Cross-referencing the documentation

When reading this manual, you will find references to other Stata manuals. For example,

[U] 26 Overview of Stata estimation commands
[R] regress
[D] reshape

The first example is a reference to chapter 26, *Overview of Stata estimation commands*, in the *User’s Guide*; the second is a reference to the `regress` entry in the *Base Reference Manual*; and the third is a reference to the `reshape` entry in the *Data Management Reference Manual*.

All the manuals in the Stata Documentation have a shorthand notation:

[GSM] Getting Started with Stata for Mac
[GSU] Getting Started with Stata for Unix
[GSW] Getting Started with Stata for Windows
[U] Stata User’s Guide
[R] Stata Base Reference Manual
[BAYES] Stata Bayesian Analysis Reference Manual
[FN] Stata Functions Reference Manual
[XT] Stata Longitudinal-Data/Panel-Data Reference Manual
[M] Stata Multiple-Imputation Reference Manual
[PSS] Stata Power and Sample-Size Reference Manual
[SVY] Stata Survey Data Reference Manual
[I] Stata Glossary and Index
Mixed-effects models are characterized as containing both fixed effects and random effects. The fixed effects are analogous to standard regression coefficients and are estimated directly. The random effects are not directly estimated (although they may be obtained postestimation) but are summarized according to their estimated variances and covariances. Random effects may take the form of either random intercepts or random coefficients, and the grouping structure of the data may consist of multiple levels of nested groups. As such, mixed-effects models are also known in the literature as multilevel models and hierarchical models. Mixed-effects commands fit mixed-effects models for a variety of distributions of the response conditional on normally distributed random effects.

**Mixed-effects linear regression**
- mixed Multilevel mixed-effects linear regression

**Mixed-effects generalized linear model**
- meglm Multilevel mixed-effects generalized linear model

**Mixed-effects binary regression**
- melogit Multilevel mixed-effects logistic regression
- meqrlogit Multilevel mixed-effects logistic regression (QR decomposition)
- meprobit Multilevel mixed-effects probit regression
- mecloglog Multilevel mixed-effects complementary log-log regression

**Mixed-effects ordinal regression**
- meologit Multilevel mixed-effects ordered logistic regression
- meoprobit Multilevel mixed-effects ordered probit regression

**Mixed-effects count-data regression**
- mepoisson Multilevel mixed-effects Poisson regression
- meqrpoisson Multilevel mixed-effects Poisson regression (QR decomposition)
- menbreg Multilevel mixed-effects negative binomial regression

**Mixed-effects multinomial regression**
Although there is no memlogit command, multilevel mixed-effects multinomial logistic models can be fit using gsem; see [SEM] example 41g.

**Mixed-effects survival model**
- mestreg Multilevel mixed-effects parametric survival models
Quick start

**Linear mixed-effects models**

Linear model of \( y \) on \( x \) with random intercepts by \( id \)

\[
mixed \ y \ x \ || \ id:
\]

Three-level linear model of \( y \) on \( x \) with random intercepts by \( doctor \) and \( patient \)

\[
mixed \ y \ x \ || \ doctor: || patient:
\]

Linear model of \( y \) on \( x \) with random intercepts and coefficients on \( x \) by \( id \)

\[
mixed \ y \ x \ || \ id: x
\]

Same model with covariance between the random slope and intercept

\[
mixed \ y \ x \ || \ id: x, \ covariance(unstructured)
\]

Linear model of \( y \) on \( x \) with crossed random effects for \( id \) and \( week \)

\[
mixed \ y \ x \ || \ _all: R.id \ || \ _all: R.week
\]

Same model specified to be more computationally efficient

\[
mixed \ y \ x \ || \ _all: R.id \ || \ week:
\]

Full factorial repeated-measures ANOVA of \( y \) on \( a \) and \( b \) with random effects by \( field \)

\[
mixed \ y \ a##b \ || \ field:
\]

**Generalized linear mixed-effects models**

Logistic model of \( y \) on \( x \) with random intercepts by \( id \), reporting odds ratios

\[
meologit \ y \ x \ || \ id: , \ or
\]

Same model specified as a GLM

\[
meglm \ y \ x \ || \ id:, \ family(bernoulli) \ link(logit)
\]

Three-level ordered probit model of \( y \) on \( x \) with random intercepts by \( doctor \) and \( patient \)

\[
meoprobit \ y \ x \ || \ doctor: || patient:
\]
Syntax

**Linear mixed-effects models**

```
mixed depvar fe_equation [ || re_equation] [ || re_equation ...] [ , options]
```

where the syntax of the fixed-effects equation, `fe_equation`, is

```
[ indepvars] [ if] [ in] [ weight] [ , fe_options]
```

and the syntax of a random-effects equation, `re_equation`, is the same as below for a generalized linear mixed-effects model.

**Generalized linear mixed-effects models**

```
me cmd depvar fe_equation [ || re_equation] [ || re_equation ...] [ , options]
```

where the syntax of the fixed-effects equation, `fe_equation`, is

```
[ indepvars] [ if] [ in] [ , fe_options]
```

and the syntax of a random-effects equation, `re_equation`, is one of the following:

- for random coefficients and intercepts
  
  `levelvar: [ varlist] [ , re_options]
  
- for random effects among the values of a factor variable
  
  `levelvar: R.varname`

`levelvar` is a variable identifying the group structure for the random effects at that level or is `_all` representing one group comprising all observations.

Remarks and examples

Remarks are presented under the following headings:

- Introduction
- Using mixed-effects commands
- Mixed-effects models
  - Linear mixed-effects models
  - Generalized linear mixed-effects models
  - Survival mixed-effects models
  - Alternative mixed-effects model specification
  - Likelihood calculation
  - Computation time and the Laplacian approximation
  - Diagnosing convergence problems
  - Distribution theory for likelihood-ratio test
- Examples
  - Two-level models
  - Covariance structures
  - Three-level models
  - Crossed-effects models
Introduction

Multilevel models have been used extensively in diverse fields, from the health and social sciences to econometrics. Mixed-effects models for binary outcomes have been used, for example, to analyze the effectiveness of toenail infection treatments (Lesaffre and Spiessens 2001) and to model union membership of young males (Vella and Verbeek 1998). Ordered outcomes have been studied by, for example, Tutz and Hennevogl (1996), who analyzed data on wine bitterness, and De Boeck and Wilson (2004), who studied verbal aggressiveness. For applications of mixed-effects models for count responses, see, for example, the study on police stops in New York City (Gelman and Hill 2007) and the analysis of the number of patents (Hall, Griliches, and Hausman 1986). Rabe-Hesketh and Skrondal (2012) provide more examples of linear and generalized linear mixed-effects models.

For a comprehensive treatment of mixed-effects models, see, for example, Searle, Casella, and McCulloch (1992); Verbeke and Molenberghs (2000); Raudenbush and Bryk (2002); Demidenko (2004); Hedeker and Gibbons (2006); McCulloch, Searle, and Neuhaus (2008); and Rabe-Hesketh and Skrondal (2012).

Using mixed-effects commands

Below we summarize general capabilities of the mixed-effects commands. We let mecmd stand for any mixed-effects command, such as mixed, melogit, or meprobit.

1. Fit a two-level random-intercept model with levelvar defining the second level:

   . mecmd depvar [indepvars] ... || levelvar: ...

2. Fit a two-level random-coefficients model containing the random-effects covariates revars at the level levelvar:

   . mecmd depvar [indepvars] ... || levelvar: revars, ...

This model assumes an independent covariance structure between the random effects; that is, all covariances are assumed to be 0. There is no statistical justification, however, for imposing any particular covariance structure between random effects at the onset of the analysis. In practice, models with an unstructured random-effects covariance matrix, which allows for distinct variances and covariances between all random-effects covariates (revars) at the same level, must be explored first; see Other covariance structures and example 3 in [ME] meprobit for details.

Stata’s commands use the default independent covariance structure for computational feasibility. Numerical methods for fitting mixed-effects models are computationally intensive—computation time increases significantly as the number of parameters increases; see Computation time and the Laplacian approximation for details. The unstructured covariance is the most general and contains many parameters, which may result in an unreasonable computation time even for relatively simple random-effects models. Whenever feasible, however, you should start your statistical analysis by fitting mixed-effects models with an unstructured covariance between random effects, as we show next.

3. Specify the unstructured covariance between the random effects in the above:

   . mecmd depvar [indepvars] ... || levelvar: revars, covariance(unstructured) ...

4. Fit a three-level nested model with levelvar1 defining the third level and levelvar2 defining the second level:

   . mecmd depvar [indepvars] ... || levelvar1: || levelvar2: ..., ...
5. Fit the above three-level nested model as a two-level model with exchangeable covariance structure at the second level (mixed, meqrlogit, and meqrpoisson only):

\[ \text{mecmd depvar} \quad \text{[indepvars]} \quad \ldots \quad || \quad \text{levelvar1: R.levelvar2, cov(exchangeable)} \quad \ldots \]

See example 11 in [ME] mixed for details about this equivalent specification. This specification may be useful for a more efficient fitting of random-effects models with a mixture of crossed and nested effects.

6. Fit higher-level nested models:

\[ \text{mecmd depvar} \quad \text{[indepvars]} \quad \ldots \quad || \quad \text{levelvar1: || levelvar2: || levelvar3: ||} \ldots \]

7. Fit a two-way crossed-effects model with the _all: notation for each random-effects equation:

\[ \text{mecmd depvar} \quad \text{[indepvars]} \quad \ldots \quad || \quad \text{_all: R.factor1 || _all: R.factor2} \ldots \]

When you use the _all: notation for each random-effects equation, the total dimension of the random-effects design equals \( r_1 + r_2 \), where \( r_1 \) and \( r_2 \) are the numbers of levels in factor1 and factor2, respectively. This specification may be infeasible for some mixed-effects models; see item 8 below for a more efficient specification of this model.

8. Fit a two-way crossed-effects model with the _all: notation for the first random-effects equation only:

\[ \text{mecmd depvar} \quad \text{[indepvars]} \quad \ldots \quad || \quad \text{_all: R.factor1 || factor2: \ldots} \]

Compared with the specification in item 7, this specification requires only \( r_1 + 1 \) parameters and is thus more efficient; see Crossed-effects models for details.

9. Fit a two-way full-factorial random-effects model:

\[ \text{mecmd depvar} \quad \text{[indepvars]} \quad \ldots \quad || \quad \text{_all: R.factor1 || factor2: || factor1: \ldots} \]

10. Fit a two-level mixed-effects model with a blocked-diagonal covariance structure between revars1 and revars2:

\[ \text{mecmd depvar} \quad \text{[indepvars]} \quad \ldots \quad || \quad \text{levelvar: revars1, noconstant} \quad \text{///} \quad \text{levelvar: revars2, noconstant} \quad \ldots \]

11. Fit a linear mixed-effects model where the correlation between the residual errors follows an autoregressive process of order 1:

\[ \text{mixed depvar} \quad \text{[indepvars]} \quad \ldots \quad || \quad \text{levelvar:, residuals(ar 1, t(time))} \quad \ldots \]

More residual error structures are available; see [ME] mixed for details.

12. Fit a two-level linear mixed-effects model accounting for sampling weights expr1 at the first (residual) level and for sampling weights expr2 at the level of levelvar:

\[ \text{mixed depvar} \quad \text{[indepvars]} \quad [\text{pweight=expr1}] \quad \ldots \quad || \quad \text{levelvar:, pweight(expr2)} \quad \ldots \]

Mixed-effects commands—with the exception of mixed, meqrlogit, and meqrpoisson—allow constraints on both fixed-effects and random-effects parameters. We provide several examples below of imposing constraints on variance components.

13. Fit a mixed-effects model with the variance of the random intercept on levelvar constrained to be 16:

\[ \text{constraint 1 \quad \_b[\text{var(_cons[levelvar]):_cons}]=16} \]

\[ \text{mecmd depvar} \quad \text{[indepvars]} \quad \ldots \quad || \quad \text{levelvar:, constraints(1)} \quad \ldots \]
14. Fit a mixed-effects model with the variance of the random intercept on `levelvar` and the variance of the random slope on `revar` to be equal:

```
. constraint 1 _b[var(revar[levelvar]):_cons] = _b[var(_cons[levelvar]):_cons]
. mecmd depvar [indepvars] ... || levelvar: revar, constraints(1) ... 
```

Note that the constraints above are equivalent to imposing an identity covariance structure for the random-effects equation:

```
. mecmd depvar [indepvars] ... || levelvar: revar, cov(identity) ... 
```

15. Assuming four random slopes `revars`, fit a mixed-effects model with the variance components at the level of `levelvar` constrained to have a banded structure:

```
. mat p = (1,..., 2,1,..., 3,2,1,., 4,3,2,1)
. mecmd depvar [indepvars] ... || levelvar: revars, noconstant ///
   covariance(pattern(p)) ... 
```

16. Assuming four random slopes `revars`, fit a mixed-effects model with the variance components at the level of `levelvar` constrained to the specified numbers, and with all the covariances constrained to be 0:

```
. mat f = diag((1,2,3,4))
. mecmd depvar [indepvars] ... || levelvar: revars, noconstant ///
   covariance(fixed(f)) ... 
```

The variance components in models in items 15 and 16 can also be constrained by using the constraints() option, but using covariance(pattern()) or covariance(fixed()) is more convenient.

**Mixed-effects models**

**Linear mixed-effects models**

Mixed-effects models for continuous responses, or linear mixed-effects (LME) models, are a generalization of linear regression allowing for the inclusion of random deviations (effects) other than those associated with the overall error term. In matrix notation,

\[
\mathbf{y} = \mathbf{X}\beta + \mathbf{Z}\mathbf{u} + \mathbf{\epsilon}
\]

(1)

where \( \mathbf{y} \) is the \( n \times 1 \) vector of responses, \( \mathbf{X} \) is an \( n \times p \) design/covariate matrix for the fixed effects \( \beta \), and \( \mathbf{Z} \) is the \( n \times q \) design/covariate matrix for the random effects \( \mathbf{u} \). The \( n \times 1 \) vector of errors \( \mathbf{\epsilon} \) is assumed to be multivariate normal with mean 0 and variance matrix \( \sigma^2_\epsilon \mathbf{R} \).

The fixed portion of (1), \( \mathbf{X}\beta \), is analogous to the linear predictor from a standard OLS regression model with \( \beta \) being the regression coefficients to be estimated. For the random portion of (1), \( \mathbf{Z}\mathbf{u} + \mathbf{\epsilon} \), we assume that \( \mathbf{u} \) has variance–covariance matrix \( \mathbf{G} \) and that \( \mathbf{u} \) is orthogonal to \( \mathbf{\epsilon} \) so that

\[
\text{Var} \begin{bmatrix} \mathbf{u} \\ \mathbf{\epsilon} \end{bmatrix} = \begin{bmatrix} \mathbf{G} & 0 \\ 0 & \sigma^2_\epsilon \mathbf{R} \end{bmatrix}
\]

The random effects \( \mathbf{u} \) are not directly estimated (although they may be predicted) but instead are characterized by the elements of \( \mathbf{G} \), known as variance components, that are estimated along with the overall residual variance \( \sigma^2_\epsilon \) and the residual-variance parameters that are contained within \( \mathbf{R} \).
The general forms of the design matrices $X$ and $Z$ allow estimation for a broad class of linear models: blocked designs, split-plot designs, growth curves, multilevel or hierarchical designs, etc. They also allow a flexible method of modeling within-cluster correlation. Subjects within the same cluster can be correlated as a result of a shared random intercept, or through a shared random slope on age (for example), or both. The general specification of $G$ also provides additional flexibility: the random intercept and random slope could themselves be modeled as independent, or correlated, or independent with equal variances, and so forth. The general structure of $R$ also allows for residual errors to be heteroskedastic and correlated and allows flexibility in exactly how these characteristics can be modeled.

In clustered-data situations, it is convenient not to consider all $n$ observations at once but instead to organize the mixed model as a series of $M$ independent groups (or clusters)

$$y_j = X_j\beta + Z_ju_j + \epsilon_j$$

for $j = 1, \ldots, M$, with cluster $j$ consisting of $n_j$ observations. The response $y_j$ comprises the rows of $y$ corresponding with the $j$th cluster, with $X_j$ and $\epsilon_j$ defined analogously. The random effects $u_j$ can now be thought of as $M$ realizations of a $q \times 1$ vector that is normally distributed with mean 0 and $q \times q$ variance matrix $\Sigma$. The matrix $Z_j$ is the $n_j \times q$ design matrix for the $j$th cluster random effects. Relating this to (1),

$$Z = \begin{bmatrix} Z_1 & 0 & \cdots & 0 \\ 0 & Z_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & Z_M \end{bmatrix}; \quad u = \begin{bmatrix} u_1 \\ \vdots \\ u_M \end{bmatrix}; \quad G = I_M \otimes \Sigma; \quad R = I_M \otimes \Lambda$$

where $\Lambda$ denotes the variance matrix of the level-1 residuals and $\otimes$ is the Kronecker product.

The mixed-model formulation (2) is from Laird and Ware (1982) and offers two key advantages. First, it makes specifications of random-effects terms easier. If the clusters are schools, you can simply specify a random effect at the school level, as opposed to thinking of what a school-level random effect would mean when all the data are considered as a whole (if it helps, think Kronecker products). Second, representing a mixed-model with (2) generalizes easily to more than one set of random effects. For example, if classes are nested within schools, then (2) can be generalized to allow random effects at both the school and the class-within-school levels.

In Stata, you can use `mixed` to fit linear mixed-effects models; see [ME] `mixed` for a detailed discussion and examples. Various predictions, statistics, and diagnostic measures are available after fitting an LME model with `mixed`. For the most part, calculation centers around obtaining estimates of random effects; see [ME] `mixed postestimation` for a detailed discussion and examples.

**Generalized linear mixed-effects models**

Generalized linear mixed-effects (GLME) models, also known as generalized linear mixed models (GLMMs), are extensions of generalized linear models allowing for the inclusion of random deviations (effects). In matrix notation,

$$g\{E(y|X,u)\} = X\beta + Zu, \quad y \sim F$$

where $y$ is the $n \times 1$ vector of responses from the distributional family $F$, $X$ is an $n \times p$ design/covariate matrix for the fixed effects $\beta$, and $Z$ is an $n \times q$ design/covariate matrix for the random effects $u$. The $X\beta + Zu$ part is called the linear predictor and is often denoted as $\eta$. $g(\cdot)$ is called the link function and is assumed to be invertible such that

$$E(y|u) = g^{-1}(X\beta + Zu) = H(\eta) = \mu$$
For notational convenience here and throughout this manual entry, we suppress the dependence of $y$ on $X$. Substituting various definitions for $g(\cdot)$ and $F$ results in a wide array of models. For instance, if $g(\cdot)$ is the logit function and $y$ is distributed as Bernoulli, we have

$$\text{logit}\{E(y)\} = X\beta + Zu, \quad y \sim \text{Bernoulli}$$

or mixed-effects logistic regression. If $g(\cdot)$ is the natural log function and $y$ is distributed as Poisson, we have

$$\ln\{E(y)\} = X\beta + Zu, \quad y \sim \text{Poisson}$$

or mixed-effects Poisson regression.

In Stata, you can use `meglm` to fit mixed-effects models for nonlinear responses. Some combinations of families and links are so common that we implemented them as separate commands in terms of `meglm`.

<table>
<thead>
<tr>
<th>Command</th>
<th><code>meglm</code> equivalent</th>
</tr>
</thead>
<tbody>
<tr>
<td>melogit</td>
<td><code>family(bernoulli) link(logit)</code></td>
</tr>
<tr>
<td>meprobit</td>
<td><code>family(bernoulli) link(probit)</code></td>
</tr>
<tr>
<td>mecloglog</td>
<td><code>family(bernoulli) link(cloglog)</code></td>
</tr>
<tr>
<td>meologit</td>
<td><code>family(ordinal) link(logit)</code></td>
</tr>
<tr>
<td>meoprobit</td>
<td><code>family(ordinal) link(probit)</code></td>
</tr>
<tr>
<td>mepoisson</td>
<td><code>family(poisson) link(log)</code></td>
</tr>
<tr>
<td>menbreg</td>
<td><code>family(nbinomial) link(log)</code></td>
</tr>
</tbody>
</table>

When no family–link combination is specified, `meglm` defaults to a Gaussian family with an identity link. Thus `meglm` can be used to fit linear mixed-effects models; however, for those models we recommend using the more specialized `mixed`, which, in addition to `meglm` capabilities, accepts frequency and sampling weights and allows for modeling of the structure of the residual errors; see [ME] `mixed` for details.

Various predictions, statistics, and diagnostic measures are available after fitting a GLME model with `meglm` and other `me` commands. For the most part, calculation centers around obtaining estimates of random effects; see [ME] `meglm postestimation` for a detailed discussion and examples.

For the random portion of (4), $Zu$, we assume that $u$ has variance–covariance matrix $G$ such that

$$\text{Var}(u) = G$$

The random effects $u$ are not directly estimated (although they may be predicted) but instead are characterized by the elements of $G$, known as variance components.

Analogously to (2), in clustered-data situations, we can write

$$E(y_j|u_j) = g^{-1}(X_j\beta + Z_ju_j), \quad y_j \sim F$$

with all the elements defined as before. In terms of the whole dataset, we now have

$$Z = \begin{bmatrix} Z_1 & 0 & \cdots & 0 \\ 0 & Z_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & Z_M \end{bmatrix}; \quad u = \begin{bmatrix} u_1 \\ \vdots \\ u_M \end{bmatrix}; \quad G = I_M \otimes \Sigma$$
By our convention on counting and ordering model levels, models (2) and (5) are two-level models, with extensions to three, four, or any number of levels. The observation $y_{ij}$ is for individual $i$ within cluster $j$, and the individuals comprise the first level while the clusters comprise the second level of the model. In our hypothetical three-level model with classes nested within schools, the observations within classes (the students, presumably) would constitute the first level, the classes would constitute the second level, and the schools would constitute the third level. This differs from certain citations in the classical ANOVA literature and texts such as Pinheiro and Bates (2000) but is the standard in the vast literature on hierarchical models, for example, Skrondal and Rabe-Hesketh (2004).

Survival mixed-effects models

Parametric survival mixed-effects models use a trivariate response variable $(t_0, t, d)$, where each response corresponds to a period under observation $(t_0, t]$ and results in either failure $(d = 1)$ or right-censoring $(d = 0)$ at time $t$. See [ST] streg for background information on parametric survival models. Two often-used models for adjusting survivor functions for the effects of covariates are the accelerated failure-time (AFT) model and the multiplicative or proportional hazards (PH) model.

In the AFT parameterization, the natural logarithm of the survival time, $log(t)$, is expressed as a linear function of the covariates. When we incorporate random effects, this yields the model

$$log(t_j) = X_j\beta + Z_j u_j + v_j$$

where $log(\cdot)$ is an elementwise function, and $v_j$ is a vector of observation-level errors. The distributional form of the error term determines the regression model.

In the PH model, the covariates have a multiplicative effect on the hazard function

$$h(t_j) = h_0(t_j) \exp(X_j\beta + Z_j u_j)$$

where all the functions are elementwise, and $h_0(\cdot)$ is a baseline hazard function. The functional form of $h_0(\cdot)$ determines the regression model.

In Stata, you can use mestreg to fit multilevel mixed-effects parametric survival models for the following distributions and parameterizations.

<table>
<thead>
<tr>
<th>Distribution</th>
<th>Parameterization</th>
</tr>
</thead>
<tbody>
<tr>
<td>exponential</td>
<td>PH, AFT</td>
</tr>
<tr>
<td>loglogistic</td>
<td>AFT</td>
</tr>
<tr>
<td>weibull</td>
<td>PH, AFT</td>
</tr>
<tr>
<td>lognormal</td>
<td>AFT</td>
</tr>
<tr>
<td>gamma</td>
<td>AFT</td>
</tr>
</tbody>
</table>

mestreg is suitable only for data that have been set using the stset command. By using stset on your data, you define the variables _t0, _t, and _d, which serve as the trivariate response. See [ME] mestreg for more details about the command. Various predictions, statistics, and diagnostic measures are available after fitting a mixed-effects survival model with mestreg; see [ME] mestreg postestimation for a detailed discussion and examples.
Alternative mixed-effects model specification

In this section, we present a hierarchical or multistage formulation of mixed-effects models where each level is described by its own set of equations.

Consider a random-intercept model that we write here in general terms:

\[ y_{ij} = \beta_0 + \beta_1 x_{ij} + u_j + \epsilon_{ij} \]  

(7)

This single-equation specification contains both level-1 and level-2 effects. In the hierarchical form, we specify a separate equation for each level.

\[ y_{ij} = \gamma_{0j} + \beta_1 x_{ij} + \epsilon_{ij} \]
\[ \gamma_{0j} = \beta_{00} + u_{0j} \]  

(8)

The equation for the intercept \( \gamma_{0j} \) consists of the overall mean intercept \( \beta_{00} \) and a cluster-specific random intercept \( u_{0j} \). To fit this model in Stata, we must translate the multiple-equation notation into a single-equation form. We substitute the second equation into the first one and rearrange terms.

\[ y_{ij} = \beta_{00} + u_{0j} + \beta_1 x_{ij} + \epsilon_{ij} = \beta_{00} + \beta_1 x_{ij} + u_{0j} + \epsilon_{ij} \]  

(9)

Note that model (9) is the same as model (7) with \( \beta_{00} \equiv \beta_0 \) and \( u_{0j} \equiv u_j \). Thus the Stata syntax for our generic random-intercept model is

```
.mixed y x || id:
```

where \( \text{id} \) is the variable designating the clusters.

We can extend model (8) to include a random slope. We do so by specifying an additional equation for the slope on \( x_{ij} \).

\[ y_{ij} = \gamma_{0j} + \gamma_{1j} x_{ij} + \epsilon_{ij} \]
\[ \gamma_{0j} = \beta_{00} + u_{0j} \]
\[ \gamma_{1j} = \beta_{10} + u_{1j} \]  

(10)

The additional equation for the slope \( \gamma_{1j} \) consists of the overall mean slope \( \beta_{10} \) and a cluster-specific random slope \( u_{1j} \). We substitute the last two equations into the first one to obtain a reduced-form model.

\[ y_{ij} = (\beta_{00} + u_{0j}) + (\beta_{10} + u_{1j}) x_{ij} + \epsilon_{ij} = \beta_{00} + \beta_{10} x_{ij} + u_{0j} + u_{1j} x_{ij} + \epsilon_{ij} \]

The Stata syntax for this model becomes

```
.mixed y x || id: x, covariance(unstructured)
```

where we specified an unstructured covariance structure for the level-2 \( u \) terms.
Here we further extend the random-slope random-intercept model (10) by adding a level-2 covariate $z_j$ into the level-2 equations.

\[
\begin{align*}
y_{ij} &= \gamma_{0j} + \gamma_{1j}x_{ij} + \epsilon_{ij} \\
\gamma_{0j} &= \beta_{00} + \beta_{01}z_j + u_{0j} \\
\gamma_{1j} &= \beta_{10} + \beta_{11}z_j + u_{1j}
\end{align*}
\]

We substitute as before to obtain a single-equation form:

\[
y_{ij} = (\beta_{00} + \beta_{01}z_j + u_{0j}) + (\beta_{10} + \beta_{11}z_j + u_{1j})x_{ij} + \epsilon_{ij}
\]

Now the fixed-effects portion of the equation contains a constant and variables $x$, $z$, and their interaction. Assuming both $x$ and $z$ are continuous variables, we can use the following Stata syntax to fit this model:

```
. mixed y x z c.x#c.z || id: x, covariance(unstructured)
```

We refer you to Raudenbush and Bryk (2002) and Rabe-Hesketh and Skrondal (2012) for a more thorough discussion and further examples of multistage mixed-model formulations, including three-level models.

**Likelihood calculation**

The key to fitting mixed models lies in estimating the variance components, and for that there exist many methods. Most of the early literature in LME models dealt with estimating variance components in ANOVA models. For simple models with balanced data, estimating variance components amounts to solving a system of equations obtained by setting expected mean-squares expressions equal to their observed counterparts. Much of the work in extending the ANOVA method to unbalanced data for general ANOVA designs is attributed to Henderson (1953).

The ANOVA method, however, has its shortcomings. Among these is a lack of uniqueness in that alternative, unbiased estimates of variance components could be derived using other quadratic forms of the data in place of observed mean squares (Searle, Casella, and McCulloch 1992, 38–39). As a result, ANOVA methods gave way to more modern methods, such as minimum norm quadratic unbiased estimation (MINQUE) and minimum variance quadratic unbiased estimation (MIVQUE); see Rao (1973) for MINQUE and LaMotte (1973) for MIVQUE. Both methods involve finding optimal quadratic forms of the data that are unbiased for the variance components.

Stata uses maximum likelihood (ML) to fit LME and GLME models. The ML estimates are based on the usual application of likelihood theory, given the distributional assumptions of the model. In addition, for linear mixed-effects models, mixed offers the method of restricted maximum likelihood (REML). The basic idea behind REML (Thompson 1962) is that you can form a set of linear contrasts of the response that do not depend on the fixed effects $\beta$ but instead depend only on the variance components to be estimated. You then apply ML methods by using the distribution of the linear contrasts to form the likelihood; see the Methods and formulas section of [ME] mixed for a detailed discussion of ML and REML methods in the context of linear mixed-effects models.

Log-likelihood calculations for fitting any LME or GLME model require integrating out the random effects. For LME models, this integral has a closed-form solution; for GLME models, it does not. In dealing with this difficulty, early estimation methods avoided the integration altogether. Two such popular methods are the closely related penalized quasi-likelihood (PQL) and marginal quasi-likelihood (MQL) (Breslow and Clayton 1993). Both PQL and MQL use a combination of iterative reweighted
least squares (see [R] \texttt{glm}) and standard estimation techniques for fitting LME models. Efficient computational methods for fitting LME models have existed for some time (Bates and Pinheiro 1998; Littell et al. 2006), and PQL and MQL inherit this computational efficiency. However, both of these methods suffer from two key disadvantages. First, they have been shown to be biased, and this bias can be severe when clusters are small or intracluster correlation is high (Rodríguez and Goldman 1995; Lin and Breslow 1996). Second, because they are “quasi-likelihood” methods and not true likelihood methods, their use prohibits comparing nested models via likelihood-ratio (LR) tests, blocking the main avenue of inference involving variance components.

The advent of modern computers has brought with it the development of more computationally intensive methods, such as bias-corrected PQL (Lin and Breslow 1996), Bayesian Markov-Chain Monte Carlo, and simulated maximum likelihood, just to name a few; see Ng et al. (2006) for a discussion of these alternate strategies (and more) for mixed-effects models for binary outcomes.

One widely used modern method is to directly estimate the integral required to calculate the log likelihood by Gauss–Hermite quadrature or some variation thereof. Because the log likelihood itself is estimated, this method has the advantage of permitting LR tests for comparing nested models. Also, if done correctly, quadrature approximations can be quite accurate, thus minimizing bias. \texttt{meglm} and the other \texttt{me} commands support three types of Gauss–Hermite quadratures: mean–variance adaptive Gauss–Hermite quadrature (MV AGH), mode-curvature adaptive Gauss–Hermite quadrature (MCAGH), and nonadaptive Gauss–Hermite quadrature (GHQ); see \textit{Methods and formulas} of [ME] \texttt{meglm} for a detailed discussion of these quadrature methods. A fourth method, the Laplacian approximation, that does not involve numerical integration is also offered; see \textit{Computation time and the Laplacian approximation} below and \textit{Methods and formulas} of [ME] \texttt{meglm} for a detailed discussion of the Laplacian approximation method.

**Computation time and the Laplacian approximation**

Like many programs that fit generalized linear mixed models, \texttt{me} commands can be computationally intensive. This is particularly true for large datasets with many lowest-level clusters, models with many random coefficients, models with many estimable parameters (both fixed effects and variance components), or any combination thereof.

Computation time will also depend on hardware and other external factors but in general is (roughly) a function of $p^2\{M + M(N_Q)^{q_t}\}$, where $p$ is the number of estimable parameters, $M$ is the number of lowest-level (smallest) clusters, $N_Q$ is the number of quadrature points, and $q_t$ is the total dimension of the random effects, that is, the total number of random intercepts and coefficients at all levels.

For a given model and a given dataset, the only prevailing factor influencing computation time is $(N_Q)^{q_t}$. However, because this is a power function, this factor can get prohibitively large. For example, using five quadrature points for a model with one random intercept and three random coefficients, we get $(N_Q)^{q_t} = 5^4 = 625$. Even a modest increase to seven quadrature points would increase this factor by almost fourfold $(7^4 = 2,401)$, which, depending on $M$ and $p$, could drastically slow down estimation. When fitting mixed-effects models, you should always assess whether the approximation is adequate by refitting the model with a larger number of quadrature points. If the results are essentially the same, the lower number of quadrature points can be used.

However, we do not deny a tradeoff between speed and accuracy, and in that spirit we give you the option to choose a (possibly) less accurate solution in the interest of getting quicker results. Toward this end is the limiting case of $N_Q = 1$, otherwise known as the Laplacian approximation; see \textit{Methods and formulas} of [ME] \texttt{meglm}. The computational benefit is evident—1 raised to any power equals 1—and the Laplacian approximation has been shown to perform well in certain situations (Liu and Pierce 1994; Tierney and Kadane 1986). When using Laplacian approximation, keep the following in mind:
1. Fixed-effects parameters and their standard errors are well approximated by the Laplacian method. Therefore, if your interest lies primarily here, then the Laplacian approximation may be a viable alternative.

2. Estimates of variance components exhibit bias, particularly the variances.

3. The model log likelihood and comparison LR test are in fair agreement with statistics obtained via quadrature methods.

Although this is by no means the rule, we find the above observations to be fairly typical based on our own experience. Pinheiro and Chao (2006) also make observations similar to points 1 and 2 on the basis of their simulation studies: bias due to Laplace (when present) tends to exhibit itself more in the estimated variance components than in the estimates of the fixed effects as well as at the lower levels in higher-level models.

Item 3 is of particular interest, because it demonstrates that the Laplacian approximation can produce a decent estimate of the model log likelihood. Consequently, you can use the Laplacian approximation during the model building phase of your analysis, during which you are comparing competing models by using LR tests. Once you settle on a parsimonious model that fits well, you can then increase the number of quadrature points and obtain more accurate parameter estimates for further study.

Of course, sometimes the Laplacian approximation will perform either better or worse than observed here. This behavior depends primarily on cluster size and intracluster correlation, but the relative influence of these factors is unclear. The idea behind the Laplacian approximation is to approximate the posterior density of the random effects given the response with a normal distribution; see Methods and formulas of [ME] meglm. Asymptotic theory dictates that this approximation improves with larger clusters. Of course, the key question, as always, is “How large is large enough?” Also, there are data situations where the Laplacian approximation performs well even with small clusters. Therefore, it is difficult to make a definitive call as to when you can expect the Laplacian approximation to yield accurate results across all aspects of the model.

In conclusion, consider our above advice as a rule of thumb based on empirical evidence.

Diagnosing convergence problems

Given the flexibility of mixed-effects models, you will find that some models fail to converge when used with your data. The default gradient-based method used by mixed-effects commands is the Newton–Raphson algorithm, requiring the calculation of a gradient vector and Hessian (second-derivative) matrix; see [R] ml.

A failure to converge can take any one of three forms:
1. repeated nonconcave or backed-up iterations without convergence;
2. a Hessian (second-derivative) calculation that has become asymmetric, unstable, or has missing values; or
3. the message “standard-error calculation has failed” when computing standard errors.

All three situations essentially amount to the same thing: the Hessian calculation has become unstable, most likely because of a ridge in the likelihood function, a subsurface of the likelihood in which all points give the same value of the likelihood and for which there is no unique solution.

Such behavior is usually the result of one of the following two situations:
A. A model that is not identified given the data, for example, fitting the three-level nested random intercept model

\[ y_{jk} = x_{jk} \beta + u_{k}^{(3)} + u_{jk}^{(2)} + \epsilon_{jk} \]
without any replicated measurements at the \((j,k)\) level, that is, with only one \(i\) per \((j,k)\) combination. This model is unidentified for such data because the random intercepts \(u_{jk}^{(2)}\) are confounded with the overall errors \(\epsilon_{jk}\).

B. A model that contains a variance component whose estimate is really close to 0. When this occurs, a ridge is formed by an interval of values near 0, which produce the same likelihood and look equally good to the optimizer.

For LME models, one useful way to diagnose problems of nonconvergence is to rely on the expectation-maximization (EM) algorithm (Dempster, Laird, and Rubin 1977), normally used by `mixed` only as a means of refining starting values; see Diagnosing convergence problems of `[ME] mixed` for details.

If your data and model are nearly unidentified, as opposed to fully unidentified, you may be able to obtain convergence with standard errors by changing some of the settings of the gradient-based optimization. Adding the `difficult` option can be particularly helpful if you are seeing many “nonconcave” messages; you may also consider changing the `technique()` or using the `nonrtolerance` option; see `[R] maximize`.

Regardless of how the convergence problem revealed itself, you may try to obtain better starting values; see Obtaining better starting values in `[ME] meglm` for details.

### Distribution theory for likelihood-ratio test

When determining the asymptotic distribution of an LR test comparing two nested mixed-effects models, issues concerning boundary problems imposed by estimating strictly positive quantities (that is, variances) can complicate the situation. For example, when performing LR tests involving linear mixed-effects models (whether comparing with linear regression within `mixed` or comparing two separate linear mixed-effects models with `lrtest`), you may thus sometimes see a test labeled as `chibar` rather than the usual `chi2`, or you may see a `chi2` test with a note attached stating that the test is conservative or possibly conservative depending on the hypothesis being tested.

At the heart of the issue is the number of variances being restricted to 0 in the reduced model. If there are none, the usual asymptotic theory holds, and the distribution of the test statistic is \(\chi^2\) with degrees of freedom equal to the difference in the number of estimated parameters between both models.

When there is only one variance being set to 0 in the reduced model, the asymptotic distribution of the LR test statistic is a 50:50 mixture of a \(\chi^2_p\) and a \(\chi^2_{p+1}\) distribution, where \(p\) is the number of other restricted parameters in the reduced model that are unaffected by boundary conditions. Stata labels such test statistics as `chibar` and adjusts the significance levels accordingly. See Self and Liang (1987) for the appropriate theory or Gutierrez, Carter, and Drukker (2001) for a Stata-specific discussion.

When more than one variance parameter is being set to 0 in the reduced model, however, the situation becomes more complicated. For example, consider a comparison test versus linear regression for a mixed model with two random coefficients and unstructured covariance matrix

\[
\Sigma = \begin{bmatrix}
\sigma^2_0 & \sigma_{01} \\
\sigma_{01} & \sigma^2_1
\end{bmatrix}
\]

Because the random component of the mixed model comprises three parameters \((\sigma^2_0, \sigma_{01}, \sigma^2_1)\), on the surface it would seem that the LR comparison test would be distributed as \(\chi^2_3\). However, two complications need to be considered. First, the variances \(\sigma^2_0\) and \(\sigma^2_1\) are restricted to be positive, and
second, constraints such as \( \sigma_1^2 = 0 \) implicitly restrict the covariance \( \sigma_{01} \) to be 0 as well. From a technical standpoint, it is unclear how many parameters must be restricted to reduce the model to linear regression.

Because of these complications, appropriate and sufficiently general distribution theory for the more-than-one-variance case has yet to be developed. Theory (for example, Stram and Lee [1994]) and empirical studies (for example, McLachlan and Basford [1988]) have demonstrated that, whatever the distribution of the LR test statistic, its tail probabilities are bounded above by those of the \( \chi^2 \) distribution with degrees of freedom equal to the full number of restricted parameters (three in the above example).

The `mixed` and `me` commands use this reference distribution, the \( \chi^2 \) with full degrees of freedom, to produce a conservative test and place a note in the output labeling the test as such. Because the displayed significance level is an upper bound, rejection of the null hypothesis based on the reported level would imply rejection on the basis of the actual level.

**Examples**

**Two-level models**

> **Example 1: Growth-curve model**

Consider a longitudinal dataset, used by both Ruppert, Wand, and Carroll (2003) and Diggle et al. (2002), consisting of weight measurements of 48 pigs on 9 successive weeks. Pigs are identified by the variable `id`. Each pig experiences a linear trend in growth, but overall weight measurements vary from pig to pig. Because we are not really interested in these particular 48 pigs per se, we instead treat them as a random sample from a larger population and model the between-pig variability as a random effect, or in the terminology of (2), as a random-intercept term at the pig level. We thus wish to fit the model

\[
weight_{ij} = \beta_0 + \beta_1 \text{week}_{ij} + u_j + \epsilon_{ij}
\]

for \( i = 1, \ldots, 9 \) weeks and \( j = 1, \ldots, 48 \) pigs. The fixed portion of the model, \( \beta_0 + \beta_1 \text{week}_{ij} \), simply states that we want one overall regression line representing the population average. The random effect \( u_j \) serves to shift this regression line up or down according to each pig. Because the random effects occur at the pig level (`id`), we fit the model by typing
16 me — Introduction to multilevel mixed-effects models

. use http://www.stata-press.com/data/r14/pig
(Longitudinal analysis of pig weights)
. mixed weight week || id:
Performing EM optimization:
Performing gradient-based optimization:
Iteration 0:  log likelihood = -1014.9268
Iteration 1:  log likelihood = -1014.9268
Computing standard errors:
Mixed-effects ML regression Number of obs = 432
Group variable: id Number of groups = 48
Obs per group:
  min = 9
  avg = 9.0
  max = 9
Wald chi2(1)       = 25337.49
Log likelihood = -1014.9268  Prob > chi2 = 0.0000

                   | Coef.     Std. Err.      z    P>|z|     [95% Conf. Interval]
-------------|-------------------------------|-------------------------------|-------------------------------|-------------------------------|-------------------------------|
weight       | 6.209896          .0390124  159.18  0.000     6.133433   6.286359
_cons        | 19.35561          .5974059   32.40  0.000     18.18472   20.52651

Random-effects Parameters   | Estimate  Std. Err. [95% Conf. Interval]
-------------|-------------------------------|-------------------------------|-------------------------------|-------------------------------|
id: Identity  | var(_cons)        14.81751    3.124226       9.801716    22.40002       |
var(Residual) | 4.383264          .3163348   3.805112    5.04926       |
LR test vs. linear model:  chibar2(01) = 472.65 Prob >= chibar2 = 0.0000

We explain the output in detail in example 1 of [ME] mixed. Here we only highlight the most important points.

1. The first estimation table reports the fixed effects. We estimate $\beta_0 = 19.36$ and $\beta_1 = 6.21$.
2. The second estimation table shows the estimated variance components. The first section of the table is labeled id: Identity, meaning that these are random effects at the id (pig) level and that their variance–covariance matrix is a multiple of the identity matrix; that is, $\Sigma = \sigma^2_u I$. The estimate of $\hat{\sigma}^2_u$ is 14.82 with standard error 3.12.
3. The row labeled var(Residual) displays the estimated standard deviation of the overall error term; that is, $\hat{\sigma}^2_\epsilon = 4.38$. This is the variance of the level-one errors, that is, the residuals.
4. An LR test comparing the model with one-level ordinary linear regression is provided and is highly significant for these data.
We can predict the random intercept \( u_j \) and list the predicted random intercept for the first 10 pigs by typing

```
. predict r_int, reffects
. egen byte tag = tag(id)
. list id r_int if id<=10 & tag
```

<table>
<thead>
<tr>
<th>id</th>
<th>r_int</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>-1.683105</td>
</tr>
<tr>
<td>10</td>
<td>0.8987018</td>
</tr>
<tr>
<td>19</td>
<td>-1.952043</td>
</tr>
<tr>
<td>28</td>
<td>-1.79068</td>
</tr>
<tr>
<td>37</td>
<td>-3.189159</td>
</tr>
<tr>
<td>46</td>
<td>-3.780823</td>
</tr>
<tr>
<td>55</td>
<td>-2.382344</td>
</tr>
<tr>
<td>64</td>
<td>-1.952043</td>
</tr>
<tr>
<td>73</td>
<td>-6.739143</td>
</tr>
<tr>
<td>82</td>
<td>1.16764</td>
</tr>
</tbody>
</table>

In example 3 of [ME] mixed, we show how to fit a random-slope model for these data, and in example 1 of [ME] mixed postestimation, we show how to plot the estimated regression lines for each of the pigs.

> Example 2: Split-plot design

Here we replicate the example of a split-plot design from Kuehl (2000, 477). The researchers investigate the effects of nitrogen in four different chemical forms and the effects of thatch accumulation on the quality of golf turf. The experimental plots were arranged in a randomized complete block design with two replications. After two years of nitrogen treatment, the second treatment factor, years of thatch accumulation, was added to the experiment. Each of the eight experimental plots was split into three subplots. Within each plot, the subplots were randomly assigned to accumulate thatch for a period of 2, 5, and 8 years.
Nitrogen treatment is stored in the variable `nitrogen`, and the chemicals used are urea, ammonium sulphate, isobutylidene diurea (IBDU), and sulphur-coated urea (urea SC). The length of thatch accumulation is stored in the variable `thatch`. The response is the chlorophyll content of grass clippings, recorded in mg/g and stored in the variable `chlorophyll`. The `block` variable identifies the replication group.

There are two sources of variation in this example corresponding to the whole-plot errors and the subplot errors. The subplot errors are the residual errors. The whole-plot errors represents variation in the chlorophyll content across nitrogen treatments and replications. We create the variable `wpunit` to represent the whole-plot units that correspond to the levels of the nitrogen treatment and block interaction.

```
. egen wpunit = group(nitrogen block)
. mixed chlorophyll ibn.nitrogen##ibn.thatch ibn.block, noomitted noconstant || wpunit:, reml
```

Performing EM optimization:
Performing gradient-based optimization:

Iteration 0:  log restricted-likelihood = -13.212401
Iteration 1:  log restricted-likelihood = -13.203149
Iteration 2:  log restricted-likelihood = -13.203125
Iteration 3:  log restricted-likelihood = -13.203125
Computing standard errors:
Mixed-effects REML regression
Group variable: wpunit
Obs per group:
  min = 3
  avg = 3.0
  max = 3
Wald chi2(13) = 2438.36
Log restricted-likelihood = -13.203125
Prob > chi2 = 0.0000

| chlorophyll | Coef. | Std. Err. | z    | P>|z| | [95% Conf. Interval] |
|-------------|-------|-----------|------|------|----------------------|
| nitrogen    |       |           |      |      |                      |
| urea        | 5.245 | .3986     | 13.16| 0.000| 4.464589             | 6.027078 |
| ammonium s..| 5.946 | .3986     | 14.92| 0.000| 5.164589             | 6.727078 |
| IBDU        | 7.946 | .3986     | 19.93| 0.000| 7.164589             | 8.727078 |
| Urea (SC)   | 8.596 | .3986     | 21.56| 0.000| 7.814589             | 9.377078 |
| thatch      |       |           |      |      |                      |
| 2           | -1.1  | .4632     | -2.37| 0.018| -2.007917            | -1.920828|
| 5           | .150  | .4632     | 0.32 | 0.746| -.7579163            | 1.057917 |
| nitrogen#   |       |           |      |      |                      |
| thatch      |       |           |      |      |                      |
| urea#2      | -.150 | .6551     | -0.23| 0.819| -1.433989            | 1.133988 |
| urea#5      | .0999 | .6551     | 0.15 | 0.879| -1.183989            | 1.383988 |
| ammonium s..|       |           |      |      |                      |
| 2           | .8999 | .6551     | 1.37 | 0.169| -.383987            | 2.183988 |
| ammonium s..|       |           |      |      |                      |
| 5           | -.100 | .6551     | -0.15| 0.879| -1.383989            | 1.183988 |
| IBDU#2      | -.200 | .6551     | -0.31| 0.760| -1.483989            | 1.083988 |
| IBDU#5      | -.195 | .6551     | -2.98| 0.003| -3.233989            | -.6660124|
| block       |       |           |      |      |                      |
| 1           | -.291 | .2643     | -1.10| 0.270| -.8097955            | .2264622 |

Random-effects Parameters

<table>
<thead>
<tr>
<th>wpunit: Identity</th>
<th>Estimate</th>
<th>Std. Err.</th>
<th>[95% Conf. Interval]</th>
</tr>
</thead>
<tbody>
<tr>
<td>var(_cons)</td>
<td>.0682407</td>
<td>.1195933</td>
<td>.0021994 2.117344</td>
</tr>
<tr>
<td>var(Residual)</td>
<td>.2145833</td>
<td>.1072917</td>
<td>.080537  .5717376</td>
</tr>
</tbody>
</table>

LR test vs. linear model: chibar2(01) = 0.53
Prob >= chibar2 = 0.2324
We can calculate the cell means for source of nitrogen and years of thatch accumulation by using `margins`.

```
.margins thatch#nitrogen
Predictive margins Number of obs = 24
Expression : Linear prediction, fixed portion, predict()
```

| thatch#nitrogen | Margin | Std. Err. | z     | P>|z| | [95% Conf. Interval] |
|-----------------|--------|-----------|-------|------|----------------------|
| 2#urea          | 3.85   | .3760479  | 10.24 | 0.000| 3.11296 4.58704     |
| ammonium s.     | 5.6    | .3760479  | 14.89 | 0.000| 4.86296 6.33704     |
| 2#IBDU          | 6.5    | .3760479  | 17.29 | 0.000| 5.76296 7.23704     |
| 2#Urea (SC)     | 7.35   | .3760479  | 19.55 | 0.000| 6.61296 8.087041    |
| 5#urea          | 5.35   | .3760479  | 14.23 | 0.000| 4.61296 6.087041    |
| ammonium s.     | 5.85   | .3760479  | 15.56 | 0.000| 5.11296 6.58704     |
| 5#IBDU          | 6      | .3760479  | 15.96 | 0.000| 5.26296 6.73704     |
| 5#Urea (SC)     | 8.6    | .3760479  | 22.87 | 0.000| 7.86296 9.337041    |
| 8#urea          | 5.1    | .3760479  | 13.56 | 0.000| 4.36296 5.837041    |
| ammonium s.     | 5.8    | .3760479  | 15.42 | 0.000| 5.06296 6.53704     |
| 8#IBDU          | 7.8    | .3760479  | 20.74 | 0.000| 7.06296 8.537041    |
| 8#Urea (SC)     | 8.45   | .3760479  | 22.47 | 0.000| 7.712959 9.18704    |

It is easier to see the effect of the treatments if we plot the impact of the four nitrogen and the three thatch treatments. We can use `marginsplot` to plot the means of chlorophyll content versus years of thatch accumulation by nitrogen source.
We can see an increase in the mean chlorophyll content over the years of thatch accumulation for all but one nitrogen source.

The marginal means can be obtained by using `margins` on one variable at a time.

```
. margins thatch
```

| thatch | Margin | Std. Err. | z    | P>|z|    | [95% Conf. Interval]   |
|--------|--------|-----------|------|--------|------------------------|
| 2      | 5.825  | .188024   | 30.98| 0.000  | 5.45648 - 6.19352      |
| 5      | 6.45   | .188024   | 34.30| 0.000  | 6.08148 - 6.81852      |
| 8      | 6.7875 | .188024   | 36.10| 0.000  | 6.41898 - 7.15602      |
. margins nitrogen
Predictive margins Number of obs = 24
Expression : Linear prediction, fixed portion, predict()

|          | Delta-method | Margin       | Std. Err. | z    | P>|z| | [95% Conf. Interval] |
|----------|--------------|--------------|-----------|------|------|---------------------|
| nitrogen |              | 4.766667     | .2643563  | 18.03| 0.000| 4.248538 5.284796   |
| urea     |              | 5.75         | .2643563  | 21.75| 0.000| 5.231871 6.268129   |
| ammonium |              | 6.766667     | .2643563  | 25.60| 0.000| 6.248538 7.284796   |
| IBDU     |              | 8.133333     | .2643563  | 30.77| 0.000| 7.615205 8.651462   |

Marchenko (2006) shows more examples of fitting other experimental designs using linear mixed-effects models.

Example 3: Binomial counts

We use the data taken from Agresti (2013, 219) on graduate school applications to the 23 departments within the College of Liberal Arts and Sciences at the University of Florida during the 1997–1998 academic year. The dataset contains the department ID (department), the number of applications (napplied), and the number of students admitted (nadmitted) cross-classified by gender (female).

. use http://www.stata-press.com/data/r14/admissions, clear
(Graduate school admissions data)
. describe
Contains data from http://www.stata-press.com/data/r14/admissions.dta
obs: 46 Graduate school admissions data
total size: 460 (_dta has notes)
storage display value
variable name type format label variable label
department long %8.0g dept department id
nadmitted byte %8.0g number of admissions
napplied float %9.0g number of applications
female byte %8.0g =1 if female, =0 if male

We wish to investigate whether admission decisions are independent of gender. Given department and gender, the probability of admission follows a binomial model, that is, \( \Pr(Y_{ij} = y_{ij}) = \text{Binomial}(n_{ij}, \pi_{ij}) \), where \( i = \{0, 1\} \) and \( j = 1, \ldots, 23 \). We fit a mixed-effects binomial logistic model with a random intercept at the department level.
. melogit nadmitted female || department:, binomial(napplied) or

Fitting fixed-effects model:
Iteration 0:  log likelihood = -302.47786
Iteration 1:  log likelihood = -300.00004
Iteration 2:  log likelihood = -299.99934
Iteration 3:  log likelihood = -299.99934
Refining starting values:
Grid node 0:  log likelihood = -145.08843

Fitting full model:
Iteration 0:  log likelihood = -145.08843
Iteration 1:  log likelihood = -140.8514
Iteration 2:  log likelihood = -140.61709
Iteration 3:  log likelihood = -140.61628
Iteration 4:  log likelihood = -140.61628

Mixed-effects logistic regression
Binomial variable:  napplied
Group variable:  department

Obs per group:
  min =  2
  avg =  2.0
  max =  2

Integration method: mvaghermite
Integration pts. =  7

Wald chi2(1) =  2.14  Prob > chi2 =  0.1435
Log likelihood = -140.61628

|           | Odds Ratio | Std. Err. | z     | P>|z|  | [95% Conf. Interval] |
|-----------|------------|-----------|-------|------|----------------------|
| female    | 1.176898   | .1310535  | 1.46  | 0.144| 0.9461357  1.463944 |
| _cons     | .7907009   | .2057191  | -0.90 | 0.367| 0.4748457  1.316655 |
| department|            |           |       |      |                      |
| var(_cons)| 1.345383   | .460702   |       |      | .6876497  2.632234  |

LR test vs. logistic model: chi2bar(01) = 318.77  Prob >= chi2bar2 = 0.0000

The odds of being admitted are higher for females than males but without statistical significance. The estimate of $\hat{\sigma}_u^2$ is 1.35 with the standard error of 0.46. An LR test comparing the model with the one-level binomial regression model favors the random-intercept model, indicating that there is a significant variation in the number of admissions between departments.

We can further assess the model fit by performing a residual analysis. For example, here we predict and plot Anscombe residuals.
Anscombe residuals are constructed to be approximately normally distributed, thus residuals that are above two in absolute value are usually considered outliers. In the graph above, the residual for female admissions in department 2 is a clear outlier, suggesting a poor fit for that particular observation; see [ME] meglm postestimation for more information about Anscombe residuals and other model diagnostics tools.

**Covariance structures**

> **Example 4: Growth-curve model with correlated random effects**

Here we extend the model from example 1 of [ME] me to allow for a random slope on week and an unstructured covariance structure between the random intercept and the random slope on week.
. use http://www.stata-press.com/data/r14/pig, clear
(Longitudinal analysis of pig weights)
. mixed weight week || id: week, covariance(unstructured)
Performing EM optimization:
Performing gradient-based optimization:
Iteration 0:  log likelihood = -868.96185
Iteration 1:  log likelihood = -868.96185
Computing standard errors:
Mixed-effects ML regression
Number of obs = 432
Group variable: id
Number of groups = 48
Obs per group:
  min = 9
  avg = 9.0
  max = 9
Wald chi2(1) = 4649.17
Log likelihood = -868.96185  Prob > chi2 = 0.0000

|         | Coef.     | Std. Err. | z    | P>|z|     | [95% Conf. Interval] |
|---------|-----------|-----------|------|---------|----------------------|
| weight  |           |           |      |         |                      |
| week    | 6.209896  | .0910745  | 68.18| 0.000   | 6.031393 6.388399    |
| _cons   | 19.35561  | .3996387  | 48.43| 0.000   | 18.57234 20.13889   |

<table>
<thead>
<tr>
<th>Random-effects Parameters</th>
<th>Estimate</th>
<th>Std. Err.</th>
<th>[95% Conf. Interval]</th>
</tr>
</thead>
<tbody>
<tr>
<td>id: Unstructured</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>var(week)</td>
<td>.3715251</td>
<td>.0812958</td>
<td>.2419532 .570486</td>
</tr>
<tr>
<td>var(_cons)</td>
<td>6.823363</td>
<td>1.566194</td>
<td>4.351297 10.69986</td>
</tr>
<tr>
<td>cov(week,_cons)</td>
<td>-.0984378</td>
<td>.2545767</td>
<td>-.5973991 .4005234</td>
</tr>
<tr>
<td>var(Residual)</td>
<td>1.596829</td>
<td>.123198</td>
<td>1.372735 1.857505</td>
</tr>
</tbody>
</table>

LR test vs. linear model: chi2(3) = 764.58  Prob > chi2 = 0.0000
Note: LR test is conservative and provided only for reference.

The unstructured covariance structure allows for correlation between the random effects. Other covariance structures supported by mixed, besides the default independent, include identity and exchangeable; see [ME] mixed for details. You can also specify multiple random-effects equations at the same level, in which case the covariance types can be combined to form more complex blocked-diagonal covariance structures; see example 5 below.

We can predict the fitted values and plot the estimated regression line for each of the pigs. The fitted values are based on both the fixed and the random effects.
Example 5: Blocked-diagonal covariance structures

In this example, we fit a logistic mixed-effects model with a blocked-diagonal covariance structure of random effects.

We use the data from the 1989 Bangladesh fertility survey (Huq and Cleland 1990), which polled 1,934 Bangladeshi women on their use of contraception. The women sampled were from 60 districts, identified by the variable `district`. Each district contained either urban or rural areas (variable `urban`) or both. The variable `c_use` is the binary response, with a value of 1 indicating contraceptive use. Other covariates include mean-centered `age` and three indicator variables recording number of children. Below we fit a standard logistic regression model amended to have random coefficients on each indicator variable for children and an overall district random intercept.
. use http://www.stata-press.com/data/r14/bangladesh, clear
(Bangladesh Fertility Survey, 1989)
. melogit c_use urban age child* || district: child*, cov(exchangeable)
> || district:, or
Fitting fixed-effects model:
Iteration 0:  log likelihood =  -1229.5485
Iteration 1:  log likelihood =  -1228.5268
Iteration 2:  log likelihood =  -1228.5263
Iteration 3:  log likelihood =  -1228.5263
Refining starting values:
Grid node 0:  log likelihood =  -1234.3979
Fitting full model:
Iteration 0:  log likelihood =  -1234.3979  (not concave)
Iteration 1:  log likelihood =  -1208.0052
Iteration 2:  log likelihood =  -1206.4497
Iteration 3:  log likelihood =  -1206.2417
Iteration 4:  log likelihood =  -1206.2397
Iteration 5:  log likelihood =  -1206.2397
Mixed-effects logistic regression
Number of obs =  1,934
Group variable:  district
Number of groups =  60
Obs per group:
    min =     2
    avg =   32.2
    max =  118
Integration method: mvaghermite  Integration pts. =  7
Log likelihood =  -1206.2397
Wald chi2(5) =   100.01
Prob > chi2 =  0.0000
( 1)  [var(child1[district])]_cons - [var(child3[district])]_cons = 0
( 2)  [cov(child2[district],child1[district])_cons -
    [cov(child3[district],child2[district])_cons = 0
( 3)  [cov(child3,district),child1[district])]_cons -
    [cov(child3[district],child2[district])]_cons = 0
( 4)  [var(child2[district])]_cons - [var(child3[district])]_cons = 0

  c_use   Odds Ratio   Std. Err.      z    P>|z|     [95% Conf. Interval]
----------- ----------- ---------- ------- -------- -----------------------
  urban     2.105163    .2546604    6.15    0.000     1.660796     2.668426
  age       .9735765    .0077461   -3.37    0.001     .9585122     .9888775
  child1    2.992596    .5021490    6.53    0.000     2.153867     4.157931
  child2    3.879345    .7094125    7.41    0.000     2.710815     5.551584
  child3    3.774627    .7055812    7.11    0.000     2.616744     5.444863
  _cons     .1859471    .0274813  -11.38    0.000     .1391841     .2484214

  district
  var(child1)  .0841518    .0880698    0.98    0.328     .654479      .654479
  var(child2)  .0841518    .0880698    0.98    0.328     .654479      .654479
  var(child3)  .0841518    .0880698    0.98    0.328     .654479      .654479
  var(_cons)   .1870273    .0787274  -11.38    0.000     .1391841     .2484214

  district
  cov(child2, child1)  .0616875    .0844681   0.73    0.465    -.1038669     .2272419
  cov(child3, child1)  .0616875    .0844681   0.73    0.465    -.1038669     .2272419
  cov(child3, child2)  .0616875    .0844681   0.73    0.465    -.1038669     .2272419

LR test vs. logistic model:  chi2(3) =   44.57  Prob > chi2 =  0.0000
Note: LR test is conservative and provided only for reference.
The fixed effects can be interpreted just as you would the output from `logit`. Urban women have roughly double the odds of using contraception as compared with their rural counterparts. Having any number of children will increase the odds from three- to fourfold when compared with the base category of no children. Contraceptive use also decreases with age.

Because we specified `cov(exchangeable)`, the estimated variances on each indicator variable for children are constrained to be the same, and the estimated covariances on each indicator variable for children are constrained to be the same. More complex covariance structures with constraints can be specified using `covariance(pattern())` and `covariance(fixed())`; see example 6 below.

Example 6: Meta analysis

In this example, we present a mixed-effects model for meta analysis of clinical trials. The term “meta-analysis” refers to a statistical analysis that involves summary data from similar but independent studies.

Turner et al. (2000) performed a study of nine clinical trials examining the effect of taking diuretics during pregnancy on the risk of pre-eclampsia. The summary data consist of the log odds-ratio (variable `or`) estimated from each study, and the corresponding estimated variance (variable `varor`). The square root of the variance is stored in the variable `std` and the trial identifier is stored in the variable `trial`.

```
use http://www.stata-press.com/data/r14/diuretics
(Meta analysis of clinical trials studying diuretics and pre-eclampsia)
list
```

<table>
<thead>
<tr>
<th>trial</th>
<th>or</th>
<th>varor</th>
<th>std</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>1</td>
<td>.04</td>
<td>.16</td>
</tr>
<tr>
<td>2.</td>
<td>2</td>
<td>-.92</td>
<td>.12</td>
</tr>
<tr>
<td>3.</td>
<td>3</td>
<td>-1.12</td>
<td>.18</td>
</tr>
<tr>
<td>4.</td>
<td>4</td>
<td>-1.47</td>
<td>.3</td>
</tr>
<tr>
<td>5.</td>
<td>5</td>
<td>-1.39</td>
<td>.11</td>
</tr>
<tr>
<td>6.</td>
<td>6</td>
<td>-.3</td>
<td>.01</td>
</tr>
<tr>
<td>7.</td>
<td>7</td>
<td>-.26</td>
<td>.12</td>
</tr>
<tr>
<td>8.</td>
<td>8</td>
<td>1.09</td>
<td>.69</td>
</tr>
<tr>
<td>9.</td>
<td>9</td>
<td>.14</td>
<td>.07</td>
</tr>
</tbody>
</table>

In a random-effects modeling of summary data, the observed log odds-ratios are treated as a continuous outcome and assumed to be normally distributed, and the true treatment effect varies randomly among the trials. The random-effects model can be written as

\[ y_i \sim N(\theta + \nu_i, \sigma_i^2) \]
\[ \nu_i \sim N(0, \tau^2) \]

where \( y_i \) is the observed treatment effect corresponding to the \( i \)th study, \( \theta + \nu_i \) is the true treatment effect, \( \sigma_i^2 \) is the variance of the observed treatment effect, and \( \tau \) is the between-trial variance component. Our aim is to estimate \( \theta \), the global mean.

Notice that the responses \( y_i \) do not provide enough information to estimate this model, because we cannot estimate the group-level variance component from a dataset that contains one observation per group. However, we already have estimates for the \( \sigma_i^2 \)'s, therefore we can constrain each \( \sigma_i \) to
be equal to its estimated value, which will allow us to estimate $\theta$ and $\tau$. We use `meglm` to estimate this model because the `mixed` command does not support constraints.

In `meglm`, one way to constrain a group of individual variances to specific values is by using the fixed covariance structure (an alternative way is to define each constraint individually with the `constraint` command and specify them in the `constraints()` option). The `covariance(fixed())` option requires a Stata matrix defining the constraints, thus we first create matrix $f$ with the values of $\sigma_i$, stored in variable `varor`, on the main diagonal. We will use this matrix to constrain the variances.

```
. mkmat varor, mat(f)
. mat f = diag(f)
```

In the random-effects equation part, we need to specify nine random slopes, one for each trial. Because random-effects equations do not support factor variables (see [U] 11.4.3 Factor variables), we cannot use the `i.trial` notation. Instead, we tabulate the variable `trial` and use the `generate()` option to create nine dummy variables named `tr1`, `tr2`, ..., `tr9`. We can then fit the model. Because the model is computationally demanding, we use Laplacian approximation instead of the default mean-variance adaptive quadrature; see Computation time and the Laplacian approximation above for details.
. qui tabulate trial, gen(tr)
. mglm or || _all: tr1-tr9, nocons cov(fixed(f)) intm(laplace) nocnsreport
Fitting fixed-effects model:
Iteration 0: log likelihood = -10.643432
Iteration 1: log likelihood = -10.643432

Refining starting values:
Grid node 0: log likelihood = -10.205455
Fitting full model:
Iteration 0: log likelihood = -10.205455
Iteration 1: log likelihood = -9.4851561 (backed up)
Iteration 2: log likelihood = -9.4587068
Iteration 3: log likelihood = -9.4552982
Iteration 4: log likelihood = -9.4552759
Iteration 5: log likelihood = -9.4552759

Mixed-effects GLM Number of obs = 9
Family: Gaussian
Link: identity
group variable: _all Number of groups = 1
Obs per group:
  min = 9
  avg = 9.0
  max = 9

Integration method: laplace

Log likelihood = -9.4552759
Wald chi2(0) = .
Prob > chi2 = .

| Coef. | Std. Err. | z   | P>|z|   | [95% Conf. Interval] |
|-------|-----------|-----|-------|----------------------|
| _cons | -.5166151 | .2059448 | -2.51 | .012 | -.9202594  -.1129707 |

<table>
<thead>
<tr>
<th></th>
<th>_cons</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>_all</td>
<td>var(tr1)</td>
<td>.16</td>
<td>(constrained)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>var(tr2)</td>
<td>.12</td>
<td>(constrained)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>var(tr3)</td>
<td>.18</td>
<td>(constrained)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>var(tr4)</td>
<td>.3</td>
<td>(constrained)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>var(tr5)</td>
<td>.11</td>
<td>(constrained)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>var(tr6)</td>
<td>.01</td>
<td>(constrained)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>var(tr7)</td>
<td>.12</td>
<td>(constrained)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>var(tr8)</td>
<td>.69</td>
<td>(constrained)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>var(tr9)</td>
<td>.07</td>
<td>(constrained)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

|     | var(e.or)| .2377469 | .1950926 | .0476023 | 1.187413 |

We estimate $\hat{\theta} = -0.52$, which agrees with the estimate reported by Turner et al. (2000).

We can fit the above model in a more efficient way. We can consider the trials as nine independent random variables, each with variance unity, and each being multiplied by a different standard error. To accomplish this, we treat trial as a random-effects level, use the standard deviations of the log odds-ratios as a random covariate at the trial level, and constrain the variance component of trial to unity.
. constraint 1 _b[var(std[trial]):_cons] = 1
. meglm or || trial: std, nocons constraints(1)

Fitting fixed-effects model:
Iteration 0:  log likelihood = -10.643432
Iteration 1:  log likelihood = -10.643432

Refining starting values:
Grid node 0:  log likelihood = -10.205455

Fitting full model:
Iteration 0:  log likelihood = -10.205455
Iteration 1:  log likelihood = -9.4851164 (backed up)
Iteration 2:  log likelihood = -9.45869
Iteration 3:  log likelihood = -9.4552794
Iteration 4:  log likelihood = -9.4552759
Iteration 5:  log likelihood = -9.4552759

Mixed-effects GLM
Family: Gaussian
Link: identity
Number of obs = 9
Number of groups = 9
Obs per group:  min = 1
avg = 1.0
max = 1
Integration method: mvaghermite
Integration pts. = 7
Wald chi2(0) =  .
Prob > chi2 =  .
Log likelihood = -9.4552759
( 1) [var(std[trial])]:_cons = 1

| or | Coef. | Std. Err. | z   | P>|z| | [95% Conf. Interval] |
|----|-------|-----------|-----|------|----------------------|
| _cons | -.5166151 | .2059448 | -2.51 | 0.012 | -.9202594 - .1129708 |
| trial | var(std) | 1 (constrained) |
| var(e.or) | .2377469 | .1950926 | .0476023 | 1.187413 |

The results are the same, but this model took a fraction of the time compared with the less efficient specification.

Three-level models

The methods we have discussed so far extend from two-level models to models with three or more levels with nested random effects. By “nested”, we mean that the random effects shared within lower-level subgroups are unique to the upper-level groups. For example, assuming that classroom effects would be nested within schools would be natural, because classrooms are unique to schools. Below we illustrate a three-level mixed-effects ordered probit model.

Example 7: Three-level ordinal response model

In this example, we fit a three-level ordered probit model. The data are from the Television, School, and Family Smoking Prevention and Cessation Project (Flay et al. 1988; Rabe-Hesketh and Skrondal 2012, chap. 11), where schools were randomly assigned into one of four groups defined
by two treatment variables. Students within each school are nested in classes, and classes are nested in schools. The dependent variable is the tobacco and health knowledge (THK) scale score collapsed into four ordered categories. We regress the outcome on the treatment variables and their interaction and control for the pretreatment score.

```
. use http://www.stata-press.com/data/r14/tvsfpors, clear
. meoprobit thk prethk cc#tv || school: || class:
Fitting fixed-effects model:
Iteration 0:  log likelihood = -2212.775
Iteration 1:  log likelihood = -2127.8111
Iteration 2:  log likelihood = -2127.7612
Iteration 3:  log likelihood = -2127.7612
Refining starting values:
Grid node 0:  log likelihood = -2195.6424
Fitting full model:
Iteration 0:  log likelihood = -2195.6424  (not concave)
Iteration 1:  log likelihood = -2167.9576  (not concave)
Iteration 2:  log likelihood = -2140.2644  (not concave)
Iteration 3:  log likelihood = -2128.6948  (not concave)
Iteration 4:  log likelihood = -2119.9225
Iteration 5:  log likelihood = -2117.0947
Iteration 6:  log likelihood = -2116.7004
Iteration 7:  log likelihood = -2116.6981
Iteration 8:  log likelihood = -2116.6981
Mixed-effects oprobit regression  Number of obs = 1,600

<table>
<thead>
<tr>
<th>Group Variable</th>
<th>No. of Groups</th>
<th>Minimum</th>
<th>Average</th>
<th>Maximum</th>
</tr>
</thead>
<tbody>
<tr>
<td>school</td>
<td>28</td>
<td>18</td>
<td>57.1</td>
<td>137</td>
</tr>
<tr>
<td>class</td>
<td>135</td>
<td>1</td>
<td>11.9</td>
<td>28</td>
</tr>
</tbody>
</table>
Integration method: mvaghermite
Integration pts. = 7
Wald chi2(4) = 124.20
Log likelihood = -2116.6981
Prob > chi2 = 0.0000

| thk       | Coef.  | Std. Err. | z     | P>|z| | [95% Conf. Interval]  |
|-----------|--------|-----------|-------|------|-----------------------|
| prethk    | .238841| .0231446  | 10.32 | 0.000| .1934784 .2842036      |
| 1.cc      | .5254813| .1285816 | 4.09  | 0.000| .2734659 .7774967      |
| 1.tv      | .1455573| .1255827 | 1.16  | 0.246| -.1005803 .3916949     |
| cc#tv     |        |           |       |      |                       |
| 1 1       | -.2426203| .1811999 | -1.34| 0.181| -.5977656 .1125251     |
| /cut1     | -.074617| .1029791 | -0.72| 0.469| -.2764523 .1272184     |
| /cut2     | .6863046| .1034813 | 6.63  | 0.000| .4834849 .8891242      |
| /cut3     | 1.413686| .1064889 | 13.28 | 0.000| 1.204972 1.622401      |
| school    |        |           |       |      |                       |
| var(_cons)| .0186456| .0160226 |       |      | .0034604 .1004695      |
| school>class|        |           |       |      |                       |
| var(_cons)| .0519974| .0224014 |       |      | .0223496 .1209745      |
LR test vs. oprobit model:  chi2(2) =  22.13  Prob > chi2 = 0.0000
Note: LR test is conservative and provided only for reference.
Notes:
1. Our model now has two random-effects equations, separated by ||. The first is a random intercept (constant only) at the school level (level three), and the second is a random intercept at the class level (level two). The order in which these are specified (from left to right) is significant—meoprobit assumes that class is nested within school.
2. The information on groups is now displayed as a table, with one row for each grouping. You can suppress this table with the nongroup or the noheader option, which will also suppress the rest of the header.
3. The variance-component estimates are now organized and labeled according to level. The variance component for class is labeled school>class to emphasize that classes are nested within schools.

The above extends to models with more than two levels of nesting in the obvious manner, by adding more random-effects equations, each separated by ||. The order of nesting goes from left to right as the groups go from biggest (highest level) to smallest (lowest level).

Crossed-effects models
Not all mixed-effects models contain nested levels of random effects.

Example 8: Crossed random effects
Returning to our longitudinal analysis of pig weights, suppose that we wish to fit

\[
\text{weight}_{ij} = \beta_0 + \beta_1 \text{week}_{ij} + u_i + v_j + \epsilon_{ij}
\]

for the \(i = 1, \ldots, 9\) weeks and \(j = 1, \ldots, 48\) pigs and

\[
\begin{align*}
    u_i &\sim N(0, \sigma_u^2); \\
v_j &\sim N(0, \sigma_v^2); \\
\epsilon_{ij} &\sim N(0, \sigma_\epsilon^2)
\end{align*}
\]

all independently. That is, we assume an overall population-average growth curve \(\beta_0 + \beta_1 \text{week}\) and a random pig-specific shift. In other words, the effect due to week, \(u_i\), is systematic to that week and common to all pigs. The rationale behind (11) could be that, assuming that the pigs were measured contemporaneously, we might be concerned that week-specific random factors such as weather and feeding patterns had significant systematic effects on all pigs.

Model (11) is an example of a two-way crossed-effects model, with the pig effects \(v_j\) being crossed with the week effects \(u_i\). One way to fit such models is to consider all the data as one big cluster, and treat \(u_i\) and \(v_j\) as a series of \(9 + 48 = 57\) random coefficients on indicator variables for week and pig. The random effects \(u\) and the variance components \(G\) are now represented as
\[
\begin{bmatrix}
  u_1 \\
  \vdots \\
  u_9 \\
  v_1 \\
  \vdots \\
  v_{48}
\end{bmatrix}
\sim N(0, G);
\quad G =
\begin{bmatrix}
  \sigma^2_u I_9 & 0 \\
  0 & \sigma^2_v I_{48}
\end{bmatrix}
\]

Because \(G\) is block diagonal, it can be represented as repeated-level equations. All we need is an ID variable to identify all the observations as one big group and a way to tell mixed-effects commands to treat \textit{week} and \textit{pig} as factor variables (or equivalently, as two sets of overparameterized indicator variables identifying weeks and pigs, respectively). The mixed-effects commands support the special group designation \textit{all} for the former and the \texttt{R.varname} notation for the latter.

```stata
. use http://www.stata-press.com/data/r14/pig
(Longitudinal analysis of pig weights)
. mixed weight week || _all: R.id || _all: R.week
Performing EM optimization:
Performing gradient-based optimization:
Iteration 0: log likelihood = -1013.824
Iteration 1: log likelihood = -1013.824
Computing standard errors:
Mixed-effects ML regression Number of obs = 432
Group variable: _all Number of groups = 1
Obs per group:
  min = 432  avg = 432.0  max = 432
Wald chi2(1) = 13258.28 Log likelihood = -1013.824 Prob > chi2 = 0.0000

weight

| Coef.  | Std. Err. | z    | P>|z|  | [95% Conf. Interval] |
|--------|-----------|------|-------|---------------------|
| week   | 6.209896  | .0539313 | 115.14 | 0.000  | 6.104192  | 6.315599   |
| _cons  | 19.35561  | .6333982 | 30.56  | 0.000  | 18.11418  | 20.59705   |

Random-effects Parameters

\begin{array}{lll}
\_all: Identity & \text{Estimate} & \text{Std. Err.} & [95\% \text{ Conf. Interval}] \\
\text{var(R.id)} & 14.83623 & 3.126142 & 9.816733 & 22.42231 \\
\text{var(R.week)} & 0.0849874 & 0.0868856 & 0.0114588 & 0.6303302 \\
\text{var(Residual)} & 4.297328 & 0.3134404 & 3.724888 & 4.957741 \\
\end{array}

LR test vs. linear model: \text{chi2}(2) = 474.85 Prob > \text{chi2} = 0.0000

Note: LR test is conservative and provided only for reference.

We estimate \( \hat{\sigma}^2_u = 0.08 \) and \( \hat{\sigma}^2_v = 14.84 \).

The \texttt{R.varname} notation is equivalent to giving a list of overparameterized (none dropped) indicator variables for use in a random-effects specification. When you use \texttt{R.varname}, mixed-effects commands handle the calculations internally rather than creating the indicators in the data. Because the set of indicators is overparameterized, \texttt{R.varname} implies \texttt{noconstant}.
Note that the column dimension of our random-effects design is 57. Computation time and memory requirements grow (roughly) quadratically with the dimension of the random effects. As a result, fitting such crossed-effects models is feasible only when the total column dimension is small to moderate. For this reason, mixed-effects commands use the Laplacian approximation as the default estimation method for crossed-effects models; see *Computation time and the Laplacian approximation* above for more details.

It is often possible to rewrite a mixed-effects model in a way that is more computationally efficient. For example, we can treat pigs as nested within the _all group, yielding the equivalent and more efficient (total column dimension 10) way to fit (11):

```
. mixed weight week || _all: R.week || id:
```

The results of both estimations are identical, but the latter specification, organized at the cluster (pig) level with random-effects dimension 1 (a random intercept) is much more computationally efficient. Whereas with the first form we are limited in how many pigs we can analyze, there is no such limitation with the second form.

All the mixed-effects commands—except *mixed*, *meqrlogit*, and *meqroisson*—automatically attempt to recast the less efficient model specification into a more efficient one. However, this automatic conversion may not be sufficient for some complicated mixed-effects specifications, especially if both crossed and nested effects are involved. Therefore, we strongly encourage you to always specify the more efficient syntax; see *Rabe-Hesketh and Skrondal* (2012) and *Marchenko* (2006) for additional techniques to make calculations more efficient in more complex mixed-effects models.

### Acknowledgments

We are indebted to Sophia Rabe-Hesketh of the University of California, Berkeley; Anders Skrondal of the University of Oslo and the Norwegian Institute of Public Health; and Andrew Pickles of the University of Manchester for their extensive body of work in Stata, both previous and ongoing, in this area.

### References


**Also see**

[ME] **Glossary**
Description

mecloglog fits mixed-effects models for binary or binomial responses. The conditional distribution of the response given the random effects is assumed to be Bernoulli, with probability of success determined by the inverse complementary log-log function.

Quick start

Two-level complementary log-log model of y on x with random intercepts by lev2
mecloglog y x || lev2:

Add binary variable a and random coefficients for a
mecloglog y x a || lev2: a

As above, but allow the random effects to be correlated
mecloglog y x a || lev2: a, covariance(unstructured)

Three-level random-intercept model of y on x with lev2 nested within lev3
mecloglog y x || lev3: || lev2:

Crossed-effects model of y on x with two-way crossed random effects by factors a and b
mecloglog y x || _all:R.a || b:

Menu

Statistics > Multilevel mixed-effects models > Complementary log-log regression
mecloglog *depvar* fe\_equation [ || re\_equation ] [ || re\_equation . . . ] [ , options ]

where the syntax of *fe\_equation* is

\[ \text{indepvars} \] \[ \text{if} \] \[ \text{in} \] \[ \text{weight} \] [ , fe\_options ]

and the syntax of *re\_equation* is one of the following:

for random coefficients and intercepts

*levelvar*: \[ varlist \] [ , re\_options ]

for random effects among the values of a factor variable

*levelvar*: R. *varname*

*levelvar* is a variable identifying the group structure for the random effects at that level or is *_all* representing one group comprising all observations.

<table>
<thead>
<tr>
<th>fe_options</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>noconstant</strong></td>
<td>suppress constant term from the fixed-effects equation</td>
</tr>
<tr>
<td><strong>offset(varname)</strong></td>
<td>include <em>varname</em> in model with coefficient constrained to 1</td>
</tr>
<tr>
<td><strong>asis</strong></td>
<td>retain perfect predictor variables</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>re_options</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>covariance(vartype)</strong></td>
<td>variance–covariance structure of the random effects</td>
</tr>
<tr>
<td><strong>noconstant</strong></td>
<td>suppress constant term from the random-effects equation</td>
</tr>
<tr>
<td><strong>fweight(varname)</strong></td>
<td>frequency weights at higher levels</td>
</tr>
<tr>
<td><strong>iweight(varname)</strong></td>
<td>importance weights at higher levels</td>
</tr>
<tr>
<td><strong>pweight(varname)</strong></td>
<td>sampling weights at higher levels</td>
</tr>
</tbody>
</table>
### mecloglog — Multilevel mixed-effects complementary log-log regression

#### options

<table>
<thead>
<tr>
<th>Description</th>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model</td>
<td></td>
</tr>
<tr>
<td>`binomial(varname</td>
<td>#)`</td>
</tr>
<tr>
<td><code>constraints(constraints)</code></td>
<td>apply specified linear constraints</td>
</tr>
<tr>
<td><code>collinear</code></td>
<td>keep collinear variables</td>
</tr>
<tr>
<td>SE/Robust</td>
<td></td>
</tr>
<tr>
<td><code>vce(vcetype)</code></td>
<td><code>vcetype</code> may be <code>oim</code>, <code>robust</code>, or <code>cluster clustvar</code></td>
</tr>
<tr>
<td>Reporting</td>
<td></td>
</tr>
<tr>
<td><code>level(#)</code></td>
<td>set confidence level; default is <code>level(95)</code></td>
</tr>
<tr>
<td><code>eform</code></td>
<td>report exponentiated coefficients</td>
</tr>
<tr>
<td><code>nocsrreport</code></td>
<td>do not display constraints</td>
</tr>
<tr>
<td><code>notable</code></td>
<td>suppress coefficient table</td>
</tr>
<tr>
<td><code>noheader</code></td>
<td>suppress output header</td>
</tr>
<tr>
<td><code>nogroup</code></td>
<td>suppress table summarizing groups</td>
</tr>
<tr>
<td><code>nlrtest</code></td>
<td>do not perform likelihood-ratio test comparing with complementary log-log regression</td>
</tr>
<tr>
<td><code>display_options</code></td>
<td>control columns and column formats, row spacing, line width, display of omitted variables and base and empty cells, and factor-variable labeling</td>
</tr>
<tr>
<td>Integration</td>
<td></td>
</tr>
<tr>
<td><code>intmethod(intmethod)</code></td>
<td>integration method</td>
</tr>
<tr>
<td><code>intpoints(#)</code></td>
<td>set the number of integration (quadrature) points for all levels; default is <code>intpoints(7)</code></td>
</tr>
<tr>
<td>Maximization</td>
<td></td>
</tr>
<tr>
<td><code>maximize_options</code></td>
<td>control the maximization process; seldom used</td>
</tr>
<tr>
<td><code>startvalues(svmethod)</code></td>
<td>method for obtaining starting values</td>
</tr>
<tr>
<td><code>startgrid[ gridspec]</code></td>
<td>perform a grid search to improve starting values</td>
</tr>
<tr>
<td><code>noestimate</code></td>
<td>do not fit the model; show starting values instead</td>
</tr>
<tr>
<td><code>dnumerical</code></td>
<td>use numerical derivative techniques</td>
</tr>
<tr>
<td><code>coeflegend</code></td>
<td>display legend instead of statistics</td>
</tr>
</tbody>
</table>

#### vartype

<table>
<thead>
<tr>
<th>Description</th>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>independent</code></td>
<td>one unique variance parameter per random effect, all covariances 0; the default unless the R. notation is used</td>
</tr>
<tr>
<td><code>exchangeable</code></td>
<td>equal variances for random effects, and one common pairwise covariance</td>
</tr>
<tr>
<td><code>identity</code></td>
<td>equal variances for random effects, all covariances 0; the default if the R. notation is used</td>
</tr>
<tr>
<td><code>unstructured</code></td>
<td>all variances and covariances to be distinctly estimated</td>
</tr>
<tr>
<td><code>fixed(matname)</code></td>
<td>user-selected variances and covariances constrained to specified values; the remaining variances and covariances unrestricted</td>
</tr>
<tr>
<td><code>pattern(matname)</code></td>
<td>user-selected variances and covariances constrained to be equal; the remaining variances and covariances unrestricted</td>
</tr>
</tbody>
</table>
intmethod

<table>
<thead>
<tr>
<th>intmethod</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mvaghermite</td>
<td>mean–variance adaptive Gauss–Hermite quadrature; the default unless a crossed random-effects model is fit</td>
</tr>
<tr>
<td>mcaghermite</td>
<td>mode-curvature adaptive Gauss–Hermite quadrature</td>
</tr>
<tr>
<td>ghermite</td>
<td>nonadaptive Gauss–Hermite quadrature</td>
</tr>
<tr>
<td>laplace</td>
<td>Laplacian approximation; the default for crossed random-effects models</td>
</tr>
</tbody>
</table>

indepvars may contain factor variables; see [U] 11.4.3 Factor variables.
depvar, indepvars, and varlist may contain time-series operators; see [U] 11.4.4 Time-series varlists.

by and svy are allowed; see [U] 11.1.10 Prefix commands.
vce() and weights are not allowed with the svy prefix; see [SVY] svy.

fweights, iweights, and pweights are allowed; see [U] 11.1.6 weight. Only one type of weight may be specified.

startvalues(), startgrid, noestimate, dnumerical, and coeflegend do not appear in the dialog box.

See [U] 20 Estimation and postestimation commands for more capabilities of estimation commands.

Options

Model

noconstant suppresses the constant (intercept) term and may be specified for the fixed-effects equation and for any of or all the random-effects equations.

offset(varname) specifies that varname be included in the fixed-effects portion of the model with the coefficient constrained to be 1.

asis forces retention of perfect predictor variables and their associated, perfectly predicted observations and may produce instabilities in maximization; see [R] probit.

covariance(vartype) specifies the structure of the covariance matrix for the random effects and may be specified for each random-effects equation. vartype is one of the following: independent, exchangeable, identity, unstructured, fixed(matname), or pattern(matname).

covariance(independent) covariance structure allows for a distinct variance for each random effect within a random-effects equation and assumes that all covariances are 0. The default is covariance(independent) unless a crossed random-effects model is fit, in which case the default is covariance(identity).

covariance(exchangeable) structure specifies one common variance for all random effects and one common pairwise covariance.

covariance(identity) is short for “multiple of the identity”; that is, all variances are equal and all covariances are 0.

covariance(unstructured) allows for all variances and covariances to be distinct. If an equation consists of \( p \) random-effects terms, the unstructured covariance matrix will have \( p(p + 1)/2 \) unique parameters.

covariance(fixed(matname)) and covariance(pattern(matname)) covariance structures provide a convenient way to impose constraints on variances and covariances of random effects. Each specification requires a matname that defines the restrictions placed on variances and covariances. Only elements in the lower triangle of matname are used, and row and column names of matname are ignored. A missing value in matname means that a given element is unrestricted.
In a fixed(matname) covariance structure, (co)variance \((i,j)\) is constrained to equal the value specified in the \(i,j\)th entry of matname. In a pattern(matname) covariance structure, (co)variances \((i,j)\) and \((k,l)\) are constrained to be equal if matname\([i,j]\) = matname\([k,l]\).

\textbf{fweight(varname)} specifies frequency weights at higher levels in a multilevel model, whereas frequency weights at the first level (the observation level) are specified in the usual manner, for example, \([fw=fwtvar1]\). varname can be any valid Stata variable name, and you can specify fweight() at levels two and higher of a multilevel model. For example, in the two-level model

\begin{verbatim}
.mecmd fixed_portion [fw = wt1] || school: ... , fweight(wt2) ...
\end{verbatim}

the variable wt1 would hold the first-level (the observation-level) frequency weights, and wt2 would hold the second-level (the school-level) frequency weights.

\textbf{iweight(varname)} specifies importance weights at higher levels in a multilevel model, whereas importance weights at the first level (the observation level) are specified in the usual manner, for example, \([iw=iwtvar1]\). varname can be any valid Stata variable name, and you can specify iweight() at levels two and higher of a multilevel model. For example, in the two-level model

\begin{verbatim}
.mecmd fixed_portion [iw = wt1] || school: ... , iweight(wt2) ...
\end{verbatim}

the variable wt1 would hold the first-level (the observation-level) importance weights, and wt2 would hold the second-level (the school-level) importance weights.

\textbf{pweight(varname)} specifies sampling weights at higher levels in a multilevel model, whereas sampling weights at the first level (the observation level) are specified in the usual manner, for example, \([pw=pwtvar1]\). varname can be any valid Stata variable name, and you can specify pweight() at levels two and higher of a multilevel model. For example, in the two-level model

\begin{verbatim}
.mecmd fixed_portion [pw = wt1] || school: ... , pweight(wt2) ...
\end{verbatim}

variable wt1 would hold the first-level (the observation-level) sampling weights, and wt2 would hold the second-level (the school-level) sampling weights.

\textbf{binomial(varname | #)} specifies that the data are in binomial form; that is, \texttt{depvar} records the number of successes from a series of binomial trials. This number of trials is given either as varname, which allows this number to vary over the observations, or as the constant #. If binomial() is not specified (the default), \texttt{depvar} is treated as Bernoulli, with any nonzero, nonmissing values indicating positive responses.

\textbf{constraints(constraints), collinear; see [R] estimation options.}

\begin{verbatim}
\underline{SE/Robust}
\end{verbatim}
\textbf{vce(vcetype)} specifies the type of standard error reported, which includes types that are derived from asymptotic theory (oim), that are robust to some kinds of misspecification (robust), and that allow for intragroup correlation (cluster clustvar); see [R] vce_option. If vce(robust) is specified, robust variances are clustered at the highest level in the multilevel model.

\begin{verbatim}
\underline{Reporting}
\end{verbatim}
\textbf{level(#); see [R] estimation options.}
\textbf{eform} reports exponentiated coefficients and corresponding standard errors and confidence intervals. This option may be specified either at estimation or upon replay.
\textbf{nocnsreport; see [R] estimation options.}
\textbf{notable} suppresses the estimation table, either at estimation or upon replay.
\textbf{noheader} suppresses the output header, either at estimation or upon replay.
nogroup suppresses the display of group summary information (number of groups, average group size, minimum, and maximum) from the output.

nolrtest prevents mecloglog from performing a likelihood-ratio test that compares the mixed-effects complementary log-log model with standard (marginal) complementary log-log regression. This option may also be specified upon replay to suppress this test from the output.

display_options: noic, nopvalues, noomitted, vsquish, noemptycells, baselevels, allbaselevels, oflabel, fwrap(#), fwrapon(style), cformat(’%fmt’), pformat(’%fmt’), sformat(’%fmt’), and nolstretch; see [R] estimation options.

Integration

tintmethod(intmethod) specifies the integration method to be used for the random-effects model. mvaghermite performs mean–variance adaptive Gauss–Hermite quadrature; mcaghermite performs mode-curvature adaptive Gauss–Hermite quadrature; ghermite performs nonadaptive Gauss–Hermite quadrature; and laplace performs the Laplacian approximation, equivalent to mode-curvature adaptive Gaussian quadrature with one integration point.

The default integration method is mvaghermite unless a crossed random-effects model is fit, in which case the default integration method is laplace. The Laplacian approximation has been known to produce biased parameter estimates; however, the bias tends to be more prominent in the estimates of the variance components rather than in the estimates of the fixed effects.

For crossed random-effects models, estimation with more than one quadrature point may be prohibitively intensive even for a small number of levels. For this reason, the integration method defaults to the Laplacian approximation. You may override this behavior by specifying a different integration method.

intpoints(#) sets the number of integration points for quadrature. The default is intpoints(7), which means that seven quadrature points are used for each level of random effects. This option is not allowed with intmethod(laplace).

The more integration points, the more accurate the approximation to the log likelihood. However, computation time increases as a function of the number of quadrature points raised to a power equaling the dimension of the random-effects specification. In crossed random-effects models and in models with many levels or many random coefficients, this increase can be substantial.

Maximization

maximize_options: difficult, technique(algorithm_spec), iterate(#), [no]log, trace, gradient, showstep, hessian, showtolerance, tolerance(#), ltolerance(#), nrtolerance(#), nonrtolerance, and from(init_specs); see [R] maximize. Those that require special mention for mecloglog are listed below.

from() accepts a properly labeled vector of initial values or a list of coefficient names with values. A list of values is not allowed.

The following options are available with mecloglog but are not shown in the dialog box: startvalues(svmethod), startgrid[ gridspec], noestimate, and dnumerical; see [ME] meglm.

coeflegend; see [R] estimation options.
Remarks and examples

For a general introduction to `me` commands, see [ME] `me`.

`mecloglog` is a convenience command for `meglm` with a cloglog link and a bernoulli or binomial family; see [ME] `meglm`.

Remarks are presented under the following headings:

- Introduction
- Two-level models
- Three-level models

Introduction

Mixed-effects complementary log-log regression is complementary log-log regression containing both fixed effects and random effects. In longitudinal data and panel data, random effects are useful for modeling intracluster correlation; that is, observations in the same cluster are correlated because they share common cluster-level random effects.

Comprehensive treatments of mixed models are provided by, for example, Searle, Casella, and McCulloch (1992); Verbeke and Molenberghs (2000); Raudenbush and Bryk (2002); Demidenko (2004); Hedeker and Gibbons (2006); McCulloch, Searle, and Neuhaus (2008); and Rabe-Hesketh and Skrondal (2012). Guo and Zhao (2000) and Rabe-Hesketh and Skrondal (2012, chap. 10) are good introductory readings on applied multilevel modeling of binary data.

`mecloglog` allows for not just one, but many levels of nested clusters of random effects. For example, in a three-level model you can specify random effects for schools and then random effects for classes nested within schools. In this model, the observations (presumably, the students) comprise the first level, the classes comprise the second level, and the schools comprise the third.

However, for simplicity, we here consider the two-level model, where for a series of $M$ independent clusters, and conditional on a set of fixed effects $x_{ij}$ and a set of random effects $u_{j}$,

$$
Pr(y_{ij} = 1 | x_{ij}, u_{j}) = H(x_{ij} \beta + z_{ij} u_{j})
$$

for $j = 1, \ldots, M$ clusters, with cluster $j$ consisting of $i = 1, \ldots, n_{j}$ observations. The responses are the binary-valued $y_{ij}$, and we follow the standard Stata convention of treating $y_{ij} = 1$ if $depvar_{ij} \neq 0$ and treating $y_{ij} = 0$ otherwise. The $1 \times p$ row vector $x_{ij}$ are the covariates for the fixed effects, analogous to the covariates you would find in a standard cloglog regression model, with regression coefficients (fixed effects) $\beta$. For notational convenience here and throughout this manual entry, we suppress the dependence of $y_{ij}$ on $x_{ij}$.

The $1 \times q$ vector $z_{ij}$ are the covariates corresponding to the random effects and can be used to represent both random intercepts and random coefficients. For example, in a random-intercept model, $z_{ij}$ is simply the scalar 1. The random effects $u_{j}$ are $M$ realizations from a multivariate normal distribution with mean $0$ and $q \times q$ variance matrix $\Sigma$. The random effects are not directly estimated as model parameters but are instead summarized according to the unique elements of $\Sigma$, known as variance components. One special case of (1) places $z_{ij} = x_{ij}$, so that all covariate effects are essentially random and distributed as multivariate normal with mean $\beta$ and variance $\Sigma$.

Finally, because this is cloglog regression, $H(\cdot)$ is the inverse of the complementary log-log function that maps the linear predictor to the probability of a success ($y_{ij} = 1$) with $H(v) = 1 - \exp\{-\exp(v)\}$.

Model (1) may also be stated in terms of a latent linear response, where only $y_{ij} = I(y_{ij}^* > 0)$ is observed for the latent

$$
y_{ij}^* = x_{ij} \beta + z_{ij} u_{j} + \epsilon_{ij}
$$
The errors $\epsilon_{ij}$ are independent and identically extreme-value (Gumbel) distributed with the mean equal to Euler’s constant and variance $\sigma^2_\epsilon = \pi^2/6$, independently of $u_j$. This nonsymmetric error distribution is an alternative to the symmetric error distribution underlying logistic and probit analysis and is usually used when the positive (or negative) outcome is rare.

Model (1) is an example of a generalized linear mixed model (GLMM), which generalizes the linear mixed-effects (LME) model to non-Gaussian responses. You can fit LMEs in Stata by using mixed and fit GLMMs by using meglm. Because of the relationship between LMEs and GLMMs, there is insight to be gained through examination of the linear mixed model. This is especially true for Stata users because the terminology, syntax, options, and output for fitting these types of models are nearly identical. See [ME] mixed and the references therein, particularly in Introduction, for more information.

Log-likelihood calculations for fitting any generalized mixed-effects model require integrating out the random effects. One widely used modern method is to directly estimate the integral required to calculate the log likelihood by Gauss–Hermite quadrature or some variation thereof. Because the log likelihood itself is estimated, this method has the advantage of permitting likelihood-ratio tests for comparing nested models. Also, if done correctly, quadrature approximations can be quite accurate, thus minimizing bias.

mecloglog supports three types of Gauss–Hermite quadrature and the Laplacian approximation method; see Methods and formulas of [ME] meglm for details. The simplest random-effects model you can fit using mecloglog is the two-level model with a random intercept,

$$\Pr(y_{ij} = 1|u_j) = H(x_{ij}\beta + u_j)$$

This model can also be fit using xtcloglog with the re option; see [XT] xtcloglog.

Below we present two short examples of mixed-effects cloglog regression; refer to [ME] melogit for additional examples including crossed-effects models and to [ME] me and [ME] meglm for examples of other random-effects models.

Two-level models

We begin with a simple application of (1) as a two-level model, because a one-level model, in our terminology, is just standard cloglog regression; see [R] cloglog.

> Example 1

In example 1 of [XT] xtcloglog, we analyze unionization of women in the United States over the period 1970–1988. The women are identified by the variable idcode. Here we refit that model with mecloglog. Because the original example used 12 integration points by default, we request 12 integration points as well.
. use http://www.stata-press.com/data/r14/union
(NLS Women 14-24 in 1968)
. mecloglog union age grade not_smsa south##c.year || idcode:, intpoints(12)
Fitting fixed-effects model:
Iteration 0: log likelihood = -14237.139
Iteration 1: log likelihood = -13546.159
Iteration 2: log likelihood = -13540.611
Iteration 3: log likelihood = -13540.607
Iteration 4: log likelihood = -13540.607
Refining starting values:
Grid node 0: log likelihood = -11104.448
Fitting full model:
Iteration 0: log likelihood = -11104.448
Iteration 1: log likelihood = -10617.891
Iteration 2: log likelihood = -10537.919
Iteration 3: log likelihood = -10535.946
Iteration 4: log likelihood = -10535.941
Iteration 5: log likelihood = -10535.941
Mixed-effects cloglog regression
Number of obs = 26,200
Group variable: idcode Number of groups = 4,434
Obs per group:
  min = 1
  avg = 5.9
  max = 12
Integration method: mvaghermite
Integration pts. = 12
Log likelihood = -10535.941
Wald chi2(6) = 248.12
Prob > chi2 = 0.0000

|            | Coef. | Std. Err. | z     | P>|z|  | [95% Conf. Interval] |
|------------|-------|-----------|-------|------|----------------------|
| union      |       |           |       |      |                      |
| age        | .0128542 | .0119441  | 1.08  | 0.282 | -.0105559 - .0362642 |
| grade      | .0699965 | .0138551  | 5.05  | 0.000 | .0428409 - .097152  |
| not_smsa   | -.1982009 | .0649258 | -3.05 | 0.002 | - .3254531 - .0709488 |
| 1.south    | -2.049901 | .4892644 | -4.19 | 0.000 | -3.008842 -1.090961  |
| year       | -.0006158 | .0123999 | -0.05 | 0.960 | - .0249191 .0236875 |
| south#c.year 1 | .0164457 | .0060685 | 2.71  | 0.007 | .0045516 .0283399   |
| _cons      | -3.277375 | .6610552 | -4.96 | 0.000 | -4.57302 -1.981731  |
| idcode     | var(_cons) | 3.489803 | .1630921 | 3.184351 | 3.824555 |

LR test vs. cloglog model: chibar2(01) = 6009.33 Prob >= chibar2 = 0.0000

The estimates are practically the same. xtcloglog reports the estimated variance component as a standard deviation, \( \hat{\sigma}_u = 1.86 \). mecloglog reports \( \hat{\sigma}_u^2 = 3.49 \), the square root of which is 1.87. We find that age and education each have a positive effect on union membership, although the former is not statistically significant. Women who live outside of metropolitan areas are less likely to unionize.

The estimated variance of the random intercept at the individual level, \( \sigma^2 \), is 3.49 with standard error 0.16. The reported likelihood-ratio test shows that there is enough variability between women to favor a mixed-effects cloglog regression over an ordinary cloglog regression; see Distribution theory for likelihood-ratio test in [ME] me for a discussion of likelihood-ratio testing of variance components.
Three-level models

Two-level models extend naturally to models with three or more levels with nested random effects. Below we analyze the data from example 2 of [ME]melogit with mecloglog.

Example 2

Rabe-Hesketh, Toulopoulou, and Murray (2001) analyzed data from a study that measured the cognitive ability of patients with schizophrenia compared with their relatives and control subjects. Cognitive ability was measured as the successful completion of the “Tower of London”, a computerized task, measured at three levels of difficulty. For all but one of the 226 subjects, there were three measurements (one for each difficulty level). Because patients’ relatives were also tested, a family identifier, family, was also recorded.

We fit a cloglog model with response dtlm, the indicator of cognitive function, and with covariates difficulty and a set of indicator variables for group, with the controls (group==1) being the base category. We also allow for random effects due to families and due to subjects within families.
. use http://www.stata-press.com/data/r14/towerlondon
   (Tower of London data)
. mecloglog dtlm difficulty i.group || family: || subject:

Fitting fixed-effects model:
Iteration 0:  log likelihood = -337.21921
Iteration 1:  log likelihood = -313.79023
Iteration 2:  log likelihood = -313.56906
Iteration 3:  log likelihood = -313.56888
Iteration 4:  log likelihood = -313.56888

Refining starting values:
Grid node 0:  log likelihood = -314.57061

Fitting full model:
Iteration 0:  log likelihood = -314.57061  (not concave)
Iteration 1:  log likelihood = -308.82101
Iteration 2:  log likelihood = -305.71841
Iteration 3:  log likelihood = -305.26804
Iteration 4:  log likelihood = -305.26516
Iteration 5:  log likelihood = -305.26516

Mixed-effects cloglog regression

<table>
<thead>
<tr>
<th>Group Variable</th>
<th>No. of Groups</th>
<th>Observations per Group</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Minimum</td>
<td>Average</td>
</tr>
<tr>
<td>family</td>
<td>118</td>
<td>2</td>
</tr>
<tr>
<td>subject</td>
<td>226</td>
<td>2</td>
</tr>
</tbody>
</table>

Integration method: mvaghermite
Integration pts. = 7
Log likelihood = -305.26516
Wald chi2(3) = 83.32
Prob > chi2 = 0.0000

| dtlm          | Coef.    | Std. Err. | z       | P>|z|   | [95% Conf. Interval] |
|---------------|----------|-----------|---------|-------|----------------------|
| difficulty    | -1.342844 | .1501508  | -8.94   | 0.000 | -1.637135 -1.048554  |
| group         |          |           |         |       |                      |
| 2             | -.1331007 | .269389   | -0.49   | 0.621 | -.6610935 .3948922   |
| 3             | -.7714314 | .3097099  | -2.49   | 0.013 | -1.378452 -.164411   |
| _cons         | -1.6718   | .2290325  | -7.30   | 0.000 | -2.120695 -1.222905  |
| family>       |          |           |         |       |                      |
| var(_cons)    | .2353453  | .2924064  | .0206122 | 2.687117 |
| family>       |          |           |         |       |                      |
| subject       |          |           |         |       |                      |
| var(_cons)    | .7737687  | .4260653  | .2629714 | 2.276742 |

LR test vs. cloglog model: chi2(2) = 16.61
Prob > chi2 = 0.0002
Note: LR test is conservative and provided only for reference.

Notes:
1. This is a three-level model with two random-effects equations, separated by ||. The first is a random intercept (constant only) at the family level, and the second is a random intercept at the subject level. The order in which these are specified (from left to right) is significant—mecloglog assumes that subject is nested within family.
2. The information on groups is now displayed as a table, with one row for each upper level. Among other things, we see that we have 226 subjects from 118 families. You can suppress this table with the nogroup or the noheader option, which will suppress the rest of the header as well.

After adjusting for the random-effects structure, the probability of successful completion of the Tower of London decreases dramatically as the level of difficulty increases. Also, schizophrenics (group==3) tended not to perform as well as the control subjects.

The above extends to models with more than two levels of nesting in the obvious manner, by adding more random-effects equations, each separated by ||. The order of nesting goes from left to right as the groups go from biggest (highest level) to smallest (lowest level).

Stored results

mecloglog stores the following in e():

Scalars

- `e(N)` number of observations
- `e(k)` number of parameters
- `e(k_dv)` number of dependent variables
- `e(k_eq)` number of equations in `e(b)`
- `e(k_eq_model)` number of equations in overall model test
- `e(k_f)` number of fixed-effects parameters
- `e(k_r)` number of random-effects parameters
- `e(k_rs)` number of variances
- `e(k_rc)` number of covariances
- `e(df_m)` model degrees of freedom
- `e(ll)` log likelihood
- `e(N_clust)` number of clusters
- `e(chi2)` $\chi^2$
- `e(p)` significance
- `e(ll_c)` log likelihood, comparison model
- `e(chi2_c)` $\chi^2$, comparison model
- `e(df_c)` degrees of freedom, comparison model
- `e(p_c)` significance, comparison model
- `e(rank)` rank of `e(V)`
- `e(ic)` number of iterations
- `e(rc)` return code
- `e(converged)` 1 if converged, 0 otherwise

Macros

- `e(cmd)` meglm
- `e(cmd2)` mecloglog
- `e(cmdline)` command as typed
- `e(depvar)` name of dependent variable
- `e(wtype)` weight type
- `e(wexp)` weight expression (first-level weights)
- `e(fweightk)` fweight variable for kth highest level, if specified
- `e(iweightk)` iweight variable for kth highest level, if specified
- `e(pweightk)` pweight variable for kth highest level, if specified
- `e(covariates)` list of covariates
- `e(ivars)` grouping variables
- `e(model)` cloglog
- `e(title)` title in estimation output
- `e(link)` cloglog
- `e(family)` bernoulli or binomial
- `e(clustvar)` name of cluster variable
- `e(offset)` offset
- `e(binomial)` binomial number of trials
Methods and formulas

Model (1) assumes Bernoulli data, a special case of the binomial. Because binomial data are also supported by `mecloglog` (option `binomial()`), the methods presented below are in terms of the more general binomial mixed-effects model.

For a two-level binomial model, consider the response $y_{ij}$ as the number of successes from a series of $r_{ij}$ Bernoulli trials (replications). For cluster $j$, $j = 1, \ldots, M$, the conditional distribution of $y_j = (y_{j1}, \ldots, y_{jn_j})'$, given a set of cluster-level random effects $u_j$, is

$$f(y_j|u_j) = \prod_{i=1}^{n_j} \left[ \binom{r_{ij}}{y_{ij}} H(\eta_{ij})^{y_{ij}} (1 - H(\eta_{ij}))^{r_{ij} - y_{ij}} \right]$$

$$= \exp \left( \sum_{i=1}^{n_j} y_{ij} \log \{ H(\eta_{ij}) \} - (r_{ij} - y_{ij}) \exp(\eta_{ij}) + \log \binom{r_{ij}}{y_{ij}} \right)$$

for $\eta_{ij} = x_{ij} \beta + z_{ij} u_j + \text{offset}_{ij}$ and $H(v) = 1 - \exp\{-\exp(v)\}$. 
Defining \( r_j = (r_{j1}, \ldots, r_{jn_j})' \) and
\[
c(y_j, r_j) = \sum_{i=1}^{n_j} \log \left( \frac{r_{ij}}{y_{ij}} \right)
\]
where \( c(y_j, r_j) \) does not depend on the model parameters, we can express the above compactly in matrix notation,
\[
f(y_j | u_j) = \exp \left[ y_j' \log \{ H(\eta_j) \} - (r_j - y_j)' \exp(\eta_j) + c(y_j, r_j) \right]
\]
where \( \eta_j \) is formed by stacking the row vectors \( \eta_{ij} \). We extend the definitions of the functions \( H(\cdot) \), \( \log(\cdot) \), and \( \exp(\cdot) \) to be vector functions where necessary.

Because the prior distribution of \( u_j \) is multivariate normal with mean \( 0 \) and \( q \times q \) variance matrix \( \Sigma \), the likelihood contribution for the \( j \)th cluster is obtained by integrating \( u_j \) out of the joint density \( f(y_j, u_j) \),
\[
L_j(\beta, \Sigma) = (2\pi)^{-q/2} |\Sigma|^{-1/2} \int f(y_j | u_j) \exp \left( -u_j' \Sigma^{-1} u_j / 2 \right) du_j
\]
\[
= \exp \left\{ c(y_j, r_j) \right\} (2\pi)^{-q/2} |\Sigma|^{-1/2} \int \exp \left\{ h(\beta, \Sigma, u_j) \right\} du_j
\]
where
\[
h(\beta, \Sigma, u_j) = y_j' \log \{ H(\eta_j) \} - (r_j - y_j)' \exp(\eta_j) - u_j' \Sigma^{-1} u_j / 2
\]
and for convenience, in the arguments of \( h(\cdot) \) we suppress the dependence on the observable data \( (y_j, r_j, X_j, Z_j) \).

The integration in (2) has no closed form and thus must be approximated. \texttt{mecloglog} offers four approximation methods: mean–variance adaptive Gauss–Hermite quadrature (default), mode-curvature adaptive Gauss–Hermite quadrature, nonadaptive Gauss–Hermite quadrature, and Laplacian approximation.

The Laplacian approximation is based on a second-order Taylor expansion of \( h(\beta, \Sigma, u_j) \) about the value of \( u_j \) that maximizes it; see \textit{Methods and formulas} in \texttt{ME megglm} for details.

Gaussian quadrature relies on transforming the multivariate integral in (2) into a set of nested univariate integrals. Each univariate integral can then be evaluated using a form of Gaussian quadrature; see \textit{Methods and formulas} in \texttt{ME megglm} for details.

The log likelihood for the entire dataset is simply the sum of the contributions of the \( M \) individual clusters, namely, \( \mathcal{L}(\beta, \Sigma) = \sum_{j=1}^{M} L_j(\beta, \Sigma) \).

Maximization of \( \mathcal{L}(\beta, \Sigma) \) is performed with respect to \( (\beta, \sigma^2) \), where \( \sigma^2 \) is a vector comprising the unique elements of \( \Sigma \). Parameter estimates are stored in \( e(\beta) \) as \( (\hat{\beta}, \hat{\sigma}^2) \), with the corresponding variance–covariance matrix stored in \( e(V) \).

\texttt{mecloglog} supports multilevel weights and survey data; see \textit{Methods and formulas} in \texttt{ME megglm} for details.
References


Also see

[ME] mecloglog postestimation — Postestimation tools for mecloglog
[ME] melogit — Multilevel mixed-effects logistic regression
[ME] meprobit — Multilevel mixed-effects probit regression
[ME] me — Introduction to multilevel mixed-effects models
[SEM] intro 5 — Tour of models (Multilevel mixed-effects models)
[SVY] svy estimation — Estimation commands for survey data
[XT] xtcloglog — Random-effects and population-averaged cloglog models
[U] 20 Estimation and postestimation commands
Postestimation commands

The following postestimation command is of special interest after `mecloglog`:

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>estat group</code></td>
<td>summarize the composition of the nested groups</td>
</tr>
</tbody>
</table>

The following standard postestimation commands are also available:

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>contrast</code></td>
<td>contrasts and ANOVA-style joint tests of estimates</td>
</tr>
<tr>
<td><code>estat ic</code></td>
<td>Akaike’s and Schwarz’s Bayesian information criteria (AIC and BIC)</td>
</tr>
<tr>
<td><code>estat summarize</code></td>
<td>summary statistics for the estimation sample</td>
</tr>
<tr>
<td><code>estat vce</code></td>
<td>variance–covariance matrix of the estimators (VCE)</td>
</tr>
<tr>
<td><code>estat (svy)</code></td>
<td>postestimation statistics for survey data</td>
</tr>
<tr>
<td><code>estimates</code></td>
<td>cataloging estimation results</td>
</tr>
<tr>
<td><code>* hausman</code></td>
<td>Hausman’s specification test</td>
</tr>
<tr>
<td><code>lincom</code></td>
<td>point estimates, standard errors, testing, and inference for linear</td>
</tr>
<tr>
<td></td>
<td>combinations of coefficients</td>
</tr>
<tr>
<td><code>* lrtest</code></td>
<td>likelihood-ratio test</td>
</tr>
<tr>
<td><code>margins</code></td>
<td>marginal means, predictive margins, marginal effects, and average marginal</td>
</tr>
<tr>
<td></td>
<td>effects</td>
</tr>
<tr>
<td><code>marginsplot</code></td>
<td>graph the results from margins (profile plots, interaction plots, etc.)</td>
</tr>
<tr>
<td><code>nlcom</code></td>
<td>point estimates, standard errors, testing, and inference for nonlinear</td>
</tr>
<tr>
<td></td>
<td>combinations of coefficients</td>
</tr>
<tr>
<td><code>predict</code></td>
<td>predictions, residuals, influence statistics, and other diagnostic measures</td>
</tr>
<tr>
<td><code>predictnl</code></td>
<td>point estimates, standard errors, testing, and inference for generalized</td>
</tr>
<tr>
<td></td>
<td>predictions</td>
</tr>
<tr>
<td><code>pwcompare</code></td>
<td>pairwise comparisons of estimates</td>
</tr>
<tr>
<td><code>test</code></td>
<td>Wald tests of simple and composite linear hypotheses</td>
</tr>
<tr>
<td><code>testnl</code></td>
<td>Wald tests of nonlinear hypotheses</td>
</tr>
</tbody>
</table>

* `hausman` and `lrtest` are not appropriate with `svy` estimation results.
predict

Description for predict

`predict` creates a new variable containing predictions such as mean responses; linear predictions; density and distribution functions; standard errors; and Pearson, deviance, and Anscombe residuals.

Menu for predict

Statistics > Postestimation

Syntax for predict

Syntax for obtaining predictions of the outcome and other statistics

```
predict [type] newvarsspec [if] [in] [ , statistic options ]
```

Syntax for obtaining estimated random effects and their standard errors

```
predict [type] newvarsspec [if] [in], reffects [re_options]
```

Syntax for obtaining ML scores

```
predict [type] newvarsspec [if] [in], scores
```

`newvarsspec` is `stub*` or `newvarlist`.

<table>
<thead>
<tr>
<th>statistic</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mu</td>
<td>mean response; the default</td>
</tr>
<tr>
<td>eta</td>
<td>fitted linear predictor</td>
</tr>
<tr>
<td>xb</td>
<td>linear predictor for the fixed portion of the model only</td>
</tr>
<tr>
<td>stdp</td>
<td>standard error of the fixed-portion linear prediction</td>
</tr>
<tr>
<td>density</td>
<td>predicted density function</td>
</tr>
<tr>
<td>distribution</td>
<td>predicted distribution function</td>
</tr>
<tr>
<td>pearson</td>
<td>Pearson residuals</td>
</tr>
<tr>
<td>deviance</td>
<td>deviance residuals</td>
</tr>
<tr>
<td>anscombe</td>
<td>Anscombe residuals</td>
</tr>
</tbody>
</table>

These statistics are available both in and out of sample; type `predict ... if e(sample) ...` if wanted only for the estimation sample.
**options**

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Main</strong></td>
</tr>
<tr>
<td><strong>conditional(ctype)</strong></td>
</tr>
<tr>
<td><strong>marginal</strong></td>
</tr>
<tr>
<td><strong>nooffset</strong></td>
</tr>
<tr>
<td><strong>Integration</strong></td>
</tr>
<tr>
<td><strong>int_options</strong></td>
</tr>
</tbody>
</table>

**Integration**

pearson, deviance, anscombe may not be combined with marginal.

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ctype</strong></td>
</tr>
<tr>
<td><strong>ebmeans</strong></td>
</tr>
<tr>
<td><strong>ebmodes</strong></td>
</tr>
<tr>
<td><strong>fixedonly</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>re_options</strong></td>
</tr>
<tr>
<td><strong>Main</strong></td>
</tr>
<tr>
<td><strong>ebmeans</strong></td>
</tr>
<tr>
<td><strong>ebmodes</strong></td>
</tr>
<tr>
<td>*<em>reffects(stub</em></td>
</tr>
<tr>
<td><strong>Integration</strong></td>
</tr>
<tr>
<td><strong>int_options</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>int_options</strong></td>
</tr>
<tr>
<td><strong>intpoints(#)</strong></td>
</tr>
<tr>
<td><strong>iterate(#)</strong></td>
</tr>
<tr>
<td><strong>tolerance(#)</strong></td>
</tr>
</tbody>
</table>

**Options for predict**

- **mu**, the default, calculates the predicted mean, that is, the probability of a positive outcome.
- **eta, xb, stdp, density, distribution, pearson, deviance, anscombe, scores, conditional(), marginal, and nooffset**; see [ME] meglm postestimation.
- **reffects, ebmeans, ebmodes, and reses();** see [ME] meglm postestimation.

- **Integration**

  intpoints(), iterate(), and tolerance(); see [ME] meglm postestimation.
margins

Description for margins

margins estimates margins of response for mean responses and linear predictions.

Menu for margins

Statistics > Postestimation

Syntax for margins

```
margins [marginlist] [ , options ]
margins [marginlist] , predict(statistic ... ) [predict(statistic ... ) ... ] [options]
```

<table>
<thead>
<tr>
<th>statistic</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mu</td>
<td>mean response; the default</td>
</tr>
<tr>
<td>eta</td>
<td>fitted linear predictor</td>
</tr>
<tr>
<td>xb</td>
<td>linear predictor for the fixed portion of the model only</td>
</tr>
<tr>
<td>stdp</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>density</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>distribution</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>pearson</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>deviance</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>anscombe</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>reffects</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>scores</td>
<td>not allowed with margins</td>
</tr>
</tbody>
</table>

Options conditional(ebmeans) and conditional(ebmodes) are not allowed with margins. Option marginal is assumed where applicable if conditional(fixedonly) is not specified.

Statistics not allowed with margins are functions of stochastic quantities other than e(b). For the full syntax, see [R] margins.
estat

Description for estat

estat group reports the number of groups and minimum, average, and maximum group sizes for each level of the model. Model levels are identified by the corresponding group variable in the data. Because groups are treated as nested, the information in this summary may differ from what you would get if you used the tabulate command on each group variable individually.

Menu for estat

Statistics > Postestimation

Syntax for estat

estat group

Remarks and examples

Various predictions, statistics, and diagnostic measures are available after fitting a mixed-effects complementary log-log model with mecloglog. Here we show a short example of predicted probabilities and predicted random effects; refer to [ME] meglm postestimation for additional examples.

Example 1

In example 2 of [ME] mecloglog, we analyzed the cognitive ability (dtlm) of patients with schizophrenia compared with their relatives and control subjects, by using a three-level complementary log-log model with random effects at the family and subject levels. Cognitive ability was measured as the successful completion of the “Tower of London”, a computerized task, measured at three levels of difficulty.
.F. mecloglog postestimation — Postestimation tools for mecloglog

.use http://www.stata-press.com/data/r14/towerlondon
(Tower of London data)
.mecloglog dtlm difficulty i.group || family: || subject:
Fitting fixed-effects model:
(output omitted)

Mixed-effects cloglog regression

<table>
<thead>
<tr>
<th>Group Variable</th>
<th>No. of Groups</th>
<th>Observations per Group</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Minimum</td>
<td>Average</td>
</tr>
<tr>
<td>family</td>
<td>118</td>
<td>2</td>
</tr>
<tr>
<td>subject</td>
<td>226</td>
<td>2</td>
</tr>
</tbody>
</table>

Integration method: mvaghermite
Integration pts. = 7
Wald chi2(3) = 83.32
Log likelihood = -305.26516
Prob > chi2 = 0.0000

| dtlm | Coef.  | Std. Err. | z      | P>|z| | [95% Conf. Interval] |
|------|--------|-----------|--------|-----|----------------------|
| difficulty | -1.342844 | 0.1501508 | -8.94 | 0.000 | -1.637135 to -1.048554 |
| group | -0.1331007 | 0.269389 | -0.49 | 0.621 | -0.6610935 to 0.3948922 |
| 2     | -0.7714314 | 0.3097099 | -2.49 | 0.013 | -1.378452 to -0.164411 |
| 3     | -1.6718   | 0.2290325 | -7.30 | 0.000 | -2.120695 to -1.222905 |

family
| var(_cons) | 0.2353453 | 0.2924064 | 0.0206122 | 2.687117 |

family>
| var(_cons) | 0.7737687 | 0.4260653 | 0.2629714 | 2.276742 |

subject
| var(_cons) | 0.2353453 | 0.2924064 | 0.0206122 | 2.687117 |

LR test vs. cloglog model: chi2(2) = 16.61
Prob > chi2 = 0.0002
Note: LR test is conservative and provided only for reference.

We obtain predicted probabilities based on the contribution of both fixed effects and random effects by typing

.predict pr
(predictions based on fixed effects and posterior means of random effects)
(option mu assumed)
(using 7 quadrature points)

As the note says, the predicted values are based on the posterior means of random effects. You can use the modes option to obtain predictions based on the posterior modes of random effects.

We obtain predictions of the posterior means themselves by typing

.predict re*, reffects
(calculating posterior means of random effects)
(using 7 quadrature points)

Because we have one random effect at the family level and another random effect at the subject level, Stata saved the predicted posterior means in the variables re1 and re2, respectively. If you are not sure which prediction corresponds to which level, you can use the describe command to show the variable labels.
Here we list the data for family 16:

```
. list family subject dtlm pr re1 re2 if family==16, sepby(subject)
```

<table>
<thead>
<tr>
<th>family</th>
<th>subject</th>
<th>dtlm</th>
<th>pr</th>
<th>re1</th>
<th>re2</th>
</tr>
</thead>
<tbody>
<tr>
<td>208.</td>
<td>16</td>
<td>5</td>
<td>1</td>
<td>.486453</td>
<td>.4184933</td>
</tr>
<tr>
<td>209.</td>
<td>16</td>
<td>5</td>
<td>0</td>
<td>.1597047</td>
<td>.4184933</td>
</tr>
<tr>
<td>210.</td>
<td>16</td>
<td>5</td>
<td>0</td>
<td>.0444156</td>
<td>.4184933</td>
</tr>
<tr>
<td>211.</td>
<td>16</td>
<td>34</td>
<td>1</td>
<td>.9659582</td>
<td>.4184933</td>
</tr>
<tr>
<td>212.</td>
<td>16</td>
<td>34</td>
<td>1</td>
<td>.5862808</td>
<td>.4184933</td>
</tr>
<tr>
<td>213.</td>
<td>16</td>
<td>34</td>
<td>1</td>
<td>.205816</td>
<td>.4184933</td>
</tr>
<tr>
<td>214.</td>
<td>16</td>
<td>35</td>
<td>0</td>
<td>.5571261</td>
<td>.4184933</td>
</tr>
<tr>
<td>215.</td>
<td>16</td>
<td>35</td>
<td>0</td>
<td>.1915688</td>
<td>.4184933</td>
</tr>
<tr>
<td>216.</td>
<td>16</td>
<td>35</td>
<td>0</td>
<td>.0540124</td>
<td>.4184933</td>
</tr>
</tbody>
</table>

We can see that the predicted random effects (re1) at the family level are the same for all members of the family. Similarly, the predicted random effects (re2) at the individual level are constant within each individual. Based on a cutoff of 0.5, the predicted probabilities (pr) for this family do not match the observed outcomes (dtlm) as well as the predicted probabilities from the logistic example; see example 1 in [ME] melogit postestimation.

Methods and formulas

Methods and formulas for predicting random effects and other statistics are given in Methods and formulas of [ME] meglm postestimation.

Also see

[ME] mecloglog — Multilevel mixed-effects complementary log-log regression

[ME] meglm postestimation — Postestimation tools for meglm

[U] 20 Estimation and postestimation commands
meglm — Multilevel mixed-effects generalized linear model

**Description**

meglm fits multilevel mixed-effects generalized linear models. meglm allows a variety of distributions for the response conditional on normally distributed random effects.

**Quick start**

**Without weights**

Random-effects probit regression of y on x1 with random intercepts by lev2

```
meglm y x1 || lev2:, family(binomial) link(probit)
```

As above, but fit a logit model and report odds ratios

```
meglm y x1 || lev2:, family(binomial) or
```

Two-level gamma model of y with fixed and random coefficients on x1

```
meglm y x1 || lev2: x1, family(gamma)
```

Nested three-level random-intercept Poisson model reporting incidence-rate ratios

```
meglm y x1 || lev3: || lev2:, family(poisson) irr
```

Two-level linear regression of y on x1 and x2 with random intercepts by lev2, random coefficients on x2, and robust standard errors

```
meglm y x1 x2 || lev2: x2, vce(robust)
```

**With weights**

Two-level linear regression of y on x with random intercepts by psu for two-stage sampling with PSU-level and observation-level sampling weights wvar2 and wvar1, respectively

```
meglm y x [pweight=wvar1] || psu:, pweight(wvar2)
```

Add secondary sampling stage with units identified by ssu having weights wvar2 and PSU-level weights wvar3 for a three-level random-intercept model

```
meglm y x [pw=wvar1] || psu:, pw(wvar3) || ssu:, pw(wvar2)
```

Same as above, but svyset data first

```
svyset psu, weight(wvar3) || ssu, weight(wvar2) || _n, weight(wvar1)
svy: meglm y x || psu: || ssu:
```
Syntax

```
meglm depvar fe_equation [ || re_equation] [ || re_equation ...] [ , options]```

where the syntax of `fe_equation` is

```
[ indepvars] [ if] [ in] [ weight] [ , fe_options]
```

and the syntax of `re_equation` is one of the following:

for random coefficients and intercepts

```
levelvar: [ varlist] [ , re_options]
```

for random effects among the values of a factor variable

```
levelvar: R.varname
```

`levelvar` is a variable identifying the group structure for the random effects at that level or is `_all` representing one group comprising all observations.

---

**fe_options**

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>noconstant</code></td>
</tr>
<tr>
<td><code>exposure(varname_e)</code></td>
</tr>
<tr>
<td><code>offset(varname_o)</code></td>
</tr>
<tr>
<td><code>asis</code></td>
</tr>
</tbody>
</table>

---

**re_options**

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>covariance(vartype)</code></td>
</tr>
<tr>
<td><code>noconstant</code></td>
</tr>
<tr>
<td><code>fweight(varname)</code></td>
</tr>
<tr>
<td><code>iweight(varname)</code></td>
</tr>
<tr>
<td><code>pweight(varname)</code></td>
</tr>
<tr>
<td>options</td>
</tr>
<tr>
<td>------------------------------</td>
</tr>
<tr>
<td><strong>family</strong> (family)</td>
</tr>
<tr>
<td>link(link)</td>
</tr>
<tr>
<td>constraints(constraints)</td>
</tr>
<tr>
<td>collinear</td>
</tr>
<tr>
<td>vce(vcetype)</td>
</tr>
<tr>
<td>level(#)</td>
</tr>
<tr>
<td>eform</td>
</tr>
<tr>
<td>irr</td>
</tr>
<tr>
<td>or</td>
</tr>
<tr>
<td>nocnsreport</td>
</tr>
<tr>
<td>notable</td>
</tr>
<tr>
<td>noheader</td>
</tr>
<tr>
<td>nogroup</td>
</tr>
<tr>
<td>nolrtest</td>
</tr>
<tr>
<td>display_options</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>intmethod(intmethod)</td>
</tr>
<tr>
<td>intpoints(#)</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>maximize_options</td>
</tr>
<tr>
<td>startvalues(svmethod)</td>
</tr>
<tr>
<td>startgrid[ gridspec ]</td>
</tr>
<tr>
<td>noestimate</td>
</tr>
<tr>
<td>dnumerical</td>
</tr>
<tr>
<td>coeflegend</td>
</tr>
</tbody>
</table>
### vartype

<table>
<thead>
<tr>
<th>Description</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>independent</td>
<td>one unique variance parameter per random effect, all covariances 0; the default unless the R notation is used</td>
</tr>
<tr>
<td>exchangeable</td>
<td>equal variances for random effects, and one common pairwise covariance</td>
</tr>
<tr>
<td>identity</td>
<td>equal variances for random effects, all covariances 0; the default if the R notation is used</td>
</tr>
<tr>
<td>unstructured</td>
<td>all variances and covariances to be distinctly estimated</td>
</tr>
<tr>
<td>fixed(matname)</td>
<td>user-selected variances and covariances constrained to specified values; the remaining variances and covariances unrestricted</td>
</tr>
<tr>
<td>pattern(matname)</td>
<td>user-selected variances and covariances constrained to be equal; the remaining variances and covariances unrestricted</td>
</tr>
</tbody>
</table>

### family

<table>
<thead>
<tr>
<th>Description</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>gaussian</td>
<td>Gaussian (normal); the default</td>
</tr>
<tr>
<td>bernoulli</td>
<td>Bernoulli</td>
</tr>
<tr>
<td>binomial [#</td>
<td>varname]</td>
</tr>
<tr>
<td>gamma</td>
<td>gamma</td>
</tr>
<tr>
<td>nbinomial [mean</td>
<td>constant]</td>
</tr>
<tr>
<td>ordinal</td>
<td>ordinal</td>
</tr>
<tr>
<td>poisson</td>
<td>Poisson</td>
</tr>
</tbody>
</table>

### link

<table>
<thead>
<tr>
<th>Description</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>identity</td>
<td>identity</td>
</tr>
<tr>
<td>log</td>
<td>log</td>
</tr>
<tr>
<td>logit</td>
<td>logit</td>
</tr>
<tr>
<td>probit</td>
<td>probit</td>
</tr>
<tr>
<td>cloglog</td>
<td>complementary log-log</td>
</tr>
</tbody>
</table>

### intmethod

<table>
<thead>
<tr>
<th>Description</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>mvaghermite</td>
<td>mean–variance adaptive Gauss–Hermite quadrature; the default unless a crossed random-effects model is fit</td>
</tr>
<tr>
<td>mcaghermite</td>
<td>mode-curvature adaptive Gauss–Hermite quadrature</td>
</tr>
<tr>
<td>ghermite</td>
<td>nonadaptive Gauss–Hermite quadrature</td>
</tr>
<tr>
<td>laplace</td>
<td>Laplacian approximation; the default for crossed random-effects models</td>
</tr>
</tbody>
</table>

### Notes

- indepvars may contain factor variables; see [U] 11.4.3 Factor variables.
- depvar, indepvars, and varlist may contain time-series operators; see [U] 11.4.4 Time-series varlists.
- by and svy are allowed; see [U] 11.1.10 Prefix commands.
- vce() and weights are not allowed with the svy prefix; see [SVY] svy.
- fweights, iweights, and pweights are allowed; see [U] 11.1.6 weight. Only one type of weight may be specified.
- Weights are not supported under the Laplacian approximation or for crossed models.
- startvalues(), startgrid, noestimate, dnumerical, and coeflegend do not appear in the dialog box.
- See [U] 20 Estimation and postestimation commands for more capabilities of estimation commands.
Options

`noconstant` suppresses the constant (intercept) term and may be specified for the fixed-effects equation and for any of or all the random-effects equations.

`exposure(varname)` specifies a variable that reflects the amount of exposure over which the `depvar` events were observed for each observation; `ln(varname)` is included in the fixed-effects portion of the model with the coefficient constrained to be 1.

`offset(varname)` specifies that `varname` be included in the fixed-effects portion of the model with the coefficient constrained to be 1.

`asis` forces retention of perfect predictor variables and their associated, perfectly predicted observations and may produce instabilities in maximization; see [R] `probit`.

`covariance(vartype)` specifies the structure of the covariance matrix for the random effects and may be specified for each random-effects equation. `vartype` is one of the following: `independent`, `exchangeable`, `identity`, `unstructured`, `fixed(matname)`, or `pattern(matname)`.

- `covariance(independent)` covariance structure allows for a distinct variance for each random effect within a random-effects equation and assumes that all covariances are 0. The default is `covariance(independent)` unless a crossed random-effects model is fit, in which case the default is `covariance(identity)`.
- `covariance(exchangeable)` structure specifies one common variance for all random effects and one common pairwise covariance.
- `covariance(identity)` is short for “multiple of the identity”; that is, all variances are equal and all covariances are 0.
- `covariance(unstructured)` allows for all variances and covariances to be distinct. If an equation consists of `p` random-effects terms, the unstructured covariance matrix will have `p(p + 1)/2` unique parameters.

`covariance(fixed(matname))` and `covariance(pattern(matname))` covariance structures provide a convenient way to impose constraints on variances and covariances of random effects. Each specification requires a `matname` that defines the restrictions placed on variances and covariances. Only elements in the lower triangle of `matname` are used, and row and column names of `matname` are ignored. A missing value in `matname` means that a given element is unrestricted. In a `fixed(matname)` covariance structure, (co)variance `(i, j)` is constrained to equal the value specified in the `i, j`th entry of `matname`. In a `pattern(matname)` covariance structure, (co)variances `(i, j)` and `(k, l)` are constrained to be equal if `matname[i, j] = matname[k, l]`.

`fweight(varname)` specifies frequency weights at higher levels in a multilevel model, whereas frequency weights at the first level (the observation level) are specified in the usual manner, for example, `[fw=fwtvar1]`. `varname` can be any valid Stata variable name, and you can specify `fweight()` at levels two and higher of a multilevel model. For example, in the two-level model

```
  . mecmd fixed_portion [fw = wt1] || school: ... , fweight(wt2) ...
```

the variable `wt1` would hold the first-level (the observation-level) frequency weights, and `wt2` would hold the second-level (the school-level) frequency weights.

`iweight(varname)` specifies importance weights at higher levels in a multilevel model, whereas importance weights at the first level (the observation level) are specified in the usual manner, for example, `[iw=iwtvar1]`. `varname` can be any valid Stata variable name, and you can specify `iweight()` at levels two and higher of a multilevel model. For example, in the two-level model

```
  . mecmd fixed_portion [iw = wt1] || school: ... , iweight(wt2) ...
```
the variable \texttt{wt1} would hold the first-level (the observation-level) importance weights, and \texttt{wt2} would hold the second-level (the school-level) importance weights.

\texttt{pweight(varname)} specifies sampling weights at higher levels in a multilevel model, whereas sampling weights at the first level (the observation level) are specified in the usual manner, for example, \texttt{[pw=pwtvar1]}. \texttt{varname} can be any valid Stata variable name, and you can specify \texttt{pweight()} at levels two and higher of a multilevel model. For example, in the two-level model

\begin{verbatim}
  . mecmd fixed_portion [pw = wt1] || school: ... , pweight(wt2) ...
\end{verbatim}

variable \texttt{wt1} would hold the first-level (the observation-level) sampling weights, and \texttt{wt2} would hold the second-level (the school-level) sampling weights.

\texttt{family(family)} specifies the distribution of \texttt{depvar}; \texttt{family(gaussian)} is the default.

\texttt{link(link)} specifies the link function; the default is the canonical link for the \texttt{family()} specified except for the gamma and negative binomial families.

If you specify both \texttt{family()} and \texttt{link()}, not all combinations make sense. You may choose from the following combinations:

\begin{verbatim}

<table>
<thead>
<tr>
<th>family</th>
<th>link</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gaussian</td>
<td>identity</td>
</tr>
<tr>
<td>Bernoulli</td>
<td>D</td>
</tr>
<tr>
<td>binomial</td>
<td>D</td>
</tr>
<tr>
<td>gamma</td>
<td>D</td>
</tr>
<tr>
<td>negative binomial</td>
<td>D</td>
</tr>
<tr>
<td>ordinal</td>
<td>D</td>
</tr>
<tr>
<td>Poisson</td>
<td>D</td>
</tr>
</tbody>
</table>

D denotes the default.
\end{verbatim}

\texttt{constraints(constraints)}, \texttt{collinear}; see \cite{R} estimation options.

\texttt{vce(vcetype)} specifies the type of standard error reported, which includes types that are derived from asymptotic theory (\texttt{oim}), that are robust to some kinds of misspecification (\texttt{robust}), and that allow for intragroup correlation (\texttt{cluster clustvar}); see \cite{R} \texttt{vce_option}. If \texttt{vce(robust)} is specified, robust variances are clustered at the highest level in the multilevel model.

\texttt{level(#)}; see \cite{R} estimation options.

\texttt{eform} reports exponentiated fixed-effects coefficients and corresponding standard errors and confidence intervals. This option may be specified either at estimation or upon replay.

\texttt{irr} reports estimated fixed-effects coefficients transformed to incidence-rate ratios, that is, \(\exp(\beta)\) rather than \(\beta\). Standard errors and confidence intervals are similarly transformed. This option affects how results are displayed, not how they are estimated or stored. \texttt{irr} may be specified either at estimation or upon replay. This option is allowed for count models only.

\texttt{or} reports estimated fixed-effects coefficients transformed to odds ratios, that is, \(\exp(\beta)\) rather than \(\beta\). Standard errors and confidence intervals are similarly transformed. This option affects how results are displayed, not how they are estimated. \texttt{or} may be specified at estimation or upon replay. This option is allowed for logistic models only.
nocnsreport; see [R] estimation options.
notable suppresses the estimation table, either at estimation or upon replay.
noheader suppresses the output header, either at estimation or upon replay.
nogroup suppresses the display of group summary information (number of groups, average group size, minimum, and maximum) from the output header.
nolrtest prevents meglm from fitting a reference linear regression model and using this model to calculate a likelihood-ratio test comparing the mixed model with ordinary regression. This option may also be specified upon replay to suppress this test from the output.

display_options: noci, nopvalues, noomitted, vsquish, noemptycells, baselevels, allbaselevels, nolvlabel, fvwrap(#), fvwrapon(style), cformat(,%fmt), pformat(,%fmt), sformat(,%fmt), and nolstretch; see [R] estimation options.

Integration

intmethod(intmethod) specifies the integration method to be used for the random-effects model. mvaghermite performs mean–variance adaptive Gauss–Hermite quadrature; mcaghermite performs mode-curvature adaptive Gauss–Hermite quadrature; ghermite performs nonadaptive Gauss–Hermite quadrature; and laplace performs the Laplacian approximation, equivalent to mode-curvature adaptive Gaussian quadrature with one integration point.

The default integration method is mvaghermite unless a crossed random-effects model is fit, in which case the default integration method is laplace. The Laplacian approximation has been known to produce biased parameter estimates; however, the bias tends to be more prominent in the estimates of the variance components rather than in the estimates of the fixed effects.

For crossed random-effects models, estimation with more than one quadrature point may be prohibitively intensive even for a small number of levels. For this reason, the integration method defaults to the Laplacian approximation. You may override this behavior by specifying a different integration method.

intpoints(#) sets the number of integration points for quadrature. The default is intpoints(7), which means that seven quadrature points are used for each level of random effects. This option is not allowed with intmethod(laplace).

The more integration points, the more accurate the approximation to the log likelihood. However, computation time increases as a function of the number of quadrature points raised to a power equaling the dimension of the random-effects specification. In crossed random-effects models and in models with many levels or many random coefficients, this increase can be substantial.

Maximization

maximize_options: difficult, technique(algorithm_spec), iterate(#), [no]log, trace, gradient, showstep, hessian, showtolerance, tolerance(#), ltolerance(#), nrtolerance(#), nonrtolerance, and from(init_specs); see [R] maximize. Those that require special mention for meglm are listed below.

from() accepts a properly labeled vector of initial values or a list of coefficient names with values. A list of values is not allowed.
The following options are available with `meglm` but are not shown in the dialog box:

`startvalues(svmethod)` specifies how starting values are to be computed. Starting values specified in `from()` override the computed starting values.

`startvalues(zero)` specifies that starting values be set to 0.

`startvalues(constantonly)` builds on `startvalues(zero)` by fitting a constant-only model to obtain estimates of the intercept and auxiliary parameters, and it substitutes 1 for the variances of random effects.

`startvalues(fixedonly)` builds on `startvalues(constantonly)` by fitting a full fixed-effects model to obtain estimates of coefficients along with intercept and auxiliary parameters, and it continues to use 1 for the variances of random effects. This is the default behavior.

`startvalues(iv)` builds on `startvalues(fixedonly)` by using instrumental-variable methods with generalized residuals to obtain variances of random effects.

`startgrid[(gridspec)]` performs a grid search on variance components of random effects to improve starting values. No grid search is performed by default unless the starting values are found to be not feasible, in which case `meglm` runs `startgrid()` to perform a “minimal” search involving $q^3$ likelihood evaluations, where $q$ is the number of random effects. Sometimes this resolves the problem. Usually, however, there is no problem and `startgrid()` is not run by default. There can be benefits from running `startgrid()` to get better starting values even when starting values are feasible.

`startgrid()` is a brute-force approach that tries various values for variances and covariances and chooses the ones that work best. You may already be using a default form of `startgrid()` without knowing it. If you see `meglm` displaying Grid node 1, Grid node 2, ... following Grid node 0 in the iteration log, that is `meglm` doing a default search because the original starting values were not feasible. The default form tries 0.1, 1, and 10 for all variances of all random effects.

`startgrid(numlist)` specifies values to try for variances of random effects.

`startgrid(covspec)` specifies the particular variances and covariances in which grid searches are to be performed. `covspec` is `name[level]` for variances and `name1[level]*name2[level]` for covariances. For example, the variance of the random intercept at level id is specified as `_cons[id]`, and the variance of the random slope on variable `week` at the same level is specified as `week[id]`. The residual variance for the linear mixed-effects model is specified as `e.depvar`, where `depvar` is the name of the dependent variable. The covariance between the random slope and the random intercept above is specified as `_cons[id]*week[id]`.

`startgrid(numlist covspec)` combines the two syntaxes. You may also specify `startgrid()` multiple times so that you can search the different ranges for different variances and covariances.

`noestimate` specifies that the model is not to be fit. Instead, starting values are to be shown (as modified by the above options if modifications were made), and they are to be shown using the `coeflegend` style of output.

`dnumerical` specifies that during optimization, the gradient vector and Hessian matrix be computed using numerical techniques instead of analytical formulas. By default, analytical formulas for computing the gradient and Hessian are used for all integration methods except `intmethod(laplace)`. `coeflegend`; see [R] estimation options.
Remarks and examples

For a general introduction to \texttt{me} commands, see [ME] \texttt{me}. For additional examples of mixed-effects models for binary and binomial outcomes, see [ME] \texttt{melogit}, [ME] \texttt{meprobit}, and [ME] \texttt{mecloglog}. For additional examples of mixed-effects models for ordinal responses, see [ME] \texttt{meologit} and [ME] \texttt{meoprobit}. For additional examples of mixed-effects models for multinomial outcomes, see [SEM] example 41g. For additional examples of mixed-effects models for count outcomes, see [ME] \texttt{mepoisson} and [ME] \texttt{menbreg}.

Remarks are presented under the following headings:

- Introduction
- Two-level models for continuous responses
- Two-level models for nonlinear responses
- Three-level models for nonlinear responses
- Crossed-effects models
- Obtaining better starting values
- Survey data
- Video example

Introduction

\texttt{meglm} fits multilevel mixed-effects generalized linear models of the form

\[ g\{ E(y|X,u) \} = X\beta + Zu, \quad y \sim F \]  \hspace{1cm} (1)

where \( y \) is the \( n \times 1 \) vector of responses from the distributional family \( F \), \( X \) is an \( n \times p \) design/covariate matrix for the fixed effects \( \beta \), and \( Z \) is the \( n \times q \) design/covariate matrix for the random effects \( u \). The \( X\beta + Zu \) part is called the linear predictor, and it is often denoted as \( \eta \). The linear predictor also contains the offset or exposure variable when \texttt{offset()} or \texttt{exposure()} is specified. \( g(\cdot) \) is called the link function and is assumed to be invertible such that

\[ E(y|X,u) = g^{-1}(X\beta + Zu) = H(\eta) = \mu \]

For notational convenience here and throughout this manual entry, we suppress the dependence of \( y \) on \( X \). Substituting various definitions for \( g(\cdot) \) and \( F \) results in a wide array of models. For instance, if \( y \) is distributed as Gaussian (normal) and \( g(\cdot) \) is the identity function, we have

\[ E(y) = X\beta + Zu, \quad y \sim \text{normal} \]

or mixed-effects linear regression. If \( g(\cdot) \) is the logit function and \( y \) is distributed as Bernoulli, we have

\[ \logit\{ E(y) \} = X\beta + Zu, \quad y \sim \text{Bernoulli} \]

or mixed-effects logistic regression. If \( g(\cdot) \) is the natural log function and \( y \) is distributed as Poisson, we have

\[ \ln\{ E(y) \} = X\beta + Zu, \quad y \sim \text{Poisson} \]

or mixed-effects Poisson regression. In fact, some combinations of families and links are so common that we implemented them as separate commands in terms of \texttt{meglm}. 
When no family–link combination is specified, `meglm` defaults to a Gaussian family with an identity link. Thus `meglm` can be used to fit linear mixed-effects models; however, for those models we recommend using the more specialized `mixed`, which, in addition to `meglm` capabilities, allows for modeling of the structure of the residual errors; see \[ME\] mixed for details.

The random effects $u$ are assumed to be distributed as multivariate normal with mean $0$ and $q \times q$ variance matrix $\Sigma$. The random effects are not directly estimated (although they may be predicted), but instead are characterized by the variance components, the elements of $G = \text{Var}(u)$.

The general forms of the design matrices $X$ and $Z$ allow estimation for a broad class of generalized mixed-effects models: blocked designs, split-plot designs, growth curves, multilevel or hierarchical designs, etc. They also allow a flexible method of modeling within-cluster correlation. Subjects within the same cluster can be correlated as a result of a shared random intercept, or through a shared random slope on a covariate, or both. The general specification of variance components also provides additional flexibility—the random intercept and random slope could themselves be modeled as independent, or correlated, or independent with equal variances, and so forth.

Comprehensive treatments of mixed models are provided by, for example, Searle, Casella, and McCulloch (1992); Verbeke and Molenberghs (2000); Raudenbush and Bryk (2002); Demidenko (2004); Hedeker and Gibbons (2006); McCulloch, Searle, and Neuhaus (2008); and Rabe-Hesketh and Skrondal (2012).

The key to fitting mixed models lies in estimating the variance components, and for that there exist many methods; see, for example, Breslow and Clayton (1993); Lin and Breslow (1996); Bates and Pinheiro (1998); and Ng et al. (2006). `meglm` uses maximum likelihood (ML) to estimate model parameters. The ML estimates are based on the usual application of likelihood theory, given the distributional assumptions of the model.

Returning to (1): in clustered-data situations, it is convenient not to consider all $n$ observations at once but instead to organize the mixed model as a series of $M$ independent groups (or clusters)

$$g\{E(y_j)\} = X_j\beta + Z_j u_j$$

for $j = 1, \ldots, M$, with cluster $j$ consisting of $n_j$ observations. The response $y_j$ comprises the rows of $y$ corresponding with the $j$th cluster, with $X_j$ defined analogously. The random effects $u_j$ can now be thought of as $M$ realizations of a $q \times 1$ vector that is normally distributed with mean $0$ and $q \times q$ variance matrix $\Sigma$. The matrix $Z_j$ is the $n_j \times q$ design matrix for the $j$th cluster random effects. Relating this to (1), note that

<table>
<thead>
<tr>
<th>Command</th>
<th><code>meglm</code> equivalent</th>
</tr>
</thead>
<tbody>
<tr>
<td>melogit</td>
<td><code>family(bernoulli) link(logit)</code></td>
</tr>
<tr>
<td>meprobit</td>
<td><code>family(bernoulli) link(probit)</code></td>
</tr>
<tr>
<td>mecloglog</td>
<td><code>family(bernoulli) link(cloglog)</code></td>
</tr>
<tr>
<td>meologit</td>
<td><code>family(ordinal) link(logit)</code></td>
</tr>
<tr>
<td>meoprobit</td>
<td><code>family(ordinal) link(probit)</code></td>
</tr>
<tr>
<td>mepoisson</td>
<td><code>family(poisson) link(log)</code></td>
</tr>
<tr>
<td>menbreg</td>
<td><code>family(nbinomial) link(log)</code></td>
</tr>
</tbody>
</table>
\[
Z = \begin{bmatrix}
Z_1 & 0 & \cdots & 0 \\
0 & Z_2 & \cdots & 0 \\
\vdots & \vdots & \ddots & \vdots \\
0 & 0 & 0 & Z_M
\end{bmatrix}; \quad u = \begin{bmatrix}
u_1 \\
\vdots \\
u_M
\end{bmatrix}; \quad G = I_M \otimes \Sigma
\]

where \( I_M \) is the \( M \times M \) identity matrix and \( \otimes \) is the Kronecker product.

The mixed-model formula (2) is from Laird and Ware (1982) and offers two key advantages. First, it makes specifications of random-effects terms easier. If the clusters are schools, you can simply specify a random effect at the school level, as opposed to thinking of what a school-level random effect would mean when all the data are considered as a whole (if it helps, think Kronecker products). Second, representing a mixed-model with (2) generalizes easily to more than one set of random effects. For example, if classes are nested within schools, then (2) can be generalized to allow random effects at both the school and the class-within-school levels.

### Two-level models for continuous responses

We begin with a simple application of (2).

**Example 1**

Consider a longitudinal dataset, used by both Ruppert, Wand, and Carroll (2003) and Diggle et al. (2002), consisting of weight measurements of 48 pigs on 9 successive weeks. Pigs are identified by the variable `id`. Each pig experiences a linear trend in growth but overall weight measurements vary from pig to pig. Because we are not really interested in these particular 48 pigs per se, we instead treat them as a random sample from a larger population and model the between-pig variability as a random effect, or in the terminology of (2), as a random-intercept term at the pig level. We thus wish to fit the model

\[
weight_{ij} = \beta_0 + \beta_1 week_{ij} + u_j + \epsilon_{ij}
\]

for \( i = 1, \ldots, 9 \) weeks and \( j = 1, \ldots, 48 \) pigs. The fixed portion of the model, \( \beta_0 + \beta_1 week_{ij} \), simply states that we want one overall regression line representing the population average. The random effect \( u_j \) serves to shift this regression line up or down according to each pig. Because the random effects occur at the pig level (`id`), we fit the model by typing

```
. use http://www.stata-press.com/data/r14/pig
(Longitudinal analysis of pig weights)
. meglm weight week || id:
```

Fitting fixed-effects model:
Iteration 0:  log likelihood = -1251.2506
Iteration 1:  log likelihood = -1251.2506
Refining starting values:
Grid node 0:  log likelihood = -1150.6253
Fitting full model:
Iteration 0:  log likelihood = -1150.6253  (not concave)
Iteration 1:  log likelihood = -1036.1793
Iteration 2:  log likelihood = -1017.9120
Iteration 3:  log likelihood = -1014.9537
Iteration 4:  log likelihood = -1014.9268
Iteration 5:  log likelihood = -1014.9268
Mixed-effects GLM

Number of obs = 432
Family: Gaussian
Link: identity
Group variable: id Number of groups = 48

Obs per group:
min = 9
avg = 9.0
max = 9

Integration method: mvaghermite Integration pts. = 7

Wald chi2(1) = 25337.48
Log likelihood = -1014.9268 Prob > chi2 = 0.0000

weight

| Coef.     | Std. Err. | z  | P>|z| | [95% Conf. Interval] |
|-----------|-----------|----|------|----------------------|
| week      | 6.209896  | .0390124 | 159.18 | 0.000 | 6.133433  | 6.286359 |
| _cons     | 19.35561  | .5974047 | 32.40  | 0.000 | 18.18472  | 20.52651 |

id

| var(_cons) | 14.81745  | 3.124202 | 9.801687 | 0.000 | 22.39989  |
| var(e.weight) | 4.383264  | .3163349 | 3.805112 | 0.000 | 5.049261  |

LR test vs. linear model: chibar2(01) = 472.65 Prob >= chibar2 = 0.0000

At this point, a guided tour of the model specification and output is in order:

1. By typing weight week, we specified the response, weight, and the fixed portion of the model in the same way that we would if we were using regress or any other estimation command. Our fixed effects are a coefficient on week and a constant term.

2. When we added || id:, we specified random effects at the level identified by the group variable id, that is, the pig level (level two). Because we wanted only a random intercept, that is all we had to type.

3. The estimation log displays a set of iterations from optimizing the log likelihood. By default, these are Newton–Raphson iterations, but other methods are available by specifying the appropriate maximize_options; see [R] maximize.

4. The header describes the model, presents a summary of the random-effects group, reports the integration method used to fit the model, and reports a Wald test against the null hypothesis that all the coefficients on the independent variables in the mean equation are 0. Here the null hypothesis is rejected at all conventional levels. You can suppress the group information with the nogroup or the noheader option, which will suppress the rest of the header as well.

5. The estimation table reports the fixed effects, followed by the random effects, followed by the overall error term.
   a. For the fixed-effects part, we estimate $\beta_0 = 19.36$ and $\beta_1 = 6.21$.
   b. The random-effects equation is labeled id, meaning that these are random effects at the id (pig) level. We have only one random effect at this level, the random intercept. The variance of the level-two errors, $\sigma_u^2$, is estimated as 14.82 with standard error 3.12.
   c. The row labeled var(e.weight) displays the estimated variance of the overall error term: $\hat{\sigma}_\epsilon^2 = 4.38$. This is the variance of the level-one errors, that is, the residuals.

6. Finally, a likelihood-ratio test comparing the model with ordinary linear regression is provided and is highly significant for these data. See Distribution theory for likelihood-ratio test in [ME] me for a discussion of likelihood-ratio testing of variance components.
Two-level models for nonlinear responses

By specifying different family–link combinations, we can fit a variety of mixed-effects models for nonlinear responses. Here we replicate the model from example 2 of `meqrlogit`.

Example 2

Ng et al. (2006) analyzed a subsample of data from the 1989 Bangladesh fertility survey (Huq and Cleland 1990), which polled 1,934 Bangladeshi women on their use of contraception. The women sampled were from 60 districts, identified by the variable `district`. Each district contained either urban or rural areas (variable `urban`) or both. The variable `c_use` is the binary response, with a value of 1 indicating contraceptive use. Other covariates include mean-centered `age` and three indicator variables recording number of children.

We fit a standard logistic regression model, amended to have a random intercept for each district and a random slope on the indicator variable `urban`. We fit the model by typing
. use http://www.stata-press.com/data/r14/bangladesh
(Bangladesh Fertility Survey, 1989)
. meglm c_use urban age child* || district: urban, family(bernoulli) link(logit)

Fitting fixed-effects model:
Iteration 0: log likelihood = -1229.5485
Iteration 1: log likelihood = -1228.5268
Iteration 2: log likelihood = -1228.5263
Iteration 3: log likelihood = -1228.5263

Refining starting values:
Grid node 0: log likelihood = -1215.8592

Fitting full model:
Iteration 0: log likelihood = -1215.8592 (not concave)
Iteration 1: log likelihood = -1209.6285
Iteration 2: log likelihood = -1205.7903
Iteration 3: log likelihood = -1205.1337
Iteration 4: log likelihood = -1205.0034
Iteration 5: log likelihood = -1205.0025
Iteration 6: log likelihood = -1205.0025

Mixed-effects GLM
Number of obs = 1,934
Family: Bernoulli
Link: logit
Group variable: district Number of groups = 60
Obs per group:
  min = 2
  avg = 32.2
  max = 118
Integration method: mvaghermite Integration pts. = 7
Wald chi2(5) = 97.30
Log likelihood = -1205.0025 Prob > chi2 = 0.0000

| Coef. | Std. Err. | z     | P>|z|   | [95% Conf. Interval] |
|-------|-----------|-------|-------|-----------------------|
| c_use |           |       |       |                       |
| urban | .7143927  | .1513595 | 4.72  | 0.000     | .4177335 -1.011052 |
| age   | -.0262261 | .0079656 | -3.29 | 0.001     | -.0418384 -.0106138 |
| child1| 1.128973  | .1599347 | 7.06  | 0.000     | .815507  1.442439  |
| child2| 1.363165  | .1761804 | 7.74  | 0.000     | 1.017857 1.708472  |
| child3| 1.352238  | .1815608 | 7.45  | 0.000     | .9963853 1.708091  |
| _cons | -1.698137 | .1505019 | -11.28| 0.000     | -1.993115 -1.403159 |

| district |          |       |       |                       |
| var(urban)| .2741013 | .2131526 | .059701 | 1.258463  |
| var(_cons)| .2390807 | .0857012 | .118419 | .4826891  |

LR test vs. logistic model: chi2(2) = 47.05 Prob > chi2 = 0.0000

Note: LR test is conservative and provided only for reference.

Because we did not specify a covariance structure for the random effects \((u_{1j}, u_{0j})\), meglm used the default independent structure:

\[
\Sigma = \text{Var} \begin{bmatrix} u_{1j} \\ u_{0j} \end{bmatrix} = \begin{bmatrix} \sigma_{u_{1}}^2 & 0 \\ 0 & \sigma_{u_{0}}^2 \end{bmatrix}
\]

with \(\hat{\sigma}_{u_{1}}^2 = 0.27\) and \(\hat{\sigma}_{u_{0}}^2 = 0.24\). You can request a different covariance structure by specifying the covariance() option. See Two-level models in [ME] meqrlogit for further analysis of these data, and see [ME] me and [ME] mixed for further examples of covariance structures.
Three-level models for nonlinear responses

Two-level models extend naturally to models with three or more levels with nested random effects. Here we replicate the model from example 2 of [ME] meologit.

Example 3

We use the data from the Television, School, and Family Smoking Prevention and Cessation Project (Flay et al. 1988; Rabe-Hesketh and Skrondal 2012, chap. 11), where schools were randomly assigned into one of four groups defined by two treatment variables. Students within each school are nested in classes, and classes are nested in schools. The dependent variable is the tobacco and health knowledge (THK) scale score collapsed into four ordered categories. We regress the outcome on the treatment variables, social resistance classroom curriculum and TV intervention, and their interaction and control for the pretreatment score.

```
use http://www.stata-press.com/data/r14/tvsfpors
meglm thk prethk cc##tv || school: || class:, family(ordinal) link(logit)
```

```
Fitting fixed-effects model:
Iteration 0: log likelihood = -2212.775
Iteration 1: log likelihood = -2125.509
Iteration 2: log likelihood = -2125.1034
Iteration 3: log likelihood = -2125.1032
Refining starting values:
Grid node 0: log likelihood = -2152.1514
Fitting full model:
Iteration 0: log likelihood = -2152.1514 (not concave)
Iteration 1: log likelihood = -2125.9213 (not concave)
Iteration 2: log likelihood = -2120.1861
Iteration 3: log likelihood = -2115.6177
Iteration 4: log likelihood = -2114.5896
Iteration 5: log likelihood = -2114.5881
Iteration 6: log likelihood = -2114.5881
Mixed-effects GLM Family:               Number of obs = 1,600
                     ordinal                     Link:            logit
                  Family:               Link:            logit
  Group Variable |   No.   Observations per Group
                 | of  Minimum Average Maximum
    school      | 28    18        57.1      137
     class      | 135   1         11.9      28
```
Integration method: mvaghermite  
Integration pts. = 7
Wald chi2(4) = 124.39
Log likelihood = -2114.5881  
Prob > chi2 = 0.0000

| thk  | Coef.     | Std. Err. | z     | P>|z| | [95% Conf. Interval] |
|------|-----------|-----------|-------|------|-----------------------|
| prethk  | .4085273  | .039616   | 10.31 | 0.000 | .3308814 .4861731 |
| 1.cc | .8844369  | .2099124  | 4.21  | 0.000 | .4730161 1.295858 |
| 1.tv  | .236448   | .2049065  | 1.15  | 0.249 | -.1651614 .6380575 |
| cc#tv 1 1 | -.3717699 | .2958887 | -1.26 | 0.209 | -.951701 .2081612 |
| /cut1 | -.0959459 | .1688988  | -0.57 | 0.570 | -.4269815 .2350896 |
| /cut2 | 1.177478  | .1704946  | 6.91  | 0.000 | .8433151 1.511642 |
| /cut3 | 2.383672  | .1786736  | 13.34 | 0.000 | 2.033478 2.733865 |

school
| var(_cons) | .0448735 | .0425387 | .0069997 | .2876749 |

school>class
| var(_cons) | .1482157 | .0637521 | .063792 | .3443674 |

LR test vs. ologit model: chi2(2) = 21.03  
Prob > chi2 = 0.0000

Note: LR test is conservative and provided only for reference.

Notes:
1. Our model now has two random-effects equations, separated by ||. The first is a random intercept (constant only) at the school level (level three), and the second is a random intercept at the class level (level two). The order in which these are specified (from left to right) is significant—meglm assumes that class is nested within school.

2. The information on groups is now displayed as a table, with one row for each grouping. You can suppress this table with the nogroup or noheader option, which will suppress the rest of the header, as well.

3. The variance-component estimates are now organized and labeled according to level. The variance component for class is labeled school>class to emphasize that classes are nested within schools.

We refer you to example 2 of [ME] meologit and example 1 of [ME] meologit postestimation for a substantive interpretation of the results.

The above extends to models with more than two levels of nesting in the obvious manner, by adding more random-effects equations, each separated by ||. The order of nesting goes from left to right as the groups go from biggest (highest level) to smallest (lowest level).

### Crossed-effects models

Not all mixed models contain nested levels of random effects. In this section, we consider a crossed-effects model, that is, a mixed-effects model in which the levels of random effects are not nested; see [ME] me for more information on crossed-effects models.
Example 4

We use the salamander cross-breeding data from Karim and Zeger (1992) as analyzed in Rabe-Hesketh and Skrondal (2012, chap. 16.10). The salamanders come from two populations—whiteside and roughbutt—and are labeled whiteside males (wsm), whiteside females (wsf), roughbutt males (rbm), and roughbutt females (rbf). Male identifiers are recorded in the variable male, and female identifiers are recorded in the variable female. The salamanders were divided into groups such that each group contained 60 male–female pairs, with each salamander having three potential partners from the same population and three potential partners from the other population. The outcome (y) is coded 1 if there was a successful mating and is coded 0 otherwise; see the references for a detailed description of the mating experiment.

We fit a crossed-effects logistic regression for successful mating, where each male has the same value of his random intercept across all females, and each female has the same value of her random intercept across all males.

To fit a crossed-effects model in Stata, we use the _all: R.varname syntax. We treat the entire dataset as one super cluster, denoted _all, and we nest each gender within the super cluster by using the R.varname notation. R.male requests a random intercept for each level of male and imposes an identity covariance structure on the random effects; that is, the variances of the random intercepts are restricted to be equal for all male salamanders. R.female accomplishes the same for the female salamanders. In Stata, we type
. use http://www.stata-press.com/data/r14/salamander
. meglm y wsm##wsf || _all: R.male || _all: R.female, family(bernoulli)
> link(logit) or
note: crossed random-effects model specified; option intmethod(laplace) implied

Fitting fixed-effects model:
Iteration 0:  log likelihood = -223.13998
Iteration 1:  log likelihood = -222.78752
Iteration 2:  log likelihood = -222.78735
Iteration 3:  log likelihood = -222.78735

Refining starting values:
Grid node 0:  log likelihood = -211.58149

Fitting full model:
Iteration 0:  log likelihood = -211.58149
Iteration 1:  log likelihood = -209.32315
Iteration 2:  log likelihood = -209.28178
Iteration 3:  log likelihood = -209.27699
Iteration 4:  log likelihood = -209.27659
Iteration 5:  log likelihood = -209.27659

Mixed-effects GLM
Number of obs = 360
Family: Bernoulli
Link: logit
Group variable: _all  Number of groups = 1
Obs per group:  min = 360
               avg = 360.0
               max = 360
Integration method: laplace
Wald chi2(3) = 37.44
Log likelihood = -209.27659  Prob > chi2 = 0.0000

| y       | Odds Ratio | Std. Err. |     z  |   P>|z|  |   [95% Conf. Interval] |
|--------|------------|-----------|-------|-------|-----------------------|
| 1.wsm  | 0.4955495  | 0.2231633 | -1.56 | 0.119 | 0.2050035  1.197879  |
| 1.wsf  | 0.0547913  | 0.0308325 | -5.16 | 0.000 | 0.0181852  0.1650842 |
| wsm#wsf| 36.17644   | 23.08862  |  5.62 | 0.000 | 10.35547   126.381   |
| _cons  | 2.740895   | 1.06429   |  2.60 | 0.009 | 1.280474   5.866972  |

Similar to above but for _all>male

LR test vs. logistic model:  chi2(2) = 27.02  Prob > chi2 = 0.0000
Note: LR test is conservative and provided only for reference.

Because we specified a crossed-effects model, `meglm` defaulted to the method of Laplacian approximation to calculate the likelihood; see *Computation time and the Laplacian approximation* in [ME] me for a discussion of computational complexity of mixed-effects models, and see *Methods and formulas* below for the formulas used by the Laplacian approximation method.

The estimates of the random intercepts suggest that the heterogeneity among the female salamanders, 1.17, is larger than the heterogeneity among the male salamanders, 1.04.
Setting both random intercepts to 0, the odds of successful mating for a roughbutt male–female pair are given by the estimate of \(_\text{cons}\), 2.74. Rabe-Hesketh and Skrondal (2012, chap. 16.10) show how to calculate the odds ratios for the other three salamander pairings.

The \texttt{R.varname} notation is equivalent to giving a list of overparameterized (none dropped) indicator variables for use in a random-effects specification. When you specify \texttt{R.varname}, \texttt{meglm} handles the calculations internally rather than creating the indicators in the data. Because the set of indicators is overparameterized, \texttt{R.varname} implies \texttt{noconstant}. You can include factor variables in the fixed-effects specification by using standard methods; see [U] 11.4.3 Factor variables. However, random-effects equations support only the \texttt{R.varname factor} specification. For more complex factor specifications (such as interactions) in random-effects equations, use \texttt{generate} to form the variables manually.

\section*{Technical note}

We fit the salamander model by using

\begin{verbatim}
.meglm y wsm##wsf || _all: R.male || _all: R.female ...
\end{verbatim}

as a direct way to demonstrate the \texttt{R.} notation. However, we can technically treat female salamanders as nested within the \texttt{_all} group, yielding the equivalent way to fit the model:

\begin{verbatim}
.meglm y wsm##wsf || _all: R.male || female: ...
\end{verbatim}

We leave it to you to verify that both produce identical results. As we note in example 8 of [ME] \texttt{me}, the latter specification, organized at the cluster (female) level with random-effects dimension one (a random intercept) is, in general, much more computationally efficient.

\section*{Obtaining better starting values}

Given the flexibility of mixed-effects models, you will find that some models “fail to converge” when used with your data; see \textit{Diagnosing convergence problems} in [ME] \texttt{me} for details. What we say below applies regardless of how the convergence problem revealed itself. You might have seen the error message “initial values not feasible” or some other error message, or you might have an infinite iteration log.

\texttt{meglm} provides two options to help you obtain better starting values: \texttt{startvalues()} and \texttt{startgrid()}.

\texttt{startvalues(svmethod)} allows you to specify one of four starting-value calculation methods: \texttt{zero}, \texttt{constantonly}, \texttt{fixedonly}, or \texttt{iv}. By default, \texttt{meglm} uses \texttt{startvalues(fixedonly)}. Evidently, that did not work for you. Try the other methods, starting with \texttt{startvalues(iv)}:

\begin{verbatim}
.meglm ..., ... startvalues(iv)
\end{verbatim}

If that does not solve the problem, proceed through the others.

By the way, if you have starting values for some parameters but not others—perhaps you fit a simplified model to get them—you can combine the options \texttt{startvalues()} and \texttt{from()}:

\begin{verbatim}
.meglm ..., ... // simplified model
.matrix b = e(b)
.meglm ..., ... from(b) startvalues(iv) // full model
\end{verbatim}
The other special option `meglm` provides is `startgrid()`, which can be used with or without `startvalues()`. `startgrid()` is a brute-force approach that tries various values for variances and covariances and chooses the ones that work best.

1. You may already be using a default form of `startgrid()` without knowing it. If you see `meglm` displaying Grid node 1, Grid node 2, ... following Grid node 0 in the iteration log, that is `meglm` doing a default search because the original starting values were not feasible.

   The default form tries 0.1, 1, and 10 for all variances of all random effects and, if applicable, for the residual variance.

2. `startgrid(numlist)` specifies values to try for variances of random effects.

3. `startgrid(covspec)` specifies the particular variances and covariances in which grid searches are to be performed. Variances and covariances are specified in the usual way. `startgrid(_cons[id] x[id] _cons[id]*x[id])` specifies that 0.1, 1, and 10 be tried for each member of the list.

4. `startgrid(numlist covspec)` combines the two syntaxes. You can specify `startgrid()` multiple times so that you can search the different ranges for different variances and covariances.

Our advice to you is the following:

1. If you receive an iteration log and it does not contain Grid node 1, Grid node 2, ..., then specify `startgrid(.1 1 10)`. Do that whether the iteration log was infinite or ended with some other error. In this case, we know that `meglm` did not run `startgrid()` on its own because it did not report Grid node 1, Grid node 2, etc. Your problem is poor starting values, not infeasible ones.

   A synonym for `startgrid(.1 1 10)` is just `startgrid()` without parentheses.

   Be careful, however, if you have many random effects. Specifying `startgrid()` could run a long time because it runs all possible combinations. If you have 10 random effects, that means $10^3 = 1,000$ likelihood evaluations.

   If you have many random effects, rerun your difficult `meglm` command including option `iterate(#)` and look at the results. Identify the problematic variances and search across them only. Do not just look for variances going to 0. Variances getting really big can be a problem, too, and even reasonable values can be a problem. Use your knowledge and intuition about the model.

   Perhaps you will try to fit your model by specifying `startgrid(.1 1 10 _cons[id] x[id] _cons[id]*x[id]).`

   Values 0.1, 1, and 10 are the default. Equivalent to specifying `startgrid(.1 1 10 _cons[id] x[id] _cons[id]*x[id])` is `startgrid(_cons[id] x[id] _cons[id]*x[id])`.

   Look at covariances as well as variances. If you expect a covariance to be negative but it is positive, then try negative starting values for the covariance by specifying `startgrid(-.1 -1 -10 _cons[id]*x[id])`.

   Remember that you can specify `startgrid()` multiple times. Thus you might specify both `startgrid(_cons[id] x[id])` and `startgrid(-.1 -1 -10 _cons[id]*x[id]).`
2. If you receive the message “initial values not feasible”, you know that `meglm` already tried the default `startgrid()`.

The default `startgrid()` only tried the values 0.1, 1, and 10, and only tried them on the variances of random effects. You may need to try different values or try the same values on covariances or variances of errors of observed endogenous variables.

We suggest you first rerun the model causing difficulty and include the `noestimate` option. If, looking at the results, you have an idea of which variance or covariance is a problem, or if you have few variances and covariances, we would recommend running `startgrid()` first. On the other hand, if you have no idea as to which variance or covariance is the problem and you have many of them, you will be better off if you first simplify the model. After doing that, if your simplified model does not include all the variances and covariances, you can specify a combination of `from()` and `startgrid()`.

Survey data

Multilevel modeling of survey data is a little different from standard modeling in that weighted sampling can take place at multiple levels in the model, resulting in multiple sampling weights. Most survey datasets, regardless of the design, contain one overall inclusion weight for each observation in the data. This weight reflects the inverse of the probability of ultimate selection, and by “ultimate” we mean that it factors in all levels of clustered sampling, corrections for noninclusion and oversampling, poststratification, etc.

For simplicity, in what follows, assume a simple two-stage sampling design where groups are randomly sampled and then individuals within groups are sampled. Also assume that no additional weight corrections are performed; that is, sampling weights are simply the inverse of the probability of selection. The sampling weight for observation $i$ in cluster $j$ in our two-level sample is then $w_{ij} = 1/\pi_{ij}$, where $\pi_{ij}$ is the probability that observation $i, j$ is selected. If you were performing a standard analysis such as OLS regression with `regress`, you would simply use a variable holding $w_{ij}$ as your `pweight` variable, and the fact that it came from two levels of sampling would not concern you. Perhaps you would type `vce(cluster groupvar)` where `groupvar` identifies the top-level groups to get standard errors that control for correlation within these groups, but you would still use only one weight variable.

Now take these same data and fit a two-level model with `meglm`. As seen in (5) in Methods and formulas later in this entry, it is not sufficient to use the single sampling weight $w_{ij}$, because weights enter the log likelihood at both the group level and the individual level. Instead, what is required for a two-level model under this sampling design is $w_j$, the inverse of the probability that group $j$ is selected in the first stage, and $w_{ij|j}$, the inverse of the probability that individual $i$ from group $j$ is selected at the second stage conditional on group $j$ already being selected. You cannot use $w_{ij}$ without making any assumptions about $w_j$.

Given the rules of conditional probability, $w_{ij} = w_j w_{ij|j}$. If your dataset has only $w_{ij}$, then you will need to either assume equal probability sampling at the first stage ($w_j = 1$ for all $j$) or find some way to recover $w_j$ from other variables in your data; see Rabe-Hesketh and Skrondal (2006) and the references therein for some suggestions on how to do this, but realize that there is little yet known about how well these approximations perform in practice.

What you really need to fit your two-level model are data that contain $w_j$ in addition to either $w_{ij}$ or $w_{ij|j}$. If you have $w_{ij}$—that is, the unconditional inclusion weight for observation $i, j$—then you need to divide $w_{ij}$ by $w_j$ to obtain $w_{ij|j}$.
Rabe-Hesketh and Skrondal (2006) analyzed data from the 2000 Programme for International Student Assessment (PISA) study on reading proficiency among 15-year-old American students, as performed by the Organisation for Economic Co-operation and Development (OECD). The original study was a three-stage cluster sample, where geographic areas were sampled at the first stage, schools at the second, and students at the third. Our version of the data does not contain the geographic-areas variable, so we treat this as a two-stage sample where schools are sampled at the first stage and students at the second.

```
   (Programme for International Student Assessment (PISA) 2000 data)
. describe
obs: 2,069 Programme for International Student Assessment (PISA) 2000 data
vars: 11 12 Jun 2014 10:08
size: 37,242 (_dta has notes)

variable name     storage   display value label                  variable label
                  type       format   label
female           byte      %8.0g  1 if female
isei             byte      %8.0g  International socio-economic index
w_fstuwt         float     %9.0g  Student-level weight
wnrschbw         float     %9.0g  School-level weight
high_school      byte      %8.0g  1 if highest level by either
                              parent is high school
college          byte      %8.0g  1 if highest level by either
                              parent is college
one_for          byte      %8.0g  1 if one parent foreign born
both_for         byte      %8.0g  1 if both parents are foreign born
test_lang        byte      %8.0g  1 if English (the test language) is spoken at home
pass_read        byte      %8.0g  1 if passed reading proficiency threshold
id_school        int       %8.0g  School ID
```

For student \( i \) in school \( j \), where the variable `id_school` identifies the schools, the variable `w_fstuwt` is a student-level overall inclusion weight \((w_{ij}, \text{not } w_{ij})\) adjusted for noninclusion and nonparticipation of students, and the variable `wnrschbw` is the school-level weight \(w_j\) adjusted for oversampling of schools with more minority students. The weight adjustments do not interfere with the methods prescribed above, and thus we can treat the weight variables simply as \(w_{ij}\) and \(w_j\), respectively.

Rabe-Hesketh and Skrondal (2006) fit a two-level logistic model for passing a reading proficiency threshold. We will do the same using `meglm`, but first we must reproduce the “method 1” adjusted weight variables that were used. The “method 1” adjustment scales the first-level weights so that they sum to the effective sample size of their corresponding second-level cluster.

```
. sort id_school
. generate sqw = w_fstuwt * w_fstuwt
. by id_school: egen sumw = sum(w_fstuwt)
. by id_school: egen sumsqw = sum(sqw)
. generate pst1s1 = w_fstuwt*sumw/sumsqw
```
The new variable pst1s1 holds the adjusted first-level weights. Rabe-Hesketh and Skrondal (2006) also included the school mean socioeconomic index as a covariate in their analysis. We reproduce this variable using egen.

. by id_school: egen mn_isei = mean(isei)

Here is the fitted model:

. meglm pass_read female isei mn_isei high_school college test_lang one_for > both_for [pw=pst1s1], family(beroulli) link(logit) > || id_school:, pweight(wnrschbw)

(output omitted)

Mixed-effects GLM

Number of obs = 2,069
Family: Bernoulli
Link: logit
Group variable: id_school Number of groups = 148

Obs per group:
min = 1
avg = 14.0
max = 28

Integration method: mvaghermite Integration pts. = 7
Wald chi2(8) = 88.30
Prob > chi2 = 0.0000

Log pseudolikelihood = -197395.98

(Std. Err. adjusted for 148 clusters in id_school)

pass_read | Coef. Std. Err. z P>|z| [95% Conf. Interval]
-----------|----------------------------------------------------------------------------------------------
     female | .6221369 .1540088 4.04  0.000 .3202852 .9239887
      isei  | .018215  .0048057 3.79  0.000 .0087959 .027634
     mn_isei | .0682472 .0164337 4.15  0.000 .0360378 .1004566
   high_school | .1028108 .477141 0.22  0.829 -.8323683 1.03799
   college  | .4531688 .5053447 0.90  0.370 -.5372885 1.443626
  test_lang | .6251822 .3821182 1.64  0.102 -.8323683 1.03799
    one_for | -.1089314 .2739724 -0.40  0.691 -.6459075 .4280447
   both_for | -.2804038 .3264681 -0.86  0.390 -.9202696 .359462
    _cons  | -5.877565 .954525 -6.16  0.000 -7.7484 -4.006731

id_school  var(_cons) | .2955769 .1243375 .1295996 .6741201

Notes:
1. We specified the level-one weights using standard Stata weight syntax, that is, [pw=pst1s1].
2. We specified the level-two weights via the pweight(wnrschbw) option as part of the random-effects specification for the id_school level. As such, it is treated as a school-level weight. Accordingly, wnrschbw needs to be constant within schools, and meglm did check for that before estimating.
3. As is the case with other estimation commands in Stata, standard errors in the presence of sampling weights are robust.
4. Robust standard errors are clustered at the top level of the model, and this will always be true unless you specify vce(cluster clustvar), where clustvar identifies an even higher level of grouping.
Example 6

meglm also supports the svy prefix (see [SVY] svy) for the linearized variance estimator. Here we refit the model from the previous example using the svy prefix after we svyset (see [SVY] svyset) the survey design variables.

```
.svyset id_school, weight(wnrschbw) || _n, weight(pst1s1)
Note: Stage 1 is sampled with replacement; further stages will be ignored for variance estimation.
pweight: <none>
VCE: linearized
Single unit: missing
Strata 1: <one>
   SU 1: id_school
   FPC 1: <zero>
Weight 1: wnrschbw
Strata 2: <one>
   SU 2: <observations>
   FPC 2: <zero>
Weight 2: pst1s1
```

```
.svy: meglm pass_read female isei mn_iseli high_school college test_lang > one_for both_for, family(bernoulli) link(logit) || id_school:
   (running meglm on estimation sample)
Survey: Mixed-effects GLM
Number of strata = 1 Number of obs = 2,069
Number of PSUs = 148 Population size = 346,373.74
Design df = 147
F( 8, 140) = 10.51
Prob > F = 0.0000

| Coef. | Std. Err. | t    | P>|t|   | [95% Conf. Interval] |
|-------|-----------|------|-------|----------------------|
| pass_read | | | | |
| female   | .6221369  | .1540088 | 4.04  | 0.000 | .3177796  | .9264943  |
| isei     | .018215   | .0048057 | 3.79  | 0.000 | .0087177  | .0277122  |
| mn_iseli | .0682472  | .0164337 | 4.15  | 0.000 | .0357704  | .1007240  |
| high_school | .1028108 | .477141  | 0.22  | 0.830 | -.8401311 | 1.045753  |
| college  | .4531688  | .5053447 | 0.90  | 0.371 | -.5455101 | 1.451848  |
| test_lang | .6251822  | .3821182 | 1.64  | 0.104 | -.1299725 | 1.380337  |
| one_for  | -.1089314 | .2739724 | -0.40 | 0.692 | -.6503648 | .432502   |
| both_for | -.2804038 | .3264681 | -0.86 | 0.392 | -.925581  | .3647734  |
| _cons    | -5.877565 | .9545256 | -6.16 | 0.000 | -7.763929 | -3.991201 |
| id_school | | | | |
| var(_cons) | .2955769  | .1243375 | 1.287156 | .6787495 |
```

Notes:

1. We svyset the design variables: id_school is the PSU variable, wnrschbw contains weights at the PSU level, _n specifies that the students are identified by the individual observations, and pst1s1 contains our adjusted student-level conditional weights.
2. svyset notes the lack of a finite population correction in the first stage and informs us that only the first-stage unit information will be used in the linearized variance estimator. However, the svy prefix will still pass the stage-two weights to meglm.
3. svy produces a different header, giving us an estimate of the population size, the design degrees of freedom, and the number of first-stage sampling units.
Video example

Tour of multilevel GLMs

Stored results

`meglm` stores the following in `e()`:

Scalars

- `e(N)` number of observations
- `e(k)` number of parameters
- `e(k_dv)` number of dependent variables
- `e(k_eq)` number of equations in `e(b)`
- `e(k_eq_model)` number of equations in overall model test
- `e(k_cat)` number of categories (with ordinal outcomes)
- `e(k_f)` number of fixed-effects parameters
- `e(k_r)` number of random-effects parameters
- `e(k_rs)` number of variances
- `e(k_rc)` number of covariances
- `e(df_m)` model degrees of freedom
- `e(ll)` log likelihood
- `e(chi2)` \( \chi^2 \)
- `e(p)` significance
- `e(ll_c)` log likelihood, comparison model
- `e(chi2_c)` \( \chi^2 \), comparison model
- `e(df_c)` degrees of freedom, comparison model
- `e(p_c)` significance, comparison model
- `e(N_clust)` number of clusters
- `e(rank)` rank of `e(V)`
- `e(ic)` number of iterations
- `e(rc)` return code
- `e(converged)` 1 if converged, 0 otherwise

Macros

- `e(cmd)` `gsem`
- `e(cmd2)` `meglm`
- `e(cmdline)` command as typed
- `e(depvar)` name of dependent variable
- `e(wtype)` weight type
- `e(wexp)` weight expression (first-level weights)
- `e(fweight)` `fweight` variable for \( k \)th highest level, if specified
- `e(iweight)` `iweight` variable for \( k \)th highest level, if specified
- `e(pweight)` `pweight` variable for \( k \)th highest level, if specified
- `e(covariates)` list of covariates
- `e(ivars)` grouping variables
- `e(model)` name of marginal model
- `e(title)` title in estimation output
- `e(link)` link
- `e(family)` family
- `e(clustvar)` name of cluster variable
- `e(offset)` offset
- `e(binomial)` binomial number of trials (with binomial models)
- `e(dispersion)` mean or constant (with negative binomial models)
- `e(intmethod)` integration method
- `e(n_quad)` number of integration points
- `e(chi2type)` \( \chi^2 \) Wald; type of model \( \chi^2 \)
- `e(vce)` `vcetype` specified in `vce()`
- `e(vcetype)` title used to label Std. Err.
- `e(opt)` type of optimization
- `e(which)` \( \max \) or \( \min \); whether optimizer is to perform maximization or minimization
- `e(ml_method)` type of `ml` method
- `e(user)` name of likelihood-evaluator program
Without a loss of generality, consider a two-level generalized mixed-effects model

\[ E(y_j | X_j, u_j) = g^{-1}(X_j \beta + Z_j u_j), \quad y \sim F \]

for \( j = 1, \ldots, M \) clusters, with the \( j \)th cluster consisting of \( n_j \) observations, where, for the \( j \)th cluster, \( y_j \) is the \( n_j \times 1 \) response vector, \( X_j \) is the \( n_j \times p \) matrix of fixed predictors, \( Z_j \) is the \( n_j \times q \) matrix of random predictors, \( u_j \) is the \( q \times 1 \) vector of random effects, \( \beta \) is the \( p \times 1 \) vector of regression coefficients on the fixed predictors, and we use \( \Sigma \) to denote the unknown \( q \times q \) variance matrix of the random effects. For simplicity, we consider a model with no auxiliary parameters.

Let \( \eta_j \) be the linear predictor, \( \eta_j = X_j \beta + Z_j u_j \), that also includes the offset or the exposure variable when \( \text{offset()} \) or \( \text{exposure()} \) is specified. Let \( y_{ij} \) and \( \eta_{ij} \) be the \( i \)th individual elements of \( y_j \) and \( \eta_j \), \( i = 1, \ldots, n_j \). Let \( f(y_{ij} | \eta_{ij}) \) be the conditional density function for the response at observation \( i \). Because the observations are assumed to be conditionally independent, we can overload the definition of \( f(\cdot) \) with vector inputs to mean

\[ \log f(y_j | \eta_j) = \sum_{i=1}^{n_j} \log f(y_{ij} | \eta_{ij}) \]
The random effects $u_j$ are assumed to be multivariate normal with mean $0$ and variance $\Sigma$. The likelihood function for cluster $j$ is given by

$$L_j(\beta, \Sigma) = (2\pi)^{-q/2} |\Sigma|^{-1/2} \int_{\mathbb{R}^q} f(y_j | \eta_j) \exp \left( -\frac{1}{2} u_j' \Sigma^{-1} u_j \right) du_j$$

where $\mathbb{R}$ denotes the set of values on the real line and $\mathbb{R}^q$ is the analog in $q$-dimensional space.

The multivariate integral in (3) is generally not tractable, so we must use numerical methods to approximate the integral. We can use a change-of-variables technique to transform this multivariate integral into a set of nested univariate integrals. Each univariate integral can then be evaluated using a form of Gaussian quadrature.

meglm supports three types of Gauss–Hermite quadratures: mean–variance adaptive Gauss–Hermite quadrature (MVAGH), mode-curvature adaptive Gauss–Hermite quadrature (MCAGH), and nonadaptive Gauss–Hermite quadrature (GHQ). meglm also offers the Laplacian-approximation method, which is used as a default method for crossed mixed-effects models. Below we describe the four methods. The methods described below are based on Skrondal and Rabe-Hesketh (2004, chap. 6.3).

**Gauss–Hermite quadrature**

Let $u_j = L v_j$, where $v_j$ is a $q \times 1$ random vector whose elements are independently standard normal variables and $L$ is the Cholesky decomposition of $\Sigma$, $\Sigma = LL'$. Then $\eta_j = X_j \beta + Z_j L v_j$, and the likelihood in (3) becomes

$$L_j(\beta, \Sigma) = (2\pi)^{-q/2} \int_{\mathbb{R}^q} \exp \left\{ \log f(y_j | \eta_j) - \frac{1}{2} v_j' v_j \right\} dv_j$$

$$= (2\pi)^{-q/2} \int_{-\infty}^{\infty} \ldots \int_{-\infty}^{\infty} \exp \left\{ \log f(y_j | \eta_j) - \frac{1}{2} \sum_{k=1}^{q} v_{jk}^2 \right\} dv_{j1}, \ldots, dv_{jq}$$

Consider a $q$-dimensional quadrature grid containing $r$ quadrature points in each dimension. Let $a_k = (a_{k1}, \ldots, a_{kq})'$ be a point on this grid, and let $w_k = (w_{k1}, \ldots, w_{kq})'$ be the vector of corresponding weights. The GHQ approximation to the likelihood is

$$L_j^{GHQ}(\beta, \Sigma) = \sum_{k_1=1}^{r} \ldots \sum_{k_q=1}^{r} \left[ \exp \left\{ \log f(y_j | \eta_{jk}) \right\} \prod_{p=1}^{q} w_{kp} \right]$$

$$= \sum_{k_1=1}^{r} \ldots \sum_{k_q=1}^{r} \left[ \exp \left\{ \sum_{i=1}^{n_j} \log f(y_{ij} | \eta_{ijk}) \right\} \prod_{p=1}^{q} w_{kp} \right]$$

where

$$\eta_{jk} = X_j \beta + Z_j L a_k$$

and $\eta_{ijk}$ is the $i$th element of $\eta_{jk}$.
Adaptive Gauss–Hermite quadrature

This section sets the stage for MV AGH quadrature and MCAGH quadrature.

Let’s reconsider the likelihood in (4). We use $\phi(v_j)$ to denote a multivariate standard normal with mean $0$ and variance $I_q$, and we use $\phi(v_j|\mu_j, \Lambda_j)$ to denote a multivariate normal with mean $\mu_j$ and variance $\Lambda_j$.

For fixed model parameters, the posterior density for $v_j$ is proportional to

$$\phi(v_j)f(y_j|\eta_j)$$

where

$$\eta_j = X_j\beta + Z_jLv_j$$

It is reasonable to assume that this posterior density can be approximated by a multivariate normal density with mean vector $\mu_j$ and variance matrix $\Lambda_j$. Instead of using the prior density of $v_j$ as the weighting distribution in the integral, we can use our approximation for the posterior density,

$$\mathcal{L}_j(\beta, \Sigma) = \int_{\mathbb{R}^q} \frac{f(y_j|\eta_j)\phi(v_j)}{\phi(v_j|\mu_j, \Lambda_j)} \phi(v_j|\mu_j, \Lambda_j) dv_j$$

Then the MVAGH approximation to the likelihood is

$$\mathcal{L}_j^{MVAGH}(\beta, \Sigma) = \sum_{k_1=1}^{r} \cdots \sum_{k_q=1}^{r} \left[ \exp \left\{ \log f(y_j|\eta_{jk}) \right\} \prod_{p=1}^{q} w_{jk_p}^{*} \right]$$

where

$$\eta_{jk} = X_j\beta + Z_jL\alpha_{jk}^{*}$$

and $\alpha_{jk}^{*}$ and $w_{jk_p}^{*}$ are the abscissas and weights after an orthogonalizing transformation of $\alpha_{jk}$ and $w_{jk_p}$, respectively, which eliminates posterior covariances between the random effects.

Estimates of $\mu_j$ and $\Lambda_j$ are computed using one of two different methods. The mean $\mu_j$ and variance $\Lambda_j$ are computed iteratively by updating the posterior moments with the MVAGH approximation, starting with a $0$ mean vector and identity variance matrix. For the MCAGH approximation, $\mu_j$ and $\Lambda_j$ are computed by optimizing the integrand with respect to $v_j$, where $\mu_j$ is the optimal value and $\Lambda_j$ is the curvature at $\mu_j$.

Laplacian approximation

Consider the likelihood in (3) and denote the argument in the exponential function by

$$h(\beta, \Sigma, u_j) = \log f(y_j|X_j\beta + Z_ju_j) - \frac{1}{2}u_j^T\Sigma^{-1}u_j$$
The Laplacian approximation is based on a second-order Taylor expansion of \( h(\beta, \Sigma, u_j) \) about the value of \( u_j \) that maximizes it. The first and second partial derivatives with respect to \( u_j \) are

\[
\begin{align*}
 h'(\beta, \Sigma, u_j) &= \frac{\partial h(\beta, \Sigma, u_j)}{\partial u_j} = Z'_j \frac{\partial \log f(y_j|\eta_j)}{\partial \eta_j} - \Sigma^{-1} u_j \\
 h''(\beta, \Sigma, u_j) &= \frac{\partial^2 h(\beta, \Sigma, u_j)}{\partial u_j \partial u'_j} = Z'_j \frac{\partial^2 \log f(y_j|\eta_j)}{\partial \eta_j \partial \eta'_j} Z_j - \Sigma^{-1}
\end{align*}
\]

The maximizer of \( h(\beta, \Sigma, u_j) \) is \( \hat{u}_j \) such that \( h'(\beta, \Sigma, \hat{u}_j) = 0 \). The integral in (3) is proportional to the posterior density of \( u_j \) given the data, so \( \hat{u}_j \) is also the posterior mode.

Let

\[
\begin{align*}
 \hat{p}_j &= X_j \beta + Z_j \hat{u}_j \\
 S_1 &= \frac{\partial \log f(y_j|\hat{p}_j)}{\partial \hat{p}_j} \\
 S_2 &= \frac{\partial S_1}{\partial \hat{p}_j} = \frac{\partial^2 \log f(y_j|\hat{p}_j)}{\partial \hat{p}_j \partial \hat{p}'_j} \\
 H_j &= h''(\beta, \Sigma, \hat{u}_j) = Z'_j S_2 Z_j - \Sigma^{-1}
\end{align*}
\]

then

\[
0 = h'(\beta, \Sigma, \hat{u}_j) = Z'_j S_1 - \Sigma^{-1} \hat{u}_j
\]

Given the above, the second-order Taylor approximation takes the form

\[
h(\beta, \Sigma, u_j) \approx h(\beta, \Sigma, \hat{u}_j) + \frac{1}{2} (u_j - \hat{u}_j)' H_j (u_j - \hat{u}_j)
\]

because the first-order derivative term is 0. The integral is approximated by

\[
\int_{\mathbb{R}^q} \exp\{h(\beta, \Sigma, u_j)\} \, du_j \approx (2\pi)^{q/2} |-H_j|^{-1/2} \exp\{h(\beta, \Sigma, \hat{u}_j)\}
\]

Thus the Laplacian approximated log likelihood is

\[
\log L_j^\text{Lap}(\beta, \Sigma) = -\frac{1}{2} \log |\Sigma| - \frac{1}{2} \log |-H_j| + h(\beta, \Sigma, \hat{u}_j)
\]

The log likelihood for the entire dataset is simply the sum of the contributions of the \( M \) individual clusters, namely, \( L(\beta, \Sigma) = \sum_{j=1}^M L_j(\beta, \Sigma) \).

Maximization of \( L(\beta, \Sigma) \) is performed with respect to \((\beta, \sigma^2)\), where \( \sigma^2 \) is a vector comprising the unique elements of \( \Sigma \). Parameter estimates are stored in \( e(b) \) as \((\hat{\beta}, \hat{\sigma}^2)\), with the corresponding variance–covariance matrix stored in \( e(V) \). In the presence of auxiliary parameters, their estimates and standard errors are included in \( e(b) \) and \( e(V) \), respectively.
Survey data

In the presence of sampling weights, following Rabe-Hesketh and Skrondal (2006), the weighted log pseudolikelihood for a two-level model is given as

\[
L(\beta, \Sigma) = \sum_{j=1}^{M} w_j \log \int_{-\infty}^{\infty} \exp \left\{ \sum_{i=1}^{n_j} w_{ij} \log f(y_{ij} | \eta_{ij}) \right\} \phi(v_{j1}) \, dv_{j1}
\]

(5)

where \( w_j \) is the inverse of the probability of selection for the \( j \)th cluster; \( w_{ij} \) is the inverse of the conditional probability of selection of individual \( i \), given the selection of cluster \( j \), \( f(\cdot) \) is as defined previously; and \( \phi(\cdot) \) is the standard multivariate normal density.

Weighted estimation is achieved through the direct application of \( w_j \) and \( w_{ij} \) into the likelihood calculations as detailed above to reflect replicated clusters for \( w_j \) and replicated observations within clusters for \( w_{ij} \). Because this estimation is based on replicated clusters and observations, frequency weights are handled similarly.

Weights are not allowed with crossed models or the Laplacian approximation.

References


Also see

[ME] meglm postestimation — Postestimation tools for meglm

[ME] mixed — Multilevel mixed-effects linear regression

[ME] me — Introduction to multilevel mixed-effects models

[R] glm — Generalized linear models

[SEM] intro 5 — Tour of models (Multilevel mixed-effects models)

[SVY] svy estimation — Estimation commands for survey data

[U] 20 Estimation and postestimation commands
Postestimation commands

The following postestimation command is of special interest after `meglm`:

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>estat group</code></td>
<td>summarize the composition of the nested groups</td>
</tr>
</tbody>
</table>

The following standard postestimation commands are also available:

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>contrast</code></td>
<td>contrasts and ANOVA-style joint tests of estimates</td>
</tr>
<tr>
<td><code>estat ic</code></td>
<td>Akaike’s and Schwarz’s Bayesian information criteria (AIC and BIC)</td>
</tr>
<tr>
<td><code>estat summarize</code></td>
<td>summary statistics for the estimation sample</td>
</tr>
<tr>
<td><code>estat vce</code></td>
<td>variance–covariance matrix of the estimators (VCE)</td>
</tr>
<tr>
<td><code>estat (svy)</code></td>
<td>postestimation statistics for survey data</td>
</tr>
<tr>
<td><code>estimates</code></td>
<td>cataloging estimation results</td>
</tr>
<tr>
<td><code>*hausman</code></td>
<td>Hausman’s specification test</td>
</tr>
<tr>
<td><code>lincom</code></td>
<td>point estimates, standard errors, testing, and inference for linear combinations of coefficients</td>
</tr>
<tr>
<td><code>*lrtest</code></td>
<td>likelihood-ratio test</td>
</tr>
<tr>
<td><code>margins</code></td>
<td>marginal means, predictive margins, marginal effects, and average marginal effects</td>
</tr>
<tr>
<td><code>marginsplot</code></td>
<td>graph the results from margins (profile plots, interaction plots, etc.)</td>
</tr>
<tr>
<td><code>nlcom</code></td>
<td>point estimates, standard errors, testing, and inference for nonlinear combinations of coefficients</td>
</tr>
<tr>
<td><code>predict</code></td>
<td>predictions, residuals, influence statistics, and other diagnostic measures</td>
</tr>
<tr>
<td><code>predictnl</code></td>
<td>point estimates, standard errors, testing, and inference for generalized predictions</td>
</tr>
<tr>
<td><code>pwcompare</code></td>
<td>pairwise comparisons of estimates</td>
</tr>
<tr>
<td><code>test</code></td>
<td>Wald tests of simple and composite linear hypotheses</td>
</tr>
<tr>
<td><code>testnl</code></td>
<td>Wald tests of nonlinear hypotheses</td>
</tr>
</tbody>
</table>

* `hausman` and `lrtest` are not appropriate with `svy` estimation results.
predict

Description for predict

predict creates a new variable containing predictions such as mean responses; linear predictions; density and distribution functions; standard errors; and raw, Pearson, deviance, and Anscombe residuals.

Menu for predict

Statistics  >  Postestimation

Syntax for predict

Syntax for obtaining predictions of the outcome and other statistics

\[\text{predict [type] newvarsspec [if] [in] [, statistic options]}\]

Syntax for obtaining estimated random effects and their standard errors

\[\text{predict [type] newvarsspec [if] [in], reffects [re_options]}\]

Syntax for obtaining ML scores

\[\text{predict [type] newvarsspec [if] [in], scores}\]

newvarsspec is stub* or newvarlist.

\[
\begin{array}{ll}
\text{statistic} & \text{Description} \\
\hline
\text{Main} & \\
\text{mu} & \text{mean response; the default} \\
\text{pr} & \text{synonym for mu for ordinal and binary response models} \\
\text{eta} & \text{fitted linear predictor} \\
\text{xb} & \text{linear predictor for the fixed portion of the model only} \\
\text{stdp} & \text{standard error of the fixed-portion linear prediction} \\
\text{density} & \text{predicted density function} \\
\text{distribution} & \text{predicted distribution function} \\
\text{residuals} & \text{raw residuals; available only with the Gaussian family} \\
\text{pearson} & \text{Pearson residuals} \\
\text{deviance} & \text{deviance residuals} \\
\text{anscombe} & \text{Anscombe residuals} \\
\end{array}
\]

These statistics are available both in and out of sample; type \text{predict ... if \_e(sample) ... if wanted} only for the estimation sample.
### options

<table>
<thead>
<tr>
<th>Description</th>
<th>Main</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>conditional(ctype)</code></td>
<td>compute statistic conditional on estimated random effects; default is <code>conditional(ebmeans)</code></td>
</tr>
<tr>
<td><code>marginal</code></td>
<td>compute statistic marginally with respect to the random effects</td>
</tr>
<tr>
<td><code>nooffset</code></td>
<td>make calculation ignoring offset or exposure</td>
</tr>
<tr>
<td><code>outcome(outcome)</code></td>
<td>outcome category for predicted probabilities for ordinal models</td>
</tr>
</tbody>
</table>

**Integration**

<table>
<thead>
<tr>
<th>Description</th>
<th><code>int_options</code></th>
</tr>
</thead>
<tbody>
<tr>
<td>integration options</td>
<td>pearson, deviance, anscombe may not be combined with marginal.</td>
</tr>
<tr>
<td>For ordinal outcomes, you specify one or $k$ new variables in <code>newvarlist</code> with <code>mu</code> and <code>pr</code>, where $k$ is the number of outcomes. If you do not specify <code>outcome()</code>, these options assume <code>outcome(#1)</code>.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Description</th>
<th><code>ctype</code></th>
</tr>
</thead>
<tbody>
<tr>
<td>empirical Bayes means of random effects; the default</td>
<td><code>ebmeans</code></td>
</tr>
<tr>
<td>empirical Bayes modes of random effects</td>
<td><code>ebmodes</code></td>
</tr>
<tr>
<td>prediction for the fixed portion of the model only</td>
<td><code>fixedonly</code></td>
</tr>
</tbody>
</table>

**Integration**

<table>
<thead>
<tr>
<th>Description</th>
<th><code>int_options</code></th>
</tr>
</thead>
<tbody>
<tr>
<td>integration options</td>
<td><code>intp</code>oints(#) use # quadrature points to compute marginal predictions and empirical Bayes means</td>
</tr>
<tr>
<td>set maximum number of iterations in computing statistics involving empirical Bayes estimators</td>
<td><code>iterate(#)</code></td>
</tr>
<tr>
<td>set convergence tolerance for computing statistics involving empirical Bayes estimators</td>
<td><code>tolerance(#)</code></td>
</tr>
</tbody>
</table>

### Options for predict

- `mu`, the default, calculates the expected value of the outcome.
- `pr` calculates predicted probabilities and is a synonym for `mu`. This option is available only for ordinal and binary response models.
- `eta` calculates the fitted linear prediction.
xb calculates the linear prediction $x\beta$ using the estimated fixed effects (coefficients) in the model. This is equivalent to fixing all random effects in the model to their theoretical (prior) mean value of 0.

stdp calculates the standard error of the fixed-effects linear predictor $x\beta$.

density calculates the density function. This prediction is computed using the current values of the observed variables, including the dependent variable.

distribution calculates the distribution function. This prediction is computed using the current values of the observed variables, including the dependent variable.

residuals calculates raw residuals, that is, responses minus the fitted values. This option is available only for the Gaussian family.

pearson calculates Pearson residuals. Pearson residuals that are large in absolute value may indicate a lack of fit.

deviance calculates deviance residuals. Deviance residuals are recommended by McCullagh and Nelder (1989) as having the best properties for examining the goodness of fit of a GLM. They are approximately normally distributed if the model is correctly specified. They can be plotted against the fitted values or against a covariate to inspect the model fit.

anscombe calculates Anscombe residuals, which are designed to closely follow a normal distribution.

conditional(ctype) and marginal specify how random effects are handled in computing statistic.

conditional() specifies that statistic will be computed conditional on specified or estimated random effects.

conditional(ebmeans), the default, specifies that empirical Bayes means be used as the estimates of the random effects. These estimates are also known as posterior mean estimates of the random effects.

conditional(ebmode) specifies that empirical Bayes modes be used as the estimates of the random effects. These estimates are also known as posterior mode estimates of the random effects.

conditional(fixedonly) specifies that all random effects be set to zero, equivalent to using only the fixed portion of the model.

marginal specifies that the predicted statistic be computed marginally with respect to the random effects, which means that statistic is calculated by integrating the prediction function with respect to all the random effects over their entire support.

Although this is not the default, marginal predictions are often very useful in applied analysis. They produce what are commonly called population-averaged estimates. They are also required by margins.

nooffset is relevant only if you specified offset(varname_o) or exposure(varname_e) with meglm. It modifies the calculations made by predict so that they ignore the offset or the exposure variable; the linear prediction is treated as $X\beta + Zu$ rather than $X\beta + Zu + offset$, or $X\beta + Zu + \ln(exposure)$, whichever is relevant.

outcome(outcome) specifies the outcome for which the predicted probabilities are to be calculated. outcome() should contain either one value of the dependent variable or one of #1, #2, ..., with #1 meaning the first category of the dependent variable, #2 meaning the second category, etc.
meglm postestimation — Postestimation tools for meglm

reffects calculates estimates of the random effects using empirical Bayes predictions. By default, or if the ebmeans option is specified, empirical Bayes means are computed. With the ebmodes option, empirical Bayes modes are computed. You must specify \( q \) new variables, where \( q \) is the number of random-effects terms in the model. However, it is much easier to just specify \( stub^* \) and let Stata name the variables \( stub1, stub2, \ldots, stubq \) for you.

ebmeans specifies that empirical Bayes means be used to predict the random effects.

ebmodes specifies that empirical Bayes modes be used to predict the random effects.

reses(\( stub^* \) | \( newvarlist \)) calculates standard errors of the empirical Bayes estimators and stores the result in \( newvarlist \). This option requires the reffects option. You must specify \( q \) new variables, where \( q \) is the number of random-effects terms in the model. However, it is much easier to just specify \( stub^* \) and let Stata name the variables \( stub1, stub2, \ldots, stubq \) for you. The new variables will have the same storage type as the corresponding random-effects variables.

The reffects and reses() options often generate multiple new variables at once. When this occurs, the random effects (and standard errors) contained in the generated variables correspond to the order in which the variance components are listed in the output of meglm. The generated variables are also labeled to identify their associated random effect.

scores calculates the scores for each coefficient in \( e(b) \). This option requires a new variable list of length equal to the number of columns in \( e(b) \). Otherwise, use the \( stub^* \) option to have predict generate enumerated variables with prefix \( stub \).

\underline{Integration}

intpoints(#) specifies the number of quadrature points used to compute marginal predictions and the empirical Bayes means; the default is the value from estimation.

iterate(#) specifies the maximum number of iterations when computing statistics involving empirical Bayes estimators; the default is the value from estimation.

tolerance(#) specifies convergence tolerance when computing statistics involving empirical Bayes estimators; the default is the value from estimation.
margins

Description for margins

margins estimates margins of response for mean responses and linear predictions.

Menu for margins

Statistics > Postestimation

Syntax for margins

margins [marginlist] [, options]
margins [marginlist], predict(statistic ...) [predict(statistic ...) ...] [options]

<table>
<thead>
<tr>
<th>statistic</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mu</td>
<td>mean response; the default</td>
</tr>
<tr>
<td>pr</td>
<td>synonym for mu for ordinal and binary response models</td>
</tr>
<tr>
<td>eta</td>
<td>fitted linear predictor</td>
</tr>
<tr>
<td>xb</td>
<td>linear predictor for the fixed portion of the model only</td>
</tr>
<tr>
<td>stdp</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>density</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>distribution</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>residuals</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>pearson</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>deviance</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>anscome</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>reffects</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>scores</td>
<td>not allowed with margins</td>
</tr>
</tbody>
</table>

Options conditional(ebmeans) and conditional(ebmode) are not allowed with margins.
Option marginal is assumed where applicable if conditional(fixedonly) is not specified.

Statistics not allowed with margins are functions of stochastic quantities other than e(b).

For the full syntax, see [R] margins.
estat

Description for estat

    estat group reports the number of groups and minimum, average, and maximum group sizes for each level of the model. Model levels are identified by the corresponding group variable in the data. Because groups are treated as nested, the information in this summary may differ from what you would get if you used the tabulate command on each group variable individually.

Menu for estat

    Statistics  Postestimation

Syntax for estat

    estat group

Remarks and examples

    Various predictions, statistics, and diagnostic measures are available after fitting a mixed-effects model using meglm. For the most part, calculation centers around obtaining predictions of the random effects. Random effects are not estimated when the model is fit but instead need to be predicted after estimation.

➢ Example 1

    In example 2 of [ME] meglm, we modeled the probability of contraceptive use among Bangladeshi women by fitting a mixed-effects logistic regression model. To facilitate a more direct comparison between urban and rural women, we express rural status in terms of urban status and eliminate the constant from both the fixed-effects part and the random-effects part.
. use http://www.stata-press.com/data/r14/bangladesh
(Bangladesh Fertility Survey, 1989)
. generate byte rural = 1 - urban
. meglm c_use rural urban age child*, nocons || district: rural urban, nocons
> family(bernoulli) link(logit)

Fitting fixed-effects model:
Iteration 0:  log likelihood = -1229.5485
Iteration 1:  log likelihood = -1228.5268
Iteration 2:  log likelihood = -1228.5263
Iteration 3:  log likelihood = -1228.5263

Refining starting values:
Grid node 0:  log likelihood = -1208.3922

Fitting full model:
Iteration 0:  log likelihood = -1208.3922 (not concave)
Iteration 1:  log likelihood = -1203.6498 (not concave)
Iteration 2:  log likelihood = -1200.6662
Iteration 3:  log likelihood = -1199.9939
Iteration 4:  log likelihood = -1199.3784
Iteration 5:  log likelihood = -1199.3272
Iteration 6:  log likelihood = -1199.3268
Iteration 7:  log likelihood = -1199.3268

Mixed-effects GLM Number of obs = 1,934
Family: Bernoulli
Link: logit
Group variable: district Number of groups = 60
Obs per group:
  min =  2
  avg = 32.2
  max = 118

Integration method: mvaghermite Integration pts. = 7
Wald chi2(6) = 120.59
Log likelihood = -1199.3268 Prob > chi2 = 0.0000
( 1)  [c_use]_cons = 0

| Coef.  | Std. Err. | z   | P>|z|  | [95% Conf. Interval] |
|--------|-----------|-----|------|----------------------|
| rural  | -1.712549 | .1603689 | -10.68 | 0.000 | -2.026866 | -1.398232 |
| urban  | -.9004495 | .1674683 | -5.38 | 0.000 | -1.228681 | -.5722176 |
| age    | -.0264472 | .0080196 | -3.30 | 0.001 | -.0421652 | -.0107291 |
| child1 | 1.132291  | .1603052 | 7.06 | 0.000 | .8180983 | 1.446483  |
| child2 | 1.358692  | .1769369 | 7.68 | 0.000 | 1.011902 | 1.705482  |
| child3 | 1.354788  | .1827459 | 7.41 | 0.000 | .9966122 | 1.712963  |
| _cons  | 0 (omitted)| | | |

<table>
<thead>
<tr>
<th>Coef.</th>
<th>Std. Err.</th>
</tr>
</thead>
<tbody>
<tr>
<td>var(rural)</td>
<td>.3882825</td>
</tr>
<tr>
<td>var(urban)</td>
<td>.239777</td>
</tr>
</tbody>
</table>

LR test vs. logistic model: chi2(2) = 58.40  Prob > chi2 = 0.0000

Note: LR test is conservative and provided only for reference.

We used the binary variables, rural and urban, instead of the factor notation i.urban because,
although supported in the fixed-effects specification of the model, such notation is not supported in
random-effects specifications.

This particular model allows for district random effects that are specific to the rural and urban
areas of that district and that can be interpreted as such. We can obtain predictions of posterior means
of the random effects and their standard errors by typing
. predict re_rural re_urban, reffects reses(se_rural se_urban)  
(calculating posterior means of random effects)  
(using 7 quadrature points)

The order in which we specified the variables to be generated corresponds to the order in which the variance components are listed in `meglm` output. If in doubt, a simple `describe` will show how these newly generated variables are labeled just to be sure.

Having generated estimated random effects and standard errors, we can now list them for the first 10 districts:

```
. by district, sort: generate tag = (_n==1)
. list district re_rural se_rural re_urban se_urban if district <= 10 & tag,
> sep(0)
```

<table>
<thead>
<tr>
<th>district</th>
<th>re_rural</th>
<th>se_rural</th>
<th>re_urban</th>
<th>se_urban</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>-.952374</td>
<td>.316291</td>
<td>-.5619418</td>
<td>.2329456</td>
</tr>
<tr>
<td>118.</td>
<td>-.0425217</td>
<td>.3819309</td>
<td>3.80e-18</td>
<td>.4896702</td>
</tr>
<tr>
<td>138.</td>
<td>-9.33e-17</td>
<td>.6231232</td>
<td>.2229486</td>
<td>.4658747</td>
</tr>
<tr>
<td>140.</td>
<td>-.2703357</td>
<td>.3980832</td>
<td>.574464</td>
<td>.3962131</td>
</tr>
<tr>
<td>170.</td>
<td>.0691029</td>
<td>.3101591</td>
<td>.0074569</td>
<td>.4650451</td>
</tr>
<tr>
<td>209.</td>
<td>-.3939819</td>
<td>.2759802</td>
<td>.2622263</td>
<td>.4177785</td>
</tr>
<tr>
<td>274.</td>
<td>-.1904756</td>
<td>.4043461</td>
<td>8.69e-18</td>
<td>.4896702</td>
</tr>
<tr>
<td>292.</td>
<td>.0382993</td>
<td>.3177392</td>
<td>.2250237</td>
<td>.4654329</td>
</tr>
<tr>
<td>329.</td>
<td>-.3715211</td>
<td>.3919996</td>
<td>.0628076</td>
<td>.453568</td>
</tr>
<tr>
<td>352.</td>
<td>-.5624707</td>
<td>.4763545</td>
<td>-5.67e-18</td>
<td>.4896702</td>
</tr>
</tbody>
</table>

The estimated standard errors are conditional on the values of the estimated model parameters: $\beta$ and the components of $\Sigma$. Their interpretation is therefore not one of standard sample-to-sample variability but instead one that does not incorporate uncertainty in the estimated model parameters; see `Methods and formulas`. That stated, conditional standard errors can still be used as a measure of relative precision, provided that you keep this caveat in mind.

You can also obtain predictions of posterior modes and compare them with the posterior means:

```
. predict mod_rural mod_urban, reffects ebmodes  
(calculating posterior modes of random effects)
. list district re_rural mod_rural re_urban mod_urban if district <= 10 & tag,
> sep(0)
```

<table>
<thead>
<tr>
<th>district</th>
<th>re_rural</th>
<th>mod_rural</th>
<th>re_urban</th>
<th>mod_urban</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>-.952374</td>
<td>-.9295366</td>
<td>-.5619418</td>
<td>-.5584528</td>
</tr>
<tr>
<td>118.</td>
<td>-.0425217</td>
<td>-.0306312</td>
<td>3.80e-18</td>
<td>0</td>
</tr>
<tr>
<td>138.</td>
<td>-9.33e-17</td>
<td>0</td>
<td>.2229486</td>
<td>.2223551</td>
</tr>
<tr>
<td>140.</td>
<td>-.2703357</td>
<td>-.2529507</td>
<td>.574464</td>
<td>.5644512</td>
</tr>
<tr>
<td>170.</td>
<td>.0691029</td>
<td>.0789803</td>
<td>.0074569</td>
<td>.0077525</td>
</tr>
<tr>
<td>209.</td>
<td>-.3939819</td>
<td>-.3803784</td>
<td>.2622263</td>
<td>.2595116</td>
</tr>
<tr>
<td>274.</td>
<td>-.1904756</td>
<td>-.1737696</td>
<td>8.69e-18</td>
<td>0</td>
</tr>
<tr>
<td>292.</td>
<td>.0382993</td>
<td>.0488528</td>
<td>.2250237</td>
<td>.2244676</td>
</tr>
<tr>
<td>329.</td>
<td>-.3715211</td>
<td>-.3540084</td>
<td>.0628076</td>
<td>.060546</td>
</tr>
<tr>
<td>352.</td>
<td>-.5624707</td>
<td>-.535444</td>
<td>-5.67e-18</td>
<td>0</td>
</tr>
</tbody>
</table>

The two set of predictions are fairly close.

Because not all districts contain both urban and rural areas, some of the posterior modes are 0 and some of the posterior means are practically 0. A closer examination of the data reveals that district 3 has no rural areas, and districts 2, 7, and 10 have no urban areas.
Had we imposed an unstructured covariance structure in our model, the estimated posterior modes and posterior means in the cases in question would not be exactly 0 because of the correlation between urban and rural effects. For instance, if a district has no urban areas, it can still yield a nonzero (albeit small) random-effects estimate for a nonexistent urban area because of the correlation with its rural counterpart; see example 1 of [ME] meqrlogit postestimation for details.

Example 2

Continuing with the model from example 1, we can obtain predicted probabilities, and unless we specify the fixedonly option, these predictions will incorporate the estimated subject-specific random effects $\tilde{u}_j$.

```
.predit pr, pr
   (predictions based on fixed effects and posterior means of random effects)
   (using 7 quadrature points)
```

The predicted probabilities for observation $i$ in cluster $j$ are obtained by applying the inverse link function to the linear predictor, $\hat{p}_{ij} = g^{-1}(x_{ij}\hat{\beta} + z_{ij}\tilde{u}_j)$; see Methods and formulas for details. Because we specified the means option, the calculation uses posterior means for $\tilde{u}_j$. You can use the modes option to obtain predictions based on the posterior modes for $\tilde{u}_j$.

```
.predit prm, pr conditional(ebmodes)
   (predictions based on fixed effects and posterior modes of random effects)
```

We can list the two sets of predicted probabilities together with the actual outcome for some district, let's say district 38:

```
.list c_use pr prm if district == 38
```

<table>
<thead>
<tr>
<th>c_use</th>
<th>pr</th>
<th>prm</th>
</tr>
</thead>
<tbody>
<tr>
<td>yes</td>
<td>.5783408</td>
<td>.5780864</td>
</tr>
<tr>
<td>no</td>
<td>.5326623</td>
<td>.5324027</td>
</tr>
<tr>
<td>yes</td>
<td>.6411679</td>
<td>.6409279</td>
</tr>
<tr>
<td>yes</td>
<td>.5326623</td>
<td>.5324027</td>
</tr>
<tr>
<td>yes</td>
<td>.5718783</td>
<td>.5716228</td>
</tr>
<tr>
<td>no</td>
<td>.3447686</td>
<td>.344533</td>
</tr>
<tr>
<td>no</td>
<td>.4507973</td>
<td>.4505391</td>
</tr>
<tr>
<td>no</td>
<td>.1940524</td>
<td>.1976133</td>
</tr>
<tr>
<td>no</td>
<td>.2846738</td>
<td>.2893007</td>
</tr>
<tr>
<td>no</td>
<td>.1264883</td>
<td>.1290078</td>
</tr>
<tr>
<td>no</td>
<td>.206763</td>
<td>.2104961</td>
</tr>
<tr>
<td>no</td>
<td>.202459</td>
<td>.2061346</td>
</tr>
<tr>
<td>no</td>
<td>.206763</td>
<td>.2104961</td>
</tr>
<tr>
<td>no</td>
<td>.1179788</td>
<td>.1203522</td>
</tr>
</tbody>
</table>

The two sets of predicted probabilities are fairly close.

For mixed-effects models with many levels or many random effects, the calculation of the posterior means of random effects or any quantities that are based on the posterior means of random effects may take a long time. This is because we must resort to numerical integration to obtain the posterior means. In contrast, the calculation of the posterior modes of random effects is usually orders of magnitude faster because there is no numerical integration involved. For this reason, empirical modes are often used in practice as an approximation to empirical means. Note that for linear mixed-effects models, the two predictors are the same.
We can compare the observed values with the predicted values by constructing a classification table. Defining success as \( \hat{y}_{ij} = 1 \) if \( \hat{p}_{ij} > 0.5 \) and defining \( \hat{y}_{ij} = 0 \) otherwise, we obtain the following table.

\[
\begin{array}{ccc}
\text{contraception} & \text{Predicted outcome} & \text{Total} \\
\hline
\text{no} & \begin{array}{cc}
991 & 184 \\
84.34 & 15.66 \\
\end{array} & 1,175 \\
\hline
\text{yes} & \begin{array}{cc}
423 & 336 \\
55.73 & 44.27 \\
\end{array} & 759 \\
\hline
\text{Total} & \begin{array}{cc}
1,414 & 520 \\
73.11 & 26.89 \\
\end{array} & 1,934 \\
\end{array}
\]

The model correctly classified 84% of women who did not use contraceptives but only 44% of women who did. In the next example, we will look at some residual diagnostics.

**Technical note**

Out-of-sample predictions are permitted after `meglm`, but if these predictions involve estimated random effects, the integrity of the estimation data must be preserved. If the estimation data have changed since the model was fit, `predict` will be unable to obtain predicted random effects that are appropriate for the fitted model and will give an error. Thus to obtain out-of-sample predictions that contain random-effects terms, be sure that the data for these predictions are in observations that augment the estimation data.

**Example 3**

Continuing our discussion from example 2, here we look at residual diagnostics. `meglm` offers three kinds of predicted residuals for nonlinear responses—Pearson, Anscombe, and deviance. Of the three, Anscombe residuals are designed to be approximately normally distributed; thus we can check for outliers by plotting Anscombe residuals against observation numbers and seeing which residuals are greater than 2 in absolute value.
predict anscombe, anscombe
(predictions based on fixed effects and posterior means of random effects)
(using 7 quadrature points)
generate n = _n
label var n "observation number"
twoway (scatter anscombe n if c_use) (scatter anscombe n if !c_use),
yline(-2 2) legend(off) text(2.5 1400 "contraceptive use")
text(-.1 500 "no contraceptive use")

There seem to be some outliers among residuals that identify women who use contraceptives. We could examine the observations corresponding to the outliers, or we could try fitting a model with perhaps a different covariance structure, which we leave as an exercise.

Example 4

In example 3 of [ME] meglm, we estimated the effects of two treatments on the tobacco and health knowledge (THK) scale score of students in 28 schools. The dependent variable was collapsed into four ordered categories, and we fit a three-level ordinal logistic regression.
. use http://www.stata-press.com/data/r14/tvsfpors, clear
. meologit thk prethk i.cc##i.tv || school: || class:

Fitting fixed-effects model:
Iteration 0: log likelihood = -2212.775
Iteration 1: log likelihood = -2125.509
Iteration 2: log likelihood = -2125.1034
Iteration 3: log likelihood = -2125.1032

Refining starting values:
Grid node 0: log likelihood = -2152.1514

Fitting full model:
(output omitted)

Mixed-effects ologit regression

<table>
<thead>
<tr>
<th>Group Variable</th>
<th>No. of Groups</th>
<th>Observations per Group</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Minimum</td>
<td>Average</td>
</tr>
<tr>
<td>school</td>
<td>28</td>
<td>18</td>
</tr>
<tr>
<td>class</td>
<td>135</td>
<td>1</td>
</tr>
</tbody>
</table>

Integration method: mvaghermite
Integration pts. = 7
Wald chi2(4) = 124.39
Log likelihood = -2114.5881
Prob > chi2 = 0.0000

| thk          | Coef.  | Std. Err. | z     | P>|z| | [95% Conf. Interval] |
|--------------|--------|-----------|-------|------|---------------------|
| prethk       | .4085273 | .039616 | 10.31 | 0.000 | .3308814 - .4861731 |
| 1.cc         | .8844369 | .2099124 | 4.21  | 0.000 | .4730161 - 1.295858 |
| 1.tv         | .236448  | .2049065 | 1.15  | 0.249 | -.1651614 - .6380575 |
| cc#tv        | -.3717699 | .2958887 | -1.26 | 0.209 | -.951701 - .2081612 |
| /cut1        | -.0959459 | .1688988 | -0.57 | 0.570 | -.4269815 - .2350896 |
| /cut2        | 1.177478  | .1704946 | 6.91  | 0.000 | .8433151 - 1.511642 |
| /cut3        | 2.383672  | .1786736 | 13.34 | 0.000 | 2.033478 - 2.733865 |

| school var(_cons) | 0.0448735 | 0.0425387 | 0.0069997 | 0.2876749 |
| school>class var(_cons) | 0.1482157 | 0.0637521 | 0.063792 | 0.3443674 |

LR test vs. ologit model: chi2(2) = 21.03
Prob > chi2 = 0.0000

Note: LR test is conservative and provided only for reference.

Not surprisingly, the level of knowledge before the intervention is a good predictor of the level of knowledge after the intervention. The social resistance classroom curriculum is effective in raising the knowledge score, but the TV intervention and the interaction term are not.

We can rank schools by their institutional effectiveness by plotting the random effects at the school level.
Although there is some variability in the predicted posterior means, we cannot see significant differences among the schools in this example.

Methods and formulas

Continuing the discussion in Methods and formulas of [ME] meglm and using the definitions and formulas defined there, we begin by considering the prediction of the random effects $\mathbf{u}_j$ for the $j$th cluster in a two-level model. Prediction of random effects in multilevel generalized linear models involves assigning values to random effects, and there are many methods for doing so; see Skrondal and Rabe-Hesketh (2009) and Skrondal and Rabe-Hesketh (2004, chap. 7) for a comprehensive review. Stata offers two methods of predicting random effects: empirical Bayes means (also known as posterior means) and empirical Bayes modes (also known as posterior modes). Below we provide more details about the two methods.

Let $\hat{\theta}$ denote the estimated model parameters comprising $\hat{\beta}$ and the unique elements of $\hat{\Sigma}$. Empirical Bayes (EB) predictors of the random effects are the means or modes of the empirical posterior distribution with the parameter estimates $\theta$ replaced with their estimates $\hat{\theta}$. The method is called “empirical” because $\hat{\theta}$ is treated as known. EB combines the prior information about the random effects with the likelihood to obtain the conditional posterior distribution of random effects. Using Bayes’s theorem, the empirical conditional posterior distribution of random effects for cluster $j$ is
The denominator is just the likelihood contribution of the \( j \)th cluster.

EB mean predictions of random effects, \( \tilde{u} \), also known as posterior means, are calculated as

\[
\tilde{u} = \int_{\mathbb{R}^q} u_j \omega(u_j | y_j, X_j, Z_j; \hat{\theta}) \, du_j
\]

where we use the notation \( \tilde{u} \) rather than \( \hat{u} \) to distinguish predicted values from estimates. This multivariate integral is approximated by the mean–variance adaptive Gaussian quadrature; see Methods and formulas of [ME] meglm for details about the quadrature. If you have multiple random effects within a level or random effects across levels, the calculation involves orthogonalizing transformations using the Cholesky transformation because the random effects are no longer independent under the posterior distribution.

In a linear mixed-effects model, the posterior density is multivariate normal, and EB means are also best linear unbiased predictors (BLUPs); see Skrondal and Rabe-Hesketh (2004, 227). In generalized mixed-effects models, the posterior density tends to multivariate normal as cluster size increases. EB modal predictions can be approximated by solving for the mode \( \tilde{u}_j \) in

\[
\frac{\partial}{\partial u_j} \log \omega(\tilde{u}_j | y_j, X_j, Z_j; \hat{\theta}) = 0
\]

Because the denominator in \( \omega(\cdot) \) does not depend on \( u \), we can omit it from the calculation to obtain

\[
\frac{\partial}{\partial u_j} \log \left\{ f(y_j | u_j, X_j, Z_j; \hat{\beta}) \phi(u_j; \hat{\Sigma}) \right\} = \frac{\partial}{\partial u_j} \log f(y_j | u_j, X_j, Z_j; \hat{\beta}) + \frac{\partial}{\partial u_j} \log \phi(u_j; \hat{\Sigma}) = 0
\]

The calculation of EB modes does not require numerical integration, and for that reason they are often used in place of EB means. As the posterior density gets closer to being multivariate normal, EB modes get closer and closer to EB means.

Just like there are many methods of assigning values to the random effects, there exist many methods of calculating standard errors of the predicted random effects; see Skrondal and Rabe-Hesketh (2009) for a comprehensive review.

Stata uses the posterior standard deviation as the standard error of the posterior means predictor of random effects. The EB posterior covariance matrix of the random effects is given by
\[
\text{cov}(\tilde{u}_j | y_j, X_j, Z_j; \hat{\theta}) = \int_{\mathbb{R}^q} (u_j - \tilde{u}_j)(u_j - \tilde{u}_j)' \omega(u_j | y_j, X_j, Z_j; \hat{\theta}) \, du_j
\]

The posterior covariance matrix and the integrals are approximated by the mean–variance adaptive Gaussian quadrature; see Methods and formulas of [ME] meglm for details about the quadrature.

Conditional standard errors for the estimated posterior modes are derived from standard theory of maximum likelihood, which dictates that the asymptotic variance matrix of \(\tilde{\mu}_j\) is the negative inverse of the Hessian, \(g''(\beta, \Sigma, \tilde{\mu}_j)\).

In what follows, we show formulas using the posterior means estimates of random effects \(\tilde{u}_j\), which are used by default or if the means option is specified. If the modes option is specified, \(\tilde{u}_j\) are simply replaced with the posterior modes \(\tilde{\mu}_j\) in these formulas.

For any \(i\)th observation in the \(j\)th cluster in a two-level model, define the linear predictor as

\[
\hat{\eta}_{ij} = x_{ij} \hat{\beta} + z_{ij} \tilde{u}_j
\]

The linear predictor includes the offset or exposure variable if one was specified during estimation, unless the nooffset option is specified. If the fixedonly option is specified, \(\hat{\eta}\) contains the linear predictor for only the fixed portion of the model, \(\hat{\eta}_{ij} = x_{ij} \hat{\beta}\).

The predicted mean, conditional on the random effects \(\tilde{u}_j\), is

\[
\hat{\mu}_{ij} = g^{-1}(\hat{\eta}_{ij})
\]

where \(g^{-1}(\cdot)\) is the inverse link function for \(\mu_{ij} = g^{-1}(\eta_{ij})\) defined as follows for the available links in link:\n
<table>
<thead>
<tr>
<th>link</th>
<th>Inverse link</th>
</tr>
</thead>
<tbody>
<tr>
<td>identity</td>
<td>(\eta_{ij})</td>
</tr>
<tr>
<td>logit</td>
<td>(1/{1 + \exp(-\eta_{ij})})</td>
</tr>
<tr>
<td>probit</td>
<td>(\Phi(\eta_{ij}))</td>
</tr>
<tr>
<td>log</td>
<td>(\exp(\eta_{ij}))</td>
</tr>
<tr>
<td>cloglog</td>
<td>(1 - \exp{-\exp(\eta_{ij})})</td>
</tr>
</tbody>
</table>

By default, random effects and any statistic based on them—mu, fitted, pearson, deviance, anscombe—are calculated using posterior means of random effects unless option modes is specified, in which case the calculations are based on posterior modes of random effects.

Raw residuals are calculated as the difference between the observed and fitted outcomes,

\[
u_{ij} = y_{ij} - \hat{\mu}_{ij}\]

and are only defined for the Gaussian family.

Let \(r_{ij}\) be the number of Bernoulli trials in a binomial model, \(\alpha\) be the conditional overdispersion parameter under the mean parameterization of the negative binomial model, and \(\delta\) be the conditional overdispersion parameter under the constant parameterization of the negative binomial model.
Pearson residuals are raw residuals divided by the square root of the variance function

\[
\nu_{ij}^P = \frac{\nu_{ij}}{\{V(\hat{\mu}_{ij})\}^{1/2}}
\]

where \(V(\hat{\mu}_{ij})\) is the family-specific variance function defined as follows for the available families in `family(family)`:

<table>
<thead>
<tr>
<th>family</th>
<th>Variance function (V(\hat{\mu}_{ij}))</th>
</tr>
</thead>
<tbody>
<tr>
<td>bernoulli</td>
<td>(\hat{\mu}<em>{ij}(1 - \hat{\mu}</em>{ij}))</td>
</tr>
<tr>
<td>binomial</td>
<td>(\hat{\mu}<em>{ij}(1 - \hat{\mu}</em>{ij}/r_{ij}))</td>
</tr>
<tr>
<td>gamma</td>
<td>(\hat{\mu}_{ij}^2)</td>
</tr>
<tr>
<td>gaussian</td>
<td>1</td>
</tr>
<tr>
<td>nbinomial mean</td>
<td>(\hat{\mu}<em>{ij}(1 + \alpha\hat{\mu}</em>{ij}))</td>
</tr>
<tr>
<td>nbinomial constant</td>
<td>(\hat{\mu}_{ij}(1 + \delta))</td>
</tr>
<tr>
<td>ordinal</td>
<td>not defined</td>
</tr>
<tr>
<td>poisson</td>
<td>(\hat{\mu}_{ij})</td>
</tr>
</tbody>
</table>
Deviance residuals are calculated as

$$\nu_{ij}^D = \text{sign}(\nu_{ij}) \sqrt{\hat{d}_{ij}^2}$$

where the squared deviance residual $\hat{d}_{ij}^2$ is defined as follows:

<table>
<thead>
<tr>
<th>family</th>
<th>Squared deviance $\hat{d}_{ij}^2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>bernoulli</td>
<td>$-2\log(1 - \hat{\mu}<em>{ij})$ if $y</em>{ij} = 0$</td>
</tr>
<tr>
<td></td>
<td>$-2\log(\hat{\mu}<em>{ij})$ if $y</em>{ij} = 1$</td>
</tr>
<tr>
<td>binomial</td>
<td>$2r_{ij} \log \left( \frac{r_{ij}}{r_{ij} - \hat{\mu}<em>{ij}} \right)$ if $y</em>{ij} = 0$</td>
</tr>
<tr>
<td></td>
<td>$2y_{ij} \log \left( \frac{y_{ij}}{\hat{\mu}<em>{ij}} \right) + 2(r</em>{ij} - y_{ij}) \log \left( \frac{r_{ij} - y_{ij}}{r_{ij} - \hat{\mu}<em>{ij}} \right)$ if $0 &lt; y</em>{ij} &lt; r_{ij}$</td>
</tr>
<tr>
<td></td>
<td>$2r_{ij} \log \left( \frac{r_{ij}}{\hat{\mu}<em>{ij}} \right)$ if $y</em>{ij} = r_{ij}$</td>
</tr>
<tr>
<td>gamma</td>
<td>$-2 \left{ \log \left( \frac{y_{ij}}{\hat{\mu}<em>{ij}} \right) - \hat{\nu}</em>{ij} \right}$</td>
</tr>
<tr>
<td>gaussian</td>
<td>$\hat{\nu}_{ij}^2$</td>
</tr>
<tr>
<td>nbinomial mean</td>
<td>$2\log \left( 1 + \alpha \hat{\mu}<em>{ij} \right) \alpha$ if $y</em>{ij} = 0$</td>
</tr>
<tr>
<td></td>
<td>$2y_{ij} \log \left( \frac{y_{ij}}{\hat{\mu}<em>{ij}} \right) - \frac{2}{\alpha} \left( 1 + \alpha y</em>{ij} \right) \log \left( \frac{1 + \alpha y_{ij}}{1 + \alpha \hat{\mu}_{ij}} \right)$ otherwise</td>
</tr>
<tr>
<td>nbinomial constant</td>
<td>not defined</td>
</tr>
<tr>
<td>ordinal</td>
<td>not defined</td>
</tr>
<tr>
<td>poisson</td>
<td>$2\hat{\mu}<em>{ij}$ if $y</em>{ij} = 0$</td>
</tr>
<tr>
<td></td>
<td>$2y_{ij} \log \left( \frac{y_{ij}}{\hat{\mu}<em>{ij}} \right) - 2\hat{\nu}</em>{ij}$ otherwise</td>
</tr>
</tbody>
</table>
Anscombe residuals, denoted $\nu_{ij}$, are calculated as follows:

<table>
<thead>
<tr>
<th>family</th>
<th>Anscombe residual $\nu_{ij}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>bernoulli</td>
<td>$\frac{3\left(y_{ij}^{2/3}\mathcal{H}(y_{ij}) - \hat{\mu}<em>{ij}^{2/3}\mathcal{H}(\hat{\mu}</em>{ij})\right)}{2\left(\hat{\mu}<em>{ij} - \hat{\mu}</em>{ij}^2\right)^{1/6}}$</td>
</tr>
<tr>
<td>binomial</td>
<td>$\frac{3\left(y_{ij}^{2/3}\mathcal{H}(y_{ij}/r_{ij}) - \hat{\mu}<em>{ij}^{2/3}\mathcal{H}(\hat{\mu}</em>{ij}/r_{ij})\right)}{2\left(\hat{\mu}<em>{ij} - \hat{\mu}</em>{ij}^2/r_{ij}\right)^{1/6}}$</td>
</tr>
<tr>
<td>gamma</td>
<td>$\frac{3(y_{ij}^{1/3} - \hat{\mu}<em>{ij}^{1/3})}{\hat{\mu}</em>{ij}^{1/3}}$</td>
</tr>
<tr>
<td>gaussian</td>
<td>$\nu_{ij}$</td>
</tr>
<tr>
<td>nbinomial mean</td>
<td>$\frac{\mathcal{H}(-\alpha y_{ij}) - \mathcal{H}(-\alpha \hat{\mu}<em>{ij}) + 1.5(y</em>{ij}^{2/3} - \hat{\mu}<em>{ij}^{2/3})}{(\hat{\mu}</em>{ij} + \alpha \hat{\mu}_{ij}^2)^{1/6}}$</td>
</tr>
<tr>
<td>nbinomial constant</td>
<td>not defined</td>
</tr>
<tr>
<td>ordinal</td>
<td>not defined</td>
</tr>
<tr>
<td>poisson</td>
<td>$\frac{3(y_{ij}^{2/3} - \hat{\mu}<em>{ij}^{2/3})}{2\hat{\mu}</em>{ij}^{1/6}}$</td>
</tr>
</tbody>
</table>

where $\mathcal{H}(t)$ is a specific univariate case of the Hypergeometric2F1 function (Wolfram 1999, 771–772), defined here as $\mathcal{H}(t) = _2F_1(2/3, 1/3, 5/3, t)$.

For a discussion of the general properties of the various residuals, see Hardin and Hilbe (2012, chap. 4).

### References


### Also see

[ME] `meglm` — Multilevel mixed-effects generalized linear model

[U] 20 Estimation and postestimation commands
melogit — Multilevel mixed-effects logistic regression

Description

melogit fits mixed-effects models for binary and binomial responses. The conditional distribution of the response given the random effects is assumed to be Bernoulli, with success probability determined by the logistic cumulative distribution function.

melogit performs optimization using the original metric of variance components. When variance components are near the boundary of the parameter space, you may consider using the meqrlogit command, which provides alternative parameterizations of variance components; see [ME] meqrlogit.

Quick start

Without weights

Two-level logistic regression of y on x with random intercepts by lev2
melogit y x || lev2:

Mixed-effects model adding random coefficients for x
melogit y x || lev2: x

As above, but allow the random effects to be correlated
melogit y x || lev2: x, covariance(unstructured)

Three-level random-intercept model of y on x with lev2 nested within lev3
melogit y x || lev3: || lev2:

Crossed-effects model of y on x with two-way crossed random effects by factors a and b
melogit y x || _all:R.a || b:

With weights

Two-level logistic regression of y on x with random intercepts by lev2 and observation-level frequency weights wvar1
melogit y x [fweight=wvar1] || lev2:

Two-level random-intercept model from a two-stage sampling design with PSUs identified by psu using PSU-level and observation-level sampling weights wvar2 and wvar1, respectively
melogit y x [pweight=wvar1] || psu:, pweight(wvar2)

Add secondary sampling stage with units identified by ssu having weights wvar2 and PSU-level weights wvar3 for a three-level random-intercept model
melogit y x [pw=wvar1] || psu:, pw(wvar3) || ssu:, pw(wvar2)
Same as above, but **svyset** data first

```
svyset psu [pw=wvar3] || ssu, weight(wvar2) || _n, weight(wvar1)
svy: melogit y x || psu: || ssu:
```

**Menu**

Statistics  \(\rightarrow\) Multilevel mixed-effects models  \(\rightarrow\) Logistic regression

**Syntax**

```
melogit  *depvar* fe\_equation \[ || re\_equation \] \[ || re\_equation . . . \] \[ , options \]
```

where the syntax of \(fe\_equation\) is

```
[ indepvars ] [ if ] [ in ] [ weight ] [ , fe\_options ]
```

and the syntax of \(re\_equation\) is one of the following:

- for random coefficients and intercepts
  ```
  levelvar: [ varlist ] [ , re\_options ]
  ```
- for random effects among the values of a factor variable
  ```
  levelvar: R.\*varname
  ```

\(levelvar\) is a variable identifying the group structure for the random effects at that level or is \(_all\) representing one group comprising all observations.

**fe\_options**

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\text{noconstant})</td>
</tr>
<tr>
<td>(\text{offset}(\text{varname}))</td>
</tr>
<tr>
<td>(\text{asis})</td>
</tr>
</tbody>
</table>

**re\_options**

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\text{covariance}(\text{vartype}))</td>
</tr>
<tr>
<td>(\text{noconstant})</td>
</tr>
<tr>
<td>(\text{fweight}(\text{varname}))</td>
</tr>
<tr>
<td>(\text{iweight}(\text{varname}))</td>
</tr>
<tr>
<td>(\text{pweight}(\text{varname}))</td>
</tr>
</tbody>
</table>
### options

<table>
<thead>
<tr>
<th>Description</th>
<th>Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>set binomial trials if data are in binomial form</td>
<td>**binomial(varname</td>
</tr>
<tr>
<td>apply specified linear constraints</td>
<td><strong>constraints(constraints)</strong></td>
</tr>
<tr>
<td>keep collinear variables</td>
<td><strong>collinear</strong></td>
</tr>
<tr>
<td>vcetype may be oim, robust, or cluster clustvar</td>
<td><strong>vce(vcetype)</strong></td>
</tr>
<tr>
<td>set confidence level; default is level(95)</td>
<td><strong>level(#)</strong></td>
</tr>
<tr>
<td>do not display constraints</td>
<td><strong>nocnsreport</strong></td>
</tr>
<tr>
<td>suppress coefficient table</td>
<td><strong>notable</strong></td>
</tr>
<tr>
<td>suppress output header</td>
<td><strong>noheader</strong></td>
</tr>
<tr>
<td>suppress table summarizing groups</td>
<td><strong>nogroup</strong></td>
</tr>
<tr>
<td>do not perform likelihood-ratio test comparing with logistic regression</td>
<td><strong>nolrtest</strong></td>
</tr>
<tr>
<td>control columns and column formats, row spacing, line width, display of omitted variables and base and empty cells, and factor-variable labeling</td>
<td><strong>display_options</strong></td>
</tr>
<tr>
<td>integration method</td>
<td><strong>intmethod(intmethod)</strong></td>
</tr>
<tr>
<td>set the number of integration (quadrature) points for all levels; default is intpoints(7)</td>
<td><strong>intpoints(#)</strong></td>
</tr>
<tr>
<td>control the maximization process; seldom used</td>
<td><strong>maximize_options</strong></td>
</tr>
<tr>
<td>method for obtaining starting values</td>
<td><strong>startvalues(svmethod)</strong></td>
</tr>
<tr>
<td>perform a grid search to improve starting values</td>
<td><strong>startgrid[ gridspec]</strong></td>
</tr>
<tr>
<td>do not fit the model; show starting values instead</td>
<td><strong>noestimate</strong></td>
</tr>
<tr>
<td>use numerical derivative techniques</td>
<td><strong>dnumerical</strong></td>
</tr>
<tr>
<td>display legend instead of statistics</td>
<td><strong>coeflegend</strong></td>
</tr>
<tr>
<td>one unique variance parameter per random effect, all covariances 0; the default unless the R. notation is used</td>
<td><strong>independent</strong></td>
</tr>
<tr>
<td>equal variances for random effects, and one common pairwise covariance</td>
<td><strong>exchangeable</strong></td>
</tr>
<tr>
<td>equal variances for random effects, all covariances 0; the default if the R. notation is used</td>
<td><strong>identity</strong></td>
</tr>
<tr>
<td>all variances and covariances to be distinctly estimated</td>
<td><strong>unstructured</strong></td>
</tr>
<tr>
<td>user-selected variances and covariances constrained to specified values; the remaining variances and covariances unrestricted</td>
<td><strong>fixed(matname)</strong></td>
</tr>
<tr>
<td>user-selected variances and covariances constrained to be equal; the remaining variances and covariances unrestricted</td>
<td><strong>pattern(matname)</strong></td>
</tr>
</tbody>
</table>
### Options

- **noconstant** suppresses the constant (intercept) term and may be specified for the fixed-effects equation and for any of or all the random-effects equations.

- **offset(varname)** specifies that varname be included in the fixed-effects portion of the model with the coefficient constrained to be 1.

- **asis** forces retention of perfect predictor variables and their associated, perfectly predicted observations and may produce instabilities in maximization; see [R] probit.

- **covariance(vartype)** specifies the structure of the covariance matrix for the random effects and may be specified for each random-effects equation. vartype is one of the following: independent, exchangeable, identity, unstructured, fixed(matname), or pattern(matname).

  - **covariance(independent)** covariance structure allows for a distinct variance for each random effect within a random-effects equation and assumes that all covariances are 0. The default is covariance(independent) unless a crossed random-effects model is fit, in which case the default is covariance(identity).

  - **covariance(exchangeable)** structure specifies one common variance for all random effects and one common pairwise covariance.

  - **covariance(identity)** is short for “multiple of the identity”; that is, all variances are equal and all covariances are 0.

  - **covariance(unstructured)** allows for all variances and covariances to be distinct. If an equation consists of p random-effects terms, the unstructured covariance matrix will have p(p + 1)/2 unique parameters.

  - **covariance(fixed(matname))** and **covariance(pattern(matname))** covariance structures provide a convenient way to impose constraints on variances and covariances of random effects. Each specification requires a matname that defines the restrictions placed on variances and covariances. Only elements in the lower triangle of matname are used, and row and column names of matname are ignored. A missing value in matname means that a given element is unrestricted.
In a fixed(matname) covariance structure, (co)variance \((i,j)\) is constrained to equal the value specified in the \(i,j\)th entry of matname. In a pattern(matname) covariance structure, (co)variances \((i,j)\) and \((k,l)\) are constrained to be equal if \(matname[i,j] = matname[k,l]\).

fweight(varname) specifies frequency weights at higher levels in a multilevel model, whereas frequency weights at the first level (the observation level) are specified in the usual manner, for example, \([fw=fwtvar1]\). varname can be any valid Stata variable name, and you can specify fweight() at levels two and higher of a multilevel model. For example, in the two-level model

\[
\text{mecmd fixed.portion} [fw = wt1] || school: ..., fweight(wt2) ...
\]

the variable \(wt1\) would hold the first-level (the observation-level) frequency weights, and \(wt2\) would hold the second-level (the school-level) frequency weights.

iweight(varname) specifies importance weights at higher levels in a multilevel model, whereas importance weights at the first level (the observation level) are specified in the usual manner, for example, \([iw=iwtvar1]\). varname can be any valid Stata variable name, and you can specify iweight() at levels two and higher of a multilevel model. For example, in the two-level model

\[
\text{mecmd fixed.portion} [iw = wt1] || school: ..., iweight(wt2) ...
\]

the variable \(wt1\) would hold the first-level (the observation-level) importance weights, and \(wt2\) would hold the second-level (the school-level) importance weights.

pweight(varname) specifies sampling weights at higher levels in a multilevel model, whereas sampling weights at the first level (the observation level) are specified in the usual manner, for example, \([pw=pwtvar1]\). varname can be any valid Stata variable name, and you can specify pweight() at levels two and higher of a multilevel model. For example, in the two-level model

\[
\text{mecmd fixed.portion} [pw = wt1] || school: ..., pweight(wt2) ...
\]

variable \(wt1\) would hold the first-level (the observation-level) sampling weights, and \(wt2\) would hold the second-level (the school-level) sampling weights.

binomial(varname | #) specifies that the data are in binomial form; that is, \(depvar\) records the number of successes from a series of binomial trials. This number of trials is given either as varname, which allows this number to vary over the observations, or as the constant #. If binomial() is not specified (the default), \(depvar\) is treated as Bernoulli, with any nonzero, nonmissing values indicating positive responses.

constraints(constraints), collinear; see [R] estimation options.

vce(vcetype) specifies the type of standard error reported, which includes types that are derived from asymptotic theory (oim), that are robust to some kinds of misspecification (robust), and that allow for intragroup correlation (cluster clustvar); see [R] vce_option. If vce(robust) is specified, robust variances are clustered at the highest level in the multilevel model.
or reports estimated fixed-effects coefficients transformed to odds ratios, that is, \( \exp(\beta) \) rather than \( \beta \). Standard errors and confidence intervals are similarly transformed. This option affects how results are displayed, not how they are estimated. or may be specified either at estimation or upon replay.

nocnsreport; see [R] estimation options.

notable suppresses the estimation table, either at estimation or upon replay.

nogroup suppresses the display of group summary information (number of groups, average group size, minimum, and maximum) from the output header.

nolrtest prevents melogit from performing a likelihood-ratio test that compares the mixed-effects logistic model with standard (marginal) logistic regression. This option may also be specified upon replay to suppress this test from the output.

display_options: noci, nopvalues, noomitted, vsquish, noemptycells, baselevels, allbaselevels, noflabel, fvwrap(#), fvwrapon(style), cformat(%, fmt), pformat(%, fmt), sformat(%, fmt), and nolstretch; see [R] estimation options.

Integration

intmethod(intmethod) specifies the integration method to be used for the random-effects model. mvaghermite performs mean–variance adaptive Gauss–Hermite quadrature; mcaghermite performs mode-curvature adaptive Gauss–Hermite quadrature; ghermite performs nonadaptive Gauss–Hermite quadrature; and laplace performs the Laplacian approximation, equivalent to mode-curvature adaptive Gaussian quadrature with one integration point.

The default integration method is mvaghermite unless a crossed random-effects model is fit, in which case the default integration method is laplace. The Laplacian approximation has been known to produce biased parameter estimates; however, the bias tends to be more prominent in the estimates of the variance components rather than in the estimates of the fixed effects. For crossed random-effects models, estimation with more than one quadrature point may be prohibitively intensive even for a small number of levels. For this reason, the integration method defaults to the Laplacian approximation. You may override this behavior by specifying a different integration method.

intpoints(#) sets the number of integration points for quadrature. The default is intpoints(7), which means that seven quadrature points are used for each level of random effects. This option is not allowed with intmethod(laplace).

The more integration points, the more accurate the approximation to the log likelihood. However, computation time increases as a function of the number of quadrature points raised to a power equaling the dimension of the random-effects specification. In crossed random-effects models and in models with many levels or many random coefficients, this increase can be substantial.

Maximization

maximize_options: difficult, technique(algorithm_spec), iterate(#), [no]log, trace, gradient, showstep, hessian, showtolerance, tolerance(#), ltolerance(#), nrtolerance(#), nonrtolerance, and from(init_specs); see [R] maximize. Those that require special mention for melogit are listed below.
from() accepts a properly labeled vector of initial values or a list of coefficient names with values. A list of values is not allowed.

The following options are available with melogit but are not shown in the dialog box:

startvalues(svmethod), startgrid[(gridspec)], noestimate, and dnumerical; see [ME] meglm.

coefflegend; see [R] estimation options.

Remarks and examples

For a general introduction to me commands, see [ME] me.

melogit is a convenience command for meglm with a logit link and a bernoulli or binomial family; see [ME] meglm.

Remarks are presented under the following headings:

Introduction
Two-level models
Three-level models

Introduction

Mixed-effects logistic regression is logistic regression containing both fixed effects and random effects. In longitudinal data and panel data, random effects are useful for modeling intracluster correlation; that is, observations in the same cluster are correlated because they share common cluster-level random effects.

Comprehensive treatments of mixed models are provided by, for example, Searle, Casella, and McCulloch (1992); Verbeke and Molenberghs (2000); Raudenbush and Bryk (2002); Demidenko (2004); Hedeker and Gibbons (2006); McCulloch, Searle, and Neuhaus (2008); and Rabe-Hesketh and Skrondal (2012). Guo and Zhao (2000) and Rabe-Hesketh and Skrondal (2012, chap. 10) are good introductory readings on applied multilevel modeling of binary data.

melogit allows for not just one, but many levels of nested clusters of random effects. For example, in a three-level model you can specify random effects for schools and then random effects for classes nested within schools. In this model, the observations (presumably, the students) comprise the first level, the classes comprise the second level, and the schools comprise the third level.

However, for simplicity, for now we consider the two-level model, where for a series of $M$ independent clusters, and conditional on a set of random effects $u_j$,

$$\Pr(y_{ij} = 1|x_{ij}, u_j) = H(x_{ij}\beta + z_{ij}u_j)$$ (1)

for $j = 1, \ldots, M$ clusters, with cluster $j$ consisting of $i = 1, \ldots, n_j$ observations. The responses are the binary-valued $y_{ij}$, and we follow the standard Stata convention of treating $y_{ij} = 1$ if $\text{depvar}_{ij} \neq 0$ and treating $y_{ij} = 0$ otherwise. The $1 \times p$ row vector $x_{ij}$ are the covariates for the fixed effects, analogous to the covariates you would find in a standard logistic regression model, with regression coefficients (fixed effects) $\beta$. For notational convenience here and throughout this manual entry, we suppress the dependence of $y_{ij}$ on $x_{ij}$. 
The $1 \times q$ vector $\mathbf{z}_{ij}$ are the covariates corresponding to the random effects and can be used to represent both random intercepts and random coefficients. For example, in a random-intercept model, $\mathbf{z}_{ij}$ is simply the scalar 1. The random effects $\mathbf{u}_j$ are $M$ realizations from a multivariate normal distribution with mean $\mathbf{0}$ and $q \times q$ variance matrix $\mathbf{\Sigma}$. The random effects are not directly estimated as model parameters but are instead summarized according to the unique elements of $\mathbf{\Sigma}$, known as variance components. One special case of (1) places $\mathbf{z}_{ij} = x_{ij}$, so that all covariate effects are essentially random and distributed as multivariate normal with mean $\beta$ and variance $\mathbf{\Sigma}$.

Finally, because this is logistic regression, $H(\cdot)$ is the logistic cumulative distribution function, which maps the linear predictor to the probability of a success ($y_{ij} = 1$) with $H(v) = \exp(v)/(1 + \exp(v))$.

Model (1) may also be stated in terms of a latent linear response, where only $y_{ij} = I(y^*_ij > 0)$ is observed for the latent

$$y^*_ij = x_{ij}\beta + z_{ij}\mathbf{u}_j + \epsilon_{ij}$$

The errors $\epsilon_{ij}$ are distributed as logistic with mean 0 and variance $\pi^2/3$ and are independent of $\mathbf{u}_j$.

Model (1) is an example of a generalized linear mixed model (GLMM), which generalizes the linear mixed-effects (LME) model to non-Gaussian responses. You can fit LMEs in Stata by using `mixed` and fit GLMMs by using `meglm`. Because of the relationship between LMEs and GLMMs, there is insight to be gained through examination of the linear mixed model. This is especially true for Stata users because the terminology, syntax, options, and output for fitting these types of models are nearly identical. See [ME] mixed and the references therein, particularly in Introduction, for more information.

Log-likelihood calculations for fitting any generalized mixed-effects model require integrating out the random effects. One widely used modern method is to directly estimate the integral required to calculate the log likelihood by Gauss–Hermite quadrature or some variation thereof. Because the log likelihood is computed, this method has the advantage of permitting likelihood-ratio tests for comparing nested models. Also, if done correctly, quadrature approximations can be quite accurate, thus minimizing bias.

`melogit` supports three types of Gauss–Hermite quadrature and the Laplacian approximation method; see Methods and formulas of [ME] meglm for details. The simplest random-effects model you can fit using `melogit` is the two-level model with a random intercept,

$$\Pr(y_{ij} = 1|\mathbf{u}_j) = H(x_{ij}\beta + u_j)$$

This model can also be fit using `xtlogit` with the `re` option; see [XT] xtlogit.

Below we present two short examples of mixed-effects logit regression; refer to [ME] me and [ME] meglm for additional examples including crossed random-effects models.

Two-level models

We begin with a simple application of (1) as a two-level model, because a one-level model, in our terminology, is just standard logistic regression; see [R] logistic.

Example 1

Ng et al. (2006) analyzed a subsample of data from the 1989 Bangladesh fertility survey (Huq and Cleland 1990), which polled 1,934 Bangladeshi women on their use of contraception.
The women sampled were from 60 districts, identified by the variable `district`. Each district contained either urban or rural areas (variable `urban`) or both. The variable `c_use` is the binary response, with a value of 1 indicating contraceptive use. Other covariates include mean-centered `age` and three indicator variables recording number of children. Below we fit a standard logistic regression model amended to have random effects for each district.
melogit — Multilevel mixed-effects logistic regression

. melogit c_use urban age child* || district:

Fitting fixed-effects model:
Iteration 0: log likelihood = -1229.5485
Iteration 1: log likelihood = -1228.5268
Iteration 2: log likelihood = -1228.5263
Iteration 3: log likelihood = -1228.5263

Refining starting values:
Grid node 0: log likelihood = -1219.2681

Fitting full model:
Iteration 0: log likelihood = -1219.2681 (not concave)
Iteration 1: log likelihood = -1207.5978
Iteration 2: log likelihood = -1206.8428
Iteration 3: log likelihood = -1206.8322
Iteration 4: log likelihood = -1206.8322

Mixed-effects logistic regression
Number of obs = 1,934
Group variable: district
Number of groups = 60

Obs per group:
    min = 2
    avg = 32.2
    max = 118

Integration method: mvaghermite
Integration pts. = 7

Wald chi2(5) = 109.60
Log likelihood = -1206.8322
Prob > chi2 = 0.0000

                 c_use |   Coef.   Std. Err.      z    P>|z|     [95% Conf. Interval]
------------- |-----------|------------------|------|---------|------------------|------------------|
urban        |   .732276  |   .1194857       | 6.13 | 0.000   |     .4980888     |     .9664641     |
age          | -0.026498 |   .0078916       | -3.36| 0.001   |    -0.0419654    |    -0.0110309    |
child1       |   1.116001 |   .1580921       | 7.06 | 0.000   |     .8061465     |     1.425856     |
child2       |   1.365895 |   .1746691       | 7.82 | 0.000   |     1.02355      |     1.70824      |
child3       |   1.344031 |   .1796549       | 7.48 | 0.000   |     .9919139     |     1.696148     |
_cons        |  -1.68929  |   .1477591       |-11.43| 0.000   |   -1.978892      |   -1.399687      |

district     |           |                 |      |         |                 |                 |
var(_cons)   |   .215618  |   .0733222       | 2.94 | 0.003   |     .0729238     |     0.358306     |

LR test vs. logistic model: chibar2(01) = 43.39    Prob >= chibar2 = 0.0000

The estimation table reports the fixed effects and the estimated variance components. The fixed effects can be interpreted just as you would the output from logit. You can also specify the or option at estimation or on replay to display the fixed effects as odds ratios instead. If you did display results as odds ratios, you would find urban women to have roughly double the odds of using contraception as that of their rural counterparts. Having any number of children will increase the odds from three-to fourfold when compared with the base category of no children. Contraceptive use also decreases with age.

Underneath the fixed effect, the table shows the estimated variance components. The random-effects equation is labeled district, meaning that these are random effects at the district level. Because we have only one random effect at this level, the table shows only one variance component. The estimate of $\sigma^2_u$ is 0.22 with standard error 0.07.

A likelihood-ratio test comparing the model to ordinary logistic regression is provided and is highly significant for these data.
Three-level models

Two-level models extend naturally to models with three or more levels with nested random effects. By “nested”, we mean that the random effects shared within lower-level subgroups are unique to the upper-level groups. For example, assuming that classroom effects would be nested within schools would be natural, because classrooms are unique to schools.

Example 2

Rabe-Hesketh, Toulopoulou, and Murray (2001) analyzed data from a study measuring the cognitive ability of patients with schizophrenia compared with their relatives and control subjects. Cognitive ability was measured as the successful completion of the “Tower of London”, a computerized task, measured at three levels of difficulty. For all but one of the 226 subjects, there were three measurements (one for each difficulty level). Because patients’ relatives were also tested, a family identifier, family, was also recorded.

. use http://www.stata-press.com/data/r14/towerlondon
   (Tower of London data)
. describe
Contains data from http://www.stata-press.com/data/r14/towerlondon.dta
obs: 677 Tower of London data
vars: 5 31 May 2014 10:41
size: 4,739 (_dta has notes)

<table>
<thead>
<tr>
<th>variable name</th>
<th>storage</th>
<th>display</th>
<th>value</th>
<th>label</th>
<th>variable label</th>
</tr>
</thead>
<tbody>
<tr>
<td>family</td>
<td>int</td>
<td>%8.0g</td>
<td>Family ID</td>
<td></td>
<td></td>
</tr>
<tr>
<td>subject</td>
<td>int</td>
<td>%9.0g</td>
<td>Subject ID</td>
<td></td>
<td></td>
</tr>
<tr>
<td>dtlm</td>
<td>byte</td>
<td>%9.0g</td>
<td>1 = task completed</td>
<td></td>
<td></td>
</tr>
<tr>
<td>difficulty</td>
<td>byte</td>
<td>%9.0g</td>
<td>Level of difficulty: -1, 0, or 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>group</td>
<td>byte</td>
<td>%8.0g</td>
<td>1: controls; 2: relatives; 3: schizophrenics</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Sorted by: family subject

We fit a logistic model with response dtlm, the indicator of cognitive function, and with covariates difficulty and a set of indicator variables for group, with the controls (group==1) being the base category. We allow for random effects due to families and due to subjects within families. We also request to display odds ratios for the fixed-effects parameters.
. melogit dtlm difficulty i.group || family: || subject: , or

Fitting fixed-effects model:
Iteration 0:  log likelihood = -317.35042
Iteration 1:  log likelihood = -313.90007
Iteration 2:  log likelihood = -313.89079
Iteration 3:  log likelihood = -313.89079

Refining starting values:
Grid node 0:  log likelihood = -310.28429

Fitting full model:
Iteration 0:  log likelihood = -310.28429
Iteration 1:  log likelihood = -307.36653
Iteration 2:  log likelihood = -305.19357
Iteration 3:  log likelihood = -305.12073
Iteration 4:  log likelihood = -305.12041
Iteration 5:  log likelihood = -305.12041

Mixed-effects logistic regression  Number of obs   =   677

<table>
<thead>
<tr>
<th>Group Variable</th>
<th>No. of Groups</th>
<th>Observations per Group</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Minimum</td>
<td>Average</td>
</tr>
<tr>
<td>family</td>
<td>118</td>
<td>2</td>
</tr>
<tr>
<td>subject</td>
<td>226</td>
<td>2</td>
</tr>
</tbody>
</table>

Integration method: mvaghermite  Integration pts. = 7
Wald chi2(3)   = 74.90
Log likelihood = -305.12041  Prob > chi2   = 0.0000

| dtlm         | Odds Ratio | Std. Err. | z     | P>|z| | [95% Conf. Interval] |
|--------------|------------|-----------|-------|------|----------------------|
| difficulty   | .1923372   | .037161   | -8.53 | 0.000 | .1317057 .2808806   |
| group 2      | .7798263   | .2763763  | -0.70 | 0.483 | .3893369 1.561961   |
| 3            | .3491318   | .13965    | -2.63 | 0.009 | .15941   .764651    |
| _cons        | .226307    | .0644625  | -5.22 | 0.000 | .1294902 .3955112   |

family
var(_cons)   | .5692105   | .5215654  | .0944757 | 3.429459 |

family>
subject
var(_cons)   | 1.137917   | .6854853  | .3494165 | 3.705762 |

LR test vs. logistic model: chi2(2) = 17.54  Prob > chi2   = 0.0002
Note: LR test is conservative and provided only for reference.

Notes:

1. This is a three-level model with two random-effects equations, separated by ||. The first is a random intercept (constant only) at the family level, and the second is a random intercept at the subject level. The order in which these are specified (from left to right) is significant—melogit assumes that subject is nested within family.
2. The information on groups is now displayed as a table, with one row for each upper level. Among other things, we see that we have 226 subjects from 118 families. You can suppress this table with the nogroup or the noheader option, which will suppress the rest of the header as well.

After adjusting for the random-effects structure, the probability of successful completion of the Tower of London decreases dramatically as the level of difficulty increases. Also, schizophrenics (group==3) tended not to perform as well as the control subjects. Of course, we would make similar conclusions from a standard logistic model fit to the same data, but the odds ratios would differ somewhat.

The above extends to models with more than two levels of nesting in the obvious manner, by adding more random-effects equations, each separated by ||. The order of nesting goes from left to right as the groups go from biggest (highest level) to smallest (lowest level).

Stored results

`melogit` stores the following in `e()`:

Scalars

- `e(N)`: number of observations
- `e(k)`: number of parameters
- `e(k_dv)`: number of dependent variables
- `e(k_eq)`: number of equations in `e(b)`
- `e(k_eq_model)`: number of equations in overall model test
- `e(k_f)`: number of fixed-effects parameters
- `e(k_r)`: number of random-effects parameters
- `e(k_rs)`: number of variances
- `e(k_rc)`: number of covariances
- `e(df_m)`: model degrees of freedom
- `e(ll)`: log likelihood
- `e(N_clust)`: number of clusters
- `e(chi2)`: $\chi^2$
- `e(p)`: significance
- `e(ll_c)`: log likelihood, comparison model
- `e(chi2_c)`: $\chi^2$, comparison model
- `e(df_c)`: degrees of freedom, comparison model
- `e(p_c)`: significance, comparison model
- `e(rank)`: rank of `e(V)`
- `e(ic)`: number of iterations
- `e(rc)`: return code
- `e(converged)`: 1 if converged, 0 otherwise

Macros

- `e(cmd)`: `meglm`
- `e(cmd2)`: `melogit`
- `e(cmdline)`: command as typed
- `e(depvar)`: name of dependent variable
- `e(wtype)`: weight type
- `e(wexp)`: weight expression (first-level weights)
- `e(fweightk)`: fweight variable for kth highest level, if specified
- `e(iweightk)`: iweight variable for kth highest level, if specified
- `e(pweightk)`: pweight variable for kth highest level, if specified
- `e(covariates)`: list of covariates
- `e(ivars)`: grouping variables
- `e(model)`: `logistic`
- `e(title)`: title in estimation output
- `e(link)`: `logit`
- `e(family)`: bernoulli or binomial
Methods and formulas

Model (1) assumes Bernoulli data, a special case of the binomial. Because binomial data are also supported by `melogit` (option `binomial()`) , the methods presented below are in terms of the more general binomial mixed-effects model.

For a two-level binomial model, consider the response \( y_{ij} \) as the number of successes from a series of \( r_{ij} \) Bernoulli trials (replications). For cluster \( j, j = 1, \ldots, M \), the conditional distribution of \( y_{j} = (y_{j1}, \ldots, y_{jn_j})' \), given a set of cluster-level random effects \( u_j \), is

\[
\begin{align*}
  f(y_j | u_j) &= \prod_{i=1}^{n_j} \left[ \frac{r_{ij}}{y_{ij}} \right]^{y_{ij}} \left[ 1 - \frac{r_{ij}}{y_{ij}} \right]^{r_{ij} - y_{ij}} \\
  &= \exp \left( \sum_{i=1}^{n_j} y_{ij} \eta_{ij} - r_{ij} \log \{ 1 + \exp(\eta_{ij}) \} + \log \left( \frac{r_{ij}}{y_{ij}} \right) \right)
\end{align*}
\]

for \( \eta_{ij} = x_{ij} \beta + z_{ij} u_j + \text{offset}_{ij} \) and \( H(v) = \exp(v) / \{ 1 + \exp(v) \} \).
Defining $r_j = (r_{j1}, \ldots, r_{jn_j})'$ and

$$c(y_j, r_j) = \sum_{i=1}^{n_j} \log \left( \frac{r_{ij}}{y_{ij}} \right)$$

where $c(y_j, r_j)$ does not depend on the model parameters, we can express the above compactly in matrix notation,

$$f(y_j | u_j) = \exp \left[ y_j' \eta_j - r_j' \log \left\{ 1 + \exp(\eta_j) \right\} + c(y_j, r_j) \right]$$

where $\eta_j$ is formed by stacking the row vectors $\eta_{ij}$. We extend the definitions of the functions $\log(\cdot)$ and $\exp(\cdot)$ to be vector functions where necessary.

Because the prior distribution of $u_j$ is multivariate normal with mean 0 and $q \times q$ variance matrix $\Sigma$, the likelihood contribution for the $j$th cluster is obtained by integrating $u_j$ out of the joint density $f(y_j, u_j)$,

$$L_j(\beta, \Sigma) = (2\pi)^{-q/2} |\Sigma|^{-1/2} \int f(y_j | u_j) \exp \left( -u_j' \Sigma^{-1} u_j / 2 \right) du_j$$

$$= \exp \{ c(y_j, r_j) \} (2\pi)^{-q/2} |\Sigma|^{-1/2} \int \exp \{ h(\beta, \Sigma, u_j) \} du_j$$

where

$$h(\beta, \Sigma, u_j) = y_j' \eta_j - r_j' \log \left\{ 1 + \exp(\eta_j) \right\} - u_j' \Sigma^{-1} u_j / 2$$

and for convenience, in the arguments of $h(\cdot)$ we suppress the dependence on the observable data $(y_j, r_j, X_j, Z_j)$.

The integration in (2) has no closed form and thus must be approximated. melogit offers four approximation methods: mean–variance adaptive Gauss–Hermite quadrature (default unless a crossed random-effects model is fit), mode-curvature adaptive Gauss–Hermite quadrature, nonadaptive Gauss–Hermite quadrature, and Laplacian approximation (default for crossed random-effects models).

The Laplacian approximation is based on a second-order Taylor expansion of $h(\beta, \Sigma, u_j)$ about the value of $u_j$ that maximizes it; see Methods and formulas in [ME] meglm for details.

Gaussian quadrature relies on transforming the multivariate integral in (2) into a set of nested univariate integrals. Each univariate integral can then be evaluated using a form of Gaussian quadrature; see Methods and formulas in [ME] meglm for details.

The log likelihood for the entire dataset is simply the sum of the contributions of the $M$ individual clusters, namely, $L(\beta, \Sigma) = \sum_{j=1}^{M} L_j(\beta, \Sigma)$.

Maximization of $L(\beta, \Sigma)$ is performed with respect to $(\beta, \sigma^2)$, where $\sigma^2$ is a vector comprising the unique elements of $\Sigma$. Parameter estimates are stored in $e(b)$ as $(\hat{\beta}, \hat{\sigma}^2)$, with the corresponding variance–covariance matrix stored in $e(V)$.

melogit supports multilevel weights and survey data; see Methods and formulas in [ME] meglm for details.
References


Also see

[ME] melogit postestimation — Postestimation tools for melogit

[ME] mecloglog — Multilevel mixed-effects complementary log-log regression

[ME] meprobit — Multilevel mixed-effects probit regression

[ME] meqrlogit — Multilevel mixed-effects logistic regression (QR decomposition)

[ME] me — Introduction to multilevel mixed-effects models

[SEM] intro 5 — Tour of models (*Multilevel mixed-effects models*)

[SVY] svy estimation — Estimation commands for survey data

[XT] xtlogit — Fixed-effects, random-effects, and population-averaged logit models

[U] 20 Estimation and postestimation commands
Postestimation commands

The following postestimation commands are of special interest after `melogit`:

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>estat group</code></td>
<td>summarize the composition of the nested groups</td>
</tr>
<tr>
<td><code>estat icc</code></td>
<td>estimate intraclass correlations</td>
</tr>
</tbody>
</table>

The following standard postestimation commands are also available:

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>contrast</code></td>
<td>contrasts and ANOVA-style joint tests of estimates</td>
</tr>
<tr>
<td><code>estat ic</code></td>
<td>Akaike’s and Schwarz’s Bayesian information criteria (AIC and BIC)</td>
</tr>
<tr>
<td><code>estat summarize</code></td>
<td>summary statistics for the estimation sample</td>
</tr>
<tr>
<td><code>estat vce</code></td>
<td>variance–covariance matrix of the estimators (VCE)</td>
</tr>
<tr>
<td><code>estat (svy)</code></td>
<td>postestimation statistics for survey data</td>
</tr>
<tr>
<td><code>estimates</code></td>
<td>cataloging estimation results</td>
</tr>
<tr>
<td><code>*hausman</code></td>
<td>Hausman’s specification test</td>
</tr>
<tr>
<td><code>lincom</code></td>
<td>point estimates, standard errors, testing, and inference for linear combinations of coefficients</td>
</tr>
<tr>
<td><code>*lrtest</code></td>
<td>likelihood-ratio test</td>
</tr>
<tr>
<td><code>margins</code></td>
<td>marginal means, predictive margins, marginal effects, and average marginal effects</td>
</tr>
<tr>
<td><code>marginsplot</code></td>
<td>graph the results from margins (profile plots, interaction plots, etc.)</td>
</tr>
<tr>
<td><code>nlcom</code></td>
<td>point estimates, standard errors, testing, and inference for nonlinear combinations of coefficients</td>
</tr>
<tr>
<td><code>predict</code></td>
<td>predictions, residuals, influence statistics, and other diagnostic measures</td>
</tr>
<tr>
<td><code>predictnl</code></td>
<td>point estimates, standard errors, testing, and inference for generalized predictions</td>
</tr>
<tr>
<td><code>pwcompare</code></td>
<td>pairwise comparisons of estimates</td>
</tr>
<tr>
<td><code>test</code></td>
<td>Wald tests of simple and composite linear hypotheses</td>
</tr>
<tr>
<td><code>testnl</code></td>
<td>Wald tests of nonlinear hypotheses</td>
</tr>
</tbody>
</table>

* `hausman` and `lrtest` are not appropriate with `svy` estimation results.
predict

Description for predict

predict creates a new variable containing predictions such as mean responses; linear predictions; density and distribution functions; standard errors; and Pearson, deviance, and Anscombe residuals.

Menu for predict

Statistics > Postestimation

Syntax for predict

Syntax for obtaining predictions of the outcome and other statistics

```
predict [type] newvarsspec [if] [in] [, statistic options]
```

Syntax for obtaining estimated random effects and their standard errors

```
predict [type] newvarsspec [if] [in], reffects [re_options]
```

Syntax for obtaining ML scores

```
predict [type] newvarsspec [if] [in], scores
```

newvarsspec is stub* or newvarlist.

<table>
<thead>
<tr>
<th>statistic</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mu</td>
<td>mean response; the default</td>
</tr>
<tr>
<td>eta</td>
<td>fitted linear predictor</td>
</tr>
<tr>
<td>xb</td>
<td>linear predictor for the fixed portion of the model only</td>
</tr>
<tr>
<td>stdp</td>
<td>standard error of the fixed-portion linear prediction</td>
</tr>
<tr>
<td>density</td>
<td>predicted density function</td>
</tr>
<tr>
<td>distribution</td>
<td>predicted distribution function</td>
</tr>
<tr>
<td>pearson</td>
<td>Pearson residuals</td>
</tr>
<tr>
<td>deviance</td>
<td>deviance residuals</td>
</tr>
<tr>
<td>anscombe</td>
<td>Anscombe residuals</td>
</tr>
</tbody>
</table>

These statistics are available both in and out of sample; type predict ... if e(sample) ... if wanted only for the estimation sample.
### options

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Main</strong></td>
</tr>
<tr>
<td><strong>conditional(ctype)</strong></td>
</tr>
<tr>
<td><strong>marginal</strong></td>
</tr>
<tr>
<td><strong>nooffset</strong></td>
</tr>
<tr>
<td><strong>Integration</strong></td>
</tr>
<tr>
<td><strong>int_options</strong></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ctype</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ebmeans</strong></td>
</tr>
<tr>
<td><strong>ebmodes</strong></td>
</tr>
<tr>
<td><strong>fixedonly</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>re_options</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Main</strong></td>
</tr>
<tr>
<td><strong>ebmeans</strong></td>
</tr>
<tr>
<td><strong>ebmodes</strong></td>
</tr>
<tr>
<td>*<em>reses(stub</em></td>
</tr>
<tr>
<td><strong>Integration</strong></td>
</tr>
<tr>
<td><strong>int_options</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>int_options</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Main</strong></td>
</tr>
<tr>
<td><strong>intpoints(#)</strong></td>
</tr>
<tr>
<td><strong>iterate(#)</strong></td>
</tr>
<tr>
<td><strong>tolerance(#)</strong></td>
</tr>
<tr>
<td><strong>Integration</strong></td>
</tr>
<tr>
<td><strong>int_options</strong></td>
</tr>
</tbody>
</table>

### Options for predict

- **mu**, the default, calculates the predicted mean, that is, the probability of a positive outcome.
- **eta, xb, stdp, density, distribution, pearson, deviance, anscombe, scores, conditional(), marginal, and nooffset**; see [ME] `meglm postestimation`.
- **reffects, ebmeans, ebmodes, and reses()**; see [ME] `meglm postestimation`.
- **intpoints(), iterate(), and tolerance()**; see [ME] `meglm postestimation`. 
margins

Description for margins

margins estimates margins of response for mean responses and linear predictions.

Menu for margins

Statistics > Postestimation

Syntax for margins

margins [marginlist] [, options]
margins [marginlist], predict(statistic ...) [predict(statistic ...) ...] [options]

<table>
<thead>
<tr>
<th>statistic</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mu</td>
<td>mean response; the default</td>
</tr>
<tr>
<td>eta</td>
<td>fitted linear predictor</td>
</tr>
<tr>
<td>xb</td>
<td>linear predictor for the fixed portion of the model only</td>
</tr>
<tr>
<td>stdp</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>density</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>distribution</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>pearson</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>deviance</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>anscombe</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>reffects</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>scores</td>
<td>not allowed with margins</td>
</tr>
</tbody>
</table>

Options conditional(ebmeans) and conditional(ebmodes) are not allowed with margins. Option marginal is assumed where applicable if conditional(fixedonly) is not specified.

Statistics not allowed with margins are functions of stochastic quantities other than e(b). For the full syntax, see [R] margins.
estat

Description for estat

estat group reports the number of groups and minimum, average, and maximum group sizes for each level of the model. Model levels are identified by the corresponding group variable in the data. Because groups are treated as nested, the information in this summary may differ from what you would get if you used the tabulate command on each group variable individually.

estat icc displays the intraclass correlation for pairs of latent linear responses at each nested level of the model. Intraclass correlations are available for random-intercept models or for random-coefficient models conditional on random-effects covariates being equal to 0. They are not available for crossed-effects models.

Menu for estat

Statistics > Postestimation

Syntax for estat

Summarize the composition of the nested groups

estat group

Estimate intraclass correlations

estat icc [, level(#)]

Option for estat icc

level(#) specifies the confidence level, as a percentage, for confidence intervals. The default is level(95) or as set by set level; see [U] 20.7 Specifying the width of confidence intervals.

Remarks and examples

Various predictions, statistics, and diagnostic measures are available after fitting a logistic mixed-effects model with melogit. Here we show a short example of predicted probabilities and predicted random effects; refer to [ME] meglm postestimation for additional examples.

Example 1

In example 2 of [ME] melogit, we analyzed the cognitive ability (dtlm) of patients with schizophrenia compared with their relatives and control subjects, by using a three-level logistic model with random effects at the family and subject levels. Cognitive ability was measured as the successful completion of the “Tower of London”, a computerized task, measured at three levels of difficulty.

use http://www.stata-press.com/data/r14/towerlondon
(Tower of London data)
melogit dtlm difficulty i.group || family: || subject: , or
(output omitted)
We obtain predicted probabilities based on the contribution of both fixed effects and random effects by typing

```
.predict pr
```

(predictions based on fixed effects and posterior means of random effects)

(option \texttt{mu} assumed)

(using 7 quadrature points)

As the note says, the predicted values are based on the posterior means of random effects. You can use the \texttt{modes} option to obtain predictions based on the posterior modes of random effects.

We obtain predictions of the posterior means themselves by typing

```
.predict re*, reffects
```

(calculating posterior means of random effects)

(using 7 quadrature points)

Because we have one random effect at the family level and another random effect at the subject level, Stata saved the predicted posterior means in the variables \texttt{re1} and \texttt{re2}, respectively. If you are not sure which prediction corresponds to which level, you can use the \texttt{describe} command to show the variable labels.

Here we list the data for family 16:

```
.list family subject dtlm pr re1 re2 if family==16, sepby(subject)
```

<table>
<thead>
<tr>
<th>family</th>
<th>subject</th>
<th>dtlm</th>
<th>pr</th>
<th>re1</th>
<th>re2</th>
</tr>
</thead>
<tbody>
<tr>
<td>208.</td>
<td>16</td>
<td>5</td>
<td>1</td>
<td>.5337746</td>
<td>.8045555</td>
</tr>
<tr>
<td>209.</td>
<td>16</td>
<td>5</td>
<td>0</td>
<td>.1804649</td>
<td>.8045555</td>
</tr>
<tr>
<td>210.</td>
<td>16</td>
<td>5</td>
<td>0</td>
<td>.0406325</td>
<td>.8045555</td>
</tr>
<tr>
<td>211.</td>
<td>16</td>
<td>34</td>
<td>1</td>
<td>.8956181</td>
<td>.8045555</td>
</tr>
<tr>
<td>212.</td>
<td>16</td>
<td>34</td>
<td>1</td>
<td>.6226832</td>
<td>.8045555</td>
</tr>
<tr>
<td>213.</td>
<td>16</td>
<td>34</td>
<td>1</td>
<td>.2409364</td>
<td>.8045555</td>
</tr>
<tr>
<td>214.</td>
<td>16</td>
<td>35</td>
<td>0</td>
<td>.6627467</td>
<td>.8045555</td>
</tr>
<tr>
<td>215.</td>
<td>16</td>
<td>35</td>
<td>1</td>
<td>.2742936</td>
<td>.8045555</td>
</tr>
<tr>
<td>216.</td>
<td>16</td>
<td>35</td>
<td>0</td>
<td>.0677705</td>
<td>.8045555</td>
</tr>
</tbody>
</table>

The predicted random effects at the family level (\texttt{re1}) are the same for all members of the family. Similarly, the predicted random effects at the individual level (\texttt{re2}) are constant within each individual.

The predicted probabilities (\texttt{pr}) for this family seem to be in fair agreement with the response (\texttt{dtlm}) based on a cutoff of 0.5.

We can use \texttt{estat icc} to estimate the residual intraclass correlation (conditional on the difficulty level and the individual's category) between the latent responses of subjects within the same family or between the latent responses of the same subject and family:

```
.estat icc
```

Residual intraclass correlation

<table>
<thead>
<tr>
<th>Level</th>
<th>ICC</th>
<th>Std. Err.</th>
<th>[95% Conf. Interval]</th>
</tr>
</thead>
<tbody>
<tr>
<td>family</td>
<td>.1139105</td>
<td>.0997727</td>
<td>.0181851 .4715289</td>
</tr>
<tr>
<td>subject</td>
<td>family</td>
<td>.3416307</td>
<td>.0889471</td>
</tr>
</tbody>
</table>

\texttt{estat icc} reports two intraclass correlations for this three-level nested model. The first is the level-3 intraclass correlation at the family level, the correlation between latent measurements of the
cognitive ability in the same family. The second is the level-2 intraclass correlation at the subject-within-family level, the correlation between the latent measurements of cognitive ability in the same subject and family.

There is not a strong correlation between individual realizations of the latent response, even within the same subject.

Stored results

`estat icc` stores the following in r():

Scalars

- `r(icc#)` level-# intraclass correlation
- `r(se#)` standard errors of level-# intraclass correlation
- `r(level)` confidence level of confidence intervals

Macros

- `r(label#)` label for level #

Matrices

- `r(ci#)` vector of confidence intervals (lower and upper) for level-# intraclass correlation

For a $G$-level nested model, # can be any integer between 2 and $G$.

Methods and formulas

Methods and formulas are presented under the following headings:

Prediction

Intraclass correlations

Prediction

Methods and formulas for predicting random effects and other statistics are given in Methods and formulas of [ME] `meglm postestimation`.

Intraclass correlations

Consider a simple, two-level random-intercept model, stated in terms of a latent linear response, where only $y_{ij} = I(y_{ij}^* > 0)$ is observed for the latent variable,

$$y_{ij}^* = \beta + u_{ij}^{(2)} + \epsilon_{ij}^{(1)}$$

with $i = 1, \ldots, n_j$ and level-2 groups $j = 1, \ldots, M$. Here $\beta$ is an unknown fixed intercept, $u_{ij}^{(2)}$ is a level-2 random intercept, and $\epsilon_{ij}^{(1)}$ is a level-1 error term. Errors are assumed to be logistic with mean 0 and variance $\sigma_1^2 = \pi^2/3$; random intercepts are assumed to be normally distributed with mean 0 and variance $\sigma_2^2$ and to be independent of error terms.

The intraclass correlation for this model is

$$\rho = \text{Corr}(y_{ij}^*, y_{ij'}^*) = \frac{\sigma_2^2}{\pi^2/3 + \sigma_2^2}$$

It corresponds to the correlation between the latent responses $i$ and $i'$ from the same group $j$. 
Now consider a three-level nested random-intercept model,

\[ y_{ijk}^* = \beta + u_{jk}^{(2)} + u_{k}^{(3)} + \epsilon_{ijk}^{(1)} \]

for measurements \( i = 1, \ldots, n_{jk} \) and level-2 groups \( j = 1, \ldots, M_{1k} \) nested within level-3 groups \( k = 1, \ldots, M_2 \). Here \( u_{jk}^{(2)} \) is a level-2 random intercept, \( u_{k}^{(3)} \) is a level-3 random intercept, and \( \epsilon_{ijk}^{(1)} \) is a level-1 error term. The error terms have a logistic distribution with mean 0 and variance \( \sigma_1^2 = \pi^2 / 3 \). The random intercepts are assumed to be normally distributed with mean 0 and variances \( \sigma_2^2 \) and \( \sigma_3^2 \), respectively, and to be mutually independent. The error terms are also independent of the random intercepts.

We can consider two types of intraclass correlations for this model. We will refer to them as level-2 and level-3 intraclass correlations. The level-3 intraclass correlation is

\[ \rho^{(3)} = \text{Corr}(y_{ijk}^*, y_{i'j'k}^*) = \frac{\sigma_3^2}{\pi^2 / 3 + \sigma_2^2 + \sigma_3^2} \]

This is the correlation between latent responses \( i \) and \( i' \) from the same level-3 group \( k \) and from different level-2 groups \( j \) and \( j' \).

The level-2 intraclass correlation is

\[ \rho^{(2)} = \text{Corr}(y_{ijk}^*, y_{ij'k}^*) = \frac{\sigma_2^2 + \sigma_3^2}{\pi^2 / 3 + \sigma_2^2 + \sigma_3^2} \]

This is the correlation between latent responses \( i \) and \( i' \) from the same level-3 group \( k \) and level-2 group \( j \). (Note that level-1 intraclass correlation is undefined.)

More generally, for a \( G \)-level nested random-intercept model, the \( g \)-level intraclass correlation is defined as

\[ \rho^{(g)} = \frac{\sum_{l=g}^{G} \sigma_l^2}{\pi^2 / 3 + \sum_{l=2}^{G} \sigma_l^2} \]

The above formulas also apply in the presence of fixed-effects covariates \( X \) in a random-effects model. In this case, intraclass correlations are conditional on fixed-effects covariates and are referred to as residual intraclass correlations. \texttt{estat icc} also uses the same formulas to compute intraclass correlations for random-coefficient models, assuming 0 baseline values for the random-effects covariates, and labels them as conditional intraclass correlations.

Intraclass correlations will always fall in \([0,1]\) because variance components are nonnegative. To accommodate the range of an intraclass correlation, we use the logit transformation to obtain confidence intervals. We use the delta method to estimate the standard errors of the intraclass correlations.
Let \( \hat{\rho}^{(g)} \) be a point estimate of the intraclass correlation and \( \hat{SE}(\hat{\rho}^{(g)}) \) be its standard error. The \((1 - \alpha) \times 100\%\) confidence interval for \( \text{logit}(\rho^{(g)}) \) is

\[
\text{logit}(\hat{\rho}^{(g)}) \pm z_{\alpha/2} \frac{\hat{SE}(\hat{\rho}^{(g)})}{\hat{\rho}^{(g)}(1 - \hat{\rho}^{(g)})}
\]

where \( z_{\alpha/2} \) is the \( 1 - \alpha/2 \) quantile of the standard normal distribution and \( \text{logit}(x) = \ln\{x/(1-x)\} \). Let \( k_u \) be the upper endpoint of this interval, and let \( k_l \) be the lower. The \((1 - \alpha) \times 100\%\) confidence interval for \( \rho^{(g)} \) is then given by

\[
\left( \frac{1}{1 + e^{-k_l}}, \frac{1}{1 + e^{-k_u}} \right)
\]

Also see

[ME] melogit — Multilevel mixed-effects logistic regression
[ME] meglm postestimation — Postestimation tools for meglm
[U] 20 Estimation and postestimation commands
menbreg — Multilevel mixed-effects negative binomial regression

Description

menbreg fits mixed-effects negative binomial models to count data. The conditional distribution of the response given random effects is assumed to follow a Poisson-like process, except that the variation is greater than that of a true Poisson process.

Quick start

Mixed-effects negative binomial regression of y on x with random intercepts by v1
  menbreg y x || v1:

Add evar measuring exposure
  menbreg y x, exposure(evar) || v1:

As above, but report incidence-rate ratios instead of coefficients
  menbreg y x, exposure(evar) || v1:, irr

Add random coefficients for x
  menbreg y x, exposure(evar) || v1: x, irr

Three-level random-intercept model of y on x with v1 nested within v2
  menbreg y x || v2: || v1:

Menu

Statistics  >  Multilevel mixed-effects models  >  Negative binomial regression
Syntax

```bash
menbreg depvar fe_equation [ || re_equation] [ || re_equation ...] [ , options]
```

where the syntax of `fe_equation` is

```
[ indepvars] [ if] [ in] [ weight] [ , fe_options]
```

and the syntax of `re_equation` is one of the following:

for random coefficients and intercepts

```
levelvar: [ varlist] , re_options
```

for random effects among the values of a factor variable

```
levelvar: R.varname
```

`levelvar` is a variable identifying the group structure for the random effects at that level or is `_all` representing one group comprising all observations.

---

### fe_options

<table>
<thead>
<tr>
<th>Model</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>noconstant</code></td>
<td>suppress the constant term from the fixed-effects equation</td>
</tr>
<tr>
<td><code>exposure(varname_e)</code></td>
<td>include ln(varname_e) in model with coefficient constrained to 1</td>
</tr>
<tr>
<td><code>offset(varname_o)</code></td>
<td>include varname_o in model with coefficient constrained to 1</td>
</tr>
</tbody>
</table>

---

### re_options

<table>
<thead>
<tr>
<th>Model</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>covariance(vartype)</code></td>
<td>variance–covariance structure of the random effects</td>
</tr>
<tr>
<td><code>noconstant</code></td>
<td>suppress constant term from the random-effects equation</td>
</tr>
<tr>
<td><code>fweight(varname)</code></td>
<td>frequency weights at higher levels</td>
</tr>
<tr>
<td><code>iweight(varname)</code></td>
<td>importance weights at higher levels</td>
</tr>
<tr>
<td><code>pweight(varname)</code></td>
<td>sampling weights at higher levels</td>
</tr>
<tr>
<td>options</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>Model</strong></td>
<td></td>
</tr>
<tr>
<td><code>dispersion(dispersion)</code></td>
<td>parameterization of the conditional overdispersion; ( dispersion ) may be mean (default) or constant</td>
</tr>
<tr>
<td><code>constraints(constraints)</code></td>
<td>apply specified linear constraints</td>
</tr>
<tr>
<td><code>collinear</code></td>
<td>keep collinear variables</td>
</tr>
<tr>
<td><strong>SE/Robust</strong></td>
<td></td>
</tr>
<tr>
<td><code>vce(vcetype)</code></td>
<td><code>vcetype</code> may be <code>oim</code>, <code>robust</code>, or <code>cluster clustvar</code></td>
</tr>
<tr>
<td><strong>Reporting</strong></td>
<td></td>
</tr>
<tr>
<td><code>level(#)</code></td>
<td>set confidence level; default is <code>level(95)</code></td>
</tr>
<tr>
<td><code>irr</code></td>
<td>report fixed-effects coefficients as incidence-rate ratios</td>
</tr>
<tr>
<td><code>nocnsreport</code></td>
<td>do not display constraints</td>
</tr>
<tr>
<td><code>notable</code></td>
<td>suppress coefficient table</td>
</tr>
<tr>
<td><code>noheader</code></td>
<td>suppress output header</td>
</tr>
<tr>
<td><code>nogroup</code></td>
<td>suppress table summarizing groups</td>
</tr>
<tr>
<td><code>nlrtest</code></td>
<td>do not perform likelihood-ratio test comparing with negative binomial regression</td>
</tr>
<tr>
<td><code>display_options</code></td>
<td>control columns and column formats, row spacing, line width, display of omitted variables and base and empty cells, and factor-variable labeling</td>
</tr>
<tr>
<td><strong>Integration</strong></td>
<td></td>
</tr>
<tr>
<td><code>intmethod(intmethod)</code></td>
<td>integration method</td>
</tr>
<tr>
<td><code>intpoints(#)</code></td>
<td>set the number of integration (quadrature) points for all levels; default is <code>intpoints(7)</code></td>
</tr>
<tr>
<td><strong>Maximization</strong></td>
<td></td>
</tr>
<tr>
<td><code>maximize_options</code></td>
<td>control the maximization process; seldom used</td>
</tr>
<tr>
<td><code>startvalues(svmethod)</code></td>
<td>method for obtaining starting values</td>
</tr>
<tr>
<td><code>startgrid</code></td>
<td>perform a grid search to improve starting values</td>
</tr>
<tr>
<td><code>noestimate</code></td>
<td>do not fit the model; show starting values instead</td>
</tr>
<tr>
<td><code>dnnumerical</code></td>
<td>use numerical derivative techniques</td>
</tr>
<tr>
<td><code>coeflegend</code></td>
<td>display legend instead of statistics</td>
</tr>
<tr>
<td><strong>vartype</strong></td>
<td></td>
</tr>
<tr>
<td><code>independent</code></td>
<td>one unique variance parameter per random effect, all covariances 0; the default unless the R. notation is used</td>
</tr>
<tr>
<td><code>exchangeable</code></td>
<td>equal variances for random effects, and one common pairwise covariance</td>
</tr>
<tr>
<td><code>identity</code></td>
<td>equal variances for random effects, all covariances 0; the default if the R. notation is used</td>
</tr>
<tr>
<td><code>unstructured</code></td>
<td>all variances and covariances to be distinctly estimated</td>
</tr>
<tr>
<td><code>fixed(matname)</code></td>
<td>user-selected variances and covariances constrained to specified values; the remaining variances and covariances unrestricted</td>
</tr>
<tr>
<td><code>pattern(matname)</code></td>
<td>user-selected variances and covariances constrained to be equal; the remaining variances and covariances unrestricted</td>
</tr>
</tbody>
</table>
**intmethod**

<table>
<thead>
<tr>
<th>intmethod</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mvaghermite</td>
<td>mean–variance adaptive Gauss–Hermite quadrature; the default unless a crossed random-effects model is fit</td>
</tr>
<tr>
<td>mcaghermite</td>
<td>mode-curvature adaptive Gauss–Hermite quadrature</td>
</tr>
<tr>
<td>ghermite</td>
<td>nonadaptive Gauss–Hermite quadrature</td>
</tr>
<tr>
<td>laplace</td>
<td>Laplacian approximation; the default for crossed random-effects models</td>
</tr>
</tbody>
</table>

**Options**

- `noconstant` suppresses the constant (intercept) term and may be specified for the fixed-effects equation and for any of or all the random-effects equations.

- `exposure(varname_e)` specifies a variable that reflects the amount of exposure over which the `depvar` events were observed for each observation; ln(varname\_e) is included in the fixed-effects portion of the model with the coefficient constrained to be 1.

- `offset(varname_o)` specifies that varname\_o be included in the fixed-effects portion of the model with the coefficient constrained to be 1.

- `covariance(vartype)` specifies the structure of the covariance matrix for the random effects and may be specified for each random-effects equation. vartype is one of the following: independent, exchangeable, identity, unstructured, fixed(matname), or pattern(matname).

  - covariance(independent) covariance structure allows for a distinct variance for each random effect within a random-effects equation and assumes that all covariances are 0. The default is covariance(independent) unless a crossed random-effects model is fit, in which case the default is covariance(identity).
  
  - covariance(exchangeable) structure specifies one common variance for all random effects and one common pairwise covariance.
  
  - covariance(identity) is short for “multiple of the identity”; that is, all variances are equal and all covariances are 0.
  
  - covariance(unstructured) allows for all variances and covariances to be distinct. If an equation consists of p random-effects terms, the unstructured covariance matrix will have p(p + 1)/2 unique parameters.
  
  - covariance(fixed(matname)) and covariance(pattern(matname)) covariance structures provide a convenient way to impose constraints on variances and covariances of random effects. Each specification requires a matname that defines the restrictions placed on variances and covariances. Only elements in the lower triangle of matname are used, and row and column names...
of matname are ignored. A missing value in matname means that a given element is unrestricted. In a fixed(matname) covariance structure, (co)variance \((i,j)\) is constrained to equal the value specified in the \(i,j\)th entry of matname. In a pattern(matname) covariance structure, (co)variances \((i,j)\) and \((k,l)\) are constrained to be equal if matname\([i,j]\) = matname\([k,l]\).

fweight(varname) specifies frequency weights at higher levels in a multilevel model, whereas frequency weights at the first level (the observation level) are specified in the usual manner, for example, [fw=fwtvar1]. varname can be any valid Stata variable name, and you can specify fweight() at levels two and higher of a multilevel model. For example, in the two-level model

```
. mecmd fixed_portion [fw = wt1] || school: ..., fweight(wt2) ...
```

the variable wt1 would hold the first-level (the observation-level) frequency weights, and wt2 would hold the second-level (the school-level) frequency weights.

iweight(varname) specifies importance weights at higher levels in a multilevel model, whereas importance weights at the first level (the observation level) are specified in the usual manner, for example, [iw=iwtvar1]. varname can be any valid Stata variable name, and you can specify iweight() at levels two and higher of a multilevel model. For example, in the two-level model

```
. mecmd fixed_portion [iw = wt1] || school: ..., iweight(wt2) ...
```

the variable wt1 would hold the first-level (the observation-level) importance weights, and wt2 would hold the second-level (the school-level) importance weights.

pweight(varname) specifies sampling weights at higher levels in a multilevel model, whereas sampling weights at the first level (the observation level) are specified in the usual manner, for example, [pw=pwtvar1]. varname can be any valid Stata variable name, and you can specify pweight() at levels two and higher of a multilevel model. For example, in the two-level model

```
. mecmd fixed_portion [pw = wt1] || school: ..., pweight(wt2) ...
```

variable wt1 would hold the first-level (the observation-level) sampling weights, and wt2 would hold the second-level (the school-level) sampling weights.

dispersion(mean | constant) specifies the parameterization of the conditional overdispersion given random effects. dispersion(mean), the default, yields a model where the conditional overdispersion is a function of the conditional mean given random effects. For example, in a two-level model, the conditional overdispersion is equal to \(1 + \alpha E(y_{ij}\mid u_j)\). dispersion(constant) yields a model where the conditional overdispersion is constant and is equal to \(1 + \delta\). \(\alpha\) and \(\delta\) are the respective conditional overdispersion parameters.

constraints(constraints), collinear; see [R] estimation options.

vce(vcetype) specifies the type of standard error reported, which includes types that are derived from asymptotic theory (oim), that are robust to some kinds of misspecification (robust), and that allow for intragroup correlation (cluster clustvar); see [R] vce_option. If vce(robust) is specified, robust variances are clustered at the highest level in the multilevel model.
Reporting level(#); see [R] estimation options.

irr reports estimated fixed-effects coefficients transformed to incidence-rate ratios, that is, \( \exp(\beta) \) rather than \( \beta \). Standard errors and confidence intervals are similarly transformed. This option affects how results are displayed, not how they are estimated or stored. \textit{irr} may be specified either at estimation or upon replay.

\texttt{nocnsreport}; see [R] estimation options.

\texttt{notable} suppresses the estimation table, either at estimation or upon replay.

\texttt{noheader} suppresses the output header, either at estimation or upon replay.

\texttt{nogroup} suppresses the display of group summary information (number of groups, average group size, minimum, and maximum) from the output header.

\texttt{nolrtest} prevents \texttt{menbreg} from performing a likelihood-ratio test that compares the mixed-effects negative binomial model with standard (marginal) negative binomial regression. This option may also be specified upon replay to suppress this test from the output.

display\_options: \texttt{noci}, \texttt{nopvalues}, \texttt{noomitted}, \texttt{vsquish}, \texttt{noemptycells}, \texttt{baselevels}, \texttt{ailbaselevels}, \texttt{nofvlabel}, \texttt{fvwrap(\#)}, \texttt{fvwraupon(style)}, \texttt{cformat(\%fmt)}, \texttt{pformat(\%fmt)}, \texttt{sformat(\%fmt)}, and \texttt{nolstretch}; see [R] estimation options.

Integration\_options: \texttt{intmethod(intmethod)} specifies the integration method to be used for the random-effects model. \texttt{mvaghermite} performs mean–variance adaptive Gauss–Hermite quadrature; \texttt{mcaghermite} performs mode-curvature adaptive Gauss–Hermite quadrature; \texttt{ghermite} performs nonadaptive Gauss–Hermite quadrature; and \texttt{laplace} performs the Laplacian approximation, equivalent to mode-curvature adaptive Gaussian quadrature with one integration point.

The default integration method is \texttt{mvaghermite} unless a crossed random-effects model is fit, in which case the default integration method is \texttt{laplace}. The Laplacian approximation has been known to produce biased parameter estimates; however, the bias tends to be more prominent in the estimates of the variance components rather than in the estimates of the fixed effects.

For crossed random-effects models, estimation with more than one quadrature point may be prohibitively intensive even for a small number of levels. For this reason, the integration method defaults to the Laplacian approximation. You may override this behavior by specifying a different integration method.

\texttt{intpoints(#)} sets the number of integration points for quadrature. The default is \texttt{intpoints(7)}, which means that seven quadrature points are used for each level of random effects. This option is not allowed with \texttt{intmethod(laplace)}.

The more integration points, the more accurate the approximation to the log likelihood. However, computation time increases as a function of the number of quadrature points raised to a power equaling the dimension of the random-effects specification. In crossed random-effects models and in models with many levels or many random coefficients, this increase can be substantial.

Maximization\_options: \texttt{difficult}, \texttt{technique(algorithm\_spec)}, \texttt{iterate(#)}, [\texttt{no}] \texttt{log}, \texttt{trace}, \texttt{gradient}, \texttt{showstep}, \texttt{hessian}, \texttt{showtolerance}, \texttt{tolerance(#)}, \texttt{ltolerance(#)}, \texttt{nrtolerance(#)}, \texttt{nonrtolerance}, and \texttt{from(init\_specs)}; see [R] \texttt{maximize}. Those that require special mention for \texttt{menbreg} are listed below.
from() accepts a properly labeled vector of initial values or a list of coefficient names with values. A list of values is not allowed.

The following options are available with `menbreg` but are not shown in the dialog box:

- `startvalues(svmethod)`, `startgrid[(gridspec)]`, `noestimate`, and `dnumerical`; see `[ME] meglm`.
- `coeflegend`; see `[R] estimation options`.

**Remarks and examples**

For a general introduction to `me` commands, see `[ME] me`.

`menbreg` is a convenience command for `meglm` with a log link and an nbinomial family; see `[ME] meglm`.

Remarks are presented under the following headings:

- Introduction
- Two-level models

**Introduction**

Mixed-effects negative binomial regression is negative binomial regression containing both fixed effects and random effects. In longitudinal data and panel data, random effects are useful for modeling intracluster correlation; that is, observations in the same cluster are correlated because they share common cluster-level random effects.

Comprehensive treatments of mixed models are provided by, for example, Searle, Casella, and McCulloch (1992); Verbeke and Molenberghs (2000); Raudenbush and Bryk (2002); Demidenko (2004); Hedeker and Gibbons (2006); McCulloch, Searle, and Neuhaus (2008); and Rabe-Hesketh and Skrondal (2012). Rabe-Hesketh and Skrondal (2012, chap. 13) is a good introductory reading on applied multilevel modeling of count data.

`menbreg` allows for not just one, but many levels of nested clusters of random effects. For example, in a three-level model you can specify random effects for schools and then random effects for classes nested within schools. In this model, the observations (presumably, the students) comprise the first level, the classes comprise the second level, and the schools comprise the third.

However, for simplicity, consider a two-level model, where for a series of $M$ independent clusters, and conditional on the latent variable $\zeta_{ij}$ and a set of random effects $u_j$,

$$y_{ij}|\zeta_{ij} \sim \text{Poisson}(\zeta_{ij})$$

and

$$\zeta_{ij}|u_j \sim \text{Gamma}(r_{ij}, p_{ij})$$

and

$$u_j \sim N(0, \Sigma)$$

where $y_{ij}$ is the count response of the $i$th observation, $i = 1, \ldots, n_j$, from the $j$th cluster, $j = 1, \ldots, M$, and $r_{ij}$ and $p_{ij}$ have two different parameterizations, (2) and (3) below. The random effects $u_j$ are $M$ realizations from a multivariate normal distribution with mean 0 and $q \times q$ variance matrix $\Sigma$. The random effects are not directly estimated as model parameters but are instead summarized according to the unique elements of $\Sigma$, known as variance components.
The probability that a random response $y_{ij}$ takes the value $y$ is then given by

$$\Pr(y_{ij} = y | u_j) = \frac{\Gamma(y + r_{ij})}{\Gamma(y + 1)\Gamma(r_{ij})}p_{ij}^{r_{ij}}(1 - p_{ij})^y$$

(1)

where for convenience we suppress the dependence of the observable data $y_{ij}$ on $r_{ij}$ and $p_{ij}$.

Model (1) is an extension of the standard negative binomial model (see [R nbreg]) to incorporate normally distributed random effects at different hierarchical levels. The standard negative binomial model itself can be viewed as a random-effects model, a Poisson model with a gamma-distributed random effect. The standard negative binomial model is used to model overdispersed count data for which the variance is greater than that of a Poisson model. In a Poisson model, the variance is equal to the mean, and thus overdispersion is defined as the extra variability compared with the mean. According to this definition, the negative binomial model presents two different parameterizations of the overdispersion: the mean parameterization, where the overdispersion is a function of the mean, $1 + \alpha E(Y | x), \alpha > 0$; and the constant parameterization, where the overdispersion is a constant function, $1 + \delta, \delta \geq 0$. We refer to $\alpha$ and $\delta$ as conditional overdispersion parameters.

Let $\mu_{ij} = E(y_{ij} | x, u_j) = \exp(x_{ij}\beta + z_{ij}u_j)$, where $x_{ij}$ is the $1 \times p$ row vector of the fixed-effects covariates, analogous to the covariates you would find in a standard negative binomial regression model, with regression coefficients (fixed effects) $\beta$; $z_{ij}$ is the $1 \times q$ vector of the random-effects covariates and can be used to represent both random intercepts and random coefficients. For example, in a random-intercept model, $z_{ij}$ is simply the scalar 1. One special case places $z_{ij} = x_{ij}$, so that all covariate effects are essentially random and distributed as multivariate normal with mean $\beta$ and variance $\Sigma$.

Similarly to the standard negative binomial model, we can consider two parameterizations of what we call the conditional overdispersion, the overdispersion conditional on random effects, in a random-effects negative binomial model. For the mean-overdispersion (or, more technically, mean-conditional-overdispersion) parameterization,

$$r_{ij} = 1/\alpha \quad \text{and} \quad p_{ij} = \frac{1}{1 + \alpha \mu_{ij}}$$

(2)

and the conditional overdispersion is equal to $1 + \alpha \mu_{ij}$. For the constant-overdispersion (or, more technically, constant-conditional-overdispersion) parameterization,

$$r_{ij} = \mu_{ij}/\delta \quad \text{and} \quad p_{ij} = \frac{1}{1 + \delta}$$

(3)

and the conditional overdispersion is equal to $1 + \delta$. In what follows, for brevity, we will use the term overdispersion parameter to mean conditional overdispersion parameter, unless stated otherwise.

In the context of random-effects negative binomial models, it is important to decide which model is used as a reference model for the definition of the overdispersion. For example, if we consider a corresponding random-effects Poisson model as a comparison model, the parameters $\alpha$ and $\delta$ can still be viewed as unconditional overdispersion parameters, as we show below, although the notion of a constant overdispersion is no longer applicable.

If we retain the definition of the overdispersion as the excess variation with respect to a Poisson process for which the variance is equal to the mean, we need to carefully distinguish between the marginal (unconditional) mean with random effects integrated out and the conditional mean given random effects.
In what follows, for simplicity, we omit the dependence of the formulas on \( x \). Conditionally on random effects, the (conditional) dispersion \( \text{Var}(y_{ij} | u_j) = (1 + \alpha \mu_{ij}) \mu_{ij} \) for the mean parameterization and \( \text{Var}(y_{ij} | u_j) = (1 + \delta) \mu_{ij} \) for the constant parameterization; the usual interpretation of the parameters holds (conditionally).

If we consider the marginal mean or, specifically, the marginal dispersion for, for example, a two-level random-intercept model, then

\[
\text{Var}(y_{ij}) = \left[ 1 + \left\{ \exp(\sigma^2)(1 + \alpha) - 1 \right\} E(y_{ij}) \right] E(y_{ij})
\]

for the mean parameterization and

\[
\text{Var}(y_{ij}) = \left[ 1 + \delta + \left\{ \exp(\sigma^2) - 1 \right\} E(y_{ij}) \right] E(y_{ij})
\]

for the constant parameterization, where \( \sigma^2 \) is the variance component corresponding to the random intercept.

A few things of interest compared with the standard negative binomial model. First, the random-effects negative binomial model is not strictly an overdispersed model. The combination of values of \( \alpha \) and \( \sigma^2 \) can lead to an underdispersed model, a model with smaller variability than the Poisson variability. Underdispersed models are not as common in practice, so we will concentrate on the overdispersion in this entry. Second, \( \alpha \) (or \( \delta \)) no longer solely determine the overdispersion and thus cannot be viewed as unconditional overdispersion parameters. Overdispersion is now a function of both \( \alpha \) (or \( \delta \)) and \( \sigma^2 \). Third, the notion of a constant overdispersion is not applicable.

Two special cases are worth mentioning. When \( \sigma^2 = 0 \), the dispersion reduces to that of a standard negative binomial model. When \( \alpha = 0 \) (or \( \delta = 0 \)), the dispersion reduces to that of a two-level random-intercept Poisson model, which itself is, in general, an overdispersed model; see Rabe-Hesketh and Skrondal (2012, chap. 13.7) for more details. As such, \( \alpha \) and \( \delta \) retain the typical interpretation as dispersion parameters relative to a random-intercept Poisson model.

Model (1) is an example of a generalized linear mixed model (GLMM), which generalizes the linear mixed-effects (LME) model to non-Gaussian responses. You can fit LMEs in Stata by using \texttt{mixed} and fit GLMMs by using \texttt{meglm}. Because of the relationship between LMEs and GLMMs, there is insight to be gained through examination of the linear mixed model. This is especially true for Stata users because the terminology, syntax, options, and output for fitting these types of models are nearly identical. See [ME] \texttt{mixed} and the references therein, particularly in the Introduction of [ME] \texttt{mixed}, for more information.

Log-likelihood calculations for fitting any generalized mixed-effects model require integrating out the random effects. One widely used modern method is to directly estimate the integral required to calculate the log likelihood by Gauss–Hermite quadrature or some variation thereof. Because the log likelihood itself is estimated, this method has the advantage of permitting likelihood-ratio tests for comparing nested models. Also, if done correctly, quadrature approximations can be quite accurate, thus minimizing bias.

\texttt{menbreg} supports three types of Gauss–Hermite quadrature and the Laplacian approximation method; see Methods and formulas of [ME] \texttt{meglm} for details.

Below we present two short examples of mixed-effects negative binomial regression; refer to [ME] \texttt{me} and [ME] \texttt{meglm} for more examples including crossed-effects models.
Two-level models

Example 1

Rabe-Hesketh and Skrondal (2012, chap. 13.7) analyze the data from Winkelmann (2004) on the impact of the 1997 health reform in Germany on the number of doctor visits. The intent of policymakers was to reduce government expenditures on health care. We use a subsample of the data restricted to 1,158 women who were employed full time the year before or after the reform.

. use http://www.stata-press.com/data/r14/drvisits
. describe
Contains data from http://www.stata-press.com/data/r14/drvisits.dta
obs: 2,227
vars: 8 23 Jan 2014 18:39
size: 71,264

<table>
<thead>
<tr>
<th>variable name</th>
<th>storage</th>
<th>display</th>
<th>label</th>
<th>variable label</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>float</td>
<td>%9.0g</td>
<td></td>
<td>person id</td>
</tr>
<tr>
<td>numvisit</td>
<td>float</td>
<td>%9.0g</td>
<td></td>
<td>number of doctor visits in the last 3 months before interview</td>
</tr>
<tr>
<td>age</td>
<td>float</td>
<td>%9.0g</td>
<td></td>
<td>age in years</td>
</tr>
<tr>
<td>educ</td>
<td>float</td>
<td>%9.0g</td>
<td></td>
<td>education in years</td>
</tr>
<tr>
<td>married</td>
<td>float</td>
<td>%9.0g</td>
<td></td>
<td>=1 if married, 0 otherwise</td>
</tr>
<tr>
<td>badh</td>
<td>float</td>
<td>%9.0g</td>
<td></td>
<td>self-reported health status, =1 if bad</td>
</tr>
<tr>
<td>loginc</td>
<td>float</td>
<td>%9.0g</td>
<td></td>
<td>log of household income</td>
</tr>
<tr>
<td>reform</td>
<td>float</td>
<td>%9.0g</td>
<td></td>
<td>=0 if interview before reform, =1 if interview after reform</td>
</tr>
</tbody>
</table>

Sorted by:

The dependent variable, numvisit, is a count of doctor visits. The covariate of interest is a dummy variable, reform, which indicates whether a doctor visit took place before or after the reform. Other covariates include a self-reported health status, age, education, marital status, and a log of household income.
We first fit a two-level random-intercept Poisson model. We specify the random intercept at the id level, that is, an individual-person level.

```
.mepoisson numvisit reform age educ married badh loginc || id:, irr
Fitting fixed-effects model:
Iteration 0:  log likelihood = -9326.8542
Iteration 1:  log likelihood = -5989.7308
Iteration 2:  log likelihood = -5942.7581
Iteration 3:  log likelihood = -5942.7243
Iteration 4:  log likelihood = -5942.7243
Refining starting values:
Grid node 0:  log likelihood = -4761.1257
Fitting full model:
Iteration 0:  log likelihood = -4761.1257
Iteration 1:  log likelihood = -4683.2239
Iteration 2:  log likelihood = -4646.9329
Iteration 3:  log likelihood = -4645.736
Iteration 4:  log likelihood = -4645.7371
Iteration 5:  log likelihood = -4645.7371
Mixed-effects Poisson regression
Number of obs = 2,227
Group variable:  id
Number of groups = 1,518
Obs per group:
  min = 1
  avg = 1.5
  max = 2
Integration method:  mvaghermite
  Integration pts. = 7
Log likelihood = -4645.7371
Prob > chi2 = 0.0000
```

| numvisit   | IRR     | Std. Err. | z     | P>|z|   | [95% Conf. Interval] |
|------------|---------|-----------|-------|-------|----------------------|
| reform     | .9517026| .0309352  | -1.52 | 0.128 | .8929617 1.014308    |
| age        | 1.005821| .002817   | 2.07  | 0.038 | 1.000315 1.011357    |
| educ       | 1.008788| .0127394  | 0.69  | 0.488 | .9841258 1.034068    |
| married    | 1.082078| .0596331  | 1.43  | 0.152 | .9712905 1.205503    |
| badh       | 2.471857| .151841   | 14.73 | 0.000 | 2.191471 2.788116    |
| loginc     | 1.094144| .0743018  | 1.32  | 0.185 | .9577909 1.249909    |
| _cons      | .5216748| .2668604  | -1.27 | 0.203 | .191413 1.421766     |
| var(_cons) | .8177932| .0503902  |       |       | .724761 .9227673     |

LR test vs. Poisson model: chibar2(01) = 2593.97    Prob >= chibar2 = 0.0000
.estimates store mepoisson

Because we specified the `irr` option, the parameters are reported as incidence-rate ratios. The health care reform seems to reduce the expected number of visits by 5%; but without statistical significance.

Because we have only one random effect at the id level, the table shows only one variance component. The estimate of $\sigma^2_u$ is 0.82 with standard error 0.05. The reported likelihood-ratio test shows that there is enough variability between women to favor a mixed-effects Poisson regression over a standard Poisson regression; see *Distribution theory for likelihood-ratio test* in [ME] _me_ for a discussion of likelihood-ratio testing of variance components.

It is possible that after conditioning on the person-level random effect, the counts of doctor visits are overdispersed. For example, medical problems occurring during the time period leading to the survey can result in extra doctor visits. We thus reexamine the data with `menbreg`. 
The estimated effect of the health care reform now corresponds to the reduction in the number of doctor visits by 10%—twice as much compared with the Poisson model—and this effect is significant at the 5% level.

The estimate of the variance component $\sigma_u^2$ drops down to 0.47 compared with `mepoisson`, which is not surprising given that now we have an additional parameter that controls the variability of the data.

Because the conditional overdispersion $\alpha$ is assumed to be greater than 0, it is parameterized on the log scale, and its log estimate is reported as `/lnalpha` in the output. In our model, $\hat{\alpha} = \exp(-0.80) = 0.45$. We can also compute the unconditional overdispersion in this model by using the corresponding formula in the Introduction above: $\exp(0.47) \times (1 + 0.45) - 1 = 1.32$.

The reported likelihood-ratio test shows that there is enough variability between women to favor a mixed-effects negative binomial regression over negative binomial regression without random effects.
We can also perform a likelihood-ratio test comparing the mixed-effects negative binomial model to the mixed-effects Poisson model. Because we are comparing two different estimators, we need to use the `force` option with `lrtest`. In general, there is no guarantee as to the validity or interpretability of the resulting likelihood-ratio test, but in our case we know the test is valid because the mixed-effects Poisson model is nested within the mixed-effects negative binomial model.

```
    . lrtest mepoisson ., force
```

```
Likelihood-ratio test
(Assumption: mepoisson nested in .)

    LR chi2(1) =    265.10
    Prob > chi2 =  0.0000
```

Note: The reported degrees of freedom assumes the null hypothesis is not on the boundary of the parameter space. If this is not true, then the reported test is conservative.

The reported likelihood-ratio test favors the mixed-effects negative binomial model. The reported test is conservative because the test of $H_0: \alpha = 0$ occurs on the boundary of the parameter space; see *Distribution theory for likelihood-ratio test* in [ME] `me` for details.

The above extends to models with more than two levels of nesting in the obvious manner, by adding more random-effects equations, each separated by `||`. The order of nesting goes from left to right as the groups go from biggest (highest level) to smallest (lowest level). To demonstrate a three-level model, we revisit example 2 from [ME] `meqrpoisson`.

### Example 2


```
    . use http://www.stata-press.com/data/r14/melanoma
    (Skin cancer (melanoma) data)
    . describe
    Contains data from http://www.stata-press.com/data/r14/melanoma.dta
    obs: 354  Skin cancer (melanoma) data
    vars: 6       30 May 2014 17:10
    size: 4,956 (_dta has notes)
```

```
    storage  display    value
      variable name    type  format   label      variable label
        nation     byte  %11.0g   n    Nation ID
        region     byte  %9.0g   Region ID: EEC level-I areas
        county    int    %9.0g   County ID: EEC level-II/level-III areas
        deaths    int    %9.0g   No. deaths during 1971–1980
        expected  float  %9.0g   No. expected deaths
        uv        float  %9.0g   UV dose, mean-centered
```

Sorted by:

Nine European nations (variable `nation`) are represented, and data were collected over geographical regions defined by EEC statistical services as level I areas (variable `region`), with deaths being recorded for each of 354 counties, which are level II or level III EEC-defined areas (variable `county`, which identifies the observations). Counties are nested within regions, and regions are nested within nations.
The variable `deaths` records the number of deaths for each county, and `expected` records the expected number of deaths (the exposure) on the basis of crude rates for the combined countries. The variable `uv` is a measure of exposure to ultraviolet (UV) radiation.

In example 2 of [ME] `mepoisson`, we noted that because counties also identified the observations, we could model overdispersion by using a four-level Poisson model with a random intercept at the county level. Here we fit a three-level negative binomial model with the default mean-dispersion parameterization.

```
.menbreg deaths uv, exposure(expected) || nation: || region:
Fitting fixed-effects model:
Iteration 0: log likelihood = -1361.855
Iteration 1: log likelihood = -1230.0211
Iteration 2: log likelihood = -1211.049
Iteration 3: log likelihood = -1202.5641
Iteration 4: log likelihood = -1202.5329
Iteration 5: log likelihood = -1202.5329
Refining starting values:
Grid node 0: log likelihood = -1209.6951
Fitting full model:
Iteration 0: log likelihood = -1209.6951 (not concave)
(output omitted)
Iteration 11: log likelihood = -1086.3902
Mixed-effects nb regression
Number of obs = 354
Overdispersion: mean
```

<table>
<thead>
<tr>
<th>Group Variable</th>
<th>No. of Groups</th>
<th>Observations per Group</th>
</tr>
</thead>
<tbody>
<tr>
<td>nation</td>
<td>9</td>
<td>3 39.3 95</td>
</tr>
<tr>
<td>region</td>
<td>78</td>
<td>1 4.5 13</td>
</tr>
</tbody>
</table>

Integration method: mvaghermite Integration pts. = 7
Wald chi2(1) = 8.73
Log likelihood = -1086.3902
Prob > chi2 = 0.0031

| deaths | Coef. | Std. Err. | z     | P>|z| | 95% Conf. Interval |
|--------|-------|-----------|-------|------|-------------------|
| uv     | -.0335933 | .0113725 | -2.95 | 0.003 | -.055883 -.0113035 |
| _cons  | -.0790606 | .1295931 | -0.61 | 0.542 | -.3330538 .1749372 |
| ln(expected) | 1 (exposure) | | | | |

| /lnalpha | Coef. | Std. Err. | z     | P>|z| | 95% Conf. Interval |
|----------|-------|-----------|-------|------|-------------------|
|         | -4.182603 | .3415036 | -12.25 | 0.000 | -4.851937 -3.513268 |

| nation var(_cons) | Coef. | Std. Err. | z     | P>|z| | 95% Conf. Interval |
|-------------------|-------|-----------|-------|------|-------------------|
|                   | .1283614 | .0678971 | .0455187 | .3619758 |
| nation>region var(_cons) | Coef. | Std. Err. | z     | P>|z| | 95% Conf. Interval |
|                    | .0401818 | .0104855 | .0240938 | .067012 |

LR test vs. nbinomial model: chi2(2) = 232.29
Prob > chi2 = 0.0000
Note: LR test is conservative and provided only for reference.
The estimates are very close to those of `meqrpoisson`. The conditional overdispersion in our model is \( \hat{\alpha} = \exp(-4.18) = 0.0153 \). It is in agreement with the estimate of the random intercept at the county level, 0.0146, in a four-level random-effects Poisson model reported by `meqrpoisson`. Because the negative binomial is a three-level model, we gained some computational efficiency over the four-level Poisson model.

### Stored results

`menbreg` stores the following in `e()`:

**Scalars**
- `e(N)` number of observations
- `e(k)` number of parameters
- `e(k_dv)` number of dependent variables
- `e(k_eq)` number of equations in `e(b)`
- `e(k_eq_model)` number of equations in overall model test
- `e(k_f)` number of fixed-effects parameters
- `e(k_r)` number of random-effects parameters
- `e(k_rs)` number of variances
- `e(k_rc)` number of covariances
- `e(df_m)` model degrees of freedom
- `e(ll)` log likelihood
- `e(N_clust)` number of clusters
- `e(chi2)` \( \chi^2 \)
- `e(p)` significance
- `e(ll_c)` log likelihood, comparison model
- `e(chi2_c)` \( \chi^2 \), comparison model
- `e(df_c)` degrees of freedom, comparison model
- `e(p_c)` significance, comparison model
- `e(rank)` rank of `e(V)`
- `e(ic)` number of iterations
- `e(rc)` return code
- `e(converged)` 1 if converged, 0 otherwise

**Macros**
- `e(cmd)` `meglm`
- `e(cmd2)` `menbreg`
- `e(cmdline)` command as typed
- `e(depvar)` name of dependent variable
- `e(wtype)` weight type
- `e(wexp)` weight expression (first-level weights)
- `e(fweight)` `fweight` variable for \( k \)th highest level, if specified
- `e(iweight)` `iweight` variable for \( k \)th highest level, if specified
- `e(pweight)` `pweight` variable for \( k \)th highest level, if specified
- `e(covariates)` list of covariates
- `e(ivars)` grouping variables
- `e(model)` `nbreg`
- `e(title)` title in estimation output
- `e(link)` `log`
- `e(family)` `nbinomial`
- `e(clustvar)` name of cluster variable
- `e(dispersion)` mean or constant
- `e(offset)` offset
- `e(exposure)` exposure variable
- `e(intmethod)` integration method
- `e(n_quad)` number of integration points
- `e(chi2type)` Wald; type of model \( \chi^2 \)
- `e(vcetype)` `vcetype` specified in `vce()`
- `e(vcetype)` title used to label Std. Err.
Methods and formulas

Without a loss of generality, consider a two-level negative binomial model. For cluster \( j, j = 1, \ldots, M \), the conditional distribution of \( y_j = (y_{j1}, \ldots, y_{jn_j})' \), given a set of cluster-level random effects \( u_j \) and the conditional overdispersion parameter \( \alpha \) in a mean-overdispersion parameterization, is

\[
f(y_j|u_j, \alpha) = \prod_{i=1}^{n_j} \left\{ \frac{\Gamma(y_{ij} + r)}{\Gamma(y_{ij} + 1)\Gamma(r)} p_{ij}^{y_{ij}}(1 - p_{ij})^{y_{ij}} \right\}
\]

\[
= \exp \left[ \sum_{i=1}^{n_j} \{ \log \Gamma(y_{ij} + r) - \log \Gamma(y_{ij} + 1) - \log \Gamma(r) + c(y_{ij}, \alpha) \} \right]
\]

where \( c(y_{ij}, \alpha) \) is defined as

\[
-\frac{1}{\alpha} \log\{1 + \exp(\eta_{ij} + \log \alpha)\} - y_{ij} \log\{1 + \exp(-\eta_{ij} - \log \alpha)\}
\]

and \( r = 1/\alpha, \ p_{ij} = 1/(1 + \alpha \mu_{ij}), \) and \( \eta_{ij} = x_{ij} \beta + z_{ij} u_j \).
For the constant-overdispersion parameterization with the conditional overdispersion parameter \( \delta \), the conditional distribution of \( y_j \) is

\[
f(y_j | u_j, \delta) = \prod_{i=1}^{n_j} \left\{ \frac{\Gamma(y_{ij} + r_{ij})}{\Gamma(y_{ij} + 1) \Gamma(r_{ij})} p^{r_{ij}} (1 - p)^{y_{ij}} \right\}
\]

\[
= \exp \left[ \sum_{i=1}^{n_j} \{ \log \Gamma(y_{ij} + r_{ij}) - \log \Gamma(y_{ij} + 1) - \log \Gamma(r_{ij}) + c(y_{ij}, \delta) \} \right]
\]

where \( c(y_{ij}, \delta) \) is defined as

\[
- \left( \frac{\mu_{ij}}{\delta} + y_{ij} \right) \log(1 + \delta) + y_{ij} \log \delta
\]

and \( r_{ij} = \mu_{ij} / \delta \) and \( p = 1 / (1 + \delta) \).

For conciseness, let \( \gamma \) denote either conditional overdispersion parameter. Because the prior distribution of \( u_j \) is multivariate normal with mean \( 0 \) and \( q \times q \) variance matrix \( \Sigma \), the likelihood contribution for the \( j \)th cluster is obtained by integrating \( u_j \) out of the joint density \( f(y_j, u_j, \gamma) \),

\[
L_j(\beta, \Sigma, \gamma) = (2\pi)^{-q/2} |\Sigma|^{-1/2} \int f(y_j | u_j, \gamma) \exp \left( -u_j' \Sigma^{-1} u_j / 2 \right) du_j
\]

\[
= (2\pi)^{-q/2} |\Sigma|^{-1/2} \int \exp \{ h(\beta, \Sigma, u_j, \gamma) \} du_j
\]

where

\[
h(\beta, \Sigma, u_j, \gamma) = f(y_j | u_j, \gamma) - u_j' \Sigma^{-1} u_j / 2
\]

and for convenience, in the arguments of \( h(\cdot) \) we suppress the dependence on the observable data \((y_j, X_j, Z_j)\).

The Laplacian approximation is based on a second-order Taylor expansion of \( h(\beta, \Sigma, u_j, \gamma) \) about the value of \( u_j \) that maximizes it; see Methods and formulas in [ME] meglm for details.

Gaussian quadrature relies on transforming the multivariate integral in (4) into a set of nested univariate integrals. Each univariate integral can then be evaluated using a form of Gaussian quadrature; see Methods and formulas in [ME] meglm for details.

The log likelihood for the entire dataset is simply the sum of the contributions of the \( M \) individual clusters, namely, \( \mathcal{L}(\beta, \Sigma, \gamma) = \sum_{j=1}^{M} L_j(\beta, \Sigma, \gamma) \).

Maximization of \( \mathcal{L}(\beta, \Sigma, \gamma) \) is performed with respect to \((\beta, \ln \gamma, \sigma^2)\), where \( \sigma^2 \) is a vector comprising the unique elements of \( \Sigma \). Parameter estimates are stored in e(b) as \((\hat{\beta}, \hat{\ln \gamma}, \hat{\sigma^2})\), with the corresponding variance–covariance matrix stored in e(V).

menbreg supports multilevel weights and survey data; see Methods and formulas in [ME] meglm for details.
References


Also see

[ME] `membreg postestimation` — Postestimation tools for membreg

[ME] `mepoisson` — Multilevel mixed-effects Poisson regression

[ME] `meqrpoisson` — Multilevel mixed-effects Poisson regression (QR decomposition)

[ME] `me` — Introduction to multilevel mixed-effects models

[SEM] `intro5` — Tour of models (*Multilevel mixed-effects models*)

[SVY] `svy estimation` — Estimation commands for survey data

[XT] `xtmembreg` — Fixed-effects, random-effects, & population-averaged negative binomial models

[U] `20 Estimation and postestimation commands`
### Postestimation commands

The following postestimation command is of special interest after `menbreg`:

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>estat group</td>
<td>summarize the composition of the nested groups</td>
</tr>
</tbody>
</table>

The following standard postestimation commands are also available:

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>contrast</td>
<td>contrasts and ANOVA-style joint tests of estimates</td>
</tr>
<tr>
<td>estat ic</td>
<td>Akaike’s and Schwarz’s Bayesian information criteria (AIC and BIC)</td>
</tr>
<tr>
<td>estat summarize</td>
<td>summary statistics for the estimation sample</td>
</tr>
<tr>
<td>estat vce</td>
<td>variance–covariance matrix of the estimators (VCE)</td>
</tr>
<tr>
<td>estat (svy)</td>
<td>postestimation statistics for survey data</td>
</tr>
<tr>
<td>estimates</td>
<td>cataloging estimation results</td>
</tr>
<tr>
<td>*hausman</td>
<td>Hausman’s specification test</td>
</tr>
<tr>
<td>lincom</td>
<td>point estimates, standard errors, testing, and inference for linear</td>
</tr>
<tr>
<td></td>
<td>combinations of coefficients</td>
</tr>
<tr>
<td>*lrtest</td>
<td>likelihood-ratio test</td>
</tr>
<tr>
<td>margins</td>
<td>marginal means, predictive margins, marginal effects, and average marginal</td>
</tr>
<tr>
<td></td>
<td>effects</td>
</tr>
<tr>
<td>marginsplot</td>
<td>graph the results from margins (profile plots, interaction plots, etc.)</td>
</tr>
<tr>
<td>nlcom</td>
<td>point estimates, standard errors, testing, and inference for nonlinear</td>
</tr>
<tr>
<td></td>
<td>combinations of coefficients</td>
</tr>
<tr>
<td>predict</td>
<td>predictions, residuals, influence statistics, and other diagnostic measures</td>
</tr>
<tr>
<td>predictnl</td>
<td>point estimates, standard errors, testing, and inference for generalized</td>
</tr>
<tr>
<td></td>
<td>predictions</td>
</tr>
<tr>
<td>pwcompare</td>
<td>pairwise comparisons of estimates</td>
</tr>
<tr>
<td>test</td>
<td>Wald tests of simple and composite linear hypotheses</td>
</tr>
<tr>
<td>testnl</td>
<td>Wald tests of nonlinear hypotheses</td>
</tr>
</tbody>
</table>

* `hausman` and `lrtest` are not appropriate with `svy` estimation results.
predict

Description for predict

predict creates a new variable containing predictions such as mean responses; linear predictions; density and distribution functions; standard errors; and Pearson, deviance, and Anscombe residuals.

Menu for predict

Statistics > Postestimation

Syntax for predict

Syntax for obtaining predictions of the outcome and other statistics

```
predict [type] newvarsspec [if] [in] [ , statistic options ]
```

Syntax for obtaining estimated random effects and their standard errors

```
predict [type] newvarsspec [if] [in] , reffects [re_options ]
```

Syntax for obtaining ML scores

```
predict [type ] newvarsspec [if ] [in] , scores
```

`newvarsspec` is `stub*` or `newvarlist`.

<table>
<thead>
<tr>
<th>statistic</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Main</td>
<td></td>
</tr>
<tr>
<td>mu</td>
<td>mean response; the default</td>
</tr>
<tr>
<td>eta</td>
<td>fitted linear predictor</td>
</tr>
<tr>
<td>xb</td>
<td>linear predictor for the fixed portion of the model only</td>
</tr>
<tr>
<td>stdp</td>
<td>standard error of the fixed-portion linear prediction</td>
</tr>
<tr>
<td>density</td>
<td>predicted density function</td>
</tr>
<tr>
<td>distribution</td>
<td>predicted distribution function</td>
</tr>
<tr>
<td>pearson</td>
<td>Pearson residuals</td>
</tr>
<tr>
<td>deviance</td>
<td>deviance residuals</td>
</tr>
<tr>
<td>anscombe</td>
<td>Anscombe residuals</td>
</tr>
</tbody>
</table>

These statistics are available both in and out of sample; type `predict ... if e(sample) ...` if wanted only for the estimation sample.
### options

<table>
<thead>
<tr>
<th>Description</th>
<th>Main</th>
</tr>
</thead>
<tbody>
<tr>
<td>compute statistic conditional on estimated random effects; default is <code>conditional(ebmeans)</code></td>
<td><code>conditional(ctype)</code></td>
</tr>
<tr>
<td>compute statistic marginally with respect to the random effects</td>
<td><code>marginal</code></td>
</tr>
<tr>
<td>make calculation ignoring offset or exposure</td>
<td><code>nooffset</code></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Description</th>
<th>Integration</th>
</tr>
</thead>
<tbody>
<tr>
<td>integration options</td>
<td><code>int_options</code></td>
</tr>
</tbody>
</table>

Option `int_options` may not be combined with `marginal`.

### ctye

<table>
<thead>
<tr>
<th>Description</th>
<th>Main</th>
</tr>
</thead>
<tbody>
<tr>
<td>empirical Bayes means of random effects; the default</td>
<td><code>ebmeans</code></td>
</tr>
<tr>
<td>empirical Bayes modes of random effects</td>
<td><code>ebmodes</code></td>
</tr>
<tr>
<td>prediction for the fixed portion of the model only</td>
<td><code>fixedonly</code></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Description</th>
<th>Main</th>
</tr>
</thead>
<tbody>
<tr>
<td>use empirical Bayes means of random effects; the default</td>
<td><code>ebmeans</code></td>
</tr>
<tr>
<td>use empirical Bayes modes of random effects</td>
<td><code>ebmodes</code></td>
</tr>
<tr>
<td>calculate standard errors of empirical Bayes estimates</td>
<td>`reses(stub*</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Description</th>
<th>Integration</th>
</tr>
</thead>
<tbody>
<tr>
<td>integration options</td>
<td><code>int_options</code></td>
</tr>
</tbody>
</table>

### re_options

<table>
<thead>
<tr>
<th>Description</th>
<th>Main</th>
</tr>
</thead>
<tbody>
<tr>
<td>use # quadrature points to compute marginal predictions and empirical Bayes means</td>
<td><code>intpoints(#)</code></td>
</tr>
<tr>
<td>set maximum number of iterations in computing statistics involving empirical Bayes estimators</td>
<td><code>iterate(#)</code></td>
</tr>
<tr>
<td>set convergence tolerance for computing statistics involving empirical Bayes estimators</td>
<td><code>tolerance(#)</code></td>
</tr>
</tbody>
</table>

### Options for predict

- **Main**
  - `mu`, the default, calculates the predicted mean, that is, the predicted number of events.
  - `eta, xb, stdp, density, distribution, pearson, deviance, anscombe, scores, conditional(), marginal, and nooffset`; see [ME] `meglm postestimation`.
  - `reffects, ebmeans, ebmodes, and reses()`; see [ME] `meglm postestimation`.

- **Integration**
  - `intpoints(), iterate(), and tolerance()`; see [ME] `meglm postestimation`. 

margins

Description for margins

margins estimates margins of response for mean responses and linear predictions.

Menu for margins

Statistics > Postestimation

Syntax for margins

\[
\text{margins} \ [ \text{marginlist} ] \ [ , \ \text{options} ] \\
\text{margins} \ [ \text{marginlist} ], \ \text{predict(} \text{statistic} \ ... \ \text{)} \ [ \text{predict(} \text{statistic} \ ... \ \text{)} \ ... ] \ [ \text{options} ]
\]

<table>
<thead>
<tr>
<th>statistic</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>\text{mu}</td>
<td>mean response; the default</td>
</tr>
<tr>
<td>\text{eta}</td>
<td>fitted linear predictor</td>
</tr>
<tr>
<td>\text{xb}</td>
<td>linear predictor for the fixed portion of the model only</td>
</tr>
<tr>
<td>\text{stdp}</td>
<td>not allowed with \text{margins}</td>
</tr>
<tr>
<td>\text{density}</td>
<td>not allowed with \text{margins}</td>
</tr>
<tr>
<td>\text{distribution}</td>
<td>not allowed with \text{margins}</td>
</tr>
<tr>
<td>\text{pearson}</td>
<td>not allowed with \text{margins}</td>
</tr>
<tr>
<td>\text{deviance}</td>
<td>not allowed with \text{margins}</td>
</tr>
<tr>
<td>\text{anscombe}</td>
<td>not allowed with \text{margins}</td>
</tr>
<tr>
<td>\text{reffects}</td>
<td>not allowed with \text{margins}</td>
</tr>
<tr>
<td>\text{scores}</td>
<td>not allowed with \text{margins}</td>
</tr>
</tbody>
</table>

Options conditional(ebmeans) and conditional(ebModes) are not allowed with margins.

Option marginal is assumed where applicable if conditional(fixedonly) is not specified.

Statistics not allowed with \text{margins} are functions of stochastic quantities other than \text{e(b)}.

For the full syntax, see [R] margins.
estat

Description for estat

estat group reports the number of groups and minimum, average, and maximum group sizes for each level of the model. Model levels are identified by the corresponding group variable in the data. Because groups are treated as nested, the information in this summary may differ from what you would get if you used the tabulate command on each group variable individually.

Menu for estat

Statistics > Postestimation

Syntax for estat

estat group

Remarks and examples

Various predictions, statistics, and diagnostic measures are available after fitting a mixed-effects negative binomial model with menbreg. For the most part, calculation centers around obtaining estimates of the subject/group-specific random effects. Random effects are not estimated when the model is fit but instead need to be predicted after estimation.

Here we show a short example of predicted counts and predicted random effects; refer to [ME] meglm postestimation for additional examples applicable to mixed-effects generalized linear models.

Example 1

In example 2 of [ME] menbreg, we modeled the number of deaths among males in nine European nations as a function of exposure to ultraviolet radiation (uv). We used a three-level negative binomial model with random effects at the nation and region levels.
. use http://www.stata-press.com/data/r14/melanoma
(Skin cancer (melanoma) data)
. menbreg deaths uv, exposure(expected) || nation: || region:
(output omitted)

We can use `predict` to obtain the predicted counts as well as the estimates of the random effects at the nation and region levels.

. predict mu
(predictions based on fixed effects and posterior means of random effects)
(option `mu` assumed)
(using 7 quadrature points)
. predict re_nat re_reg, reffects
(calculating posterior means of random effects)
(using 7 quadrature points)

Stata displays a note that the predicted values of `mu` are based on the posterior means of random effects. You can use option `modes` to obtain predictions based on the posterior modes of random effects.

Here we list the data for the first nation in the dataset, which happens to be Belgium:

. list nation region deaths mu re_nat re_reg if nation==1, sepby(region)

<table>
<thead>
<tr>
<th>nation</th>
<th>region</th>
<th>deaths</th>
<th>mu</th>
<th>re_nat</th>
<th>re_reg</th>
</tr>
</thead>
<tbody>
<tr>
<td>Belgium</td>
<td>1</td>
<td>79</td>
<td>64.4892</td>
<td>-.0819939</td>
<td>.2937711</td>
</tr>
<tr>
<td>Belgium</td>
<td>2</td>
<td>80</td>
<td>77.64736</td>
<td>-.0819939</td>
<td>.024005</td>
</tr>
<tr>
<td>Belgium</td>
<td>2</td>
<td>51</td>
<td>44.56528</td>
<td>-.0819939</td>
<td>.024005</td>
</tr>
<tr>
<td>Belgium</td>
<td>2</td>
<td>43</td>
<td>53.10434</td>
<td>-.0819939</td>
<td>.024005</td>
</tr>
<tr>
<td>Belgium</td>
<td>2</td>
<td>89</td>
<td>65.35963</td>
<td>-.0819939</td>
<td>.024005</td>
</tr>
<tr>
<td>Belgium</td>
<td>2</td>
<td>19</td>
<td>35.18457</td>
<td>-.0819939</td>
<td>.024005</td>
</tr>
<tr>
<td>Belgium</td>
<td>3</td>
<td>19</td>
<td>8.770186</td>
<td>-.0819939</td>
<td>-.3434432</td>
</tr>
<tr>
<td>Belgium</td>
<td>3</td>
<td>15</td>
<td>43.95521</td>
<td>-.0819939</td>
<td>-.3434432</td>
</tr>
<tr>
<td>Belgium</td>
<td>3</td>
<td>33</td>
<td>34.17878</td>
<td>-.0819939</td>
<td>-.3434432</td>
</tr>
<tr>
<td>Belgium</td>
<td>3</td>
<td>9</td>
<td>7.332448</td>
<td>-.0819939</td>
<td>-.3434432</td>
</tr>
<tr>
<td>Belgium</td>
<td>3</td>
<td>12</td>
<td>12.93873</td>
<td>-.0819939</td>
<td>-.3434432</td>
</tr>
</tbody>
</table>

We can see that the predicted random effects at the nation level, `re_nat`, are the same for all the observations. Similarly, the predicted random effects at the region level, `re_reg`, are the same within each region. The predicted counts, `mu`, are not as close to the observed deaths as the predicted counts from the mixed-effects Poisson model in example 1 of [ME] mepoisson postestimation.

### Methods and formulas

Methods and formulas for predicting random effects and other statistics are given in Methods and formulas of [ME] meglm postestimation.

### Also see

- [ME] menbreg — Multilevel mixed-effects negative binomial regression
- [ME] meglm postestimation — Postestimation tools for meglm
- [U] 20 Estimation and postestimation commands
meologit — Multilevel mixed-effects ordered logistic regression

Description

meologit fits mixed-effects logistic models for ordered responses. The actual values taken on by the response are irrelevant except that larger values are assumed to correspond to “higher” outcomes. The conditional distribution of the response given the random effects is assumed to be multinomial, with success probability determined by the logistic cumulative distribution function.

Quick start

Two-level ordered logit regression of y on indicators for levels of a and random intercepts by lev2

```stata
meologit y i.a || lev2:
```

Two-level model including fixed and random coefficients for x

```stata
meologit y i.a x || lev2: x
```

As above, but report odds ratios instead of coefficients

```stata
meologit y i.a x || lev2: x, or
```

Three-level model of y on a, x, and their interaction using factor variable notation and random intercepts by lev2 and lev3 with lev2 nested within lev3

```stata
meologit y a##c.x || lev3: || lev2:
```

Menu

Statistics > Multilevel mixed-effects models > Ordered logistic regression
Syntax

meologit depvar fe_equation [ || re_equation] [ || re_equation ...] [ , options ]

where the syntax of fe_equation is

[ indepvars ] [ if ] [ in ] [ weight ] [ , fe_options ]

and the syntax of re_equation is one of the following:

for random coefficients and intercepts

levelvar: [ varlist ] [ , re_options ]

for random effects among the values of a factor variable

levelvar: R. varname

levelvar is a variable identifying the group structure for the random effects at that level or is _all representing one group comprising all observations.

<table>
<thead>
<tr>
<th>fe_options</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>offset(varname)</td>
<td>include varname in model with coefficient constrained to 1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>re_options</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>covariance(vartype)</td>
<td>variance–covariance structure of the random effects</td>
</tr>
<tr>
<td>noconstant</td>
<td>suppress constant term from the random-effects equation</td>
</tr>
<tr>
<td>fweight(varname)</td>
<td>frequency weights at higher levels</td>
</tr>
<tr>
<td>iweight(varname)</td>
<td>importance weights at higher levels</td>
</tr>
<tr>
<td>pweight(varname)</td>
<td>sampling weights at higher levels</td>
</tr>
</tbody>
</table>
options | Description
--- | ---
**Model**
- `constraints(constraints)` | apply specified linear constraints
- `collinear` | keep collinear variables
**SE/Robust**
- `vce(vcetype)` | `vcetype` may be `oim`, `robust`, or `cluster clustvar`
**Reporting**
- `level(#)` | set confidence level; default is `level(95)`
- `nocnsreport` | do not display constraints
- `notable` | suppress coefficient table
- `noheader` | suppress output header
- `nogroup` | suppress table summarizing groups
- `nolrtest` | do not perform likelihood-ratio test comparing with ordered logistic regression
- `display_options` | control columns and column formats, row spacing, line width, display of omitted variables and base and empty cells, and factor-variable labeling
**Integration**
- `intmethod(intmethod)` | integration method
- `intpoints(#)` | set the number of integration (quadrature) points for all levels; default is `intpoints(7)`
**Maximization**
- `maximize_options` | control the maximization process; seldom used
- `startvalues(svmethod)` | method for obtaining starting values
- `startgrid[ (gridspec) ]` | perform a grid search to improve starting values
- `noestimate` | do not fit the model; show starting values instead
- `dnumerical` | use numerical derivative techniques
- `coeflegend` | display legend instead of statistics
**vartype** | Description
- `independent` | one unique variance parameter per random effect, all covariances 0; the default unless the R. notation is used
- `exchangeable` | equal variances for random effects, and one common pairwise covariance
- `identity` | equal variances for random effects, all covariances 0; the default if the R. notation is used
- `unstructured` | all variances and covariances to be distinctly estimated
- `fixed(matname)` | user-selected variances and covariances constrained to specified values; the remaining variances and covariances unrestricted
- `pattern(matname)` | user-selected variances and covariances constrained to be equal; the remaining variances and covariances unrestricted
intmethod

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mvaghermite</td>
<td>mean–variance adaptive Gauss–Hermite quadrature; the default unless a crossed random-effects model is fit</td>
</tr>
<tr>
<td>mcaghermite</td>
<td>mode-curvature adaptive Gauss–Hermite quadrature</td>
</tr>
<tr>
<td>ghermite</td>
<td>nonadaptive Gauss–Hermite quadrature</td>
</tr>
<tr>
<td>laplace</td>
<td>Laplacian approximation; the default for crossed random-effects models</td>
</tr>
</tbody>
</table>

indepvars may contain factor variables; see [U] 11.4.3 Factor variables.

depvar, indepvars, and varlist may contain time-series operators; see [U] 11.4.4 Time-series varlists.

by and svy are allowed; see [U] 11.1.10 Prefix commands.

covariance(vartype) specifies the structure of the covariance matrix for the random effects and may be specified for each random-effects equation. vartype is one of the following: independent, exchangeable, identity, unstructured, fixed(matname), or pattern(matname).

covariance(independent) covariance structure allows for a distinct variance for each random effect within a random-effects equation and assumes that all covariances are 0. The default is covariance(independent) unless a crossed random-effects model is fit, in which case the default is covariance(identity).

covariance(exchangeable) structure specifies one common variance for all random effects and one common pairwise covariance.

covariance(identity) is short for “multiple of the identity”; that is, all variances are equal and all covariances are 0.

covariance(unstructured) allows for all variances and covariances to be distinct. If an equation consists of p random-effects terms, the unstructured covariance matrix will have p(p + 1)/2 unique parameters.

noconstant suppresses the constant (intercept) term; may be specified for any of or all the random-effects equations.
**fweight(varname)** specifies frequency weights at higher levels in a multilevel model, whereas frequency weights at the first level (the observation level) are specified in the usual manner, for example, `[fw=fwtvar1]`. *varname* can be any valid Stata variable name, and you can specify `fweight()` at levels two and higher of a multilevel model. For example, in the two-level model

```
    . mecmd fixed_portion [fw = wt1] || school: ... , fweight(wt2) ...
```

the variable *wt1* would hold the first-level (the observation-level) frequency weights, and *wt2* would hold the second-level (the school-level) frequency weights.

**iweight(varname)** specifies importance weights at higher levels in a multilevel model, whereas importance weights at the first level (the observation level) are specified in the usual manner, for example, `[iw=iwtvar1]`. *varname* can be any valid Stata variable name, and you can specify `iweight()` at levels two and higher of a multilevel model. For example, in the two-level model

```
    . mecmd fixed_portion [iw = wt1] || school: ... , iweight(wt2) ...
```

the variable *wt1* would hold the first-level (the observation-level) importance weights, and *wt2* would hold the second-level (the school-level) importance weights.

**pweight(varname)** specifies sampling weights at higher levels in a multilevel model, whereas sampling weights at the first level (the observation level) are specified in the usual manner, for example, `[pw=pwtvar1]`. *varname* can be any valid Stata variable name, and you can specify `pweight()` at levels two and higher of a multilevel model. For example, in the two-level model

```
    . mecmd fixed_portion [pw = wt1] || school: ... , pweight(wt2) ...
```

variable *wt1* would hold the first-level (the observation-level) sampling weights, and *wt2* would hold the second-level (the school-level) sampling weights.

**constraints(constraints)**, collinear; see [R] estimation options.

---

**vce(vcetype)** specifies the type of standard error reported, which includes types that are derived from asymptotic theory (oim), that are robust to some kinds of misspecification (robust), and that allow for intragroup correlation (cluster clustvar); see [R] vce_option. If `vce(robust)` is specified, robust variances are clustered at the highest level in the multilevel model.

---

**level(#)**; see [R] estimation options.

or reports estimated fixed-effects coefficients transformed to odds ratios, that is, \( \exp(\beta) \) rather than \( \beta \). Standard errors and confidence intervals are similarly transformed. This option affects how results are displayed, not how they are estimated. or may be specified either at estimation or upon replay.

**nocnsreport**; see [R] estimation options.

**notable** suppresses the estimation table, either at estimation or upon replay.

**nogroup** suppresses the display of group summary information (number of groups, average group size, minimum, and maximum) from the output header.

**nolrtest** prevents `meologit` from performing a likelihood-ratio test that compares the mixed-effects ordered logistic model with standard (marginal) ordered logistic regression. This option may also be specified upon replay to suppress this test from the output.
Integration

intmethod(intmethod) specifies the integration method to be used for the random-effects model. mvaghermite performs mean–variance adaptive Gauss–Hermite quadrature; mcaghermite performs mode-curvature adaptive Gauss–Hermite quadrature; ghermite performs nonadaptive Gauss–Hermite quadrature; and laplace performs the Laplacian approximation, equivalent to mode-curvature adaptive Gaussian quadrature with one integration point.

The default integration method is mvaghermite unless a crossed random-effects model is fit, in which case the default integration method is laplace. The Laplacian approximation has been known to produce biased parameter estimates; however, the bias tends to be more prominent in the estimates of the variance components rather than in the estimates of the fixed effects.

For crossed random-effects models, estimation with more than one quadrature point may be prohibitively intensive even for a small number of levels. For this reason, the integration method defaults to the Laplacian approximation. You may override this behavior by specifying a different integration method.

intpoints(#) sets the number of integration points for quadrature. The default is intpoints(7), which means that seven quadrature points are used for each level of random effects. This option is not allowed with intmethod(laplace).

The more integration points, the more accurate the approximation to the log likelihood. However, computation time increases as a function of the number of quadrature points raised to a power equaling the dimension of the random-effects specification. In crossed random-effects models and in models with many levels or many random coefficients, this increase can be substantial.

Maximization

maximize_options: difficult, technique(algorithm_spec), iterate(#), [no]log, trace, gradient, showstep, hessian, showtolerance, tolerance(#), ltolerance(#), nrtolerance(#), nonrtolerance, and from(init_specs); see [R] maximize. Those that require special mention for meologit are listed below.

from() accepts a properly labeled vector of initial values or a list of coefficient names with values. A list of values is not allowed.

The following options are available with meologit but are not shown in the dialog box:

startvalues(svmethod), startgrid[gridspec], noestimate, and dnumerical; see [ME] meglm.

coefflegend; see [R] estimation options.
Remarks and examples

For a general introduction to \texttt{me} commands, see [ME] \texttt{me}.

\texttt{meologit} is a convenience command for \texttt{meglm} with a logit link and an ordinal family; see [ME] \texttt{meglm}.

Remarks are presented under the following headings:

\begin{itemize}
  \item Introduction
  \item Two-level models
  \item Three-level models
\end{itemize}

Introduction

Mixed-effects ordered logistic regression is ordered logistic regression containing both fixed effects and random effects. An ordered response is a variable that is categorical and ordered, for instance, “poor”, “good”, and “excellent”, which might indicate a person’s current health status or the repair record of a car. In the absence of random effects, mixed-effects ordered logistic regression reduces to ordered logistic regression; see [R] \texttt{ologit}.

Comprehensive treatments of mixed models are provided by, for example, Searle, Casella, and McCulloch (1992); Verbeke and Molenberghs (2000); Raudenbush and Bryk (2002); Demidenko (2004); Hedeker and Gibbons (2006); McCulloch, Searle, and Neuhaus (2008); and Rabe-Hesketh and Skrondal (2012). Agresti (2010, chap. 10) and Rabe-Hesketh and Skrondal (2012, chap. 11) are good introductory readings on applied multilevel modeling of ordinal data.

\texttt{meologit} allows for many levels of nested clusters of random effects. For example, in a three-level model you can specify random effects for schools and then random effects for classes nested within schools. In this model, the observations (presumably, the students) comprise the first level, the classes comprise the second level, and the schools comprise the third.

However, for simplicity, for now we consider the two-level model, where for a series of $M$ independent clusters, and conditional on a set of fixed effects $x_{ij}$, a set of cutpoints $\kappa$, and a set of random effects $u_j$, the cumulative probability of the response being in a category higher than $k$ is

$$
\Pr(y_{ij} > k|x_{ij}, \kappa, u_j) = H(x_{ij}\beta + z_{ij}u_j - \kappa_k)
$$

for $j = 1, \ldots, M$ clusters, with cluster $j$ consisting of $i = 1, \ldots, n_j$ observations. The cutpoints $\kappa$ are labeled $\kappa_1, \kappa_2, \ldots, \kappa_{K-1}$, where $K$ is the number of possible outcomes. $H(\cdot)$ is the logistic cumulative distribution function that represents cumulative probability.

The $1 \times p$ row vector $x_{ij}$ are the covariates for the fixed effects, analogous to the covariates you would find in a standard logistic regression model, with regression coefficients (fixed effects) $\beta$. In our parameterization, $x_{ij}$ does not contain a constant term because its effect is absorbed into the cutpoints. For notational convenience here and throughout this manual entry, we suppress the dependence of $y_{ij}$ on $x_{ij}$.

The $1 \times q$ vector $z_{ij}$ are the covariates corresponding to the random effects and can be used to represent both random intercepts and random coefficients. For example, in a random-intercept model, $z_{ij}$ is simply the scalar 1. The random effects $u_j$ are $M$ realizations from a multivariate normal distribution with mean $0$ and $q \times q$ variance matrix $\Sigma$. The random effects are not directly estimated as model parameters but are instead summarized according to the unique elements of $\Sigma$, known as variance components. One special case of (1) places $z_{ij} = x_{ij}$, so that all covariate effects are essentially random and distributed as multivariate normal with mean $\beta$ and variance $\Sigma$. 
From (1), we can derive the probability of observing outcome \(k\) as

\[
Pr(y_{ij} = k | \kappa, u_j) = Pr(\kappa_{k-1} < x_{ij}\beta + z_{ij}u_j + \epsilon_{ij} \leq \kappa_k)
\]

\[
= Pr(\kappa_{k-1} - x_{ij}\beta - z_{ij}u_j < \epsilon_{ij} \leq \kappa_k - x_{ij}\beta - z_{ij}u_j)
\]

\[
= H(\kappa_k - x_{ij}\beta - z_{ij}u_j) - H(\kappa_{k-1} - x_{ij}\beta - z_{ij}u_j)
\]

where \(\kappa_0\) is taken as \(-\infty\) and \(\kappa_K\) is taken as \(+\infty\).

From the above, we may also write the model in terms of a latent linear response, where observed ordinal responses \(y_{ij}\) are generated from the latent continuous responses, such that

\[
y_{ij}^* = x_{ij}\beta + z_{ij}u_j + \epsilon_{ij}
\]

and

\[
y_{ij} = \begin{cases} 
1 & \text{if } y_{ij}^* \leq \kappa_1 \\
2 & \text{if } \kappa_1 < y_{ij}^* \leq \kappa_2 \\
& \vdots \\
K & \text{if } \kappa_{K-1} < y_{ij}^*
\end{cases}
\]

The errors \(\epsilon_{ij}\) are distributed as logistic with mean 0 and variance \(\pi^2/3\) and are independent of \(u_j\).

Model (1) is an example of a generalized linear mixed model (GLMM), which generalizes the linear mixed-effects (LME) model to non-Gaussian responses. You can fit LMEs in Stata by using `mixed` and fit GLMMs by using `meglm`. Because of the relationship between LMEs and GLMMs, there is insight to be gained through examination of the linear mixed model. This is especially true for Stata users because the terminology, syntax, options, and output for fitting these types of models are nearly identical. See `[ME] mixed` and the references therein, particularly in the Introduction, for more information.

Log-likelihood calculations for fitting any generalized mixed-effects model require integrating out the random effects. One widely used modern method is to directly estimate the integral required to calculate the log likelihood by Gauss–Hermite quadrature or some variation thereof. Because the log likelihood itself is estimated, this method has the advantage of permitting likelihood-ratio tests for comparing nested models. Also, if done correctly, quadrature approximations can be quite accurate, thus minimizing bias.

`meologit` supports three types of Gauss–Hermite quadrature and the Laplacian approximation method; see Methods and formulas of [ME] meglm for details.

Below we present two short examples of mixed-effects ordered logistic regression; refer to `[ME] melogit` for additional examples including crossed random-effects models and to [ME] me and [ME] meglm for examples of other random-effects models.

### Two-level models

We begin with a simple application of (1) as a two-level model, because a one-level model, in our terminology, is just standard ordered logistic regression; see [R] ologit.
Example 1

We use the data from the Television, School, and Family Smoking Prevention and Cessation Project (Flay et al. 1988; Rabe-Hesketh and Skrondal 2012, chap. 11), where schools were randomly assigned into one of four groups defined by two treatment variables. Students within each school are nested in classes, and classes are nested in schools. In this example, we ignore the variability of classes within schools and fit a two-level model; we incorporate classes in a three-level model in example 2. The dependent variable is the tobacco and health knowledge (THK) scale score collapsed into four ordered categories. We regress the outcome on the treatment variables and their interaction and control for the pretreatment score.

```
use http://www.stata-press.com/data/r14/tvsfpors
meologit thk prethk cc##tv || school:
```

```
Fitting fixed-effects model:
Iteration 0: log likelihood = -2212.775
Iteration 1: log likelihood = -2125.509
Iteration 2: log likelihood = -2125.1034
Iteration 3: log likelihood = -2125.1032
Refining starting values:
Grid node 0: log likelihood = -2136.2426
Fitting full model:
Iteration 0: log likelihood = -2136.2426 (not concave)
Iteration 1: log likelihood = -2120.2577
Iteration 2: log likelihood = -2119.7574
Iteration 3: log likelihood = -2119.7428
Iteration 4: log likelihood = -2119.7428
```

```
Mixed-effects ologit regression
Number of obs = 1,600
Group variable: school
Number of groups = 28
Obs per group:
    min = 18
    avg = 57.1
    max = 137
Integration method: mvaghermite
Integration pts. = 7
Wald chi2(4) = 128.06
Prob > chi2 = 0.0000
```

```
  | thk            | Coef. | Std. Err. | z     | P>|z|  | [95% Conf. Interval]
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>prethk</td>
<td>.4033</td>
<td>.0389</td>
<td>10.38</td>
<td>0.00</td>
<td>.3271 - .4795</td>
</tr>
<tr>
<td>1.cc</td>
<td>.9238</td>
<td>.2041</td>
<td>4.53</td>
<td>0.00</td>
<td>.5238 - 1.3238</td>
</tr>
<tr>
<td>1.tv</td>
<td>.2750</td>
<td>.1977</td>
<td>1.39</td>
<td>0.16</td>
<td>-.1126 - .6626</td>
</tr>
<tr>
<td>cc#tv 1 1</td>
<td>-.466</td>
<td>.2846</td>
<td>-1.64</td>
<td>0.10</td>
<td>-1.02 - .0919</td>
</tr>
<tr>
<td>/cut1</td>
<td>-.085</td>
<td>.1641</td>
<td>-0.54</td>
<td>0.59</td>
<td>-.4101 - .2331</td>
</tr>
<tr>
<td>/cut2</td>
<td>1.153</td>
<td>.1656</td>
<td>6.96</td>
<td>0.00</td>
<td>.8287 - 1.478</td>
</tr>
<tr>
<td>/cut3</td>
<td>2.332</td>
<td>.1734</td>
<td>13.45</td>
<td>0.00</td>
<td>1.9920 - 2.6718</td>
</tr>
<tr>
<td>school</td>
<td>.0735</td>
<td>.0383</td>
<td>1.90</td>
<td>0.31</td>
<td>.0065 - .1406</td>
</tr>
<tr>
<td>var(_cons)</td>
<td>.0267</td>
<td>.0204</td>
<td>1.28</td>
<td>0.20</td>
<td>.0065 - .0472</td>
</tr>
</tbody>
</table>
```

LR test vs. ologit model: chi2(01) = 10.72  Prob >= chi22 = 0.0005

Those of you familiar with the mixed command or other me commands will recognize the syntax and output. Below we comment on the items specific to ordered outcomes.
1. The estimation table reports the fixed effects, the estimated cutpoints \((\kappa_1, \kappa_2, \kappa_3)\), and the estimated variance components. The fixed effects can be interpreted just as you would the output from `ologit`. We find that students with higher preintervention scores tend to have higher postintervention scores. Because of their interaction, the impact of the treatment variables on the knowledge score is not straightforward; we defer this discussion to example 1 of [ME] `meologit postestimation`. You can also specify the `or` option at estimation or on replay to display the fixed effects as odds ratios instead.

2. Underneath the fixed effects and the cutpoints, the table shows the estimated variance components. The random-effects equation is labeled `school`, meaning that these are random effects at the `school` level. Because we have only one random effect at this level, the table shows only one variance component. The estimate of \(\sigma_u^2\) is 0.07 with standard error 0.04. The reported likelihood-ratio test shows that there is enough variability between schools to favor a mixed-effects ordered logistic regression over a standard ordered logistic regression; see `Distribution theory for likelihood-ratio test` in [ME] `me` for a discussion of likelihood-ratio testing of variance components.

We now store our estimates for later use.

```
. estimates store r_2
```

### Three-level models

Two-level models extend naturally to models with three or more levels with nested random effects. Below we continue with example 1.

**Example 2**

In this example, we fit a three-level model incorporating classes nested within schools as an additional level. The fixed-effects part remains the same.
. meologit thk prethk cc#tv || school: || class:

Fitting fixed-effects model:
Iteration 0:  log likelihood = -2212.775
Iteration 1:  log likelihood = -2125.509
Iteration 2:  log likelihood = -2125.1034
Iteration 3:  log likelihood = -2125.1032

Refining starting values:
Grid node 0:  log likelihood = -2152.1514

Fitting full model:
Iteration 0:  log likelihood = -2152.1514 (not concave)
Iteration 1:  log likelihood = -2125.9213 (not concave)
Iteration 2:  log likelihood = -2120.1861
Iteration 3:  log likelihood = -2115.6177
Iteration 4:  log likelihood = -2114.5896
Iteration 5:  log likelihood = -2114.5881
Iteration 6:  log likelihood = -2114.5881

Mixed-effects ologit regression

<table>
<thead>
<tr>
<th>Group Variable</th>
<th>No. of Groups</th>
<th>Observations per Group</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Minimum</td>
<td>Average</td>
</tr>
<tr>
<td>school</td>
<td>28</td>
<td>18</td>
</tr>
<tr>
<td>class</td>
<td>135</td>
<td>1</td>
</tr>
</tbody>
</table>

Integration method: mvaghermite
Wald chi2(4) = 124.39
Prob > chi2 = 0.0000

| thk         | Coef. | Std. Err. | z     | P>|z|  | [95% Conf. Interval] |
|-------------|-------|-----------|-------|------|----------------------|
| prethk      | .4085273 | .039616 | 10.31 | 0.000 | .3308814 .4861731    |
| 1.cc        | .8844369 | .2099124 | 4.21  | 0.000 | .4730161 1.295858    |
| 1.tv        | .236448  | .2049065 | 1.15  | 0.249 | -.1651614 .6380575  |
| cc#tv 1 1   | -.3717699 | .2958887 | -1.26 | 0.209 | -.951701 .2081612    |
| /cut1       | -.0959459 | .1688988 | -0.57 | 0.570 | -.4269815 .2350896  |
| /cut2       | 1.177478  | .1704946 | 6.91  | 0.000 | .8433151 1.511642    |
| /cut3       | 2.383672  | .1786376 | 13.34 | 0.000 | 2.033478 2.733865    |

<table>
<thead>
<tr>
<th>school</th>
<th>var(_cons)</th>
<th>.0448735</th>
<th>.0425387</th>
</tr>
</thead>
<tbody>
<tr>
<td>school&gt;class</td>
<td>var(_cons)</td>
<td>.1482157</td>
<td>.0637521</td>
</tr>
</tbody>
</table>

LR test vs. ologit model: chi2(2) = 21.03
Prob > chi2 = 0.0000

Note: LR test is conservative and provided only for reference.

Notes:
1. Our model now has two random-effects equations, separated by ||. The first is a random intercept (constant only) at the school level (level three), and the second is a random intercept at the class level (level two). The order in which these are specified (from left to right) is significant—meologit assumes that class is nested within school.
2. The information on groups is now displayed as a table, with one row for each grouping. You can suppress this table with the `nogroup` or the `noheader` option, which will suppress the rest of the header as well.

3. The variance-component estimates are now organized and labeled according to level. The variance component for `class` is labeled `school>class` to emphasize that classes are nested within schools.

   Compared with the two-level model from example 1, the estimate of the variance of the random intercept at the school level dropped from 0.07 to 0.04. This is not surprising because we now use two random components versus one random component to account for unobserved heterogeneity among students. We can use `lrtest` and our stored estimation result from example 1 to see which model provides a better fit:

   ```
   . lrtest r_2 .
   Likelihood-ratio test                      LR chi2(1) = 10.31
   (Assumption: r_2 nested in .)       Prob > chi2 = 0.0013

   Note: The reported degrees of freedom assumes the null hypothesis is not on
   the boundary of the parameter space. If this is not true, then the
   reported test is conservative.
   ```

   The likelihood-ratio test favors the three-level model. For more information about the likelihood-ratio test in the context of mixed-effects models, see `Distribution theory for likelihood-ratio test` in [ME] me.

   The above extends to models with more than two levels of nesting in the obvious manner, by adding more random-effects equations, each separated by `||`. The order of nesting goes from left to right as the groups go from biggest (highest level) to smallest (lowest level).

### Stored results

`meologit` stores the following in `e()`:

Scalars

- `e(N)` number of observations
- `e(k)` number of parameters
- `e(k_dv)` number of dependent variables
- `e(k_cat)` number of categories
- `e(k_eq)` number of equations in `e(b)`
- `e(k_eq_model)` number of equations in overall model test
- `e(k_f)` number of fixed-effects parameters
- `e(k_r)` number of random-effects parameters
- `e(k_rs)` number of variances
- `e(k_rc)` number of covariances
- `e(df_m)` model degrees of freedom
- `e(ll)` log likelihood
- `e(N_clust)` number of clusters
- `e(chi2)` $\chi^2$
- `e(p)` significance
- `e(ll_c)` log likelihood, comparison model
- `e(chi2_c)` $\chi^2$, comparison model
- `e(df_c)` degrees of freedom, comparison model
- `e(p_c)` significance, comparison model
- `e(rank)` rank of `e(V)`
- `e(ic)` number of iterations
- `e(rc)` return code
- `e(converged)` 1 if converged, 0 otherwise
Macros

e(cmd) meglm
e(cmd2) meologit
e(cmdline) command as typed
e(depvar) name of dependent variable
e(wtype) weight type
e(wexp) weight expression (first-level weights)
e(fweightk) fweight variable for kth highest level, if specified
e(iweightk) iweight variable for kth highest level, if specified
e(pweightk) pweight variable for kth highest level, if specified
e(covariates) list of covariates
e(ivars) grouping variables
e(model) ologit
e(title) title in estimation output
e(link) logit
e(family) ordinal
e(clustvar) name of cluster variable
e(offset) offset
e(intmethod) integration method
e(_quad) number of integration points
e(chi2type) Wald; type of model $\chi^2$
e(vce) vcetype specified in vce()
e(vcetype) title used to label Std. Err.
e(opt) type of optimization
e(which) max or min; whether optimizer is to perform maximization or minimization
e(ml_method) type of ml method
e(user) name of likelihood-evaluator program
e(technique) maximization technique
e(datasignature) the checksum
e(datasignaturevars) variables used in calculation of checksum
e(properties) b V

e(estat_cmd) program used to implement estat
e(predict) program used to implement predict
e(marginsnotok) predictions disallowed by margins
e(marginswtype) weight type for margins
e(marginswexp) weight expression for margins
e(marginsdefault) default predict() specification for margins

e(asbalanced) factor variables fvset as asbalanced
e(asobserved) factor variables fvset as asobserved

Matrices

e(b) coefficient vector
e(Cns) constraints matrix
e(ilog) iteration log (up to 20 iterations)
e(gradient) gradient vector

e(N_g) group counts

e(g_min) group-size minimums

e(g_avg) group-size averages

e(g_max) group-size maximums

e(V) variance–covariance matrix of the estimators

e(V_modelbased) model-based variance

Functions

e(sample) marks estimation sample
Methods and formulas

Without a loss of generality, consider a two-level ordered logistic model. The probability of observing outcome \( k \) for response \( y_{ij} \) is then

\[
p_{ij} = \Pr(y_{ij} = k | \kappa, u_j) = \Pr(\kappa_{k-1} < \eta_{ij} + \epsilon_{it} \leq \kappa_k)
\]

\[
= \frac{1}{1 + \exp(-\kappa_k + \eta_{ij})} - \frac{1}{1 + \exp(-\kappa_{k-1} + \eta_{ij})}
\]

where \( \eta_{ij} = x_{ij} \beta + z_{ij} u_j + \text{offset}_{ij} \), \( \kappa_0 \) is taken as \(-\infty\), and \( \kappa_K \) is taken as \( +\infty \). Here \( x_{ij} \) does not contain a constant term because its effect is absorbed into the cutpoints.

For cluster \( j \), \( j = 1, \ldots, M \), the conditional distribution of \( y_j = (y_{j1}, \ldots, y_{jn_j})' \) given a set of cluster-level random effects \( u_j \) is

\[
f(y_j | \kappa, u_j) = \prod_{i=1}^{n_j} p_{ij}^{I_k(y_{ij})}
\]

\[
= \exp \sum_{i=1}^{n_j} \left\{ I_k(y_{ij}) \log(p_{ij}) \right\}
\]

where

\[
I_k(y_{ij}) = \begin{cases} 1 & \text{if } y_{ij} = k \\ 0 & \text{otherwise} \end{cases}
\]

Because the prior distribution of \( u_j \) is multivariate normal with mean \( 0 \) and \( q \times q \) variance matrix \( \Sigma \), the likelihood contribution for the \( j \)th cluster is obtained by integrating \( u_j \) out of the joint density \( f(y_j, u_j) \),

\[
L_j(\beta, \kappa, \Sigma) = (2\pi)^{-q/2} |\Sigma|^{-1/2} \int f(y_j | \kappa, u_j) \exp\left(-u_j'\Sigma^{-1}u_j/2\right) \, du_j
\]

\[
= (2\pi)^{-q/2} |\Sigma|^{-1/2} \int \exp\{h(\beta, \kappa, \Sigma, u_j)\} \, du_j
\]

where

\[
h(\beta, \kappa, \Sigma, u_j) = \sum_{i=1}^{n_j} \left\{ I_k(y_{ij}) \log(p_{ij}) \right\} - u_j'\Sigma^{-1}u_j/2
\]

and for convenience, in the arguments of \( h(\cdot) \) we suppress the dependence on the observable data \((y_j, r_j, X_j, Z_j)\).

The integration in (2) has no closed form and thus must be approximated. meologit offers four approximation methods: mean–variance adaptive Gauss–Hermite quadrature (default unless a crossed random-effects model is fit), mode-curvature adaptive Gauss–Hermite quadrature, nonadaptive Gauss–Hermite quadrature, and Laplacian approximation (default for crossed random-effects models).

The Laplacian approximation is based on a second-order Taylor expansion of \( h(\beta, \kappa, \Sigma, u_j) \) about the value of \( u_j \) that maximizes it; see Methods and formulas in [ME] meglm for details.

Gaussian quadrature relies on transforming the multivariate integral in (2) into a set of nested univariate integrals. Each univariate integral can then be evaluated using a form of Gaussian quadrature; see Methods and formulas in [ME] meglm for details.
The log likelihood for the entire dataset is simply the sum of the contributions of the $M$ individual clusters, namely, $L(\beta, \kappa, \Sigma) = \sum_{j=1}^{M} L_j(\beta, \kappa, \Sigma)$.

Maximization of $L(\beta, \kappa, \Sigma)$ is performed with respect to $(\beta, \kappa, \sigma^2)$, where $\sigma^2$ is a vector comprising the unique elements of $\Sigma$. Parameter estimates are stored in $e(b)$ as $(\hat{\beta}, \hat{\kappa}, \hat{\sigma}^2)$, with the corresponding variance–covariance matrix stored in $e(V)$.

meologit supports multilevel weights and survey data; see Methods and formulas in [ME] meglm for details.

References


Also see

[ME] meologit postestimation — Postestimation tools for meologit
[ME] meoprobit — Multilevel mixed-effects ordered probit regression
[ME] me — Introduction to multilevel mixed-effects models
[SEM] intro 5 — Tour of models (Multilevel mixed-effects models)
[SVY] svy estimation — Estimation commands for survey data
[XT] xtologit — Random-effects ordered logistic models
[U] 20 Estimation and postestimation commands
Postestimation commands

The following postestimation command is of special interest after `meologit`:

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>estat group</code></td>
<td>summarize the composition of the nested groups</td>
</tr>
</tbody>
</table>

The following standard postestimation commands are also available:

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>contrast</code></td>
<td>contrasts and ANOVA-style joint tests of estimates</td>
</tr>
<tr>
<td><code>estat ic</code></td>
<td>Akaike’s and Schwarz’s Bayesian information criteria (AIC and BIC)</td>
</tr>
<tr>
<td><code>estat summarize</code></td>
<td>summary statistics for the estimation sample</td>
</tr>
<tr>
<td><code>estat vce</code></td>
<td>variance–covariance matrix of the estimators (VCE)</td>
</tr>
<tr>
<td><code>estat (svy)</code></td>
<td>postestimation statistics for survey data</td>
</tr>
<tr>
<td><code>estimates</code></td>
<td>cataloging estimation results</td>
</tr>
<tr>
<td><code>*hausman</code></td>
<td>Hausman’s specification test</td>
</tr>
<tr>
<td><code>lincom</code></td>
<td>point estimates, standard errors, testing, and inference for linear</td>
</tr>
<tr>
<td></td>
<td>combinations of coefficients</td>
</tr>
<tr>
<td><code>*lrtest</code></td>
<td>likelihood-ratio test</td>
</tr>
<tr>
<td><code>margins</code></td>
<td>marginal means, predictive margins, marginal effects, and average marginal</td>
</tr>
<tr>
<td></td>
<td>effects</td>
</tr>
<tr>
<td><code>marginsplot</code></td>
<td>graph the results from margins (profile plots, interaction plots, etc.)</td>
</tr>
<tr>
<td><code>nlcom</code></td>
<td>point estimates, standard errors, testing, and inference for nonlinear</td>
</tr>
<tr>
<td></td>
<td>combinations of coefficients</td>
</tr>
<tr>
<td><code>predict</code></td>
<td>predictions, residuals, influence statistics, and other diagnostic measures</td>
</tr>
<tr>
<td><code>predictnl</code></td>
<td>point estimates, standard errors, testing, and inference for generalized</td>
</tr>
<tr>
<td></td>
<td>predictions</td>
</tr>
<tr>
<td><code>pwcompare</code></td>
<td>pairwise comparisons of estimates</td>
</tr>
<tr>
<td><code>test</code></td>
<td>Wald tests of simple and composite linear hypotheses</td>
</tr>
<tr>
<td><code>testnl</code></td>
<td>Wald tests of nonlinear hypotheses</td>
</tr>
</tbody>
</table>

* `hausman` and `lrtest` are not appropriate with `svy` estimation results.
predict

Description for predict

predict creates a new variable containing predictions such as probabilities, linear predictions, density and distribution functions, and standard errors.

Menu for predict

Statistics > Postestimation

Syntax for predict

Syntax for obtaining predictions of the outcome and other statistics

predict [type] newvarsspec [if] [in] [ , statistic options ]

Syntax for obtaining estimated random effects and their standard errors

predict [type] newvarsspec [if] [in] , reffects [re_options]

Syntax for obtaining ML scores

predict [type] newvarsspec [if] [in] , scores

newvarsspec is stub* or newvarlist.

<table>
<thead>
<tr>
<th>statistic</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pr</td>
<td>predicted probabilities; the default</td>
</tr>
<tr>
<td>eta</td>
<td>fitted linear predictor</td>
</tr>
<tr>
<td>xb</td>
<td>linear predictor for the fixed portion of the model only</td>
</tr>
<tr>
<td>stdp</td>
<td>standard error of the fixed-portion linear prediction</td>
</tr>
<tr>
<td>density</td>
<td>predicted density function</td>
</tr>
<tr>
<td>distribution</td>
<td>predicted distribution function</td>
</tr>
</tbody>
</table>

These statistics are available both in and out of sample; type predict ... if e(sample) ... if wanted only for the estimation sample.
Options for predict

\underline{Main}

\texttt{pr}, the default, calculates the predicted probabilities.

You specify one or \(k\) new variables, where \(k\) is the number of categories of the dependent variable. If you specify the \texttt{outcome()} option, the probabilities will be predicted for the requested outcome only, in which case you specify only one new variable. If you specify one new variable and do not specify \texttt{outcome()}, \texttt{outcome(#1)} is assumed.

\texttt{eta}, \texttt{xb}, \texttt{stdp}, \texttt{density}, \texttt{distribution}, \texttt{scores}, \texttt{conditional()}, \texttt{marginal}, and \texttt{nooffset}; see [ME] \texttt{meglm postestimation}. 

---

<table>
<thead>
<tr>
<th>options</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>\texttt{conditional(ctype)}</td>
<td>compute statistic conditional on estimated random effects; default is \texttt{conditional(ebmeans)}</td>
</tr>
<tr>
<td>\texttt{marginal}</td>
<td>compute statistic marginally with respect to the random effects</td>
</tr>
<tr>
<td>\texttt{nooffset}</td>
<td>make calculation ignoring offset or exposure</td>
</tr>
<tr>
<td>\texttt{outcome(outcome)}</td>
<td>outcome category for predicted probabilities</td>
</tr>
</tbody>
</table>

\underline{Integration}

\textit{int\_options} integration options

You specify one or \(k\) new variables in \texttt{newvarlist} with \texttt{pr}, where \(k\) is the number of outcomes. If you do not specify \texttt{outcome()}, these options assume \texttt{outcome(#1)}.

\underline{ctype} Description

| \texttt{ebmeans}              | empirical Bayes means of random effects; the default                        |
| \texttt{ebmodes}              | empirical Bayes modes of random effects                                      |
| \texttt{fixedonly}            | prediction for the fixed portion of the model only                           |

\underline{re\_options} Description

| \texttt{ebmeans}              | use empirical Bayes means of random effects; the default                     |
| \texttt{ebmodes}              | use empirical Bayes modes of random effects                                  |
| \texttt{reses(stub* newvarlist)} | calculate standard errors of empirical Bayes estimates                       |

\underline{int\_options} integration options

<table>
<thead>
<tr>
<th>\textit{int_options}</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>\texttt{intpoints(#)}</td>
<td>use # quadrature points to compute marginal predictions and empirical Bayes means</td>
</tr>
<tr>
<td>\texttt{iterate(#)}</td>
<td>set maximum number of iterations in computing statistics involving empirical Bayes estimators</td>
</tr>
<tr>
<td>\texttt{tolerance(#)}</td>
<td>set convergence tolerance for computing statistics involving empirical Bayes estimators</td>
</tr>
</tbody>
</table>
outcome(*outcome*) specifies the outcome for which the predicted probabilities are to be calculated.

outcome() should contain either one value of the dependent variable or one of #1, #2, ..., with #1 meaning the first category of the dependent variable, #2 meaning the second category, etc.

reffects, ebmeans, ebmodes, and reses(), see [ME] meglm postestimation.

Integration: intpoints(), iterate(), tolerance(); see [ME] meglm postestimation.

margins

Description for margins

margins estimates margins of response for probabilities and linear predictions.

Menu for margins

Statistics > Postestimation

Syntax for margins

```
margins [ *marginlist* ] [ , options ]
margins [ *marginlist* ] , predict(statistic ... ) [ predict(statistic ... ) ... ] [ options ]
```

<table>
<thead>
<tr>
<th>statistic</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>default</td>
<td>probabilities for each outcome</td>
</tr>
<tr>
<td>pr</td>
<td>predicted probabilities for a specified outcome</td>
</tr>
<tr>
<td>eta</td>
<td>fitted linear predictor</td>
</tr>
<tr>
<td>xb</td>
<td>linear predictor for the fixed portion of the model only</td>
</tr>
<tr>
<td>stdp</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>density</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>distribution</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>reffects</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>scores</td>
<td>not allowed with margins</td>
</tr>
</tbody>
</table>

pr defaults to the first outcome.

Options conditional(ebmeans) and conditional(ebmodes) are not allowed with margins.

Option marginal is assumed where applicable if conditional(fixedonly) is not specified.

Statistics not allowed with margins are functions of stochastic quantities other than e(b).

For the full syntax, see [R] margins.
estat

Description for estat

   estat group reports the number of groups and minimum, average, and maximum group sizes for each level of the model. Model levels are identified by the corresponding group variable in the data. Because groups are treated as nested, the information in this summary may differ from what you would get if you used the `tabulate` command on each group variable individually.

Menu for estat

   Statistics > Postestimation

Syntax for estat

   estat group

Remarks and examples

Various predictions, statistics, and diagnostic measures are available after fitting an ordered logistic mixed-effects model with `meologit`. Here we show a short example of predicted probabilities and predicted random effects; refer to [ME] `meglm postestimation` for additional examples applicable to mixed-effects generalized linear models.

> Example 1

   In example 2 of [ME] `meologit`, we modeled the tobacco and health knowledge (`thk`) score—coded 1, 2, 3, 4—among students as a function of two treatments (`cc` and `tv`) by using a three-level ordered logistic model with random effects at the school and class levels.

   . use http://www.stata-press.com/data/r14/tvsfpors
   . meologit thk prethk cc##tv || school: || class:

      (output omitted)

   We obtain predicted probabilities for all four outcomes based on the contribution of both fixed effects and random effects by typing

   . predict pr*
      (predictions based on fixed effects and posterior means of random effects)
         (option `mu` assumed)
      (using 7 quadrature points)

   As the note says, the predicted values are based on the posterior means of random effects. You can use the `modes` option to obtain predictions based on the posterior modes of random effects.

   Because we specified a stub name, Stata saved the predicted random effects in variables `pr1` through `pr4`. Here we list the predicted probabilities for the first two classes for school 515:
For each observation, our best guess for the predicted outcome is the one with the highest predicted probability. For example, for the very first observation in the table above, we would choose outcome 4 as the most likely to occur.

We obtain predictions of the posterior means themselves at the school and class levels by typing

```
. predict re_s re_c, reffects
```

(calculating posterior means of random effects)

(using 7 quadrature points)

Here we list the predicted random effects for the first two classes for school 515:

```
. list class re_s re_c if school==515 & (class==515101 | class==515102), > sepby(class)
```

We can see that the predicted random effects at the school level (re_s) are the same for all classes and that the predicted random effects at the class level (re_c) are constant within each class.
Methods and formulas

Methods and formulas for predicting random effects and other statistics are given in Methods and formulas of [ME] meglm postestimation.

Also see

[ME] meologit — Multilevel mixed-effects ordered logistic regression
[ME] meglm postestimation — Postestimation tools for meglm
[U] 20 Estimation and postestimation commands
meoprobit — Multilevel mixed-effects ordered probit regression

Description

meoprobit fits mixed-effects probit models for ordered responses. The actual values taken on by the response are irrelevant except that larger values are assumed to correspond to “higher” outcomes. The conditional distribution of the response given the random effects is assumed to be multinomial, with success probability determined by the standard normal cumulative distribution function.

Quick start

Two-level ordered probit regression of $y$ on $x$ and random intercepts by lev2

```
meoprobit y x || lev2:
```

Add random coefficients for $x$

```
meoprobit y x || lev2: x
```

Nested three-level ordered probit model with random intercepts by lev2 and lev3 for lev2 nested within lev3

```
meoprobit y x || lev3: || lev2:
```

Menu

Statistics > Multilevel mixed-effects models > Ordered probit regression
**Syntax**

```plaintext
meoprobit depvar fe_equation [ || re_equation ] [ || re_equation ... ] [ , options ]
```

where the syntax of `fe_equation` is

```plaintext
[ indepvars ] [ if ] [ in ] [ weight ] [ , fe_options ]
```

and the syntax of `re_equation` is one of the following:

- for random coefficients and intercepts
  ```plaintext
  levelvar: [ varlist ] [ , re_options ]
  ```
- for random effects among the values of a factor variable
  ```plaintext
  levelvar: R.varname
  ```

`levelvar` is a variable identifying the group structure for the random effects at that level or is `all` representing one group comprising all observations.

**fe_options**

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>offset(varname)</code></td>
</tr>
</tbody>
</table>

**re_options**

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>covariance(vartype)</code></td>
</tr>
<tr>
<td><code>noconstant</code></td>
</tr>
<tr>
<td><code>fweight(varname)</code></td>
</tr>
<tr>
<td><code>iweight(varname)</code></td>
</tr>
<tr>
<td><code>pweight(varname)</code></td>
</tr>
</tbody>
</table>
### meoprobit — Multilevel mixed-effects ordered probit regression

**options**

<table>
<thead>
<tr>
<th>Model</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>constraints</strong> <em>(constraints)</em></td>
<td>apply specified linear constraints</td>
</tr>
<tr>
<td><strong>collinear</strong></td>
<td>keep collinear variables</td>
</tr>
</tbody>
</table>

**SE/Robust**

| vce(vctype) | vcetype may be oim, robust, or cluster clustvar |

**Reporting**

| level(#) | set confidence level; default is level(95) |
| noconsreport | do not display constraints |
| notable | suppress coefficient table |
| noheader | suppress output header |
| nogroup | suppress table summarizing groups |
| nolrtest | do not perform likelihood-ratio test comparing with ordered probit regression |

**display_options**

| control columns and column formats, row spacing, line width, display of omitted variables and base and empty cells, and factor-variable labeling |

**Integration**

| intmethod(intmethod) | integration method |
| intpoints(#) | set the number of integration (quadrature) points for all levels; default is intpoints(7) |

**Maximization**

| maximize_options | control the maximization process; seldom used |
| startvalues(svmethod) | method for obtaining starting values |
| startgrid[(gridspec)] | perform a grid search to improve starting values |
| noestimate | do not fit the model; show starting values instead |
| dnumerical | use numerical derivative techniques |
| coeflegend | display legend instead of statistics |

**vartype**

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>independent</td>
</tr>
<tr>
<td>exchangeable</td>
</tr>
<tr>
<td>identity</td>
</tr>
<tr>
<td>unstructured</td>
</tr>
<tr>
<td>fixed(matname)</td>
</tr>
<tr>
<td>pattern(matname)</td>
</tr>
</tbody>
</table>
### intmethod

<table>
<thead>
<tr>
<th>intmethod</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mvaghermite</td>
<td>mean–variance adaptive Gauss–Hermite quadrature; the default unless a crossed random-effects model is fit</td>
</tr>
<tr>
<td>mcaghermite</td>
<td>mode-curvature adaptive Gauss–Hermite quadrature</td>
</tr>
<tr>
<td>ghermite</td>
<td>nonadaptive Gauss–Hermite quadrature</td>
</tr>
<tr>
<td>laplace</td>
<td>Laplacian approximation; the default for crossed random-effects models</td>
</tr>
</tbody>
</table>

*indepvars* may contain factor variables; see [U] 11.4.3 Factor variables.

*depvar*, *indepvars*, and *varlist* may contain time-series operators; see [U] 11.4.4 Time-series varlists.

by and *svy* are allowed; see [U] 11.1.10 Prefix commands.

*vce()* and weights are not allowed with the *svy* prefix; see [SVY] svy.

*fweight*, *iweight*, and *pweight* are allowed; see [U] 11.1.6 weight. Only one type of weight may be specified.

Weights are not supported under the Laplacian approximation or for crossed models.

*startvalues()*-, *startgrid*,-, *noestimate*, *dnumerical*, and *coeflegend* do not appear in the dialog box.

See [U] 20 Estimation and postestimation commands for more capabilities of estimation commands.

### Options

- **Model**

  - **offset**(*varname*) specifies that *varname* be included in the fixed-effects portion of the model with the coefficient constrained to be 1.

  - **covariance**(*vartype*) specifies the structure of the covariance matrix for the random effects and may be specified for each random-effects equation. *vartype* is one of the following: independent, exchangeable, identity, unstructured, fixed(*matname*), or pattern(*matname*).

    - **covariance(independent)** covariance structure allows for a distinct variance for each random effect within a random-effects equation and assumes that all covariances are 0. The default is covariance(independent) unless a crossed random-effects model is fit, in which case the default is covariance(identity).

    - **covariance(exchangeable)** structure specifies one common variance for all random effects and one common pairwise covariance.

    - **covariance(identity)** is short for “multiple of the identity”; that is, all variances are equal and all covariances are 0.

    - **covariance(unstructured)** allows for all variances and covariances to be distinct. If an equation consists of *p* random-effects terms, the unstructured covariance matrix will have *p*(*p* + 1)/2 unique parameters.

    - **covariance(fixed(*matname*)) and covariance(pattern(*matname*))** covariance structures provide a convenient way to impose constraints on variances and covariances of random effects. Each specification requires a *matname* that defines the restrictions placed on variances and covariances. Only elements in the lower triangle of *matname* are used, and row and column names of *matname* are ignored. A missing value in *matname* means that a given element is unrestricted. In a fixed(*matname*) covariance structure, (co)variance (i, j) is constrained to equal the value specified in the i, jth entry of *matname*. In a pattern(*matname*) covariance structure, (co)variances (i, j) and (k, l) are constrained to be equal if *matname*[i, j] = *matname*[k, l].

  - **noconstant** suppresses the constant (intercept) term; may be specified for any of or all the random-effects equations.
fweight(varname) specifies frequency weights at higher levels in a multilevel model, whereas frequency weights at the first level (the observation level) are specified in the usual manner, for example, [fw=fwtvar1]. varname can be any valid Stata variable name, and you can specify fweight() at levels two and higher of a multilevel model. For example, in the two-level model

```
  . mecmd fixed_portion [fw = wt1] || school: ... , fweight(wt2) ...
```

the variable wt1 would hold the first-level (the observation-level) frequency weights, and wt2 would hold the second-level (the school-level) frequency weights.

iweight(varname) specifies importance weights at higher levels in a multilevel model, whereas importance weights at the first level (the observation level) are specified in the usual manner, for example, [iw=iwtvar1]. varname can be any valid Stata variable name, and you can specify iweight() at levels two and higher of a multilevel model. For example, in the two-level model

```
  . mecmd fixed_portion [iw = wt1] || school: ... , iweight(wt2) ...
```

the variable wt1 would hold the first-level (the observation-level) importance weights, and wt2 would hold the second-level (the school-level) importance weights.

pweight(varname) specifies sampling weights at higher levels in a multilevel model, whereas sampling weights at the first level (the observation level) are specified in the usual manner, for example, [pw=pwtvar1]. varname can be any valid Stata variable name, and you can specify pweight() at levels two and higher of a multilevel model. For example, in the two-level model

```
  . mecmd fixed_portion [pw = wt1] || school: ... , pweight(wt2) ...
```

variable wt1 would hold the first-level (the observation-level) sampling weights, and wt2 would hold the second-level (the school-level) sampling weights.

constraints(constraints), collinear; see [R] estimation options.
Integration

`intmethod(intmethod)` specifies the integration method to be used for the random-effects model. `mvaghermite` performs mean–variance adaptive Gauss–Hermite quadrature; `mcaghermite` performs mode-curvature adaptive Gauss–Hermite quadrature; `ghermite` performs nonadaptive Gauss–Hermite quadrature; and `laplace` performs the Laplacian approximation, equivalent to mode-curvature adaptive Gaussian quadrature with one integration point.

The default integration method is `mvaghermite` unless a crossed random-effects model is fit, in which case the default integration method is `laplace`. The Laplacian approximation has been known to produce biased parameter estimates; however, the bias tends to be more prominent in the estimates of the variance components rather than in the estimates of the fixed effects.

For crossed random-effects models, estimation with more than one quadrature point may be prohibitively intensive even for a small number of levels. For this reason, the integration method defaults to the Laplacian approximation. You may override this behavior by specifying a different integration method.

`intpoints(#)` sets the number of integration points for quadrature. The default is `intpoints(7)`, which means that seven quadrature points are used for each level of random effects. This option is not allowed with `intmethod(laplace)`.

The more integration points, the more accurate the approximation to the log likelihood. However, computation time increases as a function of the number of quadrature points raised to a power equaling the dimension of the random-effects specification. In crossed random-effects models and in models with many levels or many random coefficients, this increase can be substantial.

Maximization

`maximize_options: difficult, technique(algorithm_spec), iterate(#), [no]log, trace, gradient, showstep, hessian, showtolerance, tolerance(#), ltolerance(#), nrtolerance(#), nonrtolerance, and from(init_specs);` see `[R] maximize. Those that require special mention for `meoprobit` are listed below.

`from()` accepts a properly labeled vector of initial values or a list of coefficient names with values. A list of values is not allowed.

The following options are available with `meoprobit` but are not shown in the dialog box:

`startvalues(svmethod), startgrid[(gridspec)], noestimate, and dnumerical;` see `[ME] meglm.

`coeflegend;` see `[R] estimation options.

Remarks and examples

For a general introduction to `me` commands, see `[ME] me.`

`meoprobit` is a convenience command for `meglm` with a probit link and an ordinal family; see `[ME] meglm.

Remarks are presented under the following headings:

- Introduction
- Two-level models
- Three-level models
Introduction

Mixed-effects ordered probit regression is ordered probit regression containing both fixed effects and random effects. An ordered response is a variable that is categorical and ordered, for instance, “poor”, “good”, and “excellent”, which might indicate a person’s current health status or the repair record of a car. In the absence of random effects, mixed-effects ordered probit regression reduces to ordered probit regression; see \( \text{R oprobit} \).

Comprehensive treatments of mixed models are provided by, for example, Searle, Casella, and McCulloch (1992); Verbeke and Molenberghs (2000); Raudenbush and Bryk (2002); Demidenko (2004); Hedeker and Gibbons (2006); McCulloch, Searle, and Neuhaus (2008); and Rabe-Hesketh and Skrondal (2012). Agresti (2010, chap. 10) and Rabe-Hesketh and Skrondal (2012, chap. 11) are good introductory readings on applied multilevel modeling of ordinal data.

\text{meoprobit} allows for many levels of nested clusters of random effects. For example, in a three-level model you can specify random effects for schools and then random effects for classes nested within schools. In this model, the observations (presumably, the students) comprise the first level, the classes comprise the second level, and the schools comprise the third.

However, for simplicity, for now we consider the two-level model, where for a series of \( M \) independent clusters, and conditional on a set of fixed effects \( x_{ij} \), a set of cutpoints \( \kappa \), and a set of random effects \( u_j \), the cumulative probability of the response being in a category higher than \( k \) is

\[
\Pr(y_{ij} > k | x_{ij}, \kappa, u_j) = \Phi(x_{ij} \beta + z_{ij} u_j - \kappa_k)
\]

for \( j = 1, \ldots, M \) clusters, with cluster \( j \) consisting of \( i = 1, \ldots, n_j \) observations. The cutpoints are labeled \( \kappa_1, \kappa_2, \ldots, \kappa_{K-1} \), where \( K \) is the number of possible outcomes. \( \Phi(\cdot) \) is the standard normal cumulative distribution function that represents cumulative probability.

The \( 1 \times p \) row vector \( x_{ij} \) are the covariates for the fixed effects, analogous to the covariates you would find in a standard probit regression model, with regression coefficients (fixed effects) \( \beta \). In our parameterization, \( x_{ij} \) does not contain a constant term because its effect is absorbed into the cutpoints. For notational convenience here and throughout this manual entry, we suppress the dependence of \( y_{ij} \) on \( x_{ij} \).

The \( 1 \times q \) vector \( z_{ij} \) are the covariates corresponding to the random effects and can be used to represent both random intercepts and random coefficients. For example, in a random-intercept model, \( z_{ij} \) is simply the scalar 1. The random effects \( u_j \) are \( M \) realizations from a multivariate normal distribution with mean 0 and \( q \times q \) variance matrix \( \Sigma \). The random effects are not directly estimated as model parameters but are instead summarized according to the unique elements of \( \Sigma \), known as variance components. One special case of (1) places \( z_{ij} = x_{ij} \) so that all covariate effects are essentially random and distributed as multivariate normal with mean \( \beta \) and variance \( \Sigma \).

From (1), we can derive the probability of observing outcome \( k \) as

\[
\Pr(y_{ij} = k | \kappa, u_j) = \Pr(\kappa_{k-1} < x_{ij} \beta + z_{ij} u_j + \epsilon_{ij} \leq \kappa_k) = \Phi(\kappa_k - x_{ij} \beta - z_{ij} u_j) - \Phi(\kappa_{k-1} - x_{ij} \beta - z_{ij} u_j)
\]

where \( \kappa_0 \) is taken as \(-\infty\) and \( \kappa_K \) is taken as \(+\infty\).
From the above, we may also write the model in terms of a latent linear response, where observed ordinal responses \( y_{ij} \) are generated from the latent continuous responses, such that

\[
y^*_ij = x_{ij} \beta + z_{ij} u_j + \epsilon_{ij}
\]

and

\[
y_{ij} = \begin{cases} 
1 & \text{if} \quad y^*_ij \leq \kappa_1 \\
2 & \text{if} \quad \kappa_1 < y^*_ij \leq \kappa_2 \\
\vdots & \text{if} \quad \kappa_{K-1} < y^*_ij \\
K & \text{if} \quad \kappa_K < y^*_ij
\end{cases}
\]

The errors \( \epsilon_{ij} \) are distributed as standard normal with mean 0 and variance 1 and are independent of \( u_j \).

Model (1) is an example of a generalized linear mixed model (GLMM), which generalizes the linear mixed-effects (LME) model to non-Gaussian responses. You can fit LMEs in Stata by using `mixed` and GLMMs by using `meglm`. Because of the relationship between LMEs and GLMMs, there is insight to be gained through examination of the linear mixed model. This is especially true for Stata users because the terminology, syntax, options, and output for fitting these types of models are nearly identical. See [ME] `mixed` and the references therein, particularly in the Introduction, for more information.

Log-likelihood calculations for fitting any generalized mixed-effects model require integrating out the random effects. One widely used modern method is to directly estimate the integral required to calculate the log likelihood by Gauss–Hermite quadrature or some variation thereof. Because the log likelihood itself is estimated, this method has the advantage of permitting likelihood-ratio tests for comparing nested models. Also, if done correctly, quadrature approximations can be quite accurate, thus minimizing bias.

`meoprobit` supports three types of Gauss–Hermite quadrature and the Laplacian approximation method; see Methods and formulas of [ME] `meglm` for details.

Below we present two short examples of mixed-effects ordered probit regression; refer to [ME] `melogit` for additional examples including crossed random-effects models and to [ME] `me` and [ME] `meglm` for examples of other random-effects models.

### Two-level models

We begin with a simple application of (1) as a two-level model, because a one-level model, in our terminology, is just standard ordered probit regression; see [R] `oprobit`.

#### Example 1

We use the data from the Television, School, and Family Smoking Prevention and Cessation Project (Flay et al. 1988; Rabe-Hesketh and Skrondal 2012, chap. 11), where schools were randomly assigned into one of four groups defined by two treatment variables. Students within each school are nested in classes, and classes are nested in schools. In this example, we ignore the variability of classes within schools and fit a two-level model; we incorporate classes in a three-level model in example 2. The dependent variable is the tobacco and health knowledge (THK) scale score collapsed into four ordered categories. We regress the outcome on the treatment variables and their interaction and control for the pretreatment score.
. use http://www.stata-press.com/data/r14/tvsfpors
. meoprobit thk prethk cc##tv || school:

Fitting fixed-effects model:
Iteration 0: log likelihood = -2212.775
Iteration 1: log likelihood = -2127.8111
Iteration 2: log likelihood = -2127.7612
Iteration 3: log likelihood = -2127.7612

Refining starting values:
Grid node 0: log likelihood = -2149.7302

Fitting full model:
Iteration 0: log likelihood = -2149.7302 (not concave)
Iteration 1: log likelihood = -2129.6838 (not concave)
Iteration 2: log likelihood = -2123.5143
Iteration 3: log likelihood = -2122.2896
Iteration 4: log likelihood = -2121.7949
Iteration 5: log likelihood = -2121.7716
Iteration 6: log likelihood = -2121.7715

Mixed-effects oprobit regression
Group variable: school
Obs per group:
min = 18
avg = 57.1
max = 137
Integration method: mvaghermite
Integration pts. = 7
Wald chi2(4) = 128.05
Log likelihood = -2121.7715
Prob > chi2 = 0.0000

| Coef. | Std. Err. | z  | P>|z|  | [95% Conf. Interval] |
|-------|-----------|----|------|----------------------|
| thk   |           |    |      |                      |
| prethk| 0.2369804 | 0.0227739 | 10.41 | 0.000 | 0.1923444 | 0.2816164 |
| 1.cc  | 0.5490957 | 0.1255108 | 4.37  | 0.000 | 0.303099  | 0.7950923 |
| 1.tv  | 0.1695405 | 0.1215889 | 1.39  | 0.163 | -0.0687693| 0.4078504 |
| cc#tv |           |    |      |                      |
| 1 1   | -0.2951837| 0.1751969 | -1.68 | 0.092 | -0.6385634| 0.0481959 |
| /cut1 | 0.0682011 | 0.1003374 | -0.68 | 0.497 | -0.2648857| 0.1284565 |
| /cut2 | 0.67681   | 0.1008836 | 6.71  | 0.000 | 0.4790817 | 0.8745382 |
| /cut3 | 1.390649  | 0.1037494 | 13.40 | 0.000 | 1.187304 | 1.593995 |
| school|           |    |      |                      |
| var(_cons)| 0.0288527 | 0.0146201 | 0.0106874 | 0.0778937 |

LR test vs. oprobit model: chibar2(01) = 11.98 Prob >= chibar2 = 0.0003

Those of you familiar with the mixed command or other me commands will recognize the syntax and output. Below we comment on the items specific to ordered outcomes.

1. The estimation table reports the fixed effects, the estimated cutpoints \( (\kappa_1, \kappa_2, \kappa_3) \), and the estimated variance components. The fixed effects can be interpreted just as you would the output from oprobit. We find that students with higher preintervention scores tend to have higher postintervention scores. Because of their interaction, the impact of the treatment variables on the knowledge score is not straightforward; we defer this discussion to example 1 of [ME] meoprobit postestimation.
2. Underneath the fixed effects and the cutpoints, the table shows the estimated variance components. The random-effects equation is labeled school, meaning that these are random effects at the school level. Because we have only one random effect at this level, the table shows only one variance component. The estimate of $\sigma^2_u$ is 0.03 with standard error 0.01. The reported likelihood-ratio test shows that there is enough variability between schools to favor a mixed-effects ordered probit regression over a standard ordered probit regression; see Distribution theory for likelihood-ratio test in [ME] me for a discussion of likelihood-ratio testing of variance components.

We now store our estimates for later use.

```stata
    . estimates store r_2
```

### Three-level models

Two-level models extend naturally to models with three or more levels with nested random effects. Below we continue with example 1.

#### Example 2

In this example, we fit a three-level model incorporating classes nested within schools as an additional level. The fixed-effects part remains the same.
. meoprobit thk prethk cc#tv || school: || class:  

Fitting fixed-effects model:  
Iteration 0:  log likelihood = -2212.775  
Iteration 1:  log likelihood = -2127.811  
Iteration 2:  log likelihood = -2127.7612  
Iteration 3:  log likelihood = -2127.7612  

Refining starting values:  
Grid node 0:  log likelihood = -2195.6424  

Fitting full model:  
Iteration 0:  log likelihood = -2195.6424 (not concave)  
Iteration 1:  log likelihood = -2167.9576 (not concave)  
Iteration 2:  log likelihood = -2140.2644 (not concave)  
Iteration 3:  log likelihood = -2128.6948 (not concave)  
Iteration 4:  log likelihood = -2119.9225  
Iteration 5:  log likelihood = -2117.0947  
Iteration 6:  log likelihood = -2116.7004  
Iteration 7:  log likelihood = -2116.6981  
Iteration 8:  log likelihood = -2116.6981  

Mixed-effects oprobit regression  
Number of obs = 1,600  

<table>
<thead>
<tr>
<th>Group Variable</th>
<th>No. of Groups</th>
<th>Observations per Group</th>
<th>Minimum</th>
<th>Average</th>
<th>Maximum</th>
</tr>
</thead>
<tbody>
<tr>
<td>school</td>
<td>28</td>
<td>18</td>
<td>57.1</td>
<td>137</td>
<td></td>
</tr>
<tr>
<td>class</td>
<td>135</td>
<td>1</td>
<td>11.9</td>
<td>28</td>
<td></td>
</tr>
</tbody>
</table>

Integration method: mvaghermite  
Integration pts. = 7  
Log likelihood = -2116.6981  
Wald ch2(4) = 124.20  
Prob > ch2 = 0.0000  

| thk | Coef. | Std. Err. | z      | P>|z|  | [95% Conf. Interval] |
|-----|-------|-----------|--------|------|----------------------|
| prethk | .238841 | .0231446 | 10.32  | 0.000 | .1934784 .2842036 |
| 1.cc  | .5254813 | .1285816 | 4.09  | 0.000 | .2734659 .7774967 |
| 1.tv  | .1455573 | .1255827 | 1.16  | 0.246 | -.1005803 .3916949 |
| cc#tv 1 1 | -.2426203 | .1811999 | -1.34  | 0.181 | -.5977656 .1125251 |
| /cut1 | -.074617 | .1029791 | -0.72  | 0.469 | -.2764523 .1272184 |
| /cut2 | .6863046 | .1034813 | 6.63  | 0.000 | .4834849 .8891242 |
| /cut3 | 1.413686 | .1064889 | 13.28  | 0.000 | 1.204972 1.622401 |

| school | var(_cons) | .0186456 | .0160226 | .0034604 | .1004695 |
| school>class | var(_cons) | .0519974 | .0224014 | .0223496 | .1209745 |

LR test vs. oprobit model: ch2(2) = 22.13  
Prob > ch2 = 0.0000  

Notes:  
1. Our model now has two random-effects equations, separated by ||. The first is a random intercept (constant only) at the school level (level three), and the second is a random intercept at the class level (level two). The order in which these are specified (from left to right) is significant—meoprobit assumes that class is nested within school.
The information on groups is now displayed as a table, with one row for each grouping. You can suppress this table with the \texttt{nogroup} or the \texttt{noheader} option, which will suppress the rest of the header as well.

3. The variance-component estimates are now organized and labeled according to level. The variance component for \texttt{class} is labeled \texttt{school}>\texttt{class} to emphasize that classes are nested within schools.

Compared with the two-level model from example 1, the estimate of the random intercept at the school level dropped from 0.03 to 0.02. This is not surprising because we now use two random components versus one random component to account for unobserved heterogeneity among students. We can use \texttt{lrtest} and our stored estimation result from example 1 to see which model provides a better fit:

\begin{verbatim}
   . lrtest r_2 .
   Likelihood-ratio test               LR chi2(1) =    10.15
                   (Assumption: r_2 nested in .)   Prob > chi2 =    0.0014

Note: The reported degrees of freedom assumes the null hypothesis is not on
the boundary of the parameter space. If this is not true, then the
reported test is conservative.
\end{verbatim}

The likelihood-ratio test favors the three-level model. For more information about the likelihood-ratio test in the context of mixed-effects models, see \textit{Distribution theory for likelihood-ratio test} in \cite{me}.

The above extends to models with more than two levels of nesting in the obvious manner, by adding more random-effects equations, each separated by \texttt{||}. The order of nesting goes from left to right as the groups go from biggest (highest level) to smallest (lowest level).

\section*{Stored results}
\texttt{meoprobit} stores the following in \texttt{e()}:  

\begin{verbatim}
Scalars
   e(N)         number of observations
   e(k)         number of parameters
   e(k_dv)      number of dependent variables
   e(k_cat)     number of categories
   e(k_eq)      number of equations in \texttt{e(b)}
   e(k_eq_model)number of equations in overall model test
   e(k_f)       number of fixed-effects parameters
   e(k_r)       number of random-effects parameters
   e(k_rs)      number of variances
   e(k_rc)      number of covariances
   e(df_m)      model degrees of freedom
   e(ll)        log likelihood
   e(N_clust)   number of clusters
   e(chi2)      \( \chi^2 \)
   e(p)         significance
   e(ll_c)      log likelihood, comparison model
   e(chi2_c)    \( \chi^2 \), comparison model
   e(df_c)      degrees of freedom, comparison model
   e(p_c)       significance, comparison model
   e(rank)      rank of \texttt{e(V)}
   e(ic)        number of iterations
   e(rc)        return code
   e(converged) 1 if converged, 0 otherwise
\end{verbatim}
Macros

- `e(cmd)` meglm
- `e(cmd2)` meoprobit
- `e(cmdline)` command as typed
- `e(depvar)` name of dependent variable
- `e(wtype)` weight type
- `e(wexp)` weight expression (first-level weights)
- `e(fweight_k)` fweight variable for kth highest level, if specified
- `e(iweight_k)` iweight variable for kth highest level, if specified
- `e(pweight_k)` pweight variable for kth highest level, if specified
- `e(covariates)` list of covariates
- `e(ivars)` grouping variables
- `e(model)` oprobit
- `e(title)` title in estimation output
- `e(link)` probit
- `e(family)` ordinal
- `e(clustvar)` name of cluster variable
- `e(offset)` offset
- `e(intmethod)` integration method
- `e(_quad)` number of integration points
- `e(chi2type)` Wald; type of model $\chi^2$
- `e(vce)` vcetype specified in vce()
- `e(vcetype)` title used to label Std. Err.
- `e(opt)` type of optimization
- `e(which)` max or min; whether optimizer is to perform maximization or minimization
- `e(ml_method)` type of ml method
- `e(user)` name of likelihood-evaluator program
- `e(technique)` maximization technique
- `e(datasignature)` the checksum
- `e(datasignaturevars)` variables used in calculation of checksum
- `e(properties)` b V
- `e(estat_cmd)` program used to implement estat
- `e(predict)` program used to implement predict
- `e(marginsnotok)` predictions disallowed by margins
- `e(marginswtype)` weight type for margins
- `e(marginswexp)` weight expression for margins
- `e(marginsdefault)` default predict() specification for margins
- `e(asbalanced)` factor variables fvset as asbalanced
- `e(asobserved)` factor variables fvset as asobserved

Matrices

- `e(b)` coefficient vector
- `e(Cns)` constraints matrix
- `e(ilog)` iteration log (up to 20 iterations)
- `e(gradient)` gradient vector
- `e(N_g)` group counts
- `e(g_min)` group-size minimums
- `e(g_avg)` group-size averages
- `e(g_max)` group-size maximums
- `e(V)` variance–covariance matrix of the estimators
- `e(V_modelbased)` model-based variance

Functions

- `e(sample)` marks estimation sample
Methods and formulas

Without a loss of generality, consider a two-level ordered probit model. The probability of observing outcome \( k \) for response \( y_{ij} \) is then

\[
p_{ij} = \Pr(y_{ij} = k | \kappa, u_j) = \Pr(\kappa_{k-1} < \eta_{ij} + \epsilon_{it} \leq \kappa_k)
= \Phi(\kappa_k - \eta_{ij}) - \Phi(\kappa_{k-1} - \eta_{ij})
\]

where \( \eta_{ij} = x_{ij}\beta + z_{ij}u_j + \text{offset}_{ij} \), \( \kappa_0 \) is taken as \( -\infty \), and \( \kappa_K \) is taken as \( +\infty \). Here \( x_{ij} \) does not contain a constant term because its effect is absorbed into the cutpoints.

For cluster \( j, j = 1, \ldots, M \), the conditional distribution of \( y_j = (y_{j1}, \ldots, y_{jn_j})' \) given a set of cluster-level random effects \( u_j \) is

\[
f(y_j | u_j) = \prod_{i=1}^{n_j} p_{ij}(y_{ij})
= \exp \left\{ \sum_{i=1}^{n_j} \left\{ I_k(y_{ij}) \log(p_{ij}) \right\} \right\}
\]

where

\[
I_k(y_{ij}) = \begin{cases} 1 & \text{if } y_{ij} = k \\ 0 & \text{otherwise} \end{cases}
\]

Because the prior distribution of \( u_j \) is multivariate normal with mean \( 0 \) and \( q \times q \) variance matrix \( \Sigma \), the likelihood contribution for the \( j \)th cluster is obtained by integrating \( u_j \) out of the joint density \( f(y_j, u_j) \),

\[
L_j(\beta, \kappa, \Sigma) = (2\pi)^{-q/2} |\Sigma|^{-1/2} \int f(y_j | \kappa, u_j) \exp \left\{ -u_j'\Sigma^{-1}u_j/2 \right\} du_j
= (2\pi)^{-q/2} |\Sigma|^{-1/2} \int \exp \left\{ h(\beta, \kappa, \Sigma, u_j) \right\} du_j
\]

where

\[
h(\beta, \kappa, \Sigma, u_j) = \sum_{i=1}^{n_j} \left\{ I_k(y_{ij}) \log(p_{ij}) \right\} - u_j'\Sigma^{-1}u_j/2
\]

and for convenience, in the arguments of \( h(\cdot) \) we suppress the dependence on the observable data \((y_j, r_j, X_j, Z_j)\).

The integration in (2) has no closed form and thus must be approximated. \texttt{meoprobit} offers four approximation methods: mean–variance adaptive Gauss–Hermite quadrature (default unless a crossed random-effects model is fit), mode-curvature adaptive Gauss–Hermite quadrature, nonadaptive Gauss–Hermite quadrature, and Laplacian approximation (default for crossed random-effects models).

The Laplacian approximation is based on a second-order Taylor expansion of \( h(\beta, \kappa, \Sigma, u_j) \) about the value of \( u_j \) that maximizes it; see Methods and formulas in \texttt{ME} \texttt{meglm} for details.

Gaussian quadrature relies on transforming the multivariate integral in (2) into a set of nested univariate integrals. Each univariate integral can then be evaluated using a form of Gaussian quadrature; see Methods and formulas in \texttt{ME} \texttt{meglm} for details.

The log likelihood for the entire dataset is simply the sum of the contributions of the \( M \) individual clusters, namely, \( \mathcal{L}(\beta, \kappa, \Sigma) = \sum_{j=1}^{M} L_j(\beta, \kappa, \Sigma) \).
Maximization of $L(\beta, \kappa, \Sigma)$ is performed with respect to $(\beta, \kappa, \sigma^2)$, where $\sigma^2$ is a vector comprising the unique elements of $\Sigma$. Parameter estimates are stored in $e(b)$ as $(\hat{\beta}, \hat{\kappa}, \hat{\sigma}^2)$, with the corresponding variance–covariance matrix stored in $e(V)$.

meoprobit supports multilevel weights and survey data; see Methods and formulas in [ME] meglm for details.

References


Also see

[ME] meoprobit postestimation — Postestimation tools for meoprobit

[ME] meologit — Multilevel mixed-effects ordered logistic regression

[ME] me — Introduction to multilevel mixed-effects models

[SEM] intro 5 — Tour of models (*Multilevel mixed-effects models*)

[SVY] svy estimation — Estimation commands for survey data

[XT] xtoprobit — Random-effects ordered probit models

[U] 20 Estimation and postestimation commands
Postestimation commands

The following postestimation command is of special interest after `meoprobit`:

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>estat group</code></td>
<td>summarize the composition of the nested groups</td>
</tr>
</tbody>
</table>

The following standard postestimation commands are also available:

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>contrast</code></td>
<td>contrasts and ANOVA-style joint tests of estimates</td>
</tr>
<tr>
<td><code>estat ic</code></td>
<td>Akaike’s and Schwarz’s Bayesian information criteria (AIC and BIC)</td>
</tr>
<tr>
<td><code>estat summarize</code></td>
<td>summary statistics for the estimation sample</td>
</tr>
<tr>
<td><code>estat vce</code></td>
<td>variance–covariance matrix of the estimators (VCE)</td>
</tr>
<tr>
<td><code>estat (svy)</code></td>
<td>postestimation statistics for survey data</td>
</tr>
<tr>
<td><code>estimates</code></td>
<td>cataloging estimation results</td>
</tr>
<tr>
<td>* <code>hausman</code></td>
<td>Hausman’s specification test</td>
</tr>
<tr>
<td><code>lincom</code></td>
<td>point estimates, standard errors, testing, and inference for linear combinations of coefficients</td>
</tr>
<tr>
<td>* <code>lrtest</code></td>
<td>likelihood-ratio test</td>
</tr>
<tr>
<td><code>margins</code></td>
<td>marginal means, predictive margins, marginal effects, and average marginal effects</td>
</tr>
<tr>
<td><code>marginsplot</code></td>
<td>graph the results from margins (profile plots, interaction plots, etc.)</td>
</tr>
<tr>
<td><code>nlcom</code></td>
<td>point estimates, standard errors, testing, and inference for nonlinear combinations of coefficients</td>
</tr>
<tr>
<td><code>predict</code></td>
<td>predictions, residuals, influence statistics, and other diagnostic measures</td>
</tr>
<tr>
<td><code>predictnl</code></td>
<td>point estimates, standard errors, testing, and inference for generalized predictions</td>
</tr>
<tr>
<td><code>pwcompare</code></td>
<td>pairwise comparisons of estimates</td>
</tr>
<tr>
<td><code>test</code></td>
<td>Wald tests of simple and composite linear hypotheses</td>
</tr>
<tr>
<td><code>testnl</code></td>
<td>Wald tests of nonlinear hypotheses</td>
</tr>
</tbody>
</table>

* `hausman` and `lrtest` are not appropriate with `svy` estimation results.
predict

Description for predict

predict creates a new variable containing predictions such as probabilities, linear predictions, density and distribution functions, and standard errors.

Menu for predict

Statistics > Postestimation

Syntax for predict

Syntax for obtaining predictions of the outcome and other statistics

```
predict [type] newvarsspec [if] [in] [, statistic options]
```

Syntax for obtaining estimated random effects and their standard errors

```
predict [type] newvarsspec [if] [in], reffects [re_options]
```

Syntax for obtaining ML scores

```
predict [type] newvarsspec [if] [in], scores
```

`newvarsspec` is `stub*` or `newvarlist`.

<table>
<thead>
<tr>
<th>statistic</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pr</td>
<td>predicted probabilities; the default</td>
</tr>
<tr>
<td>eta</td>
<td>fitted linear predictor</td>
</tr>
<tr>
<td>xb</td>
<td>linear predictor for the fixed portion of the model only</td>
</tr>
<tr>
<td>stdp</td>
<td>standard error of the fixed-portion linear prediction</td>
</tr>
<tr>
<td>density</td>
<td>predicted density function</td>
</tr>
<tr>
<td>distribution</td>
<td>predicted distribution function</td>
</tr>
</tbody>
</table>

These statistics are available both in and out of sample; type `predict ... if e(sample) ...` if wanted only for the estimation sample.
meoprobit postestimation — Postestimation tools for meoprobit 199

**options**

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Main</strong></td>
</tr>
<tr>
<td><strong>conditional(ctype)</strong></td>
</tr>
<tr>
<td><strong>marginal</strong></td>
</tr>
<tr>
<td><strong>nooffset</strong></td>
</tr>
<tr>
<td><strong>outcome(outcome)</strong></td>
</tr>
<tr>
<td><strong>Integration</strong></td>
</tr>
<tr>
<td><strong>int_options</strong></td>
</tr>
<tr>
<td>You specify one or ( k ) new variables in <code>newvarlist</code> with <code>pr</code>, where ( k ) is the number of outcomes. If you do not specify <code>outcome()</code>, these options assume <code>outcome(#1)</code>.</td>
</tr>
</tbody>
</table>

**ctype**

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ebmeans</strong></td>
</tr>
<tr>
<td><strong>ebmodes</strong></td>
</tr>
<tr>
<td><strong>fixedonly</strong></td>
</tr>
</tbody>
</table>

**re_options**

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Main</strong></td>
</tr>
<tr>
<td><strong>ebmeans</strong></td>
</tr>
<tr>
<td><strong>ebmodes</strong></td>
</tr>
<tr>
<td><em><em>reses(stub</em> newvarlist)</em>*</td>
</tr>
<tr>
<td><strong>Integration</strong></td>
</tr>
<tr>
<td><strong>int_options</strong></td>
</tr>
</tbody>
</table>

**Integration**

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>int_options</strong></td>
</tr>
</tbody>
</table>

**int_options**

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>intpoints(#)</strong></td>
</tr>
<tr>
<td><strong>iterate(#)</strong></td>
</tr>
<tr>
<td><strong>tolerance(#)</strong></td>
</tr>
</tbody>
</table>

**Options for predict**

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>pr</strong>, the default, calculates the predicted probabilities.</td>
</tr>
<tr>
<td>You specify one or ( k ) new variables, where ( k ) is the number of categories of the dependent variable. If you specify the <code>outcome()</code> option, the probabilities will be predicted for the requested outcome only, in which case you specify only one new variable. If you specify one new variable and do not specify <code>outcome()</code>, <code>outcome(#1)</code> is assumed.</td>
</tr>
<tr>
<td><strong>eta, xb, stdp, density, distribution, scores, conditional(), marginal, and nooffset</strong>; see [ME] megglm postestimation.</td>
</tr>
</tbody>
</table>
outcome() specifies the outcome for which the predicted probabilities are to be calculated. outcome() should contain either one value of the dependent variable or one of #1, #2, ... with #1 meaning the first category of the dependent variable, #2 meaning the second category, etc.

reffects, emeans, ebmodes, and reses(), see [ME] meglm postestimation.

Integration
intpoints(), iterate(), tolerance(); see [ME] meglm postestimation.

margins

Description for margins
margins estimates margins of response for probabilities and linear predictions.

Menu for margins
Statistics > Postestimation

Syntax for margins

margins [marginlist] [ , options]
margins [marginlist], predict(statistic ...) [predict(statistic ...) ...] [options]

<table>
<thead>
<tr>
<th>statistic</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>default</td>
<td>probabilities for each outcome</td>
</tr>
<tr>
<td>pr</td>
<td>predicted probabilities for a specified outcome</td>
</tr>
<tr>
<td>eta</td>
<td>fitted linear predictor</td>
</tr>
<tr>
<td>xb</td>
<td>linear predictor for the fixed portion of the model only</td>
</tr>
<tr>
<td>stdp</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>density</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>distribution</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>reffects</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>scores</td>
<td>not allowed with margins</td>
</tr>
</tbody>
</table>

pr defaults to the first outcome.

Options conditional(emeans) and conditional(ebmodes) are not allowed with margins.

Option marginal is assumed where applicable if conditional(fixedonly) is not specified.

Statistics not allowed with margins are functions of stochastic quantities other than e(b).

For the full syntax, see [R] margins.
estat

Description for estat

estat group reports the number of groups and minimum, average, and maximum group sizes for each level of the model. Model levels are identified by the corresponding group variable in the data. Because groups are treated as nested, the information in this summary may differ from what you would get if you used the `tabulate` command on each group variable individually.

Menu for estat

Statistics > Postestimation

Syntax for estat

    estat group

Remarks and examples

Various predictions, statistics, and diagnostic measures are available after fitting an ordered probit mixed-effects model using `meoprobit`. Here we show a short example of predicted probabilities and predicted random effects; refer to [ME] `meglm postestimation` for additional examples applicable to mixed-effects generalized linear models.

Example 1

In example 2 of [ME] `meoprobit`, we modeled the tobacco and health knowledge (thk) score—coded 1, 2, 3, 4—among students as a function of two treatments (cc and tv) using a three-level ordered probit model with random effects at the school and class levels.

    . use http://www.stata-press.com/data/r14/tvsfpors
    . meoprobit thk prethk cc##tv || school: || class:
     (output omitted)

We obtain predicted probabilities for all four outcomes based on the contribution of both fixed effects and random effects by typing

    . predict pr*
     (predictions based on fixed effects and posterior means of random effects)
     (option `mu` assumed)
     (using 7 quadrature points)

As the note says, the predicted values are based on the posterior means of random effects. You can use the `modes` option to obtain predictions based on the posterior modes of random effects.

Because we specified a stub name, Stata saved the predicted random effects in variables `pr1` through `pr4`. Here we list the predicted probabilities for the first two classes for school 515:
For each observation, our best guess for the predicted outcome is the one with the highest predicted probability. For example, for the very first observation in the table above, we would choose outcome 4 as the most likely to occur.

We obtain predictions of the posterior means themselves at the school and class levels by typing

```
.predict re_s re_c, reffects
```

(calculating posterior means of random effects)

(using 7 quadrature points)

Here we list the predicted random effects for the first two classes for school 515:

```
.list class re_s re_c if school==515 & (class==515101 | class==515102), 
> sepby(class)
```

<table>
<thead>
<tr>
<th>class</th>
<th>re_s</th>
<th>re_c</th>
</tr>
</thead>
<tbody>
<tr>
<td>515101</td>
<td>-.0340769</td>
<td>.0390243</td>
</tr>
<tr>
<td>515102</td>
<td>-.0340769</td>
<td>-.0834322</td>
</tr>
</tbody>
</table>

We can see that the predicted random effects at the school level (re_s) are the same for all classes and that the predicted random effects at the class level (re_c) are constant within each class.
Methods and formulas

Methods and formulas for predicting random effects and other statistics are given in *Methods and formulas* of [ME] *meglm postestimation*.

Also see

[ME] *meoprobit* — Multilevel mixed-effects ordered probit regression
[ME] *meglm postestimation* — Postestimation tools for *meglm*
[U] 20 Estimation and postestimation commands
Description

depoisson fits mixed-effects models for count responses. The conditional distribution of the response given the random effects is assumed to be Poisson.
depoisson performs optimization with the original metric of variance components. When variance components are near the boundary of the parameter space, you may consider using the meqrpoisson command, which provides alternative parameterizations of variance components; see [ME] meqrpoisson.

Quick start

Without weights

Two-level Poisson regression of y on x with random intercepts by lev2
mepoisson y x || lev2:

Add evar measuring exposure
mepoisson y x, exposure(evar) || lev2:

As above, but report incidence-rate ratios
mepoisson y x, exposure(evar) || lev2:, irr

Add indicators for levels of categorical variable a and random coefficients on x
mepoisson y x i.a || lev2: x, irr

Three-level random-intercept model of y on x with lev2 nested within lev3
mepoisson y x || lev3: || lev2:

With weights

Two-level Poisson regression of y on x with random intercepts by lev2 and observation-level frequency weights wvar1
mepoisson y x [fweight=wvar1] || lev2:

Two-level random-intercept model from a two-stage sampling design with PSUs identified by psu using PSU-level and observation-level sampling weights wvar2 and wvar1, respectively
mepoisson y x [pweight=wvar1] || psu:, pweight(wvar2)

Add secondary sampling stage with units identified by ssu having weights wvar2 and PSU-level weights wvar3 for a three-level random-intercept model
mepoisson y x [pw=wvar1] || psu:, pw(wvar3) || ssu:, pw(wvar2)
Same as above, but *svyset* data first

```
svyset psu [pw=wvar3] || ssu, weight(wvar2) || _n, weight(wvar1)
svy: mepoisson y x || psu: || ssu:
```

Menu

Statistics > Multilevel mixed-effects models > Poisson regression

Syntax

```
mepoisson depvar fe_equation [ [ [ re_equation ] [ [ re_equation ... ] ] , options ] ]
```

where the syntax of `fe_equation` is

```
[ indepvars ] [ if ] [ in ] [ weight ] , fe_options
```

and the syntax of `re_equation` is one of the following:

- for random coefficients and intercepts
  
  `levelvar: [ varlist ] , re_options`

- for random effects among the values of a factor variable
  
  `levelvar: R.varname`

`levelvar` is a variable identifying the group structure for the random effects at that level or is `_all` representing one group comprising all observations.

### `fe_options`

<table>
<thead>
<tr>
<th>Description</th>
<th><code>noconstant</code></th>
<th><code>exposure(varname_e)</code></th>
<th><code>offset(varname_o)</code></th>
</tr>
</thead>
<tbody>
<tr>
<td>suppress the constant term from the fixed-effects equation</td>
<td>include ( \ln(varname_e) ) in model with coefficient constrained to 1</td>
<td>include ( varname_o ) in model with coefficient constrained to 1</td>
<td></td>
</tr>
</tbody>
</table>

### `re_options`

<table>
<thead>
<tr>
<th>Description</th>
<th><code>covariance(vartype)</code></th>
<th><code>noconstant</code></th>
<th><code>fweight(varname)</code></th>
<th><code>iweight(varname)</code></th>
<th><code>pweight(varname)</code></th>
</tr>
</thead>
<tbody>
<tr>
<td>variance–covariance structure of the random effects</td>
<td>suppress constant term from the random-effects equation</td>
<td>frequency weights at higher levels</td>
<td>importance weights at higher levels</td>
<td>sampling weights at higher levels</td>
<td></td>
</tr>
</tbody>
</table>
### options

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Model</strong></td>
</tr>
<tr>
<td><code>constraints(constraints)</code> apply specified linear constraints</td>
</tr>
<tr>
<td><code>collinear</code> keep collinear variables</td>
</tr>
<tr>
<td><strong>SE/Robust</strong></td>
</tr>
<tr>
<td><code>vce(vcetype)</code> <code>vcetype</code> may be oim, robust, or cluster clustvar</td>
</tr>
<tr>
<td><strong>Reporting</strong></td>
</tr>
<tr>
<td><code>level(#)</code> set confidence level; default is level(95)</td>
</tr>
<tr>
<td><code>irr</code> report fixed-effects coefficients as incidence-rate ratios</td>
</tr>
<tr>
<td><code>nocnsreport</code> do not display constraints</td>
</tr>
<tr>
<td><code>notable</code> suppress coefficient table</td>
</tr>
<tr>
<td><code>noheader</code> suppress output header</td>
</tr>
<tr>
<td><code>nogroup</code> suppress table summarizing groups</td>
</tr>
<tr>
<td><code>nolrtest</code> do not perform likelihood-ratio test comparing with Poisson regression</td>
</tr>
<tr>
<td><strong>display_options</strong> control columns and column formats, row spacing, line width, display of omitted variables and base and empty cells, and factor-variable labeling</td>
</tr>
<tr>
<td><strong>Integration</strong></td>
</tr>
<tr>
<td><code>intmethod(intmethod)</code> integration method</td>
</tr>
<tr>
<td><code>intpoints(#)</code> set the number of integration (quadrature) points for all levels; default is intpoints(7)</td>
</tr>
<tr>
<td><strong>Maximization</strong></td>
</tr>
<tr>
<td><code>maximize_options</code> control the maximization process; seldom used</td>
</tr>
<tr>
<td><code>startvalues(svmethod)</code> method for obtaining starting values</td>
</tr>
<tr>
<td><code>startgrid[ gridspec ]</code> perform a grid search to improve starting values</td>
</tr>
<tr>
<td><code>noestimate</code> do not fit the model; show starting values instead</td>
</tr>
<tr>
<td><code>dnnumerical</code> use numerical derivative techniques</td>
</tr>
<tr>
<td><code>coeflegend</code> display legend instead of statistics</td>
</tr>
</tbody>
</table>

### vartype

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>independent</code> one unique variance parameter per random effect, all covariances 0; the default unless the R. notation is used</td>
</tr>
<tr>
<td><code>exchangeable</code> equal variances for random effects, and one common pairwise covariance</td>
</tr>
<tr>
<td><code>identity</code> equal variances for random effects, all covariances 0; the default if the R. notation is used</td>
</tr>
<tr>
<td><code>unstructured</code> all variances and covariances to be distinctly estimated</td>
</tr>
<tr>
<td><code>fixed(matname)</code> user-selected variances and covariances constrained to specified values; the remaining variances and covariances unrestricted</td>
</tr>
<tr>
<td><code>pattern(matname)</code> user-selected variances and covariances constrained to be equal; the remaining variances and covariances unrestricted</td>
</tr>
</tbody>
</table>
### intmethod

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mvaghermite</td>
<td>mean–variance adaptive Gauss–Hermite quadrature; the default unless a crossed random-effects model is fit</td>
</tr>
<tr>
<td>mcaghermite</td>
<td>mode-curvature adaptive Gauss–Hermite quadrature</td>
</tr>
<tr>
<td>ghermite</td>
<td>nonadaptive Gauss–Hermite quadrature</td>
</tr>
<tr>
<td>laplace</td>
<td>Laplacian approximation; the default for crossed random-effects models</td>
</tr>
</tbody>
</table>

*indepvars* may contain factor variables; see [U] 11.4.3 Factor variables.

*depvar, indepvars, and varlist* may contain time-series operators; see [U] 11.4.4 Time-series varlists.

by and svy are allowed; see [U] 11.1.10 Prefix commands.

vce() and weights are not allowed with the svy prefix; see [SVY] svy.

fweights, iweights, and pweights are allowed; see [U] 11.1.6 weight. Only one type of weight may be specified.

startvalues(), startgrid, noestimate, dnumerical, and coeflegend do not appear in the dialog box.

See [U] 20 Estimation and postestimation commands for more capabilities of estimation commands.

### Options

#### Model

**noconstant** suppresses the constant (intercept) term and may be specified for the fixed-effects equation and for any of or all the random-effects equations.

**exposure**(*varname*) specifies a variable that reflects the amount of exposure over which the *depvar* events were observed for each observation; ln(*varname*) is included in the fixed-effects portion of the model with the coefficient constrained to be 1.

**offset**(*varname*) specifies that *varname* be included in the fixed-effects portion of the model with the coefficient constrained to be 1.

**covariance**(*vartype*) specifies the structure of the covariance matrix for the random effects and may be specified for each random-effects equation. *vartype* is one of the following: independent, exchangeable, identity, unstructured, fixed(matname), or pattern(matname).

**covariance(independent)** covariance structure allows for a distinct variance for each random effect within a random-effects equation and assumes that all covariances are 0. The default is covariance(independent) unless a crossed random-effects model is fit, in which case the default is covariance(identity).

**covariance(exchangeable)** structure specifies one common variance for all random effects and one common pairwise covariance.

**covariance(identity)** is short for “multiple of the identity”; that is, all variances are equal and all covariances are 0.

**covariance(unstructured)** allows for all variances and covariances to be distinct. If an equation consists of *p* random-effects terms, the unstructured covariance matrix will have *p*(*p* + 1)/2 unique parameters.

**covariance(fixed(matname))** and **covariance(pattern(matname))** covariance structures provide a convenient way to impose constraints on variances and covariances of random effects. Each specification requires a *matname* that defines the restrictions placed on variances and covariances. Only elements in the lower triangle of *matname* are used, and row and column names
of matname are ignored. A missing value in matname means that a given element is unrestricted.

In a fixed(matname) covariance structure, (co)variance \((i,j)\) is constrained to equal the value specified in the \(i,j\)th entry of matname. In a pattern(matname) covariance structure, (co)variances \((i,j)\) and \((k,l)\) are constrained to be equal if \(matname[i,j] = matname[k,l]\).

fweight(varname) specifies frequency weights at higher levels in a multilevel model, whereas frequency weights at the first level (the observation level) are specified in the usual manner, for example, \([fw=fwtvar1]\). varname can be any valid Stata variable name, and you can specify fweight() at levels two and higher of a multilevel model. For example, in the two-level model

```
. mecmd fixed_portion [fw = wt1] || school: ..., fweight(wt2) ...
```

the variable \(wt1\) would hold the first-level (the observation-level) frequency weights, and \(wt2\) would hold the second-level (the school-level) frequency weights.

iweight(varname) specifies importance weights at higher levels in a multilevel model, whereas importance weights at the first level (the observation level) are specified in the usual manner, for example, \([iw=iwtvar1]\). varname can be any valid Stata variable name, and you can specify iweight() at levels two and higher of a multilevel model. For example, in the two-level model

```
. mecmd fixed_portion [iw = wt1] || school: ..., iweight(wt2) ...
```

the variable \(wt1\) would hold the first-level (the observation-level) importance weights, and \(wt2\) would hold the second-level (the school-level) importance weights.

pweight(varname) specifies sampling weights at higher levels in a multilevel model, whereas sampling weights at the first level (the observation level) are specified in the usual manner, for example, \([pw=pwtvar1]\). varname can be any valid Stata variable name, and you can specify pweight() at levels two and higher of a multilevel model. For example, in the two-level model

```
. mecmd fixed_portion [pw = wt1] || school: ..., pweight(wt2) ...
```

variable \(wt1\) would hold the first-level (the observation-level) sampling weights, and \(wt2\) would hold the second-level (the school-level) sampling weights.

constraints(constraints), collinear; see [R] estimation options.

vce(vcetype) specifies the type of standard error reported, which includes types that are derived from asymptotic theory (oim), that are robust to some kinds of misspecification (robust), and that allow for intragroup correlation (cluster clustvar); see [R] vce_option. If vce(robust) is specified, robust variances are clustered at the highest level in the multilevel model.

Reporting

level(#) ; see [R] estimation options.

irr reports estimated fixed-effects coefficients transformed to incidence-rate ratios, that is, \(\exp(\beta)\) rather than \(\beta\). Standard errors and confidence intervals are similarly transformed. This option affects how results are displayed, not how they are estimated or stored. irr may be specified either at estimation or upon replay.

nocnsreport; see [R] estimation options.

notable suppresses the estimation table, either at estimation or upon replay.

noheader suppresses the output header, either at estimation or upon replay.

nogroup suppresses the display of group summary information (number of groups, average group size, minimum, and maximum) from the output header.
nolrtest prevents mepoisson from performing a likelihood-ratio test that compares the mixed-effects Poisson model with standard (marginal) Poisson regression. This option may also be specified upon replay to suppress this test from the output.

**display_options:** noci, nopvalues, noomitted, vsquish, noemptycells, baselevels, alllbaselevels, notol, fwrap(#), fwrapon(style), cformat(%,fmt), pformat(%,fmt), sformat(%,fmt), and nolstretch; see [R] estimation options.

**Integration**

`intmethod(intmethod)` specifies the integration method to be used for the random-effects model. `mvaghermite` performs mean–variance adaptive Gauss–Hermite quadrature; `mcaghermite` performs mode-curvature adaptive Gauss–Hermite quadrature; `ghermite` performs nonadaptive Gauss–Hermite quadrature; and `laplace` performs the Laplacian approximation, equivalent to mode-curvature adaptive Gaussian quadrature with one integration point.

The default integration method is `mvaghermite` unless a crossed random-effects model is fit, in which case the default integration method is `laplace`. The Laplacian approximation has been known to produce biased parameter estimates; however, the bias tends to be more prominent in the estimates of the variance components rather than in the estimates of the fixed effects.

For crossed random-effects models, estimation with more than one quadrature point may be prohibitively intensive even for a small number of levels. For this reason, the integration method defaults to the Laplacian approximation. You may override this behavior by specifying a different integration method.

`intpoints(#)` sets the number of integration points for quadrature. The default is `intpoints(7)`, which means that seven quadrature points are used for each level of random effects. This option is not allowed with `intmethod(laplace)`. The more integration points, the more accurate the approximation to the log likelihood. However, computation time increases as a function of the number of quadrature points raised to a power equaling the dimension of the random-effects specification. In crossed random-effects models and in models with many levels or many random coefficients, this increase can be substantial.

**Maximization**

`maximize_options:` difficult, technique(algorithm_spec), iterate(#), [no]log, trace, gradient, showstep, hessian, showtolerance, tolerance(#), itolerance(#), nrtolerance(#), nonrtolerance, and from(init_specs); see [R] maximize. Those that require special mention for mepoisson are listed below.

`from()` accepts a properly labeled vector of initial values or a list of coefficient names with values. A list of values is not allowed.

The following options are available with mepoisson but are not shown in the dialog box:

`startvalues(svmethod), startgrid[(gridspec)], noestimate, and dnumerical; see [ME] meglm.

coeflegend; see [R] estimation options.
Remarks and examples

For a general introduction to me commands, see [ME] me.

Remarks are presented under the following headings:

Introduction
A two-level model
A three-level model

Introduction

Mixed-effects Poisson regression is Poisson regression containing both fixed effects and random effects. In longitudinal data and panel data, random effects are useful for modeling intracluster correlation; that is, observations in the same cluster are correlated because they share common cluster-level random effects.

Comprehensive treatments of mixed models are provided by, for example, Searle, Casella, and McCulloch (1992); Verbeke and Molenberghs (2000); Raudenbush and Bryk (2002); Demidenko (2004); Hedeker and Gibbons (2006); McCulloch, Searle, and Neuhaus (2008); and Rabe-Hesketh and Skrondal (2012). Rabe-Hesketh and Skrondal (2012, chap. 13) is a good introductory read on applied multilevel modeling of count data.

mepoisson allows for not just one, but many levels of nested clusters. For example, in a three-level model you can specify random effects for schools and then random effects for classes nested within schools. In this model, the observations (presumably, the students) comprise the first level, the classes comprise the second level, and the schools comprise the third level.

However, for simplicity, for now we consider the two-level model, where for a series of \( M \) independent clusters, and conditional on a set of random effects \( u_j \),

\[
\Pr(y_{ij} = y|x_{ij}, u_j) = \exp(-\mu_{ij}) \frac{\mu_{ij}^y}{y!}
\]

for \( \mu_{ij} = \exp(x_{ij}\beta + z_{ij}u_j) \), \( j = 1, \ldots, M \) clusters, with cluster \( j \) consisting of \( i = 1, \ldots, n_j \) observations. The responses are counts \( y_{ij} \). The \( 1 \times p \) row vector \( x_{ij} \) are the covariates for the fixed effects, analogous to the covariates you would find in a standard Poisson regression model, with regression coefficients (fixed effects) \( \beta \). For notational convenience here and throughout this manual entry, we suppress the dependence of \( y_{ij} \) on \( x_{ij} \).

The \( 1 \times q \) vector \( z_{ij} \) are the covariates corresponding to the random effects and can be used to represent both random intercepts and random coefficients. For example, in a random-intercept model, \( z_{ij} \) is simply the scalar 1. The random effects \( u_j \) are \( M \) realizations from a multivariate normal distribution with mean 0 and \( q \times q \) variance matrix \( \Sigma \). The random effects are not directly estimated as model parameters but are instead summarized according to the unique elements of \( \Sigma \), known as variance components. One special case of (1) places \( z_{ij} = x_{ij} \) so that all covariate effects are essentially random and distributed as multivariate normal with mean \( \beta \) and variance \( \Sigma \).

As noted in chapter 13.7 of Rabe-Hesketh and Skrondal (2012), the inclusion of a random intercept causes the marginal variance of \( y_{ij} \) to be greater than the marginal mean, provided the variance of the random intercept is not 0. Thus the random intercept in a mixed-effects Poisson model produces overdispersion, a measure of variability above and beyond that allowed by a Poisson process; see [R] nbreg and [ME] membreg.

Model (1) is a member of the class of generalized linear mixed models (GLMMs), which generalize the linear mixed-effects (LME) model to non-Gaussian responses. You can fit LMEs in Stata by using mixed and fit GLMMs by using meglm. Because of the relationship between LMEs and GLMMs, there
is insight to be gained through examination of the linear mixed model. This is especially true for Stata users because the terminology, syntax, options, and output for fitting these types of models are nearly identical. See [ME] mixed and the references therein, particularly in the Introduction, for more information.

Log-likelihood calculations for fitting any generalized mixed-effects model require integrating out the random effects. One widely used modern method is to directly estimate the integral required to calculate the log likelihood by Gauss–Hermite quadrature or some variation thereof. Because the log likelihood itself is estimated, this method has the advantage of permitting likelihood-ratio tests for comparing nested models. Also, if done correctly, quadrature approximations can be quite accurate, thus minimizing bias.

mepoisson supports three types of Gauss–Hermite quadrature and the Laplacian approximation method; see Methods and formulas of [ME] meglm for details.

Below we present two short examples of mixed-effects Poisson regression; refer to [ME] me and [ME] meglm for additional examples including crossed random-effects models.

A two-level model

We begin with a simple application of (1) as a two-level model, because a one-level model, in our terminology, is just standard Poisson regression; see [R] poisson.

> Example 1

Breslow and Clayton (1993) fit a mixed-effects Poisson model to data from a randomized trial of the drug progabide for the treatment of epilepsy.

```
. use http://www.stata-press.com/data/r14/epilepsy
   (Epilepsy data; progabide drug treatment)
. describe
Contains data from http://www.stata-press.com/data/r14/epilepsy.dta
    obs:      236  Epilepsy data; progabide drug treatment
    vars:     8   31 May 2014 14:09
    size:   4,956    (_dta has notes)

  variable name   type    format     label                      variable label
      subject    byte     %9.0g    Subject ID: 1-59              Subject ID: 1-59
       seizures  int      %9.0g    No. of seizures               No. of seizures
        treat    byte     %9.0g    1: progabide; 0: placebo       Dr. visit; coded as (-.3, -.1,.1,.3)
        visit   float    %9.0g    log(age), mean-centered        log(0.25*baseline seizures),
       lage     float    %9.0g    log(0.25*baseline seizures),    mean-centered
       lbas     float    %9.0g    log(0.25*baseline seizures),    mean-centered
     lbas_trt  float    %9.0g    lbas/treat interaction          lbas/treat interaction
        v4      byte     %8.0g    Fourth visit indicator          Fourth visit indicator

```

Originally from Thall and Vail (1990), data were collected on 59 subjects (31 progabide, 28 placebo). The number of epileptic seizures (seizures) was recorded during the two weeks prior to each of four doctor visits (visit). The treatment group is identified by the indicator variable treat. Data were also collected on the logarithm of age (lage) and the logarithm of one-quarter the number of seizures during the eight weeks prior to the study (lbas). The variable lbas_trt represents the
interaction between lbas and treatment. lage, lbas, and lbas_trt are mean centered. Because the study originally noted a substantial decrease in seizures prior to the fourth doctor visit, an indicator v4 for the fourth visit was also recorded.

Breslow and Clayton (1993) fit a random-effects Poisson model for the number of observed seizures,

$$\log(\mu_{ij}) = \beta_0 + \beta_1 t_{reat_{ij}} + \beta_2 l_{bas_{ij}} + \beta_3 l_{bas_{trt_{ij}}} + \beta_4 l_{age_{ij}} + \beta_5 v_{4_{ij}} + u_j$$

for \(j = 1, \ldots, 59\) subjects and \(i = 1, \ldots, 4\) visits. The random effects \(u_j\) are assumed to be normally distributed with mean 0 and variance \(\sigma^2_u\).

```
. mepoisson seizures treat lbas lbas_trt lage v4 || subject:
```

Fitting fixed-effects model:
Iteration 0: log likelihood = -1016.4106
Iteration 1: log likelihood = -819.20112
Iteration 2: log likelihood = -817.66006
Iteration 3: log likelihood = -817.65925
Iteration 4: log likelihood = -817.65925

Refining starting values:
Grid node 0: log likelihood = -680.40523

Fitting full model:
Iteration 0: log likelihood = -680.40523 (not concave)
Iteration 1: log likelihood = -672.95766 (not concave)
Iteration 2: log likelihood = -667.14039
Iteration 3: log likelihood = -665.51823
Iteration 4: log likelihood = -665.29165
Iteration 5: log likelihood = -665.29067
Iteration 6: log likelihood = -665.29067

Mixed-effects Poisson regression
Number of obs = 236
Group variable: subject
Number of groups = 59

Obs per group:
min = 4
avg = 4.0
max = 4

Integration method: mvaghermite
Integration pts. = 7
Wald chi2(5) = 121.70
Log likelihood = -665.29067
Prob > chi2 = 0.0000

|               | Coef. | Std. Err. | z     | P>|z| | [95% Conf. Interval] |
|---------------|-------|-----------|-------|------|----------------------|
| treat         | -0.9330306 | 0.4007512 | -2.33 | 0.020 | -1.718489 -1.147527 |
| lbas          | 0.8844225 | 0.1312033 | 6.74  | 0.000 | 0.6272689 1.141576  |
| lbas_trt      | 0.3382561 | 0.2033021 | 1.66  | 0.096 | -0.0602087 0.736721 |
| lage          | 0.4842226 | 0.3471906 | 1.39  | 0.163 | -0.1962582 1.164703 |
| v4            | -0.1610871 | 0.0545758 | -2.95 | 0.003 | -0.2680536 -0.0541206 |
| _cons         | 2.154578  | 0.2199926 | 9.79  | 0.000 | 1.7234 2.585756    |

subject var(_cons) = 0.2528664 0.0589844 0.1600801 0.399434

LR test vs. Poisson model: chibar2(01) = 304.74

The number of seizures before the fourth visit does exhibit a significant drop, and the patients on progabide demonstrate a decrease in frequency of seizures compared with the placebo group. The subject-specific random effects also appear significant: \(\hat{\sigma}^2_u = 0.25\) with standard error 0.06.

Because this is a simple random-intercept model, you can obtain equivalent results by using xtpoisson with the re and normal options.
A three-level model

mepoisson can also fit higher-level models with multiple levels of nested random effects.

Example 2


```
. use http://www.stata-press.com/data/r14/melanoma
(Skin cancer (melanoma) data)
. describe
```

<table>
<thead>
<tr>
<th>variable</th>
<th>type</th>
<th>format</th>
<th>label</th>
</tr>
</thead>
<tbody>
<tr>
<td>nation</td>
<td>byte</td>
<td>%11.0g</td>
<td>n</td>
</tr>
<tr>
<td>region</td>
<td>byte</td>
<td>%9.0g</td>
<td>Region ID: EEC level-I areas</td>
</tr>
<tr>
<td>county</td>
<td>int</td>
<td>%9.0g</td>
<td>County ID: EEC level-II/level-III areas</td>
</tr>
<tr>
<td>deaths</td>
<td>int</td>
<td>%9.0g</td>
<td>No. deaths during 1971-1980</td>
</tr>
<tr>
<td>expected</td>
<td>float</td>
<td>%9.0g</td>
<td>No. expected deaths</td>
</tr>
<tr>
<td>uv</td>
<td>float</td>
<td>%9.0g</td>
<td>UV dose, mean-centered</td>
</tr>
</tbody>
</table>

```
Sorted by:

Nine European nations (variable `nation`) are represented, and data were collected over geographical regions defined by EEC statistical services as level I areas (variable `region`), with deaths being recorded for each of 354 counties, which are level II or level III EEC-defined areas (variable `county`, which identifies the observations). Counties are nested within regions, and regions are nested within nations.

The variable `deaths` records the number of deaths for each county, and `expected` records the expected number of deaths (the exposure) on the basis of crude rates for the combined countries. Finally, the variable `uv` is a measure of exposure to ultraviolet (UV) radiation.

In modeling the number of deaths, one possibility is to include dummy variables for the nine nations as fixed effects. Another is to treat these as random effects and fit the three-level random-intercept Poisson model,

$$
\log(\mu_{ijk}) = \log(\text{expected}_{ijk}) + \beta_0 + \beta_1 \text{uv}_{ijk} + u_k + v_{jk}
$$

for nation $k$, region $j$, and county $i$. The model includes an exposure term for expected deaths.
By including an exposure variable that is an expected rate, we are in effect specifying a linear model for the log of the standardized mortality ratio, the ratio of observed deaths to expected deaths that is based on a reference population, the reference population being all nine nations.
Looking at the estimated variance components, we can see that there is more unobserved variability between nations than between regions within each nation. This may be due to, for example, country-specific informational campaigns on the risks of sun exposure.

Stored results

mepoisson stores the following in e():

Scalars

\begin{itemize}
\item \texttt{e(N)}: number of observations
\item \texttt{e(k)}: number of parameters
\item \texttt{e(k\_dv)}: number of dependent variables
\item \texttt{e(k\_eq)}: number of equations in \texttt{e(b)}
\item \texttt{e(k\_eq\_model)}: number of equations in overall model test
\item \texttt{e(k\_f)}: number of fixed-effects parameters
\item \texttt{e(k\_r)}: number of random-effects parameters
\item \texttt{e(k\_rs)}: number of variances
\item \texttt{e(k\_rc)}: number of covariances
\item \texttt{e(df\_m)}: model degrees of freedom
\item \texttt{e(ll)}: log likelihood
\item \texttt{e(N\_clust)}: number of clusters
\item \texttt{e(chi2)}: \(\chi^2\)
\item \texttt{e(p)}: significance
\item \texttt{e(ll\_c)}: log likelihood, comparison model
\item \texttt{e(chi2\_c)}: \(\chi^2\), comparison model
\item \texttt{e(df\_c)}: degrees of freedom, comparison model
\item \texttt{e(p\_c)}: significance, comparison model
\item \texttt{e(rank)}: rank of \texttt{e(V)}
\item \texttt{e(ic)}: number of iterations
\item \texttt{e(rc)}: return code
\item \texttt{e(converged)}: 1 if converged, 0 otherwise
\end{itemize}

Macros

\begin{itemize}
\item \texttt{e(cmd)}: \texttt{meglm}
\item \texttt{e(cmd2)}: \texttt{mepoisson}
\item \texttt{e(cmdline)}: command as typed
\item \texttt{e(depvar)}: name of dependent variable
\item \texttt{e(wtype)}: weight type
\item \texttt{e(wexp)}: weight expression (first-level weights)
\item \texttt{e(fweight\_k)}: fweight variable for \texttt{k}th highest level, if specified
\item \texttt{e(iweight\_k)}: iweight variable for \texttt{k}th highest level, if specified
\item \texttt{e(pweight\_k)}: pweight variable for \texttt{k}th highest level, if specified
\item \texttt{e(covariates)}: list of covariates
\item \texttt{e(ivar)}: grouping variables
\item \texttt{e(model)}: poisson
\item \texttt{e(title)}: title in estimation output
\item \texttt{e(link)}: log
\item \texttt{e(family)}: poisson
\item \texttt{e(clustvar)}: name of cluster variable
\item \texttt{e(offset)}: offset
\item \texttt{e(exposure)}: exposure variable
\item \texttt{e(intmethod)}: integration method
\item \texttt{e(n\_quad)}: number of integration points
\item \texttt{e(chi2type)}: Wald; type of model \(\chi^2\)
\item \texttt{e(vce)}: vcetype specified in \texttt{vce()}
\item \texttt{e(vcetype)}: title used to label Std. Err.
\item \texttt{e(opt)}: type of optimization
\item \texttt{e(which)}: max or min; whether optimizer is to perform maximization or minimization
\item \texttt{e(ml\_method)}: type of ml method
Methods and formulas

In a two-level Poisson model, for cluster $j$, $j = 1, \ldots, M$, the conditional distribution of $y_j = (y_{j1}, \ldots, y_{jn_j})'$, given a set of cluster-level random effects $u_j$, is

$$f(y_j | u_j) = \prod_{i=1}^{n_j} \left[ \exp \left( x_{ij}^T \beta + z_{ij} u_j \right) \right]^{y_{ij}} \exp \left\{ - \exp \left( x_{ij}^T \beta + z_{ij} u_j \right) \right\} / y_{ij}!$$

$$= \exp \left[ \sum_{i=1}^{n_j} \left( y_{ij} (x_{ij}^T \beta + z_{ij} u_j) - \exp (x_{ij}^T \beta + z_{ij} u_j) - \log(y_{ij}!) \right) \right]$$

Defining $c(y_j) = \sum_{i=1}^{n_j} \log(y_{ij}!)$, where $c(y_j)$ does not depend on the model parameters, we can express the above compactly in matrix parameters, we can express the above compactly in matrix notation,

$$f(y_j | u_j) = \exp \left\{ y_j^T (X_j \beta + Z_j u_j) - 1^T \exp (X_j \beta + Z_j u_j) - c(y_j) \right\}$$

where $X_j$ is formed by stacking the row vectors $x_{ij}$ and $Z_j$ is formed by stacking the row vectors $z_{ij}$. We extend the definition of $\exp(\cdot)$ to be a vector function where necessary.

Because the prior distribution of $u_j$ is multivariate normal with mean $0$ and $q \times q$ variance matrix $\Sigma$, the likelihood contribution for the $j$th cluster is obtained by integrating $u_j$ out of the joint density $f(y_j, u_j)$,

$$L_j(\beta, \Sigma) = (2\pi)^{-q/2} |\Sigma|^{-1/2} \int f(y_j | u_j) \exp \left\{ -u_j^T \Sigma^{-1} u_j / 2 \right\} du_j$$

$$= \exp \left\{ -c(y_j) \right\} (2\pi)^{-q/2} |\Sigma|^{-1/2} \int \exp \left\{ h(\beta, \Sigma, u_j) \right\} du_j$$

$$= \exp \left\{ -c(y_j) \right\} (2\pi)^{-q/2} |\Sigma|^{-1/2} \int \exp \left\{ h(\beta, \Sigma, u_j) \right\} du_j$$

$$= \exp \left\{ -c(y_j) \right\} (2\pi)^{-q/2} |\Sigma|^{-1/2} \int \exp \left\{ h(\beta, \Sigma, u_j) \right\} du_j$$
where
\[ h(\beta, \Sigma, u_j) = y'_j(X_j\beta + Z_ju_j) - 1' \exp(X_j\beta + Z_ju_j) - u'_j\Sigma^{-1}u_j/2 \]
and for convenience, in the arguments of \( h(\cdot) \) we suppress the dependence on the observable data \((y_j, X_j, Z_j)\).

The integration in (2) has no closed form and thus must be approximated. `mepoisson` offers four approximation methods: mean–variance adaptive Gauss–Hermite quadrature (default unless a crossed random-effects model is fit), mode-curvature adaptive Gauss–Hermite quadrature, nonadaptive Gauss–Hermite quadrature, and Laplacian approximation (default for crossed random-effects models).

The Laplacian approximation is based on a second-order Taylor expansion of \( g(\beta, \Sigma, u_j) \) about the value of \( u_j \) that maximizes it; see *Methods and formulas* in [ME] _meglm_ for details.

Gaussian quadrature relies on transforming the multivariate integral in (2) into a set of nested univariate integrals. Each univariate integral can then be evaluated using a form of Gaussian quadrature; see *Methods and formulas* in [ME] _meglm_ for details.

The log likelihood for the entire dataset is simply the sum of the contributions of the \( M \) individual clusters, namely, \( \mathcal{L}(\beta, \Sigma) = \sum_{j=1}^{M} \mathcal{L}_j(\beta, \Sigma) \).

Maximization of \( \mathcal{L}(\beta, \Sigma) \) is performed with respect to \((\beta, \sigma^2)\), where \( \sigma^2 \) is a vector comprising the unique elements of \( \Sigma \). Parameter estimates are stored in \( e(b) \) as \((\hat{\beta}, \hat{\sigma}^2)\), with the corresponding variance–covariance matrix stored in \( e(V) \).

`mepoisson` supports multilevel weights and survey data; see *Methods and formulas* in [ME] _meglm_ for details.

References


Also see

[ME] mepoisson postestimation — Postestimation tools for mepoisson

[ME] membreg — Multilevel mixed-effects negative binomial regression

[ME] meqrpoisson — Multilevel mixed-effects Poisson regression (QR decomposition)

[ME] me — Introduction to multilevel mixed-effects models

[SEM] intro 5 — Tour of models (Multilevel mixed-effects models)

[SVY] svy estimation — Estimation commands for survey data

[XT] xtpoisson — Fixed-effects, random-effects, and population-averaged Poisson models

[U] 20 Estimation and postestimation commands
mepoisson postestimation — Postestimation tools for mepoisson

**Postestimation commands**

The following postestimation command is of special interest after mepoisson:

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>estat group</td>
<td>summarize the composition of the nested groups</td>
</tr>
</tbody>
</table>

The following standard postestimation commands are also available:

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>contrast</td>
<td>contrasts and ANOVA-style joint tests of estimates</td>
</tr>
<tr>
<td>estat ic</td>
<td>Akaike’s and Schwarz’s Bayesian information criteria (AIC and BIC)</td>
</tr>
<tr>
<td>estat summarize</td>
<td>summary statistics for the estimation sample</td>
</tr>
<tr>
<td>estat vce</td>
<td>variance–covariance matrix of the estimators (VCE)</td>
</tr>
<tr>
<td>estat (svy)</td>
<td>postestimation statistics for survey data</td>
</tr>
<tr>
<td>estimates</td>
<td>cataloging estimation results</td>
</tr>
<tr>
<td>*hausman</td>
<td>Hausman’s specification test</td>
</tr>
<tr>
<td>lincom</td>
<td>point estimates, standard errors, testing, and inference for linear</td>
</tr>
<tr>
<td></td>
<td>combinations of coefficients</td>
</tr>
<tr>
<td>*lrtest</td>
<td>likelihood-ratio test</td>
</tr>
<tr>
<td>margins</td>
<td>marginal means, predictive margins, marginal effects, and average marginal</td>
</tr>
<tr>
<td></td>
<td>effects</td>
</tr>
<tr>
<td>marginsplot</td>
<td>graph the results from margins (profile plots, interaction plots, etc.)</td>
</tr>
<tr>
<td>nlcom</td>
<td>point estimates, standard errors, testing, and inference for nonlinear</td>
</tr>
<tr>
<td></td>
<td>combinations of coefficients</td>
</tr>
<tr>
<td>predict</td>
<td>predictions, residuals, influence statistics, and other diagnostic measures</td>
</tr>
<tr>
<td>predictnl</td>
<td>point estimates, standard errors, testing, and inference for generalized</td>
</tr>
<tr>
<td></td>
<td>predictions</td>
</tr>
<tr>
<td>pwcompare</td>
<td>pairwise comparisons of estimates</td>
</tr>
<tr>
<td>test</td>
<td>Wald tests of simple and composite linear hypotheses</td>
</tr>
<tr>
<td>testnl</td>
<td>Wald tests of nonlinear hypotheses</td>
</tr>
</tbody>
</table>

* hausman and lrtest are not appropriate with svy estimation results.
predict

Description for predict

predict creates a new variable containing predictions such as mean responses; linear predictions; density and distribution functions; standard errors; and Pearson, deviance, and Anscombe residuals.

Menu for predict

Statistics > Postestimation

Syntax for predict

Syntax for obtaining predictions of the outcome and other statistics

```
predict [type] newvarsspec [if] [in] [, statistic options]
```

Syntax for obtaining estimated random effects and their standard errors

```
predict [type] newvarsspec [if] [in], reffects [re_options]
```

Syntax for obtaining ML scores

```
predict [type] newvarsspec [if] [in], scores
```

newvarsspec is stub* or newvarlist.

<table>
<thead>
<tr>
<th>statistic</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mu</td>
<td>mean response; the default</td>
</tr>
<tr>
<td>eta</td>
<td>fitted linear predictor</td>
</tr>
<tr>
<td>xb</td>
<td>linear predictor for the fixed portion of the model only</td>
</tr>
<tr>
<td>stdp</td>
<td>standard error of the fixed-portion linear prediction</td>
</tr>
<tr>
<td>density</td>
<td>predicted density function</td>
</tr>
<tr>
<td>distribution</td>
<td>predicted distribution function</td>
</tr>
<tr>
<td>pearson</td>
<td>Pearson residuals</td>
</tr>
<tr>
<td>deviance</td>
<td>deviance residuals</td>
</tr>
<tr>
<td>anscombe</td>
<td>Anscombe residuals</td>
</tr>
</tbody>
</table>

These statistics are available both in and out of sample; type `predict ... if e(sample) ...` if wanted only for the estimation sample.
**options**

<table>
<thead>
<tr>
<th>Main</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>conditional(ctype)</code></td>
<td>compute statistic conditional on estimated random effects; default is conditional(ebmeans)</td>
</tr>
<tr>
<td><code>marginal</code></td>
<td>compute statistic marginally with respect to the random effects</td>
</tr>
<tr>
<td><code>nooffset</code></td>
<td>make calculation ignoring offset or exposure</td>
</tr>
</tbody>
</table>

**Integration**

<table>
<thead>
<tr>
<th><code>-options</code></th>
<th>integration options</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>pearson</code>, <code>deviance</code>, <code>anscombe</code></td>
<td>may not be combined with <code>marginal</code>.</td>
</tr>
</tbody>
</table>

**ctype**

| `ebmeans`                | empirical Bayes means of random effects; the default                        |
| `ebmodes`                | empirical Bayes modes of random effects                                     |
| `fixedonly`              | prediction for the fixed portion of the model only                           |

**re_options**

<table>
<thead>
<tr>
<th>Main</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ebmeans</code></td>
<td>use empirical Bayes means of random effects; the default</td>
</tr>
<tr>
<td><code>ebmodes</code></td>
<td>use empirical Bayes modes of random effects</td>
</tr>
<tr>
<td>`reses(stub*</td>
<td>newvarlist)`</td>
</tr>
</tbody>
</table>

**Integration**

<table>
<thead>
<tr>
<th><code>-options</code></th>
<th>integration options</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>intpoints(#)</code></td>
<td>use # quadrature points to compute marginal predictions and empirical Bayes means</td>
</tr>
<tr>
<td><code>iterate(#)</code></td>
<td>set maximum number of iterations in computing statistics involving empirical Bayes estimators</td>
</tr>
<tr>
<td><code>tolerance(#)</code></td>
<td>set convergence tolerance for computing statistics involving empirical Bayes estimators</td>
</tr>
</tbody>
</table>

**Options for predict**

- **Main**
  - `mu`, the default, calculates the predicted mean, that is, the predicted number of events.
  - `eta`, `xb`, `stdp`, `density`, `distribution`, `pearson`, `deviance`, `anscombe`, `scores`, `conditional()`, `marginal`, and `nooffset`; see [ME] `meglm postestimation`.
  - `reffects`, `ebmeans`, `ebmodes`, and `reses()`; see [ME] `meglm postestimation`.

- **Integration**
  - `intpoints()`, `iterate()`, and `tolerance()`; see [ME] `meglm postestimation`. 
margins

Description for margins

margins estimates margins of response for mean responses and linear predictions.

Menu for margins

Statistics > Postestimation

Syntax for margins

margins [marginlist] [ , options ]
margins [marginlist] , predict(statistic ...) [ predict(statistic ...) ...] [ options ]

<table>
<thead>
<tr>
<th>statistic</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mu</td>
<td>mean response; the default</td>
</tr>
<tr>
<td>eta</td>
<td>fitted linear predictor</td>
</tr>
<tr>
<td>xb</td>
<td>linear predictor for the fixed portion of the model only</td>
</tr>
<tr>
<td>stdp</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>density</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>distribution</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>pearson</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>deviance</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>anscombe</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>reffects</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>scores</td>
<td>not allowed with margins</td>
</tr>
</tbody>
</table>

Statistics not allowed with margins are functions of stochastic quantities other than e(b).

For the full syntax, see [R] margins.
estat

Description for estat

estat group reports the number of groups and minimum, average, and maximum group sizes for each level of the model. Model levels are identified by the corresponding group variable in the data. Because groups are treated as nested, the information in this summary may differ from what you would get if you used the `tabulate` command on each group variable individually.

Menu for estat

Statistics  >  Postestimation

Syntax for estat

    estat group

Remarks and examples

Various predictions, statistics, and diagnostic measures are available after fitting a mixed-effects Poisson model with mepoisson. For the most part, calculation centers around obtaining estimates of the subject/group-specific random effects. Random effects are not estimated when the model is fit but instead need to be predicted after estimation.

Here we show a short example of predicted counts and predicted random effects; refer to [ME] meglm postestimation for additional examples applicable to mixed-effects generalized linear models.

Example 1

In example 2 of [ME] mepoisson, we modeled the number of deaths among males in nine European nations as a function of exposure to ultraviolet radiation (uv). We used a three-level Poisson model with random effects at the nation and region levels.
We can use `predict` to obtain the predicted counts as well as the estimates of the random effects at the nation and region levels.

```stata
.predict mu
(predictions based on fixed effects and posterior means of random effects)
(option mu assumed)
(using 7 quadrature points)
.predict re_nat re_reg, reffects
(calculating posterior means of random effects)
(using 7 quadrature points)
```

Stata displays a note that the predicted values of `mu` are based on the posterior means of random effects. You can use option `modes` to obtain predictions based on the posterior modes of random effects.

Here we list the data for the first nation in the dataset, which happens to be Belgium:

```stata
.list nation region deaths mu re_nat re_reg if nation==1, sepby(region)
```

<table>
<thead>
<tr>
<th>nation</th>
<th>region</th>
<th>deaths</th>
<th>mu</th>
<th>re_nat</th>
<th>re_reg</th>
</tr>
</thead>
<tbody>
<tr>
<td>Belgium</td>
<td>1</td>
<td>79</td>
<td>69.17982</td>
<td>-.123059</td>
<td>.3604518</td>
</tr>
<tr>
<td>Belgium</td>
<td>2</td>
<td>80</td>
<td>78.14297</td>
<td>-.123059</td>
<td>.049466</td>
</tr>
<tr>
<td>Belgium</td>
<td>2</td>
<td>51</td>
<td>46.21698</td>
<td>-.123059</td>
<td>.049466</td>
</tr>
<tr>
<td>Belgium</td>
<td>2</td>
<td>43</td>
<td>54.25965</td>
<td>-.123059</td>
<td>.049466</td>
</tr>
<tr>
<td>Belgium</td>
<td>2</td>
<td>89</td>
<td>66.78156</td>
<td>-.123059</td>
<td>.049466</td>
</tr>
<tr>
<td>Belgium</td>
<td>2</td>
<td>19</td>
<td>34.83411</td>
<td>-.123059</td>
<td>.049466</td>
</tr>
<tr>
<td>Belgium</td>
<td>3</td>
<td>19</td>
<td>8.166062</td>
<td>-.123059</td>
<td>-.4354829</td>
</tr>
<tr>
<td>Belgium</td>
<td>3</td>
<td>15</td>
<td>40.92741</td>
<td>-.123059</td>
<td>-.4354829</td>
</tr>
<tr>
<td>Belgium</td>
<td>3</td>
<td>33</td>
<td>30.78324</td>
<td>-.123059</td>
<td>-.4354829</td>
</tr>
<tr>
<td>Belgium</td>
<td>3</td>
<td>9</td>
<td>6.914059</td>
<td>-.123059</td>
<td>-.4354829</td>
</tr>
<tr>
<td>Belgium</td>
<td>3</td>
<td>12</td>
<td>12.16361</td>
<td>-.123059</td>
<td>-.4354829</td>
</tr>
</tbody>
</table>

We can see that the predicted random effects at the nation level, `re_nat`, are the same for all the observations. Similarly, the predicted random effects at the region level, `re_reg`, are the same within each region. The predicted counts, `mu`, are closer to the observed deaths than the predicted counts from the negative binomial mixed-effects model in example 1 of [ME] `membreg postestimation`.

### Methods and formulas

Methods and formulas for predicting random effects and other statistics are given in *Methods and formulas* of [ME] `meglm postestimation`.

### Also see

[ME] `mepoisson` — Multilevel mixed-effects Poisson regression
[ME] `meglm postestimation` — Postestimation tools for `meglm`
[U] 20 Estimation and postestimation commands
meprobit fits mixed-effects models for binary or binomial responses. The conditional distribution of the response given the random effects is assumed to be Bernoulli, with success probability determined by the standard normal cumulative distribution function.

Two-level probit model of \( y \) and covariate \( x \) and random intercepts by \( \text{lev2} \)

\[
\text{meprobit } y \ x \ || \ \text{lev2}:
\]

Add random coefficients for \( x \)

\[
\text{meprobit } y \ x \ || \ \text{lev2}: x
\]

As above, but specify that \( y \) records the number of successes from 10 trials

\[
\text{meprobit } y \ x \ || \ \text{lev2}: x, \ \text{binomial}(10)
\]

As above, but with the number of trials stored in variable \( n \)

\[
\text{meprobit } y \ x \ || \ \text{lev2}: x, \ \text{binomial}(n)
\]

Three-level random-intercept model of \( y \) and covariate \( x \) with \( \text{lev2} \) nested within \( \text{lev3} \)

\[
\text{meprobit } y \ x \ || \ \text{lev3}: || \ \text{lev2}:
\]

Two-way crossed random effects by factors \( a \) and \( b \)

\[
\text{meprobit } y \ x \ || \ _\text{all}:R.a \ || \ b:
\]
Syntax

\texttt{meprobit } \textit{depvar } \textit{fe\_equation} [ \textit{re\_equation} ] [ \textit{re\_equation} ... ] [ , \textit{options} ]

where the syntax of \textit{fe\_equation} is

\[ \textit{indepvars} \text{ [ if ] [ in ] [ weight ] [ , \textit{fe\_options} ]} \]

and the syntax of \textit{re\_equation} is one of the following:

for random coefficients and intercepts

\textit{levelvar}: [ \textit{varlist} ] [ , \textit{re\_options} ]

for random effects among the values of a factor variable

\textit{levelvar}: \texttt{R.varname}

\textit{levelvar} is a variable identifying the group structure for the random effects at that level or is \texttt{_all} representing one group comprising all observations.

\textbf{fe\_options} \hspace{2cm} Description

\begin{tabular}{ll}
\hline
Model & \\
\texttt{noconstant} & suppress constant term from the fixed-effects equation \\
\texttt{offset(varname)} & include \textit{varname} in model with coefficient constrained to 1 \\
\texttt{asis} & retain perfect predictor variables \\
\hline
\end{tabular}

\textbf{re\_options} \hspace{2cm} Description

\begin{tabular}{ll}
\hline
Model & \\
\texttt{covariance(vartype)} & variance–covariance structure of the random effects \\
\texttt{noconstant} & suppress constant term from the random-effects equation \\
\texttt{fweight(varname)} & frequency weights at higher levels \\
\texttt{iweight(varname)} & importance weights at higher levels \\
\texttt{pweight(varname)} & sampling weights at higher levels \\
\hline
\end{tabular}
## meprobit — Multilevel mixed-effects probit regression

### options

<table>
<thead>
<tr>
<th>Model</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>binomial</strong> (varname</td>
<td>#)</td>
</tr>
<tr>
<td><strong>constraints</strong> (constraints)</td>
<td></td>
</tr>
<tr>
<td><strong>collinear</strong></td>
<td></td>
</tr>
</tbody>
</table>

### SE/Robust

| **vce(vctype)** | vcetype may be oim, robust, or cluster clustvar |

### Reporting

| **level(#)** | set confidence level; default is level(95) |
| **nocnsreport** | do not display constraints |
| **notable** | suppress coefficient table |
| **noheader** | suppress output header |
| **nogroup** | suppress table summarizing groups |
| **nolrtest** | do not perform likelihood-ratio test comparing with probit regression |
| **display_options** | control columns and column formats, row spacing, line width, display of omitted variables and base and empty cells, and factor-variable labeling |

### Integration

| **intmethod(intmethod)** | integration method |
| **intpoints(#)** | set the number of integration (quadrature) points for all levels; default is intpoints(7) |

### Maximization

| **maximize_options** | control the maximization process; seldom used |
| **startvalues(svmethod)** | method for obtaining starting values |
| **startgrid[(gridspec)]** | perform a grid search to improve starting values |
| **noestimate** | do not fit the model; show starting values instead |
| **dnnumerical** | use numerical derivative techniques |
| **coeflegend** | display legend instead of statistics |

### vartype

| **independent** | one unique variance parameter per random effect, all covariances 0; the default unless the R. notation is used |
| **exchangeable** | equal variances for random effects, and one common pairwise covariance |
| **identity** | equal variances for random effects, all covariances 0; the default if the R. notation is used |
| **unstructured** | all variances and covariances to be distinctly estimated |
| **fixed(matname)** | user-selected variances and covariances constrained to specified values; the remaining variances and covariances unrestricted |
| **pattern(matname)** | user-selected variances and covariances constrained to be equal; the remaining variances and covariances unrestricted |
### Options

- **noconstant** suppresses the constant (intercept) term and may be specified for the fixed-effects equation and for any of or all the random-effects equations.

- **offset(varname)** specifies that varname be included in the fixed-effects portion of the model with the coefficient constrained to be 1.

- **asis** forces retention of perfect predictor variables and their associated, perfectly predicted observations and may produce instabilities in maximization; see [R] probit.

- **covariance(vartype)** specifies the structure of the covariance matrix for the random effects and may be specified for each random-effects equation. vartype is one of the following: independent, exchangeable, identity, unstructured, fixed(matname), or pattern(matname).

- **covariance(independent)** covariance structure allows for a distinct variance for each random effect within a random-effects equation and assumes that all covariances are 0. The default is covariance(independent) unless a crossed random-effects model is fit, in which case the default is covariance(identity).

- **covariance(exchangeable)** structure specifies one common variance for all random effects and one common pairwise covariance.

- **covariance(identity)** is short for “multiple of the identity”; that is, all variances are equal and all covariances are 0.

- **covariance(unstructured)** allows for all variances and covariances to be distinct. If an equation consists of $p$ random-effects terms, the unstructured covariance matrix will have $p(p+1)/2$ unique parameters.

- **covariance(fixed(matname))** and **covariance(pattern(matname))** covariance structures provide a convenient way to impose constraints on variances and covariances of random effects. Each specification requires a matname that defines the restrictions placed on variances and covariances. Only elements in the lower triangle of matname are used, and row and column names of matname are ignored. A missing value in matname means that a given element is unrestricted. In a fixed(matname) covariance structure, (co)variance $(i,j)$ is constrained to equal the...
value specified in the \( i,j \)th entry of \( \text{matname} \). In a \textit{pattern(matname)} covariance structure, (co)variances \( (i,j) \) and \( (k,l) \) are constrained to be equal if \( \text{matname}[i,j] = \text{matname}[k,l] \).

\text{fweight(varname)} specifies frequency weights at higher levels in a multilevel model, whereas frequency weights at the first level (the observation level) are specified in the usual manner, for example, \([\text{fw} = \text{fwtvar1}]. \text{varname} \) can be any valid Stata variable name, and you can specify \text{fweight()} \) at levels two and higher of a multilevel model. For example, in the two-level model

\[
\text{mecmd fixed}_\text{portion} [\text{fw} = \text{wt1} \] \| school: ... , fweight(wt2) ...
\]

the variable \text{wt1} would hold the first-level (the observation-level) frequency weights, and \text{wt2} would hold the second-level (the school-level) frequency weights.

\text{iweight(varname)} specifies importance weights at higher levels in a multilevel model, whereas importance weights at the first level (the observation level) are specified in the usual manner, for example, \([\text{iw} = \text{iwtvar1}]. \text{varname} \) can be any valid Stata variable name, and you can specify \text{iweight()} \) at levels two and higher of a multilevel model. For example, in the two-level model

\[
\text{mecmd fixed}_\text{portion} [\text{iw} = \text{wt1} \] \| school: ... , iweight(wt2) ...
\]

the variable \text{wt1} would hold the first-level (the observation-level) importance weights, and \text{wt2} would hold the second-level (the school-level) importance weights.

\text{pweight(varname)} specifies sampling weights at higher levels in a multilevel model, whereas sampling weights at the first level (the observation level) are specified in the usual manner, for example, \([\text{pw} = \text{pwtvar1}]. \text{varname} \) can be any valid Stata variable name, and you can specify \text{pweight()} \) at levels two and higher of a multilevel model. For example, in the two-level model

\[
\text{mecmd fixed}_\text{portion} [\text{pw} = \text{wt1} \] \| school: ... , pweight(wt2) ...
\]

variable \text{wt1} would hold the first-level (the observation-level) sampling weights, and \text{wt2} would hold the second-level (the school-level) sampling weights.

\text{binomial(varname | #)} specifies that the data are in binomial form; that is, \text{depvar} records the number of successes from a series of binomial trials. This number of trials is given either as \text{varname}, which allows this number to vary over the observations, or as the constant \#. If \text{binomial()} \) is not specified (the default), \text{depvar} is treated as Bernoulli, with any nonzero, nonmissing values indicating positive responses.

\text{constraints(constraints), collinear; see [R] estimation options.}

\text{vce(vcetype)} specifies the type of standard error reported, which includes types that are derived from asymptotic theory (\text{oim}), that are robust to some kinds of misspecification (\text{robust}), and that allow for intragroup correlation (\text{cluster clustvar}); see [R] \text{vce_option}. If \text{vce(robust)} \) is specified, robust variances are clustered at the highest level in the multilevel model.

\text{level(#), nocnsreport, ; see [R] estimation options.}

\text{notable} suppresses the estimation table, either at estimation or upon replay.

\text{noheader} suppresses the output header, either at estimation or upon replay.

\text{nogroup} suppresses the display of group summary information (number of groups, average group size, minimum, and maximum) from the output header.
nolrtest prevents meprobit from performing a likelihood-ratio test that compares the mixed-effects probit model with standard (marginal) probit regression. This option may also be specified upon replay to suppress this test from the output.

display_options: noci, nopvalues, noomitted, vsquish, noemptycells, baselevels, allbaselevels, nofvlabel, fwrap(#), fwrapon(style), cformat(%,fmt), pformat(%,fmt), sformat(%,fmt), and nolstretch; see [R] estimation options.

Integration

intmethod(intmethod) specifies the integration method to be used for the random-effects model. 
mvaghermite performs mean–variance adaptive Gauss–Hermite quadrature; mcaghermite performs mode-curvature adaptive Gauss–Hermite quadrature; ghermite performs nonadaptive Gauss–Hermite quadrature; and laplace performs the Laplacian approximation, equivalent to mode-curvature adaptive Gaussian quadrature with one integration point.

The default integration method is mvaghermite unless a crossed random-effects model is fit, in which case the default integration method is laplace. The Laplacian approximation has been known to produce biased parameter estimates; however, the bias tends to be more prominent in the estimates of the variance components rather than in the estimates of the fixed effects.

For crossed random-effects models, estimation with more than one quadrature point may be prohibitively intensive even for a small number of levels. For this reason, the integration method defaults to the Laplacian approximation. You may override this behavior by specifying a different integration method.

intpoints(#) sets the number of integration points for quadrature. The default is intpoints(7), which means that seven quadrature points are used for each level of random effects. This option is not allowed with intmethod(laplace).

The more integration points, the more accurate the approximation to the log likelihood. However, computation time increases as a function of the number of quadrature points raised to a power equaling the dimension of the random-effects specification. In crossed random-effects models and in models with many levels or many random coefficients, this increase can be substantial.

Maximization

maximize_options: difficult, technique(algorithm_spec), iterate(#), [no]log, trace, gradient, showstep, hessian, showtolerance, tolerance(#), ltolerance(#), nrtolerance(#), nonrtolerance, and from(init_specs); see [R] maximize. Those that require special mention for meprobit are listed below.

from() accepts a properly labeled vector of initial values or a list of coefficient names with values. A list of values is not allowed.

The following options are available with meprobit but are not shown in the dialog box:

startvalues(svmethod), startgrid[(gridspec)], noestimate, and dnumerical; see [ME] meglm.

coeflegend; see [R] estimation options.
Remarks and examples

For a general introduction to me commands, see [ME] me.

meprobit is a convenience command for meglm with a probit link and a bernoulli or binomial family; see [ME] meglm.

Remarks are presented under the following headings:

Introduction
Two-level models
Three-level models

Introduction

Mixed-effects probit regression is probit regression containing both fixed effects and random effects. In longitudinal data and panel data, random effects are useful for modeling intracluster correlation; that is, observations in the same cluster are correlated because they share common cluster-level random effects.

Comprehensive treatments of mixed models are provided by, for example, Searle, Casella, and McCulloch (1992); Verbeke and Molenberghs (2000); Raudenbush and Bryk (2002); Demidenko (2004); Hedeker and Gibbons (2006); McCulloch, Searle, and Neuhaus (2008); and Rabe-Hesketh and Skrondal (2012). Guo and Zhao (2000) and Rabe-Hesketh and Skrondal (2012, chap. 10) are good introductory readings on applied multilevel modeling of binary data.

meprobit allows for not just one, but many levels of nested clusters of random effects. For example, in a three-level model you can specify random effects for schools and then random effects for classes nested within schools. In this model, the observations (presumably, the students) comprise the first level, the classes comprise the second level, and the schools comprise the third.

However, for simplicity, we here consider the two-level model, where for a series of $M$ independent clusters, and conditional on a set of fixed effects $x_{ij}$ and a set of random effects $u_j$,

$$
\Pr(y_{ij} = 1 | x_{ij}, u_j) = H(x_{ij} \beta + z_{ij} u_j)
$$

for $j = 1, \ldots, M$ clusters, with cluster $j$ consisting of $i = 1, \ldots, n_j$ observations. The responses are the binary-valued $y_{ij}$, and we follow the standard Stata convention of treating $y_{ij} = 1$ if $\text{depvar}_{ij} \neq 0$ and treating $y_{ij} = 0$ otherwise. The $1 \times p$ row vector $x_{ij}$ are the covariates for the fixed effects, analogous to the covariates you would find in a standard probit regression model, with regression coefficients (fixed effects) $\beta$. For notational convenience here and throughout this manual entry, we suppress the dependence of $y_{ij}$ on $x_{ij}$.

The $1 \times q$ vector $z_{ij}$ are the covariates corresponding to the random effects and can be used to represent both random intercepts and random coefficients. For example, in a random-intercept model, $z_{ij}$ is simply the scalar 1. The random effects $u_j$ are $M$ realizations from a multivariate normal distribution with mean $0$ and $q \times q$ variance matrix $\Sigma$. The random effects are not directly estimated as model parameters but are instead summarized according to the unique elements of $\Sigma$, known as variance components. One special case of (1) places $z_{ij} = x_{ij}$, so that all covariate effects are essentially random and distributed as multivariate normal with mean $\beta$ and variance $\Sigma$.

Finally, because this is probit regression, $H(\cdot)$ is the standard normal cumulative distribution function, which maps the linear predictor to the probability of a success ($y_{ij} = 1$) with $H(v) = \Phi(v)$.

Model (1) may also be stated in terms of a latent linear response, where only $y_{ij} = I(y_{ij}^* > 0)$ is observed for the latent

$$
y_{ij}^* = x_{ij} \beta + z_{ij} u_j + \epsilon_{ij}
$$
The errors $\epsilon_{ij}$ are distributed as a standard normal with mean 0 and variance 1 and are independent of $u_j$.

Model (1) is an example of a generalized linear mixed model (GLMM), which generalizes the linear mixed-effects (LME) model to non-Gaussian responses. You can fit LMEs in Stata by using `mixed` and fit GLMMs by using `meglm`. Because of the relationship between LMEs and GLMMs, there is insight to be gained through examination of the linear mixed model. This is especially true for Stata users because the terminology, syntax, options, and output for fitting these types of models are nearly identical. See [ME] mixed and the references therein, particularly in Introduction, for more information.

Log-likelihood calculations for fitting any generalized mixed-effects model require integrating out the random effects. One widely used modern method is to directly estimate the integral required to calculate the log likelihood by Gauss–Hermite quadrature or some variation thereof. Because the log likelihood itself is estimated, this method has the advantage of permitting likelihood-ratio tests for comparing nested models. Also, if done correctly, quadrature approximations can be quite accurate, thus minimizing bias.

`meprobit` supports three types of Gauss–Hermite quadrature and the Laplacian approximation method; see Methods and formulas of [ME] meglm for details. The simplest random-effects model you can fit using `meprobit` is the two-level model with a random intercept,

$$\Pr(y_{ij} = 1|u_j) = \Phi(x_{ij}\beta + u_j)$$

This model can also be fit using `xtprobit` with the `re` option; see [XT] xtprobit.

Below we present two short examples of mixed-effects probit regression; refer to [ME] melogit for additional examples including crossed random-effects models and to [ME] me and [ME] meglm for examples of other random-effects models.

### Two-level models

We begin with a simple application of (1) as a two-level model, because a one-level model, in our terminology, is just standard probit regression; see [R] probit.

#### Example 1

In example 1 of [ME] melogit, we analyzed a subsample of data from the 1989 Bangladesh fertility survey (Huq and Cleland 1990), which polled 1,934 Bangladeshi women on their use of contraception. The women sampled were from 60 districts, identified by the variable `district`. Each district contained either urban or rural areas (variable `urban`) or both. The variable `c_use` is the binary response, with a value of 1 indicating contraceptive use. Other covariates include mean-centered `age` and three indicator variables recording number of children. Here we refit that model with `meprobit`:
. use http://www.stata-press.com/data/r14/bangladesh
(Bangladesh Fertility Survey, 1989)
. meprobit c_use urban age child* || district:

Fitting fixed-effects model:
Iteration 0: log likelihood = -1228.8313
Iteration 1: log likelihood = -1228.2466
Iteration 2: log likelihood = -1228.2466

Refining starting values:
Grid node 0: log likelihood = -1237.3973

Fitting full model:
Iteration 0: log likelihood = -1237.3973 (not concave)
Iteration 1: log likelihood = -1221.2111 (not concave)
Iteration 2: log likelihood = -1207.4451
Iteration 3: log likelihood = -1206.7002
Iteration 4: log likelihood = -1206.5346
Iteration 5: log likelihood = -1206.5336
Iteration 6: log likelihood = -1206.5336

Mixed-effects probit regression
Number of obs = 1,934
Group variable: district
Number of groups = 60
Obs per group:
  min =  2
  avg = 32.2
  max = 118
Integration method: mvaghermite
Integration pts. = 7
Wald chi2(5) = 115.36
Log likelihood = -1206.5336
Prob > chi2 = 0.0000

|              | Coef. | Std. Err. | z     | P>|z|  | [95% Conf. Interval] |
|--------------|-------|-----------|-------|------|---------------------|
| c_use        |       |           |       |      |                     |
| urban        | .4490191 | .0727176  | 6.17  | 0.000| .3064953 - .5915429 |
| age          | -.0162203 | .0048005  | -3.38 | 0.001| -.0256291 - -.0068114 |
| child1       | .674377  | .0947829  | 7.11  | 0.000| .488606  - .8601481 |
| child2       | .8281581 | .1048136  | 7.90  | 0.000| .6227272 -1.035389 |
| child3       | .8137876 | .1073951  | 7.58  | 0.000| .6032972 -1.024278 |
| _cons        | -1.02799 | .0870307  | -11.81| 0.000| -1.198567 - .8574132 |

|              |       |           |       |      |                     |
| district     |       |           |       |      |                     |
| var(_cons)   | .0798719 | .026886  | .0412921 | .1544972 |

LR test vs. probit model: chibar2(01) = 43.43
Prob >= chibar2 = 0.0000

Comparing the estimates of meprobit with those of melogit, we observe the familiar result
where the probit estimates are closer to 0 in absolute value due to the smaller variance of the error
term in the probit model. Example 1 of [ME] meprobit postestimation shows that the marginal effect
of covariates is nearly the same between the two models.

Unlike a logistic regression, coefficients from a probit regression cannot be interpreted in terms of
odds ratios. Most commonly, probit regression coefficients are interpreted in terms of partial effects,
as we demonstrate in example 1 of [ME] meprobit postestimation. For now, we only note that urban
women and women with more children are more likely to use contraceptives and that contraceptive
use decreases with age. The estimated variance of the random intercept at the district level, \( \sigma^2 \), is
0.08 with standard error 0.03. The reported likelihood-ratio test shows that there is enough variability
between districts to favor a mixed-effects probit regression over an ordinary probit regression; see
Distribution theory for likelihood-ratio test in [ME] me for a discussion of likelihood-ratio testing of
variance components.
Three-level models

Two-level models extend naturally to models with three or more levels with nested random effects. Below we replicate example 2 of [ME] melogit with meprobit.

Example 2

Rabe-Hesketh, Toulopoulou, and Murray (2001) analyzed data from a study that measured the cognitive ability of patients with schizophrenia compared with their relatives and control subjects. Cognitive ability was measured as the successful completion of the “Tower of London”, a computerized task, measured at three levels of difficulty. For all but one of the 226 subjects, there were three measurements (one for each difficulty level). Because patients' relatives were also tested, a family identifier, family, was also recorded.

We fit a probit model with response dtlm, the indicator of cognitive function, and with covariates difficulty and a set of indicator variables for group, with the controls (group==1) being the base category. We also allow for random effects due to families and due to subjects within families.
. use http://www.stata-press.com/data/r14/towerlondon
(Tower of London data)
. meprobit dtlm difficulty i.group || family: || subject:

Fitting fixed-effects model:
Iteration 0: log likelihood = -317.11238
Iteration 1: log likelihood = -314.50535
Iteration 2: log likelihood = -314.50121
Iteration 3: log likelihood = -314.50121

Refining starting values:
Grid node 0: log likelihood = -326.18533

Fitting full model:
Iteration 0: log likelihood = -326.18533 (not concave)
Iteration 1: log likelihood = -313.16256 (not concave)
Iteration 2: log likelihood = -308.47528
Iteration 3: log likelihood = -305.02228
Iteration 4: log likelihood = -304.88972
Iteration 5: log likelihood = -304.88845
Iteration 6: log likelihood = -304.88845

Mixed-effects probit regression
Number of obs = 677

<table>
<thead>
<tr>
<th>Group Variable</th>
<th>No. of Groups</th>
<th>Observations per Group</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Minimum</td>
<td>Average</td>
</tr>
<tr>
<td>family</td>
<td>118</td>
<td>2</td>
</tr>
<tr>
<td>subject</td>
<td>226</td>
<td>2</td>
</tr>
</tbody>
</table>

Integration method: mvaghermite
Integration pts. = 7
Log likelihood = -304.88845
Wald chi2(3) = 83.28
Prob > chi2 = 0.0000

| Coef. | Std. Err. | z     | P>|z|  | [95% Conf. Interval] |
|-------|-----------|-------|-----|----------------------|
| dtlm  | -.9329891 | .1037376 | -8.99 | 0.000 | -1.136311 - .7296672 |
|       | difficulty|        |       |         |                      |
| group | -.1632243 | .204265 | -0.80 | 0.424 | -.5635763 .2371276  |
| 2     |           |        |       |         |                      |
| 3     | -.6220196 | .228063 | -2.73 | 0.006 | -1.069015 -.1750244 |
| _cons | -.8405154 | .1597223 | -5.26 | 0.000 | -1.153565 -.5274654 |

family
| var(_cons) | .2120948 | .1736281 | .0426292 | 1.055244 |

family>
| subject
| var(_cons) | .3559141 | .219331 | .106364 | 1.190956 |

LR test vs. probit model: chi2(2) = 19.23
Prob > chi2 = 0.0001

Notes:
1. This is a three-level model with two random-effects equations, separated by ||. The first is a random intercept (constant only) at the family level, and the second is a random intercept at the subject level. The order in which these are specified (from left to right) is significant—meprobit assumes that subject is nested within family.
The information on groups is now displayed as a table, with one row for each upper level. Among other things, we see that we have 226 subjects from 118 families. You can suppress this table with the \texttt{nogroup} or the \texttt{noheader} option, which will suppress the rest of the header as well.

After adjusting for the random-effects structure, the probability of successful completion of the Tower of London decreases dramatically as the level of difficulty increases. Also, schizophrenics (\texttt{group==3}) tended not to perform as well as the control subjects.

The above extends to models with more than two levels of nesting in the obvious manner, by adding more random-effects equations, each separated by \texttt{||}. The order of nesting goes from left to right as the groups go from biggest (highest level) to smallest (lowest level).

**Stored results**

\texttt{meprobit} stores the following in \texttt{e()}:

** Scalars**

- \texttt{e(N)} number of observations
- \texttt{e(k)} number of parameters
- \texttt{e(k_dv)} number of dependent variables
- \texttt{e(k_eq)} number of equations in \texttt{e(b)}
- \texttt{e(k_eq_model)} number of equations in overall model test
- \texttt{e(k_f)} number of fixed-effects parameters
- \texttt{e(k_r)} number of random-effects parameters
- \texttt{e(k_rs)} number of variances
- \texttt{e(k_rc)} number of covariances
- \texttt{e(df_m)} model degrees of freedom
- \texttt{e(ll)} log likelihood
- \texttt{e(N_clust)} number of clusters
- \texttt{e(chi2)} \( \chi^2 \)
- \texttt{e(p)} significance
- \texttt{e(ll_c)} log likelihood, comparison model
- \texttt{e(chi2_c)} \( \chi^2 \), comparison model
- \texttt{e(df_c)} degrees of freedom, comparison model
- \texttt{e(p_c)} significance, comparison model
- \texttt{e(rank)} rank of \texttt{e(V)}
- \texttt{e(ic)} number of iterations
- \texttt{e(rc)} return code
- \texttt{e(converged)} 1 if converged, 0 otherwise

** Macros**

- \texttt{e(cmd)} \texttt{meglm}
- \texttt{e(cmd2)} \texttt{meprobit}
- \texttt{e(cmdline)} command as typed
- \texttt{e(depvar)} name of dependent variable
- \texttt{e(wtype)} weight type
- \texttt{e(wexp)} weight expression (first-level weights)
- \texttt{e(fweightk)} \texttt{fweight} variable for \( k \)th highest level, if specified
- \texttt{e(iweightk)} \texttt{iweight} variable for \( k \)th highest level, if specified
- \texttt{e(pweightk)} \texttt{pweight} variable for \( k \)th highest level, if specified
- \texttt{e(covariates)} list of covariates
- \texttt{e(ivars)} grouping variables
- \texttt{e(model)} \texttt{probit}
- \texttt{e(title)} title in estimation output
- \texttt{e(link)} \texttt{probit}
- \texttt{e(family)} \texttt{bernoulli} or \texttt{binomial}
- \texttt{e(clustvar)} name of cluster variable
- \texttt{e(offset)} offset
- \texttt{e(binomial)} binomial number of trials
Methods and formulas

Model (1) assumes Bernoulli data, a special case of the binomial. Because binomial data are also supported by `meprobit` (option `binomial()`), the methods presented below are in terms of the more general binomial mixed-effects model.

For a two-level binomial model, consider the response $y_{ij}$ as the number of successes from a series of $r_{ij}$ Bernoulli trials (replications). For cluster $j, j = 1, \ldots, M$, the conditional distribution of $y_j = (y_{j1}, \ldots, y_{jn_j})'$, given a set of cluster-level random effects $u_j$, is

$$f(y_j | u_j) = \prod_{i=1}^{n_j} \left[ \frac{r_{ij}}{y_{ij}} \left\{ \Phi(\eta_{ij}) \right\}^{y_{ij}} \left\{ 1 - \Phi(\eta_{ij}) \right\}^{r_{ij} - y_{ij}} \right]$$

$$= \exp \left( \sum_{i=1}^{n_j} \left[ y_{ij} \log \left\{ \Phi(\eta_{ij}) \right\} - (r_{ij} - y_{ij}) \log \left\{ \Phi(-\eta_{ij}) \right\} + \log \left( \frac{r_{ij}}{y_{ij}} \right) \right] \right)$$

for $\eta_{ij} = x_{ij} \beta + z_{ij} u_j + \text{offset}_{ij}$. 
Defining \( r_j = (r_{j1}, \ldots, r_{jn_j})' \) and

\[
c(y_j, r_j) = \sum_{i=1}^{n_j} \log \left( \frac{r_{ij}}{y_{ij}} \right)
\]

where \( c(y_j, r_j) \) does not depend on the model parameters, we can express the above compactly in matrix notation,

\[
f(y_j | u_j) = \exp \left[ y_j' \log \left\{ \Phi(\eta_j) \right\} - (r_j - y_j)' \log \left\{ \Phi(-\eta_j) \right\} + c(y_j, r_j) \right]
\]

where \( \eta_j \) is formed by stacking the row vectors \( \eta_{ij} \). We extend the definitions of \( \Phi(\cdot), \log(\cdot), \) and \( \exp(\cdot) \) to be vector functions where necessary.

Because the prior distribution of \( u_j \) is multivariate normal with mean \( 0 \) and \( q \times q \) variance matrix \( \Sigma \), the likelihood contribution for the \( j \)th cluster is obtained by integrating \( u_j \) out of the joint density \( f(y_j, u_j) \),

\[
L_j(\beta, \Sigma) = (2\pi)^{-q/2} |\Sigma|^{-1/2} \int f(y_j | u_j) \exp \left( -u_j' \Sigma^{-1} u_j / 2 \right) du_j
\]

where

\[
h(\beta, \Sigma, u_j) = y_j' \log \left\{ \Phi(\eta_j) \right\} - (r_j - y_j)' \log \left\{ \Phi(-\eta_j) \right\} - u_j' \Sigma^{-1} u_j / 2
\]

and for convenience, in the arguments of \( h(\cdot) \) we suppress the dependence on the observable data \((y_j, r_j, X_j, Z_j)\).

The integration in (2) has no closed form and thus must be approximated. \texttt{meprobit} offers four approximation methods: mean–variance adaptive Gauss–Hermite quadrature (default unless a crossed random-effects model is fit), mode-curvature adaptive Gauss–Hermite quadrature, nonadaptive Gauss–Hermite quadrature, and Laplacian approximation (default for crossed random-effects models).

The Laplacian approximation is based on a second-order Taylor expansion of \( h(\beta, \Sigma, u_j) \) about the value of \( u_j \) that maximizes it; see \textit{Methods and formulas} in \cite{MEmeglm} for details.

Gaussian quadrature relies on transforming the multivariate integral in (2) into a set of nested univariate integrals. Each univariate integral can then be evaluated using a form of Gaussian quadrature; see \textit{Methods and formulas} in \cite{MEmeglm} for details.

The log likelihood for the entire dataset is simply the sum of the contributions of the \( M \) individual clusters, namely, \( L(\beta, \Sigma) = \sum_{j=1}^{M} L_j(\beta, \Sigma) \).

Maximization of \( L(\beta, \Sigma) \) is performed with respect to \((\beta, \sigma^2)\), where \( \sigma^2 \) is a vector comprising the unique elements of \( \Sigma \). Parameter estimates are stored in \( e(b) \) as \((\hat{\beta}, \hat{\sigma}^2)\), with the corresponding variance–covariance matrix stored in \( e(V) \).

\texttt{meprobit} supports multilevel weights and survey data; see \textit{Methods and formulas} in \cite{MEmeglm} for details.
References


Also see

[ME] **meprobit postestimation** — Postestimation tools for meprobit
[ME] **mecloglog** — Multilevel mixed-effects complementary log-log regression
[ME] **melogit** — Multilevel mixed-effects logistic regression
[ME] **me** — Introduction to multilevel mixed-effects models
[SEM] **intro 5** — Tour of models (*Multilevel mixed-effects models*)
[SVY] **svy estimation** — Estimation commands for survey data
[XT] **xtprobit** — Random-effects and population-averaged probit models
[U] **20** Estimation and postestimation commands
## meprobit postestimation — Postestimation tools for meprobit

### Postestimation commands

The following postestimation commands are of special interest after `meprobit`:

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>estat group</code></td>
<td>summarize the composition of the nested groups</td>
</tr>
<tr>
<td><code>estat icc</code></td>
<td>estimate intraclass correlations</td>
</tr>
</tbody>
</table>

The following standard postestimation commands are also available:

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>contrast</code></td>
<td>contrasts and ANOVA-style joint tests of estimates</td>
</tr>
<tr>
<td><code>estat ic</code></td>
<td>Akaike’s and Schwarz’s Bayesian information criteria (AIC and BIC)</td>
</tr>
<tr>
<td><code>estat summarize</code></td>
<td>summary statistics for the estimation sample</td>
</tr>
<tr>
<td><code>estat vce</code></td>
<td>variance–covariance matrix of the estimators (VCE)</td>
</tr>
<tr>
<td><code>estat (svy)</code></td>
<td>postestimation statistics for survey data</td>
</tr>
<tr>
<td><code>estimates</code></td>
<td>cataloging estimation results</td>
</tr>
<tr>
<td><code>*hausman</code></td>
<td>Hausman’s specification test</td>
</tr>
<tr>
<td><code>lincom</code></td>
<td>point estimates, standard errors, testing, and inference for linear combinations of coefficients</td>
</tr>
<tr>
<td><code>*lrtest</code></td>
<td>likelihood-ratio test</td>
</tr>
<tr>
<td><code>margins</code></td>
<td>marginal means, predictive margins, marginal effects, and average marginal effects</td>
</tr>
<tr>
<td><code>marginsplot</code></td>
<td>graph the results from margins (profile plots, interaction plots, etc.)</td>
</tr>
<tr>
<td><code>nlcom</code></td>
<td>point estimates, standard errors, testing, and inference for nonlinear combinations of coefficients</td>
</tr>
<tr>
<td><code>predict</code></td>
<td>predictions, residuals, influence statistics, and other diagnostic measures</td>
</tr>
<tr>
<td><code>predictnl</code></td>
<td>point estimates, standard errors, testing, and inference for generalized predictions</td>
</tr>
<tr>
<td><code>pwcompare</code></td>
<td>pairwise comparisons of estimates</td>
</tr>
<tr>
<td><code>test</code></td>
<td>Wald tests of simple and composite linear hypotheses</td>
</tr>
<tr>
<td><code>testnl</code></td>
<td>Wald tests of nonlinear hypotheses</td>
</tr>
</tbody>
</table>

* `hausman` and `lrtest` are not appropriate with `svy` estimation results.
**predict**

**Description for predict**

`predict` creates a new variable containing predictions such as mean responses; linear predictions; density and distribution functions; standard errors; and Pearson, deviance, and Anscombe residuals.

**Menu for predict**

Statistics > Postestimation

**Syntax for predict**

**Syntax for obtaining predictions of the outcome and other statistics**

`predict [type] newvarsspec [if] [in] [ , statistic options ]`

**Syntax for obtaining estimated random effects and their standard errors**

`predict [type] newvarsspec [if] [in] , reffects [re_options ]`

**Syntax for obtaining ML scores**

`predict [type] newvarsspec [if] [in] , scores`

`newvarsspec` is `stub*` or `newvarlist`.

**statistic**

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Main</strong></td>
</tr>
<tr>
<td>mu</td>
</tr>
<tr>
<td>eta</td>
</tr>
<tr>
<td>xb</td>
</tr>
<tr>
<td>stdp</td>
</tr>
<tr>
<td>density</td>
</tr>
<tr>
<td>distribution</td>
</tr>
<tr>
<td>pearson</td>
</tr>
<tr>
<td>deviance</td>
</tr>
<tr>
<td>anscombe</td>
</tr>
</tbody>
</table>

These statistics are available both in and out of sample; type `predict ... if e(sample) ...` if wanted only for the estimation sample.
options | Description
--- | ---

Main

**conditional(ctype)** | compute statistic conditional on estimated random effects; default is conditional(ebmeans)

**marginal** | compute statistic marginally with respect to the random effects

**nooffset** | make calculation ignoring offset or exposure

Integration

**int_options** | integration options

Pearson, deviance, anscombe may not be combined with marginal.

**ctype** | Description

| | Description
--- | ---

**ebmeans** | empirical Bayes means of random effects; the default

**ebmodes** | empirical Bayes modes of random effects

**fixedonly** | prediction for the fixed portion of the model only

**re_options** | Description

Main

**ebmeans** | use empirical Bayes means of random effects; the default

**ebmodes** | use empirical Bayes modes of random effects

**reses(stub* | newvarlist)** | calculate standard errors of empirical Bayes estimates

Integration

**int_options** | integration options

Integration

**int_options** | Description

| | Description
--- | ---

**intpoints(#)** | use # quadrature points to compute marginal predictions and empirical Bayes means

**iterate(#)** | set maximum number of iterations in computing statistics involving empirical Bayes estimators

**tolerance(#)** | set convergence tolerance for computing statistics involving empirical Bayes estimators

---

**Options for predict**

--- | Description
--- | ---

**mu**, the default, calculates the predicted mean, that is, the probability of a positive outcome.

**eta**, xb, stdp, density, distribution, pearson, deviance, anscombe, scores, conditional(), marginal, and nooffset; see [ME] meglm postestimation.

**reffffects**, ebmeans, ebmodes, and reses(); see [ME] meglm postestimation.

**intpoints()**, **iterate()**, and **tolerance()**; see [ME] meglm postestimation.
margins

Description for margins

margins estimates margins of response for mean responses and linear predictions.

Menu for margins

Statistics > Postestimation

Syntax for margins

margins [marginlist] [, options]
margins [marginlist], predict(statistic ...) [predict(statistic ...) ...] [options]

<table>
<thead>
<tr>
<th>statistic</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mu</td>
<td>mean response; the default</td>
</tr>
<tr>
<td>eta</td>
<td>fitted linear predictor</td>
</tr>
<tr>
<td>xb</td>
<td>linear predictor for the fixed portion of the model only</td>
</tr>
<tr>
<td>stdp</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>density</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>distribution</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>pearson</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>deviance</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>anscombe</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>reffects</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>scores</td>
<td>not allowed with margins</td>
</tr>
</tbody>
</table>

Options conditional(ebmeans) and conditional(ebmodes) are not allowed with margins.
Option marginal is assumed where applicable if conditional(fixedonly) is not specified.

Statistics not allowed with margins are functions of stochastic quantities other than e(b).
For the full syntax, see [R] margins.
estat

Description for estat

`estat group` reports the number of groups and minimum, average, and maximum group sizes for each level of the model. Model levels are identified by the corresponding group variable in the data. Because groups are treated as nested, the information in this summary may differ from what you would get if you used the `tabulate` command on each group variable individually.

`estat icc` displays the intraclass correlation for pairs of latent linear responses at each nested level of the model. Intraclass correlations are available for random-intercept models or for random-coefficient models conditional on random-effects covariates being equal to 0. They are not available for crossed-effects models.

Menu for estat

Statistics > Postestimation

Syntax for estat

`Summarize the composition of the nested groups`

`estat group`

`Estimate intraclass correlations`

`estat icc [, level(#)]`

Option for estat icc

`level(#)` specifies the confidence level, as a percentage, for confidence intervals. The default is `level(95)` or as set by `set level`; see [U] 20.7 Specifying the width of confidence intervals.

Remarks and examples

Various predictions, statistics, and diagnostic measures are available after fitting a mixed-effects probit model using `meprobit`. Here we show a short example of predicted probabilities and predicted random effects; refer to [ME] meglm postestimation for additional examples.

Example 1

In example 2 of [ME] meprobit, we analyzed the cognitive ability (dtlm) of patients with schizophrenia compared with their relatives and control subjects, by using a three-level probit model with random effects at the family and subject levels. Cognitive ability was measured as the successful completion of the “Tower of London”, a computerized task, measured at three levels of difficulty.

```
use http://www.stata-press.com/data/r14/towerlondon
(Tower of London data)
.meprobit dtlm difficulty i.group || family: || subject:
(output omitted)
```
We obtain predicted probabilities based on the contribution of both fixed effects and random effects by typing:

```
.predict pr
```

(predictions based on fixed effects and posterior means of random effects)
(option mu assumed)
(using 7 quadrature points)

As the note says, the predicted values are based on the posterior means of random effects. You can use the `modes` option to obtain predictions based on the posterior modes of random effects.

We obtain predictions of the posterior means themselves by typing:

```
.predict re*, reffects
```

(calculating posterior means of random effects)
(using 7 quadrature points)

Because we have one random effect at the family level and another random effect at the subject level, Stata saved the predicted posterior means in the variables `re1` and `re2`, respectively. If you are not sure which prediction corresponds to which level, you can use the `describe` command to show the variable labels.

Here we list the data for family 16:

```
.list family subject dtlm pr re1 re2 if family==16, sepby(subject)
```

<table>
<thead>
<tr>
<th>family</th>
<th>subject</th>
<th>dtlm</th>
<th>pr</th>
<th>re1</th>
<th>re2</th>
</tr>
</thead>
<tbody>
<tr>
<td>208.</td>
<td>16</td>
<td>5</td>
<td>1</td>
<td>.5301687</td>
<td>.5051272</td>
</tr>
<tr>
<td>209.</td>
<td>16</td>
<td>5</td>
<td>0</td>
<td>.1956408</td>
<td>.5051272</td>
</tr>
<tr>
<td>210.</td>
<td>16</td>
<td>5</td>
<td>0</td>
<td>.0367041</td>
<td>.5051272</td>
</tr>
<tr>
<td>211.</td>
<td>16</td>
<td>34</td>
<td>1</td>
<td>.8876646</td>
<td>.5051272</td>
</tr>
<tr>
<td>212.</td>
<td>16</td>
<td>34</td>
<td>1</td>
<td>.6107262</td>
<td>.5051272</td>
</tr>
<tr>
<td>213.</td>
<td>16</td>
<td>34</td>
<td>1</td>
<td>.2572725</td>
<td>.5051272</td>
</tr>
<tr>
<td>214.</td>
<td>16</td>
<td>35</td>
<td>0</td>
<td>.6561904</td>
<td>.5051272</td>
</tr>
<tr>
<td>215.</td>
<td>16</td>
<td>35</td>
<td>1</td>
<td>.2977437</td>
<td>.5051272</td>
</tr>
<tr>
<td>216.</td>
<td>16</td>
<td>35</td>
<td>0</td>
<td>.071612</td>
<td>.5051272</td>
</tr>
</tbody>
</table>

The predicted random effects at the family level (`re1`) are the same for all members of the family. Similarly, the predicted random effects at the individual level (`re2`) are constant within each individual. The predicted probabilities (`pr`) for this family seem to be in fair agreement with the response (`dtlm`) based on a cutoff of 0.5.

We can use `estat icc` to estimate the residual intraclass correlation (conditional on the difficulty level and the individual’s category) between the latent responses of subjects within the same family or between the latent responses of the same subject and family:

```
.estat icc
```

**Residual intraclass correlation**

<table>
<thead>
<tr>
<th>Level</th>
<th>ICC</th>
<th>Std. Err.</th>
<th>[95% Conf. Interval]</th>
</tr>
</thead>
<tbody>
<tr>
<td>family</td>
<td>.1352637</td>
<td>.1050492</td>
<td>.0261998 -.4762821</td>
</tr>
<tr>
<td>subject</td>
<td>family</td>
<td>.3622485</td>
<td>.0877459</td>
</tr>
</tbody>
</table>

`estat icc` reports two intraclass correlations for this three-level nested model. The first is the level-3 intraclass correlation at the family level, the correlation between latent measurements of the
cognitive ability in the same family. The second is the level-2 intraclass correlation at the subject-
within-family level, the correlation between the latent measurements of cognitive ability in the same
subject and family.

There is not a strong correlation between individual realizations of the latent response, even within
the same subject.

Stored results

`estat icc` stores the following in `r()`:

Scalars
- `r(icc#)`: level-# intraclass correlation
- `r(se#)`: standard errors of level-# intraclass correlation
- `r(level)`: confidence level of confidence intervals

Macros
- `r(label#)`: label for level #

Matrices
- `r(ci#)`: vector of confidence intervals (lower and upper) for level-# intraclass correlation

For a $G$-level nested model, # can be any integer between 2 and $G$.

Methods and formulas

Methods and formulas are presented under the following headings:

Prediction
Intraclass correlations

Prediction

Methods and formulas for predicting random effects and other statistics are given in `Methods and formulas` of [ME] `meglm postestimation`.

Intraclass correlations

Consider a simple, two-level random-intercept model, stated in terms of a latent linear response,
where only $y_{ij} = I(y^*_{ij} > 0)$ is observed for the latent variable,

$$y^*_{ij} = \beta + u_j^{(2)} + \epsilon_{ij}^{(1)}$$

with $i = 1, \ldots, n_j$ and level-2 groups $j = 1, \ldots, M$. Here $\beta$ is an unknown fixed intercept, $u_j^{(2)}$ is
a level-2 random intercept, and $\epsilon_{ij}^{(1)}$ is a level-1 error term. Errors are assumed to be distributed as
standard normal with mean 0 and variance 1; random intercepts are assumed to be normally distributed
with mean 0 and variance $\sigma^2_2$ and to be independent of error terms.

The intraclass correlation for this model is

$$\rho = \text{Corr}(y^*_i, y^*_{i'}) = \frac{\sigma^2_2}{1 + \sigma^2_2}$$

It corresponds to the correlation between the latent responses $i$ and $i'$ from the same group $j$. 
Now consider a three-level nested random-intercept model,

\[ y_{ijk}^* = \beta + u_{jk}^{(2)} + u_k^{(3)} + \epsilon_{ijk}^{(1)} \]

for measurements \( i = 1, \ldots, n_{jk} \) and level-2 groups \( j = 1, \ldots, M_1 \) nested within level-3 groups \( k = 1, \ldots, M_2 \). Here \( u_{jk}^{(2)} \) is a level-2 random intercept, \( u_k^{(3)} \) is a level-3 random intercept, and \( \epsilon_{ijk}^{(1)} \) is a level-1 error term. The error terms have a standard normal distribution with mean 0 and variance 1. The random intercepts are assumed to be normally distributed with mean 0 and variances \( \sigma_2^2 \) and \( \sigma_3^2 \), respectively, and to be mutually independent. The error terms are also independent of the random intercepts.

We can consider two types of intraclass correlations for this model. We will refer to them as level-2 and level-3 intraclass correlations. The level-3 intraclass correlation is

\[ \rho^{(3)} = \text{Corr}(y_{ijk}^*, y_{i'jk}^*) = \frac{\sigma_3^2}{1 + \sigma_2^2 + \sigma_3^2} \]

This is the correlation between latent responses \( i \) and \( i' \) from the same level-3 group \( k \) and from different level-2 groups \( j \) and \( j' \).

The level-2 intraclass correlation is

\[ \rho^{(2)} = \text{Corr}(y_{ijk}^*, y_{i'jk}^*) = \frac{\sigma_2^2 + \sigma_3^2}{1 + \sigma_2^2 + \sigma_3^2} \]

This is the correlation between latent responses \( i \) and \( i' \) from the same level-3 group \( k \) and level-2 group \( j \). (Note that level-1 intraclass correlation is undefined.)

More generally, for a \( G \)-level nested random-intercept model, the \( g \)-level intraclass correlation is defined as

\[ \rho^{(g)} = \frac{\sum_{l=g}^{G} \sigma_l^2}{1 + \sum_{l=2}^{G} \sigma_l^2} \]

The above formulas also apply in the presence of fixed-effects covariates \( X \) in a random-effects model. In this case, intraclass correlations are conditional on fixed-effects covariates and are referred to as residual intraclass correlations. \texttt{estat icc} also uses the same formulas to compute intraclass correlations for random-coefficient models, assuming 0 baseline values for the random-effects covariates, and labels them as conditional intraclass correlations.

Intraclass correlations will always fall in \([0,1]\) because variance components are nonnegative. To accommodate the range of an intraclass correlation, we use the logit transformation to obtain confidence intervals. We use the delta method to estimate the standard errors of the intraclass correlations.
Let \( \hat{\rho}(g) \) be a point estimate of the intraclass correlation and \( \hat{SE}(\hat{\rho}(g)) \) be its standard error. The \((1 - \alpha) \times 100\%\) confidence interval for \( \text{logit}(\rho(g)) \) is

\[
\text{logit}(\hat{\rho}(g)) \pm z_{\alpha/2} \frac{\hat{SE}(\hat{\rho}(g))}{\hat{\rho}(g)(1 - \hat{\rho}(g))}
\]

where \( z_{\alpha/2} \) is the \(1 - \alpha/2\) quantile of the standard normal distribution and \( \text{logit}(x) = \ln\{x/(1-x)\} \). Let \( k_u \) be the upper endpoint of this interval, and let \( k_l \) be the lower. The \((1 - \alpha) \times 100\%\) confidence interval for \( \rho(g) \) is then given by

\[
\left( \frac{1}{1 + e^{-k_l}}, \frac{1}{1 + e^{-k_u}} \right)
\]

Also see

[ME] meprob — Multilevel mixed-effects probit regression
[ME] meglm postestimation — Postestimation tools for meglm
[U] 20 Estimation and postestimation commands
Description

meqrlogit, like melogit, fits mixed-effects models for binary or binomial responses. The conditional distribution of the response given the random effects is assumed to be Bernoulli, with success probability determined by the logistic cumulative distribution function.

meqrlogit provides an alternative estimation method, which uses the QR decomposition of the variance-components matrix. This method may aid convergence when variance components are near the boundary of the parameter space.

Quick start

Two-level logistic regression of $y$ on $x$ with random intercepts by lev2 using QR decomposition

```
meqrlogit y x || lev2:
```

Add random coefficients for $x$

```
meqrlogit y x || lev2: x
```

As above, but allow the random effects to be correlated

```
meqrlogit y x || lev2: x, covariance(unstructured)
```

Three-level random-intercept model of $y$ on $x$ with lev2 nested within lev3

```
meqrlogit y x || lev3: || lev2:
```

Crossed-effects model of $y$ on $x$ with two-way crossed random effects by factors $a$ and $b$

```
meqrlogit y x || _all:R.a || b:
```

Menu

Statistics > Multilevel mixed-effects models > Estimation by QR decomposition > Logistic regression
Syntax

```
meqrlogit depvar fe_equation || re_equation [ || re_equation ... ] [ , options ]
```

where the syntax of `fe_equation` is

```
[ indepvars ] [ if ] [ in ] [ , fe_options ]
```

and the syntax of `re_equation` is one of the following:

- for random coefficients and intercepts
  
  ```
  levelvar: [ varlist ] [ , re_options ]
  ```

- for random effects among the values of a factor variable
  
  ```
  levelvar: R.varname [ , re_options ]
  ```

`levelvar` is a variable identifying the group structure for the random effects at that level or is `_all` representing one group comprising all observations.

---

**fe_options**

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>noconstant</code> suppress constant term from the fixed-effects equation</td>
</tr>
<tr>
<td><code>offset(varname)</code> include <code>varname</code> in model with coefficient constrained to 1</td>
</tr>
</tbody>
</table>

---

**re_options**

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>covariance(vartype)</code> variance–covariance structure of the random effects</td>
</tr>
<tr>
<td><code>noconstant</code> suppress constant term from the random-effects equation</td>
</tr>
<tr>
<td><code>collinear</code> keep collinear variables</td>
</tr>
</tbody>
</table>
**options**

<table>
<thead>
<tr>
<th>Model</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>**binomial(varname</td>
<td>#)**</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Reporting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>level(#)</strong></td>
<td>set confidence level; default is level(95)</td>
</tr>
<tr>
<td><strong>variance</strong></td>
<td>report fixed-effects coefficients as odds ratios</td>
</tr>
<tr>
<td><strong>stddeviations</strong></td>
<td>show random-effects parameter estimates as variances and covariances; the default</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Reporting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>noret</strong></td>
<td>suppress random-effects table</td>
</tr>
<tr>
<td><strong>nofet</strong></td>
<td>suppress fixed-effects table</td>
</tr>
<tr>
<td><strong>estmetric</strong></td>
<td>show parameter estimates in the estimation metric</td>
</tr>
<tr>
<td><strong>noheader</strong></td>
<td>suppress output header</td>
</tr>
<tr>
<td><strong>nogroup</strong></td>
<td>suppress table summarizing groups</td>
</tr>
<tr>
<td><strong>nolrtest</strong></td>
<td>do not perform likelihood-ratio test comparing with logistic regression</td>
</tr>
</tbody>
</table>

| display_options | control columns and column formats, row spacing, line width, display of omitted variables and base and empty cells, and factor-variable labeling |

<table>
<thead>
<tr>
<th>Integration</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>intpoints(# [# ...])</strong></td>
<td>set the number of integration (quadrature) points; default is intpoints(7)</td>
</tr>
<tr>
<td><strong>laplace</strong></td>
<td>use Laplacian approximation; equivalent to intpoints(1)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Maximization</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>maximize_options</strong></td>
<td>control the maximization process; seldom used</td>
</tr>
<tr>
<td><strong>retoleance(#)</strong></td>
<td>tolerance for random-effects estimates; default is retolerance(1e-8); seldom used</td>
</tr>
<tr>
<td><strong>reiterate(#)</strong></td>
<td>maximum number of iterations for random-effects estimation; default is reiterate(50); seldom used</td>
</tr>
<tr>
<td><strong>matsqrt</strong></td>
<td>parameterize variance components using matrix square roots; the default</td>
</tr>
<tr>
<td><strong>matlog</strong></td>
<td>parameterize variance components using matrix logarithms</td>
</tr>
<tr>
<td><strong>refineopts(maximize_options)</strong></td>
<td>control the maximization process during refinement of starting values</td>
</tr>
<tr>
<td><strong>coeflegend</strong></td>
<td>display legend instead of statistics</td>
</tr>
</tbody>
</table>
### vartype

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>independent</td>
</tr>
<tr>
<td>exchangeable</td>
</tr>
<tr>
<td>identity</td>
</tr>
<tr>
<td>unstructured</td>
</tr>
</tbody>
</table>

*indepvars* may contain factor variables; see [U] 11.4.3 Factor variables.

*indepvars* and *varlist* may contain time-series operators; see [U] 11.4.4 Time-series varlists.

*bootstrap*, by, *jackknife*, *mi estimate*, rolling, and *statsby* are allowed; see [U] 11.1.10 Prefix commands.

*coeflegend* does not appear in the dialog box.

See [U] 20 Estimation and postestimation commands for more capabilities of estimation commands.

### Options

- **noconstant** suppresses the constant (intercept) term and may be specified for the fixed-effects equation and for any of or all the random-effects equations.

- **offset(varname)** specifies that *varname* be included in the fixed-effects portion of the model with the coefficient constrained to be 1.

- **covariance(vartype)** specifies the structure of the covariance matrix for the random effects and may be specified for each random-effects equation. *vartype* is one of the following: independent, exchangeable, identity, and unstructured.

  - **covariance(independent)** covariance structure allows for a distinct variance for each random effect within a random-effects equation and assumes that all covariances are 0. The default is **covariance(independent)**, except when the R. notation is used, in which case the default is **covariance(identity)** and only **covariance(identity)** and **covariance(exchangeable)** are allowed.

  - **covariance(exchangeable)** structure specifies one common variance for all random effects and one common pairwise covariance.

  - **covariance(identity)** is short for “multiple of the identity”; that is, all variances are equal and all covariances are 0.

  - **covariance(unstructured)** allows for all variances and covariances to be distinct. If an equation consists of *p* random-effects terms, the unstructured covariance matrix will have \( p(p + 1)/2 \) unique parameters.

- **collinear** specifies that meqrologit not omit collinear variables from the random-effects equation. Usually, there is no reason to leave collinear variables in place; in fact, doing so usually causes the estimation to fail because of the matrix singularity caused by the collinearity. However, with certain models (for example, a random-effects model with a full set of contrasts), the variables may be collinear, yet the model is fully identified because of restrictions on the random-effects covariance structure. In such cases, using the **collinear** option allows the estimation to take place with the random-effects equation intact.
binomial(varname | #) specifies that the data are in binomial form; that is, depvar records the number of successes from a series of binomial trials. This number of trials is given either as varname, which allows this number to vary over the observations, or as the constant #. If binomial() is not specified (the default), depvar is treated as Bernoulli, with any nonzero, nonmissing values indicating positive responses.

Reporting

level(#); see [R] estimation options.

or reports estimated fixed-effects coefficients transformed to odds ratios, that is, \(\exp(\beta)\) rather than \(\beta\). Standard errors and confidence intervals are similarly transformed. This option affects how results are displayed, not how they are estimated. or may be specified either at estimation or upon replay.

variance, the default, displays the random-effects parameter estimates as variances and covariances.

stddeviations displays the random-effects parameter estimates as standard deviations and correlations.

noretable suppresses the random-effects table.

nofetable suppresses the fixed-effects table.

estmetric displays all parameter estimates in the estimation metric. Fixed-effects estimates are unchanged from those normally displayed, but random-effects parameter estimates are displayed as log-standard deviations and hyperbolic arctangents of correlations, with equation names that organize them by model level.

noheader suppresses the output header, either at estimation or upon replay.

nogroup suppresses the display of group summary information (number of groups, average group size, minimum, and maximum) from the output header.

nolrtest prevents meqrlogit from performing a likelihood-ratio test that compares the mixed-effects logistic model with standard (marginal) logistic regression. This option may also be specified upon replay to suppress this test from the output.

display_options: noci, nopvalues, noomitted, vsquish, noemptycells, baselevels, allbaselevels, nolabel, fvwrap(#), fvwrapon(style), cformat(%fmt), pformat(%fmt), sformat(%fmt), and nolstretch; see [R] estimation options.

Integration

intpoints(# | # | ...) sets the number of integration points for adaptive Gaussian quadrature. The more integration points, the more accurate the approximation to the log likelihood. However, computation time increases with the number of quadrature points, and in models with many levels or many random coefficients, this increase can be substantial.

You may specify one number of integration points applying to all levels of random effects in the model, or you may specify distinct numbers of points for each level. intpoints(7) is the default; that is, by default seven quadrature points are used for each level.

laplace specifies that log likelihoods be calculated using the Laplacian approximation, equivalent to adaptive Gaussian quadrature with one integration point for each level in the model; laplace is equivalent to intpoints(1). Computation time increases as a function of the number of quadrature points raised to a power equaling the dimension of the random-effects specification. The computational time saved by using laplace can thus be substantial, especially when you have many levels or random coefficients.
The Laplacian approximation has been known to produce biased parameter estimates, but the bias tends to be more prominent in the estimates of the variance components rather than in the estimates of the fixed effects. If your interest lies primarily with the fixed-effects estimates, the Laplace approximation may be a viable faster alternative to adaptive quadrature with multiple integration points.

When the R.varname notation is used, the dimension of the random effects increases by the number of distinct values of varname. Even when this number is small to moderate, it increases the total random-effects dimension to the point where estimation with more than one quadrature point is prohibitively intensive.

For this reason, when you use the R. notation in your random-effects equations, the laplace option is assumed. You can override this behavior by using the intpoints() option, but doing so is not recommended.

Maximization

\[ \text{Maximize options: difficult, technique(algorithm_spec), iterate(#), [no]log, trace, gradient, showstep, hessian, showt2olerance, tolerance(#), ltolerance(#), nrtolerance(#), nonrtolerance, and from(init_specs); see [R] maximize.} \]

Those that require special mention for meqrlogit are listed below.

For the technique() option, the default is technique(nr). The bhhh algorithm may not be specified.

from(init_specs) is particularly useful when combined with refineopts(iterate(0)) (see the description below), which bypasses the initial optimization stage.

retolerance(#) specifies the convergence tolerance for the estimated random effects used by adaptive Gaussian quadrature. Although not estimated as model parameters, random-effects estimators are used to adapt the quadrature points. Estimating these random effects is an iterative procedure, with convergence declared when the maximum relative change in the random effects is less than retolerance(). The default is retolerance(1e-8). You should seldom have to use this option.

reiterate(#) specifies the maximum number of iterations used when estimating the random effects to be used in adapting the Gaussian quadrature points; see the retolerance() option. The default is reiterate(50). You should seldom have to use this option.

matsqrt (the default), during optimization, parameterizes variance components by using the matrix square roots of the variance–covariance matrices formed by these components at each model level. matlog, during optimization, parameterizes variance components by using the matrix logarithms of the variance–covariance matrices formed by these components at each model level.

The matsqrt parameterization ensures that variance–covariance matrices are positive semidefinite, while matlog ensures matrices that are positive definite. For most problems, the matrix square root is more stable near the boundary of the parameter space. However, if convergence is problematic, one option may be to try the alternate matlog parameterization. When convergence is not an issue, both parameterizations yield equivalent results.

refineopts(maximize_options) controls the maximization process during the refinement of starting values. Estimation in meqrlogit takes place in two stages. In the first stage, starting values are refined by holding the quadrature points fixed between iterations. During the second stage, quadrature points are adapted with each evaluation of the log likelihood. Maximization options specified within refineopts() control the first stage of optimization; that is, they control the refining of starting values.

maximize_options specified outside refineopts() control the second stage.
The one exception to the above rule is the nolog option, which when specified outside refineopts() applies globally.

from(init_specs) is not allowed within refineopts() and instead must be specified globally. Refining starting values helps make the iterations of the second stage (those that lead toward the solution) more numerically stable. In this regard, of particular interest is refineopts(iterate(#)), with two iterations being the default. Should the maximization fail because of instability in the Hessian calculations, one possible solution may be to increase the number of iterations here.

The following option is available with meqrrlogit but is not shown in the dialog box: coeflegend; see [R] estimation options.

Remarks and examples

Remarks are presented under the following headings:

- Introduction
- Two-level models
- Other covariance structures
- Three-level models
- Crossed-effects models

Introduction

Mixed-effects logistic regression is logistic regression containing both fixed effects and random effects. In longitudinal data and panel data, random effects are useful for modeling intracluster correlation; that is, observations in the same cluster are correlated because they share common cluster-level random effects.

meqrrlogit allows for not just one, but many levels of nested clusters of random effects. For example, in a three-level model you can specify random effects for schools and then random effects for classes nested within schools. In this model, the observations (presumably, the students) comprise the first level, the classes comprise the second level, and the schools comprise the third.

However, for simplicity, for now we consider the two-level model, where for a series of $M$ independent clusters, and conditional on a set of random effects $u_j$,

$$
Pr(y_{ij} = 1|u_j) = H(x_{ij}\beta + z_{ij}u_j)
$$

for $j = 1,\ldots,M$ clusters, with cluster $j$ consisting of $i = 1,\ldots,n_j$ observations. The responses are the binary-valued $y_{ij}$, and we follow the standard Stata convention of treating $y_{ij} = 1$ if $depvar_{ij} \neq 0$ and treating $y_{ij} = 0$ otherwise. The $1 \times p$ row vector $x_{ij}$ are the covariates for the fixed effects, analogous to the covariates you would find in a standard logistic regression model, with regression coefficients (fixed effects) $\beta$.

The $1 \times q$ vector $z_{ij}$ are the covariates corresponding to the random effects and can be used to represent both random intercepts and random coefficients. For example, in a random-intercept model, $z_{ij}$ is simply the scalar 1. The random effects $u_j$ are $M$ realizations from a multivariate normal distribution with mean $0$ and $q \times q$ variance matrix $\Sigma$. The random effects are not directly estimated as model parameters but are instead summarized according to the unique elements of $\Sigma$, known as variance components. One special case of (1) places $z_{ij} = x_{ij}$ so that all covariate effects are essentially random and distributed as multivariate normal with mean $\beta$ and variance $\Sigma$.

Finally, because this is logistic regression, $H(\cdot)$ is the logistic cumulative distribution function, which maps the linear predictor to the probability of a success ($y_{ij} = 1$), with $H(v) = \exp(v)/(1 + \exp(v))$. 
Model (1) may also be stated in terms of a latent linear response, where only $y_{ij} = I(y_{ij}^* > 0)$ is observed for the latent

$$y_{ij}^* = \mathbf{x}_{ij}\beta + \mathbf{z}_{ij}\mathbf{u}_j + \epsilon_{ij}$$

The errors $\epsilon_{ij}$ are distributed as logistic with mean 0 and variance $\pi^2/3$ and are independent of $\mathbf{u}_j$.

Model (1) is an example of a generalized linear mixed model (GLMM), which generalizes the linear mixed-effects (LME) model to non-Gaussian responses. You can fit LMEs in Stata by using \texttt{mixed} and fit GLMMs by using \texttt{meglm}. Because of the relationship between LMEs and GLMMs, there is insight to be gained through examination of the linear mixed model. This is especially true for Stata users because the terminology, syntax, options, and output for fitting these types of models are nearly identical. See [ME] \texttt{mixed} and the references therein, particularly in \textit{Introduction}, for more information.

Multilevel models with binary responses have been used extensively in the health and social sciences. As just one example, Leyland and Goldstein (2001, sec. 3.6) describe a study of equity of health care in Great Britain. Multilevel models with binary and other limited dependent responses also have a long history in econometrics; Rabe-Hesketh, Skrondal, and Pickles (2005) provide an excellent survey.

Log-likelihood calculations for fitting any generalized mixed-effects model require integrating out the random effects. One widely used modern method is to directly estimate the integral required to calculate the log likelihood by Gauss–Hermite quadrature or some variation thereof. The estimation method used by \texttt{meqrlogit} is a multicoefficient and multilevel extension of one of these quadrature types, namely, adaptive Gaussian quadrature (AGQ) based on conditional modes, with the multicoefficient extension from Pinheiro and Bates (1995) and the multilevel extension from Pinheiro and Chao (2006); see \textit{Methods and formulas}.

Two-level models

We begin with a simple application of (1) as a two-level model, because a one-level model, in our terminology, is just standard logistic regression; see [R] \texttt{logistic}.

Example 1

Ng et al. (2006) analyze a subsample of data from the 1989 Bangladesh fertility survey (Huq and Cleland 1990), which polled 1,934 Bangladeshi women on their use of contraception.
The women sampled were from 60 districts, identified by the variable `district`. Each district contained either urban or rural areas (variable `urban`) or both. The variable `c_use` is the binary response, with a value of 1 indicating contraceptive use. Other covariates include mean-centered age and three indicator variables recording number of children.

Consider a standard logistic regression model, amended to have random effects for each district. Defining $\pi_{ij} = \Pr(c_{use_{ij}} = 1)$, we have

$$\text{logit}(\pi_{ij}) = \beta_0 + \beta_1 \text{urban}_{ij} + \beta_2 \text{age}_{ij} + \beta_3 \text{child1}_{ij} + \beta_4 \text{child2}_{ij} + \beta_5 \text{child3}_{ij} + u_j$$

for $j = 1, \ldots, 60$ districts, with $i = 1, \ldots, n_j$ women in district $j$. 
. meqrlogit c_use urban age child* || district:

Refining starting values:
Iteration 0:  log likelihood = -1219.2682
Iteration 1:  log likelihood = -1209.3544
Iteration 2:  log likelihood = -1207.1895

Performing gradient-based optimization:
Iteration 0:  log likelihood = -1207.1895
Iteration 1:  log likelihood = -1206.8323
Iteration 2:  log likelihood = -1206.8322
Iteration 3:  log likelihood = -1206.8322

Mixed-effects logistic regression
Number of obs = 1,934
Group variable: district
Number of groups = 60

Obs per group:
    min = 2
    avg = 32.2
    max = 118

Integration points = 7
Wald chi2(5) = 109.60
Log likelihood = -1206.8322
Prob > chi2 = 0.0000

| Coef. Std. Err.  z  P>|z|  [95% Conf. Interval] |
|-----------------|-----------------|------|--------|-----------------|
| c_use           |                 |      |        |                 |
| urban           | .7322764        | .1194857 | 6.13   | 0.000           | .4980887 - .9664641 |
| age             | -.0264982       | .0078916  | -3.36  | 0.001           | -.0419654 - .0110309 |
| child1          | 1.116002        | .1580921  | 7.06   | 0.000           | .8061466 1.425856 |
| child2          | 1.365895        | .174669   | 7.82   | 0.000           | 1.02355   1.70824 |
| child3          | 1.344031        | .1796549  | 7.48   | 0.000           | .9919141 1.696148 |
| _cons           | -1.68929        | .1477592  | -11.43 | 0.000           | -1.978892-1.399687 |

Random-effects Parameters

| Estimate Std. Err.  [95% Conf. Interval] |
|-----------------|-----------------|-----------------|
| district: Identity var(_cons) | .2156188 | .0733234 | .1107202 .4199007 |

LR test vs. logistic model: chi2(01) = 43.39  Prob >= chibar2 = 0.0000

Notes:
1. The estimation log consists of two parts:
   (a) A set of iterations aimed at refining starting values. These are designed to be relatively quick iterations aimed at getting the parameter estimates within a neighborhood of the eventual solution, making the iterations in (b) more numerically stable.
   (b) A set of gradient-based iterations. By default, these are Newton–Raphson iterations, but other methods are available by specifying the appropriate maximize options; see [R] maximize.
2. The first estimation table reports the fixed effects, and these can be interpreted just as you would the output from logit. You can also specify the or option at estimation or on replay to display the fixed effects as odds ratios instead.
   If you did display results as odds ratios, you would find urban women to have roughly double the odds of using contraception as that of their rural counterparts. Having any number of children will increase the odds from three- to fourfold when compared with the base category of no children. Contraceptive use also decreases with age.
3. The second estimation table shows the estimated variance components. The first section of the table is labeled district: Identity, meaning that these are random effects at the district level and that their variance–covariance matrix is a multiple of the identity matrix; that is, \( \Sigma = \sigma_u^2 I \).
Because we have only one random effect at this level, meqrlogit knew that Identity is the only possible covariance structure. In any case, $\sigma^2_u$ was estimated as 0.22 with standard error 0.07. If you prefer standard deviation estimates $\hat{\sigma}_u$ to variance estimates $\hat{\sigma}^2_u$, specify the stddeviations option either at estimation or on replay.

4. A likelihood-ratio test comparing the model to ordinary logistic regression, (2) without $u_j$, is provided and is highly significant for these data.

5. Finally, because (2) is a simple random-intercept model, you can also fit it with xtlogit, specifying the re option.

We now store our estimates for later use.

. estimates store r_int

In what follows, we will be extending (2), focusing on the variable urban. Before we begin, to keep things short we restate (2) as

$$\logit(\pi_{ij}) = \beta_0 + \beta_1 \text{urban}_{ij} + \mathcal{F}_{ij} + u_j$$

where $\mathcal{F}_{ij}$ is merely shorthand for the portion of the fixed-effects specification having to do with age and children.

▶ Example 2

Extending (2) to allow for a random slope on the indicator variable urban yields the model

$$\logit(\pi_{ij}) = \beta_0 + \beta_1 \text{urban}_{ij} + \mathcal{F}_{ij} + u_j + v_j \text{urban}_{ij}$$

(3)

which we can fit by typing

. meqrlogit c_use urban age child* || district: urban
  (output omitted)
. estimates store r_urban

Extending the model was as simple as adding urban to the random-effects specification so that the model now includes a random intercept and a random coefficient on urban. We dispense with the output because, although this is an improvement over the random-intercept model (2),

. lrtest r_int r_urban

Likelihood-ratio test LR chi2(1) = 3.66
(Assumption: r_int nested in r_urban) Prob > chi2 = 0.0558

Note: The reported degrees of freedom assumes the null hypothesis is not on the boundary of the parameter space. If this is not true, then the reported test is conservative.

we find the default covariance structure for $(u_j, v_j)$, covariance(Independent),

$$\Sigma = \text{Var} \begin{bmatrix} u_j \\ v_j \end{bmatrix} = \begin{bmatrix} \sigma^2_u & 0 \\ 0 & \sigma^2_v \end{bmatrix}$$

to be inadequate. We state that the random-coefficient model is an “improvement” over the random-intercept model because the null hypothesis of the likelihood-ratio comparison test ($H_0: \sigma^2_v = 0$) is on the boundary of the parameter test. This makes the reported $p$-value, 5.6%, an upper bound on the actual $p$-value, which is actually half of that; see Distribution theory for likelihood-ratio test in [ME] me.
We see below that we can reject this model in favor of one that allows correlation between \( u_j \) and \( v_j \).

```
. meqrlogit c_use urban age child* || district: urban, covariance(unstructured)
```

Refining starting values:

<table>
<thead>
<tr>
<th>Iteration</th>
<th>Log likelihood</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>-1215.8594</td>
</tr>
<tr>
<td>1</td>
<td>-1204.0802</td>
</tr>
<tr>
<td>2</td>
<td>-1199.7994</td>
</tr>
</tbody>
</table>

Performing gradient-based optimization:

<table>
<thead>
<tr>
<th>Iteration</th>
<th>Log likelihood</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>-1199.7994</td>
</tr>
<tr>
<td>1</td>
<td>-1199.4801</td>
</tr>
<tr>
<td>2</td>
<td>-1199.3158</td>
</tr>
<tr>
<td>3</td>
<td>-1199.315</td>
</tr>
<tr>
<td>4</td>
<td>-1199.315</td>
</tr>
</tbody>
</table>

Mixed-effects logistic regression

|       | Coef.    | Std. Err. | z     | P>|z| | [95% Conf. Interval] |
|-------|----------|-----------|-------|------|----------------------|
| c_use |          |           |       |      |                      |
| urban | .8157872 | .1715519  | 4.76  | 0.000| .4795516              | 1.152023              |
| age   | -.026415 | .008023   | -3.29 | 0.001| -.0421398             | -.0106902             |
| child1| 1.13252  | .1603285  | 7.06  | 0.000| .818282              | 1.446758              |
| child2| 1.357739 | .1770522  | 7.67  | 0.000| 1.010724             | 1.704755              |
| child3| 1.353827 | .1828801  | 7.40  | 0.000| .9953882             | 1.712265              |
| _cons | -1.71165 | .1605617  | -10.66| 0.000| -2.026345            | -1.396954             |

Random-effects Parameters

<table>
<thead>
<tr>
<th></th>
<th>Estimate</th>
<th>Std. Err.</th>
<th>[95% Conf. Interval]</th>
</tr>
</thead>
<tbody>
<tr>
<td>district: Unstructured</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>var(urban)</td>
<td>.6663222</td>
<td>.3224715</td>
<td>.2580709 1.7204</td>
</tr>
<tr>
<td>var(_cons)</td>
<td>.3897434</td>
<td>.1292458</td>
<td>.2034723  .7465387</td>
</tr>
<tr>
<td>cov(urban,_cons)</td>
<td>-.4058846</td>
<td>.1755418</td>
<td>-.7499402 -.0618289</td>
</tr>
</tbody>
</table>

LR test vs. logistic model: chi2(3) = 58.42 Prob > chi2 = 0.0000

Note: LR test is conservative and provided only for reference.

```
. estimates store r_urban_corr
. lrtest r_urban r_urban_corr
```

Likelihood-ratio test: LR chi2(1) = 11.38 Prob > chi2 = 0.0007

(Assumption: r_urban nested in r_urban_corr)

By specifying covariance(unstructured) above, we told meqrlogit to allow correlation between random effects at the district level; that is,

\[
\Sigma = \text{Var} \begin{bmatrix} u_j \\ v_j \end{bmatrix} = \begin{bmatrix} \sigma_u^2 & \sigma_{uv} \\ \sigma_{uv} & \sigma_v^2 \end{bmatrix}
\]
Example 3

The purpose of introducing a random coefficient on the binary variable `urban` in (3) was to allow for separate random effects, within each district, for the urban and rural areas of that district. Hence, if we have the binary variable `rural` in our data such that \( r_{ij} = 1 - u_{ij} \), then we can reformulate (3) as

\[
\logit(\pi_{ij}) = \beta_0 r_{ij} + (\beta_0 + \beta_1) u_{ij} + F_{ij} + u_j r_{ij} + (u_j + v_j) u_{ij} \tag{3a}
\]

where we have translated both the fixed portion and the random portion to be in terms of `rural` rather than a random intercept. Translating the fixed portion is not necessary to make the point we make below, but we do so anyway for uniformity.

Translating the estimated random-effects parameters from the previous output to ones appropriate for (3a), we get \( \text{Var}(u_j) = \hat{\sigma}^2_u = 0.39 \),

\[
\text{Var}(u_j + v_j) = \hat{\sigma}^2_u + \hat{\sigma}^2_v + 2\hat{\sigma}_{uv} = 0.39 + 0.67 - 2(0.41) = 0.24
\]

and \( \text{Cov}(u_j, u_j + v_j) = \hat{\sigma}^2_u + \hat{\sigma}_{uv} = 0.39 - 0.41 = -0.02 \).

An alternative that does not require remembering how to calculate variances and covariances involving sums—and one that also gives you standard errors—is to let Stata do the work for you:

```stata
. generate byte rural = 1 - urban
. meqrlogit c_use rural urban age child*, noconstant || district: rural urban, > noconstant cov(unstr)
(output omitted)
```

Mixed-effects logistic regression

Number of obs = 1,934
Group variable: district
Number of groups = 60
Obs per group:
  min = 2
  avg = 32.2
  max = 118

Integration points = 7
Wald chi2(6) = 120.24
Log likelihood = -1199.315 Prob > chi2 = 0.0000

| Coef. Std. Err. z  P>|z| [95% Conf. Interval] |
|-----------------|------------------|------------------|------------------|
| c_use rural | -1.71165 | .1605618 | -10.66 | 0.000 | -2.026345 | -1.396954 |
| urban | -0.8958623 | .1704961 | -5.25 | 0.000 | -1.230028 | -.561961 |
| age | -0.026415 | .008023 | -3.29 | 0.001 | -0.0421398 | -0.0106902 |
| child1 | 1.13252 | .1603285 | 7.06 | 0.000 | .818282 | 1.446758 |
| child2 | 1.357739 | .1770522 | 7.67 | 0.000 | 1.010724 | 1.704755 |
| child3 | 1.353827 | .1828801 | 7.40 | 0.000 | .9953882 | 1.712265 |

Random-effects Parameters

| Estimate Std. Err. [95% Conf. Interval] |
|-----------------|------------------|------------------|
| district: Unstructured | var(rural) | .3897439 | .1292459 | .2034726 | .7465393 |
| var(urban) | .2442965 | .1450674 | .0762886 | .782303 |
| cov(rural,urban) | -.0161411 | .1057469 | -.2234012 | .1911189 |

LR test vs. logistic model: chi2(3) = 58.42 Prob > chi2 = 0.0000

Note: LR test is conservative and provided only for reference.
The above output demonstrates an equivalent fit to that we displayed for model (3), with the added benefit of a more direct comparison of the parameters for rural and urban areas.

**Technical note**

We used the binary variables `rural` and `urban` instead of the factor notation `i.urban` because, although supported in the fixed-effects specification of the model, such notation is not supported in random-effects specifications.

**Technical note**

Our model fits for (3) and (3a) are equivalent only because we allowed for correlation in the random effects for both. Had we used the default `Independent` covariance structure, we would be fitting different models; in (3) we would be making the restriction that $\text{Cov}(u_j, v_j) = 0$, whereas in (3a) we would be assuming that $\text{Cov}(u_j, u_j + v_j) = 0$.

The moral here is that although `meqrlogit` will do this by default, one should be cautious when imposing an independent covariance structure, because the correlation between random effects is not invariant to model translations that would otherwise yield equivalent results in standard regression models. In our example, we remapped an intercept and binary coefficient to two complementary binary coefficients, something we could do in standard logistic regression without consequence but that here required more consideration.

Rabe-Hesketh and Skrondal (2012, sec. 11.4) provide a nice discussion of this phenomenon in the related case of recentering a continuous covariate.

**Other covariance structures**

In the above examples, we demonstrated the `Independent` and `Unstructured` covariance structures. Also available are `Identity` (seen previously in output but not directly specified), which restricts random effects to be uncorrelated and share a common variance, and `Exchangeable`, which assumes a common variance and a common pairwise covariance.

You can also specify multiple random-effects equations at the same level, in which case the above four covariance types can be combined to form more complex blocked-diagonal covariance structures. This could be used, for example, to impose an equality constraint on a subset of variance components or to otherwise group together a set of related random effects.

Continuing the previous example, typing

```
. meqrlogit c_use urban age child* || district: child*, cov(exchangeable) ||
> district:
```

would fit a model with the same fixed effects as (3) but with random-effects structure

$$\text{logit}(\pi_{ij}) = \beta_0 + \cdots + u_{1j}\text{child1}_{ij} + u_{2j}\text{child2}_{ij} + u_{3j}\text{child3}_{ij} + v_j$$

That is, we have random coefficients on each indicator variable for children (the first `district:` specification) and an overall district random intercept (the second `district:` specification). The above syntax fits a model with overall covariance structure

$$\Sigma = \text{Var} \begin{bmatrix} u_{1j} \\ u_{2j} \\ u_{3j} \\ v_j \end{bmatrix} = \begin{bmatrix} \sigma_u^2 & \sigma_c & \sigma_c & 0 \\ \sigma_c & \sigma_u^2 & \sigma_c & 0 \\ \sigma_c & \sigma_c & \sigma_u^2 & 0 \\ 0 & 0 & 0 & \sigma_v^2 \end{bmatrix}$$
reflecting the relationship among the random coefficients for children. We did not have to specify noconstant on the first district: specification. meqrlogit automatically avoids collinearity by including an intercept on only the final specification among repeated-level equations.

Of course, if we fit the above model, we would heed our own advice from the previous technical note and make sure that not only our data but also our specification characterization of the random effects permitted the above structure. That is, we would check the above against a model that had an Unstructured covariance for all four random effects and then perhaps against a model that assumed an Unstructured covariance among the three random coefficients on children, coupled with independence with the random intercept. All comparisons can be made by storing estimates (command estimates store) and then using lrtest, as demonstrated previously.

Three-level models

The methods we have discussed so far extend from two-level models to models with three or more levels with nested random effects.

Example 4

Rabe-Hesketh, Toulopoulou, and Murray (2001) analyzed data from a study measuring the cognitive ability of patients with schizophrenia compared with their relatives and control subjects. Cognitive ability was measured as the successful completion of the “Tower of London”, a computerized task, measured at three levels of difficulty. For all but one of the 226 subjects, there were three measurements (one for each difficulty level). Because patients’ relatives were also tested, a family identifier, family, was also recorded.

```
. use http://www.stata-press.com/data/r14/towerlondon, clear
   (Tower of London data)
. describe
Contains data from http://www.stata-press.com/data/r14/towerlondon.dta
    obs:        677   Tower of London data
    vars:        5
    size:   4,739 (_dta has notes)

variable name   storage   display  value
       format      label
family     int     %8.0g   Family ID
subject    int     %9.0g   Subject ID
dtlm       byte     %9.0g   1 = task completed
difficulty byte     %9.0g   Level of difficulty: -1, 0, or 1
group      byte     %8.0g   1: controls; 2: relatives; 3: schizophrenics
```

We fit a logistic model with response dtlm, the indicator of cognitive function, and with covariates difficulty and a set of indicator variables for group, with the controls (group==1) being the base category. We allow for random effects due to families and due to subjects within families, and we request to see odds ratios.
. meqrlogit dtlm difficulty i.group || family: || subject: , or

Refining starting values:
Iteration 0: log likelihood = -310.28433
Iteration 1: log likelihood = -306.42785 (not concave)
Iteration 2: log likelihood = -305.25996

Performing gradient-based optimization:
Iteration 0: log likelihood = -305.25996
Iteration 1: log likelihood = -305.12097
Iteration 2: log likelihood = -305.12043
Iteration 3: log likelihood = -305.12043

Mixed-effects logistic regression

<table>
<thead>
<tr>
<th>Group Variable</th>
<th>No. of Groups</th>
<th>Observations per Group</th>
<th>Integration Points</th>
</tr>
</thead>
<tbody>
<tr>
<td>family</td>
<td>118</td>
<td>2</td>
<td>5.7</td>
</tr>
<tr>
<td>subject</td>
<td>226</td>
<td>2</td>
<td>3.0</td>
</tr>
</tbody>
</table>

Number of obs = 677

Log likelihood = -305.12043
Wald chi2(3) = 74.89
Prob > chi2 = 0.0000

| dtlm  | Odds Ratio | Std. Err. | z     | P>|z| | [95% Conf. Interval] |
|-------|------------|-----------|-------|-------|----------------------|
| difficulty | .192337   | .0371622 | -8.53 | 0.000 | .131704 .2808839 |
| group  |            |           |       |       |                      |
| 2      | .7798295  | .2763766 | -0.70 | 0.483 | .3893393 1.561964  |
| 3      | .3491338  | .1396499 | -2.63 | 0.009 | .1594117 .7646517  |
| _cons | .2263075   | .064463  | -5.22 | 0.000 | .1294902 .3955133  |

Random-effects Parameters

<table>
<thead>
<tr>
<th>family: Identity</th>
<th>Estimate</th>
<th>Std. Err.</th>
<th>[95% Conf. Interval]</th>
</tr>
</thead>
<tbody>
<tr>
<td>var(_cons)</td>
<td>.569182</td>
<td>.5216584</td>
<td>.0944322 3.430694</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>subject: Identity</th>
<th>Estimate</th>
<th>Std. Err.</th>
<th>[95% Conf. Interval]</th>
</tr>
</thead>
<tbody>
<tr>
<td>var(_cons)</td>
<td>1.137931</td>
<td>.6857498</td>
<td>.3492672 3.707441</td>
</tr>
</tbody>
</table>

LR test vs. logistic model: chi2(2) = 17.54
Prob > chi2 = 0.0002

Note: LR test is conservative and provided only for reference.

Notes:
1. This is a three-level model with two random-effects equations, separated by ||. The first is a random intercept (constant only) at the family level, and the second is a random intercept at the subject level. The order in which these are specified (from left to right) is significant—meqrlogit assumes that subject is nested within family.

2. The information on groups is now displayed as a table, with one row for each upper level. Among other things, we see that we have 226 subjects from 118 families. Also the number of integration points for adaptive Gaussian quadrature is displayed within this table, because you can choose to have it vary by model level. As with two-level models, the default is seven points. You can suppress this table with the nogroup or the noheader option, which will suppress the rest of the header as well.

3. The variance-component estimates are now organized and labeled according to level.
After adjusting for the random-effects structure, the odds of successful completion of the Tower of London decrease dramatically as the level of difficulty increases. Also, schizophrenics (group==3) tended not to perform as well as the control subjects. Of course, we would make similar conclusions from a standard logistic model fit to the same data, but the odds ratios would differ somewhat.

Technical note

In the previous example, the subjects are coded with unique values between 1 and 251 (with some gaps), but such coding is not necessary to produce nesting within families. Once we specified the nesting structure to meqrlogit, all that was important was the relative coding of subject within each unique value of family. We could have coded subjects as the numbers 1, 2, 3, and so on, restarting at 1 with each new family, and meqrlogit would have produced the same results.

Group identifiers may also be coded using string variables.

The above extends to models with more than two levels of nesting in the obvious manner, by adding more random-effects equations, each separated by ||. The order of nesting goes from left to right as the groups go from biggest (highest level) to smallest (lowest level).

Crossed-effects models

Not all mixed-effects models contain nested random effects.

Example 5

Rabe-Hesketh and Skrondal (2012, 443–460) perform an analysis on school data from Fife, Scotland. The data, originally from Paterson (1991), are from a study measuring students’ attainment as an integer score from 1 to 10, based on the Scottish school exit examination taken at age 16. The study comprises 3,435 students who first attended any one of 148 primary schools and then any one of 19 secondary schools.

```
. use http://www.stata-press.com/data/r14/fifeschool
(School data from Fife, Scotland)
. describe
Contains data from http://www.stata-press.com/data/r14/fifeschool.dta
obs: 3,435 School data from Fife, Scotland
vars: 5 28 May 2014 10:08
size: 24,045 (_dta has notes)

 variable name type format label variable label
     pid int %9.0g Primary school ID
    sid byte %9.0g Secondary school ID
   attain byte %9.0g Attainment score at age 16
      vrq int %9.0g Verbal-reasoning score from final year of primary school
     sex byte %9.0g 1: female; 0: male
```

Sorted by:

```
. generate byte attain_gt_6 = attain > 6
```
To make the analysis relevant to our present discussion, we focus not on the attainment score itself but instead on whether the score is greater than 6. We wish to model this indicator as a function of the fixed effect `sex` and of random effects due to primary and secondary schools.

For this analysis, it would make sense to assume that the random effects are not nested, but instead crossed, meaning that the effect due to primary school is the same regardless of the secondary school attended. Our model is thus

\[
\logit \{ \Pr(\text{attain}_{ijk} > 6) \} = \beta_0 + \beta_1 \text{sex}_{ijk} + u_j + v_k
\]

for student \( i, i = 1, \ldots, n_{jk} \), who attended primary school \( j, j = 1, \ldots, 148 \), and then secondary school \( k, k = 1, \ldots, 19 \).

Because there is no evident nesting, one solution would be to consider the data as a whole and fit a two-level, one-cluster model with random-effects structure

\[
\begin{bmatrix}
  u_1 \\
  \vdots \\
  u_{148} \\
  v_1 \\
  \vdots \\
  v_{19}
\end{bmatrix} \sim N(\mathbf{0}, \Sigma); \quad \Sigma = \begin{bmatrix}
  \sigma_u^2 I_{148} & 0 \\
  0 & \sigma_v^2 I_{19}
\end{bmatrix}
\]

We can fit such a model by using the group designation `_all:`, which tells `meqrlogit` to treat the whole dataset as one cluster, and the `R.varname` notation, which mimics the creation of indicator variables identifying schools:

```
. meqrlogit attain_gt_6 sex || _all:R.pid || _all:R.sid, or
```

But we do not recommend fitting the model this way because of high total dimension \((148 + 19 = 167)\) of the random effects. This would require working with matrices of column dimension 167, which is probably not a problem for most current hardware, but would be a problem if this number got much larger.

An equivalent way to fit (4) that has a smaller dimension is to treat the clusters identified by primary schools as nested within all the data, that is, as nested within the `_all` group.
Choosing the primary schools as those to nest was no accident; because there are far fewer secondary schools than primary schools, the above required only 19 random coefficients for the secondary schools and one random intercept at the primary school level, for a total dimension of 20. Our data also include a measurement of verbal reasoning, the variable `vrq`. Adding a fixed effect due to `vrq` in (4) would negate the effect due to secondary school, a fact we leave to you to verify as an exercise.

See [ME] `mixed` for a similar discussion of crossed effects in the context of linear mixed models. Also see Rabe-Hesketh and Skrondal (2012) for more examples of crossed-effects models, including models with random interactions, and for more techniques on how to avoid high-dimensional estimation.

Technical note

The estimation in the previous example was performed using a Laplacian approximation, even though we did not specify this. Whenever the R. notation is used in random-effects specifications, estimation reverts to the Laplacian method because of the high dimension induced by having the R. variables.

In the above example, through some creative nesting, we reduced the dimension of the random effects to 20, but this is still too large to permit estimation via adaptive Gaussian quadrature; see Computation time and the Laplacian approximation in [ME] `me`. Even with two quadrature points, our rough formula for computation time would contain within it a factor of $2^{20} = 1,048,576$. 
The *laplace* option is therefore assumed when you use R. notation. If the number of distinct levels of your R. variables is small enough (say, five or fewer) to permit estimation via AGQ, you can override the imposition of *laplace* by specifying the *intpoints()* option.

### Stored results

`meqrlogit` stores the following in `e()`:

**Scalars**

- `e(N)` number of observations
- `e(k)` number of parameters
- `e(k_f)` number of fixed-effects parameters
- `e(k_r)` number of random-effects parameters
- `e(k_rs)` number of variances
- `e(k_rc)` number of covariances
- `e(df_m)` model degrees of freedom
- `e(ll)` log likelihood
- `e(chi2)` \(\chi^2\)
- `e(p)` significance
- `e(ll_c)` log likelihood, comparison model
- `e(chi2_c)` \(\chi^2\), comparison model
- `e(df_c)` degrees of freedom, comparison model
- `e(p_c)` significance, comparison model
- `e(rank)` rank of `e(V)`
- `e(reparm_rc)` return code, final reparameterization
- `e(rc)` return code
- `e(converged)` 1 if converged, 0 otherwise

**Macros**

- `e(cmd)` `meqrlogit`
- `e(cmdline)` command as typed
- `e(depvar)` name of dependent variable
- `e(ivars)` grouping variables
- `e(model)` logistic
- `e(title)` title in estimation output
- `e(offset)` offset
- `e(binomial)` binomial number of trials
- `e(redim)` random-effects dimensions
- `e(vartypes)` variance-structure types
- `e(revars)` random-effects covariates
- `e(n_quad)` number of integration points
- `e(laplace)` *laplace*, if Laplace approximation
- `e(chi2type)` Wald; type of model \(\chi^2\)
- `e(vce)` bootstrap or *jackknife* if defined
- `e(vcetype)` title used to label Std. Err.
- `e(method)` ML
- `e(opt)` type of optimization
- `e(ml_method)` type of *ml* method
- `e(technique)` maximization technique
- `e(datasignature)` the checksum
- `e(datasignaturevars)` variables used in calculation of checksum
- `e(properties)` `b V`
- `e(estat_cmd)` program used to implement *estat*
- `e(predict)` program used to implement *predict*
- `e(marginsnotok)` predictions disallowed by *margins*
- `e(marginsdefault)` default *predict()* specification for *margins*
- `e(asbalanced)` factor variables *fvset* as *asbalanced*
- `e(asobserved)` factor variables *fvset* as *asobserved*
Methods and formulas

Model (1) assumes Bernoulli data, a special case of the binomial. Because binomial data are also supported by `meqrlogit` (option `binomial()`), the methods presented below are in terms of the more general binomial mixed-effects model.

For a two-level binomial model, consider the response $y_{ij}$ as the number of successes from a series of $r_{ij}$ Bernoulli trials (replications). For cluster $j$, $j = 1, \ldots, M$, the conditional distribution of $y_j = (y_{j1}, \ldots, y_{jn_j})'$, given a set of cluster-level random effects $u_j$, is

$$f(y_j|u_j) = \prod_{i=1}^{n_j} \left[ H(x_{ij}\beta + z_{ij}u_j) \right]^{y_{ij}} \left[ 1 - H(x_{ij}\beta + z_{ij}u_j) \right]^{r_{ij} - y_{ij}}$$

$$= \exp\left( \sum_{i=1}^{n_j} y_{ij} (x_{ij}\beta + z_{ij}u_j) - r_{ij} \log \left\{ 1 + \exp (x_{ij}\beta + z_{ij}u_j) \right\} + \log \left( \frac{r_{ij}}{y_{ij}} \right) \right)$$

for $H(v) = \exp(v)/\{1 + \exp(v)\}$.

Defining $r_j = (r_{j1}, \ldots, r_{jn_j})'$ and

$$c(y_j, r_j) = \sum_{i=1}^{n_j} \log \left( \frac{r_{ij}}{y_{ij}} \right)$$

where $c(y_j, r_j)$ does not depend on the model parameters, we can express the above compactly in matrix notation,

$$f(y_j|u_j) = \exp \left[ y_j' (X_j\beta + Z_ju_j) - r_j' \log \left\{ 1 + \exp (X_j\beta + Z_ju_j) \right\} + c(y_j, r_j) \right]$$

where $X_j$ is formed by stacking the row vectors $x_{ij}$ and $Z_j$ is formed by stacking the row vectors $z_{ij}$. We extend the definitions of the functions $\log(\cdot)$ and $\exp(\cdot)$ to be vector functions where necessary.

Because the prior distribution of $u_j$ is multivariate normal with mean $\mathbf{0}$ and $q \times q$ variance matrix $\Sigma$, the likelihood contribution for the $j$th cluster is obtained by integrating $u_j$ out of the joint density $f(y_j, u_j)$,

$$L_j(\beta, \Sigma) = (2\pi)^{-q/2} |\Sigma|^{-1/2} \int f(y_j|u_j) \exp \left\{ -u_j'\Sigma^{-1}u_j/2 \right\} \, du_j$$

$$= \exp \left\{ c(y_j, r_j) \right\} (2\pi)^{-q/2} |\Sigma|^{-1/2} \int \exp \left\{ h(\beta, \Sigma, u_j) \right\} \, du_j$$

(5)
where

\[ h(\beta, \Sigma, u_j) = y'_j (X_j \beta + Z_j u_j) - r'_j \log \{ 1 + \exp (X_j \beta + Z_j u_j) \} - u'_j \Sigma^{-1} u_j / 2 \]

and for convenience, in the arguments of \( h(\cdot) \) we suppress the dependence on the observable data \((y_j, r_j, X_j, Z_j)\).

The integration in (5) has no closed form and thus must be approximated. The Laplacian approximation (Tierney and Kadane 1986; Pinheiro and Bates 1995) is based on a second-order Taylor expansion of \( h(\beta, \Sigma, u_j) \) about the value of \( u_j \) that maximizes it. Taking first and second derivatives, we obtain

\[ h'(\beta, \Sigma, u_j) = \frac{\partial h(\beta, \Sigma, u_j)}{\partial u_j} = Z'_j \{ y_j - m(\beta, u_j) \} - \Sigma^{-1} u_j \]

\[ h''(\beta, \Sigma, u_j) = \frac{\partial^2 h(\beta, \Sigma, u_j)}{\partial u_j \partial u'_j} = - \{ Z'_j V(\beta, u_j) Z_j + \Sigma^{-1} \} \]

where \( m(\beta, u_j) \) is the vector function with the \( i \)th element equal to the conditional mean of \( y_{ij} \) given \( u_j \), that is, \( r_{ij} H(x_{ij} \beta + z_{ij} u_j) \). \( V(\beta, u_j) \) is the diagonal matrix whose diagonal entries \( v_{ij} \) are the conditional variances of \( y_{ij} \) given \( u_j \), namely,

\[ v_{ij} = r_{ij} H(x_{ij} \beta + z_{ij} u_j) \{ 1 - H(x_{ij} \beta + z_{ij} u_j) \} \]

The maximizer of \( h(\beta, \Sigma, u_j) \) is \( \hat{u}_j \) such that \( h'(\beta, \Sigma, \hat{u}_j) = 0 \). The integrand in (5) is proportional to the posterior density \( f(u_j | y_j) \), so \( \hat{u}_j \) also represents the posterior mode, a plausible estimator of \( u_j \) in its own right.

Given the above derivatives, the second-order Taylor approximation then takes the form

\[ h(\beta, \Sigma, u_j) \approx h(\beta, \Sigma, \hat{u}_j) + \frac{1}{2} (u_j - \hat{u}_j)' h''(\beta, \Sigma, \hat{u}_j) (u_j - \hat{u}_j) \]  \hspace{1cm} (6)

The first-derivative term vanishes because \( h'(\beta, \Sigma, \hat{u}_j) = 0 \). Therefore,

\[
\int \exp \{ h(\beta, \Sigma, u_j) \} \, du_j \approx \exp \{ h(\beta, \Sigma, \hat{u}_j) \} \times \int \exp \left[ -\frac{1}{2} (u_j - \hat{u}_j)' \{ -h''(\beta, \Sigma, \hat{u}_j) \} (u_j - \hat{u}_j) \right] \, du_j \]  \hspace{1cm} (7)

because the latter integrand can be recognized as the “kernel” of a multivariate normal density.

Combining the above with (5) (and taking logs) gives the Laplacian log-likelihood contribution of the \( j \)th cluster,

\[ \mathcal{L}^{\text{Lap}}_j(\beta, \Sigma) = -\frac{1}{2} \log |\Sigma| - \log |R_j| + h(\beta, \Sigma, \hat{u}_j) + c(y_j, r_j) \]

where \( R_j \) is an upper-triangular matrix such that \(-h''(\beta, \Sigma, \hat{u}_j) = R_j R'_j\). Pinheiro and Chao (2006) show that \( \hat{u}_j \) and \( R_j \) can be efficiently computed as the iterative solution to a least-squares problem by using matrix decomposition methods similar to those used in fitting LME models (Bates and Pinheiro 1998; Pinheiro and Bates 2000; [ME] mixed).
The fidelity of the Laplacian approximation is determined wholly by the accuracy of the approximation in (6). An alternative that does not depend so heavily on this approximation is integration via AGQ (Naylor and Smith 1982; Liu and Pierce 1994).

The application of AGQ to this particular problem is from Pinheiro and Bates (1995). When we reexamine the integral in question, a transformation of integration variables yields

\[
\int \exp \{ h (\beta, \Sigma, u_j) \} \, du_j = |R_j|^{-1} \int \exp \{ h (\beta, \Sigma, \hat{u}_j + R_j^{-1} t) \} \, dt
\]

\[
= (2\pi)^{q/2} |R_j|^{-1} \int \exp \{ h (\beta, \Sigma, \hat{u}_j + R_j^{-1} t) + t'T/2 \} \phi(t) \, dt
\]

where \( \phi(\cdot) \) is the standard multivariate normal density. Because the integrand is now expressed as some function multiplied by a normal density, it can be estimated by applying the rules of standard Gauss–Hermite quadrature. For a predetermined number of quadrature points \( N_Q \), define \( a_k = \sqrt{2} a^*_k \) and \( w_k = w^*_k/\sqrt{\pi} \), for \( k = 1, \ldots, N_Q \), where \( (a^*_k, w^*_k) \) are a set of abscissas and weights for Gauss–Hermite quadrature approximations of \( \int \exp(-x^2)f(x) \, dx \), as obtained from Abramowitz and Stegun (1972, 924).

Define \( a_k = (a_{k1}, a_{k2}, \ldots, a_{kq})' \); that is, \( a_k \) is a vector that spans the \( N_Q \) abscissas over the dimension \( q \) of the random effects. Applying quadrature rules to (8) yields the AGQ approximation,

\[
\int \exp \{ h (\beta, \Sigma, u_j) \} \, du_j
\]

\[
\approx (2\pi)^{q/2} |R_j|^{-1} \sum_{k_1=1}^{N_Q} \cdots \sum_{k_q=1}^{N_Q} \left[ \exp \{ h (\beta, \Sigma, \hat{u}_j + R_j^{-1} a_k) + a_k a_k/2 \} \prod_{p=1}^{q} w_{k_p} \right]
\]

\[
\equiv (2\pi)^{q/2} \hat{G}_j(\beta, \Sigma)
\]

resulting in the AGQ log-likelihood contribution of the \( i \)th cluster,

\[
L^\text{AGQ}_j(\beta, \Sigma) = -\frac{1}{2} \log |\Sigma| + \log \left\{ \hat{G}_j(\beta, \Sigma) \right\} + c(y_j, r_j)
\]

The “adaptive” part of adaptive Gaussian quadrature lies in the translation and rescaling of the integration variables in (8) by using \( \hat{u}_j \) and \( R_j^{-1} \), respectively. This transformation of quadrature abscissas (centered at 0 in standard form) is chosen to better capture the features of the integrand, which through (7) can be seen to resemble a multivariate normal distribution with mean \( \hat{u}_j \) and variance \( R_j^{-1} R_j^{-T} \). AGQ is therefore not as dependent as the Laplace method upon the approximation in (6). In AGQ, (6) serves merely to redirect the quadrature abscissas, with the AGQ approximation improving as the number of quadrature points \( N_Q \) increases. In fact, Pinheiro and Bates (1995) point out that AGQ with only one quadrature point \( (a = 0 \text{ and } w = 1) \) reduces to the Laplacian approximation.

The log likelihood for the entire dataset is then simply the sum of the contributions of the \( M \) individual clusters, namely, \( L(\beta, \Sigma) = \sum_{j=1}^{M} L^\text{Lap}_j(\beta, \Sigma) \) for Laplace and \( L(\beta, \Sigma) = \sum_{j=1}^{M} L^\text{AGQ}_j(\beta, \Sigma) \) for AGQ.

Maximization of \( L(\beta, \Sigma) \) is performed with respect to \( (\beta, \theta) \), where \( \theta \) is a vector comprising the unique elements of the matrix square root of \( \Sigma \). This is done to ensure that \( \Sigma \) is always positive semidefinite. If the \texttt{matlog} option is specified, then \( \theta \) instead consists of the unique elements of the matrix logarithm of \( \Sigma \). For well-conditioned problems, both methods produce equivalent results, yet our experience deems the former as more numerically stable near the boundary of the parameter space.
Once maximization is achieved, parameter estimates are mapped from \((\hat{\beta}, \hat{\theta})\) to \((\hat{\beta}, \hat{\gamma})\), where \(\hat{\gamma}\) is a vector containing the unique (estimated) elements of \(\Sigma\), expressed as logarithms of standard deviations for the diagonal elements and hyperbolic arctangents of the correlations for off-diagonal elements. This last step is necessary to (a) obtain a parameterization under which parameter estimates can be displayed and interpreted individually, rather than as elements of a matrix square root (or logarithm), and (b) parameterize these elements such that their ranges each encompass the entire real line.

Parameter estimates are stored in \(e(b)\) as \((\hat{\beta}, \hat{\gamma})\), with the corresponding variance–covariance matrix stored in \(e(V)\). Parameter estimates can be displayed in this metric by specifying the \texttt{estmetric} option. However, in \texttt{meqrlogit} output, variance components are most often displayed either as variances and covariances (the default) or as standard deviations and correlations (option \texttt{stddeviations}).

The approach outlined above can be extended from two-level models to higher-level models; see Pinheiro and Chao (2006) for details.

References


Also see

- [ME] `meqrlogit postestimation` — Postestimation tools for meqrlogit
- [ME] `mecloglog` — Multilevel mixed-effects complementary log-log regression
- [ME] `melogit` — Multilevel mixed-effects logistic regression
- [ME] `meprobit` — Multilevel mixed-effects probit regression
- [ME] `me` — Introduction to multilevel mixed-effects models
- [MI] `estimation` — Estimation commands for use with mi estimate
- [SEM] `intro 5` — Tour of models (*Multilevel mixed-effects models*)
- [XT] `xtlogit` — Fixed-effects, random-effects, and population-averaged logit models
- [U] `20 Estimation and postestimation commands`
meqrlogit postestimation — Postestimation tools for meqrlogit

Postestimation commands

The following postestimation commands are of special interest after meqrlogit:

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>estat group</td>
<td>summarize the composition of the nested groups</td>
</tr>
<tr>
<td>estat recovariance</td>
<td>display the estimated random-effects covariance matrix (or matrices)</td>
</tr>
<tr>
<td>estat icc</td>
<td>estimate intraclass correlations</td>
</tr>
</tbody>
</table>

The following standard postestimation commands are also available:

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>contrast</td>
<td>contrasts and ANOVA-style joint tests of estimates</td>
</tr>
<tr>
<td>estat ic</td>
<td>Akaike’s and Schwarz’s Bayesian information criteria (AIC and BIC)</td>
</tr>
<tr>
<td>estat summarize</td>
<td>summary statistics for the estimation sample</td>
</tr>
<tr>
<td>estat vce</td>
<td>variance–covariance matrix of the estimators (VCE)</td>
</tr>
<tr>
<td>estimates</td>
<td>cataloging estimation results</td>
</tr>
<tr>
<td>hausman</td>
<td>Hausman’s specification test</td>
</tr>
<tr>
<td>lincom</td>
<td>point estimates, standard errors, testing, and inference for linear</td>
</tr>
<tr>
<td></td>
<td>combinations of coefficients</td>
</tr>
<tr>
<td>lrtest</td>
<td>likelihood-ratio test</td>
</tr>
<tr>
<td>margins</td>
<td>marginal means, predictive margins, marginal effects, and average</td>
</tr>
<tr>
<td></td>
<td>marginal effects</td>
</tr>
<tr>
<td>marginsplot</td>
<td>graph the results from margins (profile plots, interaction plots, etc.)</td>
</tr>
<tr>
<td>nlcom</td>
<td>point estimates, standard errors, testing, and inference for nonlinear</td>
</tr>
<tr>
<td></td>
<td>combinations of coefficients</td>
</tr>
<tr>
<td>predict</td>
<td>predictions, residuals, influence statistics, and other diagnostic measures</td>
</tr>
<tr>
<td>predictnl</td>
<td>point estimates, standard errors, testing, and inference for generalized</td>
</tr>
<tr>
<td></td>
<td>predictions</td>
</tr>
<tr>
<td>pwcompare</td>
<td>pairwise comparisons of estimates</td>
</tr>
<tr>
<td>test</td>
<td>Wald tests of simple and composite linear hypotheses</td>
</tr>
<tr>
<td>testnl</td>
<td>Wald tests of nonlinear hypotheses</td>
</tr>
</tbody>
</table>

274
predict

Description for predict

predict creates a new variable containing predictions such as mean responses; linear predictions; standard errors; and Pearson, deviance, and Anscombe residuals.

Menu for predict

Statistics > Postestimation

Syntax for predict

Syntax for obtaining estimated random effects and their standard errors

```
predict [type] newvarsspec [if] [in], reffects [reses(newvarsspec)
relevel(levelvar)]
```

Syntax for obtaining other predictions

```
predict [type] newvar [if] [in] [, statistic nooffset fixedonly]
```

newvarsspec is stub* or newvarlist.

<table>
<thead>
<tr>
<th>statistic</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mu</td>
<td>mean response; the default</td>
</tr>
<tr>
<td>xb</td>
<td>linear predictor for the fixed portion of the model only</td>
</tr>
<tr>
<td>stdp</td>
<td>standard error of the fixed-portion linear prediction</td>
</tr>
<tr>
<td>pearson</td>
<td>Pearson residuals</td>
</tr>
<tr>
<td>deviance</td>
<td>deviance residuals</td>
</tr>
<tr>
<td>anscombe</td>
<td>Anscombe residuals</td>
</tr>
</tbody>
</table>

These statistics are available both in and out of sample; type `predict ... if e(sample) ...` if wanted only for the estimation sample.

Options for predict

reffects calculates posterior modal estimates of the random effects. By default, estimates for all random effects in the model are calculated. However, if the `relevel(levelvar)` option is specified, then estimates for only level `levelvar` in the model are calculated. For example, if `classes` are nested within `schools`, then typing

```
predict b*, reffects relevel(school)
```

would yield random-effects estimates at the school level. You must specify `q` new variables, where `q` is the number of random-effects terms in the model (or level). However, it is much easier to just specify `stub*` and let Stata name the variables `stub1, stub2, ..., stubq` for you.
**resex(newvarsspec)** calculates standard errors for the random-effects estimates. By default, standard errors for all random effects in the model are calculated. However, if the **relevel(levelvar)** option is specified, then standard errors for only level *levelvar* in the model are calculated; see the **reffects** option.

You must specify *q* new variables, where *q* is the number of random-effects terms in the model (or level). However, it is much easier to just specify *stub* and let Stata name the variables *stub1, stub2, ..., stubq* for you. The new variables will have the same storage type as the corresponding random-effects variables.

The **reffects** and **reses()** options often generate multiple new variables at once. When this occurs, the random effects (or standard errors) contained in the generated variables correspond to the order in which the variance components are listed in the output of **meqrlogit**. Still, examining the variable labels of the generated variables (with the **describe** command, for instance) can be useful in deciphering which variables correspond to which terms in the model.

**relevel(levelvar)** specifies the level in the model at which predictions for random effects and their standard errors are to be obtained. *levelvar* is the name of the model level and is either the name of the variable describing the grouping at that level or is _all_, a special designation for a group comprising all the estimation data.

**mu**, the default, calculates the predicted mean. By default, this is based on a linear predictor that includes both the fixed effects and the random effects, and the predicted mean is conditional on the values of the random effects. Use the **fixedonly** option (see below) if you want predictions that include only the fixed portion of the model, that is, if you want random effects set to 0.

**xb** calculates the linear prediction $X\beta$ based on the estimated fixed effects (coefficients) in the model. This is equivalent to fixing all random effects in the model to their theoretical (prior) mean value of 0.

**stdp** calculates the standard error of the fixed-effects linear predictor $X\beta$.

**pearson** calculates Pearson residuals. Pearson residuals large in absolute value may indicate a lack of fit. By default, residuals include both the fixed portion and the random portion of the model. The **fixedonly** option modifies the calculation to include the fixed portion only.

**deviance** calculates deviance residuals. Deviance residuals are recommended by McCullagh and Nelder (1989) as having the best properties for examining the goodness of fit of a GLM. They are approximately normally distributed if the model is correctly specified. They may be plotted against the fitted values or against a covariate to inspect the model’s fit. By default, residuals include both the fixed portion and the random portion of the model. The **fixedonly** option modifies the calculation to include the fixed portion only.

**anscombe** calculates Anscombe residuals, which are designed to closely follow a normal distribution. By default, residuals include both the fixed portion and the random portion of the model. The **fixedonly** option modifies the calculation to include the fixed portion only.

**nooffset** is relevant only if you specified **offset(varname)** with **meqrlogit**. It modifies the calculations made by **predict** so that they ignore the offset variable; the linear prediction is treated as $X\beta + Zu$ rather than $X\beta + Zu + offset$.

**fixedonly** modifies predictions to include only the fixed portion of the model, equivalent to setting all random effects equal to 0; see the **mu** option.
margins

Description for margins

margins estimates margins of response for linear predictions.

Menu for margins

Statistics > Postestimation

Syntax for margins

margins [marginlist] [, options]
margins [marginlist], predict(statistic ...) [options]

<table>
<thead>
<tr>
<th>statistic</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>xb</td>
<td>linear predictor for the fixed portion of the model only; the default</td>
</tr>
<tr>
<td>reffects</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>mu</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>stdp</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>pearson</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>deviance</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>anscombe</td>
<td>not allowed with margins</td>
</tr>
</tbody>
</table>

Statistics not allowed with margins are functions of stochastic quantities other than \( \text{e}(b) \).

For the full syntax, see [R] margins.

estat

Description for estat

estat group reports the number of groups and minimum, average, and maximum group sizes for each level of the model. Model levels are identified by the corresponding group variable in the data. Because groups are treated as nested, the information in this summary may differ from what you would get if you used the tabulate command on each group variable individually.

estat recovariance displays the estimated variance–covariance matrix of the random effects for each level in the model. Random effects can be either random intercepts, in which case the corresponding rows and columns of the matrix are labeled as \_cons, or random coefficients, in which case the label is the name of the associated variable in the data.

estat icc displays the intraclass correlation for pairs of latent linear responses at each nested level of the model. Intraclass correlations are available for random-intercept models or for random-coefficient models conditional on random-effects covariates being equal to 0. They are not available for crossed-effects models.
Menu for estat

Statistics > Postestimation

Syntax for estat

Summarize the composition of the nested groups

\texttt{estat group}

Display the estimated random-effects covariance matrix (or matrices)

\texttt{estat recovariance [ , relevel(levelvar) correlation matlist_options ]}

Estimate intraclass correlations

\texttt{estat icc [ , level(#) ]}

Options for estat recovariance

\texttt{relevel(levelvar)} specifies the level in the model for which the random-effects covariance matrix is to be displayed and returned in \texttt{r(cov)}. By default, the covariance matrices for all levels in the model are displayed. \texttt{levelvar} is the name of the model level and is either the name of the variable describing the grouping at that level or is \texttt{_all}, a special designation for a group comprising all the estimation data.

\texttt{correlation} displays the covariance matrix as a correlation matrix and returns the correlation matrix in \texttt{r(corr)}.

\texttt{matlist_options} are style and formatting options that control how the matrix (or matrices) is displayed; see \texttt{[P] matlist} for a list of options that are available.

Option for estat icc

\texttt{level(#)} specifies the confidence level, as a percentage, for confidence intervals. The default is \texttt{level(95)} or as set by \texttt{set level}; see \texttt{[U] 20.7 Specifying the width of confidence intervals}. 
Remarks and examples

Various predictions, statistics, and diagnostic measures are available after fitting a logistic mixed-effects model with \texttt{meqrlogit}. For the most part, calculation centers around obtaining estimates of the subject/group-specific random effects. Random effects are not provided as estimates when the model is fit but instead need to be predicted after estimation. Calculation of intraclass correlations, estimating the dependence between latent linear responses for different levels of nesting, may also be of interest.

\subsection*{Example 1}

Following Rabe-Hesketh and Skrondal (2012, chap. 10), we consider a two-level mixed-effects model for onycholysis (separation of toenail plate from nail bed) among those who contract toenail fungus. The data are obtained from De Backer et al. (1998) and were also studied by Lesaffre and Spiessens (2001). The onycholysis outcome is dichotomously coded as 1 (moderate or severe onycholysis) or 0 (none or mild onycholysis). Fixed-effects covariates include treatment (0: itraconazole; 1: terbinafine), the month of measurement, and their interaction.

We fit the two-level model with \texttt{meqrlogit}:

\begin{verbatim}
. use http://www.stata-press.com/data/r14/toenail
(Onycholysis severity)
. meqrlogit outcome treatment month trt_month || patient:, intpoints(30)
Refining starting values:
  Iteration 0:  log likelihood =  -749.959
  Iteration 1:  log likelihood =  -630.116
  Iteration 2:  log likelihood =  -625.679
Performing gradient-based optimization:
  Iteration 0:  log likelihood =  -625.679
  Iteration 1:  log likelihood =  -625.365
  Iteration 2:  log likelihood =  -625.365
  Iteration 3:  log likelihood =  -625.365
Mixed-effects logistic regression Number of obs =  1908
Group variable:  patient Number of groups = 294
Obs per group:
  min = 1
  avg = 6.5
  max =  7
Integration points = 30 Wald chi2(3) = 150.52
Log likelihood = -625.365 Prob > chi2 =  0.0000

outcome                  Coef.   Std. Err.      z    P>|z|     [95% Conf. Interval]
-------------            -------  --------  ------   --------
treatment       -.1609377    .584208    -0.28  0.783    -1.305964    .984089
month            -.3910603    .0443957   -8.81  0.000    -.4780744   -.3040463
trt_month        -.1368073    .0680235   -2.01  0.044    -.2701310   -.0034836
_cons            -1.618961    .4347771   -3.72  0.000    -2.471108   -.7668132

Random-effects Parameters  Estimate    Std. Err.     [95% Conf. Interval]
patient: Identity
  var(_cons)         16.06538    3.057361     11.06371    23.32818

LR test vs. logistic model: chibar2(01) = 565.22    Prob >= chibar2 = 0.0000
\end{verbatim}
It is of interest to determine the dependence among responses for the same subject (between-subject heterogeneity). Under the latent-linear-response formulation, this dependence can be obtained with the intraclass correlation. Formally, in a two-level random-effects model, the intraclass correlation corresponds to the correlation of latent responses within the same individual and also to the proportion of variance explained by the individual random effect.

In the presence of fixed-effects covariates, `estat icc` reports the residual intraclass correlation, which is the correlation between latent linear responses conditional on the fixed-effects covariates.

We use `estat icc` to estimate the residual intraclass correlation:

```
. estat icc
Residual intraclass correlation

<table>
<thead>
<tr>
<th>Level</th>
<th>ICC</th>
<th>Std. Err.</th>
<th>[95% Conf. Interval]</th>
</tr>
</thead>
<tbody>
<tr>
<td>patient</td>
<td>.830027</td>
<td>.026849</td>
<td>.7707981 .8764046</td>
</tr>
</tbody>
</table>
```

Conditional on treatment and month of treatment, we estimate that latent responses within the same patient have a large correlation of approximately 0.83. Further, 83% of the variance of a latent response is explained by the between-patient variability.

Example 2

In example 3 of [ME] `meqrlogit`, we represented the probability of contraceptive use among Bangladeshi women by using the model (stated with slightly different notation here)

\[
\text{logit}(\pi_{ij}) = \beta_0 \text{rural}_{ij} + \beta_1 \text{urban}_{ij} + \beta_2 \text{age}_{ij} + \\
\phantom{=} \beta_3 \text{child1}_{ij} + \beta_4 \text{child2}_{ij} + \beta_5 \text{child3}_{ij} + a_j \text{rural}_{ij} + b_j \text{urban}_{ij}
\]

where \(\pi_{ij}\) is the probability of contraceptive use, \(j = 1, \ldots, 60\) districts, \(i = 1, \ldots, n_j\) women within each district, and \(a_j\) and \(b_j\) are normally distributed with mean 0 and variance–covariance matrix

\[
\Sigma = \text{Var} \begin{bmatrix} a_j \\ b_j \end{bmatrix} = \begin{bmatrix} \sigma_a^2 & \sigma_{ab} \\ \sigma_{ab} & \sigma_b^2 \end{bmatrix}
\]
. use http://www.stata-press.com/data/r14/bangladesh
   (Bangladesh Fertility Survey, 1989)
. generate byte rural = 1 - urban
. meqrlogit c_use rural urban age child*, noconstant || district: rural urban,
   > noconstant cov(unstructured)

Refining starting values:
Iteration 0:  log likelihood = -1208.3924
Iteration 1:  log likelihood = -1204.1317
Iteration 2:  log likelihood = -1200.6012

Performing gradient-based optimization:
Iteration 0:  log likelihood = -1200.6012
Iteration 1:  log likelihood = -1199.3332
Iteration 2:  log likelihood = -1199.315
Iteration 3:  log likelihood = -1199.315

Mixed-effects logistic regression Number of obs = 1,934
Group variable: district Number of groups = 60

Obs per group:
    min  = 2
    avg  = 32.2
    max  = 118

Integration points = 7 Wald chi2(6) = 120.24
Log likelihood = -1199.315 Prob > chi2 = 0.0000

|            | Coef. | Std. Err. | z     | P>|z|   | [95% Conf. Interval] |
|------------|-------|-----------|-------|-------|-----------------------|
| c_use      |       |           |       |       |                       |
| rural      | -1.71165 | 0.1605618 | -10.66 | 0.000 | -2.026345 -1.396954  |
| urban      | -0.8958623 | 0.1704961 | -5.25  | 0.000 | -1.230028 -0.561961  |
| age        | -0.026415 | 0.008023  | -3.29  | 0.001 | -0.042139 -0.0106902 |
| child1     | 1.13252 | 0.1603285 | 7.06   | 0.000 | 0.818282 1.446758    |
| child2     | 1.357739 | 0.1770522 | 7.67   | 0.000 | 1.010724 1.704755    |
| child3     | 1.353827 | 0.1828801 | 7.40   | 0.000 | 0.9953882 1.712265   |

Random-effects Parameters

<table>
<thead>
<tr>
<th></th>
<th>Estimate</th>
<th>Std. Err.</th>
<th>[95% Conf. Interval]</th>
</tr>
</thead>
<tbody>
<tr>
<td>district: Unstructured</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>var(rural)</td>
<td>.3897439</td>
<td>.1292459</td>
<td>.2034726 .7465393</td>
</tr>
<tr>
<td>var(urban)</td>
<td>.2442965</td>
<td>.1450674</td>
<td>.0762886 .782303</td>
</tr>
<tr>
<td>cov(rural,urban)</td>
<td>-.0161411</td>
<td>.1057469</td>
<td>-.2234012 .1911189</td>
</tr>
</tbody>
</table>

LR test vs. logistic model: chi2(3) = 58.42 Prob > chi2 = 0.0000
Note: LR test is conservative and provided only for reference.

Rather than see the estimated variance components listed as variance and covariances as above,
we can instead see them as correlations and standard deviations in matrix form; that is, we can see
\( \hat{\Sigma} \) as a correlation matrix:

. estat recovariance, correlation

Random-effects correlation matrix for level district

<table>
<thead>
<tr>
<th></th>
<th>rural</th>
<th>urban</th>
</tr>
</thead>
<tbody>
<tr>
<td>rural</td>
<td>1</td>
<td>-.05231</td>
</tr>
<tr>
<td>urban</td>
<td>- -.05231</td>
<td>1</td>
</tr>
</tbody>
</table>

The purpose of using this particular model was to allow for district random effects that were
specific to the rural and urban areas of that district and that could be interpreted as such. We can
obtain predictions of these random effects and their corresponding standard errors,
predict re_rural re_urban, reffects reses(se_rural se_urban)

The order in which we specified the variables to be generated corresponds to the order in which the variance components are listed in `meqrlogit` output. If in doubt, a simple `describe` will show how these newly generated variables are labeled just to be sure.

Having generated estimated random effects and standard errors, we can now list them for the first 10 districts:

```
. by district, sort: generate tolist = (_n==1)
. list district re_rural se_rural re_urban se_urban if district <= 10 & tolist,
> sep(0)
```

<table>
<thead>
<tr>
<th>district</th>
<th>re_rural</th>
<th>se_rural</th>
<th>re_urban</th>
<th>se_urban</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>-.9206641</td>
<td>.3129662</td>
<td>-.5551252</td>
<td>.2321872</td>
</tr>
<tr>
<td>118.</td>
<td>-.0307772</td>
<td>.3784629</td>
<td>.0012746</td>
<td>.4938357</td>
</tr>
<tr>
<td>138.</td>
<td>-.0149148</td>
<td>.6242095</td>
<td>.2257356</td>
<td>.4689535</td>
</tr>
<tr>
<td>140.</td>
<td>-.2684802</td>
<td>.3951617</td>
<td>.5760575</td>
<td>.3970433</td>
</tr>
<tr>
<td>170.</td>
<td>.0787537</td>
<td>.3078451</td>
<td>.004534</td>
<td>.4675104</td>
</tr>
<tr>
<td>209.</td>
<td>-.3842217</td>
<td>.2741989</td>
<td>.272722</td>
<td>.4184852</td>
</tr>
<tr>
<td>274.</td>
<td>-.1742786</td>
<td>.4008164</td>
<td>.0072177</td>
<td>.4938666</td>
</tr>
<tr>
<td>292.</td>
<td>.0447142</td>
<td>.315396</td>
<td>.2256406</td>
<td>.46799</td>
</tr>
<tr>
<td>329.</td>
<td>-.3561363</td>
<td>.3885605</td>
<td>.0733451</td>
<td>.4555067</td>
</tr>
<tr>
<td>352.</td>
<td>-.5368572</td>
<td>.4743089</td>
<td>.0222338</td>
<td>.4939776</td>
</tr>
</tbody>
</table>

Technical note

When these data were first introduced in `ME meqrlogit`, we noted that not all districts contained both urban and rural areas. This fact is somewhat demonstrated by the random effects that are nearly 0 in the above. A closer examination of the data would reveal that district 3 has no rural areas, and districts 2, 7, and 10 have no urban areas.

The estimated random effects are not exactly 0 in these cases because of the correlation between urban and rural effects. For instance, if a district has no urban areas, it can still yield a nonzero (albeit small) random-effects estimate for a nonexistent urban area because of the correlation with its rural counterpart.

Had we imposed an independent covariance structure in our model, the estimated random effects in the cases in question would be exactly 0.

Technical note

The estimated standard errors produced above with the `reses()` option are conditional on the values of the estimated model parameters: $\beta$ and the components of $\Sigma$. Their interpretation is therefore not one of standard sample-to-sample variability but instead one that does not incorporate uncertainty in the estimated model parameters; see `Methods and formulas`.

That stated, conditional standard errors can still be used as a measure of relative precision, provided that you keep this caveat in mind.
Example 3

Continuing with example 2, we can obtain predicted probabilities, the default prediction:

```
. predict p
(option mu assumed; predicted means)
```

These predictions are based on a linear predictor that includes both the fixed effects and the random effects due to district. Specifying the `fixedonly` option gives predictions that set the random effects to their prior mean of 0. Below we compare both over the first 20 observations:

```
. predict p_fixed, fixedonly
(option mu assumed; predicted means)
. list c_use p p_fixed age child* in 1/20
```

<table>
<thead>
<tr>
<th>c_use</th>
<th>p</th>
<th>p_fixed</th>
<th>age</th>
<th>child1</th>
<th>child2</th>
<th>child3</th>
</tr>
</thead>
<tbody>
<tr>
<td>no</td>
<td>.3579543</td>
<td>.4927183</td>
<td>18.44</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>no</td>
<td>.2134724</td>
<td>.3210403</td>
<td>-5.56</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>no</td>
<td>.4672256</td>
<td>.6044016</td>
<td>1.44</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>no</td>
<td>.4206505</td>
<td>.5584864</td>
<td>8.44</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>no</td>
<td>.2510909</td>
<td>.3687281</td>
<td>-13.56</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>no</td>
<td>.2412878</td>
<td>.3565185</td>
<td>-11.56</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>no</td>
<td>.3579543</td>
<td>.4927183</td>
<td>18.44</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>no</td>
<td>.4992191</td>
<td>.6345999</td>
<td>-3.56</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>no</td>
<td>.4572049</td>
<td>.594723</td>
<td>-5.56</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>no</td>
<td>.4662518</td>
<td>.6034657</td>
<td>1.44</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>yes</td>
<td>.2412878</td>
<td>.3565185</td>
<td>-11.56</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>no</td>
<td>.2004691</td>
<td>.3040173</td>
<td>-2.56</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>no</td>
<td>.4506573</td>
<td>.5883407</td>
<td>-4.56</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>no</td>
<td>.4400747</td>
<td>.5779263</td>
<td>5.44</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>no</td>
<td>.4794194</td>
<td>.6160359</td>
<td>-0.56</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>yes</td>
<td>.4465936</td>
<td>.5843561</td>
<td>4.44</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>no</td>
<td>.2134724</td>
<td>.3210403</td>
<td>-5.56</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>yes</td>
<td>.4794194</td>
<td>.6160359</td>
<td>-0.56</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>yes</td>
<td>.4637673</td>
<td>.6010735</td>
<td>-6.56</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>no</td>
<td>.5001973</td>
<td>.6355067</td>
<td>-3.56</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

Technical note

Out-of-sample predictions are permitted after `meqreg`, but if these predictions involve estimated random effects, the integrity of the estimation data must be preserved. If the estimation data have changed since the model was fit, `predict` will be unable to obtain predicted random effects that are appropriate for the fitted model and will give an error. Thus to obtain out-of-sample predictions that contain random-effects terms, be sure that the data for these predictions are in observations that augment the estimation data.
Continuing with example 2, we can also compute intraclass correlations for the model.

In the presence of random-effects covariates, the intraclass correlation is no longer constant and depends on the values of the random-effects covariates. In this case, `estat icc` reports conditional intraclass correlations assuming 0 values for all random-effects covariates. For example, in a two-level model, this conditional correlation represents the correlation of the latent responses for two measurements on the same subject, both of which have random-effects covariates equal to 0. Similarly to the interpretation of intercept variances in random-coefficient models (Rabe-Hesketh and Skrondal 2012, chap. 16), interpretation of this conditional intraclass correlation relies on the usefulness of the 0 baseline values of random-effects covariates. For example, mean centering of the covariates is often used to make a 0 value a useful reference.

Estimation of the conditional intraclass correlation in the Bangladeshi contraceptive study setting of example 2 is of interest. In example 2, the random-effects covariates `rural` and `urban` for the random level `district` are mutually exclusive indicator variables and can never be simultaneously 0. Thus we could not use `estat icc` to estimate the conditional intraclass correlation for this model, because `estat icc` requires that the random intercept is included in all random-effects specifications.

Instead, we consider an alternative model for the Bangladeshi contraceptive study. In example 2 of [ME] `meqrlogit`, we represented the probability of contraceptive use among Bangladeshi women with fixed-effects for urban residence (`urban`), age (`age`), and the number of children (`child1`–`child3`). The random effects for urban and rural residence are represented as a random slope for urban residence and a random intercept at the district level.

We fit the model with `meqrlogit`:

```
use http://www.stata-press.com/data/r14/bangladesh, clear
(Bangladesh Fertility Survey, 1989)
.meqrlogit c_use urban age child* || district: urban, covariance(unstructured)
Refining starting values:
Iteration 0: log likelihood = -1215.8594 (not concave)
Iteration 1: log likelihood = -1204.0802
Iteration 2: log likelihood = -1199.7994
Performing gradient-based optimization:
Iteration 0: log likelihood = -1199.7994
(output omitted)
Iteration 4: log likelihood = -1199.315
Mixed-effects logistic regression Number of obs = 1,934
Group variable: district Number of groups = 60
Obs per group:
    min = 2
    avg = 32.2
    max = 118
Integration points = 7 Wald chi2(5) = 97.50
Log likelihood = -1199.315 Prob > chi2 = 0.0000
```

| c_use | Coef.  | Std. Err. | z     | P>|z|   |  [95% Conf. Interval] |
|-------|--------|-----------|-------|-------|----------------------|
| urban | .8157872 | .1715519 | 4.76  | 0.000 | .4795516 – 1.152023 |
| age   | -.026415 | .008023  | -3.29 | 0.001 | -.0421398 – -.0106902 |
| child1| 1.13252  | .1603285 | 7.06  | 0.000 | .818282 – 1.446758  |
| child2| 1.357739 | .1770522 | 7.67  | 0.000 | 1.010724 – 1.704755 |
| child3| 1.353827 | .1828801 | 7.40  | 0.000 | .9953882 – 1.712265 |
| _cons | -1.71165 | .1605617 | -10.66| 0.000 | -2.026345 – -1.396954|
Random-effects Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Estimate</th>
<th>Std. Err.</th>
<th>[95% Conf. Interval]</th>
</tr>
</thead>
<tbody>
<tr>
<td>district: Unstructured</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>var(urban)</td>
<td>.6663222</td>
<td>.3224715</td>
<td>.2580709 1.7204</td>
</tr>
<tr>
<td>var(_cons)</td>
<td>.3897434</td>
<td>.1292458</td>
<td>.2034723 .7465387</td>
</tr>
<tr>
<td>cov(urban,_cons)</td>
<td>-.4058846</td>
<td>.1755418</td>
<td>-.7499402 -.0618289</td>
</tr>
</tbody>
</table>

LR test vs. logistic model: chi2(3) = 58.42 Prob > chi2 = 0.0000
Note: LR test is conservative and provided only for reference.

We use `estat icc` to estimate the intraclass correlation conditional on `urban` being equal to 0:

```stata
. estat icc
```
Conditional intraclass correlation

<table>
<thead>
<tr>
<th>Level</th>
<th>ICC</th>
<th>Std. Err.</th>
<th>[95% Conf. Interval]</th>
</tr>
</thead>
<tbody>
<tr>
<td>district</td>
<td>.1059197</td>
<td>.0314044</td>
<td>.0582458 .1849513</td>
</tr>
</tbody>
</table>

Note: ICC is conditional on zero values of random-effects covariates.

This estimate suggests that the latent responses are not strongly correlated for rural residents (`urban == 0`) within the same district, conditional on the fixed-effects covariates.

Example 5

In example 4 of [ME] `meqrlogit`, we fit a three-level model for the cognitive ability of schizophrenia patients as compared with their relatives and a control. Fixed-effects covariates include the difficulty of the test, `difficulty`, and an individual’s category, `group` (control, family member of patient, or patient). Family units (`family`) represent the third nesting level, and individual subjects (`subject`) represent the second nesting level. Three measurements were taken on all but one subject, one for each difficulty measure.

We fit the model with `meqrlogit`:

```stata
. use http://www.stata-press.com/data/r14/towerlondon
    (Tower of London data)
. meqrlogit dtlm difficulty i.group || family: || subject:
```
Refining starting values:

Iteration 0:  log likelihood = -310.28433
Iteration 1:  log likelihood = -306.42785  (not concave)
Iteration 2:  log likelihood = -305.25996
Performing gradient-based optimization:

Iteration 0:  log likelihood = -305.25996
Iteration 1:  log likelihood = -305.12097
Iteration 2:  log likelihood = -305.12043
Iteration 3:  log likelihood = -305.12043

Mixed-effects logistic regression

<table>
<thead>
<tr>
<th>Group Variable</th>
<th>No. of Groups</th>
<th>Observations per Group</th>
<th>Integration Points</th>
</tr>
</thead>
<tbody>
<tr>
<td>family</td>
<td>118</td>
<td>2</td>
<td>7</td>
</tr>
<tr>
<td>subject</td>
<td>226</td>
<td>2</td>
<td>7</td>
</tr>
</tbody>
</table>

Number of obs = 677
We can use `estat icc` to estimate the residual intraclass correlation (conditional on the difficulty level and the individual’s category) between the latent responses of subjects within the same family or between the latent responses of the same subject and family:

```
. estat icc

Residual intraclass correlation

<table>
<thead>
<tr>
<th>Level</th>
<th>ICC</th>
<th>Std. Err.</th>
<th>[95% Conf. Interval]</th>
</tr>
</thead>
<tbody>
<tr>
<td>family</td>
<td>.1139052</td>
<td>.0997976</td>
<td>.0181741 .4716556</td>
</tr>
<tr>
<td>subject</td>
<td>family</td>
<td>.3416289</td>
<td>.0889531</td>
</tr>
</tbody>
</table>
```

`estat icc` reports two intraclass correlations for this three-level nested model. The first is the level-3 intraclass correlation at the family level, the correlation between latent measurements of the cognitive ability in the same family. The second is the level-2 intraclass correlation at the subject-within-family level, the correlation between the latent measurements of cognitive ability in the same subject and family.

There is not a strong correlation between individual realizations of the latent response, even within the same subject.
Stored results

`estat recovariance` stores the following in `r()`:

Scalars
- \texttt{r(relevels)}: number of levels

Matrices
- \texttt{r(Cov#)}: level-# random-effects covariance matrix
- \texttt{r(Corr#)}: level-# random-effects correlation matrix (if option \texttt{correlation} was specified)

For a \(G\)-level nested model, \# can be any integer between 2 and \(G\).

`estat icc` stores the following in `r()`:

Scalars
- \texttt{r(icc#)}: level-# intraclass correlation
- \texttt{r(se#)}: standard errors of level-# intraclass correlation
- \texttt{r(level)}: confidence level of confidence intervals

Macros
- \texttt{r(label#)}: label for level \#

Matrices
- \texttt{r(ci#)}: vector of confidence intervals (lower and upper) for level-# intraclass correlation

For a \(G\)-level nested model, \# can be any integer between 2 and \(G\).

Methods and formulas

Methods and formulas are presented under the following headings:

- \texttt{Prediction}
- \texttt{Intraclass correlations}

Prediction

Continuing the discussion in \textit{Methods and formulas} of \textit{[ME] meqrlogit}, and using the definitions and formulas defined there, we begin by considering the prediction of the random effects \(u_j\) for the \(j\)th cluster in a two-level model.

Given a set of estimated \textit{meqrlogit} parameters \((\hat{\beta}, \hat{\Sigma})\), a profile likelihood in \(u_j\) is derived from the joint distribution \(f(y_j, u_j)\) as

\[
L_j(u_j) = \exp \left\{ c(y_j, r_j) \right\} (2\pi)^{-q/2} |\hat{\Sigma}|^{-1/2} \exp \left\{ g(\hat{\beta}, \hat{\Sigma}, u_j) \right\}
\]

(1)

The conditional maximum likelihood estimator of \(u_j\)—conditional on fixed \((\hat{\beta}, \hat{\Sigma})\)—is the maximizer of \(L_j(u_j)\) or, equivalently, the value of \(\hat{u}_j\) that solves

\[
0 = g'(\hat{\beta}, \hat{\Sigma}, \hat{u}_j) = Z_j' \left\{ y_j - m(\hat{\beta}, \hat{u}_j) \right\} - \hat{\Sigma}^{-1} \hat{u}_j
\]

Because (1) is proportional to the conditional density \(f(u_j|y_j)\), you can also refer to \(\hat{u}_j\) as the conditional mode (or posterior mode if you lean toward Bayesian terminology). Regardless, you are referring to the same estimator.
Conditional standard errors for the estimated random effects are derived from standard theory of maximum likelihood, which dictates that the asymptotic variance matrix of \( \hat{u}_j \) is the negative inverse of the Hessian, which is estimated as

\[
 g'' \left( \hat{\beta}, \hat{\Sigma}, \hat{u}_j \right) = - \left\{ Z'_j V(\hat{\beta}, \hat{u}_j) Z_j + \hat{\Sigma}^{-1} \right\}
\]

Similar calculations extend to models with more than one level of random effects; see Pinheiro and Chao (2006).

For any observation \( i \) in the \( j \)th cluster in a two-level model, define the linear predictor as

\[
 \hat{\eta}_{ij} = x_{ij} \hat{\beta} + z_{ij} \hat{u}_j
\]

In a three-level model, for the \( i \)th observation within the \( j \)th level-two cluster within the \( k \)th level-three cluster,

\[
 \hat{\eta}_{ijk} = x_{ijk} \hat{\beta} + z_{ijk}^{(3)} \hat{u}_{jk}^{(3)} + z_{ijk}^{(2)} \hat{u}_{jk}^{(2)}
\]

where \( z^{(p)} \) and \( u^{(p)} \) refer to the level \( p \) design variables and random effects, respectively. For models with more than three levels, the definition of \( \hat{\eta} \) extends in the natural way, with only the notation becoming more complicated.

If the fixedonly option is specified, \( \hat{\eta} \) contains the linear predictor for only the fixed portion of the model, for example, in a two-level model \( \hat{\eta}_{ij} = x_{ij} \hat{\beta} \). In what follows, we assume a two-level model, with the only necessary modification for multilevel models being the indexing.

The predicted mean conditional on the random effects \( \hat{u}_j \) is

\[
 \hat{\mu}_{ij} = r_{ij} H(\hat{\eta}_{ij})
\]

Pearson residuals are calculated as

\[
 \nu_{ij}^P = \frac{y_{ij} - \hat{\mu}_{ij}}{\left\{ V(\hat{\mu}_{ij}) \right\}^{1/2}}
\]

for \( V(\hat{\mu}_{ij}) = \hat{\mu}_{ij}(1 - \hat{\mu}_{ij}/r_{ij}) \).

Deviance residuals are calculated as

\[
 \nu_{ij}^D = \text{sign}(y_{ij} - \hat{\mu}_{ij}) \sqrt{\hat{d}_{ij}^2}
\]

where

\[
 \hat{d}_{ij}^2 = \begin{cases} 2r_{ij} \log \left( \frac{r_{ij}}{r_{ij} - \hat{\mu}_{ij}} \right) & \text{if } y_{ij} = 0 \\ 2y_{ij} \log \left( \frac{y_{ij}}{\hat{\mu}_{ij}} \right) + 2(r_{ij} - y_{ij}) \log \left( \frac{r_{ij} - y_{ij}}{r_{ij} - \hat{\mu}_{ij}} \right) & \text{if } 0 < y_{ij} < r_{ij} \\ 2r_{ij} \log \left( \frac{r_{ij}}{\hat{\mu}_{ij}} \right) & \text{if } y_{ij} = r_{ij} \end{cases}
\]

Anscombe residuals are calculated as

\[
 \nu_{ij}^A = \frac{3 \left\{ y_{ij}^{2/3} H(y_{ij}/r_{ij}) - \hat{\mu}_{ij}^{2/3} H(\hat{\mu}_{ij}/r_{ij}) \right\}}{2 \left( \hat{\mu}_{ij} - \hat{\mu}_{ij}^2 / r_{ij} \right)^{1/6}}
\]

where \( H(t) \) is a specific univariate case of the Hypergeometric2F1 function (Wolfram 1999, 771–772). For Anscombe residuals for binomial regression, the specific form of the Hypergeometric2F1 function that we require is \( H(t) = _2F_1(2/3, 1/3, 5/3, t) \).

For a discussion of the general properties of the above residuals, see Hardin and Hilbe (2012, chap. 4).
Intraclss correlations

Consider a simple, two-level random-intercept model, stated in terms of a latent linear response, where only \( y_{ij} = I(y_{ij}^* > 0) \) is observed for the latent variable,

\[ y_{ij}^* = \beta + u_j^{(2)} + \epsilon_{ij}^{(1)} \]

with \( i = 1, \ldots, n_j \) and level-2 groups \( j = 1, \ldots, M \). Here \( \beta \) is an unknown fixed intercept, \( u_j^{(2)} \) is a level-2 random intercept, and \( \epsilon_{ij}^{(1)} \) is a level-1 error term. Errors are assumed to be logistic with mean 0 and variance \( \sigma_1^2 = \pi^2/3 \); random intercepts are assumed to be normally distributed with mean 0 and variance \( \sigma_2^2 \) and to be independent of error terms.

The intraclass correlation for this model is

\[ \rho = \text{Corr}(y_{ij}^*, y_{i'j}^*) = \frac{\sigma_2^2}{\pi^2/3 + \sigma_2^2} \]

It corresponds to the correlation between the latent responses \( i \) and \( i' \) from the same group \( j \).

Now consider a three-level nested random-intercept model,

\[ y_{ijk}^* = \beta + u_{jk}^{(2)} + u_k^{(3)} + \epsilon_{ijk}^{(1)} \]

for measurements \( i = 1, \ldots, n_{jk} \) and level-2 groups \( j = 1, \ldots, M_{1k} \) nested within level-3 groups \( k = 1, \ldots, M_2 \). Here \( u_{jk}^{(2)} \) is a level-2 random intercept, \( u_k^{(3)} \) is a level-3 random intercept, and \( \epsilon_{ijk}^{(1)} \) is a level-1 error term. The error terms have a logistic distribution with mean 0 and variance \( \sigma_1^2 = \pi^2/3 \). The random intercepts are assumed to be normally distributed with mean 0 and variances \( \sigma_2^2 \) and \( \sigma_3^2 \), respectively, and to be mutually independent. The error terms are also independent of the random intercepts.

We can consider two types of intraclass correlations for this model. We will refer to them as level-2 and level-3 intraclass correlations. The level-3 intraclass correlation is

\[ \rho^{(3)} = \text{Corr}(y_{ijk}^*, y_{i'jk}^*) = \frac{\sigma_3^2}{\pi^2/3 + \sigma_2^2 + \sigma_3^2} \]

This is the correlation between latent responses \( i \) and \( i' \) from the same level-3 group \( k \) and from different level-2 groups \( j \) and \( j' \).

The level-2 intraclass correlation is

\[ \rho^{(2)} = \text{Corr}(y_{ijk}^*, y_{i'jk}^*) = \frac{\sigma_2^2 + \sigma_3^2}{\pi^2/3 + \sigma_2^2 + \sigma_3^2} \]

This is the correlation between latent responses \( i \) and \( i' \) from the same level-3 group \( k \) and level-2 group \( j \). (Note that level-1 intraclass correlation is undefined.)

More generally, for a \( G \)-level nested random-intercept model, the \( g \)-level intraclass correlation is defined as

\[ \rho^{(g)} = \frac{\sum_{l=g}^{G} \sigma_l^2}{\pi^2/3 + \sum_{l=2}^{G} \sigma_l^2} \]
The above formulas also apply in the presence of fixed-effects covariates $X$ in a random-effects model. In this case, intraclass correlations are conditional on fixed-effects covariates and are referred to as residual intraclass correlations. *estat icc* also uses the same formulas to compute intraclass correlations for random-coefficient models, assuming 0 baseline values for the random-effects covariates, and labels them as conditional intraclass correlations.

Intraclass correlations will always fall in $[0,1]$ because variance components are nonnegative. To accommodate the range of an intraclass correlation, we use the logit transformation to obtain confidence intervals. We use the delta method to estimate the standard errors of the intraclass correlations.

Let $\hat{\rho}(g)$ be a point estimate of the intraclass correlation and $\hat{SE}(\hat{\rho}(g))$ be its standard error. The $(1-\alpha) \times 100\%$ confidence interval for logit$(\rho(g))$ is

$$\text{logit}(\hat{\rho}(g)) \pm z_{\alpha/2} \frac{\hat{SE}(\hat{\rho}(g))}{\hat{\rho}(g)(1-\hat{\rho}(g))}$$

where $z_{\alpha/2}$ is the $1-\alpha/2$ quantile of the standard normal distribution and logit$(x) = \ln\{x/(1-x)\}$. Let $k_u$ be the upper endpoint of this interval, and let $k_l$ be the lower. The $(1-\alpha) \times 100\%$ confidence interval for $\rho(g)$ is then given by

$$\left(\frac{1}{1+e^{-k_l}}, \frac{1}{1+e^{-k_u}}\right)$$

References


Also see

[ME] meqrlogit — Multilevel mixed-effects logistic regression (QR decomposition)

[U] 20 Estimation and postestimation commands
**Description**

`megrpoisson`, like `mepoisson`, fits mixed-effects models for count responses, for which the conditional distribution of the response given the random effects is assumed to be Poisson.

`megrpoisson` provides an alternative estimation method that uses the QR decomposition of the variance-components matrix. This method may aid convergence when variance components are near the boundary of the parameter space.

**Quick start**

Two-level Poisson regression of \(y\) on \(x\) with random intercepts by \(\text{lev2}\) using QR decomposition

```
meqrpoisson y x || lev2:
```

As above, but report incidence-rate ratios

```
meqrpoisson y x || lev2:, irr
```

Add indicator variables for each level of categorical variable \(a\) and random coefficients on \(x\)

```
meqrpoisson y x i.a || lev2: x, irr
```

Three-level random-intercept model of \(y\) on \(x\) with \(\text{lev2}\) nested within \(\text{lev3}\)

```
meqrpoisson y x || lev3: || lev2:
```
Syntax

```
meqrpoisson depvar fe_equation || re_equation [|| re_equation ...] [, options]
```

where the syntax of `fe_equation` is

```
[ indepvars ] [ if ] [ in ] [, fe_options]
```

and the syntax of `re_equation` is one of the following:

for random coefficients and intercepts

```
levelvar: [ varlist ] [, re_options]
```

for random effects among the values of a factor variable

```
levelvar: R.varname [, re_options]
```

`levelvar` is a variable identifying the group structure for the random effects at that level or is `all` representing one group comprising all observations.

### fe_options

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>noconstant</code></td>
</tr>
<tr>
<td><code>exposure(varname_e)</code></td>
</tr>
<tr>
<td><code>offset(varname_o)</code></td>
</tr>
</tbody>
</table>

### re_options

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>covariance(vartype)</code></td>
</tr>
<tr>
<td><code>noconstant</code></td>
</tr>
<tr>
<td><code>collinear</code></td>
</tr>
</tbody>
</table>
## Reporting

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>level(#)</code></td>
<td>set confidence level; default is <code>level(95)</code></td>
</tr>
<tr>
<td><code>irr</code></td>
<td>report fixed-effects coefficients as incidence-rate ratios</td>
</tr>
<tr>
<td><code>variance</code></td>
<td>show random-effects parameter estimates as variances and covariances</td>
</tr>
<tr>
<td><code>stddeviations</code></td>
<td>show random-effects parameter estimates as standard deviations</td>
</tr>
<tr>
<td><code>norettable</code></td>
<td>suppress random-effects table</td>
</tr>
<tr>
<td><code>nofetable</code></td>
<td>suppress fixed-effects table</td>
</tr>
<tr>
<td><code>estmetric</code></td>
<td>show parameter estimates in the estimation metric</td>
</tr>
<tr>
<td><code>noheader</code></td>
<td>suppress output header</td>
</tr>
<tr>
<td><code>nogroup</code></td>
<td>suppress table summarizing groups</td>
</tr>
<tr>
<td><code>nolrtest</code></td>
<td>do not perform likelihood-ratio test comparing with Poisson regression</td>
</tr>
</tbody>
</table>

## Display Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>display_options</code></td>
<td>control columns and column formats, row spacing, line width, display of omitted variables and base and empty cells, and factor-variable labeling</td>
</tr>
</tbody>
</table>

## Integration

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>intpoints(# [#...])</code></td>
<td>set the number of integration (quadrature) points; default is <code>intpoints(7)</code></td>
</tr>
<tr>
<td><code>laplace</code></td>
<td>use Laplacian approximation; equivalent to <code>intpoints(1)</code></td>
</tr>
</tbody>
</table>

## Maximization

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>maximize_options</code></td>
<td>control the maximization process; seldom used</td>
</tr>
<tr>
<td><code>retolerance(#)</code></td>
<td>tolerance for random-effects estimates; default is <code>retolerance(1e-8)</code>; seldom used</td>
</tr>
<tr>
<td><code>reiterate(#)</code></td>
<td>maximum number of iterations for random-effects estimation; default is <code>reiterate(50)</code>; seldom used</td>
</tr>
<tr>
<td><code>matsqrt</code></td>
<td>parameterize variance components using matrix square roots; the default</td>
</tr>
<tr>
<td><code>matlog</code></td>
<td>parameterize variance components using matrix logarithms</td>
</tr>
<tr>
<td><code>refineopts(maximize_options)</code></td>
<td>control the maximization process during refinement of starting values</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>coeflegend</code></td>
<td>display legend instead of statistics</td>
</tr>
</tbody>
</table>

## Vartype

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>independent</code></td>
<td>one unique variance parameter per random effect, all covariances 0; the default unless the R. notation is used</td>
</tr>
<tr>
<td><code>exchangeable</code></td>
<td>equal variances for random effects, and one common pairwise covariance</td>
</tr>
<tr>
<td><code>identity</code></td>
<td>equal variances for random effects, all covariances 0; the default if the R. notation is used</td>
</tr>
<tr>
<td><code>unstructured</code></td>
<td>all variances and covariances to be distinctly estimated</td>
</tr>
</tbody>
</table>
indevars may contain factor variables; see [U] 11.4.3 Factor variables.

indevars and varlist may contain time-series operators; see [U] 11.4.4 Time-series varlists.

bootstrap, by, jackknife, mi estimate, rolling, and statsby are allowed; see [U] 11.1.10 Prefix commands.

coeflegend does not appear in the dialog box.

See [U] 20 Estimation and postestimation commands for more capabilities of estimation commands.

### Options

- **noconstant** suppresses the constant (intercept) term and may be specified for the fixed-effects equation and for any of or all the random-effects equations.

- **exposure(varname_e)** specifies a variable that reflects the amount of exposure over which the depvar events were observed for each observation; \( \ln(varname_e) \) is included in the fixed-effects portion of the model with the coefficient constrained to be 1.

- **offset(varname_o)** specifies that varname_o be included in the fixed-effects portion of the model with the coefficient constrained to be 1.

- **covariance(vartype)** specifies the structure of the covariance matrix for the random effects and may be specified for each random-effects equation. vartype is one of the following: independent, exchangeable, identity, and unstructured.
  - **covariance(independent)** covariance structure allows for a distinct variance for each random effect within a random-effects equation and assumes that all covariances are 0. The default is **covariance(independent)**, except when the R. notation is used, in which case the default is **covariance(identity)** and only covariance(identity) and covariance(exchangeable) are allowed.
  - **covariance(exchangeable)** structure specifies one common variance for all random effects and one common pairwise covariance.
  - **covariance(identity)** is short for “multiple of the identity”; that is, all variances are equal and all covariances are 0.
  - **covariance(unstructured)** allows for all variances and covariances to be distinct. If an equation consists of \( p \) random-effects terms, the unstructured covariance matrix will have \( p(p + 1)/2 \) unique parameters.

- **collinear** specifies that meqrpoisson not omit collinear variables from the random-effects equation.
  Usually, there is no reason to leave collinear variables in place; in fact, doing so usually causes the estimation to fail because of the matrix singularity caused by the collinearity. However, with certain models (for example, a random-effects model with a full set of contrasts), the variables may be collinear, yet the model is fully identified because of restrictions on the random-effects covariance structure. In such cases, using the **collinear** option allows the estimation to take place with the random-effects equation intact.

- **level(#)**; see [R] estimation options.

  \( \text{irr} \) reports estimated fixed-effects coefficients transformed to incidence-rate ratios, that is, \( \exp(\beta) \) rather than \( \beta \). Standard errors and confidence intervals are similarly transformed. This option affects how results are displayed, not how they are estimated. \( \text{irr} \) may be specified at estimation or upon replay.

- **variance**, the default, displays the random-effects parameter estimates as variances and covariances.
stddeviations displays the random-effects parameter estimates as standard deviations and correlations.

noretable suppresses the random-effects table.

nofetable suppresses the fixed-effects table.

estmetric displays all parameter estimates in the estimation metric. Fixed-effects estimates are unchanged from those normally displayed, but random-effects parameter estimates are displayed as log-standard deviations and hyperbolic arctangents of correlations, with equation names that organize them by model level.

noheader suppresses the output header, either at estimation or upon replay.

nogroup suppresses the display of group summary information (number of groups, average group size, minimum, and maximum) from the output header.

nolrtest prevents meqrpoisson from performing a likelihood-ratio test that compares the mixed-effects Poisson model with standard (marginal) Poisson regression. This option may also be specified upon replay to suppress this test from the output.

display_options: noci, nopvalues, noomitted, vsquish, noemptycells, baselevels, allbaselevels, nofvlabel, fvwrap(§), fvwrapon(style), cformat(%fmt), pformat(%fmt), sformat(%fmt), and nolstretch; see [R] estimation options.

Integration

intpoints(# | # | ... |) sets the number of integration points for adaptive Gaussian quadrature. The more integration points, the more accurate the approximation to the log likelihood. However, computation time increases with the number of quadrature points, and in models with many levels or many random coefficients, this increase can be substantial.

You may specify one number of integration points applying to all levels of random effects in the model, or you may specify distinct numbers of points for each level. intpoints(7) is the default; that is, by default seven quadrature points are used for each level.

laplace specifies that log likelihoods be calculated using the Laplacian approximation, equivalent to adaptive Gaussian quadrature with one integration point for each level in the model; laplace is equivalent to intpoints(1). Computation time increases as a function of the number of quadrature points raised to a power equaling the dimension of the random-effects specification. The computational time saved by using laplace can thus be substantial, especially when you have many levels or random coefficients.

The Laplacian approximation has been known to produce biased parameter estimates, but the bias tends to be more prominent in the estimates of the variance components rather than in the estimates of the fixed effects. If your interest lies primarily with the fixed-effects estimates, the Laplace approximation may be a viable faster alternative to adaptive quadrature with multiple integration points.

When the R. varname notation is used, the dimension of the random effects increases by the number of distinct values of varname. Even when this number is small to moderate, it increases the total random-effects dimension to the point where estimation with more than one quadrature point is prohibitively intensive.

For this reason, when you use the R. notation in your random-effects equations, the laplace option is assumed. You can override this behavior by using the intpoints() option, but doing so is not recommended.
Maximization

\texttt{maximize} \ options: \ \texttt{difficult}, \ \texttt{technique(algorithm\_spec)}, \ \texttt{iterate(#)}, \ \texttt{[no]log}, \ \texttt{trace}, \ \texttt{gradient}, \ \texttt{showstep}, \ \texttt{hessian}, \ \texttt{showtolerance}, \ \texttt{tolerance(#)}, \ \texttt{ltolerance(#)}, \ \texttt{nrtolerance(#)}, \ \texttt{nonrtolerance}, \ and \ \texttt{from(init\_specs)}; \ see \ [R] \ \texttt{maximize}. \ Those \ that \ require \ special \ mention \ for \ \texttt{meqrpoisson} \ are \ listed \ below.

For \ the \ \texttt{technique()} \ option, \ the \ default \ is \ \texttt{technique(nr)}. \ The \ \texttt{bhhh} \ algorithm \ may \ not \ be \ specified.

\texttt{from(init\_specs)} \ is \ particularly \ useful \ when \ combined \ with \ \texttt{refineopts(iterate(0))} \ (see \ the \ description \ below), \ which \ bypasses \ the \ initial \ optimization \ stage.

\texttt{retolerance(#)} \ specifies \ the \ convergence \ tolerance \ for \ the \ estimated \ random \ effects \ used \ by \ adaptive \ Gaussian \ quadrature. \ Although \ not \ estimated \ as \ model \ parameters, \ random-effects \ estimators \ are \ used \ to \ adapt \ the \ quadrature \ points. \ Estimating \ these \ random \ effects \ is \ an \ iterative \ procedure, \ with \ convergence \ declared \ when \ the \ maximum \ relative \ change \ in \ the \ random \ effects \ is \ less \ than \ \texttt{retolerance()}. \ The \ default \ is \ \texttt{retolerance(1e-8)}. \ You \ should \ seldom \ have \ to \ use \ this \ option.

\texttt{reiterate(#)} \ specifies \ the \ maximum \ number \ of \ iterations \ used \ when \ estimating \ the \ random \ effects \ to \ be \ used \ in \ adapting \ the \ Gaussian \ quadrature \ points; \ see \ the \ \texttt{retolerance()} \ option. \ The \ default \ is \ \texttt{reiterate(50)}. \ You \ should \ seldom \ have \ to \ use \ this \ option.

\texttt{matsqrt} \ (the \ default), \ during \ optimization, \ parameterizes \ variance \ components \ by \ using \ the \ matrix \ square \ roots \ of \ the \ variance–covariance \ matrices \ formed \ by \ these \ components \ at \ each \ model \ level.

\texttt{matlog}, \ during \ optimization, \ parameterizes \ variance \ components \ by \ using \ the \ matrix \ logarithms \ of \ the \ variance–covariance \ matrices \ formed \ by \ these \ components \ at \ each \ model \ level.

The \ \texttt{matsqrt} \ parameterization \ ensures \ that \ variance–covariance \ matrices \ are \ positive \ semidefinite, \ while \ \texttt{matlog} \ ensures \ matrices \ that \ are \ positive \ definite. \ For \ most \ problems, \ the \ matrix \ square \ root \ is \ more \ stable \ near \ the \ boundary \ of \ the \ parameter \ space. \ However, \ if \ convergence \ is \ problematic, \ one \ option \ may \ be \ to \ try \ the \ alternate \ \texttt{matlog} \ parameterization. \ When \ convergence \ is \ not \ an \ issue, \ both \ parameterizations \ yield \ equivalent \ results.

\texttt{refineopts(maximize\_options)} \ controls \ the \ maximization \ process \ during \ the \ refinement \ of \ starting \ values. \ Estimation \ in \ \texttt{meqrpoisson} \ takes \ place \ in \ two \ stages. \ In \ the \ first \ stage, \ starting \ values \ are \ refined \ by \ holding \ the \ quadrature \ points \ fixed \ between \ iterations. \ During \ the \ second \ stage, \ quadrature \ points \ are \ adapted \ with \ each \ evaluation \ of \ the \ log \ likelihood. \ Maximization \ options \ specified \ within \ \texttt{refineopts()} \ control \ the \ first \ stage \ of \ optimization; \ that \ is, \ they \ control \ the \ refining \ of \ starting \ values.

\texttt{maximize\_options} \ specified \ outside \ \texttt{refineopts()} \ control \ the \ second \ stage.

The \ one \ exception \ to \ the \ above \ rule \ is \ the \ \texttt{nolog} \ option, \ which \ when \ specified \ outside \ \texttt{refineopts()} \ applies \ globally.

\texttt{from(init\_specs)} \ is \ not \ allowed \ within \ \texttt{refineopts()} \ and \ instead \ must \ be \ specified \ globally.

Refining \ starting \ values \ helps \ make \ the \ iterations \ of \ the \ second \ stage \ (those \ that \ lead \ toward \ the \ solution) \ more \ numerically \ stable. \ In \ this \ regard, \ of \ particular \ interest \ is \ \texttt{refineopts(iterate(#))}, \ with \ two \ iterations \ being \ the \ default. \ Should \ the \ maximization \ fail \ because \ of \ instability \ in \ the \ Hessian \ calculations, \ one \ possible \ solution \ may \ be \ to \ increase \ the \ number \ of \ iterations \ here.

The \ following \ option \ is \ available \ with \ \texttt{meqrpoisson} \ but \ is \ not \ shown \ in \ the \ dialog \ box:

\texttt{coeflegend}; \ see \ [R] \ \texttt{estimation\_options}.
Remarks and examples

Remarks are presented under the following headings:

Introduction
A two-level model
A three-level model

Introduction

Mixed-effects Poisson regression is Poisson regression containing both fixed effects and random effects. In longitudinal data and panel data, random effects are useful for modeling intracluster correlation; that is, observations in the same cluster are correlated because they share common cluster-level random effects.

meqrpoisson allows for not just one, but many levels of nested clusters. For example, in a three-level model you can specify random effects for schools and then random effects for classes nested within schools. The observations (students, presumably) would comprise level one of the model, the classes would comprise level two, and the schools would comprise level three.

However, for simplicity, for now we consider the two-level model, where for a series of \( M \) independent clusters, and conditional on a set of random effects \( u_j \),

\[
Pr(y_{ij} = y_j | u_j) = \exp(-\mu_{ij}) \mu_{ij}^y / y!
\]

for \( \mu_{ij} = \exp(x_{ij} \beta + z_{ij} u_j) \), \( j = 1, \ldots, M \) clusters, and with cluster \( j \) consisting of \( i = 1, \ldots, n_j \) observations. The responses are counts \( y_{ij} \). The \( 1 \times p \) row vector \( x_{ij} \) are the covariates for the fixed effects, analogous to the covariates you would find in a standard Poisson regression model, with regression coefficients (fixed effects) \( \beta \).

The \( 1 \times q \) vector \( z_{ij} \) are the covariates corresponding to the random effects and can be used to represent both random intercepts and random coefficients. For example, in a random-intercept model, \( z_{ij} \) is simply the scalar 1. The random effects \( u_j \) are \( M \) realizations from a multivariate normal distribution with mean 0 and \( q \times q \) variance matrix \( \Sigma \). The random effects are not directly estimated as model parameters but are instead summarized according to the unique elements of \( \Sigma \), known as variance components. One special case of (1) places \( z_{ij} = x_{ij} \) so that all covariate effects are essentially random and distributed as multivariate normal with mean \( \beta \) and variance \( \Sigma \).

Model (1) is an example of a generalized linear mixed model (GLMM), which generalizes the linear mixed-effects (LME) model to non-Gaussian responses. You can fit LMEs in Stata by using \textit{mixed} and fit GLMMs by using \textit{meglm}. Because of the relationship between LMEs and GLMMs, there is insight to be gained through examination of the linear mixed model. This is especially true for Stata users because the terminology, syntax, options, and output for fitting these types of models are nearly identical. See [ME] \textit{mixed} and the references therein, particularly in the \textit{Introduction}, for more information.

Log-likelihood calculations for fitting any generalized mixed-effects model require integrating out the random effects. One widely used modern method is to directly estimate the integral required to calculate the log likelihood by Gauss–Hermite quadrature or some variation thereof. The estimation method used by meqrpoisson is a multicoefficient and multilevel extension of one of these quadrature types, namely, adaptive Gaussian quadrature (AGQ) based on conditional modes, with the multicoefficient extension from Pinheiro and Bates (1995) and the multilevel extension from Pinheiro and Chao (2006); see \textit{Methods and formulas}.

Below we present two short examples of mixed-effects Poisson regression; refer to [ME] \textit{me} and [ME] \textit{meglm} for additional examples.
A two-level model

In this section, we begin with a two-level mixed-effects Poisson regression, because a one-level model, in multilevel-model terminology, is just standard Poisson regression; see [R] poisson.

Example 1

Breslow and Clayton (1993) fit a mixed-effects Poisson model to data from a randomized trial of the drug progabide for the treatment of epilepsy.

```
. use http://www.stata-press.com/data/r14/epilepsy
(Epilepsy data; progabide drug treatment)
. describe
Contains data from http://www.stata-press.com/data/r14/epilepsy.dta
    obs:        236   Epilepsy data; progabide drug treatment
    vars:        8
    size:     4,956 (_dta has notes)

    storage  display value
variable name type    format    label      variable label
    subject  byte     %9.0g    Subject ID: 1-59
    seizures int      %9.0g    No. of seizures
    treat    byte     %9.0g    1: progabide; 0: placebo
    visit    float    %9.0g    Dr. visit; coded as (-.3, -.1, .1, .3)
    lage     float    %9.0g    log(age), mean-centered
    lbas     float    %9.0g    log(0.25*baseline seizures), mean-centered
    lbas_trt float    %9.0g    lbas/treat interaction
    v4       byte     %8.0g    Fourth visit indicator

Sorted by: subject
```

Originally from Thall and Vail (1990), data were collected on 59 subjects (31 on progabide, 28 on placebo). The number of epileptic seizures (seizures) was recorded during the two weeks prior to each of four doctor visits (visit). The treatment group is identified by the indicator variable treat. Data were also collected on the logarithm of age (lage) and the logarithm of one-quarter the number of seizures during the eight weeks prior to the study (lbas). The variable lbas_trt represents the interaction between lbas and treatment. lage, lbas, and lbas_trt are mean centered. Because the study originally noted a substantial decrease in seizures prior to the fourth doctor visit, an indicator, v4, for the fourth visit was also recorded.

Breslow and Clayton (1993) fit a random-effects Poisson model for the number of observed seizures

\[
\log(\mu_{ij}) = \beta_0 + \beta_1 \text{treat}_{ij} + \beta_2 \text{lbas}_{ij} + \beta_3 \text{lbas/trt}_{ij} + \beta_4 \text{lage}_{ij} + \beta_5 \text{v4}_{ij} + u_j
\]

for \( j = 1, \ldots, 59 \) subjects and \( i = 1, \ldots, 4 \) visits. The random effects \( u_j \) are assumed to be normally distributed with mean 0 and variance \( \sigma_u^2 \).
. meqrpoisson seizures treat lbas lbas_trt lage v4 || subject:

Refining starting values:
Iteration 0:  log likelihood = -680.40577  (not concave)
Iteration 1:  log likelihood = -668.60112
Iteration 2:  log likelihood = -666.37635
Performing gradient-based optimization:
Iteration 0:  log likelihood = -666.37635
Iteration 1:  log likelihood = -665.4589
Iteration 2:  log likelihood = -665.29074
Iteration 3:  log likelihood = -665.29068

Mixed-effects Poisson regression Number of obs = 236
Group variable: subject Number of groups = 59
Obs per group: 
  min = 4 
  avg = 4.0 
  max = 4
Integration points = 7 Wald chi2(5) = 121.67
Log likelihood = -665.29068 Prob > chi2 = 0.0000

|          | Coef. | Std. Err. | z    | P>|z| | [95% Conf. Interval] |
|----------|-------|-----------|------|-----|----------------------|
| seizures |       |           |      |     |                      |
| treat    | -.9330388 | .4008346 | -2.33 | 0.020 | -.171866 -.1474175 |
| lbas     | .884433 | .1312313 | 6.74 | 0.000 | .6272244 1.141642 |
| lbas_trt | .3382609 | .2033384 | 1.66 | 0.096 | -.0602752 .7367969 |
| lage     | .4842389 | .3472775 | 1.39 | 0.163 | -.1964126 .16489 |
| v4       | -.1610871 | .0545758 | -2.95 | 0.003 | -.2680537 -.0541206 |
| _cons    | 2.154574 | .2200426 | 9.79 | 0.000 | 1.723299 2.585855 |

Random-effects Parameters                     Estimate  Std. Err. [95% Conf. Interval]
subject: Identity                            var(_cons) .2528265 .058956 .1600785 .3993118

LR test vs. Poisson model:  chibar2(01) = 304.74  Prob >= chibar2 = 0.0000

The number of seizures before the fourth visit does exhibit a significant drop, and the patients
on progabide demonstrate a decrease in frequency of seizures compared with the placebo group.
The subject-specific random effects also appear significant: \( \hat{\sigma}_u^2 = 0.25 \) with standard error 0.06. The
above results are also in good agreement with those of Breslow and Clayton (1993, table 4), who fit
this model by the method of penalized quasi-likelihood (PQL).

Because this is a simple random-intercept model, you can obtain equivalent results by using
xtpoisson with the re and normal options.

> Example 2

In their study of PQL, Breslow and Clayton (1993) also fit a model where they dropped the fixed
effect on v4 and replaced it with a random subject-specific linear trend over the four doctor visits.
The model they fit is

\[
\log(\mu_{ij}) = \beta_0 + \beta_1 \text{treat}_{ij} + \beta_2 \text{lbas}_{ij} + \beta_3 \text{lbas_trt}_{ij} + \\
\quad \beta_4 \text{lage}_{ij} + \beta_5 \text{visit}_{ij} + u_j + v_j \text{visit}_{ij}
\]
where \((u_j, v_j)\) are bivariate normal with 0 mean and variance–covariance matrix

\[
\Sigma = \text{Var} \begin{bmatrix} u_j \\ v_j \end{bmatrix} = \begin{bmatrix} \sigma_u^2 & \sigma_{uv} \\ \sigma_{uv} & \sigma_v^2 \end{bmatrix}
\]

```
. meqrpoisson seizures treat lbas lbas_trt lage visit || subject: visit, > cov(unstructured) intpoints(9)
```

Refining starting values:

Iteration 0:  log likelihood = -672.17188 (not concave)
Iteration 1:  log likelihood = -660.46056
Iteration 2:  log likelihood = -655.8713

Performing gradient-based optimization:

Iteration 0:  log likelihood = -655.8713
Iteration 1:  log likelihood = -655.68217
Iteration 2:  log likelihood = -655.68103
Iteration 3:  log likelihood = -655.68103

Mixed-effects Poisson regression

| Coef. | Std. Err. | z     | P>|z| | [95% Conf. Interval] |
|-------|-----------|-------|------|---------------------|
| treat | -.9286589 | .4021642 | -2.31 | 0.021 | -1.716886 | -.1404315 |
| lbas  | .8849766  | .131252  | 6.74  | 0.000 | .6277275  | 1.142226  |
| lbas_trt | .3379758  | .2044444  | 1.65  | 0.098 | -.0627279 | .7386794  |
| lage  | .4767192  | .3536221  | 1.35  | 0.178 | -.2163673 | 1.169806  |
| visit | -.2664098 | .1647096  | -1.62 | 0.106 | -.5892347 | .0564151  |
| _cons | 2.099555  | .2203711  | 9.53  | 0.000 | 1.667635  | 2.531474  |

Random-effects Parameters

<table>
<thead>
<tr>
<th>Estimate</th>
<th>Std. Err.</th>
<th>[95% Conf. Interval]</th>
</tr>
</thead>
<tbody>
<tr>
<td>var(visit)</td>
<td>.5314808</td>
<td>.2293851</td>
</tr>
<tr>
<td>var(_cons)</td>
<td>.2514928</td>
<td>.0587892</td>
</tr>
<tr>
<td>cov(visit,_cons)</td>
<td>.0028715</td>
<td>.0887018</td>
</tr>
</tbody>
</table>

LR test vs. Poisson model: chi2(3) = 324.54  Prob > chi2 = 0.0000
Note: LR test is conservative and provided only for reference.

In the above, we specified the \texttt{cov(unstructured)} option to allow correlation between \(u_j\) and \(v_j\), although on the basis of the above output it probably was not necessary—the default \texttt{Independent} structure would have sufficed. In the interest of getting more accurate estimates, we also increased the number of quadrature points to nine, although the estimates do not change much when compared with estimates based on the default seven quadrature points.

The essence of the above-fitted model is that after adjusting for other covariates, the log trend in seizures is modeled as a random subject-specific line, with intercept distributed as \(N(\beta_0, \sigma_u^2)\) and slope distributed as \(N(\beta_5, \sigma_v^2)\). From the above output, \(\hat{\beta}_0 = 2.10, \hat{\sigma}_u^2 = 0.25, \hat{\beta}_5 = -0.27, \) and \(\hat{\sigma}_v^2 = 0.53.\)
You can predict the random effects \( u_j \) and \( v_j \) by using `predict` after `meqrpoisson`; see 
`[ME] meqrpoisson postestimation`. Better still, you can obtain a predicted number of seizures that takes these random effects into account.

**A three-level model**

`meqrpoisson` can also fit higher-level models with multiple levels of nested random effects.

### Example 3


```
. use http://www.stata-press.com/data/r14/melanoma
(Skin cancer (melanoma) data)
. describe
Contains data from http://www.stata-press.com/data/r14/melanoma.dta
obs: 354  
vars: 6  
size: 4,956  
  _dta has notes

storage  display value
variable name type format label      variable label
  nation    byte  %11.0g  n    Nation ID
  region    byte  %9.0g     Region ID: EEC level-I areas
  county    int   %9.0g     County ID: EEC level-II/level-III areas
  deaths    int   %9.0g     No. deaths during 1971-1980
  expected  float %9.0g     No. expected deaths
  uv        float %9.0g     UV dose, mean-centered
```

Nine European nations (variable `nation`) are represented, and data were collected over geographical regions defined by EEC statistical services as level I areas (variable `region`), with deaths being recorded for each of 354 counties, which are level II or level III EEC-defined areas (variable `county`, which identifies the observations). Counties are nested within regions, and regions are nested within nations.

The variable `deaths` records the number of deaths for each county, and `expected` records the expected number of deaths (the exposure) on the basis of crude rates for the combined countries. Finally, the variable `uv` is a measure of exposure to ultraviolet (UV) radiation.

In modeling the number of deaths, one possibility is to include dummy variables for the nine nations as fixed effects. Another is to treat these as random effects and fit the three-level random-intercept Poisson model,

\[
\log(\mu_{ijk}) = \log(\text{expected}_{ijk}) + \beta_0 + \beta_1 \text{uv}_{ijk} + u_k + v_{jk}
\]

for nation \( k \), region \( j \), and county \( i \). The model includes an exposure term for expected deaths.
. meqrpoisson deaths uv, exposure(expected) || nation: || region:

Refining starting values:
Iteration 0:  log likelihood = -1169.0851 (not concave)
Iteration 1:  log likelihood = -1156.523 (not concave)
Iteration 2:  log likelihood = -1101.8488

Performing gradient-based optimization:
Iteration 0:  log likelihood = -1101.8488
Iteration 1:  log likelihood = -1099.5579
Iteration 2:  log likelihood = -1097.8162
Iteration 3:  log likelihood = -1097.7151
Iteration 4:  log likelihood = -1097.714
Iteration 5:  log likelihood = -1097.714

Mixed-effects Poisson regression  Number of obs =  354

<table>
<thead>
<tr>
<th>Group Variable</th>
<th>No. of Groups</th>
<th>Observations per Group</th>
<th>Integration Points</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Minimum</td>
<td>Average</td>
</tr>
<tr>
<td>nation</td>
<td>9</td>
<td>3</td>
<td>39.3</td>
</tr>
<tr>
<td>region</td>
<td>78</td>
<td>1</td>
<td>4.5</td>
</tr>
</tbody>
</table>

Wald chi2(1) = 6.12  Prob > chi2 = 0.0134

Log likelihood = -1097.714

| deaths | Coef.   | Std. Err. | z    | P>|z|   | [95% Conf. Interval] |
|--------|---------|-----------|------|------|---------------------|
| uv     | -.0281991 | .0114027  | -2.47 | 0.013 | -.050548 -.0058503 |
| _cons  | -.0639472 | .1335247  | -0.48 | 0.632 | -.3256508 .1977565 |

ln(expected)  1 (exposure)

Random-effects Parameters  

<table>
<thead>
<tr>
<th>nation: Identity</th>
<th>Estimate</th>
<th>Std. Err.</th>
<th>[95% Conf. Interval]</th>
</tr>
</thead>
<tbody>
<tr>
<td>var(_cons)</td>
<td>.1370344</td>
<td>.0722804</td>
<td>.0487364 .3853059</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>region: Identity</th>
<th>Estimate</th>
<th>Std. Err.</th>
<th>[95% Conf. Interval]</th>
</tr>
</thead>
<tbody>
<tr>
<td>var(_cons)</td>
<td>.0483854</td>
<td>.010927</td>
<td>.0310803 .0753258</td>
</tr>
</tbody>
</table>

LR test vs. Poisson model: chi2(2) = 1252.12  Prob > chi2 = 0.0000

Note: LR test is conservative and provided only for reference.

By including an exposure variable that is an expected rate, we are in effect specifying a linear model for the log of the standardized mortality ratio, the ratio of observed deaths to expected deaths that is based on a reference population. Here the reference population is all nine nations.

We now add a random intercept for counties nested within regions, making this a four-level model. Because counties also identify the observations, the corresponding variance component can be interpreted as a measure of overdispersion, variability above and beyond that allowed by a Poisson process; see [R] nbreg and [ME] menbreg.
. meqrpoisson deaths uv, exposure(expected) || nation: || region: || county:, > laplace

Refining starting values:
Iteration 0: log likelihood = -1381.1202 (not concave)
Iteration 1: log likelihood = -1144.7025 (not concave)
Iteration 2: log likelihood = -1138.7095

Performing gradient-based optimization:
Iteration 0: log likelihood = -1138.7095
Iteration 1: log likelihood = -1129.8553
Iteration 2: log likelihood = -1097.7408 (not concave)
Iteration 3: log likelihood = -1088.4381
Iteration 4: log likelihood = -1086.8459
Iteration 5: log likelihood = -1086.7321
Iteration 6: log likelihood = -1086.7309
Iteration 7: log likelihood = -1086.7309

Mixed-effects Poisson regression
Number of obs = 354

Group Variable | No. of Groups | Observations per Group | Integration Points
--- | --- | --- | ---
nation | 9 | Minimum = 3 | Average = 39.3 | Maximum = 95 | Points = 1
region | 78 | Minimum = 1 | Average = 4.5 | Maximum = 13 | Points = 1
county | 354 | Minimum = 1 | Average = 1.0 | Maximum = 1 | Points = 1

Wald chi2(1) = 8.63
Prob > chi2 = 0.0033

Log likelihood = -1086.7309

| deaths | Coef. | Std. Err. | z | P>|z| | [95% Conf. Interval] |
--- | --- | --- | --- | --- | --- |
uv | -.0334681 | .0113919 | -2.94 | 0.003 | -.0557959 -.0111404 |
_cons | -.0864109 | .1298712 | -0.67 | 0.506 | -.3409538 .168132 |
ln(expected) | 1 (exposure) |

| Random-effects Parameters | Estimate | Std. Err. | [95% Conf. Interval] |
nation: Identity | var(_cons) | .1287416 | .0680887 | .0456601 .3629954 |
region: Identity | var(_cons) | .0405965 | .0105002 | .0244527 .0673986 |
county: Identity | var(_cons) | .0146027 | .0050766 | .0073878 .0288636 |

LR test vs. Poisson model: chi2(3) = 1274.08 Prob > chi2 = 0.0000
Note: LR test is conservative and provided only for reference.
Note: Log-likelihood calculations are based on the Laplacian approximation.

In the above, we used a Laplacian approximation, which is not only faster but also produces estimates that closely agree with those obtained with the default seven quadrature points.

See Computation time and the Laplacian approximation in [ME] me for a discussion comparing Laplacian approximation with adaptive quadrature.
 Stored results

`meqrpoisson` stores the following in `e()`:

Scalars

- `e(N)` \(\text{number of observations}\)
- `e(k)` \(\text{number of parameters}\)
- `e(k_f)` \(\text{number of fixed-effects parameters}\)
- `e(k_r)` \(\text{number of random-effects parameters}\)
- `e(k_rs)` \(\text{number of variances}\)
- `e(k_rc)` \(\text{number of covariances}\)
- `e(df_m)` \(\text{model degrees of freedom}\)
- `e(ll)` \(\log \text{likelihood}\)
- `e(chi2)` \(\chi^2\)
- `e(p)` \(\text{significance}\)
- `e(ll_c)` \(\log \text{likelihood, comparison model}\)
- `e(chi2_c)` \(\chi^2, \text{comparison model}\)
- `e(df_c)` \(\text{degrees of freedom, comparison model}\)
- `e(p_c)` \(\text{significance, comparison model}\)
- `e(rank)` \(\text{rank of } e(V)\)
- `e(reparm_rc)` \(\text{return code, final reparameterization}\)
- `e(rc)` \(\text{return code}\)
- `e(converged)` \(1\) if converged, \(0\) otherwise

Macros

- `e(cmd)` `meqrpoisson`
- `e(cmdline)` \(\text{command as typed}\)
- `e(depvar)` \(\text{name of dependent variable}\)
- `e(ivars)` \(\text{grouping variables}\)
- `e(exposurevar)` \(\text{exposure variable}\)
- `e(model)` \(\text{Poisson}\)
- `e(title)` \(\text{title in estimation output}\)
- `e(offset)` \(\text{offset}\)
- `e(redim)` \(\text{random-effects dimensions}\)
- `e(vartypes)` \(\text{variance-structure types}\)
- `e(revars)` \(\text{random-effects covariates}\)
- `e(n_quad)` \(\text{number of integration points}\)
- `e(laplace)` \(\text{laplace, if Laplace approximation}\)
- `e(chi2type)` \(\text{Wald; type of model } \chi^2\)
- `e(vce)` \(\text{bootstrap or jackknife if defined}\)
- `e(vcetype)` \(\text{title used to label Std. Err.}\)
- `e(method)` \(\text{ML}\)
- `e(opt)` \(\text{type of optimization}\)
- `e(ml_method)` \(\text{type of } \text{ml method}\)
- `e(technique)` \(\text{maximization technique}\)
- `e(datasignature)` \(\text{the checksum}\)
- `e(datasignaturevars)` \(\text{variables used in calculation of checksum}\)
- `e(properties)` \(b \ V\)
- `e(estat_cmd)` \(\text{program used to implement } \text{estat}\)
- `e(predict)` \(\text{program used to implement } \text{predict}\)
- `e(marginsnotok)` \(\text{predictions disallowed by } \text{margins}\)
- `e(marginsdefault)` \(\text{default } \text{predict() specification for } \text{margins}\)
- `e(asbalanced)` \(\text{factor variables } \text{fvset as } \text{asbalanced}\)
- `e(asobserved)` \(\text{factor variables } \text{fvset as } \text{asobserved}\)

Matrices

- `e(b)` \(\text{coefficient vector}\)
- `e(Cns)` \(\text{constraints matrix}\)
- `e(N_g)` \(\text{group counts}\)
- `e(g_min)` \(\text{group-size minimums}\)
- `e(g_avg)` \(\text{group-size averages}\)
- `e(g_max)` \(\text{group-size maximums}\)
- `e(V)` \(\text{variance–covariance matrix of the estimators}\)

Functions

- `e(sample)` \(\text{marks estimation sample}\)
Methods and formulas

In a two-level Poisson model, for cluster \( j, j = 1, \ldots, M \), the conditional distribution of \( y_j = (y_{j1}, \ldots, y_{jn_j})' \), given a set of cluster-level random effects \( u_j \), is

\[
f(y_j|u_j) = \prod_{i=1}^{n_j} \left\{ \exp \left( x_{ij}\beta + z_{ij}u_j \right) \right\}^{y_{ij}} \exp \left\{ - \exp \left( x_{ij}\beta + z_{ij}u_j \right) \right\} / \gamma_{ij}!
\]

where \( \gamma_{ij} = \exp \left( x_{ij}\beta + z_{ij}u_j \right) \). Because equality of mean and variance is a characteristic of the Poisson distribution.

\[
y_{ij} \text{ are the conditional variances of } y_{ij}
\]

where \( u_j \) is the diagonal matrix whose diagonal entries are the conditional means of \( y_{ij} \) given \( u_j \). We extend the definition of \( \exp(\cdot) \) to be a vector function where necessary.

Because the prior distribution of \( u_j \) is multivariate normal with mean \( \mathbf{0} \) and \( q \times q \) variance matrix \( \Sigma \), the likelihood contribution for the \( j \)th cluster is obtained by integrating \( u_j \) out of the joint density \( f(y_j, u_j) \),

\[
\mathcal{L}_j(\beta, \Sigma) = (2\pi)^{-q/2} |\Sigma|^{-1/2} \int f(y_j|u_j) \exp \left( -u'_j\Sigma^{-1}u_j/2 \right) du_j
\]

\[
= \exp \left\{ -c(y_j) \right\} (2\pi)^{-q/2} |\Sigma|^{-1/2} \int \exp \left\{ h(\beta, \Sigma, u_j) \right\} du_j
\]

where

\[
h(\beta, \Sigma, u_j) = y'_j (X_j\beta + Z_ju_j) - 1' \exp (X_j\beta + Z_ju_j) - u'_j\Sigma^{-1}u_j/2
\]

and for convenience, in the arguments of \( h(\cdot) \) we suppress the dependence on the observable data \((y_j, X_j, Z_j)\).

The integration in (2) has no closed form and thus must be approximated. The Laplacian approximation (Tierney and Kadane 1986; Pinheiro and Bates 1995) is based on a second-order Taylor expansion of \( h(\beta, \Sigma, u_j) \) about the value of \( u_j \) that maximizes it. Taking first and second derivatives, we obtain

\[
h'(\beta, \Sigma, u_j) = \frac{\partial h(\beta, \Sigma, u_j)}{\partial u_j} = Z'_j \{ y_j - m(\beta, u_j) \} - \Sigma^{-1}u_j
\]

\[
h''(\beta, \Sigma, u_j) = \frac{\partial^2 h(\beta, \Sigma, u_j)}{\partial u_j \partial u'_j} = -\{ Z'_j V(\beta, u_j) Z_j + \Sigma^{-1} \}
\]

where \( m(\beta, u_j) \) is the vector function with the \( i \)th element equal to the conditional mean of \( y_{ij} \) given \( u_j \), that is, \( \exp(x_{ij}\beta + z_{ij}u_j) \). \( V(\beta, u_j) \) is the diagonal matrix whose diagonal entries \( v_{ij} \) are the conditional variances of \( y_{ij} \) given \( u_j \), namely,

\[
v_{ij} = \exp(x_{ij}\beta + z_{ij}u_j)
\]

because equality of mean and variance is a characteristic of the Poisson distribution.
The maximizer of \( h(\beta, \Sigma, \mathbf{u}_j) \) is \( \hat{\mathbf{u}}_j \) such that \( h'(\beta, \Sigma, \hat{\mathbf{u}}_j) = 0 \). The integrand in (2) is proportional to the posterior density \( f(\mathbf{u}_j | y_j) \), so \( \hat{\mathbf{u}}_j \) also represents the posterior mode, a plausible estimator of \( \mathbf{u}_j \) in its own right.

Given the above derivatives, the second-order Taylor approximation then takes the form

\[
  h(\beta, \Sigma, \mathbf{u}_j) \approx h(\beta, \Sigma, \hat{\mathbf{u}}_j) + \frac{1}{2} (\mathbf{u}_j - \hat{\mathbf{u}}_j)' h''(\beta, \Sigma, \hat{\mathbf{u}}_j) (\mathbf{u}_j - \hat{\mathbf{u}}_j)
\]

The first-derivative term vanishes because \( h'(\beta, \Sigma, \hat{\mathbf{u}}_j) = 0 \). Therefore,

\[
  \int \exp \{ h(\beta, \Sigma, \mathbf{u}_j) \} d\mathbf{u}_j \approx \exp \{ h(\beta, \Sigma, \hat{\mathbf{u}}_j) \} \times \int \exp \left[ -\frac{1}{2} (\mathbf{u}_j - \hat{\mathbf{u}}_j)' \left\{ -h''(\beta, \Sigma, \hat{\mathbf{u}}_j) \right\} (\mathbf{u}_j - \hat{\mathbf{u}}_j) \right] d\mathbf{u}_j
\]

\[
  = \exp \{ h(\beta, \Sigma, \hat{\mathbf{u}}_j) \} (2\pi)^{q/2} | -h''(\beta, \Sigma, \hat{\mathbf{u}}_j) |^{-1/2}
\]

because the latter integrand can be recognized as the “kernel” of a multivariate normal density.

Combining the above with (2) (and taking logs) gives the Laplacian log-likelihood contribution of the \( j \)th cluster,

\[
  \mathcal{L}_j^{\text{Lap}}(\beta, \Sigma) = -\frac{1}{2} \log |\Sigma| - \log |\mathbf{R}_j| + h(\beta, \Sigma, \hat{\mathbf{u}}_j) - c(y_j)
\]

where \( \mathbf{R}_j \) is an upper-triangular matrix such that \(-h''(\beta, \Sigma, \hat{\mathbf{u}}_j) = \mathbf{R}_j \mathbf{R}_j'\). Pinheiro and Chao (2006) show that \( \hat{\mathbf{u}}_j \) and \( \mathbf{R}_j \) can be efficiently computed as the iterative solution to a least-squares problem by using matrix decomposition methods similar to those used in fitting LME models (Bates and Pinheiro 1998; Pinheiro and Bates 2000; [ME] mixed).

The fidelity of the Laplacian approximation is determined wholly by the accuracy of the approximation in (3). An alternative that does not depend so heavily on this approximation is integration via AGQ (Naylor and Smith 1982; Liu and Pierce 1994).

The application of AGQ to this particular problem is from Pinheiro and Bates (1995). When we reexamine the integral in question, a transformation of integration variables yields

\[
  \int \exp \{ h(\beta, \Sigma, \mathbf{u}_j) \} d\mathbf{u}_j = |\mathbf{R}_j|^{-1} \int \exp \left\{ h(\beta, \Sigma, \hat{\mathbf{u}}_j + \mathbf{R}_j^{-1} t) \right\} dt
\]

\[
  = (2\pi)^{q/2} |\mathbf{R}_j|^{-1} \int \exp \left\{ h(\beta, \Sigma, \hat{\mathbf{u}}_j + \mathbf{R}_j^{-1} t) + t' t / 2 \right\} \phi(t) dt
\]

where \( \phi(\cdot) \) is the standard multivariate normal density. Because the integrand is now expressed as some function multiplied by a normal density, it can be estimated by applying the rules of standard Gauss–Hermite quadrature. For a predetermined number of quadrature points \( N_Q \), define \( a_k = \sqrt{2} a^*_k \) and \( w_k = w^*_k / \sqrt{\pi} \), for \( k = 1, \ldots, N_Q \), where \( (a^*_k, w^*_k) \) are a set of abscissas and weights for Gauss–Hermite quadrature approximations of \( \int \exp(-x^2) f(x) dx \), as obtained from Abramowitz and Stegun (1972, 924).
Define $a_k = (a_{k_1}, a_{k_2}, \ldots, a_{k_q})'$; that is, $a_k$ is a vector that spans the $N_Q$ abscissas over the dimension $q$ of the random effects. Applying quadrature rules to (5) yields the AGQ approximation,

$$
\int \exp \left\{ h(\beta, \Sigma, u_j) \right\} du_j
\approx (2\pi)^{q/2} |R_j|^{-1} \sum_{k_1=1}^{N_Q} \cdots \sum_{k_q=1}^{N_Q} \left[ \exp \left\{ h(\beta, \Sigma, \hat{u}_j + R_j^{-1}a_k) + a_k'a_k/2 \right\} \prod_{p=1}^{q} w_{kp} \right]
\equiv (2\pi)^{q/2}\hat{G}_j(\beta, \Sigma)
$$

resulting in the AGQ log-likelihood contribution of the $j$th cluster,

$$
\mathcal{L}^{AGQ}_j(\beta, \Sigma) = -\frac{1}{2} \log |\Sigma| + \log \left\{ \hat{G}_j(\beta, \Sigma) \right\} - c(y_j)
$$

The “adaptive” part of adaptive Gaussian quadrature lies in the translation and rescaling of the integration variables in (5) by using $\hat{u}_j$ and $R_j^{-1}$, respectively. This transformation of quadrature abscissas (centered at 0 in standard form) is chosen to better capture the features of the integrand, through which (4) can be seen to resemble a multivariate normal distribution with mean $\hat{u}_j$ and variance $R_j^{-1}R_j^{-T}$. AGQ is therefore not as dependent as the Laplace method upon the approximation in (3). In AGQ, (3) serves merely to redirect the quadrature abscissas, with the AGQ approximation improving as the number of quadrature points, $N_Q$, increases. In fact, Pinheiro and Bates (1995) point out that AGQ with only one quadrature point ($a = 0$ and $w = 1$) reduces to the Laplacian approximation.

The log likelihood for the entire dataset is then simply the sum of the contributions of the $M$ individual clusters, namely, $\mathcal{L}(\beta, \Sigma) = \sum_{j=1}^{M} \mathcal{L}^{\text{Lap}}_j(\beta, \Sigma)$ for Laplace and $\mathcal{L}(\beta, \Sigma) = \sum_{j=1}^{M} \mathcal{L}^{AGQ}_j(\beta, \Sigma)$ for AGQ.

Maximization of $\mathcal{L}(\beta, \Sigma)$ is performed with respect to $(\beta, \theta)$, where $\theta$ is a vector comprising the unique elements of the matrix square root of $\Sigma$. This is done to ensure that $\Sigma$ is always positive semidefinite. If the matlog option is specified, then $\theta$ instead consists of the unique elements of the matrix logarithm of $\Sigma$. For well-conditioned problems, both methods produce equivalent results, yet our experience deems the former as more numerically stable near the boundary of the parameter space.

Once maximization is achieved, parameter estimates are mapped from $(\hat{\beta}, \hat{\theta})$ to $(\hat{\beta}, \hat{\gamma})$, where $\hat{\gamma}$ is a vector containing the unique (estimated) elements of $\Sigma$, expressed as logarithms of standard deviations for the diagonal elements and hyperbolic arctangents of the correlations for off-diagonal elements. This last step is necessary to (a) obtain a parameterization under which parameter estimates can be displayed and interpreted individually, rather than as elements of a matrix square root (or logarithm), and (b) parameterize these elements such that their ranges each encompass the entire real line.

Parameter estimates are stored in $e(b)$ as $(\hat{\beta}, \hat{\gamma})$, with the corresponding variance–covariance matrix stored in $e(V)$. Parameter estimates can be displayed in this metric by specifying the estmetric option. However, in meqrpoisson output, variance components are most often displayed either as variances and covariances (the default) or as standard deviations and correlations (option stddeviations).

The approach outlined above can be extended from two-level models to models with three or more levels; see Pinheiro and Chao (2006) for details.
References


Also see

- [ME] meqrpoisson postestimation — Postestimation tools for meqrpoisson
- [ME] membreg — Multilevel mixed-effects negative binomial regression
- [ME] mepoisson — Multilevel mixed-effects Poisson regression
- [ME] me — Introduction to multilevel mixed-effects models
- [MI] estimation — Estimation commands for use with mi estimate
- [SEM] intro 5 — Tour of models (Multilevel mixed-effects models)
- [XT] xtpoisson — Fixed-effects, random-effects, and population-averaged Poisson models
- [U] 20 Estimation and postestimation commands
### Postestimation commands

The following postestimation commands are of special interest after `meqrpoisson`:

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>estat group</code></td>
<td>summarize the composition of the nested groups</td>
</tr>
<tr>
<td><code>estat recovariance</code></td>
<td>display the estimated random-effects covariance matrix (or matrices)</td>
</tr>
</tbody>
</table>

The following standard postestimation commands are also available:

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>contrast</code></td>
<td>contrasts and ANOVA-style joint tests of estimates</td>
</tr>
<tr>
<td><code>estat ic</code></td>
<td>Akaike’s and Schwarz’s Bayesian information criteria (AIC and BIC)</td>
</tr>
<tr>
<td><code>estat summarize</code></td>
<td>summary statistics for the estimation sample</td>
</tr>
<tr>
<td><code>estat vce</code></td>
<td>variance–covariance matrix of the estimators (VCE)</td>
</tr>
<tr>
<td><code>estimates</code></td>
<td>cataloging estimation results</td>
</tr>
<tr>
<td><code>hausman</code></td>
<td>Hausman’s specification test</td>
</tr>
<tr>
<td><code>lincom</code></td>
<td>point estimates, standard errors, testing, and inference for linear combinations of coefficients</td>
</tr>
<tr>
<td><code>lrtest</code></td>
<td>likelihood-ratio test</td>
</tr>
<tr>
<td><code>margins</code></td>
<td>marginal means, predictive margins, marginal effects, and average marginal effects</td>
</tr>
<tr>
<td><code>marginsplot</code></td>
<td>graph the results from margins (profile plots, interaction plots, etc.)</td>
</tr>
<tr>
<td><code>nlcom</code></td>
<td>point estimates, standard errors, testing, and inference for nonlinear combinations of coefficients</td>
</tr>
<tr>
<td><code>predict</code></td>
<td>predictions, residuals, influence statistics, and other diagnostic measures</td>
</tr>
<tr>
<td><code>predictnl</code></td>
<td>point estimates, standard errors, testing, and inference for generalized predictions</td>
</tr>
<tr>
<td><code>pwcompare</code></td>
<td>pairwise comparisons of estimates</td>
</tr>
<tr>
<td><code>test</code></td>
<td>Wald tests of simple and composite linear hypotheses</td>
</tr>
<tr>
<td><code>testnl</code></td>
<td>Wald tests of nonlinear hypotheses</td>
</tr>
</tbody>
</table>
**predict**

**Description for predict**

`predict` creates a new variable containing predictions such as mean responses; linear predictions; standard errors; and Pearson, deviance, and Anscombe residuals.

**Menu for predict**

Statistics > Postestimation

**Syntax for predict**

**Syntax for obtaining estimated random effects and their standard errors**

```
predict [type] newvarsspec [if] [in], reffects [reses(newvarsspec) relevel(levelvar)]
```

**Syntax for obtaining other predictions**

```
predict [type] newvar [if] [in], statistic nooffset fixedonly
```

`newvarsspec` is `stub*` or `newvarlist`.

<table>
<thead>
<tr>
<th>statistic</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>mu</code></td>
<td>predicted mean; the default</td>
</tr>
<tr>
<td><code>xb</code></td>
<td>linear predictor for the fixed portion of the model only</td>
</tr>
<tr>
<td><code>stdp</code></td>
<td>standard error of the fixed-portion linear prediction</td>
</tr>
<tr>
<td><code>pearson</code></td>
<td>Pearson residuals</td>
</tr>
<tr>
<td><code>deviance</code></td>
<td>deviance residuals</td>
</tr>
<tr>
<td><code>anscombe</code></td>
<td>Anscombe residuals</td>
</tr>
</tbody>
</table>

These statistics are available both in and out of sample; type `predict ... if e(sample) ... if wanted` only for the estimation sample.

**Options for predict**

`reffects` calculates posterior modal estimates of the random effects. By default, estimates for all random effects in the model are calculated. However, if the `relevel(levelvar)` option is specified, then estimates for only level `levelvar` in the model are calculated. For example, if classes are nested within schools, then typing

```
.predict b*, reffects relevel(school)
```

would yield random-effects estimates at the school level. You must specify `q` new variables, where `q` is the number of random-effects terms in the model (or level). However, it is much easier to just specify `stub*` and let Stata name the variables `stub1, stub2, ..., stubq` for you.
reses(newvarsspec) calculates standard errors for the random-effects estimates. By default, standard errors for all random effects in the model are calculated. However, if the relevel(levelvar) option is specified, then standard errors for only level levelvar in the model are calculated; see the reffects option.

You must specify q new variables, where q is the number of random-effects terms in the model (or level). However, it is much easier to just specify stub* and let Stata name the variables stub1, stub2, ..., stubq for you. The new variables will have the same storage type as the corresponding random-effects variables.

The reffects and reses() options often generate multiple new variables at once. When this occurs, the random effects (or standard errors) contained in the generated variables correspond to the order in which the variance components are listed in the output of meqrpoisson. Still, examining the variable labels of the generated variables (with the describe command, for instance) can be useful in deciphering which variables correspond to which terms in the model.

relevel(levelvar) specifies the level in the model at which predictions for random effects and their standard errors are to be obtained. levelvar is the name of the model level and is either the name of the variable describing the grouping at that level or is _all, a special designation for a group comprising all the estimation data.

mu, the default, calculates the predicted mean, that is, the predicted count. By default, this is based on a linear predictor that includes both the fixed effects and the random effects, and the predicted mean is conditional on the values of the random effects. Use the fixedonly option (see below) if you want predictions that include only the fixed portion of the model, that is, if you want random effects set to 0.

xb calculates the linear prediction $x_\beta$ based on the estimated fixed effects (coefficients) in the model. This is equivalent to fixing all random effects in the model to their theoretical (prior) mean value of 0.

stdp calculates the standard error of the fixed-effects linear predictor $x_\beta$.

pearson calculates Pearson residuals. Pearson residuals large in absolute value may indicate a lack of fit. By default, residuals include both the fixed portion and the random portion of the model. The fixedonly option modifies the calculation to include the fixed portion only.

deviance calculates deviance residuals. Deviance residuals are recommended by McCullagh and Nelder (1989) as having the best properties for examining the goodness of fit of a GLM. They are approximately normally distributed if the model is correctly specified. They may be plotted against the fitted values or against a covariate to inspect the model’s fit. By default, residuals include both the fixed portion and the random portion of the model. The fixedonly option modifies the calculation to include the fixed portion only.

anscombe calculates Anscombe residuals, which are designed to closely follow a normal distribution. By default, residuals include both the fixed portion and the random portion of the model. The fixedonly option modifies the calculation to include the fixed portion only.

nooffset is relevant only if you specified offset(varname_o) or exposure(varname_e) for meqrpoisson. It modifies the calculations made by predict so that they ignore the offset/exposure variable; the linear prediction is treated as $X_\beta + Zu$ rather than $X_\beta + Zu + \text{offset}$, or $X_\beta + Zu + \ln(\text{exposure})$, whichever is relevant.

fixedonly modifies predictions to include only the fixed portion of the model, equivalent to setting all random effects equal to 0; see the mu option.
margins

Description for margins

margins estimates margins of response for linear predictions.

Menu for margins

Statistics > Postestimation

Syntax for margins

margins [marginlist] [ , options ]
margins [marginlist] , predict(statistic ...) [ options ]

<table>
<thead>
<tr>
<th>statistic</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>xb</td>
<td>linear predictor for the fixed portion of the model only; the default</td>
</tr>
<tr>
<td>reffects</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>mu</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>stdp</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>pearson</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>deviance</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>anscombe</td>
<td>not allowed with margins</td>
</tr>
</tbody>
</table>

Statistics not allowed with margins are functions of stochastic quantities other than \( \mathbf{e}(b) \).

For the full syntax, see \[R\] margins.

estat

Description for estat

estat group reports the number of groups and minimum, average, and maximum group sizes for each level of the model. Model levels are identified by the corresponding group variable in the data. Because groups are treated as nested, the information in this summary may differ from what you would get if you used the \texttt{tabulate} command on each group variable individually.

estat recovariance displays the estimated variance–covariance matrix of the random effects for each level in the model. Random effects can be either random intercepts, in which case the corresponding rows and columns of the matrix are labeled as \_cons, or random coefficients, in which case the label is the name of the associated variable in the data.
Menu for estat

Statistics > Postestimation

Syntax for estat

*Summarize the composition of the nested groups*

```
estat group
```

*Display the estimated random-effects covariance matrix (or matrices)*

```
estat recovariance [ , relevel(levelvar) correlation matlist_options ]
```

Options for estat recovariance

relevel(levelvar) specifies the level in the model for which the random-effects covariance matrix
is to be displayed and returned in r(cov). By default, the covariance matrices for all levels in the
model are displayed. levelvar is the name of the model level and is either the name of the variable
describing the grouping at that level or is _all, a special designation for a group comprising all
the estimation data.

correlation displays the covariance matrix as a correlation matrix and returns the correlation matrix
in r(corr).

*matlist_options* are style and formatting options that control how the matrix (or matrices) is displayed;
see [P] matlist for a list of options that are available.

Remarks and examples

Various predictions, statistics, and diagnostic measures are available after fitting a Poisson mixed-
effects model with *meqrpoisson*. For the most part, calculation centers around obtaining estimates
of the subject/group-specific random effects. Random effects are not estimated when the model is fit
but instead need to be predicted after estimation.

Example 1

In example 2 of [ME] *meqrpoisson*, we modeled the number of observed epileptic seizures as a
function of treatment with the drug progabide and other covariates,

\[
\log(\mu_{ij}) = \beta_0 + \beta_1 \text{treat}_{ij} + \beta_2 lbas_{ij} + \beta_3 lbas_{trt}_{ij} + \beta_4 \text{age}_{ij} + \beta_5 \text{visit}_{ij} + u_j + v_j \text{visit}_{ij}
\]

where \((u_j, v_j)\) are bivariate normal with 0 mean and variance–covariance matrix

\[
\Sigma = \text{Var} \begin{bmatrix} u_j \\ v_j \end{bmatrix} = \begin{bmatrix} \sigma_u^2 & \sigma_{uv} \\ \sigma_{uv} & \sigma_v^2 \end{bmatrix}
\]
. use http://www.stata-press.com/data/r14/epilepsy  
(Epilepsy data; progabide drug treatment)
. meqrpoisson seizures treat lbas lbas_trt lage visit || subject: visit,  
> cov(unstructured) intpoints(9)

Refining starting values:
Iteration 0:  log likelihood = -672.17188 (not concave)
Iteration 1:  log likelihood = -660.46056
Iteration 2:  log likelihood = -655.8713

Performing gradient-based optimization:
Iteration 0:  log likelihood = -655.8713
Iteration 1:  log likelihood = -655.68217
Iteration 2:  log likelihood = -655.68103
Iteration 3:  log likelihood = -655.68103

Mixed-effects Poisson regression  
Number of obs = 236  
Group variable: subject  
Number of groups = 59

Obs per group:  
min = 4
avg = 4.0
max = 4

Integration points = 9  
Wald chi2(5) = 115.56
Log likelihood = -655.68103  
Prob > chi2 = 0.0000

| Coef.  | Std. Err. | z     | P>|z|  | [95% Conf. Interval] |
|--------|-----------|-------|------|-----------------------|
| seizures |           |       |      |                       |
| treat   | -.9286589 | .4021642 | -2.31 | 0.021      | -1.716886     | -.1404315  |
| lbas    | .8849766  | .131252 | 6.74 | 0.000      | .6277275     | 1.142226   |
| lbas_trt| .3379758  | .2044444 | 1.65 | 0.098      | .0627279     | .7386794   |
| lage    | .4767192  | .3536221 | 1.35 | 0.178      | -.2163673    | 1.169806   |
| visit   | -.2664098 | .1647096 | -1.62 | 0.106      | -.5892347    | .0564151   |
| _cons   | 2.099555  | .2203711 | 9.53 | 0.000      | 1.667635     | 2.531474   |

Random-effects Parameters  
Number of obs = 236  
Number of groups = 59

<table>
<thead>
<tr>
<th>Estimate</th>
<th>Std. Err.</th>
<th>[95% Conf. Interval]</th>
</tr>
</thead>
<tbody>
<tr>
<td>subject: Unstructured</td>
<td></td>
<td></td>
</tr>
<tr>
<td>var(visit)</td>
<td>.5314808</td>
<td>.2293851</td>
</tr>
<tr>
<td>var(_cons)</td>
<td>.2514928</td>
<td>.0587892</td>
</tr>
<tr>
<td>cov(visit,_cons)</td>
<td>.0028715</td>
<td>.0887018</td>
</tr>
</tbody>
</table>

LR test vs. Poisson model:  
chi2(3) = 324.54  
Prob > chi2 = 0.0000

Note: LR test is conservative and provided only for reference.

The purpose of this model was to allow subject-specific linear log trends over each subject’s four doctor visits, after adjusting for the other covariates. The intercepts of these lines are distributed $N(\beta_0, \sigma_u^2)$, and the slopes are distributed $N(\beta_5, \sigma_v^2)$, based on the fixed effects and assumed distribution of the random effects.

We can use predict to obtain estimates of the random effects $u_j$ and $v_j$ and combine these with our estimates of $\beta_0$ and $\beta_5$ to obtain the intercepts and slopes of the linear log trends.

. predict re_visit re_cons, reffects
. generate b1 = _b[visit] + re_visit
. generate b0 = _b[_cons] + re_cons
. by subject, sort: generate tolist = _n==1
We list these slopes (b1) and intercepts (b0) for five control subjects and five subjects on the treatment.

```
. count if tolist & treat
   31
. count if tolist & treat & b1 < 0
   25
. count if tolist & !treat
   28
. count if tolist & !treat & b1 < 0
   20
```

We also find that 25 of the 31 subjects taking progabide were estimated to have a downward trend in seizures over their four doctor visits, compared with 20 of the 28 control subjects.

We also obtain predictions for number of seizures, and unless we specify the fixedonly option, these predictions will incorporate the estimated subject-specific random effects.

```
. predict n
   (option mu assumed; predicted means)
. list subject treat visit seizures n if subject <= 2 | subject >= 58, sep(0)
```

We list these slopes (b1) and intercepts (b0) for five control subjects and five subjects on the treatment.

```
subject        treat b1     b0
1.                1  0  -.4284563  2.164691
5.                2  0  -.2727145  2.179111
9.                3  0  .0026486   2.450811
13.               4  0  -.3194157  2.268827
17.               5  0  .6063656   2.123723
217.              55 1  -.2304782  2.311494
221.              56 1  .2904741   3.211369
225.              57 1  -.4831492  1.457486
229.              58 1  -.252236   1.168154
233.              59 1  -.1266651  2.204869
```
Technical note

Out-of-sample predictions are permitted after `meqrpoisson`, but if these predictions involve estimated random effects, the integrity of the estimation data must be preserved. If the estimation data have changed since the model was fit, `predict` will be unable to obtain predicted random effects that are appropriate for the fitted model and will give an error. Thus to obtain out-of-sample predictions that contain random-effects terms, be sure that the data for these predictions are in observations that augment the estimation data.

Stored results

`estat recovariance` stores the following in `r()`:

Scalars
- `r(relevels)` number of levels

Matrices
- `r(Cov#)` level-# random-effects covariance matrix
- `r(Corr#)` level-# random-effects correlation matrix (if option `correlation` was specified)

For a \( G \)-level nested model, \# can be any integer between 2 and \( G \).

Methods and formulas

Continuing the discussion in `Methods and formulas` of [ME] `meqrpoisson` and using the definitions and formulas defined there, we begin by considering the prediction of the random effects \( \mathbf{u}_j \) for the \( j \)th cluster in a two-level model.

Given a set of estimated `meqrpoisson` parameters, \( (\hat{\beta}, \hat{\Sigma}) \), a profile likelihood in \( \mathbf{u}_j \) is derived from the joint distribution \( f(y_j, \mathbf{u}_j) \) as

\[
\mathcal{L}_j(\mathbf{u}_j) = \exp\left\{ -c(y_j) \right\} (2\pi)^{-q/2} |\hat{\Sigma}|^{-1/2} \exp\left\{ g(\hat{\beta}, \hat{\Sigma}, \mathbf{u}_j) \right\}
\]

(1)

The conditional maximum likelihood estimator of \( \mathbf{u}_j \)—conditional on fixed \( (\hat{\beta}, \hat{\Sigma}) \)—is the maximizer of \( \mathcal{L}_j(\mathbf{u}_j) \) or, equivalently, the value of \( \hat{\mathbf{u}}_j \) that solves

\[
0 = g'(\hat{\beta}, \hat{\Sigma}, \hat{\mathbf{u}}_j) = Z_j' \left\{ y_j - m(\hat{\beta}, \hat{\mathbf{u}}_j) \right\} - \hat{\Sigma}^{-1}\hat{\mathbf{u}}_j
\]

Because (1) is proportional to the conditional density \( f(\mathbf{u}_j|y_j) \), you can also refer to \( \hat{\mathbf{u}}_j \) as the conditional mode (or posterior mode if you lean toward Bayesian terminology). Regardless, you are referring to the same estimator.

Conditional standard errors for the estimated random effects are derived from standard theory of maximum likelihood, which dictates that the asymptotic variance matrix of \( \hat{\mathbf{u}}_j \) is the negative inverse of the Hessian, which is estimated as

\[
g''(\hat{\beta}, \hat{\Sigma}, \hat{\mathbf{u}}_j) = -\left\{ Z_j' V(\hat{\beta}, \hat{\mathbf{u}}_j) Z_j + \hat{\Sigma}^{-1}\right\}
\]

Similar calculations extend to models with more than one level of random effects; see Pinheiro and Chao (2006).
For any observation \( i \) in the \( j \)th cluster in a two-level model, define the linear predictor as
\[
\hat{\eta}_{ij} = x_{ij} \hat{\beta} + z_{ij} \hat{u}_j
\]
In a three-level model, for the \( i \)th observation within the \( j \)th level-two cluster within the \( k \)th level-three cluster,
\[
\hat{\eta}_{ijk} = x_{ijk} \hat{\beta} + z_{ijk}^{(3)} \hat{u}_{(3)}^{(k)} + z_{ijk}^{(2)} \hat{u}_{(2)}^{jk}
\]
where \( z^{(p)} \) and \( u^{(p)} \) refer to the level \( p \) design variables and random effects, respectively. For models with more than three levels, the definition of \( \hat{\eta} \) extends in the natural way, with only the notation becoming more complicated.

If the fixedonly option is specified, \( \hat{\eta} \) contains the linear predictor for only the fixed portion of the model, for example, in a two-level model \( \hat{\eta}_{ij} = x_{ij} \hat{\beta} \). In what follows, we assume a two-level model, with the only necessary modification for multilevel models being the indexing.

The predicted mean conditional on the random effects \( \hat{u}_j \) is
\[
\hat{\mu}_{ij} = \exp(\hat{\eta}_{ij})
\]
Pearson residuals are calculated as
\[
\nu_{ij}^P = \frac{y_{ij} - \hat{\mu}_{ij}}{\sqrt{\nu(\hat{\mu}_{ij})}}
\]
for \( V(\hat{\mu}_{ij}) = \hat{\mu}_{ij} \).

Deviance residuals are calculated as
\[
\nu_{ij}^D = \text{sign}(y_{ij} - \hat{\mu}_{ij}) \sqrt{\hat{d}_{ij}^2}
\]
where
\[
\hat{d}_{ij}^2 = \begin{cases} 
2\hat{\mu}_{ij} & \text{if } y_{ij} = 0 \\
2 \left\{ y_{ij} \log \left( \frac{y_{ij}}{\hat{\mu}_{ij}} \right) - (y_{ij} - \hat{\mu}_{ij}) \right\} & \text{otherwise}
\end{cases}
\]

Anscombe residuals are calculated as
\[
\nu_{ij}^A = \frac{3 \left( \frac{y_{ij}^{2/3} - \hat{\mu}_{ij}^{2/3}}{2\hat{\mu}_{ij}^{1/6}} \right)}{2\hat{\mu}_{ij}^{1/6}}
\]

For a discussion of the general properties of the above residuals, see Hardin and Hilbe (2012, chap. 4).

References


Also see

[ME] meqrpoisson — Multilevel mixed-effects Poisson regression (QR decomposition)

[U] 20 Estimation and postestimation commands
**Description**

`mestreg` fits a mixed-effects parametric survival-time model. The conditional distribution of the response given the random effects is assumed to be an exponential, Weibull, lognormal, loglogistic, or gamma distribution. `mestreg` can be used with single- or multiple-record `st` data.

**Quick start**

**Without weights**

Two-level Weibull survival model with covariates `x1` and `x2` and random intercepts by `lev2` using `stset` data

```
mestreg x1 x2 || lev2:, distribution(weibull)
```

Mixed-effects model adding random coefficients for `x1`

```
mestreg x1 x2 || lev2:x1, distribution(weibull)
```

Three-level random-intercept model with `lev2` nested within `lev3`

```
mestreg x1 x2 || lev3: || lev2:, distribution(weibull)
```

**With weights**

Two-level Weibull survival model with covariates `x1` and `x2`, random intercepts by `lev2`, and observation-level frequency weights `wvar1` using `stset` data

```
mestreg x1 x2 [fweight=wvar1] || lev2:, distribution(weibull)
```

Two-level random-intercept model from a two-stage sampling design with PSUs identified by `psu` using `PSU-level` and observation-level sampling weights `wvar2` and `wvar1`

```
mestreg x1 x2 [pweight=wvar1] || psu:, pweight(wvar2)
```

Same as above, but `svyset` the data first

```
svyset psu [pweight=wvar2] || _n, weight(wvar1)
svy: mestreg x1 x2 || psu:, distribution(weibull)
```

Note: Any supported parametric survival `distribution` may be specified in place of `weibull` above.

**Menu**

```
Statistics > Multilevel mixed-effects models > Parametric survival regression
```
Syntax

```plaintext
mestreg fe_equation || re_equation [|| re_equation ...],
distribution(distname) [options]
```

where the syntax of `fe_equation` is

```plaintext
[ indepvars ] [ if ] [ in ] [ weight ] [ , fe_options ]
```

and the syntax of `re_equation` is one of the following:

- for random coefficients and intercepts
  ```plaintext
  levelvar: [ varlist ] [ , re_options ]
  ```
  for random effects among the values of a factor variable
  ```plaintext
  levelvar: R.varname
  ```

`levelvar` is a variable identifying the group structure for the random effects at that level or is `_all` representing one group comprising all observations.

### fe_options

<table>
<thead>
<tr>
<th>Model</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>noconstant</td>
<td>suppress constant term from the fixed-effects equation</td>
</tr>
<tr>
<td>offset(varname)</td>
<td>include varname in model with coefficient constrained to 1</td>
</tr>
</tbody>
</table>

### re_options

<table>
<thead>
<tr>
<th>Model</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>covariance(vartype)</td>
<td>variance–covariance structure of the random effects</td>
</tr>
<tr>
<td>noconstant</td>
<td>suppress constant term from the random-effects equation</td>
</tr>
<tr>
<td>fweight(varname)</td>
<td>frequency weights at higher levels</td>
</tr>
<tr>
<td>iweight(varname)</td>
<td>importance weights at higher levels</td>
</tr>
<tr>
<td>pweight(varname)</td>
<td>sampling weights at higher levels</td>
</tr>
</tbody>
</table>
### options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>*distribution(distname)</td>
<td>specify survival distribution</td>
</tr>
<tr>
<td>time</td>
<td>use accelerated failure-time metric</td>
</tr>
<tr>
<td>constraints(constraints)</td>
<td>apply specified linear constraints</td>
</tr>
<tr>
<td>collinear</td>
<td>keep collinear variables</td>
</tr>
</tbody>
</table>

#### SE/Robust

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>vce(vcetype)</td>
<td>vcetype may be oim, robust, or cluster clustvar</td>
</tr>
</tbody>
</table>

#### Reporting

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>level(#)</td>
<td>set confidence level; default is level(95)</td>
</tr>
<tr>
<td>nohr</td>
<td>do not report hazard ratios</td>
</tr>
<tr>
<td>tratio</td>
<td>report time ratios</td>
</tr>
<tr>
<td>noshow</td>
<td>do not show st setting information</td>
</tr>
<tr>
<td>nocnsreport</td>
<td>do not display constraints</td>
</tr>
<tr>
<td>notable</td>
<td>suppress coefficient table</td>
</tr>
<tr>
<td>noheader</td>
<td>suppress output header</td>
</tr>
<tr>
<td>nogroup</td>
<td>suppress table summarizing groups</td>
</tr>
<tr>
<td>nolrtest</td>
<td>do not perform likelihood-ratio test</td>
</tr>
<tr>
<td>display_options</td>
<td>control columns and column formats, row spacing, line width, display of omitted variables and base and empty cells, and factor-variable labeling</td>
</tr>
</tbody>
</table>

#### Integration

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>intmethod(intmethod)</td>
<td>integration method</td>
</tr>
<tr>
<td>intpoints(#)</td>
<td>set the number of integration (quadrature) points for all levels; default is intpoints(7)</td>
</tr>
</tbody>
</table>

#### Maximization

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>maximize_options</td>
<td>control the maximization process; seldom used</td>
</tr>
<tr>
<td>startvalues(svmethod)</td>
<td>method for obtaining starting values</td>
</tr>
<tr>
<td>startgrid[gridspec]</td>
<td>perform a grid search to improve starting values</td>
</tr>
<tr>
<td>noestimate</td>
<td>do not fit the model; show starting values instead</td>
</tr>
<tr>
<td>dnumerical</td>
<td>use numerical derivative techniques</td>
</tr>
<tr>
<td>coeflegend</td>
<td>display legend instead of statistics</td>
</tr>
</tbody>
</table>

*distribution(distname) is required.
You must \texttt{stset} your data before using \texttt{mestreg}; see \cite{[ST] stset}.

\texttt{indepvars} may contain factor variables; see \cite{[U] 11.4.3 Factor variables}.

\texttt{by} and \texttt{svy} are allowed; see \cite{[U] 11.1.10 Prefix commands}.

\texttt{vce()} and weights are not allowed with the \texttt{svy} prefix; see \cite{[SVY] svy}.

\texttt{fweights}, \texttt{iweights}, and \texttt{pweights} are allowed; see \cite{[U] 11.1.6 weight}. Only one type of weight may be specified.

Weights are not supported under the Laplacian approximation or for crossed models.

\texttt{startvalues()}, \texttt{startgrid}, \texttt{noestimate}, \texttt{dnumerical}, and \texttt{coeflegend} do not appear in the dialog box.

See \cite{[U] 20 Estimation and postestimation commands} for more capabilities of estimation commands.

\section*{Options}

\begin{itemize}
\item \texttt{noconstant} suppresses the constant (intercept) term and may be specified for the fixed-effects equation and for any of or all the random-effects equations.
\end{itemize}
offset(varname) specifies that varname be included in the fixed-effects portion of the model with the coefficient constrained to be 1.

covariance(vartype) specifies the structure of the covariance matrix for the random effects and may be specified for each random-effects equation. vartype is one of the following: independent, exchangeable, identity, unstructured, fixed(matname), or pattern(matname).

covariance(independent) covariance structure allows for a distinct variance for each random effect within a random-effects equation and assumes that all covariances are 0. The default is covariance(independent) unless a crossed random-effects model is fit, in which case the default is covariance(identity).

covariance(exchangeable) structure specifies one common variance for all random effects and one common pairwise covariance.

covariance(identity) is short for “multiple of the identity”; that is, all variances are equal and all covariances are 0.

covariance(unstructured) allows for all variances and covariances to be distinct. If an equation consists of \( p \) random-effects terms, the unstructured covariance matrix will have \( p(p + 1)/2 \) unique parameters.

covariance(fixed(matname)) and covariance(pattern(matname)) covariance structures provide a convenient way to impose constraints on variances and covariances of random effects. Each specification requires a matname that defines the restrictions placed on variances and covariances. Only elements in the lower triangle of matname are used, and row and column names of matname are ignored. A missing value in matname means that a given element is unrestricted. In a fixed(matname) covariance structure, (co)variance \((i,j)\) is constrained to equal the value specified in the \(i,j\)th entry of matname. In a pattern(matname) covariance structure, (co)variances \((i,j)\) and \((k,l)\) are constrained to be equal if \(\text{matname}[i,j] = \text{matname}[k,l]\).

fweight(varname) specifies frequency weights at higher levels in a multilevel model, whereas frequency weights at the first level (the observation level) are specified in the usual manner, for example, [fw=fwtvar1]. varname can be any valid Stata variable name, and you can specify fweight() at levels two and higher of a multilevel model. For example, in the two-level model

```
. mecmd fixed_portion [fw = wt1] || school: ..., fweight(wt2) ...
```

the variable wt1 would hold the first-level (the observation-level) frequency weights, and wt2 would hold the second-level (the school-level) frequency weights.

iweight(varname) specifies importance weights at higher levels in a multilevel model, whereas importance weights at the first level (the observation level) are specified in the usual manner, for example, [iw=iwtvar1]. varname can be any valid Stata variable name, and you can specify iweight() at levels two and higher of a multilevel model. For example, in the two-level model

```
. mecmd fixed_portion [iw = wt1] || school: ..., iweight(wt2) ...
```

the variable wt1 would hold the first-level (the observation-level) importance weights, and wt2 would hold the second-level (the school-level) importance weights.

pweight(varname) specifies sampling weights at higher levels in a multilevel model, whereas sampling weights at the first level (the observation level) are specified in the usual manner, for example, [pw=pwtvar1]. varname can be any valid Stata variable name, and you can specify pweight() at levels two and higher of a multilevel model. For example, in the two-level model

```
. mecmd fixed_portion [pw = wt1] || school: ..., pweight(wt2) ...
```

variable wt1 would hold the first-level (the observation-level) sampling weights, and wt2 would hold the second-level (the school-level) sampling weights.
distribution(*distname*) specifies the survival model to be fit. *distname* is one of the following: exponential, loglogistic, llogistic, weibull, lognormal, lnormal, or gamma. This option is required.

time specifies that the model be fit in the accelerated failure-time metric rather than in the log relative-hazard metric. This option is valid only for the exponential and Weibull models because these are the only models that have both a proportional-hazards and an accelerated failure-time parameterization. Regardless of metric, the likelihood function is the same, and models are equally appropriate in either metric; it is just a matter of changing interpretation.

time must be specified at estimation.

canstraints(*constraints*), collinear; see [R] estimation options.

vce(*vcetype*) specifies the type of standard error reported, which includes types that are derived from asymptotic theory (oim), that are robust to some kinds of misspecification (robust), and that allow for intragroup correlation (cluster clustvar); see [R] vce_option. If vce(robust) is specified, robust variances are clustered at the highest level in the multilevel model.

level(#) ; see [R] estimation options.

nohr, which may be specified at estimation or upon redisplaying results, specifies that coefficients rather than exponentiated coefficients be displayed, that is, that coefficients rather than hazard ratios be displayed. This option affects only how coefficients are displayed, not how they are estimated.

This option is valid only for the exponential and Weibull models because they have a natural proportional-hazards parameterization. These two models, by default, report hazards ratios (exponentiated coefficients).

tratio specifies that exponentiated coefficients, which are interpreted as time ratios, be displayed. tratio is appropriate only for the loglogistic, lognormal, and gamma models or for the exponential and Weibull models when fit in the accelerated failure-time metric.

tratio may be specified at estimation or upon replay.

noshow prevents mestreg from showing the key st variables. This option is rarely used because most users type stset, show or stset, noshow to set once and for all whether they want to see these variables mentioned at the top of the output of every st command; see [ST] stset.

nocnsreport; see [R] estimation options.

notable suppresses the estimation table, either at estimation or upon replay.

noheader suppresses the output header, either at estimation or upon replay.

nogroup suppresses the display of group summary information (number of groups, average group size, minimum, and maximum) from the output header.

nolrtest is valid only with frailty models, in which case it suppresses the likelihood-ratio test for significant frailty. This option may also be specified upon replay to suppress this test from the output.

display_options: noci, nopvalues, noomitted, vsquish, noemptycells, baselevels, allbaselevels,nofvlabel,fvwrap(#), fvwrapon(style), cformat(%fmt), pformat(%fmt), sformat(%fmt), and nolstretch; see [R] estimation options.
Integration

`intmethod(intmethod)` specifies the integration method to be used for the random-effects model. `mvaghermite` performs mean–variance adaptive Gauss–Hermite quadrature; `mcaghermite` performs mode-curvature adaptive Gauss–Hermite quadrature; `ghermite` performs nonadaptive Gauss–Hermite quadrature; and `laplace` performs the Laplacian approximation, equivalent to mode-curvature adaptive Gaussian quadrature with one integration point.

The default integration method is `mvaghermite` unless a crossed random-effects model is fit, in which case the default integration method is `laplace`. The Laplacian approximation has been known to produce biased parameter estimates; however, the bias tends to be more prominent in the estimates of the variance components rather than in the estimates of the fixed effects.

For crossed random-effects models, estimation with more than one quadrature point may be prohibitively intensive even for a small number of levels. For this reason, the integration method defaults to the Laplacian approximation. You may override this behavior by specifying a different integration method.

`intpoints(#)` sets the number of integration points for quadrature. The default is `intpoints(7)`, which means that seven quadrature points are used for each level of random effects. This option is not allowed with `intmethod(laplace)`.

The more integration points, the more accurate the approximation to the log likelihood. However, computation time increases as a function of the number of quadrature points raised to a power equaling the dimension of the random-effects specification. In crossed random-effects models and in models with many levels or many random coefficients, this increase can be substantial.

Maximization

`maximize_options: difficult, technique(algorithm_spec), iterate(#), [no]log, trace, gradient, showstep, hessian, showtolerance, tolerance(#), ltolerance(#), nrtolerance(#), nonrtolerance, and from(init_specs); see [R] maximize. Those that require special mention for `mestreg` are listed below.

`from()` accepts a properly labeled vector of initial values or a list of coefficient names with values. A list of values is not allowed.

The following options are available with `mestreg` but are not shown in the dialog box:
`startvalues(svmethod), startgrid([gridspec]), noestimate, and dnumerical; see [ME] meglm.`

`coeflegend; see [R] estimation options.`

Remarks and examples

For a general introduction to `me` commands, see [ME] me.

Remarks are presented under the following headings:

Introduction
Two-level models
Three-level models
Introduction

Mixed-effects survival models contain both fixed effects and random effects. In longitudinal data and panel data, random effects are useful for modeling intracluster correlation; that is, observations in the same cluster are correlated because they share common cluster-level random effects.

Comprehensive treatments of mixed models are provided by, for example, Searle, Casella, and McCulloch (1992); Verbeke and Molenberghs (2000); Raudenbush and Bryk (2002); Demidenko (2004); Hedeker and Gibbons (2006); and McCulloch, Searle, and Neuhaus (2008). Rabe-Hesketh and Skrondal (2012, chap. 14–15) is a good introductory read on applied multilevel modeling of survival data.

mestreg allows for not just one but many levels of nested clusters of random effects. For example, in a three-level model, you can specify random effects for schools and then random effects for classes nested within schools. In this model, the observations (presumably, the students) comprise the first level, the classes comprise the second level, and the schools comprise the third level.

However, for simplicity, we now consider only two-level models, where we have a series of M independent clusters and a set of random effects \( u_j \) corresponding to those clusters. Two often-used models for adjusting survivor functions for the effects of covariates are the accelerated failure-time (AFT) model and the multiplicative or proportional hazards (PH) model.

In the AFT model, the natural logarithm of the survival time, \( \log t \), is expressed as a linear function of the covariates; when we incorporate random-effects, this yields the model

\[
\log t_{ji} = x_{ji} \beta + z_{ji} u_j + v_{ji}
\]

for \( j = 1, \ldots, M \) clusters, with cluster \( j \) consisting of \( i = 1, \ldots, n_j \) observations. The \( 1 \times p \) row vector \( x_{ji} \) contains the covariates for the fixed effects, with regression coefficients (fixed effects) \( \beta \).

The \( 1 \times q \) vector \( z_{ji} \) contains the covariates corresponding to the random effects and can be used to represent both random intercepts and random coefficients. For example, in a random-intercept model, \( z_{ji} \) is simply the scalar 1. The random effects \( u_j \) are \( M \) realizations from a multivariate normal distribution with mean 0 and \( q \times q \) variance matrix \( \Sigma \). The random effects are not directly estimated as model parameters but are instead summarized according to the unique elements of \( \Sigma \), known as variance components.

Finally, \( v_{ji} \) are the observation-level errors with density \( \varphi(\cdot) \). The distributional form of the error term determines the regression model. Five regression models are implemented in mestreg using the AFT parameterization: exponential, gamma, loglogistic, lognormal, and Weibull. The lognormal regression model is obtained by letting \( \varphi(\cdot) \) be the normal density. Similarly, by letting \( \varphi(\cdot) \) be the logistic density, one obtains the loglogistic regression. Setting \( \varphi(\cdot) \) equal to the extreme-value density yields the exponential and the Weibull regression models.

In the PH models fit by mestreg, the covariates have a multiplicative effect on the hazard function

\[
h(t_{ji}) = h_0(t_{ji}) \exp(x_{ji} \beta + z_{ji} u_j)
\]

for some baseline hazard function \( h_0(t) \). For the mestreg command, \( h_0(t) \) is assumed to be parametric. The exponential and Weibull models are implemented in mestreg for the PH parameterization. These two models are implemented using both the AFT and PH parameterizations.

mestreg is suitable only for data that have been stset. By using stset on your data, you define the variables \(-t0, -t, \) and \(-d\), which serve as the trivariate response variable \((t_0, t, d)\). Each response corresponds to a period under observation, \((t_0, t]\), resulting in either failure \((d = 1)\) or right-censoring \((d = 0)\) at time \( t \).
mestreg does not allow delayed entry or gaps. However, mestreg is appropriate for data exhibiting multiple records per subject and time-varying covariates. mestreg requires subjects to be nested within clusters.

stset weights are not used; instead, weights must be specified at estimation. Weights are not allowed with crossed models or the Laplacian approximation. See Survey estimation in Methods and formulas for details.

Two-level models

Example 1

In example 11 of [ST] streg, we fit a Weibull model with an inverse-Gaussian shared frailty to the recurrence times for catheter-insertion point infection for 38 kidney dialysis patients. In this example, the subjects are the catheter insertions, not the patients themselves. This is a function of how the data were recorded—the onset of risk occurs at the time the catheter is inserted and not, say, at the time of admission of the patient into the study. Thus we have two subjects (insertions) within each group (patient). Each catheter insertion results in either infection (infect==1) or right-censoring (infect==0). The stset results are shown below.

```
. use http://www.stata-press.com/data/r14/catheter
( Kidney data, McGilchrist and Aisbett, Biometrics, 1991)
. stset time, failure(infect)
  failure event: infect != 0 & infect < .
  obs. time interval: (0, time]
  exit on or before: failure
```

76 total observations
0 exclusions

76 observations remaining, representing
58 failures in single-record/single-failure data
7424 total analysis time at risk and under observation
  at risk from t = 0
  earliest observed entry t = 0
  last observed exit t = 562

While it is reasonable to assume independence of patients, we would not want to assume that recurrence times within each patient are independent. The model used in [ST] streg allowed us to model the correlation by assuming that it was the result of a latent patient-level effect, or frailty.

The random-effects approach used by mestreg is more flexible because it allows you to experiment with several levels of random effects, including random coefficients, or both. You can then choose the model that best suits your data. Here we use mestreg to fit a random-effects Weibull model with normally distributed random effects. This model can be viewed as a shared frailty model with lognormal frailty.
. mestreg age female || patient:, distribution(weibull)
    failure _d: infect
    analysis time _t: time

Fitting fixed-effects model:
Iteration 0:  log likelihood = -1700758.8
Iteration 1:  log likelihood = -209.06397
Iteration 2:  log likelihood = -105.48579
Iteration 3:  log likelihood = -103.51354
Iteration 4:  log likelihood = -103.44376
Iteration 5:  log likelihood = -103.44362
Iteration 6:  log likelihood = -103.44362
Refining starting values:
Grid node 0:  log likelihood = -104.90022

Fitting full model:
Iteration 0:  log likelihood = -104.90022 (not concave)
Iteration 1:  log likelihood = -101.93677
Iteration 2:  log likelihood = -101.67709
Iteration 3:  log likelihood = -98.880108
Iteration 4:  log likelihood = -98.744275
Iteration 5:  log likelihood = -98.742493
Iteration 6:  log likelihood = -98.742496

Mixed-effects Weibull regression

| Number of obs | 76                  |
| Group variable: patient | Number of groups = 38 |

<table>
<thead>
<tr>
<th></th>
<th>Obs per group:</th>
</tr>
</thead>
<tbody>
<tr>
<td>min</td>
<td>2</td>
</tr>
<tr>
<td>avg</td>
<td>2.0</td>
</tr>
<tr>
<td>max</td>
<td>2</td>
</tr>
</tbody>
</table>

Integration method: mvaghermite
Integration pts. = 7
Wald chi2(2) = 10.12
Log likelihood = -98.742496
Prob > chi2 = 0.0063

| _t      | Haz. Ratio | Std. Err. | z    | P>|z| | [95% Conf. Interval] |
|---------|------------|-----------|------|------|-----------------------|
| age     | 1.007348   | .013788   | 0.53 | 0.593| .9806827 1.034737    |
| female  | .1904719   | .0999918  | -3.16| 0.002| .0680733 .5329482   |
| _cons   | .0072901   | .0072274  | -4.96| 0.000| .0010444 .0508879   |
| /ln_p   | .2243249   | .1402794  | 1.60 | 0.110| -.0506177 .4992675  |

| patient | var(_cons) | .8308563 | .4978461 | .2567373 | 2.688827 |

LR test vs. Weibull model: chibar2(01) = 9.40
Prob >= chibar2 = 0.0011

The results are similar to those in [ST] streg. The likelihood-ratio test compares the random-effects model with a survival model with fixed-effects only. The results support the random-effects model.

By default, when fitting a model with the PH parameterization, mestreg displays exponentiated coefficients, labeled as hazard ratios. These hazard ratios should be interpreted as “conditional hazard ratios”, that is, conditional on the random effects.

For example, the hazard ratio for age is 1.01. This means that according to the model, for a given patient, the hazard would increase 1% with each year of age. However, at the population level, marginal hazards corresponding to different levels of the covariates are not necessarily proportional. Example 5 in [ME] mestreg postestimation illustrates this point with simulated data.

The exponentiated coefficients of covariates that usually remain constant within a group do not have a natural interpretation as conditional hazard ratios. However, the magnitude of the exponentiated
coefficients always gives an idea of the effect of the covariates. In this example, female is constant within the group. The estimated hazard ratio for female is 0.19, which indicates that hazard functions for females tend to be smaller than hazard functions for males. Both conditional and unconditional predictions can be obtained with predict. Unconditional predictions can be visualized by using stcurve. Unconditional effects can be tested and visualized by using margins and marginsplot. See example 1 in [ME] mestreg postestimation for an example using predict, margins, and marginsplot.

Example 2

Although the PH parameterization is more popular in the literature because the output is easier to interpret, the AFT parameterization is useful when we need to make comparisons with other models that have only an AFT parameterization. For example, we might want to compare the Weibull results from example 1 with the results from a gamma model.

Let’s redisplay the results of a Weibull PH model from example 1 as coefficients:

```
. mestreg, nohr
Mixed-effects Weibull regression
Group variable: patient

Obs per group:
  min = 2
  avg = 2.0
  max = 2
Integration method: mvaghermite

Wald chi2(2) = 10.12
Log likelihood = -98.742496 Prob > chi2 = 0.0063

_t     Coef.   Std. Err.     z  P>|z|     [95% Conf. Interval]
------------- ------------- -------- -------- ---------------
  age        .0073207   .0136875   0.53   0.593     -.0195063   .0341476
female     -1.658251   .5249686   -3.16   0.002    -2.68717   -.6293311
_cons       -4.921242   .9914022   -4.96   0.000    -6.864354   -2.978129
------------- ------------- -------- -------- ---------------
/ln_p        .2243249   .1402794    1.60   0.110     -.0506177    .4992675
------------- ------------- -------- -------- ---------------
patient      .8308563   .4978461
                  .2567373    2.688827
LR test vs. Weibull model: chibar2(01) = 9.40 Prob >= chibar2 = 0.0011
```
We can refit the Weibull model using the AFT parameterization by specifying option \texttt{time}.

```
. mestreg age female || patient:, distribution(weibull) time
   failure _d: infect
   analysis time _t: time
```

Fitting fixed-effects model:

<table>
<thead>
<tr>
<th>Iteration</th>
<th>Log likelihood</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>-115.32903</td>
</tr>
<tr>
<td>1</td>
<td>-112.15933</td>
</tr>
<tr>
<td>2</td>
<td>-103.91548</td>
</tr>
<tr>
<td>3</td>
<td>-103.44725</td>
</tr>
<tr>
<td>4</td>
<td>-103.44362</td>
</tr>
<tr>
<td>5</td>
<td>-103.44362</td>
</tr>
</tbody>
</table>

Refining starting values:

<table>
<thead>
<tr>
<th>Grid node</th>
<th>Log likelihood</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>-103.96845</td>
</tr>
</tbody>
</table>

Fitting full model:

<table>
<thead>
<tr>
<th>Iteration</th>
<th>Log likelihood</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>-103.96845</td>
</tr>
<tr>
<td>1</td>
<td>-100.99964</td>
</tr>
<tr>
<td>2</td>
<td>-98.8804</td>
</tr>
<tr>
<td>3</td>
<td>-98.744307</td>
</tr>
<tr>
<td>4</td>
<td>-98.742497</td>
</tr>
<tr>
<td>5</td>
<td>-98.742495</td>
</tr>
</tbody>
</table>

Mixed-effects Weibull regression -- AFT

|        | Coef. | Std. Err. | z     | P>|z| | [95% Conf. Interval] |
|--------|-------|-----------|-------|-------|----------------------|
| age    | -0.0058496 | 0.010872 | -0.54 | 0.591 | -0.0271585 - 0.0154592 |
| female | 1.325034 | 0.3719102 | 3.56 | 0.000 | 0.596103 - 2.053964 |
| _cons  | 3.932346 | 0.5663757 | 6.94 | 0.000 | 2.822227 - 5.042422 |

|        | Coef. | Std. Err. | z     | P>|z| | [95% Conf. Interval] |
|--------|-------|-----------|-------|-------|----------------------|
| /ln_p  | 0.2243237 | 0.1402794 | 1.60 | 0.110 | -0.0506189 - 0.4992663 |
| patient| 0.5304902 | 0.2343675 | 2.231626 | 1.261053 |

LR test vs. Weibull model: \texttt{chibar2(01) = 9.40} \hspace{1cm} \text{Prob >= chibar2 = 0.0011}

The estimates of coefficients and variance components are different between the two models. In fact, the coefficients have the opposite signs. This is expected because the two models have different parameterizations. The relationship between the coefficients and variances of the two parameterizations for the Weibull model is

\[
\beta_{PH} = -p \times \beta_{AFT} \\
\text{var}_{PH} = p^2 \times \text{var}_{AFT}
\]

where \( p \) denotes the ancillary parameter. It is estimated in the logarithmic metric and is displayed in the output as \(/\ln_p\).

For example, we could calculate \( \beta_{PH} \) for \texttt{female} as approximately \(- \exp(0.22) \times 1.33 = -1.66\). If we exponentiate this to obtain the hazard ratio that was reported in example 1, we obtain the same reported result, 0.19.
For a discussion of the differences between the PH and AFT parameterizations, see, for example, Cleves et al. (2010).

Now, we can compare the results from our Weibull specification with the results from a gamma specification.

```
. mestreg age female || patient:, distribution(gamma)
  failure _d: infect
  analysis time _t: time
Fitting fixed-effects model:
Iteration 0:  log likelihood = -351.17349
Iteration 1:  log likelihood = -337.04571
Iteration 2:  log likelihood = -335.10167
Iteration 3:  log likelihood = -335.09115
Iteration 4:  log likelihood = -335.09115
Refining starting values:
Grid node 0:  log likelihood = -334.49759
Fitting full model:
Iteration 0:  log likelihood = -334.49759
Iteration 1:  log likelihood = -331.87827
Iteration 2:  log likelihood = -329.64795
Iteration 3:  log likelihood = -329.52682
Iteration 4:  log likelihood = -329.52635
Iteration 5:  log likelihood = -329.52634
Mixed-effects gamma regression
Number of obs = 76
Group variable:  patient  Number of groups = 38
  Obs per group:
    min =  2
    avg =  2.0
    max =  2
Integration method: mvaghermite
  Integration pts. =  7
Log likelihood = -329.52634  Wald chi2(2) = 13.23
                   Prob > chi2 =  0.0013

                     _t       Coef.   Std. Err.     z     P>|z|    [95% Conf. Interval]
    age         -.0060276   .0108267   -0.56   0.578   -.0272475 -.0151924
    female     1.324745     .3685132    3.59   0.000     .6024726  2.047018
    _cons       3.873854     .5628993    6.88   0.000   2.770592   4.977117
   /logs   -.1835075     .1008892   -1.82   0.069   -.3812467   .0142317

patient  var(_cons)   .5071823   .2241959    2.26   0.024     .0873937   1.027876
LR test vs. gamma model:  chibar2(01) = 11.13  Prob >= chibar2 =  0.0004
```

The coefficients and the random-effects variance are very similar for the two AFT models.

We can compare the marginal distributions or hazard functions for the two models by using `stcurve`; see example 2 in [ME] `mestreg postestimation`. 
Example 3

In this example, we use a modified form of the dataset from Rabe-Hesketh and Skrondal (2012, chap. 15.7), previously published in Danahy et al. (1977) and analyzed by Pickles and Crouchley (1994,1995) and Rabe-Hesketh, Skrondal, and Pickles (2004).

angina.dta includes data on 21 patients with coronary heart disease who participated in a randomized crossover trial comparing a drug to prevent angina (chest pain) with a placebo. The participants are identified by pid.

Before receiving the drug (or placebo), participants were asked to exercise on exercise bikes to the onset of angina or, if angina did not occur, to exhaustion. The exercise time, seconds, and the result of the exercise, angina—angina (angina=1) or exhaustion (angina=0)—were recorded. The drug (treat=1) or placebo (treat=0) was then taken orally, and the exercise test was repeated one, three, and five hours (variable occasion) after drug or placebo administration. Because each exercise test can have a failure (the occurrence of angina), the test is the subject. Each test is identified by tid. Failure is indicated by the variable angina. In this case, we have eight repeated measures per study participant.

Before fitting the model, we stset our data:

```
. use http://www.stata-press.com/data/r14/angina
(Angina drug data, Rabe-Hesketh and Skrondal, 2012, ch 15.7)
. stset seconds, failure(angina) id(tid)
    id: tid
    failure event: angina != 0 & angina < .
    obs. time interval: (seconds[_n-1], seconds]
    exit on or before: failure
```

168 total observations
0 exclusions

168 observations remaining, representing
168 subjects
155 failures in single-failure-per-subject data
47267 total analysis time at risk and under observation

at risk from t = 0
earliest observed entry t = 0
last observed exit t = 743

To reiterate, we specify seconds as the time variable, angina as the failure variable, and tid as the variable identifying multiple observations per test.

Rabe-Hesketh and Skrondal (2012) apply several models to this dataset, including a lognormal model and a Cox model with random effects. We fit a Weibull model with covariates occasion and treat and interaction between occasion and treat. We include a random effect at the subject level.
. mestreg occasion#treat || pid:, distribution(weibull)

failure _d: angina
analysis time _t: seconds
id: tid

note: 1.occasion#1.treat identifies no observations in the sample
note: 4.occasion#1.treat omitted because of collinearity
(output omitted)

Mixed-effects Weibull regression  Number of obs = 168
Group variable: pid  Number of groups = 21
Obs per group:
  min = 8
  avg = 8.0
  max = 8

Integration method: mvaghermite  Integration pts. = 7
Wald chi2(6) = 78.14
Log likelihood = -40.058632  Prob > chi2 = 0.0000

|          | Haz. Ratio | Std. Err. | z    | P>|z|  | [95% Conf. Interval] |
|----------|-----------|-----------|------|------|----------------------|
| _t       |           |           |      |      |                      |
| occasion |           |           |      |      |                      |
| 2        | 0.719456  | 0.2031744 | -1.17| 0.244| 0.4136423 1.251364   |
| 3        | 0.902988  | 0.2542746 | -0.36| 0.717| 0.5200146 1.568009   |
| 4        | 1.264262  | 0.3516346 | 0.84 | 0.399| 0.7329746 2.180648   |
| treat    |           |           |      |      |                      |
| Drug     | 0.3825531 | 0.128784  | -2.85| 0.004| 0.1977608 0.7400195  |
| occasion# |           |           |      |      |                      |
| treat    |           |           |      |      |                      |
| 1#Drug   | 1 (empty) |           |      |      |                      |
| 2#Drug   | 0.1576401 | 0.0804767 | -3.62| 0.000| 0.0579589 0.4287586  |
| 3#Drug   | 0.4512793 | 0.2127706 | -1.69| 0.091| 0.1791093 1.137032   |
| 4#Drug   | 1 (omitted)|           |      |      |                      |
| _cons    | 4.90e-13  | 9.98e-13  | -13.91| 0.000| 9.03e-15 2.66e-11   |
| /ln_p    | 1.640297  | 0.0689544 | 23.79| 0.000| 1.505149 1.775445   |
| pid      |           |           |      |      |                      |
| var(_cons)| 4.529709 | 1.544218  | 2.322138 | 8.835934 |

LR test vs. Weibull model: chibar2(01) = 177.40  Prob >= chibar2 = 0.0000

Because individuals were exercising without the administration of a placebo or treatment at the first occasion (occasion==1), the category for interaction between occasion==1 and treat==1 is empty.

The estimated variance at the individual level (that is, the variance between individuals) is equal to 4.53. The likelihood-ratio test shows evidence in favor of the random-effects model versus the fixed-effects model.

The parameter $p$ is $\exp(1.640297) = 5.16$, which is larger than 1. This means that the estimated hazard (conditional on the covariates and on the random effects) is a monotonically increasing function if we assume a Weibull distribution.
The model contains interaction terms for occasion and treat. Interpretation of interaction terms is usually less straightforward. Briefly, to interpret the exponentiated coefficients as conditional hazard ratios, we need to examine all the covariates in the interaction. The hazard function for pid = j, when we set occasion = k and treat = l, will be

\[
h(t) = h_0(t) \times \exp(\beta_{occk} + \beta_{treatl} + \beta_{occk \times treatl} + \_cons + u_j)
\]

where \( \beta_{occk}, \beta_{treatl}, \) and \( \beta_{occk \times treatl} \) are, respectively, the coefficients for the dummies for occasion = k and treat = l and the interaction (occasion = \( k \times \) treatment = \( l \)).

For example, when treat = 0, the hazard function is

\[
h(t | \text{treat} = 0, \text{occasion} = k, \text{pid} = j) = h_0(t) \times \exp(\beta_{occk} + \_cons + u_j)
\]

where \( \beta_{occk} \) is equal to 0 because occasion = 1 is the base category. This means that for a given pid,

\[
\frac{h(t | \text{treat} = 0, \text{occ} = k, \text{pid} = j)}{h(t | \text{treat} = 0, \text{occ} = 1, \text{pid} = j)} = \exp(\beta_{occk})
\]

Notice that this is only true within pid, because different participants have different \( u_j \)s.

The coefficients have already been exponentiated, so we can see clearly that according to this model, when there is no treatment, the hazard for occasion 2 is smaller than the hazard for occasion 1. The increasing ratios indicate that the hazard increases with the occasion. Similar calculations could be performed for other interaction terms.

The easiest way to interpret models with interactions is by using \texttt{margins} and \texttt{marginsplot}, which allow us to compute and then visualize unconditional predictions and marginal effects. See \texttt{[R] margins} for more information.

Above we assumed a constant treatment effect for all individuals for each occasion. However, we may instead believe that the treatment effect varies also with individuals. This can be modeled by adding a random coefficient for the treatment, treat, at the individual level; we also include the \texttt{covariance(unstructured)} option to estimate a covariance term between the random intercept and the random slope for treat.
. mestreg occasion#treat || pid: treat, distribution(weibull)  
  > covariance(unstructured)  
  failure _d: angina  
  analysis time _t: seconds  
  id: tid  

Note: 1. occasion#1.treat identifies no observations in the sample  
Note: 4. occasion#1.treat omitted because of collinearity  
(output omitted)

Mixed-effects Weibull regression  
Number of obs = 168  
Group variable: pid  
Number of groups = 21  
Obs per group:  
  min = 8  
  avg = 8.0  
  max = 8  
Integration method: mvaghermite  
Integration pts. = 7  
Log likelihood = -13.887663  
Wald chi2(6) = 50.18  
Prob > chi2 = 0.0000

|   | Haz. Ratio | Std. Err. | z   | P>|z| | [95% Conf. Interval] |
|---|-----------|-----------|-----|-----|---------------------|
| occasion |         |           |     |     |                     |
| 2 | .5993579 | .1861741  | -1.65 | 0.099 | .3260497 1.101764 |
| 3 | .8643304 | .2560241  | -0.49 | 0.623 | .4836649 1.544596 |
| 4 | 1.333201 | .3843217  | 1.00  | 0.318 | .7577397 2.345694 |
| treat |         |           |     |     |                     |
| Drug | .2147832 | .1280133  | -2.58 | 0.010 | .0667843 .6907586 |
| occasion#treat |         |           |     |     |                     |
| 1#Drug |         |           |     |     |                     |
| 2#Drug | .1594337 | .0885643  | -3.31 | 0.001 | .0536715 .4736052 |
| 3#Drug | .4632939 | .2273925  | -1.57 | 0.117 | .1770404 1.212385 |
| 4#Drug |         |           |     |     |                     |
| _cons | 6.21e-17 | 1.75e-16  | -13.20 | 0.000 | 2.44e-19 1.58e-14 |

/ln_p  
  1.919312 .0736148 26.07 0.000 1.775029 2.063594

pid  
  var(treat) 4.682494 1.95688 2.064183 10.622
  var(_cons) 6.939258 2.373094 3.549918 13.56462

pid  
  cov(_cons, treat) 1.737728 1.313064 1.32 0.186 -.8358292 4.311286

LR test vs. Weibull model: chi2(3) = 229.74  
Prob > chi2 = 0.0000

Note: LR test is conservative and provided only for reference.

We obtain somewhat different estimates of hazard ratios, but our inferential conclusions remain the same. We now observe two variances in the output, the variance for the intercept at the individual level and the variance for the coefficient for treatment at the individual level. The variance for the intercept is smaller because some of the variability is now explained by varying coefficients for treatment. The covariance is positive, meaning that the random slope tends to be larger for individuals who have a larger random intercept. See example 4 in [ME] mestreg postestimation for an application of predict that presents a graphical analysis of this relationship.
Three-level models

Example 4

Blossfeld, Golsch, and Rohwer (2007) analyze a dataset based on the German Life History Study of Mayer and Brückner (1989), collected in the years 1981–1983. The jobhistory dataset contains a modified version of Blossfeld, Golsch, and Rohwer’s anonymization of a random sample of 201 respondents from the original data. Each of the 600 observations in the dataset corresponds to a job episode. Variable id contains identification of the individual, tstart contains the starting point of the job (in months from the beginning of the century), tend is the end of the job episode, and failure indicates whether the date in tend corresponds to the actual end of the employment in a certain job or whether it is a censored observation.

We first stset the data. As explained in Cleves (1999) and Therneau and Grambsch (2000), when analyzing multiple-failure data, we can consider two main approaches. One approach is to define the study time from the first time that an individual starts being at risk. The second approach is to define the study time from the last failure. We will take the second approach, which means that we treat each job episode as the subject.

Therefore, the origin is defined as the start of each job episode, and the study time will be the time from the start of each episode until the jobs end or the episode is censored.

```
use http://www.stata-press.com/data/r14/jobhistory
(Job history data, Blossfeld et al 2007)
.stset tend, origin(tstart) failure(failure)
  failure event: failure != 0 & failure < .
obs. time interval: (origin, tend]
exit on or before: failure
  t for analysis: (time-origin)
  origin: time tstart

600  total observations
  0  exclusions
600  observations remaining, representing
458  failures in single-record/single-failure data
40782  total analysis time at risk and under observation
  at risk from t = 0
  earliest observed entry t = 0
  last observed exit t = 428
```

We want to fit a Weibull model using the education level, the number of previous jobs, the prestige of the current job, and gender as explanatory variables. education records the highest education level before entering the labor market, njobs contains the number of previous jobs for each individual, and prestige is an index for the prestige of the current job. The birthyear variable indicates the year of birth. female is 1 for women, 0 for men. To account for individual heterogeneity, we include a random effect at the individual level.
. mestreg education njobs prestige i.female || id:, distribution(weibull)
    failure _d: failure
    analysis time _t: (tend-origin)
    origin: time tstart

Fitting fixed-effects model:
Iteration 0:  log likelihood = -5735328.5
Iteration 1:  log likelihood = -1088.6966
Iteration 2:  log likelihood = -908.73089
Iteration 3:  log likelihood = -901.38382
Iteration 4:  log likelihood = -901.2818
Iteration 5:  log likelihood = -901.28171
Iteration 6:  log likelihood = -901.28171

Refining starting values:
Grid node 0:  log likelihood = -915.16707

Fitting full model:
Iteration 0:  log likelihood = -915.16707  (not concave)
Iteration 1:  log likelihood = -896.53044
Iteration 2:  log likelihood = -881.07771
Iteration 3:  log likelihood = -869.73564
Iteration 4:  log likelihood = -866.85736
Iteration 5:  log likelihood = -866.82219
Iteration 6:  log likelihood = -866.82263
Iteration 7:  log likelihood = -866.82262

Mixed-effects Weibull regression  Number of obs = 600
Group variable: id  Number of groups = 201
Obs per group:
   min = 1
   avg = 3.0
   max = 9
Integration method: mvaghermite  Integration pts. = 7
Log likelihood = -866.82262
Wald chi2(4) = 87.38  Prob > chi2 = 0.0000

|       | Haz. Ratio | Std. Err. | z     | P>|z|   | [95% Conf. Interval] |
|-------|------------|-----------|-------|-------|----------------------|
| _t    |            |           |       |       |                      |
| education | 1.11897    | .0463468  | 2.71  | 0.007 | 1.031722 1.213597    |
| njobs  | .7071195   | .0357624  | -6.85 | 0.000 | .6403884 .7808043   |
| prestige| .9677567   | .0069576  | -4.56 | 0.000 | .9542157 .98149     |
| i.female| 1.75651    | .3188527  | 3.11  | 0.002 | 1.231063 2.506228   |
| _cons  | .0053352   | .0029015  | -9.62 | 0.000 | .0018376 .0154904   |
| /ln_p  | .1695545   | .0453649  | 3.74  | 0.000 | .0806409 .2584681   |

id
var(_cons) | 1.016459 | .2149037 | .671623 | 1.538347

LR test vs. Weibull model: chibar2(01) = 68.92  Prob >= chibar2 = 0.0000

The estimated variance of the random intercept is equal to 1.02

According to this model, an increase in the number of previous jobs is negatively associated with job mobility; the same is true for an increase in the prestige of the current job. By contrast, an increase in the years of education is positively associated with job mobility. Also, women seem to be more mobile than men.

We now store our estimates for later use:

. estimates store randint
The dataset has only two natural levels. However, for illustration purposes, let's consider the following situation. Assume that we want to account for unobserved variables associated with the date of birth, such as life experience, level of familiarity with new technologies, and family situation. We therefore add a random effect for the year of birth. Now, individuals will be nested within birth years.

```
. mestreg education njobs prestige i.female || birthyear: || id:,
    > distribution(weibull)
    failure _d:  failure
    analysis time _t:  (tend-origin)
    origin:  time tstart
```

(output omitted)

Mixed-effects Weibull regression

<table>
<thead>
<tr>
<th>Group Variable</th>
<th>No. of Observations per Group</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Groups</td>
</tr>
<tr>
<td>birthyear</td>
<td>12</td>
</tr>
<tr>
<td>id</td>
<td>201</td>
</tr>
</tbody>
</table>

Integration method: mvaghermite
Integration pts. = 7

Log likelihood = -863.85463
Wald chi2(4) = 83.20
Prob > chi2 = 0.0000

| _t        | Haz. Ratio | Std. Err. | z   | P>|z| | [95% Conf. Interval] |
|-----------|------------|------------|-----|------|----------------------|
| education | 1.120373   | .0452029   | 2.82| 0.005| 1.035189 1.212566   |
| njobs     | .7181197   | .0372039   | -6.39| 0.000| .6487813 .7948686   |
| prestige  | .966567    | .0069189   | -4.75| 0.000| .9531009 .9802234   |
| i.female  | 1.734236   | .3022478   | 3.16| 0.002| 1.232418 2.440383   |
| _cons     | .059091    | .0031758   | -9.55| 0.000| .0020609 .0169429   |
| /ln_p     | .1685641   | .0458242   | 3.71| 0.000| .0794203 .2577079   |

| _t        | Haz. Ratio | Std. Err. | z   | P>|z| | [95% Conf. Interval] |
|-----------|------------|------------|-----|------|----------------------|
| birthyear | .0950325   | .0741398   | 1.28| 0.200| .020597  .4384694   |
| var(_cons)| .8728391   | .202094    | .000| 1.000| .000000  .000000    |

LR test vs. Weibull model: chi2(2) = 74.85
Prob > chi2 = 0.0000

Note: LR test is conservative and provided only for reference.

The results for the fixed part of the model are similar to the ones in the previous model.

Now, we have two estimated variances—one estimate for the random intercept at the individual level and one estimate for the random intercept at the birth-year level.

The variance component for the individual level is smaller for this model, and it looks as if the first model might have been trying to explain a variance component at the birth-year level by incorporating it into the individual-level variance. We can perform a likelihood-ratio test to compare the stored model `randint` with the current model:

```
. lrtest randint .
Likelihood-ratio test
(Assumption: randint nested in .)  LR chi2(1) = 5.94
Prob > chi2 = 0.0148
```

Note: The reported degrees of freedom assumes the null hypothesis is not on the boundary of the parameter space. If this is not true, then the reported test is conservative.
The test is conservative because we are testing on the boundary of the parameter space; see Distribution theory for likelihood-ratio test in [ME] me for details. Provided that we are testing only one variance component, we can adjust the \( p \)-value accordingly by dividing the reported value by two, which results in an adjusted \( p \)-value equal to 0.0074.

The test is significant at the 0.05 level. It supports the three-level model with the additional variance component at the birth-year level.

### Stored results

*mestreg* stores the following in e():

**Scalars**
- \( e(N) \): number of observations
- \( e(k) \): number of parameters
- \( e(k_eq) \): number of equations in \( e(b) \)
- \( e(k_eq_model) \): number of equations in overall model test
- \( e(k_dv) \): number of dependent variables
- \( e(k_f) \): number of fixed-effects parameters
- \( e(k_r) \): number of random-effects parameters
- \( e(k_rs) \): number of variances
- \( e(k_rc) \): number of covariances
- \( e(df_m) \): model degrees of freedom
- \( e(ll) \): log likelihood
- \( e(chi2) \): \( \chi^2 \)
- \( e(p) \): significance
- \( e(ll_c) \): log likelihood, comparison model
- \( e(chi2_c) \): \( \chi^2 \), comparison model
- \( e(df_c) \): degrees of freedom, comparison model
- \( e(p_c) \): significance, comparison model
- \( e(N_clust) \): number of clusters
- \( e(rank) \): rank of \( e(V) \)
- \( e(ic) \): number of iterations
- \( e(rc) \): return code
- \( e(converged) \): 1 if converged, 0 otherwise

**Macros**
- \( e(cmd) \): gsem
- \( e(cmd2) \): mestreg
- \( e(cmdline) \): command as typed
- \( e(depvar) \): name of dependent variable
- \( e(vtype) \): weight type
- \( e(wexp) \): weight expression (first-level weights)
- \( e(fweightk) \): \( fweight \) variable for \( k \)th highest level, if specified
- \( e(iweightk) \): \( iweight \) variable for \( k \)th highest level, if specified
- \( e(pweightk) \): \( pweight \) variable for \( k \)th highest level, if specified
- \( e(covariates) \): list of covariates
- \( e(ivars) \): grouping variables
- \( e(model) \): model name
- \( e(title) \): title in estimation output
- \( e(distribution) \): distribution
- \( e(clustvar) \): name of cluster variable
- \( e(offset) \): offset
- \( e(exposure) \): exposure variable
- \( e(intmethod) \): integration method
- \( e(n_quad) \): number of integration points
- \( e(chi2type) \): Wald; type of model \( \chi^2 \)
- \( e(vce) \): \( vcetype \) specified in vce()
Methods and formulas are presented under the following headings:

Survival models
Survey data

Survival models

Survival models have a trivariate response \((t_0, t, d)\):

- \(t_0\) is the starting time under observation \(t_0 \geq 0\);
- \(t\) is the ending time under observation \(t \geq t_0\); and
- \(d\) is an indicator for failure \(d \in \{0, 1\}\).

The survival function for a given family is the complement of the cumulative distribution function, \(S(t) = 1 - F(t)\). The unconditional density for a failure at time \(t\) is given by

\[
g(t) = \frac{\partial F(t)}{\partial t} = -\frac{\partial S(t)}{\partial t}
\]

Some distributions contain ancillary parameters that are not denoted here.

The conditional density for a failure at time \(t\) is

\[
g(t|t \geq t_0, d = 1) = \frac{g(t)}{S(t_0)}
\]
and the conditional probability of survival without failure up to time \( t \) is
\[
P(T \geq t | t \geq t_0, d = 0) = S(t)/S(t_0)
\]
The conditional likelihood is given by
\[
L(t, t_0, d) = \left\{ \frac{g(t)}{S(t_0)} \right\}^d \left\{ \frac{S(t)}{S(t_0)} \right\}^{1-d}
\]
See Survival distributions in [SEM] methods and formulas for gsem for the specific density function corresponding to each distribution.

Given a set of cluster-level random effects \( u_j \) for \( j = 1, \ldots, M \), the conditional distribution of \( t_j = (t_{j1}, \ldots, t_{jn_j})' \) on \( \eta_j = X_j\beta + Z_j u_j = (x_{j11}+ z_{j1} u_j, \ldots, x_{jn_j}+ z_{jn_j} u_j) \) for cluster \( j \) is
\[
f(t_j | \eta_j) = \prod_{i=1}^{n_j} f(t_{ji} | \eta_{ji})
\]
where \( f(t_{ji} | \eta_{ji}) \) is the contribution to the likelihood from observation \( ji \); that is,
\[
f(t_{ji} | \eta_{ji}) = \left\{ \frac{g(t_{ji} | x_{ji}\beta + z_{ji} u_j)}{S(t_{0ji} | x_{ji}\beta + z_{ji} u_j)} \right\}^{d_{ji}} \left\{ \frac{S(t_{ji} | x_{ji}\beta + z_{ji} u_j)}{S(t_{0ji} | x_{ji}\beta + z_{ji} u_j)} \right\}^{1-d_{ji}}
\]
(1)
where \( g(t | \eta) \) and \( S(t | \eta) \) are, respectively, the density and the survivor function conditional on the linear prediction \( \eta \).

As mentioned in Introduction under Remarks and examples, mestreg does not allow delayed entry or gaps. Therefore, the first observation for a given subject will have a value of \( t_0 = 0 \), and subsequent spells for the subject must start at the end of the previous spell. That is, if observations \( ji \) and \( j, i+1 \) belong to the same subject, then \( t_{0ji,i+1} = t_{ji} \).

Because the prior distribution of \( u_j \) is multivariate normal with mean \( \mathbf{0} \) and \( q \times q \) variance matrix \( \Sigma \), the likelihood contribution for the \( j \)th cluster is obtained by integrating \( u_j \) out of the joint density \( f(t_j, u_j) \),
\[
L_j(\beta, \Sigma) = (2\pi)^{-q/2} |\Sigma|^{-1/2} \int f(t_j | X_j\beta + Z_j u_j) \exp \left(-u_j' \Sigma^{-1} u_j / 2\right) \, du_j
\]
(2)
The integration in (2) has no closed form and thus must be approximated. mestreg offers four approximation methods: mean–variance adaptive Gauss–Hermite quadrature (default unless a crossed random-effects model is fit), mode-curvature adaptive Gauss–Hermite quadrature, nonadaptive Gauss–Hermite quadrature, and Laplacian approximation (default for crossed random-effects models).

The Laplacian approximation is based on a second-order Taylor expansion; see Methods and formulas in [ME] meglm for details.

Gaussian quadrature relies on transforming the multivariate integral in (2) into a set of nested univariate integrals. Each univariate integral can then be evaluated using a form of Gaussian quadrature; see Methods and formulas in [ME] meglm for details.

The log likelihood for the entire dataset is simply the sum of the contributions of the \( M \) individual clusters, namely, \( \mathcal{L}(\beta, \Sigma) = \sum_{j=1}^{M} L_j(\beta, \Sigma) \).

Maximization of \( \mathcal{L}(\beta, \Sigma) \) is performed with respect to \( (\beta, \sigma^2) \), where \( \sigma^2 \) is a vector comprising the unique elements of \( \Sigma \). Parameter estimates are stored in \( \hat{e(b)} \) as \( (\hat{\beta}, \hat{\sigma}^2) \), with the corresponding variance–covariance matrix stored in \( \hat{e(V)} \).
Survey data

In the presence of sampling weights, following Rabe-Hesketh and Skrondal (2006), the weighted log pseudolikelihood for a two-level model is given as

\[
\mathcal{L}(\beta, \Sigma) = \sum_{j=1}^{M} w_j \log \int_{-\infty}^{\infty} \exp \left\{ \sum_{i=1}^{n_j} w_{ij} \log f(t_{ji} | \eta_{ji}) \right\} \phi(v_{j1}) \, dv_{j1}
\]

where \( w_j \) is the inverse of the probability of selection for the \( j \)th cluster; \( w_{ij} \) is the inverse of the conditional probability of selection of individual \( i \), given the selection of cluster \( j \); \( f(t_{ji} | \eta_{ji}) \) is as in (1); and \( \eta_{ji}, \phi(\cdot), v_{j1} \) are defined as in Methods and formulas in [ME] \texttt{meglm}.

Weighted estimation is achieved through the direct application of \( w_j \) and \( w_{ij} \) into the likelihood calculations as detailed above to reflect replicated clusters for \( w_j \) and replicated observations within clusters for \( w_{ij} \). Because this estimation is based on replicated clusters and observations, frequency weights are handled similarly.

References


Also see

[ME] mestreg postestimation — Postestimation tools for mestreg

[ME] me — Introduction to multilevel mixed-effects models

[ST] streg — Parametric survival models

[ST] Glossary

[SVY] svy estimation — Estimation commands for survey data

[XT] xtstreg — Random-effects parametric survival models

[U] 20 Estimation and postestimation commands
Postestimation commands

The following postestimation commands are of special interest after `mestreg`:

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>stcurve</td>
<td>plot the survivor, hazard, and cumulative hazard functions</td>
</tr>
<tr>
<td>estat group</td>
<td>summarize the composition of the nested groups</td>
</tr>
</tbody>
</table>

The following standard postestimation commands are also available:

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>contrast</td>
<td>contrasts and ANOVA-style joint tests of estimates</td>
</tr>
<tr>
<td>estat ic</td>
<td>Akaike’s and Schwarz’s Bayesian information criteria (AIC and BIC)</td>
</tr>
<tr>
<td>estat summarize</td>
<td>summary statistics for the estimation sample</td>
</tr>
<tr>
<td>estat vce</td>
<td>variance–covariance matrix of the estimators (VCE)</td>
</tr>
<tr>
<td>estat (svy)</td>
<td>postestimation statistics for survey data</td>
</tr>
<tr>
<td>estimates</td>
<td>cataloging estimation results</td>
</tr>
<tr>
<td>*hausman</td>
<td>Hausman’s specification test</td>
</tr>
<tr>
<td>lincom</td>
<td>point estimates, standard errors, testing, and inference for linear</td>
</tr>
<tr>
<td></td>
<td>combinations of coefficients</td>
</tr>
<tr>
<td>*lrtest</td>
<td>likelihood-ratio test</td>
</tr>
<tr>
<td>margins</td>
<td>marginal means, predictive margins, marginal effects, and average marginal</td>
</tr>
<tr>
<td></td>
<td>effects</td>
</tr>
<tr>
<td>marginsplot</td>
<td>graph the results from margins (profile plots, interaction plots, etc.)</td>
</tr>
<tr>
<td>nlcom</td>
<td>point estimates, standard errors, testing, and inference for nonlinear</td>
</tr>
<tr>
<td></td>
<td>combinations of coefficients</td>
</tr>
<tr>
<td>predict</td>
<td>predictions, residuals, influence statistics, and other diagnostic measures</td>
</tr>
<tr>
<td>predictnl</td>
<td>point estimates, standard errors, testing, and inference for generalized</td>
</tr>
<tr>
<td></td>
<td>predictions</td>
</tr>
<tr>
<td>pwcompare</td>
<td>pairwise comparisons of estimates</td>
</tr>
<tr>
<td>test</td>
<td>Wald tests of simple and composite linear hypotheses</td>
</tr>
<tr>
<td>testnl</td>
<td>Wald tests of nonlinear hypotheses</td>
</tr>
</tbody>
</table>

* `hausman` and `lrtest` are not appropriate with `svy` estimation results.
predict

Description for predict

predict creates a new variable containing predictions such as mean and median survival times, hazards, survivor functions, linear predictions, and standard errors.

Menu for predict

Statistics > Postestimation

Syntax for predict

Syntax for obtaining predictions of the outcome and other statistics

predict [type] newvarsspec [if] [in] [ , statistic options ]

Syntax for obtaining estimated random effects and their standard errors

predict [type] newvarsspec [if] [in], reffects [re_options ]

Syntax for obtaining ML scores

predict [type] newvarsspec [if] [in], scores

newvarsspec is stub* or newvarlist.

<table>
<thead>
<tr>
<th>statistic</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mean</td>
<td>mean survival time; the default</td>
</tr>
<tr>
<td>median</td>
<td>median survival time</td>
</tr>
<tr>
<td>hazard</td>
<td>hazard</td>
</tr>
<tr>
<td>eta</td>
<td>fitted linear predictor</td>
</tr>
<tr>
<td>xb</td>
<td>linear predictor for the fixed portion of the model only</td>
</tr>
<tr>
<td>stdp</td>
<td>standard error of the fixed-portion linear prediction</td>
</tr>
<tr>
<td>surv</td>
<td>predicted survivor function</td>
</tr>
<tr>
<td>density</td>
<td>predicted density function</td>
</tr>
<tr>
<td>distribution</td>
<td>predicted distribution function</td>
</tr>
</tbody>
</table>

These statistics are available both in and out of sample; type predict ... if e(sample) ... if wanted only for the estimation sample.
options

<table>
<thead>
<tr>
<th>Main</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>conditional(cctype)</strong></td>
<td>compute statistic conditional on estimated random effects; default is conditional(ebmeans)</td>
</tr>
<tr>
<td>marginal</td>
<td>compute statistic marginally with respect to the random effects</td>
</tr>
<tr>
<td>nooffset</td>
<td>make calculation ignoring offset or exposure</td>
</tr>
</tbody>
</table>

Integration

<table>
<thead>
<tr>
<th><strong>int_options</strong></th>
<th>integration options</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>ctype</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ebmeans</td>
<td>empirical Bayes means of random effects; the default</td>
</tr>
<tr>
<td>ebmodes</td>
<td>empirical Bayes modes of random effects</td>
</tr>
<tr>
<td>fixedonly</td>
<td>prediction for the fixed portion of the model only</td>
</tr>
</tbody>
</table>

Integration

<table>
<thead>
<tr>
<th><strong>int_options</strong></th>
<th>integration options</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Main</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ebmeans</td>
<td>use empirical Bayes means of random effects; the default</td>
</tr>
<tr>
<td>ebmodes</td>
<td>use empirical Bayes modes of random effects</td>
</tr>
<tr>
<td>reses(stub*</td>
<td>newvarlist)</td>
</tr>
</tbody>
</table>

Integration

<table>
<thead>
<tr>
<th><strong>int_options</strong></th>
<th>integration options</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th><strong>int_options</strong></th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>intpoints(#)</td>
<td>use # quadrature points to compute marginal predictions and empirical Bayes means</td>
</tr>
<tr>
<td>iterate(#)</td>
<td>set maximum number of iterations in computing statistics involving empirical Bayes estimators</td>
</tr>
<tr>
<td>tolerance(#)</td>
<td>set convergence tolerance for computing statistics involving empirical Bayes estimators</td>
</tr>
</tbody>
</table>

Options for predict

- **Main**
  - mean, the default, calculates the mean survival time.
  - median calculates the median survival time.
  - hazard calculates the hazard. When marginal is specified, marginal hazard is calculated as a ratio of the marginal density to the marginal survivor function.
  - surv calculates the predicted survivor function.
  - eta, xb, stdp, density, distribution, scores, conditional(), marginal, and nooffset; see [ME] meglm postestimation.
  - reffects, ebmeans, ebmodes, and reses(); see [ME] meglm postestimation.
margins

Description for margins

margins estimates margins of response for mean and median survival times and linear predictions.

Menu for margins

Statistics > Postestimation

Syntax for margins

margins [marginlist] [ , options]
margins [marginlist], predict(statistic ...) [predict(statistic ...) ...] [options]

<table>
<thead>
<tr>
<th>statistic</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mean</td>
<td>mean survival time; the default</td>
</tr>
<tr>
<td>median</td>
<td>median survival time</td>
</tr>
<tr>
<td>xb</td>
<td>linear predictor for the fixed portion of the model only</td>
</tr>
<tr>
<td>hazard</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>eta</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>stdp</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>surv</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>density</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>distribution</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>reffects</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>scores</td>
<td>not allowed with margins</td>
</tr>
</tbody>
</table>

Options conditional(ebmeans) and conditional(ebmoses) are not allowed with margins. Option marginal is assumed where applicable if conditional(fixedonly) is not specified.

Statistics not allowed with margins are functions of stochastic quantities other than e(b).

For the full syntax, see [R] margins.

Remarks and examples

Various predictions, statistics, and diagnostic measures are available after fitting a mixed-effects parametric survival model with mestreg. For the most part, predictions center on obtaining estimates of the survival times or hazard functions. Conditional predictions are based on the computation of the group-specific random effects, and marginal predictions are obtained by numerically integrating out the random effects.
Example 1

In example 1 of [ME] mestreg, we analyzed the time to infection of the catheter insertion point for 38 kidney dialysis patients. We fit the following model:

```
. use http://www.stata-press.com/data/r14/catheter
   (Kidney data, McGilchrist and Aisbett, Biometrics, 1991)
. stset time, failure(infect)
   (output omitted)
. mestreg age female || patient:, distribution(weibull)
   (output omitted)
```

The `predict` command allows us to compute marginal and conditional predictions. Unless stated differently, we use the word “conditional” to mean “conditional on the empirical Bayes predictions of the random effects”. Below we compute marginal and conditional means for the mean survival time.

```
. predict m_marg, mean marginal
. predict m_cond, mean conditional
   (predictions based on fixed effects and posterior means of random effects)
   (using 7 quadrature points)
```

Now, we can display the predictions for some of the patients.

```
. sort female age patient
. list patient female age m_* in 15/20, sepby(patient)

<table>
<thead>
<tr>
<th>patient</th>
<th>female</th>
<th>age</th>
<th>m_marg</th>
<th>m_cond</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>29</td>
<td>53</td>
<td>52.7936</td>
<td>22.36021</td>
</tr>
<tr>
<td></td>
<td>29</td>
<td>53</td>
<td>52.7936</td>
<td>22.36021</td>
</tr>
<tr>
<td></td>
<td>16</td>
<td>60</td>
<td>50.67549</td>
<td>28.01287</td>
</tr>
<tr>
<td></td>
<td>16</td>
<td>60</td>
<td>50.67549</td>
<td>28.01287</td>
</tr>
<tr>
<td></td>
<td>38</td>
<td>60</td>
<td>50.67549</td>
<td>49.47013</td>
</tr>
<tr>
<td></td>
<td>38</td>
<td>60</td>
<td>50.67549</td>
<td>49.47013</td>
</tr>
</tbody>
</table>
```

We see in the output that the predicted expected conditional mean for patient 29 is equal to 22.36 (shown in `m_cond`). This is the expected time to infection for this patient. However, the predicted marginal mean for this patient is 52.79 (shown in `m_marg`). This is the expected time to infection for a patient from the population who is male and is 53 years old. This particular patient seems to be more prone to infection than would be expected based on his age and gender.

Conditional predictions are specific to each group, while marginal predictions are the same within each covariate pattern through the data. Patients 16 and 38 have the same covariate patterns; therefore, their marginal predicted means are the same. However, conditional predicted means differ.
margins and marginsplot show the changes in the marginal means for different ages.

```
. margins, predict(mean marginal) at(female=0 age=(20(5)70)) noatlegend
Adjusted predictions Number of obs = 76
Model VCE : OIM
Expression : Marginal predicted mean, predict(mean marginal)
```

| _at   | Marg | Std. Err. | z     | P>|z|  | [95% Conf. Interval] |
|-------|------|-----------|-------|------|---------------------|
| 1     | 64.03488 | 28.99891  | 2.21  | 0.027 | 7.19805             | 120.8717 |
| 2     | 62.1891   | 26.33293  | 2.36  | 0.018 | 10.5775             | 113.8007 |
| 3     | 60.39652  | 24.11465  | 2.50  | 0.012 | 13.13268            | 107.6604 |
| 4     | 58.65662  | 22.37009  | 2.62  | 0.009 | 14.81105            | 102.5002 |
| 5     | 56.96489  | 21.11496  | 2.70  | 0.007 | 15.58032            | 98.34946 |
| 6     | 55.3229   | 20.34546  | 2.72  | 0.007 | 15.44653            | 95.19927 |
| 7     | 53.72824  | 20.03199  | 2.68  | 0.007 | 14.46626            | 92.90023 |
| 8     | 52.17955  | 20.12007  | 2.59  | 0.010 | 12.74494            | 91.61415 |
| 9     | 50.67549  | 20.53858  | 2.47  | 0.014 | 10.42062            | 90.93037 |
| 10    | 49.21479  | 21.2114   | 2.32  | 0.020 | 7.641204            | 90.78838 |
| 11    | 47.7962   | 22.06721  | 2.17  | 0.030 | 4.545267            | 91.04712 |

```
. marginsplot
Variables that uniquely identify margins: age
```

We see that the predicted marginal mean decreases with age; older patients are expected to have an event earlier. This is consistent with the findings from example 1 of [ME] mestreg that the hazard is increasing with age.
Example 2

Continuing with example 1, we now predict survivor functions.

\[ \text{. predict } S_{\text{marg}}, \text{ surv marginal} \]
(\text{using 7 quadrature points})

\[ \text{. predict } S_{\text{cond}}, \text{ surv conditional} \]
(\text{predictions based on fixed effects and posterior means of random effects})
(\text{using 7 quadrature points})

\[ \text{. sort female age patient } _{t} \]
\[ \text{. list patient female age } _{t} S_{*} \text{ in 15/20, sepby(patient)} \]

<table>
<thead>
<tr>
<th>patient</th>
<th>female</th>
<th>age</th>
<th>_t</th>
<th>S_marg</th>
<th>S_cond</th>
</tr>
</thead>
<tbody>
<tr>
<td>15.</td>
<td>29</td>
<td>0</td>
<td>53</td>
<td>2.9628681</td>
<td>0.9564018</td>
</tr>
<tr>
<td>16.</td>
<td>29</td>
<td>0</td>
<td>53</td>
<td>2.516502</td>
<td>0.3493612</td>
</tr>
<tr>
<td>17.</td>
<td>16</td>
<td>0</td>
<td>60</td>
<td>2.912224</td>
<td>0.9230725</td>
</tr>
<tr>
<td>18.</td>
<td>16</td>
<td>0</td>
<td>60</td>
<td>2.62736</td>
<td>0.6129258</td>
</tr>
<tr>
<td>19.</td>
<td>38</td>
<td>0</td>
<td>60</td>
<td>2.8141541</td>
<td>0.9107043</td>
</tr>
<tr>
<td>20.</td>
<td>38</td>
<td>0</td>
<td>60</td>
<td>2.2048696</td>
<td>0.2900459</td>
</tr>
</tbody>
</table>

Survival predictions vary with the value of the study time variable because they are predictions of the survivor function at the study time \(_t\). For example, patient 29 has a 0.96 probability that a new insertion remains at least 2 days without infection and a 0.35 probability that a new insertion remains at least 25 days without infection. For a random patient from the population, the probabilities to remain at least 2 or 25 days without infection are, respectively, 0.96 and 0.52.

We can use \text{stcurve} to plot these predictions simultaneously for males and females of the same age.

\[ \text{. stcurve, surv at1(female=0 age=53) at2(female=1 age=53)} \]
(option unconditional assumed)

We see that the survivor function for females is above the survivor function for males, which means that females have a greater probability of not having an episode by study time \(_t\).
Example 3

In example 2 of [ME] mestreg, we estimated two different distributions with random effects on patient and covariates age and female. Here we compare the marginal hazards using stcurve. By default, stcurve plots predictions at the mean of the covariates, computed over the whole estimation sample. We plot the predictions for female==1.

. mestreg age female || patient:, dist(weibull) time
  (output omitted)
. stcurve, hazard at(female=1)
  (option unconditional assumed)
. graph save g1
  (file g1.gph saved)
. mestreg age female || patient:, dist(gamma)
  (output omitted)
. stcurve, hazard at(female=1)
  (option unconditional assumed)
. graph save g2
  (file g2.gph saved)
. graph combine g1.gph g2.gph

The two estimated marginal hazards are similar. The marginal hazard has a very different shape from the conditional hazards. The conditional hazard function for a Weibull or a gamma distribution are both monotonic (increasing, constant, or decreasing, depending on the parameters).
Example 4

In example 3 of [ME] `mestreg`, we fit a Weibull model with random intercepts and random coefficients at the subject level. We obtained a positive covariance between the random effects. We refit the model here and then use `predict` with the option `reffects` to obtain predictions of the random effects based on the empirical Bayes posterior means.

```
. use http://www.stata-press.com/data/r14/angina, clear
   (Angina drug data, Rabe-Hesketh and Skrondal, 2012, ch 15.7)
. mestreg occasion##treat || pid: treat, distribution(weibull)
   > covariance(unstructured)
   (output omitted)
. predict re*, reffects
   (calculating posterior means of random effects)
   (using 7 quadrature points)
```

Plotting the predictions of the predicted random coefficient versus the random intercept shows the pattern we discussed in the main section: individuals with a larger random slope tend also to have a larger random intercept.

```
. twoway scatter re1 re2
```

Individuals with large random intercepts have individual hazards that are larger than those of other individuals with the same covariate patterns. Also, individuals with large random coefficients have individual conditional hazard ratios for treatment that are larger than those of other individuals with the same covariate pattern.

In other words, if the aim of the treatment is to decrease the hazard, then the positive correlation means that the treatment tends to be less effective for individuals who have a higher individual hazard (within the same occasion number).

Example 5

In example 1 of [ME] `mestreg`, we mentioned that hazard ratios should be interpreted as conditional on the random effects. Here we use `predict` to illustrate this concept. We use a simulated dataset for a Weibull model with random effects for `group` and a binary covariate `x`.
We show that for a given group, the conditional hazard function satisfies the proportional-hazards (PH) assumption. That is, for a given group $j$,

$$h(t|x=1, \text{group} = j) = \exp(\beta_x) \times h(t|x=0, \text{group} = j)$$

is equivalent to

$$\log\{h(t|x=1, \text{group} = j)\} = \beta_x + \log\{h(t|x=0, \text{group} = j)\}$$

This property of the log hazard-function translates to one curve being a shifted version of the other, which is easier to see than the proportionality of the (untransformed) hazard function.

After fitting the model, we use `predict` to compute the conditional prediction of the hazard function for group 1; we create the variables `hcond0` and `hcond1`. `hcond0` will contain the conditional hazard for group 1 when $x==0$; `hcond1` will contain the conditional hazard for group 1 when $x==1$.

We also create $zcond = \log(hcond0) + \beta_x$. If the PH assumption is satisfied, then the plotted values of $zcond$ will be superimposed on those of $\log(hcond1)$.

```stata
use http://www.stata-press.com/data/r14/weibre, clear
mestreg i.x || group:, distribution(weibull) nolog
    failure _d: 1 (meaning all fail)
analysis time _t: t
Mixed-effects Weibull regression
Group variable: group  Number of obs = 100,000
Number of groups = 500
Obs per group:  min = 200
                avg = 200.0
                max = 200
Integration method: mvaghermite
Integration pts. = 7
Log likelihood = -228103.73  Wald chi2(1) = 21447.86
Prob > chi2 = 0.0000

|       | Haz. Ratio | Std. Err. | z    | P>|z|  | [95% Conf. Interval] |
|-------|------------|-----------|------|------|----------------------|
| _t    |            |           |      |      |                      |
| 1.x   | 2.713137   | .0184908  | 146.45 | 0.000 | 2.677137             | 2.749622 |
| _cons | 2.564135   | .0797383  | 30.28 | 0.000 | 2.412518             | 2.72528  |
| /ln_p | -.6925791  | .0024746  | -279.88 | 0.000 | -.6974291            | -.687729 |

group var(_cons)  .4728021  .0303093    .4169774  .5361006

LR test vs. Weibull model: chibar2(01) = 35800.39  Prob >= chibar2 = 0.0000
```

. predict hcond, hazard conditional(ebmeans)
(predictions based on fixed effects and posterior means of random effects)

. gen loghcond0 = log(hcond) if x==0
(49,991 missing values generated)

. gen loghcond1 = log(hcond) if x==1
(50,009 missing values generated)

. gen zcond = loghcond0 + _b[_t:1.x]
(49,991 missing values generated)

. sort _t group
In the graph above, the line for \texttt{loghcond1} cannot be distinguished from the line for \texttt{zcond} for most of the distribution. This illustrates that the PH assumption is satisfied for the conditional hazard. Notice that you can still see a part of \texttt{loghcond1} near the origin. This is because the two variables correspond to different values of _\texttt{t} and only \texttt{loghcond1} happens to be defined at the early values.

Now, we make the same computation for the marginal hazard.

```
. predict hmarg, hazard marginal
. gen loghmarg0 = log(hmarg) if x==0
   (49,991 missing values generated)
. gen loghmarg1 = log(hmarg) if x==1
   (50,009 missing values generated)
. gen zmarg = loghmarg0 + _b[_t:1.x]
   (49,991 missing values generated)
. sort _t group
. twoway line loghmarg0 loghmarg1 zmarg _t
```

The curve for zmarg is clearly different from the curve for loghmarg1, demonstrating that the marginal distribution does not meet the PH assumption. Notice that the line for loghmarg1 is shorter than the others. This is because predictions are obtained at the values of \( t \) in the dataset. These values of \( t \) were simulated based on the model, which determines that observations with \( x=1 \) fail earlier.

### Methods and formulas

Methods and formulas for predicting random effects and other statistics are given in Methods and formulas of [ME] mestreg. Statistics of special interest for survival analysis are described below.

predict newvar with the conditional() option computes the following predictions:

**median:**

newvar\( _{ji} \) = \( \{ t : \hat{S}(t|x_{ji}, \hat{u}_{ji}) = 1/2 \} \)

where \( \hat{S}(t|x_{ji}, \hat{u}_{ji}) \) is \( S(t|x_{ji}\hat{\beta} + \hat{u}_{ji}) \), where \( \hat{u}_{ji} \) are the empirical Bayes predictions for \( u_{ji} \). If conditional(fixedonly) is specified, then 0 is substituted for \( \hat{u}_{ji} \).

**mean:**

\[
newvar_{ji} = \int_{0}^{\infty} \hat{S}(t|x_{ji}, u_{ji}) dt
\]

**surv:**

\[
newvar_{ji} = \hat{S}(t_{ji}|x_{ji}, \hat{u}_{ji})
\]

**hazard:**

\[
newvar_{ji} = \hat{g}(t_{ji}|x_{ji}, \hat{u}_{ji}) / \hat{S}(t_{ji}|x_{ji}, \hat{u}_{ji})
\]

where \( \hat{g}(t|x_{ji}, u_{ji}) \) is the density \( g(t|x_{ji}\hat{\beta} + \hat{u}_{ji}) \).

When the marginal option is used with mean or surv, the prediction is computed marginally with respect to the random effects. That is, the prediction is integrated over the random-effects distributions. When the marginal option is used with hazard, the hazard for the marginal distribution is computed. That is, the predicted hazard is computed as the quotient of the marginal hazard and the marginal survivor function.

### Also see

[ME] mestreg — Multilevel mixed-effects parametric survival models
[ME] meglm postestimation — Postestimation tools for meglm
[ME] mixed postestimation — Postestimation tools for mixed
[ST] stcurve — Plot survivor, hazard, cumulative hazard, or cumulative incidence function
[U] 20 Estimation and postestimation commands
mixed — Multilevel mixed-effects linear regression

Description

mixed fits linear mixed-effects models. These models are also known as multilevel models or hierarchical linear models. The overall error distribution of the linear mixed-effects model is assumed to be Gaussian, and heteroskedasticity and correlations within lowest-level groups also may be modeled.

Quick start

Linear mixed-effects model of y on x with random intercepts by lev2
   mixed y x || lev2:

As above, but perform restricted maximum-likelihood (REML) estimation instead of the default maximum likelihood (ML) estimation
   mixed y x || lev2:, reml

As above, but perform small-sample inference on x using the Kenward–Roger degrees of freedom (DF) method
   mixed y x || lev2:, reml dfmethod(kroger)

Add random coefficients on x
   mixed y x || lev2: x

As above, but allow correlation between the random slopes and intercepts
   mixed y x || lev2: x, covariance(unstructured)

Three-level model with random intercepts by lev2 and lev3 for lev2 nested within lev3
   mixed y x || lev3: || lev2:

Crossed-effects model with two-way crossed effects by factors a and b
   mixed y x || _all:R.a || b:

Menu

Statistics > Multilevel mixed-effects models > Linear regression
Syntax

\[ \text{mixed} \ depvar \ fe\_equation \ [ \| \ re\_equation] \ [ \| \ re\_equation \ldots] \ [ , \ options] \]

where the syntax of \textit{fe\_equation} is

\[ \text{indepvars} \ [ \ if \] \ [ in] \ [ weight] \ [ , \ fe\_options] \]

and the syntax of \textit{re\_equation} is one of the following:

for random coefficients and intercepts

\[ \text{levelvar:} \ [ \text{varlist} \ [ , \ re\_options] \]

for random effects among the values of a factor variable

\[ \text{levelvar: R.\textit{varname} [ , re\_options]} \]

\textit{levelvar} is a variable identifying the group structure for the random effects at that level or is _all_ representing one group comprising all observations.

<table>
<thead>
<tr>
<th>\textit{fe_options}</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>\textit{noconstant}</td>
<td>suppress constant term from the fixed-effects equation</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>\textit{re_options}</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>\textit{covariance(vartype)}</td>
<td>variance–covariance structure of the random effects</td>
</tr>
<tr>
<td>\textit{noconstant}</td>
<td>suppress constant term from the random-effects equation</td>
</tr>
<tr>
<td>\textit{collinear}</td>
<td>keep collinear variables</td>
</tr>
<tr>
<td>\textit{fweight(exp)}</td>
<td>frequency weights at higher levels</td>
</tr>
<tr>
<td>\textit{pweight(exp)}</td>
<td>sampling weights at higher levels</td>
</tr>
</tbody>
</table>
### options

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Model</strong></td>
</tr>
<tr>
<td><code>mle</code> fit model via maximum likelihood; the default</td>
</tr>
<tr>
<td><code>reml</code> fit model via restricted maximum likelihood</td>
</tr>
<tr>
<td><code>dfmethod(df_method)</code> specify method for computing DF of a t distribution</td>
</tr>
<tr>
<td><code>pwscale(scale_method)</code> control scaling of sampling weights in two-level models</td>
</tr>
<tr>
<td><code>residuals(rspec)</code> structure of residual errors</td>
</tr>
<tr>
<td><strong>SE/Robust</strong></td>
</tr>
<tr>
<td><code>vce(vcetype)</code> vcetype may be oim, robust, or cluster clustvar; types other than oim may not be combined with dfmethod()</td>
</tr>
<tr>
<td><strong>Reporting</strong></td>
</tr>
<tr>
<td><code>level(#)</code> set confidence level; default is level(95)</td>
</tr>
<tr>
<td><code>variance</code> show random-effects and residual-error parameter estimates as variances and covariances; the default</td>
</tr>
<tr>
<td><code>stddeviations</code> show random-effects and residual-error parameter estimates as standard deviations</td>
</tr>
<tr>
<td><code>dftable(dftable)</code> specify contents of fixed-effects table; requires dfmethod() at estimation</td>
</tr>
<tr>
<td><code>noret</code> suppress random-effects table</td>
</tr>
<tr>
<td><code>nofetable</code> suppress fixed-effects table</td>
</tr>
<tr>
<td><code>estmetric</code> show parameter estimates in the estimation metric</td>
</tr>
<tr>
<td><code>noheader</code> suppress output header</td>
</tr>
<tr>
<td><code>nogroup</code> suppress table summarizing groups</td>
</tr>
<tr>
<td><code>nostderr</code> do not estimate standard errors of random-effects parameters</td>
</tr>
<tr>
<td><code>nolrtest</code> do not perform likelihood-ratio test comparing with linear regression</td>
</tr>
<tr>
<td><code>display_options</code> control columns and column formats, row spacing, line width, display of omitted variables and base and empty cells, and factor-variable labeling</td>
</tr>
<tr>
<td><strong>EM options</strong></td>
</tr>
<tr>
<td><code>emiterate(#)</code> number of EM iterations; default is emiterate(20)</td>
</tr>
<tr>
<td><code>emtolerance(#)</code> EM convergence tolerance; default is emtolerance(1e-10)</td>
</tr>
<tr>
<td><code>emonly</code> fit model exclusively using EM</td>
</tr>
<tr>
<td><code>emlog</code> show EM iteration log</td>
</tr>
<tr>
<td><code>emdots</code> show EM iterations as dots</td>
</tr>
<tr>
<td><strong>Maximization</strong></td>
</tr>
<tr>
<td><code>maximize_options</code> control the maximization process; seldom used</td>
</tr>
<tr>
<td><code>matsqrt</code> parameterize variance components using matrix square roots; the default</td>
</tr>
<tr>
<td><code>matlog</code> parameterize variance components using matrix logarithms</td>
</tr>
<tr>
<td><code>small</code> replay small-sample inference results</td>
</tr>
<tr>
<td><code>coeflegend</code> display legend instead of statistics</td>
</tr>
</tbody>
</table>
### vartype

<table>
<thead>
<tr>
<th>vartype</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>independent</td>
<td>one unique variance parameter per random effect, all covariances 0; the default unless the R. notation is used</td>
</tr>
<tr>
<td>exchangeable</td>
<td>equal variances for random effects, and one common pairwise covariance</td>
</tr>
<tr>
<td>identity</td>
<td>equal variances for random effects, all covariances 0; the default if the R. notation is used</td>
</tr>
<tr>
<td>unstructured</td>
<td>all variances and covariances to be distinctly estimated</td>
</tr>
</tbody>
</table>

### df_method

<table>
<thead>
<tr>
<th>df_method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>residual</td>
<td>residual degrees of freedom, ( n - \text{rank}(X) )</td>
</tr>
<tr>
<td>repeated</td>
<td>repeated-measures ANOVA</td>
</tr>
<tr>
<td>anova</td>
<td>ANOVA</td>
</tr>
<tr>
<td>satterthwaite</td>
<td>generalized Satterthwaite approximation; REML estimation only</td>
</tr>
<tr>
<td>kroger</td>
<td>Kenward–Roger; REML estimation only</td>
</tr>
</tbody>
</table>

### df_table

<table>
<thead>
<tr>
<th>df_table</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>default</td>
<td>test statistics, ( p )-values, and confidence intervals; the default</td>
</tr>
<tr>
<td>ci</td>
<td>DFs and confidence intervals</td>
</tr>
<tr>
<td>pvalue</td>
<td>DFs, test statistics, and ( p )-values</td>
</tr>
</tbody>
</table>

**Options**

* indepvars may contain factor variables; see [U] 11.4.3 Factor variables.  
* depvar, indepvars, and varlist may contain time-series operators; see [U] 11.4.4 Time-series varlists.  
* bootstrap, by, jackknife, mi estimate, rolling, and statsby are allowed; see [U] 11.1.10 Prefix commands.  
* mi estimate is not allowed if dfmethod() is specified.  
* Weights are not allowed with the bootstrap prefix; see [R] bootstrap.  
* pweights and fweights are allowed; see [U] 11.1.6 weight. However, no weights are allowed if either option reml or option dfmethod() is specified.  
* small and coeflegend do not appear in the dialog box.  
* See [U] 20 Estimation and postestimation commands for more capabilities of estimation commands.

---

**noconstant** suppresses the constant (intercept) term and may be specified for the fixed-effects equation and for any of or all the random-effects equations.

**covariance(vartype)** specifies the structure of the covariance matrix for the random effects and may be specified for each random-effects equation. \( \text{vartype} \) is one of the following: independent, exchangeable, identity, or unstructured.

**independent** allows for a distinct variance for each random effect within a random-effects equation and assumes that all covariances are 0.
mixed — Multilevel mixed-effects linear regression

mixed — Multilevel mixed-effects linear regression

exchangeable specifies one common variance for all random effects and one common pairwise covariance.

identity is short for “multiple of the identity”; that is, all variances are equal and all covariances are 0.

unstructured allows for all variances and covariances to be distinct. If an equation consists of \( p \) random-effects terms, the unstructured covariance matrix will have \( p(p+1)/2 \) unique parameters.

covariance(independent) is the default, except when the R. notation is used, in which case covariance(identity) is the default and only covariance(identity) and covariance(exchangeable) are allowed.

collinear specifies that mixed not omit collinear variables from the random-effects equation. Usually, there is no reason to leave collinear variables in place; in fact, doing so usually causes the estimation to fail because of the matrix singularity caused by the collinearity. However, with certain models (for example, a random-effects model with a full set of contrasts), the variables may be collinear, yet the model is fully identified because of restrictions on the random-effects covariance structure. In such cases, using the collinear option allows the estimation to take place with the random-effects equation intact.

fweight(\( \exp \)) specifies frequency weights at higher levels in a multilevel model, whereas frequency weights at the first level (the observation level) are specified in the usual manner, for example, [fw=fwtvar1]. \( \exp \) can be any valid Stata variable, and you can specify fweight() at levels two and higher of a multilevel model. For example, in the two-level model

```
.mixed fixed_portion [fw = wt1] || school: ..., fweight(wt2) ...
```

the variable wt1 would hold the first-level (the observation-level) frequency weights, and wt2 would hold the second-level (the school-level) frequency weights.

pweight(\( \exp \)) specifies sampling weights at higher levels in a multilevel model, whereas sampling weights at the first level (the observation level) are specified in the usual manner, for example, [pw=pwtvar1]. \( \exp \) can be any valid Stata variable, and you can specify pweight() at levels two and higher of a multilevel model. For example, in the two-level model

```
.mixed fixed_portion [pw = wt1] || school: ..., pweight(wt2) ...
```

variable wt1 would hold the first-level (the observation-level) sampling weights, and wt2 would hold the second-level (the school-level) sampling weights.

See Survey data in Remarks and examples below for more information regarding the use of sampling weights in multilevel models.

mle and reml specify the statistical method for fitting the model.

mle, the default, specifies that the model be fit using ML. Options dfmethod(satterthwaite) and dfmethod(kroger) are not supported under ML estimation.

reml specifies that the model be fit using REML, also known as residual maximum likelihood.

dfmethod(\( df\_method \)) requests that reported hypothesis tests for the fixed effects (coefficients) use a small-sample adjustment. By default, inference is based on a large-sample approximation of the sampling distributions of the test statistics by normal and \( \chi^2 \) distributions. Caution should be exercised when choosing a DF method; see Small-sample inference for fixed effects in Remarks and examples for details.
When \texttt{dfmethod(df\_method)} is specified, the sampling distributions of the test statistics are approximated by a \( t \) distribution, according to the requested method for computing the DF. \texttt{df\_method} is one of the following: \texttt{residual}, \texttt{repeated}, \texttt{anova}, \texttt{satterthwaite}, or \texttt{kroger}.

\texttt{residual} uses the residual degrees of freedom, \( n - \text{rank}(X) \), as the DF for all tests of fixed effects. For a linear model without random effects with independent and identically distributed (i.i.d.) errors, the distributions of the test statistics for fixed effects are \( t \) distributions with the residual DF. For other mixed-effects models, this method typically leads to poor approximations of the actual sampling distributions of the test statistics.

\texttt{repeated} uses the repeated-measures ANOVA method for computing the DF. It is used with balanced repeated-measures designs with spherical correlation error structures. It partitions the residual degrees of freedom into the between-subject degrees of freedom and the within-subject degrees of freedom. \texttt{repeated} is supported only with two-level models. For more complex mixed-effects models or with unbalanced data, this method typically leads to poor approximations of the actual sampling distributions of the test statistics.

\texttt{anova} uses the traditional ANOVA method for computing the DF. According to this method, the DF for a test of a fixed effect of a given variable depends on whether that variable is also included in any of the random-effects equations. For traditional ANOVA models with balanced designs, this method provides exact sampling distributions of the test statistics. For more complex mixed-effects models or with unbalanced data, this method typically leads to poor approximations of the actual sampling distributions of the test statistics.

\texttt{satterthwaite}, \texttt{dfopts} implements a generalization of the Satterthwaite (1946) approximation of the unknown sampling distributions of test statistics for complex linear mixed-effect models. This method is supported only with REML estimation.

\texttt{kroger}, \texttt{dfopts} implements the Kenward and Roger (1997) method, which is designed to approximate unknown sampling distributions of test statistics for complex linear mixed-effects models. This method is supported only with REML estimation.

\texttt{dfopts} is either \texttt{eim} or \texttt{oim}.

\texttt{eim} specifies that the expected information matrix be used to compute Satterthwaite or Kenward–Roger degrees of freedom. This is the default.

\texttt{oim} specifies that the observed information matrix be used to compute Satterthwaite or Kenward–Roger degrees of freedom.

Residual, repeated, and ANOVA methods are suitable only when the sampling distributions of the test statistics are known to be \( t \) or \( F \). This is usually only known for certain classes of linear mixed-effects models with simple covariance structures and when data are balanced. These methods are available with both ML and REML estimation.

For unbalanced data or balanced data with complicated covariance structures, the sampling distributions of the test statistics are unknown and can only be approximated. The Satterthwaite and Kenward–Roger methods provide approximations to the distributions in these cases. According to Schaalje, McBride, and Fellingham (2002), the Kenward–Roger method should, in general, be preferred to the Satterthwaite method. However, there are situations in which the two methods are expected to perform similarly, such as with compound symmetry covariance structures. The Kenward–Roger method is more computationally demanding than the Satterthwaite method. Both methods are available only with REML estimation. See \textit{Small-sample inference for fixed effects} in Remarks and examples for examples and more detailed descriptions of the DF methods.

\texttt{dfmethod()} may not be combined with weighted estimation, the \texttt{mi estimate} prefix, or \texttt{vce()}, unless it is the default \texttt{vce(oim)}. 
**mixed** — Multilevel mixed-effects linear regression

`pwscale(scale_method)` controls how sampling weights (if specified) are scaled in two-level models. `scale_method` is one of the following: `size`, `effective`, or `gk`.

- **size** specifies that first-level (observation-level) weights be scaled so that they sum to the sample size of their corresponding second-level cluster. Second-level sampling weights are left unchanged.
- **effective** specifies that first-level weights be scaled so that they sum to the effective sample size of their corresponding second-level cluster. Second-level sampling weights are left unchanged.
- **gk** specifies the Graubard and Korn (1996) method. Under this method, second-level weights are set to the cluster averages of the products of the weights at both levels, and first-level weights are then set equal to 1.

`pwscale()` is supported only with two-level models. See Survey data in Remarks and examples below for more details on using `pwscale()`. `pwscale()` may not be combined with the `dfmethod()` option.

**residuals(rspec)** specifies the structure of the residual errors within the lowest-level groups (the second level of a multilevel model with the observations comprising the first level) of the linear mixed model. For example, if you are modeling random effects for classes nested within schools, then `residuals()` refers to the residual variance–covariance structure of the observations within classes, the lowest-level groups. `rspec` has the following syntax:

```
restype [ , residual_options ]
```

- **restype** is one of the following: `independent`, `exchangeable`, `ar #`, `ma #`, `unstructured`, `banded #`, `toeplitz #`, or `exponential`.

  - `independent`, the default, specifies that all residuals be i.i.d. Gaussian with one common variance. When combined with `by(varname)`, independence is still assumed, but you estimate a distinct variance for each level of `varname`. Unlike with the structures described below, `varname` does not need to be constant within groups.

  - `exchangeable` estimates two parameters, one common within-group variance and one common pairwise covariance. When combined with `by(varname)`, these two parameters are distinctly estimated for each level of `varname`. Because you are modeling a within-group covariance, `varname` must be constant within lowest-level groups.

  - `ar #` assumes that within-group errors have an autoregressive (AR) structure of order `#`; `ar 1` is the default. The `t(varname)` option is required, where `varname` is an integer-valued time variable used to order the observations within groups and to determine the lags between successive observations. Any nonconsecutive time values will be treated as gaps. For this structure, `# + 1` parameters are estimated (`#` AR coefficients and one overall error variance). `restype ar` may be combined with `by(varname)`, but `varname` must be constant within groups.

  - `ma #` assumes that within-group errors have a moving-average (MA) structure of order `#`; `ma 1` is the default. The `t(varname)` option is required, where `varname` is an integer-valued time variable used to order the observations within groups and to determine the lags between successive observations. Any nonconsecutive time values will be treated as gaps. For this structure, `# + 1` parameters are estimated (`#` MA coefficients and one overall error variance). `restype ma` may be combined with `by(varname)`, but `varname` must be constant within groups.

  - `unstructured` is the most general structure; it estimates distinct variances for each within-group error and distinct covariances for each within-group error pair. The `t(varname)` option is required, where `varname` is a nonnegative-integer–valued variable that identifies the observations within each group. The groups may be unbalanced in that not all levels of `t()` need to be observed within every group, but you may not have repeated `t()` values within any particular group. When you have `p` levels of `t()`, then `p(p + 1)/2` parameters are estimated.
unstructured may be combined with by(varname), but varname must be constant within groups.

banded # is a special case of unstructured that restricts estimation to the covariances within the first # off-diagonals and sets the covariances outside this band to 0. The t(varname) option is required, where varname is a nonnegative-integer-valued variable that identifies the observations within each group. # is an integer between 0 and p − 1, where p is the number of levels of t(). By default, # = p − 1; that is, all elements of the covariance matrix are estimated. When # is 0, only the diagonal elements of the covariance matrix are estimated. restype banded may be combined with by(varname), but varname must be constant within groups.

toeplitz # assumes that within-group errors have Toeplitz structure of order #, for which correlations are constant with respect to time lags less than or equal to # and are 0 for lags greater than #. The t(varname) option is required, where varname is an integer-valued time variable used to order the observations within groups and to determine the lags between successive observations. # is an integer between 1 and the maximum observed lag (the default). Any nonconsecutive time values will be treated as gaps. For this structure, # + 1 parameters are estimated (# correlations and one overall error variance). restype toeplitz may be combined with by(varname), but varname must be constant within groups.

exponential is a generalization of the AR covariance model that allows for unequally spaced and noninteger time values. The t(varname) option is required, where varname is real-valued. For the exponential covariance model, the correlation between two errors is the parameter ρ, raised to a power equal to the absolute value of the difference between the t() values for those errors. For this structure, two parameters are estimated (the correlation parameter ρ and one overall error variance). restype exponential may be combined with by(varname), but varname must be constant within groups.

residual_options are by(varname) and t(varname).

by(varname) is for use within the residuals() option and specifies that a set of distinct residual-error parameters be estimated for each level of varname. In other words, you use by() to model heteroskedasticity.

t(varname) is for use within the residuals() option to specify a time variable for the ar, ma, toeplitz, and exponential structures, or to identify the observations when restype is unstructured or banded.

vce(vcetype) specifies the type of standard error reported, which includes types that are derived from asymptotic theory (oim), that are robust to some kinds of misspecification (robust), and that allow for intragroup correlation (cluster clustvar); see [R] vce_option. If vce(robust) is specified, robust variances are clustered at the highest level in the multilevel model.

vce(robust) and vce(cluster clustvar) are not supported with REML estimation. Only vce(oim) is allowed in combination with dfmethod().

level(#) ; see [R] estimation options.

variance, the default, displays the random-effects and residual-error parameter estimates as variances and covariances.

stddeviations displays the random-effects and residual-error parameter estimates as standard deviations and correlations.
dfetable(dftable) specifies the contents of the fixed-effects table for small-sample inference when dfmethod() is used during estimation. dftable is one of the following: default, ci, or pvalue. default displays the default standard fixed-effects table that contains test statistics, p-values, and confidence intervals.

ci displays the fixed-effects table in which the columns containing statistics and p-values are replaced with a column containing coefficient-specific DFs. Confidence intervals are also displayed.

pvalue displays the fixed-effects table that includes a column containing DFs with the standard columns containing test statistics and p-values. Confidence intervals are not displayed.

noret able suppresses the random-effects table from the output.

nofetable suppresses the fixed-effects table from the output.

estmetric displays all parameter estimates in the estimation metric. Fixed-effects estimates are unchanged from those normally displayed, but random-effects parameter estimates are displayed as log-standard deviations and hyperbolic arctangents of correlations, with equation names that organize them by model level. Residual-variance parameter estimates are also displayed in their original estimation metric.

noheader suppresses the output header, either at estimation or upon replay.

nogroup suppresses the display of group summary information (number of groups, average group size, minimum, and maximum) from the output header.

nostderr prevents mixed from calculating standard errors for the estimated random-effects parameters, although standard errors are still provided for the fixed-effects parameters. Specifying this option will speed up computation times. nostderr is available only when residuals are modeled as independent with constant variance.

nolrtest prevents mixed from fitting a reference linear regression model and using this model to calculate a likelihood-ratio test comparing the mixed model to ordinary regression. This option may also be specified upon replay to suppress this test from the output.

display_options: noci, nopvalues, noomitted, vsquish, noemptycells, baselevels, allbaselevels, nofvlabel, fvwrap(#), fvwrapon(style), cformat(%,fmt), pformat(%,fmt), sformat(%,fmt), and nolstretch; see [R] estimation options.

**EM options**

These options control the expectation-maximization (EM) iterations that take place before estimation switches to a gradient-based method. When residuals are modeled as independent with constant variance, EM will either converge to the solution or bring parameter estimates close to the solution. For other residual structures or for weighted estimation, EM is used to obtain starting values.

emiterate(#) specifies the number of EM iterations to perform. The default is emiterate(20).

emtolerance(#) specifies the convergence tolerance for the EM algorithm. The default is emtolerance(1e-10). EM iterations will be halted once the log (restricted) likelihood changes by a relative amount less than #. At that point, optimization switches to a gradient-based method, unless emonly is specified, in which case maximization stops.

emonly specifies that the likelihood be maximized exclusively using EM. The advantage of specifying emonly is that EM iterations are typically much faster than those for gradient-based methods. The disadvantages are that EM iterations can be slow to converge (if at all) and that EM provides no facility for estimating standard errors for the random-effects parameters. emonly is available only with unweighted estimation and when residuals are modeled as independent with constant variance.
emlog specifies that the EM iteration log be shown. The EM iteration log is, by default, not displayed unless the emonly option is specified.

emdots specifies that the EM iterations be shown as dots. This option can be convenient because the EM algorithm may require many iterations to converge.

Maximization

\texttt{maximize\_options: difficult, technique(algorithm\_spec), iterate(#), [no]log, trace, gradient, showstep, hessian, showtolerance, tolerance(#), ltolerance(#), nrtolerance(#), and nonrtolerance; see [R] maximize.} Those that require special mention for mixed are listed below.

For the technique() option, the default is technique(nr). The bhhh algorithm may not be specified.

\texttt{matsqrt (the default), during optimization, parameterizes variance components by using the matrix square roots of the variance–covariance matrices formed by these components at each model level.}

\texttt{matlog}, during optimization, parameterizes variance components by using the matrix logarithms of the variance–covariance matrices formed by these components at each model level.

The matsqrt parameterization ensures that variance–covariance matrices are positive semidefinite, while matlog ensures matrices that are positive definite. For most problems, the matrix square root is more stable near the boundary of the parameter space. However, if convergence is problematic, one option may be to try the alternate matlog parameterization. When convergence is not an issue, both parameterizations yield equivalent results.

The following options are available with mixed but are not shown in the dialog box:

\texttt{small replays previously obtained small-sample results. This option is available only upon replay and requires that the dfmethod() option be used during estimation. small is equivalent to dftable(default) upon replay.}

\texttt{coeflegend; see [R] estimation options.}

Remarks and examples

Remarks are presented under the following headings:

Introduction
Two-level models
Covariance structures
Likelihood versus restricted likelihood
Three-level models
Blocked-diagonal covariance structures
Heteroskedastic random effects
Heteroskedastic residual errors
Other residual-error structures
Crossed-effects models
Diagnosing convergence problems
Survey data
Small-sample inference for fixed effects
Introduction

Linear mixed models are models containing both fixed effects and random effects. They are a generalization of linear regression allowing for the inclusion of random deviations (effects) other than those associated with the overall error term. In matrix notation,

$$ y = X\beta + Zu + \epsilon $$

(1)

where $y$ is the $n \times 1$ vector of responses, $X$ is an $n \times p$ design/covariate matrix for the fixed effects $\beta$, and $Z$ is the $n \times q$ design/covariate matrix for the random effects $u$. The $n \times 1$ vector of errors $\epsilon$ is assumed to be multivariate normal with mean 0 and variance matrix $\sigma^2 \epsilon R$.

The fixed portion of (1), $X\beta$, is analogous to the linear predictor from a standard OLS regression model with $\beta$ being the regression coefficients to be estimated. For the random portion of (1), $Zu + \epsilon$, we assume that $u$ has variance–covariance matrix $G$ and that $u$ is orthogonal to $\epsilon$ so that

$$ \text{Var} \left[ \begin{array}{c} u \\ \epsilon \end{array} \right] = \left[ \begin{array}{cc} G & 0 \\ 0 & \sigma^2 \epsilon R \end{array} \right] $$

The random effects $u$ are not directly estimated (although they may be predicted), but instead are characterized by the elements of $G$, known as variance components, that are estimated along with the overall residual variance $\sigma^2 \epsilon$ and the residual-variance parameters that are contained within $R$.

The general forms of the design matrices $X$ and $Z$ allow estimation for a broad class of linear models: blocked designs, split-plot designs, growth curves, multilevel or hierarchical designs, etc. They also allow a flexible method of modeling within-cluster correlation. Subjects within the same cluster can be correlated as a result of a shared random intercept, or through a shared random slope on (say) age, or both. The general specification of $G$ also provides additional flexibility—the random intercept and random slope could themselves be modeled as independent, or correlated, or independent with equal variances, and so forth. The general structure of $R$ also allows for residual errors to be heteroskedastic and correlated, and allows flexibility in exactly how these characteristics can be modeled.

Comprehensive treatments of mixed models are provided by, among others, Searle, Casella, and McCulloch (1992); McCulloch, Searle, and Neuhaus (2008); Verbeke and Molenberghs (2000); Raudenbush and Bryk (2002); Demidenko (2004); and Pinheiro and Bates (2000). In particular, chapter 2 of Searle, Casella, and McCulloch (1992) provides an excellent history.

The key to fitting mixed models lies in estimating the variance components, and for that there exist many methods. Most of the early literature in mixed models dealt with estimating variance components in ANOVA models. For simple models with balanced data, estimating variance components amounts to solving a system of equations obtained by setting expected mean-squares expressions equal to their observed counterparts. Much of the work in extending the ANOVA method to unbalanced data for general ANOVA designs is due to Henderson (1953).

The ANOVA method, however, has its shortcomings. Among these is a lack of uniqueness in that alternative, unbiased estimates of variance components could be derived using other quadratic forms of the data in place of observed mean squares (Searle, Casella, and McCulloch 1992, 38–39). As a result, ANOVA methods gave way to more modern methods, such as minimum norm quadratic unbiased estimation (MINQUE) and minimum variance quadratic unbiased estimation (MIVQUE); see Rao (1973) for MINQUE and LaMotte (1973) for MIVQUE. Both methods involve finding optimal quadratic forms of the data that are unbiased for the variance components.

The most popular methods, however, are ML and REML, and these are the two methods that are supported by mixed. The ML estimates are based on the usual application of likelihood theory, given
the distributional assumptions of the model. The basic idea behind REML (Thompson 1962) is that you can form a set of linear contrasts of the response that do not depend on the fixed effects $\beta$, but instead depend only on the variance components to be estimated. You then apply ML methods by using the distribution of the linear contrasts to form the likelihood.

Returning to (1): in clustered-data situations, it is convenient not to consider all $n$ observations at once but instead to organize the mixed model as a series of $M$ independent groups or clusters

$$y_j = X_j\beta + Z_j u_j + \epsilon_j$$

for $j = 1, \ldots, M$, with cluster $j$ consisting of $n_j$ observations. The response $y_j$ comprises the rows of $y$ corresponding with the $j$th cluster, with $X_j$ and $\epsilon_j$ defined analogously. The random effects $u_j$ can now be thought of as $M$ realizations of a $q \times 1$ vector that is normally distributed with mean 0 and $q \times q$ variance matrix $\Sigma$. The matrix $Z_i$ is the $n_j \times q$ design matrix for the $j$th cluster random effects. Relating this to (1), note that

$$Z = \begin{bmatrix} Z_1 & 0 & \cdots & 0 \\ 0 & Z_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & Z_M \end{bmatrix}; \quad u = \begin{bmatrix} u_1 \\ \vdots \\ u_M \end{bmatrix}; \quad G = I_M \otimes \Sigma; \quad R = I_M \otimes \Lambda \quad (3)$$

The mixed-model formulation (2) is from Laird and Ware (1982) and offers two key advantages. First, it makes specifications of random-effects terms easier. If the clusters are schools, you can simply specify a random effect at the school level, as opposed to thinking of what a school-level random effect would mean when all the data are considered as a whole (if it helps, think Kronecker products). Second, representing a mixed-model with (2) generalizes easily to more than one set of random effects. For example, if classes are nested within schools, then (2) can be generalized to allow random effects at both the school and the class-within-school levels. This we demonstrate later.

In the sections that follow, we assume that residuals are independent with constant variance; that is, in (3) we treat $\Lambda$ equal to the identity matrix and limit ourselves to estimating one overall residual variance, $\sigma^2_\epsilon$. Beginning in Heteroskedastic residual errors, we relax this assumption.

### Two-level models

We begin with a simple application of (2) as a two-level model, because a one-level linear model, by our terminology, is just standard OLS regression.

#### Example 1

Consider a longitudinal dataset, used by both Ruppert, Wand, and Carroll (2003) and Diggle et al. (2002), consisting of weight measurements of 48 pigs on 9 successive weeks. Pigs are identified by the variable `id`. Below is a plot of the growth curves for the first 10 pigs.
It seems clear that each pig experiences a linear trend in growth and that overall weight measurements vary from pig to pig. Because we are not really interested in these particular 48 pigs per se, we instead treat them as a random sample from a larger population and model the between-pig variability as a random effect, or in the terminology of (2), as a random-intercept term at the pig level. We thus wish to fit the model

$$weight_{ij} = \beta_0 + \beta_1 week_{ij} + u_j + \epsilon_{ij}$$

(4)

for \(i = 1, \ldots, 9\) weeks and \(j = 1, \ldots, 48\) pigs. The fixed portion of the model, \(\beta_0 + \beta_1 \text{week}_{ij}\), simply states that we want one overall regression line representing the population average. The random effect \(u_j\) serves to shift this regression line up or down according to each pig. Because the random effects occur at the pig level (id), we fit the model by typing
. mixed weight week || id:
Performing EM optimization:
Performing gradient-based optimization:
Iteration 0:  log likelihood = -1014.9268
Iteration 1:  log likelihood = -1014.9268
Computing standard errors:
Mixed-effects ML regression
Group variable: id

<table>
<thead>
<tr>
<th></th>
<th>Estimate</th>
<th>Std. Err.</th>
<th>[95% Conf. Interval]</th>
</tr>
</thead>
<tbody>
<tr>
<td>weight week</td>
<td>6.209896</td>
<td>.0390124</td>
<td>6.133433 6.286359</td>
</tr>
<tr>
<td>_cons</td>
<td>19.35561</td>
<td>.5974059</td>
<td>18.18472 20.52651</td>
</tr>
</tbody>
</table>

Random-effects Parameters  

<table>
<thead>
<tr>
<th></th>
<th>Estimate</th>
<th>Std. Err.</th>
<th>[95% Conf. Interval]</th>
</tr>
</thead>
<tbody>
<tr>
<td>id: Identity</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>var(_cons)</td>
<td>14.81751</td>
<td>3.124226</td>
<td>9.801716 22.40002</td>
</tr>
<tr>
<td>var(Residual)</td>
<td>4.383264</td>
<td>.3163348</td>
<td>3.805112 5.04926</td>
</tr>
</tbody>
</table>

LR test vs. linear model: chibar2(1) = 472.65 Prob >= chibar2 = 0.0000

Notes:

1. By typing weight week, we specified the response, weight, and the fixed portion of the model in the same way that we would if we were using regress or any other estimation command. Our fixed effects are a coefficient on week and a constant term.

2. When we added || id:, we specified random effects at the level identified by the group variable id, that is, the pig level (level two). Because we wanted only a random intercept, that is all we had to type.

3. The estimation log consists of three parts:
   a. A set of EM iterations used to refine starting values. By default, the iterations themselves are not displayed, but you can display them with the emlog option.
   b. A set of gradient-based iterations. By default, these are Newton–Raphson iterations, but other methods are available by specifying the appropriate maximize options; see [R] maximize.
   c. The message “Computing standard errors”. This is just to inform you that mixed has finished its iterative maximization and is now reparameterizing from a matrix-based parameterization (see Methods and formulas) to the natural metric of variance components and their estimated standard errors.

4. The output title, “Mixed-effects ML regression”, informs us that our model was fit using ML, the default. For REML estimates, use the reml option.

Because this model is a simple random-intercept model fit by ML, it would be equivalent to using xtreg with its mle option.

5. The first estimation table reports the fixed effects. We estimate $\beta_0 = 19.36$ and $\beta_1 = 6.21$. 
6. The second estimation table shows the estimated variance components. The first section of the table is labeled \( \text{id: Identity} \), meaning that these are random effects at the \( \text{id} \) (pig) level and that their variance–covariance matrix is a multiple of the identity matrix; that is, \( \Sigma = \sigma^2_u I \). Because we have only one random effect at this level, mixed knew that \( \text{Identity} \) is the only possible covariance structure. In any case, the variance of the level-two errors, \( \sigma^2_u \), is estimated as 14.82 with standard error 3.12.

7. The row labeled \( \text{var(Residual)} \) displays the estimated variance of the overall error term; that is, \( \hat{\sigma}^2_\epsilon = 4.38 \). This is the variance of the level-one errors, that is, the residuals.

8. Finally, a likelihood-ratio test comparing the model with one-level ordinary linear regression, model (4) without \( u_j \), is provided and is highly significant for these data.

We now store our estimates for later use:

```
. estimates store randint
```

### Example 2

Extending (4) to allow for a random slope on \( \text{week} \) yields the model

\[
\text{weight}_{ij} = \beta_0 + \beta_1 \text{week}_{ij} + u_{0j} + u_{1j} \text{week}_{ij} + \epsilon_{ij}
\]

and we fit this with mixed:

```
. mixed weight week || id: week
Performing EM optimization:
Performing gradient-based optimization:
Iteration 0: log likelihood = -869.03825
Iteration 1: log likelihood = -869.03825
Computing standard errors:
Mixed-effects ML regression
Group variable: id

| weight | Coef. | Std. Err. | z   | P>|z| | [95% Conf. Interval] |
|--------|-------|-----------|-----|-----|---------------------|
| week   | 6.209896 | .0906819 | 68.48 | 0.000 | 6.032163 - 6.387629 |
| _cons  | 19.35561 | .3979159 | 48.64 | 0.000 | 18.57571 - 20.13551 |

Random-effects Parameters
<table>
<thead>
<tr>
<th>id: Independent</th>
<th>Estimate</th>
<th>Std. Err.</th>
<th>[95% Conf. Interval]</th>
</tr>
</thead>
<tbody>
<tr>
<td>var(week)</td>
<td>.3680668</td>
<td>.0801181</td>
<td>.2402389 - .5639103</td>
</tr>
<tr>
<td>var(_cons)</td>
<td>6.756364</td>
<td>1.543503</td>
<td>4.317721 - 10.57235</td>
</tr>
</tbody>
</table>

| var(Residual)   | 1.598811 | .1233988 | 1.374358 - 1.85992 |

LR test vs. linear model: chi2(2) = 764.42 Prob > chi2 = 0.0000
Note: LR test is conservative and provided only for reference.
```

```
. estimates store randslope
```
Because we did not specify a covariance structure for the random effects \((u_{0j}, u_{1j})'\), \texttt{mixed} used the default \texttt{Independent} structure; that is,

\[
\Sigma = \text{Var} \begin{bmatrix} u_{0j} \\ u_{1j} \end{bmatrix} = \begin{bmatrix} \sigma_{u0}^2 & 0 \\ 0 & \sigma_{u1}^2 \end{bmatrix}
\]

(6)

with \(\hat{\sigma}_{u0}^2 = 6.76\) and \(\hat{\sigma}_{u1}^2 = 0.37\). Our point estimates of the fixed effects are essentially identical to those from model (4), but note that this does not hold generally. Given the 95% confidence interval for \(\hat{\sigma}_{u1}^2\), it would seem that the random slope is significant, and we can use \texttt{lrtest} and our two stored estimation results to verify this fact:

\[
\text{. lrtest randslope randint}
\]

Likelihood-ratio test LR chi2(1) = 291.78
(Assumption: randint nested in randslope) Prob > chi2 = 0.0000

Note: The reported degrees of freedom assumes the null hypothesis is not on the boundary of the parameter space. If this is not true, then the reported test is conservative.

The near-zero significance level favors the model that allows for a random pig-specific regression line over the model that allows only for a pig-specific shift.

\textbf{Covariance structures}

In example 2, we fit a model with the default \texttt{Independent} covariance given in (6). Within any random-effects level specification, we can override this default by specifying an alternative covariance structure via the \texttt{covariance()} option.

\textbf{Example 3}

We generalize (6) to allow \(u_{0j}\) and \(u_{1j}\) to be correlated; that is,

\[
\Sigma = \text{Var} \begin{bmatrix} u_{0j} \\ u_{1j} \end{bmatrix} = \begin{bmatrix} \sigma_{u0}^2 & \sigma_{01} \\ \sigma_{01} & \sigma_{u1}^2 \end{bmatrix}
\]
mixed — Multilevel mixed-effects linear regression

Performing EM optimization:
Performing gradient-based optimization:
Iteration 0: log likelihood = -868.96185
Iteration 1: log likelihood = -868.96185
Computing standard errors:
Mixed-effects ML regression
Group variable: id

Number of obs = 432
Number of groups = 48
Obs per group:
min = 9
avg = 9.0
max = 9
Wald chi2(1) = 4649.17
Log likelihood = -868.96185
Prob > chi2 = 0.0000

| weight | Coef.  | Std. Err. | z      | P>|z|  | [95% Conf. Interval] |
|--------|--------|-----------|--------|------|---------------------|
| week   | 6.209896 | 0.0910745 | 68.18  | 0.000 | 6.031393 6.388399   |
| _cons  | 19.35561 | 0.3996387 | 48.43  | 0.000 | 18.57234 20.13889  |

Random-effects Parameters
id: Unstructured

| var(week)   | .3715251 | .0812958 | .2419532 | .570486 |
| var(_cons)  | 6.823363 | 1.566194 | 4.351297 | 10.69986 |
| cov(week,_cons) | -.0984378 | .2545767 | -.5973991 | .4005234 |
| var(Residual)| 1.596829 | .123198 | 1.372735 | 1.857505 |

LR test vs. linear model: chi2(3) = 764.58
Prob > chi2 = 0.0000
Note: LR test is conservative and provided only for reference.

But we do not find the correlation to be at all significant.

. lrtest . randslope
Likelihood-ratio test
(Lassumption: randslope nested in .) LR chi2(1) = 0.15
Prob > chi2 = 0.6959

Instead, we could have also specified covariance(identity), restricting $u_{0j}$ and $u_{1j}$ to not only be independent but also to have common variance, or we could have specified covariance(exchangeable), which imposes a common variance but allows for a nonzero correlation.

Likelihood versus restricted likelihood

Thus far, all our examples have used ML to estimate variance components. We could have just as easily asked for REML estimates. Refitting the model in example 2 by REML, we get
Mixed weight week || id: week, reml

Performing EM optimization:
Performing gradient-based optimization:
Iteration 0: log restricted-likelihood = -870.51473
Iteration 1: log restricted-likelihood = -870.51473

Computing standard errors:

Mixed-effects REML regression Number of obs = 432
Group variable: id Number of groups = 48
Obs per group:
  min = 9
  avg = 9.0
  max = 9

Log restricted-likelihood = -870.51473 Prob > chi2 = 0.0000

| weight  | Coef.  | Std. Err. | z     | P>|z|   | [95% Conf. Interval] |
|---------|--------|-----------|-------|-------|----------------------|
| week    | 6.209896 | .0916387  | 67.77 | 0.000 | 6.030287 6.389504   |
| _cons   | 19.35561 | .4021144  | 48.13 | 0.000 | 18.56748 20.14374   |

Random-effects Parameters Estimate Std. Err. [95% Conf. Interval]

id: Independent
  var(week) .3764405 .0827027 .2447317 .5790317
  var(_cons) 6.917604 1.593247 4.404624 10.86432

  var(Residual) 1.598784 .1234011 1.374328 1.859898

LR test vs. linear model: chi2(2) = 765.92 Prob > chi2 = 0.0000

Note: LR test is conservative and provided only for reference.

Although ML estimators are based on the usual likelihood theory, the idea behind REML is to transform the response into a set of linear contrasts whose distribution is free of the fixed effects $\beta$. The restricted likelihood is then formed by considering the distribution of the linear contrasts. Not only does this make the maximization problem free of $\beta$, it also incorporates the degrees of freedom used to estimate $\beta$ into the estimation of the variance components. This follows because, by necessity, the rank of the linear contrasts must be less than the number of observations.

As a simple example, consider a constant-only regression where $y_i \sim N(\mu, \sigma^2)$ for $i = 1, \ldots, n$. The ML estimate of $\sigma^2$ can be derived theoretically as the $n$-divided sample variance. The REML estimate can be derived by considering the first $n-1$ error contrasts, $y_i - \bar{y}$, whose joint distribution is free of $\mu$. Applying maximum likelihood to this distribution results in an estimate of $\sigma^2$, that is, the $(n-1)$-divided sample variance, which is unbiased for $\sigma^2$.

The unbiasedness property of REML extends to all mixed models when the data are balanced, and thus REML would seem the clear choice in balanced-data problems, although in large samples the difference between ML and REML is negligible. One disadvantage of REML is that likelihood-ratio (LR) tests based on REML are inappropriate for comparing models with different fixed-effects specifications. ML is appropriate for such LR tests and has the advantage of being easy to explain and being the method of choice for other estimators.

Another factor to consider is that ML estimation under mixed is more feature-rich, allowing for weighted estimation and robust variance–covariance matrices, features not supported under REML. In the end, which method to use should be based both on your needs and on personal taste.
Examining the REML output, we find that the estimates of the variance components are slightly larger than the ML estimates. This is typical, because ML estimates, which do not incorporate the degrees of freedom used to estimate the fixed effects, tend to be biased downward.

Three-level models

The clustered-data representation of the mixed model given in (2) can be extended to two nested levels of clustering, creating a three-level model once the observations are considered. Formally,

\[ y_{jk} = X_{jk}\beta + Z_{jk}^{(3)} u_{k}^{(3)} + Z_{jk}^{(2)} u_{jk}^{(2)} + \epsilon_{jk} \]  

(7)

for \( i = 1, \ldots, n_{jk} \) first-level observations nested within \( j = 1, \ldots, M_k \) second-level groups, which are nested within \( k = 1, \ldots, M \) third-level groups. Group \( j, k \) consists of \( n_{jk} \) observations, so \( y_{jk}, X_{jk}, \) and \( \epsilon_{jk} \) each have row dimension \( n_{jk} \). \( Z_{jk}^{(3)} \) is the \( n_{jk} \times q_3 \) design matrix for the third-level random effects \( u_{k}^{(3)} \), and \( Z_{jk}^{(2)} \) is the \( n_{jk} \times q_2 \) design matrix for the second-level random effects \( u_{jk}^{(2)} \). Furthermore, assume that

\[ u_{k}^{(3)} \sim N(0, \Sigma_3); \quad u_{jk}^{(2)} \sim N(0, \Sigma_2); \quad \epsilon_{jk} \sim N(0, \sigma^2 \epsilon I) \]

and that \( u_{k}^{(3)}, u_{jk}^{(2)}, \) and \( \epsilon_{jk} \) are independent.

Fitting a three-level model requires you to specify two random-effects equations: one for level three and then one for level two. The variable list for the first equation represents \( Z_{jk}^{(3)} \) and for the second equation represents \( Z_{jk}^{(2)} \); that is, you specify the levels top to bottom in \texttt{mixed}.

Example 4

Baltagi, Song, and Jung (2001) estimate a Cobb–Douglas production function examining the productivity of public capital in each state’s private output. Originally provided by Munnell (1990), the data were recorded over 1970–1986 for 48 states grouped into nine regions.
. use http://www.stata-press.com/data/r14/productivity
(Public Capital Productivity)
. describe
Contains data from http://www.stata-press.com/data/r14/productivity.dta
obs: 816 Public Capital Productivity
vars: 11 29 Mar 2014 10:57
size: 29,376 (_dta has notes)

storage  display  value
variable name  type format label variable label
state  byte  %9.0g  states 1-48
region  byte  %9.0g  regions 1-9
year  int  %9.0g  years 1970-1986
public  float  %9.0g  public capital stock
hwy  float  %9.0g  log(highway component of public)
water  float  %9.0g  log(water component of public)
other  float  %9.0g  log(bldg/other component of public)
private  float  %9.0g  log(private capital stock)
gsp  float  %9.0g  log(gross state product)
emp  float  %9.0g  log(non-agriculture payrolls)
unemp  float  %9.0g  state unemployment rate

Sorted by:

Because the states are nested within regions, we fit a three-level mixed model with random intercepts at
both the region and the state-within-region levels. That is, we use (7) with both $Z_{jk}^{(3)}$ and $Z_{jk}^{(2)}$ set
to the $n_{jk} \times 1$ column of ones, and $\Sigma_3 = \sigma_3^2$ and $\Sigma_2 = \sigma_2^2$ are both scalars.

. mixed gsp private emp hwy water other unemp || region: || state: (output omitted)
Mixed-effects ML regression
Number of obs = 816

Wald chi2(6) = 18829.06
Log likelihood = 1430.5017  Prob > chi2 = 0.0000

<table>
<thead>
<tr>
<th>Group Variable</th>
<th>No. of Groups</th>
<th>Observations per Group</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Minimum</td>
<td>Average</td>
</tr>
<tr>
<td>region</td>
<td>9</td>
<td>51</td>
</tr>
<tr>
<td>state</td>
<td>48</td>
<td>17</td>
</tr>
</tbody>
</table>

| gsp | Coef. | Std. Err. | z     | P>|z| | [95% Conf. Interval] |
|-----|-------|-----------|-------|------|---------------------|
| gsp | .2671484 | .0212591 | 12.57 | 0.000 | .2254814 .3088154 |
| emp | .754072 | .0261868 | 28.80 | 0.000 | .7027468 .8053973 |
| hwy | .0709767 | .023041 | 3.08 | 0.002 | .0258172 .1161363 |
| water | .0761187 | .0139248 | 5.47 | 0.000 | .0488266 .1034109 |
| other | -.0999955 | .0169366 | -5.90 | 0.000 | -.1331906 -.0668004 |
| unemp | -.0058983 | .0009031 | -6.53 | 0.000 | -.0076684 -.0041282 |
| _cons | 2.128823 | .1543854 | 13.79 | 0.000 | 1.826233 2.431413 |
Random-effects Parameters | Estimate | Std. Err. | [95% Conf. Interval]
--- | --- | --- | ---
region: Identity | var(_cons) | .0014506 | .0012995 | .0002506 | .0083957
state: Identity | var(_cons) | .0062757 | .0014871 | .0039442 | .0099855
 | var(Residual) | .0013461 | .0000689 | .0012176 | .0014882
LR test vs. linear model: chi2(2) = 1154.73 Prob > chi2 = 0.0000
Note: LR test is conservative and provided only for reference.

Notes:

1. Our model now has two random-effects equations, separated by ||. The first is a random intercept (constant only) at the region level (level three), and the second is a random intercept at the state level (level two). The order in which these are specified (from left to right) is significant—mixed assumes that state is nested within region.

2. The information on groups is now displayed as a table, with one row for each grouping. You can suppress this table with the nogroup or the noheader option, which will suppress the rest of the header, as well.

3. The variance-component estimates are now organized and labeled according to level.

   After adjusting for the nested-level error structure, we find that the highway and water components of public capital had significant positive effects on private output, whereas the other public buildings component had a negative effect.

Technical note

In the previous example, the states are coded 1–48 and are nested within nine regions. mixed treated the states as nested within regions, regardless of whether the codes for each state were unique between regions. That is, even if codes for states were duplicated between regions, mixed would have enforced the nesting and produced the same results.

The group information at the top of the mixed output and that produced by the postestimation command estat group (see [ME] mixed postestimation) take the nesting into account. The statistics are thus not necessarily what you would get if you instead tabulated each group variable individually.

Model (7) extends in a straightforward manner to more than three levels, as does the specification of such models in mixed.

Blocked-diagonal covariance structures

Covariance matrices of random effects within an equation can be modeled either as a multiple of the identity matrix, as diagonal (that is, Independent), as exchangeable, or as general symmetric (Unstructured). These may also be combined to produce more complex block-diagonal covariance structures, effectively placing constraints on the variance components.
Example 5

Returning to our productivity data, we now add random coefficients on hwy and unemp at the region level. This only slightly changes the estimates of the fixed effects, so we focus our attention on the variance components:

```
. mixed gsp private emp hwy water other unemp || region: hwy unemp || state:,
    > nolog nogroup nofetable
```

<table>
<thead>
<tr>
<th>Mixed-effects ML regression</th>
<th>Number of obs =  816</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wald chi2(6) =  17137.94</td>
<td></td>
</tr>
<tr>
<td>Prob &gt; chi2 =  0.0000</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Estimate</th>
<th>Std. Err.</th>
<th>[95% Conf. Interval]</th>
</tr>
</thead>
<tbody>
<tr>
<td>region: Independent</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>var(hwy)</td>
<td>.0000209</td>
<td>.0001103</td>
<td>6.71e-10 .6506957</td>
</tr>
<tr>
<td>var(unemp)</td>
<td>.0000238</td>
<td>.0000135</td>
<td>7.84e-06 .0000722</td>
</tr>
<tr>
<td>var(_cons)</td>
<td>.0030349</td>
<td>.0086684</td>
<td>.0000112 .8191291</td>
</tr>
<tr>
<td>state: Identity</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>var(_cons)</td>
<td>.0063658</td>
<td>.0015611</td>
<td>.0039365 .0102943</td>
</tr>
<tr>
<td>var(Residual)</td>
<td>.0012469</td>
<td>.0000643</td>
<td>.001127 .0013795</td>
</tr>
</tbody>
</table>

LR test vs. linear model: chi2(4) = 1189.08  Prob > chi2 = 0.0000

Note: LR test is conservative and provided only for reference.

```
. estimates store prodrce
```

This model is the same as that fit in example 4 except that $Z_{jk}^{(3)}$ is now the $n_{jk} \times 3$ matrix with columns determined by the values of hwy, unemp, and an intercept term (one), in that order, and (because we used the default Independent structure) $\Sigma_3$ is

$$
\Sigma_3 = \begin{pmatrix}
\sigma_a^2 & 0 & 0 \\
0 & \sigma_b^2 & 0 \\
0 & 0 & \sigma_c^2
\end{pmatrix}
$$

The random-effects specification at the state level remains unchanged; that is, $\Sigma_2$ is still treated as the scalar variance of the random intercepts at the state level.

An LR test comparing this model with that from example 4 favors the inclusion of the two random coefficients, a fact we leave to the interested reader to verify.

The estimated variance components, upon examination, reveal that the variances of the random coefficients on hwy and unemp could be treated as equal. That is,

$$
\Sigma_3 = \begin{pmatrix}
\sigma_a^2 & 0 & 0 \\
0 & \sigma_a^2 & 0 \\
0 & 0 & \sigma_c^2
\end{pmatrix}
$$

looks plausible. We can impose this equality constraint by treating $\Sigma_3$ as block diagonal: the first block is a $2 \times 2$ multiple of the identity matrix, that is, $\sigma_a^2 I_2$; the second is a scalar, equivalently, a $1 \times 1$ multiple of the identity.
We construct block-diagonal covariances by repeating level specifications:

```
. mixed gsp private emp hwy water other unemp || region: hwy unemp,
    > cov(identity) || region: || state:, nolog nogroup nofetable
```

Mixed-effects ML regression

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Estimate</th>
<th>Std. Err.</th>
<th>[95% Conf. Interval]</th>
</tr>
</thead>
<tbody>
<tr>
<td>region: Identity</td>
<td>var(hwy unemp)</td>
<td>0.000238</td>
<td>0.000134</td>
</tr>
<tr>
<td>region: Identity</td>
<td>var(_cons)</td>
<td>0.0028191</td>
<td>0.0030429</td>
</tr>
<tr>
<td>state: Identity</td>
<td>var(_cons)</td>
<td>0.006358</td>
<td>0.0015309</td>
</tr>
<tr>
<td>var(Residual)</td>
<td>0.0012469</td>
<td>0.0000643</td>
<td>0.001127</td>
</tr>
</tbody>
</table>

LR test vs. linear model: chi2(3) = 1189.08 Prob > chi2 = 0.0000

Note: LR test is conservative and provided only for reference.

We specified two equations for the `region` level: the first for the random coefficients on `hwy` and `unemp` with covariance set to `Identity` and the second for the random intercept `_cons`, whose covariance defaults to `Identity` because it is of dimension 1. `mixed` labeled the estimate of $\sigma_a^2$ as `var(hwy unemp)` to designate that it is common to the random coefficients on both `hwy` and `unemp`.

An LR test shows that the constrained model fits equally well.

```
. lrtest . prodrc
```

Likelihood-ratio test

<table>
<thead>
<tr>
<th>LR chi2(1)</th>
<th>Prob &gt; chi2</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.00</td>
<td>0.9784</td>
</tr>
</tbody>
</table>

Note: The reported degrees of freedom assumes the null hypothesis is not on the boundary of the parameter space. If this is not true, then the reported test is conservative.

Because the null hypothesis for this test is one of equality ($H_0: \sigma_a^2 = \sigma_b^2$), it is not on the boundary of the parameter space. As such, we can take the reported significance as precise rather than a conservative estimate.

You can repeat level specifications as often as you like, defining successive blocks of a block-diagonal covariance matrix. However, repeated-level equations must be listed consecutively; otherwise, `mixed` will give an error.

**Technical note**

In the previous estimation output, there was no constant term included in the first `region` equation, even though we did not use the `noconstant` option. When you specify repeated-level equations, `mixed` knows not to put constant terms in each equation because such a model would be unidentified. By default, it places the constant in the last repeated-level equation, but you can use `noconstant` creatively to override this.

Linear mixed-effects models can also be fit using `meglm` with the default `gaussian` family. `meglm` provides two more covariance structures through which you can impose constraints on variance components; see [ME] `meglm` for details.
Heteroskedastic random effects

Blocked-diagonal covariance structures and repeated-level specifications of random effects can also be used to model heteroskedasticity among random effects at a given level.

Example 6

Following Rabe-Hesketh and Skrondal (2012, sec. 7.2), we analyze data from Asian children in a British community who were weighed up to four times, roughly between the ages of 6 weeks and 27 months. The dataset is a random sample of data previously analyzed by Goldstein (1986) and Prosser, Rasbash, and Goldstein (1991).

```
. use http://www.stata-press.com/data/r14/childweight
(Weight data on Asian children)
. describe
Contains data from http://www.stata-press.com/data/r14/childweight.dta
    obs: 198    Weight data on Asian children
    vars: 5    23 May 2014 15:12
    size: 3,168  (_dta has notes)

                      storage  display value
variable name    type    format    label      variable label
 id              int     %8.0g    child identifier
 age             float   %8.0g    age in years
 weight          float   %8.0g    weight in Kg
 brthwt           int     %8.0g    Birth weight in g
 girl            float   %9.0g    bg    gender
```

Sorted by: id age

```
. graph twoway (line weight age, connect(ascending)), by(girl)
> xtitle(Age in years) ytitle(Weight in kg)
```

Ignoring gender effects for the moment, we begin with the following model for the $i$th measurement on the $j$th child:

\[
\text{weight}_{ij} = \beta_0 + \beta_1 \text{age}_{ij} + \beta_2 \text{age}_{ij}^2 + u_{j0} + u_{j1} \text{age}_{ij} + \epsilon_{ij}
\]
This models overall mean growth as quadratic in age and allows for two child-specific random
effects: a random intercept $u_{i0}$, which represents each child’s vertical shift from the overall mean ($\beta_0$), and a random age slope $u_{i1}$, which represents each child’s deviation in linear growth rate from the overall mean linear growth rate ($\beta_1$). For simplicity, we do not consider child-specific changes in the quadratic component of growth.

```
mixed weight age c.age#c.age || id: age, nolog
```

Mixed-effects ML regression
Number of obs = 198
Group variable: id
Number of groups = 68

Obs per group:
min = 1
avg = 2.9
max = 5

Wald chi2(2) = 1863.46
Prob > chi2 = 0.0000

Log likelihood = -258.51915

|              | Coef. | Std. Err. | z   | P>|z| | [95% Conf. Interval] |
|--------------|-------|-----------|-----|------|---------------------|
| age          | 7.693701 | .2381076 | 32.31 | .0000 | 7.227019 - 8.160384 |
| c.age#c.age  | -1.654542 | .0874987 | -18.91 | .0000 | -1.826037 -1.483048 |
| _cons        | 3.497628 | .1416914 | 24.68 | .0000 | 3.219918 3.775338   |

Random-effects Parameters
<table>
<thead>
<tr>
<th></th>
<th>Estimate</th>
<th>Std. Err.</th>
<th>[95% Conf. Interval]</th>
</tr>
</thead>
<tbody>
<tr>
<td>id: Independent</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>var(age)</td>
<td>.2987207</td>
<td>.0827569</td>
<td>.1735603 .5141388</td>
</tr>
<tr>
<td>var(_cons)</td>
<td>.5023857</td>
<td>.141263</td>
<td>.2895294 .8717298</td>
</tr>
<tr>
<td>var(Residual)</td>
<td>.3092897</td>
<td>.0474887</td>
<td>.2289133 .417888</td>
</tr>
</tbody>
</table>

LR test vs. linear model: chi2(2) = 114.70  Prob > chi2 = 0.0000

Note: LR test is conservative and provided only for reference.

Because there is no reason to believe that the random effects are uncorrelated, it is always a good idea to first fit a model with the covariance(unstructured) option. We do not include the output for such a model because for these data the correlation between random effects is not significant; however, we did check this before reverting to mixed’s default Independent structure.

Next we introduce gender effects into the fixed portion of the model by including a main gender effect and a gender–age interaction for overall mean growth:
```
mixed weight i.girl i.girl#c.age c.age#c.age || id: age, nolog

Mixed-effects ML regression  Number of obs   =    198
Group variable: id  Number of groups =     68
    Obs per group:
         min  =        1
         avg  =      2.9
         max  =        5
Wald chi2(4)    =   1942.30
Log likelihood = -253.182  Prob > chi2    =   0.0000

| Coef.  | Std. Err. | z     | P>|z|   | [95% Conf. Interval] |
|--------|-----------|-------|-------|-----------------------|
| girl   |           |       |       |                       |
| girl   | -0.510468 | 0.214553 | -2.38 | 0.017  | -.9309835 -.0899516 |
| girl#c.age |         |       |       |                       |
| boy    | 7.806765  | 0.252453 | 30.92 | 0.000  | 7.311956  8.301574  |
| girl   | 7.577296  | 0.253132 | 29.93 | 0.000  | 7.081166  8.073425  |
| c.age#c.age |       |       |       |                       |
|        | -1.654323 | 0.087175 | -18.98 | 0.000  | -1.825183 -1.483463 |
| _cons  | 3.754275  | 0.172640 | 21.75 | 0.000  | 3.415906  4.092644  |

Random-effects Parameters  | Estimate  | Std. Err.  | [95% Conf. Interval] |
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>id: Independent</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>var(age)</td>
<td>0.2772846</td>
<td>0.0769233</td>
<td>0.1609861 0.4775987</td>
</tr>
<tr>
<td>var(_cons)</td>
<td>0.4076892</td>
<td>0.12386</td>
<td>0.2247635 0.7394906</td>
</tr>
<tr>
<td>var(Residual)</td>
<td>0.3131704</td>
<td>0.047684</td>
<td>0.2323672 0.422072</td>
</tr>
</tbody>
</table>

LR test vs. linear model: chi2(2) = 104.39  Prob > chi2 = 0.0000
Note: LR test is conservative and provided only for reference.
```

The main gender effect is significant at the 5% level, but the gender–age interaction is not:
```
.test 0.girl#c.age = 1.girl#c.age

( 1) [weight]0b.girl#c.age - [weight]1.girl#c.age = 0
    chi2(  1) =  1.66
    Prob > chi2 = 0.1978
```

On average, boys are heavier than girls, but their average linear growth rates are not significantly different.

In the above model, we introduced a gender effect on average growth, but we still assumed that the variability in child-specific deviations from this average was the same for boys and girls. To check this assumption, we introduce gender into the random component of the model. Because support for factor-variable notation is limited in specifications of random effects (see Crossed-effects models below), we need to generate the interactions ourselves.
. generate boy = !girl
. generate boyXage = boy*age
. generate girlXage = girl*age
. mixed weight i.girl i.girl#c.age c.age#c.age || id: boy boyXage, noconstant
> || id: girl girlXage, noconstant nolog nofetable

Mixed-effects ML regression
Number of obs = 198
Group variable: id
Number of groups = 68

Obs per group:
min = 1
avg = 2.9
max = 5

Wald chi2(4) = 2358.11
Log likelihood = -248.94752

Random-effects Parameters
<table>
<thead>
<tr>
<th>Estimate</th>
<th>Std. Err.</th>
<th>[95% Conf. Interval]</th>
</tr>
</thead>
<tbody>
<tr>
<td>var(boy)</td>
<td>.3161091</td>
<td>.1557911 .1203181 .8305061</td>
</tr>
<tr>
<td>var(boyXage)</td>
<td>.4734482</td>
<td>.1574626 .2467028 .9085962</td>
</tr>
<tr>
<td>var(girl)</td>
<td>.5798676</td>
<td>.1959725 .2989896 1.124609</td>
</tr>
<tr>
<td>var(girlXage)</td>
<td>.0664634</td>
<td>.0553274 .0130017 .3397538</td>
</tr>
<tr>
<td>var(Residual)</td>
<td>.3078826</td>
<td>.046484 .2290188 .4139037</td>
</tr>
</tbody>
</table>

LR test vs. linear model: chi2(4) = 112.86
Prob > chi2 = 0.0000

In the above, we suppress displaying the fixed portion of the model (the nofetable option) because it does not differ much from that of the previous model.

Our previous model had the random-effects specification

|| id: age

which we have replaced with the dual repeated-level specification

|| id: boy boyXage, noconstant || id: girl girlXage, noconstant

The former models a random intercept and random slope on age, and does so treating all children as a random sample from one population. The latter also specifies a random intercept and random slope on age, but allows for the variability of the random intercepts and slopes to differ between boys and girls. In other words, it allows for heteroskedasticity in random effects due to gender. We use the noconstant option so that we can separate the overall random intercept (automatically provided by the former syntax) into one specific to boys and one specific to girls.

There seems to be a large gender effect in the variability of linear growth rates. We can compare both models with an LR test, recalling that we stored the previous estimation results under the name homoskedastic:

. lrtest homoskedastic heteroskedastic

Likelihood-ratio test LR chi2(2) = 8.47 (Assumption: homoskedastic nested in heteroskedas-c) Prob > chi2 = 0.0145

Note: The reported degrees of freedom assumes the null hypothesis is not on the boundary of the parameter space. If this is not true, then the reported test is conservative.
Because the null hypothesis here is one of equality of variances and not that variances are 0, the above does not test on the boundary; thus we can treat the significance level as precise and not conservative. Either way, the results favor the new model with heteroskedastic random effects.

Heteroskedastic residual errors

Up to this point, we have assumed that the level-one residual errors—the $\epsilon$’s in the stated models—have been i.i.d. Gaussian with variance $\sigma^2_\epsilon$. This is demonstrated in `mixed` output in the random-effects table, where up until now we have estimated a single residual-error variance, labeled as `var(Residual)`.

To relax the assumptions of homoskedasticity or independence of residual errors, use the `residuals()` option.

Example 7

West, Welch, and Gałecki (2015, chap. 7) analyze data studying the effect of ceramic dental veneer placement on gingival (gum) health. Data on 55 teeth located in the maxillary arches of 12 patients were considered.

```
. use http://www.stata-press.com/data/r14/veneer, clear
(Dental veneer data)
. describe
Contains data from http://www.stata-press.com/data/r14/veneer.dta
obs: 110 Dental veneer data
vars: 7 24 May 2014 12:11
size: 1,100 (_dta has notes)

storage  display value
variable name type format label variable label
patient  byte  %8.0g Patient ID
tooth  byte  %8.0g Tooth number with patient
gcf  byte  %8.0g Gingival crevicular fluid (GCF)
age  byte  %8.0g Patient age
base_gcf  byte  %8.0g Baseline GCF
cda  float  %9.0g Average contour difference after veneer placement
followup  byte  %9.0g t Follow-up time: 3 or 6 months

Sorted by:
```

Veneers were placed to match the original contour of the tooth as closely as possible, and researchers were interested in how contour differences (variable `cda`) impacted gingival health. Gingival health was measured as the amount of gingival crevicular fluid (GCF) at each tooth, measured at baseline (variable `base_gcf`) and at two posttreatment follow-ups at 3 and 6 months. The variable `gcf` records GCF at follow-up, and the variable `followup` records the follow-up time.

Because two measurements were taken for each tooth and there exist multiple teeth per patient, we fit a three-level model with the following random effects: a random intercept and random slope on follow-up time at the patient level, and a random intercept at the tooth level. For the $i$th measurement of the $j$th tooth from the $k$th patient, we have

$$ gcf_{ijk} = \beta_0 + \beta_1 followup_{ijk} + \beta_2 base_{gcf}_{ijk} + \beta_3 cda_{ijk} + \beta_4 age_{ijk} + u_{0k} + u_{1k} followup_{ijk} + v_{0jk} + \epsilon_{ijk} $$
which we can fit using mixed:

```
    . mixed gcf followup base_gcf cda age || patient: followup, cov(un) || tooth: ,
    > reml nolog
```

**Mixed-effects REML regression**  \( \text{Number of obs} = 110 \)

<table>
<thead>
<tr>
<th>Group Variable</th>
<th>No. of Groups</th>
<th>Observations per Group</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Minimum</td>
<td>Average</td>
</tr>
<tr>
<td>patient</td>
<td>12</td>
<td>2</td>
</tr>
<tr>
<td>tooth</td>
<td>55</td>
<td>2</td>
</tr>
</tbody>
</table>

Wald chi2(4) = 7.48

Log restricted-likelihood = -420.92761

| gcf            | Coef. | Std. Err. | z     | P>|z| | [95% Conf. Interval] |
|----------------|-------|-----------|-------|-------|----------------------|
| followup       | .3009815 | 1.936863 | 0.16 | 0.877 | -3.4952 4.097163 |
| base_gcf       | -.0183127 | .1433094 | -0.13 | 0.898 | -0.399194 .2625685 |
| cda            | -.329303 | .5292525 | -0.62 | 0.534 | -1.366619 .7080128 |
| age            | -.5773932 | .2139656 | -2.70 | 0.007 | -.9967582 -.1580283 |
| _cons          | 45.73862 | 12.55497 | 3.64 | 0.000 | 21.13133 70.34591 |

**Random-effects Parameters**  \( \text{Estimate} \)  \( \text{Std. Err.} \)  \( \text{[95% Conf. Interval]} \)

<table>
<thead>
<tr>
<th>patient: Unstructured</th>
<th>Estimate</th>
<th>Std. Err.</th>
<th>[95% Conf. Interval]</th>
</tr>
</thead>
<tbody>
<tr>
<td>var(followup)</td>
<td>41.88772</td>
<td>18.79997</td>
<td>17.38009 100.9535</td>
</tr>
<tr>
<td>var(_cons)</td>
<td>524.9851</td>
<td>253.0205</td>
<td>204.1287 1350.175</td>
</tr>
<tr>
<td>cov(followup,_cons)</td>
<td>-140.4229</td>
<td>66.57623</td>
<td>-270.9099 -9.935907</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>tooth: Identity</th>
<th>Estimate</th>
<th>Std. Err.</th>
<th>[95% Conf. Interval]</th>
</tr>
</thead>
<tbody>
<tr>
<td>var(_cons)</td>
<td>47.45738</td>
<td>16.63034</td>
<td>23.8792 94.3165</td>
</tr>
</tbody>
</table>

\( \text{var(Residual)} \)  \( 48.86704 \)  \( 10.50523 \)  \( 32.06479 \)  \( 74.47382 \)

LR test vs. linear model: chi2(4) = 91.12

Prob > chi2 = 0.0000

Note: LR test is conservative and provided only for reference.

We used REML estimation for no other reason than variety.

Among the other features of the model fit, we note that the residual variance \( \sigma^2 \epsilon \) was estimated as 48.87 and that our model assumed that the residuals were independent with constant variance (homoskedastic). Because it may be the case that the precision of gcf measurements could change over time, we modify the above to estimate two distinct error variances: one for the 3-month follow-up and one for the 6-month follow-up.

To fit this model, we add the residuals(independent, by(followup)) option, which maintains independence of residual errors but allows for heteroskedasticity with respect to follow-up time.
. mixed gcf followup base_gcf cda age || patient: followup, cov(un) || tooth:, > residuals(independent, by(followup)) reml nolog

Mixed-effects REML regression  Number of obs  = 110

No. of Observations per Group

<table>
<thead>
<tr>
<th>Group Variable</th>
<th>Groups</th>
<th>Minimum</th>
<th>Average</th>
<th>Maximum</th>
</tr>
</thead>
<tbody>
<tr>
<td>patient</td>
<td>12</td>
<td>2</td>
<td>9.2</td>
<td>12</td>
</tr>
<tr>
<td>tooth</td>
<td>55</td>
<td>2</td>
<td>2.0</td>
<td>2</td>
</tr>
</tbody>
</table>

Wald chi2(4)  = 7.51
Log restricted-likelihood = -420.4576  Prob > chi2  = 0.1113

| gcf     | Coef.  | Std. Err. | z   | P>|z| | [95% Conf. Interval] |
|---------|--------|-----------|-----|------|----------------------|
| followup| .2703944 | 1.933096  | 0.14| 0.889| -3.518405 4.059193  |
| base_gcf| .0062144 | .1419121 | 0.04| 0.965| -.2719283 .284357  |
| cda     | -.2947235 | .5245126 | -0.56| 0.574| -.1322749 .7333023 |
| age     | -.5743755 | .2142249 | -2.68| 0.007| -.9942487 -.1545024 |
| _cons   | 45.15089  | 12.51452  | 3.61| 0.000| 20.62288 69.6789  |

Random-effects Parameters

<table>
<thead>
<tr>
<th>patient: Unstructured</th>
<th>Estimate</th>
<th>Std. Err.</th>
<th>[95% Conf. Interval]</th>
</tr>
</thead>
<tbody>
<tr>
<td>var(followup)</td>
<td>41.75169</td>
<td>18.72989</td>
<td>17.33099 100.583</td>
</tr>
<tr>
<td>var(_cons)</td>
<td>515.2018</td>
<td>251.9661</td>
<td>197.5542 1343.595</td>
</tr>
<tr>
<td>cov(followup,_cons)</td>
<td>-139.0496</td>
<td>66.27806</td>
<td>-268.9522 -9.146946</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>tooth: Identity</th>
<th>Estimate</th>
<th>Std. Err.</th>
<th>[95% Conf. Interval]</th>
</tr>
</thead>
<tbody>
<tr>
<td>var(_cons)</td>
<td>47.35914</td>
<td>16.48931</td>
<td>23.93514 93.70693</td>
</tr>
</tbody>
</table>

Residual: Independent, by followup

<table>
<thead>
<tr>
<th></th>
<th>Estimate</th>
<th>Std. Err.</th>
<th>[95% Conf. Interval]</th>
</tr>
</thead>
<tbody>
<tr>
<td>3 months: var(e)</td>
<td>61.36785</td>
<td>18.38913</td>
<td>34.10946 110.4096</td>
</tr>
<tr>
<td>6 months: var(e)</td>
<td>36.42861</td>
<td>14.97501</td>
<td>16.27542 81.53666</td>
</tr>
</tbody>
</table>

LR test vs. linear model: chi2(5) = 92.06  Prob > chi2 = 0.0000

Note: LR test is conservative and provided only for reference.

Comparison of both models via an LR test reveals the difference in residual variances to be not significant, something we leave to you to verify as an exercise.

The default residual-variance structure is independent, and when specified without by() is equivalent to the default behavior of mixed: estimating one overall residual standard variance for the entire model.

### Other residual-error structures

Besides the default independent residual-error structure, mixed supports four other structures that allow for correlation between residual errors within the lowest-level (smallest or level two) groups. For purposes of notation, in what follows we assume a two-level model, with the obvious extension to higher-level models.
The exchangeable structure assumes one overall variance and one common pairwise covariance; that is,

\[
\text{Var}(\epsilon_j) = \text{Var}
\begin{bmatrix}
\epsilon_{j1} \\
\epsilon_{j2} \\
\vdots \\
\epsilon_{jn_j}
\end{bmatrix}
= 
\begin{bmatrix}
\sigma^2_{\epsilon} & \sigma_1 & \cdots & \sigma_1 \\
\sigma_1 & \sigma^2_{\epsilon} & \cdots & \sigma_1 \\
\vdots & \vdots & \ddots & \vdots \\
\sigma_1 & \sigma_1 & \cdots & \sigma^2_{\epsilon}
\end{bmatrix}
\]

By default, mixed will report estimates of the two parameters as estimates of the common variance \(\sigma^2_{\epsilon}\) and of the covariance \(\sigma_1\). If the \texttt{stddeviations} option is specified, you obtain estimates of \(\sigma_\epsilon\) and the pairwise correlation. When the \texttt{by(varname)} option is also specified, these two parameters are estimated for each level \texttt{varname}.

The \texttt{ar p} structure assumes that the errors have an AR structure of order \(p\). That is,

\[
\epsilon_{ij} = \phi_1 \epsilon_{i-1,j} + \cdots + \phi_p \epsilon_{i-p,j} + u_{ij}
\]

where \(u_{ij}\) are i.i.d. Gaussian with mean 0 and variance \(\sigma^2_u\). mixed reports estimates of \(\phi_1, \ldots, \phi_p\) and the overall error variance \(\sigma^2_\epsilon\), which can be derived from the above expression. The \texttt{t(varname)} option is required, where \texttt{varname} is a time variable used to order the observations within lowest-level groups and to determine any gaps between observations. When the \texttt{by(varname)} option is also specified, the set of \(p + 1\) parameters is estimated for each level of \texttt{varname}. If \(p = 1\), then the estimate of \(\phi_1\) is reported as \texttt{rho}, because in this case it represents the correlation between successive error terms.

The \texttt{ma q} structure assumes that the errors are an MA process of order \(q\). That is,

\[
\epsilon_{ij} = u_{ij} + \theta_1 u_{i-1,j} + \cdots + \theta_q u_{i-q,j}
\]

where \(u_{ij}\) are i.i.d. Gaussian with mean 0 and variance \(\sigma^2_u\). mixed reports estimates of \(\theta_1, \ldots, \theta_q\) and the overall error variance \(\sigma^2_\epsilon\), which can be derived from the above expression. The \texttt{t(varname)} option is required, where \texttt{varname} is a time variable used to order the observations within lowest-level groups and to determine any gaps between observations. When the \texttt{by(varname)} option is also specified, the set of \(q + 1\) parameters is estimated for each level of \texttt{varname}.

The unstructured structure is the most general and estimates unique variances and unique pairwise covariances for all residuals within the lowest-level grouping. Because the data may be unbalanced and the ordering of the observations is arbitrary, the \texttt{t(varname)} option is required, where \texttt{varname} is an identification variable that matches error terms in different groups. If \texttt{varname} has \(n\) distinct levels, then \(n(n+1)/2\) parameters are estimated. Not all \(n\) levels need to be observed within each group, but duplicated levels of \texttt{varname} within a given group are not allowed because they would cause a singularity in the estimated error-variance matrix for that group. When the \texttt{by(varname)} option is also specified, the set of \(n(n+1)/2\) parameters is estimated for each level of \texttt{varname}.

The banded \(q\) structure is a special case of unstructured that confines estimation to within the first \(q\) off-diagonal elements of the residual variance–covariance matrix and sets the covariances outside this band to 0. As is the case with \texttt{unstructured}, the \texttt{t(varname)} option is required, where \texttt{varname} is an identification variable that matches error terms in different groups. However, with banded variance structures, the ordering of the values in \texttt{varname} is significant because it determines which covariances are to be estimated and which are to be set to 0. For example, if \texttt{varname} has \(n = 5\) distinct values \(t = 1, 2, 3, 4, 5\), then a banded variance–covariance structure of order \(q = 2\) would estimate the following:
\[ \text{Var}(\epsilon_j) = \text{Var} \begin{bmatrix} \epsilon_{1j} \\ \epsilon_{2j} \\ \epsilon_{3j} \\ \epsilon_{4j} \\ \epsilon_{5j} \end{bmatrix} = \begin{bmatrix} \sigma_1^2 & \sigma_{12} & \sigma_{13} & 0 & 0 \\ \sigma_{12} & \sigma_2^2 & \sigma_{23} & \sigma_{24} & 0 \\ \sigma_{13} & \sigma_{23} & \sigma_3^2 & \sigma_{34} & \sigma_{35} \\ 0 & \sigma_{24} & \sigma_{34} & \sigma_4^2 & \sigma_{45} \\ 0 & 0 & \sigma_{35} & \sigma_{45} & \sigma_5^2 \end{bmatrix} \]

In other words, you would have an unstructured variance matrix that constrains \( \sigma_{14} = \sigma_{15} = \sigma_{25} = 0 \).

If \( \text{varname} \) has \( n \) distinct levels, then \( (q+1)(2n-q)/2 \) parameters are estimated. Not all \( n \) levels need to be observed within each group, but duplicated levels of \( \text{varname} \) within a given group are not allowed because they would cause a singularity in the estimated error-variance matrix for that group. When the \( \text{by}(\text{varname}) \) option is also specified, the set of parameters is estimated for each level of \( \text{varname} \). If \( q \) is left unspecified, then \( \text{banded} \) is equivalent to \( \text{unstructured} \); that is, all variances and covariances are estimated. When \( q = 0 \), \( \text{Var}(\epsilon_j) \) is treated as diagonal and can thus be used to model uncorrelated yet heteroskedastic residual errors.

The \( \text{toeplitz} \ q \) structure assumes that the residual errors are homoskedastic and that the correlation between two errors is determined by the time lag between the two. That is, \( \text{Var}(\epsilon_{ij}) = \sigma_\epsilon^2 \) and

\[ \text{Corr}(\epsilon_{ij}, \epsilon_{i+k,j}) = \rho_k \]

If the lag \( k \) is less than or equal to \( q \), then the pairwise correlation \( \rho_k \) is estimated; if the lag is greater than \( q \), then \( \rho_k \) is assumed to be 0. If \( q \) is left unspecified, then \( \rho_k \) is estimated for each observed lag \( k \). The \( \text{t}(\text{varname}) \) option is required, where \( \text{varname} \) is a time variable \( t \) used to determine the lags between pairs of residual errors. As such, \( \text{t}() \) must be integer-valued. \( q+1 \) parameters are estimated: one overall variance \( \sigma_\epsilon^2 \) and \( q \) correlations. When the \( \text{by}(\text{varname}) \) option is also specified, the set of \( q+1 \) parameters is estimated for each level of \( \text{varname} \).

The \( \text{exponential} \) structure is a generalization of the AR structure that allows for noninteger and irregularly spaced time lags. That is, \( \text{Var}(\epsilon_{ij}) = \sigma_\epsilon^2 \) and

\[ \text{Corr}(\epsilon_{ij}, \epsilon_{k,j}) = \rho^{|i-k|} \]

for \( 0 \leq \rho \leq 1 \), with \( i \) and \( k \) not required to be integers. The \( \text{t}(\text{varname}) \) option is required, where \( \text{varname} \) is a time variable used to determine \( i \) and \( k \) for each residual-error pair. \( \text{t}() \) is real-valued.

\( \text{mixed} \) reports estimates of \( \sigma_\epsilon^2 \) and \( \rho \). When the \( \text{by}(\text{varname}) \) option is also specified, these two parameters are estimated for each level of \( \text{varname} \).

**Example 8**

Pinheiro and Bates (2000, chap. 5) analyze data from a study of the estrus cycles of mares. Originally analyzed in Pierson and Ginther (1987), the data record the number of ovarian follicles larger than 10mm, daily over a period ranging from three days before ovulation to three days after the subsequent ovulation.
The $stime$ variable is time that has been scaled so that ovulation occurs at scaled times 0 and 1, and the $time$ variable records the time ordering within mares. Because graphical evidence suggests a periodic behavior, the analysis includes the $\sin1$ and $\cos1$ variables, which are sine and cosine transformations of scaled time, respectively.

We consider the following model for the $i$th measurement on the $j$th mare:

$$ \text{follicles}_{ij} = \beta_0 + \beta_1 \sin1_{ij} + \beta_2 \cos1_{ij} + u_j + \epsilon_{ij} $$

The above model incorporates the cyclical nature of the data as affecting the overall average number of follicles and includes mare-specific random effects $u_j$. Because we believe successive measurements within each mare are probably correlated (even after controlling for the periodicity in the average), we also model the within-mare errors as being AR of order 2.
We picked an order of 2 as a guess, but we could have used LR tests of competing AR models to determine the optimal order, because models of smaller order are nested within those of larger order.

Example 9

Fitzmaurice, Laird, and Ware (2011, chap. 7) analyzed data on 37 subjects who participated in an exercise therapy trial.

Subjects (variable id) were placed on either an increased-repetition regimen (program==1) or a program that kept the repetitions constant but increased weight (program==2). Muscle-strength measurements (variable strength) were taken at baseline (day==0) and then every two days over the next twelve days.

Following Fitzmaurice, Laird, and Ware (2011, chap. 7), and to demonstrate fitting residual-error structures to data collected at uneven time points, we confine our analysis to those data collected at baseline and at days 4, 6, 8, and 12. We fit a full two-way factorial model of strength on program and day, with an unstructured residual-error covariance matrix over those repeated measurements taken on the same subject:
mixed — Multilevel mixed-effects linear regression

keep if inlist(day, 0, 4, 6, 8, 12)
(74 observations deleted)
mixed strength i.program##i.day || id:, > noconstant residuals(unstructured, t(day)) nolog

Mixed-effects ML regression
Number of obs = 173
Group variable: id
Number of groups = 37
Obs per group:
   min = 3
   avg = 4.7
   max = 5
Wald chi2(9) = 45.85
Log likelihood = -296.58215
Prob > chi2 = 0.0000

strength Coef. Std. Err. z P>|z| [95% Conf. Interval]
2.program 1.360119 1.003549 1.36 0.175 -.6068016 3.32704
   day
   4 1.125 .3322583 3.39 0.001 .4737858 1.776214
   6 1.360127 .3766894 3.61 0.000 .6220827 2.098425
   8 1.583563 .4905876 3.23 0.001 .6218298 2.545097
   12 1.623576 .5372947 3.02 0.003 .5704977 2.676654
program#day
   2 4 -.169034 .4423472 -0.38 0.702 -1.036019 .6979506
   2 6 .2113012 .4982385 0.42 0.671 -.7652283 1.187831
   2 8 -.1299763 .6524813 -0.20 0.842 -1.408816 1.148864
   2 12 .3212829 .7306782 0.44 0.660 -1.11082 1.753386
   _cons 79.6875 .7560448 105.40 0.000 78.20568 81.16932

Random-effects Parameters

id: (empty)
Residual: Unstructured
   var(e0) 9.14566 2.126233 5.798599 14.42471
   var(e4) 11.87114 2.761187 7.524987 18.72747
   var(e6) 10.06571 2.348835 6.371125 15.90275
   var(e8) 13.22464 3.118885 8.336026 20.98014
   var(e12) 13.16909 3.167316 8.219245 21.09985
   cov(e0,e4) 9.625235 2.331946 5.054705 14.19577
   cov(e0,e6) 8.489042 2.106352 4.360668 12.61742
   cov(e0,e8) 9.280413 2.369524 4.636232 13.92459
   cov(e0,e12) 8.898006 2.348212 4.295594 13.50042
   cov(e4,e6) 10.49184 2.492498 5.606639 15.37705
   cov(e4,e8) 11.89787 2.848714 6.314492 17.48125
   cov(e4,e12) 11.28344 2.804991 5.785767 16.78112
   cov(e6,e8) 11.0507 2.669555 5.632762 16.23863
   cov(e6,e12) 10.5006 2.590246 5.423812 15.57739
   cov(e8,e12) 12.4091 3.010761 6.508121 18.31009

LR test vs. linear model: ch2(14) = 314.67 Prob > ch2 = 0.0000

Note: The reported degrees of freedom assumes the null hypothesis is not on
the boundary of the parameter space. If this is not true, then the
reported test is conservative.

Because we are using the variable id only to group the repeated measurements and not to introduce random effects at the subject level, we use the noconstant option to omit any subject-level effects.
The unstructured covariance matrix is the most general and contains many parameters. In this example, we estimate a distinct residual variance for each day and a distinct covariance for each pair of days.

That there is positive covariance between all pairs of measurements is evident, but what is not as evident is whether the covariances may be more parsimoniously represented. One option would be to explore whether the correlation diminishes as the time gap between strength measurements increases and whether it diminishes systematically. Given the irregularity of the time intervals, an exponential structure would be more appropriate than, say, an AR or MA structure.

```
. estimates store unstructured
. mixed strength i.program##i.day || id:, noconstant
> residuals(exponential, t(day)) nolog nofetable
Mixed-effects ML regression
Group variable: id

Obs per group:
  min =     3
  avg =    4.7
  max =    5

Wald chi2(9) =  36.77
Log likelihood = -307.83324

Random-effects Parameters |   Estimate   Std. Err.      [95% Conf. Interval]
-----------------------------+-----------------------------------------------
id:                          (empty)
Residual: Exponential       +-------------------+
                          rho     .9786462   .0051238     .9659207     .9866854
                          var(e)   11.22349    2.338371     7.460762    16.88389

LR test vs. linear model: chi2(1) = 292.17   Prob > chi2 = 0.0000
Note: The reported degrees of freedom assumes the null hypothesis is not on
the boundary of the parameter space. If this is not true, then the
reported test is conservative.
```

In the above example, we suppressed displaying the main regression parameters because they did not differ much from those of the previous model. While the unstructured model estimated 15 variance–covariance parameters, the exponential model claims to get the job done with just 2, a fact that is not disputed by an LR test comparing the two nested models (at least not at the 0.01 level).

```
. lrtest unstructured
Likelihood-ratio test   LR chi2(13) =  22.50   Prob > chi2 = 0.0481
(Assumption: . nested in unstructured)
Note: The reported degrees of freedom assumes the null hypothesis is not on
the boundary of the parameter space. If this is not true, then the
reported test is conservative.
```
Crossed-effects models

Not all mixed models contain nested levels of random effects.

Example 10

Returning to our longitudinal analysis of pig weights, suppose that instead of (5) we wish to fit

\[ \text{weight}_{ij} = \beta_0 + \beta_1 \text{week}_{ij} + u_i + v_j + \epsilon_{ij} \quad (8) \]

for the \( i = 1, \ldots, 9 \) weeks and \( j = 1, \ldots, 48 \) pigs and

\[ u_i \sim N(0, \sigma_u^2); \quad v_j \sim N(0, \sigma_v^2); \quad \epsilon_{ij} \sim N(0, \sigma^2) \]

all independently. Both (5) and (8) assume an overall population-average growth curve \( \beta_0 + \beta_1 \text{week} \) and a random pig-specific shift.

The models differ in how week enters into the random part of the model. In (5), we assume that the effect due to week is linear and pig specific (a random slope); in (8), we assume that the effect due to week, \( u_i \), is systematic to that week and common to all pigs. The rationale behind (8) could be that, assuming that the pigs were measured contemporaneously, we might be concerned that week-specific random factors such as weather and feeding patterns had significant systematic effects on all pigs.

Model (8) is an example of a two-way crossed-effects model, with the pig effects \( v_j \) being crossed with the week effects \( u_i \). One way to fit such models is to consider all the data as one big cluster, and treat the \( u_i \) and \( v_j \) as a series of \( 9 + 48 = 57 \) random coefficients on indicator variables for week and pig. In the notation of (2),

\[
\mathbf{u} = \begin{bmatrix} u_1 \\ \vdots \\ u_9 \\ v_1 \\ \vdots \\ v_{48} \end{bmatrix} \sim N(0, \mathbf{G}); \quad \mathbf{G} = \begin{bmatrix} \sigma_u^2 \mathbf{I}_9 & 0 \\ 0 & \sigma_v^2 \mathbf{I}_{48} \end{bmatrix}
\]

Because \( \mathbf{G} \) is block diagonal, it can be represented in \texttt{mixed} as repeated-level equations. All we need is an identification variable to identify all the observations as one big group and a way to tell \texttt{mixed} to treat \texttt{week} and \texttt{pig} as factor variables (or equivalently, as two sets of overparameterized indicator variables identifying weeks and pigs, respectively). \texttt{mixed} supports the special group designation \_all for the former and the \texttt{R.varname} notation for the latter.
Thus we estimate \( \hat{\sigma}_u^2 = 0.08 \) and \( \hat{\sigma}_v^2 = 14.84 \). Both (5) and (8) estimate a total of five parameters: two fixed effects and three variance components. The models, however, are not nested within each other, which precludes the use of an LR test to compare both models. Refitting model (5) and looking at the Akaike information criteria values by using \textit{estimates stats},

\begin{verbatim}
. quietly mixed weight week || id:week
. estimates stats crossed
\end{verbatim}

definitely favors model (5). This finding is not surprising given that our rationale behind (8) was somewhat fictitious. In our \textit{estimates stats} output, the values of \texttt{ll(null)} are missing. \textit{mixed} does not fit a constant-only model as part of its usual estimation of the full model, but you can use \textit{mixed} to fit a constant-only model directly, if you wish.

<table>
<thead>
<tr>
<th>Model</th>
<th>Obs</th>
<th>ll(null)</th>
<th>ll(model)</th>
<th>df</th>
<th>AIC</th>
<th>BIC</th>
</tr>
</thead>
<tbody>
<tr>
<td>crossed</td>
<td>432</td>
<td>-1013.824</td>
<td>2037.648</td>
<td>5</td>
<td>2057.99</td>
<td></td>
</tr>
<tr>
<td>.</td>
<td>432</td>
<td>-869.0383</td>
<td>1748.077</td>
<td>5</td>
<td>1768.419</td>
<td></td>
</tr>
</tbody>
</table>

Note: \( N=\text{Obs used in calculating BIC}; \) see [B] BIC note.
mixed — Multilevel mixed-effects linear regression

The \texttt{R.varname} notation is equivalent to giving a list of overparameterized (none dropped) indicator variables for use in a random-effects specification. When you specify \texttt{R.varname}, \texttt{mixed} handles the calculations internally rather than creating the indicators in the data. Because the set of indicators is overparameterized, \texttt{R.varname} implies \texttt{noconstant}. You can include factor variables in the fixed-effects specification by using standard methods; see \cite{[U] 11.4.3 Factor variables}. However, random-effects equations support only the \texttt{R.varname} factor specification. For more complex factor specifications (such as interactions) in random-effects equations, use \texttt{generate} to form the variables manually, as we demonstrated in \textbf{example 6}.

\textbf{Technical note}

Although we were able to fit the crossed-effects model (8), it came at the expense of increasing the column dimension of our random-effects design from 2 in model (5) to 57 in model (8). Computation time and memory requirements grow (roughly) quadratically with the dimension of the random effects. As a result, fitting such crossed-effects models is feasible only when the total column dimension is small to moderate.

Reexamining model (8), we note that if we drop \(u_i\), we end up with a model equivalent to (4), meaning that we could have fit (4) by typing

\begin{verbatim}
   . mixed weight week || _all: R.id
\end{verbatim}

instead of

\begin{verbatim}
   . mixed weight week || id:
\end{verbatim}

as we did when we originally fit the model. The results of both estimations are identical, but the latter specification, organized at the cluster (pig) level with random-effects dimension 1 (a random intercept) is much more computationally efficient. Whereas with the first form we are limited in how many pigs we can analyze, there is no such limitation with the second form.

Furthermore, we fit model (8) by using

\begin{verbatim}
   . mixed weight week || _all: R.week || _all: R.id
\end{verbatim}

as a direct way to demonstrate the \texttt{R.} notation. However, we can technically treat pigs as nested within the _all group, yielding the equivalent and more efficient (total column dimension 10) way to fit (8):

\begin{verbatim}
   . mixed weight week || _all: R.week || id:
\end{verbatim}

We leave it to you to verify that both produce identical results. See Rabe-Hesketh and Skrondal (2012) for additional techniques to make calculations more efficient in more complex models.

\textbf{Example 11}

As another example of how the same model may be fit in different ways by using \texttt{mixed} (and as a way to demonstrate covariance(exchangeable)), consider the three-level model used in \textbf{example 4}:

\[ y_{jk} = X_{jk}\beta + u_{k}^{(3)} + u_{jk}^{(2)} + \epsilon_{jk} \]
where $y_{jk}$ represents the logarithms of gross state products for the $n_{jk} = 17$ observations from state $j$ in region $k$, $X_{jk}$ is a set of regressors, $u^{(3)}_k$ is a random intercept at the region level, and $u^{(2)}_{jk}$ is a random intercept at the state (nested within region) level. We assume that $u^{(3)}_k \sim N(0, \sigma^2_3)$ and $u^{(2)}_{jk} \sim N(0, \sigma^2_2)$ independently. Define

$$v_k = \begin{bmatrix}
    u^{(3)}_k + u^{(2)}_{1k} \\
    (3) \\
    (2) \\
    (3) + u^{(2)}_{2k} \\
    \vdots \\
    (3) + u^{(2)}_{M_k,k}
\end{bmatrix}$$

where $M_k$ is the number of states in region $k$. Making this substitution, we can stack the observations for all the states within region $k$ to get

$$y_k = X_k \beta + Z_k v_k + \epsilon_k$$

where $Z_k$ is a set of indicators identifying the states within each region; that is,

$$Z_k = I_{M_k} \otimes J_{17}$$

for a $k$-column vector of 1s $J_k$, and

$$\Sigma = \text{Var}(v_k) = \begin{bmatrix}
    \sigma^2_3 + \sigma^2_2 & \sigma^2_3 & \cdots & \sigma^2_3 \\
    \sigma^2_3 & \sigma^2_3 + \sigma^2_2 & \cdots & \sigma^2_3 \\
    \vdots & \vdots & \ddots & \vdots \\
    \sigma^2_3 & \sigma^2_3 & \sigma^2_3 & \sigma^2_3 + \sigma^2_2
\end{bmatrix}_{M_k \times M_k}$$

Because $\Sigma$ is an exchangeable matrix, we can fit this alternative form of the model by specifying the exchangeable covariance structure.
mixed — Multilevel mixed-effects linear regression

use http://www.stata-press.com/data/r14/productivity

. mixed gsp private emp hwy water other unemp || region: R.state, cov(exchangeable)
(output omitted)

Mixed-effects ML regression

Number of obs = 816

Group variable: region
  Number of groups = 9

Obs per group:
  min = 51
  avg = 90.7
  max = 136

Wald chi2(6) = 18829.06

Log likelihood = 1430.5017  Prob > chi2 = 0.0000

| Coef.  Std. Err. | z     | P>|z| |  [95% Conf. Interval] |
|----------|-------|-------|---------|-----------------------|
gsp       |       |       |         |                       |
private   | 0.267 | 0.021 | 12.57   | 0.000                 | 0.2254813  | 0.3088154 |
emp       | 0.754 | 0.026 | 28.80   | 0.000                 | 0.7027468  | 0.8053973 |
hwy       | 0.071 | 0.023 | 3.08    | 0.002                 | 0.0258172  | 0.1161363 |
water     | 0.076 | 0.014 | 5.47    | 0.000                 | 0.0488266  | 0.1034109 |
other     | -0.099| 0.017 | -5.90   | 0.000                 | -0.1331907 | -0.0668004|
unemp     | -0.006| 0.001 | -6.53   | 0.000                 | -0.0076684 | -0.0041282|
_cons     | 2.129 | 0.154 | 13.79   | 0.000                 | 1.826233   | 2.431413  |

Random-effects Parameters

<table>
<thead>
<tr>
<th>Estimate</th>
<th>Std. Err.</th>
<th>[95% Conf. Interval]</th>
</tr>
</thead>
</table>
region: Exchangeable
  var(R.state) | 0.0077263  | 0.0017926  | 0.0049032  | 0.0121749 |
  cov(R.state) | 0.0014506  | 0.0012995  | -0.0010963 | 0.0039975 |

LR test vs. linear model: chi2(2) = 1154.73  Prob > chi2 = 0.0000

Note: LR test is conservative and provided only for reference.

The estimates of the fixed effects and their standard errors are equivalent to those from example 4, and remapping the variance components from \( (\sigma_3^2 + \sigma_2^2, \sigma_3^2, \sigma_\epsilon^2) \), as displayed here, to \( (\sigma_3^2, \sigma_2^2, \sigma_\epsilon^2) \), as displayed in example 4, will show that they are equivalent as well.

Of course, given the discussion in the previous technical note, it is more efficient to fit this model as we did originally, as a three-level model.

Diagnosing convergence problems

Given the flexibility of mixed-effects models, you will find that some models fail to converge when used with your data; see Diagnosing convergence problems in [ME] me for advice applicable to mixed-effects models in general.

In unweighted LME models with independent and homoskedastic residuals, one useful way to diagnose problems of nonconvergence is to rely on the EM algorithm (Dempster, Laird, and Rubin 1977), normally used by mixed only as a means of refining starting values. The advantages of EM are that it does not require a Hessian calculation, each successive EM iteration will result in a larger likelihood, iterations can be calculated quickly, and iterations will quickly bring parameter estimates into a
neighborhood of the solution. The disadvantages of EM are that, once in a neighborhood of the solution, it can be slow to converge, if at all, and EM provides no facility for estimating standard errors of the estimated variance components. One useful property of EM is that it is always willing to provide a solution if you allow it to iterate enough times, if you are satisfied with being in a neighborhood of the optimum rather than right on the optimum, and if standard errors of variance components are not crucial to your analysis.

If you encounter a nonconvergent model, try using the `emonly` option to bypass gradient-based optimization. Use `emiterate(#)` to specify the maximum number of EM iterations, which you will usually want to set much higher than the default of 20. If your EM solution shows an estimated variance component that is near 0, a ridge is formed by an interval of values near 0, which produces the same likelihood and looks equally good to the optimizer. In this case, the solution is to drop the offending variance component from the model.

**Survey data**

Multilevel modeling of survey data is a little different from standard modeling in that weighted sampling can take place at multiple levels in the model, resulting in multiple sampling weights. Most survey datasets, regardless of the design, contain one overall inclusion weight for each observation in the data. This weight reflects the inverse of the probability of ultimate selection, and by “ultimate” we mean that it factors in all levels of clustered sampling, corrections for noninclusion and oversampling, poststratification, etc.

For simplicity, in what follows assume a simple two-stage sampling design where groups are randomly sampled and then individuals within groups are sampled. Also assume that no additional weight corrections are performed; that is, sampling weights are simply the inverse of the probability of selection. The sampling weight for observation $i$ in cluster $j$ in our two-level sample is then $w_{ij} = 1/\pi_{ij}$, where $\pi_{ij}$ is the probability that observation $i, j$ is selected. If you were performing a standard analysis such as OLS regression with `regress`, you would simply use a variable holding $w_{ij}$ as your `pweight` variable, and the fact that it came from two levels of sampling would not concern you. Perhaps you would type `vce(cluster groupvar)` where `groupvar` identifies the top-level groups to get standard errors that control for correlation within these groups, but you would still use only a single weight variable.

Now take these same data and fit a two-level model with `mixed`. As seen in (14) in Methods and formulas later in this entry, it is not sufficient to use the single sampling weight $w_{ij}$, because weights enter into the log likelihood at both the group level and the individual level. Instead, what is required for a two-level model under this sampling design is $w_j$, the inverse of the probability that group $j$ is selected in the first stage, and $w_{i|j}$, the inverse of the probability that individual $i$ from group $j$ is selected at the second stage conditional on group $j$ already being selected. It simply will not do to just use $w_{ij}$ without making any assumptions about $w_j$.

Given the rules of conditional probability, $w_{ij} = w_j w_{i|j}$. If your dataset has only $w_{ij}$, then you will need to either assume equal probability sampling at the first stage ($w_j = 1$ for all $j$) or find some way to recover $w_j$ from other variables in your data; see Rabe-Hesketh and Skrondal (2006) and the references therein for some suggestions on how to do this, but realize that there is little yet known about how well these approximations perform in practice.

What you really need to fit your two-level model are data that contain $w_{ij}$ in addition to either $w_{ij}$ or $w_{i|j}$. If you have $w_{ij}$—that is, the unconditional inclusion weight for observation $i, j$—then you need to either divide $w_{ij}$ by $w_j$ to obtain $w_{i|j}$ or rescale $w_{ij}$ so that its dependence on $w_j$ disappears. If you already have $w_{i|j}$, then rescaling becomes optional (but still an important decision to make).
Weight rescaling is not an exact science, because the scale of the level-one weights is at issue regardless of whether they represent $w_{ij}$ or $w_{i|j}$: because $w_{ij}$ is unique to group $j$, the group-to-group magnitudes of these weights need to be normalized so that they are “consistent” from group to group. This is in stark contrast to a standard analysis, where the scale of sampling weights does not factor into estimation, instead only affecting the estimate of the total population size.

`mixed` offers three methods for standardizing weights in a two-level model, and you can specify which method you want via the `pwscale()` option. If you specify `pwscale(size)`, then the $w_{i|j}$ (or $w_{ij}$, it does not matter) are scaled to sum to the cluster size $n_j$. Method `pwscale(effective)` adds in a dependence on the sum of the squared weights so that level-one weights sum to the “effective” sample size. Just like `pwscale(size)`, `pwscale(effective)` also behaves the same whether you have $w_{i|j}$ or $w_{ij}$, and so it can be used with either.

Although both `pwscale(size)` and `pwscale(effective)` leave $w_j$ untouched, the `pwscale(gk)` method is a little different in that 1) it changes the weights at both levels and 2) it does assume you have $w_{i|j}$ for level-one weights and not $w_{ij}$ (if you have the latter, then first divide by $w_j$). Using the method of Graubard and Korn (1996), it sets the weights at the group level (level two) to the cluster averages of the products of both level weights (this product being $w_{ij}$). It then sets the individual weights to 1 everywhere; see Methods and formulas for the computational details of all three methods.

Determining which method is “best” is a tough call and depends on cluster size (the smaller the clusters, the greater the sensitivity to scale), whether the sampling is informative (that is, the sampling weights are correlated with the residuals), whether you are interested primarily in regression coefficients or in variance components, whether you have a simple random-intercept model or a more complex random-coefficients model, and other factors; see Rabe-Hesketh and Skrondal (2006), Carle (2009), and Pfeffermann et al. (1998) for some detailed advice. At the very least, you want to compare estimates across all three scaling methods (four, if you add no scaling) and perform a sensitivity analysis.

If you choose to rescale level-one weights, it does not matter whether you have $w_{i|j}$ or $w_{ij}$. For the `pwscale(size)` and `pwscale(effective)` methods, you get identical results, and even though `pwscale(gk)` assumes $w_{i|j}$, you can obtain this as $w_{i|j} = w_{ij}/w_j$ before proceeding.

If you do not specify `pwscale()`, then no scaling takes place, and thus at a minimum, you need to make sure you have $w_{i|j}$ in your data and not $w_{ij}$.

JUnit 12

Rabe-Hesketh and Skrondal (2006) analyzed data from the 2000 Programme for International Student Assessment (PISA) study on reading proficiency among 15-year-old American students, as performed by the Organisation for Economic Co-operation and Development (OECD). The original study was a three-stage cluster sample, where geographic areas were sampled at the first stage, schools at the second, and students at the third. Our version of the data does not contain the geographic-areas variable, so we treat this as a two-stage sample where schools are sampled at the first stage and students at the second.
( Programme for International Student Assessment (PISA) 2000 data)
. describe

obs: 2,069 Programme for International Student Assessment (PISA) 2000 data
vars: 11 12 Jun 2014 10:08
size: 37,242 (_dta has notes)

<table>
<thead>
<tr>
<th>variable name</th>
<th>storage</th>
<th>display</th>
<th>value</th>
<th>variable label</th>
</tr>
</thead>
<tbody>
<tr>
<td>female</td>
<td>byte</td>
<td>%8.0g</td>
<td>1</td>
<td>if female</td>
</tr>
<tr>
<td>isei</td>
<td>byte</td>
<td>%8.0g</td>
<td>1</td>
<td>International socio-economic index</td>
</tr>
<tr>
<td>w_fstuwt</td>
<td>float</td>
<td>%9.0g</td>
<td>1</td>
<td>Student-level weight</td>
</tr>
<tr>
<td>wnrschbw</td>
<td>float</td>
<td>%9.0g</td>
<td>1</td>
<td>School-level weight</td>
</tr>
<tr>
<td>high_school</td>
<td>byte</td>
<td>%8.0g</td>
<td>1</td>
<td>if highest level by either parent is high school</td>
</tr>
<tr>
<td>college</td>
<td>byte</td>
<td>%8.0g</td>
<td>1</td>
<td>if highest level by either parent is college</td>
</tr>
<tr>
<td>one_for</td>
<td>byte</td>
<td>%8.0g</td>
<td>1</td>
<td>if one parent foreign born</td>
</tr>
<tr>
<td>both_for</td>
<td>byte</td>
<td>%8.0g</td>
<td>1</td>
<td>if both parents are foreign born</td>
</tr>
<tr>
<td>test_lang</td>
<td>byte</td>
<td>%8.0g</td>
<td>1</td>
<td>if English (the test language) is spoken at home</td>
</tr>
<tr>
<td>pass_read</td>
<td>byte</td>
<td>%8.0g</td>
<td>1</td>
<td>if passed reading proficiency threshold</td>
</tr>
<tr>
<td>id_school</td>
<td>int</td>
<td>%8.0g</td>
<td>1</td>
<td>School ID</td>
</tr>
</tbody>
</table>

Sorted by:

For student \( i \) in school \( j \), where the variable \( \text{id\_school} \) identifies the schools, the variable \( w_{\text{fstuwt}} \) is a student-level overall inclusion weight \( (w_{ij}, \text{not } w_{i|j}) \) adjusted for noninclusion and nonparticipation of students, and the variable \( w_{\text{nrscbw}} \) is the school-level weight \( w_j \) adjusted for oversampling of schools with more minority students. The weight adjustments do not interfere with the methods prescribed above, and thus we can treat the weight variables simply as \( w_{ij} \) and \( w_j \), respectively.

Rabe-Hesketh and Skrondal (2006) fit a two-level logistic model for passing a reading proficiency threshold. We fit a two-level linear random-intercept model for socioeconomic index. Because we have \( w_{ij} \) and not \( w_{i|j} \), we rescale using \text{pwscale(size)} and thus obtain results as if we had \( w_{i|j} \).
mixed isei female high_school college one_for both_for test_lang
> [pw=w_fstuwt] || id_school:, pweight(wnrschbw) pwscale(size)

(output omitted)

Mixed-effects regression
Number of obs = 2,069
Group variable: id_school
Number of groups = 148

Obs per group:
  min = 1
  avg = 14.0
  max = 28

Wald chi2(6) = 187.23

Log pseudolikelihood = -1443093.9
Prob > chi2 = 0.0000

(Std. Err. adjusted for 148 clusters in id_school)

isei | Coef. Std. Err. z P>|z| [95% Conf. Interval]
-------|--------------------------------------------------
  female | .59379 .8732886 0.68 0.497 -1.117824 2.305404
high_school | 6.410618 1.500337 4.27 0.000 3.470011 9.351224
college | 19.39494 2.121145 9.14 0.000 15.23757 23.55231
  one_for | -.9584613 1.789947 -0.54 0.592 -4.466692 2.54977
both_for | -.2021101 2.32633 -0.09 0.931 -4.761633 4.357413
test_lang | 2.519539 2.393165 1.05 0.292 -2.170978 7.210056
     _cons | 28.10788 2.435712 11.54 0.000 23.33397 32.88179

Random-effects Parameters
| Estimate Std. Err. [95% Conf. Interval]
id_school: Identity
  var(_cons) | 34.69374 8.574865 21.37318 56.31617

Notes:
1. We specified the level-one weights using standard Stata weight syntax, that is, [pw=w_fstuwt].
2. We specified the level-two weights via the pweight(wnrschbw) option as part of the random-
   effects specification for the id_school level. As such, it is treated as a school-level weight.
   Accordingly, wnrschbw needs to be constant within schools, and mixed did check for that before
   estimating.
3. Because our level-one weights are unconditional, we specified pwscale(size) to rescale them.
4. As is the case with other estimation commands in Stata, standard errors in the presence of sampling
   weights are robust.
5. Robust standard errors are clustered at the top level of the model, and this will always be true unless
   you specify vce(cluster clustvar), where clustvar identifies an even higher level of grouping.
   As a form of sensitivity analysis, we compare the above with scaling via pwscale(gk). Because
   pwscale(gk) assumes $w_{ij}$, you want to first divide $w_{ij}$ by $w_j$. But you can handle that within the
   weight specification itself.
mixed isei female high_school college one_for both_for test_lang
> [pw=w_fstwtw/wnrschbw] || id_school:, pweight(wnrschbw) pwscale(gk)
(output omitted)

Mixed-effects regression
Group variable: id_school

<table>
<thead>
<tr>
<th></th>
<th>Number of obs</th>
<th>Number of groups</th>
<th>Obs per group:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>min</td>
</tr>
<tr>
<td></td>
<td>2,069</td>
<td>148</td>
<td>1</td>
</tr>
</tbody>
</table>

Wald chi2(6) = 291.37
Log pseudolikelihood = -7270505.6
Prob > chi2 = 0.0000
(Std. Err. adjusted for 148 clusters in id_school)

|              | Robust Coef. | Std. Err. | z     | P>|z| | [95% Conf. Interval] |
|--------------|--------------|-----------|-------|------|----------------------|
| female       | -.3519458    | .7436334  | -0.47 | 0.636 | -1.80944 1.105549    |
| high_school  | 7.074911     | 1.139777  | 6.21  | 0.000 | 4.84099 9.308833    |
| college      | 19.27285     | 1.286029  | 14.99 | 0.000 | 16.75228 21.79342  |
| one_for      | -.9142879    | 1.783091  | -0.51 | 0.608 | -4.409082 2.580506 |
| both_for     | 1.214151     | 1.611885  | 0.75  | 0.451 | -1.945085 4.373388 |
| test_lang    | 2.661866     | 1.556491  | 1.71  | 0.087 | -.3887996 5.712532 |
| _cons        | 31.20145     | 1.907413  | 16.36 | 0.000 | 27.46299 34.93991  |

Random-effects Parameters

<table>
<thead>
<tr>
<th></th>
<th>Robust Estimate</th>
<th>Std. Err.</th>
<th>[95% Conf. Interval]</th>
</tr>
</thead>
<tbody>
<tr>
<td>id_school: Identity</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>var(_cons)</td>
<td>31.67522</td>
<td>6.792239</td>
<td>20.80622 48.22209</td>
</tr>
<tr>
<td>var(Residual)</td>
<td>226.2429</td>
<td>8.150714</td>
<td>210.8188 242.7955</td>
</tr>
</tbody>
</table>

The results are somewhat similar to before, which is good news from a sensitivity standpoint. Note that we specified [pw=w_fstwtw/wnrschbw] and thus did the conversion from \( w_{ij} \) to \( w_{ij} | j \) within our call to mixed.

We close this section with a bit of bad news. Although weight rescaling and the issues that arise have been well studied for two-level models, as pointed out by Carle (2009), “... a best practice for scaling weights across multiple levels has yet to be advanced.” As such, pwscale() is currently supported only for two-level models. If you are fitting a higher-level model with survey data, you need to make sure your sampling weights are conditional on selection at the previous stage and not overall inclusion weights, because there is currently no rescaling option to fall back on if you do not.

Small-sample inference for fixed effects

Researchers are often interested in making inferences about fixed effects in a linear mixed-effects model. In the special case where the data are balanced and the mixed-effects model has a simple covariance structure, the sampling distributions of the statistics for testing hypotheses about fixed effects are known to follow an \( F \) distribution with specific denominator degrees of freedom (DDF) under the null hypothesis. For example, the test statistics for testing hypotheses about fixed effects in balanced split-plot designs and balanced repeated-measures designs have exact \( t \) or \( F \) distributions. In general, however, the null sampling distributions of test statistics for fixed effects are not known and can only be approximated in more complicated mixed-effects models.
For a large sample, the null sampling distributions of the test statistics can be approximated by a normal distribution for a one-hypothesis test and a $\chi^2$ distribution for a multiple-hypotheses test. This is the default behavior of `mixed`. However, these large-sample approximations may not be appropriate in small samples, and $t$ and $F$ distributions may provide better approximations.

You can specify the `dfmethod()` option to request small-sample inference for fixed effects. `mixed` with the `dfmethod()` option uses a $t$ distribution for one-hypothesis tests and an $F$ distribution for multiple-hypotheses tests for inference about fixed effects. We use DF to refer to degrees of freedom of a $t$ distribution, and we use DDF to refer to denominator degrees of freedom of an $F$ distribution.

Researchers have proposed various approximations that use $t$ and $F$ distributions but differ in how respective DF and DDF are computed (for example, Khuri, Mathew, and Sinha [1998]; Brown and Prescott [2015]; Schluchter and Elashoff [1990]; Elston [1998]; Kackar and Harville [1984]; Giesbrecht and Burns [1985]; Fai and Cornelius [1996]; and Kenward and Roger [1997, 2009]). `mixed` provides five methods with the `dfmethod()` option for calculating the DF of a $t$ distribution:

**Residual DDF (DF).** This method uses the residual degrees of freedom, $n - p$, as the DDF for all tests of fixed effects. For a linear model without random effects and with i.i.d errors, the distributions of the test statistics for testing the fixed effects are exact $t$ or $F$ distributions with the residual DF.

**Repeated DDF (DF).** This method partitions the residual degrees of freedom into the between-subject degrees of freedom and the within-subject degrees of freedom. This partitioning of the degrees of freedom arises from balanced repeated-measures ANOVA analysis. If levels of a fixed effect change within a subject, then the within-subject degrees of freedom is assigned to the fixed effect of interest; otherwise, the between-subject degrees of freedom is assigned to that fixed effect. Winer, Brown, and Michels (1991) showed that this method is appropriate only when the data are balanced and the correlation structure is assumed to be spherical. The repeated DDF method is supported only with two-level models. For DDF methods accounting for unbalanced repeated measures, see, for example, Schluchter and Elashoff (1990).

**ANOVA DDF (DF).** This method mimics the traditional ANOVA method. It determines the DDF for a fixed effect depending on whether the corresponding covariate is contained in any of the random-effects equations. If the covariate is contained in a random-effects equation, the DDF for the fixed effect is computed as the number of levels of the level variable from that equation minus one. If the covariate is specified in more than one random-effects equation, the DDF for the fixed effect is computed as the smallest number of levels of the level variables from those equations minus one and is a conservative estimate of the true DDF. If the covariate is specified only in the fixed-effects equation, the DDF is computed as $\nu_{ddf} = n - \text{rank}(X, Z)$. This method leads to an exact sampling distribution of the test statistics only when random effects are balanced and the residuals are i.i.d; see, for example, chapter 1.6 in Brown and Prescott (2015) for details.

**Satterthwaite DDF (DF).** This method performs a generalization of the Satterthwaite approximation based on Kackar and Harville (1984), Giesbrecht and Burns (1985), and Fai and Cornelius (1996). Giesbrecht and Burns (1985) developed a method of computing the DDF for a single-hypothesis test that is analogous to Satterthwaite’s approximation of the degrees of freedom of a linear combination of ANOVA mean squares. For a multiple-hypotheses test, Fai and Cornelius (1996) proposed an extension of the Giesbrecht–Burns single-degree-of-freedom method. This method involves the spectral decomposition of the contrast matrix of the hypothesis test and repeated application of the single-degree-of-freedom $t$ test. See Denominator degrees of freedom in Methods and formulas for more computational details.

**Kenward–Roger DDF (DF).** This method, developed by Kenward and Roger (1997), was designed to provide an approximation that improves the performance of hypothesis tests about fixed effects in small samples for complicated mixed-effects models and reproduces the exact inference available
for simpler mixed-effects models. It provides adjusted test statistics, more appropriate DDFs for the approximate $F$ distributions when exact inference is not available, and yields the exact $t$ and $F$ distributions when exact inference is available. This method first accounts for the small-sample bias and the variability of the estimated random effects to obtain an adjusted estimator of the fixed-effects covariance matrix. Then, it proposes an approximate $F$ test based on a scaled Wald test statistic that uses the adjusted variance–covariance estimator. See *Denominator degrees of freedom* in *Methods and formulas* for more computational details.

Residual, repeated, and ANOVA are known as “exact” methods in the literature. These methods are suitable only when the sampling distributions of the test statistics are known to be $t$ or $F$. This is usually only known for certain classes of linear mixed-effects models with simple covariance structures and when data are balanced. These methods are available with both ML and REML estimation.

Satterthwaite and Kenward–Roger are known as “approximation” methods in the literature. These methods are for unbalanced data and complicated covariance structures where the sampling distributions of test statistics are unknown and can only be approximated. Both methods are available only with REML estimation. For single-hypothesis tests, DDFs calculated with the Kenward–Roger method are the same as those calculated with the Satterthwaite method, but they differ for multiple-hypotheses tests. Although DDFs of the two methods are the same for single-hypothesis tests, the inference is not the same because the Kenward–Roger method uses bias-adjusted standard errors.

Except for the special cases for which the sampling distributions are known, there is no definitive recommendation for which approximation performs best. Schaalje, McBride, and Fellingham (2002) compared the Satterthwaite method with the Kenward–Roger method via simulation using different covariance structures and various sample sizes. They concluded that the Kenward–Roger method outperforms the Satterthwaite method in most situations. They recommend using the Satterthwaite method only when the covariance structure of the data is compound symmetry and the sample size is moderately large. The Kenward–Roger method, however, is not guaranteed to work well in all situations. For example, for more complicated covariance structures and very small-sample sizes, the Kenward–Roger method may produce inflated type I error rates. In conclusion, you should choose your DDF method carefully. See, for example, Schaalje, McBride, and Fellingham (2002), Chen and Wei (2003), Vallejo et al. (2004), and West, Welch, and Gałecki (2015) for a comparison of different approximations.

Both types of methods, exact and approximation, are available for single-hypothesis tests. For multiple-hypotheses tests, exact methods are available only if DDFs associated with fixed effects are the same for all tested covariates. See *Denominator degrees of freedom* in *Methods and formulas* for details.

Example 13

Consider an example from Winer, Brown, and Michels (1991, table 4.3), also analyzed in example 15 of [R] anova, which reports the reaction time for five subjects who were tested with four drugs. The reaction time was recorded in the variable score. Assume that person is random (that is, we wish to infer to the larger population of possible subjects) and drug is fixed (that is, only four drugs are of interest). This is an example of a mixed-effects model with a simple covariance structure—a balanced repeated-measures design. The dataset contains only 20 observations, so we would like to account for the small sample in our analysis. Because this is a balanced repeated-measures design, we can use the repeated method to obtain small-sample inference for fixed effects. We specify the dfmethod(repeated) option with mixed. We also request REML estimates by specifying the reml option to account for the small number of groups.
. use http://www.stata-press.com/data/r14/t43 (T4.3 -- Winer, Brown, Michels)
mixed score i.drug || person:, reml dfmethod(repeated)
Performing EM optimization:
Performing gradient-based optimization:
Iteration 0: log restricted-likelihood = -49.640099
Iteration 1: log restricted-likelihood = -49.640099
Computing standard errors:
Computing degrees of freedom:
Mixed-effects REML regression
Number of obs = 20
Group variable: person
Number of groups = 5
Obs per group:
    min = 4
    avg = 4.0
    max = 4
DF method: Repeated
DF:
    min = 4.00
    avg = 10.00
    max = 12.00
F(3, 12.00) = 24.76
Log restricted-likelihood = -49.640099
Prob > F = 0.0000

| score | Coef.  | Std. Err. | t     | P>|t|  | [95% Conf. Interval] |
|-------|--------|-----------|-------|------|---------------------|
| drug  |       |           |       |      |                     |
| 2     | -.8   | 1.939072  | -0.41 | 0.687| -5.024874 3.424874 |
| 3     | -10.8 | 1.939072  | -5.57 | 0.000| -15.02487 -6.575126 |
| 4     | 5.6   | 1.939072  | 2.89  | 0.014| 1.375126 9.824874  |
| _cons | 26.4  | 3.149604  | 8.38  | 0.001| 17.6553 35.1447    |

Random-effects Parameters
<table>
<thead>
<tr>
<th>Estimate</th>
<th>Std. Err.</th>
<th>[95% Conf. Interval]</th>
</tr>
</thead>
</table>
| person: Identity
| var(_cons) | 40.20004 | 30.10272 | 9.264606 | 174.4319 |
| var(Residual) | 9.399997 | 3.837532 | 4.22305 | 20.92325 |

LR test vs. linear model: chibar2(01) = 15.03 Prob >= chibar2 = 0.0001

In the table for fixed effects, \( t \) statistics are reported instead of the default \( z \) statistics. We can compare our small-sample inference with the corresponding large-sample inference for fixed effects. We do not need to rerun the estimation command, because we can obtain large-sample results upon replay by default.
. mixed
Mixed-effects REML regression
Number of obs = 20
Group variable: person
Number of groups = 5
Obs per group:
    min = 4
    avg = 4.0
    max = 4
Wald chi2(3) = 74.28
Log restricted-likelihood = -49.640099 Prob > chi2 = 0.0000

|    | Coef.  | Std. Err. | z       | P>|z|  | [95% Conf. Interval] |
|----|--------|-----------|---------|------|---------------------|
| drug |        |           |         |      |                     |
| 2   | -.8    | 1.939072  | -0.41   | 0.680| -4.600511           | 3.000511 |
| 3   | -10.8  | 1.939072  | -5.57   | 0.000| -14.60051           | -6.999489|
| 4   | 5.6    | 1.939072  | 2.89    | 0.004| 1.799489            | 9.400511 |
| _cons | 26.4   | 3.149604  | 8.38    | 0.000| 20.22689            | 32.57311 |

Random-effects Parameters

<table>
<thead>
<tr>
<th></th>
<th>Estimate</th>
<th>Std. Err.</th>
<th>[95% Conf. Interval]</th>
</tr>
</thead>
<tbody>
<tr>
<td>person: Identity</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>var(_cons)</td>
<td>40.20004</td>
<td>30.10272</td>
<td>9.264606</td>
</tr>
<tr>
<td>var(Residual)</td>
<td>9.399997</td>
<td>3.837532</td>
<td>4.22305</td>
</tr>
</tbody>
</table>

LR test vs. linear model: chibar2(01) = 15.03 Prob >= chibar2 = 0.0001

Comparing the above large-sample inference for fixed effects of drug with the small-sample inference, we see that the p-value for the level 4 of drug changes from 0.014 to 0.004.

If we wanted to replay our small-sample estimation results, we would type

. mixed, small
(output omitted)

The specified DF method and summaries of the coefficient-specific DFs are reported in the output header. We can use the dftable() option to display a fixed-effects table containing coefficient-specific DFs. dftable(pvalue) reports the fixed-effects table containing DFs, t statistics, and p-values, and dftable(ci) reports the fixed-effects table containing DFs and confidence intervals.
Because levels of `drug` vary within `person`, the within-subject degrees of freedom, 12, are assigned to the coefficients for the levels of `drug`. The DF for the constant term is always the between-subject degrees of freedom, 4 in this example, because it is constant within random-effects levels.

The model $F$ test is reported in the output header instead of the default $\chi^2$ test. The $F$ statistic for testing $\text{drug} = 0$ is 24.76 with DDF = 12, which agrees with the results of `anova, repeated()`:

```
. anova score person drug, repeated(drug)
```

```
Number of obs = 20  R-squared = 0.9244
Root MSE = 3.06594  Adj R-squared = 0.8803

Source       Partial SS   df  MS      F    Prob>F
------------- ----------------- -------- -------- ------ --------
Model        1379.0         7  197.0    20.96  0.0000
person       680.8          4  170.2    18.11  0.0001
drug          698.2          3 232.73    24.76  0.0000
Residual     112.8          12   9.4      Prob>F
Total        1491.8         19  78.52

Between-subjects error term: person
Levels: 5 (4 df)
Lowest b.s.e. variable: person
Repeated variable: drug

Huynh-Feldt epsilon = 1.0789
*Huynh-Feldt epsilon reset to 1.0000
Greenhouse-Geisser epsilon = 0.6049
Box's conservative epsilon = 0.3333

Source         df  F   Regular   H-F    G-G    Box
------------- ------ ------ -------- ------ -------- --------
drug         3 24.76  0.0000  0.0000  0.0006  0.0076
Residual     12
```

## Mixed-effects REML regression

Number of obs = 20
Group variable: person

<table>
<thead>
<tr>
<th>Obs per group:</th>
<th>min</th>
<th>avg</th>
<th>max</th>
</tr>
</thead>
<tbody>
<tr>
<td>min</td>
<td>4</td>
<td>4.0</td>
<td></td>
</tr>
<tr>
<td>avg</td>
<td></td>
<td></td>
<td>4.0</td>
</tr>
<tr>
<td>max</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

DF method: Repeated

<table>
<thead>
<tr>
<th>DF:</th>
<th>min</th>
<th>avg</th>
<th>max</th>
</tr>
</thead>
<tbody>
<tr>
<td>min</td>
<td>4.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>avg</td>
<td>10.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>max</td>
<td>12.00</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Log restricted-likelihood = -49.640099
Prob > F = 0.0000

| score | Coef. Std. Err. | DF  | t     | P>|t| |
|-------|-----------------|-----|-------|-----|
| drug  |                 |     |       |     |
| 2     | -.8             | 1.939072 | 12.0 | -0.41| 0.687 |
| 3     | -10.8           | 1.939072 | 12.0 | -5.57| 0.000 |
| 4     | 5.6             | 1.939072 | 12.0 | 2.89 | 0.014 |
| _cons| 26.4            | 3.149604 | 4.0  | 8.38 | 0.001 |
Example 14

Consider West, Welch, and Galecki’s (2015) dental veneer dataset from example 7, containing two measurements on each tooth from multiple teeth per patient. Because of small-sample size, we would like to obtain small-sample inference for fixed effects.

Some patients are missing observations for some teeth:

```
use http://www.stata-press.com/data/r14/veneer, clear
(Dental veneer data)
table patient tooth
```

<table>
<thead>
<tr>
<th>Patient ID</th>
<th>Tooth number with patient</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>6 7 8 9 10 11</td>
</tr>
<tr>
<td>1</td>
<td>2 2 2 2 2 2</td>
</tr>
<tr>
<td>3</td>
<td>2 2 2 2 2 2</td>
</tr>
<tr>
<td>4</td>
<td>2 2 2 2 2 2</td>
</tr>
<tr>
<td>5</td>
<td>2 2 2 2 2 2</td>
</tr>
<tr>
<td>6</td>
<td>2 2 2 2 2 2</td>
</tr>
<tr>
<td>7</td>
<td>2 2 2 2 2 2</td>
</tr>
<tr>
<td>8</td>
<td>2 2 2 2 2 2</td>
</tr>
<tr>
<td>9</td>
<td>2 2 2 2 2 2</td>
</tr>
<tr>
<td>10</td>
<td>2 2 2 2 2 2</td>
</tr>
<tr>
<td>12</td>
<td>2 2 2 2 2 2</td>
</tr>
<tr>
<td>13</td>
<td>2 2 2 2 2 2</td>
</tr>
<tr>
<td>14</td>
<td>2 2 2 2 2 2</td>
</tr>
</tbody>
</table>

The dataset is unbalanced; therefore, exact $F$ tests for fixed effects are unavailable. As such, we will use the Satterthwaite and the Kenward–Roger approximation methods for calculating DF. Let’s fit the model using the Kenward–Roger method first by specifying `dfmethod(kroger)`.

```
mixed gcf followup base_gcf cda age || patient: followup, cov(un)>
> || tooth:, reml nolog dfmethod(kroger)
mixed-effects REML regression
```

<table>
<thead>
<tr>
<th>Group Variable</th>
<th>No. of Groups</th>
<th>Observations per Group</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Minimum</td>
<td>Average</td>
</tr>
<tr>
<td>patient</td>
<td>12</td>
<td>9.2</td>
</tr>
<tr>
<td>tooth</td>
<td>55</td>
<td>2.0</td>
</tr>
</tbody>
</table>

DF method: Kenward–Roger

| gcf     | Coef.  | Std. Err. | t    | P>|t|  | [95% Conf. Interval] |
|---------|--------|-----------|------|------|---------------------|
| followup| .3009815 | 1.938641 | 0.16 | 0.879 | -3.96767 4.569633   |
| base_gcf| -.0183127 | .1466261 | -0.12 | 0.901 | -.3132419 .2766164 |
| cda     | -.329303 | .5533506 | -0.60 | 0.554 | -1.440355 .7817493 |
| age     | -.5773932 | .2350491 | -2.46 | 0.033 | -1.098324 -.056462 |
| _cons   | 45.73862 | 13.21824 | 3.46 | 0.002 | 18.53866 72.93858  |
Mixed-effects REML regression  

<table>
<thead>
<tr>
<th>Group Variable</th>
<th>No. of Groups</th>
<th>Observations per Group</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Minimum</td>
<td>Average</td>
</tr>
<tr>
<td>patient</td>
<td>2</td>
<td>9.2</td>
</tr>
<tr>
<td>tooth</td>
<td>2</td>
<td>2.0</td>
</tr>
</tbody>
</table>

DF method: Satterthwaite  
DF:  
- min = 10.41  
- avg = 28.96  
- max = 50.71  

F(4, 16.49) = 1.87  
Log restricted-likelihood = -420.92761  
Prob > F = 0.1638  

Compared with the \( p \)-values of the large-sample results from example 7, the \( p \)-values for age and _cons change substantially from 0.007 and 0.000 to 0.033 and 0.002, respectively. Note that for the Kenward–Roger method, not only the \( p \)-values and confidence intervals differ from those of the large-sample results but also the standard errors for the fixed effects differ. The standard errors differ because this method uses a bias-adjusted estimator of the variance–covariance matrix of fixed effects.

Now, let’s fit the model using the Satterthwaite approximation:

```
mixed gcf followup base_gcf cda age || patient: followup, cov(un) > || tooth:, reml nolog dfmethod(satterthwaite)
```

---

| gcf   | Coef.  | Std. Err. | t     | P>|t|     | [95% Conf. Interval] |
|-------|--------|-----------|-------|---------|---------------------|
| followup | .3009815 | 1.936863  | 0.16  | 0.879   | -3.963754 - 4.565717  |
| base_gcf | -.0183127 | .1433094  | -.13  | 0.899   | -.3065704 - 0.269945  |
| cda    | -.329303  | .5292525  | -.62  | 0.537   | -1.39197 - 0.7333636  |
| age    | -.5773932 | .2139656  | -.270 | 0.022   | -1.051598 - 0.1031885 |
| _cons  | 45.73862  | 12.55497  | 3.64  | 0.001   | 19.90352 - 71.57372   |
### Random-effects Parameters

<table>
<thead>
<tr>
<th></th>
<th>Estimate</th>
<th>Std. Err.</th>
<th>[95% Conf. Interval]</th>
</tr>
</thead>
<tbody>
<tr>
<td>patient: Unstructured</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>var(followup)</td>
<td>41.88772</td>
<td>18.79997</td>
<td>17.38009 100.9535</td>
</tr>
<tr>
<td>var(_cons)</td>
<td>524.9851</td>
<td>253.0205</td>
<td>204.1287 1350.175</td>
</tr>
<tr>
<td>cov(followup,_cons)</td>
<td>-140.4229</td>
<td>66.57623</td>
<td>-270.9099 -9.935907</td>
</tr>
<tr>
<td>tooth: Identity</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>var(_cons)</td>
<td>47.45738</td>
<td>16.63034</td>
<td>23.8792 94.3165</td>
</tr>
<tr>
<td>var(Residual)</td>
<td>48.86704</td>
<td>10.50523</td>
<td>32.06479 74.47382</td>
</tr>
<tr>
<td>LR test vs. linear model: chi2(4) = 91.12 Prob &gt; chi2 = 0.0000</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Note: LR test is conservative and provided only for reference.

Using the Satterthwaite method, we see that the \( p \)-value for \texttt{age} is 0.022 and for \texttt{cons} is 0.001 and that these are again substantially different from their large-sample counterparts. On the other hand, unlike the standard errors for the Kenward–Roger method, those for the Satterthwaite method are the same as the standard errors from the large-sample results.

Looking at the DF summaries in the output header of the two methods, we notice that they are exactly the same. This is because DFs for fixed effects obtained using the Kenward–Roger and Satterthwaite methods are the same for single-hypothesis tests. (You can verify this by specifying, for example, \texttt{dftable(pvalue)} with the above commands or by using \texttt{estat df}; see \[ME\] mixed postestimation.) The DFs differ, however, for multiple-hypotheses tests. For example, DDF computed for the overall model test using \texttt{dfmethod(satterthwaite)} (16.49) is smaller than that computed using \texttt{dfmethod(kroger)} (27.96).

There are no general guidelines to which method should be preferred, but according to Schaalje, McBride, and Fellingham (2002), the Kenward–Roger method outperforms the Satterthwaite method when the variance–covariance structure of the random effects is unstructured, which is the case in our example.

Determining which DDF method is best is a difficult task and may often need simulation. The choice of the method depends on the specified covariance structure, sample size, and imbalance of the data. No method applies to all situations; thus you should use caution when choosing among methods.
mixed — Multilevel mixed-effects linear regression

Stored results

mixed stores the following in e():

Scalars

- `e(N)` number of observations
- `e(k)` number of parameters
- `e(k_f)` number of fixed-effects parameters
- `e(k_r)` number of random-effects parameters
- `e(k_rs)` number of variances
- `e(k_rc)` number of covariances
- `e(k_res)` number of residual-error parameters
- `e(N_clust)` number of clusters
- `e(nrgrps)` number of residual-error by() groups
- `e(ar_p)` AR order of residual errors, if specified
- `e(ma_q)` MA order of residual errors, if specified
- `e(res_order)` order of residual-error structure, if appropriate
- `e(dff_m)` model degrees of freedom
- `e(small)` 1 if dfmethod() option specified, 0 otherwise
- `e(F)` overall F test statistic when dfmethod() is specified
- `e(ddf_m)` model DDF
- `e(df_max)` maximum DF
- `e(df_avg)` average DF
- `e(df_min)` minimum DF
- `e(ll)` log (restricted) likelihood
- `e(chi2)` \( \chi^2 \)
- `e(p)` significance
- `e(ll_c)` log likelihood, comparison model
- `e(chi2_c)` \( \chi^2 \), comparison model
- `e(df_c)` degrees of freedom, comparison model
- `e(p_c)` significance, comparison model
- `e(rank)` rank of e(V)
- `e(rc)` return code
- `e(converged)` 1 if converged, 0 otherwise

Macros

- `e(cmd)` mixed
- `e(cmdline)` command as typed
- `e(depvar)` name of dependent variable
- `e(vtype)` weight type (first-level weights)
- `e(wtype)` weight expression (first-level weights)
- `e(fweight_k)` fweight variable for \( k \)th highest level, if specified
- `e(pweight_k)` pweight variable for \( k \)th highest level, if specified
- `e(ivars)` grouping variables
- `e(title)` title in estimation output
- `e(redim)` random-effects dimensions
- `e(vartypes)` variance-structure types
- `e(revars)` random-effects covariates
- `e(resopt)` residuals() specification, as typed
- `e(rstructure)` residual-error structure
- `e(rstructlab)` residual-error structure output label
- `e(rbyvar)` residual-error by() variable, if specified
- `e(rbylabels)` residual-error by() groups labels
- `e(pwscaler)` sampling-weight scaling method
- `e(timevar)` residual-error \( t() \) variable, if specified
- `e(dfmethod)` DF method specified in dfmethod()
- `e(dftitle)` title for DF method
- `e(chi2type)` Wald; type of model \( \chi^2 \) test
- `e(clustvar)` name of cluster variable
- `e(vcetype)` vcetype specified in vce()
- `e(vcetype)` title used to label Std. Err.
- `e(method)` ML or REML
- `e(opt)` type of optimization
- `e(optmetric)` matsqrt or matlog; random-effects matrix parameterization
e(émonly) emonly, if specified
e(ml_method) type of ml method
e(technique) maximization technique
e(datasignature) the checksum
e(datasignaturevars) variables used in calculation of checksum
e(properties) b V

Matrices
e(b) coefficient vector
e(N_g) group counts
e(g_min) group-size minimums
e(g_avg) group-size averages
e(g_max) group-size maximums
e(tmap) ID mapping for unstructured residual errors
e(V) variance–covariance matrix of the estimators
e(V_modelbased) model-based variance
e(df) parameter-specific DF for fixed effects
e(V_df) variance–covariance matrix of the estimators when dfmethod(kroger) is specified

Functions
e(sample) marks estimation sample

### Methods and formulas

As given by (1), in the absence of weights we have the linear mixed model

\[ y = X\beta + Zu + \epsilon \]

where \( y \) is the \( n \times 1 \) vector of responses, \( X \) is an \( n \times p \) design/covariate matrix for the fixed effects \( \beta \), and \( Z \) is the \( n \times q \) design/covariate matrix for the random effects \( u \). The \( n \times 1 \) vector of errors \( \epsilon \) is for now assumed to be multivariate normal with mean 0 and variance matrix \( \sigma^2_\epsilon I_n \). We also assume that \( u \) has variance–covariance matrix \( G \) and that \( u \) is orthogonal to \( \epsilon \) so that

\[
\text{Var} \begin{bmatrix} u \\ \epsilon \end{bmatrix} = \begin{bmatrix} G & 0 \\ 0 & \sigma^2_\epsilon I_n \end{bmatrix}
\]

Considering the combined error term \( Zu + \epsilon \), we see that \( y \) is multivariate normal with mean \( X\beta \) and \( n \times n \) variance–covariance matrix

\[ V = ZGZ' + \sigma^2_\epsilon I_n \]

Defining \( \theta \) as the vector of unique elements of \( G \) results in the log likelihood

\[
L(\beta, \theta, \sigma^2_\epsilon) = -\frac{1}{2} \left\{ n \log(2\pi) + \log |V| + (y - X\beta)'V^{-1}(y - X\beta) \right\} \tag{9}
\]

which is maximized as a function of \( \beta, \theta, \) and \( \sigma^2_\epsilon \). As explained in chapter 6 of Searle, Casella, and McCulloch (1992), considering instead the likelihood of a set of linear contrasts \( Ky \) that do not depend on \( \beta \) results in the restricted log likelihood

\[
L_R(\beta, \theta, \sigma^2_\epsilon) = L(\beta, \theta, \sigma^2_\epsilon) - \frac{1}{2} \log |X'V^{-1}X| \tag{10}
\]
Given the high dimension of $V$, however, the log-likelihood and restricted log-likelihood criteria are not usually computed by brute-force application of the above expressions. Instead, you can simplify the problem by subdividing the data into independent clusters (and subclusters if possible) and using matrix decomposition methods on the smaller matrices that result from treating each cluster one at a time.

Consider the two-level model described previously in (2),

$$y_j = X_j \beta + Z_j u_j + \epsilon_j$$

for $j = 1, \ldots, M$ clusters with cluster $j$ containing $n_j$ observations, with $\text{Var}(u_j) = \Sigma$, a $q \times q$ matrix.

Efficient methods for computing (9) and (10) are given in chapter 2 of Pinheiro and Bates (2000). Namely, for the two-level model, define $\Delta$ to be the Cholesky factor of $\sigma^2 \Sigma^{-1}$, such that $\sigma^2 \Sigma^{-1} = \Delta' \Delta$. For $j = 1, \ldots, M$, decompose

$$\begin{bmatrix} Z_j \\ \Delta \end{bmatrix} = Q_j \begin{bmatrix} R_{11j} \\ 0 \end{bmatrix}$$

by using an orthogonal-triangular (QR) decomposition, with $Q_j$ a $(n_j + q)$-square matrix and $R_{11j}$ a $q$-square matrix. We then apply $Q_j$ as follows:

$$\begin{bmatrix} R_{10j} \\ R_{00j} \end{bmatrix} = Q_j \begin{bmatrix} X_j \\ 0 \end{bmatrix}; \quad \begin{bmatrix} c_{1j} \\ c_{0j} \end{bmatrix} = Q_j' \begin{bmatrix} y_j \\ 0 \end{bmatrix}$$

Stack the $R_{00j}$ and $c_{0j}$ matrices, and perform the additional QR decomposition

$$\begin{bmatrix} R_{001} & c_{01} \\ \vdots & \vdots \\ R_{00M} & c_{0M} \end{bmatrix} = Q_0 \begin{bmatrix} R_{00} & c_0 \\ 0 & c_1 \end{bmatrix}$$

Pinheiro and Bates (2000) show that ML estimates of $\beta$, $\sigma^2$, and $\Delta$ (the unique elements of $\Delta$, that is) are obtained by maximizing the profile log likelihood (profiled in $\Delta$)

$$L(\Delta) = \frac{n}{2} \left\{ \log n - \log(2\pi) - 1 \right\} - n \log ||c_1|| + \sum_{j=1}^{M} \log \left| \frac{\det(\Delta)}{\det(R_{11j})} \right|$$  

where $|| \cdot ||$ denotes the 2-norm. Following this maximization with

$$\hat{\beta} = R_{00}^{-1} c_0; \quad \hat{\sigma}^2 = n^{-1} ||c_1||^2$$
REML estimates are obtained by maximizing
\[
L_R(\Delta) = \frac{n - p}{2} \{ \log(n - p) - \log(2\pi) - 1 \} - (n - p) \log |c_1| \\
- \log |\det(R_{00})| + \sum_{j=1}^{M} \log \left| \frac{\det(\Delta)}{\det(R_{11j})} \right|
\]
followed by
\[
\hat{\beta} = R_{00}^{-1} c_0; \quad \hat{\sigma}^2 = (n - p)^{-1} |c_1|^2
\]

For numerical stability, maximization of (11) and (13) is not performed with respect to the unique elements of \( \Delta \) but instead with respect to the unique elements of the matrix square root (or matrix logarithm if the matlog option is specified) of \( \Sigma / \sigma^2 \); define \( \gamma \) to be the vector containing these elements.

Once maximization with respect to \( \gamma \) is completed, \( (\gamma, \sigma^2) \) is reparameterized to \{\( \alpha \), \( \log(\sigma) \)\}, where \( \alpha \) is a vector containing the unique elements of \( \Sigma \), expressed as logarithms of standard deviations for the diagonal elements and hyperbolic arctangents of the correlations for off-diagonal elements. This last step is necessary 1) to obtain a joint variance–covariance estimate of the elements of \( \Sigma \) and \( \sigma^2 \); 2) to obtain a parameterization under which parameter estimates can be interpreted individually, rather than as elements of a matrix square root (or logarithm); and 3) to parameterize these elements such that their ranges each encompass the entire real line.

Obtaining a joint variance–covariance matrix for the estimated \{\( \alpha \), \( \log(\sigma) \)\} requires the evaluation of the log likelihood (or log-restricted likelihood) with only \( \beta \) profiled out. For ML, we have
\[
L^* \{\alpha, \log(\sigma)\} = L(\Delta(\alpha, \sigma^2), \sigma^2)
\]
\[
= -\frac{n}{2} \log(2\pi\sigma^2) - \frac{|c_1|^2}{2\sigma^2} + \sum_{j=1}^{M} \log \left| \frac{\det(\Delta)}{\det(R_{11j})} \right|
\]
with the analogous expression for REML.

The variance–covariance matrix of \( \hat{\beta} \) is estimated as
\[
\hat{\text{Var}}(\hat{\beta}) = \hat{\sigma}^2 R_{00}^{-1} (R_{00}^{-1})'
\]
but this does not mean that \( \hat{\text{Var}}(\hat{\beta}) \) is identical under both ML and REML because \( R_{00} \) depends on \( \Delta \). Because \( \hat{\beta} \) is asymptotically uncorrelated with \{\( \hat{\alpha} \), \( \log(\hat{\sigma}) \)\}, the covariance of \( \hat{\beta} \) with the other estimated parameters is treated as 0.

Parameter estimates are stored in \( e(b) \) as \{\( \hat{\beta} \), \( \hat{\alpha} \), \( \log(\hat{\sigma}) \)\}, with the corresponding (block-diagonal) variance–covariance matrix stored in \( e(V) \). Parameter estimates can be displayed in this metric by specifying the estmetric option. However, in mixed output, variance components are most often displayed either as variances and covariances or as standard deviations and correlations.

EM iterations are derived by considering the \( u_j \) in (2) as missing data. Here we describe the procedure for maximizing the log likelihood via EM; the procedure for maximizing the restricted log likelihood is similar. The log likelihood for the full data \( (y, u) \) is
\[
L_F(\beta, \Sigma, \sigma^2) = \sum_{j=1}^{M} \left\{ \log f_1(y_j | u_j, \beta, \sigma^2) + \log f_2(u_j | \Sigma) \right\}
\]
where \( f_1(\cdot) \) is the density function for multivariate normal with mean \( X_j \beta + Z_j u_j \) and variance \( \sigma_\epsilon^2 I_n_j \), and \( f_2(\cdot) \) is the density for multivariate normal with mean \( \mathbf{0} \) and \( q \times q \) covariance matrix \( \Sigma \). As before, we can profile \( \beta \) and \( \sigma_\epsilon^2 \) out of the optimization, yielding the following EM iterative procedure:

1. For the current iterated value of \( \Sigma^{(t)} \), fix \( \hat{\beta} = \hat{\beta}(\Sigma^{(t)}) \) and \( \hat{\sigma}_\epsilon^2 = \hat{\sigma}_\epsilon^2(\Sigma^{(t)}) \) according to (12).

2. Expectation step: Calculate

\[
D(\Sigma) \equiv E \left\{ L_F(\hat{\beta}, \Sigma, \hat{\sigma}_\epsilon^2) | \mathbf{y} \right\} = C - \frac{M}{2} \log \det (\Sigma) - \frac{1}{2} \sum_{j=1}^{M} E \left( u_j' \Sigma^{-1} u_j | \mathbf{y} \right)
\]

where \( C \) is a constant that does not depend on \( \Sigma \), and the expected value of the quadratic form \( u_j' \Sigma^{-1} u_j \) is taken with respect to the conditional density \( f(u_j | \mathbf{y}, \hat{\beta}, \Sigma^{(t)}, \hat{\sigma}_\epsilon^2) \).

3. Maximization step: Maximize \( D(\Sigma) \) to produce \( \Sigma^{(t+1)} \).

For general, symmetric \( \Sigma \), the maximizer of \( D(\Sigma) \) can be derived explicitly, making EM iterations quite fast.

For general, residual-error structures,

\[
\text{Var}(\epsilon_j) = \sigma_\epsilon^2 \Lambda_j
\]

where the subscript \( j \) merely represents that \( \epsilon_j \) and \( \Lambda_j \) vary in dimension in unbalanced data, the data are first transformed according to

\[
\mathbf{y}_j^* = \hat{\Lambda}_j^{-1/2} \mathbf{y}_j; \quad \mathbf{X}_j^* = \hat{\Lambda}_j^{-1/2} \mathbf{X}_j; \quad \mathbf{Z}_j^* = \hat{\Lambda}_j^{-1/2} \mathbf{Z}_j;
\]

and the likelihood-evaluation techniques described above are applied to \( \mathbf{y}_j^*, \mathbf{X}_j^*, \) and \( \mathbf{Z}_j^* \) instead. The unique elements of \( \Lambda, \rho, \) are estimated along with the fixed effects and variance components. Because \( \sigma_\epsilon^2 \) is always estimated and multiplies the entire \( \Lambda_j \) matrix, \( \hat{\rho} \) is parameterized to take this into account.

In the presence of sampling weights, following Rabe-Hesketh and Skrondal (2006), the weighted log pseudolikelihood for a two-level model is given as

\[
L(\beta, \Sigma, \sigma_\epsilon^2) = \sum_{j=1}^{M} w_j \log \left[ \int \exp \left\{ \sum_{i=1}^{n_j} w_{i|j} \log f_1(y_{ij} | u_j, \beta, \sigma_\epsilon^2) \right\} f_2(u_j | \Sigma) du_j \right]
\]

(14)

where \( w_j \) is the inverse of the probability of selection for the \( j \)th cluster, \( w_{i|j} \) is the inverse of the conditional probability of selection of individual \( i \) given the selection of cluster \( j \), and \( f_1(\cdot) \) and \( f_2(\cdot) \) are the multivariate normal densities previously defined.

Weighted estimation is achieved through incorporating \( w_j \) and \( w_{i|j} \) into the matrix decomposition methods detailed above to reflect replicated clusters for \( w_j \) and replicated observations within clusters for \( w_{i|j} \). Because this estimation is based on replicated clusters and observations, frequency weights are handled similarly.
Rescaling of sampling weights can take one of three available forms:

Under `pwscale(size)`,

\[ w^*_i|j = n_j w_{i|j} \left( \sum_{i=1}^{n_j} w_{i|j} \right)^{-1} \]

Under `pwscale(effective)`,

\[ w^*_i|j = w_{i|j} \left[ \left( \sum_{i=1}^{n_j} w_{i|j} \right) \left( \sum_{i=1}^{n_j} w_{i|j}^2 \right) \right]^{-1} \]

Under both the above, \( w_j \) remains unchanged. For method `pwscale(gk)`, however, both weights are modified:

\[ w^*_j = n_j^{-1} \sum_{i=1}^{n_j} w_{i|j} w_j; \quad w^*_i|j = 1 \]

Under ML estimation, robust standard errors are obtained in the usual way (see [P] `robust`) with the one distinction being that in multilevel models, robust variances are, at a minimum, clustered at the highest level. This is because given the form of the log likelihood, scores aggregate at the top-level clusters. For a two-level model, scores are obtained as the partial derivatives of \( L_j(\beta, \Sigma, \sigma^2_{\epsilon}) \) with respect to \( \{\beta, \alpha, \log(\sigma_{\epsilon})\} \), where \( L_j \) is the log likelihood for cluster \( j \) and \( L = \sum_{j=1}^{M} L_j \). Robust variances are not supported under REML estimation because the form of the log restricted likelihood does not lend itself to separation by highest-level clusters.

EM iterations always assume equal weighting and an independent, homoskedastic error structure. As such, with weighted data or when error structures are more complex, EM is used only to obtain starting values.

For extensions to models with three or more levels, see Bates and Pinheiro (1998) and Rabe-Hesketh and Skrondal (2006).

**Denominator degrees of freedom**

When the `dfmethod()` option is specified, `mixed` uses a \( t \) distribution with \( \nu_{ddf} \) degrees of freedom to perform single-hypothesis tests for fixed effects \( H_0: \beta_i = 0 \) for \( i = 1, 2, \ldots, p \) or an \( F \) distribution with model numerator degrees of freedom and \( \nu_{ddf,m} \) DDF for a model (joint) test of all coefficients (except the constant) being equal to zero. Denominator degrees of freedom \( \nu_{ddf} \) and \( \nu_{ddf,m} \) are computed according to the specified DDF method.

**Residual DDF**

This method uses the residual degrees of freedom as the DDF, \( \nu_{ddf} = n - p \), where \( n \) is the total number of observations, and \( p \) is the rank of the design matrix \( X \).
Repeated DDF

This method partitions the residual degrees of freedom into the between-subject degrees of freedom and the within-subject degrees of freedom. This partitioning of the degrees of freedom arises from balanced repeated-measures ANOVA analysis. If levels of a fixed effect change within a subject, then the within-subject degrees of freedom is assigned to the fixed effect of interest; otherwise, the between-subject degrees of freedom is assigned to that fixed effect. See Schluchter and Elashoff (1990) for more computational details and, specifically, for the expressions of between-subject and within-subject degrees of freedom.

ANOVA DDF

This method determines the DDF for a fixed effect depending on whether the corresponding covariate is contained in any of the random-effects equations. If the covariate is contained in a random-effects equation, the DDF $\nu_{ddf}$ for the fixed effect is computed as the number of levels of the level variable from that equation minus one. If the covariate is specified in more than one random-effects equation, the DDF $\nu_{ddf}$ for the fixed effect is computed as the smallest number of levels of the level variables from those equations minus one and is a conservative estimate of the true DDF. If the covariate is specified only in the fixed-effects equation, the DDF is computed as $\nu_{ddf} = n - \text{rank}(X, Z)$.

For example, suppose we have the following mixed model,

$$\text{mixed } y \ A \ B \ C \ | \ | \ D: A \ | \ | E: A \ B$$

where A, B, and C are fixed effects, and D and E are nested random effects. For the fixed effect A, $\nu_{ddf}$ is the smaller number of levels of variables D and E minus one because A is included in random-effects equations at both levels D and E. For the fixed effect B, $\nu_{ddf}$ is the number of levels of variable E minus one because B is included in the random-effects equation at the level E. For the fixed effect C, $\nu_{ddf} = n - \text{rank}(X, Z)$ because C is not included in any of the random-effects equations.

For the three methods above, the DDF for a model test of $H_0: \beta = 0$ is computed as follows. If all corresponding single-hypothesis tests $H_0: \beta_i = 0$ have the same DDF $\nu_{ddf}$, then model DDF $\nu_{ddf_m} = \nu_{ddf}$. If the single-hypothesis DDF differs, then $\nu_{ddf_m}$ is not defined, and the large-sample $\chi^2$ test is reported instead of the $F$ test.

To provide formulas for the Satterthwaite and Kenward–Roger methods, consider a general linear-hypotheses test of fixed effects $H_0: C^\prime \beta = b$ with a $p \times l$ matrix of linear hypotheses $C$ of rank $l$.

The variance–covariance matrix of $y$ is $\text{Var}(y) = V = ZGZ^\prime + R = V(\sigma)$ and can be viewed as a function of variance components $\sigma (r \times 1)$. Suppose that the first two partial derivatives of $V(\sigma)$ with respect to $\sigma$ exist.

Let $\hat{\sigma}$ be the REML estimator of $\sigma$. Then, the REML estimator of the fixed effects $\beta$ is the generalized least-squares estimator

$$\hat{\beta} = \{X^\prime V^{-1}(\hat{\sigma})X\}^{-1} X^\prime V^{-1}(\hat{\sigma})Y$$

where $\text{Var}(\hat{\beta}) = \hat{\Phi} = \Phi(\hat{\sigma}) = \{X^\prime V^{-1}(\hat{\sigma})X\}^{-1}$ is the conventional estimator of the variance–covariance matrix of the fixed effects $\hat{\beta}$, and $V(\hat{\sigma})$ is the estimator of the covariance matrix of $y$.

Under the null $H_0: C^\prime \beta = b$, the $F$ test statistic is

$$F = \frac{l}{l} (C^\prime \hat{\beta} - b)'(C^\prime \hat{\Phi} C)^{-1}(C^\prime \beta - b)$$

and it has an $F$ distribution with $l$ numerator and $\nu_{ddf_{C}}$ DDF.
Satterthwaite DDF

This method is derived from the DDF formula of the original approximation attributable to Satterthwaite (1946):

$$ \text{ddf} = \frac{2(C'\hat{\Phi}C)^2}{\text{Var}(C'\hat{\Phi}C)} $$

For a single-hypothesis test of $H_0: c'\beta = b$, where $c$ and $b$ are vectors of known constants, Giesbrecht and Burns (1985) proposed using

$$ \nu_{ddf} = \frac{2(c'\hat{\Phi}c)^2}{\text{Var}(c'\hat{\Phi}c)} = \frac{2(c'\hat{\Phi}c)^2}{d'Wd} \tag{15} $$

where $d$ is a vector of partial derivatives of $c'\hat{\Phi}(\sigma)c$ with respect to $\sigma$ evaluated at $\hat{\sigma}$, and $\hat{\text{Var}}(\hat{\sigma}) = W$ is the estimator of the variance–covariance matrix of $\hat{\sigma}$ computed based on the expected information matrix $I_E$ in (17) or on the observed information matrix if suboption $oim$ of $\text{dfmethod()}$ is specified.

For a multiple-hypotheses test (when the rank of $C$ is greater than 1), Fai and Cornelius (1996) proposed an extension of the Giesbrecht–Burns single-degree-of-freedom method. Their method involves the spectral decomposition $C'\hat{\Phi}C = P'DP$, where $P = (p_1, p_2, \ldots, p_l)$ is an orthogonal matrix of eigenvectors, and $D = \text{diag}(\lambda_1, \lambda_2, \ldots, \lambda_l)$ is a diagonal matrix of the corresponding eigenvalues. Using this decomposition, we can write the $F$-test statistic as a sum of $l$ independent approximate $t$ random variates, $F = Q/l$ with

$$ Q = \sum_{k=1}^{l} \frac{\{p_k'(C'\hat{\beta} - b)\}^2}{\lambda_k} = \sum_{k=1}^{l} t_{v_k}^2 $$

where $v_k$ is computed using (15). Because $t_{v_k}$ s are independent and have approximate $t$ distributions with $v_k$ degrees of freedom,

$$ E(Q) = \sum_{k=1}^{l} \frac{v_k}{v_k - 2} I(v_k > 2) $$

Then, the DDF for a multiple-hypotheses test can be approximately written as

$$ \nu_{ddf_C} = \frac{2E(Q)}{E(Q) - l} $$

For more computational details of the Satterthwaite method, see Fai and Cornelius (1996).

Kenward–Roger DDF

This method was developed by Kenward and Roger (1997). It is based on adjusting the conventional variance–covariance estimator of fixed effects $\hat{\Phi}$ for small-sample bias and introducing a scaled $F$ test that improves the small-sample performance of the conventional $F$ test of fixed effects.
Kenward and Roger (1997) propose the adjusted estimator,

$$\hat{\Phi}_A = \hat{\Phi} + 2\hat{\Phi} \left\{ \sum_{i=1}^{r} \sum_{j=1}^{r} W_{ij} (Q_{ij} - P_i \hat{\Phi} P_j - \frac{1}{4} R_{ij}) \right\} \hat{\Phi} \quad (16)$$

where $P_i = X'\{\frac{\partial V^{-1}(\sigma)}{\partial \sigma_i}\}X$, $Q_{ij} = X'\{\frac{\partial V^{-1}(\sigma)}{\partial \sigma_i}\}V(\sigma)\{\frac{\partial V^{-1}(\sigma)}{\partial \sigma_j}\}X$ evaluated at $\hat{\sigma}$ and $W_{ij}$ is the $(i,j)$th element of $W$, the estimator of the variance–covariance matrix of $\hat{\sigma}$ computed from the inverse of the expected information matrix $I_E$, where the element $I_E^{ij}$ is defined as

$$2I_E^{ij} = \text{tr} \left( \frac{\partial \Sigma^{-1}}{\partial \sigma_i} \Sigma \frac{\partial \Sigma^{-1}}{\partial \sigma_j} \Sigma \right) - \text{tr}(2\Phi Q_{ij} - \Phi P_i \Phi P_j) \quad (17)$$

Alternatively, you can use the observed information matrix as $W$ by specifying suboption oim in dfmethod().

All terms in (16), except those involving $R_{ij}$, are invariant under reparameterization of the covariance structures. Also, the second derivative requires more computational resources and may not be numerically stable. For these reasons, the $R_{ij}$ terms are ignored in the computation of $\hat{\Phi}_A$ in (16).

For multiple-hypotheses testing, Kenward and Roger (1997) propose the scaled $F$-test statistic, which under the null hypothesis can be written as

$$F_{KR} = \frac{\lambda}{l} (C'\hat{\beta} - b)'(C'\hat{\Phi}_A C)^{-1}(C'\hat{\beta} - b)$$

and has an $F$ distribution with $l$ numerator and $\nu_{ddfC}$ DDF. The scale factor $\lambda = \nu_{ddfC} / (l-1+\nu_{ddfC})$.

The DDF $\nu_{ddfC}$ and $\lambda$ are approximated as

$$\nu_{ddfC} = 4 + \frac{l+2}{l \times \rho - 1} \quad \text{and} \quad \lambda = \frac{\nu_{ddfC}}{E^*(\nu_{ddfC} - 2)}$$

where $\rho = V^*/(E^*)^2$ and $E^*$ and $V^*$ are the respective approximate mean and variance of the $F_{KR}$ statistic; see Kenward and Roger (1997, 987) for expressions for $E^*$ and $V^*$.

**Acknowledgments**

We thank Badi Baltagi of the Department of Economics at Syracuse University and Ray Carroll of the Department of Statistics at Texas A&M University for each providing us with a dataset used in this entry.

We also thank Mike Kenward of the Medical Statistics Unit at the London School of Hygiene and Tropical Medicine and James Roger of the Research Statistics Unit at GlaxoSmithKline for answering our questions about their methods.
Charles Roy Henderson (1911–1989) was born in Iowa and grew up on the family farm. His education in animal husbandry, animal nutrition, and statistics at Iowa State was interspersed with jobs in the Iowa Extension Service, Ohio University, and the U.S. Army. After completing his PhD, Henderson joined the Animal Science faculty at Cornell. He developed and applied statistical methods in the improvement of farm livestock productivity through genetic selection, with particular focus on dairy cattle. His methods are general and have been used worldwide in livestock breeding and beyond agriculture. Henderson’s work on variance components and best linear unbiased predictions has proved to be one of the main roots of current mixed-model methods.

References


### Also see

[ME] **mixed postestimation** — Postestimation tools for mixed

[ME] **me** — Introduction to multilevel mixed-effects models

[MI] **estimation** — Estimation commands for use with mi estimate

[SEM] **intro 5** — Tour of models (*Multilevel mixed-effects models*)

[XT] **xtreg** — Random-coefficients model

[XT] **xtreg** — Fixed-, between-, and random-effects and population-averaged linear models

[U] **20 Estimation and postestimation commands**
### Postestimation commands

The following postestimation commands are of special interest after `mixed`:

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>estat df</code></td>
<td>calculate and display degrees of freedom for fixed effects</td>
</tr>
<tr>
<td><code>estat group</code></td>
<td>summarize the composition of the nested groups</td>
</tr>
<tr>
<td><code>estat icc</code></td>
<td>estimate intraclass correlations</td>
</tr>
<tr>
<td><code>estat recovariance</code></td>
<td>display the estimated random-effects covariance matrix (or matrices)</td>
</tr>
<tr>
<td><code>estat wcorrelation</code></td>
<td>display model-implied within-cluster correlations and standard deviations</td>
</tr>
</tbody>
</table>

The following standard postestimation commands are also available:

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>contrast</code></td>
<td>contrasts and ANOVA-style joint tests of estimates</td>
</tr>
<tr>
<td><code>estat ic</code></td>
<td>Akaike’s and Schwarz’s Bayesian information criteria (AIC and BIC)</td>
</tr>
<tr>
<td><code>estat summarize</code></td>
<td>summary statistics for the estimation sample</td>
</tr>
<tr>
<td><code>estat vce</code></td>
<td>variance–covariance matrix of the estimators (VCE)</td>
</tr>
<tr>
<td><code>estimates</code></td>
<td>cataloging estimation results</td>
</tr>
<tr>
<td><code>hausman</code></td>
<td>Hausman’s specification test</td>
</tr>
<tr>
<td><code>lincom</code></td>
<td>point estimates, standard errors, testing, and inference for linear combinations of coefficients</td>
</tr>
<tr>
<td><code>lrtest</code></td>
<td>likelihood-ratio test</td>
</tr>
<tr>
<td><code>margins</code></td>
<td>marginal means, predictive margins, marginal effects, and average marginal effects</td>
</tr>
<tr>
<td><code>marginsplot</code></td>
<td>graph the results from margins (profile plots, interaction plots, etc.)</td>
</tr>
<tr>
<td><code>nlcom</code></td>
<td>point estimates, standard errors, testing, and inference for nonlinear combinations of coefficients</td>
</tr>
<tr>
<td><code>predict</code></td>
<td>predictions, residuals, influence statistics, and other diagnostic measures</td>
</tr>
<tr>
<td><code>predictnl</code></td>
<td>point estimates, standard errors, testing, and inference for generalized predictions</td>
</tr>
<tr>
<td><code>pwcompare</code></td>
<td>pairwise comparisons of estimates</td>
</tr>
<tr>
<td><code>test</code></td>
<td>Wald tests of simple and composite linear hypotheses</td>
</tr>
<tr>
<td><code>testnl</code></td>
<td>Wald tests of nonlinear hypotheses</td>
</tr>
</tbody>
</table>
predict

Description for predict

predict creates a new variable containing predictions such as linear predictions, standard errors, fitted values, residuals, and standardized residuals.

Menu for predict

Statistics > Postestimation

Syntax for predict

Syntax for obtaining BLUPs of random effects and the BLUPs’ standard errors

```
predict [type] newvarsspec [if] [in], reffects [reses(newvarsspec)] relevel(levelvar)
```

Syntax for obtaining scores after ML estimation

```
predict [type] newvarsspec [if] [in], scores
```

Syntax for obtaining other predictions

```
predict [type] newvar [if] [in], statistic relevel(levelvar)
```

newvarsspec is stub* or newvarlist.

<table>
<thead>
<tr>
<th>statistic</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Main</td>
<td></td>
</tr>
<tr>
<td>xb</td>
<td>linear prediction for the fixed portion of the model only; the default</td>
</tr>
<tr>
<td>stdp</td>
<td>standard error of the fixed-portion linear prediction</td>
</tr>
<tr>
<td>fitted</td>
<td>fitted values, fixed-portion linear prediction plus contributions based on</td>
</tr>
<tr>
<td>residuals</td>
<td>predicted random effects</td>
</tr>
<tr>
<td>*rstandard</td>
<td>standardized residuals</td>
</tr>
</tbody>
</table>

Unstarred statistics are available both in and out of sample; type `predict ... if e(sample) ...` if wanted only for the estimation sample. Starred statistics are calculated only for the estimation sample, even when `if e(sample)` is not specified.
Options for predict

xb, the default, calculates the linear prediction $x\beta$ based on the estimated fixed effects (coefficients) in the model. This is equivalent to fixing all random effects in the model to their theoretical mean value of 0.

stdp calculates the standard error of the linear predictor $x\beta$.

reffects calculates best linear unbiased predictions (BLUPs) of the random effects. By default, BLUPs for all random effects in the model are calculated. However, if the relevel(levelvar) option is specified, then BLUPs for only level levelvar in the model are calculated. For example, if classes are nested within schools, then typing

```
. predict b*, reffects relevel(school)
```

would produce BLUPs at the school level. You must specify $q$ new variables, where $q$ is the number of random-effects terms in the model (or level). However, it is much easier to just specify stub* and let Stata name the variables stub1, stub2, ..., stub$q$ for you.

Rabe-Hesketh and Skrondal (2012, sec. 2.11.2) discuss the link between the empirical Bayes predictions and BLUPs and how these predictions are unbiased. They are unbiased when the groups associated with the random effects are expected to vary in repeated samples. If you expect the groups to be fixed in repeated samples, then these predictions are no longer unbiased.

reses(newvarsspec) calculates the standard errors of the BLUPs of the random effects. By default, standard errors for all BLUPs in the model are calculated. However, if the relevel(levelvar) option is specified, then standard errors for only level levelvar in the model are calculated; see the reffects option.

You must specify $q$ new variables, where $q$ is the number of random-effects terms in the model (or level). However, it is much easier to just specify stub* and let Stata name the variables stub1, stub2, ..., stub$q$ for you. The new variables will have the same storage type as the corresponding random-effects variables.

The reffects and reses() options often generate multiple new variables at once. When this occurs, the random effects (or standard errors) contained in the generated variables correspond to the order in which the variance components are listed in the output of mixed. Still, examining the variable labels of the generated variables (with the describe command, for instance) can be useful in deciphering which variables correspond to which terms in the model.

fitted calculates fitted values, which are equal to the fixed-portion linear predictor plus contributions based on predicted random effects, or in mixed-model notation, $x\beta + Zu$. By default, the fitted values take into account random effects from all levels in the model; however, if the relevel(levelvar) option is specified, then the fitted values are fit beginning with the topmost level down to and including level levelvar. For example, if classes are nested within schools, then typing

```
. predict yhat_school, fitted relevel(school)
```

would produce school-level predictions. That is, the predictions would incorporate school-specific random effects but not those for each class nested within each school.

residuals calculates residuals, equal to the responses minus fitted values. By default, the fitted values take into account random effects from all levels in the model; however, if the relevel(levelvar) option is specified, then the fitted values are fit beginning at the topmost level down to and including level levelvar.

rstandard calculates standardized residuals, equal to the residuals multiplied by the inverse square root of the estimated error covariance matrix.
scores calculates the parameter-level scores, one for each parameter in the model including regression coefficients and variance components. The score for a parameter is the first derivative of the log likelihood (or log pseudolikelihood) with respect to that parameter. One score per highest-level group is calculated, and it is placed on the last record within that group. Scores are calculated in the estimation metric as stored in \( e(b) \).

scores is not available after restricted maximum-likelihood (REML) estimation.

`relevel(levelvar)` specifies the level in the model at which predictions involving random effects are to be obtained; see the options above for the specifics. `levelvar` is the name of the model level and is either the name of the variable describing the grouping at that level or is `all`, a special designation for a group comprising all the estimation data.

---

**margins**

**Description for margins**

`margins` estimates margins of response for linear predictions.

**Menu for margins**

Statistics > Postestimation

**Syntax for margins**

\[
\text{margins } [marginlist] \ [, \ options] \\
\text{margins } [marginlist], \text{ predict(statistic ...)} [options]
\]

<table>
<thead>
<tr>
<th>statistic</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>xb</td>
<td>linear predictor for the fixed portion of the model only; the default</td>
</tr>
<tr>
<td>reffects</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>scores</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>stdp</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>fitted</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>residuals</td>
<td>not allowed with margins</td>
</tr>
<tr>
<td>standard</td>
<td>not allowed with margins</td>
</tr>
</tbody>
</table>

Statistics not allowed with `margins` are functions of stochastic quantities other than \( e(b) \).

For the full syntax, see \[R\] `margins`. 
estat

Description for estat

estat group reports the number of groups and minimum, average, and maximum group sizes for each level of the model. Model levels are identified by the corresponding group variable in the data. Because groups are treated as nested, the information in this summary may differ from what you would get if you used the tabulate command on each group variable individually.

estat icc displays the intraclass correlation for pairs of responses at each nested level of the model. Intraclass correlations are available for random-intercept models or for random-coefficient models conditional on random-effects covariates being equal to 0. They are not available for crossed-effects models or with residual error structures other than independent structures.

estat recovariance displays the estimated variance–covariance matrix of the random effects for each level in the model. Random effects can be either random intercepts, in which case the corresponding rows and columns of the matrix are labeled as _cons, or random coefficients, in which case the label is the name of the associated variable in the data.

estat wcorrelation displays the overall correlation matrix for a given cluster calculated on the basis of the design of the random effects and their assumed covariance and the correlation structure of the residuals. This allows for a comparison of different multilevel models in terms of the ultimate within-cluster correlation matrix that each model implies.

estat df calculates and displays the degrees of freedom (DF) for each fixed effect using the specified methods. This allows for a comparison of different DF methods. estat df can also be used to continue with postestimation using a different DF method without rerunning the mixed model, which is useful if one wants to change the current DF method after comparing methods.

Menu for estat

Statistics  >  Postestimation
Syntax for estat

Summarize the composition of the nested groups

\texttt{estat group}

Estimate intraclass correlations

\texttt{estat icc [ , level(#) ]}

Display the estimated random-effects covariance matrix (or matrices)

\texttt{estat recovariance [ , relevel(levelvar) correlation matlist\_options ]}

Display model-implied within-cluster correlations and standard deviations

\texttt{estat wcorrelation [ , wcor\_options ]}

Report or calculate degrees of freedom for fixed effects

\texttt{estat df [ , method(df\_methods) post[(df\_method)] eim oim]}

\begin{tabular}{ll}
\textbf{wcor\_options} & Description \\
\hline
\texttt{at(at\_spec)} & specify the cluster for which you want the correlation matrix; default is the first two-level cluster encountered in the data \\
\texttt{all} & display correlation matrix for all the data \\
\texttt{covariance} & display the covariance matrix instead of the correlation matrix \\
\texttt{list} & list the data corresponding to the correlation matrix \\
\texttt{nosort} & list the rows and columns of the correlation matrix in the order they were originally present in the data \\
\texttt{format(\%fmt)} & set the display format; default is \texttt{format(\%6.3f)} \\
\texttt{matlist\_options} & style and formatting options that control how matrices are displayed \\
\end{tabular}

Option for estat icc

\texttt{level(#)} specifies the confidence level, as a percentage, for confidence intervals. The default is \texttt{level(95)} or as set by \texttt{set level}; see \texttt{[U] 20.7 Specifying the width of confidence intervals}.

Options for estat recovariance

\texttt{relevel(levelvar)} specifies the level in the model for which the random-effects covariance matrix is to be displayed. By default, the covariance matrices for all levels in the model are displayed. \texttt{levelvar} is the name of the model level and is either the name of the variable describing the grouping at that level or is \texttt{all}, a special designation for a group comprising all the estimation data.

\texttt{correlation} displays the covariance matrix as a correlation matrix.

\texttt{matlist\_options} are style and formatting options that control how the matrix (or matrices) is displayed; see \texttt{[P] matlist} for a list of options that are available.
Options for estat wcorrelation

at(at_spec) specifies the cluster of observations for which you want the within-cluster correlation matrix. at_spec is

\[ \text{relevel}_\text{var} = \text{value} \left[ , \text{relevel}_\text{var} = \text{value} \ldots \right] \]

For example, if you specify

```
. estat wcorrelation, at(school = 33)
```

you get the within-cluster correlation matrix for those observations in school 33. If you specify

```
. estat wcorrelation, at(school = 33 classroom = 4)
```

you get the correlation matrix for classroom 4 in school 33.

If at() is not specified, then you get the correlations for the first level-two cluster encountered in the data. This is usually what you want.

all specifies that you want the correlation matrix for all the data. This is not recommended unless you have a relatively small dataset or you enjoy seeing large \( N \times N \) matrices. However, this can prove useful in some cases.

covariance specifies that the within-cluster covariance matrix be displayed instead of the default correlations and standard deviations.

list lists the model data for those observations depicted in the displayed correlation matrix. This option is useful if you have many random-effects design variables and you wish to see the represented values of these design variables.

nosort lists the rows and columns of the correlation matrix in the order that they were originally present in the data. Normally, estat wcorrelation will first sort the data according to level variables, by-group variables, and time variables to produce correlation matrices whose rows and columns follow a natural ordering. nosort suppresses this.

format(\%fmt) sets the display format for the standard-deviation vector and correlation matrix. The default is format(\%6.3f).

matlist_options are style and formatting options that control how the matrix (or matrices) is displayed; see [P] matlist for a list of options that are available.

Options for estat df

method(df_methods) specifies a list of methods to compute DF. The supported methods are residual, repeated, anova, satterthwaite, and kroger; more than one method may be specified. Methods satterthwaite and kroger are only available with REML estimation. If option dfmethod() was not specified in the most recently fit mixed model, then option method() is required. See Small-sample inference for fixed effects under Remarks and examples in [ME] mixed for more details.

post causes estat df to behave like a Stata estimation command. When post is specified, estat df will post the DF for each fixed effect as well as everything related to the DF computation to e() for the method specified in method(). Thus, after posting, you could continue to use this DF for other postestimation commands. For example, you could use test, small to perform Wald \( F \) tests on linear combination of the fixed effects.

post may also be specified using the syntax post(df_method). You must use this syntax if you specify multiple df_methods in option method(). With this syntax, estat df computes the DF
using the method specified in \texttt{post()} and stores the results in \texttt{e()}. Only one computation method may be specified using the syntax \texttt{post()}.

The \texttt{df\_method} specified in \texttt{post()} must be one of the DF methods specified in option \texttt{method()}. If only one method is specified in option \texttt{method()}, then one can simply use \texttt{post} to make this DF method active for postestimation and for \texttt{mixed} replay.

\texttt{eim} specifies that the expected information matrix be used in the DF computation. It can be used only when \texttt{method()} contains \texttt{kroger} or \texttt{satterthwaite}. \texttt{eim} is the default and may not be specified with \texttt{oim}.

\texttt{oim} specifies that the observed information matrix be used in the DF computation. It can be used only when \texttt{method()} contains \texttt{kroger} or \texttt{satterthwaite} and may not be specified with \texttt{eim}.

**test and testparm**

**Description for test and testparm**

\texttt{test} and \texttt{testparm}, by default, perform $\chi^2$ tests of simple and composite linear hypotheses about the parameters for the most recently fit \texttt{mixed} model. They also support $F$ tests with a small-sample adjustment for fixed effects.

**Menu for test and testparm**

\texttt{Statistics > Postestimation}

**Syntax for test and testparm**

\begin{quote}
\texttt{test (spec) [(spec) ...] [, test\_options small]}
\end{quote}

\begin{quote}
\texttt{testparm varlist[, testparm\_options small]}
\end{quote}

**Options for test and testparm**

\texttt{test\_options}; see \cite{R} \texttt{test} options. Options \texttt{df()}, \texttt{common}, and \texttt{nosvyadjust} may not be specified together with \texttt{small}.

\texttt{testparm\_options}; see options of \texttt{testparm} in \cite{R} \texttt{test}. Options \texttt{df()} and \texttt{nosvyadjust} may not be specified together with \texttt{small}.

\texttt{small} specifies that $F$ tests for fixed effects be carried out with the denominator degrees of freedom (DDF) obtained by the same method used in the most recently fit \texttt{mixed} model. If option \texttt{dfmethod()} is not specified in the previous \texttt{mixed} command, option \texttt{small} is not allowed. For certain methods, the DDF for some tests may not be available. See \cite{Small-sample inference for fixed effects} in \cite{ME} \texttt{mixed} for more details.
**lincom**

**Description for lincom**

`lincom`, by default, computes point estimates, standard errors, $z$ statistics, $p$-values, and confidence intervals for linear combinations of coefficients after `mixed`. `lincom` also provides $t$ statistics for linear combinations of the fixed effects, with the degrees of freedom calculated by the DF method specified in option `dfmethod()` of `mixed`.

**Menu for lincom**

Statistics > Postestimation

**Syntax for lincom**

```
lincom exp [ , lincom_options small]
```

**Options for lincom**

`lincom_options`; see `[R] lincom options`. Option `df()` may not be specified together with `small`. `small` specifies that $t$ statistics for linear combinations of fixed effects be displayed with the degrees of freedom obtained by the same method used in the most recently fit `mixed` model. If option `dfmethod()` is not specified in the previous `mixed` command, option `small` is not allowed. For certain methods, the degrees of freedom for some linear combinations may not be available. See *Small-sample inference for fixed effects* in `[ME] mixed* for more details.

**contrast**

**Description for contrast**

`contrast`, by default, performs $\chi^2$ tests of linear hypotheses and forms contrasts involving factor variables and their interactions for the most recently fit `mixed` model. `contrast` also supports tests with small-sample adjustments after `mixed, dfmethod()`.

**Menu for contrast**

Statistics > Postestimation

**Syntax for contrast**

```
contrast termlist [ , contrast_options small]
```
Options for contrast

_postestimation contrast_options_; see [R] contrast options. Options df() and nosvyadjust may not be specified together with small.

small specifies that tests for contrasts be carried out with the DDF obtained by the same method used in the most recently fit mixed model. If option dfmethod() is not specified in the previous mixed command, option small is not allowed. For certain methods, the DDF for some contrasts may not be available. See Small-sample inference for fixed effects in [ME] mixed for more details.

pwcompare

Description for pwcompare

pwcompare performs pairwise comparisons across the levels of factor variables from the most recently fit mixed model. pwcompare, by default, reports the comparisons as contrasts (differences) of margins along with z tests or confidence intervals for the pairwise comparisons. pwcompare also supports t tests with small-sample adjustments after mixed, dfmethod().

Menu for pwcompare

Statistics > Postestimation

Syntax for pwcompare

pwcompare marginlist [ , pwcompare_options small ]

Options for pwcompare

pwcompare_options; see [R] pwcompare options. Option df() may not be specified together with small.

small specifies that t tests for pairwise comparisons be carried out with the degrees of freedom obtained by the same method used in the most recently fit mixed model with the dfmethod() option. If option dfmethod() is not specified in the previous mixed command, option small is not allowed. For certain methods, the degrees of freedom for some pairwise comparisons may not be available. See Small-sample inference for fixed effects in [ME] mixed for more details.

Remarks and examples

Various predictions, statistics, and diagnostic measures are available after fitting a mixed model using mixed. For the most part, calculation centers around obtaining BLUPs of the random effects. Random effects are not estimated when the model is fit but instead need to be predicted after estimation. Calculation of intraclass correlations, estimating the dependence between responses for different levels of nesting, may also be of interest.
Example 1

In example 3 of [ME] mixed, we modeled the weights of 48 pigs measured on nine successive weeks as

$$weight_{ij} = \beta_0 + \beta_1 \text{week}_{ij} + u_{0j} + u_{1j} \text{week}_{ij} + \epsilon_{ij}$$  \hspace{1cm} (1)

for $i = 1, \ldots, 9$, $j = 1, \ldots, 48$, $\epsilon_{ij} \sim N(0, \sigma^2_{\epsilon})$, and $u_{0j}$ and $u_{1j}$ normally distributed with mean 0 and variance–covariance matrix

$$\Sigma = \text{Var} \begin{bmatrix} u_{0j} \\ u_{1j} \end{bmatrix} = \begin{bmatrix} \sigma^2_{u0} & \sigma_{01} \\ \sigma_{01} & \sigma^2_{u1} \end{bmatrix}$$

```
use http://www.stata-press.com/data/r14/pig
(Longitudinal analysis of pig weights)
```

. mixed weight week || id: week, covariance(unstructured)
Performing EM optimization:
Performing gradient-based optimization:
Iteration 0: log likelihood = -868.96185
Iteration 1: log likelihood = -868.96185
Computing standard errors:
Mixed-effects ML regression
Group variable: id
Number of obs = 432
Number of groups = 48
Obs per group:
  min = 9
  avg = 9.0
  max = 9
Log likelihood = -868.96185
Wald chi2(1) = 4649.17
Prob > chi2 = 0.0000

| weight | Coef.  | Std. Err. | z     | P>|z|  | [95% Conf. Interval] |
|--------|--------|-----------|-------|------|---------------------|
| week   | 6.209896 | .0910745 | 68.18 | 0.000 | 6.031393  6.388399 |
| _cons  | 19.35561 | .3996387 | 48.43 | 0.000 | 18.57234  20.13889 |

<table>
<thead>
<tr>
<th>Random-effects Parameters</th>
<th>Estimate</th>
<th>Std. Err.</th>
<th>[95% Conf. Interval]</th>
</tr>
</thead>
<tbody>
<tr>
<td>id: Unstructured</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>var(week)</td>
<td>.3715251</td>
<td>.0812958</td>
<td>.2419532  .570486</td>
</tr>
<tr>
<td>var(_cons)</td>
<td>6.823363</td>
<td>1.566194</td>
<td>4.351297  10.69986</td>
</tr>
<tr>
<td>cov(week,_cons)</td>
<td>-.0984378</td>
<td>.2545767</td>
<td>-.5973991  .4005234</td>
</tr>
<tr>
<td>var(Residual)</td>
<td>1.596829</td>
<td>.123198</td>
<td>1.372735  1.857505</td>
</tr>
</tbody>
</table>

LR test vs. linear model: chi2(3) = 764.58 Prob > chi2 = 0.0000
Note: LR test is conservative and provided only for reference.

Rather than see the estimated variance components listed as variance and covariances as above, we can instead see them as correlations and standard deviations in matrix form; that is, we can see $\hat{\Sigma}$ as a correlation matrix:
We can use `estat wcorrelation` to display the within-cluster marginal standard deviations and correlations for one of the clusters.

```
. estat wcorrelation, format(%4.2g)
Standard deviations and correlations for id = 1:
Standard deviations:
   obs  1  2  3  4  5  6  7  8  9
      sd  2.9 3.1 3.3 3.7 4.1 4.5 5  5.5 6.1
Correlations:
   obs  1  2  3  4  5  6  7  8  9
      1  1
      2  .8  1
      3  .77  .83  1
      4  .72  .81  .86  1
      5  .67  .78  .85  .89  1
      6  .63  .75  .83  .88  .91  1
      7  .59  .72  .81  .87  .91  .93  1
      8  .55  .69  .79  .86  .9  .93  .94  1
      9  .52  .66  .77  .85  .89  .92  .94  .95  1
```

Because within-cluster correlations can vary between clusters, `estat wcorrelation` by default displays the results for the first cluster. In this example, each cluster (pig) has the same number of observations, and the timings of measurements (`week`) are the same between clusters. Thus the within-cluster correlations are the same for all the clusters. In example 4, we fit a model where different clusters have different within-cluster correlations and show how to display these correlations.

We can also obtain BLUPs of the pig-level random effects (\(u_{0j}\) and \(u_{1j}\)). We need to specify the variables to be created in the order \(u_1 u_0\) because that is the order in which the corresponding variance components are listed in the output (`week _cons`). We obtain the predictions and list them for the first 10 pigs.

```
. predict u1 u0, reffects
. by id, sort: generate tolist = (_n==1)
. list id u0 u1 if id <=10 & tolist

<table>
<thead>
<tr>
<th>id</th>
<th>u0</th>
<th>u1</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>.2369444</td>
<td>-.3957636</td>
</tr>
<tr>
<td>10</td>
<td>-1.584127</td>
<td>.510038</td>
</tr>
<tr>
<td>19</td>
<td>-3.526551</td>
<td>.3200372</td>
</tr>
<tr>
<td>28</td>
<td>1.964378</td>
<td>-.7719702</td>
</tr>
<tr>
<td>37</td>
<td>1.299236</td>
<td>-.9241479</td>
</tr>
<tr>
<td>46</td>
<td>-1.147302</td>
<td>-.5448151</td>
</tr>
<tr>
<td>55</td>
<td>-2.590529</td>
<td>.0394454</td>
</tr>
<tr>
<td>64</td>
<td>-1.137067</td>
<td>-.1696566</td>
</tr>
<tr>
<td>73</td>
<td>-3.189545</td>
<td>-.7365507</td>
</tr>
<tr>
<td>82</td>
<td>1.160324</td>
<td>.0030772</td>
</tr>
</tbody>
</table>
```
mixed postestimation — Postestimation tools for mixed 435

If you forget how to order your variables in predict, or if you use predict stub*, remember that predict labels the generated variables for you to avoid confusion.

. describe u0 u1

<table>
<thead>
<tr>
<th>variable name</th>
<th>storage type</th>
<th>display format</th>
<th>value label</th>
</tr>
</thead>
<tbody>
<tr>
<td>u0</td>
<td>float</td>
<td>%9.0g</td>
<td>BLUP r.e. for id: _cons</td>
</tr>
<tr>
<td>u1</td>
<td>float</td>
<td>%9.0g</td>
<td>BLUP r.e. for id: week</td>
</tr>
</tbody>
</table>

Examining (1), we see that within each pig, the successive weight measurements are modeled as simple linear regression with intercept $\beta_0 + u_{j0}$ and slope $\beta_1 + u_{j1}$. We can generate estimates of the pig-level intercepts and slopes with

. generate intercept = _b[_cons] + u0
. generate slope = _b[week] + u1
. list id intercept slope if id<=10 & tolist

<table>
<thead>
<tr>
<th>id</th>
<th>intercept</th>
<th>slope</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>19.59256</td>
<td>5.814132</td>
</tr>
<tr>
<td>10</td>
<td>17.77149</td>
<td>6.719934</td>
</tr>
<tr>
<td>19</td>
<td>15.82906</td>
<td>6.529933</td>
</tr>
<tr>
<td>28</td>
<td>21.31999</td>
<td>5.437926</td>
</tr>
<tr>
<td>37</td>
<td>20.65485</td>
<td>5.285748</td>
</tr>
<tr>
<td>46</td>
<td>18.20831</td>
<td>5.665081</td>
</tr>
<tr>
<td>55</td>
<td>16.76509</td>
<td>6.249341</td>
</tr>
<tr>
<td>64</td>
<td>18.21855</td>
<td>6.040239</td>
</tr>
<tr>
<td>73</td>
<td>16.16607</td>
<td>5.473345</td>
</tr>
<tr>
<td>82</td>
<td>20.51594</td>
<td>6.212973</td>
</tr>
</tbody>
</table>

Thus we can plot estimated regression lines for each of the pigs. Equivalently, we can just plot the fitted values because they are based on both the fixed and the random effects:

. predict fitweight, fitted
. twoway connected fitweight week if id<=10, connect(L)

We can also generate standardized residuals and see whether they follow a standard normal distribution, as they should in any good-fitting model:
. predict rs, rstandard
. summarize rs

<table>
<thead>
<tr>
<th>Variable</th>
<th>Obs</th>
<th>Mean</th>
<th>Std. Dev.</th>
<th>Min</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>rs</td>
<td>432</td>
<td>1.01e-09</td>
<td>.8929356</td>
<td>-3.621446</td>
<td>3.000929</td>
</tr>
</tbody>
</table>

. qnorm rs

Example 2

Following Rabe-Hesketh and Skrondal (2012, chap. 2), we fit a two-level random-effects model for human peak-expiratory-flow rate. The subjects were each measured twice with the Mini-Wright peak-flow meter. It is of interest to determine how reliable the meter is as a measurement device. The intraclass correlation provides a measure of reliability. Formally, in a two-level random-effects model, the intraclass correlation corresponds to the correlation of measurements within the same individual and also to the proportion of variance explained by the individual random effect.
First, we fit the two-level model with mixed:

```
. use http://www.stata-press.com/data/r14/pefrate, clear
(Peak-expiratory-flow rate)
. mixed wm || id:
```

Performing EM optimization:
Performing gradient-based optimization:
Iteration 0: log likelihood = -184.57839
Iteration 1: log likelihood = -184.57839
Computing standard errors:

Mixed-effects ML regression
Group variable: id

<table>
<thead>
<tr>
<th></th>
<th>Number of obs = 34</th>
<th>Number of groups = 17</th>
</tr>
</thead>
<tbody>
<tr>
<td>Obs per group:</td>
<td>min = 2</td>
<td>avg = 2</td>
</tr>
<tr>
<td></td>
<td>max = 2</td>
<td></td>
</tr>
</tbody>
</table>

Log likelihood = -184.57839

| wm         | Coef. | Std. Err. | z     | P>|z|  | [95% Conf. Interval] |
|------------|-------|-----------|-------|------|---------------------|
| _cons      | 453.9118 | 26.18617 | 17.33 | 0.000 | 402.5878 505.2357   |

Random-effects Parameters

<table>
<thead>
<tr>
<th>id: Identity</th>
<th>Estimate</th>
<th>Std. Err.</th>
<th>[95% Conf. Interval]</th>
</tr>
</thead>
<tbody>
<tr>
<td>var(_cons)</td>
<td>11458.94</td>
<td>3998.952</td>
<td>5782.176 22708.98</td>
</tr>
<tr>
<td>var(Residual)</td>
<td>396.441</td>
<td>135.9781</td>
<td>202.4039 776.4942</td>
</tr>
</tbody>
</table>

LR test vs. linear model: chibar2(01) = 46.27  Prob >= chibar2 = 0.0000

Now we use estat icc to estimate the intraclass correlation:

```
. estat icc
Intraclass correlation

<table>
<thead>
<tr>
<th>Level</th>
<th>ICC</th>
<th>Std. Err.</th>
<th>[95% Conf. Interval]</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>.9665602</td>
<td>.0159495</td>
<td>.9165853  .9870185</td>
</tr>
</tbody>
</table>
```

This correlation is close to 1, indicating that the Mini-Wright peak-flow meter is reliable. But as noted by Rabe-Hesketh and Skrondal (2012), the reliability is not only a characteristic of the instrument but also of the between-subject variance. Here we see that the between-subject standard deviation, sd(_cons), is much larger than the within-subject standard deviation, sd(Residual).

In the presence of fixed-effects covariates, estat icc reports the residual intraclass correlation, the correlation between measurements conditional on the fixed-effects covariates. This is equivalent to the correlation of the model residuals.

In the presence of random-effects covariates, the intraclass correlation is no longer constant and depends on the values of the random-effects covariates. In this case, estat icc reports conditional intraclass correlations assuming 0 values for all random-effects covariates. For example, in a two-level model, this conditional correlation represents the correlation of the residuals for two measurements on the same subject, which both have random-effects covariates equal to 0. Similarly to the interpretation of intercept variances in random-coefficient models (Rabe-Hesketh and Skrondal 2012, chap. 4),
interpretation of this conditional intraclass correlation relies on the usefulness of the 0 baseline values of random-effects covariates. For example, mean centering of the covariates is often used to make a 0 value a useful reference.

Example 3

In example 4 of [ME] mixed, we estimated a Cobb–Douglas production function with random intercepts at the region level and at the state-within-region level:

\[
y_{jk} = X_{jk}\beta + u^{(3)}_k + u^{(2)}_{jk} + \epsilon_{jk}
\]

\[
. \text{use http://www.stata-press.com/data/r14/productivity}
\]
\[
. \text{(Public Capital Productivity)}
\]
\[
. \text{mixed gsp private emp hwy water other unemp || region: || state: (output omitted)}
\]

We can use \texttt{estat group} to see how the data are broken down by state and region:

\[
. \text{estat group}
\]

<table>
<thead>
<tr>
<th>Group Variable</th>
<th>No. of Groups</th>
<th>Observations per Group</th>
</tr>
</thead>
<tbody>
<tr>
<td>region</td>
<td>9</td>
<td>51</td>
</tr>
<tr>
<td>state</td>
<td>48</td>
<td>17</td>
</tr>
</tbody>
</table>

We are reminded that we have balanced productivity data for 17 years for each state.

We can use \texttt{predict, fitted} to get the fitted values

\[
\hat{y}_{jk} = X_{jk}\hat{\beta} + \hat{u}^{(3)}_k + \hat{u}^{(2)}_{jk}
\]

but if we instead want fitted values at the region level, that is,

\[
\hat{y}_{jk} = X_{jk}\hat{\beta} + \hat{u}^{(3)}_k
\]

we need to use the \texttt{relevel()} option:

\[
. \text{predict gsp_region, fitted relevel(region)}
\]
\[
. \text{list gsp gsp_region in 1/10}
\]

<table>
<thead>
<tr>
<th>gsp</th>
<th>gsp_region</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>10.25478</td>
</tr>
<tr>
<td>2</td>
<td>10.2879</td>
</tr>
<tr>
<td>3</td>
<td>10.35147</td>
</tr>
<tr>
<td>4</td>
<td>10.41721</td>
</tr>
<tr>
<td>5</td>
<td>10.42671</td>
</tr>
<tr>
<td>6</td>
<td>10.4224</td>
</tr>
<tr>
<td>7</td>
<td>10.4847</td>
</tr>
<tr>
<td>8</td>
<td>10.53111</td>
</tr>
<tr>
<td>9</td>
<td>10.59573</td>
</tr>
<tr>
<td>10</td>
<td>10.62082</td>
</tr>
</tbody>
</table>
Technical note

Out-of-sample predictions are permitted after mixed, but if these predictions involve BLUPs of random effects, the integrity of the estimation data must be preserved. If the estimation data have changed since the mixed model was fit, predict will be unable to obtain predicted random effects that are appropriate for the fitted model and will give an error. Thus to obtain out-of-sample predictions that contain random-effects terms, be sure that the data for these predictions are in observations that augment the estimation data.

We can use estat icc to estimate residual intraclass correlations between productivity years in the same region and in the same state and region.

```
estat icc
Residual intraclass correlation
```

<table>
<thead>
<tr>
<th>Level</th>
<th>ICC</th>
<th>Std. Err.</th>
<th>[95% Conf. Interval]</th>
</tr>
</thead>
<tbody>
<tr>
<td>region</td>
<td>.159893</td>
<td>.127627</td>
<td>.0287143 .5506202</td>
</tr>
<tr>
<td>state</td>
<td>region</td>
<td>.8516265</td>
<td>.0301733</td>
</tr>
</tbody>
</table>

estat icc reports two intraclass correlations for this three-level nested model. The first is the level-3 intraclass correlation at the region level, the correlation between productivity years in the same region. The second is the level-2 intraclass correlation at the state-within-region level, the correlation between productivity years in the same state and region.

Conditional on the fixed-effects covariates, we find that annual productivity is only slightly correlated within the same region, but it is highly correlated within the same state and region. We estimate that state and region random effects compose approximately 85% of the total residual variance.

Example 4

In example 1, we fit a model where each cluster had the same model-implied within-cluster correlations. Here we fit a model where different clusters have different within-cluster correlations, and we show how to display them for different clusters. We use the Asian children weight data from example 6 of [ME] mixed.
mixed postestimation — Postestimation tools for mixed

. use http://www.stata-press.com/data/r14/childweight, clear
(Weight data on Asian children)
. mixed weight age || id: age, covariance(unstructured)
Performing EM optimization:
Performing gradient-based optimization:
Iteration 0: log likelihood = -344.37065
Iteration 1: log likelihood = -342.83887
Iteration 2: log likelihood = -342.71863
Iteration 3: log likelihood = -342.71777
Iteration 4: log likelihood = -342.71777
Computing standard errors:
Mixed-effects ML regression
Group variable: id
Obs per group:
  min = 1
  avg = 2.9
  max = 5
Wald chi2(1) = 755.27
Log likelihood = -342.71777
Prob > chi2 = 0.0000

|    | Coef.  | Std. Err. |   z  | P>|z| | [95% Conf. Interval] |
|----|--------|-----------|------|-----|---------------------|
| age| 3.459671 | .1258877  | 27.48| 0.000 | 3.212936 3.706406  |
| _cons| 5.110496 | .1494781  | 34.19| 0.000 | 4.817524 5.403468  |

Random-effects Parameters

<table>
<thead>
<tr>
<th></th>
<th>Estimate</th>
<th>Std. Err.</th>
<th>[95% Conf. Interval]</th>
</tr>
</thead>
<tbody>
<tr>
<td>id: Unstructured</td>
<td>var(age)</td>
<td>.202392</td>
<td>.1242868</td>
</tr>
<tr>
<td></td>
<td>var(_cons)</td>
<td>.0970272</td>
<td>.1107998</td>
</tr>
<tr>
<td></td>
<td>cov(age,_cons)</td>
<td>.140134</td>
<td>.0566901</td>
</tr>
<tr>
<td></td>
<td>var(Residual)</td>
<td>1.357922</td>
<td>.1650502</td>
</tr>
</tbody>
</table>

LR test vs. linear model: chi2(3) = 27.38
Prob > chi2 = 0.0000
Note: LR test is conservative and provided only for reference.
We use `estat wcorrelation` to display the within-cluster correlations for the first cluster.

```
estat wcorrelation, list
```

**Standard deviations and correlations for id = 45:**

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>sd</strong></td>
<td>1.224</td>
<td>1.314</td>
<td>1.448</td>
<td>1.506</td>
<td>1.771</td>
</tr>
</tbody>
</table>

**Correlations:**

```
<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1.000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0.141</td>
<td>1.000</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>0.181</td>
<td>0.274</td>
<td>1.000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>0.193</td>
<td>0.293</td>
<td>0.376</td>
<td>1.000</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>0.230</td>
<td>0.348</td>
<td>0.447</td>
<td>0.477</td>
<td>1.000</td>
</tr>
</tbody>
</table>
```

**Data:**

```
id  weight  age
1. 45  5.171  .136893
2. 45  10.86  .657084
3. 45  13.15  1.21834
4. 45  13.2  1.42916
5. 45  15.88  2.27242
```

We specified the `list` option to display the data associated with the cluster. The next cluster in the dataset has ID 258. To display the within-cluster correlations for this cluster, we specify the `at()` option.

```
estat wcorrelation, at(id=258) list
```

**Standard deviations and correlations for id = 258:**

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>sd</strong></td>
<td>1.231</td>
<td>1.320</td>
<td>1.424</td>
<td>1.782</td>
</tr>
</tbody>
</table>

**Correlations:**

```
<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1.000</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0.152</td>
<td>1.000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>0.186</td>
<td>0.270</td>
<td>1.000</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>0.244</td>
<td>0.356</td>
<td>0.435</td>
<td>1.000</td>
</tr>
</tbody>
</table>
```

**Data:**

```
id  weight  age
1. 258  5.3  .19165
2. 258  9.74  .687201
3. 258  9.98  1.12799
4. 258  11.34  2.30527
```

The within-cluster correlations for this model depend on age. The values for `age` in the two clusters are different, as are the corresponding within-cluster correlations.
Example 5

To illustrate the use of estat df, we refit the dental veneer data from example 14 of [ME] mixed using the Kenward–Roger method (option dfmethod(kroger)) to compute the DF for fixed effects.

```
use http://www.stata-press.com/data/r14/veneer, clear
(Dental veneer data)
mixed gcf followup base_gcf cda age || patient: followup, cov(un)
> || tooth:, reml nolog dfmethod(kroger)
```

Mixed-effects REML regression Number of obs = 110

No. of Observations per Group

<table>
<thead>
<tr>
<th>Group Variable</th>
<th>No. of Groups</th>
<th>Observations per Group</th>
<th>Minimum</th>
<th>Average</th>
<th>Maximum</th>
</tr>
</thead>
<tbody>
<tr>
<td>patient</td>
<td>12</td>
<td>2</td>
<td>9.2</td>
<td>12</td>
<td></td>
</tr>
<tr>
<td>tooth</td>
<td>55</td>
<td>2</td>
<td>2.0</td>
<td>2</td>
<td></td>
</tr>
</tbody>
</table>

DF method: Kenward-Roger

DF:            
min = 10.41
avg = 28.96
max = 50.71
F(4,  27.96) = 1.47
Log restricted-likelihood = -420.92761 Prob > F = 0.2370

gcf | Coef. Std. Err. t P>|t| [95% Conf. Interval]
---- | -------- -------- -------- -------- ------------------
followup  |  0.3009815  1.938641 0.16 0.879 -3.96767  4.569633
base_gcf |  -0.0183127  0.146626  0.12 0.901 -0.313241  0.276616
   cda |  -0.329303  0.553350  0.60 0.554 -1.440355  0.781749
  age |  -0.5773932  0.235049  2.46 0.033 -1.098324  0.056462
   _cons |  45.73862  13.21824  3.46 0.002  18.53866  72.93858

Random-effects Parameters | Estimate Std. Err. [95% Conf. Interval]
--------------------------- | -------- -------- ------------------
patient: Unstructured
var(followup) |  41.88772  18.79997  17.38009  100.9535
var(_cons) |  524.9851  253.0205  204.1287  1350.175
cov(followup,_cons) |  -140.4229  66.57623 -270.9099  -9.935907

tooth: Identity
var(_cons) |  47.45738  16.63034  23.8792  94.3165

LR test vs. linear model: chi2(4) = 91.12 Prob > chi2 = 0.0000

Note: LR test is conservative and provided only for reference.

Rather than specifying option dftable(pvalue) or dftable(ci) at estimation, we can display the covariate-specific DFs during postestimation by typing
mixed postestimation — Postestimation tools for mixed

. estat df

Degrees of freedom

<table>
<thead>
<tr>
<th></th>
<th>Kenward-Roger</th>
</tr>
</thead>
<tbody>
<tr>
<td>gcf</td>
<td></td>
</tr>
<tr>
<td>followup</td>
<td>10.96355</td>
</tr>
<tr>
<td>base_gcf</td>
<td>47.2708</td>
</tr>
<tr>
<td>cda</td>
<td>50.70932</td>
</tr>
<tr>
<td>age</td>
<td>10.41127</td>
</tr>
<tr>
<td>_cons</td>
<td>25.43377</td>
</tr>
</tbody>
</table>

estat df can also compare different DF methods using the method() option. For example, we can compare the Kenward–Roger method with the Satterthwaite method by typing

. estat df, method(kroger satterthwaite)

Degrees of freedom

<table>
<thead>
<tr>
<th></th>
<th>Kenward-Roger</th>
<th>Satterthwaite</th>
</tr>
</thead>
<tbody>
<tr>
<td>gcf</td>
<td></td>
<td></td>
</tr>
<tr>
<td>followup</td>
<td>10.96355</td>
<td>10.96355</td>
</tr>
<tr>
<td>base_gcf</td>
<td>47.2708</td>
<td>47.2708</td>
</tr>
<tr>
<td>cda</td>
<td>50.70932</td>
<td>50.70932</td>
</tr>
<tr>
<td>age</td>
<td>10.41127</td>
<td>10.41127</td>
</tr>
<tr>
<td>_cons</td>
<td>25.43377</td>
<td>25.43377</td>
</tr>
</tbody>
</table>

The two methods produce the same estimates of DFs for single-hypothesis tests, but the results differ for multiple-hypotheses tests; see example 6 for details.

Suppose that we decide to proceed with the Satterthwaite method in subsequent analysis. Rather than retyping our mixed command with the dfmethod(satterthwaite) option, we can post the Satterthwaite DFs using the post option of estat df.

. estat df, method(satterthwaite) post

Degrees of freedom

<table>
<thead>
<tr>
<th></th>
<th>Satterthwaite</th>
</tr>
</thead>
<tbody>
<tr>
<td>gcf</td>
<td></td>
</tr>
<tr>
<td>followup</td>
<td>10.96355</td>
</tr>
<tr>
<td>base_gcf</td>
<td>47.2708</td>
</tr>
<tr>
<td>cda</td>
<td>50.70932</td>
</tr>
<tr>
<td>age</td>
<td>10.41127</td>
</tr>
<tr>
<td>_cons</td>
<td>25.43377</td>
</tr>
</tbody>
</table>

The returned values associated with dfmethod(kroger) from the mixed command will be replaced with those of dfmethod(satterthwaite).

Example 6

Continuing with example 5, after posting coefficient-specific DFs computed using the dfmethod(satterthwaite) method, we can use test or testparm with the small option for small-sample adjusted tests for fixed effects. For example, we can test the hypotheses that all fixed effects are zero by typing

. test

The following output shows the results of the test:
The $F$ statistic for the overall test is 1.87, and the DDF is estimated to be 16.49. These results are different from the model test using the Kenward–Roger DDF method reported in the header of the estimation output in example 5 (the $F$ statistic is 1.47, and the model DDF is 27.96).

The results differ because the Kenward–Roger method uses an adjusted $F$-test statistic and adjusts the fixed-effects variance–covariance estimator for a small sample. Both methods, however, lead to the same conclusion of no joint significance of the fixed effects.

Without option `small`, the commands `test` and `testparm` report large-sample $\chi^2$ Wald tests. We can compare the small-sample and large-sample tests of the joint hypotheses that the coefficient on `followup` and the coefficient on `age` equal zero.

```
. test followup = age = 0, small
   ( 1) [gcf]followup - [gcf]age = 0
   ( 2) [gcf]followup = 0
   F( 2, 10.75) =  3.65
   Prob > F =  0.0617
```

The DDF of the $F$ test, which is computed using the Satterthwaite method from our posted results, is 10.75. The $p$-values are very different (0.0617 versus 0.0260), and they lead to different conclusions of whether we should reject the null hypotheses at the $\alpha = 0.05$ level.

Similarly, you can use the `small` option with `lincom` to perform small-sample inference for linear combinations of fixed effects.

Example 7

As we did with `test`, after fitting a mixed model with the `dfmethod()` option for small-sample adjustment, we can use the `small` option with `contrast` to adjust for a small sample when estimating contrasts. Suppose we have collected data on a vigilance performance test. This experiment has been designed to test the response latency scores of two modes of signal during a four-hour monitoring period. This is a split-plot factorial design where `signal` is the whole-plot factor, `hour` is the subplot factor, and `subject` is the block factor. The whole-plot factor and the subplot factor are fixed; the block factor is random. Also, suppose that two measurements are missing in this dataset.
mixed postestimation — Postestimation tools for mixed 445

. use http://www.stata-press.com/data/r14/vptscores, clear
(Vigilance Performance Test Scores With Missing Data)
. tabdisp subject hour, cellvar(score) by(signal) concise missing

<table>
<thead>
<tr>
<th>Signal and Subject ID</th>
<th>Monitoring period</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1 2 3 4</td>
</tr>
<tr>
<td>Auditory</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>3 4 7 7</td>
</tr>
<tr>
<td>2</td>
<td>6 5 . 8</td>
</tr>
<tr>
<td>3</td>
<td>3 4 7 9</td>
</tr>
<tr>
<td>4</td>
<td>3 3 6 8</td>
</tr>
<tr>
<td>Visual</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>1 2 5 10</td>
</tr>
<tr>
<td>6</td>
<td>2 3 6 .</td>
</tr>
<tr>
<td>7</td>
<td>2 4 5 9</td>
</tr>
<tr>
<td>8</td>
<td>2 3 6 11</td>
</tr>
</tbody>
</table>

We start by fitting a mixed model. Because the dataset is small and unbalanced, we apply the Kenward–Roger method for small-sample adjustment:

. mixed score signal##hour || subject:, reml dfmethod(kroger) nolog nogroup

Mixed-effects REML regression
Number of obs = 30
DF method: Kenward-Roger
DF: min = 16.02
avg = 16.76
max = 18.29
F(7, 16.08) = 43.84
Log restricted-likelihood = -32.9724

| score          | Coef. | Std. Err. | t     | P>|t| | [95% Conf. Interval] |
|----------------|-------|-----------|-------|-------|---------------------|
| signal         |       |           |       |       |                     |
| Visual         | -2    | .6288677  | -3.18 | 0.005 | -3.319693 -.6803071 |
| hour           |       |           |       |       |                     |
| 2              | .25   | .5359916  | 0.47  | 0.647 | -.8861371 1.386137 |
| 3              | 3.108222 | .5911044 | 5.26  | 0.000 | 1.859163 4.357281 |
| 4              | 4.25  | .5359916  | 7.93  | 0.000 | 3.113863 5.386137 |
| signal#hour    |       |           |       |       |                     |
| Visual#2       | 1     | .7580066  | 1.32  | 0.206 | -.6067405 2.606741 |
| Visual#3       | .6417778 | .7979294 | 0.80  | 0.433 | -1.046666 2.330221 |
| Visual#4       | 4.044205 | .7979294 | 5.07  | 0.000 | 2.355762 5.732649 |
| _cons          | 3.75  | .4446766  | 8.43  | 0.000 | 2.816836 4.683164 |

Random-effects Parameters
<table>
<thead>
<tr>
<th>Estimate</th>
<th>Std. Err.</th>
<th>[95% Conf. Interval]</th>
</tr>
</thead>
</table>
| subject: Identity
| var(_cons) | .2163751  | .2345718  | .0258477  | 1.811312 |
| var(Residual) | .574574  | .2062107  | .2843515  | 1.161011 |

LR test vs. linear model: chibar2(01) = 1.55
Prob >= chibar2 = 0.1069
We can test the main effects and the interaction effects by typing the `contrast` command. With the `small` option, `contrast` reports small-sample adjusted $F$ tests. Without the `small` option, `contrast` performs large-sample $\chi^2$ Wald tests. Below is the comparison of the small-sample and the large-sample contrasts:

```
. contrast signal#hour, small
Contrasts of marginal linear predictions
Margins : asbalanced

<table>
<thead>
<tr>
<th></th>
<th>df</th>
<th>ddf</th>
<th>F</th>
<th>P&gt;F</th>
</tr>
</thead>
<tbody>
<tr>
<td>score</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>signal</td>
<td>1</td>
<td>5.95</td>
<td>1.78</td>
<td>0.2307</td>
</tr>
<tr>
<td>hour</td>
<td>3</td>
<td>16.35</td>
<td>100.62</td>
<td>0.0000</td>
</tr>
<tr>
<td>signal#hour</td>
<td>3</td>
<td>16.35</td>
<td>9.66</td>
<td>0.0007</td>
</tr>
</tbody>
</table>
```

```
. contrast signal#hour
Contrasts of marginal linear predictions
Margins : asbalanced

<table>
<thead>
<tr>
<th></th>
<th>df</th>
<th>chi2</th>
<th>P&gt;chi2</th>
</tr>
</thead>
<tbody>
<tr>
<td>score</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>signal</td>
<td>1</td>
<td>1.79</td>
<td>0.1810</td>
</tr>
<tr>
<td>hour</td>
<td>3</td>
<td>304.95</td>
<td>0.0000</td>
</tr>
<tr>
<td>signal#hour</td>
<td>3</td>
<td>29.35</td>
<td>0.0000</td>
</tr>
</tbody>
</table>
```

From these results, we can see that the $p$-values for the main effect of `signal` and the interaction effect vary between small-sample and large-sample tests. However, both tests indicate that the `hour` effect and the interaction effects are significant. We can decompose the interaction effect into separate interaction contrasts for further investigation.
From previous analysis, we already knew the overall interaction was significant. From the decomposition, we can easily see that the overall significance is driven by differences in the third and fourth hours; the change in response latency from hour three to hour four is greater for visual signals than for auditory signals.

We can also calculate the pairwise differences of the hourly marginal means by typing the `pwcompare` command. With the `small` option, `pwcompare` reports small-sample adjusted pairwise comparisons along with the degrees of freedom for each pairwise comparison.

When we compare these results with the large-sample results below, we can see that the confidence interval of hour 2 versus hour 1 changes to include 0. Therefore, after adjusting for small-sample
size, we would not reject the hypothesis that the means for hour 1 and hour 2 are equivalent at the 5% significance level.

```
. pwcompare hour
Pairwise comparisons of marginal linear predictions
Margins : asbalanced

<table>
<thead>
<tr>
<th>score</th>
<th>Contrast</th>
<th>Std. Err.</th>
<th>[95% Conf. Interval]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>hour</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2 vs 1</td>
<td>.75</td>
<td>.3790033</td>
<td>.0071672 1.492833</td>
</tr>
<tr>
<td>3 vs 1</td>
<td>3.429111</td>
<td>.3971529</td>
<td>2.650706 4.207516</td>
</tr>
<tr>
<td>4 vs 1</td>
<td>6.272103</td>
<td>.3971529</td>
<td>5.493697 7.050508</td>
</tr>
<tr>
<td>3 vs 2</td>
<td>2.679111</td>
<td>.3971529</td>
<td>1.900706 3.457516</td>
</tr>
<tr>
<td>4 vs 2</td>
<td>5.522103</td>
<td>.3971529</td>
<td>4.743697 6.300508</td>
</tr>
<tr>
<td>4 vs 3</td>
<td>2.842991</td>
<td>.4145085</td>
<td>2.03057 3.655413</td>
</tr>
</tbody>
</table>
```

Stored results

`estat icc` stores the following in `r()`:

Scalars
- `r(icc#)`: level-# intraclass correlation
- `r(se#)`: standard errors of level-# intraclass correlation
- `r(level)`: confidence level of confidence intervals

Macros
- `r(label#)`: label for level #

Matrices
- `r(ci#)`: vector of confidence intervals (lower and upper) for level-# intraclass correlation

For a G-level nested model, # can be any integer between 2 and G.

`estat recovariance` stores the following in `r()`:

Scalars
- `r(relevels)`: number of levels

Matrices
- `r(Cov#)`: level-# random-effects covariance matrix
- `r(Corr#)`: level-# random-effects correlation matrix (if option `correlation` was specified)

For a G-level nested model, # can be any integer between 2 and G.

`estat wcorrelation` stores the following in `r()`:

Matrices
- `r(sd)`: standard deviations
- `r(Corr)`: within-cluster correlation matrix
- `r(Cov)`: within-cluster variance–covariance matrix
- `r(G)`: variance–covariance matrix of random effects
- `r(Z)`: model-based design matrix
- `r(R)`: variance–covariance matrix of level-one errors
**mixed postestimation — Postestimation tools for mixed**

**estat df** stores the following in `r()`:

**Macros**
- `r(dfmethods)` — DF methods

**Matrices**
- `r(df)` — parameter-specific DFs for each method specified in `method()`
- `r(V_df)` — variance–covariance matrix of the estimators when kroger method is specified

If option `post()` is specified, **estat df** also stores the following in `e()`:

**Scalars**
- `e(F)` — overall $F$ test statistic for the method specified in `post()`
- `e(ddf_m)` — model DDF for the method specified in `post()`
- `e(df_max)` — maximum DF for the method specified in `post()`
- `e(df_avg)` — average DF for the method specified in `post()`
- `e(df_min)` — minimum DF for the method specified in `post()`

**Macros**
- `e(dfmethod)` — DF method specified in `post()`
- `e(dftitle)` — title for DF method

**Matrices**
- `e(df)` — parameter-specific DFs for the method specified in `post()`
- `e(V_df)` — variance–covariance matrix of the estimators when kroger method is posted

**pwcompare** with option `small` stores the following in `r()`:

**Matrices**
- `r(L_df)` — degrees of freedom for each margin difference
- `r(M_df)` — degrees of freedom for each margin estimate

**pwcompare** with options `post` and `small` stores the following in `e()`:

**Matrices**
- `e(L_df)` — degrees of freedom for each margin difference
- `e(M_df)` — degrees of freedom for each margin estimate

---

**Methods and formulas**

Methods and formulas are presented under the following headings:

*Prediction*
*Intraclass correlations*
*Within-cluster covariance matrix*
*Small-sample inference*

**Prediction**

Following the notation defined throughout [*ME* mixed], BLUPs of random effects $\mathbf{u}$ are obtained as

$$
\tilde{\mathbf{u}} = \tilde{\mathbf{G}} \tilde{\mathbf{V}}^{-1} \left( \mathbf{y} - \mathbf{X} \hat{\mathbf{\beta}} \right)
$$

where $\tilde{\mathbf{G}}$ and $\tilde{\mathbf{V}}$ are $\mathbf{G}$ and $\mathbf{V} = \mathbf{ZGZ}' + \sigma_e^2 \mathbf{R}$ with maximum likelihood (ML) or REML estimates of the variance components plugged in. Standard errors for BLUPs are calculated based on the iterative technique of Bates and Pinheiro (1998, sec. 3.3) for estimating the BLUPs themselves. If estimation is done by REML, these standard errors account for uncertainty in the estimate of $\mathbf{\beta}$, while for ML the standard errors treat $\mathbf{\beta}$ as known. As such, standard errors of REML-based BLUPs will usually be larger.
Fitted values are given by $X\hat{\beta} + Z\tilde{u}$, residuals as $\hat{\epsilon} = y - X\hat{\beta} - Z\tilde{u}$, and standardized residuals as

$$
\hat{\epsilon}_* = \frac{1}{\hat{\sigma}} \hat{R}^{-1/2} \hat{\epsilon}
$$

If the `relevel(levelvar)` option is specified, fitted values, residuals, and standardized residuals consider only those random-effects terms up to and including level `levelvar` in the model.

For details concerning the calculation of scores, see Methods and formulas in [ME] mixed.

**Intraclass correlations**

Consider a simple, two-level random-intercept model,

$$y_{ij} = \beta + u_{j}^{(2)} + \epsilon_{ij}^{(1)}$$

for measurements $i = 1, \ldots, n_j$ and level-2 groups $j = 1, \ldots, M$, where $y_{ij}$ is a response, $\beta$ is an unknown fixed intercept, $u_{j}$ is a level-2 random intercept, and $\epsilon_{ij}^{(1)}$ is a level-1 error term. Errors are assumed to be normally distributed with mean 0 and variance $\sigma_1^2$; random intercepts are assumed to be normally distributed with mean 0 and variance $\sigma_2^2$ and to be independent of error terms.

The intraclass correlation for this model is

$$
\rho = \text{Corr}(y_{ij}, y_{i'j}) = \frac{\sigma_2^2}{\sigma_1^2 + \sigma_2^2}
$$

It corresponds to the correlation between measurements $i$ and $i'$ from the same group $j$.

Now consider a three-level nested random-intercept model,

$$y_{ijk} = \beta + u_{jk}^{(2)} + u_{k}^{(3)} + \epsilon_{ijk}^{(1)}$$

for measurements $i = 1, \ldots, n_{jk}$ and level-2 groups $j = 1, \ldots, M_{1k}$ nested within level-3 groups $k = 1, \ldots, M_2$. Here $u_{jk}^{(2)}$ is a level-2 random intercept, $u_{k}^{(3)}$ is a level-3 random intercept, and $\epsilon_{ijk}^{(1)}$ is a level-1 error term. The error terms and random intercepts are assumed to be normally distributed with mean 0 and variances $\sigma_1^2$, $\sigma_2^2$, and $\sigma_3^2$, respectively, and to be mutually independent.

We can consider two types of intraclass correlations for this model. We will refer to them as level-2 and level-3 intraclass correlations. The level-3 intraclass correlation is

$$
\rho^{(3)} = \text{Corr}(y_{ijk}, y_{i'jk}) = \frac{\sigma_3^2}{\sigma_1^2 + \sigma_2^2 + \sigma_3^2}
$$

This is the correlation between measurements $i$ and $i'$ from the same level-3 group $k$ and from different level-2 groups $j$ and $j'$.

The level-2 intraclass correlation is

$$
\rho^{(2)} = \text{Corr}(y_{ijk}, y_{i'jk}) = \frac{\sigma_2^2 + \sigma_3^2}{\sigma_1^2 + \sigma_2^2 + \sigma_3^2}
$$

This is the correlation between measurements $i$ and $i'$ from the same level-3 group $k$ and level-2 group $j$. (Note that level-1 intraclass correlation is undefined.)
More generally, for a $G$-level nested random-intercept model, the $g$-level intraclass correlation is defined as

$$
\rho^{(g)} = \frac{\sum_{l=g}^{G} \sigma^2_l}{\sum_{l=1}^{G} \sigma^2_l}
$$

The above formulas also apply in the presence of fixed-effects covariates $X$ in a random-effects model. In this case, intraclass correlations are conditional on fixed-effects covariates and are referred to as residual intraclass correlations. `estat icc` also uses the same formulas to compute intraclass correlations for random-coefficient models, assuming 0 baseline values for the random-effects covariates, and labels them as conditional intraclass correlations. The above formulas assume independent residual structures.

Intraclass correlations are estimated using the delta method and will always fall in (0,1) because variance components are nonnegative. To accommodate the range of an intraclass correlation, we use the logit transformation to obtain confidence intervals.

Let $\hat{\rho}^{(g)}$ be a point estimate of the intraclass correlation and $\hat{SE}(\hat{\rho}^{(g)})$ be its standard error. The $(1 - \alpha) \times 100\%$ confidence interval for $\logit(\rho^{(g)})$ is

$$
\logit(\hat{\rho}^{(g)}) \pm z_{\alpha/2} \frac{\hat{SE}(\hat{\rho}^{(g)})}{\hat{\rho}^{(g)} (1 - \hat{\rho}^{(g)})}
$$

where $z_{\alpha/2}$ is the $1 - \alpha / 2$ quantile of the standard normal distribution and $\logit(x) = \ln\{x / (1 - x)\}$. Let $k_u$ be the upper endpoint of this interval, and let $k_l$ be the lower. The $(1 - \alpha) \times 100\%$ confidence interval for $\rho^{(g)}$ is then given by

$$
\left(\frac{1}{1 + e^{-k_l}}, \frac{1}{1 + e^{-k_u}}\right)
$$

**Within-cluster covariance matrix**

A two-level linear mixed model of the form

$$
y_j = X_j \beta + Z_j u_j + \epsilon_j
$$

implies the marginal model

$$
y_j = X_j \beta + \epsilon_j^*
$$

where $\epsilon_j^* \sim N(0, V_j)$, $V_j = Z_j G Z'_j + R$. In a marginal model, the random part is described in terms of the marginal or total residuals $\epsilon_j^*$, and $V_j$ is the covariance structure of these residuals.

`estat wcorrelation` calculates the marginal covariance matrix $\tilde{V}_j$ for cluster $j$ and by default displays the results in terms of standard deviations and correlations. This allows for a comparison of different multilevel models in terms of the ultimate within-cluster correlation matrix that each model implies.

Calculation of the marginal covariance matrix extends naturally to higher-level models; see, for example, chapter 4.8 in *West, Welch, and Gałecki (2015)*.
Small-sample inference

For small-sample computations performed when the `small` option is used with `test`, `testparm`, `lincom`, `contrast`, or `pwcompare`, see *Denominator degrees of freedom* in *Methods and formulas* of [ME] mixed.

References


Also see

[ME] mixed — Multilevel mixed-effects linear regression

[U] 20 Estimation and postestimation commands
Glossary

ANOVA denominator degrees of freedom (DFD) method. This method uses the traditional ANOVA for computing DDF. According to this method, the DDF for a test of a fixed effect of a given variable depends on whether that variable is also included in any of the random-effects equations. For traditional ANOVA models with balanced designs, this method provides exact sampling distributions of the test statistics. For more complex mixed-effects models or with unbalanced data, this method typically leads to poor approximations of the actual sampling distributions of the test statistics.

Approximation denominator degrees of freedom (DFD) methods. The Kenward–Roger and Satterthwaite DDF methods are referred to as approximation methods because they approximate the sampling distributions of test statistics using t and F distributions with the DDF specific to the method for complicated mixed-effects models and for simple mixed models with unbalanced data. Also see exact denominator degrees of freedom (DFD) methods.

Between–within denominator degrees of freedom (DFD) method. See repeated denominator degrees of freedom (DFD) method.

BLUPs. BLUPs are best linear unbiased predictions of either random effects or linear combinations of random effects. In linear models containing random effects, these effects are not estimated directly but instead are integrated out of the estimation. Once the fixed effects and variance components have been estimated, you can use these estimates to predict group-specific random effects. These predictions are called BLUPs because they are unbiased and have minimal mean squared errors among all linear functions of the response.

Canonical link. Corresponding to each family of distributions in a generalized linear model (GLM) is a canonical link function for which there is a sufficient statistic with the same dimension as the number of parameters in the linear predictor. The use of canonical link functions provides the GLM with desirable statistical properties, especially when the sample size is small.

Conditional hazard function. In the context of mixed-effects survival models, the conditional hazard function is the hazard function computed conditionally on the random effects. Even within the same covariate pattern, the conditional hazard function varies among individuals who belong to different random-effects clusters.

Conditional hazard ratio. In the context of mixed-effects survival models, the conditional hazard ratio is the ratio of two conditional hazard functions evaluated at different values of the covariates. Unless stated differently, the denominator corresponds to the conditional hazard function at baseline, that is, with all the covariates set to zero.

Conditional overdispersion. In a negative binomial mixed-effects model, conditional overdispersion is overdispersion conditional on random effects. Also see overdispersion.

Containment denominator degrees of freedom (DFD) method. See ANOVA denominator degrees of freedom (DFD) method.

Covariance structure. In a mixed-effects model, covariance structure refers to the variance–covariance structure of the random effects.

Crossed-effects model. A crossed-effects model is a mixed-effects model in which the levels of random effects are not nested. A simple crossed-effects model for cross-sectional time-series data would contain a random effect to control for panel-specific variation and a second random effect to control for time-specific random variation. Rather than being nested within panel, in this model a random effect due to a given time is the same for all panels.
crossed-random effects. See crossed-effects model.

EB. See empirical Bayes.

empirical Bayes. In generalized linear mixed-effects models, empirical Bayes refers to the method of prediction of the random effects after the model parameters have been estimated. The empirical Bayes method uses Bayesian principles to obtain the posterior distribution of the random effects, but instead of assuming a prior distribution for the model parameters, the parameters are treated as given.

empirical Bayes mean. See posterior mean.

empirical Bayes mode. See posterior mode.

exact denominator degrees of freedom (DDF) methods. Residual, repeated, and ANOVA DDF methods are referred to as exact methods because they provide exact \( t \) and \( F \) sampling distributions of test statistics for special classes of mixed-effects models—linear regression, repeated-measures designs, and traditional ANOVA models—with balanced data. Also see approximation denominator degrees of freedom (DDF) methods.

fixed effects. In the context of multilevel mixed-effects models, fixed effects represent effects that are constant for all groups at any level of nesting. In the ANOVA literature, fixed effects represent the levels of a factor for which the inference is restricted to only the specific levels observed in the study. See also fixed-effects model in [XT] Glossary.

Gauss–Hermite quadrature. In the context of generalized linear mixed models, Gauss–Hermite quadrature is a method of approximating the integral used in the calculation of the log likelihood. The quadrature locations and weights for individual clusters are fixed during the optimization process.

generalized linear mixed-effects model. A generalized linear mixed-effects model is an extension of a generalized linear model allowing for the inclusion of random deviations (effects).

generalized linear model. The generalized linear model is an estimation framework in which the user specifies a distributional family for the dependent variable and a link function that relates the dependent variable to a linear combination of the regressors. The distribution must be a member of the exponential family of distributions. The generalized linear model encompasses many common models, including linear, probit, and Poisson regression.

GHQ. See Gauss–Hermite quadrature.

GLM. See generalized linear model.

GLME model. See generalized linear mixed-effects model.

GLMM. Generalized linear mixed model. See generalized linear mixed-effects model.

hierarchical model. A hierarchical model is one in which successively more narrowly defined groups are nested within larger groups. For example, in a hierarchical model, patients may be nested within doctors who are in turn nested within the hospital at which they practice.

intraclass correlation. In the context of mixed-effects models, intraclass correlation refers to the correlation for pairs of responses at each nested level of the model.

Kenward–Roger denominator degrees of freedom (DDF) method. This method implements the Kenward and Roger (1997) method, which is designed to approximate unknown sampling distributions of test statistics for complex linear mixed-effects models. This method is supported only with restricted maximum-likelihood estimation.
Laplacian approximation. Laplacian approximation is a technique used to approximate definite integrals without resorting to quadrature methods. In the context of mixed-effects models, Laplacian approximation is as a rule faster than quadrature methods at the cost of producing biased parameter estimates of variance components.

linear mixed model. See linear mixed-effects model.

linear mixed-effects model. A linear mixed-effects model is an extension of a linear model allowing for the inclusion of random deviations (effects).

link function. In a generalized linear model or a generalized linear mixed-effects model, the link function relates a linear combination of predictors to the expected value of the dependent variable. In a linear regression model, the link function is simply the identity function.

LME model. See linear mixed-effects model.

MCAGH. See mode-curvature adaptive Gauss–Hermite quadrature.

mean–variance adaptive Gauss–Hermite quadrature. In the context of generalized linear mixed models, mean–variance adaptive Gauss–Hermite quadrature is a method of approximating the integral used in the calculation of the log likelihood. The quadrature locations and weights for individual clusters are updated during the optimization process by using the posterior mean and the posterior standard deviation.

mixed model. See mixed-effects model.

mixed-effects model. A mixed-effects model contains both fixed and random effects. The fixed effects are estimated directly, whereas the random effects are summarized according to their (co)variances. Mixed-effects models are used primarily to perform estimation and inference on the regression coefficients in the presence of complicated within-subject correlation structures induced by multiple levels of grouping.

mode-curvature adaptive Gauss–Hermite quadrature. In the context of generalized linear mixed models, mode-curvature adaptive Gauss–Hermite quadrature is a method of approximating the integral used in the calculation of the log likelihood. The quadrature locations and weights for individual clusters are updated during the optimization process by using the posterior mode and the standard deviation of the normal density that approximates the log posterior at the mode.

MVAGH. See mean–variance adaptive Gauss–Hermite quadrature.

nested random effects. In the context of mixed-effects models, nested random effects refer to the nested grouping factors for the random effects. For example, we may have data on students who are nested in classes that are nested in schools.

one-level model. A one-level model has no multilevel structure and no random effects. Linear regression is a one-level model.

overdispersion. In count-data models, overdispersion occurs when there is more variation in the data than would be expected if the process were Poisson.

posterior mean. In generalized linear mixed-effects models, posterior mean refers to the predictions of random effects based on the mean of the posterior distribution.

posterior mode. In generalized linear mixed-effects models, posterior mode refers to the predictions of random effects based on the mode of the posterior distribution.

QR decomposition. QR decomposition is an orthogonal-triangular decomposition of an augmented data matrix that speeds up the calculation of the log likelihood; see Methods and formulas in [ME] mixed for more details.

quadrature. Quadrature is a set of numerical methods to evaluate a definite integral.
random coefficient. In the context of mixed-effects models, a random coefficient is a counterpart to a slope in the fixed-effects equation. You can think of a random coefficient as a randomly varying slope at a specific level of nesting.

random effects. In the context of mixed-effects models, random effects represent effects that may vary from group to group at any level of nesting. In the ANOVA literature, random effects represent the levels of a factor for which the inference can be generalized to the underlying population represented by the levels observed in the study. See also random-effects model in [XT] Glossary.

random intercept. In the context of mixed-effects models, a random intercept is a counterpart to the intercept in the fixed-effects equation. You can think of a random intercept as a randomly varying intercept at a specific level of nesting.

REML. See restricted maximum likelihood.

repeated denominator degrees of freedom (DDF) method. This method uses the repeated-measures ANOVA for computing DDF. It is used with balanced repeated-measures designs with spherical correlation error structures. It partitions the residual degrees of freedom into the between-subject degrees of freedom and the within-subject degrees of freedom. The repeated method is supported only with two-level models. For more complex mixed-effects models or with unbalanced data, this method typically leads to poor approximations of the actual sampling distributions of the test statistics.

residual denominator degrees of freedom (DDF) method. This method uses the residual degrees of freedom, $n - \text{rank}(X)$, as the DDF for all tests of fixed effects. For a linear model without random effects with independent and identically distributed errors, the distributions of the test statistics for fixed effects are $t$ or $F$ distributions with the residual DDF. For other mixed-effects models, this method typically leads to poor approximations of the actual sampling distributions of the test statistics.

restricted maximum likelihood. Restricted maximum likelihood is a method of fitting linear mixed-effects models that involves transforming out the fixed effects to focus solely on variance–component estimation.

Satterthwaite denominator degrees of freedom (DDF) method. This method implements a generalization of the Satterthwaite (1946) approximation of the unknown sampling distributions of test statistics for complex linear mixed-effects models. This method is supported only with restricted maximum-likelihood estimation.

three-level model. A three-level mixed-effects model has one level of observations and two levels of grouping. Suppose that you have a dataset consisting of patients overseen by doctors at hospitals, and each doctor practices at one hospital. Then a three-level model would contain a set of random effects to control for hospital-specific variation, a second set of random effects to control for doctor-specific random variation within a hospital, and a random-error term to control for patients’ random variation.

two-level model. A two-level mixed-effects model has one level of observations and one level of grouping. Suppose that you have a panel dataset consisting of patients at hospitals; a two-level model would contain a set of random effects at the hospital level (the second level) to control for hospital-specific random variation and a random-error term at the observation level (the first level) to control for within-hospital variation.

variance components. In a mixed-effects model, the variance components refer to the variances and covariances of the various random effects.
References


Subject and author index

See the combined subject index and the combined author index in the Glossary and Index.