## Contents

<table>
<thead>
<tr>
<th>Intro</th>
<th>Introduction to DSGEs</th>
<th>Learning the syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intro 1</td>
<td>Classic DSGE examples</td>
<td>New Keynesian model</td>
</tr>
<tr>
<td>Intro 2</td>
<td>Financial frictions model</td>
<td>New Classical model</td>
</tr>
<tr>
<td>Intro 3</td>
<td>Nonlinear New Keynesian model</td>
<td>Nonlinear New Classical model</td>
</tr>
<tr>
<td>Intro 3a</td>
<td>Nonlinear New Classical model</td>
<td>Stochastic growth model</td>
</tr>
<tr>
<td>Intro 3b</td>
<td>Writing a DSGE in a solvable form</td>
<td>Including a lag of a state variable</td>
</tr>
<tr>
<td>Intro 3c</td>
<td>Including an expectation dated by more than one period ahead</td>
<td>Including a second-order lag of a control</td>
</tr>
<tr>
<td>Intro 3d</td>
<td>Including an observed exogenous variable</td>
<td>Correlated state variables</td>
</tr>
<tr>
<td>Intro 3e</td>
<td>Stability conditions</td>
<td>Identification</td>
</tr>
<tr>
<td>Intro 3f</td>
<td>Convergence problems</td>
<td>Wald tests vary with nonlinear transforms</td>
</tr>
<tr>
<td>Intro 4</td>
<td>Linear dynamic stochastic general equilibrium models</td>
<td>Postestimation tools for dsge</td>
</tr>
<tr>
<td>Intro 4a</td>
<td>Nonlinear dynamic stochastic general equilibrium models</td>
<td>Postestimation tools for dsgenl</td>
</tr>
<tr>
<td>Intro 4b</td>
<td>Display estimated covariances of model variables</td>
<td>Display policy matrix</td>
</tr>
<tr>
<td>Intro 4c</td>
<td>Check stability of system</td>
<td>Display steady state of nonlinear DSGE model</td>
</tr>
<tr>
<td>Intro 4d</td>
<td>Display state transition matrix</td>
<td></td>
</tr>
</tbody>
</table>

Glossary | Subject and author index |
When reading this manual, you will find references to other Stata manuals, for example, \([U]\) 27 Overview of Stata estimation commands; \([R]\) regress; and \([D]\) reshape. The first example is a reference to chapter 27, 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:

- \(\text{GSM}\) Getting Started with Stata for Mac
- \(\text{GSU}\) Getting Started with Stata for Unix
- \(\text{GSW}\) Getting Started with Stata for Windows
- \([U]\) Stata User’s Guide
- \([R]\) Stata Base Reference Manual
- \([BAYES]\) Stata Bayesian Analysis Reference Manual
- \([D]\) Stata Data Management Reference Manual
- \([ERM]\) Stata Extended Regression Models Reference Manual
- \([FMM]\) Stata Finite Mixture Models Reference Manual
- \([FN]\) Stata Functions Reference Manual
- \([IRT]\) Stata Item Response Theory Reference Manual
- \([LASSO]\) Stata Lasso Reference Manual
- \([XT]\) Stata Longitudinal-Data/Panel-Data Reference Manual
- \([META]\) Stata Meta-Analysis Reference Manual
- \([MI]\) Stata Multiple-Imputation Reference Manual
- \([PSS]\) Stata Power, Precision, and Sample-Size Reference Manual
- \([P]\) Stata Programming Reference Manual
- \([RPT]\) Stata Reporting Reference Manual
- \([SP]\) Stata Spatial Autoregressive Models Reference Manual
- \([SVY]\) Stata Survey Data Reference Manual
- \([ST]\) Stata Survival Analysis Reference Manual
- \([TS]\) Stata Time-Series Reference Manual
- \([I]\) Stata Glossary and Index
- \([M]\) Mata Reference Manual
DSGE stands for dynamic stochastic general equilibrium. DSGE models are multivariate time-series models that are used in economics, in particular, macroeconomics, for policy analysis and forecasting. These models are systems of equations that are typically derived from economic theory. As such, the parameters are often directly interpretable based on theory. DSGE models are unique in that equations in the system allow current values of variables to depend not only on past values but also on expectations of future values.

The `dsge1` command estimates parameters of nonlinear DSGE models.

The `dsge` command estimates parameters of linear DSGE models.

**Remarks and examples**

We recommend that you read this manual beginning with [DSGE] Intro 1 and then continue with the remaining introductions. In these introductions, we will introduce DSGE models, show you how to use the `dsge1` and `dsge` commands, walk you through worked examples of classic models, and present solutions to common stumbling blocks.

[DSGE] Intro 1 and [DSGE] Intro 2 are essential reading. Read them first. Here you will find an overview of DSGE models, descriptions of concepts used throughout the manual, discussion of assumptions, a first worked example, and an introduction to the syntax.

[DSGE] Intro 1 Introduction to DSGEs
[DSGE] Intro 2 Learning the syntax

[DSGE] Intro 3 focuses on classical DSGE models. It includes a series of examples that illustrate model solution, model estimation, and postestimation procedures for simple variants of common models.

[DSGE] Intro 3 Classic DSGE examples
[DSGE] Intro 3a New Keynesian model
[DSGE] Intro 3b New Classical model
[DSGE] Intro 3c Financial frictions model
[DSGE] Intro 3d Nonlinear New Keynesian model
[DSGE] Intro 3e Nonlinear New Classical model
[DSGE] Intro 3f Stochastic growth model

[DSGE] Intro 4 discusses some features commonly found in DSGE models and how to specify models with those features to `dsge` and `dsge1`. The structural equations of the DSGE model must have a specific structure so that the model can be solved. Often, DSGE models are written using intuitive forms that do not have this structure. These intuitive forms can be rewritten in a logically equivalent form that has the structure required for solution. [DSGE] Intro 4 provides an overview of this topic and examples demonstrating solutions.
Writing a DSGE in a solvable form
Specifying a shock on a control variable
Including a lag of a control variable
Including a lag of a state variable
Including an expectation dated by more than one period ahead
Including a second-order lag of a control
Including an observed exogenous variable
Correlated state variables

Stability conditions
Identification
Convergence problems
Wald tests vary with nonlinear transforms

The main command entries are references for syntax and implementation details. All the examples are in the introductions discussed above.

Linear dynamic stochastic general equilibrium models
Postestimation tools for dsge
Nonlinear dynamic stochastic general equilibrium models
Postestimation tools for dsgenl
Display estimated covariances of model variables
Display policy matrix
Check stability of system
Display steady state of nonlinear DSGE model
Display state transition matrix

Also see
Introduction to DSGEs
In this entry, we introduce DSGE models and the two commands that estimate the parameters of DSGE models, \texttt{dsgenl} and \texttt{dsge}. We begin with an overview of DSGE models. We then illustrate the process of DSGE modeling by working an example. In this example, we describe a model in its original nonlinear form and estimate its parameters using \texttt{dsgenl}. We also write the model in a corresponding linearized form and estimate the parameters using \texttt{dsge}. We conclude by showing how this example fits into the general DSGE modeling framework.

### Remarks and examples

Remarks are presented under the following headings:

- **Introduction to DSGE models**
  - An example: A nonlinear DSGE model
  - Writing down nonlinear DSGEs
  - Data preparation
  - Specifying the model to \texttt{dsgenl}
  - Parameter estimation and interpretation of nonlinear DSGEs
- **An example: A linear DSGE model**
  - Writing down linearized DSGEs
  - Specifying the model to \texttt{dsge}
  - Parameter estimation and interpretation of linear DSGEs
- **Postestimation**
  - Policy and transition matrices
  - Impulse responses
  - Forecasts
  - Structural and reduced forms of DSGE models

### Introduction to DSGE models

DSGE models are models for multiple time series used in macroeconomics and finance. These models are systems of equations that are motivated by economic theory and in which expectations of future values of variables play an important role. Because these models come from theory, the parameters of these models can typically be directly interpreted in terms of the motivating theory. DSGE models are used for macroeconomic policy analysis and forecasting.

In DSGE models, individuals’ actions are summarized by decision rules that take the form of nonlinear systems of dynamic equations. These decision rules often come from dynamic stochastic optimization problems. A DSGE model consists of these decision rules, plus any aggregation conditions, resource or budget constraints, and stochastic processes for exogenous variables.

Because the model’s equations are the solution to dynamic optimization problems, model equations can feature expectations of future variables. These expectations are endogenous. In DSGE models, expectations of future values of variables correspond to their conditional mean as implied by the model. In other words, individuals’ expectations of future values are correct, on average. Such expectations are said to be model-consistent expectations or rational expectations.
There are three kinds of variables in DSGE models: control variables, state variables, and shocks. The terminology is taken from the state-space and optimal control literatures. In DSGE models, the concepts of exogeneity and endogeneity are understood relative to a time period. A state variable is fixed, or exogenous, in a given time period. The system of equations then determines the value of the state variable one period in the future. On the other hand, the system of equations determines the value of a control variable in the current time period. Control variables in a DSGE model can be either observed or unobserved. State variables are always unobserved.

DSGE models can be written in multiple forms. The model may consist of equations that are nonlinear both in the variables and in the parameters. Such a model is said to be nonlinear. A DSGE model is said to be linear, or linearized, when the model equations are linear in the variables. Linear models may still be nonlinear in their parameters.

Nonlinear DSGE models are commonly linearized prior to analysis. In Stata, you have two choices for handling the linearization of a nonlinear model. You can write your model in its nonlinear form and use the `dsgenl` command. `dsgenl` will take a linear approximation to the nonlinear model for you. Alternatively, you can derive the linear form of the model and use the `dsge` command. Although linearizing the equations yourself requires extra effort, there can be advantages. Linearized DSGE models are easier to manipulate than nonlinear ones. For instance, it is easier to include additional lag terms in a linearized model than in a nonlinear one.

After linearization, the DSGE model must be solved prior to estimation. In any analysis of simultaneous equations systems, to solve a model means to write the model’s endogenous variables as functions of its exogenous variables. In DSGE models, the analogous concept is to write the model’s control variables in terms of its state variables. The model’s solution consists of a system of equations relating the control variables to the state variables and a system of equations describing the evolution of the state variables over time. The solution to a DSGE model thus takes the form of a state-space model. The solution to a DSGE model is a crucial object for both estimation and analyses after estimation. Both the likelihood function and the impulse–response functions are formed from the model solution.

`dsgenl` solves and estimates the parameters of nonlinear DSGE models. It linearizes the model by taking a first-order approximation to the model’s equations at the model’s steady state, solves the linearized model, and estimates the parameters of the model using the linearized solution.

`dsge` solves and estimates the parameters of linearized DSGE models.

An example: A nonlinear DSGE model

Writing down nonlinear DSGEs

Consider the following nonlinear model, similar to that in Woodford (2003, chap. 4). The model consists of equations that describe the behavior of households, firms, and a central bank. Interactions among these actors produce a model of inflation, output growth, and the interest rate. Models of this type are popular in academic and policy settings and are used to describe monetary policy.

Household optimization generates an equation that relates current output $Y_t$ to the expected value of a function of tomorrow’s output $Y_{t+1}$, tomorrow’s inflation $\Pi_{t+1}$, and the current nominal interest rate $R_t$, 

$$
\frac{1}{Y_t} = \beta E_t \left( \frac{1}{Y_{t+1}} \frac{R_t}{\Pi_{t+1}} \right) 
$$

where $\beta$ is a parameter that captures households’ willingness to delay consumption.

Optimization by firms generates an equation that links the current deviation of inflation from its steady state, $\Pi_t - \Pi$, to the expected value of the deviation of inflation from its steady state in the future, $E_t \left( \Pi_{t+1} - \Pi \right)$, and to the ratio of actual output, $Y_t$, to the natural level of output, $Z_t$,

$$(\Pi_t - \Pi) + \frac{1}{\phi} = \phi \left( \frac{Y_t}{Z_t} \right) + \beta E_t \left( \Pi_{t+1} - \Pi \right)$$

where $\phi$ is a parameter linked to the pricing decision of firms. Firms are not affected by inflation per se; they are affected only by deviations of inflation from its steady-state value.

Finally, there is an equation that describes central bank policy. The central bank adjusts the interest rate in response to inflation and, perhaps, to other factors that we leave unmodeled. The equation for the central bank policy is

$$
\frac{R_t}{R} = \left( \frac{\Pi_t}{\Pi} \right)^{1/\beta} U_t
$$

where $R$ is the steady-state value of the interest rate and $U_t$ is a state variable that captures all movements in the interest rate that are not driven by inflation.

We make a few modifications to the model equations before estimating the parameters. Woodford (2003) rewrites the model in (1)–(3) by defining $X_t = Y_t/Z_t$ as the output gap. We do the same. Substituting in $X_t$ gives us the three-equation system

$$
1 = \beta E_t \left( \frac{X_t}{X_{t+1}} \frac{1}{G_t} \frac{R_t}{\Pi_{t+1}} \right) 
$$

$$
(\Pi_t - \Pi) + \frac{1}{\phi} = \phi X_t + \beta E_t \left( \Pi_{t+1} - \Pi \right)
$$

$$
\frac{R_t}{R} = \left( \frac{\Pi_t}{\Pi} \right)^{1/\beta} U_t
$$

where $G_t = Z_{t+1}/Z_t$ is a state variable that captures changes in $Z_t$. 
The model is completed by adding equations describing the evolution of the state variables $G_t$ and $U_t$. We write them as autoregressive processes in logarithms,

$$\ln(G_{t+1}) = \rho_g \ln(G_t) + \xi_{t+1}$$  \hspace{1cm} (7a)
$$\ln(U_{t+1}) = \rho_u \ln(U_t) + \epsilon_{t+1}$$  \hspace{1cm} (8a)

The variables $\xi_{t+1}$ and $\epsilon_{t+1}$ are shocks to the state variables. We now have a complete nonlinear DSGE model. To estimate its parameters, we need to get some data.

**Data preparation**

There are three requirements of the data used with dsgenl and dsge. First, the data must be `tsset` so that Stata understands the time structure of your data. Second, the series in your model must have zero mean. This requirement is actually handled for you. Before solving the models, these commands first transform variables into differences from their steady state or, with dsgenl, percent deviations from their steady state. The third requirement is that variables in your data must be weakly stationary.

We have data on the price level and the nominal interest rate in `rates.dta`. These data were obtained from the Federal Reserve Economic Database (FRED), which contains many macroeconomic and financial time series; see [D] `import fred`. We can type `tsset` to see that `dateq` has already been set as the quarterly time variable in this dataset.

```
. use https://www.stata-press.com/data/r16/rates2
(Federal Reserve Economic Data - St. Louis Fed, 2017-02-10)
. tsset
time variable: dateq, 1947q1 to 2017q1
delta: 1 quarter
. describe
Contains data from https://www.stata-press.com/data/r16/rates2.dta
obs: 281 Federal Reserve Economic Data - St. Louis Fed, 2017-02-10
vars: 5 26 Apr 2019 21:22

    storage  display value
variable name type format label                      variable label
     datestr  str10  %-10s Observation date
     daten    int   %td         Numeric (daily) date
    gdpdef   float  %9.0g         GDP deflator GDPDEF
              float  %9.0g         Federal funds rate FEDFUNDS
     dateq    int   %tq         Quarterly date
```

The dataset includes the price-level variable `gdpdef`. But the model is written in terms of the inflation rate. For quarterly data, the inflation rate is conventionally obtained as 400 times the difference in log of the price level. Therefore, we begin by generating an inflation rate variable `p` by using the `L.` lag operator.

```
. generate p = 400*(ln(gdpdef) - ln(L.gdpdef))
(2 missing values generated)
. label variable p "Inflation rate"
```

We now have inflation and interest rates.
Specifying the model to dsgenl

Equations (4a)–(8a) are the nonlinear DSGE model. The command to estimate the parameters of the system (4a)–(8a) is

```
. dsgenl (1 = {beta}*(x/F.x)*(1/g)*(r/F.p) ) ///
(1/{phi} + (p-1) = {phi}*x + {beta}*(F.p-1)) ///
({beta}*r = p^({1/beta})*u) ///
(ln(F.u) = {rhou}*ln(u)) ///
(ln(F.g) = {rhog}*ln(g)), ///
exostate(u g) observed(p r) unobserved(x)
```

Each equation is bound in parentheses. The equations in the command look similar to the equations we wrote down. The equations can be written in nearly any form that is logically equivalent to the model equations. In addition to providing the equations, you must specify which variable plays which role in the model through the options. The option `exostate()` lists all exogenous state variables, those that are subject to shocks. The option `observed()` lists observed control variables. The option `unobserved()` lists unobserved (latent) control variables. A fourth option, `endostate()`, is available for endogenous state variables, those that are not subject to shocks. Each variable must be listed in one and only one of the options. The number of exogenous state variables must be the same as the number of observed control variables.

Parameters are bound in curly braces; this is how Stata distinguishes between parameters and variables. Parameters can appear nonlinearly, they can appear more than once, and they can appear in multiple equations. A parameter repeated across equations is considered the same parameter; for example, in the above model, beta appears in each of the first three equations, and it is assumed that there is only one parameter beta to be estimated. These cross-equation restrictions are common in DSGE models.

The operator F. can be applied to both state and control variables to denote the expected future value of a variable. For example, we typed F.x to include x_{t+1} in the model. However, unlike F. used in other time-series applications, the value of x_{t+1} is not determined from the value of x at time t + 1 in your data. Instead, it is an expectation of the value of x at time t + 1 based on your model. In fact, the entire model can be written in terms of expectations.

Notice that in (4a) and (5a), the $E_t$ operator appears. This is the statistical expectation operator. It refers to expected value of the term in brackets conditional on information known at time $t$. In dsgenl, this expectation operator is assumed in front of each equation. Hence, the dsgenl equation

$$(1 = \{beta\}*(x/F.x)*(1/g)*(r/F.p))$$

is interpreted as

$$E_t \left( 1 - \beta \frac{X_t}{X_{t+1}} \frac{1}{G_t} \frac{R_t}{\Pi_{t+1}} \right) = 0$$

which is equivalent to what we wrote in (4a).
Parameter estimation and interpretation of nonlinear DSGEs

The model’s parameters are estimated by taking a linear approximation to the model’s equations, solving the linearized system for its state-space form, and estimating the parameters of the state-space model by maximum likelihood.

\[
\text{dsgenl} \ (1 = \{\beta\}*(x/F.x)*(1/g)*(r/F.p)) \\
> (1/\phi) + (p-1) = \{\phi\}x + \{\beta\}*(F.p-1) \\
> (\{\beta\}r = p^{1/\{\beta\}}u) \\
> (\ln(F.u) = \{\rho_u\}ln(u)) \\
> (\ln(F.g) = \{\rho_g\}ln(g))
\]

Solving at initial parameter vector ...
Checking identification ...

First-order DSGE model
Sample: 1954q3 - 2016q4 Number of obs = 250
Log likelihood = -768.09383

<table>
<thead>
<tr>
<th></th>
<th>DGM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Coef.</td>
</tr>
<tr>
<td>/structural</td>
<td></td>
</tr>
<tr>
<td>beta</td>
<td>.5112873</td>
</tr>
<tr>
<td>phi</td>
<td>5.895197</td>
</tr>
<tr>
<td>rho_u</td>
<td>.6989192</td>
</tr>
<tr>
<td>rho_g</td>
<td>.9556406</td>
</tr>
<tr>
<td>sd(e.u)</td>
<td>2.317592</td>
</tr>
<tr>
<td>sd(e.g)</td>
<td>.6147361</td>
</tr>
</tbody>
</table>

We see header information describing the sample date range, the number of observations, and the log likelihood.

In the estimation table, we see estimates of model parameters. Some of these parameters have a structural interpretation. Parameter \(\beta\) plays two roles in the model: it is the discount factor in the household and firm equations, and its inverse captures the degree to which the central bank responds to inflation in the interest rate equation. Its point estimate is about one half, so a point estimate for its inverse is about 2.

We use `nlcom` to obtain an estimate of the parameter \(1/\beta\) directly.

\[
\text{nlcom} \ 1/_b[\beta] \\
\_nl_1: \ 1/_b[\beta]
\]

|                | Coef.     | Std. Err.    | z     | P>|z| | [95% Conf. Interval] |
|----------------|-----------|--------------|-------|------|----------------------|
| _nl_1          | 1.955847  | .2899274     | 6.75  | 0.000 | 1.3876 .. 2.524095   |

Values for this parameter in the literature are typically around 1.5. Our estimated value is about 2, but the common value of 1.5 lies within the confidence interval.

The \(\phi\) parameter is a measure of price adjustment. The remaining four parameters describe the evolution of the model’s exogenous state variables. Both state variables are highly persistent, with persistence parameters of 0.70 and 0.96 on u and g, respectively. The estimated standard deviations of the shocks are also displayed. The shock to the state variable u is denoted e.u and has standard deviation 2.32. The shock to the state variable g is denoted e.g and has standard deviation 0.61.
An example: A linear DSGE model

Writing down linearized DSGEs

The model in (4a)–(8a) is nonlinear. To use dsge, we first write the model in its corresponding linearized form, which we show below. Throughout this manual, we use lowercase letters to denote deviations of variables from the steady state. The linearized versions of (4a)–(8a) are

\[
\begin{align*}
    x_t &= E_t x_{t+1} - (r_t - E_t \pi_{t+1} - g_t) \\
    \pi_t &= \beta E_t \pi_{t+1} + \kappa x_t \\
    r_t &= \frac{1}{\beta} \pi_t + u_t \\
    u_{t+1} &= \rho_u u_t + \epsilon_{t+1} \\
    g_{t+1} &= \rho_g g_t + \xi_{t+1}
\end{align*}
\]

The new parameter \( \kappa \) is a function of the underlying parameter \( \phi \) from (5a).

We have implicitly constrained some of the coefficients; for example, the coefficient on the interest rate in the output equation is constrained to \(-1\).

Specifying the model to dsge

The dsge command to estimate the parameters in the system of (4b)–(8b) is

```plaintext
. dsge (p = {beta}*F.p + {kappa}*x) ///
   (x = F.x - (r - F.p - g), unobserved) ///
   (r = (1/{beta})*p + u) ///
   (F.u = {rhou}*u, state) ///
   (F.g = {rhog}*g, state)
```

As with dsgenl, each equation is bound in parentheses. The equations look almost identical to the system in (4b)–(8b). Because a model has as many variables as it has equations, each variable will appear on the left-hand side of one and only one equation but can appear on the right-hand side of as many equations as we like. Equations may be specified in any order.

We identify types of variables differently with dsge than we did with dsgenl. Equation options that appear within the parentheses indicate which type of variable appears on the left-hand side of the equation. Equations specified without any options are equations for observed control variables. We add the unobserved option to specify that a left-hand-side variable is an unobserved control variable. We use the state option to identify state variables.

In our model, \( p \) and \( r \) are the observed control variables \( p_t \) and \( r_t \), and their equations are specified without options. The output gap, \( x_t \), is an unobserved control variable, so we specify the equation for \( x \) using the unobserved option.

\( u \) and \( g \) are the state variables \( u_t \) and \( g_t \), and option state is included when we type their equations. Recall that state variables are fixed in the current period, so we specify how they evolve through time by modeling the one-period lead—hence, the F. on the left-hand side of each state equation. The state equations specify how the state variable evolves as a function of the current state variables and, possibly, the control variables.

The shocks \( \epsilon_t \) and \( \xi_t \) enter the system through the state equations of their corresponding variable. By default, a shock is attached to each state equation. The number of shocks must be the same as the number of observed control variables in the model.
When we typed

\[ . \text{dsge} \ldots (F.u = \{\text{rhou}\}u, \text{state}) \ldots \]

the underlying equation is what we wrote in (7b). If a state variable is treated as deterministic in your model, then it will not have a shock. For example, capital accumulation is often treated as deterministic. To include an equation for a state variable without a shock, we would include the \text{noshock} option within the equation.

As with \text{dsgenl}, the operator \( F. \) can be applied to both state and control variables to denote the expected future value of a variable. For example, we can type \( F.x \) to specify \( x_{t+1} \). Unlike \( F. \) used in other time-series applications, the value of \( x_{t+1} \) is not taken from the value of \( x \) at time \( t + 1 \) in your data. Instead, it is an expectation of the value of \( x \) at time \( t + 1 \) based on your model. \text{dsge} interprets equations with \( F. \) operators in the following manner. The equation

\[ . \text{dsge} \ (p = \{\beta\}p + \{\kappa\}x) \ldots \]

is interpreted as

\[ E_t (p_t - \beta p_{t+1} - \kappa x_t) = 0 \]

which is equivalent to (4b) above. In a linear DSGE model, it is immaterial whether the expectation operator is thought of as applying to a variable, or to an equation, or indeed to the entire system of equations jointly.

The parameters we want to estimate are bound in curly braces.

For more details on the \text{dsge} syntax, see [DSGE] Intro 2.

**Parameter estimation and interpretation of linear DSGEs**

We estimate the parameters of the model in (4b)–(8b). These equations are much discussed in the monetary economics literature. Equation (4b) is known as the output-gap Euler equation. Equation (5b) is known as a New Keynesian Phillips Curve, and the parameter \( \kappa \) is known as the slope of the Phillips curve. In New Keynesian models, prices depend on output, and \( \kappa \) is a measure of that dependence. Equation (6a) is known as a Taylor rule, after Taylor (1993). The coefficient on inflation in a Taylor rule is a commonly discussed parameter. \( \beta \) has two roles in the model above. It relates current inflation deviations to expected future inflation deviations, and it relates interest rate deviations to inflation deviations.
Intro 1 — Introduction to DSGEs

\[ dsge \ (p = \{\beta\}*F.p + \{kappa\}*x) \]
> \[ (x = F.x - (r - F.p - g), \text{unobserved}) \]
> \[ (r = (1/\{\beta\})*p + u) \]
> \[ (F.u = \{rhou\}*u, \text{state}) \]
> \[ (F.g = \{rhog\}*g, \text{state}) \]

(setting technique to bfgs)

Iteration 0: log likelihood = -13931.564
Iteration 1: log likelihood = -1301.5118 (backed up)
Iteration 2: log likelihood = -1039.6984 (backed up)
Iteration 3: log likelihood = -905.70867 (backed up)
Iteration 4: log likelihood = -842.76867 (backed up)

(switching technique to nr)

Iteration 5: log likelihood = -812.04209 (backed up)
Iteration 6: log likelihood = -786.76609
Iteration 7: log likelihood = -777.19779
Iteration 8: log likelihood = -768.8383
Iteration 9: log likelihood = -768.1368
Iteration 10: log likelihood = -768.09519
Iteration 11: log likelihood = -768.09383
Iteration 12: log likelihood = -768.09383

DSGE model

Sample: 1954q3 - 2016q4

Log likelihood = -768.09383

|                | Coef.     | Std. Err. | z     | P>|z| | [95% Conf. Interval] |
|----------------|-----------|-----------|-------|-----|---------------------|
| /structural    |           |           |       |     |                     |
| beta           | .5112881  | .075791   | 6.75  | 0.000 | .3627404 .6598359   |
| kappa          | .1696296  | .0475492  | 3.57  | 0.000 | .076435 .2628243    |
| rhou           | .6989189  | .0449192  | 15.56 | 0.000 | .6108789 .7869588   |
| rhog           | .9556407  | .0181342  | 52.70 | 0.000 | .9200983 .9911831   |
| sd(e.u)        | .3175889  | .3098024  | 1.73  | .083 | 1.90323            |
| sd(e.g)        | .6147348  | .0973279  | 6.35  | 0.000 | 1.387601 .524088    |

Two of the parameters have structural interpretations. The parameter \( \kappa \) is the slope of the Phillips curve. Theory predicts that this parameter will be positive, and indeed our estimate is positive.

The parameter \( \beta \) is the inverse of the coefficient on inflation in the interest rate equation. We can obtain an estimate of \( 1/\beta \), which is interpreted as the degree to which the central bank responds to movements in inflation, by using \texttt{nlcom}.

```
. nlcom 1/_b[beta]
   _nl_1: 1/_b[beta]

|                | Coef.     | Std. Err. | z     | P>|z| | [95% Conf. Interval] |
|----------------|-----------|-----------|-------|-----|---------------------|
| _nl_1          | 1.955844  | .2899255  | 6.75  | 0.000 | 1.387601 .524088    |
```

A typical value for \( 1/\beta \) in the literature is 1.5. Our point estimate is around 2.
Postestimation

Several tools are available to help you analyze your model after you have estimated its parameters. These tools are available after using either \texttt{dsge} or \texttt{dsge}. Below, we continue with the model we fit using \texttt{dsge}. The results would be the same if you used these commands after \texttt{dsge}.

Policy and transition matrices

The matrix of parameters in the state-space form that specifies how the state variables affect the control variables is known as the policy matrix. Each policy matrix parameter is the effect of a one-unit shock to a state variable on a control variable.

We use \texttt{estat policy} to view these results.

```
. estat policy
Policy matrix

<table>
<thead>
<tr>
<th></th>
<th>Delta-method</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Coef.</td>
</tr>
<tr>
<td>p</td>
<td></td>
</tr>
<tr>
<td>u</td>
<td>-.417</td>
</tr>
<tr>
<td>g</td>
<td>.882</td>
</tr>
<tr>
<td>x</td>
<td></td>
</tr>
<tr>
<td>u</td>
<td>-1.58</td>
</tr>
<tr>
<td>g</td>
<td>2.66</td>
</tr>
<tr>
<td>r</td>
<td></td>
</tr>
<tr>
<td>u</td>
<td>.184</td>
</tr>
<tr>
<td>g</td>
<td>1.72</td>
</tr>
</tbody>
</table>
```

Results are listed equation by equation. The first block is the policy equation for inflation \( p \) and writes it as a function of the state variables alone. A unit shock to the state \( u \) reduces inflation by an estimated 0.417, and a unit shock to \( g \) raises inflation by an estimated 0.882.

The matrix of parameters that specifies the dynamic process for the state variables is known as the state transition matrix. The state transition equation relates the future values of the state variables to their values in the current period. Each state transition matrix parameter is the effect of a one-unit shock to a state variable on its one-period-ahead mean.
. estat transition

Transition matrix of state variables

<table>
<thead>
<tr>
<th></th>
<th>Delta-method</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Coef.</td>
<td>Std. Err.</td>
<td>z</td>
<td>P&gt;</td>
<td>z</td>
</tr>
<tr>
<td>F.u</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>u</td>
<td>.6989189</td>
<td>.0449192</td>
<td>15.56</td>
<td>.000</td>
<td>.6108789 , .7869588</td>
</tr>
<tr>
<td>g</td>
<td>1.11e-16</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>F.g</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>u</td>
<td>0 (omitted)</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>g</td>
<td>.9556407</td>
<td>.0181342</td>
<td>52.70</td>
<td>.000</td>
<td>.9200983 , .9911831</td>
</tr>
</tbody>
</table>

Note: Standard errors reported as missing for constrained transition matrix values.

Both state variables are modeled as autoregressive processes, so the results in estat transition repeat the estimates of \( \rho_{uu} \) and \( \rho_{ug} \) from the dsge output. In this case, the other entries in the state transition matrix are 0 or differ from 0 only because of lack of numerical precision. In more complicated models, such as those in which a state equation depends on a control variable, the state transition matrix will contain new information about that state variable.

**Impulse responses**

The state-space form allows us to trace the path of a control or state in response to a shock to a state. This path is known as an impulse–response function (IRF). irf after dsge or dsgenl estimates IRFs, and it puts the named set of estimates into an .irf file, whose results can be displayed using irf graph or irf table.

To graph the IRF, we first create the nkirf.irf file and set it as the active .irf file using the irf set command.

```
. irf set nkirf.irf
(file nkirf.irf created)
(file nkirf.irf now active)
```

Next, we use irf create to estimate a complete set of impulse responses based on our dsge command. A complete set of impulse responses is an impulse to each shock and the response of each state and control variable to that impulse. For the model in this example, irf create generates an impulse to e.u and e.g, then stores the response to that impulse on p, x, r, g, and u. The results are stored in the nkirf.irf file.

```
. irf create model1
(file nkirf.irf updated)
```

We now use irf graph to plot the impulse responses. The impulse() and response() options control which impulse and which responses are chosen. To view the response of p, x, r, and u to a shock to u, we type
The state variable $u$ models movements in the interest rate that occur for reasons other than the feedback between inflation and the interest rate. A shock to $u$ is effectively a surprise increase in the interest rate, and the IRF traces out how this shock causes temporary decreases to inflation (top-left graph) and to the output gap (bottom-right graph).

**Forecasts**

The forecast suite of commands produces dynamic forecasts from the fitted model.

We first store the dsge estimation results.

```
. estimates store dsge_est
```

We use `tsappend()` to extend the dataset by 3 years, or 12 quarters.

```
. tsappend, add(12)
```

To set up a forecast, we perform three steps. First, we type `forecast create` to initialize a new forecasting model, which we name `dsgemodel`.

```
. forecast create dsgemodel
Forecast model dsgemodel started.
```

Next, we add the estimates from `dsge` to the forecasting model using `forecast estimates`.

```
. forecast estimates dsge_est
  Added estimation results from dsge.
Forecast model dsgemodel now contains 2 endogenous variables.
```

This command adds the estimates stored in `dsge_est` to the model `dsgemodel`. Finally, we produce dynamic forecasts beginning in the first quarter of 2017 using `forecast solve`. The `prefix(d1_)` option specifies the `d1_` prefix that will be given to the variables created by `forecast`. We also request that dynamic forecasts begin in the first quarter of 2017 with the `begin(tq(2017q1))` option.
. forecast solve, prefix(d1_) begin(tq(2017q1))
Computing dynamic forecasts for model dsgemodel.

Starting period: 2017q1
Ending period: 2020q1
Forecast prefix: d1_
2017q1: ..............
2017q2: ..............
2017q3: ..............
2017q4: ..............
2018q1: ..............
2018q2: ..............
2018q3: ..............
2018q4: ..............
2019q1: ..............
2019q2: ..............
2019q3: ..............
2019q4: ..............
2020q1: ..............

Forecast 2 variables spanning 13 periods.

The dynamic forecast begins in the first quarter of 2017, so all forecasts are out of sample. We can graph the forecast for inflation d1_p using tsline.

. tsline d1_p if tin(2010q1, 2021q1), tline(2017q1)

The model forecasts that inflation will smoothly return to its long-run value, the sample mean. We can also begin the forecast during a time period for which observations are available.
Specifying the `begin(tq(2014q1))` option produces dynamic forecasts beginning in the first quarter of 2014, so we can compare the forecast for 2014–2016 with the actual observations over that period.

```
. forecast solve, prefix(d2_) begin(tq(2014q1))
Computing dynamic forecasts for model dsgemodel.
```

```
Starting period:  2014q1
Ending period:   2020q1
Forecast prefix: d2_

2014q1: ..............
2014q2: ..............
2014q3: ..............
2014q4: ..............
2015q1: ..............
2015q2: ..............
2015q3: ..............
2015q4: ..............
2016q1: ..............
2016q2: ..............
2016q3: ..............
2016q4: ..............
2017q1: ..............
2017q2: ..............
2017q3: ..............
2017q4: ..............
2018q1: ..............
2018q2: ..............
2018q3: ..............
2018q4: ..............
2019q1: ..............
2019q2: ..............
2019q3: ..............
2019q4: ..............
2020q1: ..............
```

Forecast 2 variables spanning 25 periods.

```
. tsline p d2_p if tin(2010q1, 2021q1), tline(2014q1)
```
We plot both the observed inflation and the forecast.

The forecast captures the general upward trend in inflation from 2014–2016, but it does not predict the variation in inflation around the upward trend.

**Structural and reduced forms of DSGE models**

Now that we have worked an example, we show how it fits in the more general formulation of DSGE models. The model in (4a)–(8a) is an example of a nonlinear DSGE model. In general, a nonlinear DSGE model can be expressed as

\[ E_t \{ f(x_{t+1}, y_{t+1}, x_t, y_t, \theta) \} = 0 \]  

where \( f \) is a vector of equations, \( x_t \) is a vector of state variables, \( y_t \) is a vector of control variables, and \( \theta \) denotes the vector of structural parameters. To solve a DSGE model means to write it in state-space form. In general, the state-space form of a nonlinear DSGE model is nonlinear. \texttt{dsge\_nl} takes a linear approximation to the model equations to solve for a linear approximation to the state-space form.

The model in (4b)–(8b) is an example of a linearized DSGE model. In general, a linearized DSGE model can be expressed as

\[ A_0 y_t = A_1 E_t (y_{t+1}) + A_2 y_t + A_3 x_t \]  
\[ B_0 x_{t+1} = B_1 E_t (y_{t+1}) + B_2 y_t + B_3 x_t + C \epsilon_{t+1} \]  

where as before \( y_t \) is a vector of control variables, \( x_t \) is a vector of state variables, and \( \epsilon_t \) is a vector of shocks. \( A_0 \) through \( A_3 \) and \( B_0 \) through \( B_3 \) are matrices of parameters. We require that \( A_0 \) and \( B_0 \) be diagonal matrices. The entries in all of these matrices are functions of the structural parameters, which we denote by vector \( \theta \). Economic theory places restrictions on the entries in the matrices. \( C \) is a selection matrix that determines which state variables are subject to shocks.

The reduced form of a DSGE model expresses the control variables as functions of the state variables alone and specifies how state variables evolve over time. The reduced form is a state-space model. The state-space form of the model is given by

\[ y_t = G x_t \]  
\[ x_{t+1} = H x_t + M \epsilon_{t+1} \]
where $y_t$ is a vector of control variables, $x_t$ is a vector of state variables, and $\epsilon_t$ is a vector of shocks. $G$ is the policy matrix, and $H$ is the state transition matrix. $M$ is diagonal and contains the standard deviations of the shocks.

$y_t$ is partitioned into observed and unobserved controls, $y_t = (y_{1,t}, y_{2,t})$. The observed control variables are related to the control variables by the equation

$$y_{1,t} = Dy_t$$

where $D$ is a selection matrix. Only observed control variables play a role in estimation. The number of observed control variables must be the same as the number of state equations that include shocks.

You specify a model of the form (9) to `dsge`1 and a model of the form (10)–(11) to `dsge`. Many models require some manipulation to fit into the structure in (9) and (10); see [DSGE] Intro 4 for details. Postestimation commands `estat policy` and `estat transition` will display the policy and transition matrices in (12) and (13), respectively.

References


Also see

[DSGE] dsge — Linear dynamic stochastic general equilibrium models
[DSGE] dsge postestimation — Postestimation tools for dsge
[DSGE] dsge postestimation — Postestimation tools for dsge
[DSGE] dsge postestimation — Postestimation tools for dsge
[TS] forecast — Econometric model forecasting
[TS] irf — Create and analyze IRFs, dynamic-multiplier functions, and FEVDs
[TS] sspace — State-space models
Description

In this introduction, we demonstrate how to specify a DSGE model using the `dsge` and `dsgenl` commands. We focus on two unique aspects of DSGEs that must be considered when writing the syntax—writing the system of equations and identifying each type of variable within those equations.

Remarks and examples

Remarks are presented under the following headings:

- **Introduction**
- **Syntax for linear DSGE models**
  - Preview of `dsge` syntax
  - Specifying the system of linear equations
  - Control variables
  - State variables and shocks
  - Expectations of future values of control variables
  - Specifying parameters using `dsge`'s substitutable expressions
- **Syntax for nonlinear DSGE models**
  - Preview of `dsgenl` syntax
  - Specifying the system of nonlinear equations
  - State and control variables
  - Expectations in nonlinear models

Introduction

If you have not read [DSGE Intro 1](#), we recommend that you read it first. In particular, see *Structural and reduced forms of DSGE models* in [DSGE Intro 1](#), where we discuss the structural forms of a DSGE model that are required by `dsge` and `dsgenl`. Below, we assume that your model has this structural form, and we discuss the syntax for specifying such a model with the `dsge` and `dsgenl` commands.

`dsge` estimates the parameters of linear DSGE models. `dsgenl` estimates the parameters of nonlinear DSGE models. The two commands have elements in common. Both center around a list of equations and a collection of variables. The two commands differ somewhat in how they expect you to specify equations and variables. Below, we first describe the syntax for `dsge` and then for `dsgenl`. 
Syntax for linear DSGE models

Preview of dsge syntax

As an example, if we wanted to fit the DSGE model

\[
\begin{align*}
\pi_t &= \beta E_t(\pi_{t+1}) + \kappa y_t \\
y_t &= E_t(y_{t+1}) - \{r_t - E_t(\pi_{t+1}) - \rho z_t\} \\
\beta r_t &= \pi_t + \beta u_t \\
z_{t+1} &= \rho z_t + \epsilon_{t+1} \\
u_{t+1} &= \delta u_t + \xi_{t+1}
\end{align*}
\]

we would type

```
. dsge (p = {beta}*F.p + {kappa}*y, unobserved) ///
    (y = F.y -(r - F.p - {rho}*z), unobserved) ///
    ({beta}*r = p + {beta}*u) ///
    (F.z = {rho}*z, state) ///
    (F.u = {delta}*u, state)
```

The syntax looks a lot like the equations. In what follows, we discuss each element of the `dsge` syntax that you will need to specify the equations of your DSGE model.

In [DSGE] Intro 1, we told you that three types of variables—control variables, state variables, and shocks—appear in DSGE models. We will demonstrate how to specify equations involving each of these types of variables. We will also show you how to specify the required types of equations that are linear in these variables and nonlinear in the parameters.

Specifying the system of linear equations

For a linear DSGE model, we specify one equation for each control variable and one equation for each state variable. The equations are bound by parentheses. Therefore, the basic structure of the `dsge` command is

```
. dsge (eq1) (eq2) ...,
```

where `eq1` and `eq2` will be replaced with the syntax representing one of the equations in our model. We use `...` to indicate that we can include more than two equations as well as options.

The basic form of an equation within the parentheses is

```
term = term [ + term [ + term [ ...]]]
```

where each `term` includes a variable name. The variable name may be preceded by a parameter or nonlinear combination of parameters. For instance, a valid equation might look something like

```
(y = {kappa}*z)
```

or

```
(y = {kappa}*z + {kappa}*{beta}*x)
```

or

```
(1/{beta}*y = {gamma}*z + x)
```
We will explain this later. For now, simply note that these equations are written in \texttt{dsge} syntax much like we would write the math, but the parameters we want to estimate are offset with curly braces, \{\}. If you have used any other Stata commands that work with substitutable expressions, you may recognize this notation. In fact, \texttt{dsge} uses a special form of substitutable expressions.

Before we discuss how to use \texttt{dsge}'s substitutable expressions, we will focus on how to specify each type of variable.

**Control variables**

Control variables can be modeled as a function of other control variables, expectations of the future value of control variables, and state variables. The equations for control variables do not include shocks.

We continue from the basic \texttt{dsge} command in the previous section, which was

\[
\texttt{dsge (eq1) (eq2) \ldots, \ldots}
\]

If we have an observed control variable \(y\), and for \(eq1\), we want to specify that \(y_t\) be modeled as a function of \(z_t\) and \(x_t\), we can type this equation in our \texttt{dsge} command as

\[
\texttt{dsge (y = pexp*z + pexp*x) ///}
\]

\[
\texttt{(eq2) ///}
\]

\[
\ldots, \ldots
\]

where \(pexp\) is a possibly nonlinear expression of parameters in each term.

Control variables can be observed or unobserved. Because we did not include any options within this set of parentheses, \(y\) is assumed to be an observed control variable. If \(y\) were unobserved, we would add the \texttt{unobserved} option as follows:

\[
\texttt{(y = pexp*z + pexp*x, unobserved)}
\]

Note that each control variable in the model must be included on the left-hand side of one, and only one, equation.

**State variables and shocks**

To model a state variable, we specify an equation with the one-period lead of that state variable on the left-hand side. On the right-hand side of the equation, we can include state variables, control variables, expectations of the future value of control variables, and shocks.

When we specify an equation for a state variable, we include the \texttt{state} option within the parentheses defining the equation.

Continuing with the syntax above, let's suppose \(eq2\) is an equation for a state variable \(x\), and we model \(x_{t+1}\) as a function of \(x_t\). We expand our \texttt{dsge} command to

\[
\texttt{dsge (y = pexp*z + pexp*x) ///}
\]

\[
\texttt{(F.x = pexp*x, state) ///}
\]

\[
\ldots, \ldots
\]

We used the \texttt{F.} lead operator to specify the one-period lead of \(x\) as \texttt{F.x}; see \[U\] \texttt{11.4.4 Time-series varlists}. However, notice that the full list of time-series operators is not available here. We can specify only equations for one-period leads of state variables; we could not replace \texttt{F.x} with \texttt{F2.x} in the equation above. Note that this restriction does not limit the types of models we can fit; see \[DSGE\] \texttt{Intro 4c}. 

By default, the equation for a state variable includes an unobserved shock. However, equations for state variables are not required to include a shock. Within the system of equations, the number of shocks should be equal to the number of observed control variables. If we did not wish to include a shock in the equation for $x_{t+1}$, we could add the noshock option,

$$(F.x = pexp*x, \text{state noshock})$$

Note that the one-period lead of each state variable must be included on the left-hand side of one, and only one, equation.

**Expectations of future values of control variables**

Expectations of the one-period lead of control variables can appear in equations for both control and state variables. Mathematically, we write these expectations as $E_t(\cdot)$. For instance, we write the expectation of $y$ in time $t + 1$ as $E_t(y_{t+1})$. In dsge, we write this expectation as $F.y$. Any time dsge sees the $F.$ operator applied to a control variable, it interprets this as the expectation of that variable at time $t + 1$.

If we model $y_t$ as a function of $E_t(y_{t+1})$ in addition to $z_t$ and $x_t$, we can expand our previous dsge command to

```
 dsge (y = pexp*F.y + pexp*z + pexp*x) ///
 (F.x = pexp*x, \text{state}) ///
 ..., ...
```

Expectations are strictly for one-period leads of control variables. You cannot, for instance, use $F2.y$ to include the expectation of $y$ in time $t + 2$ in the model. This does not prevent you, however, from including such terms in your model. See [DSGE] Intro 4d for details of fitting models, including expectations of control variables more than one period in the future.

**Specifying parameters using dsge’s substitutable expressions**

At this point, we know how to specify each type of variable that may arise in our DSGE model. We now turn to specifying the parameters that we want to estimate.

Recall that the basic form of an equation is

$$term = term [ + term [ + term [ ... ]]]$$

where each $term$ includes one variable name. The variable name may be preceded by a parameter or a nonlinear combination of parameters. For $terms$ on the right-hand side of the equation, variable names can also be followed by a parameter specification.

We specify the $terms$ using a special type of substitutable expressions that we call scalar substitutable expressions. In scalar substitutable expressions, parameters are enclosed in curly braces, ${}$, and may enter the model either linearly or nonlinearly. The restriction that each term includes only one variable implies that the variables must enter the equation linearly with these scalar substitutable expressions.

If our model for $y$ in the equation above is

$$y_t = \beta E_t(y_{t+1}) + \kappa z_t + \gamma x_t$$

we could extend our previous dsge command as follows:

```
 dsge (y = {beta}*F.y + {kappa}*z + {gamma}*x) ///
 (F.x = pexp*x, state) ///
 ..., ...
```
However, the equation for $y$ need not be linear in the parameters. If instead we wanted to model $y$ as

$$y_t = (1/\beta)E_t(y_{t+1}) + \kappa z_t + (\gamma/\beta)x_t$$

we would change our command to

```
. dsge (y = 1/{beta}*F.y + {kappa}*z + ({gamma}/beta)*x) ///
   (F.x = pexp*x, state) ///
   ..., ...
```

We can even include parameters on the left-hand side of the equation. For instance, we could model $y$ as

$$\frac{1}{\beta}y_t = E_t(y_{t+1}) + \kappa z_t + \frac{\gamma}{\beta}x_t$$

and change our `dsge` command to

```
. dsge ((1/{beta})*y = F.y + {gamma}*x + {kappa}*z) ///
   (F.x = pexp*x, state) ///
   ..., ...
```

Finally, there is an extension to the basic form of the equation that we should mention. Sometimes, it is more convenient to write an equation so that a scalar value or a parameter or a nonlinear combination of parameters is multiplied by a linear combination of terms. The `dsge` command allows this. For instance, instead of

$$y_t = (1/\beta)E_t(y_{t+1}) + \kappa z_t + \frac{\gamma}{\beta}x_t$$

we can write

$$y_t = (1/\beta)\{E_t(y_{t+1}) + \gamma x_t\} + \kappa z_t$$

Similarly, we could write this equation in the `dsge` command as

```plaintext
(y = (1/{beta}) * (F.y + {gamma}*x) + {kappa}*z)
```

Now you know the general syntax of `dsge` and are ready to fit your own DSGE model with `dsge`. For examples of fitting classic DSGE models using this syntax, see [DSGE] Intro 3a, [DSGE] Intro 3b, and [DSGE] Intro 3c. For more details on the `dsge` syntax, see [DSGE] dsge.

### Syntax for nonlinear DSGE models

The `dsge` syntax builds on the `dsge` syntax. However, because of the additional complexity of nonlinear models, the syntax does differ. As we discuss the `dsge` syntax below, we will highlight similarities and differences from `dsge`.

---

---
Introduction to Learning the Syntax

Preview of 
Suppose we wanted to estimate the parameters of the nonlinear DSGE model,
\[ R_t = \frac{Y_t}{K_t} \]
\[ 1 = \beta E_t \left\{ \left( \frac{C_t}{C_{t+1}} \right) (1 + R_{t+1} - \delta) \right\} \]
\[ Y_t = Z_t K_t^{\alpha} \]
\[ K_{t+1} = Y_t - C_t + (1 - \delta) K_t \]
\[ \ln(Z_{t+1}) = \rho \ln(Z_t) + e_{t+1} \]

To estimate the parameters of this model, we would type

\[
\text{. dsgenl (r = \{alpha\}*y/k) /// } \\
(1 = \{beta\}*(c/F.c)*(1 + F.r - \{delta\})) /// \\
(y = z*k^{\{alpha\}}) /// \\
(F.k = y - c + (1-\{delta\})*k) /// \\
(ln(F.z) = \{rho\}*ln(z)) , /// \\
\text{observed(y) unobserved(c r) exostate(z) endostate(k)}
\]

As with \texttt{dsge}, the \texttt{dsgenl} syntax looks a lot like the equations. Variables are written as normal, and parameters are written bound by curly braces. We first focus on two syntax elements: the system of equations that specifies the interrelations among variables and the options that specify how each variable is treated.

Specifying the system of nonlinear equations

For a nonlinear DSGE model, we specify a system of equations that jointly determines the structure of the model. There must be as many equations as there are control and state variables. The equations are bound by parentheses. Therefore, the basic structure of the \texttt{dsgenl} command is

\[
\text{. dsgenl (eq1) (eq2) \ldots, \ldots}
\]

where \texttt{eq1} and \texttt{eq2} will be replaced with the syntax representing one of the equations in our model. We use \ldots to indicate that we can include more than two equations as well as options.

The equations for a nonlinear model fit with \texttt{dsgenl} take on a much more flexible form than the equations we described for a linear model fit by \texttt{dsge}. For \texttt{dsgenl}, the equation within the parentheses may include a nonlinear substitutable expression on both sides of the equal sign. An example of such an expression is the first equation in the model above, which reads

\[
(r = \{alpha\}*y/k)
\]

The equations written in \texttt{dsgenl}, like those in \texttt{dsge}, use curly braces to identify the parameters such as \texttt{alpha} that are to be estimated.

This equation could have been expressed in other ways. As written above, it reads like an equation for \texttt{r}. We could just as well have written it as

\[
(k = \{alpha\}*y/r)
\]

or

\[
(r*k/y = \{alpha\})
\]
or even
\[(r \times k / y - \{\text{alpha}\} = 0)\]

All four of the above Stata equations specify the same mathematical equation, and it does not matter which parameterization you choose.

The second equation in the model above reads
\[(1 = \{\beta}\times(c/F.c)\times(1 + F.r - \{\text{delta}\}))\]

We could have written it as
\[(1/c = \{\beta}\times(1/F.c)\times(1 + F.r - \{\text{delta}\}))\]
or
\[(1 = \{\beta}\times(F.c/c)^{-1}\times(1 + F.r - \{\text{delta}\}))\]

The three equations just listed specify the same mathematical equation, and it does not matter which parameterization you choose.

In the second equation of the model, the \(F\). lead operator denotes the future value of a variable, just as it did in \textit{dsge}. More information on how \textit{dsgenl} treats the \(F\). operator is below, under \textit{Expectations in nonlinear models}.

### State and control variables

So far, we have discussed only the specification of model equations. We have seen that you can specify the model equations in nearly any form. The price of that freedom is that you must specify a set of options—\textit{exostate()}, \textit{endostate()}, \textit{observed()}, and \textit{unobserved()}—that tell Stata what role each variable plays in the model.

Recall that a DSGE model has control and state variables. Each control variable may be observed or unobserved, and each state variable may be subject to shocks. Hence, each variable fits into one of four types: observed controls, unobserved controls, a state variable that is not subject to a shock, and a state variable that is subject to a shock. The above model has variables of all four types.

Using the jargon common in macroeconomics, we call a state variable that is subject to a shock an exogenous state variable. We call a state variable that is not subject to a shock an endogenous state variable. You must have exactly as many exogenous state variables (and thus, as many shocks) as you have observed control variables.

In the model above, the option list
\[. dsgenl \ldots, \ldots \text{observed}(y) \text{ unobserved}(c r) \text{ exostate}(z) \text{ endostate}(k) \ldots\]
specifies that \(c\), \(r\), and \(y\) are control variables; of those, \(y\) is observed. The option list specifies \(k\) and \(z\) as state variables, of which \(z\) is subject to shocks.

### Expectations in nonlinear models

In \textit{Expectations of future values of control variables}, we saw that terms in models fit with \textit{dsge} can include the expectation of the one-period lead of a variable. Such expectations can also be included in models fit with \textit{dsgenl}. Because the equations in \textit{dsgenl} are more flexible, we need to think of the expectations in a more general way, as we describe below.

Expectations of the one-period lead of variables can appear in any equation. Mathematically, we write these expectations as \(E_t(\cdot)\). For instance, we write the expectation of \(y\) in time \(t + 1\) as \(E_t(y_{t+1})\). As we did in \textit{dsge}, we write this expectation as \(F.y\).
In a nonlinear model, however, the expectation may not be applied directly to one variable. We might have

\[ 1 = \beta E_t \left\{ \left( \frac{C_t}{C_{t+1}} \right) (1 + R_{t+1} - \delta) \right\} \]

where the expectation is written as if it is applied to everything in the curly brackets. Really, the expectation can be thought of as being applied to the entire equation. This means we can use the F. lead operator and write the equation involving the expectation above as

```
    . dsgenl (1 = {beta}*(c/F.c)*(1 + F.r - {delta})) ... , ...
```

in dsgenl, and it will be interpreted as

\[ E_t \left\{ 1 - \beta \left( \frac{C_t}{C_{t+1}} \right) (1 + R_{t+1} - \delta) \right\} = 0 \]

The upshot is that you can specify expected future values by using the F.x notation, and both dsgenl and dsge will take the appropriate action.

Now you have seen all elements of the general syntax dsgenl and are ready to fit your own model. For examples of fitting classic nonlinear DSGE models, see [DSGE] Intro 3d, [DSGE] Intro 3e, and [DSGE] Intro 3f. For more details on the dsgenl syntax, see [DSGE] dsgenl.

Also see

[DSGE] dsge — Linear dynamic stochastic general equilibrium models
[DSGE] dsgenl — Nonlinear dynamic stochastic general equilibrium models
[DSGE] Intro 3 — Classic DSGE examples
[DSGE] Intro 4 — Writing a DSGE in a solvable form
Description

In this entry, we present several classic DSGE examples. These include linear and nonlinear versions of a New Keynesian model, linear and nonlinear versions of a New Classical model (Real Business Cycle model), a linear financial frictions model, and a nonlinear stochastic growth model.

Remarks and examples

In [DSGE Intro 3a], [DSGE Intro 3b], and [DSGE Intro 3c], we fit simple variants of common linearized DSGE models. In [DSGE Intro 3d], [DSGE Intro 3e], and [DSGE Intro 3f], we fit simple variants of common nonlinear DSGE models. Through these examples, we demonstrate model solution, estimation, and interpretation.

[DSGE Intro 3a] demonstrates how to fit a New Keynesian model. In this example, we interpret structural parameters, policy matrix parameters, and state transition matrix parameters. We also predict values of both observed control variables and unobserved states.

[DSGE Intro 3b] illustrates how to solve a New Classical model and plot the IRFs to compare the model’s theoretical predictions under different sets of parameter values.

[DSGE Intro 3c] fits a financial frictions model. In this example, we also estimate parameters of the policy matrix and evaluate the IRFs.

[DSGE Intro 3d] revisits the New Keynesian model. In this example, we estimate the parameters of the nonlinear model, interpret structural parameters, and explain partial identification by parameter restrictions in a DSGE model.

[DSGE Intro 3e] revisits the New Classical model. We fix some parameters and estimate others, explain the effect of fixing parameters on postestimation statistics, and explain how to compare models across different parameter settings.

[DSGE Intro 3f] demonstrates how to solve a nonlinear stochastic growth model. We explain how dsgenl takes an approximation of the model and how to interpret the steady state and approximations to the policy and transition equations. This example describes some of the technical differences between linear and log-linear approximations to nonlinear DSGE models.

Also see

[DSGE Intro 4] — Writing a DSGE in a solvable form
[DSGE dsge] — Linear dynamic stochastic general equilibrium models
[DSGE dsgenl] — Nonlinear dynamic stochastic general equilibrium models
Description

This introduction estimates and interprets the parameters of a simple New Keynesian model. In this entry, we demonstrate how to constrain parameters in the model and how to interpret structural parameters, policy matrix parameters, and state transition matrix parameters. We also predict values of both observed control variables and unobserved states.

Remarks and examples

Remarks are presented under the following headings:

The model
Parameter estimation
Policy and transition matrices
One-step-ahead predictions
Estimating an unobserved state

The model

Equations (1)–(5) specify a canonical New Keynesian model of inflation $p_t$, the output gap $x_t$, and the interest rate $r_t$. These are the linearized equations; the model’s nonlinear equations are similar to those in Writing down nonlinear DSGEs in [DSGE] Intro 1.

\begin{align*}
    p_t &= \beta E_t(p_{t+1}) + \kappa x_t \\
    x_t &= E_t(x_{t+1}) - \{r_t - E_t(p_{t+1}) - g_t\} \\
    r_t &= \psi p_t + u_t \\
    u_{t+1} &= \rho u_t + \epsilon_{t+1} \\
    g_{t+1} &= \rho g_t + \xi_{t+1}
\end{align*}

Equation (1) specifies inflation as a linear combination of expected future inflation and the output gap. Equation (2) specifies the output gap as a linear combination of the expected future output gap, the real interest rate, and a state variable $g_t$. Equation (3) specifies the interest rate as a linear combination of inflation and a state variable $u_t$. The state variables are modeled as first-order autoregressive processes. The state variable $u_t$ is the deviation of $r_t$ from its equilibrium value of $\psi p_t$. The state variable $g_t$ is also the deviation of $x_t$ from its equilibrium value.

Three of the parameters have structural interpretation. The parameter $\kappa$ is known as the slope of the Phillips curve and is predicted to be positive. The parameter $\beta$ is the discount factor that represents the degree to which agents discount the future relative to the current period. The parameter $\psi$ measures the degree to which interest rates react to movements in inflation.
Parameter estimation

Not all model parameters are identified. We constrain $\beta$ to be 0.96, a common value in the literature. The remaining parameters are identified.

```
. use https://www.stata-press.com/data/r16/usmacro2
(Federal Reserve Economic Data - St. Louis Fed, 2017-01-15)
. constraint 1 _b[beta]=0.96
. dsge (p = {beta}*F.p + {kappa}*x)
  >     (x = F.x - (r - F.p - g), unobserved)
  >     (r = {psi}*p + u)
  >     (F.u = {rhou}*u, state)
  >     (F.g = {rhog}*g, state),
  >     from(psi=1.5) constraint(1)
(setting technique to bfgs)
Iteration 0:  log likelihood =  -5736.4646
Iteration 1:  log likelihood =  -1190.0604  (backed up)
Iteration 2:  log likelihood =  -960.00953  (backed up)
Iteration 3:  log likelihood =  -928.79225  (backed up)
Iteration 4:  log likelihood =  -842.40806  (backed up)
(switching technique to nr)
Iteration 5:  log likelihood =  -810.92149  (backed up)
Iteration 6:  log likelihood =  -766.17852  (not concave)
Iteration 7:  log likelihood =  -760.42901
Iteration 8:  log likelihood =  -756.77337
Iteration 9:  log likelihood =  -754.05236
Iteration 10: log likelihood =  -753.58052
Iteration 11: log likelihood =  -753.57142
Iteration 12: log likelihood =  -753.57131
Iteration 13: log likelihood =  -753.57131
DSGE model
Sample: 1955q1 - 2015q4  Number of obs  =  244
Log likelihood =  -753.57131
( 1)  /[structural]beta = .96


<table>
<thead>
<tr>
<th></th>
<th>OIM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Coef.    Std. Err.   z     P&gt;</td>
</tr>
<tr>
<td>/structural</td>
<td></td>
</tr>
<tr>
<td>beta</td>
<td>.96     (.96)</td>
</tr>
<tr>
<td>kappa</td>
<td>.0849633  .0287693  2.95   0.003  .0285764  .1413502</td>
</tr>
<tr>
<td>psi</td>
<td>1.943007  .2957876  6.57   0.000  1.363274  2.52274</td>
</tr>
<tr>
<td>rhou</td>
<td>.7005481  .0452603  15.48  0.000  .6118395  .7892568</td>
</tr>
<tr>
<td>rhog</td>
<td>.9545256  .0186424  51.20  0.000  .9179872  .991064</td>
</tr>
<tr>
<td>sd(e.u)</td>
<td>2.318207  .3047443  1.72   0.089  1.920917  2.915495</td>
</tr>
<tr>
<td>sd(e.g)</td>
<td>.5689901  .0982974  .37   0.710  .3763308  .7616494</td>
</tr>
</tbody>
</table>

The slope of the Phillips curve, kappa, is estimated to be positive. The coefficient on inflation in the interest rate equation is estimated to be almost 1.94, meaning that interest rates are expected to rise almost two for one with increases in inflation.
Policy and transition matrices

Elements of the policy matrix represent the response of a control variable to a one-unit increase in a state variable.

```
. estat policy
Policy matrix
```

<table>
<thead>
<tr>
<th>Delta-method</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Coef. Std. Err.</td>
<td>z</td>
<td>P&gt;</td>
<td>z</td>
<td></td>
</tr>
<tr>
<td>p u</td>
<td>-0.4172517</td>
<td>0.039361</td>
<td>-10.60</td>
<td>0.000</td>
</tr>
<tr>
<td>g</td>
<td>0.9678148</td>
<td>0.2777441</td>
<td>3.48</td>
<td>0.000</td>
</tr>
<tr>
<td>x u</td>
<td>-1.608212</td>
<td>0.405024</td>
<td>-3.97</td>
<td>0.000</td>
</tr>
<tr>
<td>g</td>
<td>0.9529166</td>
<td>0.481391</td>
<td>1.98</td>
<td>0.048</td>
</tr>
<tr>
<td>r u</td>
<td>0.1892771</td>
<td>0.059166</td>
<td>3.20</td>
<td>0.001</td>
</tr>
<tr>
<td>g</td>
<td>1.880471</td>
<td>0.2615988</td>
<td>7.19</td>
<td>0.000</td>
</tr>
</tbody>
</table>

An increase in \( u \) decreases the extent to which the inflation is above its short-run equilibrium value. This change decreases the output gap and increases interest rate.

An increase in \( g \) increases the extent to which the inflation is above its short-run equilibrium value. This change also increases output gap and interest rates.

Because the states are uncorrelated with each other in this example, the elements of the state transition matrix are just the persistence parameters in the model.

```
. estat transition
Transition matrix of state variables
```

<table>
<thead>
<tr>
<th>Delta-method</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Coef. Std. Err.</td>
<td>z</td>
<td>P&gt;</td>
<td>z</td>
<td></td>
</tr>
<tr>
<td>F.u u</td>
<td>0.7005481</td>
<td>0.0452603</td>
<td>15.48</td>
<td>0.000</td>
</tr>
<tr>
<td>g</td>
<td>1.67e-16</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>F.g u</td>
<td>0 (omitted)</td>
<td>51.20</td>
<td>0.000</td>
<td>0.9179872</td>
</tr>
<tr>
<td>g</td>
<td>0.9545256</td>
<td>0.0186424</td>
<td>51.20</td>
<td>0.000</td>
</tr>
</tbody>
</table>

Note: Standard errors reported as missing for constrained transition matrix values.

See [DSGE] Intro 4g for an example in which the state variables depend on each other. The states can also depend on each other when a state variable is specified to depend on control variables, as in [DSGE] Intro 3b, or when the state vector includes lagged control variables, as in [DSGE] Intro 4a.
One-step-ahead predictions

Predictions after dsge depend on the estimated state-space parameters. Below, we obtain one-step-ahead predictions for each of the two observed control variables in the model, and we graph the actual and predicted inflation rates.

. predict dep*
   (option xb assumed; fitted values)
. tsline p dep1, legend(col(1))

The graph shows that the one-step-ahead predictions closely follow realized inflation.

Estimating an unobserved state

The observed control variables are driven by two unobserved state variables. We can use predict with the state option to estimate the state variables.
Here we estimate the unobserved state $u$ and plot it.

```
predict state1, state
.tsline state1
```

The loose monetary policy in the mid-1970s, where predicted values of $u_t$ are negative, and the subsequent Volcker contraction are apparent in this plot.

Reference


Also see

[DSGE] Intro 1 — Introduction to DSGEs
[DSGE] Intro 3b — New Classical model
[DSGE] Intro 3c — Financial frictions model
[DSGE] Intro 3d — Nonlinear New Keynesian model
[DSGE] dsge — Linear dynamic stochastic general equilibrium models
[DSGE] dsge postestimation — Postestimation tools for dsge
In this example, we solve a New Classical model similar to the one in King and Rebelo (1999). We also demonstrate how to compare a model’s theoretical predictions under different parameter values using IRFs.

Remarks and examples

Remarks are presented under the following headings:

- The model
- Solving the model
- Policy and transition matrices
- Impulse responses
- Sensitivity analysis

The model

In this model, output, consumption, investment, employment, and other variables are driven by state variables linked to production and demand. The model is similar to the one in King and Rebelo (1999) and is referred to as a real business cycle model.

The nonlinear form of the model is

\[
\frac{1}{C_t} = \beta E_t \left\{ \frac{1}{C_{t+1}}(1 + R_{t+1} - \delta) \right\}
\]  

(1)

\[ H_t^p = \frac{W_t}{C_t} \]  

(2)

\[ Y_t = C_t + X_t + G_t \]  

(3)

\[ Y_t = K_t^\alpha (Z_t H_t)^{1-\alpha} \]  

(4)

\[ W_t = (1 - \alpha) \frac{Y_t}{H_t} \]  

(5)

\[ R_t = \alpha \frac{Y_t}{K_t} \]  

(6)

\[ K_{t+1} = (1 - \delta) K_t + X_t \]  

(7)

Equation (1) specifies consumption \( C_t \) as a function of expected future consumption and the expected future interest rate \( E_t(R_{t+1}) \). Equation (2) specifies labor hours \( H_t \) as a function of the wage \( W_t \) and consumption; it is a labor supply equation. Equation (3) is the national income accounting identity for a closed economy, specifying output \( Y_t \) as the sum of consumption, investment \( X_t \), and government spending \( G_t \). Equation (4) is a production function that specifies output as a function of labor input \( H_t \), capital input \( K_t \), and productivity \( Z_t \). Equations (5) and (6) specify labor demand and capital demand, respectively. Equation (7) specifies the equation for capital accumulation. The model is completed when we add state transition equations for \( Z_t \) and \( G_t \). These state transition equations are conventionally specified after the model has been linearized.
The linearized form of the model is

\[
\begin{align*}
    c_t &= E_t(c_{t+1}) - (1 - \beta + \beta \delta)E_t(r_{t+1}) \\
    \eta h_t &= w_t - c_t \\
    \phi_1 x_t &= y_t - \phi_2 c_t - g_t \\
    y_t &= (1 - \alpha)(z_t + h_t) + \alpha k_t \\
    w_t &= y_t - h_t \\
    r_t &= y_t - k_t \\
    k_{t+1} &= \delta x_t + (1 - \delta)k_t \\
    z_{t+1} &= \rho z_t + \epsilon_{t+1} \\
    g_{t+1} &= \rho g_t + \xi_{t+1}
\end{align*}
\]

The model has six control variables and three state variables. Two of the state variables, \( z_{t+1} \) and \( g_{t+1} \), are modeled as first-order autoregressive processes. The state equation for \( k_{t+1} \) depends on the current value of a control variable, namely, \( x_t \).

**Solving the model**

The `solve` option of `dsge` places the model in state-space form without estimating parameters; it is similar to `iterate(0)` but is faster because it does not calculate standard errors. Using `solve` for different parameter values of your model is a useful way to explore the model’s theoretical properties.

The parameter values used here are similar to those used in King and Rebelo (1999). Each has an interpretation. \((1 - \text{alpha})\) is labor’s share of national income. \( \text{delta} \) is the depreciation rate of capital. \( \text{eta} \) is the slope of the labor supply curve. \( \phi_1 \) and \( \phi_2 \) are share parameters related to investment’s share of national income and consumption’s share of national income, respectively. \( \rho z \) and \( \rho g \) are autoregressive parameters on the state variables.
. use https://www.stata-press.com/data/r16/usmacro2
(Federal Reserve Economic Data - St. Louis Fed, 2017-01-15)
. dsges
(c = F.c - (1-{beta}+{beta}*{delta})*F.r, unobserved)
> ({eta}*h = w - c, unobserved)
> ({phi1}*x = y - {phi2}*c - g, unobserved)
> (y = (1-{alpha})*(z+h) + {alpha}*k)
> (r = y - k, unobserved)
> (F.k = {delta}*x+ (1-{delta})*k, state noshock)
> (F.z = {rhoz}*z, state)
> (F.g = {rhog}*g, state),
> from(beta=0.96 eta=1 alpha=0.3 delta=0.025 phi1=0.2 phi2=0.6 rhoz=0.8 rhog=0.3)
> solve noidencheck

DSGE model
Sample: 1955q1 - 2015q4 Number of obs = 244
Log likelihood = -1957.0261

<table>
<thead>
<tr>
<th></th>
<th>OIM</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Coef.</td>
<td>Std. Err.</td>
<td>z</td>
<td>P&gt;</td>
<td>z</td>
</tr>
<tr>
<td>/structural</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>beta</td>
<td>0.96</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>delta</td>
<td>0.025</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>eta</td>
<td>1</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>phi1</td>
<td>0.2</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>phi2</td>
<td>0.6</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>alpha</td>
<td>0.3</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>rhoz</td>
<td>0.8</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>rhog</td>
<td>0.3</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>sd(e.z)</td>
<td>1</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>sd(e.g)</td>
<td>1</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
</tbody>
</table>

Note: Skipped identification check.
Note: Model solved at specified parameters.

The solve option solves the model at the specified values in from(). We skip the identification check with noidencheck. Simply solving the model does not involve any reference to the data or any estimation. Still, we can explore what these parameters imply.

Policy and transition matrices

After solving, we can use many of the postestimation commands, though standard errors will be missing throughout.

The state transition matrix shows how the state vector in the next period is related to the state vector in the current period. Some state variables are specified as first-order autoregressive processes, and their transition equations will simply repeat information that is already available in the estimation table. However, if any state variable equation contains control variables, then that state variable’s transition equation will depend on the other state variables.
Transition matrix of state variables

| Delta-method Coef. | Std. Err. | z   | P>|z| | [95% Conf. Interval] |
|-------------------|-----------|-----|-------|----------------------|
| F.k | k | .9256785 | . | . | . | . |
|     | z | .1078282 | . | . | . | . |
|     | g | -.1070547 | . | . | . | . |
| F.z | k | 0 (omitted) | . | . | . | . |
|     | z | .8 | . | . | . | . |
|     | g | 2.22e-16 | . | . | . | . |
| F.g | k | 0 (omitted) | . | . | . | . |
|     | z | 0 (omitted) | . | . | . | . |
|     | g | .3 | . | . | . | . |

Note: Standard errors reported as missing for constrained transition matrix values.

The value of the state variables \( z \) and \( g \) in the next period depends only on their value in the current period, but the value of the capital stock \( k \) in the next period depends on the current value of all three state variables. This feature means that, for example, a shock to the \( z \) state variable has two effects: it increases future values of \( z \), because \( z \) is autoregressive, but it also increases future values of \( k \). Interrelationships among the state variables can generate more interesting patterns in the IRFs than the AR(1) dynamics that we saw in [DSGE] Intro 3a.

**Impulse responses**

One way to compare two parameter sets is to graph the impulse response of model variables to a shock under each parameter set. We first set the impulse–response file with `irf set` and then add impulse responses named `persistent` to the file with `irf create`.

```
. irf set rbcirf
  (file rbcirf.irf created)
  (file rbcirf.irf now active)

. irf create persistent
  (file rbcirf.irf updated)
```
The response of model variables to a shock to $z$ is graphed by typing:

```
.irf graph irf, irf(persistent) impulse(z) response(y c x h w z) noci
> byopts(yrescale)
```

Each graph is labeled with the IRF name, the impulse variable, and the response variable. For instance, the top-left graph shows the response of consumption to a shock to the $z_t$ state variable. The bottom-right graph shows the response of the state variable $z_t$ itself. The state is persistent, which is not surprising: we set the autoregressive parameter in the $z_t$ equation to 0.8.

In the top-left graph, we see that consumption $c$ rises over time before returning to steady state. The time unit is quarters, so a value of about 0.27, 4 periods after the shock, indicates that consumption is 0.27% above its steady-state value one year after the shock. Hours worked $h$ are shown in the top center graph and rise initially before falling below steady state. The real wage $w$, output $y$, and investment $x$ all rise.

**Sensitivity analysis**

The responses of variables to a shock to $z$ are persistent. Some variables, like consumption and the wage, show dynamics beyond the simple autoregressive behavior of $z$ itself. To evaluate the role of persistence in $z$ on the persistence of other model variables, we rerun the dsge command. This time, we set the persistence of $z$ to a smaller value of 0.6.
dsge (c = F.c - F.r, unobserved)
> (eta)*h = w - c, unobserved)
> (phi1)*x = y - {phi2}*c - g, unobserved)
> (y = (1-{alpha})*{z+h} + {alpha}*k)
> (w = y - h, unobserved)
> (r = y - k, unobserved)
> (F.k = {delta}*x+ (1-{delta})*k, state noshock)
> (F.z = {rhoz}*z, state)
> (F.g = {rhog}*g, state),
> from(eta=1 alpha=.3 delta=0.025 phi1=0.2 phi2=0.6 rhoz=0.6 rhog=0.3)
> solve noidencheck

DSGE model
Sample: 1955q1 - 2015q4
Number of obs = 244
Log likelihood = -1447.5902

| y       | Coef. | Std. Err. | z    | P>|z| | [95% Conf. Interval] |
|---------|-------|-----------|------|-----|----------------------|
| /structural |      |           |      |     |                       |
| eta     | 1     | .         | .    | .   | .                    |
| phi1    | .2    | .         | .    | .   | .                    |
| phi2    | .6    | .         | .    | .   | .                    |
| alpha   | .3    | .         | .    | .   | .                    |
| delta   | .025  | .         | .    | .   | .                    |
| rhoz    | .6    | .         | .    | .   | .                    |
| rhog    | .3    | .         | .    | .   | .                    |
| sd(e.z) | 1     | .         | .    | .   | .                    |
| sd(e.g) | 1     | .         | .    | .   | .                    |

Note: Skipped identification check.
Note: Model solved at specified parameters.

The only change in the parameter set is that rhoz has been set to 0.6 from its earlier setting of 0.8. We can add the impulse responses of this model to the irf file with the name transitory,

    . irf create transitory, replace
    irfname transitory not found in rbcirf.irf
    (file rbcirf.irf updated)
and graph them.

```
  . irf graph irf, irf(transitory) impulse(z) response(y c x h w z) noci
  > byopts(yrescale)
```

Model variables are much less persistent. We can use `irf ograph` to overlay the IRF for a variable under the two calibrations. This way we can view the differences across calibrations directly.

```
  . irf ograph (persistent z c irf) (transitory z c irf)
```

When the shock itself is persistent, consumption responds persistently. When the shock is transitory, consumption returns to its steady-state value quickly.
Reference


Also see

[DSGE] Intro 1 — Introduction to DSGEs
[DSGE] Intro 3a — New Keynesian model
[DSGE] Intro 3c — Financial frictions model
[DSGE] Intro 3e — Nonlinear New Classical model
[DSGE] dsge — Linear dynamic stochastic general equilibrium models
[DSGE] dsge postestimation — Postestimation tools for dsge
Description

This introduction estimates and interprets the parameters of a model that incorporates financial frictions. The model is an extension of the one in [DSGE Intro 3a].

Remarks and examples

Remarks are presented under the following headings:

The model
Parameter estimation
Policy and transition matrices
Impulse responses

The model

Equations (1)–(7) specify a model of financial frictions. The simplest such model places a wedge between two interest rates: the safe interest rate set by the central bank and the market interest rate that consumers and producers use.

\[
\begin{align*}
\pi_t &= \beta E_t \pi_{t+1} + \kappa x_t \\
x_t &= E_t x_{t+1} - (i_t - E_t \pi_{t+1} - g_t) \\
r_t &= \psi \pi_t + u_t \\
i_t &= \chi r_t + e_t \\
g_{t+1} &= \rho_g g_t + \xi_{t+1} \\
u_{t+1} &= \rho_u u_t + \epsilon_{t+1} \\
e_{t+1} &= \rho_e e_t + \eta_{t+1}
\end{align*}
\]

These are the linearized equations. The nonlinear equations are similar to those in Writing down nonlinear DSGEs in [DSGE Intro 1].

This model is an extension of the one worked in [DSGE Intro 1]. It has an additional equation for the interest rate spread. As before, (1) specifies the inflation equation (a Phillips curve), (2) specifies the output gap equation (an Euler equation), and (3) specifies the equation for the safe interest rate (a Taylor rule). Equation (3) can be thought of as an equation that specifies the safe interest rate \( r_t \). The new element is (4), which specifies an equation for the market interest rate \( i_t \) that enters the output gap equation. The market interest rate \( i_t \) is a function of the safe interest rate and a state variable \( e_t \). The state variable \( e_t \) controls the interest rate spread and can be thought of as representing the state of the financial system. A large realization of \( e_t \) represents a large interest rate spread, indicating financial distress.
Parameter estimation

We estimate the parameters of the model in (1)–(7) using U.S. data on the Federal funds rate (safe rate set by the central bank), the high-grade corporate bond interest rate (a measure of market interest rates), and the inflation rate.

As is typical in these models, we constrain the parameter $\beta$ to 0.96.

```
. use https://www.stata-press.com/data/r16/usmacro2
   (Federal Reserve Economic Data - St. Louis Fed, 2017-01-15)
. constraint 1 _b[beta]=0.96
. dsge (p = {beta}*F.p + {kappa}*x)
   > (x = F.x - (i - F.p - g), unobserved)
   > (i = {chi}*r + e)
   > (r = {psi}*p + u)
   > (F.e = {rhoe}*e, state)
   > (F.u = {rhou}*u, state)
   > (F.g = {rhoz}*g, state),
   > from(psi=2 chi=0.8) constraint(1)
   (setting technique to bfgs)
Iteration 0:  log likelihood = -4780.2037
Iteration 1:  log likelihood = -1731.5956 (backed up)
Iteration 2:  log likelihood = -1315.8819 (backed up)
Iteration 3:  log likelihood = -1161.0796 (backed up)
Iteration 4:  log likelihood = -1115.2257 (backed up)
   (switching technique to nr)
Iteration 5:  log likelihood = -1069.1254 (backed up)
Iteration 6:  log likelihood = -1016.9948 (not concave)
Iteration 7:  log likelihood = -980.93826 (not concave)
Iteration 8:  log likelihood = -922.65912 (not concave)
Iteration 9:  log likelihood = -914.78652 (not concave)
Iteration 10: log likelihood = -908.4795 (not concave)
Iteration 11: log likelihood = -905.35477 (not concave)
Iteration 12: log likelihood = -902.51101 (not concave)
Iteration 13: log likelihood = -900.48635 (not concave)
Iteration 14: log likelihood = -899.17535 (not concave)
Iteration 15: log likelihood = -897.88451 (not concave)
Iteration 16: log likelihood = -895.03731
Iteration 17: log likelihood = -891.34894
Iteration 18: log likelihood = -887.41441
Iteration 19: log likelihood = -882.82925 (not concave)
Iteration 20: log likelihood = -882.73272
Iteration 21: log likelihood = -882.20988
Iteration 22: log likelihood = -882.13462
Iteration 23: log likelihood = -882.13198
Iteration 24: log likelihood = -882.13195
```
DSGE model
Sample: 1955q1 - 2015q4
Number of obs = 244
Log likelihood = -882.13195
( 1) /[structural]beta = .96

| OIM                      | Coef.    | Std. Err. | z     | P>|z|  | [95% Conf. Interval] |
|--------------------------|----------|-----------|-------|------|---------------------|
| /structural              |          |           |       |      |                     |
| beta                     | .96      | (constrained) | | | | |
| kappa                    | .0503238 | .0184922  | 2.72  | 0.007 | .0140796 .0865679  |
| chi                      | .9067346 | .1249748  | 7.26  | 0.000 | .6617885 1.151681  |
| psi                      | 6.332415 | 2.56766   | 2.47  | 0.014 | 1.299893 11.36494 |
| rhoe                     | .8478235 | .0301215  | 28.15 | 0.000 | .7887865 .9068605 |
| rhou                     | .8153493 | .032519   | 24.52 | 0.000 | .7501768 .8805218 |
| rhoz                     | .9861866 | .009767   | 98.85 | 0.000 | .9666327 1.005741 |
| sd(e.e)                  | .8857022 | .1343     |       |      | .622479 1.149925  |
| sd(e.u)                  | 7.16076  | 2.93846   |       |      | 1.410528 12.91099 |
| sd(e.g)                  | .3911864 | .0358355  |       |      | .3209501 4.614226 |

The persistence of the financial shock rhoe is estimated to be 0.85. The slope of the Phillips curve, kappa, is somewhat flatter in this model than in the one in [DSGE] Intro 1. The coefficient on inflation in the interest rate equation is 6.3 and indicates that the central bank increases interest rates much more than one for one in response to movements in inflation.

Policy and transition matrices

We can read off the impact effect of shocks using the policy matrix. The response to the financial shock e will be of most interest.

. estat policy
Policy matrix

| Delta-method                      | Coef.    | Std. Err. | z     | P>|z|  | [95% Conf. Interval] |
|----------------------------------|----------|-----------|-------|------|---------------------|
| p                                |          |           |       |      |                     |
| e                                | -.1832602 | .0740955 | -2.47 | 0.013 | -.3284848 -.380357 |
| u                                | -.1584184 | .0641356 | -2.47 | 0.014 | -.2841219 -.32715  |
| g                                | .2096325  | .1003838  | 2.09  | 0.037 | .0128838 4.063812  |
| x                                |          |           |       |      |                     |
| e                                | -.6776679 | .348929   | -1.94 | 0.053 | -.1363445 .0081091 |
| u                                | -.6839457 | .2783497  | -2.46 | 0.014 | -.1229501 -.1383903 |
| g                                | .2218675  | .1284759  | 1.73  | 0.084 | -.0299406 .4736756 |
| i                                |          |           |       |      |                     |
| e                                | -.0522472 | .040321   | -1.30 | 0.195 | -.1312748 .0267805 |
| u                                | -.0028753 | .0041244  | -0.70 | 0.486 | -.0109589 .0052083 |
| g                                | 1.203672  | .0979354  | 12.29 | 0.000 | 1.011722 1.395622  |
| r                                |          |           |       |      |                     |
| e                                | -.1.16048 | .1361163  | -8.53 | 0.000 | -.1427263 -.8936968 |
| u                                | -.003171  | .0045309  | -0.70 | 0.484 | -.0120515 .0057094 |
| g                                | 1.32748   | .2224889  | 5.97  | 0.000 | .8914098 1.76355  |
Importantly, the financial shock causes $r$ to fall by more than $i$, indicating an increase in the interest rate spread. Inflation also falls, as does the output gap. It is the fall in the inflation rate that causes the central bank’s interest rate to fall on impact of the shock.

**Impulse responses**

We use the `irf set` command to set `finirf.irf` as the active IRF file and then use `irf create` to create and store the impulse responses under the name `param`.

```
.irf set finirf
(file finirf.irf created)
(file finirf.irf now active)
.irf create param
(file finirf.irf updated)
```

Finally, we graph the impulse response to a financial shock.

```
.irf graph irf, irf(param1) impulse(e) response(e x p i r) byopts(yrescale)
```

The top middle panel shows the effect of the shock to the $e$ equation on $e$ itself. The shock represents a persistent increase in the spread between the two interest rates. The increased interest rate spread causes both the output gap and inflation rate to fall. The safe interest rate $r$ falls dramatically, and the combined effect of the increased spread $e$ and reduced safe interest rate $r$ causes the market interest rate $i$ to be little changed.
Also see

[DSGE] Intro 1 — Introduction to DSGEs

[DSGE] Intro 3a — New Keynesian model

[DSGE] Intro 3b — New Classical model

[DSGE] dsge — Linear dynamic stochastic general equilibrium models

[DSGE] dsge postestimation — Postestimation tools for dsge
Intro 3d — Nonlinear New Keynesian model

Description

This introduction estimates and interprets the parameters of a nonlinear New Keynesian model. We demonstrate the effect of a change in constraints on a model that is only partially identified.

Remarks and examples

Remarks are presented under the following headings:

- The model
- Parameter estimation
- Policy and transition matrices
- Impulse responses
- A change in constraints

The model

Equations (1)–(5) specify a simplified nonlinear New Keynesian model of inflation $\Pi_t$, the output gap $X_t$, and the interest rate $R_t$. These three variables are driven by two state variables, $Z_t$ and $M_t$. Recall that in this manual, capital letters denote the level of a variable, while lowercase letters denote deviations from steady state.

\begin{align*}
1 &= \beta E_t \left( \frac{X_t}{X_{t+1}} \frac{1}{Z_t} \frac{R_t}{\Pi_{t+1}} \right) \\
(\theta - 1) + \phi(\pi_t - 1)\pi_t &= \theta X_t + \phi \beta E_t \{(\pi_{t+1} - 1)\pi_{t+1}\} \\
\beta R_t &= \Pi_t \psi M_t \\
\ln(M_{t+1}) &= \rho_m \ln(M_t) + u_{t+1} \\
\ln(Z_{t+1}) &= \rho_z \ln(Z_t) + e_{t+1}
\end{align*}

Equation (1) is the Euler equation linking the output gap in the current period to the expected future output gap and the real interest rate. In this equation, $Z_t$ can be interpreted as a state variable influencing household decisions. In applications, $Z_t$ stands in for the natural rate of interest, consumer spending shocks, or government expenditures. Equation (2) is the Phillips curve linking the current rate of inflation to expected future inflation and the output gap. Equation (3) is the interest-rate rule linking the interest rate to the inflation rate; in this equation, the state variable $M_t$ captures all movements in the interest rate that are due to factors other than inflation. Equation (4) specifies the stochastic process for the state variable in the interest rate equation. Equation (5) specifies the stochastic process for the state variable in the Euler equation.
Parameter estimation

We estimate the model’s parameters using data on the inflation rate and the nominal interest rate.

```
. use https://www.stata-press.com/data/r16/usmacro2
( Federal Reserve Economic Data - St. Louis Fed, 2017-01-15)
```

Not all model parameters are identified. We constrain $\beta$ to be 0.96, a common value in the literature. The parameters $\theta$ and $\phi$ are collinear and cannot be estimated together. The parameter $\theta$ is set to 5, another common value in the literature. Later, we will explore consequences of the choice $\theta = 5$. The remaining six parameters are identified.

```
. constraint 1 _b[theta]=5
. constraint 2 _b[beta]=0.96
```

To specify this model to `dsgenl`, we will write each equation and then specify four options. The first three options tell `dsgenl` about the model’s structure. The `observed()` option specifies observed control variables that are to be used in estimation. The `unobserved()` option specifies latent control variables. The `exostate()` option specifies the collection of state variables. The other option we use is the standard `constraints()` option that applies our constraints on the parameters $\beta$ and $\theta$.

```
. dsgenl (1 = {beta}*(x/F.x)*(1/z)*(r/F.p))
> (({theta}-1 + {phi}*(p -1)*p = {theta}*x + {phi}*{beta}*(F.p-1)*F.p)
> (({beta})*r = (p)^({psi=2})*m)
> (ln(F.m) = {rhom}*ln(m))
> (ln(F.z) = {rhoz}*ln(z)),
> exostate(z m) unobserved(x) observed(p r)
> constraint(1 2) nolog
```

Solving at initial parameter vector ...
Checking identification ...
First-order DSGE model

```
Sample: 1955q1 - 2015q4
Number of obs = 244
Log likelihood = -753.57131
( 1) [/structural]theta = 5
( 2) [/structural]beta = .96
```

<table>
<thead>
<tr>
<th></th>
<th>OIM</th>
<th></th>
<th></th>
<th></th>
<th>[95% Conf. Interval]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Coef.</td>
<td>Std. Err.</td>
<td>z</td>
<td>P&gt;</td>
<td>z</td>
</tr>
<tr>
<td>/structural</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>beta</td>
<td>.96</td>
<td>(constrained)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>theta</td>
<td>5</td>
<td>(constrained)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>phi</td>
<td>47.07949</td>
<td>15.94165</td>
<td>2.95</td>
<td>0.003</td>
<td>15.83443</td>
</tr>
<tr>
<td>psi</td>
<td>1.943004</td>
<td>.2957882</td>
<td>6.57</td>
<td>0.000</td>
<td>1.363269</td>
</tr>
<tr>
<td>rhom</td>
<td>.700549</td>
<td>.0452605</td>
<td>15.48</td>
<td>0.000</td>
<td>.6118401</td>
</tr>
<tr>
<td>rhoz</td>
<td>.9545257</td>
<td>.0186424</td>
<td>51.20</td>
<td>0.000</td>
<td>.9179873</td>
</tr>
<tr>
<td>sd(e.z)</td>
<td>.5689892</td>
<td>.0982979</td>
<td>.3763289</td>
<td>.7616495</td>
<td></td>
</tr>
<tr>
<td>sd(e.m)</td>
<td>2.318204</td>
<td>.3047446</td>
<td>1.720915</td>
<td>2.915492</td>
<td></td>
</tr>
</tbody>
</table>
```

Because this is the nonlinear version of the model in [DSGE] Intro 3a, it is instructive to compare the two results. The two models deliver nearly identical estimates of most parameters; the small differences seen in the fourth and higher decimal places are due to differences in the numerical techniques used in each command. The only parameter that differs for the two models is the stickiness parameter $\phi$, which is the counterpart of the Phillips curve slope $\kappa$ from [DSGE] Intro 3a. The parameterization of the Phillips curve in (2) above differs from the one in [DSGE] Intro 3a. However, the two are linked by the relationship $\kappa = (\theta - 1)/\phi$, and we can use that relationship to show that the models are equivalent.
\[ \text{nlcom} \left( \frac{\_b[\theta] - 1}{\_b[\phi]} \right) \]

\[ \_nl\_1: \left( \frac{\_b[\theta] - 1}{\_b[\phi]} \right) \]

|       | Coef. | Std. Err. | z     | P>|z|  | [95% Conf. Interval] |
|-------|-------|-----------|-------|-------|----------------------|
| \_nl\_1 | 0.0849627 | 0.0287693 | 2.95  | 0.003 | 0.0285758 - 0.1413495 |

The value above is the same as the coefficient for \( \kappa \) in [DSGE] Intro 3a.

**Policy and transition matrices**

The policy matrix is part of the state-space representation of a DSGE model. It expresses the control variables of the model as functions of the state variables alone. Elements of the policy matrix represent the response of a control variable to a one-unit increase in a state variable. In a linear DSGE model, the policy matrix will also be linear. For a nonlinear DSGE model, the policy function will typically be nonlinear. The policy matrix reported after \texttt{dsgenl} is a linear approximation of the true, nonlinear policy matrix evaluated at the estimated parameter vector. For details on the linearization of the policy function, see DeJong and Dave (2011, sec. 5.3).

\[ \text{. estat policy} \]

**Policy matrix**

<table>
<thead>
<tr>
<th></th>
<th>Delta-method</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Coef.</td>
<td>Std. Err.</td>
<td>z</td>
<td>P&gt;</td>
<td>z</td>
</tr>
<tr>
<td>( x )</td>
<td>( z )</td>
<td>.9529253</td>
<td>.4813989</td>
<td>1.98</td>
<td>0.048</td>
</tr>
<tr>
<td></td>
<td>( m )</td>
<td>-1.608221</td>
<td>.4050582</td>
<td>-3.97</td>
<td>0.000</td>
</tr>
<tr>
<td>( p )</td>
<td>( z )</td>
<td>.9678176</td>
<td>.2777463</td>
<td>3.48</td>
<td>0.000</td>
</tr>
<tr>
<td></td>
<td>( m )</td>
<td>-.417252</td>
<td>.039361</td>
<td>-10.60</td>
<td>0.000</td>
</tr>
<tr>
<td>( r )</td>
<td>( z )</td>
<td>1.880473</td>
<td>.2616009</td>
<td>7.19</td>
<td>0.000</td>
</tr>
<tr>
<td></td>
<td>( m )</td>
<td>.1892777</td>
<td>.0591663</td>
<td>3.20</td>
<td>0.001</td>
</tr>
</tbody>
</table>

An increase in \( m \) decreases inflation, decreases the output gap, and increases interest rates. We may interpret it as an unexpected contractionary monetary policy. An increase in \( z \) increases inflation, increases the output gap, and increases interest rates.

Next, we can look at the state transition matrix. Because the states are uncorrelated with each other in this example, the elements of the state transition matrix are the persistence parameters in the model.
. estat transition

Transition matrix of state variables

<table>
<thead>
<tr>
<th></th>
<th>Delta-method</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Coef.</td>
<td>Std. Err.</td>
<td>z</td>
<td>P&gt;</td>
<td>z</td>
</tr>
<tr>
<td>F.z</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>z</td>
<td>.9545257</td>
<td>.0186424</td>
<td>51.20</td>
<td>0.000</td>
<td>.9179873</td>
</tr>
<tr>
<td>m</td>
<td>0 (omitted)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>F.m</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>z</td>
<td>2.78e-16</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>m</td>
<td>.700549</td>
<td>.0452605</td>
<td>15.48</td>
<td>0.000</td>
<td>.6118401</td>
</tr>
</tbody>
</table>

Note: Standard errors reported as missing for constrained transition matrix values.

These parameters repeat the results we saw in the estimation table. When the state variables are correlated, the transition matrix will reveal new information about the correlations among state variables.

Impulse responses

The impulse–response function traces out the effect of a shock on the model’s state and control variables. We use two commands to set up the impulse responses: an `irf set` command to set the active IRF file and an `irf create` command to create the impulse–response functions.

From there, we can use `irf graph irf` to graph the impulse–response functions.

. irf set dta/nlex1irf, replace
   (file dta/nlex1irf.irf created)
   (file dta/nlex1irf.irf now active)
. irf create irf1, replace
   irfname irf1 not found in dta/nlex1irf.irf
   (file dta/nlex1irf.irf updated)
. irf graph irf, impulse(m z) response(x p r) byopts(yrescale)
Responses to a monetary shock are displayed in the first row. A shock to monetary policy leads inflation to fall, the interest rate to rise, and the output gap to fall. From the bottom row, we see that an increase in the natural rate of interest increases all three control variables.

A change in constraints

In the above example, the parameter $\theta$ was set to 5 to achieve identification of the other parameters. The following example shows the consequences of this decision by examining a case where $\theta$ was fixed to a different value. Suppose that instead of setting $\theta = 5$ above, we set $\theta = 2$. How would the change in constraints change our estimates? First, we store the current estimates with `estimates store`. We call the current estimates `theta5` because they represent the case where $\theta = 5$.

```
dsgenl
First-order DSGE model
Sample: 1955q1 - 2015q4 Number of obs = 244
Log likelihood = -753.57131
( 1) [/structural]theta = 5
( 2) [/structural]beta = .96


          OIM
Coef. Std. Err.  z  P>|z|  [95% Conf. Interval]
/structural
   beta   .96 (constrained)
   theta  5 (constrained)
    phi  47.07949  15.94165  2.95  0.003    15.83443    78.32455
    psi  1.943004  .2957882  6.57  0.000    1.363269    2.522738
   rhom  .700549  .0452605 15.48  0.000    .6118401    .7892579
   rhoz  .9545257  .0452605 15.48  0.000    .8199873    .9910641
   sd(e.z) .5689892  .0982979   .3763289    .7616495
   sd(e.m) 2.318204  .3047446   1.720915   2.915492
```

```
estimates store theta5
```

We can now recall these estimates later as needed. We set up the new constraint with

```
constraint 3 _b[theta]=2
```

and reestimate parameters. The `nolog` option is used to suppress the iteration log.
dsgenl (1 = {beta}*(x/F.x)*(1/z)*(r/F.p))
> (theta)-1 + {phi}*(p-1)*p = {theta}*x + {phi}*{beta}*(F.p-1)*F.p
> (({beta})*r = (p)^({psi=2})*m)
> (ln(F.m) = {rhom}*ln(m))
> (ln(F.z) = {rhoz}*ln(z)),
> exostate(z m) unobserved(x) observed(p r)
> constraint(2 3) nolog
Solving at initial parameter vector ...
Checking identification ...
First-order DSGE model
Sample: 1955q1 - 2015q4  Number of obs = 244
Log likelihood = -753.57131
( 1) \[/structural\]beta = .96
( 2) \[/structural\]theta = 2

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Coef.</td>
<td>Std. Err.</td>
<td>z</td>
<td>P&gt;</td>
<td>z</td>
</tr>
<tr>
<td>/structural</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>beta</td>
<td>.96</td>
<td>(.constrained)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>theta</td>
<td>2</td>
<td>(.constrained)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>phi</td>
<td>11.76985</td>
<td>3.985343</td>
<td>2.95</td>
<td>0.003</td>
<td>3.958725</td>
</tr>
<tr>
<td>psi</td>
<td>1.943002</td>
<td>.2957863</td>
<td>6.57</td>
<td>0.000</td>
<td>1.363272</td>
</tr>
<tr>
<td>rhom</td>
<td>.7005489</td>
<td>.0452598</td>
<td>15.48</td>
<td>0.000</td>
<td>.6118414</td>
</tr>
<tr>
<td>rhoz</td>
<td>.9545256</td>
<td>.0186424</td>
<td>51.20</td>
<td>0.000</td>
<td>.9179872</td>
</tr>
<tr>
<td>sd(e.z)</td>
<td>.5689888</td>
<td>.0982973</td>
<td>.3763296</td>
<td>.761648</td>
<td></td>
</tr>
<tr>
<td>sd(e.m)</td>
<td>2.318202</td>
<td>.3047427</td>
<td>1.720918</td>
<td>2.915487</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Statistics</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ll</td>
<td>-753.5713</td>
<td>-753.5713</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

We saved these estimates in theta2 because they represent the case where \( \theta = 2 \). estimates table displays the two results. The stats(ll) option displays the estimated log likelihood with the estimated structural parameters.

. estimates store theta2
We saved these estimates in theta2 because they represent the case where \( \theta = 2 \). estimates table displays the two results. The stats(ll) option displays the estimated log likelihood with the estimated structural parameters.

. estimates table theta5 theta2, b(%9.4f) stats(ll)

<table>
<thead>
<tr>
<th>Variable</th>
<th>theta5</th>
<th>theta2</th>
</tr>
</thead>
<tbody>
<tr>
<td>/structural</td>
<td></td>
<td></td>
</tr>
<tr>
<td>beta</td>
<td>0.9600</td>
<td>0.9600</td>
</tr>
<tr>
<td>theta</td>
<td>5.0000</td>
<td>2.0000</td>
</tr>
<tr>
<td>phi</td>
<td>47.0795</td>
<td>11.7699</td>
</tr>
<tr>
<td>psi</td>
<td>1.9430</td>
<td>1.9430</td>
</tr>
<tr>
<td>rhom</td>
<td>0.7005</td>
<td>0.7005</td>
</tr>
<tr>
<td>rhoz</td>
<td>0.9545</td>
<td>0.9545</td>
</tr>
<tr>
<td>sd(e.z)</td>
<td>0.5690</td>
<td>0.5690</td>
</tr>
<tr>
<td>sd(e.m)</td>
<td>2.3182</td>
<td>2.3182</td>
</tr>
<tr>
<td>Statistics</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ll</td>
<td>-753.5713</td>
<td>-753.5713</td>
</tr>
</tbody>
</table>

The reported log-likelihood values are identical across the two parameterizations. The change in constraints slides us along the top of a ridge in the likelihood function. The estimated value of \( \phi \) differs when the constraint is changed, but all other parameter estimates are identical. Even though the point estimate of \( \phi \) has changed, the \( z \) statistic and \( p \)-value are unchanged.

As discussed above, the underlying parameter that is being estimated here is \( \kappa = (\theta - 1)/\phi \). The restrictions on the parameter \( \theta \) lead to a change in the estimated value of \( \phi \) but do not change the
estimated value of \( \kappa \). We can use `nlcom` to recover \( \kappa \) first from the model with \( \theta = 5 \) and then from the model with \( \theta = 2 \).

```
. estimates restore theta5
(results theta5 are active now)
. nlcom (_b[theta]-1) / _b[phi]
   _nl_1: (_b[theta]-1) / _b[phi]
   Coef. Std. Err. z  P>|z|  [95% Conf. Interval]
   _nl_1      0.0849627   .0287693    2.95  0.003   .0285758   .1413495
```

```
. estimates restore theta2
(results theta2 are active now)
. nlcom (_b[theta]-1) / _b[phi]
   _nl_1: (_b[theta]-1) / _b[phi]
   Coef. Std. Err. z  P>|z|  [95% Conf. Interval]
   _nl_1      0.0849628   .0287689    2.95  0.003   .0285768   .1413489
```

The equivalence of the two parameterizations can also be seen through the policy matrix. The estimated policy matrix when \( \theta = 2 \) is

```
. estat policy
Policy matrix
Delta-method
Coef. Std. Err. z  P>|z|  [95% Conf. Interval]
   x
   z      .9529261   .4813946    1.98  0.048   .0094101   1.896442
   m    -1.60822   .4050525   -3.97  0.000  -2.402109  -.8143319
   p
   z      .9678188   .2777451    3.48  0.000   .4234484   1.512189
   m    -.4172524   .0393609  -10.60  0.000  -.4943984  -.3401065
   r
   z      1.880474   .2615997    7.19  0.000   1.367748   2.3932
   m     .1892776   .0591660    3.20  0.001   .0733143   .3052408
```

This policy matrix is identical to the one shown above, in which the constraint \( \theta = 5 \) was applied. The two parameter vectors imply the same reduced-form matrices.

Reference

Also see

[DSGE] Intro 1 — Introduction to DSGEs
[DSGE] Intro 3a — New Keynesian model
[DSGE] Intro 3e — Nonlinear New Classical model
[DSGE] Intro 3f — Stochastic growth model
[DSGE] dsgenl — Nonlinear dynamic stochastic general equilibrium models
[DSGE] dsgenl postestimation — Postestimation tools for dsgenl
Description

This introduction describes a nonlinear New Classical model, estimates some of its parameters, and explores the model’s impulse–response functions under two different parameter settings.

Remarks and examples

Remarks are presented under the following headings:
- The model
- Parameter estimation
- Steady state
- Model-implied covariances
- Policy and transition matrices
- Impulse responses
- Sensitivity analysis

The model

Equations (1)–(8) specify a variant on the New Classical model studied in King and Rebelo (1999). It includes equations for output $Y_t$, consumption $C_t$, investment $I_t$, hours worked $H_t$, the interest rate $R_t$, the real wage $W_t$, the capital stock $K_t$, and productivity $Z_t$. Model variables must be weakly stationary. The model contains six parameters: $\alpha$, $\beta$, $\chi$, $\delta$, $\rho$, and $\sigma$.

Equation (1) is a consumption Euler equation that links consumption in the current period to expected future consumption and the expected future interest rate. Equation (2) is a labor supply equation that links hours worked to the wage and consumption. Equation (3) is a standard national income accounting identity, stating that output is split between consumption and investment. Equation (4) is an output supply equation or production function, stating that output is produced by combining capital $K_t$ and labor $H_t$ at productivity level $Z_t$. Equation (5) is a capital demand curve. Equation (6) is a labor demand curve. Equation (7) is the capital accumulation process. Equation (8) specifies a stochastic process for productivity. The stochastic shock $e_{t+1}$ is i.i.d. normal with mean zero and standard deviation $\sigma_z$.
The model includes six parameters. \( \beta \) is the discount factor reflecting a preference for current consumption relative to future consumption. \( \alpha \) is a production parameter. \( \chi \) is a preference parameter. \( \delta \) is the depreciation rate. \( \rho \) measures the persistence of the stochastic productivity process. Finally, \( \sigma \) is the standard deviation of the innovations to the productivity process.

**Parameter estimation**

In this example, we fix some parameters at prespecified values and estimate others. Once the parameters are estimated, we analyze the model by inspecting its steady state, low-order moments, policy and transition matrices, and impulse–response functions.

We use the growth rate of output as the empirical counterpart to \( y_t \). For the calibrated parameters, we set \( \alpha = 0.33, \beta = 0.99, \delta = 0.025, \) and \( \chi = 2 \), which follows King and Rebelo (1999). We will estimate \((\rho, \sigma_z)\). We first bring in the data.

```
use https://www.stata-press.com/data/r16/usmacro2
(Federal Reserve Economic Data - St. Louis Fed, 2017-01-15)
```

We next set up all the constraints

```stata
constraint 1 _b[alpha] = 0.33
constraint 2 _b[beta] = 0.99
constraint 3 _b[delta] = 0.025
constraint 4 _b[chi] = 2
```

and estimate the remaining parameters,

```stata
dsgenl (1/c = {beta}*(1/F.c)*(1+r-{delta}))
> ({chi}*h = w/c)
> (y = c + i)
> (y = z*k-\{alpha\}*h^((1-\{alpha\}))
> (r = \{alpha\}*y/k)
> (w = (1-\{alpha\})*y/h)
> (F.k = i + (1-{delta})*k)
> (ln(F.z) = \{rho\}*ln(z))
, observed(y) unobserved(c i r w h) exostate(z) endostate(k)
> constraint(1/4)
```

Solving at initial parameter vector ...
Checking identification ...

(setting technique to bfgs)
Iteration 0:  log likelihood = -938.12868
Iteration 1:  log likelihood = -853.10706  (backed up)
Iteration 2:  log likelihood = -834.13876  (backed up)
Iteration 3:  log likelihood = -834.13876  (backed up)
Iteration 4:  log likelihood = -828.70123
Iteration 5:  log likelihood = -828.70123  (backed up)
Iteration 6:  log likelihood = -828.70123  (backed up)
Iteration 7:  log likelihood = -828.70123  (backed up)
Iteration 8:  log likelihood = -828.70123  (backed up)
Iteration 9:  log likelihood = -828.70123  (backed up)
(switching technique to nr)
Iteration 10: log likelihood = -828.70123  (not concave)
Iteration 11: log likelihood = -691.57217
Iteration 12: log likelihood = -686.86764  (not concave)
Iteration 13: log likelihood = -669.49914
Iteration 14: log likelihood = -647.58806
Iteration 15: log likelihood = -639.41443
Iteration 16: log likelihood = -639.38788
Iteration 17: log likelihood = -639.38787
First-order DSGE model

Sample: 1955q1 - 2015q4  Number of obs = 244
Log likelihood = -639.38787
( 1) \[/structural\]alpha = .33
( 2) \[/structural\]beta = .99
( 3) \[/structural\]delta = .025
( 4) \[/structural\]chi = 2

<table>
<thead>
<tr>
<th></th>
<th>OIM</th>
<th></th>
<th></th>
<th></th>
<th>[95% Conf. Interval]</th>
</tr>
</thead>
<tbody>
<tr>
<td>y</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>/structural</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>beta</td>
<td>.99 (constrained)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>delta</td>
<td>.025 (constrained)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>chi</td>
<td>2 (constrained)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>alpha</td>
<td>.33 (constrained)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>rho</td>
<td>.3132837 .0614709 5.10 0.000 .1928029 .4337645</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>sd(e.z)</td>
<td>2.249022 .101853 2.049394 2.44865</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

We specified four options related to the model structure. The exostate(z) option lists the state variable that is subject to shocks, z. The endostate(k) option specifies that k is a state variable without shocks. The observed(y) option specifies y as an observed control variable. The unobserved(c i r w h) option specifies those variables as unobserved, or latent, control variables. The remaining option we specified was constraint(1/4), which applies the constraints we set up previously. We find that the estimated productivity process is mildly persistent, \( \rho = 0.31 \). The standard deviation of the productivity shock is about 2.2, or about 2%.

**Steady state**

In the absence of shocks, the variables in a nonlinear DSGE model converge to a steady-state position in which variables are constant through time. dsge1 finds the steady state of the model at the estimated parameter values. You can view the steady state with estat steady.

```
. estat steady
Location of model steady-state
```

<table>
<thead>
<tr>
<th></th>
<th>Delta-method</th>
<th></th>
<th></th>
<th></th>
<th>[95% Conf. Interval]</th>
</tr>
</thead>
<tbody>
<tr>
<td>k</td>
<td>18.75991</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>z</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>c</td>
<td>1.526432</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>i</td>
<td>.4689977</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>r</td>
<td>.035101</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>w</td>
<td>2.02027</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>h</td>
<td>.6617621</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>y</td>
<td>1.99543</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Note: Standard errors reported as missing for constrained steady-state values.

All the standard errors for the estimated location of the steady state are missing. This is because of the constraints we placed on some of the model parameters. We estimated the persistence of the productivity state variable and the standard deviation of the shock to productivity. These dynamic parameters have no effect on the steady state. Instead, the steady state is a function of the model’s static parameters. Because we constrained all the static parameters, the steady-state location is not
subject to uncertainty, and the standard errors are all missing. Put another way, the location of the steady state is determined entirely by parameters that we have constrained in the model and by the model’s structure itself.

### Model-implied covariances

The model’s state-space matrices generate predictions for the variances and covariances of the model variables.

```stata
. estat covariance
Estimated covariances of model variables

<table>
<thead>
<tr>
<th></th>
<th>Delta-method</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Coef.</td>
<td>Std. Err.</td>
<td>z</td>
<td>P&gt;</td>
<td>z</td>
<td></td>
</tr>
<tr>
<td>c</td>
<td>var(c)</td>
<td>.8564586</td>
<td>.1674129</td>
<td>5.12</td>
<td>0.000</td>
<td>.5283354</td>
</tr>
<tr>
<td></td>
<td>cov(c,i)</td>
<td>1.699536</td>
<td>.432758</td>
<td>3.93</td>
<td>0.000</td>
<td>.8513461</td>
</tr>
<tr>
<td></td>
<td>cov(c,r)</td>
<td>-.4153308</td>
<td>.055347</td>
<td>-7.50</td>
<td>0.000</td>
<td>-.523809</td>
</tr>
<tr>
<td></td>
<td>cov(c,w)</td>
<td>.9555354</td>
<td>.198332</td>
<td>4.82</td>
<td>0.000</td>
<td>.5668118</td>
</tr>
<tr>
<td></td>
<td>cov(c,y)</td>
<td>1.054612</td>
<td>.2293833</td>
<td>4.60</td>
<td>0.000</td>
<td>.6050292</td>
</tr>
<tr>
<td>i</td>
<td>var(i)</td>
<td>207.93</td>
<td>.207228</td>
<td>10.36</td>
<td>0.000</td>
<td>168.589</td>
</tr>
<tr>
<td></td>
<td>cov(i,r)</td>
<td>50.23759</td>
<td>.422667</td>
<td>10.42</td>
<td>0.000</td>
<td>40.78534</td>
</tr>
<tr>
<td></td>
<td>cov(i,w)</td>
<td>25.93532</td>
<td>.625692</td>
<td>9.88</td>
<td>0.000</td>
<td>20.78906</td>
</tr>
<tr>
<td></td>
<td>cov(i,y)</td>
<td>24.23579</td>
<td>.325743</td>
<td>10.42</td>
<td>0.000</td>
<td>19.67741</td>
</tr>
<tr>
<td></td>
<td>cov(i,y)</td>
<td>50.17111</td>
<td>4.941598</td>
<td>10.15</td>
<td>0.000</td>
<td>40.48575</td>
</tr>
<tr>
<td>r</td>
<td>var(r)</td>
<td>12.94748</td>
<td>1.274919</td>
<td>10.16</td>
<td>0.000</td>
<td>10.44869</td>
</tr>
<tr>
<td></td>
<td>cov(r,w)</td>
<td>5.537299</td>
<td>.524047</td>
<td>10.57</td>
<td>0.000</td>
<td>4.510185</td>
</tr>
<tr>
<td></td>
<td>cov(r,h)</td>
<td>5.95263</td>
<td>.5725108</td>
<td>10.40</td>
<td>0.000</td>
<td>4.830529</td>
</tr>
<tr>
<td></td>
<td>cov(r,y)</td>
<td>11.48993</td>
<td>1.096232</td>
<td>10.48</td>
<td>0.000</td>
<td>9.341353</td>
</tr>
<tr>
<td>w</td>
<td>var(w)</td>
<td>3.89111</td>
<td>.4598352</td>
<td>8.46</td>
<td>0.000</td>
<td>2.989849</td>
</tr>
<tr>
<td></td>
<td>cov(w,h)</td>
<td>2.935574</td>
<td>.2904425</td>
<td>10.11</td>
<td>0.000</td>
<td>2.366317</td>
</tr>
<tr>
<td></td>
<td>cov(w,y)</td>
<td>6.826684</td>
<td>.743152</td>
<td>9.19</td>
<td>0.000</td>
<td>5.370133</td>
</tr>
<tr>
<td>h</td>
<td>var(h)</td>
<td>2.836497</td>
<td>.2711682</td>
<td>10.46</td>
<td>0.000</td>
<td>2.305017</td>
</tr>
<tr>
<td></td>
<td>cov(h,y)</td>
<td>5.772071</td>
<td>.561063</td>
<td>10.29</td>
<td>0.000</td>
<td>4.672408</td>
</tr>
<tr>
<td>y</td>
<td>var(y)</td>
<td>12.59876</td>
<td>1.296733</td>
<td>9.72</td>
<td>0.000</td>
<td>10.05721</td>
</tr>
</tbody>
</table>
```

One object of interest is the relative volatility of a model variable, defined as the standard deviation of that variable compared with the standard deviation of a reference variable. We can calculate the volatility of investment (model variable \( i \)) relative to output (model variable \( y \)) with \texttt{nlcom}.

Before we can use \texttt{nlcom}, we need to add the \texttt{post} option to our \texttt{estat covariance} command. With this option, the results are stored in \texttt{e()}, where \texttt{nlcom} can access them. First, however, we save the \texttt{dsgenl} estimates using \texttt{estimates store}.

```
. estimates store dsgenl
. quietly estat covariance, post
```
We can now refer to the variances of investment and output as \( \text{var}(i) \) and \( \text{var}(y) \). We use \text{nlcom} to compute the volatility of investment relative to output.

\[
\text{nlcom sqrt(_b[i:var(i)] / _b[y:var(y)])}
\]

|              | Coef.  | Std. Err. |   z   |   P>|z|   |  [95% Conf. Interval] |
|--------------|--------|-----------|-------|--------|----------------------|
| \_nl_1       | 4.062513 | 0.0314516 | 129.17 | 0.000  | 4.000869 - 4.124157  |

The model predicts that investment is about four times as volatile as output.

### Policy and transition matrices

The model’s state-space form expresses the control variables as functions of the state variables alone. It solves the system of equations locally near the steady state, resolving the expectations of future variables via the rational expectations assumption. The entries in the state-space matrices are interpretable as the contemporaneous effect of a one-unit change in the state variable on the control variable.

\[
\text{estimates restore dsgenl}
\]

(\text{results dsgenl are active now})

\[
\text{. estat policy}
\]

Policy matrix

<table>
<thead>
<tr>
<th></th>
<th>Delta-method</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Coef.</td>
</tr>
<tr>
<td>c</td>
<td></td>
</tr>
<tr>
<td>k</td>
<td>.5651362</td>
</tr>
<tr>
<td>z</td>
<td>6.12184</td>
</tr>
<tr>
<td>i</td>
<td>-.9392642</td>
</tr>
<tr>
<td>r</td>
<td>-.7884521</td>
</tr>
<tr>
<td>w</td>
<td>.3883421</td>
</tr>
<tr>
<td>h</td>
<td>-.1767941</td>
</tr>
<tr>
<td>y</td>
<td>.2115479</td>
</tr>
</tbody>
</table>

Note: Standard errors reported as missing for constrained policy matrix values.

The standard errors presented here again require some interpretation. The standard errors for the impact effect of \( k \) on all model variables are missing. These impact effects depend entirely on the parameters we constrained, so their values are fixed entirely by the structure of the model.
The state variables are correlated in this model. The capital equation shown in (7) specifies that future capital is a function of the current capital stock and current investment. In turn, investment depends on the current capital stock and on productivity; hence, the future capital stock depends on both the current capital stock and current productivity. We can see these relationships with `estat transition`.

```
estat transition
```

**Transition matrix of state variables**

<table>
<thead>
<tr>
<th></th>
<th>Delta-method</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Coef.</td>
<td>Std. Err.</td>
<td>z</td>
<td>P&gt;</td>
<td>z</td>
</tr>
<tr>
<td>F.k</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>k</td>
<td>.9515184</td>
<td></td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>z</td>
<td>.153046</td>
<td>.0005693</td>
<td>268.84</td>
<td>0.000</td>
<td>.1519302</td>
</tr>
<tr>
<td>F.z</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>k</td>
<td>0 (omitted)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>z</td>
<td>.3132837</td>
<td>.0614709</td>
<td>5.10</td>
<td>0.000</td>
<td>.1928029</td>
</tr>
</tbody>
</table>

*Note: Standard errors reported as missing for constrained transition matrix values.*

The “own” effects (k on F.k and z on F.z) are persistence parameters. The other reported effects show how a change in one state variable contemporaneously affects another. A change in the capital stock does not affect future productivity, but a change in productivity does affect the future capital stock. A 1% increase in productivity raises the future capital stock by 0.15%.
Impulse responses

The policy and transition matrices display impact effects. Because of the dynamic structure of the state, a shock to a state variable has lasting effects on both the state and the controls. An impulse–response function traces out the full dynamic effect of a shock. When state variables are correlated, a shock to one state variable will (either immediately or eventually) lead to movements in another state variable.

We use the `irf` commands to trace out the effect of a shock to productivity on itself and on each of the control variables in our model.

```
. irf set rbcirf, replace
  (file rbcirf.irf created)
  (file rbcirf.irf now active)
. irf create est, step(20) replace
  irfname est not found in rbcirf.irf
  (file rbcirf.irf updated)
. irf graph irf, impulse(z) response(y c i h w z) byopts(yrescale)
```

All model variables rise on a shock to productivity. For most model variables, the increase is short lived, and they return to their long-run values within five periods. Consumption is the exception (upper left panel). It rises smoothly for several periods, then gradually falls back to its long-run value.
Sensitivity analysis

In this section, we will explore the effect of changing some model parameters on the impulse responses. We first type the `dsgenl` command without arguments to replay the current estimates.

```
.dsgenl
First-order DSGE model
Sample: 1955q1 - 2015q4   Number of obs   =   244
Log likelihood = -639.38787

( 1) [//structural]alpha = .33
( 2) [//structural]beta = .99
( 3) [//structural]delta = .025
( 4) [//structural]chi = 2

y      Coef.  Std. Err.      z    P>|z|     [95% Conf. Interval]
/structural
    beta   .990   (.061)    16.37   0.000      .868    1.112
    delta  .025   (.007)     3.63   0.000      .011    .038
    alpha  .330   (.063)     5.25   0.000      .205    .455
    rho    .313    .061     5.10   0.000      .193    .433
sd(e.z)  2.25    .102     21.97   0.000     2.05     2.45
```

We can solve the model using different parameter values and analyze how the model behaves across the two scenarios. It is common to examine a model’s behavior for multiple parameter settings, for example, for different settings for the persistence of a shock or for different settings of preference, technology, or policy parameters. The simplest way to solve using a new set of parameters is to use the `from()` and `solve` options. First, store the current parameter vector in a Stata matrix.

```
.matrix b = e(b)
```

Next, change the entries in the matrix to the new desired parameters. In this case, we want to look at the response of the model to a shock when productivity is more persistent than was estimated, so we set $\rho = 0.6$.

```
.matrix b[1,5] = 0.6
```
Next, we solve the model again. In the following command, note the use of the \texttt{from(b)} and \texttt{solve} options.

\begin{verbatim}
. dsgenl (1/c = \{beta\}*(1/F.c)*(1+r-{delta}))
> ({chi}/(1-h) = w/c)
> (y = c + i)
> (y = z*k^{-\{alpha\}}*h^{(1-{alpha})})
> (w = (1-{alpha})*y/h)
> (r = \{alpha\}*y/k)
> (F.k = i + (1-{delta})*k)
> (ln(F.z) = \{rho\}*ln(z))
> , observed(y) unobserved(c i r w h) exostate(z) endostate(k)
> from(b) solve noidencheck
\end{verbatim}

Solving at initial parameter vector ...

First-order DSGE model

Sample: 1955q1 - 2015q4 Number of obs = 244
Log likelihood = -654.31148

|     | Coef. | Std. Err. | z  | P>|z|   | [95% Conf. Interval] |
|-----|-------|-----------|----|-------|----------------------|
| /structural |       |           |    |       |                      |
| beta |   .99  |           |    | .     |                      |
| delta|   .025 |           |    | .     |                      |
| chi  |     2  |           |    | .     |                      |
| alpha|   .33  |           |    | .     |                      |
| rho  |     .6  |           |    | .     |                      |
| sd(e.z) |  2.249022 |           |    | .     |                      |

Note: Skipped identification check.
Note: Model solved at specified parameters; maximization options ignored.

We passed the parameter vector in using \texttt{from()}, and we used the \texttt{solve} option to prevent estimation. Because we specified the \texttt{solve} option, the estimation table simply repeats the values we fed in with \texttt{from()}. Of course, because we did not estimate, we do not get standard errors. When we use \texttt{solve}, the point of the analysis is always in the \texttt{estat} and \texttt{irf} commands that come after.

We create a new set of impulse responses.

\begin{verbatim}
. irf create alt, step(20) replace
irfname alt not found in rbcirf.irf
(file rbcirf.irf updated)
\end{verbatim}
We graph the response of $y$ to a $z$ impulse across the two models.

\[
\text{. irf ograph (est z y irf) (alt z y irf)}
\]

![Graph of response of $y$ to a $z$ impulse across the two models.](image)

We see that output increases more and takes longer to return to its long-run value when we set $\rho$ to 0.60 instead of its estimated value of 0.31.

We are not interested only in the response of $y$ to a shock in $z$. We can also plot the response of $c$, $h$, $i$, $w$, $y$, and $z$ to an impulse to $z$ in panels of a combined graph. We will accomplish this task with a `foreach` loop.
The model with $\rho = 0.60$ shows more persistence overall than the estimated value of $\rho = 0.31$, as expected. For variables such as consumption and wages, the amplitude of the response is amplified as well.

Reference


Also see

[DSGE] Intro 1 — Introduction to DSGEs
[DSGE] Intro 3b — New Classical model
[DSGE] Intro 3d — Nonlinear New Keynesian model
[DSGE] Intro 3f — Stochastic growth model
[DSGE] dsgenl — Nonlinear dynamic stochastic general equilibrium models
[DSGE] dsgenl postestimation — Postestimation tools for dsgenl
In this introduction, we solve a stochastic growth model and interpret objects that are available after solving. The topics discussed in this introduction are more technical than those in the introductions describing other classic models. We provide some details about how \texttt{dsgen1} finds the model’s steady state, we describe how it solves for a first-order approximation to the model’s state-space form, and we provide guidance on the differences between and interpretations of linear versus log-linear approximations.

Remarks and examples

Remarks are presented under the following headings:

- The model
- Approximating the solution to a nonlinear DSGE model
- Specifying the model to Stata
- After solving
  - The steady state
  - Approximations to the policy and transition matrices
  - Linear and log-linear approximations

The model

The model contains equations that jointly determine output $Y_t$, the interest rate $R_t$, consumption $C_t$, capital $K_t$, and productivity $Z_t$. The model contains four parameters: $\alpha$, $\beta$, $\delta$, and $\rho$. This model is a variant on the model used in Schmitt-Grohé and Uribe (2004).

\begin{align*}
1 &= \beta E_t \left\{ \left( \frac{C_{t+1}}{C_t} \right)^{-1} (1 + R_{t+1} - \delta) \right\} \quad (1) \\
Y_t &= Z_t K_t^{\alpha} \quad (2) \\
R_t &= \alpha Z_t K_t^{\alpha-1} \quad (3) \\
K_{t+1} &= Y_t - C_t + (1 - \delta)K_t \quad (4) \\
\ln(Z_{t+1}) &= \rho \ln(Z_t) + \epsilon_{t+1} \quad (5)
\end{align*}

Equation (1) defines a relationship between consumption growth $C_{t+1}/C_t$ and the real interest rate $R_{t+1}$. Equation (2) is a production function for output $Y_t$ as a function of productivity $Z_t$ and capital $K_t$. Equation (3) is a model for the interest rate. Equation (4) is the equation for capital accumulation; capital in the next period is equal to undepreciated capital this period $(1 - \delta)K_t$, plus unconsumed output $Y_t - C_t$. Equation (5) is a law of motion for productivity $Z_t$. The parameter $\beta$ is a discount factor in the consumption equation, the parameter $\alpha$ is a production parameter in the output equation, the parameter $\delta$ is a depreciation parameter in the capital equation, and the parameter $\rho$ is a persistence parameter in the productivity equation.
The state variables are the current-period capital stock and the level of productivity, \((K_t, Z_t)\). The control variables are consumption, the interest rate, and output \((C_t, R_t, Y_t)\). To solve the model means to write the control variables as functions of the state variables and to write the future values of the state variables as functions of the current state variables. Equation (2) is already solved, for example, because it writes output as a function of the state variables alone. But (4) is not a reduced form, because it writes the future capital stock as a function of the state variable \(K_t\) and the control variables \(C_t\) and \(Y_t\).

The model equations have two key features. First, the model is nonlinear in its variables. Second, the model contains expectations of variables at time \(t+1\), conditional on information at time \(t\). \texttt{dsgenl} solves the model by taking a linear (or log-linear) approximation to the model equations at the steady state. Two concepts need to be clarified: the steady state and the approximation. We will describe these two issues in general, then return to the specific model described above.

**Approximating the solution to a nonlinear DSGE model**

We can gather up all the model equations into a vector of functions, \(f\). We denote the model’s vector of state variables by \(x_t\) and the vector of control variables by \(y_t\). Now, the DSGE model can be written as

\[
E_t \{ f(x_{t+1}, y_{t+1}, x_t, y_t, \theta) \} = 0
\]

The expectation operator applies to the entire system of equations. In a generic DSGE model, there are \(n\) equations, \(n_x\) state variables, and \(n_y\) control variables; it must be the case that \(n_x + n_y = n\).

The steady state of the model is the vector of values to which model variables converge in the absence of shocks. This is equivalent to the solution to the model’s system of equations after setting all future values of variables equal to current values and dropping the expectation operator. This location is a vector of numbers \((\bar{x}, \bar{y})\) that solves

\[
f(\bar{x}, \bar{y}, \bar{x}, \bar{y}, \theta) = 0
\]

This is a system of \(n\) equations in \(n\) unknowns, and its solution is the steady state. It is this point around which we take our approximation. The location of the steady state is influenced by the model parameters \(\theta\). It is possible that the parameter vector you specify does not admit a steady state; in such a case, \texttt{dsgenl} will halt and issue an error message. You may specify starting values for the search for the steady state with the \texttt{steadyinit()} option.

We take an approximation to the model equations at the steady state. Differentiate \(f\) with respect to each variable at the point \((\bar{x}, \bar{y}, \bar{x}, \bar{y})\). Differentiation generates a linearized model that is described by four matrices, namely,

\[
F_1 E_t \hat{x}_{t+1} + F_2 E_t \hat{y}_{t+1} + F_3 \hat{x}_t + F_4 \hat{y}_t = 0
\]

The matrix \(F_1\) contains partial derivatives of each model equation with respect to one-period-ahead state variables. The other \(F_i\) matrices are defined similarly. Each element of each \(F_i\) matrix depends on the structural parameter vector \(\theta\). There are two ways of obtaining the \(F_i\) matrices: linear approximation and log-linear, or percentage, approximation. The “hat” notation denotes the variables of the linear model. In a linear approximation,

\[
\hat{x}_t = (x_t - \bar{x})
\]

that is, \(\hat{x}_t\) is the unit difference from steady state. In a log-linear approximation,

\[
\hat{x}_t = (x_t - \bar{x})/\bar{x}
\]
so that $\hat{x}_t$ is measured in percentage difference from the steady state. Which approximation is applied will affect the $F_i$ matrices, parameter estimation, and the interpretation of all objects available after estimation.

Hatted variables are measured as deviations from steady state. At this point, the model is in the form of a linear DSGE model, and the methods of Klein (2000) may be employed to find the solution. The stability conditions of Klein (2000) apply to the $F_i$ matrices, and \texttt{dsgenl} will not proceed if the stability conditions are not met. If the stability conditions are met, we exactly solve the linear model. The solution is of the form

$$\begin{align*}
\hat{y}_t &= G\hat{x}_t \\
\hat{x}_{t+1} &= H\hat{x}_t + M\epsilon_{t+1}
\end{align*}$$

Because shocks $\epsilon_t$ are mean zero, it does not matter whether we write $\epsilon_t$ or $\hat{\epsilon}_t$.

In a nonlinear DSGE model, the linear solution matrices are a linear approximation to the true nonlinear solution functions,

$$\begin{align*}
y_t &= g(x_t) \\
x_{t+1} &= h(x_t) + M\epsilon_{t+1}
\end{align*}$$

The matrix $G$ is an approximation to the function $g(\cdot)$, and the matrix $H$ is an approximation to the function $h(\cdot)$. Once the solution has been obtained, we can view the steady state, the approximate solution matrices, the impulse responses, and other objects.

### Specifying the model to Stata

We import U.S. macro data. Although we are solving the model (hence, the data will play no role), \texttt{dsgenl} expects there to be data anyway.

```
. use https://www.stata-press.com/data/r16/usmacro2
   (Federal Reserve Economic Data - St. Louis Fed, 2017-01-15)
```

Next, we set up a parameter matrix containing the values that we will use for our model. We type

```
. matrix param_mat = (0.33, 0.96, 0.025, 0.9, 1)
. matrix colnames param_mat = alpha beta delta rho /sd(e.z)
```

These values are similar to those in King and Rebelo (1999) and Schmitt-Grohé and Uribe (2004). The elasticity of output with respect to capital $\alpha$ is set to 0.33. The discount factor $\beta$ is set to 0.96. The depreciation rate $\delta$ is set to 0.025. The persistence of productivity is set to 0.9, and the standard deviation of the shock to productivity is set to 1.

Next, we specify the model.
Intro 3f — Stochastic growth model

\[ dsgenl (1 = \{beta\}*(F.c/c)^{-1}*(1+F.r-{delta})) \]
\[ (r = \{alpha\}*z*k^{\{alpha\}-1}) \]
\[ (y = z*k^{\{alpha\}}) \]
\[ (F.k = y - c + (1-{delta})*k) \]
\[ (ln(F.z) = \{rho\}*ln(z)) \]
\[ , observed(y) unobserved(c r) exostate(z) endostate(k) \]
\[ from(param_mat) solve noidencheck \]

Solving at initial parameter vector ...

First-order DSGE model

Sample: 1955q1 - 2015q4
Number of obs = 244
Log likelihood = -2114.8044

<table>
<thead>
<tr>
<th></th>
<th>OIM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Coef.</td>
</tr>
<tr>
<td>y</td>
<td></td>
</tr>
<tr>
<td>/structural</td>
<td></td>
</tr>
<tr>
<td>beta</td>
<td>.96</td>
</tr>
<tr>
<td>delta</td>
<td>.025</td>
</tr>
<tr>
<td>alpha</td>
<td>.33</td>
</tr>
<tr>
<td>rho</td>
<td>.9</td>
</tr>
<tr>
<td>sd(e.z)</td>
<td>1</td>
</tr>
</tbody>
</table>

Note: Skipped identification check.
Note: Model solved at specified parameters; maximization options ignored.

Each equation is bound in parentheses. Variables appear without any special modification; parameters are bound in parentheses. After the equation list, we specify seven options. The first four tell dsgenl how to interpret each variable in the model. The control variables \(y, r,\) and \(c\) are specified as being either observed() or unobserved(). Because we are only solving the model, it is inconsequential how we place control variables into these categories. In an estimation exercise, data on observed control variables are used to estimate model parameters, and which variables are observable can affect parameter identification. The state variables are divided into categories according to whether they are subject to shocks. The state variable \(k\) is not subject to shocks; it is an endogenous state variable. The state variable \(z\) is subject to shocks; it is an exogenous state variable.

The from() option specifies the starting values for the parameters. The solve option declares that we wish to solve the model at the values in from(). The noidencheck option skips the check for local identification.

The output table repeats the parameter vector that we passed in; it provides no new information. The notes below the estimation table remind us that we have skipped the identification check and that we have solved the model rather than estimated the model parameters.

After solving

After solving the model by running dsgen1, you can view the steady state, the approximate policy and transition matrices, and more using the estat commands.
The steady state

In the process of finding the solution, dsgenl finds the steady state of the model. The steady state is displayed with

```
. estat steady, compact
Location of model steady-state

<table>
<thead>
<tr>
<th>Coef.</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>k</td>
<td>10.88</td>
</tr>
<tr>
<td>z</td>
<td>1.00</td>
</tr>
<tr>
<td>c</td>
<td>1.93</td>
</tr>
<tr>
<td>r</td>
<td>0.07</td>
</tr>
<tr>
<td>y</td>
<td>2.20</td>
</tr>
</tbody>
</table>
```

Because we used dsgenl’s solve option, estat steady displays the steady state of the model under the initial parameter vector. When solve is not specified, estat steady displays the steady state of the model at the converged parameter values.

The steady-state level of the capital stock is 10.88 units, and the steady-state level of output is 2.20 units. The steady-state interest rate is 0.07, or 7%. Some of these values are more meaningful when expressed as ratios rather than when they are expressed as raw values. The model’s consumption-to-output ratio is about 0.88. The model’s capital-to-output ratio is about 4.9.

Approximations to the policy and transition matrices

Next, we approximate the policy and state transition matrices. estat policy provides the approximate policy matrix.

```
. estat policy, compact
Policy matrix

<table>
<thead>
<tr>
<th></th>
<th>k</th>
<th>z</th>
</tr>
</thead>
<tbody>
<tr>
<td>c</td>
<td>.6077069</td>
<td>.3431366</td>
</tr>
<tr>
<td>r</td>
<td>-.67</td>
<td>1</td>
</tr>
<tr>
<td>y</td>
<td>.33</td>
<td>1</td>
</tr>
</tbody>
</table>
```

By default, dsgenl uses a log-linear approximation, and the results of estat policy are also based on this approximation. This means that variables are measured as percentage deviations from steady state. The interpretation of the output equation in the policy matrix is that

\[
\frac{Y_t - \bar{Y}}{\bar{Y}} \approx 0.33 \frac{K_t - \bar{K}}{\bar{K}} + \frac{Z_t - \bar{Z}}{\bar{Z}}
\]

A log-linear approximation was chosen because many DSGE model equations are linear in logs, or nearly so. For instance, in logs, (2) is

\[
\log(Y_t) = \alpha \log(K_t) + \log(Z_t)
\]

just as the policy matrix describes.
The entries in the policy matrix are impact effects. They display the percentage change in a control variable that results from a one-percent change in a state variable. From the consumption equation, a 1% increase in $z$ causes consumption to rise by 34%. From the output equation, a 1% increase in $z$ causes output to rise by 1%.

`estat transition` provides the approximate state transition matrix.

```
. estat transition, compact
Transition matrix of state variables

<table>
<thead>
<tr>
<th></th>
<th>k</th>
<th>z</th>
</tr>
</thead>
<tbody>
<tr>
<td>k</td>
<td>F1.</td>
<td>.9340903</td>
</tr>
<tr>
<td>z</td>
<td>F1.</td>
<td>0</td>
</tr>
</tbody>
</table>
```

The state transition matrix displays how state variables in the next period depend on state variables today. A 1% increase in $z$ causes capital to rise by 0.14% one period in the future. Notice that an increase in $z$ has an effect on $k$, but an increase in $k$ has no cross-effect on $z$.

**Linear and log-linear approximations**

Specifying option `linearapprox` causes `dsgenl` to take a linear approximation rather than a log-linear approximation. We first re-solve the model.

```
. dsgenl (1 = {beta}*(F.c/c)^(-1)*(1+F.r-{delta}))
> (r = {alpha}*z*k^{({alpha}-1)})
> (y = z*k^{alpha})
> (F.k = y - c + (1-{delta})*k)
> (ln(F.z) = {rho}*ln(z))
> , observed(y) unobserved(c r) exostate(z) endostate(k)
> from(param_mat) solve noidencheck linearapprox
```

Solving at initial parameter vector ...

First-order DSGE model

Sample: 1955q1 - 2015q4  Number of obs = 244
Log likelihood = -808.47167

| Coef. | Std. Err. | z | P>|z| | [95% Conf. Interval] |
|-------|-----------|---|------|----------------------|
| /structural |
| beta   | .96       | . | .    | .        | .        |
| delta  | .025      | . | .    | .        | .        |
| alpha  | .33       | . | .    | .        | .        |
| rho    | .9        | . | .    | .        | .        |
| sd(e.z)| 1         | . | .    | .        | .        |

Note: Skipped identification check.
Note: Model solved at specified parameters; maximization options ignored.
We next examine the impact that the `linearapprox` option has on postsolution objects. The choice of linear or log-linear approximation does not affect the location of the steady state.

```
. estat steady, compact
Location of model steady-state

<table>
<thead>
<tr>
<th>Coef.</th>
</tr>
</thead>
<tbody>
<tr>
<td>k</td>
</tr>
<tr>
<td>z</td>
</tr>
<tr>
<td>c</td>
</tr>
<tr>
<td>r</td>
</tr>
<tr>
<td>y</td>
</tr>
</tbody>
</table>
```

This is to be expected; the steady state is the point around which approximations are taken, but its location does not depend on the choice of approximation.

The policy matrix has changed.

```
. estat policy, compact
Policy matrix

<table>
<thead>
<tr>
<th></th>
<th>k</th>
<th>z</th>
</tr>
</thead>
<tbody>
<tr>
<td>c</td>
<td>.1075764</td>
<td>.6610193</td>
</tr>
<tr>
<td>r</td>
<td>-.0041045</td>
<td>.0666667</td>
</tr>
<tr>
<td>y</td>
<td>.0666667</td>
<td>2.198462</td>
</tr>
</tbody>
</table>
```

Each entry displays the unit increase in a control variable after a unit increase in a state variable. As an example, we use the estimates in the last line of this table. The interpretation of the output equation is

\[
(Y_t - \bar{Y}) \approx 0.067(K_t - \bar{K}) + 2.2(Z_t - \bar{Z})
\]

so that a one-unit increase in the capital stock leads output to rise by 0.067 units. The other entries are interpreted analogously.

The state transition matrix is now

```
. estat transition, compact
Transition matrix of state variables

<table>
<thead>
<tr>
<th></th>
<th>k</th>
<th>z</th>
</tr>
</thead>
<tbody>
<tr>
<td>k</td>
<td>.9340903</td>
<td>1.537443</td>
</tr>
<tr>
<td>z</td>
<td>0</td>
<td>.9</td>
</tr>
</tbody>
</table>
```

The approximate capital accumulation equation is

\[
(K_{t+1} - \bar{K}) \approx 0.93(K_t - \bar{K}) + 1.54(Z_t - \bar{Z})
\]

so that a one-unit increase in productivity \(Z_t\) leads to a 1.54-unit increase in the capital stock in the next period.
References


Also see

[DSGE] Intro 1 — Introduction to DSGEs
[DSGE] Intro 3d — Nonlinear New Keynesian model
[DSGE] Intro 3e — Nonlinear New Classical model
[DSGE] dsgenl — Nonlinear dynamic stochastic general equilibrium models
[DSGE] dsgenl postestimation — Postestimation tools for dsgenl
Many DSGE models, when written in their natural representation based on economic theory, include problematic terms that do not fit into the algebraic form required to solve a DSGE model. This entry shows how to accommodate these problematic terms by defining a new state variable or a new control variable and then rewriting the equations so that they have the required form. We demonstrate how to make the required changes using \texttt{dsge}, but the advice in this entry applies to both \texttt{dsge} and \texttt{dsgenl}.

We assume that you are already familiar with the elements of a DSGE; see [DSGE] Intro 1.

Remarks and examples

Remarks are presented under the following headings:

- Introduction
- Shocks to a control equation
- Including a lag of a control variable
- Including a lag of a state variable
- Including an expectation of a control dated by more than one period ahead
- Including a second-order lag of a control variable
- Including an observed exogenous variable

Introduction

We can estimate the parameters of a DSGE model only if we can solve for its state-space form, and we can solve a DSGE model for its state-space form only if the equations for control variables and state variables have specific structures. Unfortunately, many DSGE models contain problematic terms that do not fit into these structures without some manipulation. Fortunately, the manipulation is easy, once you know the tricks discussed below.

As we discussed in Structural and reduced forms of DSGE models in [DSGE] Intro 1, the structural form of a DSGE model can be written as

\[
E_t[f(x_{t+1}, y_{t+1}, x_t, y_t, \theta)] = 0
\]  

and the structural form of a linearized DSGE model can be written as

\[
A_0 y_t = A_1 E_t(y_{t+1}) + A_2 y_t + A_3 x_t
\]  

\[
B_0 x_{t+1} = B_1 E_t(y_{t+1}) + B_2 y_t + B_3 x_t + C \epsilon_{t+1}
\]

where \( y_t \) is a vector of control variables, \( x_t \) is a vector of state variables, and \( \epsilon_t \) is a vector of shocks. \( A_0 \) through \( A_3 \) and \( B_0 \) through \( B_3 \) are matrices of parameters. The entries in \( A_i \) and \( B_j \) are all functions of the structural parameters, which we denote by vector \( \theta \). \( A_0 \) and \( B_0 \) are diagonal matrices, and \( A_2 \) has zeros on its diagonal. \( C \) is a selection matrix that determines which state variables are subject to shocks.
For nonlinear DSGE models, (1) specifies the required structure of the equations. Equations can contain current values of state and control variables and can contain one-step-ahead values of state and control variables. All other terms, such as future values dated more than one period ahead or lagged values, are problematic.

For linearized DSGE models, (2) specifies the required structure for control variable equations, and (3) specifies the required structure for state variable equations. Per (2), a control can depend on only three types of terms—expected values of control variables in the next period, current values of other control variables, and current values of state variables. All other terms are problematic. Per (3), the values of state variables in the next period can depend on only four types of terms—expected values of control variables in the next period, current values of control variables, current values of state variables, and shocks. All other terms are problematic.

When written based on economic theory, the structural forms of many DSGE models include problematic terms that do not fit into the required form. These problematic terms are handled by defining a new state variable or a new control variable and rewriting the equations to have the required form. We define a new state variable when the problematic term involves an exogenous, also known as a predetermined, variable. We define a new control variable when the problematic term includes a new endogenous variable. The rewriting process replaces the variable in the problematic term with the new state or the new control variable.

Problematic terms usually violate one of the following four restrictions:

1. Equations for control variables may not contain shocks.
2. Equations may not contain lagged control variables.
3. Equations may not contain lagged state variables.
4. Equations may not contain expectations of control variables dated by more than one period ahead.

All of these restrictions can be overcome by adding a new state variable or a new control variable and rewriting the model so that it has the required form. Here are five examples of the four restrictions and solutions to each one.

1. If you have an equation for a control variable that contains a shock, make this shock a new state variable; see *Shocks to a control equation* below.
2. If you have an equation for a control variable that contains a lagged control variable, make the lagged control variable a new state variable; see *Including a lag of a control variable* below.
3. If you have an equation for a state variable that contains a lag of a state variable, make the lagged state variable a new state variable; see *Including a lag of a state variable* below.
4. If you have an equation for a control variable that contains an expectation of a control variable dated by more than one period ahead, make it a new control variable. We use a new control variable instead of a new state variable because the far-future expectation is endogenous, not predetermined. See *Including an expectation of a control dated by more than one period ahead* below for details.
5. If you have an equation for a control variable that contains a second-order lag of a control variable, make the second-order lagged control variable a new state variable. You will also need to create a new state variable for the first-order lag. See *Including a second-order lag of a control variable* below for details.
A nuanced issue not covered by restrictions 1–4 is that all the observed variables in a DSGE must be endogenous control variables. This structure is usually not a problem, because DSGE models are usually generated by theories that treat everything observed as endogenous. Occasionally, however, models include an observed variable that is exogenous. For example, a small, open economy might not be able to affect its real exchange rate. The solution for this problem is to model the observed variable as a control variable that is equal to an exogenous state variable. See Including an observed exogenous variable below for details.

For the most part, this entry discusses how to make models with problematic terms fit into the structure of (1) or (2) and (3). Note that correlated state variables already fit into this structure. This correlation is modeled by off-diagonal elements in $B_3$, not by elements in $C$. See [DSGE] Intro 4g for an example.

**Shocks to a control equation**

Sometimes, we observe only a control variable with error. Suppose that the equation for the control variable consumption is

$$c_t = E_t c_{t+1} - r_t + \epsilon_t \quad (4)$$

where $c_t$ is consumption, $r_t$ is the interest rate, and $\epsilon_t$ is a shock. Shocks to a control variable are not allowed, so $\epsilon_t$ is a problematic term.

The solution entails three steps. First, we define a new state variable $u_t = \epsilon_t$. We define a new state instead of a new control because the shock is exogenous. Second, we write a state equation for $u_t$. Recall that the state equation specifies tomorrow’s state value in terms of today’s state value and tomorrow’s shock. The state equation for $u_t$ is

$$u_{t+1} = \epsilon_{t+1}$$

Third, we rewrite (4), substituting $u_t$ for $\epsilon_t$. The consumption equation becomes

$$c_t = E_t c_{t+1} - r_t + u_t$$

which is in the required form. In the `dsge` command, this is expressed as

```
. dsge ... ///
  (c = F.c - r + u) ///
  (F.u = 0, state) ///
...
```

See [DSGE] Intro 4a for a complete example.

**Including a lag of a control variable**

Models that mix adaptive and rational expectations contain a lag of the control variable and the expected future value of the control variable. Suppose our model had the following equation for the control variable $y_t$,

$$y_t = \alpha y_{t-1} + (1 - \alpha) E_t(y_{t+1}) - r_t \quad (4)$$

where $y_t$ is output and $r_t$ is the interest rate. The problematic term is $\alpha y_{t-1}$ because it involves a lag of the control variable.
The solution entails three steps. First, we define a new state variable \( L_y = y_{t-1} \). We define a new state instead of new control because the lagged control is predetermined, which makes it exogenous. Second, we write its state equation as

\[
L_{y_{t+1}} = y_t
\]

Third, we rewrite (4), substituting \( L_y \) for \( y_{t-1} \),

\[
y_t = \alpha L_y + (1 - \alpha)E_t(y_{t+1}) - r_t
\]

which is in the required form. To do this using the \texttt{dsge} command, we would type

\[
\begin{align*}
\text{. dsge ...} & \quad /// \\
(y = \{\alpha\} \cdot L_y + (1-\{\alpha\}) \cdot F.y - r) & \quad /// \\
(F.Ly = y, \text{ state noshock}) & \quad /// \\
\end{align*}
\]

See \cite{DSGE} Intro 4b for a complete example.

### Including a lag of a state variable

Lagged state variables are handled in a manner analogous to lagged control variables. Suppose we have an equation for a state variable \( z_t \) that contains a lag of itself. For example, this occurs when \( z_t \) is a second-order autoregressive process instead of a first-order autoregressive process. Specifically, our model contains the equation

\[
z_{t+1} = \rho_1 z_t + \rho_2 z_{t-1} + \epsilon_{t+1}
\]

(5)

The problematic term is \( \rho_2 z_{t-1} \) because it violates the form required by (2).

The solution entails three steps. First, we define a new state variable \( L_z = z_{t-1} \). We define a new state instead of new control because the lagged state is exogenous. Second, we write its state equation as

\[
L_{z_{t+1}} = z_t
\]

Third, we rewrite (5), substituting \( L_z \) for \( z_{t-1} \),

\[
z_{t+1} = \rho_1 z_t + \rho_2 L_z + \epsilon_{t+1}
\]

which is in the required form. In the \texttt{dsge} command, we would type

\[
\begin{align*}
\text{. dsge ...} & \quad ///
(F.z = \{\rho1\}*z + \{\rho2\}*Lz, \text{ state}) & \quad ///
(F.Lz = z, \text{ state noshock}) & \quad ///
\end{align*}
\]

See \cite{DSGE} Intro 4c for a complete example.

### Including an expectation of a control dated by more than one period ahead

Many models include an expectation of a control variable dated by more than one period ahead. For example, suppose that the equation for the control variable consumption growth is

\[
c_t = (1 - h) w_t + h E_t c_{t+2} + r_t
\]

(6)
where $c_t$ is consumption growth, $w_t$ is wage growth, and $r_t$ is the interest rate. Only expectations of control variables dated one period ahead are allowed, so $E_t(c_{t+2})$ is a problematic term.

The solution entails three steps. First, we define a new control variable $Fc_t = E_t(c_{t+1})$. The new variable is a control instead of a state because $E_t(c_{t+1})$ is endogenous instead of exogenous. Second, we include this new control equation,

$$Fc_t = E_t(c_{t+1})$$

Third, we substitute $Fc_{t+1}$ for $c_{t+2}$ in (6). The equation becomes

$$c_t = (1 - h)w_t + hE_t(Fc_{t+1}) + r_t$$

which is in the required form. Using the dsge command, we would type

```
. dsge ... 
   (c = (1-{h})*w + {h}*Fc + r) 
   (Fc = F.c, unobserved) 
   ... 
```

See [DSGE] Intro 4d for a complete example.

**Including a second-order lag of a control variable**

Higher-order lags require defining more than one new state variable, as we illustrate here.

Equations for the state variable capital frequently involve “time to build”, so that the control variable investment made in time $t$ becomes available only as new capital several periods in the future. Suppose it takes three periods to convert investment $x_t$ into capital. We denote current capital by $k_t$ and write this three-period relationship as

$$k_{t+1} = (1 - \delta)k_t + x_{t-2} \quad (7)$$

which specifies that the capital stock tomorrow is the sum of investment made two periods ago and the capital today that has not depreciated at rate $\delta$.

The term $x_{t-2}$ is problematic because it is a second-order lag of the state variable $x_t$, which violates the form required by (2).

The solution entails three steps. First, we define the new state variables $L2x_t = x_{t-2}$ and $Lx_t = x_{t-1}$. Second, we write their state equations as

$$L2x_{t+1} = Lx_t$$

and

$$Lx_{t+1} = x_{t}$$

Third, we rewrite (7) as

$$k_{t+1} = (1 - \delta)k_t + L2x_t$$

With the dsge command, we could fit this model as

```
. dsge ... 
   (F.k = (1-{delta})*k + L2x) 
   (F.L2x = Lx, state noshock) 
   (F.Lx = x, state noshock) 
   ... 
```

For a complete example, see [DSGE] Intro 4e.

In general, if the lag $t - k$ appears anywhere in any of your equations, you will need $k$ new state equations.
Including an observed exogenous variable

Sometimes, we want to treat an observed variable as exogenous. Suppose we want to treat the changes in an observed exchange rate $e_t$ as exogenous. The problem is that all the observed variables in a DSGE model must be modeled as endogenous control variables. The solution is to define a control variable that is equal to a state variable that models the exogenous process. So we could set the observed exchange rate $e_t$ to equal the exchange rate state, $es_t$,

$$e_t = es_t$$

and specify, say, a first-order autoregressive process for $es_t$,

$$es_{t+1} = \rho_e es_t + \eta_{t+1}$$

To fit this type of model using the \texttt{dsge} command, we would type

```
. dsge ... ///
    (e = es) ///
    (F.es = {rhoe}*es, state) ///
... 
```

See \cite{Intro4} for a complete example.

Also see

\cite{Intro2} — Learning the syntax
\cite{Intro3} — Classic DSGE examples
\cite{dsge} — Linear dynamic stochastic general equilibrium models
Intro 4a — Specifying a shock on a control variable

Description

A shock to a control variable is problematic because it does not fit into the form of a structural model that is required to solve for the state-space form. This entry shows how to solve this problem by defining a new state variable and rewriting the equations.

Remarks and examples

Remarks are presented under the following headings:

The model
Parameter estimation

The model

Equations (1)–(3) specify a model of consumption growth and growth in hours worked. The equation for consumption growth is subject to an additive shock.

\[ c_t = (1 - h)w_t + hE_t c_{t+1} + \epsilon_t \quad (1) \]
\[ n_t = w_t - \gamma c_t \quad (2) \]
\[ w_{t+1} = \rho w_t + \xi_{t+1} \quad (3) \]

Equation (1) specifies that consumption growth \( c_t \) is a linear combination of wage growth \( w_t \), expected future consumption growth \( E_t c_{t+1} \), and a consumption shock \( \epsilon_t \). Equation (2) specifies that the growth rate of hours worked \( n_t \) depends on wage growth and consumption growth. Equation (3) specifies an autoregressive process for wage growth. The structural parameter \( h \) controls the weights on wage growth and expected future consumption growth in the consumption equation.

One cannot solve the model in (1)–(3) for the state-space form because the shock \( \epsilon_t \) is added to the control equation \( c_t \) in (1). Shocks to control variables must be reparameterized as new state variables so that the model can be solved for its state-space form. We define the shocks as new state variables instead of new control variables because the shocks are exogenous. We rewrite the model in (1)–(3) so that the additive shock is a new state variable in (4)–(7). Specifically, we define a new state variable \( z_t \) such that \( z_t = \epsilon_t \) and write the model as

\[ c_t = (1 - h)w_t + hE_t c_{t+1} + z_t \quad (4) \]
\[ n_t = w_t - \gamma c_t \quad (5) \]
\[ w_{t+1} = \rho w_t + \xi_{t+1} \quad (6) \]
\[ z_{t+1} = \epsilon_{t+1} \quad (7) \]

That we replaced \( \epsilon_t \) with the new state variable \( z_t \) is the only difference between (4) and (1). Equations (5) and (6) are identical to their counterparts, (2) and (3). Equation (7) specifies the evolution of the new state variable \( z_t \); it is similar to (6), except that it lacks an autoregressive component.
Parameter estimation

We estimate the parameters of the model in (4)–(7) using U.S. data on consumption growth and growth in hours worked. We specify \( w \) and \( z \) as state variables. We specify \( c \) and \( n \) as control variables. Both \( c \) and \( n \) are treated as observed. We specify that \( w \) and \( z \) are subject to shocks.

```
. use https://www.stata-press.com/data/r16/usmacro2
    (Federal Reserve Economic Data - St. Louis Fed, 2017-01-15)
. dsge (c = (1-{h})*(w) + {h}*F.c + z)
>   (n = w - {gamma}*c)
>   (F.w = {rho}*w, state)
>   (F.z = , state)
    (setting technique to bfgs)
    Iteration 0:  log likelihood = -2514.2527
    Iteration 1:  log likelihood = -1264.15 (backed up)
    Iteration 2:  log likelihood = -1182.8885 (backed up)
    Iteration 3:  log likelihood = -1153.1143 (backed up)
    Iteration 4:  log likelihood = -1152.7297 (backed up)
    (switching technique to nr)
    Iteration 5:  log likelihood = -1152.4548
    Iteration 6:  log likelihood = -1134.679
    Iteration 7:  log likelihood = -1131.4427
    Iteration 8:  log likelihood = -1131.283
    Iteration 9:  log likelihood = -1131.2826
    Iteration 10: log likelihood = -1131.2826

DSGE model

Sample: 1955q1 - 2015q4   Number of obs = 244
Model:  Log likelihood = -1131.2826

| Coef.  | Std. Err. | z     | P>|z|   | [95% Conf. Interval]         |
|--------|-----------|-------|-------|-----------------------------|
| /structural  |  |  | | | |
| h      | .7642505  | .0360272 | 21.21 | 0.000 | .6936386 - .8348625 |
| gamma  | .2726181  | .1059468 | 2.57  | 0.010 | .0649663 - .4802699 |
| rho    | .6545212  | .0485627 | 13.48 | 0.000 | .55934 - .7497023  |
| sd(e.w)| 2.904762  | .1958651 | 2.520873 | 3.28865 |
| sd(e.z)| 2.077219  | .1400659 | 1.802695 | 2.351743 |
```

The estimate of \( h \) is 0.76. Being greater than 0.5, this indicates that consumption growth depends more strongly on expected future consumption growth than on current wage growth.

Also see

[DSGE] Intro 2 — Learning the syntax
[DSGE] Intro 4 — Writing a DSGE in a solvable form
Description

dse and dsgenl do not allow lags of control variables to be included in the model. The structural form of the model that is required so that the model can be solved for its state-space form does not include lags of control variables. This entry shows how to fit a DSGE model that involves a lag of control variables by defining a new state variable and rewriting the equations.

Remarks and examples

Remarks are presented under the following headings:

- A model with a lagged endogenous variable
- Parameter estimation

A model with a lagged endogenous variable

The model in (1)–(5) is similar to many in monetary economics in that it includes inertia in the interest rate because the interest rate depends on its lagged value.

\[ p_t = \beta E_t p_{t+1} + \kappa y_t \]  
(1)

\[ y_t = E_t y_{t+1} - (r_t - E_t p_{t+1} - \rho z_t) \]  
(2)

\[ r_t = \rho r_{t-1} + (1 - \rho) \left( \frac{1}{\beta} p_t + u_t \right) \]  
(3)

\[ z_{t+1} = \rho z_t + \epsilon_{t+1} \]  
(4)

\[ u_{t+1} = \rho u_t + \xi_{t+1} \]  
(5)

Equation (1) specifies the structural equation for inflation \( p_t \). Inflation is a linear combination of expected future inflation \( E_t(p_{t+1}) \) and output growth \( y_t \). Equation (2) specifies the structural equation for output growth. Output growth is a linear combination of expected future output growth \( E_t(y_{t+1}) \), the interest rate \( r_t \), expected future inflation, and the state \( z_t \). The state \( z_t \) is the first-order autoregressive process that drives output growth. Equation (3) specifies the structural equation for the interest rate. The interest rate depends on its own lagged value, the inflation rate, and the state \( z_t \). The state \( u_t \) is the first-order autoregressive process that drives the interest rate. The control variables in this model are \( p_t \), \( y_t \), and \( r_t \). The state variables are \( u_t \) and \( z_t \).

The term involving \( r_{t-1} \) in (3) is problematic because the lag of a control variable does not fit into the structure required to solve the model for the state-space form. We accommodate this term by defining a new state variable \( Lr_t \) that equals \( r_{t-1} \) and replacing \( r_{t-1} \) in (3) with this new state variable. We define a new state instead of new control because the lagged control is predetermined, which makes it exogenous. These changes yield the model in (6)–(11).
\[ p_t = \beta E_t p_{t+1} + \kappa y_t \]  
\[ y_t = E_t y_{t+1} - (r_t - E_t p_{t+1} - \rho z_t) \]  
\[ r_t = \rho_r L r_t + (1 - \rho_r) \left( \frac{1}{\beta} p_t + u_t \right) \]  
\[ L r_{t+1} = r_t \]  
\[ z_{t+1} = \rho_z z_t + \epsilon_{t+1} \]  
\[ u_{t+1} = \rho_u u_t + \xi_{t+1} \]  

Parameter estimation

We will estimate the parameters in the model in (6)–(11) using data on U.S. interest rate \( r \) and inflation \( p \).

```stata
. use https://www.stata-press.com/data/r16/usmacro2
(Federal Reserve Economic Data - St. Louis Fed, 2017-01-15)
. dsge (p = {beta}*F.p + {kappa}*y)
> (y = F.y - (r - f.p - {rhoz}*z), unobserved)
> (r = {rhor}*lr + (1-{rhor})*((1/{beta})*p + u))
> (F.lr = r, state noshock)
> (F.u = {rhou}*u, state)
> (F.z = {rhoz}*z, state)
(setting technique to bfgs)
Iteration 0:  log likelihood = -158313.11
Iteration 1:  log likelihood = -5780.6489 (backed up)
Iteration 2:  log likelihood = -1028.4602 (backed up)
Iteration 3:  log likelihood = -1000.851 (backed up)
Iteration 4:  log likelihood = -873.3804 (backed up)
(switching technique to nr)
Iteration 5:  log likelihood = -855.55061 (not concave)
Iteration 6:  log likelihood = -799.322 (not concave)
Iteration 7:  log likelihood = -784.44106 (not concave)
Iteration 8:  log likelihood = -776.37874 (not concave)
Iteration 9:  log likelihood = -771.3587 (not concave)
Iteration 10: log likelihood = -767.02524 (not concave)
Iteration 11: log likelihood = -764.16952 (not concave)
Iteration 12: log likelihood = -761.74111 (not concave)
Iteration 13: log likelihood = -759.86071 (not concave)
Iteration 14: log likelihood = -758.21296
Iteration 15: log likelihood = -755.96039
Iteration 16: log likelihood = -754.36723
Iteration 17: log likelihood = -753.19361
Iteration 18: log likelihood = -753.07315
Iteration 19: log likelihood = -753.07026
Iteration 20: log likelihood = -753.07026
```
DSGE model
Sample: 1955q1 - 2015q4
Number of obs = 244
Log likelihood = -753.07026

|                  | Coef. | Std. Err. | z    | P>|z| | [95% Conf. Interval] |
|------------------|-------|-----------|------|------|----------------------|
| /structural      |       |           |      |      |                      |
| beta             | 0.5026369 | 0.0791865 | 6.35 | 0.000 | 0.3474342 - 0.6578397 |
| kappa            | 0.1760193 | 0.0511049 | 3.44 | 0.001 | 0.0758554 - 0.2761831 |
| rhoz             | 0.9591408 | 0.0181341 | 52.89 | 0.000 | 0.9235986 - 0.994683  |
| rhor             | -0.0939028 | 0.0939661 | -1.00 | 0.318 | -0.278073 - 0.0902673 |
| rhoul            | 0.7094625 | 0.0447807 | 15.84 | 0.000 | 0.6216939 - 0.7972311 |
| sd(e.u)          | 2.324311  | 0.3236236 |      |      | 1.69002 2.958602     |
| sd(e.z)          | 0.6111539 | 0.1084981 |      |      | 0.3985014 0.8238063  |

Ironically, the inertia term for the interest rate $\rho_{hor}$ was not needed in this model. The autoregressive state $u_t$ is probably modeling both the inertia and the persistence in the shocks to the interest rate.

Also see

[DSGE] Intro 2 — Learning the syntax
[DSGE] Intro 4 — Writing a DSGE in a solvable form
Lags of state variables are not allowed to be included in the structural models specified in the \texttt{dsge} and \texttt{dsgenl} commands. These lags do not fit into the form of a structural model that can be solved for the state-space form. However, this restriction does not prevent us from fitting models that involve the lag of a state variable. This entry shows how to define a new state variable and rewrite the equations in the required structural form.

### Remarks and examples

Remarks are presented under the following headings:

- \textit{A model with a lagged state variable}
- \textit{Parameter estimation}

### A model with a lagged state variable

The model in (1)–(5) is a standard monetary model in which the state-driving output growth $y_t$ is a second-order autoregressive AR($2$) process instead of a first-order autoregressive process.

\begin{align*}
  p_t &= \beta E_t(p_{t+1}) + \kappa y_t \\
  y_t &= E_t(y_{t+1}) - \{r_t - E_t(p_{t+1}) - z_t\} \\
  r_t &= \frac{1}{\beta} p_t + u_t \\
  z_{t+1} &= \rho_1 z_t + \rho_2 z_{t-1} + \epsilon_{t+1} \\
  u_{t+1} &= \rho_u u_t + \xi_{t+1}
\end{align*}

Equation (1) specifies the structural equation for inflation $p_t$. Inflation is a linear combination of expected future inflation $E_t(p_{t+1})$ and output growth $y_t$. Equation (2) specifies the structural equation for output growth. Output growth is a linear combination of expected future output growth $E_t(y_{t+1})$, the interest rate $r_t$, expected future inflation, and the state $z_t$. The state $z_t$ is an AR(2) process that drives output growth. Equation (3) specifies the structural equation for the interest rate. The interest rate depends on the inflation rate and the state $u_t$. The state $u_t$ is an AR(1) process that drives the interest rate. The control variables in this model are $p_t$, $y_t$, and $r_t$. The state variables are $u_t$ and $z_t$. 
The AR(2) term in (4) is a problematic term because a lag of a state variable does not fit into the structure required to solve the model for the state-space form. We accommodate this term by defining a new state variable $Lz_t$ that equals $z_{t-1}$ and replacing $z_{t-1}$ in (4) with this new state variable. We define a new state instead of new control because the lagged state is exogenous. These changes yield the model in (6)–(11).

\[
\begin{align*}
    p_t &= \beta E_t(p_{t+1}) + \kappa y_t & (6) \\
    y_t &= E_t(y_{t+1}) - \{r_t - E_t(p_{t+1}) - z_t\} & (7) \\
    r_t &= \frac{1}{\beta} p_t + u_t & (8) \\
    z_{t+1} &= \rho z_t + \rho_2 Lz_t + \epsilon_{t+1} & (9) \\
    u_{t+1} &= \rho u_t + \xi_{t+1} & (10) \\
    Lz_{t+1} &= z_t & (11)
\end{align*}
\]

We will estimate the parameters in the model in (6)–(11).
Parameter estimation

We estimate the parameters in the model in (6)–(11) using data on U.S. interest rate $r$ and inflation $p$.

```
. use https://www.stata-press.com/data/r16/usmacro2
(Federal Reserve Economic Data - St. Louis Fed, 2017-01-15)
. dsge (p = {beta}*F.p + {kappa}*y)
>     (y = F.y - (r - F.p - z), unobserved)
>     (r = (1/{beta})*p + u)
>     (F.u = {rhou}*u, state)
>     (F.z = {rhoz1}*z + {rhoz2}*Lz, state)
>     (F.Lz = z, state noshock)
(setting technique to bfgs)
Iteration 0: log likelihood = -9116.0985
Iteration 1: log likelihood = -997.95396 (backed up)
Iteration 2: log likelihood = -881.53897 (backed up)
Iteration 3: log likelihood = -836.47478 (backed up)
Iteration 4: log likelihood = -769.37111 (backed up)
(setting technique to nr)
Iteration 5: log likelihood = -768.80775 (not concave)
Iteration 6: log likelihood = -760.17544 (not concave)
Iteration 7: log likelihood = -755.99765 (not concave)
Iteration 8: log likelihood = -754.93526 (not concave)
Iteration 9: log likelihood = -754.42155
Iteration 10: log likelihood = -753.8269
Iteration 11: log likelihood = -753.07788
Iteration 12: log likelihood = -753.07788
Iteration 13: log likelihood = -753.07788
Iteration 14: log likelihood = -753.07788

DSGE model
Sample: 1955q1 - 2015q4 Number of obs = 244
Log likelihood = -753.07788

<table>
<thead>
<tr>
<th></th>
<th>OIM</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Coef.</td>
<td>Std. Err.</td>
<td>z</td>
<td>P&gt;</td>
<td>z</td>
</tr>
<tr>
<td>/structural</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>beta</td>
<td>.5154874</td>
<td>.0770496</td>
<td>6.69</td>
<td>0.000</td>
<td>.364473</td>
</tr>
<tr>
<td>kappa</td>
<td>.1662425</td>
<td>.0468473</td>
<td>3.55</td>
<td>0.000</td>
<td>.0744234</td>
</tr>
<tr>
<td>rhou</td>
<td>.6979877</td>
<td>.0452071</td>
<td>15.44</td>
<td>0.000</td>
<td>.6093834</td>
</tr>
<tr>
<td>rhoz1</td>
<td>.6735462</td>
<td>.2666028</td>
<td>2.53</td>
<td>0.012</td>
<td>.1510143</td>
</tr>
<tr>
<td>rhoz2</td>
<td>.2710021</td>
<td>.2564596</td>
<td>1.06</td>
<td>0.291</td>
<td>-.2316495</td>
</tr>
<tr>
<td>sd(e.u)</td>
<td>2.315262</td>
<td>.2992183</td>
<td></td>
<td></td>
<td>1.728805</td>
</tr>
<tr>
<td>sd(e.z)</td>
<td>.7720678</td>
<td>.1716214</td>
<td></td>
<td></td>
<td>.4356961</td>
</tr>
</tbody>
</table>
```

Looking at the confidence interval on rhoz2, we find no evidence that the AR(2) term is needed in this model.

Also see

[DSGE] Intro 2 — Learning the syntax
[DSGE] Intro 4 — Writing a DSGE in a solvable form
Intro 4d — Including an expectation dated by more than one period ahead

Description

Expectations of control variables dated by more than one period ahead are not allowed in the equations specified in dsge and dsgenl because they do not appear in the required form of a structural model that can be solved for the state-space form. In this entry, we demonstrate how to fit models with these types of expectations by defining a new variable and rewriting our equations. Because the expectation dated more than one period ahead is endogenous, the new variable that we define is a new control variable.

Remarks and examples

Remarks are presented under the following headings:

The model
Parameter estimation

The model

Equations (1)–(4) specify a model of consumption growth and growth in hours worked.

\[ c_t = (1 - h)w_t + hE_{t}c_{t+2} + r_t \]  
\[ n_t = w_t - \gamma c_t \]  
\[ w_{t+1} = \rho w w_t + \xi_{t+1} \]  
\[ r_{t+1} = \rho r r_t + \epsilon_{t+1} \]  

Equation (1) specifies that consumption growth \( c_t \) is a linear combination of wage growth \( w_t \), the expected value of consumption growth two periods ahead \( E_{t}c_{t+2} \), and the interest rate \( r_t \). Equation (2) specifies that the growth rate of hours worked \( n_t \) depends on wage growth and consumption growth. Equations (3) and (4) specify a first-order autoregressive process for wage growth and for the interest rate, respectively. The control variables are \( c_t \) and \( n_t \), and the state variables are \( w_t \) and \( r_t \).
The expected value of consumption growth two periods ahead in (1) is a problematic term because it does not fit into the structure required to solve for the state-space form. The structure requires that expectations be only of one-period ahead values. We accommodate this term by defining a new control variable $F_{c_t}$ that equals $E_t(c_{t+1})$ and replacing $c_{t+1}$ in (1) with this new control variable. We define a new control variable instead of a new state variable because the expected future consumption is endogenous instead of exogenous. These changes yield the model in (5)–(9).

\[
\begin{align*}
    c_t &= (1 - h)w_t + hE_t(F_{c_{t+1}}) + r_t \\
    n_t &= w_t - \gamma c_t \\
    Fc_t &= E_t(c_{t+1}) \\
    w_{t+1} &= \rho_w w_t + \xi_{t+1} \\
    r_{t+1} &= \rho_r r_t + \epsilon_{t+1}
\end{align*}
\]

New (7) defines the new control variable $F_{c_t}$ as the expected value of next period’s consumption. Equation (5) is (1) but with $c_{t+2}$ replaced with the new control $F_{c_{t+1}}$.

There are now three control variables and only two shocks, so we treat the new control variable $F_{c_t}$ as unobserved. There is a logic to this process. The one-step-ahead structure of state-space models makes it impossible to solve for terms like $E_t(c_{t+2})$. In contrast, it is possible to solve for terms like $E_t\{E_t(c_{t+1})\}$ as long as the unobserved $E_t(c_{t+1})$ is determined by another equation, which in this case is (7). In fact, this example illustrates a part of the recursive dynamic modeling approach at the heart of the DSGE approach to macroeconomics.
Parameter estimation

We estimate the parameters of the model in (5)–(9) using U.S. data on consumption growth and growth in hours worked.

```
. use https://www.stata-press.com/data/r16/usmacro2
   (Federal Reserve Economic Data - St. Louis Fed, 2017-01-15)
. dsge (c = (1-{h})*w + {h}*F.fc + r)  
   > (n = w - {gamma}*c)
   > (fc = F.c, unobserved)
   > (F.w = {rho_w}*w, state)
   > (F.r = {rho_r}*r, state)
   (setting technique to bfgs)
   Iteration 0: log likelihood = -2423.7325
   Iteration 1: log likelihood = -1284.9295 (backed up)
   Iteration 2: log likelihood = -1193.2234 (backed up)
   Iteration 3: log likelihood = -1180.1787 (backed up)
   Iteration 4: log likelihood = -1175.3563 (backed up)
   (switching technique to nr)
   Iteration 5: log likelihood = -1171.0676 (backed up)
   Iteration 6: log likelihood = -1152.0355
   Iteration 7: log likelihood = -1134.7089
   Iteration 8: log likelihood = -1130.0791
   Iteration 9: log likelihood = -1129.9372
   Iteration 10: log likelihood = -1129.9357
   Iteration 11: log likelihood = -1129.9357
DSGE model
Sample: 1955q1 - 2015q4 Number of obs = 244
Log likelihood = -1129.9357

                   | OIM               |
                   | Coef.   Std. Err. |   z     P>|z|   [95% Conf. Interval] |
---------|------------------|
/structural  |                  |
   h       | 0.661791         | .0427917 15.47 0.000   .5779208   .7456611 |
  gamma    | 0.273381         | .1120734  2.44 0.015   .0537212   .4930408 |
   rho_w   | 0.6545228        | .0485628 13.48 0.000   .5593415   .7497041 |
   rho_r   | 1.050387         | .0638171  1.65 0.100  -.0200406   .230118 |
   sd(e.w) | 2.905807         | .2023204       | 2.509267   3.302348 |
   sd(e.r) | 2.049915         | .1437862       | 1.7681   2.331731 |
```

The estimate of $h$ is greater than 0.5, so we conclude that slightly more of the unobserved wage state is allocated to the expected growth in consumption two periods ahead than to current consumption.

Also see

[DSGE] Intro 2 — Learning the syntax
[DSGE] Intro 4 — Writing a DSGE in a solvable form
Some DSGE models capture delayed effects by including a second-order lag of a control variable and excluding the first-order lag. The second-order lag is a problematic term that does not fit into the form required to solve a structural model for its state-space form. This entry shows how to solve this problem by defining new state variables and rewriting the equations.

**Remarks and examples**

Remarks are presented under the following headings:

- The model
- Parameter estimation

**The model**

Consider a model in which changes in hours worked take two periods to adjust because next period’s hours have already been budgeted. In this model, the second-order lag of changes in hours worked is included, and the first-order lag is excluded. Equations (1)–(4) specify such a model of growth in hours worked and of consumption growth.

\[
\begin{align*}
n_t &= b_1 n_{t-2} + w_t - \gamma c_t \\
c_t &= (1 - h) w_t + h E_t c_{t+1} + r_t \\
w_{t+1} &= \rho w_t + \xi_{t+1} \\
r_{t+1} &= \epsilon_{t+1}
\end{align*}
\]

Equation (1) specifies that the growth rate of hours worked \(n_t\) depends on a second-order lag of itself, wage growth \(w_t\), and consumption growth \(c_t\). Equation (2) specifies that consumption growth is a linear combination of wage growth, expected future consumption growth \(E_t c_{t+1}\), and the interest rate \(r_t\). Equation (3) specifies an autoregressive process for wage growth. Equation (4) specifies that interest rate is just a shock. The control variables are \(n_t\) and \(c_t\). The state variables are \(w_t\) and \(r_t\).

One cannot solve the model in (1)–(4) for the state-space form because the problematic term \(b_1 n_{t-2}\) does not fit into the required form. To accommodate this term, we define two new state variables, one for \(n_{t-1}\) and one for \(n_{t-2}\). We define new state variables instead of new control variables because lags of the control are predetermined and thus exogenous. The model with new state variables is
\[ n_t = b_1 L_2 n_t + w_t - \gamma c_t \]  
\[ c_t = (1 - h) w_t + h E_t c_{t+1} + r_t \]  
\[ w_{t+1} = \rho w_t + \xi_{t+1} \]  
\[ r_{t+1} = \epsilon_{t+1} \]  
\[ L n_{t+1} = n_t \]  
\[ L_2 n_{t+1} = L n_t \]  

Equation (9) defines the new state for \( n_{t-1} \), and (10) defines \( L_2 n_t \) to be the new state for \( n_{t-2} \). The \( L_2 n_t \) in (5) replaces \( n_{t-2} \) in (1).

### Parameter estimation

We specify \( n \) and \( c \) as observed control equations. We specify \( w, r, L_n, \) and \( L_2 n \) as state equations. We specify that \( w \) and \( r \) are subject to shocks; the new states to accommodate \( n_{t-2} \) are not subject to shocks.

```
. use https://www.stata-press.com/data/r16/usmacro2
   (Federal Reserve Economic Data – St. Louis Fed, 2017-01-15)
. dsge (n = {b1}*L2n + w - {gamma}*c)
  > (c = (1-{h})*w + {h}*F.c + r)
  > (F.w = {rho}*w, state)
  > (F.r = , state)
  > (F.L2n = Ln, state noshock)
  > (F.Ln = n, state noshock)
(setting technique to bfgs)
Iteration 0:  log likelihood = -2325.1996
Iteration 1:  log likelihood = -1277.0146 (backed up)
Iteration 2:  log likelihood = -1193.4512 (backed up)
Iteration 3:  log likelihood = -1189.3181 (backed up)
Iteration 4:  log likelihood = -1188.2629 (backed up)
(iterating technique to nr)
Iteration 5:  log likelihood = -1187.9872 (backed up)
Iteration 6:  log likelihood = -1147.3379
Iteration 7:  log likelihood = -1131.5153
Iteration 8:  log likelihood = -1129.0358
Iteration 9:  log likelihood = -1129.0181
Iteration 10: log likelihood = -1129.0181

DSGE model

Sample: 1955q1 – 2015q4  Number of obs = 244
Log likelihood = -1129.0181

<table>
<thead>
<tr>
<th></th>
<th>OIM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Coef.</td>
</tr>
<tr>
<td>/structural</td>
<td></td>
</tr>
<tr>
<td>b1</td>
<td>.1320845</td>
</tr>
<tr>
<td>gamma</td>
<td>.360925</td>
</tr>
<tr>
<td>h</td>
<td>.7238121</td>
</tr>
<tr>
<td>rho</td>
<td>.6177969</td>
</tr>
<tr>
<td>sd(e.w)</td>
<td>3.033799</td>
</tr>
<tr>
<td>sd(e.r)</td>
<td>1.970288</td>
</tr>
</tbody>
</table>
```
Looking at the confidence interval for $b_1$, we conclude that the second-order lag of hours’ growth impacts current hours’ growth.

Also see

[DSGE] Intro 2 — Learning the syntax

[DSGE] Intro 4 — Writing a DSGE in a solvable form
All the observed variables in a DSGE model must be modeled as endogenous control variables. This requirement implies that there is no reduced form for the endogenous variables as a function of observed exogenous variables. Any variable that is theoretically exogenous must be modeled.

Mechanically, the solution is to define a control variable that is equal to a state variable that models the exogenous process. We clarify this issue by discussing a model in which the exchange rate is theoretically exogenous.

Remarks and examples

Remarks are presented under the following headings:

The model
Parameter estimation

The model

We model an economy in which the growth rate of the trade-weighted exchange rate is exogenous and in which it affects inflation. Henceforth, we call the trade-weighted exchange rate just the exchange rate. We begin by adding the growth rate of the exchange rate to the inflation equation in the New Keynesian model of DSGE Intro 1.

\[ x_t = E_t(x_{t+1}) - \{r_t - E_t(p_{t+1}) - g_t\} \]  
\[ r_t = \frac{1}{\beta}p_t + u_t \]  
\[ p_t = \beta E_t(p_{t+1}) + \kappa x_t + \psi e_t \]  
\[ u_{t+1} = \rho_u u_t + \epsilon_{t+1} \]  
\[ g_{t+1} = \rho_g g_t + \xi_{t+1} \]  

\( x_t \) is the output gap, which is modeled as an unobserved control variable. \( r_t \) is the interest rate, which is modeled as an observed control variable. \( p_t \) is the inflation rate, which is modeled as an observed control variable. \( g_t \) and \( u_t \) are first-order autoregressive state variables. \( e_t \) is the growth rate of the exchange rate, which we want to model as an observed exogenous variable.
The model in (1)–(5) cannot be solved because there is no equation for how the observed $e_t$ evolves over time. A first-order autoregressive process [AR(1)] is a standard approximation to an exogenous variable, like the growth in exchange rate. We complete the model by adding an AR(1) for the unobserved state variable $e_{st}$ and an equation linking the unobserved $e_{st}$ to the observed $e_t$.

\[
x_t = E_t(x_{t+1}) - \{r_t - E_t(p_{t+1}) - g_t\} \\
r_t = \frac{1}{\beta}p_t + u_t \\
p_t = \beta E_t(p_{t+1}) + \kappa x_t + \psi e_{st} \\
e_t = e_{st} \\
u_{t+1} = \rho_u u_t + \epsilon_{t+1} \\
g_{t+1} = \rho_g g_t + \xi_{t+1} \\
e_{st+1} = \rho_e e_{st} + \eta_{t+1}
\]

New (8a) specifies how the unobserved state $e_{st}$ is transformed into the observed control variable $e_t$. Equation (11) specifies that the unobserved state $e_{st}$ is an AR(1) process. Other equations are unchanged.
Parameter estimation

We fit this model using data on the U.S. interest rate $r$, inflation rate $p$, and the growth rate of the exchange rate $e$. The equation ($e = es$) links the observed variable $e$ to the unobserved state variable $es$.

```
. use https://www.stata-press.com/data/r16/usmacro2
    (Federal Reserve Economic Data – St. Louis Fed, 2017-01-15)
. dsge (x = F.x - (r - F.p - g), unobserved)
>     (r = 1/\{beta}\*p + u)
>     (p = \{beta\}*F.p + \{kappa\}*x + \{psi\}*es)
>     (e = es)
>     (F.u = \{rho_u\}*u, state)
>     (F.g = \{rho_g\}*g, state)
>     (F.es = \{rho_e\}*es, state)

(setting technique to bfgs)
Iteration 0: log likelihood = -24249.537
Iteration 1: log likelihood = -8517.9588 (backed up)
Iteration 2: log likelihood = -1808.8075 (backed up)
Iteration 3: log likelihood = -1625.0334 (backed up)
Iteration 4: log likelihood = -1600.0843 (backed up)
(switching technique to nr)
Iteration 5: log likelihood = -1557.8925 (not concave)
Iteration 6: log likelihood = -1512.705 (not concave)
Iteration 7: log likelihood = -1481.8476 (not concave)
Iteration 8: log likelihood = -1278.0033 (not concave)
Iteration 9: log likelihood = -1198.5095 (not concave)
Iteration 10: log likelihood = -1186.0134
Iteration 11: log likelihood = -1177.2355 (not concave)
Iteration 12: log likelihood = -1175.3012
Iteration 13: log likelihood = -1174.3886
Iteration 14: log likelihood = -1172.7244
Iteration 15: log likelihood = -1172.3166
Iteration 16: log likelihood = -1172.0638
Iteration 17: log likelihood = -1172.0376
Iteration 18: log likelihood = -1172.0364
Iteration 19: log likelihood = -1172.0364

DSGE model
Sample: 1973q2 - 2015q4 Number of obs = 171
Log likelihood = -1172.0364

| Coef.  | Std. Err. | z     | P>|z| | [95% Conf. Interval] |
|--------|-----------|-------|------|---------------------|
| /structural
| beta   | .5106735  | .1022178 | 5.00 | 0.000 | .3103303 | .7110167 |
| kappa  | .0901563  | .0334284 | 2.70 | 0.007 | .0246378 | .1556747 |
| psi    | .01373    | .0037684 | 3.64 | 0.000 | .006344  | .021116 |
| rho_u  | .7789341  | .047395  | 16.43| 0.000 | .6860415 | .8718266 |
| rho_g  | .9597842  | .0206193 | 46.55| 0.000 | .9193712 | 1.000197 |
| rho_e  | .2372214  | .0742225 | 3.20 | 0.001 | .0917479 | .3826948 |
| sd(e.u) | 2.206087  | .3466166 |        |       | 1.526731 | 2.885443 |
| sd(e.g) | .703933   | .1545767 |        |       | .4009683 | 1.006898 |
| sd(e.es)| 10.48932  | .5671986 |        |       | 9.37763 | 11.60101 |

From the estimates for $\psi$, it appears that the growth of the exchange rate impacts inflation.
Also see

[DSGE] Intro 2 — Learning the syntax

[DSGE] Intro 4 — Writing a DSGE in a solvable form
Description

Many models include correlated state variables. We illustrate how to specify correlated state variables in a model of output growth $y_t$ and inflation $p_t$.

Remarks and examples

Remarks are presented under the following headings:

The model
Parameter estimation

The model

We model output growth $y_t$ and inflation $p_t$ as functions of domestic and international factors. The domestic factor $g_t$ that drives output growth is a first-order autoregressive process that is also affected by the international factor $z_t$. The international factor that drives inflation is a simple first-order autoregressive process.

Mathematically, the model is

\begin{align*}
y_t &= E_t y_{t+1} + \alpha p_t + g_t \\
p_t &= z_t \\
g_{t+1} &= \rho_g g_t + \rho_g z_t + \xi_{t+1} \\
z_{t+1} &= \rho_z z_t + \epsilon_{t+1}
\end{align*}

Equation (1) specifies that output growth depends on expected future output growth $E_t y_{t+1}$, inflation $p_t$, and the domestic factor $g_t$. This equation has the form of an aggregate demand curve, and the parameter $\alpha$ is referred to as the slope of the aggregate demand curve. It should be negative. Equation (2) specifies that inflation is entirely driven by the international factor $z_t$. Equations (3) and (4) specify that the factors $g_t$ and $z_t$ follow a first-order vector autoregressive process with parameters $\rho_g$, $\rho_g z$, and $\rho_z$ and with shocks $\xi_{t+1}$ and $\epsilon_{t+1}$. The factors $z_t$ and $g_t$ are the state variables, and $p_t$ and $y_t$ are the observed control variables.
Parameter estimation

From the U.S. macroeconomic data, we use the GDP-growth data in $y$ and the inflation data in $p$ and estimate the parameters of this model.

\begin{align*}
\text{dsge} (y = F.y + \{\alpha\}*p + g) \\
(p = z) \\
(F.g = \{\rho_g\}*g + \{\rho_gz\}*z, \text{state}) \\
(F.z = \{\rho_z\}*z, \text{state})
\end{align*}

(setting technique to bfgs)

Iteration 0: log likelihood = -2106.7245
Iteration 1: log likelihood = -1563.7386 (backed up)
Iteration 2: log likelihood = -1363.6417 (backed up)
Iteration 3: log likelihood = -1289.3079 (backed up)
Iteration 4: log likelihood = -1289.3079 (backed up)

(switching technique to nr)

Iteration 5: log likelihood = -1175.4471 (not concave)
Iteration 6: log likelihood = -1121.5017 (not concave)
Iteration 7: log likelihood = -1111.6839 (not concave)
Iteration 8: log likelihood = -1104.5102 (not concave)
Iteration 9: log likelihood = -1098.5517 (not concave)
Iteration 10: log likelihood = -1085.8018 (not concave)
Iteration 11: log likelihood = -1074.273 (not concave)
Iteration 12: log likelihood = -1071.4913 (not concave)
Iteration 13: log likelihood = -1069.9378 (not concave)
Iteration 14: log likelihood = -1060.2549 (not concave)
Iteration 15: log likelihood = -1057.046
Iteration 16: log likelihood = -1040.5911 (not concave)
Iteration 17: log likelihood = -1028.4304 (not concave)
Iteration 18: log likelihood = -1023.5103 (not concave)
Iteration 19: log likelihood = -1021.3008
Iteration 20: log likelihood = -1018.6421 (not concave)
Iteration 21: log likelihood = -1017.6527
Iteration 22: log likelihood = -1017.315
Iteration 23: log likelihood = -1017.164
Iteration 24: log likelihood = -1017.1592
Iteration 25: log likelihood = -1017.1592

DSGE model

Sample: 1955q1 - 2015q4 Number of obs = 244
Log likelihood = -1017.1592

<table>
<thead>
<tr>
<th></th>
<th>OIM</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Coef.</td>
<td>Std. Err.</td>
<td>z</td>
<td>P&gt;</td>
<td>z</td>
</tr>
<tr>
<td>/structural</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>alpha</td>
<td>-.1130023</td>
<td>.1554398</td>
<td>-0.73</td>
<td>0.467</td>
<td>-.4176587</td>
</tr>
<tr>
<td>rho_g</td>
<td>.335777</td>
<td>.0610763</td>
<td>5.50</td>
<td>0.000</td>
<td>.2160696</td>
</tr>
<tr>
<td>rho_gz</td>
<td>.0443504</td>
<td>.09013</td>
<td>0.49</td>
<td>0.623</td>
<td>-.1323013</td>
</tr>
<tr>
<td>rho_z</td>
<td>.8626566</td>
<td>.0319007</td>
<td>27.04</td>
<td>0.000</td>
<td>.8001324</td>
</tr>
<tr>
<td>sd(e.g)</td>
<td>2.184805</td>
<td>.2241105</td>
<td></td>
<td></td>
<td>1.745557</td>
</tr>
<tr>
<td>sd(e.z)</td>
<td>1.146947</td>
<td>.0519234</td>
<td></td>
<td></td>
<td>1.045179</td>
</tr>
</tbody>
</table>

The slope of the aggregate demand curve, $\alpha$, is estimated to be negative as we expected, but the confidence interval is wide and includes zero. The imprecision in $\rho_{gz}$ has caused imprecision in the estimate of $\alpha$. 
Also see

[DSGE] Intro 2 — Learning the syntax
[DSGE] Intro 4 — Writing a DSGE in a solvable form
Description

We can estimate only the parameters of models that we can solve, and we can solve only models that satisfy a stability condition. This entry discusses this condition, how it affects estimation in practice, and how to find initial values that satisfy it.

Remarks and examples

Remarks are presented under the following headings:

- Why we care about stability
- What if the initial values are not saddle-path stable?

Why we care about stability

DSGE models are dynamic systems subject to random shocks. DSGE models that do not spiral out of control or converge to a single point when shocked are said to be “saddle-path stable”. We can solve only saddle-path stable DSGE models, and we can estimate only the parameters of models we can solve.

The parameter values determine whether a DSGE model is saddle-path stable. Most DSGE models are saddle-path stable for some parameter values and not for other values. As we discussed in Structural and reduced forms of DSGE models in [DSGE] Intro 1, the structural form of a linear DSGE model is

\begin{align}
A_0 y_t &= A_1 E_t(y_{t+1}) + A_2 y_t + A_3 x_t \\
B_0 x_{t+1} &= B_1 E_t(y_{t+1}) + B_2 y_t + B_3 x_t + C \epsilon_{t+1}
\end{align}

Given this structural form, the values in the $A_i$ and $B_j$ matrices determine saddle-path stability. For nonlinear DSGE models, saddle-path stability is assessed on the linear approximation to the nonlinear model at the steady state.

In the process of solving the structural form in (1) and (2) for the state space, the Klein (2000) solver computes the generalized eigenvalues of a matrix formed from the values in the $A_i$ and $B_j$ matrices. An eigenvalue is said to be stable when its absolute value is less than 1. The model is saddle-path stable when the number of stable eigenvalues equals the number of states in the model.

Once the maximization algorithm gets started, saddle-path stability usually is not an issue. Problems with saddle-path stability usually occur when trying to get the maximization process started. The initial values for the maximization algorithm must imply saddle-path stability because we cannot solve a DSGE model for parameter values that do not imply saddle-path stability.

dsge and dsgenl use a series of small positive numbers as the default initial values for the structural parameters that appear in the $A$ and $B$ matrices. If you know that your model is not saddle-path stable when a parameter, say, $\beta$, is set to 0.1, you should use the from() option to specify an initial value that does imply saddle-path stability. Alternatively, you could reparameterize the model so that $\beta = 0.1$ does imply saddle-path stability, but this solution tends to be more work than changing the initial value.
In rare cases, when the true parameters are close to values that are not saddle-path stable, the maximum likelihood estimator will suffer from convergence problems. Better initial values can help find the solution in these cases.

**What if the initial values are not saddle-path stable?**

If the initial values do not imply a saddle-path stable solution, you will see

```
. dsge ...
  model is not saddle-path stable at current parameter values
  The number of stable eigenvalues is greater than the number of state variables.
  (output omitted)
```

or

```
. dsge ...
  model is not saddle-path stable at current parameter values
  The number of stable eigenvalues is less than the number of state variables.
  (output omitted)
```

In either case, you should specify the `solve` and `noidencheck` options and subsequently use `estat stable` to find the problem. `solve` instructs `dsge` and `dsgenl` to only solve the model and not proceed with estimation, and `noidencheck` suppresses the check of whether the parameters are identified. `estat stable` will display the eigenvalues so that you can determine which ones are saddle-path stable and which ones are not. The trick is to change the starting values so that the number of stable eigenvalues equals the number of states in the model. Once you have determined which parameters are problematic, you can use the `from()` option to change the initial values to ones that do not have this problem.

Textbooks like those by Canova (2007), DeJong and Dave (2011), Ljungqvist and Sargent (2018), and Woodford (2003) are full of examples in which some structural parameters must be either greater than 1 or less than 1 for the model to be saddle-path stable. Similarly, some parameters must be either greater than or less than 0 for the model to be saddle-path stable. Moving one structural parameter over one of these boundaries and looking at how the change affects the number of stable eigenvalues can help find a vector of parameters that yields a saddle-path stable solution.

That the model that you are trying to fit is likely related to other models in the literature or in the textbooks suggests another strategy. Any such references probably contain some discussion of the set of values that yield saddle-path stable solutions.

In especially difficult cases, you can use the strategy of solving the problem for a smaller model and building back up. You temporarily remove equations from the model, find a parameter vector that yields a saddle-path stable solution for the smaller model, and then progressively add back equations as you find initial values that yield saddle-path stable solutions.
Example 1: Finding saddle-path stable initial values

Equations (1)–(6) model the observed control variable (inflation) $p_t$, the unobserved control variable (output gap) $y_t$, and the observed control variable (interest rate) $r_t$ as functions of the states $u_t$ and $z_t$. $Lz_{t+1}$ is an auxiliary state that allows $z_t$ to be a second-order process instead of a first-order process.

\[ p_t = \frac{1}{\gamma} E_t(p_{t+1}) + \kappa y_t \]  
\[ y_t = E_t(y_{t+1}) - \{r_t - E_t(p_{t+1}) - z_t\} \]  
\[ r_t = \gamma p_t + u_t \]  
\[ u_{t+1} = \rho_u u_t + \xi_{t+1} \]  
\[ z_{t+1} = \rho_z z_t + \rho_z^2 Lz_t + \epsilon_{t+1} \]  
\[ Lz_{t+1} = z_t \]

We try to fit this model using default starting values below.

```
use https://www.stata-press.com/data/r16/usmacro2
(dsge (p = (1/{gamma})*F.p + {kappa}*y)
> (y = F.y - (r - F.p - z), unobserved)
> (r = {gamma}*p + u)
> (F.u = {rho_u}*u, state)
> (F.z = {rho_z1}*z + {rho_z2}*Lz, state)
> (F.Lz = z, state noshock)
model is not saddle-path stable at current parameter values
The number of stable eigenvalues is greater than the number of state variables.
r(498);
```

The default initial values specify a model that is not saddle-path stable. We specify options `solve` and `noidencheck`, which cause the command to attempt only to solve the model and to skip the identification check.
dsge (p = (1/\{gamma\})*F.p + \{kappa\}*y)
> (y = F.y - (r - F.p - z), unobserved)
> (r = \{gamma\}*p + u)
> (F.u = \{rho_u\}*u, state)
> (F.z = \{rho_z1\}*z + \{rho_z2\}*Lz, state)
> (F.Lz = z, state noshock),
> solve noidencheck

DSGE model
Sample: 1955q1 - 2015q4 Number of obs = 244
Log likelihood = .

| OIM                     | Coef. | Std. Err. | z   | P>|z| | [95% Conf. Interval] |
|-------------------------|-------|-----------|-----|------|----------------------|
| /structural             |       |           |     |      |                      |
| gamma                   | .21   | .         | .   | .    | .                    |
| kappa                   | .22   | .         | .   | .    | .                    |
| rho_u                   | .23   | .         | .   | .    | .                    |
| rho_z1                  | .24   | .         | .   | .    | .                    |
| rho_z2                  | .25   | .         | .   | .    | .                    |
| sd(e.u)                 | 1     | .         | .   | .    | .                    |
| sd(e.z)                 | 1     | .         | .   | .    | .                    |

Note: Skipped identification check.
Warning: Model cannot be solved at current parameter values. Current values imply a model that is not saddle-path stable.

dsgne displays the initial values used in the attempted solution, and it displays a warning message indicating that the model cannot be solved at these values because they imply a model that is not saddle-path stable.

We use estat stable to look at the eigenvalues implied by the initial values.

. estat stable

Stability results

<table>
<thead>
<tr>
<th>Eigenvalues</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>stable</td>
<td>-.3942</td>
</tr>
<tr>
<td>stable</td>
<td>.6342</td>
</tr>
<tr>
<td>stable</td>
<td>.21</td>
</tr>
<tr>
<td>stable</td>
<td>.23</td>
</tr>
<tr>
<td>unstable</td>
<td>2.605e+16</td>
</tr>
<tr>
<td>unstable</td>
<td>1.046</td>
</tr>
</tbody>
</table>

The process is not saddle-path stable.
The process is saddle-path stable when there are 3 stable eigenvalues and 3 unstable eigenvalues.

The model is saddle-path stable when the number of stable eigenvalues equals the number of states in the model. The initial values yield four stable eigenvalues, but there are only three states in the model. So the model is not saddle-path stable at these values.

The initial value for gamma in the output from the failed solution attempt was 0.21. From (3), we see that the coefficient on expected future inflation is greater than one when gamma is less than one. We suspect that this relationship could be causing the lack of saddle-path stability. Below, we use option from() to specify 1.2 as an initial value for gamma and attempt to solve the model.
dsge \( p = \left(1/\gamma\right)F.p + \kappa y \)
\( y = F.y - (r - F.p - z), \text{unobserved} \)
\( r = \gamma p + u \)
\( F.u = \rho_u u, \text{state} \)
\( F.z = \rho_{z1} z + \rho_{z2} Lz, \text{state} \)
\( F.Lz = z, \text{state noshock}, \)
solve noidencheck from(\gamma=1.2)

DSGE model
Sample: 1955q1 - 2015q4
Number of obs = 244
Log likelihood = -1504.7564

| OIM | Coef. | Std. Err. | z  | P>|z| | [95% Conf. Interval] |
|-----|-------|-----------|----|------|-----------------------|
| /structural | gamma | 1.2 | . | . | . |
| | kappa | .22 | . | . | . |
| | rho_u | .23 | . | . | . |
| | rho_z1 | .24 | . | . | . |
| | rho_z2 | .25 | . | . | . |
| sd(e.u) | 1 | . | . | . |
| sd(e.z) | 1 | . | . | . |

Note: Skipped identification check.
Note: Model solved at specified parameters.

There is no warning message, so the parameter values yielded a saddle-path stable solution. As a final illustration of this point, we display the eigenvalues below.

. estat stable

Stability results

<table>
<thead>
<tr>
<th>Eigenvalues</th>
</tr>
</thead>
<tbody>
<tr>
<td>stable</td>
</tr>
<tr>
<td>stable</td>
</tr>
<tr>
<td>stable</td>
</tr>
<tr>
<td>unstable</td>
</tr>
<tr>
<td>unstable</td>
</tr>
<tr>
<td>unstable</td>
</tr>
</tbody>
</table>

The process is saddle-path stable.

We could now proceed with estimation.

References


Also see

[DSGE] Intro 6 — Identification
[DSGE] Intro 7 — Convergence problems
[DSGE] estat stable — Check stability of system
We can estimate only the parameters of a DSGE model when the parameters are identified. We discuss how to find the identification problems and how to fix them.

The parameters that maximize the likelihood function are the maximum likelihood (ML) estimates of ML estimators like the one in dsge. When there is a unique vector of parameters that maximizes the likelihood function, the parameters are identified and we can estimate them. When there is more than one parameter vector that produces the same maximum value of the likelihood function, the parameters are not identified, and we cannot estimate them because there is no way to choose among the sets of equally best parameters.

dsge, dsge1, and all other commands that implement ML estimators check for identification at the solution by checking that the matrix of second derivatives, known as the Hessian, is full rank at the maximum. This test works only when the maximization algorithm arrives at a candidate maximum. In addition, it provides no information about which parameters are not identified.

For many models, when the parameters are not identified, the only output produced by a standard ML estimator is an ever growing series of “not concave” messages. This issue is especially relevant to DSGE models because it is easy to write down theoretical DSGE models whose parameters are not identified. dsge uses the Iskrev (2010) diagnostic to resolve this issue. The Iskrev diagnostic detects lack of parameter identification and reports which parameters are not identified at specific parameter values. By default, dsge performs the Iskrev diagnostic at the initial values and at the solution values. dsge1 performs the Iskrev diagnostic on the linear approximation to the nonlinear model at the steady state implied by the initial values and checks again at the solution values.

Instead of looking at the rank of the Hessian, the Iskrev (2010) diagnostic checks that all the parameters affect the autocovariances of the observed control variables. When there is a one-to-one relationship between the autocovariances for the observed control variables and the model parameters, the parameters are identified. Intuitively, this one-to-one relationship allows us to back out the model parameters from the autocovariances that we can always estimate from the data. When some of the parameters do not affect the autocovariances of the observed control variables, there is no one-to-one relationship, and the model parameters are not identified.

Simply adding restrictions to parameters can make the parameters of an unidentified model identified. This fact is frustrating but intuitive. When the parameters are not identified, they are not uniquely determined by the data generated by the true model. Only adding restrictions, either by defining constraints or by changing the equations, can identify the parameters.
Example 1: A model with two unidentified parameters

Consider the following model of inflation $p_t$, output growth $y_t$, and the interest rate $r_t$.

$$p_t = \beta E_t(p_{t+1}) + \kappa y_t$$
$$y_t = E_t(y_{t+1}) - \gamma \{r_t - E_t(p_{t+1}) - \rho z_t\}$$
$$\beta r_t = p_t + \beta u_t$$
$$z_{t+1} = \rho z_t + \epsilon_{t+1}$$
$$u_{t+1} = \delta u_t + \xi_{t+1}$$

This model specifies how the observed control variables $p_t$ and $r_t$ depend on the unobserved control variable $y_t$, on the state variables $z_t$ and $u_t$, and on the shocks $\epsilon_t$ and $\xi_t$.

Let's try to estimate the parameters of this model.

```
. use https://www.stata-press.com/data/r16/usmacro2
   (Federal Reserve Economic Data - St. Louis Fed, 2017-01-15)
. dsge (p = {beta}*F.p + {kappa}*y)
     (y = F.y -{gamma}*(r - F.p - {rhoz}*z), unobserved)
     (r = (1/{beta})*p + u)
     (F.u = {rhou}*u, state)
     (F.z = {rhoz}*z, state)
identification failure at starting values
   Constrain some parameters or specify option noidencheck. Likely source of
   identification failure: {kappa} {gamma}
r(498);
```

The error message indicates that $\kappa$ and $\gamma$ do not influence the autocovariances independently; they are linearly dependent. If a list of linearly dependent parameters is supplied, then you must constrain all but one of them prior to estimation. We constrain $\gamma$.

```
. constraint 2 _b[gamma]=1
And we apply the constraint to the model:
```

```
. dsge (p = {beta}*F.p + {kappa}*y)
     (y = F.y -{gamma}*(r - F.p - {rhoz}*z), unobserved)
     (r = (1/{beta})*p + u)
     (F.u = {rhou}*u, state)
     (F.z = {rhoz}*z, state),
     constraint(2)
(setting technique to bfgs)
Iteration 0:  log likelihood =  -128443.1
Iteration 1:  log likelihood =  -3610.8865  (backed up)
Iteration 2:  log likelihood =  -1066.3053  (backed up)
Iteration 3:  log likelihood =  -919.1875  (backed up)
Iteration 4:  log likelihood =  -815.49521  (backed up)
(switching technique to nr)
Iteration 5:  log likelihood =  -805.28492  (backed up)
Iteration 6:  log likelihood =  -790.04311
Iteration 7:  log likelihood =  -773.86016
Iteration 8:  log likelihood =  -755.6535
Iteration 9:  log likelihood =  -753.96757
Iteration 10: log likelihood =  -753.57754
Iteration 11: log likelihood =  -753.57134
Iteration 12: log likelihood =  -753.57131
```

```
DSGE model
Sample: 1955q1 - 2015q4
Number of obs = 244
Log likelihood = -753.57131
( 1) [structural]gamma = 1

<table>
<thead>
<tr>
<th></th>
<th>OIM</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Coef.</td>
<td>Std. Err.</td>
<td>z</td>
<td>P&gt;</td>
</tr>
<tr>
<td>/structural</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>beta</td>
<td>.5146687</td>
<td>.078349</td>
<td>6.57</td>
<td>0.000</td>
</tr>
<tr>
<td>kappa</td>
<td>.1659038</td>
<td>.0474064</td>
<td>3.50</td>
<td>0.000</td>
</tr>
<tr>
<td>gamma</td>
<td>1 (constrained)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>rhoz</td>
<td>.9545255</td>
<td>.0186424</td>
<td>51.20</td>
<td>0.000</td>
</tr>
<tr>
<td>rhou</td>
<td>.7005496</td>
<td>.0452601</td>
<td>15.48</td>
<td>0.000</td>
</tr>
<tr>
<td>sd(e.u)</td>
<td>2.318197</td>
<td>.3047421</td>
<td></td>
<td></td>
</tr>
<tr>
<td>sd(e.z)</td>
<td>.650711</td>
<td>.1123848</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The remaining free parameters of the model are identified and can be estimated.

Keep in mind that the Iskrev (2010) diagnostic is performed at specific parameter values. Sometimes, the parameters of a model cannot be identified at a vector of parameter values but can be identified at practically any other nearby vector. This problem arises when simple initial values cause terms to cancel out. If you suspect that the identification diagnostic is finding a problem because of initial values, you can either specify noidencheck or specify other seemingly random initial values.

> Example 2: Suppressing the identification check

We suppress the identification check in an attempt to estimate the parameters without imposing additional constraints. We specify the iterate(50) option in case the parameters are not identified.

```
dsge (p = {beta}*F.p + {kappa}*y)
>   (y = F.y -{gamma}*(r - F.p - {rhoz}*z), unobserved)
>   (r = (1/{beta})*p + u)
>   (F.u = {rhou}*u, state)
>   (F.z = {rhoz}*z, state),
>   noidencheck iterate(50)
```

(setting technique to bfgs)
Iteration 0: log likelihood = -1597221.6
Iteration 1: log likelihood = -5097.544 (backed up)
Iteration 2: log likelihood = -1073.3293 (backed up)
Iteration 3: log likelihood = -978.36896 (backed up)
Iteration 4: log likelihood = -818.00268 (backed up)
(switching technique to nr)
Iteration 5: log likelihood = -808.99197 (not concave)
Iteration 6: log likelihood = -778.94324 (not concave)
Iteration 7: log likelihood = -761.39371 (not concave)
Iteration 8: log likelihood = -755.33513 (not concave)
Iteration 9: log likelihood = -754.55068
Iteration 10: log likelihood = -753.60217 (not concave)
Iteration 11: log likelihood = -753.5846 (not concave)
Iteration 12: log likelihood = -753.58018 (not concave)
Iteration 13: log likelihood = -753.57404 (not concave)
Iteration 14: log likelihood = -753.57267 (not concave)
Iteration 15: log likelihood = -753.5721
Iteration 16: log likelihood = -753.57131
Iteration 17: log likelihood = -753.57131 (not concave)
Iteration 18: log likelihood = -753.57131 (not concave)
We see a long list of not concave messages, indicating that the parameters are indeed not identified without additional constraints. We could remove the option limiting the number of iterations, and the command should produce not concave iterations indefinitely.
Reference


Also see

[DSGE] Intro 5 — Stability conditions
[DSGE] Intro 7 — Convergence problems
Fitting DSGE models is notoriously difficult. We discuss the causes of convergence problems and some solutions.

When the iteration process never gets started or when it never ends are two types of convergence problems. When it never gets started, you will see something like

```
 dsge ...
 model could not be solved at starting values
   equilibrium is locally indeterminate
 (output omitted)
```

This error occurs when the initial values specify a model that violates the stability conditions required for DSGE estimation; see [DSGE] Intro 5 for a discussion of this issue and some solutions.

When the iteration process never ends, the cause is either that there is no unique solution or that the optimizer cannot find the unique solution. When there is no unique solution, the parameters are said to be not identified. In this case, you see \texttt{(not concave)} after each iteration, like

```
 dsge ...
 (output omitted)
 Iteration 50: log likelihood = -337504.44 (not concave)
 Iteration 51: log likelihood = -337503.52 (not concave)
 Iteration 52: log likelihood = -337502.13 (not concave)
```

See [DSGE] Intro 6 for a discussion of this issue and some solutions.

Even when the iterations get started and the parameters are identified, convergence problems are still common. In these cases, there are usually many parameters.

Changing the optimization technique is the simplest method to overcome a convergence problem, and it can be surprisingly effective. By default, \texttt{dsge} and \texttt{dsgenl} use the Broyden–Fletcher–Goldfarb–Shanno (BFGS) algorithm for five iterations before switching to a modified Newton–Raphson (NR) algorithm. The BFGS algorithm is especially effective at finding a maximum of the DSGE log-likelihood function from poor initial values. In some cases, simply specifying \texttt{technique(bfgs 200 nr)} can overcome the convergence problem.

The best way to overcome a convergence problem is to provide better initial values. The optimizer used by \texttt{dsge} and \texttt{dsgenl} will almost always converge when it is provided with good initial values and the parameters are identified. Specifying good initial values that come from theory or previous empirical work can solve your convergence problem. You can specify initial values inside the scalar substitutable expressions or by using the \texttt{from()} option; see examples 1 and 2 for details.
Sometimes, you have a convergence problem and do not have good initial values. One way to get initial values is to estimate a series of progressively less constrained models. This process usually produces convergence when the parameters are identified. See example 3 for details.

Example 1: Specifying starting values in scalar substitutable expressions

Models with lagged state variables and other models that allow for more persistent processes can exhibit convergence problems. Equations (1)–(6) model the observed control variable (inflation) $p_t$, the unobserved control variable (output gap) $y_t$, and the observed control variable (interest rate) $r_t$ as functions of the states $u_t$ and $z_t$. $Lz_{t+1}$ is an auxiliary state that allows $z_t$ to be a second-order process instead of a first-order process.

\[
\begin{align*}
    p_t &= \beta E_t(p_{t+1}) + \kappa y_t \\
    y_t &= E_t(y_{t+1}) - \{r_t - E_t(p_{t+1}) - z_t\} \\
    r_t &= \frac{1}{\beta} p_t + u_t \\
    u_{t+1} &= \rho_u u_t + \xi_{t+1} \\
    z_{t+1} &= \rho_{z1} z_t + \rho_{z2} Lz_t + \epsilon_{t+1} \\
    Lz_{t+1} &= z_t
\end{align*}
\]

Suppose that previous empirical work suggests the initial values of $\beta = 0.5$, $\kappa = 0.2$, $\rho_u = 0.7$, $\rho_{z1} = 0.7$, and $\rho_{z2} = 0.2$. Inside a scalar substitutable expression, we can specify an initial value by typing \{parameter = value\}. We specify these initial values for the parameters in the scalar substitutable expressions in our model specification below.

```
. use https://www.stata-press.com/data/r16/usmacro2
   (Federal Reserve Economic Data - St. Louis Fed, 2017-01-15)
. dsge (p = {beta=.5}*F.p + {kappa=.2}*y)
   > (y = F.y - (r - F.p - z), unobserved)
   > (r = (1/{beta})*p + u)
   > (F.u = {rho_u=.7}*u, state)
   > (F.z = {rho_z1=.7}*z + {rho_z2=.2}*Lz, state)
   > (F.Lz = z, state noshock)
   (setting technique to bfgs)
Iteration 0:  log likelihood =  -1117.6457
Iteration 1:  log likelihood =  -936.74558  (backed up)
Iteration 2:  log likelihood =  -846.69924  (backed up)
Iteration 3:  log likelihood =  -835.21274  (backed up)
Iteration 4:  log likelihood =  -782.84071  (backed up)
   (switching technique to nr)
Iteration 5:  log likelihood =  -772.42535  (not concave)
Iteration 6:  log likelihood =  -754.3824
Iteration 7:  log likelihood =  -753.14343
Iteration 8:  log likelihood =  -753.07802
Iteration 9:  log likelihood =  -753.07788
Iteration 10: log likelihood =  -753.07788
```
Note that the output includes results for the standard deviations of the shocks, for which there are no scalar substitutable expressions. In example 2, we illustrate how to specify initial values for all the parameters using the from() option.

Example 2: Specifying starting values using from()

In example 1, we specified initial values inside the scalar substitutable expressions. You cannot specify initial values for the standard deviations of the shocks this way, because the shocks do not appear in any scalar substitutable expressions. We can specify starting values for any, or all, of the parameters using the from() option.

From() has several syntaxes, but the vector of starting values is most frequently used in DSGE applications. In this syntax, we specify a row vector containing the initial values $\beta = 0.5$, $\kappa = 0.2$, $\rho_u = 0.7$, $\rho_{z1} = 0.7$, $\rho_{z2} = 0.2$, $\sigma_{e,u} = 2.3$, and $\sigma_{e,z} = 0.7$. We begin by defining and listing the vector of initial values.

```
matrix ivalues = (.5, .2, .7, .7, .2, 2.3, .7)
matrix list ivalues
ivalues[1,7]
c1 c2 c3 c4 c5 c6 c7
r1 .5 .2 .7 .7 .2 2.3 .7
```

Although we could specify and use the column names of the vector to assign elements of the vector to parameters, we use from's suboption copy to make the assignment based on order of appearance in the vector. Specifying from(), copy) implies that the first element in the vector specifies the initial value for the first model parameter, the second element in the vector specifies the initial value for the second model parameter, and so on.
. dsge (p = \{beta\}*F.p + \{kappa\}*y)
> (y = F.y - (r - F.p - z), unobserved)
> (r = (1/\{beta\})*p + u)
> (F.u = \{rho_u\}*u, state)
> (F.z = \{rho_z1\}*z + \{rho_z2\}*Lz, state)
> (F.Lz = z, state noshock), from(ivalues, copy)
(setting technique to bfgs)

Iteration 0: log likelihood = -769.09431
Iteration 1: log likelihood = -754.09564 (backed up)
Iteration 2: log likelihood = -753.70352 (backed up)
Iteration 3: log likelihood = -753.62091 (backed up)
Iteration 4: log likelihood = -753.53446 (backed up)
(switching technique to nr)
Iteration 5: log likelihood = -753.52617 (backed up)
Iteration 6: log likelihood = -753.09207
Iteration 7: log likelihood = -753.07789
Iteration 8: log likelihood = -753.07788

DSGE model
Sample: 1955q1 - 2015q4 Number of obs = 244
Log likelihood = -753.07788

| OIM          | Coef. | Std. Err. | z    | P>|z| | [95% Conf. Interval] |
|--------------|-------|-----------|------|------|----------------------|
| /structural  |       |           |      |      |                      |
| beta         | .5154871 | .0770492  | 6.69 | 0.000 | .3644734 - .6665008  |
| kappa        | .1662427 | .0468473  | 3.55 | 0.000 | .0744237 - .2580616  |
| rho_u        | .6979877 | .0452072  | 15.44| 0.000 | .6093833 - .7865921  |
| rho_z1       | .6735477 | .2666002  | 2.53 | 0.012 | .1510209 - 1.196074  |
| rho_z2       | .2710007 | .2564571  | 1.06 | 0.291 | -.2316459 - .7736473 |
| sd(e.u)      | 2.315264 | .2992175  |      |      | 1.728808 - 2.901719  |
| sd(e.z)      | .7720676 | .1716211  |      |      | .4356965 - 1.108439  |

Example 3: Starting with a more constrained model to obtain convergence

Suppose that we wanted to estimate the parameters of the model in example 1 but that the only good initial value was that theory predicted that $\beta$ should be about 0.5. Further suppose that we were having convergence problems fitting this model. In this case, one strategy would be to constrain $\beta$ to 0.5 and to constrain $\kappa$ to a small, nonzero value. The small value minimizes the interaction between $p$ and $y$. Making the value nonzero can prevent problems when solving for the state-space form of the model. The remaining parameters determine the autoregressive processes for the states. Given that we suspect that the second-order lag is part of the problem, we also constrain it to a small, nonzero value. We try to estimate the remaining parameters in the output below.
constraint define 1 _b[\text{beta}] = 0.5
constraint define 2 _b[kappa] = 0.1
constraint define 3 _b[rho_z2] = 0.01
dsge (p = \{\text{beta}\}*F.p + \{\text{kappa}\}*y)
> (y = F.y - (r - F.p - z), unobserved)
> (r = (1/\{\text{beta}\})*p + u)
> (F.u = \{\text{rho_u}\}*u, state)
> (F.z = \{\text{rho_z1}\}*z + \{\text{rho_z2}\}*Lz, state)
> (F.Lz = z, state noshock), constraints(1 2 3)
(setting technique to bfgs)
Iteration 0: log likelihood = -22446.965
Iteration 1: log likelihood = -2153.9862 (backed up)
Iteration 2: log likelihood = -1257.6437 (backed up)
Iteration 3: log likelihood = -1120.3354 (backed up)
Iteration 4: log likelihood = -1098.8062 (backed up)
(switching technique to nr)
Iteration 5: log likelihood = -1092.0434 (not concave)
Iteration 6: log likelihood = -865.7853 (not concave)
Iteration 7: log likelihood = -781.8036
Iteration 8: log likelihood = -756.40072
Iteration 9: log likelihood = -755.63161
Iteration 10: log likelihood = -755.63025
Iteration 11: log likelihood = -755.63025

DSGE model
Sample: 1955q1 - 2015q4
Log likelihood = -755.63025
( 1) /[\text{structural}]beta = .5
( 2) /[\text{structural}]kappa = .1
( 3) /[\text{structural}]rho_z2 = .01

| Coef. | Std. Err. | z     | P>|z| | [95% Conf. Interval] |
|-------|-----------|-------|-------|-----------------------|
| beta  | .5 (constrained) |
| kappa | .1 (constrained)  |
| rho_u | .7906612 .0102382 77.23 0.000 .7705947 .8107277 |
| rho_z1| .9440872 .0188166 50.17 0.000 .9072073 .980967 |
| rho_z2| .01 (constrained) |
| sd(e.u)| 2.391372 .1083319 2.179045 2.603699 |
| sd(e.z)| .697013 .0731968 .5535498 .8404761 |

The estimator converged. We will want to use these estimates as initial values, so we put them into the matrix b and list b to confirm it has the desired values.

```
.matrix b = e(b)
.matrix list b
```

b[1,7]

| Coef. | Std. Err. | z     | P>|z| | [95% Conf. Interval] |
|-------|-----------|-------|-------|-----------------------|
| beta  | .5        |
| kappa | .1        |
| rho_u | .7906617  |
| rho_z1| .94408715 |
| rho_z2| .01       |
| sd(e.u)| 2.3913719 |
| sd(e.z)| .69701298 |
```
We use the previous estimates as initial values, drop the constraints, and try to estimate all the parameters.

```
.dse (p = {beta}*F.p + {kappa}*y)
> (y = F.y - (r - F.p - z), unobserved)
> (r = (1/{beta})*p + u)
> (F.u = {rho_u}*u, state)
> (F.z = {rho_z1}*z + {rho_z2}*Lz, state)
> (F.Lz = z, state noshock), from(b, copy)
(setting technique to bfgs)
Iteration 0: log likelihood = -755.63025
Iteration 1: log likelihood = -755.57974 (backed up)
Iteration 2: log likelihood = -755.1865 (backed up)
Iteration 3: log likelihood = -754.6238 (backed up)
(moving technique to nr)
Iteration 4: log likelihood = -754.55289 (backed up)
Iteration 5: log likelihood = -754.45439 (backed up)
Iteration 6: log likelihood = -753.14544
Iteration 7: log likelihood = -753.07913
Iteration 8: log likelihood = -753.07788
Iteration 9: log likelihood = -753.07788
DSGE model
Sample: 1955q1 - 2015q4 Number of obs = 244
Log likelihood = -753.07788

<table>
<thead>
<tr>
<th></th>
<th>OIM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Coef.</td>
</tr>
<tr>
<td>/structural</td>
<td></td>
</tr>
<tr>
<td>beta</td>
<td>.5154872</td>
</tr>
<tr>
<td>kappa</td>
<td>.1662428</td>
</tr>
<tr>
<td>rho_u</td>
<td>.6979875</td>
</tr>
<tr>
<td>rho_z1</td>
<td>.6735459</td>
</tr>
<tr>
<td>rho_z2</td>
<td>.2710024</td>
</tr>
<tr>
<td>sd(e.u)</td>
<td>2.315263</td>
</tr>
<tr>
<td>sd(e.z)</td>
<td>.7720683</td>
</tr>
</tbody>
</table>
```

The estimator converged.

In larger problems, more steps are sometimes required. In these steps, you use the previous estimates as initial values, but you drop only some of the constraints at each step.

**Also see**

[DSE] Intro 5 — Stability conditions

[DSE] Intro 6 — Identification
After fitting a DSGE model, we often perform tests of structural parameters, and these tests often place nonlinear restrictions on the parameters. The values and rejection rates of a Wald test for different nonlinear expressions of the same null hypothesis are different. We illustrate this issue, show that likelihood-ratio (LR) tests do not have this problem, and illustrate that you can parameterize your model in terms of invertible transforms of each parameter.

Remarks and examples

Remarks are presented under the following headings:

- Wald tests vary with nonlinear transforms
- LR tests do not vary with nonlinear transforms

Wald tests vary with nonlinear transforms

Performing a statistical test of whether a structural parameter in a DSGE has a specific value is one of the most frequent forms of inference after dsge and dsgenl estimation. The null hypothesis in one of these tests frequently places nonlinear restrictions on the underlying parameters. Two different nonlinear expressions of the same null hypothesis produce different Wald test statistics in finite samples and have different rejection rates. In other words, the Wald test is not invariant to nonlinear transforms of the null hypothesis. The LR test, on the other hand, is invariant to nonlinear transforms of the null hypothesis.

Example 1: Different values from logically equivalent Wald tests

Equations (1)–(5) specify how the observed control variable inflation $p_t$, the unobserved control variable output growth $y_t$, and the observed control variable (interest rate) $r_t$ depend on the states $z_t$ and $u_t$, given the shocks $\epsilon_t$ and $\xi_t$.

\begin{align*}
    p_t &= \beta E_t(p_{t+1}) + \kappa y_t \\
    y_t &= E_t(y_{t+1}) - \left\{ r_t - E_t(p_{t+1}) - \rho z_t \right\} \\
    r_t &= (1/\beta)p_t + u_t \\
    z_{t+1} &= \rho z_t + \epsilon_{t+1} \\
    u_{t+1} &= \delta u_t + \xi_{t+1}
\end{align*}
We estimate the parameters of this model using the macroeconomic data for the United States in usmacro2.dta.

```
. use https://www.stata-press.com/data/r16/usmacro2
(Federal Reserve Economic Data - St. Louis Fed, 2017-01-15)
. dse (p = {beta}*F.p + {kappa}*y)
> (y = F.y - (r - F.p - {rhoz}*z), unobserved)
> (r = (1/{beta})*p + u)
> (F.u = {rhou}*u, state)
> (F.z = {rhoz}*z, state)
(setting technique to bfgs)
Iteration 0: log likelihood = -146218.64
Iteration 1: log likelihood = -5532.4212 (backed up)
Iteration 2: log likelihood = -1067.4665 (backed up)
Iteration 3: log likelihood = -938.92415 (backed up)
Iteration 4: log likelihood = -885.96401 (backed up)
(setting technique to nr)
Iteration 5: log likelihood = -880.81744 (not concave)
Iteration 6: log likelihood = -819.11157 (not concave)
Iteration 7: log likelihood = -775.50717 (not concave)
Iteration 8: log likelihood = -766.59529
Iteration 9: log likelihood = -756.78683 (not concave)
Iteration 10: log likelihood = -754.40674
Iteration 11: log likelihood = -753.85297
Iteration 12: log likelihood = -753.57309
Iteration 13: log likelihood = -753.57131
Iteration 14: log likelihood = -753.57131
DSGE model
Sample: 1955q1 - 2015q4 Number of obs  = 244
Log likelihood = -753.57131

                       | OIM
structural            | Coef.  Std. Err.     z  P>|z|    [95% Conf. Interval]
-----------------------+---------------------------------+-------------------+-------------------+-------------------+
/structural            | beta    0.5146674  0.0783489   6.57  0.000    0.3611064    0.6682284
                     | kappa   0.1659057  0.0474074   3.50  0.000    0.0729889    0.2588224
                     | rhoz    0.9545256  0.0186424  51.20  0.000    0.9179872    0.991064
                     | rhou    0.7005483  0.0452604  15.48  0.000    0.618395    0.7892571
sd(e.u)               |        2.318203   0.3047438  1.72  0.091    1.672444     2.963962
sd(e.z)               |        0.6507118  0.1123845  5.79  0.000    0.4304422    0.8709814

The interest rate equation shown in (3) links the nominal interest rate to the inflation rate. The coefficient on inflation is $1/\beta$. We test whether this parameter is 1.5, a common benchmark value in the literature.

```
. testnl 1/_b[beta] = 1.5
(1) 1/_b[beta] = 1.5
    ch2(1) = 2.24
    Prob > chi2 = 0.1342
```

We do not reject the null hypothesis that $1/\beta$ is 1.5.
If we test the logically equivalent hypothesis that $\beta = 2/3$, the statistic and $p$-value change.

```
.test _b[beta] =2/3
   ( 1) `/structural'beta = .6666667
      chi2( 1) = 3.76
      Prob > chi2 = 0.0524
```

The values of these two logically equivalent Wald tests differ because Wald tests are not invariant to nonlinear transformation. This issue is well known in the literature; see Gregory and Veall (1985) and Phillips and Park (1988) for details. In this example, the inference of failing to reject the null hypothesis remains the same when using a 5% significance level, but this is not true in general. Different formulations of Wald tests can lead to different inferences.

### LR tests do not vary with nonlinear transforms

#### Example 2: LR tests are invariant to nonlinear transforms

We illustrate this feature by performing LR tests that $\beta = 2/3$ and that $1/\beta = 1.5$. The current estimates are those of the unconstrained model. We repeat these results and store them as unconstrained.

```
.dse
DSGE model
Sample: 1955q1 - 2015q4
Log likelihood = -753.57131

--------- OIM
         Coef.    Std. Err.     z    P>|z|    [95% Conf. Interval]
---------
/structural
  beta    .5146674    .0783489    6.57    0.000    .3611064    .6682284
  kappa   .1659057    .0474074    3.50    0.000    .0729889    .2588224
  rhoz    .9545256    .0186424   51.20    0.000    .9179872    .991064
  rhou    .7005483    .0452604   15.48    0.000    .618395    .7892571
  sd(e.u) 2.318203    .3047438   1.720916  2.915489
  sd(e.z) .6507118    .1123845   .4304422   .8709814
---------
.estimates store unconstrained
```
Now, we estimate the parameters of the constrained model in which $\beta = 2/3$, store the results as constrained, and perform an LR test of the null hypothesis that $\beta = 2/3$.

```
. constraint 1 _b[beta] = 2/3
. dsge (p = {beta}*F.p + {kappa}*y)
> (y = F.y - (r - F.p - {rhoz}*z), unobserved)
> (r = (1/{beta})*p + u)
> (F.u = {rhou}*u, state)
> (F.z = {rhoz}*z, state),
> constraint(1)
```

(setting technique to bfgs)

Iteration 0: log likelihood = -119695.1
Iteration 1: log likelihood = -1425.592 (backed up)
Iteration 2: log likelihood = -984.57609 (backed up)
Iteration 3: log likelihood = -948.41524 (backed up)
Iteration 4: log likelihood = -945.83724 (backed up)

(switching technique to nr)

Iteration 5: log likelihood = -945.06881 (backed up)
Iteration 6: log likelihood = -764.18274 (not concave)
Iteration 7: log likelihood = -756.7209
Iteration 8: log likelihood = -755.12606
Iteration 9: log likelihood = -755.11016
Iteration 10: log likelihood = -755.11003
Iteration 11: log likelihood = -755.11003

DSGE model
Sample: 1955q1 - 2015q4 Number of obs = 244
Log likelihood = -755.11003

( 1) [/structural]beta = .6666667

```
. estimates store constrained
. lrtest unconstrained constrained
```

Likelihood-ratio test LR chi2(1) = 3.08
(Assumption: constrained nested in unconstrained) Prob > chi2 = 0.0794

Note that the value of the LR statistic is 3.08. We now illustrate an LR of the null hypothesis that $1/\beta = 1.5$ produces the same value.

We cannot impose nonlinear restrictions on parameters, so we must begin by reparameterizing the unconstrained model by replacing $\{\beta\}$ with $1/\{\beta\}$. To avoid having $\{\beta\}$ mean two different things, we write the model in terms of $\{\gamma\} = 1/\{\beta\}$ and estimate the parameters:
dsge (p = 1/{gamma}*F.p + {kappa}*y)
> (y = F.y - (r - F.p - {rhoz}*z), unobserved)
> (r = ({gamma})*p + u)
> (F.u = {rhou}*u, state)
> (F.z = {rhoz}*z, state),
> from(gamma=2 kappa=0.15 rhou=0.75 rhoz=0.95)
(setting technique to bfgs)
Iteration 0:  log likelihood= -1137.8808
Iteration 1:  log likelihood= -1097.9283 (backed up)
Iteration 2:  log likelihood= -1027.9554 (backed up)
Iteration 3:  log likelihood= -801.19555 (backed up)
Iteration 4:  log likelihood= -784.48041 (backed up)
(setting technique to nr)
Iteration 5:  log likelihood= -763.19407 (not concave)
Iteration 6:  log likelihood= -754.50175 (not concave)
Iteration 7:  log likelihood= -754.0848
Iteration 8:  log likelihood= -753.57364
Iteration 9:  log likelihood= -753.57131
Iteration 10: log likelihood= -753.57131

DSGE model
Sample: 1955q1 - 2015q4 Number of obs = 244
Log likelihood = -753.57131

<table>
<thead>
<tr>
<th></th>
<th>OIM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Coef.</td>
</tr>
<tr>
<td>/structural</td>
<td></td>
</tr>
<tr>
<td>gamma</td>
<td>1.943005</td>
</tr>
<tr>
<td>kappa</td>
<td>.1659061</td>
</tr>
<tr>
<td>rhoz</td>
<td>.9545256</td>
</tr>
<tr>
<td>rhou</td>
<td>.7005481</td>
</tr>
<tr>
<td>sd(e.u)</td>
<td>2.318205</td>
</tr>
<tr>
<td>sd(e.z)</td>
<td>.6507123</td>
</tr>
</tbody>
</table>

The estimates of the parameters other than gamma and the value of the log likelihood are nearly the same as those for the unconstrained model. The value for gamma = 1.94 is the same as 1/beta = 1/0.514 = 1.95. By tightening the convergence tolerance, we could make these values exactly the same. These values are nearly the same because this example is an instance of a general property of maximum likelihood estimators. Transforming a parameter by an invertible function does not change the log likelihood or the other parameter estimates. In other words, maximum likelihood estimators are invariant to invertible transformations of the parameters; see Casella and Berger (2002, 319) for details.
Having stored the estimates from the unconstrained model, we now estimate the parameters of the constrained model and store these results in `_constrained2`.

```stata
. constraint 2 _b[gamma] = 1.5
. dsge (p = 1/(gamma)*F.p + {kappa}*y) > (y = F.y - (r - F.p - {rhoz}*z), unobserved) > (r = ({gamma})*p + u) > (F.u = {rhou}*u, state) > (F.z = {rhoz}*z, state), > constraint(2)
(setting technique to bfgs)
Iteration 0: log likelihood = -119695.1
Iteration 1: log likelihood = -1425.592 (backed up)
Iteration 2: log likelihood = -984.57609 (backed up)
Iteration 3: log likelihood = -948.41524 (backed up)
Iteration 4: log likelihood = -945.83724 (backed up)
(switching technique to nr)
Iteration 5: log likelihood = -945.06881 (backed up)
Iteration 6: log likelihood = -764.18274 (not concave)
Iteration 7: log likelihood = -756.7209
Iteration 8: log likelihood = -755.12606
Iteration 9: log likelihood = -755.11016
Iteration 10: log likelihood = -755.11003
Iteration 11: log likelihood = -755.11003

DSGE model
Sample: 1955q1 - 2015q4 Number of obs = 244
Log likelihood = -755.11003
( 1) [structural]gamma = 1.5

OIM
Coef. Std. Err. z  P>|z|  [95% Conf. Interval]
/structural
  gamma 1.5 (constrained)
  kappa .1076807  .0276892 3.89  0.000   .0534109   .1619506
  rhoz .9538519  .0187789 50.79  0.000   .9170458   .9906579
  rhou .7214333  .043967 16.41  0.000   .6352595   .807607
sd(e.u) 1.915459  .0867103 21.14  0.000   .74551  2.085408
sd(e.z) .4936815  .0805138 6.13  0.000   .3358773   .6514856
```

The estimates store `_constrained2`.

The estimates of the parameters other than `gamma` and the value of the log likelihood are the same as those for the `constrained` model. This is another instance of the invariance of the maximum likelihood estimator to invertible transformations of the parameters.

Having stored the log likelihoods from the constrained and unconstrained model, we now perform an LR of the null hypothesis that \( \gamma = 1.5 \).

```stata
. lrtest unconstrained2 constrained2
 Likelihood-ratio test  LR chi2(1) = 3.08
 (Assumption: constrained2 nested in unconstrained2)  Prob > chi2 = 0.0794
```

The LR test statistic and its \( p \)-value are the same as those reported for the test against the null hypothesis that \( \beta = 2/3 \), which illustrates that LR tests are invariant to nonlinear transforms.
References


Also see

[DSGE] dsge postestimation — Postestimation tools for dsge
[DSGE] dsgenl postestimation — Postestimation tools for dsgenl
[R] lrtest — Likelihood-ratio test after estimation
[R] test — Test linear hypotheses after estimation
dsge fits linear dynamic stochastic general equilibrium (DSGE) models to multiple time series. DSGE models are systems of equations that are motivated by economic theory. In these systems, expectations of future values of variables can affect the current values. The parameters of these models are often directly interpretable in terms of economic theory.

Syntax

```
dsge eqlist [if] [in] [, options]
```

<table>
<thead>
<tr>
<th>options</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Main</strong></td>
<td></td>
</tr>
<tr>
<td><code>constraints(constraints)</code></td>
<td>apply specified linear constraints</td>
</tr>
<tr>
<td><code>noidencheck</code></td>
<td>do not check for parameter identification</td>
</tr>
<tr>
<td><code>solve</code></td>
<td>return model solution at initial values</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 oim or robust</td>
</tr>
<tr>
<td><strong>Reporting</strong></td>
<td></td>
</tr>
<tr>
<td><code>level(#)</code></td>
<td>set confidence level; default is level(95)</td>
</tr>
<tr>
<td><code>nocsreport</code></td>
<td>do not display constraints</td>
</tr>
<tr>
<td><code>display_options</code></td>
<td>control columns and column formats and line width</td>
</tr>
<tr>
<td><strong>Maximization</strong></td>
<td></td>
</tr>
<tr>
<td><code>maximize_options</code></td>
<td>control the maximization process</td>
</tr>
<tr>
<td><strong>Advanced</strong></td>
<td></td>
</tr>
<tr>
<td><code>nodemean</code></td>
<td>do not demean data prior to estimation</td>
</tr>
<tr>
<td><code>post</code></td>
<td>force posting of estimation results in the event of errors caused by lack of identification or stability</td>
</tr>
<tr>
<td><code>idtolerance(#)</code></td>
<td>set tolerance used for identification check; seldom used</td>
</tr>
<tr>
<td><code>lintolerance(#)</code></td>
<td>set tolerance used for linearity check; seldom used</td>
</tr>
<tr>
<td><code>coeflegend</code></td>
<td>display legend instead of statistics</td>
</tr>
</tbody>
</table>
You must `tsset` your data before using `dsge`; see [TS] `tsset`.
`coeflegend` does not appear in the dialog box. See [U] 20 Estimation and postestimation commands for more capabilities of estimation commands.

Below we present the full specification of `eqlist`. You may prefer to start with the syntax discussion in [DSGE] Intro 2.

`eqlist` is

```
eq
eq eqlist
```

eq is

```
cntrl_eq
state_eq
```

cntrl_eq contains

```
(ocntrl_var = termlist)
(ucntrl_var = termlist, unobserved)
(parm_exp * ocntrl_var = termlist)
(parm_exp * ucntrl_var = termlist, unobserved)
```

state_eq contains

```
(F. state_var = , state)
(F. state_var = termlist, state [noshock])
(parm_exp * F. state_var = termlist, state [noshock])
```

ocntrl_var is Stata variable to be treated as an observed control in the system

ucntrl_var is name to be treated as an unobserved control in the system

state_var is name to be treated as an unobserved state in the system

If there happens to be a Stata variable with the same name as `ucntrl_var` or `state_var`, the variable is ignored and plays no role in the estimation.

termlist is

```
term
term + termlist
term - termlist
```

term is

```
rhs_var
parm_exp * rhs_var
parm_exp * (termlist)
```

rhs_var is

```
ocntrl_var
ucntrl_var
state_var
F.ocntrl_var
F.ucntrl_var
```
parm_exp is scalar substitutable expression;  
parm_spec elements are allowed and expected  
rhs_var are not allowed

parm_spec is

{parm_name}
{parm_name=initial_val}

parm_name is name used to identify model parameter

initial_val is numeric literal; specifies an initial value

Options

- **Main**

  constraints(\textit{constraints}); see \texttt{[R] Estimation options}.

  noidencheck skips the check that the parameters are identified at the initial values. Models that are not structurally identified can still converge, thereby producing meaningless results that only appear to have meaning; thus care should be taken in specifying this option. See \texttt{[DSGE] Intro 6} for details.

  solve puts the model into state-space form at the initial parameter values. No standard errors are produced.

- **SE/Robust**

  \texttt{vce(\textit{vcetype})} specifies the type of standard error reported, which includes types that are derived from asymptotic theory (\texttt{oim}) and that are robust to some kinds of misspecification (\texttt{robust}); see \texttt{[R] vce_option}.

- **Reporting**

  level(#), noconsreport; see \texttt{[R] Estimation options}.

  \textit{display_options}: noci, nopvalues, cformat(\texttt{\%fmt}), pformat(\texttt{\%fmt}), sformat(\texttt{\%fmt}), and nostretch; see \texttt{[R] Estimation options}.

- **Maximization**

  maximize_options: difficult, technique(\texttt{algorithm_spec}), iterate(#), [no]log, trace, gradient, showstep, hessian, showtolerance, tolerance(#), ltolerance(#), nrtolerance(#), and from(\texttt{init_specs}); see \texttt{[R] Maximize}.

- **Advanced**

  nodemean prevents \texttt{dsge} from removing the mean of the observed control variables prior to estimation.

  post causes \texttt{dsge} to post the parameter vector into \texttt{e()}, even in the event of errors that arise from checking stability conditions or identification.

  idtolerance(#) specifies the tolerance used in the identification diagnostic. The default is idtolerance(1e-6).

  lintolerance(#) specifies the tolerance used in the linearity diagnostic. The default is lintolerance(1e-12).
The following option is available with \texttt{dsge} but is not shown in the dialog box: \texttt{coeflegend}; see [R] Estimation options.

**Remarks**

For an introduction to what \texttt{dsge} can do and how to use it, see [DSGE] Intro 1. It is highly recommended that you read the introduction first.

For examples of \texttt{dsge}, see the examples of classic DSGE models in [DSGE] Intro 3a, [DSGE] Intro 3b, and [DSGE] Intro 3c. Additional examples are presented in [DSGE] Intro 4a–[DSGE] Intro 4g. See [DSGE] Intro 4 for an overview of these examples.

**Stored results**

\texttt{dsge} stores the following in \texttt{e()}: 

Scalars

- \texttt{e(N)}: number of observations
- \texttt{e(k)}: number of parameters
- \texttt{e(k_eq)}: number of equations in \texttt{e(b)}
- \texttt{e(k_dv)}: number of dependent variables
- \texttt{e(k_state)}: number of state equations
- \texttt{e(k_control)}: number of control equations
- \texttt{e(k_shock)}: number of shocks
- \texttt{e(k_observed)}: number of observed control equations
- \texttt{e(k_stable)}: number of stable eigenvalues
- \texttt{e(ll)}: log likelihood
- \texttt{e(tmin)}: minimum time in sample
- \texttt{e(tmax)}: maximum time in sample
- \texttt{e(rank)}: rank of VCE
- \texttt{e(ic)}: number of iterations
- \texttt{e(rc)}: return code
- \texttt{e(converged)}: 1 if converged, 0 otherwise

Macros

- \texttt{e(cmd)}: \texttt{dsge}
- \texttt{e(cmdline)}: command as typed
- \texttt{e(depvar)}: unoperated names of dependent variables
- \texttt{e(state)}: unoperated names of state variables
- \texttt{e(control)}: unoperated names of control variables
- \texttt{e(observed)}: unoperated names of observed control variables
- \texttt{e(title)}: title in estimation output
- \texttt{e(tvar)}: variable denoting time within groups
- \texttt{e(tmins)}: formatted minimum time
- \texttt{e(tmaxs)}: formatted maximum time
- \texttt{e(timestamp)}: format for the current time variable
- \texttt{e(vcetype)}: \texttt{vcetype} specified in \texttt{vce()}
- \texttt{e(vce)}: title used to label Std. Err.
- \texttt{e(opt)}: type of optimization
- \texttt{e(method)}: likelihood method
- \texttt{e(idencheck)}: passed, failed, or skipped
- \texttt{e(technique)}: maximization technique
- \texttt{e(properties)}: \texttt{b V}
- \texttt{e(estat_cmd)}: program used to implement \texttt{estat}
- \texttt{e(predict)}: program used to implement \texttt{predict}
- \texttt{e(marginsnotok)}: predictions disallowed by \texttt{margins}
Methods and formulas

A DSGE model is a system of equations in which the values of the control variables $y_t$ and of the state variables $x_t$ depend on their values in previous periods and their one-period-ahead rational expectations; see [DSGE] Intro 1 for an introduction to control variables and state variables. The structural form of a DSGE model is

$$A_0 y_t = A_1 E_t(y_{t+1}) + A_2 y_t + A_3 x_t$$
$$B_0 x_{t+1} = B_1 E_t(y_{t+1}) + B_2 y_t + B_3 x_t + C \epsilon_{t+1}$$

where the entries in $A_i$ and $B_i$ are functions of the parameter vector $\theta$ and $C$ is a selector matrix of zeros and ones that puts shocks in the state equations that include them. The $A_0$ and $B_0$ matrices are diagonal, and $A_2$ has zeros on its diagonal. See [DSGE] Intro 1 for details about how the structural parameters $\theta$ get mapped into entries in $A_i$ and $B_i$ matrices.

dse implements the solution technique described in Klein (2000), which obtains the state-space form in (3) and (4) from the structural model in (1) and (2).

We can only solve a structural model for its state-space form when i) the structural form has the structure implied by (1) and (2) and when ii) the values of the structural parameters imply that the model neither spirals out of control nor converges to a point, a condition known as saddle-path stability.

Condition (i) seems trivial, but some minor manipulation is frequently required to make it true. Many DSGE models include a problematic term like a shock to a control variable or a lagged state variable. These problematic terms do not fit into the structure required by (1) and (2), but the model is easily rewritten to accommodate these problematic terms; see [DSGE] Intro 4 for details.
Condition (ii), the saddle-path stability condition, depends on the values of the parameters. We can only solve models for parameter values that imply a saddle-path-stable model; see \cite{DSGE Intro 5} for details.

Assuming that the shocks $\epsilon_t$ are mean-zero, independent, and normally distributed implies that (3) and (4) form a linear state-space model in which the coefficients are nonlinear functions of the structural parameters $\theta$. This structure allows \texttt{dsge} to estimate the parameters $\theta$ and the standard deviations of the shocks by maximum likelihood.

The Kalman filter is used to form the log-likelihood function. The Kalman filter is a method for recursively obtaining linear, least-squares forecasts conditional on past information. These forecasts are used to construct the log likelihood, assuming normality and stationarity. See \textit{Methods and formulas} in \cite{TS sspace} for details of the Kalman filter.

When the shocks are independent and identically distributed, but not normally distributed, the maximum likelihood estimator is consistent for $\theta$, but \texttt{vce(robust)} standard errors must be used. (The process of using a maximum likelihood estimator under weaker distributional assumptions and correcting the standard errors is known as a quasi–maximum likelihood estimator in the literature.)

One final caveat is that only a subset $y_{1,t}$ of the controls is treated as observed in some models. This caveat is handled by augmenting the state-space systems in (3) and (4) with the observation equation

$$y_{1,t} = D y_t$$

where $D$ is a selection matrix that picks out the rows in $G$ corresponding to elements in the control vector that are observed.

Asymptotic standard errors for postestimation objects such as the entries in the state-space matrices and the impulse responses are obtained using the delta method. See Serfling (1980, sec. 3.3) for a discussion of the delta method.

**References**


**Also see**

\cite{DSGE} \texttt{dsge postestimation} — Postestimation tools for \texttt{dsge}

\cite{DSGE} \texttt{Intro 2} — Learning the syntax

\cite{TS} \texttt{sspace} — State-space models

\cite{TS} \texttt{tsset} — Declare data to be time-series data

\cite{TS} \texttt{var} — Vector autoregressive models

\cite{U} \texttt{20 Estimation and postestimation commands}
## Postestimation commands

The following postestimation commands are of special interest after `dsge`:

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>estat policy</code></td>
<td>display policy matrix of estimated model</td>
</tr>
<tr>
<td><code>estat stable</code></td>
<td>assess stability of the system</td>
</tr>
<tr>
<td><code>estat transition</code></td>
<td>display transition matrix of estimated model</td>
</tr>
<tr>
<td><code>irf</code></td>
<td>create and analyze IRFs and FEVDs</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>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>forecast</code></td>
<td>dynamic forecasts and simulations</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>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>one-step-ahead predictions, prediction errors, 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>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 new variables containing predictions such as expected values. Predictions are available as static one-step-ahead predictions or as dynamic multistep predictions, and you can control when dynamic predictions begin.

Menu for predict

Statistics > Postestimation

Syntax for predict

predict [type] {stub*|newvarlist} [if] [in] [ , statistic options ]

<table>
<thead>
<tr>
<th>statistic</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>xb</td>
<td>linear prediction for observed variables</td>
</tr>
<tr>
<td>states</td>
<td>linear prediction for latent state variables</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>options</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>rmse(stub*</td>
<td>newvarlist)</td>
</tr>
<tr>
<td>dynamic(time constant)</td>
<td>begin dynamic forecast at specified time</td>
</tr>
</tbody>
</table>

Advanced

| smethod(method) | method for predicting unobserved states |

<table>
<thead>
<tr>
<th>method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>onestep</td>
<td>predict using past information</td>
</tr>
<tr>
<td>filter</td>
<td>predict using past and contemporaneous information</td>
</tr>
</tbody>
</table>

Options for predict

xb, the default, calculates the linear predictions of the observed variables.
states calculates the linear predictions of the latent state variables.
Options

\texttt{rmse(stub* | newvarlist)} puts the root mean squared errors of the predicted statistics into the specified new variables. The root mean squared errors measure the variances due to the disturbances but do not account for estimation error.

\texttt{dynamic(time_constant)} specifies when \texttt{predict} starts producing dynamic forecasts. The specified \texttt{time_constant} must be in the scale of the time variable specified in \texttt{tsset}, and the \texttt{time_constant} must be inside a sample for which observations on the dependent variables are available. For example, \texttt{dynamic(tq(2008q4))} causes dynamic predictions to begin in the fourth quarter of 2008, assuming that your time variable is quarterly; see \texttt{[D] Datetime}. If the model contains exogenous variables, they must be present for the whole predicted sample.

Advanced

\texttt{smethod(method)} specifies the method for predicting the unobserved states, \texttt{smethod(onestep)} and \texttt{smethod(filter)}, and causes different amounts of information on the dependent variables to be used in predicting the states at each time period.

\texttt{smethod(onestep)}, the default, causes \texttt{predict} to estimate the states at each time period using previous information on the dependent variables. The Kalman filter is performed on previous periods, but only the one-step predictions are made for the current period.

\texttt{smethod(filter)} causes \texttt{predict} to estimate the states at each time period using previous and contemporaneous data by the Kalman filter. The Kalman filter is performed on previous periods and the current period. \texttt{smethod(filter)} may be specified only with \texttt{states}.

Remarks and examples

For examples of \texttt{estat policy}, see \texttt{[DSGE Intro 1]}, \texttt{[DSGE Intro 3a]}, and \texttt{[DSGE Intro 3c]}.

For examples of \texttt{estat transition}, see \texttt{[DSGE Intro 1]}, \texttt{[DSGE Intro 3a]}, and \texttt{[DSGE Intro 3b]}.

For an example of \texttt{estat stable}, see \texttt{[DSGE Intro 5]}.

For examples of \texttt{irf after dsge}, see \texttt{[DSGE Intro 1]}, \texttt{[DSGE Intro 3b]}, and \texttt{[DSGE Intro 3c]}.

For an example of \texttt{forecast after dsge}, see \texttt{[DSGE Intro 1]}.

For examples of \texttt{predict after dsge}, see \texttt{[DSGE Intro 3a]}.

Methods and formulas

Estimating the unobserved states is the key to predicting the observed variables.

By default and with the \texttt{smethod(onestep)} option, \texttt{predict} estimates the state in each period by applying the Kalman filter to all previous periods and only making the one-step prediction to the current period.

With the \texttt{smethod(filter)} option, \texttt{predict} estimates the states in each period by applying the Kalman filter on all previous periods and the current period. The computational difference between \texttt{smethod(onestep)} and \texttt{smethod(filter)} is that \texttt{smethod(filter)} performs the update step on the current period while \texttt{smethod(onestep)} does not. The statistical difference between \texttt{smethod(onestep)} and \texttt{smethod(filter)} is that \texttt{smethod(filter)} uses contemporaneous information on the observed variables while \texttt{smethod(onestep)} does not.

The observed control variables are predicted by plugging in the estimated states.
Also see

[DSGE] dsge — Linear dynamic stochastic general equilibrium models
[DSGE] estat policy — Display policy matrix
[DSGE] estat stable — Check stability of system
[DSGE] estat transition — Display state transition matrix
[TS] forecast — Econometric model forecasting
[TS] irf — Create and analyze IRFs, dynamic-multiplier functions, and FEVDs
[U] 20 Estimation and postestimation commands
dsgenl — Nonlinear dynamic stochastic general equilibrium models

Description

dsgenl fits nonlinear dynamic stochastic general equilibrium (DSGE) models to multiple time series via perturbation methods. DSGE models are systems of equations that are motivated by economic theory. In these systems, expectations of future values of variables can affect the current values. The parameters of these models are often directly interpretable in terms of economic theory.

Menu

Statistics > Multivariate time series > Dynamic stochastic general equilibrium (DSGE) models > Nonlinear DSGE models

Syntax

```
dsgenl (lexp_1 = rexp_1) [ (lexp_2 = rexp_2) ...] [ if ] [ in ],
    observed(varlist) exostate(namelist) [ options ]
```

rexp_\$j and lexp_\$j are substitutable expressions on the right-hand side and left-hand side of equation \$j. These are Stata expressions that include scalars, variables, and parameters to be estimated. The variables may be state variables, observed control variables, and unobserved control variables. The lead operator, F., can be applied to a variable so that F.\$varname refers to the expected value of \$varname in the next time period. No other time-series operators are allowed; see [DSGE] Intro 4 for information on including additional lags and leads. Parameters to be estimated are enclosed in curly braces; \{\text{beta}\} is an example of a parameter. Initial values for parameters are given by including an equal sign and the initial value inside the braces; for example, \{\text{beta}=2\} sets the initial value for \text{beta} to 2. See Specifying the system of nonlinear equations in [DSGE] Intro 2.
Options

Main

*observed(varlist)  list of observed control variables
unobserved(namelist)  list of unobserved control variables
*exostate(namelist)  list of exogenous state variables
endostate(namelist)  list of endogenous state variables
constraints(constraints)  apply specified linear constraints
linearapprox  take a linear, rather than log-linear, approximation
noidencheck  do not check for parameter identification
solve  return model solution at initial values

SE/Robust

vce(vcetype)  vcetype may be oim or robust

Reporting

level(#)  set confidence level; default is level(95)
ocnsreport  do not display constraints
display_options  control columns and column formats and line width

Maximization

maximize_options  control the maximization process

Advanced

nodean  do not demean data prior to estimation
post  force posting of estimation results in the event of errors caused by lack of identification or stability
idtolerance(#)  set tolerance used for identification check; seldom used
steadytolerance(#)  set tolerance used for convergence in steady-state calculations; seldom used
steadyinit(matrix)  set initial values for steady state; seldom used
coeflegend  display legend instead of statistics

*observed() and exostate() are required.

You must tsset your data before using dsgenl; see [TS] tsset.
coeflegend does not appear in the dialog box.

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

Options

Main

observed(varlist) specifies which variables that appear in the model equations are observed control variables. observed() is required.

unobserved(namelist) specifies which variables that appear in the model equations are unobserved control variables. The names must be valid Stata names but need not exist in your dataset.

exostate(namelist) specifies which variables that appear in the model equations are exogenous state variables. Exogenous state variables are subject to shocks. The names must be valid Stata names but need not exist in your dataset. There must be the same number of exogenous state variables as there are observed control variables in your model. exostate() is required.
endostate(namelist) specifies which variables that appear in the model equations are endogenous state variables. Endogenous state variables are not subject to shocks. The names must be valid Stata names but need not exist in your dataset.

constraints(constraints); see [R] Estimation options.

linearapprox specifies that a linear approximation be applied to the DSGE model rather than a log-linear approximation. In either case, an approximation is applied. In a log-linear approximation, variables are interpreted as percentage deviations from steady state. In a linear approximation, variables are interpreted as unit deviations from steady state.

noidencheck skips the check that the parameters are identified at the initial values. Models that are not structurally identified can still converge, thereby producing meaningless results that only appear to have meaning; thus, care should be taken in specifying this option. See [DSGE] Intro 6 for details.

solve puts the model into state-space form at the initial parameter values. No standard errors are produced.

vce(vcetype) specifies the type of standard error reported, which includes types that are derived from asymptotic theory (oim) and that are robust to some kinds of misspecification (robust); see [R] vce_option.

level(#), nocnsreport; see [R] Estimation options.

display_options: noci, nopvalues, cformat(%,fmt), pformat(%,fmt), sformat(%,fmt), and nolstretch; see [R] Estimation options.

maximize_options: difficult, technique(algorithm_spec), iterate(#), [no]log, trace, gradient, showstep, hessian, showtolerance, tolerance(#), ltolerance(#), nrtolerance(#), and from(init_specs); see [R] Maximize.

nodemean prevents dsgenl from removing the mean of the observed control variables prior to estimation.

post causes dsgenl to post the parameter vector into e(), even in the event of errors that arise from checking stability conditions or identification.

idtolerance(#) specifies the tolerance used in the identification diagnostic. The default is idtolerance(1e-6).

steadytolerance(#) specifies the tolerance used in calculations of the model’s steady state. The default is steadytolerance(1e-17).

steadyinit(matrix) specifies a vector of initial values for use in finding the steady state.

The following option is available with dsgenl but is not shown in the dialog box: coeflegend; see [R] Estimation options.
Remarks

For an introduction to what `dsgenl` can do and how to use it, see [DSGE] Intro 1. It is highly recommended that you read the introduction first.

For examples of `dsgenl`, see the examples of classic DSGE models in [DSGE] Intro 3d, [DSGE] Intro 3e, and [DSGE] Intro 3f.

Stored results

`dsgenl` 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_dv)` number of dependent variables
- `e(k_state)` number of state equations
- `e(k_control)` number of control equations
- `e(k_shock)` number of shocks
- `e(k_observed)` number of observed control equations
- `e(k_stable)` number of stable eigenvalues
- `e(ll)` log likelihood
- `e(tmin)` minimum time in sample
- `e(tmax)` maximum time in sample
- `e(rank)` rank of VCE
- `e(ic)` number of iterations
- `e(rc)` return code
- `e(converged)` 1 if converged, 0 otherwise

Macros

- `e(cmd)` `dsgenl`
- `e(cmdline)` command as typed
- `e(depvar)` unoperated names of dependent variables
- `e(state)` unoperated names of state variables
- `e(shock)` unoperated names of state variables subject to shocks
- `e(control)` unoperated names of control variables
- `e(observed)` unoperated names of observed control variables
- `e(title)` title in estimation output
- `e(tvar)` variable denoting time within groups
- `e(tmins)` formatted minimum time
- `e(tmaxs)` formatted maximum time
- `e(tfmt)` format for the current time variable
- `e(vce)` vcetype specified in `vce()`
- `e(vcetype)` title used to label Std. Err.
- `e(opt)` type of optimization
- `e(method)` maximization technique
- `e(idencheck)` passed, failed, or skipped
- `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`
Methods and formulas

A DSGE model is a system of equations in which the values of the control variables \( y_t \) and of the state variables \( x_t \) depend on their values in previous periods and their one-period-ahead rational expectations; see [DSGE Intro 1] for an introduction to control variables and state variables. The structural form of a nonlinear DSGE model is

\[
E_t \{ f(x_{t+1}, y_{t+1}, x_t, y_t; \theta) \} = 0
\]

(1)

where \( f \) is an \( n \times 1 \) vector of functions, the \( n_y \) variables \( y_t \) are control variables, and the \( n_x \times 1 \) variables \( x_t \) are state variables. The parameter \( \theta \) contains the model’s structural parameters.

Fernández-Villaverde, Rubio-Ramírez, and Schorfheide (2016) contain an introduction to the solution and estimation of such models. dsgenl proceeds by linearizing (1) around the steady state and then solving the resulting linear system.

The steady state is found by solving the deterministic system of equations,

\[
f(x, y, x, y; \theta) = 0
\]

which is \( n \) equations in \( n \) unknowns. The location of the steady state is a vector \((\bar{x}, \bar{y})\), each element of which may depend on the structural parameters.

Once the steady state has been found, the vector of functions \( f \) is linearized around that point, yielding

\[
E_t \{ F_1 x_{t+1} + F_2 y_{t+1} + F_3 x_t + F_4 y_t \} = 0
\]

(2)

Equation (2) is in the form of a linearized DSGE model, and the remaining solution techniques are the same as for dsge; see Methods and formulas in [DSGE] dsge. Parameter estimation proceeds on the basis of the linearized equation shown as (2).

Reference

Also see

[DSGE] dsgenl postestimation — Postestimation tools for dsgenl
[DSGE] Intro 2 — Learning the syntax
[TS] sspace — State-space models
[TS] tsset — Declare data to be time-series data
[TS] var — Vector autoregressive models
[U] 20 Estimation and postestimation commands
Postestimation commands

The following postestimation commands are of special interest after `dsgenl`:

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>estat covariance</code></td>
<td>display model-implied covariance matrix of model variables</td>
</tr>
<tr>
<td><code>estat policy</code></td>
<td>display policy matrix of estimated model</td>
</tr>
<tr>
<td><code>estat stable</code></td>
<td>assess stability of the system</td>
</tr>
<tr>
<td><code>estat steady</code></td>
<td>display steady state of the system</td>
</tr>
<tr>
<td><code>estat transition</code></td>
<td>display transition matrix of estimated model</td>
</tr>
<tr>
<td><code>irf</code></td>
<td>create and analyze IRFs and FEVDs</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>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>forecast</code></td>
<td>dynamic forecasts and simulations</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>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>one-step-ahead predictions, prediction errors, 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>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 new variables containing predictions such as expected values. Predictions are available as static one-step-ahead predictions or as dynamic multistep predictions, and you can control when dynamic predictions begin.

Menu for predict

Statistics > Postestimation

Syntax for predict

predict [type] {stub* | newvarlist} [if] [in] [, statistic options]

<table>
<thead>
<tr>
<th>statistic</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>xb</td>
<td>linear prediction for observed variables</td>
</tr>
<tr>
<td>states</td>
<td>linear prediction for latent state variables</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>options</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>rmse(stub*</td>
<td>newvarlist)</td>
</tr>
<tr>
<td>dynamic(time constant)</td>
<td>begin dynamic forecast at specified time</td>
</tr>
</tbody>
</table>

Advanced

| smethod(method) | method for predicting unobserved states |

<table>
<thead>
<tr>
<th>method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>onestep</td>
<td>predict using past information</td>
</tr>
<tr>
<td>filter</td>
<td>predict using past and contemporaneous information</td>
</tr>
</tbody>
</table>

Options for predict

xb, the default, calculates the linear predictions of the observed variables.

states calculates the linear predictions of the latent state variables.
Options

**rmse** (*stub* | *newvarlist*) puts the root mean squared errors of the predicted statistics into the specified new variables. The root mean squared errors measure the variances due to the disturbances but do not account for estimation error.

**dynamic** (*time_constant*) specifies when *predict* starts producing dynamic forecasts. The specified *time_constant* must be in the scale of the time variable specified in *tsset*, and the *time_constant* must be inside a sample for which observations on the dependent variables are available. For example, **dynamic(tq(2008q4))** causes dynamic predictions to begin in the fourth quarter of 2008, assuming that your time variable is quarterly; see [D] Datetime. If the model contains exogenous variables, they must be present for the whole predicted sample.

Advanced

**smethod** (*method*) specifies the method for predicting the unobserved states, **smethod(onestep)** and **smethod(filter)**, and causes different amounts of information on the dependent variables to be used in predicting the states at each time period.

**smethod(onestep)**, the default, causes *predict* to estimate the states at each time period using previous information on the dependent variables. The Kalman filter is performed on previous periods, but only the one-step predictions are made for the current period.

**smethod(filter)** causes *predict* to estimate the states at each time period using previous and contemporaneous data by the Kalman filter. The Kalman filter is performed on previous periods and the current period. **smethod(filter)** may be specified only with **states**.

Remarks and examples

For examples of **estat covariance**, see [DSGE] Intro 3e and [DSGE] estat covariance.

For examples of **estat policy**, see [DSGE] Intro 1, [DSGE] Intro 3d, and [DSGE] Intro 3e.

For examples of **estat transition**, see [DSGE] Intro 1, [DSGE] Intro 3d, and [DSGE] Intro 3e.

For examples of **irf** after *dsgenl*, see [DSGE] Intro 3d and [DSGE] Intro 3e.

For examples of **predict** and **forecast** after fitting a DSGE model, see [DSGE] Intro 1. Examples are shown with estimation results from *dsge* but apply to results from *dsgenl* as well.

Methods and formulas

The most important thing to keep in mind is that postestimation predictions after *dsgenl* are made on the basis of the linearized form of the model given to *dsgenl*.

Estimating the unobserved states is the key to predicting the observed variables.

By default and with the **smethod(onestep)** option, *predict* estimates the state in each period by applying the Kalman filter to all previous periods and only making the one-step prediction to the current period.

With the **smethod(filter)** option, *predict* estimates the states in each period by applying the Kalman filter on all previous periods and the current period. The computational difference between **smethod(onestep)** and **smethod(filter)** is that **smethod(filter)** performs the update step on the current period while **smethod(onestep)** does not. The statistical difference between **smethod(onestep)** and **smethod(filter)** is that **smethod(filter)** uses contemporaneous information on the observed variables while **smethod(onestep)** does not.

The observed control variables are predicted by plugging in the estimated states.
Also see

[DSGE] **ds genl** — Nonlinear dynamic stochastic general equilibrium models

[DSGE] **estat covariance** — Display estimated covariances of model variables

[DSGE] **estat policy** — Display policy matrix

[DSGE] **estat stable** — Check stability of system

[DSGE] **estat steady** — Display steady state of nonlinear DSGE model

[DSGE] **estat transition** — Display state transition matrix

[TS] **forecast** — Econometric model forecasting

[TS] **irf** — Create and analyze IRFs, dynamic-multiplier functions, and FEVDs

[U] 20 Estimation and postestimation commands
**Description**

`estat covariance` displays model-implied covariances among control variables.

**Quick start**

Display variances and covariances of all control variables in the model

```
estat covariance
```

Include only variances and covariances of x and its lag

```
estat covariance x L.x
```

Display the variance of x1 and the covariance of x1 with the lag of x1 and with x2

```
estat covariance x1, addcovariance(L.x1 x2)
```

**Menu for estat**

Statistics > Postestimation

**Syntax**

```
estat covariance [ varlist ] [ , options ]
```

`varlist` may include control variables and their lags. If `varlist` is not specified, variances and covariances are reported for all control variables in the model.

<table>
<thead>
<tr>
<th>options</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>addcovariance(clistc)</code></td>
<td>report additional covariances</td>
</tr>
<tr>
<td><code>nocovariance</code></td>
<td>do not report covariances</td>
</tr>
<tr>
<td><code>post</code></td>
<td>post variances and covariances and their VCE as estimation results</td>
</tr>
<tr>
<td><code>level(#)</code></td>
<td>set confidence level; default is <code>level(95)</code></td>
</tr>
<tr>
<td><code>display_options</code></td>
<td>control columns and column formats and line width</td>
</tr>
</tbody>
</table>

**Options**

`addcovariance(clistc)` specifies that the covariances between the control variables specified in `clistc` and those specified in `varlist` be displayed. The variances of variables in `clistc` are not reported. `clistc` can contain lags of the control variables in the model.
nocovariance specifies that no covariance be displayed. nocovariance may not be specified with addcovariance().

post causes estat covariance to behave like a Stata estimation (e-class) command. estat covariance posts the estimated variance–covariance matrix to e(), so you can treat it just as you would results from any other estimation command.

level(%) specifies the confidence level, as a percentage, for confidence intervals. The default is level(95) or as set by set level; see [U] 20.8 Specifying the width of confidence intervals.

display_options: noci, nopvalues, cformat(%,fmt), pformat(%,fmt), sformat(%,fmt), and nolstretch; see [R] Estimation options.

Remarks and examples

estat covariance displays covariances among control variables implied by a DSGE model.

We provide examples of the use of estat covariance using the model from [DSGE] Intro 3e. In that introduction, we estimated some of the parameters of a Real Business Cycle model. Model setup and estimation were given by

```stata
use https://www.stata-press.com/data/r16/usmacro2
(Federal Reserve Economic Data - St. Louis Fed, 2017-01-15)
constraint 1 _b[alpha] = 0.33
constraint 2 _b[beta] = 0.99
constraint 3 _b[delta] = 0.025
constraint 4 _b[ch] = 2
.dsgenl (1/c = {beta}*(1/F.c)*(1+r-{delta}))
> ( {chi}*h = w/c)
> ( y = c + i)
> ( y = z*k^({alpha}*h^-1-{alpha})))
> ( r = {alpha}*y/k)
> ( w = (1-{alpha})*y/h)
> (F.k = i + (1-{delta})*k)
> (ln(F.z) = {rho}*ln(z))
> , observed(y) unobserved(c i r w h) exostate(z) endostate(k)
> constraint(1/4) nolog
```

Solving at initial parameter vector ...
Checking identification ...
First-order DSGE model
Sample: 1955q1 - 2015q4 Number of obs = 244
Log likelihood = -639.38787
( 1) [/structural]alpha = .33
( 2) [/structural]beta = .99
( 3) [/structural]delta = .025
( 4) [/structural]chi = 2

<table>
<thead>
<tr>
<th></th>
<th>OIM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Coef.</td>
</tr>
<tr>
<td>/structural</td>
<td></td>
</tr>
<tr>
<td>beta</td>
<td>.99</td>
</tr>
<tr>
<td>delta</td>
<td>.025</td>
</tr>
<tr>
<td>chi</td>
<td>2</td>
</tr>
<tr>
<td>alpha</td>
<td>.33</td>
</tr>
<tr>
<td>rho</td>
<td>.3132837</td>
</tr>
<tr>
<td>sd(e.z)</td>
<td>2.249022</td>
</tr>
</tbody>
</table>
The commands just listed imported data, set up constraints, and estimated the remaining model parameters. For details of the model, see [DSGE] Intro 3e. The control variables in this model are \( c, i, r, w, h, \) and \( y \).

By default, `estat covariance` displays the contemporaneous variances and covariances of all the control variables in the model.

```
. estat covariance
```

Estimated covariances of model variables

<table>
<thead>
<tr>
<th></th>
<th>Delta-method</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Coef.</td>
<td>Std. Err.</td>
<td>z</td>
<td>P&gt;</td>
<td>z</td>
</tr>
<tr>
<td>( c )</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>var(( c ))</td>
<td>.8564586</td>
<td>.1674129</td>
<td>5.12</td>
<td>0.000</td>
<td>.5283354</td>
</tr>
<tr>
<td>cov(( c, i ))</td>
<td>1.699536</td>
<td>.432758</td>
<td>3.93</td>
<td>0.000</td>
<td>.8513461</td>
</tr>
<tr>
<td>cov(( c, r ))</td>
<td>-.4153308</td>
<td>.056547</td>
<td>-7.50</td>
<td>0.000</td>
<td>-.523809</td>
</tr>
<tr>
<td>cov(( c, w ))</td>
<td>.9555354</td>
<td>.198332</td>
<td>4.82</td>
<td>0.000</td>
<td>.5668118</td>
</tr>
<tr>
<td>cov(( c, h ))</td>
<td>.0990768</td>
<td>.0314055</td>
<td>3.15</td>
<td>0.002</td>
<td>.0375231</td>
</tr>
<tr>
<td>cov(( c, y ))</td>
<td>1.054612</td>
<td>.2293833</td>
<td>4.60</td>
<td>0.000</td>
<td>.6050292</td>
</tr>
<tr>
<td>( i )</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>var(( i ))</td>
<td>207.93</td>
<td>20.07228</td>
<td>10.36</td>
<td>0.000</td>
<td>168.589</td>
</tr>
<tr>
<td>cov(( i, r ))</td>
<td>50.23759</td>
<td>4.822667</td>
<td>10.42</td>
<td>0.000</td>
<td>40.78534</td>
</tr>
<tr>
<td>cov(( i, w ))</td>
<td>25.93532</td>
<td>2.625692</td>
<td>9.88</td>
<td>0.000</td>
<td>20.78906</td>
</tr>
<tr>
<td>cov(( i, h ))</td>
<td>24.23579</td>
<td>2.325743</td>
<td>10.42</td>
<td>0.000</td>
<td>19.67741</td>
</tr>
<tr>
<td>cov(( i, y ))</td>
<td>50.17111</td>
<td>4.941598</td>
<td>10.15</td>
<td>0.000</td>
<td>40.48575</td>
</tr>
<tr>
<td>( r )</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>var(( r ))</td>
<td>12.94748</td>
<td>1.274919</td>
<td>10.16</td>
<td>0.000</td>
<td>10.44869</td>
</tr>
<tr>
<td>cov(( r, w ))</td>
<td>5.537299</td>
<td>.524047</td>
<td>10.57</td>
<td>0.000</td>
<td>4.510185</td>
</tr>
<tr>
<td>cov(( r, h ))</td>
<td>5.95263</td>
<td>.5725108</td>
<td>10.40</td>
<td>0.000</td>
<td>4.830529</td>
</tr>
<tr>
<td>cov(( r, y ))</td>
<td>11.48993</td>
<td>1.096232</td>
<td>10.48</td>
<td>0.000</td>
<td>9.341353</td>
</tr>
<tr>
<td>( w )</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>var(( w ))</td>
<td>3.89111</td>
<td>.4598352</td>
<td>8.46</td>
<td>0.000</td>
<td>2.989849</td>
</tr>
<tr>
<td>cov(( w, h ))</td>
<td>2.935574</td>
<td>.2904425</td>
<td>10.11</td>
<td>0.000</td>
<td>2.366317</td>
</tr>
<tr>
<td>cov(( w, y ))</td>
<td>6.826684</td>
<td>.743152</td>
<td>9.19</td>
<td>0.000</td>
<td>5.370133</td>
</tr>
<tr>
<td>( h )</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>var(( h ))</td>
<td>2.836497</td>
<td>.2711682</td>
<td>10.46</td>
<td>0.000</td>
<td>2.305017</td>
</tr>
<tr>
<td>cov(( h, y ))</td>
<td>5.772071</td>
<td>.561063</td>
<td>10.29</td>
<td>0.000</td>
<td>4.672408</td>
</tr>
<tr>
<td>( y )</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>var(( y ))</td>
<td>12.59876</td>
<td>1.296733</td>
<td>9.72</td>
<td>0.000</td>
<td>10.05721</td>
</tr>
</tbody>
</table>

Applied studies frequently look at the variances and covariances of a subset of the control variables in the model. Studies also frequently look at the covariances between a subset of the control variables in the model and the lags of one or two control variables. We illustrate how to obtain these results.
To view only the variance of the consumption \( c \) and its covariance with the control variables in the model, type

```
. estat covariance c, addcovariance(i r w h y)
```

Estimated covariances of model variables

| Coef. | Std. Err. | z    | P>|z| | [95% Conf. Interval] |
|-------|-----------|------|------|----------------------|
| \( c \) | | | | |
| \( \text{var}(c) \) | .8564586 | .1674129 | 5.12 | 0.000 | .5283354 1.184582 |
| \( \text{cov}(c,i) \) | 1.699536 | .432758 | 3.93 | 0.000 | .8513461 2.547726 |
| \( \text{cov}(c,r) \) | -.4153308 | .055347 | -7.50 | 0.000 | -.523809 -.3068527 |
| \( \text{cov}(c,w) \) | .9555354 | .198332 | 4.82 | 0.000 | .5688118 1.344259 |
| \( \text{cov}(c,h) \) | .0990768 | .0314055 | 3.15 | 0.002 | .0375231 1.1606305 |
| \( \text{cov}(c,y) \) | 1.054612 | .2293833 | 4.60 | 0.000 | .6050292 1.504195 |

Autocovariances are available as well. To view the variance of output and the first-order autocovariance of output, type

```
. estat covariance y, addcovariance(L.y)
```

Estimated covariances of model variables

| Coef. | Std. Err. | z    | P>|z| | [95% Conf. Interval] |
|-------|-----------|------|------|----------------------|
| \( y \) | | | | |
| \( \text{var}(y) \) | 12.59876 | 1.296733 | 9.72 | 0.000 | 10.05721 15.14031 |
| \( \text{cov}(y,L.y) \) | 4.367543 | 1.076622 | 4.06 | 0.000 | 2.257402 6.477685 |

To view the variances and covariance of output and consumption as well as their covariances with lagged output and lagged consumption, type

```
. estat covariance y c, addcovariance(L.y L.c)
```

Estimated covariances of model variables

| Coef. | Std. Err. | z    | P>|z| | [95% Conf. Interval] |
|-------|-----------|------|------|----------------------|
| \( y \) | | | | |
| \( \text{var}(y) \) | 12.59876 | 1.296733 | 9.72 | 0.000 | 10.05721 15.14031 |
| \( \text{cov}(y,c) \) | 1.054612 | .2293833 | 4.60 | 0.000 | .6050292 1.504195 |
| \( \text{cov}(y,L.y) \) | 4.367543 | 1.076622 | 4.06 | 0.000 | 2.257402 6.477685 |
| \( \text{cov}(y,L.c) \) | .9995189 | .2568548 | 3.89 | 0.000 | .4960927 1.502945 |

| Coef. | Std. Err. | z    | P>|z| | [95% Conf. Interval] |
|-------|-----------|------|------|----------------------|
| \( c \) | | | | |
| \( \text{var}(c) \) | .8564586 | .1674129 | 5.12 | 0.000 | .5283354 1.184582 |
| \( \text{cov}(c,y) \) | 1.054612 | .2293833 | 4.60 | 0.000 | .6050292 1.504195 |
| \( \text{cov}(c,L.y) \) | .9995189 | .1560805 | 6.40 | 0.000 | .6936067 1.305431 |
| \( \text{cov}(c,L.c) \) | .8420259 | .1655577 | 5.09 | 0.000 | .5175387 1.166513 |

What we typed requested the covariance between \( y \) and the lag of \( c \) (\( \text{cov}(y,L.c) \)) and between \( c \) and the lag of \( y \) (\( \text{cov}(c,L.y) \)). In general, these covariances can be different. The structure in this model causes their estimates to be the same, but their estimated standard errors are different. The estimated standard errors differ because the derivatives that enter the delta method are different.
Stored results

`estat covariance` stores the following in `r()`:

Matrices
- `r(b)` estimates
- `r(V)` variance–covariance matrix of the estimates

If `post` is specified, `estat covariance` also stores the following in `e()`:

Macros
- `e(properties) b V`

Matrices
- `e(b)` estimates
- `e(V)` variance–covariance matrix of the estimates

Methods and formulas

Entries in the covariance matrix are functions of the structural parameter vector $\theta$. Standard errors are calculated using the delta method.

Also see

[DSGE] `dsgenl` — Nonlinear dynamic stochastic general equilibrium models
[DSGE] `dsgenl postestimation` —[DSGE] Postestimation tools for dsgenl
[DSGE] `Intro 3e` — Nonlinear New Classical model
**Description**

`estat policy` displays the estimated policy matrix of the state-space form of a DSGE model.

**Quick start**

Display the estimated policy matrix

```
estat policy
```

As above, but with 90% confidence intervals

```
estat policy, level(90)
```

**Menu for estat**

Statistics > Postestimation

**Syntax**

```
estat policy [, compact post level(#) display_options ]
```

**Options**

- `compact` reports only the coefficient values of the estimated policy matrix and displays these coefficients in matrix form.
- `post` causes `estat policy` to behave like a Stata estimation (e-class) command. `estat policy` posts the policy matrix parameters along with the estimated variance–covariance matrix to `e()`, so you can treat the estimated policy matrix as you would results from any other estimation command.
- `level(#)` specifies the confidence level, as a percentage, for confidence intervals. The default is `level(95)` or as set by `set level`; see [U] 20.8 Specifying the width of confidence intervals.
- `display_options`: `noci`, `nopvalues`, `cformat(%,fmt)`, `pformat(%,fmt)`, `sformat(%,fmt)`, and `nolstretch`; see [R] Estimation options.
Remarks and examples

The policy matrix is part of the state-space form of a DSGE model. It specifies the model’s control variables as a function of the model’s state variables.

For examples, see [DSGE] Intro 1, [DSGE] Intro 3a, and [DSGE] Intro 3c.

Stored results

\texttt{estat policy} stores the following in \texttt{r()}:\newline

Matrices

\begin{verbatim}
  r(policy)          estimated policy matrix
  r(b)               estimates
  r(V)               variance–covariance matrix of the estimates
\end{verbatim}

If \texttt{post} is specified, \texttt{estat policy} also stores the following in \texttt{e()}:

Macros

\begin{verbatim}
  e(properties)     b V
\end{verbatim}

Matrices

\begin{verbatim}
  e(policy)         estimated policy matrix
  e(b)              estimates
  e(V)              variance–covariance matrix of the estimates
\end{verbatim}

Methods and formulas

Entries in the policy matrix \( G \) are functions of the structural parameter vector \( \theta \). Standard errors for \( \hat{G} \) are calculated using the delta method.

Also see

[DSGE] \texttt{dsge} — Linear dynamic stochastic general equilibrium models
[DSGE] \texttt{dsge postestimation} — Postestimation tools for dsge
[DSGE] \texttt{dsgenl} — Nonlinear dynamic stochastic general equilibrium models
[DSGE] \texttt{dsgenl postestimation} — Postestimation tools for dsgenl
[DSGE] \texttt{Intro 1} — Introduction to DSGEs
estat stable — Check stability of system

Description

estat stable displays functions of the model parameters that indicate whether the model is saddle-path stable at specific parameter values. These results can help you find initial values for which the model is saddle-path stable. Saddle-path stability is required for solving and estimating the parameters of DSGE models.

Menu for estat

Statistics > Postestimation

Syntax

estat stable

Remarks and examples

DSGE models are dynamic systems that are subject to random shocks. DSGE models that do not spiral out of control or converge to a single point when shocked are said to be “saddle-path stable”. We can solve for the state-space form of DSGE models only if they are saddle-path stable, and we can estimate the parameters of models only if we can solve for the state-space form.

The structural parameter values determine whether a DSGE model is saddle-path stable. In the process of solving the structural form for the state-space form, the Klein (2000) solver computes the generalized eigenvalues of a matrix formed from structural parameter values. An eigenvalue is said to be stable when its absolute value is less than 1. The model is saddle-path stable when the number of stable eigenvalues equals the number of states in the model.

estat stable displays the generalized eigenvalues implied by the parameter values in e(b), indicates which are stable and which are unstable, and displays a note indicating whether the model is saddle-path stable at these parameter values. estat stable can help you find initial values for the maximization routine when the default values imply a model that is not saddle-path stable; see [DSGE] Intro 5 for details.

Stored results

estat stable stores the following in r():

Scalars

r(stable) 1 if stable, 0 otherwise

Matrices

r(eigenvalues) eigenvalues
Methods and formulas

`estat stable` displays the generalized eigenvalues computed by the Klein (2000) solver. Values less than 1 are labeled as stable; values greater than or equal to 1 are labeled as unstable.

Reference


Also see

[DSGE] `dsge` — Linear dynamic stochastic general equilibrium models
[DSGE] `dsge postestimation` — Postestimation tools for dsge
[DSGE] `dsgenl` — Nonlinear dynamic stochastic general equilibrium models
[DSGE] `dsgenl postestimation` — Postestimation tools for dsgenl
[DSGE] `Intro 5` — Stability conditions
estat steady — Display steady state of nonlinear DSGE model

Description

estat steady displays the estimated steady-state values of variables in a nonlinear DSGE model.

Quick start

Display the estimated steady-state values

estat steady

As above, but with 90% confidence intervals

estat steady, level(90)

Menu for estat

Statistics > Postestimation

Syntax

estat steady [, compact level(#) display_options]

Options

compact reports only the coefficient values of the estimated steady-state vector.

level(#) specifies the confidence level, as a percentage, for confidence intervals. The default is level(95) or as set by set level; see [U] 20.8 Specifying the width of confidence intervals.

display_options: noci, nopvalues, cformat('%fmt'), pformat('%fmt'), sformat('%fmt), and nolstretch; see [R] Estimation options.

Remarks and examples

A nonlinear DSGE model is a system of equations describing the relationships among model variables. These relationships are dynamic and subject to random shocks. When the random shocks are set to zero, the model variables converge to fixed values over time. This vector of fixed values is known as the steady-state vector. The steady-state vector solves the system of equations when past and future values of each variable are replaced with their present value and expectations are dropped.
`estat steady` displays the estimated steady-state value of each variable.
For an example of `estat steady`, see [DSGE] Intro 3f.

### Stored results

`estat steady` stores the following in `r()`:  
Matrices  
- `r(steady)` estimated steady-state vector

### Methods and formulas

A nonlinear DSGE model can be written in the structural form  
\[ E_t \{ f(x_{t+1}, y_{t+1}, x_t, y_t; \theta) \} = 0 \]

The steady state is obtained by solving  
\[ f(x, y, x, y; \theta) = 0 \]

The resulting \((x, y)\) is a function of the parameter vector \(\theta\). Standard errors are calculated using the delta method.

### Also see

- [DSGE] `dsgenl` — Nonlinear dynamic stochastic general equilibrium models  
- [DSGE] `dsgenl postestimation` — Postestimation tools for dsgenl  
- [DSGE] Intro 3f — Stochastic growth model
### Description

`estat transition` displays the estimated state transition matrix of the state-space form of a DSGE model.

### Quick start

Display the estimated transition matrix

```
estat transition
```

As above, but with 90% confidence intervals

```
estat transition, level(90)
```

### Menu for estat

Statistics > Postestimation

### Syntax

```
estat transition [ , compact post level(#) display_options ]
```

### Options

- `compact` reports only the coefficient values of the estimated policy matrix and displays these coefficients in matrix form.
- `post` causes `estat transition` to behave like a Stata estimation (e-class) command. `estat transition` posts the state transition matrix to `e()`, so you can treat it as you would results from any other estimation command.
- `level(#)` specifies the confidence level, as a percentage, for confidence intervals. The default is `level(95)` or as set by `set level`; see [U] 20.8 Specifying the width of confidence intervals.
- `display_options`: `noci`, `nopvalues`, `cformat(%)`, `pformat(%)`, `sformat(%)`, and `nolstretch`; see [R] Estimation options.

### Also see

- `estat`
Remarks and examples

The state transition matrix is part of the state-space form of a DSGE model. It specifies the transition matrix of the model’s state variables.

For examples, see [DSGE] Intro 1, [DSGE] Intro 3a, and [DSGE] Intro 3b.

Stored results

estat transition stores the following in r():

Matrices
\[ r(\text{transition}) \] estimated transition matrix
\[ r(b) \] estimates
\[ r(V) \] variance–covariance matrix of the estimates

If post is specified, estat transition also stores the following in e():

Macros
\[ e(\text{properties}) \] b V

Matrices
\[ e(\text{transition}) \] estimated transition matrix
\[ e(b) \] estimates
\[ e(V) \] variance–covariance matrix of the estimates

Methods and formulas

Entries in the state transition matrix \( \mathbf{H} \) are functions of the structural parameter vector \( \theta \). Standard errors for entries in \( \hat{\mathbf{H}} \) are calculated using the delta method.

Also see

[DSGE] dsge — Linear dynamic stochastic general equilibrium models
[DSGE] dsge postestimation — Postestimation tools for dsge
[DSGE] dsgenl — Nonlinear dynamic stochastic general equilibrium models
[DSGE] dsgenl postestimation — Postestimation tools for dsgenl
[DSGE] Intro 1 — Introduction to DSGEs
Glossary

**autoregressive process.** An autoregressive process is a time series in which the current value of a variable is a linear function of its own past values and a white-noise error term. A first-order autoregressive process, denoted as an AR(1) process, is \( y_t = \rho y_{t-1} + \epsilon_t \). An AR\(p\) model contains \( p \) lagged values of the dependent variable.

**conditional mean.** A conditional mean expresses the mean of one variable as a function of some other variables. A regression function is a conditional mean.

**control variable.** A control variable is an endogenous variable. Control variables can be observed or unobserved.

  In a structural DSGE model, the current value of a control variable depends on the current value of other control variables, on the expected future values of any control variable, and on the current values of state variables. The current value of a control variable is found by solving the model for the state-space form.

**covariance stationary.** A covariance stationary process is a weakly stationary process.

**dynamic forecast.** A dynamic forecast uses forecasted values wherever lagged values of the endogenous variables appear in the model, allowing one to forecast multiple periods into the future.

**dynamic stochastic general equilibrium model.** A dynamic stochastic general equilibrium model is a multivariate time-series model that specifies the structural relationship between state variables and control variables and is typically derived from economic theory.

**endogenous variable.** An endogenous variable is a variable whose values are determined by the equilibrium of a structural model. The values of an endogenous variable are determined inside the system.

**equilibrium.** The equilibrium values of variables in a model are the values that satisfy all the model’s equations simultaneously.

**exogenous variable.** An exogenous variable is one whose values change independently of the other variables in a structural model. The values of an exogenous variable are determined outside the system. In a time-series model, an exogenous variable is also a predetermined variable.

**expected future value.** An expected future value is a forecast of the value of a variable in the future based on current information. In DSGE models, expected future values are computed under rational expectations.

  Under rational expectations, \( E_t(y_{t+1}) \) is the condition mean of \( y_{t+1} \) conditional on the complete history of all variables in the model and the structure of the model itself.

**forward operator.** The forward operator \( F \) denotes the value of a variable at time \( t + 1 \). Formally, \( Fy_t = y_{t+1} \), and \( F^2 y_t = Fy_{t+1} = y_{t+2} \). A forward operator is also called a lead operator.

**identified.** Identified is a condition required to estimate the parameters of a model. In other words, only identified parameters can be estimated.

  In DSGE models, the parameters are identified when there is a unique parameter vector that maximizes the likelihood function. For a discussion of identification, see [DSGE] Intro 6.

**impulse–response function.** An impulse–response function (IRF) measures the effect of a shock to an endogenous variable on itself or another endogenous variable. The \( k \)th impulse–response function of variable \( i \) on variable \( j \) measures the effect on variable \( j \) in period \( t + k \) in response to a one-unit shock to variable \( i \) in period \( t \), holding everything else constant.
independent and identically distributed. A series of observations is independent and identically distributed (i.i.d.) if each observation is an independent realization from the same underlying distribution. In some contexts, the definition is relaxed to mean only that the observations are independent and have identical means and variances; see Davidson and MacKinnon (1993, 42).

initial values. Initial values specify the starting place for the iterative maximization algorithm used by DSGE.

Kalman filter. The Kalman filter is a recursive procedure for predicting the state vector in a state-space model.

lag operator. The lag operator $L$ denotes the value of a variable at time $t - 1$. Formally, $Ly_t = y_{t-1}$, and $L^2y_t = Ly_{t-1} = y_{t-2}$.

lead operator. See forward operator.

likelihood-ratio (LR) test. The LR test is a classical testing procedure used to compare the fit of two models, one of which, the constrained model, is nested within the full (unconstrained) model. Under the null hypothesis, the constrained model fits the data as well as the full model. The LR test requires one to determine the maximal value of the log-likelihood function for both the constrained and the full models.

linearized model. A linearized model is an approximation to a model that is nonlinear in the variables and nonlinear in the parameters. The approximation is linear in variables but potentially nonlinear in the parameters. In a linearized model, variables are interpreted as unit deviations from steady state.

log-linear model. A log-linear model is an approximation to a model that is nonlinear in the variables and nonlinear in the parameters. In a log-linear model, variables are interpreted as percentage deviations from steady state.

model solution. A model solution is a function for the endogenous variables in terms of the exogenous variables. A model solution is also known as the reduced form of a model.

In DSGE terminology, a model solution expresses the control variables as a function of the state variables alone and expresses the state variables as a function of their values in the previous period and shocks. The reduced form of a DSGE model is also known as the state-space form of the DSGE model.

model-consistent expectation. A model-consistent expectation is the conditional mean of a variable implied by the model under consideration.

For example, under rational expectations the model-consistent expectation of $E_t(y_{t+1})$ is the mean of $y_{t+1}$ implied by the model, conditional on the realizations of variables dated time $t$ or previously.

nonpredetermined variable. A nonpredetermined variable is a variable whose value at time $t$ is determined by the system of equations in the model. Contrast with predetermined variable.

null hypothesis. In hypothesis testing, the null hypothesis typically represents the conjecture that one is attempting to disprove. Often the null hypothesis is that a parameter is zero or that a statistic is equal across populations.

one-step-ahead forecast. See static forecast.

policy matrix. The policy matrix in the reduced form of a DSGE model is the matrix that expresses control variables as a function of state variables.

predetermined variable. A predetermined variable is a variable whose value is fixed at time $t$, given everything that has occurred previously. More technically, the value of a predetermined variable is fixed, given the realizations of all observed and unobserved variables at times $t - 1, t - 2, \ldots$.
rational expectations. A rational expectation of a variable does not deviate from the mean of that variable in a predictable way. More technically, a rational expectation of a variable is the conditional mean of the variable implied by the model.

realization. The realization of a random variable is the value it takes on when drawn.

reduced form. The reduced form of a model expresses the endogenous variables as functions of the exogenous variables.

The reduced form of a DSGE model expresses the control variables as a function of the state variables alone and expresses the state variables as a function of their values in the previous period and shocks. The reduced form of a DSGE model is a state-space model.

saddle-path stable. A saddle-path stable model is a structural model that can be solved for its state-space form. The existence of a saddle-path stable solution depends on the parameter values of the model. For a discussion of saddle-path stability, see [DSGE] Intro 5.

shock variable. A shock variable is a random variable whose value is specified as an independently and identically distributed (i.i.d.) random variable. The maximum likelihood estimator is derived under normally distributed shocks but remains consistent under i.i.d. shocks. Robust standard errors must be specified when the errors are i.i.d. but not normally distributed.

state transition matrix. The state transition matrix in the reduced form of a DSGE model is the matrix that expresses how the future values of state variables depend on their current values.

state variable. A state variable is an unobserved exogenous variable.

In DSGE models, a state variable is an unobserved exogenous variable that may depend on its own previous value, the previous values of other state variables, and shocks.

state-space model. A state-space model describes the relationship between an observed time series and an unobservable state vector that represents the “state” of the world. The measurement equation expresses the observed series as a function of the state vector, and the transition equation describes how the unobserved state vector evolves over time. By defining the parameters of the measurement and transition equations appropriately, one can write a wide variety of time-series models in the state-space form.

For DSGE models, the state-space form is the reduced form of the structural model.

The DSGE framework changes the jargon and the structure of state-space models. The measurement equation is the vector of equations for the control variables, and the transition equation is the vector of equations for the state variables. In contrast to the standard state-space model, DSGE models allow a control variable to be unobserved.

static forecast. A static forecast uses actual values wherever lagged values of the endogenous variables appear in the model. As a result, static forecasts perform at least as well as dynamic forecasts, but static forecasts cannot produce forecasts into the future when lags of the endogenous variables appear in the model.

Because actual values will be missing beyond the last historical time period in the dataset, static forecasts can forecast only one period into the future (assuming only first lags appear in the model); thus they are often called one-step-ahead forecasts.

steady-state equilibrium. A steady-state equilibrium is a time-invariant rest point of a dynamic system.

More technically, a steady-state equilibrium is a set of values for the endogenous variables to which the dynamic system will return after an exogenous variable is changed or a random shock occurs. This set of values is time invariant in that it does not depend on the time period in which the change or shock occurs. Multistep dynamic forecasts converge to these values. A steady-state
equilibrium is also known as a long-run equilibrium because it specifies time-invariant values for the endogenous variables to which the dynamic system will return, if left unshocked.

**stochastic equation.** A stochastic equation, in contrast to an identity, is an equation in a forecast model that includes a random component, most often in the form of an additive error term. Stochastic equations include parameters that must be estimated from historical data.

**stochastic trend.** A stochastic trend is a nonstationary random process. Unit-root process and random coefficients on time are two common stochastic trends. See [TS ucm](#) for examples and discussions of more commonly applied stochastic trends.

**strict stationarity.** A process is strictly stationary if the joint distribution of \( y_1, \ldots, y_k \) is the same as the joint distribution of \( y_{1+\tau}, \ldots, y_{k+\tau} \) for all \( k \) and \( \tau \). Intuitively, shifting the origin of the series by \( \tau \) units has no effect on the joint distributions.

**structural model.** A structural model specifies the theoretical relationship among a set of variables. Structural models contain both endogenous variables and exogenous variables. Parameter estimation and interpretation require that structural models be solved for a reduced form.

**trend.** The trend specifies the long-run behavior in a time series. The trend can be deterministic or stochastic. Many economic, biological, health, and social time series have long-run tendencies to increase or decrease. Before the 1980s, most time-series analysis specified the long-run tendencies as deterministic functions of time. Since the 1980s, the stochastic trends implied by unit-root processes have become a standard part of the toolkit.

**Wald test.** A Wald test is a classical testing procedure used to compare the fit of two models, one of which, the constrained model, is nested within the full (unconstrained) model. Under the null hypothesis, the constrained model fits the data as well as the full model. The Wald test requires one to fit the full model but does not require one to fit the constrained model.

**weakly stationary.** A process is weakly stationary if the mean of the process is finite and independent of \( t \), the unconditional variance of the process is finite and independent of \( t \), and the covariance between periods \( t \) and \( t-s \) is finite and depends on \( t-s \) but not on \( t \) or \( s \) themselves. Weakly-stationary processes are also known as covariance stationary processes.

**white noise.** A variable \( u_t \) represents a white-noise process if the mean of \( u_t \) is zero, the variance of \( u_t \) is \( \sigma^2 \), and the covariance between \( u_t \) and \( u_s \) is zero for all \( s \neq t \).

**Reference**

Subject and author index

See the combined subject index and the combined author index in the Glossary and Index.