Demystifying DSGE Models

John Cuddy

The Graduate Institute

Geneva

Lab Session – Dynare Intro

All you need to know to get started with Dynare

1. Installing Dynare

Dynare is a collection of Matlab routines.

It reads in a system of DSGE model equations, solves/estimates the model for a given set of parameter values and delivers useful output.

To get ***Dynare*** , go to http://www.dynare.org/download and download the latest stable version: the file should be dynare-*-win.exe, where * is the version number.

Install it by running the .exe file. Let it install to its default folders, typically (but not necessarily) C:\dynare*, where * is the version number.

All the Matlab routines needed to run ***Dynare*** are in C:\dynare*\matlab.

After installing ***Dynare*** , Matlab needs to be directed to recognise ***Dynare*** files.

That is, the path of the ***Dynare*** Matlab files needs to be set in Matlab.

In Matlab, click ***File → SetPath***, then click ***Add Folder***..., and choose as the folder, ***C:\dynare*\matlab*** (assuming ***Dynare*** was installed to this directory).

All the ***Dynare*** files should now appear in the Matlab search path.

Now click ***Save*** and ***Close***.

To test if ***Dynare*** has installed properly, type " dynare " (without the "") into the Matlab command line.

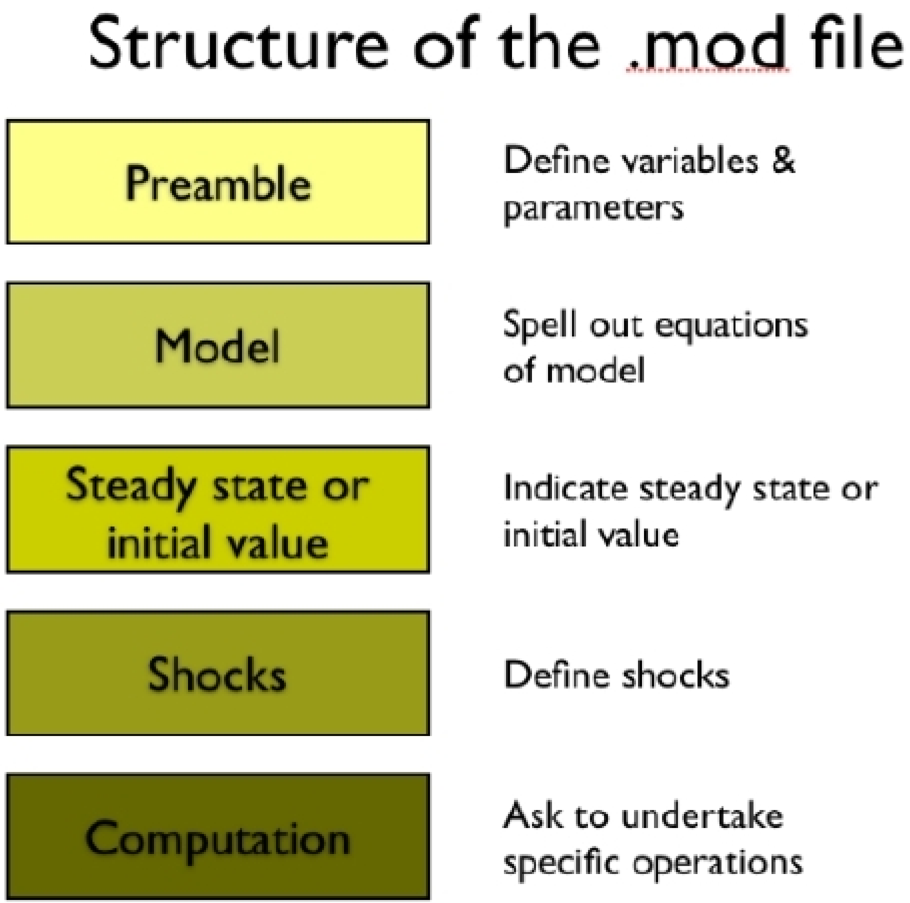
If it is installed correctly, Matlab will print the error message
"Error using dynare (line 36) Not enough input arguments."

If not installed correctly, Matlab will print the error message
"Undefined function or variable 'dynare'" .

2. Writing a model in Dynare

Dynare reads in .mod text files and parses these into Matlab files, which then call on the ***Dynare*** Matlab routines to do the model analysis.

The .mod file contains ***five distinct parts*** :



Preamble

The very first element of a ***Dynare*** programme is the Preamble.

It consists of ***four*** parts:

Part 1 .

First, it is necessary to define the ***endogenous variables*** which appear in the model.

The syntax for writing this in ***Dynare*** is **var [variable1, variable2, …];** with the line ended by a semicolon.

Note that here (and later) the square brackets [ ] are not used, and the commas " , " are not necessary.

For example

var y, c, i, k, n, r, a; ***OR*** **var y c i k n r a;**

It is possible to enter line(s) of comment by prefixing the text by **%** or **//**.

For example: **%** list of endogenous variables

Part 2 .

Next, one writes the ***exogenous*** (usually shock) variables in the model.

The syntax for writing this in ***Dynare*** is **varexo [variable1, variable2, …];**

with the line ended by a semicolon.

For example

varexo e;

Part 3 .

Next come the ***parameters*** of the model.

The syntax for writing this in ***Dynare*** is **parameters [parameter1, parameter2, …];** with the line ended by a semicolon.

For example

parameters alpha beta delta eta a rho;

Part 4 .

Finally, one gives ***initial values*** to the parameters.

The syntax for doing so is **[parameter] = [value];** which is similar to normal Matlab code. For example

```
alpha = 0.33;
rho   = 0.95;
beta  = 0.99;
delta = .015;
```

Model

In the next (“Model”) part of a ***Dynare*** programme, one writes the equations of the (usually linearised) model.

The equations are written in generally the same way as they appear in an academic paper.

The section starts with

model (linear) ;

and is completed by

end ; .

If the model is nonlinear, then only **model;** is used.

A few syntax considerations:

1. Variables which are ***multiplied (divided)*** by a parameter are separated from the parameter with * (/)

2. ***Contemporaneous*** variables do ***not*** have a time subscript in the ***Dynare*** code.
   Lagged variables x_{t-i} are written as x(-i)
   Expectational variables $E_t\{x_{t+i}\}$ are written as x(+i)
3. Equations are ended by a semicolon ;

For example:

```
model(linear);
  y = (Css_Yss)*c + (Iss_Yss)*i;
  y = a + alpha*k + (1 - alpha)*n;
  k(+1) = (Iss_Kss)*i + (1 - delta)*k;
  n = y - eta*c;
  c = c(+1) - (1/eta)*r(+1);
  r = (alpha*(Yss_Kss)/Rss)*(y - k);
  a = rho*a(-1) + e;
end;
```

One next writes **resid;** to have ***Dynare*** calculate the residuals of the static versions of the model's equations using the assigned parameter values.

These residuals should be zero if the model is internally consistent.

This completes the Model part of the mod-file.

Initial Values

After specifying the equations, one must tell ***Dynare*** where to begin simulations or impulse responses: the initial values of the variables.

This is done in a way similar to specifying the parameter initial values, but

- beginning the section with **initval ;**
- ending the section with **end ;**
- and setting the variable to a value.

Usually the initial values correspond to the ***steady-state*** of the model.

If the model is non-linear or complicated, it may require some effort (or even a separate Matlab routine) to obtain these steady-state values.

In cases where endogenous variables are transformed so as to enter the ***Dynare*** routines as deviations from their steady-state values, the steady-state of the ***transformed*** (***model)*** variables is by definition ***zero*** .

In this case, there is ***no need*** to specify an initial value for the variable, since ***Dynare*** by default will assume an initial value of zero.

If some variables are not so transformed, then a non-zero initial value is necessary for them.

Following this, one writes

steady;

which causes ***Dynare*** to calculate the model's steady -state, and

check;

which causes ***Dynare*** to check the Blanchard-Kahn conditions which must be satisfied for a model solution to be obtained.

The **Initial Values** part of the mod-file is then finished. If all goes well with ***steady*** and ***check***, Matlab will continue to process the model; if not, it will ***stop*** .

Shocks

In order to simulate the model, it is necessary to

- define the (***i.i.d.***) shocks, and
- set the non-zero elements of the shocks' variance -covariance matrix to the specified (or estimated, if the simulation follows an eventual estimation) parameters.

This section begins with

shocks ;

and finishes with

end ;

The syntax for specifying each shock is

var [shock] = [shock variance] ; or **var [shock]; stderr [shock sd];**

For example, two equivalent shocks may (since ***var = stderr***^{2}) be written ***as***

shocks;
var e; stderr 0.1;
end;

or

shocks;
var e = 0.01;
end;

Note that if the model is log-linearised around the steady-state, a shock with a variance of 0.01 is equivalent to a 1% shock.

Note also that if the shock in your model is ***not i.i.d.,*** but, say ***AR(1),*** then it will be necessary to add to the model's ***equations*** a "shock ***process***" definition , for example

eps_e = rho_e*eps_e(-1) + e;

Note that this will also require

- adding an endogenous variable (“eps_e”) to the list in the ***var*** section, and
- a parameter (“rho_e”) to the ***parameters*** section.

The Shock section remains the same as in the ***i.i.d*** . case, since the shock **process** includes an ***i.i.d.*** shock “e”.

This ends the Shock section of the mod-file.

Computation

Finally, the model is ***solved*** , to obtain ***Dynare*** 's default analytics.

This is done by using the

stoch_simul;

command without specifying any further options.

There are many options which can be used with stoch_simul, which can be found in the ***Dynare*** manual.

The default (without options) produces IRFs of 24 periods in length.

If a list of variables is specified after **stoch_simul** (and before the semi-colon) then results are displayed only for those variables and in the order in which they are listed.

3. Running a model in Dynare

To run the ***Dynare*** .mod file created, save the text file with a " .mod " extension in a directory (for example " \mymodels ")

which must be set to be Matlab's ***current directory***

For example, save the file as " \mymodels\test.mod ".

To initiate ***Dynare*** , in Matlab's command line, type

>> **dynare test.mod**

Dynare will then go to work.

It reads in the .mod contents, recognising the model's details, and solves for the rational expectations solution.

Dynare will print a lot of analytical information to the Matlab command window, save much of that to the workspace, and generate figures which plot impulse responses to the shocks for each variable.

It also saves everything that it generates to the working directory.

It is worth checking that the number of equations ***Dynare*** finds in the .mod file is the same as the number of endogenous variables in the model. For example if there are **seven** variables:

```
Starting Dynare (version 5.4).
Calling Dynare with arguments: none
Starting preprocessing of the model file ...
Found 7 equation(s).
Evaluating expressions...done
Computing static model derivatives (order 1).
Computing dynamic model derivatives (order 1).
Processing outputs ...
done
Preprocessing completed.
```

Next, ***Dynare*** will print the ***residuals*** of the static equations, the ***steady-state*** values for all the variables, the results of the ***Blanchard-Kahn*** check, and a ***summary*** of the model variables:

Residuals of the static equations:

Equation number 1 : 0 : Resource Constraint
Equation number 2 : 0 : Production Function
Equation number 3 : 0 : Law of motion of capital
Equation number 4 : 0 : Labour FOC
Equation number 5 : 0 : Euler equation
Equation number 6 : 0 : real interest rate/firm FOC capital
Equation number 7 : 0 : exogenous TFP process

The equation residuals are all **0**, so the model is internally consistent

STEADY-STATE RESULTS:

y	**0**
c	**0**
i	**0**
k	**0**
n	**0**
r	**0**
a	**0**

In this model, all variables are expressed as deviations from their steady-state values, so the steady-states of the ***transformed*** (***model)*** variables are by definition ***zero***

```
EIGENVALUES:
         Modulus                   Real          Imaginary

            0.95                   0.95                  0
           0.952                  0.952                  0
           1.061                  1.061                  0
       1.132e+16             -1.132e+16                  0


There are 2 eigenvalue(s) larger than 1 in modulus
for 2 forward-looking variable(s)

The rank condition is verified.


MODEL SUMMARY

  Number of variables:         7
  Number of stochastic shocks: 1
  Number of state variables:   2
  Number of jumpers:           2
  Number of static variables:  3
```

Next, ***Dynare*** generates the ***variance-covariance matrix*** of the shocks.

The values in the diagonals correspond to the specified variances in the .mod file.

If there are more than one shock, the covariances across the shocks are shown on the off-diagonal elements; these will be zero unless non-zero covariances have been specified in the **shocks** section (see the ***Dynare*** manual on how to do this).

For example, if there were four non-correlated shocks eps_..., the matrix would look like

```
MATRIX OF COVARIANCE OF EXOGENOUS SHOCKS
Variables        eps_r      eps_a      eps_e      eps_z
eps_r         0.000010   0.000000   0.000000   0.000000
eps_a         0.000000   0.001640   0.000000   0.000000
eps_e         0.000000   0.000000   0.000001   0.000000
eps_z         0.000000   0.000000   0.000000   0.000119
```

Dynare next computes the ***solution*** of the model as: $\hat{y}_t = A\hat{y}_{t-1} + Bu_t$ where $\hat{y}_t$ is a vector of endogenous variables as a deviation from steady-state and u_t is a vector of the exogenous shocks.

The first two lines of the output below correspond to the non-zero rows of matrix A^T, while the last row corresponds to matrix B^T.

POLICY AND TRANSITION				FUNCTIONS			
y	c	i	k	n	r	a	
k(-1)	-0.028	0.506	-2.202	0.952	-0.534	-0.026	0
a(-1)	2.035	0.416	8.628	0.129	1.619	0.051	0.95
e	2.142	0.437	9.082	0.136	1.705	0.053	1

The next section of ***Dynare*** output reports the ***moments*** of the model, using the model solution and the specified standard errors of the exogenous shock(s)

THEORETICAL MOMENTS

VARIABLE	MEAN	STD. DEV.	VARIANCE
y	0.0000	0.6741	0.4544
c	0.0000	0.4298	0.1847
i	0.0000	2.2064	4.8682
k	0.0000	0.6358	0.4043
n	0.0000	0.4008	0.1606
r	0.0000	0.0133	0.0002
a	0.0000	0.3203	0.1026

Next, in the case when there are more than one shock, ***Dynare*** reports the ***variance decomposition*** of the endogenous variables.

This gives the ***proportion*** of the variance in the endogenous variable which can be ***attributed*** to each of the exogenous shocks, so the sum for each variable across the four shocks is 100%.

For example, for one particular model with four shocks, at the calibrated parameter values, **63.5%** of the change in inflation (“ pi ”) following shocks is due to “e” shocks and **29.4%** to “r” shocks, but none to “z” shocks (see last line)

VARIANCE DECOMPOSITION (in percent)

	eps_r	eps_a	eps_e	eps_z
y	7.08	8.67	84.25	0.00
x	7.55	2.53	89.91	0.00
g	29.12	20.00	16.69	34.19
r	12.42	45.99	41.60	0.00
pi	$\boxed{29.43}$	7.03	$\boxed{63.54}$	0.00

Dynare then reports a matrix of theoretical ***correlations*** across the endogenous variables.

These values are constructed from the variance-covariance matrix of the endogenous variables, which ***Dynare*** does not output to the command window (but which it saves in a matrix called "oo_.var" buried deep in the Matlab workspace, together with much else – see section 4 below)

MATRIX OF CORRELATIONS

Variables	y	c	i	k	n	r	a
y	1.0000	0.8258	0.8944	0.7040	0.7964	0.4665	0.9998
c	0.8258	1.0000	0.4864	0.9819	0.3166	-0.1136	0.8365
i	0.8944	0.4864	1.0000	0.3120	0.9828	0.8128	0.8857
k	0.7040	0.9819	0.3120	1.0000	0.1311	-0.2998	0.7175
n	0.7964	0.3166	0.9828	0.1311	1.0000	0.9065	0.7846
r	0.4665	-0.1136	0.8128	-0.2998	0.9065	1.0000	0.4494
a	0.9998	0.8365	0.8857	0.7175	0.7846	0.4494	1.0000

Next, ***Dynare*** shows coefficients of ***autocorrelation*** up to the fifth order, that is, the correlation of an endogenous value with p-order lags of itself, where p = {1,...,5}

COEFFICIENTS OF AUTOCORRELATION

Order	1	2	3	4	5
y	0.9482	0.8990	0.8524	0.8082	0.7663
c	0.9944	0.9870	0.9779	0.9673	0.9554
i	0.9111	0.8285	0.7519	0.6807	0.6147
k	0.9987	0.9952	0.9896	0.9822	0.9731
n	0.9042	0.8155	0.7332	0.6571	0.5867
r	0.9132	0.8325	0.7575	0.6879	0.6232
a	0.9500	0.9025	0.8574	0.8145	0.7738

Finally, ***Dynare*** will generate a number of ***impulse response*** (IRF) graphics for the response of the endogenous variables to each of the exogenous shocks.

Dynare filters out impulses which are essentially zero across the horizon.

On each figure, it will plot at most 9 impulse responses; if the model has more than 9 endogenous variables, it will plot the 10th+ variables on subsequent Matlab figure

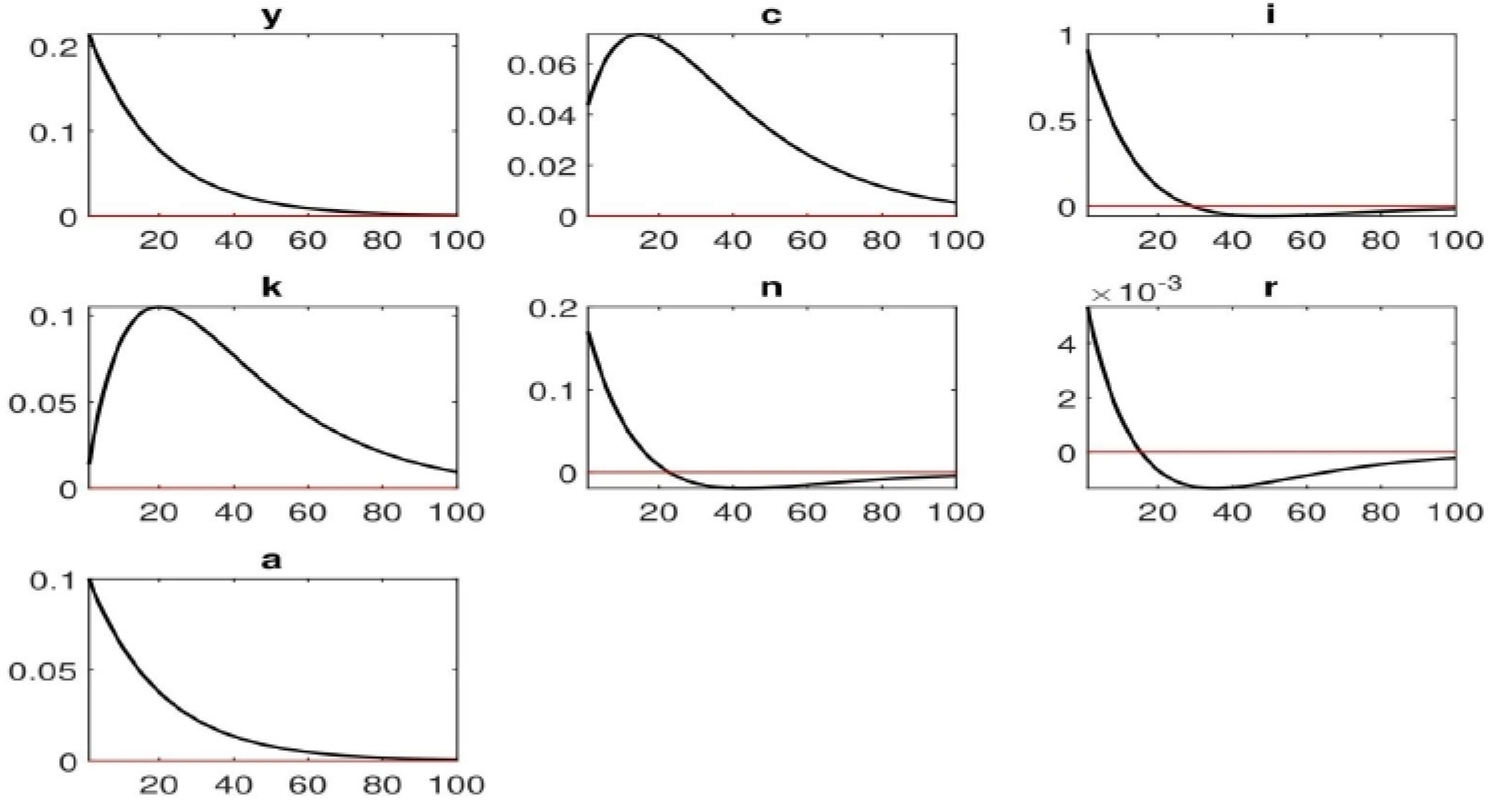


Figure 1: Impulse response functions (orthogonalized shock to e).

4. The Matlab Workspace generated by Dynare

The Table below gives information on some of what is stored by ***Dynare*** to the Matlab workspace. More generally, ***Dynare*** stores to the workspace all the parameter values and impulse responses of all endogenous variables to shocks

Table 1: **The Matlab Workspace**

Stored in	What
`oo_.exo_simul`	Simulated series of the exogenous variables (if simulating).
`oo_.endo_simul`	Simulated series of the endogenous variables (if simulating).
`oo_.dr`	Stored information about variables and policy function.
`oo_.dr.order_var`	The mapping from the declaration order to the decision rule order.
`oo_.dr.inv_order_var`	The mapping from the decision rule order to the declaration order.
`oo_.dr.ghx`	The A matrix of the transition function.
`oo_.dr.ghu`	The B matrix of the transition function.
`oo_.steady_state`	Values of the endogenous variables' steady-state.
`oo_.gamma_y{7,1}`	The variance decomposition outputted to the command window.
`oo_.mean`	The mean of endogenous variables.
`oo_.var`	The variance-covariance matrix.
`oo_.autocorr`	The autocorrelation matrices for endogenous variables; to 5 lags.
`oo_.irfs`	Impulse responses of endogenous variables to exogenous shocks.
`oo_.irfs.x_y`	Impulse response of x to an innovation in y.
`oo_.forecast`	Forecasts of endogenous variables.
`M_`	Structure with information about the model.
`M_.endo_names`	The names of endogenous variables in declaration order.
`M_.exo_names`	The names of exogenous variables in declaration order.
`M_.exo_nbr`	The number of exogenous variables.
`M_.endo_nbr`	The number of endogenous variables.
`options_`	Structure with all the options which Dynare recognises.

5. Writing L A T E X from Dynare

Dynare has in-built functions to print easily the model equations and estimation results to a L A T E X file.

This is ***not*** required, but if you want the L A T E X output, then for ***Dynare*** to know what L A T E X variables to use, you need to define the L A T E X representation of each variable and parameter.

This is done in the ***Preamble*** section, after each variable, in dollar signs. For example:

var r r, pi π, …
varexo eps_r ε^r, …
parameters beta β, alpha α, eta η, ...

It is also possible to add ***descriptive text*** to the variables/parameters, which L A T E X will use in creating its reports, for example:

parameters
alpha ${\alpha}$ (long_name='capital share')
beta ${\beta}$ (long_name='discount factor')
…

After declaring L A T E X equivalent variables and parameters, include the option "Tex," in the **stoch_simul** command and at the end of the ***Dynare*** model code add the commands:

write_latex_dynamic_model ;
write_latex_parameter_table;
write_latex_definitions;
collect_latex_files;

to generate a .tex file with name MODELNAME_tex_binder.tex where MODELNAME is the name of the .mod file.

This .tex file can then be compiled to produce a .pdf report including the L A T E X versions of the model, definitions, parameters and various tables automatically generated by ***Dynare*** , including (importantly) the IRF graphs