Introduction to Dynare and Examples

Advanced Macroeconomics I
CAEN/UFC - March 2019

Prof. Marcelo Arbex

What is Dynare?

- Dynare is a Matlab front end to solve and simulate dynamic models
- Either deterministic or stochastic
- Developed by Michel Juillard at CEPREMAP
- website: `http://www.dynare.org/`

What is Dynare?

- Dynare runs inside the computation platform MATLAB, or the open-source alternative Octave.
- Both Dynare and Octave are available for free download, the former running under MATLAB or Octave on all platforms and the latter in separate versions for Windows, Mac, or Linux.
- You will need to run your simulations on a computer on which Dynare and either MATLAB or Octave are installed (Read instruction in readme file to add Dynare to Matlab.)

How does it work?

1. Input your model, parameter values, and instructors to Dynare through a text file with a .mod suffix.
    - The Quick Start guide, Tutorial, User Guide, and Manual available at the Dynare Web site
    - website: `http://www.dynare.org/` provide details.
2. Template for several .mod files so you will be able to run the code
3. Note that the .mod file must be in plain text format. You can use a simple text editor such as Notepad to create such a file.
    - If you instead use Word or another advanced word processor, you must set the file format to plain text when you save or Dynare won't be able to read your file.

Structure of .mod file

The .mod file consists of five parts:

- The **preamble** section
  - defines your variables and parameters.
- The **model** section
  - contains the equations of your model in a simple algebraic notation similar to standard programming languages.
- The **steady state** section
  - gives the steady-state (initial) values of the model solution.
- The **shocks** section
  - in which you tell Dynare about the deterministic or random shocks that you want to simulate.
- The **computation** section
  - tells Dynare to simulate the model and report the results.

How does it work?

Before we get into details, note several features of the Dynare program:

- Variables are case-sensitive.
- All commands end with a semi-colon.
- Blocks of commands such as model equations, steady-state (initial) values, and shocks end with the `end;` statement.
- Options for any command are entered in parentheses immediate after the name of the command.

General Structure

* Preamble
  - Declaration of endogenous, exogenous variables and parameters
  - Assignment of parameter values

* Declaration of model
  - Start: model;
  - End: end;
  - In between: all equilibrium conditions

* Initial conditions
  - Start: initval
  - Provide initial conditions for steady state of model
  - End: end

General Structure

* Specify shocks
  - Start: Shocks
  - Define (all non-zero entries of) Variance-Covariance matrix of shocks
  - End: end

* Solution of Model
  - Use command stoch simul
  - More on options later

Timing Convention

- Pre-determined variables (e.g., capital stock) dated t-1 in time t equation
- Way to tell Dynare which variables are state variables
- Need to rewrite set of equations
- Lag capital stock in all equations

More on conventions

* Timing conventions
  - If variable x is decided in period t, write x
  - If variable x is decided in period t - 1, write x(-1)
  - If variable x is decided in period t + 1, write x(+1)
  - If variable x is decided in period t + 2, introduce auxiliary variable

* Solutions
  - Dynare default: linear approximation of levels of variables
  - Linear approx. in logs convenient as IRFs are in percentage terms
  - Define variables as exp(x)
  - Now x will be interpreted as log of variable

Structure of the mod file: Preamble

Assume your model takes the form

$$x_t = \rho x_{t-1} + e_t$$

with $e_t \sim N(0, \sigma^2)$

- Variable: x_t
- Exogenous Variable: e_t
- Parameters: ρ and σ^2
  - If x_t captures technology, ρ is a parameter capturing the persistence of technological progress

Structure of the mod file: Preamble

Define variables and parameters

3 major instructions:

1. `var`: Define variables
2. `varexo`: Define (truly) exogenous variables
3. `parameters`: Declare parameters
    - ... assign values to parameters

Structure of the mod file: Preamble

An example

```
// Simple AR(1) model
// This version:  01/16/17
var x;
varexo e;
parameters rho,sigma;
rho       = 0.90;
sigma     = 0.01;
```

Structure of the mod file: Model

Define model equations

1 major instruction:

```
model;
....
end;
```

Write equations as they appear in *natural* language

Structure of the mod file: Model

An example

AR(1) example: $x_t = \rho x_{t-1} + e_t$

Model writes:

```
model;
x=rho*x(-1)+e;
end;
```

Structure of the mod file: Steady State

Compute the long-run of the model

- That is: Where its deterministic dynamics will converge
- Why? Because it will take a (non-)linear approximation around this long run

Structure:

```
initval;
...
end;
steady;
check;
```

Structure of the mod file: Steady State

Steady computes the long run of the model using a non-linear solver

- Close to the Newton algorithm (more sophisticated though!)

It therefore needs initial conditions

- That's the role of the `initval;... end;` `statement`.

Better give (very) good initial conditions for all variables

Structure of the mod file: Steady State

What if you forget `steady`

- It will not compute the steady state.

What happens then?

- Simulations start from values specified in the `initval;... end;` `statement`.

`check` is optional. It checks the dynamic stability of the system

Structure of the mod file: Steady State

An example

AR(1) example: $x_t = \rho x_{t-1} + e_t$

In deterministic steady state: $e_t = e^* = 0$

$$x^* = \rho x^* \implies x^* = 0$$

```
initval;
e     = 0;
x   = 0;
end;
steady;
check;
```

Structure of the mod file: Shocks

Define the properties of the exogenous shocks

- Exogenous shocks are Gaussian innovations with $N(0, \Sigma)$

Structure:

```
shocks;
var ...;
stderr ...;
end;
```

Structure of the mod file: Shocks

AR(1) example: $x_t = \rho x_{t-1} + e_t$

```
shocks;
var e;
stderr se;
end;
```

Structure of the mod file: Solution

Final step: Compute the solution and produce some output

Solution method

- Deterministic model: Relaxation method
- Stochastic model: First or Second order perturbation method

Then compute some moments and impulse responses.

```
stoch_simul(...)  ...;
```

Structure of the mod file: Solution

AR(1) example: $x_t = \rho x_{t-1} + e_t$

Therefore (because the model is linear)

```
stoch_simul(linear);
```

Structure of the mod file: Solution

Options of the stoch_simul option

Solver

- `linear`: In case of a linear model.
- `order = 1 or 2` : order of Taylor approximation (default $= 2$).

Output (prints everything by default)

- `noprint`: cancel any printing.
- `nocorr`: doesn't print the correlation matrix.
- `nofunctions`: doesn't print the approximated solution.
- `nomoments`: doesn't print moments of the endogenous variables.
- `ar = INTEGER`:
  Order of autocorrelation coefficients to compute (5)

Structure of the mod file: Solution

Impulse Response Functions

- `irf = INTEGER`: number of periods on which to compute the IRFs (Setting `IRF=0`, suppresses the plotting of IRFs).
- `relative irf` requests the computation of normalized IRFs in percentage of the standard error of each shock.

Simulations

- `periods = INTEGER`: specifies the number of periods to use in simulations (default $= 0$).
- `replic = INTEGER`: number of simulated series used to compute the IRFs (default $= 1$ if order $= 1$, and 50 otherwise).
- `drop = INTEGER`: number of points dropped in simulations (default $= 100$).

```
// AR(1) model
// Name of the variable
var x;
// Name of the exogenous variable
varexo e;
// Parameters of the model
parameters rho se;
rho = 0.95;
se = 0.02;
model;
x = rho*x(-1)+e;
end;
initval;
e=0;
x=0;
end;
steady;
check;
shocks;
var e; stderr se;
end;
stoch_simul(linear);
```

STEADY-STATE RESULTS:

x 0

EIGENVALUES:

Modulus	Real	Imaginary
0.95	0.95	0

There are 0 eigenvalue(s) larger than 1 in modulus
for 0 forward-looking variable(s)
The rank condition is verified.

MODEL SUMMARY

Number of variables: 1
Number of stochastic shocks: 1
Number of state variables: 1
Number of jumpers: 0
Number of static variables: 0

MATRIX OF COVARIANCE OF EXOGENOUS SHOCKS

Variables	e
e	0.000400

POLICY AND TRANSITION FUNCTIONS

	x
x(-1)	0.950000
e	1.000000

THEORETICAL MOMENTS

VARIABLE	MEAN	STD. DEV.	VARIANCE
x	0.0000	0.0641	0.0041

MATRIX OF CORRELATIONS

Variables	x
x	1.0000

COEFFICIENTS OF AUTOCORRELATION

Order	1	2	3	4	5
x	0.9500	0.9025	0.8574	0.8145	0.7738

Samuelson's accelerator model

Consider the Samuelson's accelerator model (backward looking model)

$$C_t = \beta Y_{t-1} \qquad G_t = \rho G_{t-1} + (1-\rho)\bar{G} + \eta_t$$
$$I_t = \alpha(C_t - C_{t-1}) \qquad Y_t = C_t + I_t + G_t$$

- Consumption is 80% of income
- Government spending have a strong persistence (0.9)
- $\bar{G} = 1$
- Investment in period t equals 5% more than the variation in consumption between t and $t-1$
- $\eta_t \sim iid\ N(0, 0.02^2)$.

Samuelson's accelerator model

- Variables: C_t, I_t, Y_t, G_t
- Exogenous Variable: η_t
- Parameters: α, β, ρ, σ

A) Write the Dynare code to perform a stochastic simulation of this model for 2000 periods and generate impulse response functions for the endogenous variables for 60 periods.

B) Present the matrix of correlation coefficients of the endogenous variables.

C) (make sure the model is stable!) Try run the code with $\beta = 2$

Forward looking models

Consider a (linear) rational expectations model

- postpone theory to tomorrow!

The model writes

$$y_t = aE_t y_{t+1} + bx_t$$

$$x_t = \rho x_{t-1} + \varepsilon_t$$

with $\varepsilon_t \sim N(0, \sigma^2)$

- Variable: y_t, x_t
- Exogenous Variable: ε_t
- Parameters: a, b, ρ and σ

Forward looking models

- ε_t is exogenous
- x_t is a predetermined variable
- y_t is a jump variable (it's a control variable)

$$y_t = \sum_{j=0}^{\infty} a^j E_t x_{t+j}$$

* Blanchard and Kahn (Econometrica 1980)
* Fundamentally forward looking model!
* Dynare knows how to solve it!

Backward-Forward looking models

Mix of jump and predetermined (endogenous) variables

The model writes

$$E_t y_{t+1} - (\lambda + \mu) y_t + \lambda \mu y_{t-1} = b x_t$$

$$x_t = \rho x_{t-1} + \varepsilon_t$$

with $\varepsilon_t \sim N(0, \sigma^2)$

- Variable: y_t, x_t
- Exogenous Variable: ε_t
- Parameters: λ, μ, b, ρ and σ

Backward-Forward looking models

- ε_t is exogenous
- x_t is a predetermined variable
- y_t is a jump variable but it has also a predetermined component.

  * Blanchard and Kahn (Econometrica 1980) again!

Deterministic vs. Stochastic models

The important question: **is your model stochastic or deterministic?**

- The distinction hinges on whether future shocks are known.

Deterministic models: the occurrence of all future shocks is known exactly at the time of computing the model's solution.

Stochastic models: only the distribution of future shocks is known.

Deterministic vs. Stochastic models

Consider a shock to a model's innovation only in period 1.

In a deterministic context

- Agents will take their decisions knowing that future values of the innovations *will be zero in all periods to come*.

In a stochastic context

- Agents will take their decisions knowing that the future value of innovations *are random but will have zero mean.*

The solution method for each of these model types differs significantly.

Deterministic vs. Stochastic models

In deterministic models

- A highly accurate solution can be found by numerical methods.
- Solution: a series of numbers that match a given set of equations.

If an agent has perfect foresight

- Agent can specify today - at the time of her decision - what each of her precise actions will be in the future.

Deterministic vs. Stochastic models

Stochastic environment

- The best the agent can do is specify a decision, policy or feedback rule for the future:
  - What will her optimal actions be contingent on each possible realization of shocks.
- We search for a function satisfying the model's first order conditions.
  - This function may be non-linear and thus needs to be approximated.

Deterministic models have the following characteristics:

1. DSGE literature has gained attention in economics, **deterministic models have become somewhat rare**. Examples include OLG models without aggregate uncertainty.

2. Solution does not require linearization.

- It doesn't even really need a steady state.
- Numerical simulation to find the exact paths of endogenous variables that meet the model's first order conditions and shock structure.
- This solution method can therefore be useful when the economy is far away from steady state (when linearization offers a poor approximation).

Deterministic models have the following characteristics:

3. These models are usually introduced to **study the impact of a change in regime** (i.e., introduction of a new tax).

4. Full information, perfect foresight and no uncertainty around shocks.

5. Shocks can hit the economy today or at any time in the future, in which case they **would be expected with perfect foresight**.

- Shocks can also last one or several periods.
- Most often models introduce a positive shock today and zero shocks thereafter (with certainty).

Stochastic models have the following characteristics:

1. These types of models tend to be **more popular in the literature**. Examples include most RBC, or new Keynesian monetary models.
2. In these models, **shocks hit today (with a surprise)**, but thereafter their expected value is zero.
    - Expected future shocks, or permanent changes in the exogenous variables cannot be handled due to the use of Taylor approximations around a steady state.
3. When these models are linearized to the first order, **agents behave as if future shocks where equal to zero** (since their expectation is null), which is the certainty equivalence property.
    - This is an often overlooked point in the literature which misleads readers in supposing their models may be deterministic.