Solving DSGE models

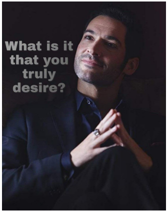
Macro II - Fluctuations - ENSAE, 2023-2024

Pablo Winant

2024-03-20

Introduction

What is the main specificity of economic modeling?
In (macro)economics, we *model* the behaviour of economic agents by specifying:



- their objective

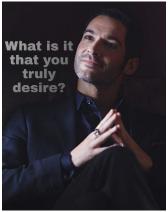
$$\max_{c_t} E_t \sum_{s \geq t} \beta^s U(c_s)$$

$$\max \pi_t$$

$$\dots$$

- their constraints (budget constraint, econ. environment...)

What is the main specificity of economic modeling?



In (macro)economics, we *model* the behaviour of economic agents by specifying:

- their objective

$$\max_{c_t} E_t \sum_{s \geq t} \beta^s U(c_s)$$

$$\max \pi_t$$

$$\ldots$$

- their constraints (budget constraint, econ. environment...)

This has important implications:

- macro models are *forward looking*
- macro models need to be **solved**

In many cases, there is not closed form for the solution -> we need numerical techniques

Dynare

- 1996: Michel Juillard created an opensource software to solve DSGE models
- It has been widely adopted:
  - early version in Gauss
  - then Matlab/Octave/Scilab
  - latest version in Julia

Figure 1: Michel Juillard

Dynare

- 1996: Michel Juillard created an opensource software to solve DSGE models
  - DSGE: Dynamic Stochastic General Equilibrium
- It has been widely adopted:
  - early version in Gauss
  - then Matlab/Octave/Scilab
  - latest version in Julia

Figure 1: Michel Juillard

Dynare

- 1996: Michel Juillard created an opensource software to solve DSGE models
  - DSGE: Dynamic Stochastic General Equilibrium
  - usually solved around a steady-state
- It has been widely adopted:
  - early version in Gauss
  - then Matlab/Octave/Scilab
  - latest version in Julia

Figure 1: Michel Juillard

Dynare

- 1996: Michel Juillard created an opensource software to solve DSGE models
  - DSGE: Dynamic Stochastic General Equilibrium
  - usually solved around a steady-state
- Now about 10 contributors.
- It has been widely adopted:
  - early version in Gauss
  - then Matlab/Octave/Scilab
  - latest version in Julia

Figure 1: Michel Juillard

DSGE Models in institutions

Nowadays most DSGE models built in institutions have a Dynare version (IMF/GIMF, EC/Quest, ECB/, NYFed/FRBNY)

- they are usually based on the *midsize model* from Smets & Wouters (10 equations)
- but have grown up a lot (»100 equations)

DSGE Models in institutions

Nowadays most DSGE models built in institutions have a Dynare version (IMF/GIMF, EC/Quest, ECB/, NYFed/FRBNY)

- they are usually based on the *midsize model* from Smets & Wouters (10 equations)
- but have grown up a lot (»100 equations)

Institutions (led by researchers) are (slowly) diversifying their model

- Computational General Equilibrium Models
- Agent-based
- Semi-structural models
- Heterogenous Agents Models

Solving a model

Model

A very concise representation of a model

$$\mathbb{E}_t \left[f(y_{t+1}, y_t, y_{t-1}, \epsilon_t) \right] = 0$$

The **problem**:

- $y_t \in \mathbb{R}^n$: the vector of endogenous variables
- $\epsilon_t \in \mathbb{R}^{n_e}$: the vector of exogenous variables
  - we assume that ϵ_t is a zero-mean gaussian process
- $f : \mathbb{R}^n \rightarrow \mathbb{R}^n$: the model equations

The **solution**:

- g such that

$$\forall t, y_t = g(y_{t-1}, \epsilon_t)$$

The timing of the equations

Tip

In dynare the model equations are coded in the `model; ... ; end;` block.

New information arrives with the innovations ϵ_t.

At date t, the information set is spanned by
$\mathcal{F}_t = \mathcal{F}(\cdots, \epsilon_{t-3}, \epsilon_{t-2}, \epsilon_{t-1}, \epsilon_t)$

By convention *an endogenous variable has a subscript t if it is known first at date t*.

Example

The timing of equations

Using Dynare's timing conventions:

- Write the production function in the RBC
- Write the law of motion for capital k, with a depreciation rate δ and investment i
  - when is capital known?
  - when is investment known?
- Add a multiplicative investment efficiency shock χ_t. Assume it is an $AR1$ driven by innovation η_t and autocorrelation ρ_χ

Steady-state

The deterministic steady-state satisfies:

$$f(\overline{y}, \overline{y}, \overline{y}, 0) = 0$$

Often, there is a closed-form solution.

Otherwise, one must resort to a numerical solver to solve

$$\overline{y} \rightarrow f(\overline{y}, \overline{y}, \overline{y}, 0)$$

Tip

In dynare the steady-state values are provided in the `steadystate_model; ... ; end;` block. One can check they are correct using the `check;` statement.
To find numerically the steady-state: `steady;`.

The implicit system

Replacing the solution

$$y_t = g(y_{t-1}, \epsilon_t)$$

in the system

$$\mathbb{E}_t \left[f(y_{t+1}, y_t, y_{t-1}, \epsilon_t) \right] = 0$$

we obtain:

$$\mathbb{E}_t \left[f(g(g(y_{t-1}, \epsilon_t), \epsilon_{t+1}), g(y_{t-1}, \epsilon_t), y_{t-1}, \epsilon_t) \right] = 0$$

It is an equation defining implicitly the function $g()$

The state-space

$$\mathbb{E}_t \left[f(g(g(y_{t-1}, \epsilon_t), \epsilon_{t+1}), g(y_{t-1}, \epsilon_t), y_{t-1}, \epsilon_t) \right] = 0$$

In this expression, y_{t-1}, ϵ_t is the state-space.

The state-space

$$\mathbb{E}_t\left[f(g(g(y_{t-1},\epsilon_t),\epsilon_{t+1}),g(y_{t-1},\epsilon_t),y_{t-1},\epsilon_t)\right]=0$$

In this expression, y_{t-1},ϵ_t is the state-space.

Dropping the time subscripts, the equation must be satisfied for any realization of (y,ϵ)

$$\forall(y,\epsilon)\ \Phi(g)(y,\epsilon)=\mathbb{E}_{\epsilon'}\left[f(g(g(y,\epsilon),\epsilon'),g(y,\epsilon),y,\epsilon)\right]=0$$

It is a functional equation $\Phi(g)=0$

Expected shocks

First order approximation:

- Assume $|y_t - \overline{y}| << 1$,$|\epsilon| << 1$,$|\epsilon'| << 1$

Perform a Taylor expansion with respect to future shock:

$$
\begin{aligned}
& \mathbb{E}_{\epsilon'}\left[f(g(g(y,\epsilon),\epsilon'),g(y,\epsilon),y,\epsilon)\right] && (1)\\
= & \mathbb{E}_{\epsilon'}\left[f(g(g(y,\epsilon),0),g(y,\epsilon),y,\epsilon)\right] && (2)\\
& +\mathbb{E}_{\epsilon'}\left[f'_{y_{t+1}}(g(g(y,\epsilon),0),g(y,\epsilon),y,\epsilon)g'_{\epsilon}\epsilon'\right] + o(\epsilon') && (3)\\
\approx & f(g(g(y,\epsilon),0),g(y,\epsilon),y,\epsilon) && (4)
\end{aligned}
$$

Expected shocks

First order approximation:

- Assume $|y_t - \overline{y}| << 1$,$|\epsilon| << 1$,$|\epsilon'| << 1$

Perform a Taylor expansion with respect to future shock:

$$
\begin{aligned}
& \mathbb{E}_{\epsilon'}\left[f(g(g(y,\epsilon),\epsilon'),g(y,\epsilon),y,\epsilon)\right] && (1)\\
= \quad & \mathbb{E}_{\epsilon'}\left[f(g(g(y,\epsilon),0),g(y,\epsilon),y,\epsilon)\right] && (2)\\
& +\mathbb{E}_{\epsilon'}\left[f'_{y_{t+1}}(g(g(y,\epsilon),0),g(y,\epsilon),y,\epsilon)g'_{\epsilon}\epsilon'\right] + o(\epsilon') && (3)\\
\approx \quad & f(g(g(y,\epsilon),0),g(y,\epsilon),y,\epsilon) && (4)
\end{aligned}
$$

This uses the fact that $\mathbb{E}\left[\epsilon'\right] = 0$.

At first order, expected shocks play no role.

To capture precautionary behaviour (like risk premia), we would need to increase the approximation order.

First order perturbation

We are left with the system:

$$F(y, \epsilon) = f(g(g(y, \epsilon), 0), g(y, \epsilon), y, \epsilon) = 0$$

We can now use a variant of the *implicit function theorem* to recover a first approximation of g as:

$$g(y, \epsilon) = \overline{y} + g'_y(y - \overline{y}) + g'_e \epsilon_t$$

First order perturbation

We are left with the system:

$$F(y,\epsilon) = f(g(g(y,\epsilon),0), g(y,\epsilon), y, \epsilon) = 0$$

We can now use a variant of the *implicit function theorem* to recover a first approximation of g as:

$$g(y,\epsilon) = \overline{y} + g'_y(y - \overline{y}) + g'_e \epsilon_t$$

We can obtain the unknown quantities g'_y, and g'_e using the *method of undeterminate coefficients*:

Plug the first approximation into the system and write the conditions

$$F'_y(\overline{y}, 0) = 0$$

$$F'_\epsilon(\overline{y}, 0) = 0$$

Computing $g_y^{'}$

Recall the system:

$$F(y,\epsilon) = f(g(g(y,0),\epsilon), g(y,\epsilon), y, \epsilon) = 0$$

We have

$$F'_y(\overline{y},0) = f'_{y_{t+1}} g'_y g'_y + f'_{y_t} g'_y + f'_{y_{t-1}} = 0$$

Computing $g_y^{'}$

Recall the system:

$$F(y, \epsilon) = f(g(g(y,0),\epsilon), g(y,\epsilon), y, \epsilon) = 0$$

We have

$$F_y'(\overline{y}, 0) = f'_{y_{t+1}} g'_y g'_y + f'_{y_t} g'_y + f'_{y_{t-1}} = 0$$

This is a specific Riccati equation

$$AX^2 + BX + C$$

where A, B, C and $X = g_y'$ are square matrices $\in \mathbb{R}^n \times \mathbb{R}^n$

First Order Deterministic Model

Let's pause a minute to observe the first order deterministic model:

$$AX^2 + BX + C$$

From our intuition in dimension 1, we know there must be multiple solutions

- how do we find them?
- how do we select the right ones?

I the absence of shocks the dynamics of the model are given by

$$y_t = Xy_{t-1}$$

What is the condition for the model to be stationary?

First Order Deterministic Model

Let's pause a minute to observe the first order deterministic model:

$$AX^2 + BX + C$$

From our intuition in dimension 1, we know there must be multiple solutions

- how do we find them?
- how do we select the right ones?

I the absence of shocks the dynamics of the model are given by

$$y_t = X y_{t-1}$$

What is the condition for the model to be stationary?

-$>$ the biggest eigenvalue of X should be smaller than 1

Multiplicity of solution

It is possible to show that the system is associated with $2n$ generalized eigenvalues:

$$|\lambda_1| \leq \cdots \leq |\lambda_{2n}|$$

For each choice C of n eigenvalues ($|C| = n$), a specific recursive solution X_C can be *constructed*. It has eigenvalues C.

Multiplicity of solution

It is possible to show that the system is associated with $2n$ generalized eigenvalues:

$$|\lambda_1| \leq \cdots \leq |\lambda_{2n}|$$

For each choice C of n eigenvalues ($|C| = n$), a specific recursive solution X_C can be *constructed*. It has eigenvalues C.

This yields at least $\binom{2n}{n}$ different combinations.

Multiplicity of solution

It is possible to show that the system is associated with $2n$ generalized eigenvalues:

$$|\lambda_1| \leq \cdots \leq |\lambda_{2n}|$$

For each choice C of n eigenvalues ($|C| = n$), a specific recursive solution X_C can be *constructed*. It has eigenvalues C.

This yields at least $\binom{2n}{n}$ different combinations.

A model is well defined when there is **exactly one solution that is non divergent**.

This is equivalent to:

$$|\lambda_1| \leq \cdots \leq |\lambda_n| \leq 1 < |\lambda_{n+1}| \leq \cdots \leq |\lambda_{2n}|$$

Example 1

Forward looking inflation:

$$\pi_t = \alpha \pi_{t+1}$$

with $\alpha < 1$.

Is it well defined?

Example 1

Forward looking inflation:

$$\pi_t = \alpha \pi_{t+1}$$

with $\alpha < 1$.

Is it well defined?

We can rewrite the system as:

$$\alpha \pi_{t+1} - \pi_t + 0 \pi_{t-1} = 0$$

or

$$\pi_{t+1} - (\frac{1}{\alpha} + 0) \pi_t + \left(\frac{1}{\alpha} 0 \right) \pi_{t-1} = 0$$

Example 1

Forward looking inflation:

$$\pi_t = \alpha \pi_{t+1}$$

with $\alpha < 1$.

Is it well defined?

We can rewrite the system as:

$$\alpha \pi_{t+1} - \pi_t + 0 \pi_{t-1} = 0$$

or

$$\pi_{t+1} - (\frac{1}{\alpha} + 0)\pi_t + \left(\frac{1}{\alpha} 0\right) \pi_{t-1} = 0$$

The generalized eigenvalues are $0 \leq 1 < \frac{1}{\alpha}$.

Example 1

Forward looking inflation:

$$\pi_t = \alpha \pi_{t+1}$$

with $\alpha < 1$.

Is it well defined?

We can rewrite the system as:

$$\alpha \pi_{t+1} - \pi_t + 0\pi_{t-1} = 0$$

or

$$\pi_{t+1} - (\frac{1}{\alpha} + 0)\pi_t + \left(\frac{1}{\alpha}0\right)\pi_{t-1} = 0$$

The generalized eigenvalues are $0 \leq 1 < \frac{1}{\alpha}$.

The unique stable solution is $\pi_t = 0\pi_{t-1}$

Example 2

Debt accumulation equation by a rational agent:

$$b_{t+1} - (1 + \frac{1}{\beta})b_t + \frac{1}{\beta}b_{t-1} = 0$$

Is it well-defined?

Example 2

Debt accumulation equation by a rational agent:

$$b_{t+1} - (1 + \frac{1}{\beta})b_t + \frac{1}{\beta}b_{t-1} = 0$$

Is it well-defined?

Two generalized eigenvalues $\lambda_1 = 1 < \lambda_2 = \frac{1}{\beta}$

Example 2

Debt accumulation equation by a rational agent:

$$b_{t+1} - (1 + \frac{1}{\beta})b_t + \frac{1}{\beta}b_{t-1} = 0$$

Is it well-defined?

Two generalized eigenvalues $\lambda_1 = 1 < \lambda_2 = \frac{1}{\beta}$

The unique non-diverging solution is $b_t = b_{t-1}$.

- it is a *unit-root*: any initial deviation in b_{t-1} has persistent effects

Example 3

Productivity process:

$$z_t = \rho z_{t-1}$$

with $\rho < 1$: well defined

Example 3

Productivity process:

$$z_t = \rho z_{t-1}$$

with $\rho < 1$: well defined

In that case there is a hidden infinite eigenvalue ∞ associated to z_{t+1}.

Example 3

Productivity process:

$$z_t = \rho z_{t-1}$$

with $\rho < 1$: well defined

In that case there is a hidden infinite eigenvalue ∞ associated to z_{t+1}.

To see why consider the system associated with eigenvalues m and ρ:

$$z_{t+1} - (m + \rho) z_t + m\rho z_{t-1} = 0$$

$$\frac{1}{m} z_{t+1} - (1 + \frac{\rho}{m}) z_t + \rho z_{t-1} = 0$$

Which corresponds to the initial model when $m = \infty$

Example 3

Productivity process:

$$z_t = \rho z_{t-1}$$

with $\rho < 1$: well defined

In that case there is a hidden infinite eigenvalue ∞ associated to z_{t+1}.

To see why consider the system associated with eigenvalues m and ρ:

$$z_{t+1} - (m+\rho)z_t + m\rho z_{t-1} = 0$$

$$\frac{1}{m} z_{t+1} - (1 + \frac{\rho}{m}) z_t + \rho z_{t-1} = 0$$

Which corresponds to the initial model when $m = \infty$

The generalized eigenvalues are $\lambda_1 = \rho \leq 1 < \lambda_2 = \infty$

More generally, any variable that does not appear in $t+1$ creates one infinite generalized eigenvalue.

A criterium for well-definedness

Looking again at the list of eigenvalues we set aside the infinite ones.

The model is well specified iff we can sort the eigenvalues as:

$$|\lambda_1| \leq \cdots \leq |\lambda_n| \leq 1 < |\lambda_{n+1}| \leq \cdots |\lambda_{n+k}| \leq \underbrace{|\lambda_{n+k+1}| \cdots \leq |\lambda_{2n}|}_{\text{infinite eigenvalues}}$$

Blanchard-Kahn criterium

The model satisfies the Blanchard-Kahn criterium if the number of eigenvalues greater than one, is exactly equal to the number of variables *appearing* in $t+1$.
In that case the model is well-defined.

Computing the solution

There are several classical methods to compute the solution to the algebraic Riccatti equation:

$$AX^2 + BX + C = 0$$

- qz decomposition
  - traditionnally used in the DSGE literature since Chris Sims
  - a little bit unintuitive
- cyclic reduction
  - new default in dynare, more adequate for big models
- linear time iteration cf @sec:linear_time_iteration
  - conceptually very simple

Computing $g_e^{'}$

Now we have g_y', how do we get g_e'?

Recall:

$$F(y,\epsilon) = f(g(g(y,\epsilon),0), g(y,\epsilon), y, \epsilon) = 0$$

We have

$$F_e'(\overline{y},0) = f_{y_{t+1}}' g_y' g_e' + f_{y_t}' g_e' + f_{\epsilon_t}' = 0$$

Now this is easy:

$$g_e' = -(f_{y_{t+1}}' g_y' + f_{y_t}')^{-1} f_{\epsilon_t}' = 0$$

The model solution

The result of the model solution:

$$y_t = g_y y_{t-1} + g_e \epsilon_t$$

It is an AR1, driven by exogenous shock ϵ_t.

The model solution

The result of the model solution:

$$y_t = g_y y_{t-1} + g_e \epsilon_t$$

It is an AR1, driven by exogenous shock ϵ_t.

Because it is a well known structure, one can investigate the model with

- impulse response functions
- stochastic simulations

The model solution

The result of the model solution:

$$y_t = g_y y_{t-1} + g_e \epsilon_t$$

It is an AR1, driven by exogenous shock ϵ_t.

Because it is a well known structure, one can investigate the model with

- impulse response functions
- stochastic simulations

Then to compare the model to the data we compute

- implied moments:
  - covariances, autocorrelation
- likelihood

Optimizing the fit to the data is called *model* estimation

Conclusion

What can you do with the solution

The solution of a model found by Dynare has an especially simple form: an AR1

- $y_t = Xy_{t-1} + Y\epsilon_t$
- where the covariances Σ of ϵ_t can be chosen by the modeler

What can you do with the solution

The solution of a model found by Dynare has an especially simple form: an AR1

- $y_t = X y_{t-1} + Y \epsilon_t$
- where the covariances Σ of ϵ_t can be chosen by the modeler

With this solution we can (cf next TD)

- compute (conditional and unconditional) moments
- perform stochastic simulations, impulse response function

What can you do with the solution

The solution of a model found by Dynare has an especially simple form: an AR1

- $y_t = X y_{t-1} + Y \epsilon_t$
- where the covariances Σ of ϵ_t can be chosen by the modeler

With this solution we can (cf next TD)

- compute (conditional and unconditional) moments
- perform stochastic simulations, impulse response function

Going Further

Taking the model to the data with Dynare

- "estimate" the model: compute the likelihood of a solution and maximize it by choosing the right parameters
- "identify" shocks in the data

Other functions

- higher order approximation
- (noninear) perfect foresight simulations
- ramsey plan
- discretionary policy
- ...

Coming Next

Many models

Appendix: Linear Time Iteration

Linear Time Iteration

Recall the system to solve:

$$F(y, \epsilon) = f(g(g(y, \epsilon), 0), g(y, \epsilon), y, \epsilon) = 0$$

but now assume the decision rules today and tomorrow are different:

- today: $y_t = g(y_{t-1}, \epsilon_t) = \overline{y} + X y_{t-1} + g_y \epsilon_t$
- tomorrow: $y_{t+1} = \tilde{g}(y_t, \epsilon_{t+1}) = \overline{y} + \tilde{X} y_{t-1} + \tilde{g}_y \epsilon_t$

Then the Ricatti equation is written:

$$A \tilde{X} X + B X + C = 0$$

Linear Time Iteration (2)

The linear time iteration algorithm consists in solving the decision rule X today as a function of decision rule tomorrow $\tilde{X}$.

This corresponds to the simple formula:

$$X = -(A\tilde{X} + B)^{-1}C$$

And the full algorithm can be described as:

- choose X_0
- for any X_n, compute $X_{n+1} = T(X_n) = -(AX_n + B)^{-1}C$
  - repeat until convergence

Linear Time Iteration (3)

It can be shown that, starting from a random initial guess, the linear time-iteration algorithm converges to the solution X with the smallest modulus:

$$\underbrace{|\lambda_1| \leq \cdots \leq |\lambda_n|}_{\text{Selected eigenvalues}} \leq |\lambda_{n+1}| \cdots \leq |\lambda_{2n}|$$

In other words, it finds the right solution when the model is well specified.

How do you check it is well specified?

- λ_n is the biggest eigenvalue of solution X
- what about λ_{n+1}?
  - $\frac{1}{\lambda_{n+1}}$ is the biggest eigenvalue of $(AX + B)^{-1}A$

Linear Time Iteration (4)

Define

$$M(\lambda) = A\lambda^2 + B\lambda + C$$

For any solution X, $M(\lambda)$ can be factorized as: [1]

$$M(\lambda) = (\lambda A + AX + B)(\lambda I - X)$$

and

$$det(M(\lambda)) = \underbrace{det(\lambda A + AX + B)}_{Q(\lambda)} det(\lambda I - X)$$

By construction $Q(\lambda)$ is a polynomial whose roots are those that are not selected by the solution i.e. $\Lambda \quad Sp(X)$.

[1]Special case of Bezout theorem. Easy to check in that case

Linear Time Iteration (5)

For $\lambda \neq 0$ we have:

$$\lambda \in Sp((AX+B)^{-1}A)$$

$$\iff det((AX+B)^{-1})A - I\lambda) = 0$$

$$\iff det(\frac{1}{\lambda}A - I(AX+B)) = 0$$

$$\iff Q(\frac{1}{\lambda}) = 0$$

$$\iff \frac{1}{\lambda} \in G \quad Sp(X)$$

In words, $(AX+B)^{-1}$ contains all the eigenvalues that have been rejected by the selection of X.

In particular, $\rho((AX+B)^{-1})A) = 1/\min(G \quad Sp(X))$