Simulating and Estimating DSGE Model with Dynare

Tao Zeng

Wuhan University

April 2016

What is Dynare?

- Dynare is a Matlab frontend to solve and simulate dynamic models
- Either deterministic or stochastic
- Developed by Michel Juillard at CEPREMAP
- website: http://www.cepremap.cnrs.fr/dynare/

How does it work?

- Write the code of the model
- Takes care of parsing the model to Dynare
- Rearrange the model
- Solves the model
- Use the solution to generate some output
- Can estimate the model

Structure of the mod file: Simulation

- Preamble: Define variables and parameters
- Model: Equations of the model
- Steady State: Compute the steady state
- Shocks: Define the properties of Shocks
- Solution: Compute the Solution and Product Output

Structure of the mod file: Preamble

- In this part, we need to define endogenous variables, shocks and parameters by three commands **var**, **varexo** and **parameters.**
  - VAR: define the endogenous variables of your model
  - VAREXO: define the list of shocks in your model
  - PARAMETERS: define the list of parameters and then assign the parameters values.
- Assume the model takes the form

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

  with $e_t \sim N\left(0, \sigma^2\right)$.
- Variable is x_t, exogenous variables is e_t and parameters are ρ and σ.

- In practice, we always want to know wether our model match data
- The model takes the form

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

  with $e_t \sim N\left(0, \sigma^2\right)$.
- We compute the sample moments such as mean, variance and covariance of the data.
- Then we compute the theoretical moments of the model.
- Compare the sample moments with the theoretical moments.
- Here we simulate data from the model, treat the simulated data as real data and compute the moments such as mean, variance and covariance.

Structure of the mod file: Preamble

An example

- The dynare code for the Preamble part

```
var x;
varexo e;
parameters rho,se;
rho = 0.90;
se = 0.01;
```

Structure of the mod file: Model

An example

- In the model block, we need to define model equations using "model;" and "end;"
- Model the AR(1) processes as
  model;
  x=rho*x(-1)+e;
  end;
- Between the command **model** and **end**, there need to be **as many equations as** you declared endogenous variables in the **var** part.
- Each line of instruction ends with a semicolon.
- Variable with a time t subscript, such as x_t,is written as x.
- Variable with a time t-n subscript, such as x_{t-n}, is written as $x\,(-n)$.
- Variable with a time t+n subscript, such as x_{t+n}, is written as $x\,(+n)$.

Structure of the mod file: Steady State

- Compute the long–run of the model which is the deterministic value that the dynamic system will converge to.
- We will take appximation around this long run.
- The structure is as follows
  - inival;
    $\cdots$
    end;
    steady;
    check;
- *Steady* computes the long run of the model using a non–linear New-type solver.
- It therefore needs initial conditions. That is the role of the *inival*;$\cdots$*end*; statement. Note that if the inival block is not followed by *steady*, the steady state computation will still be triggered by subsequent commands (*stoch_simul*, *estimation*,...).

Structure of the mod file: Steady State

- You would better give a initial value close to the exact steady state.
- *histval;...end;* block allows setting the starting point of those simulations in the state space (it does not affect the starting point for impulse response functions).
  histval;
  x(0)=0;
  end;
- *check* is optional. It checks the dynamic stability of the system by BK condition. It computes and displays the eigenvalues of the system. A necessary conditions for the uniqueness of a stable equilibrium in the neighborhood of the steady state is that there are as many eigenvalues larger than 1 in modulus as there are forward looking variables in the system.

Structure of the mod file: Steady State

- Again take the AR(1) example:

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

- In deterministic steady state: $e_t = \bar{e} = 0$, therefore

$$\bar{x} = \rho \bar{x} \Longrightarrow \bar{x} = 0$$

- Hence
  initival
  e = 0
  x = 0
  end;
  steady;
  check;

Structure of the mod file: Shocks

- Exogenous shocks are gaussian innovations with 0 mean.
- Structure:

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

- Therefore, for the AR(1) example

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

Structure of the mod file: Solution

- Final step: Compute the solution and produce some output
- Solution method: First or Second order perturbation method
- Then compute some moments and impulse responses.
- Getting solution: *stoch_simul*(...)...;
- Again take the 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

- Solver
  - linear: In case of a linear model.
  - order = 1 or 2 : order of Taylor approximation (default = 2), unless you are working with a linear model in which case the order is automatically to 1.
- 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, default is 5.
  - *hp_filter = DOUBLE:* Using HP filter to the model for theoretical moments (if *periods=0*) and the simulated moments.

- Impulse Response Functions
  - *irf* = *INTEGER*: number of periods on which to compute the IRFs (Setting IRF=0, suppresses the plotting of IRFs). Default is 40.
  - *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). Dynare's default is to produce analytical/theoretical moments of the variables.
  - Having periods not equal to zero will instead have it simulate data and take the
  - moments from the simulated data.
  - *replic* = *INTEGER*: number of simulated series used to compute the IRFs (default = 1 if order = 1, and 50 otherwise).

Structure of the mod file: Solution

Options of the stoch_simul

- Simulations
  - *drop* $= INTEGER$: number of points dropped in simulations (default = 100). By default, Dynare drops the first 100 values from a simulation, so you need to give it a number of periods greater than 100 for this to work. Hence, typing "stoch simul(periods=300);" will produce moments based on a simulation with 200 periods.
  - set_dynare_seed ($INTEGER$): set the random seeds
- To run a Dynare file, simply type "dynare filename" into the command window while in Matlab. For e.g.: "dynare *.mod"

- Dynare obtains linear approximations to the policy functions that satisfy the first-order conditions.
- State variables: $x_t = \left[x_{1t}, x_{2t}, \cdots, x_{nt}\right]'$
- The endogenous variable can be expressed as

$$y_t = \bar{y} + a\left(x_t - \bar{x}\right)$$

where a bar above a variable indicates steady state value.

- Specification of the model in level

$$\max_{\{c_t,k_t\}_{t=1}^{\infty}} E\sum_{t=1}^{\infty}\beta^{t-1}\frac{c_t^{1-v}-1}{1-v}$$

$$c_t+k_{t+1}=z_t k_{t-1}^{\alpha}+(1-\delta)\,k_{t-1}$$

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

$$k_0 \text{ given, } E_t\left(\varepsilon_{t+1}\right)=0 \text{ and } E_t\left(\varepsilon_{t+1}^2\right)=\sigma^2$$

- Model equations

$$\begin{aligned} c_t^{-v} &= E_t\left[\beta c_{t+1}^{-v}\left(\alpha z_{t+1}k_t^{\alpha-1}+1-\delta\right)\right] \\ c_t+k_t &= z_t k_{t-1}^{\alpha}+(1-\delta)\,k_{t-1} \\ z_t &= (1-\rho)+\rho z_{t-1}+\varepsilon_t \end{aligned}$$

- $x_t = [k_{t-1}, z_t]$, $y_t = [c_t, k_t, z_t]$
- Linearized solution

$$
\begin{aligned}
c_t &= \bar{c} + a_{ck}\left(k_{t-1} - \bar{k}\right) + a_{cz}\left(z_t - \bar{z}\right) && (1)\\
k_t &= \bar{k} + a_{kk}\left(k_{t-1} - \bar{k}\right) + a_{kz}\left(z_t - \bar{z}\right) && (2)\\
z_t &= \rho z_{t-1} + \varepsilon_t && (3)
\end{aligned}
$$

- Dynare does not understand what c_t is, it only generates a linear solution in what you specify as the variables.

Structure of the mod file: Some tips

Neoclassical example

- The equation (2) and (3) can of course be written (less conveniently) as

$$k_t = \bar{k} + a_{kk}\left(k_{t-1} - \bar{k}\right) + a_{kz_{-1}}\left(z_{t-1} - \bar{z}\right) + a_{kz}\varepsilon_t$$

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

  by substituting (3) into (2) with $a_{kz_{-1}} = \rho a_{kz}$.
- Dynare gives the solution in the less convenient form

$$c_t = \bar{c} + a_{ck}\left(k_{t-1} - \bar{k}\right) + a_{cz_{-1}}\left(z_{t-1} - \bar{z}\right) + a_{cz}\varepsilon_t$$

$$k_t = \bar{k} + a_{kk}\left(k_{t-1} - \bar{k}\right) + a_{kz_{-1}}\left(z_{t-1} - \bar{z}\right) + a_{kz}\varepsilon_t$$

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

- Dynare equations

```
c^(-nu)=beta*c(+1)^(-nu)*(alpha*z(+1)*k^(alpha-1)+1-delta);
c+k=z*k(-1)^alpha+(1-delta)k(-1);
z=(1-rho)+rho*z(-1)+e;
```

- $\delta = 0.025$, $v = 2$, $\alpha = 0.36$, $\beta = 0.99$, and $\rho = 0.95$ and the results from the dynare

POLICY AND TRANSITION FUNCTIONS

	k	z	c
constant	37.989254	1.000000	2.754327
k(-1)	0.976540	-0.000000	0.033561
z(-1)	2.597386	0.950000	0.921470
e	2.734091	1.000000	0.969968

Structure of the mod file: Some tips

Neoclassical example

- The table must be interpreted as follows

	k	z	c
constant	37.989254	1.000000	2.754327
k(-1)-kss	0.976540	-0.000000	0.033561
z(-1)-zss	2.597386	0.950000	0.921470
e	2.734091	1.000000	0.969968

- That is, explanatory variables are relative to steady state. (Note that steady state of e is zero by definition)
- If explanatory variables take on steady state values, then choices are equal to the constant term, which of course is simply equal to the corresponding steady state value

- Specification of the model in level

$$\max_{\{c_t, k_t\}_{t=0}^{\infty}} E \sum_{t=0}^{\infty} \beta^t \frac{c_t^{1-v} - 1}{1-v}$$

$$k_{t+1} = a_t k_t^{\alpha} - c_t + (1-\delta) k_t$$

$$\ln a_t = \rho \ln a_{t-1} + \varepsilon_t$$

$$k_0 \text{ given, } E_t(\varepsilon_{t+1}) = 0 \text{ and } E_t\left(\varepsilon_{t+1}^2\right) = \sigma^2$$

- Model equations

$$\begin{aligned} c_t^{-v} &= E_t\left[\beta c_{t+1}^{-v}\left(\alpha \exp\left(a_{t+1}\right) k_t^{\alpha-1} + 1 - \delta\right)\right] \\ k_{t+1} &= a_t k_t^{\alpha} - c_t + (1-\delta) k_t \\ \ln a_t &= \rho \ln a_{t-1} + \varepsilon_t \end{aligned}$$

- In particular, Dynare requires that predetermined variables (like the captical stock) show up as dated $t-1$ in the time t equatiion and t in the tiime $t+1$ equations.
- We rewrite the model as

$$\begin{aligned} c_t^{-v} &= E_t\left[\beta c_{t+1}^{-v}\left(\alpha \exp\left(a_{t+1}\right) k_t^{\alpha-1}+1-\delta\right)\right] \\ k_t &= a_t k_{t-1}^{\alpha}-c_t+(1-\delta) k_{t-1} \\ \ln a_t &= \rho \ln a_{t-1}+\varepsilon_t \\ y_t &= a_t k_{t-1}^{\alpha} \\ i_t &= y_t-c_t \end{aligned}$$

- If want all the variable in log form, which means that we want to get the pecentige deviation, then the model equations can be written asexp(c)^(-nu) = beta*(exp(c(+1))^(-nu)*(alpha*exp(a(+1))*exp(k)^(alpha-1) + (1-delta)));

- If want all the variable in log form, which means that we want to get the pecentige deviation, then the model equations can be written
asexp(c)^(-nu) =
beta*(exp(c(+1))^(-nu)*(alpha*exp(a(+1))*exp(k)^(alpha-1) + (1-delta)));
exp(y) = exp(a)*exp(k(-1))^(alpha);
exp(k) = exp(I) + (1-delta)*exp(k(-1));
exp(y) = exp(c) + exp(I);
a = rho*a(-1)+ e

- Dynare distinguishes four types of endogenous variables:
    - *Purely backward (or purely predetermined) variables* Those that appear only at current and past period in the model, but not at future period (i.e. at t and t-1 but not t + 1). The number of such variables is equal to *M_.npred*, such as k_t.
    - *Purely forward variables* Those that appear only at current and future period in the model, but not at past period (i.e. at t and t+1 but not t-1). The number of such variables is stored in *M_.nfwrd*, here is c_t.
    - *Mixed variables* Those that appear at current, past and future period in the model (i.e. at t, t+1 and t-1). The number of such variables is stored in *M_.nboth*, such as a_t here.
    - *Static variables* Those that appear only at current, not past and future period in the model (i.e. only at t, not at t+1 or t-1). The number of such variables is stored in *M_.nstatic*, such as I_t, y_t here.

Structure of the mod file: Some tips

Types of endougenous variables

- $M\_.npred + M\_.nboth + \quad M\_.nfwrd \quad + \quad M\_.static =$ $M\_.endo\_nbr$
- The state variables of the model are the purely backward variables and the mixed variables.
- The first order approximation of the decision rules

$$y_t = y^s + A\left(x_{t-1} - x^s\right) + Bu_t$$

- y^s is stored in *oo_.dr.ys*, x^s is part of y^s. The vector rows correspond to all endogenous in the declaration order.
- A is stored in *oo_.dr.ghx*. The matrix rows correspond to all endogenous in DR-order. The matrix columns correspond to state variables in DR-order.
- B is stored in *oo_.dr.ghu*. The matrix rows correspond to all endogenous in DR-order. The matrix columns correspond to exogenous variables in declaration order.

Structure of the mod file: Some tips

Types of endougenous variables

- Internally, Dynare uses two orderings of the endogenous variables:
  - the order of declaration (which is reflected in M_.endo_names)
  - the order based on the four types described above, which we will call the DR-order ("DR" stands for decision rules).
- Most of the time, the declaration order is used, but for elements of the decision rules, the DR-order is used.
- The DR-order is the following: static variables appear first, then purely backward variables, then mixed variables, and finally purely forward variables. Inside each category, variables are arranged according to the declaration order.
- Variable *oo_.dr.order_var* maps DR-order to declaration order, for instance, y^s is stored in declearation order, then $y^s\left(oo\_.dr.order\_var\right)$ is tranformed to DR-order.

- Variable *oo_.dr.inv_order_var* contains the inverse map, k-th declared variable is numbered oo_.dr.inv_order_var(k) in DR-order.

Excute the mod file in Matlab

- addpath ('G:\program\ dynare\4.4.3\matlab'); //Add path of dynare to matlab,
- pwd='G:\My Program\Teaching\Bayesian DSGE\dynare slides\dynare code\Neoclassical';
- cd (pwd) // set working directory
- dynare *.mod

DSGE as State Space Model

DSGE model can be casted into a linear state space form

- State equation

$$\widehat{y}_t = E_t\left[\widehat{y}_{t+1}\right] + \widehat{g}_t - E_t\left[\widehat{g}_{t+1}\right] - \frac{1}{\tau}\left(\widehat{R}_t - E_t\left[\widehat{\pi}_{t+1}\right] - E_t\left[\widehat{z}_{t+1}\right]\right)$$

$$\widehat{\pi}_t = \beta E_t\left[\widehat{\pi}_{t+1}\right] + \kappa\left(\widehat{y}_t - \frac{\tau}{\varrho}\widehat{g}_t\right)$$

$$\widehat{y}_t = \widehat{c}_t + \widehat{g}_t$$

$$\widehat{R}_t = \rho_R \widehat{R}_{t-1} + (1-\rho_R)\,\psi_1 \widehat{\pi}_t + (1-\rho_R)\,\psi_2\left(\widehat{y}_t - \frac{\tau}{\varrho}\widehat{g}_t\right)$$

where

$$\kappa = \frac{1-\nu}{\nu\phi\pi^2}\varrho, \quad \varrho = \tau + \varphi\,(1-\alpha) + \frac{\alpha}{1-\alpha}$$

$$\widehat{g}_t = \rho_g \widehat{g}_{t-1} + \epsilon_{g,t}$$

$$\widehat{z}_t = \rho_z \widehat{z}_{t-1} + \epsilon_{z,t}$$

- Space equations

$$\begin{aligned} YGR_t &= \gamma^{(Q)} + 100\left(\hat{y}_t - \hat{y}_{t-1} + \hat{z}_t\right) \\ INFL_t &= \pi^{(A)} + 400\hat{\pi}_t \\ INT_t &= \pi^{(A)} + r^{(A)} + 4\gamma^{(Q)} + 400\hat{R}_t \end{aligned}$$

- Why using Bayesian method to estimate DSGE models (An and Schorfheide, 2007)
    - It is system-based comparing to GMM estimation.
    - Prior distributions can be used to incorporate additional information into the parameter estimation.
- The state equation can be solved as

$$s_t = Ts_{t-1} + R\varepsilon_t.$$

- The space equation

$$y_t = D + Zs_t.$$

Bayesian Estimation for DSGE Model

- Dejong, Ingram, and Whiteman (2000), Schorfheide (2000) and Otrok (2001) — the first papers using Bayesian techniques to DSGE models.
- Smets and Wouters (2003, 2007) — more than a dozen hidden states and 36 estimators.
- Schorfheide (2000) and Otrok (2001) — Random Walk Metrolpolis Hasting (RWMH) algorithm
- 95% of papers published from 2005 – 2010 in eight top economics journals use the RWMH algorithm to estimate DSGE models (Herbst, 2011)
- Dynare with Matlab facilitates the use of the RWMH algorithm.

Bayesian Estimation for DSGE Model

- Choosing prior density
- Computing posterior mode
- Simulating posterior distribution
- Computing point estimates and confidence regions
- Computing posterior probabilities

MCMC for DSGE — Metropolis Hasting

For $i = 1$ to M, M is the number of draws

1. Draw θ from a proposal density $q\left(\theta^i|\theta^{i-1}\right)$
2. Draw u from $U(0,1)$
3. Set $\theta^i = \theta$ if

$$U \leq \alpha = \min\{1, \frac{p\left(y|\theta\right)p\left(\theta\right)}{p\left(y|\theta^{i-1}\right)p\left(\theta^{i-1}\right)}\frac{q\left(\theta^{i-1}|\theta\right)}{q\left(\theta|\theta^{i-1}\right)}\}$$

   and $\theta^i = \theta^{i-1}$ otherwise.
4. Go to step 1, draw until the desired number of iterations is obtained.

Random Walk Metropolis Hasting

- Random Walk Metropolis Hasting (RWMH) (Schorfheide, 2000):

$$\theta = \theta^{i-1} + \eta, \quad \alpha = \min\{1, \frac{p\left(y|\theta\right)p\left(\theta\right)}{p\left(y|\theta^{i-1}\right)p\left(\theta^{i-1}\right)}\}$$

  where η is a multivariate normal distribution with mean 0 and variance $c\widehat{\Sigma}$.

- How to choose $\hat{\Sigma}$

$$\hat{\Sigma}^{-1} = -\frac{\partial^2 \log p\left(\theta|y\right)}{\partial\theta\partial\theta'}|_{\theta=\hat{\theta}_m}$$

  where $\hat{\theta}_m = \arg\max \log p\left(\theta|y\right)$.

- c is used to control the acceptance rate, 0.234 for multivariate normal target distribution (Roberts et al., 1997) and between $0.20 - 0.40$ in practice.
- For the linearized case, $p\left(y|\theta\right)$ can be evaluated by Kalman Filter.

Structure of the mod file: Estimation

- Preamble: Define variables and parameters
- Model: Equations of the model
- Observables: Load the observed data
- Steady State: Specifying the steady state
- Prior: Define the prior distribution of the parameters.
- Estimation: Estimate the model.

The first two part are the same as in Simulating case.

Structure of the mod file: Estimation

Observables and Prior

- Observables: *varobs y*
- Note that the number of the obsevations should be less than the number of shocks.
- Prior: *estimated_params;... end;*
- The four parameters: parameter name, prior shape, prior mean, prior standard error

Structure of the mod file: Estimation

Prior

- The shape should be consistent with the domain of definition of the parameter
- Use values obtained in other studies (micro or macro)
- Check the graph of the priors
- Check the implication of your priors by running *stoch_simul* with parameters set at prior mean
- Do sensitivity tests by widening your priors

Structure of the mod file: Estimation

Steady state

- Give initial values for steady state
  initval;
  lc = -1.02;
  lk = -1.61;
  lz = 0;
  end;
- Steady state must be calculated for many different values of parameters, it is better to
  - Linearize the system yourself, then it is easy to solve for steady state.
  - Give the exact solution of steady state as initial values.
  - Provide program to calculate the steady state yourself.

- Give the exact solution of steady state as initial values, in your *.mod file include:

  ```
  steady_state_model;
  lz = 0;
  lk = log(((1-beta*(1-delta))/(alpha*beta))^(1/(alpha-1)));
  lc = log(exp(lk)^alpha-delta*exp(lk));
  ly = alpha*lk;
  li = log(delta)+lk;
  end;
  ```

- Provide program to calculate the steady state yourself.
  - If your *.mod file is called xxx.mod then write a file xxx_steadystate.m. The two files will be in the same directory.
  - Dynare checks whether a file with this name exists and will use it.
  - Sequence of output should correspond with sequence given in var list.

```
function [ys,check] = Neoclassical_estimation_ex_steadystate(ys,exo)
global M_
alpha = M_.params(1);
beta = M_.params(2);
delta = M_.params(3);
rho = M_.params(4);
nu = M_.params(5);
z = 1;
k = ((1-beta*(1-delta))/(alpha*beta))^(1/(alpha-1));
c = k^alpha-delta*k;
i = delta*k;
y =c+i;
ys =[ y; i; k; c; z ];
ys =ln(ys);
```

- estimation(datafile=cdata,mh_nblocks=5,mh_replic=10000, mh_jscale=3,mh_init_scale=12) lc;
- lc: (optional) name of the endogenous variables (e.g. if you want to plot Bayesian IRFs)
- datafile: file contains observables, the format should be .mat, .m or .xls.
- nobs: number of observations used (default all)
- first_obs: first observation (default is first)
- mh_replic: number of observations in each MCMC sequence
- mh_nblocks: number of MCMC sequences

- *mh_jscale*: variance of the jumps in MCMC chain
  - a higher value of *mh_jscale* means bigger steps through the domain of the parameters and lower acceptance ratio.
  - acceptance ratio should be around 0.234.
- *mh_init_scale*: variance of initial draw, it is important to make sure that the different MCMC sequences start in different points.

- MCMC should generate sequence as if drawn from the posterior distribution.
- Minimum requirement is that distribution is same
  - for different parts of the same sequence.
  - across sequence (if you have more than one).
- θ_{ij} the *ith* draw (out of I) in the *jth* sequence (out of J), $\bar{\theta}_j$ is the mean of *jth* sequence, $\bar{\boldsymbol{\theta}}$ is the mean across all available data (pooling all the data).
- Define the between variance

$$\frac{B}{I} = \frac{1}{J-1}\sum_{j=1}^{J}\left(\bar{\theta}_j - \bar{\boldsymbol{\theta}}\right)^2$$

where $\frac{B}{I}$ is the estimator of the variance of sample mean.

- Given a sequence $\{\theta_i\}_{i=1}^{I}$ which is i.i.d. from a random variable y with variance σ^2, the variance of the sample mean $\bar{\theta} = \sum_{i=1}^{I} \theta_i$ is σ^2/I.
- If we have J different i.i.d. sequences from θ, which is denoted by $\{\theta_{ij}\}_{i=1}^{I}$, then for each chain we have the mean $\bar{\theta}_j$, then $\{\bar{\theta}_j\}_{j=1}^{J}$ is an i.i.d. sequence with variance σ^2/I.
- $\frac{B}{I}$ is the estimator of σ^2/I.

- Define the within variance

$$\hat{W} = \frac{1}{J}\sum_{j=1}^{J}\left(\frac{1}{I}\sum_{i=1}^{I}\left(\theta_{ij} - \bar{\theta}_j\right)^2\right), W = \frac{1}{J}\sum_{j=1}^{J}\left(\frac{1}{I-1}\sum_{i=1}^{I}\left(\theta_{ij} - \bar{\theta}_j\right)^2\right)$$

  then $\hat{W}$ and W are both consistent estimators of σ^2, since $\frac{1}{I}\sum_{i=1}^{I}\left(\theta_{ij} - \bar{\theta}_j\right)^2$ is consistent estimator of σ^2.
- Between variance should go to zero, $\lim_{I\to\infty}\frac{B}{I} \longrightarrow 0$. And within variance should settle down $\lim_{I\to\infty}\hat{W} \longrightarrow \sigma^2$.
- In dynare, read line denote W as function of I and blue line denote $\widehat{V} = \hat{W} + \frac{B}{I}\left(1 + \frac{1}{m}\right)$.

- We need red and blue line to get close, and red line to settle down.
- The above can be done for any moment, not just the variance.
- Dynare computes 3 sets of MCMC statistics
  - Interval: The length of the Highest Probability Density interval covering 80% of the posterior distribution.
  - M2: Variance
  - M3: Skewness
- For each of these, dynare computes a statistic related to the within-sequence value of each of these (red) and essentially a sum of the within-sequence statistic and a between-sequence variance (blue)

- For each moment of interest you can calculate the multivariate version, such as covariance matrix.
- These higher-dimensional objects have to be transformed into scalar objects that can be plotted.
- The acceptance rate should be "around" 0.234.
  - A relatively low acceptance rate makes it more likely that the MCMC travels through the whole domain of θ.
  - If the acceptance rate is too high, you can increase *mh_jscale.*

- Impulse responses trace out the response of current and future values of each of the variables to a one-unit increase (or to a one-standard deviation increase, when the scale matters) in the current value of one of the VAR errors, assuming that this error returns to zero in subsequent periods and that all other errors are equal to zero.
- **The implied thought experiment of changing one error while holding the others constant makes most sense when the errors are uncorrelated across equations, so impulse responses are typically calculated for recursive and structural VARs.**

- If we know A, B and the diagonal covariance matrix Σ_u, we can begin from:
$$z_t = A^{-1}Bz_{t-1} + A^{-1}u_t$$
- Suppose we are interested in tracing the dynamics to a shock to the first variable in a two variable VAR: when a shock hits at time 0:
$$u_0 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \bar{z}_0 = \begin{bmatrix} Y_0 \\ R_0 \end{bmatrix} = A^{-1}u_0$$
- For every $s > 0$,
$$\bar{z}_s = A^{-1}B\bar{z}_{s-1}.$$
- Collecting $(\bar{z}_{10}, \bar{z}_{12}, \bar{z}_{13}, \cdots, \bar{z}_{1s}, \cdots)$ and $(\bar{z}_{20}, \bar{z}_{22}, \bar{z}_{23}, \cdots, \bar{z}_{2s}, \cdots)$ as the impluse response of variables z_{1t} and z_{2t} to the structural shock u_{1t} respectively.

- The SVMA (structral vector moving average) representation of VAR is

$$z_t = \Phi(L)\, e_t = \Phi_0 u_t + \Phi_1 u_{t-1} + \Phi_2 u_{t-2} + \cdots + \cdots$$

- Consider the SVMA representation at time $t+s$

$$\begin{bmatrix} z_{1t+s} \\ z_{2t+s} \end{bmatrix} = \begin{bmatrix} \phi_{11,0} & \phi_{12,0} \\ \phi_{21,0} & \phi_{22,0} \end{bmatrix} \begin{bmatrix} u_{1t+s} \\ u_{2t+s} \end{bmatrix} + \cdots + \begin{bmatrix} \phi_{11,s} & \phi_{12,s} \\ \phi_{21,s} & \phi_{22,s} \end{bmatrix} \begin{bmatrix} u_{1t} \\ u_{2t} \end{bmatrix}$$

Result Analysis
IRF

- The structural dynamic multipliers are

$$
\begin{aligned}
\frac{\partial z_{1t+s}}{u_{1t}} &= \phi_{11,s}, \frac{\partial z_{1t+s}}{u_{2t}} = \phi_{12,s} \\
\frac{\partial z_{2t+s}}{u_{1t}} &= \phi_{21,s}, \frac{\partial z_{2t+s}}{u_{2t}} = \phi_{22,s}.
\end{aligned}
$$

- The structural impulse response functions (IRFs) are the plots of $\phi_{ij,s}$ vs. s for $i, j = 1, 2$.
- These plots summarize how unit impulses of the structural shocks at time t impact the level of z at time $t + s$ for different values of s.

- In *stoch_simul* command, the option is $irf = periods$.
- In *estimation* command, the option is *bayesian_irf* which is used to trigger the computation of IRFs. The length of the IRFs are controlled by the option $irf = periods$.

Result Analysis
Forecast Error Variance Decomposition

- Variance decomposition separates the variation in an endogenous variable into the component shocks to the VAR.
- The variance decomposition provides information about the relative importance of each random innovation in affecting the variables in the VAR.
- The SVMA (structral vector moving average) representation of VAR is

$$z_t = \Phi\left(L\right) e_t = \Phi_0 u_t + \Phi_1 u_{t-1} + \Phi_2 u_{t-2} + \cdots + \cdots$$

- The error in forecasting z_t in the future is, for each horizon s:

$$z_{t+s} - E_t z_{t+s} = \Phi_0 u_{t+s} + \Phi_1 u_{t+s-1} + \Phi_2 u_{t+s-2} + \cdots + \Phi_{s-1} u_{t+1}$$

- The variance of the forcasting error is

$$var\left(z_{t+s} - E_t z_{t+s}\right) = \Phi_0 \Sigma_u \Phi_0' + \Phi_1 \Sigma_u \Phi_1' + \Phi_2 \Sigma_u \Phi_2' + \ldots + \Phi_{s-1} \Sigma_u \Phi \tag{4}$$

- Since Σ_u is diagonal, for the first equation

$$\begin{aligned} var\left(z_{1t+s} - E_t z_{1t+s}\right) &= \sigma_1^2(s) = \sigma_1^2\left(\phi_{11,0}^2 + \phi_{11,1}^2 + \cdots + \phi_{11,s-1}^2\right) \\ &\quad +\sigma_2^2\left(\phi_{12,0}^2 + \phi_{12,1}^2 + \cdots + \phi_{12,s-1}^2\right), \end{aligned}$$

- And for the second

$$\begin{aligned} var\left(z_{2t+s} - E_t z_{2t+s}\right) &= \sigma_2^2(s) = \sigma_1^2\left(\phi_{21,0}^2 + \phi_{21,1}^2 + \cdots + \phi_{21,s-1}^2\right) \\ &\quad +\sigma_2^2\left(\phi_{22,0}^2 + \phi_{22,1}^2 + \cdots + \phi_{22,s-1}^2\right). \end{aligned}$$

- The proportion of $\sigma_1^2(s)$ due to shocks in u_{1t} is then

$$\rho_{1,1} = \frac{\sigma_1^2\left(\phi_{11,0}^2 + \phi_{11,1}^2 + \cdots + \phi_{11,s-1}^2\right)}{\sigma_1^2(s)},$$

- The proportion of $\sigma_1^2(s)$ due to shocks in u_{2t} is

$$\rho_{1,1} = \frac{\sigma_1^2\left(\phi_{12,0}^2 + \phi_{12,1}^2 + \cdots + \phi_{12,s-1}^2\right)}{\sigma_1^2(s)}.$$

- The forecast error variance decompositions (FEVDs) for z_{2t+s}

$$\rho_{r,y} = \frac{\sigma_y^2\left(\phi_{21,0}^2 + \phi_{21,1}^2 + \cdots + \phi_{21,s-1}^2\right)}{\sigma_r^2(s)},$$

$$\rho_{r,r} = \frac{\sigma_r^2\left(\phi_{22,0}^2 + \phi_{22,1}^2 + \cdots + \phi_{22,s-1}^2\right)}{\sigma_r^2(s)}.$$

- *conditional_variance_decomposition* = *INTEGER*
- *conditional_variance_decomposition* = *[INTEGER1:INTEGER2]*
- *conditional_variance_decomposition* = *[INTEGER1 INTEGER2 ...]*
- In *stoch_simul* comand, the conditional variance decomposition is computed at calibrated values of parameters.
- In *estimation* comand, these options compute the posterior distribution of the conditional variance decomposition, for the specified period(s).
- Note that this option requires the option moments_varendo to be specified.

- Prior density $p\left(\theta_A|A\right)$, where A represents the model and θ_A, the parameters of that model.
- Conditional density

$$p\left(y|\theta_A, A\right)$$

- Conditional density for dynamic time series models

$$p\left(Y_T|\theta_A, A\right) = p\left(y_0|\theta_A, A\right)\prod_{t=1}^{T} p\left(y_t|Y_{T-1}, \theta_A, A\right)$$

where Y_T are the observations until period T.

- Likelihood function

$$\mathcal{L}\left(\theta_A|Y_T, A\right) = p\left(Y_T|\theta_A, A\right)$$

Result Analysis
Model Comparison

- Marginal density

$$p\left(Y_T|A\right) = \int_{\Theta_A} p\left(Y_T|\theta_A, A\right) d\theta_A = \int_{\Theta_A} p\left(Y_T|\theta_A, A\right) p\left(\theta_A|A\right) d\theta_A$$

- Posterior density

$$p\left(\theta_A|Y_T, A\right) = \frac{p\left(Y_T|\theta_A, A\right) p\left(\theta_A|A\right)}{p\left(Y_T|A\right)}$$

- Unnormalized posterior or posterior kernel

$$p\left(\theta_A|Y_T, A\right) \propto p\left(Y_T|\theta_A, A\right) p\left(\theta_A|A\right)$$

- Posterior predictive density

$$\begin{aligned} p\left(\tilde{Y}|Y_T, A\right) &= \int_{\Theta_A} p\left(\tilde{Y}, \theta_A|Y_T, A\right) d\theta_A \\ &= \int_{\Theta_A} p\left(\tilde{Y}|\theta_A, Y_T, A\right) p\left(\theta_A|Y_T, A\right) \end{aligned}$$

- The ratio of posterior probability of two models is

$$\frac{P\left(A_j|Y_T\right)}{P\left(A_k|Y_T\right)} = \frac{P\left(A_j\right)P\left(Y_T|A_j\right)/P\left(Y_T\right)}{P\left(A_k\right)P\left(Y_T|A_k\right)/P\left(Y_T\right)} = \frac{P\left(A_j\right)}{P\left(A_k\right)}\frac{P\left(Y_T|A_j\right)}{P\left(Y_T|A_k\right)}$$

  in favor of the model A_j versus the model A_k
- The prior odds is $P\left(A_j\right)/P\left(A_k\right)$
- The Bayes factor is $P\left(Y_T|A_j\right)/P\left(Y_T|A_k\right)$
- The posterior odds ratio is $P\left(A_j|Y_T\right)/P\left(A_k|Y_T\right)$, if $P\left(A_j\right) = P\left(A_k\right)$, the posterior odds ratio is same as Bayes factor.

- Laplace approximation

$$P\left(Y_T|A\right) = \int_{\Theta_A} p\left(Y_T|\theta_A, A\right) p\left(\theta_A|A\right) d\theta_A$$

$$\hat{P}\left(Y_T|A\right) = (2\pi)^{k/2} \left|\Sigma_{\theta_A^M}\right|^{-1/2} p\left(Y_T|\theta_A^M, A\right) p\left(\theta_A^M|A\right)$$

where θ_A^M is the posterior mode.

- Harmonic mean

$$P\left(Y_T|A\right) = \int_{\Theta_A} p\left(Y_T|\theta_A, A\right) p\left(\theta_A|A\right) d\theta_A$$

$$\hat{P}\left(Y_T|A\right) = \left[\frac{1}{n}\sum_{i=1}^{n} \frac{f\left(\theta_A^{(i)}\right)}{p\left(\theta_A^{(i)}|Y_T, A\right) p\left(\theta_A^{(i)}|A\right)}\right]^{-1}$$

where $\int f\left(\theta\right) d\theta = 1$, here we can take $f\left(\theta\right) = p\left(\theta_A|A\right)$.

- The convergence of the hamonic mean estimator because

$$
\begin{aligned}
& E\left[\frac{f\left(\theta_A^{(i)}\right)}{p\left(\theta_A^{(i)}|Y_T, A\right)p\left(\theta_A^{(i)}|A\right)}\right] \\
= & \int \frac{f\left(\theta_A^{(i)}\right)}{p\left(\theta_A^{(i)}|Y_T, A\right)p\left(\theta_A^{(i)}|A\right)} p\left(\theta_A^{(i)}|Y_T, A\right) d\theta_A^{(i)} \\
= & \int \frac{f\left(\theta_A^{(i)}\right)}{p\left(\theta_A^{(i)}|Y_T, A\right)p\left(\theta_A^{(i)}|A\right)} \frac{p\left(\theta_A^{(i)}|Y_T, A\right)p\left(\theta_A^{(i)}|A\right)}{P\left(Y_T|A\right)} d\theta_A^{(i)} \\
= & \int \frac{f\left(\theta_A^{(i)}\right)}{P\left(Y_T|A\right)} d\theta_A^{(i)} = \frac{1}{P\left(Y_T|A\right)} \int f\left(\theta_A^{(i)}\right) d\theta_A^{(i)} = \frac{1}{P\left(Y_T|A\right)}
\end{aligned}
$$

- Geweke (1999) modified harmonic mean

$$P\left(Y_T|A\right) = \int_{\Theta_A} p\left(Y_T|\theta_A, A\right) p\left(\theta_A|A\right) d\theta_A$$

$$\hat{P}\left(Y_T|A\right) = \left[\frac{1}{n}\sum_{i=1}^{n} \frac{f\left(\theta_A^{(i)}\right)}{p\left(Y_T|\theta_A^{(i)}, A\right) p\left(\theta_A^{(i)}|A\right)}\right]^{-1}$$

where

$$\begin{aligned} f\left(\theta\right) &= p^{-1}\left(2\pi\right)^{-k/2}\left|\Sigma_{\theta_A^M}\right|^{-1/2} \exp\left\{-\frac{1}{2}\left(\theta-\theta_A^M\right)\Sigma_{\theta_A^M}^{-1}\left(\theta-\theta_A^M\right)'\right\} \\ &\times \left\{\left(\theta-\theta_A^M\right)\Sigma_{\theta_A^M}^{-1}\left(\theta-\theta_A^M\right)' \leq F_{\mathcal{X}_k^2(p)}^{-1}\right\} \end{aligned}$$

with p an arbitray probability and k, the number of eatimated parameters.

- $\Sigma_{\theta_A^M}$ is the minus second order derivative of $p\left(Y_T|\theta_A, A\right)p\left(\theta_A|A\right)$ evaluated at the mode θ_A^M.
- Larger marginal likelihood means a better model.
- In Geweke (1999) modified harmonic mean, $f\left(\theta\right)$ is truncated multivariate normal distribution.
- In Geweke (1999) modified harmonic mean, we need to use the posterior draws and the mode.
- For Laplace method, we only need the mode.

- In Dynare, the marginal density can be compute by *estimation* command.
- The Laplace approximation of marginal density is stored in *oo._MarginalDensity.LaplaceApproximation*.
- The Modified Harmonic Mean of marginal density is stored in *oo._MarginalDensity.ModifiedHarmonicMean* which is used with *mh_replic>0* or *load_mh_file* option.

- Sometimes we need to include dynare in a loop of matlab over different parameters.
- In our AR(1) model, starting with .m file with the following code:
  rho = 0.90;
  se = 0.01;
  save parameterfile rho se
- And in the .mod file, we change the code to
  parameters rho,se; // Parameters of the model
  load parameterfile
  set_param_value('rho', rho);
  set_param_value('se', se);

- To include dynare in a loop of Matlab, we use the following code

```
rho_value=[0.85 0.90 0.95];
N = size(rho_value,2);
se = 0.01;
Simmean=zeros(N,1); % We want to get the simulated mean for
each value of rho
for i=1:N
 rho = rho_value(i);
 save parameterfile rho se
 dynare AR_demo.mod noclearall
 Simmean(i)=oo_.mean;
end
```