Demystifying DSGE Models

4. Bringing DSGE Models to the Data

Outline

- To this point used “calibrated” values for model parameters
- But for realism, need parameters determined by ***actual data*** referring to economy in question
- ***Standard*** inference framework from basic ***econometrics***: Proceed ***as if we knew nothing*** about possible values of parameters
  - ➔***suppose that*** for any parameter β to be estimated
    ($-\infty <= \beta <= +\infty$)
- Fundamental basis for parameter estimation is ***maximum likelihood*** approach

- Recall basic econometrics: system of G equations

  - $Y_{(N \times G)} = Y A_{(G \times G)} + X_{(N \times KG)} B_{(KG \times G)} + U_{(N \times G)}$
  - or $\mathbf{y = Z\Gamma + u}$

    where

Y_i are the current endogenous, and X_i are the pre-determined, variables contained in equation i; they are stacked side-by-side

$$y \equiv \begin{bmatrix} Y_1 \\ Y_2 \\ \cdot \\ \cdot \\ \cdot \\ Y_G \end{bmatrix}; y_i \equiv \begin{bmatrix} Y_{i1} \\ Y_{i2} \\ \cdot \\ \cdot \\ \cdot \\ Y_{iN} \end{bmatrix}; Z \equiv \begin{bmatrix} Y_1 \mid X_1 & 0 & \cdot & \cdot & 0 \\ 0 & \boxed{Y_2 \mid X_2} & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ 0 & \cdot & \cdot & \cdot & Y_G \mid X_G \end{bmatrix}$$

- It then becomes possible to summarise succinctly the various "least squares" estimation methods available:

Recall basic econometrics for <u>single equation:</u>
$y = X\beta + u \Rightarrow \beta = (X'X)^{-1}(X'y)$

- <u>OLS</u>: $\Gamma_{OLS} = (Z'Z)^{-1} Z'y$
- <u>2SLS:</u> $\Gamma_{2SLS} = (Z^{hat\prime}Z^{hat})^{-1} Z^{hat\prime}y$

  where Z^{hat} is just Z with the Y_i elements replaced by $Y^{hat}_i \equiv X\underbrace{(X'X)^{-1} X'y}$
- <u>3SLS:</u> $\Gamma_{3SLS} = (Z^{hat\prime}\ \hat{\Omega}^{-1}\ Z^{hat})^{-1} Z^{hat}\ \hat{\Omega}^{-1} y$

  where $\hat{\Omega} = \hat{\Sigma} \otimes I_N$ and the elements σ_{ij} of Σ are taken from the 2SLS estimation

- Preceding was about well-known "least squares" approach, involving minimising the sum of squared errors (u*) in the equations
- What about the ***maximum likelihood*** approach?
- The log-likelihood function for the system

  $y = Z\Gamma + u$ can be written as

  u !

- $\ln L(y) = -2NG \ln(2\pi) - 2N \ln(|\Sigma|)$
- $+ N \ln(|I\text{-}A|) - 2(y\text{-}Z\Gamma)'(\Sigma^{-1} \otimes I)(y\text{-}Z\Gamma)$
- Various algorithms are then used to maximise this log-likelihood function to obtain the estimated parameters Γ_{MLE}

- For MLE in ***DSGE*** models, write ***linear*** model (eg, corresponding to our log-linearised **Dynare** model) involving ***leads and lags*** as

$$\mathbf{Z_t = AZ_{t-1} + BE_tZ_{t+1} + HX_t}$$

Notice that this equation is <u>exact</u>: ∃ no "error" term – it is "hidden" in X_t

- where
- Z_t ➔ n ***endogenous*** variables at time t
- X_t ➔ k ***exogenous*** shock variables that evolve (usually) according to AR(1): $\mathbf{X_t = DX_{t-1} + \epsilon_t}$
- (If ∃"t-2" etc variables, just ***define new ones*** with only one period lag (eg, $d_t = c_{t-1}$ ➔ $d_{t-1} = c_{t-2}$)

- Binder and Pesaran (2000) showed that

$$\mathbf{Z_t = AZ_{t-1} + BE_tZ_{t+1} + HX_t} \quad [\mathit{shock}\ X_t = \mathbf{D}X_{t-1} + \epsilon_t]$$

has an ***analytical solution*** ***(involving no leads)***

$$\mathbf{Z_t = CZ_{t-1} + PX_t}$$

Notice that this equation is also **exact**: ∃ no "error" term- it is "hidden" in X_t

- where
- C is a ***function of*** coefficients in A and B
- P is a ***function of*** coefficients in A, B, H and D
- These ***cross-equation restrictions*** in DSGE models tend to be ***very limiting***
- ➔ given any values for A, B, H and D ➔ C and P must obey ***very particular patterns ("mappings")***

- ➔ MLE estimates of A, B, H, D, Σ_ϵ and Σ_u are those that ***maximise log-likelihood***

Same thing for exog vars

$$-T\log 2\pi - T\left(\log\left|\Sigma_\epsilon^{-1}\right| + \log\left|\Sigma_u^{-1}\right|\right)$$

Usual sum of squared errors

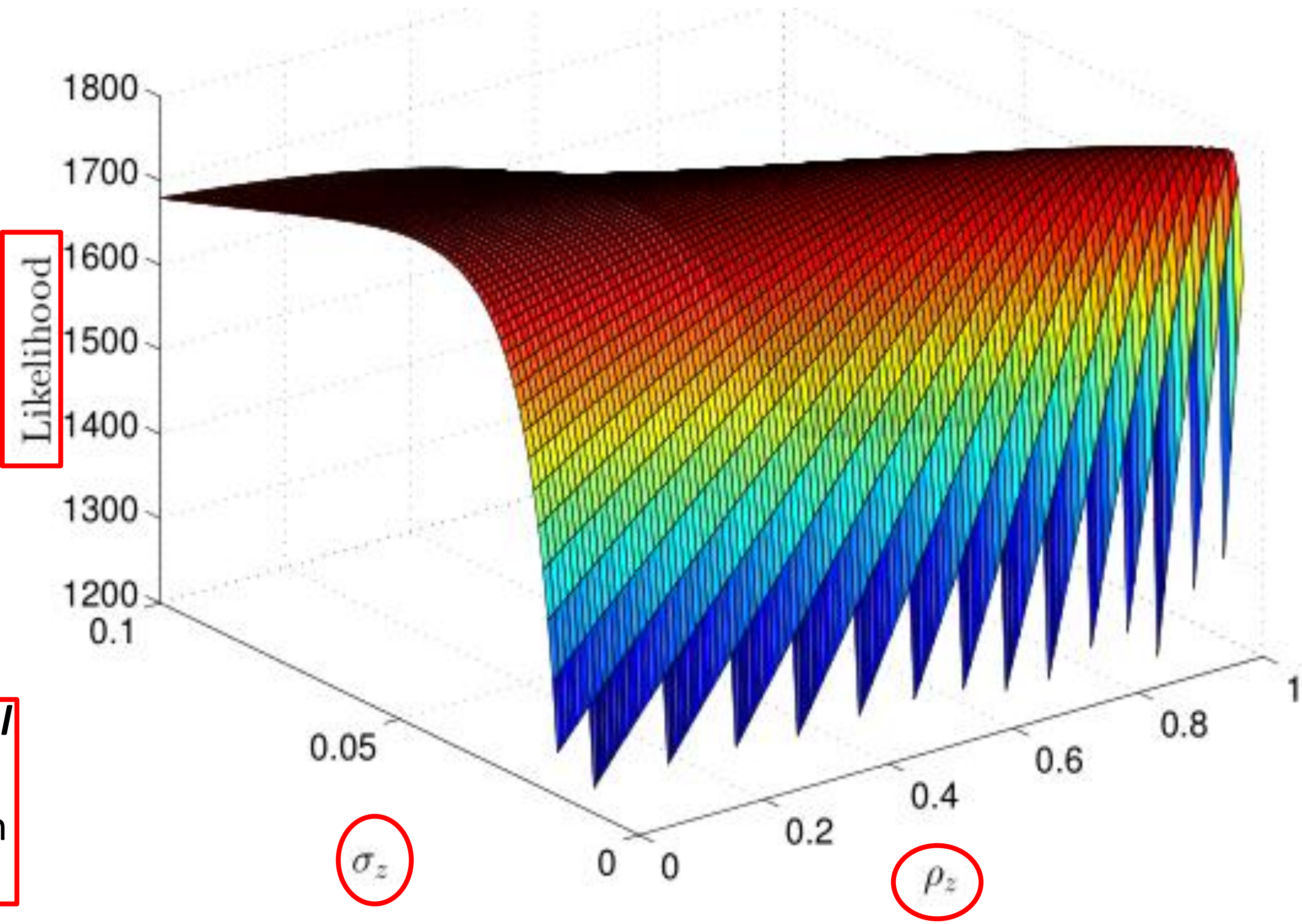
$$-\frac{1}{2}\sum_{k=1}^{T}\left(X_i - DX_{i-1}\right)'\Sigma_\epsilon^{-1}\left(X_i - DX_{i-1}\right)$$

$$-\frac{1}{2}\sum_{k=1}^{T}\left(Z_i - CZ_{i-1} - PX_i\right)'\Sigma_u^{-1}\left(Z_i - CZ_{i-1} - PX_i\right)$$

- Σ_u is var-cov matrix of errors in Binder-Pesaran solution $\mathbf{Z_t = CZ_{t-1} + PX_t}$
- And Σ_ϵ is var-cov matrix of errors in $X_t = \mathbf{D}X_{t-1} + \epsilon_t$
- ***subject to*** DSGE model ***restrictions*** that
  - ***map*** A and B ***into*** C
  - ***map*** A, B, H and D ***into*** P

- Obviously very formidable task
- ➔ not surprising that estimating parameters of even a medium-scale DSGE model is a ***very difficult*** operation
- Very frequently likelihood function has very ***flat*** (or alternatively, very ***rocky***) surface ➔ hard to distinguish which values of parameters actually maximise it
- See example on next slide from a small-scale DSGE model

- Likelihood function is very complicated – even for just two parameters: where is maximum ??



Also, ***local*** vs ***global*** maximum problem

- ***Also***, ***analytical mapping*** of parameter space to log-likelihood function is usually ***not available*** for DSGE models
- In most cases likelihood function has ***no analytical solution*** → information matrix has to be solved for ***numerically***
- But numerical differentiation is ***often highly inaccurate*** for non-linear functions such as those found in DSGE models
- And then there are the dreaded …

Blanchard-Kahn Conditions

- Not infrequently, **Dynare** simply stops with terrifying cryptic message ***"Blanchard-Kahn conditions not satisfied"***

Recall from B-P: $Z_t = CZ_{t-1} + PX_t$

- What does this mean?
- Blanchard and Kahn [1980] established that there was a ***unique stable solution*** to any ***linear*** DSGE model so long as ***two conditions*** were satisfied
- ***First*** (***"counting condition"***): there have to be as many ***unstable eigenvalues (ie, modulus >1)*** in (square) solution ***matrix C*** as there are ***forward-looking*** ("t+1") variables

For a square matrix A, x is an eigenvector if $Ax = \lambda x$ [i.e., $(A - \lambda I)x = 0$]
The scale factor λ is called an eigenvalue. The eigenvalues are the roots of the characteristic equation $|A - \lambda I| = 0$

- ***Second***: another particular matrix which links the unstable "canonical variables" and non-predetermined variables has to be of ***full rank***
- This ***"rank condition"*** can only be checked via a detailed model solution and that requires more ***matrix algebra*** than we want to get into in this course
- ***Appendix 6*** to this lecture goes through algebra in detail for those who want it
- ***Fortunately***, **Dynare** does any required checking, and if all goes well, reports results like those below for simple 3-equation Cho-Moreno model in ***Appendix 1***

- Cho-Moreno's micro-founded NK "3-equation model" is basis for many papers in DSGE literature (see details in Appendix 1):

$$\pi_t = \delta\pi_{t+1} + (1-\delta)\pi_{t-1} + \lambda y_t + \varepsilon_{ast}$$ NK Phillips Curve (AS)

$$y_t = \mu y_{t+1} + (1-\mu)y_{t-1} - \phi(r_t - \pi_{t+1}) + \varepsilon_{ist}$$ Euler Equation (IS)

$$r_t = \rho r_{t-1} + (1-\rho)(\beta\pi_{t+1} + \gamma y_t) + \varepsilon_{mpt}$$ Taylor Rule

Variable	LaTeX	Description
`del`	δ	Phillips Curve expected inflation factor
`lam`	λ	Phillips Curve output factor
`mu`	μ	IS expected output factor
`phi`	ϕ	IS real rate factor
`roe`	ρ	Taylor Rule interest smoothing
`bet`	β	Taylor Rule inflation weight
`gam`	γ	Taylor Rule output weight
`sigas`	σ_{as}	std dev of AS shock process
`sigis`	σ_{is}	std dev of IS shock process
`sigmp`	σ_{mp}	std dev of MonPol shock process

- In Cho-Moreno model we have

```
Equation number 1 : 0 : New Keynesian Phillips Curve
Equation number 2 : 0 : Dynamic IS Curve
Equation number 3 : 0 : Taylor Rule
```

The “0” means that the model equation is consistent with the initial parameter values

EIGENVALUES:

Modulus	Real	Imaginary
0.7874	0.7874	0
0.9002	0.9001	0.01533
0.9002	0.9001	-0.01533
1.008	1.008	0
1.099	1.099	0

Oscillatory

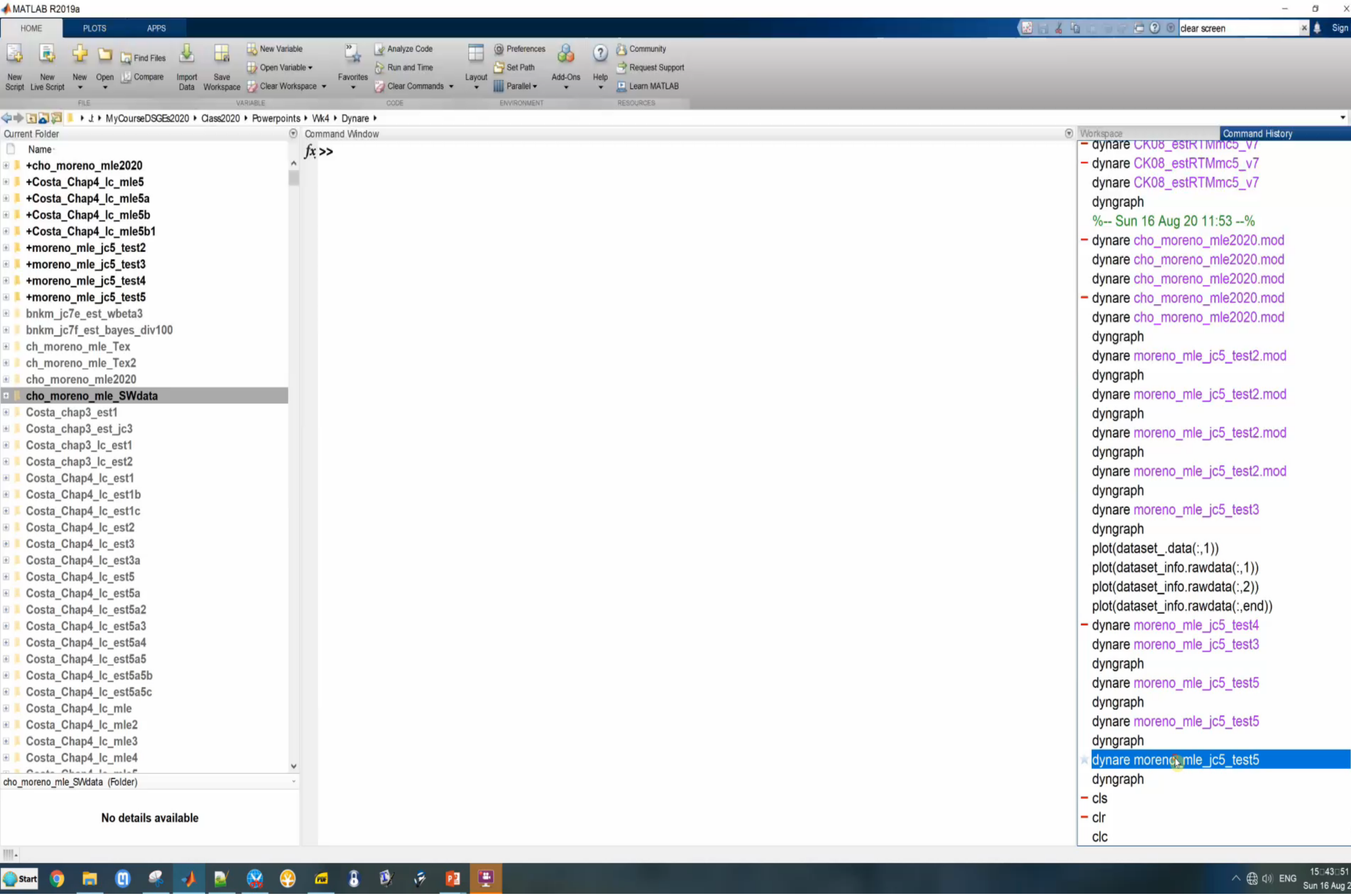
There are 2 eigenvalue(s) larger than 1 in modulus
for 2 forward-looking variable(s)

Counting condition

The rank condition is verified.

- In Cho-Moreno model, ∃ 10 parameters to be estimated ➔ ***Likelihood function*** will be ***10-dimensional*** and quite possibly rather ***flat or rocky***
- ➔ ***may be hard*** to find a solution to optimisation problem [eg, 10^{1000} combinations; 1 billion = 10^9 !!]
- In practice, ∃ many ***different algorithms*** for maximising Likelihood function [eg, ***DFP*** used in my PhD thesis]
- Most are based on doing a "line search": crawls along changing one parameter at a time, attempting to climb higher on likelihood surface
- If algorithm hits a ***cliff*** or ***flat surface***, it moves in another direction to find better parameter estimates

- Obviously this process is not very efficient and may easily fail
- **Dynare** default algorithm is so-called ***"csminwel"*** of Chris Sims
- Other (sometimes more efficient) algorithms can be chosen by including ***"mode_compute=??"*** among options in estimation command (?? for csminwel is 4)
- Let's try it on Cho-Moreno, with ***"mode_compute=8"*** which corresponds to **Dynare** implementation of ***Nelder-Mead simplex-based optimisation*** routine:


MATLAB R2019a
HOME
PLOTS
APPS
clear screen
Sign In
New Script
New Live Script
New
Open
Find Files
Compare
Import Data
Save Workspace
New Variable
Open Variable
Clear Workspace
Favorites
Analyze Code
Run and Time
Clear Commands
Layout
Preferences
Set Path
Parallel
Add-Ons
Help
Community
Request Support
Learn MATLAB
FILE
VARIABLE
CODE
ENVIRONMENT
RESOURCES
MyCourseDSGEs2020 ▸ Class2020 ▸ Powerpoints ▸ Wk4 ▸ Dynare ▸
Current Folder
Name
+cho_moreno_mle2020
+Costa_Chap4_lc_mle5
+Costa_Chap4_lc_mle5a
+Costa_Chap4_lc_mle5b
+Costa_Chap4_lc_mle5b1
+moreno_mle_jc5_test2
+moreno_mle_jc5_test3
+moreno_mle_jc5_test4
+moreno_mle_jc5_test5
bnkm_jc7e_est_wbeta3
bnkm_jc7f_est_bayes_div100
ch_moreno_mle_Tex
ch_moreno_mle_Tex2
cho_moreno_mle2020
cho_moreno_mle_SWdata
Costa_chap3_est1
Costa_chap3_est_jc3
Costa_chap3_lc_est1
Costa_chap3_lc_est2
Costa_Chap4_lc_est1
Costa_Chap4_lc_est1b
Costa_Chap4_lc_est1c
Costa_Chap4_lc_est2
Costa_Chap4_lc_est3
Costa_Chap4_lc_est3a
Costa_Chap4_lc_est5
Costa_Chap4_lc_est5a
Costa_Chap4_lc_est5a2
Costa_Chap4_lc_est5a3
Costa_Chap4_lc_est5a4
Costa_Chap4_lc_est5a5
Costa_Chap4_lc_est5a5b
Costa_Chap4_lc_est5a5c
Costa_Chap4_lc_mle
Costa_Chap4_lc_mle2
Costa_Chap4_lc_mle3
Costa_Chap4_lc_mle4
cho_moreno_mle_SWdata (Folder)
No details available
Command Window
fx >>
Workspace
Command History
dynare CK08_estRTMmc5_v7
dynare CK08_estRTMmc5_v7
dyngraph
%-- Sun 16 Aug 20 11:53 --%
dynare cho_moreno_mle2020.mod
dynare cho_moreno_mle2020.mod
dynare cho_moreno_mle2020.mod
dynare cho_moreno_mle2020.mod
dynare cho_moreno_mle2020.mod
dyngraph
dynare moreno_mle_jc5_test2.mod
dyngraph
dynare moreno_mle_jc5_test2.mod
dyngraph
dynare moreno_mle_jc5_test2.mod
dyngraph
dynare moreno_mle_jc5_test2.mod
dyngraph
dynare moreno_mle_jc5_test3
dyngraph
plot(dataset_.data(:,1))
plot(dataset_info.rawdata(:,1))
plot(dataset_info.rawdata(:,2))
plot(dataset_info.rawdata(:,end))
dynare moreno_mle_jc5_test4
dynare moreno_mle_jc5_test3
dyngraph
dynare moreno_mle_jc5_test5
dyngraph
dynare moreno_mle_jc5_test5
dyngraph
dynare moreno_mle_jc5_test5
dyngraph
cls
clr
clc
Start
ENG
Sun 16 Aug 20

- MLE works for Cho-Moreno! Output is:

```
Simplex iteration number: 2729-4-3
Simplex move:             reflect-0
Objective function value: 704.1908
Mode improvement:         0
Norm of dx:               0
Norm of dSimplex:         0.00020535
Crit. f:                  3.6954e-08
Crit. x:                  9.1275e-05
```

Nelder-Mead algorithm stuff

```
CONVERGENCE ACHIEVED AFTER 2729 ITERATIONS!

Final value of minus the log posterior (or likelihood):704.190819

RESULTS FROM MAXIMUM LIKELIHOOD ESTIMATION
parameters
    Estimate    s.d. t-stat

del   0.5251  0.0119 44.0944
lam   0.0098  0.0070  1.3916
mu    0.5157  0.0204 25.2646
phi   0.0031  0.0029  1.0583
roe   0.8869  0.0240 36.9025
bet   1.2421  0.2734  4.5424
gam   2.5596  0.6858  3.7325

standard deviation of shocks
    Estimate    s.d. t-stat

eas   0.6248  0.0350 17.8321
eis   0.4135  0.0241 17.1403
emp   0.8579  0.0458 18.7422
```

Most parameters are highly statistically significant

- What about Week 2 NK-P model? Recall:

J:\MyCourseDSGEs2025\Powerpoints\Wk2\BasicNK2025_Price.mod

```
//1-Labour Supply
sigma*c + phi*l = w - p;
//2-Euler Equation
(sigma/beta)*(c(+1)-c)=(Rss/Pss)*(r(+1)-p(+1));
//3-Law of motion of Capital
k(+1) = (1-delta)*k + delta*i;
//4-Production Function
y = a + alpha*k + (1-alpha)*l;
//5-Demand for Capital
k = y - (r - p);
//6-Demand for Labour
l = y - (w - p);
//7-Marginal Cost
mc = (1-alpha)*w + alpha*r - a;
//8-Phillips Curve
pi = beta*pi(+1)+((1-theta)*(1-beta*theta)/theta)*(mc-p);
//9-Gross Inflation Rate
pi = p - p(-1);
//10-Goods Market Equilibrium Condition
Yss*y = Css*c + Iss*i;
//11-Productivity Shock
a = rhoa*a(-1) + e;
```

New !!

- MLE ***fails*** for week 2 NK-P model:

```
OPTIMIZATION PROBLEM!
 (minus) the hessian matrix at the "mode" is not positive definite!
=> variance of the estimated parameters are not positive.
You should try to change the initial values of the parameters using
the estimated_params_init block, or use another optimization routine.
[Warning: The results below are most likely wrong!]
```

```
RESULTS FROM MAXIMUM LIKELIHOOD ESTIMATION
parameters
         Estimate    s.d.  t-stat

sigma      1.9958     NaN     NaN
phi        1.4998     NaN     NaN
alpha      0.3268     NaN     NaN
delta      0.0837     NaN     NaN
rhoa       0.9990     NaN     NaN
psi        7.9998     NaN     NaN
theta      0.7816     NaN     NaN
thetaw     0.7477     NaN     NaN
psiw      21.0000     NaN     NaN

standard deviation of shocks
         Estimate    s.d.  t-stat

e          0.5000     NaN     NaN
eps_1      0.5001     NaN     NaN
eps_2      0.5259     NaN     NaN
```

```
// for estimation MLE setup
estimated_params;
sigma, 2, 0, 100;
phi, 1.5, 0, 100;
alpha, 0.35, 0, 1;
%beta, 0.985, 0, 1;
delta, 0.025, 0, 0.99;
rhoa, 0.95, 0, 0.999;
psi, 8, 0, 100;
theta, 0.75, 0, 0.999;
thetaw, 0.75;
psiw, 21, 0, 100;

stderr e, 0.5,0,100;
stderr eps_1, 0.5,0,100;
stderr eps_2, 0.5,0,100;
end;

estimated_params_init(use_calibration);
end;
```

“Regularised” ML Estimation

- As seen, MLE is very problematic, hence basic version rarely used as such for complex DSGE estimation
- Researchers therefore turn to an alternative method: ***"Regularised" (ie, constrained) MLE***
- Recall that for basic MLE, we proceed ***as if we knew nothing*** about possible values of parameters

  ➔ suppose that for any β, range is **($-\infty <= \beta <= +\infty$)**
- But typically we have some ***prior knowledge*** which enables us to feel comfortable with ***constraining*** parameters to a given region; for example
  – a rate of time discount or a capital or labour share must be positive and less than one **($0 < \beta < 1$)**
  – similarly for Calvo price or wage parameters
  – elasticities must be positive; depreciation rates should be between 1 and 10%; etc

- In such cases, it would be ***inefficient*** to ignore prior information
- Instead, it should be used in estimation process itself
- How?
- By "penalising" likelihood function for parameter values lying ***outside*** ranges indicated by ***prior*** information

- Recall that

> Binder and Pesaran (2000) showed that
>
> $\mathbf{Z_t = AZ_{t-1} + BE_tZ_{t+1} + HX_t}$ *[shock $X_t = DX_{t-1} + \epsilon_t$]*
>
> has an ***analytical solution*** ***(involving no leads)***
>
> $\mathbf{Z_t = CZ_{t-1} + PX_t}$

- and that ML estimates of A, B, H, D, Σ_u and Σ_ϵ are those that maximise ***log-likelihood***

This must be ***positive definite*** for inversion; hence requirement earlier on "***Hessian***"

$$-T\log 2\pi - T\left(\log\left|\Sigma_\epsilon^{-1}\right| + \log\left|\Sigma_u^{-1}\right|\right)$$
$$-\tfrac{1}{2}\sum_{k=1}^{T}\left(X_i - DX_{i-1}\right)'\Sigma_\epsilon^{-1}\left(X_i - DX_{i-1}\right)$$
$$-\tfrac{1}{2}\sum_{k=1}^{T}\left(Z_i - CZ_{i-1} - PX_i\right)'\Sigma_u^{-1}\left(Z_i - CZ_{i-1} - PX_i\right)$$

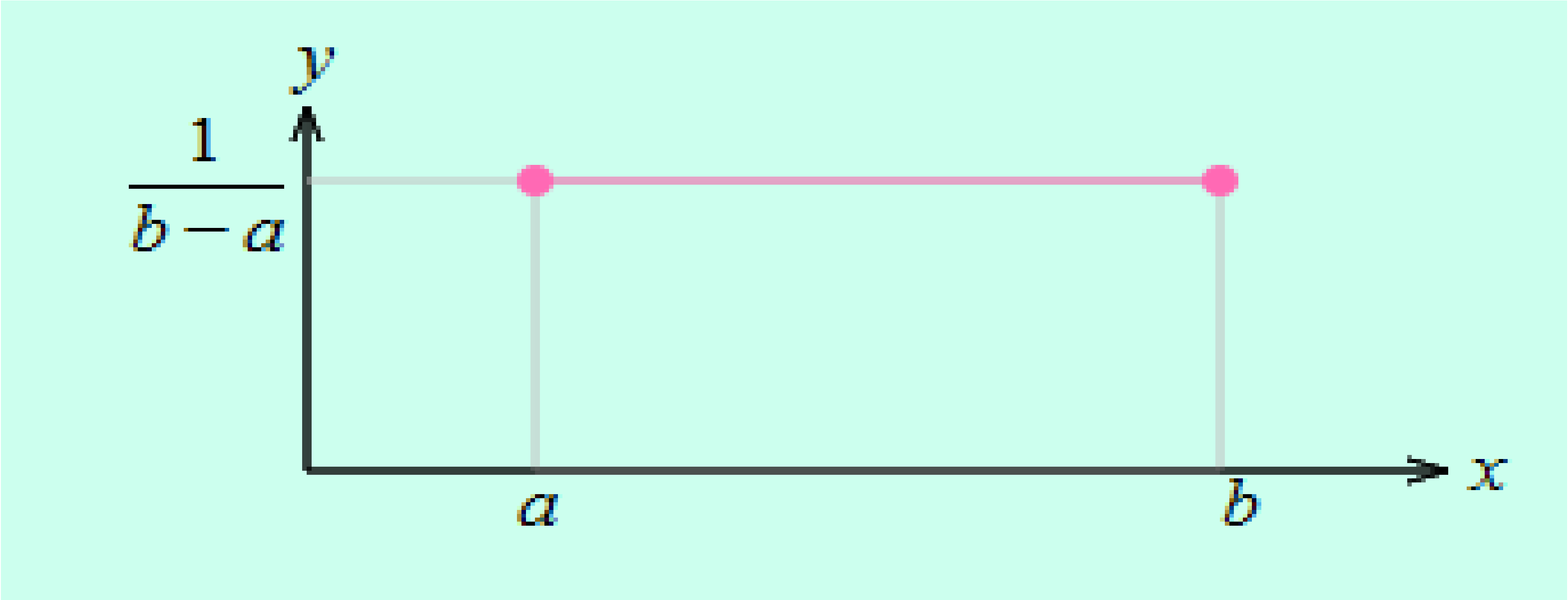
- ***subject to*** DSGE model ***restrictions*** that ***map original*** model parameters A, B, H, D ***into*** C and P above

- So MLE is ***already constrained*** (implicitly) by DSGE model ***cross-equation restrictions***
- Now ***add to constraints*** with a "***penalty function***" which ***reduces value*** of revised likelihood function whenever "candidate" parameter values fall ***outside*** ranges indicated by prior information
- How to do this?
- Need first to specify what ***prior information*** is to be used – ***eg***, range for mean (and variance?) of estimators

- Many ways of doing this have been proposed
- But ***easiest way*** is to use what is known as ***Bayesian analysis***
- (The Reverend Thomas [!]) ***Bayes*** demonstrated that a ***posterior distribution*** of estimated parameters, incorporating ***all*** constraints, could be calculated from ***likelihood LogL*** and a ***prior distribution*** (specifying mean, variance and ***basic shape*** of additional constraints)
- All that we need to add is this prior distribution – ***rest is all mechanical, done by Dynare***

- ***All information*** about parameter set θ from ***data*** is conveyed through ***likelihood***: likelihood principle ***always holds***!
- If ***prior*** distribution is defined as ***Uniform Distribution*** over entire interval (-∞, +∞) then obviously ***no additional constraint*** is imposed
- Hence result from Bayes with this prior = ***unconstrained*** [other than DSGE] MLE – see Cho-Moreno example in ***Appendix 1***
- But if Uniform prior is ***truncated*** to a particular range [eg, 0 to 1], result will be a form of ***constrained MLE*** ("Regularised MLE")

- Below is a ***truncated*** Uniform Distribution in which we give ***equal probability weight*** to all possible values ***within a given range*** (in this example range is [a,b], and therefore probability is 1/(b-a) for all points)



- ∃ ***many other possible ways*** to incorporate this prior information
- Choosing ***appropriate*** prior for parameters is both tricky and very important
- Should ***domain*** of prior over each parameter be ***bounded?***
- Should it be ***opened*** on either or both sides?
- Should ***shape*** of prior distribution be ***symmetric?***
- ***Skewed?***
- If so, on which side?

- Typically, use ***five distributions*** which relate to these characteristics: Uniform, Normal, Beta, Gamma and Inverse Gamma

Shape	Name	Support	Example(s)
normal_pdf	$\mathcal{N}(\mu, \sigma)$	$(-\infty, +\infty)$	Policy parameters, utility curvatures,
gamma_pdf	$\mathcal{G}(\mu, \sigma)$	$(0, +\infty)$	Elasticities, utility curvatures, trends
beta_pdf	$\mathcal{B}(\mu, \sigma)$	$[0, 1]$	Probabilities, AR-MA terms, shares...
inv_gamma_pdf	$\mathcal{IG}(\mu, \sigma)$	$(0, +\infty)$	Standard deviations
uniform_pdf	$\mathcal{U}(\mu, \sigma)$	$(-\infty, +\infty)$	Same as normal distribution

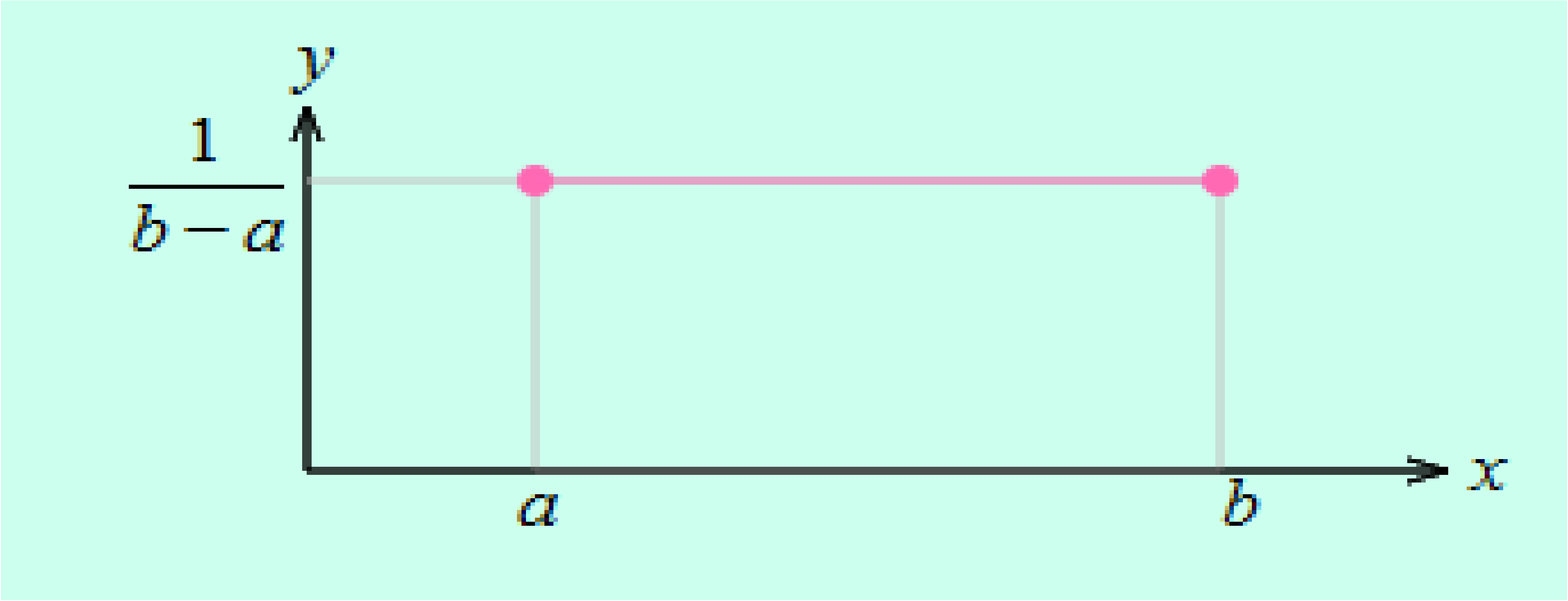
Table 1: Prior distributions available in Dynare

- More detail on each in the slides below

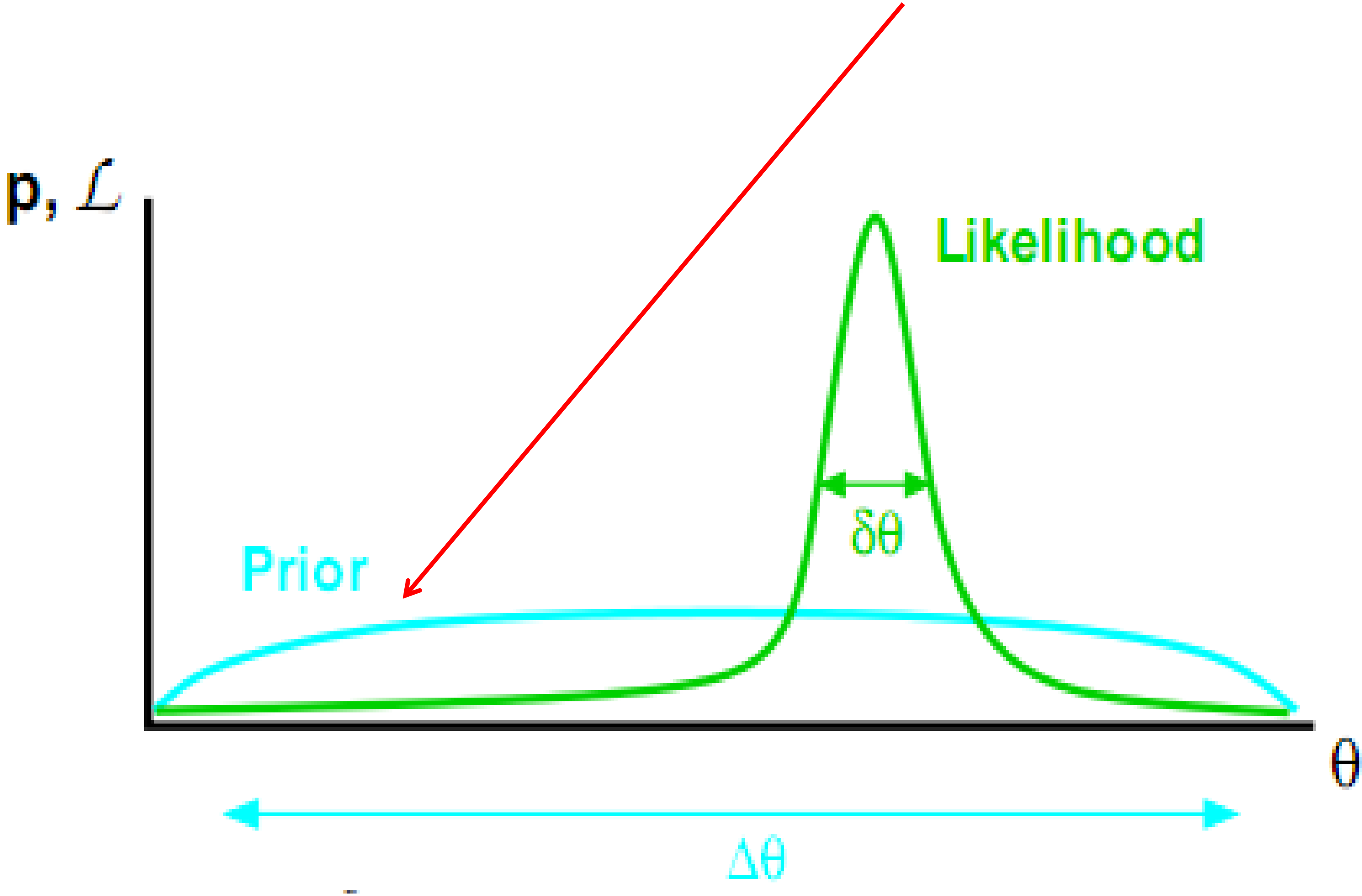
Note: Dynare now also allows for the Weibull distribution

Uniform Distribution

- Already seen - this is an "uninformative" distribution, or distribution of "***ignorance***": we have no idea what value of parameter should be, so we give ***equal probability weight*** to all possible values within a given range (in this example range is [a,b], and therefore probability is 1/(b-a) for all points)

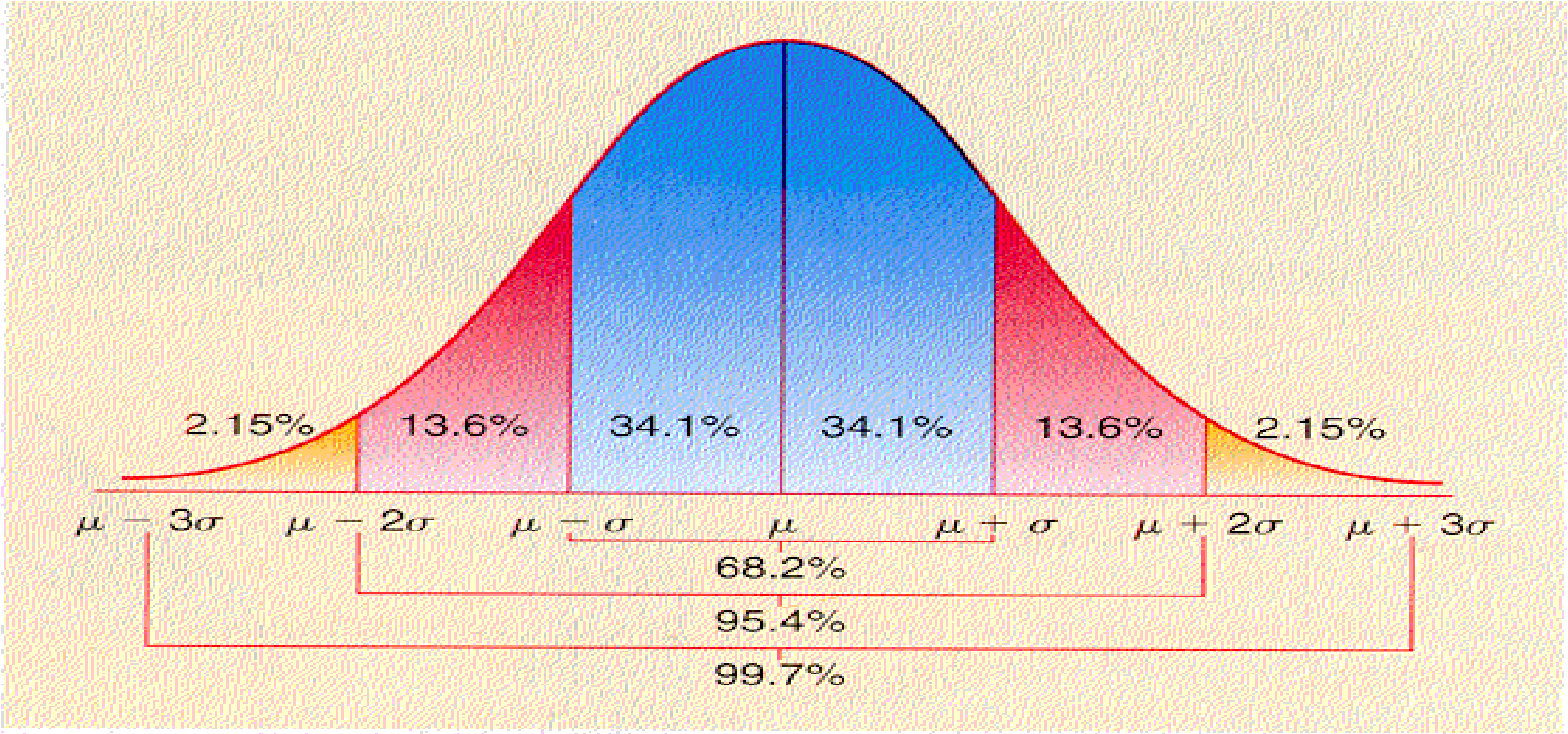


A not-quite-Uniform "flattish" prior



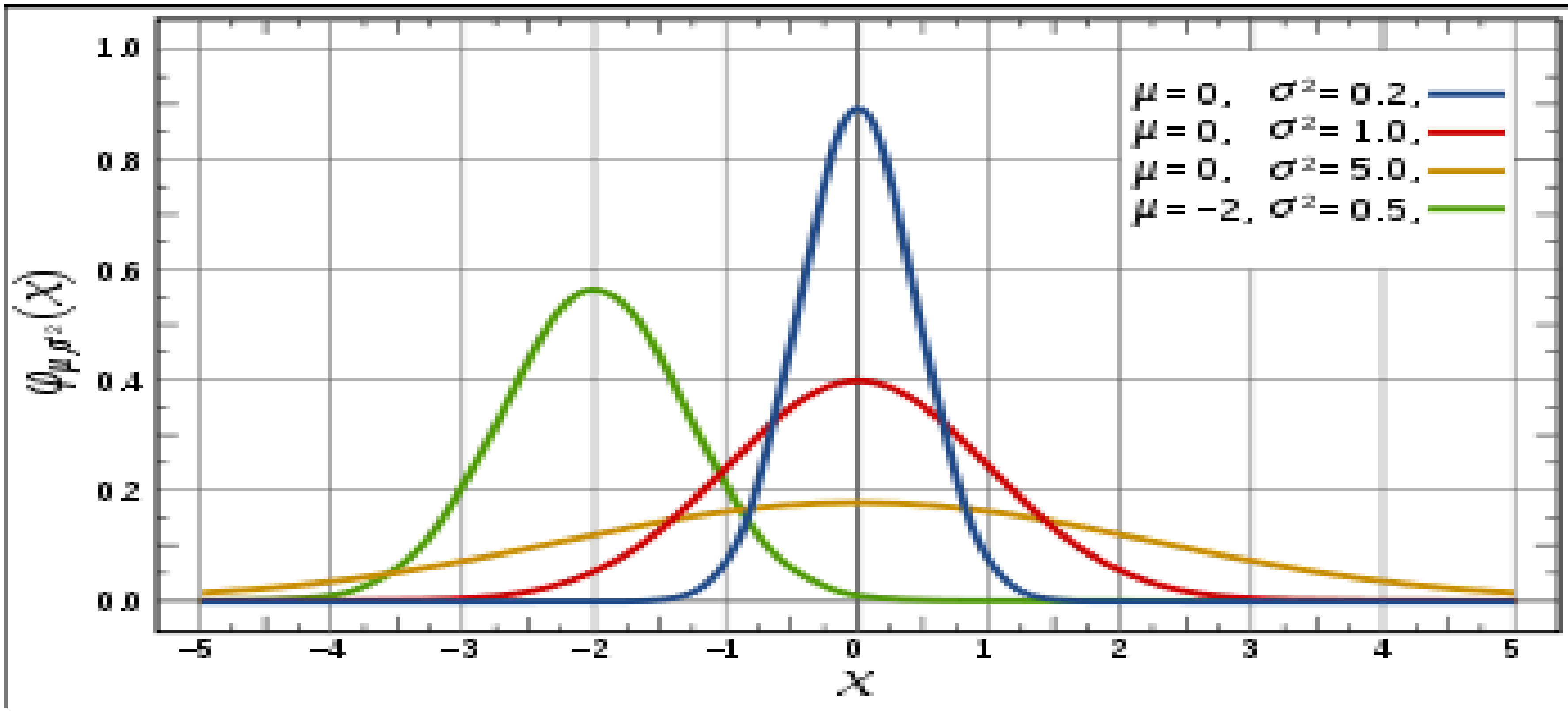
Normal Distribution

- This is an “informative” Normal distribution, in which ***68.2% of observations*** lie within 1 standard deviation (σ) of mean (μ), 95.4% within 2σ, and 99.7% ***[“all”]*** within 3σ



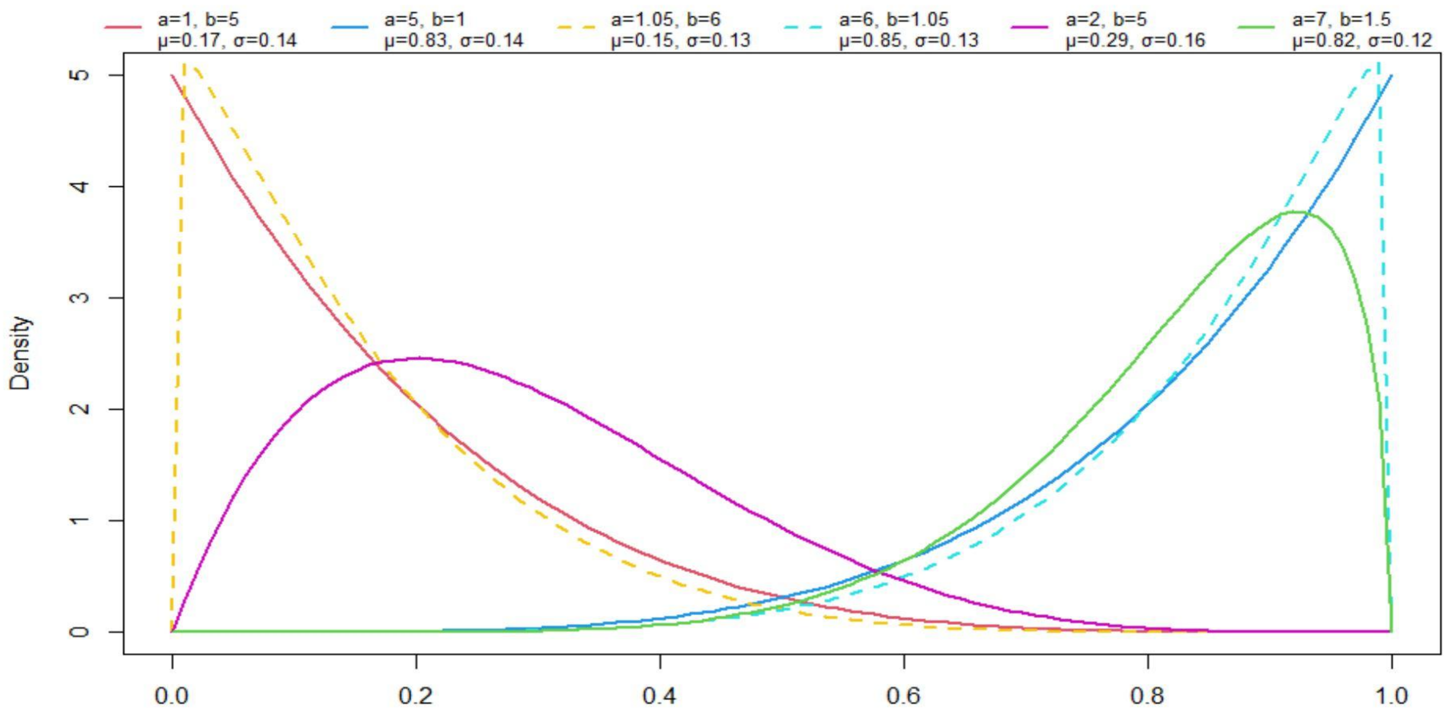
Normal Distribution

- Depending on how large is variance, Normal distribution may be very spread out (**gold**), or very sharply peaked (**blue**), as seen below



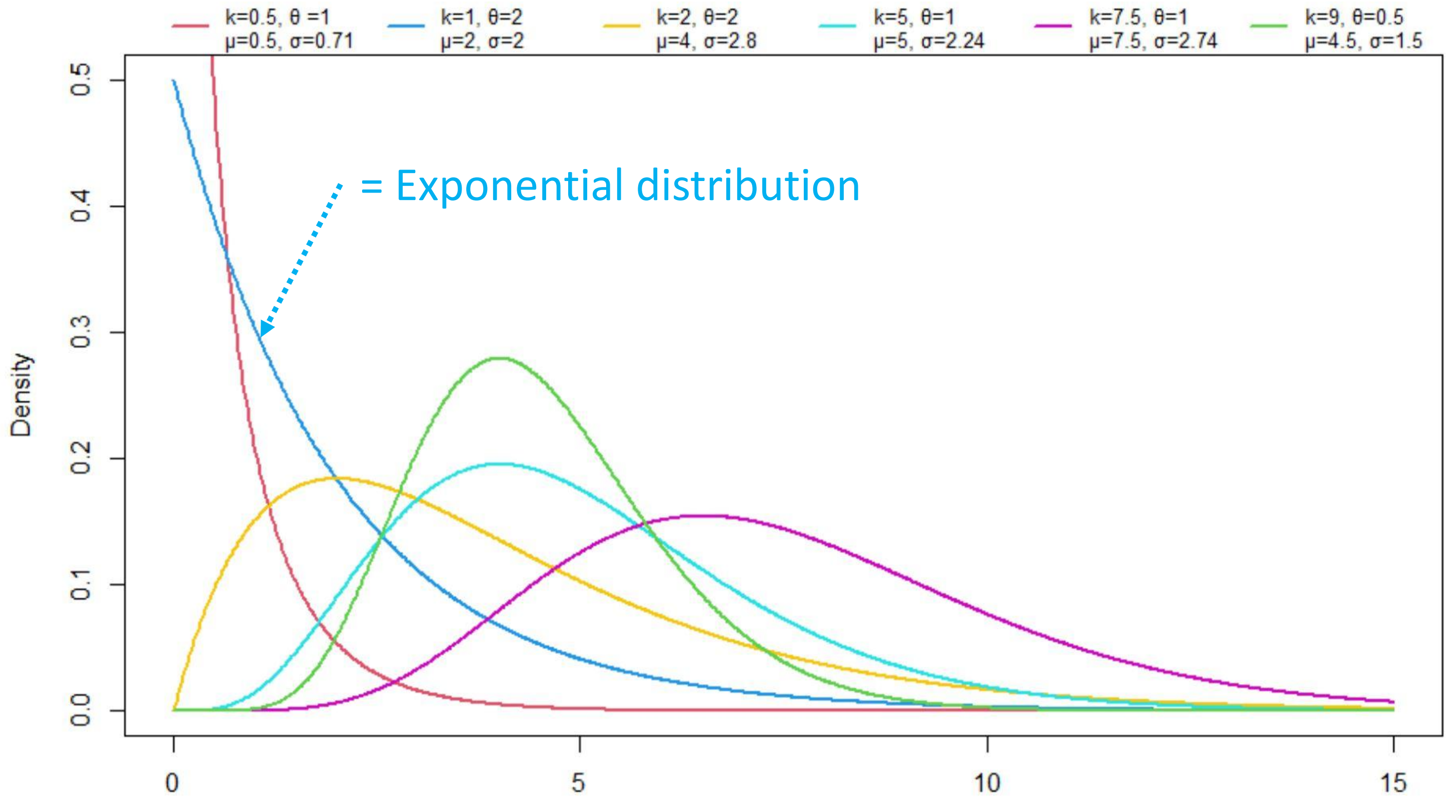
Beta Distribution

- "informative" beta distribution can take on many shapes but it is always ***bounded in interval [0, 1]***, hence useful for parameters like rate of time preference, Calvo parameters, AR(1) parameters ..



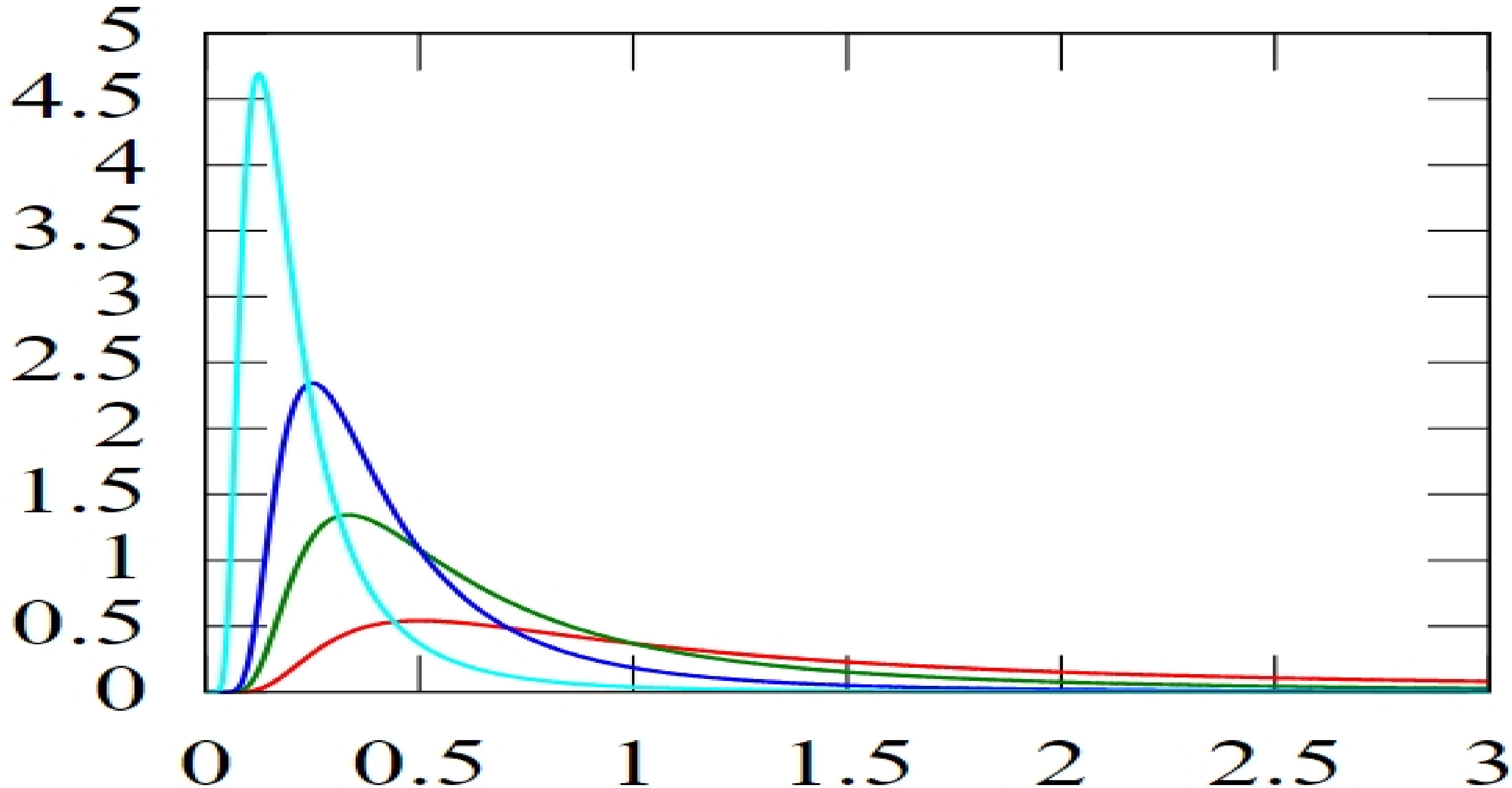
Gamma Distribution

- Like Beta, this distribution has many shapes, but its range is not limited to interval [0, 1]; it may range from 0 to ∞ (but ***always positive***, notice)

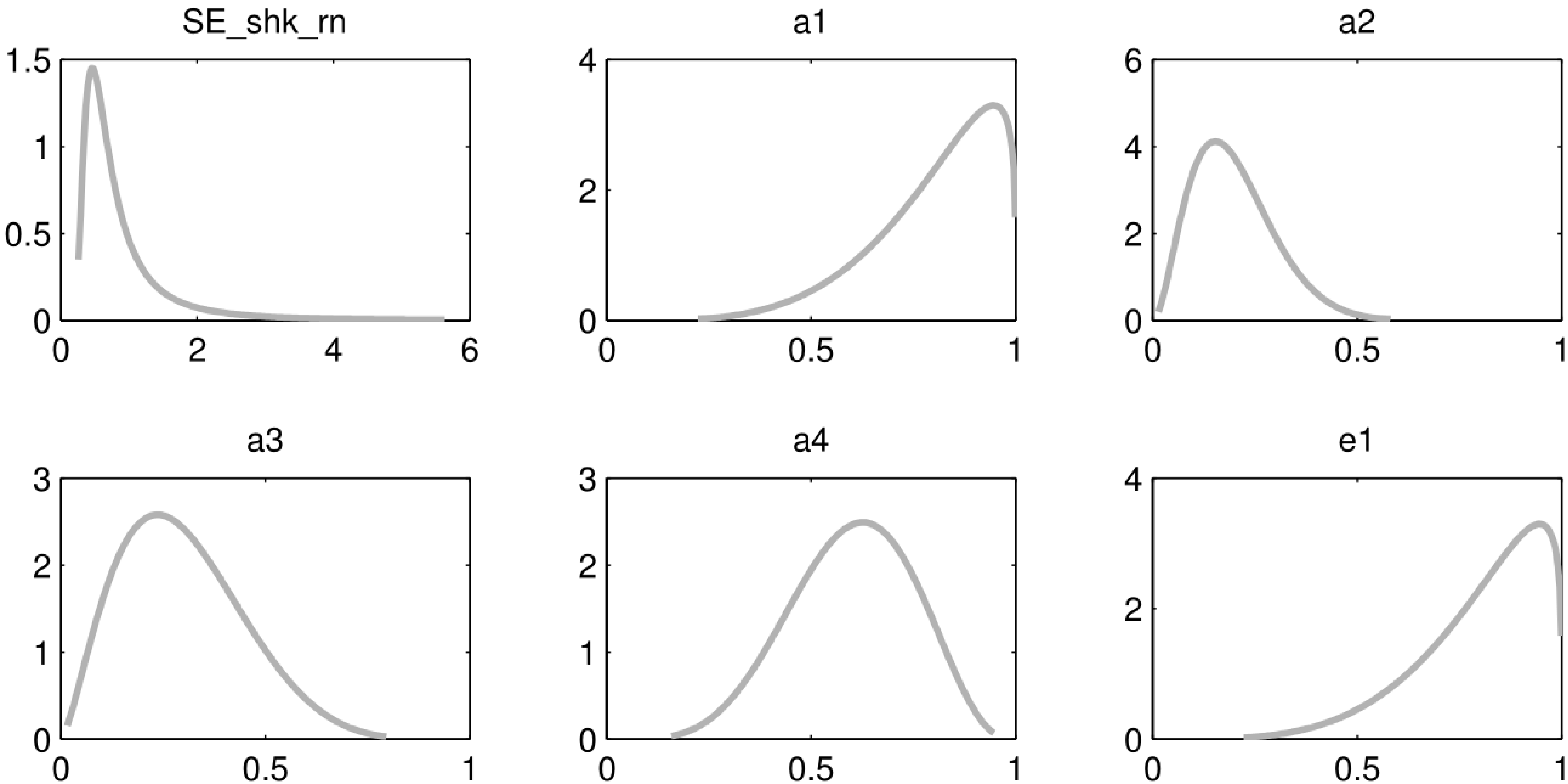


Inverse Gamma Distribution

- Derived from Gamma, this distribution is usually used for ***variances***, as its range is [0 to ∞] (***always positive***) and ***skewed to left***



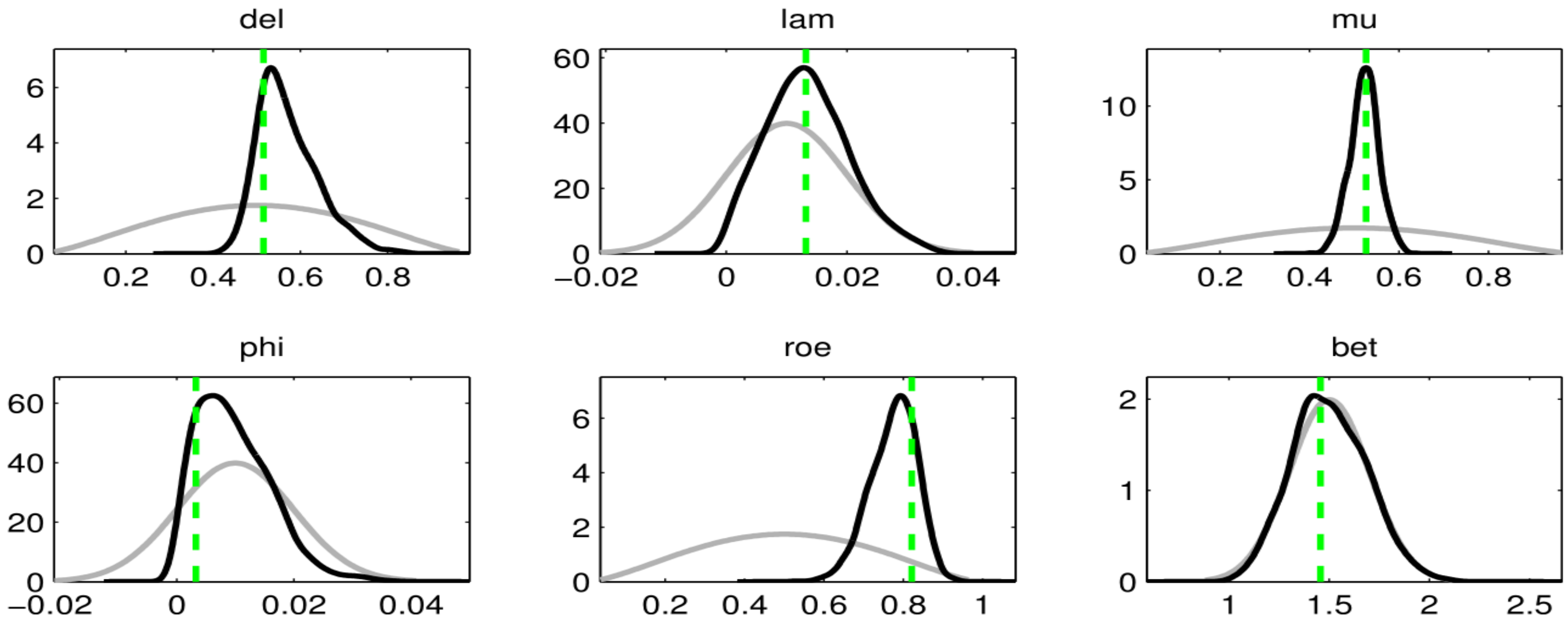
- So, one might choose priors such as those below to implement Bayesian version of Regularised MLE



- How do we implement Bayesian version of Regularised MLE (***RegMLE***) in **Dynare**?
- ➔ requires (***very simple) addition*** to code of a section on "estimated parameters":

```
estimated_params;
% param name, initval, lb, ub, prior_shape, prior_p1, prior_p2
stderr ea,0.4618,0.01,3,INV_GAMMA_PDF,0.1,2;
...
cmap,.7652,0.01,.9999,BETA_PDF,0.5,0.2;
cmaw,.8936,0.01,.9999,BETA_PDF,0.5,0.2;
csadjcost,6.3325,2,15,NORMAL_PDF,4,1.5;
csigma,1.2312,0.25,3,NORMAL_PDF,1.50,0.375;
chabb,0.7205,0.001,0.99,BETA_PDF,0.7,0.1;
cprobw,0.7937,0.3,0.95,BETA_PDF,0.5,0.1;
csigl,2.8401,0.25,10,GAMMA_PDF,2,0.75;
...
end;
```

- ***RegMLE-Bayesian*** estimation provides us with ***point estimates*** of individual parameters and their variances
- But from where do those nice graphs (like that below) found in so many DSGE papers come?

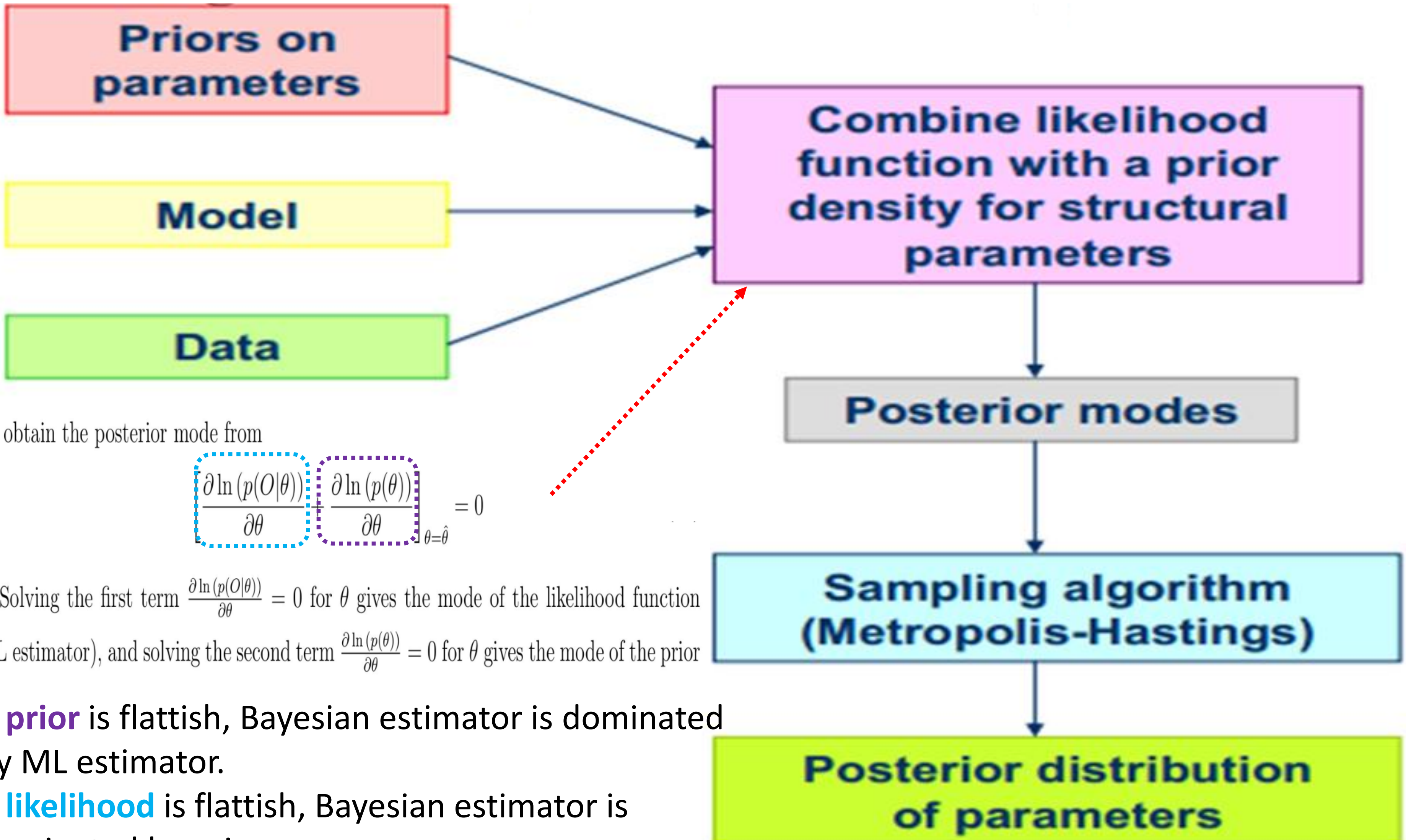


- → to obtain these graphs, need to construct a ***sampling distribution*** for parameters
- ***Random-Walk Metropolis-Hastings*** (RW-MH) is most commonly used method ***to obtain*** required sampling distribution since (usually) doing so ***analytically*** is ***impossible***
- ***General idea*** of ***RW-MH*** algorithm: build on fact that (***Central Limit Theorem***)
  – under general conditions ***distribution*** of parameters will be ***asymptotically Normal***
  – ***whatever*** is ***initial*** specification of prior (uniform, beta, gamma, etc.)

- **RW-MH** is a "***Markov Chain Monte Carlo***" (MCMC) algorithm
- Works by generating a ***sequence of sample values*** э as number of sample values increases ➔ sample distribution ***more closely approximates*** (unknown) "true" parameter distribution
- Sample values produced ***iteratively***
- Distribution of ***next*** sample value dependent ***only*** on ***current*** sample value [***Markov Chain*** aspect - zero memory]
- RW-MH algorithm ***randomly*** [***Monte Carlo*** aspect] attempts to move about sample space

- RW-MH sometimes ***accepts*** these random moves (in which case new [“candidate”] value is used in next iteration)
- RW-MH sometimes ***rejects moves*** (in which case candidate value is ***discarded***, and current value is ***reused*** in next iteration)
- So-called “***acceptance ratio***” indicates what proportion of ***possible*** moves result in ***actual*** moves
- When number of parameters (“dimensions”) is high, finding right ***jumping width (“mh_jscale”)*** to use from move-to-move can be very difficult, and ***convergence very slow***

Bayesian estimation process summary



We obtain the posterior mode from

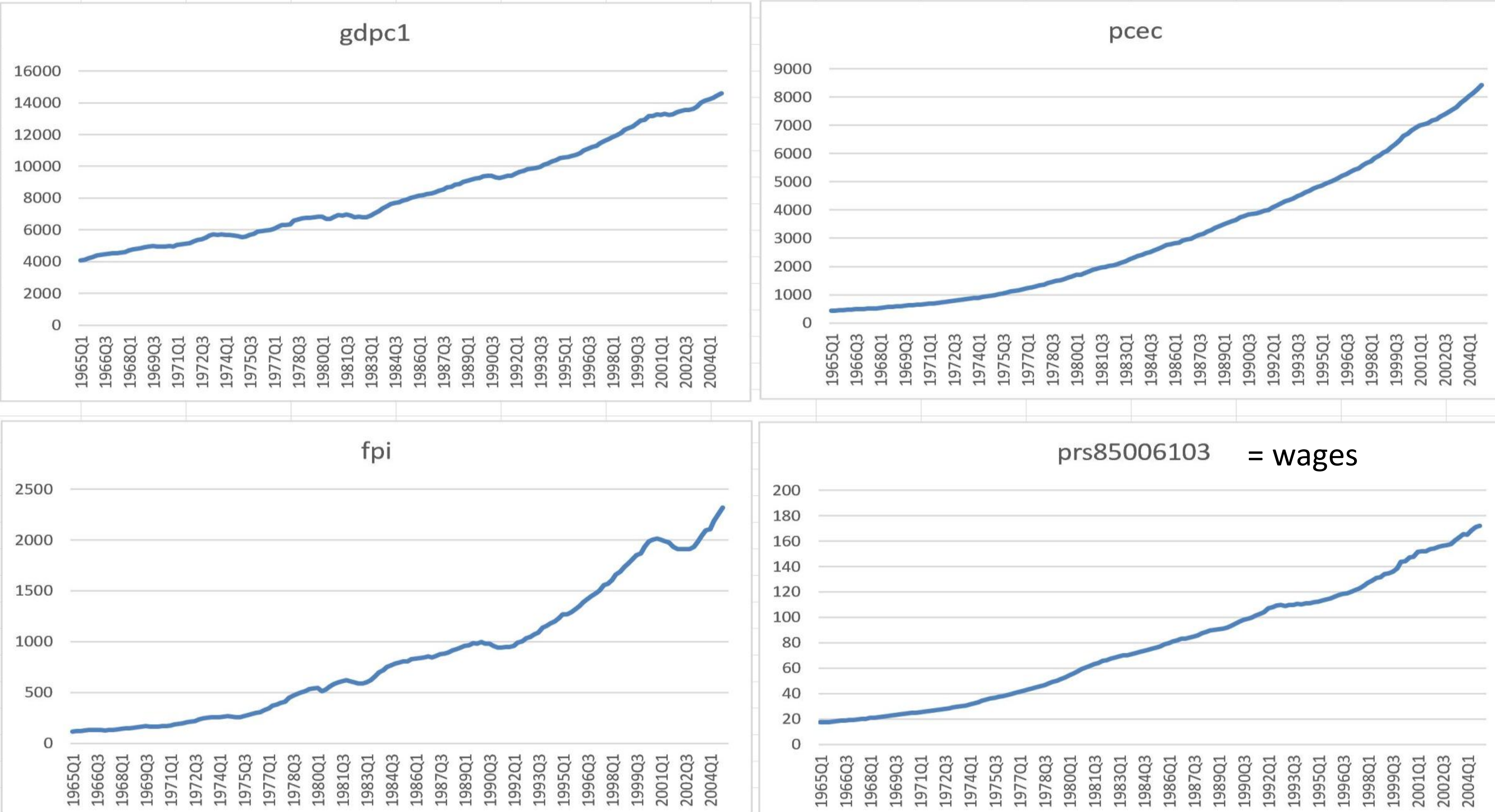
$$\left[\frac{\partial \ln\left(p(O|\theta)\right)}{\partial \theta} + \frac{\partial \ln\left(p(\theta)\right)}{\partial \theta}\right]_{\theta=\hat{\theta}} = 0$$

Solving the first term $\frac{\partial \ln(p(O|\theta))}{\partial \theta} = 0$ for θ gives the mode of the likelihood function (ML estimator), and solving the second term $\frac{\partial \ln(p(\theta))}{\partial \theta} = 0$ for θ gives the mode of the prior

If **prior** is flattish, Bayesian estimator is dominated by ML estimator.
If **likelihood** is flattish, Bayesian estimator is dominated by prior.

RegMLE on SW Model

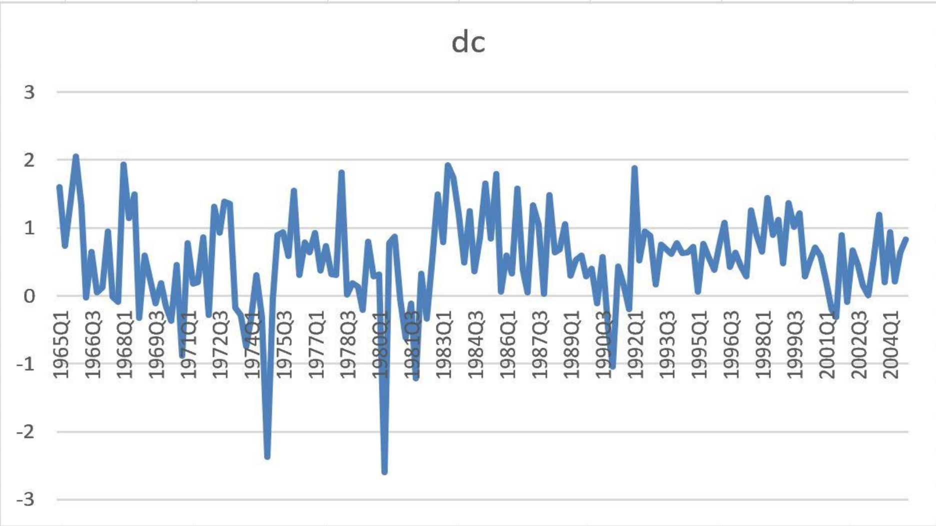
- SW used data on ***seven variables*** (output, consumption, investment, labour hours, inflation, interest rate, wage rate) from "FRED" Database
- All, except for interest rate, are ***seasonally adjusted***
- Data are ***quarterly*** and run from 1947Q1 to present (although SW used only 1965Q1 - 2004Q4)
- Now for some "nitty – gritty"!
- Recall that DSGE model ***uses zero-mean dlog*** variables whose fluctuations around steady-states are "small"
- ➔ ***stationary - I(0)*** ***[ie, zero mean, constant variance]***
- How do we ***construct*** such variables?
- Let's first take a look at ***actual data*** used by SW

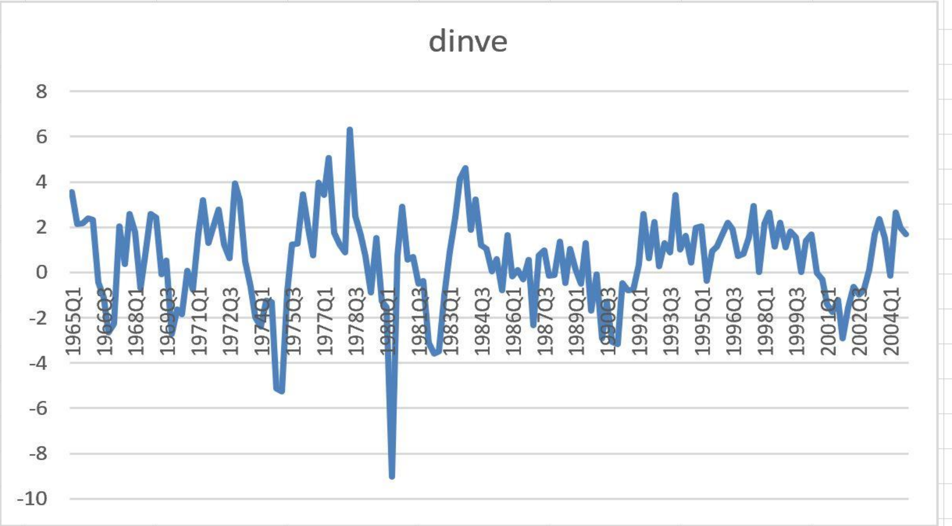


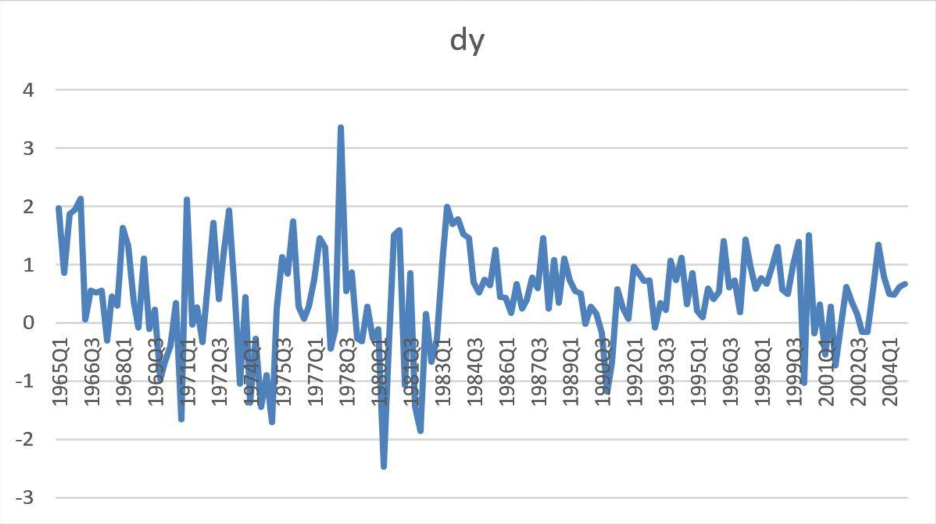
- Obviously, GDP, C, I and W ***do not*** meet our requirement of stationarity [I(0) ➔ zero mean, constant variance]

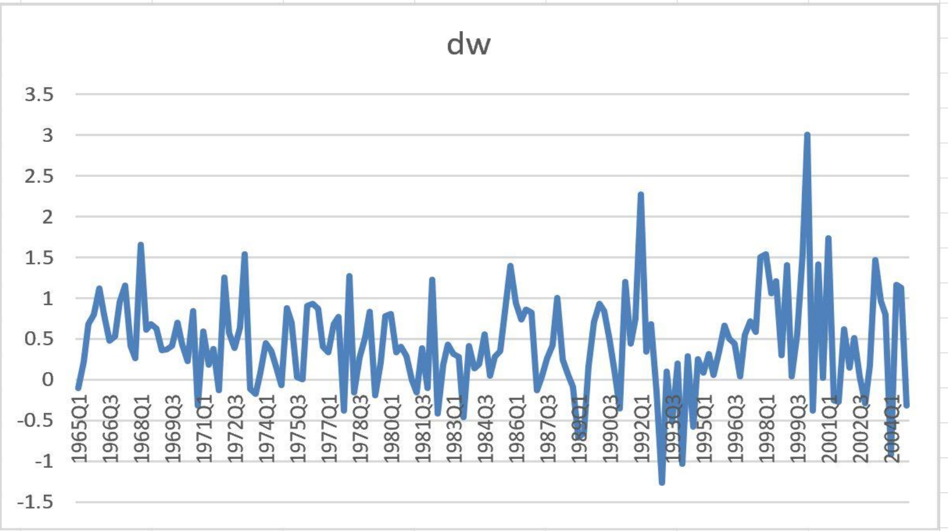
- Typically, time series researchers “stationarise” their data
  - by ***first-differencing logs*** of variables [which are specified in ***real per capita*** terms for usual aggregates]
- SW therefore put data for C, I and Y in ***log real per-capita*** terms by dividing by civilian non-institutional ***population***, age 16 and over, and a price deflator, then taking logs
- But ***distinct upward trends*** continue to appear in log per-capita variables, reflecting secular growth of US economy
- So SW ***first-differenced*** this ***log-per-capita*** data, including for W

- They also noticed that trend was quite ***similar*** for these four variables, so ***assumed*** they were ***cointegrated***
- SW therefore included a ***common trend*** for these four variables in section on “measurement equations” (see below)



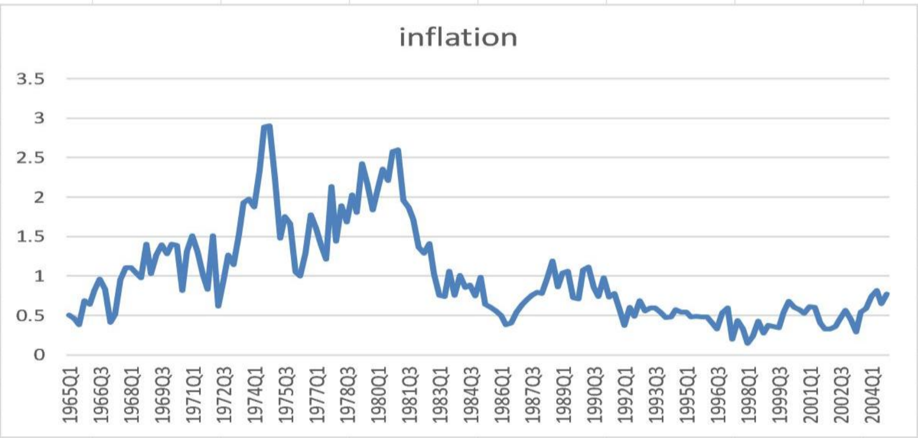


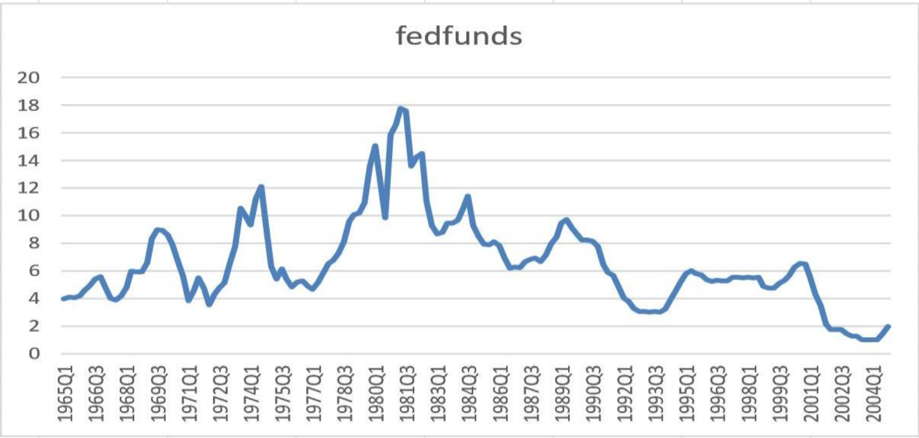


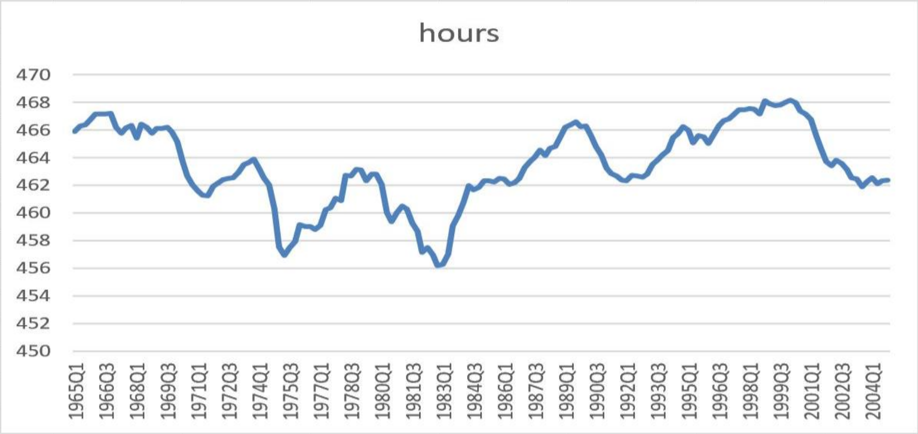


- ***Log first-differences*** of GDP, C, I and W – ie, growth rates – ***now are stationary***

- What about other three variables (interest rate, inflation, labour input)?
- These three already look (roughly) stationary:







- In **Dynare** model, SW use following ***mapping of model variables (on RHS) to data variables (on LHS)*** via their "measurement equations"

cointegration assumption

$$Y_t = \begin{bmatrix} dlGDP_t \\ dlCONS_t \\ dlINV_t \\ dlWAG_t \\ lHOURS_t \\ dlP_t \\ FEDFUNDS_t \end{bmatrix} = \begin{bmatrix} \boxed{\begin{matrix} \bar{\gamma} \\ \bar{\gamma} \\ \bar{\gamma} \\ \bar{\gamma} \end{matrix}} \\ \bar{l} \\ \bar{\pi} \\ \bar{r} \end{bmatrix} + \begin{bmatrix} y_t - y_{t-1} \\ c_t - c_{t-1} \\ i_t - i_{t-1} \\ w_t - w_{t-1} \\ l_t \\ \pi_t \\ r_t \end{bmatrix}$$

- where ***"l"*** stands for (natural) log, and ***"dl"*** for log-difference, of underlying variable

- If ***model*** variables are already in ***dlogs***, why do SW use "$y_t - y_{t-1}$" in their ***measurement*** equations?
- Answer is "measurement error":

$$Y_t = \begin{bmatrix} dlGDP_t \\ dlCONS_t \\ dlINV_t \\ dlWAG_t \\ lHOURS_t \\ dlP_t \\ FEDFUNDS_t \end{bmatrix} = \begin{bmatrix} y_t + \boxed{me_t^y} \\ c_t + me_t^c \\ i_t + me_t^i \\ w_t + me_t^w \\ l_t + \bar{l} \\ \pi_t + \bar{\pi} \\ r_t + \bar{r} \end{bmatrix}$$

me ≡ difference between (assumed) SS growth rate and growth rate observed last period

$$\begin{bmatrix} me_t^y \\ me_t^c \\ me_t^i \\ me_t^w \end{bmatrix} = \begin{bmatrix} \bar{\gamma} - y_{t-1} \\ \bar{\gamma} - c_{t-1} \\ \bar{\gamma} - i_{t-1} \\ \bar{\gamma} - w_{t-1} \end{bmatrix}$$

$$Y_t = \begin{bmatrix} dlGDP_t \\ dlCONS_t \\ dlINV_t \\ dlWAG_t \\ lHOURS_t \\ dlP_t \\ FEDFUNDS_t \end{bmatrix} = \begin{bmatrix} \bar{\gamma} \\ \bar{\gamma} \\ \bar{\gamma} \\ \bar{\gamma} \\ \bar{l} \\ \bar{\pi} \\ \bar{r} \end{bmatrix} + \begin{bmatrix} \boxed{y_t - y_{t-1}} \\ c_t - c_{t-1} \\ i_t - i_{t-1} \\ w_t - w_{t-1} \\ l_t \\ \pi_t \\ r_t \end{bmatrix}$$

- Data in hand → estimate parameters using ***Dynare***
- Doing so merely requires adding a few lines to code used for simulating model
- Start with listing of ***endogenous*** variables:

  var c inve y lab pinf w r a b g qs mc zcap rk k pk kp zcapf rkf kf pkf cf invef yf labf wf rrf kpf spinf sw ms ewma epinfma

  labobs robs pinfobs dy dc dinve dw ;

- ***New element***: second line, which defines ***observed variables*** (as these may differ from those defined in model itself)

- Next, ***exogenous*** variables and ***parameters***

  varexo ea eb eg eqs em epinf ew ;

  SW2007 mod-file notation !!

  parameters constelab constepinf constebeta cgy cmaw cmap calfa czcap cbeta csadjcost csigma chabb ccs cinvs cfc cindw cprobw cindp cprobp csigl crdpi crpi crdy cry crr crhoa crhoas crhob crhog crhols crhoqs crhoms crhopinf crhow ctrend conster cgamma clandap cbetabar cr cpie crk cw cikbar cik clk cky ciy ccy crkky cwhlc cwly

  cg ctou clandaw curvw curvp;

- ***New element***: last line, which defines values of five parameters which will ***not*** be estimated, but instead held at their "***calibrated***" values

- Now time to ***estimate*** parameters of SW model using ***RegMLE-Bayesian technique with RW-MH*** based on methodology of An and Schorfheide (2007)
- These techniques have been applied to many DSGE models, and are now ***standard practice*** in central banks of all developed, and many developing, countries [and academia of course!]
- Distributions ***actually used*** are on next slide, which shows also
  - ***initial value*** given to **Dynare** as a starting point for estimation ("InitVal")
  - ***mean*** of given distribution ("param1")
  - ***shape parameter*** for given distribution ("param2" - ***standard error***)

Param	InitVal	LowerLimit	UpperLimit	Distribution	Param1	Param2
crhoa	0.9676	0.01	0.9999	BETA_PDF	0.500	0.200
crhob	0.2703	0.01	0.9999	BETA_PDF	0.500	0.200
crhog	0.993	0.01	0.9999	BETA_PDF	0.500	0.200
crhoqs	0.5724	0.01	0.9999	BETA_PDF	0.500	0.200
crhoms	0.3	0.01	0.9999	BETA_PDF	0.500	0.200
crhopinf	0.8692	0.01	0.9999	BETA_PDF	0.500	0.200
crhow	0.9546	0.001	0.9999	BETA_PDF	0.500	0.200
cmap	0.7652	0.01	0.9999	BETA_PDF	0.500	0.200
cmaw	0.8936	0.01	0.9999	BETA_PDF	0.500	0.200
csadjcost	6.3325	2	15	NORMAL_PDF	4.000	1.500
csigma	1.2312	0.25	3	NORMAL_PDF	1.500	0.375
chabb	0.7205	0.001	0.99	BETA_PDF	0.700	0.100
cprobw	0.7937	0.3	0.95	BETA_PDF	0.500	0.100
csigl	2.8401	0.25	10	NORMAL_PDF	2.000	0.750
cprobp	0.7813	0.5	0.95	BETA_PDF	0.500	0.100
cindw	0.4425	0.01	0.99	BETA_PDF	0.500	0.150
cindp	0.3291	0.01	0.99	BETA_PDF	0.500	0.150
czcap	0.2648	0.01	1	BETA_PDF	0.500	0.150
cfc	1.4672	1	3	NORMAL_PDF	1.250	0.125
crpi	1.7985	1	3	NORMAL_PDF	1.500	0.250
crr	0.8258	0.5	0.975	BETA_PDF	0.750	0.100
cry	0.0893	0.001	0.5	NORMAL_PDF	0.125	0.050
crdy	0.2239	0.001	0.5	NORMAL_PDF	0.125	0.050
constepinf	0.7	0.1	2	GAMMA_PDF	0.625	0.100
constebeta	0.742	0.01	2	GAMMA_PDF	0.250	0.100
constelab	1.2918	-10	10	NORMAL_PDF	0.000	2.000
ctrend	0.3982	0.1	0.8	NORMAL_PDF	0.400	0.100
cgy	0.05	0.01	2	NORMAL_PDF	0.500	0.250
calfa	0.24	0.01	1	NORMAL_PDF	0.300	0.050

Defines Prior for Bayesian estimation

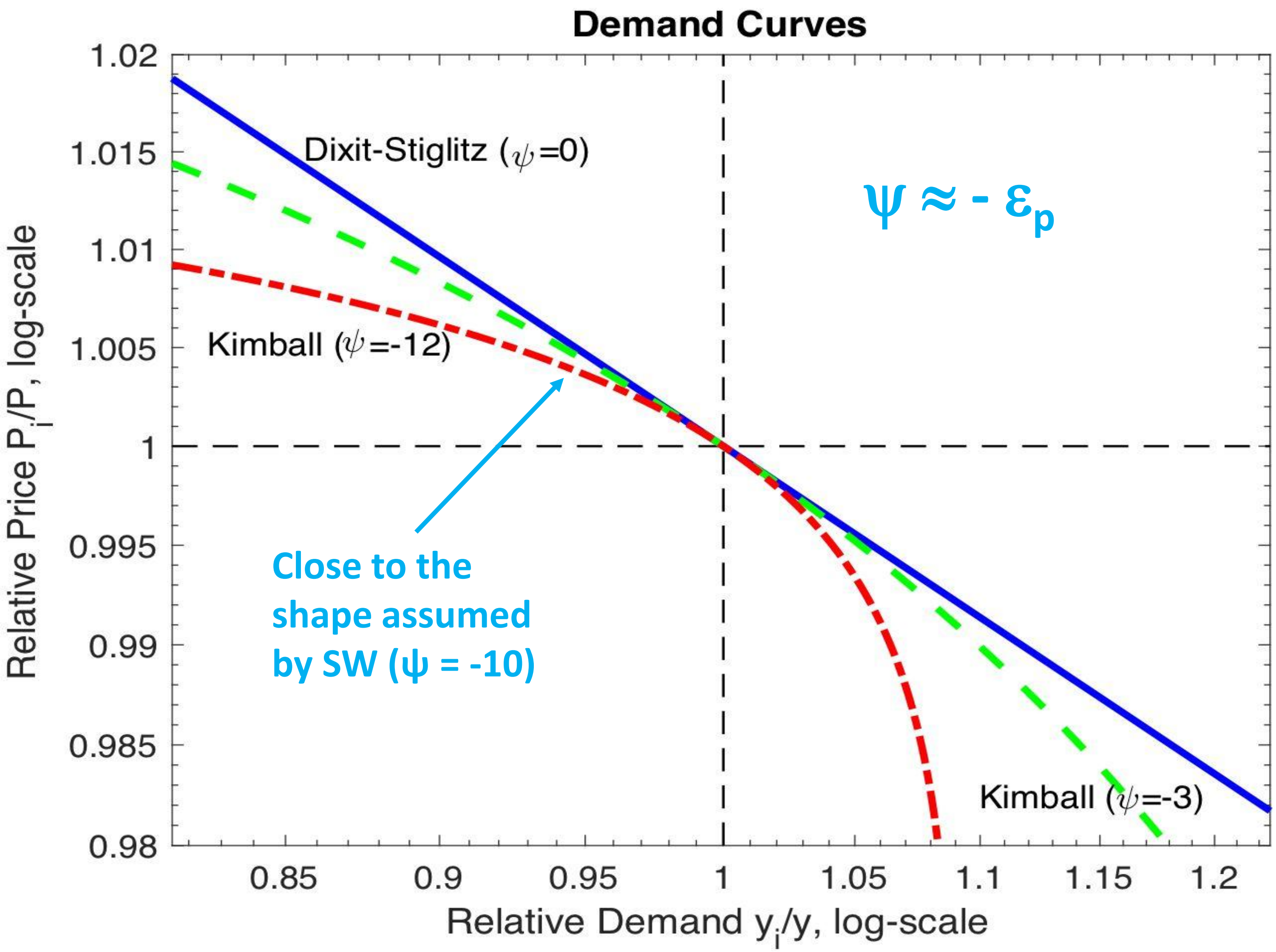
- ***At last*** we are ready to estimate parameters of SW2007 AER model
- Their data (for US) starts in 1947Q1 but they use observations only from 1965Q1 onwards (71st observation in their data file) ***[See data file]***
- ➔ ***estimation*** command contains ***option*** "first_obs=71"
- Since they use only 7 variables indicated earlier, ***varobs*** command will be
- ***varobs*** dy dc dinve labobs pinfobs dw robs

Why only 7 variables? Because a statistical condition known as "stochastic singularity" occurs if there are more variables than shocks in model – and SW model has 7 shocks

observed ("varobs")

- ∃ 34 parameters plus 7 s.d. to estimate, so likelihood surface will be ***very complex***
- Good set of starting values will therefore be ***very important*** in finding a solution
- SW2007 decided to ***fix*** 5 parameters in advance:
  - ***Depreciation rate*** δ is fixed at 0.025 (on a quarterly basis)
  - ***ratio of exogenous spending*** to GDP $\bar{g}$ is set at 18 percent
  - steady-state ***mark-up*** in labour market (ϕ_w) is ***fixed*** at 1.5
  - ***curvature parameters*** of Kimball aggregators in goods and labour markets (ε_p and ε_w) are both ***fixed*** at 10
- [See next slide for impact of this on demand curve, where $\psi \approx - \varepsilon_p$]

Figure 1: Demand Curves -- Implications of Kimball vs. Dixit-Stiglitz Aggregators.



- Once ***priors*** have been set up, proceed to ***estimation*** phase, triggered in **Dynare** code by:

varobs dy dc dinve labobs pinfobs dw robs;

estimation(Tex,datafile=usmodel_data, mode_compute=1, first_obs=71, presample=4, lik_init=2, prefilter=0, mh_replic=0, mh_nblocks=2, mh_jscale=0.20, mh_drop=0.2);

RW-MH stuff

- ***First line*** defines ***observed*** variables (which are found in data file "usmodel_data")
- ***Estimation*** command indicates how many ***MH blocks*** to use (2) and which ***estimator*** ("1")

"presample=4" uses first 4 obs to start estimation process; "prefilter=0" does NOT demean data; "lik_init=2" is a technical issue (Kalman Filter)

- ***First,*** find ***mode of posterior distribution*** by maximising ***log posterior function,*** which combines prior information on parameters with likelihood of data
- If likelihood surface is ***very flat*** and ∃ more than a few parameters, this can be a very ***tricky part*** of process and may require several attempts using ***different*** estimation algorithms and/or different parameters on priors
- If ***don't*** need those fancy graphs, use RegMLE setting option ***mh_replic = 0*** in ***estimation*** command
- ➔ begin ML estimation at ***starting values*** specified in "**estimated_params**" block
- Starting values used here are those given by SW in their original **Dynare** file

- If *do* want those fancy graphs, use ***Random-Walk Metropolis-Hastings*** algorithm to get a picture of ***variability of posterior distribution*** and to evaluate ***marginal likelihood*** of model
- ➔ setting option ***mh_replic*** $\neq$ ***0***
- Dynare's default number of MH replications is 20,000 but most papers use 500,000 or more
- Beware: this can take a loooooong time!
- Estimation using Bayesian methodology and ***mh_replic = 20000*** ➔ results shown on following slides

Bayes: C:\Quant1\SW\SW2007\Smets_Wouters_2007_45_jc3aer.mod

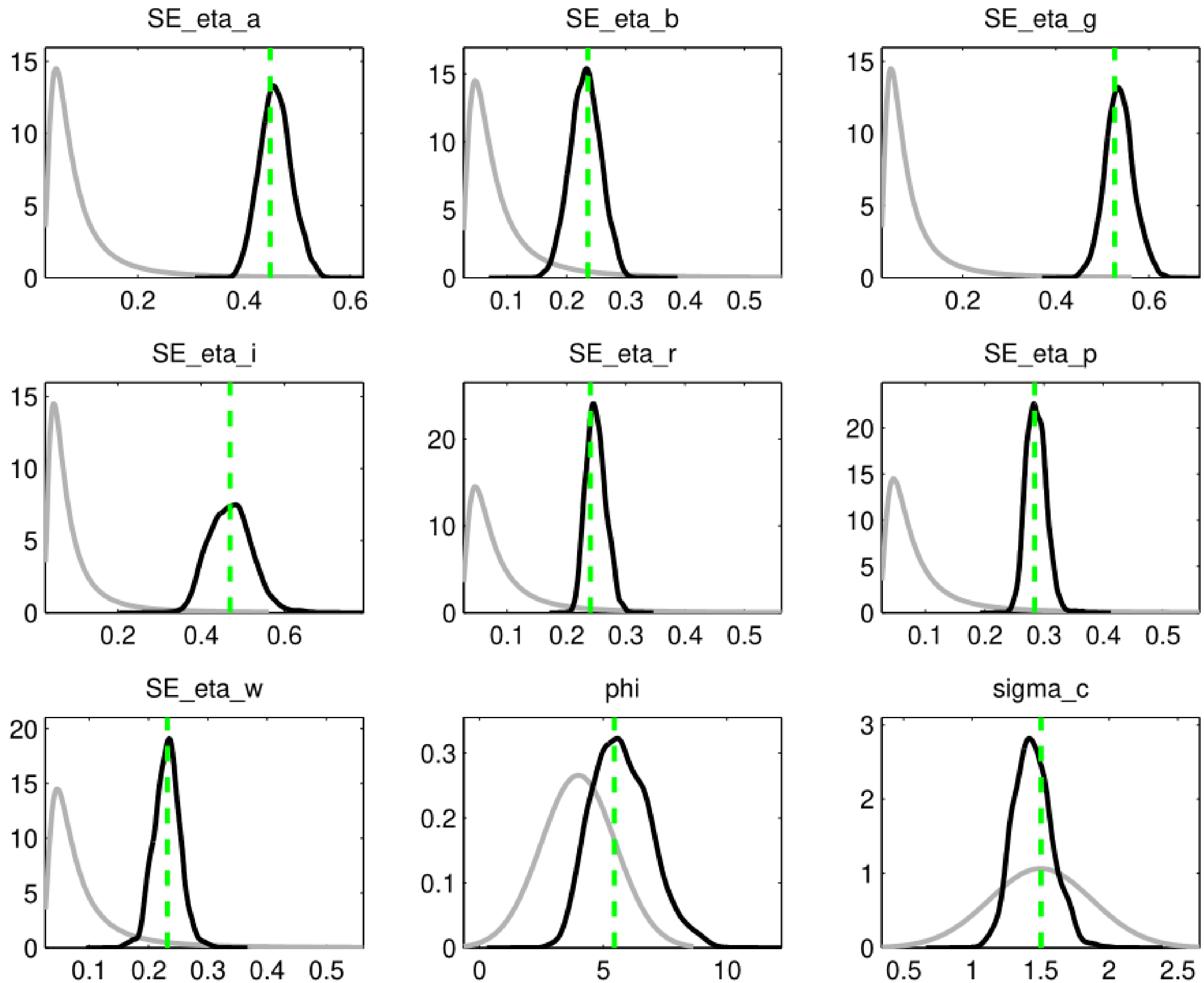


Figure 1: Priors and posteriors.

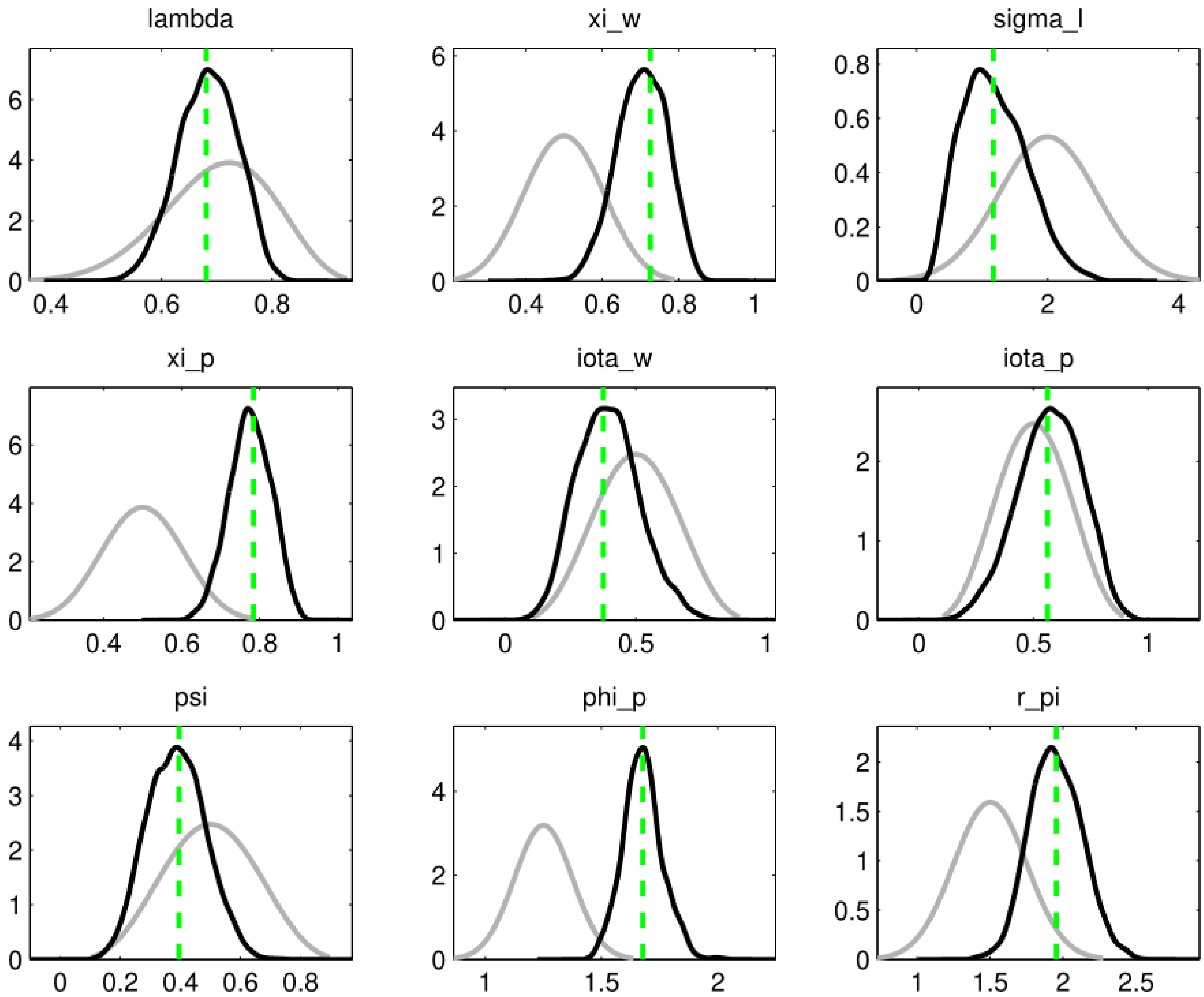


Figure 2: Priors and posteriors.

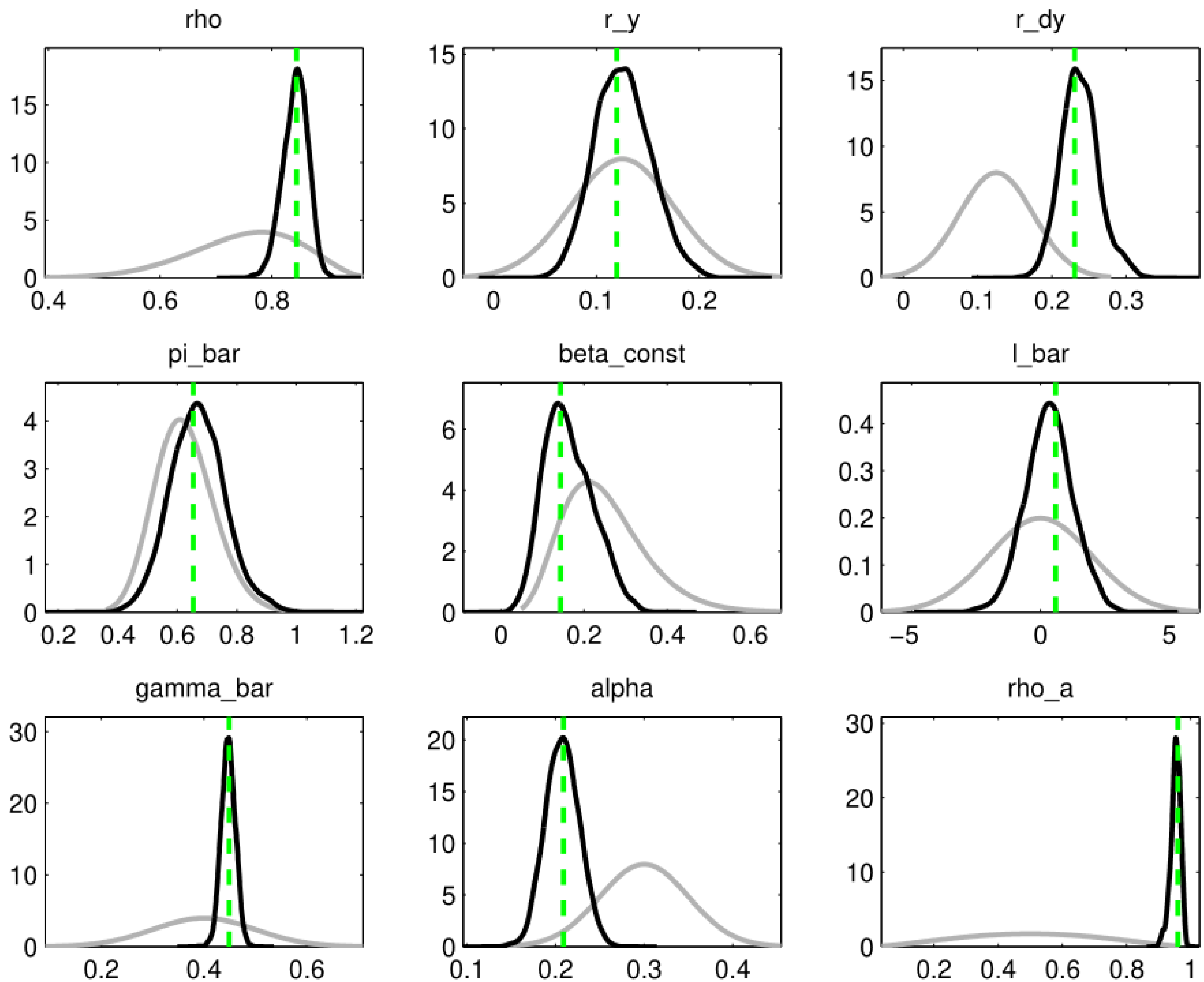


Figure 3: Priors and posteriors.

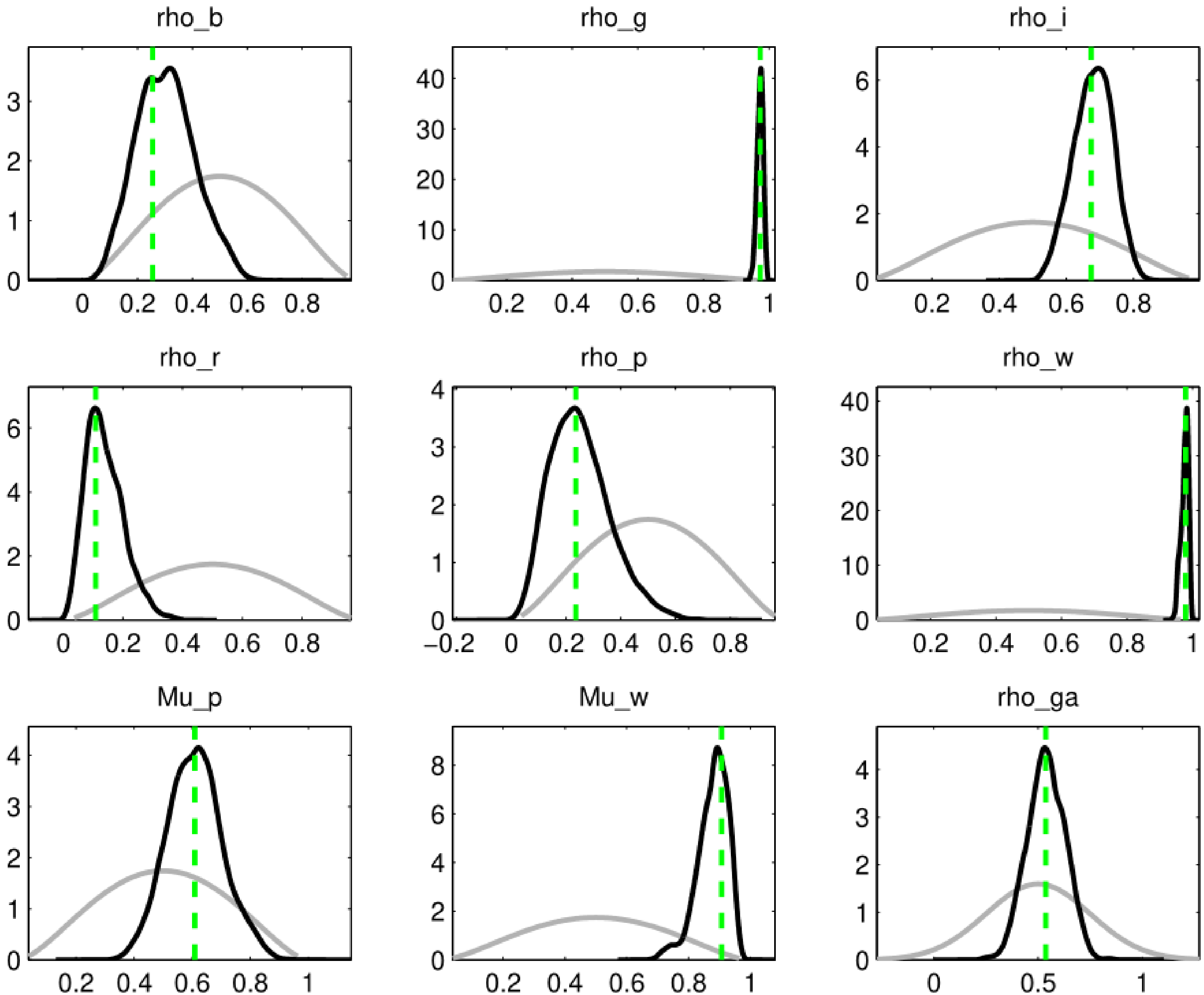


Figure 4: Priors and posteriors.

Table 4: Results from Metropolis-Hastings (parameters)

	Prior			Posterior 90% CI			
	Dist.	Mean	Stdev.	Mean	Stdev.	HPD inf	HPD sup
ρ_a	beta	0.500	0.2000	0.957	0.0112	0.9397	0.9753
ρ_b	beta	0.500	0.2000	0.222	0.0854	0.0904	0.3616
ρ_g	beta	0.500	0.2000	0.977	0.0084	0.9631	0.9902
ρ_i	beta	0.500	0.2000	0.710	0.0577	0.6148	0.8018
ρ_r	beta	0.500	0.2000	0.155	0.0650	0.0482	0.2572
ρ_p	beta	0.500	0.2000	0.891	0.0490	0.8257	0.9719
ρ_w	beta	0.500	0.2000	0.970	0.0129	0.9497	0.9903
μ_p	beta	0.500	0.2000	0.703	0.0958	0.5572	0.8490
μ_w	beta	0.500	0.2000	0.850	0.0654	0.7488	0.9573
"Deep" ↓ φ	norm	4.000	1.5000	5.681	1.0234	3.9683	7.2012
σ_c	norm	1.500	0.3750	1.396	0.1360	1.1655	1.6087
λ	beta	0.700	0.1000	0.707	0.0431	0.6382	0.7776
ξ_w	beta	0.500	0.1000	0.704	0.0635	0.6017	0.8121
σ_l	norm	2.000	0.7500	1.767	0.5309	0.8812	2.6171
ξ_p	beta	0.500	0.1000	0.648	0.0584	0.5486	0.7426
ι_w	beta	0.500	0.1500	0.594	0.1202	0.3928	0.7857
ι_p	beta	0.500	0.1500	0.248	0.0894	0.0999	0.3881
ψ	beta	0.500	0.1500	0.561	0.1115	0.3861	0.7512
Φ	norm	1.250	0.1250	1.615	0.0808	1.4852	1.7483

r_π	norm	1.500	0.2500	2.032	0.1701	1.7536	2.3132
ρ	beta	0.750	0.1000	0.808	0.0244	0.7698	0.8486
r_y	norm	0.125	0.0500	0.089	0.0214	0.0538	0.1245
$r_{\Delta y}$	norm	0.125	0.0500	0.224	0.0289	0.1755	0.2712
$\bar{\pi}$	gamm	0.625	0.1000	0.787	0.1010	0.6222	0.9585
$100(\beta^{-1}-1)$	gamm	0.250	0.1000	0.178	0.0608	0.0809	0.2753
$\bar{l}$	norm	0.000	2.0000	0.560	1.0356	-1.0957	2.2450
$\bar{\gamma}$	norm	0.400	0.1000	0.431	0.0147	0.4057	0.4537
ρ_{ga}	norm	0.500	0.2500	0.523	0.0866	0.3865	0.6741
α	norm	0.300	0.0500	0.187	0.0178	0.1585	0.2166

Table 5: Results from Metropolis-Hastings (standard deviation of structural shocks)

	Prior			Posterior			
	Dist.	Mean	Stdev.	Mean	Stdev.	HPD inf	HPD sup
η^a	invg	0.100	2.0000	0.456	0.0280	0.4114	0.5030
η^b	invg	0.100	2.0000	0.238	0.0235	0.2018	0.2782
η^g	invg	0.100	2.0000	0.533	0.0312	0.4840	0.5839
η^i	invg	0.100	2.0000	0.457	0.0500	0.3735	0.5358
η^m	invg	0.100	2.0000	0.246	0.0151	0.2204	0.2702
η^p	invg	0.100	2.0000	0.141	0.0163	0.1143	0.1672
η^w	invg	0.100	2.0000	0.246	0.0229	0.2069	0.2818

- Now consider estimates of ***main behavioural parameters*** in some detail
- First, since distinctive feature of NK model as contrasted with standard RBC model is (supposed) presence of ***sticky wages and prices***, let us look at what estimates say about this
- Degree of both ***price*** ξ_p and ***wage*** ξ_w ***stickiness*** is estimated to be ***significantly different from zero***, supporting NK hypothesis – and (at 0.65 and 0.7 respectively) higher than 0.5 assumed
- ***average duration*** of ***wage*** contracts $(1- \xi_w)^{-1}$ is somewhat ***less than a year***; whereas that of ***price*** contracts $(1- \xi_p)^{-1}$ is about ***three quarters***

- Interestingly, ***price-indexation*** parameter ι_p is now estimated to be ***significantly different*** from zero, ***overturning MLE result*** [shown in Appendix 5]
- Another feature of SW2007 model is inclusion of (external) ***habit persistence*** in consumption function, via coefficient λ:

$$(C_{t+s}(j) - \lambda C_{t+s-1})$$

- Here, λ is estimated to be ***significantly different from zero***, and at 0.7 quite high

- Recall SW2007 → ***KPR preferences:***

$$E_t \sum_{s=0}^{\infty} \beta^s \left[\frac{1}{1-\sigma_c} (C_{t+s}(j) - \lambda C_{t+s-1})^{1-\sigma_c} \right] \exp\left(\frac{\sigma_c - 1}{1+\sigma_l} L_{t+s}(j)^{1+\sigma_l} \right)$$

coefficient of relative risk aversion *elasticity of work effort*

- What does data say about σ_c , ***coefficient of relative risk aversion***? Estimated value of 1.4 is statistically significantly > 1 , so *v(L)* is decreasing and convex
- σ_l represents inverse of ***elasticity of work effort*** with respect to real wage; its value is estimated as 1.77, so ***elasticity*** is estimated to be significantly less than 1 (actually, a bit less than 0.6) – it takes a big jump in wages to induce more work !

- Finally, estimated value of quarterly ***discount factor*** β may be found by unwinding estimated value of "beta_const" which is defined as

$$100(\beta^{-1} - 1)$$

  and estimated to be 0.1794

- From this, we find a value of β = 0.9926, which implies an ***annual*** discount factor of the ***fourth power*** of this, or 0.97 ➔time rate of preference of 3% per annum, in line with historical norms

- Turning to ***investment-capital nexus***, recall that SW2007 assumed that ***adjusting*** investment (and also capacity ***utilisation***) was costly:

$$K_t(j) = (1-\delta)K_{t-1}(j) + \epsilon_t^i \left[1 - S\left(\frac{I_t(j)}{I_{t-1}(j)}\right)\right] I_t(j)$$

$$S(x_t) = \frac{\varphi}{2}(x_t - \gamma)^2$$

- Since $\partial S(x)/\partial x = \varphi$, elasticity of ***cost of changing investment*** is φ, which is estimated to be quite high (5.7) suggesting a ***slow response*** of investment to changes in value of capital (since S enters ***negatively***)

- ***Intermediate Good Producer i*** is assumed by SW to use ***Cobb-Douglas*** technology plus a fixed cost

$$Y_t(i) = \epsilon_t^a K_t^s(i)^{\alpha}\left[\gamma^t L_t(i)\right]^{1-\alpha} - \gamma^t \Phi$$

- where
  - $K_t^s(i)$ is capital services used in production
  - $L_t(i)$ is a composite labour input
  - **Φ** is a fixed cost
  - **γ^t** represents a labour-augmenting deterministic growth rate in the economy
- Posterior mean of ***fixed cost parameter Φ*** is estimated to be quite high (1.6 ➔ fixed cost share of ***60%***)
- And ***share of capital*** in production ***α*** is estimated to be 0.19, much ***lower*** than the 0.3 usually assumed

- What about ***capacity utilisation***?
- Amount of ***effective capital services*** that households can rent to firms is specified by SW2007 as

$$K_t^s(j) = Z_t(j)K_{t-1}(j)$$

- And cost of changing capital utilisation is given by

$$Z_t = \kappa + R^k(1 - \psi)/\psi$$

- Here, ***capacity utilisation cost*** parameter $\boldsymbol{\psi}$ (inverse of elasticity wrt R^k) is estimated to be 0.56, about midway between extremes of 0 and 1 allowed for it (recall, $\boldsymbol{\psi}$ **=** 1 ➔ utilisation of capital remains ***constant***) ➔ elasticity $\approx$ 1.8

- Finally, turning to ***monetary policy reaction function*** parameters, mean of long-run reaction coefficient to ***inflation*** r_{π} is estimated to be relatively high (2.0), ***greater*** than the 1.5 used in standard Taylor Rule
- There is ***substantial interest rate smoothing***, as mean of coefficient on lagged interest rate ρ is estimated to be 0.81
- Fed policy does not appear to react very strongly to ***output gap*** level (r_y = 0.09), but ***does*** respond strongly to ***changes*** in output gap ($r_{\Delta y}$ = 0.22) in short run

- Overall, it appears that data are ***quite informative*** on behavioural parameters, as indicated by ***lower variance of posterior*** distribution relative to prior distribution
- Same appears to be true for ***stochastic processes*** for exogenous disturbances
- ***Productivity***, ***government spending***, and ***wage mark-up*** processes are estimated to be most ***persistent***, with AR(1) coefficients, respectively, of $\rho_a = 0.96$, $\rho_g = 0.98$, and $\rho_w = 0.97$

- ***High persistence*** of productivity, fiscal and wage mark-up processes implies that, ***at long horizons***, much of ***forecast error variance*** of real variables will be explained by those three shocks
- In contrast, persistence of ***risk premium*** ρ_b and ***monetary policy shock*** ρ_r are quite ***low*** (0.22 and 0.16, respectively) ➔ smaller LR impact on FEV
- Two ***MA components*** (which capture ***high-frequency*** effects of a shock) are quite ***high*** (μ_w = .85 for wages and μ_p = .7 for prices) indicating a ***sharp initial impact***

- ***Trend quarterly growth rate*** $\bar{\gamma}$ is estimated to be around 0.43, which is somewhat ***smaller*** than average growth rate of output per capita over sample (4 x 0.43 = 1.72)
- Posterior mean of ***steady-state inflation rate*** $\bar{\pi}$ over full sample is just above 3 percent on an annual basis (4 x 0.787 = 3.15)
- Mean of ***discount rate*** **(1 – 1/β)** is estimated (as indicated earlier) to be about 3 percent on an annual basis
- Finally, implied mean ***steady-state*** nominal and real ***interest rates*** are, respectively, about 6 percent and 3 percent on an annual basis

- Finish with a ***subtlety***
- RegMLE-Bayesian estimation requires first finding ***posterior mode*** (ML optimising parameters)
- But often this is difficult or impossible
- What to do?
- ***Dynare*** offers a solution: ***mode_compute = 6***
- Note that this has ***nothing*** specifically to do with RegMLE-Bayesian estimation
- mode_compute = 6 may be used in ***any*** situation where posterior mode is hard to find

- Option ***mode_compute = 6*** triggers a ***Monte-Carlo*** based optimisation routine
- Actually, it is ***not a true*** optimisation routine: goal is only to ***identify an interesting region*** to start RW-MH algorithm and an ***initial estimate*** of posterior covariance matrix for estimated parameters
- RW-MH algorithm ***does not need*** to start from posterior mode to ***converge*** to posterior distribution
- It is only required to start from a point (in parameter space) with a ***high posterior density value*** and to use an estimate of covariance matrix for jumping distribution

- ***If*** Hessian (a square matrix of ***second-order partial derivatives*** of a function f) is ***positive definite*** at x, ***then*** f attains a ***local minimum*** at x

Why minimum? Because Dynare uses –LogL so min ➔ max of LogL

- Very often **Dynare**'s default optimization algorithm ***fails*** to find a minimum of –LogL with a ***positive definite Hessian matrix***
- Consequently **Dynare** cannot run RW-MH, since a positive definite Hessian matrix of –LogL is ***required*** to approximate posterior var-cov matrix
  - ***inverse*** of Hessian matrix provides an accurate ***approximation*** of ***var-cov*** matrix ***if*** posterior distribution is not too far from a Gaussian distribution

- Mode_compute=6 algorithm uses a RW-MH algorithm with a ***diagonal var-cov matrix*** (prior variances or a covariance matrix proportional to unity) and ***continuously updates*** posterior var-cov matrix and posterior mode estimates through RW-MH draws
- After each RW-MH draw, θ_t in posterior distribution, posterior mean, posterior covariance and posterior mode are updated
- Process continues until convergence

- Where to find Dynare examples?
- There are two really exceptional places:
- https://github.com/JohannesPfeifer/DSGE_mod
- Pfeifer's site contains dozens of worked Dynare examples from the literature
- http://www.macromodelbase.com/
- Wieland's Macroeconomic Model Data Base (MMB) is an archive of > 75 macroeconomic models written in Dynare (simulation versions)
- It also provides various tools for systematic model comparison

Appendices

Appendix 1: Cho-Moreno Model

- ***Cho and Moreno*** "A Small-Sample Study of the New-Keynesian Macro Model", 2006
- As in models we have studied previously, economy consists of
  - representative household
  - representative finished goods-producing firm
  - continuum of intermediate goods-producing firms indexed by $i \in [0, 1]$
- Also includes ***monetary*** (but ***not*** fiscal) authority
- Model is thus a linearized Rational Expectations model consisting of AS, IS and monetary policy rule equations with endogenous persistence

- ***AS (Supply) equation***: generalization of Calvo (1983) pricing model already seen many times
- ***IS (Demand) equation***: derived through representative agent optimization with external habit persistence
- ***Monetary policy rule***: forward looking Taylor rule proposed by Clarida, Gali, and Gertler (2000)

- ***IS (Demand) equation***: representative agent seeks to maximize its lifetime expected ***utility*** given by:

$$E_0 \left\{ \sum_{t=0}^{\infty} \psi^t \left[u \left(C_t, C_{t-1}; \xi_t \right) \right] \right\}$$

  – ψ = time discount factor
  – C_t is a consumption index of differentiated (index i) goods defined as usual by

$$C_t = \left[\int_0^1 C_{t,i}^{\frac{\theta_t - 1}{\theta_t}} di \right]$$

- Utility function exhibits ***(external) habit persistence*** but ignores leisure/labour

- Optimal intertemporal consumption choice is then given by standard ***Euler Equation***:

$$\frac{1}{1+\tilde{r}_t} = E_t \left\{ \frac{\psi u_c\left(C_{t+1}, C_t; \xi_{t+1}\right)}{u_c\left(C_t, C_{t-1}; \xi_t\right)} \frac{P_t}{P_{t+1}} \right\}$$

- ***Parametrise*** utility function as:

$$u\left(C_t, C_{t-1}; \xi_t\right) = \xi_t \frac{1}{1-\sigma} \left(\frac{C_t}{C_{t-1}^h} \right)^{1-\sigma}$$

- Log-linearise ➔ ***IS equation*** as

$$y_t = \mu_1 E_t y_{t+1} + \mu_2 y_{t-1} - \phi\left(r_t - E_t \pi_{t+1}\right) + \varepsilon_t^{IS}$$

- Parameters are defined as

$$y_t = \mu_1 E_t y_{t+1} + \mu_2 y_{t-1} - \phi\left(r_t - E_t \pi_{t+1}\right) + \varepsilon_t^{IS}$$

$$\begin{aligned}
\mu_1 &= \frac{\sigma}{\sigma(1+h)-h} \\
\mu_2 &= \frac{h(\sigma-1)}{\sigma(1+h)-h} \\
\phi &= \frac{1}{\sigma(1+h)-h}
\end{aligned}$$

- where
  - h is ***habit persistence*** parameter
  - $\mu_1 + \mu_2 = 1$ (➔ weighted average of past and expected)
  - σ is (inverse of) ***intertemporal elasticity of substitution***
  - all variables are expressed as percentage deviations from their ***steady-state*** values

- ***AS (Supply):*** Cho and Moreno use ***Calvo fairy*** to obtain pricing rule exactly like that previously studied:

$$P_t^{1-\psi} = \int_0^{\theta} P_{\boxed{t-1}}^{1-\psi}\, dj + \int_{\theta}^{1} P_t^{\boxed{*}\,1-\psi}\, dj$$

$$P_t = \left[\theta P_{t-1}^{1-\psi} + (1-\theta) P_t^{*\,1-\psi}\right]^{\frac{1}{1-\psi}}$$

- Slight difference introduced by Cho-Moreno:
- Previously, ***Calvo Rule*** was simply $P_t = P_{t-1}$ so that price is not adjusted

- Cho and Moreno instead ***index*** to inflation:

$$\log P_t = \log P_{t-1} + \vartheta \pi_{t-1}$$

- which changes ***aggregate pricing rule*** slightly to

$$P_t = \left[(1-\varphi) P_{i,t}^{*1-\theta} + \varphi \left(\frac{P_{t-1}^{1+\vartheta}}{P_{t-2}} \right)^{1-\theta_t} \right]^{\frac{1}{1-\theta}}$$

- FOCs for firm's profit-maximising function + algebra ➔ ***AS equation*** (***Phillips Curve***)

$$\pi_t = \delta E_t \pi_{t+1} + (1-\delta)\pi_{t-1} + \lambda y_t + \varepsilon_t^{AS}$$

- Third equation: ***monetary policy rule***, which Cho and Moreno define by

$$r_t = \rho \boxed{r_{t-1}} + (1-\rho)\boxed{r_t^*} + \varepsilon_{MP_t}$$

- where
  - r^* = desired nominal interest rate
  - ε_{MPt} = monetary policy shock
- ***Two parts*** to MonPol equation:

1. Lagged interest rate (r_{t-1}) captures well-known Federal Reserve interest-rate ***smoothing***
2. r^* - see next slide

2. r* ➔ "Taylor rule“: monetary authority reacts to
   - deviations of expected inflation from its ***target*** ($\bar{\pi}$ = long run equilibrium level of inflation)
   - deviations of output from its ***potential*** level

$$r_t^* = \bar{r}^* + \beta(E_t\pi_{t+1} - \bar{\pi}) + \gamma y_t$$

- Putting these two parts together yields

$$r_t = \alpha_{MP} + \rho r_{t-1} + (1-\rho)\left[\beta E_t\pi_{t+1} + \gamma y_t\right] + \varepsilon_{MP_t}$$

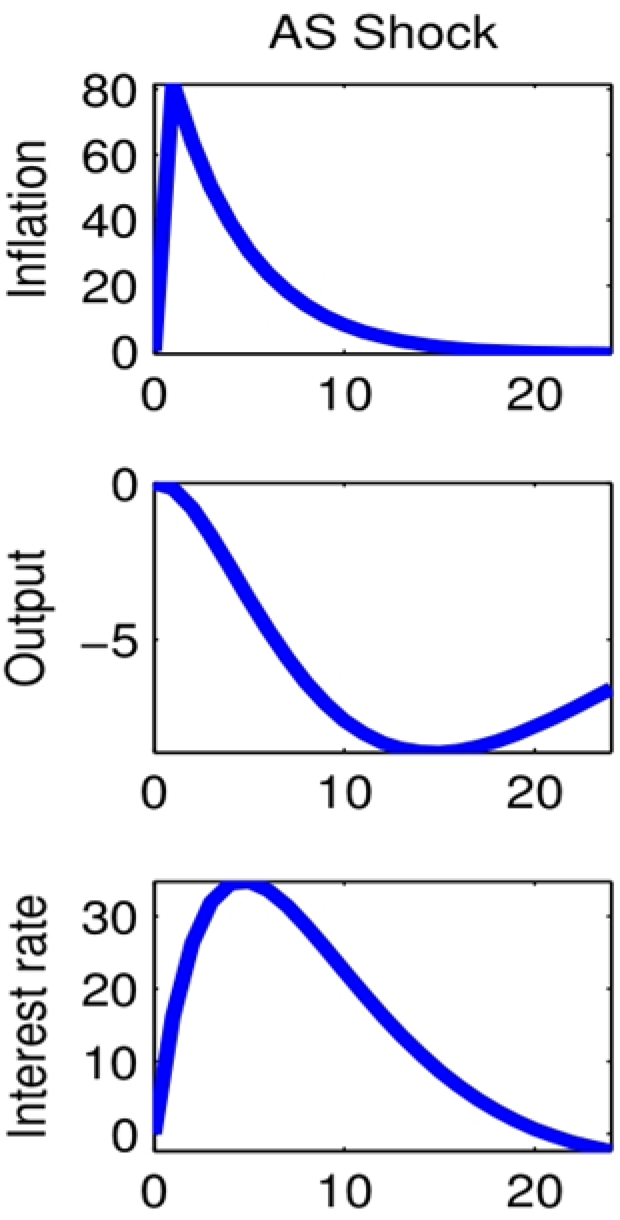
- where $\alpha_{MP} = (1-\rho)(\bar{r}^* - \beta\bar{\pi})$

- **Dynare** code:

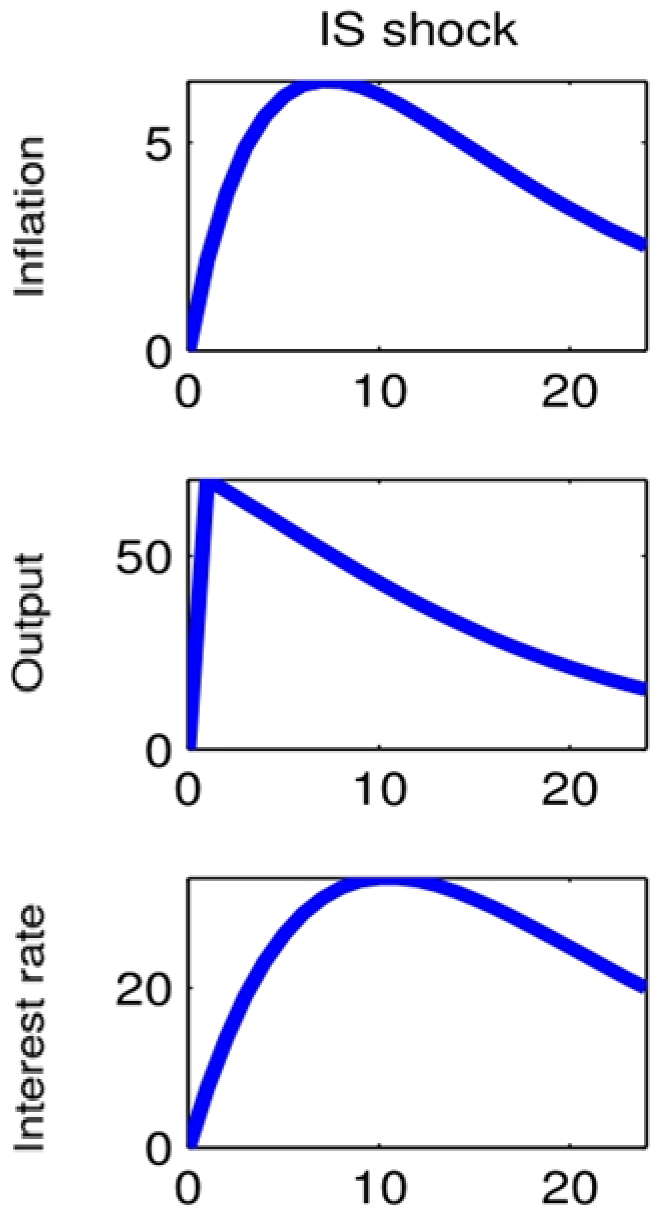
```
// IS equation
y = mu*y(+1) + (1-mu)*y(-1) - phi*(r-infl(+1)) + eis;
// AS equation (Phillips Curve)
infl = del*infl(+1) + (1-del)*infl(-1) + lam*y +eas;
// Taylor Rule (MonPol equation)
r = roe*r(-1) + (1-roe)*(bet*infl(+1)+gam*y) + emp;
```

- where
  - $\mu_1 + \mu_2 = 1$ ➔ use "mu" = μ_1
  - incorporate constant (α_{MP}) of MonPol equation into shock term (emp)
- Now simulate model to see its properties

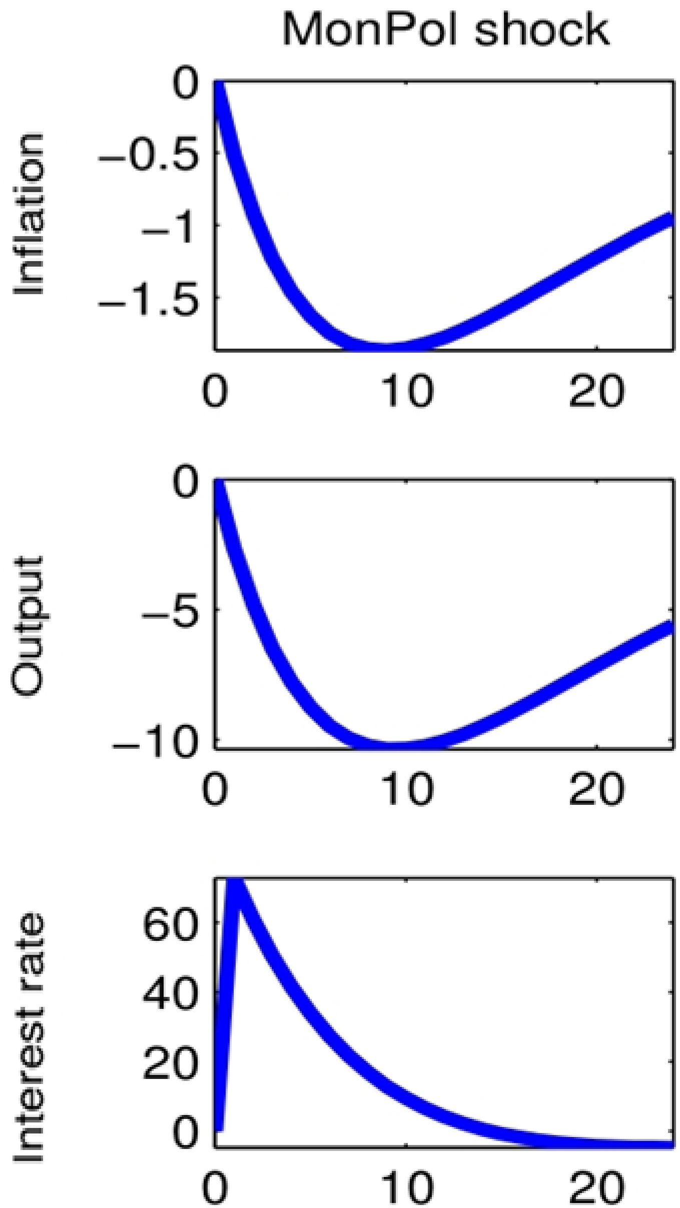
- An upward shock to inflation (AS equation) should cause central bank to raise interest rates in reaction to increased inflation, tamping down increase in inflation and ultimately driving output down – check!



- Similarly, an upward shock to output (IS equation) should cause central bank also to raise interest rates in reaction, tamping down both increase in inflation and output increase – check!



- And finally, an upward shock to interest rates (MonPol equation) should impact negatively on both inflation and output – check!



App. 2 MLE on Cho-Moreno

- IRFs evidently dependent upon values of ***parameters*** used for simulation
- Here based upon a simple ***calibration*** = first column of Table 2 of Cho-Moreno paper
- Let us now see what actual data – via ***ML estimation*** – says about how good this calibration is
- In this model, there are 10 parameters to be estimated → ***Likelihood function*** will be ***10-dimensional*** and quite possibly rather ***flat or rocky***

- ➔ ***may be hard*** to find a solution to optimisation problem
- ➔ **Dynare** provides several ***different algorithms*** for maximising Likelihood function
- **Dynare** default is so-called ***"csminwel"*** of Chris Sims
- This does very simple "line search" and if it hits a cliff or flat surface, randomly moves in another direction to find better parameter estimates
- Other algorithms can be chosen by including ***"mode_compute=??"*** among options in estimation command (?? for csminwel is 4)

- Example: use ***"mode_compute=8"*** which corresponds to **Dynare** implementation of the ***Nelder-Mead simplex-based optimization*** routine:
- ***estimation (datafile=cmdata1_jc1, prefilter=1, first_obs=10, mode_compute=8) infl y r;***
- Some options included in this command are ***new***:
  - ***"datafile=cmdata1_jc1"*** instructs **Dynare** to use a ***Matlab*** file "cmdata1_jc1" which ***constructs*** data to be used in estimation process

- ***“prefilter=1”*** causes **Dynare** to ***demean*** each data series by its empirical mean
- ***“first_obs=10”*** tells **Dynare** to drop first 9 observations (eg, because they may be perturbed by various data transformations) and start from 10th

• ***Matlab*** file “cmdata1_jc1” used here reads data from an Excel spreadsheet in folder “data”
  - [dataQ,datadate,raw] = xlsread('data/CM04.xls');

• and creates (quarterly) ***output gap*** variable by ***HP filtering*** original raw Real GDP data (in col. 1)
  - ygap = 100*log(dataQ(:,1)/4)-
    hpfilter(100*log(dataQ(:,1)/4), lambda);

This is the CM04.xls data sheet

A	B	C	D	E	F	G	H	I
Quarter	RGDP	Inf_IGDP	FFR_EndQ	FFR	FFRAVG	3MTB	3MTBAvg	OGAP
Mar-1960	2517.40000	1.7466	4	3.84	3.93333	3.44	3.9466667	-0.245983
Jun-1960	2504.80000	1.3951	3.5	3.32	3.69667	2.64	3.0933333	-1.658951
Sep-1960	2508.70000	1.5235	3	2.6	2.93667	2.49	2.3933333	-2.429684
Dec-1960	2476.20000	1.1762	3	1.98	2.29667	2.27	2.3633333	-4.670724
Mar-1961	2491.20000	0.8701	2	2.02	2.00333	2.42	2.3766667	-5.010441
Jun-1961	2538.00000	0.8494	0.5	1.73	1.73333	2.36	2.3266667	-4.103001
Sep-1961	2579.10000	1.2431	1.5	1.88	1.68333	2.3	2.3233333	-3.448815
Dec-1961	2631.80000	1.3519	2.5	2.33	2.40000	2.62	2.4766667	-2.380439
Mar-1962	2679.10000	2.3579	2.75	2.85	2.45667	2.72	2.7433333	-1.559204
Jun-1962	2708.40000	0.5953	2.75	2.68	2.60667	2.72	2.72	-1.433335
Sep-1962	2733.30000	0.966	2.75	2.9	2.84667	2.79	2.8566667	-1.485275
Dec-1962	2740.00000	1.2601	3	2.93	2.92333	2.86	2.8033333	-2.212567
Mar-1963	2775.90000	0.9052	3	2.98	2.96667	2.9	2.91	-1.891292
Jun-1963	2810.60000	0.7926	3	2.99	2.96333	3	2.9433333	-1.640914
Sep-1963	2863.50000	0.791	3.5	3.48	3.33000	3.38	3.28	-0.779216
Dec-1963	2885.80000	2.9925	3.25	3.38	3.45333	3.52	3.4966667	-1.017057
Mar-1964	2950.50000	1.1845	3.5	3.43	3.46333	3.55	3.5366667	0.1730016
Jun-1964	2984.80000	1.0356	3.5	3.5	3.49000	3.48	3.48	0.2952626
Sep-1964	3025.50000	1.5766	2.75	3.45	3.45667	3.53	3.5066667	0.6000449
Dec-1964	3033.60000	1.9856	4	3.85	3.57667	3.86	3.6866667	-0.191009

- ***Before estimation*** need to give algorithm some help by defining ***starting values*** (otherwise it assumes a starting value of ***zero*** for each parameter …)
- Block of ***Dynare*** code on next slide lists
  - all parameters to be estimated
  - initial values
  - upper and lower bounds (as necessary)
  - one line per parameter

```
// for estimation MLE setup
estimated_params;
del, 0.5; //0, 1;
lam, 0.1; // -10, 10;
mu,  0.5; // 0, 1;
phi, 0.1; // -10, 10;
roe, 0.5; // 0, 1;
bet, 1.5; // 1, 10;
gam, 0.5; // 0, 10;

stderr eas, 0.5,0,100;
stderr eis, 0.5,0,100;
stderr emp, 0.5,0,100;
end;
```

- Numbers in each line after comment indicator (“//”) are parameter ***lower and upper bounds*** (here not used since commented out – except for standard errors of shock terms)

- Output of ML estimation is:

```
Simplex iteration number: 2729-4-3
Simplex move:             reflect-0
Objective function value: 704.1908
Mode improvement:         0
Norm of dx:               0
Norm of dSimplex:         0.00020535
Crit. f:                  3.6954e-08
Crit. x:                  9.1275e-05

CONVERGENCE ACHIEVED AFTER 2729 ITERATIONS!

Final value of minus the log posterior (or likelihood):704.190819

RESULTS FROM MAXIMUM LIKELIHOOD ESTIMATION
parameters
    Estimate    s.d. t-stat

del    0.5251  0.0119 44.0944
lam    0.0098  0.0070  1.3916
mu     0.5157  0.0204 25.2646
phi    0.0031  0.0029  1.0583
roe    0.8869  0.0240 36.9025
bet    1.2421  0.2734  4.5424
gam    2.5596  0.6858  3.7325

standard deviation of shocks
    Estimate    s.d. t-stat

eas    0.6248  0.0350 17.8321
eis    0.4135  0.0241 17.1403
emp    0.8579  0.0458 18.7422
```

Nelder-Mead algorithm stuff

Most parameters are highly statistically significant

- Comparing ***calibrated*** with ***estimated*** values of parameters:

	Prior	Estimate
del	0.5586	0.5251
lam	0.0011	0.0098
mu	0.4859	0.5157
phi	0.0045	0.0031
roe	0.8458	0.887
bet	1.6409	1.2421
gam	0.6038	2.5596
	standard	deviation
	Prior	Estimate
eas	0.4585	0.6248
eis	0.3734	0.4135
emp	0.7327	0.8579

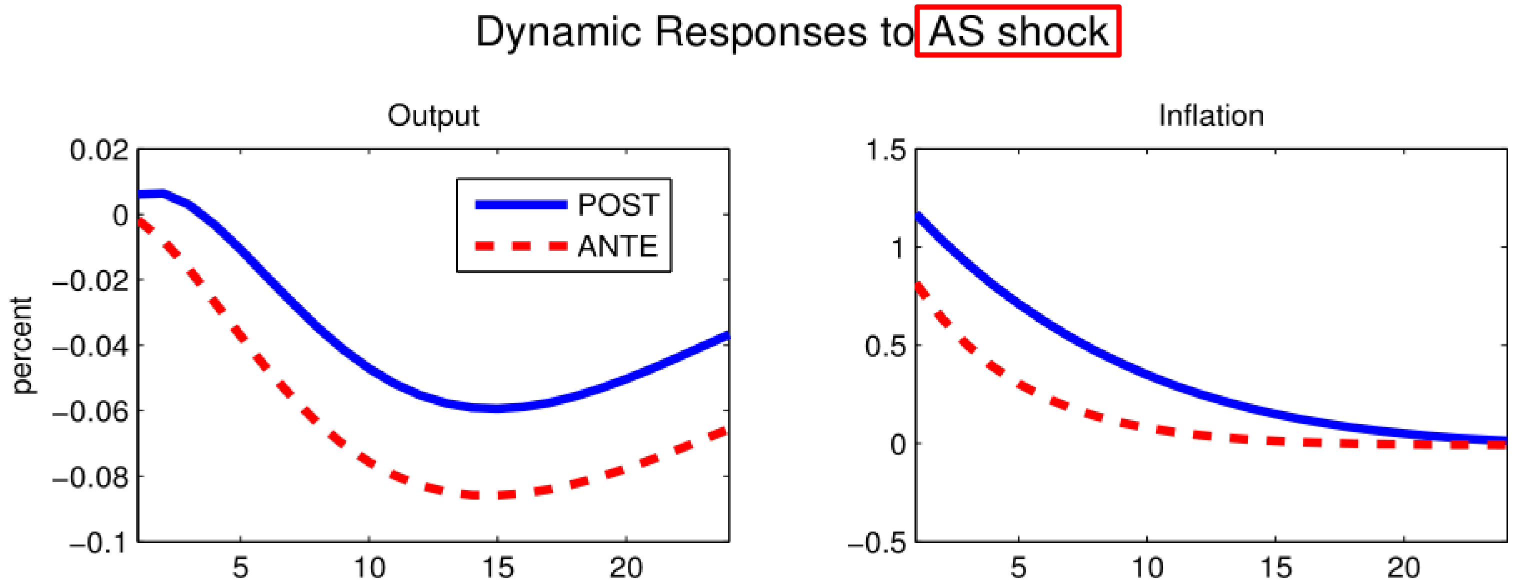
With exception of “gam”, “lam” and “phi”, there is ***not much difference***. But these parameters are ***important*** in responses of model to shocks:

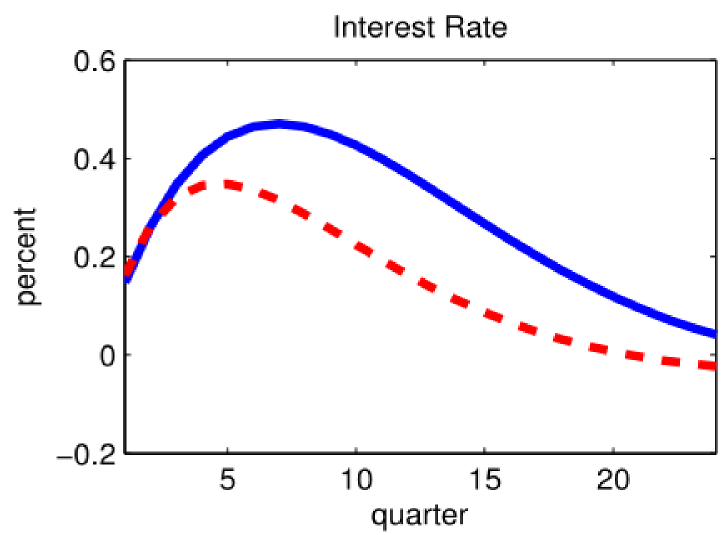
y = mu*y(+1) + (1-mu)*y(-1) - phi*(r-infl(+1)) + eis;

infl = del*infl(+1) + (1-del)*infl(-1) + lam*y +eas;

r = roe*r(-1) + (1-roe)*(bet*infl(+1)+gam*y) + emp;

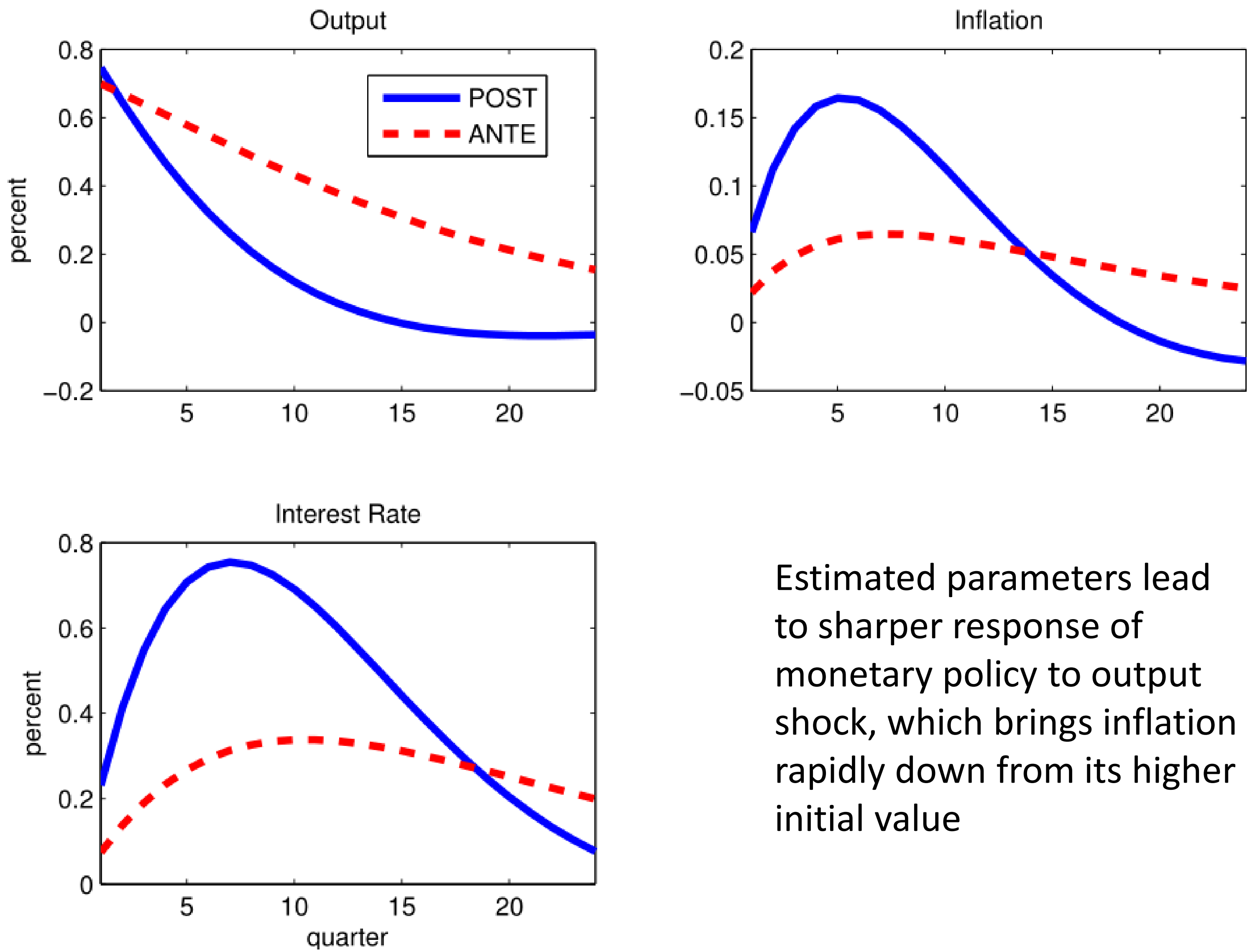
- Comparing IRFs ***before and after*** ML estimation:





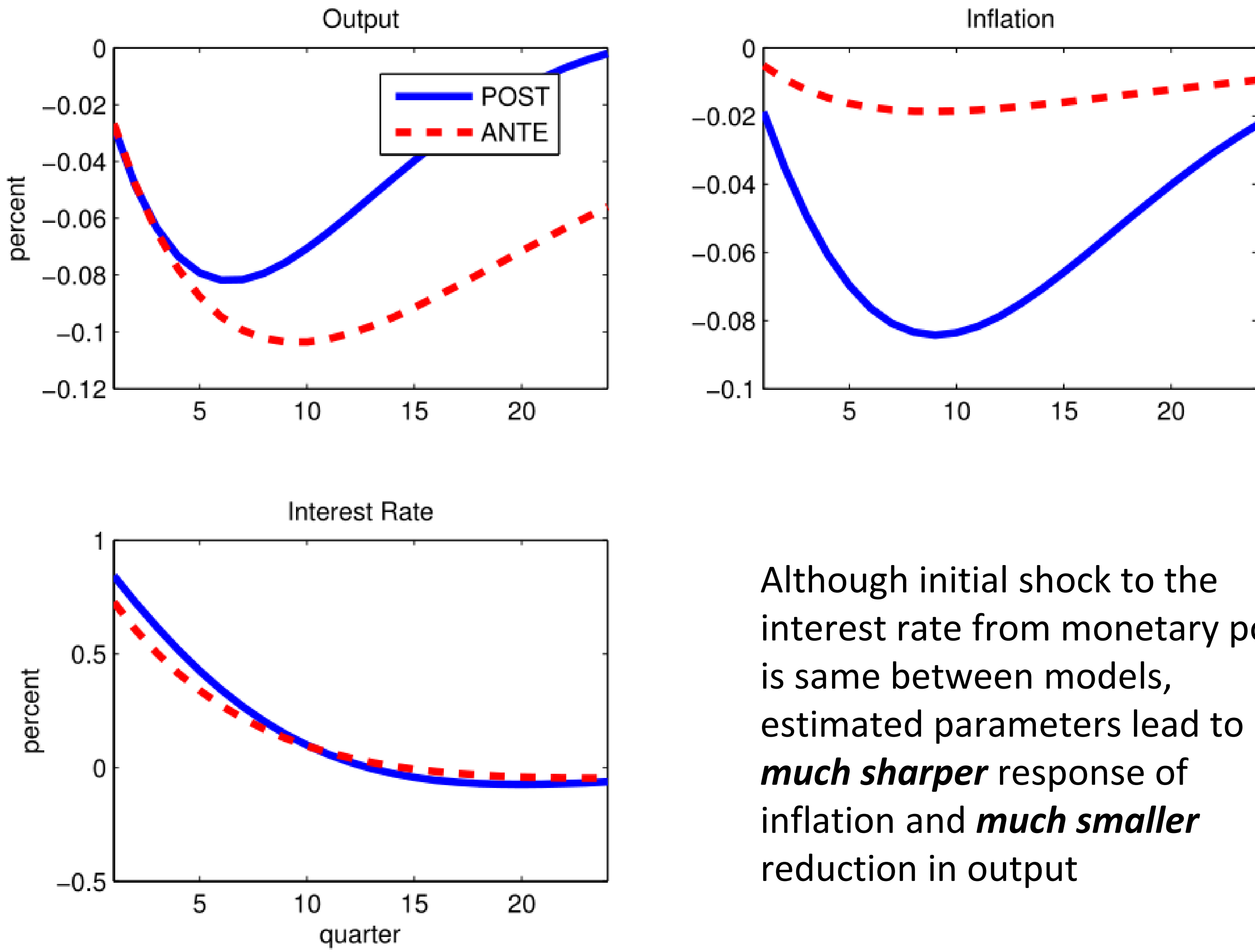
Estimated parameters lead to ***higher initial shock*** to inflation and ***sharper response*** of monetary policy to inflation shock, which brings inflation rapidly down from its higher initial value

Dynamic Responses to IS shock



Estimated parameters lead to sharper response of monetary policy to output shock, which brings inflation rapidly down from its higher initial value

Dynamic Responses to MonPol shock



Although initial shock to the interest rate from monetary policy is same between models, estimated parameters lead to ***much sharper*** response of inflation and ***much smaller*** reduction in output

- ***Dynare output log*** contains other interesting information:

MOMENTS OF SIMULATED VARIABLES

VARIABLE	MEAN	STD. DEV.	VARIANCE	SKEWNESS	KURTOSIS
infl	-0.369219	2.315141	5.359876	0.121984	-0.185298
y	-0.112923	1.336085	1.785124	0.043860	-0.097384
r	-0.457292	3.094213	9.574155	-0.077168	-0.143283

- Note that ***means*** of ***simulated*** variables are NOT zero, although steady-states ARE:

STEADY-STATE RESULTS:

infl	0
y	0
r	0

- Notice how small is ***variable correlation*** – a good sign!

CORRELATION OF SIMULATED VARIABLES

VARIABLE	infl	y	r
infl	1.0000	0.0984	0.4129
y	0.0984	1.0000	0.2922
r	0.4129	0.2922	1.0000

- And from ***variance decomposition***, we see that π is essentially driven by AS shock, **y** by IS shock, with **r** responding to everything (as Taylor Rule implies)

VARIANCE DECOMPOSITION SIMULATING ONE SHOCK AT A TIME (in percent)

	emp	eas	eis	Tot. lin. contr.
infl	1.69	96.25	3.21	101.15
y	3.42	2.29	91.62	97.34
r	26.72	21.13	55.29	103.14

Note: numbers do not add up to 100 due to non-zero correlation of simulated shocks

- ***"Flat"*** likelihood surfaces → it may be necessary to attempt ***several different*** Maximum Likelihood methodologies before finding one which is able to converge to a (local) optimum
- Not a problem here, but still of interest to compare results from estimation using different methodologies implemented in **Dynare**
- As seen in slide below, ***not all succeed***, even though all start from exactly the same point with exactly the same data …

mode_compute=4

parameters			
	Estimate	s.d.	t-stat
del	0.5251	0.0119	44.0952
lam	0.0098	0.007	1.3919
mu	0.5157	0.0204	25.271
phi	0.0031	0.0029	1.0583
roe	0.887	0.024	36.903
bet	1.2421	0.2735	4.5423
gam	2.5596	0.6858	3.7324
standard	deviation	of	shocks
	Estimate	s.d.	t-stat
eas	0.6248	0.035	17.832
eis	0.4135	0.0241	17.1412
emp	0.8579	0.0458	18.7421

mode_compute=5

Estimate	s.d.	t-stat
0.5251	0.0119	44.1002
0.0098	0.007	1.3909
0.5157	0.0204	25.2546
0.0031	0.0029	1.0583
0.8869	0.024	36.9021
1.2419	0.2734	4.5427
2.5592	0.6856	3.733
deviation	of	shocks
Estimate	s.d.	t-stat
0.6249	0.035	17.8316
0.4135	0.0241	17.1385
0.858	0.0458	18.7404

mode_compute=8

parameters			
	Estimate	s.d.	t-stat
del	0.5251	0.0119	44.0944
lam	0.0098	0.007	1.3916
mu	0.5157	0.0204	25.2646
phi	0.0031	0.0029	1.0583
roe	0.8869	0.024	36.9025
bet	1.2421	0.2734	4.5424
gam	2.5596	0.6858	3.7325
standard	deviation	of	shocks
	Estimate	s.d.	t-stat
eas	0.6248	0.035	17.8321
eis	0.4135	0.0241	17.1403
emp	0.8579	0.0458	18.7422

mode_compute=10

Estimate	s.d.	t-stat
0.5	NaN	NaN
0.1	NaN	NaN
0.5	NaN	NaN
-6919235	NaN	NaN
-8017485	NaN	NaN
-711758	NaN	NaN
-8476837	NaN	NaN
Estimate	s.d.	t-stat
0.6652	NaN	NaN
0.5	NaN	NaN
0.5	NaN	NaN

App.3 “Regularised” MLE

- This is implemented in Dynare very simply
- Going back to Cho-Moreno example, merely ***change "estimated params" section***; rest remains same

Except for ***one subtlety***: must include "***mh_replic=0***" in estimation command

```
estimated_params;
//  parameter name   initval lb  ub  prior shape    param1  param2  param3  param4  jscale
    del,             0.5,  ,   ,     uniform_pdf, , , -9999, 9999;
    lam,             0.1,  ,   ,     uniform_pdf, , , -9999, 9999;
    mu,              0.5,  ,   ,     uniform_pdf, , , -9999, 9999;
    phi,             0.1,  ,   ,     uniform_pdf, , , -9999, 9999;
    roe,             0.5,  ,   ,     uniform_pdf, , , -9999, 9999;
    bet,             1.5,  ,   ,     uniform_pdf, , , -9999, 9999;
    gam,             0.5,  ,   ,     uniform_pdf, , , -9999, 9999;

    stderr eas,      0.5,  ,   ,     uniform_pdf, , , 0.001, 9999;
    stderr eis,      0.5,  ,   ,     uniform_pdf, , , 0.001, 9999;
    stderr emp,      0.5,  ,   ,     uniform_pdf, , , 0.001, 9999;
end;
```

- Output of ***unconstrained*** ML estimation was:

```
Simplex iteration number: 2729-4-3
Simplex move:             reflect-0
Objective function value: 704.1908
Mode improvement:         0
Norm of dx:               0
Norm of dSimplex:         0.00020535
Crit. f:                  3.6954e-08
Crit. x:                  9.1275e-05

CONVERGENCE ACHIEVED AFTER 2729 ITERATIONS!

Final value of minus the log posterior (or likelihood):704.190819

RESULTS FROM MAXIMUM LIKELIHOOD ESTIMATION
parameters
    Estimate    s.d. t-stat

del   0.5251  0.0119 44.0944
lam   0.0098  0.0070  1.3916
mu    0.5157  0.0204 25.2646
phi   0.0031  0.0029  1.0583
roe   0.8869  0.0240 36.9025
bet   1.2421  0.2734  4.5424
gam   2.5596  0.6858  3.7325

standard deviation of shocks
    Estimate    s.d. t-stat

eas   0.6248  0.0350 17.8321
eis   0.4135  0.0241 17.1403
emp   0.8579  0.0458 18.7422
```

- Output of ML estimation with ***untruncated Uniform*** is:

```
Simplex iteration number: 3597-4-3
Simplex move:             reflect-0
Objective function value: 801.1453
Mode improvement:         0
Norm of dx:               0
Norm of dSimplex:         0.00025031
Crit. f:                  3.9259e-08
Crit. x:                  9.7418e-05

CONVERGENCE ACHIEVED AFTER 3597 ITERATIONS!

Final value of minus the log posterior (or likelihood):801.145253

RESULTS FROM POSTERIOR ESTIMATION
parameters
  prior mean      mode     s.d. prior pstdev

del    0.000   0.5251  0.0119 unif   5772.9253
lam    0.000   0.0098  0.0070 unif   5772.9253
mu     0.000   0.5157  0.0204 unif   5772.9253
phi    0.000   0.0031  0.0029 unif   5772.9253
roe    0.000   0.8870  0.0240 unif   5772.9253
bet    0.000   1.2421  0.2734 unif   5772.9253
gam    0.000   2.5596  0.6858 unif   5772.9253

standard deviation of shocks
  prior mean      mode     s.d. prior pstdev

eas 4999.501    0.6248  0.0350 unif   2886.4624
eis 4999.501    0.4135  0.0241 unif   2886.4624
emp 4999.501    0.8579  0.0458 unif   2886.4624
```

Parameters and std devs are ***identical*** to previous slide

- Using ***upper and lower bounds*** seen earlier (but unused) in (unconstrained) MLE

```
estimated_params;
//  parameter name    initval lb   ub   prior shape     param1  param2  param3  param4  jscale
    del,              0.5, ,   ,        uniform_pdf, , , 0, 1;
    lam,              0.1, ,   ,        uniform_pdf, , , -10, 10;
    mu,               0.5, ,   ,        uniform_pdf, , , 0, 1;
    phi,              0.1, ,   ,        uniform_pdf, , , -10, 10;
    roe,              0.5, ,   ,        uniform_pdf, , , 0, 1;
    bet,              1.5, ,   ,        uniform_pdf, , , 1, 10;
    gam,              0.5, ,   ,        uniform_pdf, , , 0, 10;

    stderr eas,       0.5, ,   ,        uniform_pdf, , , 0.5,0,100;
    stderr eis,       0.5, ,   ,        uniform_pdf, , , 0.5,0,100;
    stderr emp,       0.5, ,   ,        uniform_pdf, , , 0.5,0,100;
end;
```

- ➔ following results from ***Regularised MLE***

```
Final value of minus the log posterior (or likelihood):728.497604

RESULTS FROM POSTERIOR ESTIMATION
parameters
  prior mean     mode    s.d. prior pstdev

del   0.500   0.5251  0.0119 unif  0.2887
lam   0.000   0.0098  0.0070 unif  5.7735
mu    0.500   0.5157  0.0204 unif  0.2887
phi   0.000   0.0031  0.0029 unif  5.7735
roe   0.500   0.8870  0.0240 unif  0.2887
bet   5.500   1.2421  0.2735 unif  2.5981
gam   5.000   2.5596  0.6858 unif  2.8868

standard deviation of shocks
  prior mean     mode    s.d. prior pstdev

eas  50.000   0.6249  0.0350 unif  28.8675
eis  50.000   0.4135  0.0241 unif  28.8675
emp  50.000   0.8579  0.0458 unif  28.8675
```

- ***Identical to ordinary MLE*** since bounds, though present, are in fact ***unconstraining*** in this case

- Other bounds or other priors will give different results
- Eg, limiting gam to [0, 2.5] actually causes estimation ***to fail***

Table 5: Results from Maximum Likelihood maximization (parameters)

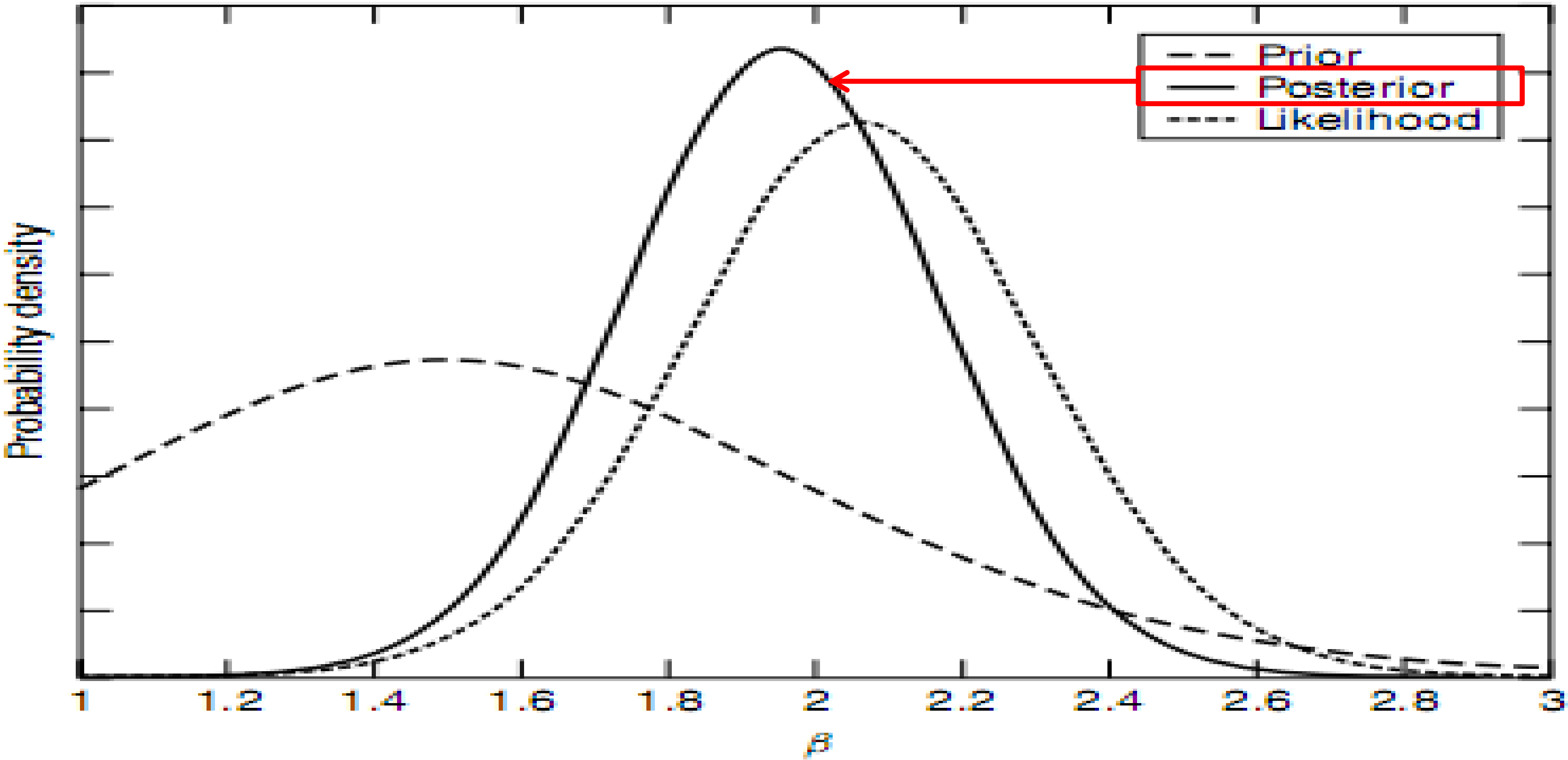
	Mode	s.d.	t-stat
del	0.5251	NaN	NaN
lam	0.0098	NaN	NaN
mu	0.5159	NaN	NaN
phi	0.0031	NaN	NaN
roe	0.8852	NaN	NaN
bet	1.2397	NaN	NaN
gam	2.5000	NaN	NaN

App.4 Bayesian Estimation

- Since it ***brings together*** elements of ***calibration*** (through priors) and ***maximum likelihood*** (through likelihood function maximisation), ***Regularised MLE*** using Bayesian estimation is able to cope with shortcomings of both calibration and maximum likelihood analysis
- By allowing consideration of priors, it avoids ***posterior distribution*** peaking at strange points where ***likelihood*** happens to peak (***rocky***)
- Moreover, inclusion of ***priors*** helps in identifying parameters in cases where ***likelihood function*** is ***flat***, which is a major ***drawback*** of MLE on its own

The effect of using a Bayesian prior

- When we combine a Bayesian prior with standard MLE, we obtain an “***averaging***” effect:



The Bayesian Prior

- Recap: “prior” distribution ***encapsulates external information*** which analyst has on range of possible values of parameters
- Prior distribution is then used, together with data, to obtain “posterior” distribution of parameters (weighted average of data and prior beliefs)
- How ***firmly*** those prior beliefs are held is specified in prior distribution by its ***variance***:
  - ***small*** variance ➔ very ***firmly-held*** belief;
  - ***large*** variance ➔ ***less certainty*** on the part of the analyst

- Comparing Models:
- Suppose there are i=1,2 models M_i with prior probability $p_i = P(M_i)$ that model M_i is true model
- Each model has a set of parameters θ_i and corresponding prior distribution and likelihood
- Then the probability of model 1 being the true model given the data, is given by

$$P(M_1|d) = \frac{P(M_1)\mathcal{L}_1(d|M_1)}{\mathcal{L}(d)} = \frac{p_1 \int \mathcal{L}_1(d,\theta_1|M_1)d\theta_1}{\mathcal{L}(d)}$$

$$= \frac{p_1 \int \mathcal{L}_1(d|\theta_1, M_1)\wp_1(\theta_1|M_1)d\theta_1}{\mathcal{L}(d)}$$

with $\mathcal{L}(d) = p_1 \int \mathcal{L}_1(d|\theta_1, M_1)\wp_1(\theta_1|M_1)d\theta_1 + p_2 \int \mathcal{L}_2(d|\theta_2, M_2)\wp_2(\theta_2|M_2)d\theta_2$

- The expected value of the likelihood given the prior distribution is the so-called ***marginal-likelihood*** for model i:

$$m_i(d) = \int \mathcal{L}_i(d|\theta_i, M_i)\wp_i(\theta_i|M_i)d\theta_i$$

- Using this, one can calculate the ***posterior-odds***:

$$PO_{12} = \frac{P(M_1|d)}{P(M_2|d)} = \underbrace{\frac{p_1}{p_2}}_{\text{Prior-Odds-Ratio}} \cdot \underbrace{\frac{m_1(d)}{m_2(d)}}_{Bayes-factor}$$

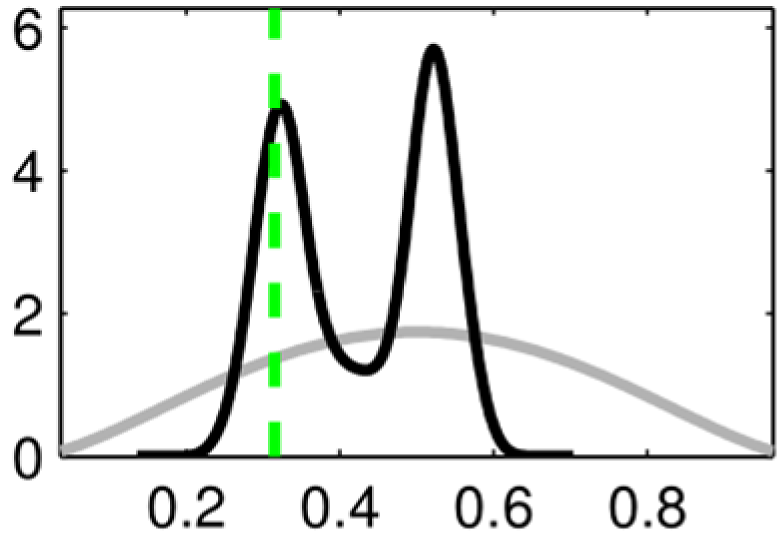
- A PO_{12} >>1 is an ***indication*** that the data as well as the priors prefer model 1
- Guidelines of Jeffrey (1961):
- 1:1 - 3:1 → ***weak*** evidence for model 1
- 10:1 - 100:1 → ***strong*** evidence for model 1
- > 100:1 → ***decisive*** evidence for model 1

Appendix 5

Maximum Likelihood on SW

- ***First phase*** of estimation is obtaining a ***maximum likelihood*** estimation of parameters from which to start Bayesian element
- If likelihood surface is ***very flat [or very rocky]*** and number of parameters more than a few, this can be a very ***tricky part*** of process and may require several attempts using ***different*** estimation algorithms and/or different parameters on priors

- ➔ Example: attempt to do MLE phase led to following posterior distribution, strongly indicative of a ***very strange*** likelihood surface



- Here, grey curve indicates prior, and black the posterior distribution, whose mode is green

- Despite this, now do ML estimation
- MLE in **Dynare** ➔ setting option ***mh_replic = 0*** (default value is 20,000) in ***estimation*** command
- ➔ begin ML estimation at ***starting values*** specified in "**estimated_params**" block (no priors!)
- Starting values used are those given by SW in their original **Dynare** file

- Partial example is

  sigma_c, $\boxed{1.2312}$, $\boxed{0.25}$, $\boxed{3}$;

  l, 0.7205, 0.001, 0.99;

  xi_w, 0.7937, 0.3, 0.95;

- recall, entries after parameter names indicate, respectively, ***initial value***, ***lower bound***, and ***upper bound***
- Results from ML estimation are shown on next slides

ML: C:\Quant1\SW\SW2007\Smets_Wouters_2007_45_jc2aer_test1.mod

- But first a minor ***practical subtlety***:
- For convenience, break programme up into four ***modules*** as follows:
- ***sw_defs.mod*** contains variable ***definitions***
- ***sw_model_def.mod*** contains ***equations***
- ***sw_estimation_def.mod*** contains ***priors***
- ***sw_RegMLE.mod*** contains ***estimation*** command as shown on next slide
- **Dynare**'s ***macro language*** uses ***@#include*** to instruct Matlab to include everything in various modules

```
@#include "sw_defs.mod"
@#include "sw_US_model_def.mod"
@#include "sw_estimation_def.mod"
estimation(Tex,
   nodiagnostic,
   optim=('Algorithm','active-set'),
   datafile=usmodel_data,
   mode_compute=1,
   first_obs=71,
   presample=4,
   lik_init=2,
   prefilter=0,
   mh_replic=0,
   mh_nblocks=2,
   mh_jscale=0.20,
   mh_drop=0.2
);
write_latex_prior_table;
write_latex_dynamic_model(write_equation_tags);
write_latex_parameter_table;
write_latex_definitions;
collect_latex_files;
```

Different data files for ***US*** and ***Euro area*** require different measurement equations; the rest is identical

These are results from ***MLE***

Table 1: Parameter Values

Parameter	Value	Description
ι_w	0.760	Indexation to past wages
ξ_w	0.881	Calvo parameter wages
ι_p	0.011	Indexation to past prices
ξ_p	0.660	Calvo parameter prices
σ_l	3.097	Frisch elasticity
σ_c	1.826	risk aversion
λ	0.695	external habit degree
Φ	1.845	fixed cost share
ψ	0.866	capacity utilization cost
φ	12.866	investment adjustment cost

r_π	2.998	Taylor rule inflation feedback
$r_{\Delta y}$	0.266	Taylor rule output growth feedback
r_y	0.172	Taylor rule output level feedback
ρ	0.884	interest rate persistence
$\bar{\gamma}$	0.403	net growth rate in percent
$\bar{l}$	-1.702	steady state hours
$\bar{\pi}$	1.241	steady state inflation rate
$100(\beta^{-1} - 1)$	0.011	time preference rate in percent
μ_w	0.969	coefficient on MA term wage markup
μ_p	0.707	coefficient on MA term price markup
α	0.171	capital share
ρ_{ga}	0.414	Feedback technology on exogenous spending

ρ_a	0.966	persistence productivity shock
ρ_b	0.063	persistence risk premium shock
ρ_g	0.983	persistence spending shock
ρ_i	0.625	persistence risk premium shock
ρ_r	0.010	persistence monetary policy shock
ρ_p	0.915	persistence price markup shock
ρ_w	0.981	persistence wage markup shock
ϕ_w	1.500	Gross markup wages
ε_w	10.000	Curvature Kimball aggregator wages
ε_p	10.000	Curvature Kimball aggregator prices
δ	0.025	depreciation rate
$\frac{\bar{g}}{\bar{y}}$	0.180	steady state exogenous spending share

These are the "calibrated" parameters set by SW

Table 2: Results from Maximum Likelihood maximization (parameters)

	Mode	s.d.	t-stat
ρ_a	0.9659	0.0079	122.0457
ρ_b	0.0627	0.0959	0.6536
ρ_g	0.9825	0.0084	117.5146
ρ_i	0.6249	0.0733	8.5253
ρ_r	0.0104	0.0874	0.1186
ρ_p	0.9151	0.0472	19.3841
ρ_w	0.9812	0.0146	67.2647
μ_p	0.7067	0.1413	5.0015
μ_w	0.9687	0.0157	61.8723
φ	12.8663	6.9673	1.8467
σ_c	1.8259	0.3712	4.9183
λ	0.6948	0.0648	10.7242

ξ_w	0.8815	0.0495	17.7968
σ_l	3.0971	1.8254	1.6967
ξ_p	0.6600	0.0725	9.0989
ι_w	0.7598	0.2327	3.2649
ι_p	0.0113	0.1298	0.0874
ψ	0.8663	0.1954	4.4328
Φ	1.8446	0.1571	11.7444
r_π	2.9977	0.6232	4.8100
ρ	0.8841	0.0257	34.3600
r_y	0.1723	0.0630	2.7369
$r_{\Delta y}$	0.2657	0.0485	5.4839
$\bar{\pi}$	1.2411	0.2784	4.4579
$100(\beta^{-1} - 1)$	0.0110	0.2031	0.0540
$\bar{l}$	-1.7017	2.0290	0.8387
$\bar{\gamma}$	0.4028	0.0214	18.7829
ρ_{ga}	0.4142	0.1170	3.5412
α	0.1713	0.0260	6.5762

- t-statistics ➔ most parameters ***highly statistically significant***
- ***Especially important*** for price ξ_p and wage ξ_w ***stickiness parameters***, which ***test*** NK theory
- ***Exceptions*** are ρ_b, ρ_r , ι_p , β and $\mathcal{T}$; of these, $\boldsymbol{\iota_p}$ is especially significant, as it is the price-indexation parameter
- ML estimates therefore seem to be indicating that there is very little, if any, ***price indexation***

Appendix 6

Blanchard-Kahn Conditions

- Related to failure of identification is ***an unfortunate reality***: failure of the so-called ***"Blanchard-Kahn"*** conditions
- Not infrequently, it happens that **Dynare** simply stops with the somewhat cryptic message ***"Blanchard-Kahn conditions not satisfied"***
- What does this mean?
- Answering this question will take us on an ***excursion into matrix algebra***, but it cannot be helped
- Blanchard and Kahn [1980] established that there was a ***unique stable solution*** to DSGE model so long as ***two conditions*** were satisfied

- ***First*** (***"counting condition"***): there have to be as many ***stable eigenvalues*** of a particular matrix as there are ***predetermined*** variables
- or, what is same thing, as many ***unstable eigenvalues*** as there are ***forward-looking*** variables
- ***Second***: another particular matrix which links unstable "canonical variables" and non-predetermined variables has to be of full rank
- This ***"rank condition"*** can only be checked in course of a detailed model solution and to that we now turn

- A typical DSGE model can be written in ***matrix form*** as follows

$$A_1 \begin{bmatrix} X_{t+1} \\ E_t P_{t+1} \end{bmatrix} = A_0 \begin{bmatrix} X_t \\ P_t \end{bmatrix} + \gamma Z_{t+1}$$

- where
  - X_t = (n x 1) vector collecting variables ***predetermined*** at time t, including backward-looking variables [the so-called ***"state variables"***]
  - P_t = (m x 1) vector collecting ***forward-looking*** variables, i.e. those entering model in form of ***expectations*** [known in Dynare as "***jumpers***"]
  - Z_{t+1} = (k x 1) vector collecting ***shocks*** (white noise)
  - A1, A0 = (n + m) x (n + m) ***coefficient*** matrices on variables
  - γ = (n + m) x k ***coefficient*** matrix on shocks

- If A_1 is ***not singular*** (ie, is ***invertible***), we may rewrite model as

$$\begin{bmatrix} X_{t+1} \\ E_t P_{t+1} \end{bmatrix} = A \begin{bmatrix} X_t \\ P_t \end{bmatrix} + R Z_{t+1}$$

- where obviously $A = A_1^{-1} A_0$ and $R = A_1^{-1}\gamma$
- According to ***Jordan decomposition*** result of matrix algebra, A can be decomposed as $A = C \Lambda C^{-1}$ where
  - Λ is a ***diagonal*** matrix with n + m distinct ***eigenvalues*** of A on its main diagonal, ***sorted*** with ***increasing*** absolute value (this is important!)
  - C is corresponding matrix of ***eigenvectors***

- Recall: A value λ_i is an ***eigenvalue*** of matrix A if ∃ a vector c_i (known as an ***eigenvector***) such that $Ac_i = \lambda_i c_i$
- Many (n × n) matrices have n distinct eigenvalues
- If C = matrix that has as its columns n eigenvectors corresponding to these eigenvalues
- Then AC = CΛ where

$$\Lambda = \begin{pmatrix} \lambda_1 & 0 & 0 & 0 \\ 0 & \lambda_2 & 0 & 0 \\ 0 & 0 & & 0 \\ 0 & 0 & 0 & \lambda_n \end{pmatrix}$$

- But AC = CΛ ➔ A = CΛC^{-1} (as on last-but-one slide)
- This tells us something about relationship between eigenvalues and higher powers of A because

$$A^n = C\Lambda^n C^{-1} = C \begin{pmatrix} \lambda_1^n & 0 & 0 & 0 \\ 0 & \lambda_2^n & 0 & 0 \\ 0 & 0 & & 0 \\ 0 & 0 & 0 & \lambda_n^n \end{pmatrix} C^{-1}$$

- If all eigenvalues are ***inside unit circle*** (i.e. ***less than one*** in absolute value) then all entries in A^n will tend towards ***zero*** as n → ∞

- But ∃ no guarantee that all eigenvalues will lie ***inside*** unit circle
- Those which lie ***outside*** unit circle will obviously be ***explosive***, since as n → ∞, they will also → ∞
- Recall that we had

$$\begin{bmatrix} X_{t+1} \\ E_t P_{t+1} \end{bmatrix} = A \begin{bmatrix} X_t \\ P_t \end{bmatrix} + RZ_{t+1}$$

- Premultiplying by C^{-1} yields

because A = CΛC^{-1}

$$C^{-1}\begin{bmatrix} X_{t+1} \\ E_t P_{t+1} \end{bmatrix} = \Lambda C^{-1} \begin{bmatrix} X_t \\ P_t \end{bmatrix} + C^{-1}RZ_{t+1}$$

- Next step is to write Λ in ***block form*** as

$$\Lambda = \begin{bmatrix} \Lambda_1 & 0 \\ 0 & \Lambda_2 \end{bmatrix}$$

- where:
- Λ_1 = (n x n) matrix collecting eigenvalues lying ***within*** unit circle (i.e. modulus <= 1)
- Λ_2 = (m x m) matrix collecting eigenvalues ***outside*** unit circle (possibly empty)

- Similarly for further use, decompose A, C^{-1} and R as follows

$$A = \begin{bmatrix} A_{11[n\times n]} & A_{12[n\times m]} \\ A_{21[m\times n]} & A_{22[m\times m]} \end{bmatrix}$$

$$C^{-1} = \begin{bmatrix} C_{11[n^*\times n]} & C_{12[n^*\times m]} \\ C_{21[m^*\times n]} & C_{22[m^*\times m]} \end{bmatrix}$$

$$R = \begin{bmatrix} R_{1[n\times k]} \\ R_{2[m\times k]} \end{bmatrix}$$

- And define

$$\begin{bmatrix} Y_t \\ Q_t \end{bmatrix} = C^{-1} \begin{bmatrix} X_t \\ P_t \end{bmatrix}$$

- All will become clear on next slide!!

- Doing this allows us to rewrite transformed model as

$$\tilde{E}_t \left[\begin{array}{c} Y_{t+1} \\ Q_{t+1} \end{array} \right] = \Lambda \left[\begin{array}{c} Y_t \\ Q_t \end{array} \right] + C^{-1} R Z_{t+1}$$

- We then ***iterate bottom blocks forward*** and using $E_t Z_{t+j+1} = 0$ find that $E_t Q_{t+j+1} = \Lambda_2^j Q_t$
- Since Λ_2 collects eigenvalues lying ***outside*** unit circle, solution is non-exploding ***only*** if $Q_t = 0 \ \forall\ t$, which in turn requires that $C_{21} X_t + C_{22} P_t = 0$
- And "rank condition" of ***Blanchard-Kahn*** is precisely that ***C_{22} be of full rank*** (m)

- To summarise, given our model

$$\begin{bmatrix} X_{t+1} \\ E_t P_{t+1} \end{bmatrix} = A \begin{bmatrix} X_t \\ P_t \end{bmatrix} + R Z_{t+1}$$

- Blanchard-Kahn "***counting condition***" is that number of eigenvalues of A lying outside unit circle (m*) is equal to number of "jumpers" [***forward-looking*** variables] (elements of P) ➔ m* = m
- Blanchard-Kahn "***rank condition***" is that C_{22} be of full rank (m) where C_{22} is lower right block of inverse of C, which is matrix of ***eigenvectors***

- In the Cho-Moreno model we have

```
Equation number 1 : 0 : New Keynesian Phillips Curve
Equation number 2 : 0 : Dynamic IS Curve
Equation number 3 : 0 : Taylor Rule
```

"0" means that model equation is consistent with the initial parameter values

```
EIGENVALUES:
        Modulus              Real          Imaginary

         0.7874            0.7874                  0
         0.9002            0.9001            0.01533
         0.9002            0.9001           -0.01533
          1.008             1.008                  0
          1.099             1.099                  0
```

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

```
The rank condition is verified.
```

- You will have forgotten by now, but this was all based on ***assumption*** that A_1 was ***invertible*** in model

$$A_1 \begin{bmatrix} X_{t+1} \\ E_t P_{t+1} \end{bmatrix} = A_0 \begin{bmatrix} X_t \\ P_t \end{bmatrix} + \gamma Z_{t+1}$$

- It turns out that this is ***often*** true, but ***not always*** and there is an extensive literature dealing with cases when it is ***not***
- ***Dynare*** is able to deal with these cases, using what is known as “generalized Schur decomposition”, also known as “***QZ Factorization***”, which solves a ***generalized*** eigenvalue problem

- Basic idea of generalized Schur decomposition is as follows: Starting from

$$A_0 E_t Y_{t+1} = A_1 Y_t + v_{t+1}, \text{ where } Y_t = \begin{bmatrix} X_{t-1} \\ X_t \end{bmatrix}$$

1. perform generalized Schur (QZ) decomposition of system
2. check for Blanchard-Kahn conditions
3. transform system into a triangular one and isolate unstable block
4. solve unstable (2nd) block forward
5. from this, solve stable block (1st) backward
6. ***solution*** of system can be written as a ***VAR(1)***:

$$Y_t = Y^s + AY_{t-1} + Bv_t$$

- Given this ***VAR(1)*** solution

$$Y_t = Y^s + AY_{t-1} + Bv_t$$

- in ***Dynare***, output is such that
- – Y^s: ***dr_.ys*** steady state values
- – A: ***dr_.ghx*** coef. on state variables
- – B: ***dr_.ghu*** coef. on exogenous variables
- This simple ***VAR(1)*** form of model is known as ***Decision Rule*** version as it makes it easy to compute moments, IRFs and forecasts

- This is all very abstract, but an example of this ***Decision Rule*** structure can be found in Cho-Moreno model we estimated earlier:

POLICY AND TRANSITION FUNCTIONS

	infl	y	r
r(-1)	-0.018915	-0.028446	0.876792
infl(-1)	0.890690	0.002845	0.115626
y(-1)	0.073827	0.856651	0.267932
emp	-0.021247	-0.031952	0.984887
eas	1.871343	0.005978	0.242931
eis	0.154606	1.793960	0.561092

- For each ***"current endogenous variable"***, columns indicate coefficients on lagged variables (**A**) and shocks (**B**) [Y^S = 0 by construction]

$$Y_t = Y^s + AY_{t-1} + Bv_t$$

State-Space Model

- We are not yet at end of our troubles!
- ML procedure used until now implied that ***all*** variables were observable
- But many DSGE models have a ***mix*** of observable and ***unobservable*** variables
- For example, although we can easily observe variables in ***national accounting identities*** (Y, C, I, G, X and M) we cannot directly observe ***technical change*** variables nor ***potential output*** nor ***capital stocks***
- And since we do not really know ***capital depreciation rates***, we cannot construct capital variable used in ***production functions***

- How to deal with this unfortunate reality?
- It turns out that a type of model known as ***"state space" model*** can handle a mixture of observable and non-observable variables
- Such models are characterised by ***two equations:***
- ***State (or "transition") equation***: describes how ***unobservable "state"*** variables evolve over time
- ***Measurement equation***: relates ***observable*** variables to ***unobservable*** state variables

- ➔ ***State Space*** model may be represented as

$$
\begin{aligned}
y_t^* &= M\bar{y}(\theta) + M\hat{y}_t + N(\theta)x_t + \eta_t \\
\hat{y}_t &= g_y(\theta)\hat{y}_{t-1} + g_u(\theta)u_t \\
E(\eta_t \eta_t') &= V(\theta) \\
E(u_t u_t') &= Q(\theta)
\end{aligned}
$$

- Here,
  - $\hat{y}_t$ are unobserved ***"state"*** variables measured in ***deviations*** from steady state
  - $\bar{y}$ is vector of ***steady state*** values
  - $y_t{}^*$ are variables ***actually observed***
  - θ is vector of deep (or structural) parameters to be estimated

- Second equation is just our ***decision rule***

$$\hat{y}_t = g_y(\theta)\hat{y}_{t-1} + g_u(\theta)u_t$$

- First equation expresses a relationship among ***true*** endogenous variables that are ***not directly observed***

$$y_t^* = M\bar{y}(\theta) + M\hat{y}_t + N(\theta)x_t + \eta_t$$

- Only y_t^* is observable, and it is related to true variables with an error η_t
- Any ***trend*** is captured by $N(\theta)x_t$ to allow for most general case in which trend depends on deep parameters

- A bit of jiggery-pokery has to be done to get model in state-space form, but ***all standard DSGE models can be re-arranged to be put in this format***
- This is important because it means we can use ….
- … (infamous) ***Kalman Filter*** to do maximum likelihood estimation of DSGE models that mix observable and unobservable variables
- Because Kalman Filter is ***standard method*** to estimate state-space models
- Luckily for you, **Dynare** implements Kalman Filter in background and you do not need to know anything about how it works

Kalman Filter

- Recall that ***all*** DSGE models may be written as state-space models, where ***measurement equation***, relating a set of ***observable*** variables Z_t to the ***unobservable*** state variables S_t is

  $$Z_t = HS_t + v_t \quad (\text{where } S_t = FS_{t-1} + u_t)$$

- We can't observe S_t but suppose we could replace it by an ***observable unbiased guess*** based on information available up to time t – 1
- Call this guess $S_{t|t-1}$ and suppose its errors are ***normally*** distributed with a known covariance matrix so that

  $$S_t - S_{t|t-1} \sim N\left(0, \Sigma^S_{t|t-1}\right)$$

- ***Trick***: ***observed*** variables may be written as

$$Z_t = HS_{t|t-1} + v_t + H(S_t - S_{t|t-1})$$

- Because $S_{t|t-1}$ is ***observable*** and ***unobservable*** elements (v_t and $S_t - S_{t|t-1}$) are normally distributed, this model can be estimated via ***maximum-likelihood*** methods!
  - (Actually, Normality is not necessary – all ML needs is a known probability distribution from which (log-) likelihood function can be computed.)
- Variance of error terms may be written as

$$v_t + H\left(S_t - S_{t|t-1}\right) \sim N\left(0, \Omega_t\right)$$
$$\Omega_t = \Sigma^v + H\Sigma^S_{t|t-1}H'$$

- Then, if θ collects all parameters of model, so that θ =(F, H, Σ^v, Σ^u), we have log-likelihood function for Z_t given observables at time t−1 as

$$\log f\left(Z_t | Z_{t-1}, \theta\right) = -\log 2\pi - \log|\Omega_t| - \frac{1}{2}\left(Z_t - HS_{t|t-1}\right)' \Omega_t^{-1}\left(Z_t - HS_{t|t-1}\right)$$

- ***Given an initial estimate*** of first-period unobservable state $S_{1|0}$, combined likelihood for all observed data is ***product*** of all period-by-period likelihoods

$$f\left(Z_1, Z_2, ..., Z_T | S_{1|0}, \theta\right) = f\left(Z_1 | S_{1|0}, \theta\right) \prod_{i=2}^{i=T} f\left(Z_i | Z_{i-1}, \theta\right)$$

- And so combined ***log-likelihood*** function for observed dataset is given by ***sum***

$$\log f\left(Z_1, Z_2, ..., Z_T | S_{1|0}, \theta\right) = -T \log 2\pi - \sum_{i=1}^{T} \log |\Omega_i|$$

$$-\frac{1}{2}\sum_{i=1}^{T}\left(Z_i - HS_{i|i-1}\right)' \Omega_i^{-1}\left(Z_i - HS_{i|i-1}\right)$$

- Maximum-likelihood parameter estimates are set of matrices $\theta = (F, H, \Sigma^v, \Sigma^u)$ that provide largest value for this function
- So we now have a method to estimate model's parameters via MLE ***provided*** we have an unbiased guess based on information available up to time t−1, which we called $S_{t|t-1}$, with normally distributed errors

- How to generate these unbiased guesses?
- Answer: ***Kalman Filter***
- It is an ***iterative*** method: starting from one period's estimates of state variables, it uses observable data for next period to ***update*** these estimates
- Start with an estimate of state variable at time t given information at time t−1:

$$S_t = FS_{t-1} + u_t \Rightarrow S_{t|t-1} = FS_{t-1|t-1}$$

- This means that in period t − 1, ***expected values*** for ***observables*** in period t are

$$Z_{t|t-1} = HS_{t|t-1} = HFS_{t-1|t-1} \quad [\text{since } E_t(u_t) = 0]$$

- Then in period t, when we observe Z_t, how do we update our guesses for state variable in light of the "news" in $Z_t - HFS_{t-1|t-1}$?
- It turns out that ***minimum variance unbiased estimate*** of S_t given observed Z_t is

$$E\,(S_t | Z_t) = S_{t|t} = FS_{t-1|t-1} + K_t\,(Z_t - HFS_{t-1|t-1})$$

- where

$$K_t = \left(H\Sigma^S_{t|t-1}\right)' \left(\Sigma^v + H\Sigma^S_{t|t-1}H'\right)^{-1}$$

- K_t matrix is known as ***Kalman Gain*** matrix
- So ***new*** estimate is just ***old*** estimate plus an adjustment using Kalman Gain matrix

- Covariance matrices required to compute this K_t matrix are updated by formulae:

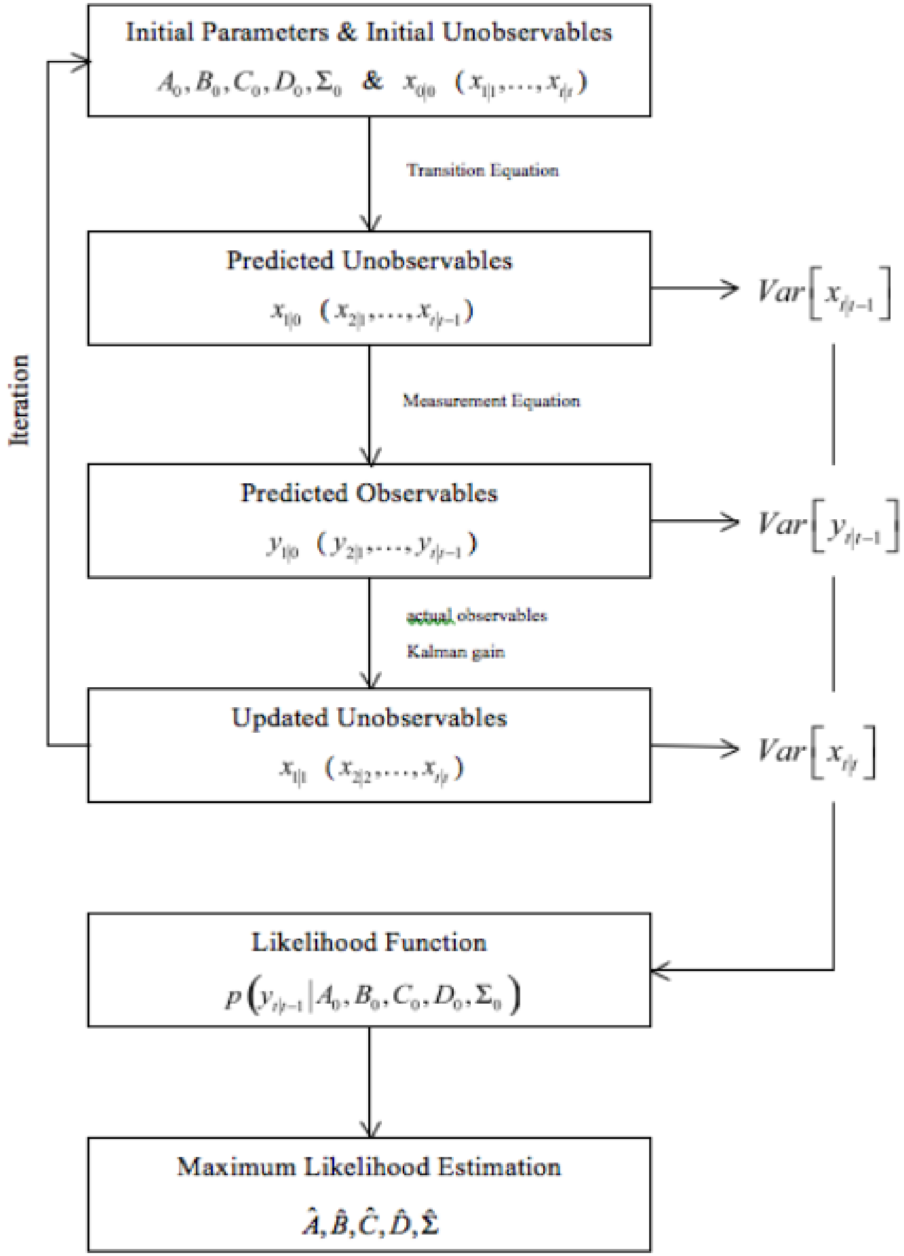
$$\Sigma^S_{t|t-1} = F\Sigma^S_{t-1|t-1}F^{'} + \Sigma^u$$
$$\Sigma^S_{t|t} = (I - K_t H)\,\Sigma^S_{t|t-1}$$

- We are nearly finished, but we still need an ***initial estimate*** $S_{1|0}$ as well as its covariance matrix to start Kalman filter process
- Initial estimate $S_{1|0}$ typically will be = ***steady-state*** value

- And initial estimate of covariance matrix typically will be equal to value of Σ which solves matrix equation $\Sigma = F\Sigma F' + \Sigma^u$
- This completes Kalman Filter ***initiation*** process and rest is just ***iteration***
- It is worth noting that Kalman Filter is what is known as a ***one-sided*** filter: Estimates of states at time t are based ***solely*** on information available at time t; ***no data after period t*** is used to calculate estimates of unobserved state variables

- This is a reasonable model for how someone might behave if they were learning about state variables in ***real time***
- But researchers have access to the ***full history*** of data set, including all observations after time t
- For this reason, economists generally estimate time-varying models using a method known as ***Kalman Smoother***
- This is a ***two-sided*** filter that uses data ***both before and after time t*** to compute expected values of state variables at time t
- It is also implemented in ***Dynare***

Estimation via Kalman Filter



Hodrick-Prescott Filter

- Hodrick-Prescott (HP) Filter is often used in macroeconomic research as a way of choosing a trend for series that has a time-varying trend Y_t*
- HP Filter minimises

$$\sum_{t=1}^{N}\left[(Y_t - Y_t^*)^2 + \lambda\left(\Delta Y_t^* - \Delta Y_{t-1}^*\right)\right]$$

- Although this seems fairly ad hoc, it can be shown that ***for large samples, HP Filter technique is same as Kalman Filter*** estimation of a particular state-space model

- Given state-space model

$$
\begin{aligned}
Y_t &= Y_t^* + C_t \\
\Delta Y_t^* &= \Delta Y_{t-1}^* + \epsilon_t^g \\
C_t &= \epsilon_t^c
\end{aligned}
$$

- where

$$\text{Var}(\epsilon_t^g) = \sigma_g^2 \text{ and } \text{Var}(\epsilon_t^c) = \sigma_c^2$$

- Here variable Y_t is split into two components
  - its "***long-term trend***" Y_t^*
  - its ***cyclical variation*** C_t
- It turns out that HP and Kalman results are ***identical*** if in HP formula we set $\lambda = \frac{\sigma_c^2}{\sigma_g^2}$

- Hodrick and Prescott used ***quarterly*** data and assumed that cyclical component C_t had a standard deviation of 5 percentage points while ϵ_t^g had a standard deviation of one-eighth of a percentage point; hence they chose

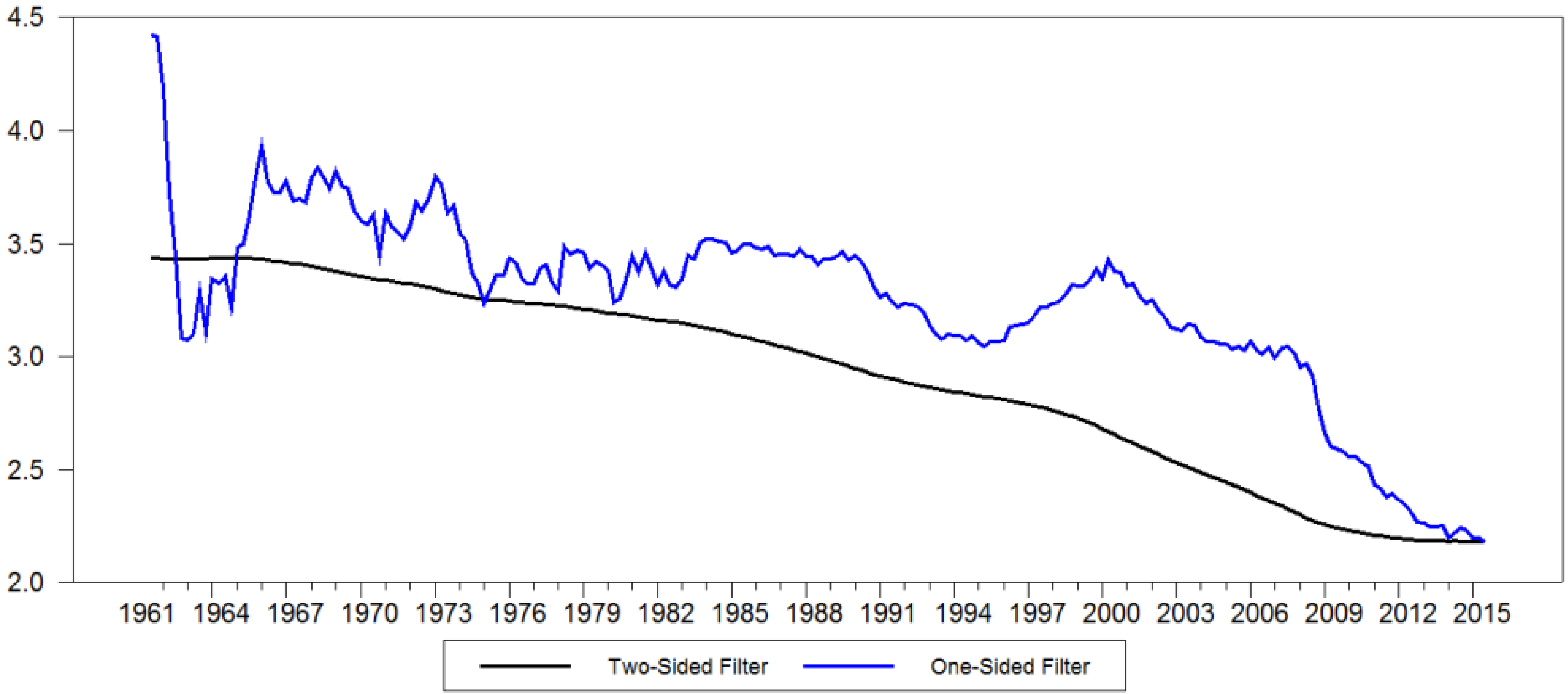
$$\lambda = \frac{5^2}{\left(\frac{1}{8}\right)^2} = (25)(64) = 1600$$

- This value of λ is ***standard value*** used in HP Filter for quarterly data; for annual data, λ = 6.25; and for monthly data, λ = 128000

- Note: since log-linearized versions of most DSGE models make use of variables which are (log) deviations from their steady-state values, researchers often apply ***HP Filter*** when constructing data for use in DSGE modelling
- However, just as for Kalman, so too HP Filter may be calculated using a ***one-sided*** or a ***two-sided*** process (and default setting – for example in Eviews – is usually two-sided)
- Several recent papers have strongly argued for using only ***one-sided*** estimates; slide below suggests why – and also why two-sided Kalman is called ***"Smoother"!***

- This comes from San Francisco Fed which updates the Laubach and Williams (2001) results

One-Sided and Two-Sided Estimates of Potential Output Growth



- Finally, you should be aware that use of HP is not universally accepted
- See paper by James Hamilton “Why You Should Never Use The Hodrick-Prescott Filter” 2018 NBER WP23429