Christiano
Assignment 9 (to be run with Dynare version 4)

Tutorial on Forecasting, Output Gap Estimation, DSGE Model Estimation and the MCMC Algorithm Using Dynare

As a reminder of example 5 from the model-solution handout, following are the equations of the Clarida-Gali-Gertler model.

$$\begin{aligned}
\beta E_t \pi_{t+1} + \kappa x_t - \pi_t &= 0 \text{ (Calvo pricing equation)} \\
-\left[r_t - E_t \pi_{t+1} - rr_t^*\right] + E_t x_{t+1} - x_t &= 0 \text{ (intertemporal equation)} \\
\alpha r_{t-1} + u_t + (1-\alpha)\phi_\pi \pi_t + (1-\alpha)\phi_x x_t - r_t &= 0 \text{ (policy rule)} \\
rr_t^* - \rho \Delta a_t - \frac{1}{1+\varphi}(1-\lambda)\tau_t &= 0 \text{ (definition of natural rate)} \\
y_t^* &= a_t - \frac{1}{1+\varphi}\tau_t \text{ (natural output)} \\
x_t &= y_t - y_t^* \text{ (output gap)}
\end{aligned}$$

You will need the Dynare files, cggsim.mod and cggest.mod, as well as the MATLAB m files, plots.m, analyzegap.m, suptitle.m and HPFAST.m, to do this assignment (you can see answers in cggsimans.mod and cggestans.mod).

The HP filter is defined as follows:

$$\min_{\left\{y_t^T\right\}_{t=1}} \sum_{t=1}^{T} \left(y_t - y_t^T\right)^2 + \lambda \sum_{t=2}^{T-1} \left[\left(y_{t+1}^T - y_t^T\right) - \left(y_t^T - y_{t-1}^T\right)\right]^2$$

The parameter, λ, controls how 'smooth' y_t^T is. If $\lambda = 0$, then $y_t = y_t^T$. If $\lambda = \infty$, then y_t^T is a time trend (i.e., a line whose second derivative is zero). In business cycle analysis, it is customary to use $\lambda = 1600$ in studying quarterly. The MATLAB m-file, [y_hp,y_hptrend]=HPFAST(y,lambda) takes y as input and puts out y_hp=$y_t - y_t^T$, y_hptrend=y_t^T.

This assignment explores three things: (i) the estimation of the output gap using the HP filter and a model (ii) estimation, by Bayesian and maximum likelihood methods, of a model, and (iii) the MCMC algorithm as a device for approximating a posterior distribution.

1. For our first exercise, we explore the MCMC algorithm and the Laplace approximation in a simple example. Technical details about both these objects are discussed in the lecture notes. One practical consideration not mentioned in the notes is relevant for the case in which the pdf of interest is of a non-negative random variable. Since the jump distribution is Normal, a negative x could be drawn (see the notes for a detailed discussion of x and the 'jump distribution'). A 'quick and dirty' approach in this case is to work with the absolute value of x.

   Hopefully, it is apparent that the MCMC algorithm is quite simple, and can be programmed by anyone with a relatively small exposure to MATLAB. A useful exercise to understand how the algorithm works, is to use it to see how well it approximates a simple known function. Thus, consider the Weibull probability distribution function (pdf),

   $$g(\theta) = ba^{-b}\theta^{b-1}e^{-\left(\frac{\theta}{a}\right)^2},\ x \geq 0,$$

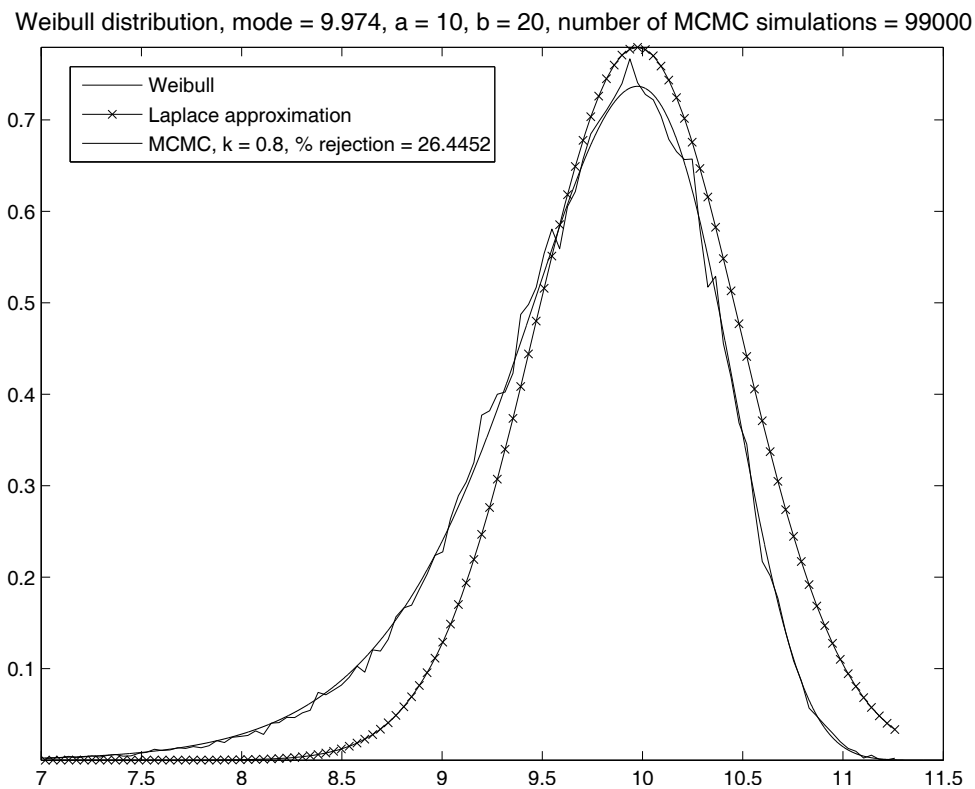
   where a, b are parameters. (For an explanation of this pdf, see the MATLAB documentation of $[g] = wblpdf(\theta, a, b)$.) Consider $a = 10$, $b = 20$. Graph this pdf over the grid, $[7, 11.5]$, with intervals 0.001 (i.e., graph g on the vertical axis, where $g = wblpdf(x, 10, 20)$, and x on the horizontal axis, where $x = 7 : .001 : 11.5$). Compute the mode of this pdf by finding the element in your grid with the highest value of g. Compute the second derivative of the Weibull at the mode point numerically, using the formula,

   $$f''(x) = \frac{f(x + 2\varepsilon) - 2f(x) + f(x - 2\varepsilon)}{4\varepsilon^2},$$

   for ε small (for example, you could set $\varepsilon = 0.000001$.) Here, x plays the role of θ^* and f plays the role of the MATLAB function, *wblpdf*. Set $V = -f''(\theta^*)^{-1}$.

   Set $M = 1,000$ (a very small number!) and set $k = 1$. Graph the density function implied by the MCMC approximation, the actual density

function, and the one implied by the Laplace approximation. The value of k, $k = 1$, seems to be a bit large. A value of 0.8 seems to work better. Note how rough the MCMC approximation is. In part, this is due to the fact that we have used a relatively small value for M. Try $M = 10,000$. When I set $M = 100,000$ and $k = 0.80$, I obtained (see the MATLAB code WeibullMCMC.m) the following result:



Note how well the MCMC approximation works. The Laplace approximation assigns too much density near the mode, and lacks the skewness of the Weibull. Still, for practical purposes the Laplace may be workable, at least as a first approximation in the initial stages of a research project. This could be verified in the early stages of the project by doing a run using the MCMC algorithm and comparing the results with those of the Laplace approximation.

2. For the remainder of this assignment, it will be useful to have a sense of how the CGG economy responds to a shock. Following is the parameterization used in the lecture:

$$\begin{aligned}\beta &= 0.97,\ \phi_x = 0,\ \phi_\pi = 1.5,\ \alpha = 0,\ \rho = 0.2,\ \lambda = 0.5,\ \delta = 0.2,\\ \varphi &= 1,\ \theta = 0.75,\ \sigma_a = \sigma_\tau = 0.02,\ \sigma_u = 0.\end{aligned}$$

   (a) In the case of the technology and preference shocks, use Dynare to compute the impulse response functions of the variables to each shock. The m file, plots.m, produces these in a format that resembles the one in the model-solving lecture notes. You may want to use these. In particular, note whether the economy over- or under- responds to the shock compared to what natural output does. What is the economic intuition in each case?

   (b) Do the calculations with $\phi_\pi = 0.99$. What sort of message does Dynare generate, and can you provide the economic intuition for it?

   (c) Return to the parameterization, $\phi_\pi = 1.5$. Now, insert r_t into the Cavlo pricing equation. Redo the calculations and note how Dynare reports indeterminacy. Provide economic intuition for your result.

   (d) Consider the following alternative parameterization, which is more appealing from an empirical point of view:

$$\begin{aligned}\beta &= 0.97,\ \phi_x = 0.15,\ \phi_\pi = 1.5,\ \alpha = 0.8,\ \rho = 0.9,\ \lambda = 0.5,\ \delta = 0.2,\\ \varphi &= 1,\ \theta = 0.75,\ \sigma_a = \sigma_\tau = 0.02,\ \sigma_u = 0.\end{aligned}$$

   Look at the impulse response functions to the preference and technology shocks. Do they make sense?

3. Generate $T = 200$ artificial observations on the 'endogenous' (in the sense of Dynare) variables of the model. These are the variables in the 'var' list. The mod file provided, cggsim.mod, has 7 variables. Before doing the simulation, you should add the growth rate of output to the equations of the model and to the var list (call it 'dy'.) That way, Dynare will also simulate output growth. The variables simulated by

Dynare are placed in the $n \times T$ matrix, oo_.endo_simul.[1] The n rows of oo_.endo_simul correspond to the $n = 8$ variables in var, *listed in the order in which you have listed them in the var statement* from the first to the last row. To verify the order that Dynare puts the variables in, see how they are ordered in M_.endo_names in the Dynare-created file, cggsim.m.

Retrieve output growth from oo_.endo_simul and get the log level of output, y, using y =cumsum($dysim$), where $dysim$ is the name I arbitrarily assigned to the row of oo_.endo_simul corresponding to output growth. Also, retrieve x from the appropriate row of oo_.endo_simul and create natural output from the relation, $y^* = y - x$.

(a) Compute the HP filter of y with $\lambda = 1$ and display a graph with y and y^T. Do this also for $\lambda = 1600$ and for λ =160,000,000. Do the results accord with what you would expect, given the formula for the HP filter above?

(b) Graph the HP filter trend, y^T, ($\lambda = 1600$) along with y and y^*. Note how actual output is somewhat more volatile than potential or natural output (recall, the economy overreacts to technology shocks). As a result, the HP filter with $\lambda = 1600$ over smooths the data. Graph $y_t - y_t^T$ and the true output gap, x_t, as well as y, y^T and y^*. Compute the correlation between $y_t - y_t^T$ and x_t. Also compute the correlation for the case where technology shocks are dominant (i.e., $\sigma_a = 2$, $\sigma_\tau = 0.02$) and for the case where preference shocks are dominant (i.e., $\sigma_a = 0.02$, $\sigma_\tau = 2$). Interpret the results. The MATLAB command for computing the correlation between two variables, w_t and u_t, is corrcoef(w,u). The result of this calculation is a 2×2 matrix with unity on the diagonal and the correlation on the off-diagonal.

The model of this question lies close to the heart of the main paradigm underlying the current view about the monetary transmission mechanism. Note that in the case of this model, the hp-filter is not terrible as a guide to the output gap. This is because the technology shock is the important shock in the dynamics of the data, and the actual data

[1] Here, endo_simul is the matrix, which is a 'field' in the structure, oo_.

overreact to the technology shock. That is, the natural rate of output is a smooth version of the data. Of course, this is only an example, and is something worth pursuing more carefully using a DSGE model that has more solid empirical foundations.

4. Now we will do some estimation. First, we generate artificial data from the baseline parameterization of the model. Place the simulated data, oo_.endo_simul, in the matrix, data. Then, save these data to a MATLAB file, data, using the instruction, save data data. Also, set periods = 5000 in the stoch_simul command. Run the mod file using Dynare. This saves the simulated data. Second, open cggest.mod.

   (a) First, do maximum likelihood estimation. Use 4,000 observations to verify that everything is working properly. Consistency of maximum likelihood implies that with this many observations, the probability that the estimates are far from the true parameter values is low. Try doing the estimation when you start far from the true parameter value, say with rho=lambda=0.9. Despite the bad initial guess about the parameter values, you should end up roughly at the true values.

   (b) Redo (a), but now with 30 observations, and you should see that maximum likelihood still works well. Note that although the point estimates look quite good, the standard error on lambda is rather large.

5. Now do Bayesian estimation, using the inverted gamma distribution as the prior on the two standard deviations and the beta distribution as the prior on the two autocorrelations.

   (a) Set the mean of the priors over the parameters to the corresponding true values. Set the standard deviation of the inverted gamma to 10 and of the beta to 0.04. (It's hard to interpret these standard deviations directly, but you will see graphs of the priors, which are easier to interpret.) Use 30 observations in the estimation. Adjust the value of k, so that you get a reasonable acceptance rate. I found that $k = 1.2$ works well. Have a look at the posteriors, and notice how, with one exception, they are much tighter than the

priors. The exception is lambda, where the posterior and prior are very similar. This is evidence that there is little information in the data about lambda.

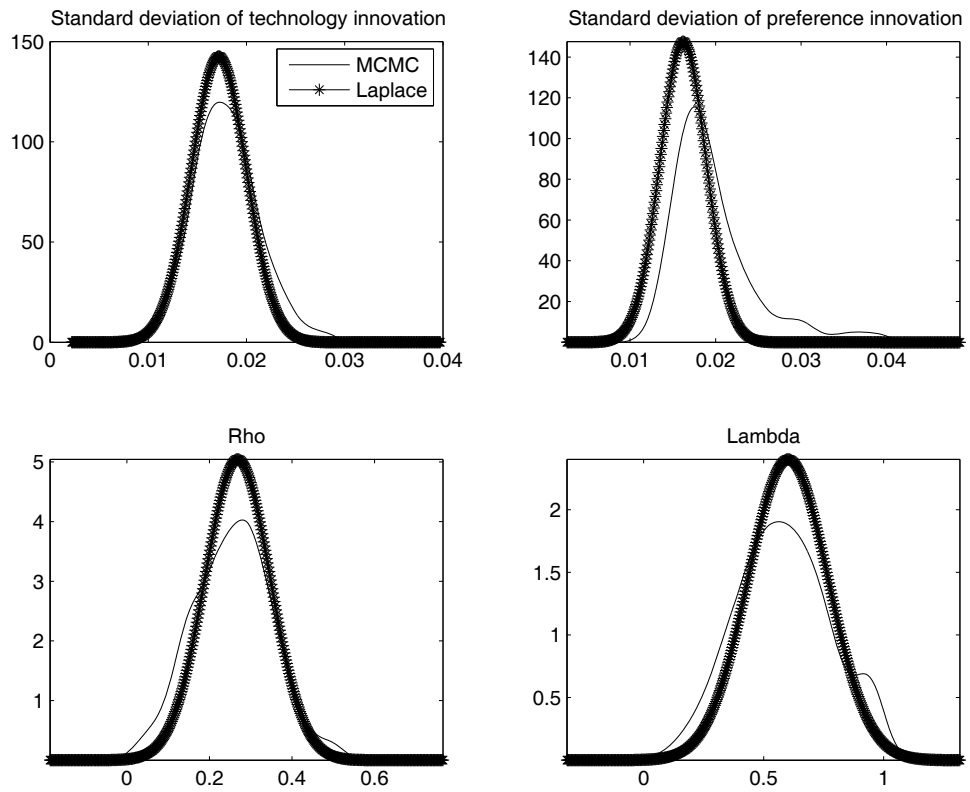
(b) Redo (a), but set the mean and standard deviation of the prior on lambda equal to 0.95 and 0.04, respectively. Note how the prior and posterior are again very similar. There is not much information in the data about the value of lambda!

(c) Note how the priors on σ_a and ρ have faint 'shoulders' on the right side. Redo (a), with $M = 4,000$ (M is mh_replic, which controls the number of MCMC replications). Note that the posteriors are now smoother. Actually, $M = 4,000$ is a small number of replications to use in practice.

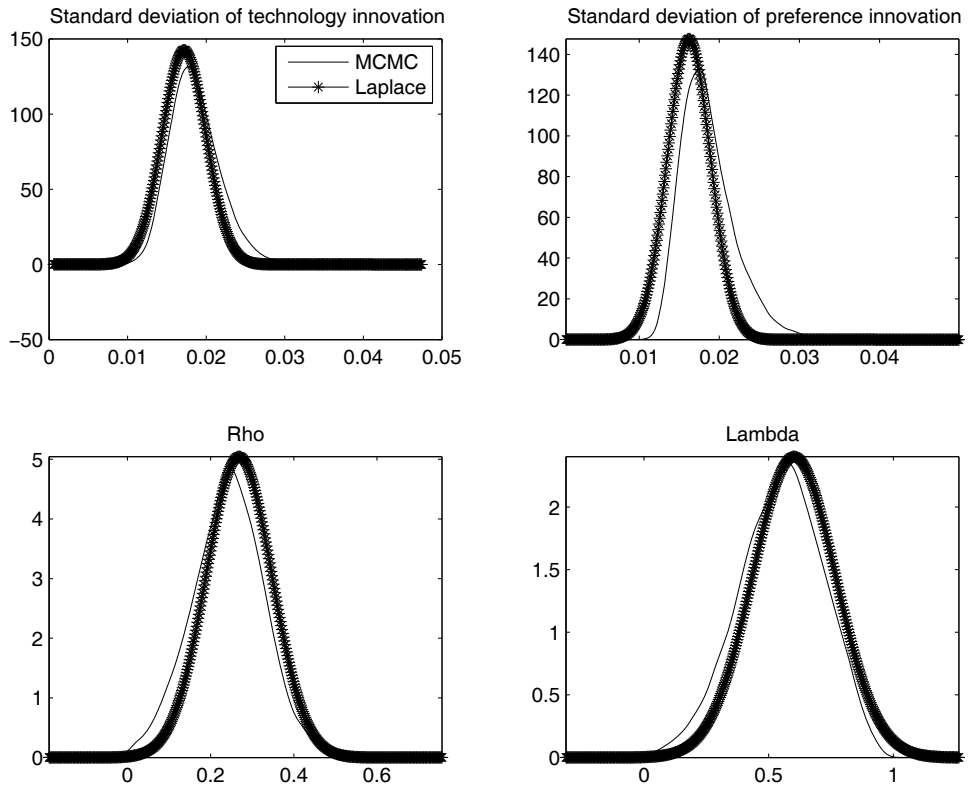
(d) Now set the mean of the priors on the standard deviations to 0.1, far from the truth. Set the prior standard deviation on the inverted gamma distributions to 1. Keep the observations at 30, and see how the posteriors compare with the priors. (Reset $M = 1,000$ so that the computations go quickly.) Note that the posteriors move sharply back into the neighborhood of 0.02. Evidently, there is a lot of information in the data about these parameters.

(e) Repeat (a) with 4,000 observations. Compare the priors and posteriors. Note how, with one exception, the posteriors are 'spikes'. The exception, of course, is lambda. Still, the difference between the prior and posterior in this case indicates there is information in the data about lambda.

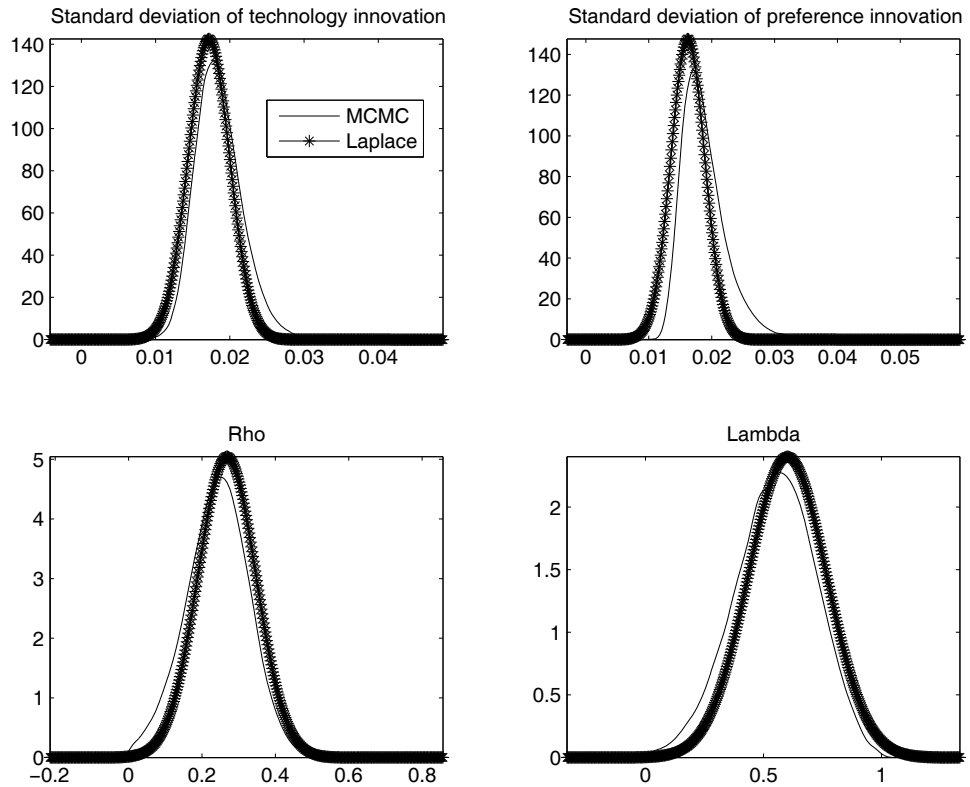
6. It is of interest to compare the posterior densities approximated by the MCMC algorithm with the Laplace approximation. Consider the setup in 5 (a). You can recover all the information you need for these calculations from the structure, oo_. The posterior distributions of the parameters and shock standard errors are in the structure oo_.posterior_density. Posterior modes are in oo_.posterior_mode. Posterior standard deviations (taken from the relevant diagonal parts of the inverse of the hessian of the log criterion) appear in oo_.posterior_variance (my code for recovering these objects is compareMCMCLaplace.m. Setting $M = 10,000$, I found



When I set $M = 100,000$, the MCMC posteriors became smoother:



Note how much more similar the MCMC and Laplace posteriors are. The tail areas of the MCMC posteriors have thinned out and now resemble more closely the Laplace. Next, I set $M = 1,000,000$ and obtained virtually the same result as with $M = 100,000$:



Thus, in this example it seems that the MCMC algorithm has roughly converged for $M = 100,000$. In addition the Laplace and MCMC approximations deliver very similar results, consistent with the conclusion that the Laplace approach can used at the start and middle of a research project, while the MCMC can be done later on. Note that in any particular project, you can 'test' this proposition doing comparison of the posterior distribution obtained by the Laplace approximation with the posterior distribution obtained by MCMC.

7. The output gap is not in the dataset used in the econometric estimation. However, as noted in the handout, it is possible to use the Kalman filter to estimate the output gap (actually, everything in the state) from the available data. There are two ways to do this: 'smoothing' uses the entire data set and 'filtering' only uses the part of the dataset prior to the date for which the estimate of the gap is formed (thus, filtered data are one-step-ahead forecasts). To activate the Kalman smoother

in Dynare, include the argument, smoother, in the estimation argument list. The smoothed estimates will then be placed in a MATLAB structure oo_.SmoothedVariables. This structure can be accessed either directly from the command line, or by selecting >desktop>workspace from the pull down menu available in the command window. You will see that inside oo_.SmoothedVariables there are a number of subcategories, with output related to the Bayesian estimation (for example, oo_.SmoothedVariables.Median.x displays the median smoothed estimate of the output gap, x). To see how well the Dynare-estimated version of the model does at producing a good guess of the output gap, include the code, analyzegap.m, at the end of your mod file. This shows you how to recover the smoothed output gap from Dynare, and allows you to compare it with the actual output gap, as well as with the hp-filtered estimate of the output gap.

8. Dynare also reports confidence intervals for the smoothed variables (e.g., oo_.SmoothedVariables.HPDinf.x contains the upper bound of the 95 percent confidence interval for x, in case you set conf_sig =.95 in the Dynare estimation command). These reflect parameter uncertainty, as well as the difficulty of recovering these variables from the observed data when they are not in the data set. If you run analyzegap.m down to line 41, you will see what this confidence interval looks like, by comparison with the actual gap. Note that occasionally, the actual gap lies outside the confidence interval, as is to be expected.

9. The analysis in the previous question suggests that the output gap can be estimated reliably using the estimated dsge model. However, in practice one needs the output gap in real time. For this, the smoothed estimates of the output gap are not a reliable indicator. Instead, it is useful to look at the filtered estimates. These are found by running analyzegap.m to line 56 (the code shows how these are recovered from oo_.FilteredVariables). Note that there is a systematic phase shift between the estimated and actual gaps. This is as expected. Turning points are hard to 'see' in real time. They become evident only after the fact. (Dynare also reports 'updated' variables. These are forecasts of the data based on current and past observations. Not surprisingly, the updated 'estimates' of variables that happen to be in the econome-

trician's data set happen to coincide with their true values. This is not so for filtered variables.)

10. It is interesting to see how the HP filter works in real time. By running analyze.m down to line 75 one obtains an estimate of this. Note that the HP filter does not exhibit the same phase shift as the filtered data. This is because for date t I have computed the HP filter using data up to and including date t.

11. Dynare will also do forecasting. For this, one includes the argument, forecast=xx, where xx indicates how many periods in the future you want to forecast. (Put in xx=12.) To obtain the forecasts, as well as forecast uncertainty, execute the rest of analyzegap.m. You can see from the analyzegap.m code where in oo_.PointForecast the forecasts as well as the forecast uncertainty is stored.