Solving DSGE Models with Dynare

Graduate Macro II, Spring 2010
The University of Notre Dame
Professor Sims

1 Introduction

This document will present some simple examples of how to solve, simulate, and estimate DSGE models using Dynare. Dynare is not its own program but is rather basically a collection of Matlab codes. To run Dynare, you must first install it. You can download it from the following website: http://www.dynare.org/download. I will only support downloads using Matlab. You need to choose a destination folder. My destination folder is under the desktop folder "Research", then in the subfolder "dynare_new", then in the subfolder "4.1.0" (that's the version I currently have), then in the subfolder "Matlab". So the full destination is: "C:\Documents and Settings\esims1\Desktop\Research\dynare_new\4.1.0\matlab".

To run Dynare, you have to create .mod files (you can create these in the m-file editor, but be sure to save them as .mod, not as .m files). Then, to run your code, simply type "dynare filename" into the command window (or you can have this command within a seperate m-file which can be in another directory so long as you call up the Dynare directory).

2 A Neoclassical Model with Fixed Labor

Consider a simple stochastic neoclassical model with fixed labor. The planner's problem can be written:

$$\max \quad E_0 \sum_{t=0}^{\infty} \beta^t \frac{c_t^{1-\sigma} - 1}{1-\sigma}$$

s.t.

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

TFP is assumed to follow a mean zero AR(1) in the log:

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

The first order conditions for this problem can be characterized with three non-linear difference equations and a transversality condition:

$$\begin{aligned} c_t^{-\sigma} &= \beta E_t c_{t+1}^{-\sigma}(\alpha a_{t+1} k_{t+1}^{\alpha-1} + (1-\delta)) \\ k_{t+1} &= a_t k_t^{\alpha} - c_t + (1-\delta)k_t \\ \ln a_t &= \rho \ln a_{t-1} + \varepsilon_t \\ \lim_{t\to\infty} \beta^t c_t^{-\sigma} k_{t+1} &= 0 \end{aligned}$$

In addition, I have two auxiliary "static" variables that define output from the production function and investment from the accounting identity:

$$\begin{aligned} y_t &= a_t k_t^{\alpha} \\ i_t &= y_t - c_t \end{aligned}$$

The parameters that need to be calibrated are σ, α, δ, β, ρ, and σ_ε. I set the coefficient of relative risk aversion to 2, capital's share to 1/3, the depreciation rate to 0.025, the discount factor to 0.99, the persistence of technology shocks to 0.95, and the standard deviation of technology shocks to 0.01 (i.e. one percent).

2.1 A Timing Convention

There is a timing convention in Dynare that requires that we (slightly) rewrite the model. In particulary, Dynare requires that predetermined variables (like the capital stock) show up as dated $t-1$ in the time t equations and t in the $t+1$ equations. As such, we need to rewrite the FOCs:

$$\begin{aligned} c_t^{-\sigma} &= \beta E_t c_{t+1}^{-\sigma}(\alpha a_{t+1} k_t^{\alpha-1} + (1-\delta)) \\ k_t &= a_t k_{t-1}^{\alpha} - c_t + (1-\delta)k_{t-1} \\ \ln a_t &= \rho \ln a_{t-1} + \varepsilon_t \\ \lim_{t\to\infty} \beta^t c_t^{-\sigma} k_t &= 0 \\ y_t &= a_t k_{t-1}^{\alpha} \\ i_t &= y_t - c_t \end{aligned}$$

In other words, we essentially just lag the capital stock one period in each relevant equation.

2.2 Writing the Dynare Code

With the first order conditions in hand, we are now ready to begin to write our dynare code (which occurs in the .mod file). The dynare code will not run unless each entry is followed by a semicolon to suppress the output.

The first step is to declare the variables of the model. Dynare automatically sorts variables into the following order: static variables (i.e. only enter FOCs at time t), purely predetermined variables (only enter FOCs at t and $t-1$), variables that are both predetermined and forward-looking (appear at $t-1$, t, and $t+1$), and variables that are purely forward-looking (only appear at dates t and $t+1$). In our model, output is static, capital is purely predetermined, technology is both predtermined and forward-looking (since future TFP shows up in the Euler equation), and consumption is forward-looking. It makes sense to declare the variables in this order. The first non-comment command in your Dynare code should be: "var" followed by the endogenous variable names and culminating with a semicolon. For the purposes of naming things, everything other than the shock(s) is considered to be endogenous My first line of code looks like:

```
var y i k a c;
```

The next line of code specifies the exogenous variables. These are just shocks, and in this model I have only one. The line of code is:

```
varexo e;
```

The next step is to declare the parameters of the model. You simply type "parameters" followed by the names of those parameters:

```
parameters alpha beta delta rho sigma sigmae;
```

Following this command you want to specify values of these parameters. You do that below:

```
alpha = 0.33;
beta = 0.99;
delta = 0.025;
rho = 0.95;
sigma = 2;
sigmae = 0.01;
```

Now that you've named variables and specified parameter values, it's now time to "declare the model". To do this, you type in "model;" followed by the first order conditions, constraints, identities, etc., followed by "end;". We typically want to approximate the solutions of the natural logs of the variables, so that impulse responses, etc. are all in percentage terms. Dynare will do linear approximations of the levels of the variables. To get it to do linear approximation of the logs of the variables, you need to specify the variables as "exp(x)". This way the variable x is interpreted as the log of the variable of interest, while exp(x) is the level (since the exponential and log are inverse functions), which is what shows up in most of the FOCs. Then you just type in the first order conditions. If a variable appears dated t, then you just type x. If a variable is $t+1$, you type $x(+1)$; if it's $t-1$, then $x(-1)$. If, for whatever reason, you need variables beyond one period away to appear, you could define auxiliary state variables to make the system work out this way. My first order conditions for the above model look like:

```
model;
exp(c)^(-sigma) = beta*(exp(c(+1))^(-sigma))*(alpha*exp(a(+1))*(exp(k))^(alpha-
1) + (1-delta));
exp(y) = exp(a)*exp(k(-1))^(alpha);
exp(k) = exp(a)*exp(k(-1))^(alpha) - exp(c) + (1-delta)*exp(k(-1));
a = rho*a(-1) + e;
exp(i) = exp(y) - exp(c);
end;
```

Again, remember the timing convention on the capital stock, and remember to enter all variables as "exp(x)" instead of as just x. The reason I don't have the exponent on the process for technology is because it already appears as a log. Since $\exp(\ln x) = x$, we're not changing any of the FOC above by writing them this way, but are linearizing the logs of the variables (i.e. percentage deviations). Thus, the initial variables that I'm declaring are interpreted as logs (that's why I don't have an exp in the process for log technology). If you ever wanted a variable to not appear as a percentage deviation, like with an interest rate, you would also not write it as an exponential. Remember to close all command lines with a semi-colon.

To verify that writing it this way will do the log-linear approximation correctly, let's consider a couple of examples. Take the production function:

$$y_t = \exp(\ln a_t) k_t^{\alpha}$$

Because I've assumed a_t is the log of technology but want the level in the production function, I need to write it with an exp here. Let's linearize this without taking logs:

$$\begin{aligned} y^* + (y_t - y^*) &= k^{*\alpha} + (a_t - a^*)k^{*\alpha} + \alpha k^{*\alpha-1}(k_t - k^*) \\ (y_t - y^*) &= (a_t - a^*)k^{*\alpha} + \alpha k^{*\alpha-1}(k_t - k^*) \end{aligned}$$

The linearization has made use of the fact that $\exp(\ln a^*) = 1$. Now let's rewrite the process using exponentials:

$$\exp(\ln y_t) = \exp(\ln a_t) \exp(\alpha \ln k_t)$$

Now linearize about the steady state:

$$\begin{aligned} \exp(\ln y^*) + \frac{1}{y^*}\exp(\ln y^*)(y_t - y^*) &= \exp(\ln a^*)\exp(\alpha \ln k^*) + (a_t - a^*)\exp(\alpha \ln k^*) + ... + \\ &\quad ... + \frac{\alpha}{k^*}\exp(\ln a^*)\exp(\alpha \ln k^*)(k_t - k^*) \\ \frac{1}{y^*}\exp(\ln y^*)(y_t - y^*) &= (a_t - a^*)\exp(\ln a^*)\exp(\alpha \ln k^*) + ... \\ &\quad ... + \frac{\alpha}{k^*}\exp(\ln a^*)\exp(\alpha \ln k^*)(k_t - k^*) \\ \frac{y_t - y^*}{y^*} &= \frac{a_t - a^*}{a^*} + \alpha \frac{k_t - k^*}{k^*} \end{aligned}$$

In other words, you get the percentage deviations if you write out the FOC using this change of variable notation.

The next step is to specifiy initial values of the variables; given these initial values Dynare will numerically search for a steady state. Problems can arise if you give it bad initial values. When giving initial values, remember that it's interpreting all variables as logs. So if you solve analytically for a steady state capital stock of 30, you need to give an initial value of the capital stock in the neighborhood of the natural log of 30, not 30. You begin this part of the code with "initval;", followed by initial values for all of the endogenous variables, capped off with "end;". My code for this part looks like:

```
initval;
k = log(29);
y = log(3);
a = 0;
c = log(2.5);
i = log(1.5);
end;
```

The next step is to specify the variance of the shocks. This code starts with "shocks;", followed by a specification of the variance, followed by "end;". My code here looks like:

```
shocks;
var e = sigmae^2;
end;
```

In the next step you simply type in "steady;" – this calculates the steady state:

```
steady;
```

The next command is the payoff. It's the "stoch_simul" command, which is what solves the model, produces the policy functions, and generates impulse responses functions and unconditional second moments. There are a number of options following this command. If you just type in "stoch_simul;", it's going to give you the default output. The default output is: (1) steady state values of endogenous variables; (2) a model summary, which counts variables by type; (3) covariance matrix of shocks (which in the example I'm giving is a scalar); (4) the policy and transition functions (in state space notation); (5) theoretical first and second moments; (6) a theoretical correlation matrix; (7) theoretical autocovariances up to order 5; and (8) impulse responses. By default, Dynare does a second order approximation about the steady state; the second order approximation involves "cross terms", and so the policy/transition functions look more complicated than you're used to seeing.

There are a number of options one can include behind "stoch_simul;" (to do this you type "stoch_simul(options);". There are several of these that you can read about in the manual, but the more important ones for our purposes are:

- hp_filter = integer: this will produce theoretical moments (variances, covariances, autocorrelations) after HP filtering the data (the default is to apply no filter to the data). We typically want to look at HP filtered moments, so this is an option you'll want to use. The integer is a number (i.e. 1600) corresponding to the penalty parameter in the HP filter. So, for example, typing "stoch_simul(hp_filter=1600);" will produce theoretical (i.e. analytical) moments of the HP filtered data. The only problem here is that it will not simultaneously do HP filtering of simulated data; you can either get moments from simulated data or analytical HP filtered moments

- irf = integer: this will change the number of periods plotted in the impulse response functions. The default value is 40. To suppress printing impulse responses altogether, type in 0 for the number of horizons. For example, to see impulse responses for only 20 periods, type "stoch_simul(irf=20);".

- nocorr: this will mean it will not print a matrix of correlations in the command window
- nofunctions: this will suppress printing the state space coefficients of the solution in the command window
- nomoments: this will suppress printing of moments
- noprint: this will suppress printing of any output at all
- order = 1 or 2: this tells Dynare the order of the (log) approximation. The default is a second order approximation. Hence, typing "order=1" will have it do a linear approximation.
- periods = integer: Dynare's default is to produce analytical/theoretical moments of the variables. Having periods not equal to zero will instead have it simulate data and take the moments from the simulated data. By default, Dynare drops the first 100 values from a simulation, so you need to give it a number of periods greater than 100 for this to work. Hence, typing "stoch_simul(periods=300);" will produce moments based on a simulation with 200 variables.
- drop = integer: You can change the number of observations to drop from the simulations. For example, typing "stoch_simul(drop=0);" will result in no observations being dropped in the simulations.
- simul_seed = integer: sets the seed used in the random number generator for the simulations.

For example, typing "stoch_simul(nofunctions,hp_filter=1600,order=1,irf=20);" will suppress the policy function output, will produce analytical HP filtered moments, will do a first order approximation, and will plot 20 periods in the impulse responses.

Dynare has a weird way of storing some of the output. The impulse responses will be stored as "x_e": this is the impulse response of the variable x to the shock e. If you want to be able to access the policy functions you have to look a little harder. The state space representation for the linearized model is as follows:

$$x_t = Cs_{t-1} + De_t$$

s is the vector of states, while x is a vector of all variables (including states and non-states). The command "oo_.dr.ghu" will recover D, the command "oo_.dr.ghx" will recover the C, and "oo_.dr.ys" will recover the steady states (Dynare will actually print out the policy functions with constants, where the constants correspond to steady state values).

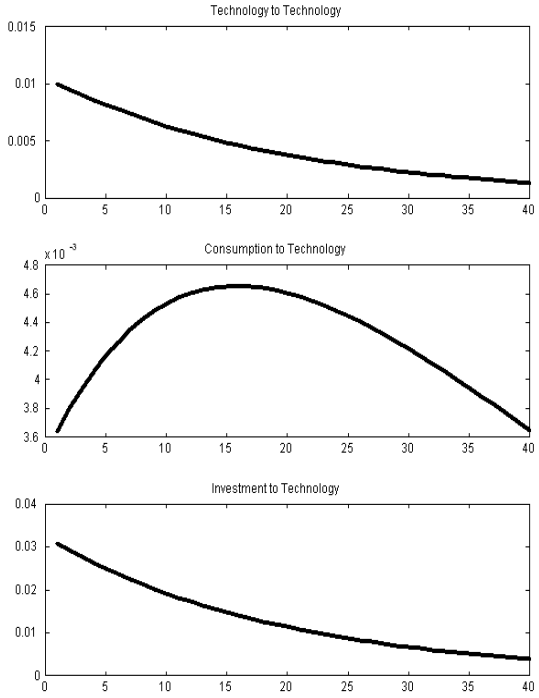
My full code looks like this:

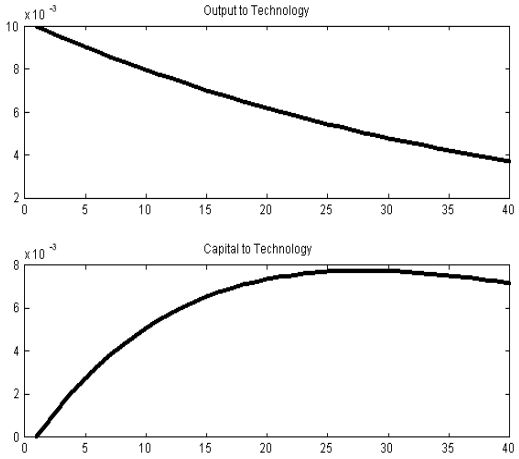
```
var y i k a c;
varexo e;
parameters alpha beta delta rho sigma sigmae;
alpha = 0.33;
beta = 0.99;
```

```
delta = 0.025;
rho = 0.95;
sigma = 2;
sigmae = 0.01;
model;
exp(c)^(-sigma) = beta*(exp(c(+1))^(-sigma))*(alpha*exp(a(+1))*(exp(k))^(alpha-
1) + (1-delta));
exp(y) = exp(a)*exp(k(-1))^(alpha);
exp(k) = exp(a)*exp(k(-1))^(alpha) - exp(c) + (1-delta)*exp(k(-1));
a = rho*a(-1) + e;
exp(i) = exp(y) - exp(c);
end;
initval;
k = log(29);
y = log(3);
a = 0;
c = log(2.5);
i = log(1.5);
end;
shocks;
var e = sigmae^2;
end;
steady;
stoch_simul(hp_filter = 1600, order = 1, irf = 40);
```

You can insert comments into the .mod file just like a regular Matlab file (by typing % before). To actually run the .mod file, you can't click F5 or type in the name of the .mod file into the Matlab command prompt. You have to type in "dynare filename" (without the .mod on the end of the file name). You have to be in the directory where your Dynare codes are stored. If you want to write a .mod file saved elsewhere you can run that too, but you need to type in "addpath('C:\Documents and Settings\esims1\Desktop\Research\dynare_new\4.1.0\matlab')" where the exact directions will change depending on where you save your Dynare.

Below are the impulse responses to a technology shock (I changed the defaults on the graph that it produced):





An annoying feature related to the timing convention is that the impulse response of capital (which will be stored as "k_e") is off by one period. Because in the model k_t is really k_{t+1}, it's going to show capital stock jumping on impact, which obviously cannot happen. You need to "lag" the IRF of capital one period to get things right.

Below is the imported output (kind of hard to read):

STEADY-STATE RESULTS:
y 1.10371
i -0.344308
k 3.34457
a 0
c 0.835782
MODEL SUMMARY
Number of variables: 5
Number of stochastic shocks: 1
Number of state variables: 2
Number of jumpers: 2
Number of static variables: 2
MATRIX OF COVARIANCE OF EXOGENOUS SHOCKS
Variables e
e 0.000100
POLICY AND TRANSITION FUNCTIONS
y i k a c
Constant 1.103709 -0.344308 3.344571 0 0.835782

k(-1) 0.330000 -0.029780 0.974256 0 0.440543
a(-1) 0.950000 2.916523 0.072913 0.950000 0.345784
e 1.000000 3.070024 0.076751 1.000000 0.363983
$THEORETICALMOMENTS(HPfilter, lambda = 1600)$
VARIABLE MEAN STD. DEV. VARIANCE
y 1.1037 0.0130 0.0002
i -0.3443 0.0400 0.0016
k 3.3446 0.0036 0.0000
a 0.0000 0.0130 0.0002
c 0.8358 0.0049 0.0000
$MATRIXOFCORRELATIONS(HPfilter, lambda = 1600)$
Variables y i k a c
y 1.0000 0.9956 0.3178 0.9959 0.9725
i 0.9956 1.0000 0.2281 1.0000 0.9466
k 0.3178 0.2281 1.0000 0.2307 0.5298
a 0.9959 1.0000 0.2307 1.0000 0.9475
c 0.9725 0.9466 0.5298 0.9475 1.0000
$COEFFICIENTSOFAUTOCORRELATION(HPfilter, lambda = 1600)$
Order 1 2 3 4 5
y 0.7195 0.4810 0.2826 0.1216 -0.0055
i 0.7131 0.4710 0.2709 0.1096 -0.0165
k 0.9603 0.8640 0.7306 0.5759 0.4128
a 0.7133 0.4711 0.2711 0.1098 -0.0163
c 0.7528 0.5341 0.3447 0.1845 0.0524

For this problem doing a second order approximation (as opposed to a first order) appears to make very little difference unless the curvature of the utility function begins to get very strong.

Now, given your output from Dynare, you may want to do some simulating of your own. The solution of the model is presented in state space form. As noted above, this takes the form:

$$x_t = Cs_{t-1} + D\varepsilon_t$$

In the model I've laid out, I have five variables, two states, and one shock. Thus, x_t is 5×1, C is 5×2, s_t is 2×1, D is 5×1, and ε_t is a scalar. The transition matrix C is stored as oo_.dr.ghx; thus, you can type "C = oo_.dr.ghx;" to recover that. The "impact matrix" D is stored as oo_.dr.ghu; hence, type "D = oo_.dr.ghu;" to get that. Dynare also reports a constant (the steady state) in the state space notation; you do NOT want to use this in any simulation you do, since the variables are expressed as deviations about a steady state. Only after you have simulated the variables do you want to add back in the steady state value/intercept. Doing it during the simulations will result in explosion.

If you wanted to do your own simulation, begin by drawing shocks from a distribution with the appropriate standard deviation. For example, to draw a sample of length T with the appropriate standard deviation from a normal distribution, you'd type:

```
e=sigmae*randn(T,1);
```

Then you have to generate a time series of states seperately. Let $\widehat{C}$ be the rows of C corresponding with the two states, and let $\widehat{D}$ be defined similarly. Specifying an initial condition of zero, you could simulate the time paths of the states as follows:

```
s(:,1) = Dhat*e(1,1);
for j = 2:T
s(:,j) = Chat*s(:,j-1) + Dhat*e(j,1);
end
```

Given a time series for the states, you can now form a time series for the rest of the variables. Define $\widehat{C}_2$ and $\widehat{D}_2$ as the rows of the C and D corresponding to non-states. Then you can simulate those time paths given the simulated paths of the states as follows:

```
c(:,1) = Dhat2*e(1,1);
for j = 2:T
c(:,j) = Chat2*s(:,j-1) + Dhat2*e(j,1);
end
```

2.3 Advanced Topic: Where to write your codes

You can write codes and .mod files in different folders than where the Dyanre codes are stored. For example, for the above I have a .mod file called "neoclassical_dynare_log.mod" stored in my teaching folder, whereas the Dynare code is stored in my research folder. I can create another m-file in my teaching folder that will run this with the following lines of code:

```
clear all;
close all;
addpath('C:\Documents and Settings\esims1\Desktop\Research\dynare_new\4.1.0\matlab')
dynare neoclassical_dynare_log
```

The .mod file needs to be in the same director that I'm currently in, but the "addpath" command will access the stuff from the Dynare directory on your machine.

3 Dealing With Trend Growth

In the model above, there is no trend growth. That's obviously a bit problematic since we obviously observe trend growth. I'm going to introduce a trend component to productivity, call it x_t. This is distinct (and uncorrelated with) TFP, a_t. The production function is now:

$$y_t = a_t x_t k_t^{\alpha}$$

Let x_t obey the following process:

$$\begin{aligned} x_t &= \exp(\gamma t) \\ \ln x_t &= \gamma t \end{aligned}$$

This just says that x_t grows at a constant, deterministic rate, γ. Because the solution techniques require that the model be stationary, we need to transform the variables of the model to deal with this. Define the following transformed variables:

$$\begin{aligned} \widehat{y}_t &\equiv \frac{y_t}{x_t^{\frac{1}{1-\alpha}}} \\ \widehat{k}_t &\equiv \frac{k_t}{x_t^{\frac{1}{1-\alpha}}} \\ \widehat{c}_t &\equiv \frac{c_t}{x_t^{\frac{1}{1-\alpha}}} \\ \widehat{i}_t &\equiv \frac{i_t}{x_t^{\frac{1}{1-\alpha}}} \end{aligned}$$

We don't need to transform a_t since it's exogenously stationary. Start with the production function. Divide both sides by $x_t^{\frac{1}{1-\alpha}}$:

$$\begin{aligned} \frac{y_t}{x_t^{\frac{1}{1-\alpha}}} &= a_t x_t^{1-\frac{1}{1-\alpha}} k_t^{\alpha} \\ \widehat{y}_t &= a_t x_t^{-\frac{\alpha}{1-\alpha}} \left(\frac{k_t}{x_t^{\frac{1}{1-\alpha}}} \right)^{\alpha} x_t^{\frac{\alpha}{1-\alpha}} \\ \widehat{y}_t &= a_t x_t^{-\frac{\alpha}{1-\alpha}+\frac{\alpha}{1-\alpha}} \widehat{k}_t^{\alpha} = \widehat{y}_t = a_t \widehat{k}_t^{\alpha} \end{aligned}$$

In other words, the transformed production function looks exactly the same as without trend growth. Now go to the capital accumulation equation:

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

Divide both sides by $x_t^{\frac{1}{1-\alpha}}$ and proceed to "play around":

$$\begin{aligned} \frac{k_{t+1}}{x_t^{\frac{1}{1-\alpha}}} &= \frac{a_t x_t k_t^{\alpha}}{x_t^{\frac{1}{1-\alpha}}} - \frac{c_t}{x_t^{\frac{1}{1-\alpha}}} + (1-\delta)\frac{k_t}{x_t^{\frac{1}{1-\alpha}}} \\ \frac{k_{t+1}}{x_t^{\frac{1}{1-\alpha}}} &= \frac{a_t x_t k_t^{\alpha}}{x_t^{\frac{1}{1-\alpha}}} - \widehat{c}_t + (1-\delta)\widehat{k}_t \\ \frac{k_{t+1}}{x_t^{\frac{1}{1-\alpha}}} &= a_t x_t^{-\frac{\alpha}{1-\alpha}} \left(\frac{k_t}{x_t^{\frac{1}{1-\alpha}}} \right)^{\alpha} x_t^{\frac{\alpha}{1-\alpha}} - \widehat{c}_t + (1-\delta)\widehat{k}_t \\ \frac{k_{t+1}}{x_t^{\frac{1}{1-\alpha}}} &= a_t \widehat{k}_t^{\alpha} - \widehat{c}_t + (1-\delta)\widehat{k}_t \end{aligned}$$

Now we need to play around on the LHS a little bit. We need to be dividing by $x_{t+1}^{\frac{1}{1-\alpha}}$, not $x_t^{\frac{1}{1-\alpha}}$. From the process for x, we know that $x_{t+1} = \exp(\gamma)x_t$, or $x_t = \frac{1}{\exp(\gamma)}x_{t+1}$. Plug that in on the left hand side:

$$\begin{aligned}\frac{k_{t+1}}{\left(\frac{1}{\exp(\gamma)}x_{t+1}\right)^{\frac{1}{1-\alpha}}} &= a_t\widehat{k}_t^{\alpha} - \widehat{c}_t + (1-\delta)\widehat{k}_t \\ \exp(\gamma)^{\frac{1}{1-\alpha}}\widehat{k}_{t+1} &= a_t\widehat{k}_t^{\alpha} - \widehat{c}_t + (1-\delta)\widehat{k}_t\end{aligned}$$

In other words, the capital accumulation equation looks very similar, but with an adjustment factor $\exp(\gamma)^{\frac{1}{1-\alpha}}$ on the left hand side. Suppose that $\gamma = 0.0033$ and $\alpha = 0.33$. This implies that the deterministic component of productivity grows at rate 0.33 percent per quarter, or about 1.25 percent per year. With those numbers, the adjustment factor is going to be equal to 1.0049, which is so small that it's not really going to make any difference whether you have it in there or not.

The accounting identity is easy to deal with:

$$y_t = c_t + i_t$$

Simply divide both sides by $x_t^{\frac{1}{1-\alpha}}$, which will leave the same equation in the transformed variables:

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

The last thing to do is to alter preferences. The within period utility function is:

$$\beta^t \frac{c_t^{1-\sigma} - 1}{1-\sigma}$$

Transform consumption without altering the value of this:

$$\beta^t \frac{\left(\frac{c_t}{x_t^{\frac{1}{1-\alpha}}}\right)^{1-\sigma} x_t^{\frac{1-\sigma}{1-\alpha}} - 1}{1-\sigma}$$

Recall that $x_t = \exp(\gamma t)$, which is equivalent to $x_t = (\exp(\gamma))^t$ given properties of exponentials. This means that we can rewrite this as:

$$\beta^t \frac{(\widehat{c}_t)^{1-\sigma} (\exp(\gamma))^{t\frac{1-\sigma}{1-\alpha}} - 1}{1-\sigma}$$

Because this adjustment term is raised to the power t, we can group it with the discount factor. Denote the transformed discount factor with a hat:

$$\widehat{\beta} = \beta(\exp(\gamma))^{\frac{1-\sigma}{1-\alpha}}$$

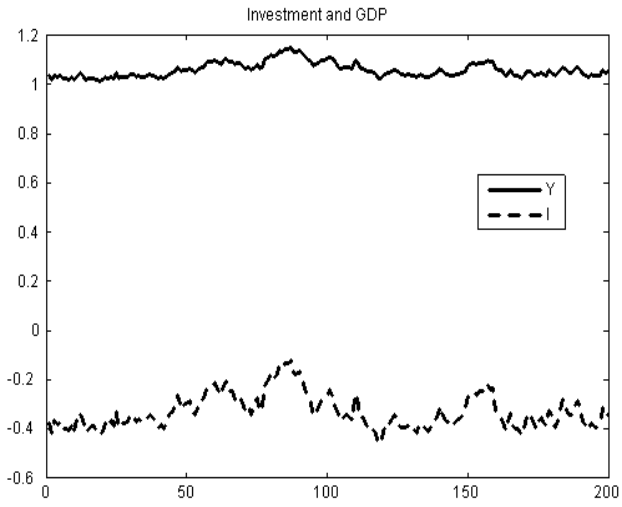
If $\sigma = 1$ (i.e. log utility), then no transformation is needed. Suppose $\gamma = 0.0033$, $\alpha = 0.33$, $\sigma = 2$, and $\beta = 0.99$. Then $\widehat{\beta} = 0.985$. Again, this is a very small adjustment, so it's often ignored completely.

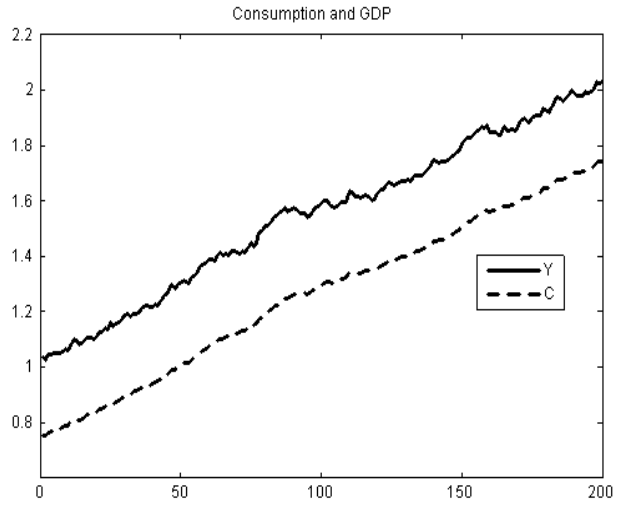
The first order conditions of transformed model can be written:

$$\begin{aligned}
\widehat{c}_t^{-\sigma} &= \widehat{\beta} E_t \widehat{c}_{t+1}^{-\sigma} (\alpha a_{t+1} \widehat{k}_{t+1}^{\alpha-1} + (1-\delta)) \\
\exp(\gamma)^{\frac{1}{1-\alpha}} \widehat{k}_t &= a_t \widehat{k}_t^{\alpha} - \widehat{c}_t + (1-\delta)\widehat{k}_t \\
\ln a_t &= \rho \ln a_{t-1} + \varepsilon_t \\
\lim_{t\to\infty} \widehat{\beta}^t \widehat{c}_t^{-\sigma} \widehat{k}_{t+1} &= 0 \\
\widehat{y}_t &= a_t \widehat{k}_{t-1}^{\alpha} \\
\widehat{i}_t &= \widehat{y}_t - \widehat{c}_t
\end{aligned}$$

The only two differences with the original model are the adjustment factor in the capital accumulation equation and the adjusted discount factor. Since these adjustments are small, you're usually fine just not worrying about them at all in the first place. There's really very little difference (for plausible parameters) in the properties of the model whether you account for the trend growth or not. As such, it's common to not worry about it.

Once you simulate the detrended model, you may want to add back in the trend growth. In my code, I interpret the underlying variables to already be logged. Thus: $\widehat{c}_t = c_t - \frac{1}{1-\alpha} \ln x_t = c_t - \frac{1}{1-\alpha}\gamma t$. Thus, to generate the levels of the series, I need to add $\frac{1}{1-\alpha}\gamma t$ to the simulated value of $\widehat{c}_t$, for example. Below are plots of the simulated series without and with adding back in trend growth:





It makes absolutely no difference for the second moment properties whether you HP filter the series with or without adding back in the trend. Note also that the scales above are log scales. If you look at investment without the trend added back in for example, its mean appears to be roughly -0.4. This is 0.67 in the levels.

4 Random Walk Technology

Suppose that we take the same basic structure of the stochastic growth model above, but instead suppose that technology follows a random walk with drift in its log. This means that shocks to technology never die out. It also means that that the trend is stochastic. In particular, suppose:

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

This process may equivalently be written:

$$\exp(\ln a_t) = \exp(\gamma + \varepsilon_t) \exp(\ln a_{t-1})$$

The production function can be written:

$$y_t = a_t k_t^{\alpha}$$

You can verify that, in a non-stohchastic setting (i.e. all shocks equal to zero at all times), this is identical to the deterministic case above.

We still need to come up with a way to render the variables of the model stationary. It's basically the same as we did before. Define the transformed variables as follows:

$$
\begin{aligned}
y_t &\equiv \frac{y_t}{a_t^{\frac{1}{1-\alpha}}} \\
k_t &\equiv \frac{k_t}{a_t^{\frac{1}{1-\alpha}}} \\
c_t &\equiv \frac{c_t}{a_t^{\frac{1}{1-\alpha}}} \\
i_t &\equiv \frac{i_t}{a_t^{\frac{1}{1-\alpha}}}
\end{aligned}
$$

Begin with the production function written in Dynare friendly notation:

$$
\begin{aligned}
y_t &= a_t k_{t-1}^{\alpha} \\
y_t &= a_t \left(\frac{k_{t-1}}{a_{t-1}^{\frac{1}{1-\alpha}}} \right)^{\alpha} a_{t-1}^{\frac{\alpha}{1-\alpha}}
\end{aligned}
$$

Using our notation:

$$
y_t = a_t \widehat{k}_{t-1}^{\alpha} a_{t-1}^{\frac{\alpha}{1-\alpha}}
$$

Now divide both sides by $a_t^{\frac{1}{1-\alpha}}$:

$$
\begin{aligned}
\frac{y_t}{a_t^{\frac{1}{1-\alpha}}} &= a_t^{1-\frac{1}{1-\alpha}} \widehat{k}_{t-1}^{\alpha} a_{t-1}^{\frac{\alpha}{1-\alpha}} \\
\widehat{y}_t &= a_t^{-\frac{\alpha}{1-\alpha}} \widehat{k}_{t-1}^{\alpha} a_{t-1}^{\frac{\alpha}{1-\alpha}} \\
\widehat{y}_t &= \left(\frac{a_t}{a_{t-1}} \right)^{-\frac{\alpha}{1-\alpha}} \widehat{k}_{t-1}^{\alpha}
\end{aligned}
$$

Look at the process for technology:

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

We know that $a_t = \exp(\ln a_t)$. We can equivalently write the process above as:

$$
\exp(\ln a_t) = \exp(\ln a_{t-1}) \exp(\gamma + \varepsilon_t)
$$

Thus, the ratio of of technology across time is:

$$
\frac{a_t}{a_{t-1}} = \frac{\exp(\ln a_t)}{\exp(\ln a_{t-1})} = \exp(\gamma + \varepsilon_t)
$$

We can plug this back into the production function:

$$\widehat{y}_t = (\exp(\gamma + \varepsilon_t))^{-\frac{\alpha}{1-\alpha}} \widehat{k}_{t-1}^{\alpha}$$

Now let's look at the capital accumulation equation, and again use Dynare's timing convention:

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

Divide both sides by the scaling factor:

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

Using our notation, this reduces to:

$$\widehat{k}_t = a_t^{-\frac{\alpha}{1-\alpha}} k_{t-1}^{\alpha} - \widehat{c}_t + (1-\delta)\frac{k_{t-1}}{a_t^{\frac{1}{1-\alpha}}}$$

Simplify a bit:

$$\widehat{k}_t = \left(\frac{a_t}{a_{t-1}}\right)^{-\frac{\alpha}{1-\alpha}} \widehat{k}_{t-1}^{\alpha} - \widehat{c}_t + (1-\delta)\frac{k_{t-1}}{a_t^{\frac{1}{1-\alpha}}}$$

From above, we know that $a_t = a_{t-1}\exp(\gamma + \varepsilon_t)$. Thus:

$$\widehat{k}_t = (\exp(\gamma + \varepsilon_t))^{-\frac{\alpha}{1-\alpha}} \widehat{k}_{t-1}^{\alpha} - \widehat{c}_t + (1-\delta)\,(\exp(\gamma + \varepsilon_t))^{-\frac{1}{1-\alpha}} \widehat{k}_{t-1}$$

Finally, we deal with preferences. Normalizing the initial level of technology to be unity (i.e. $a_0 = 1$), the expected value of future technology can be written as: $\exp(\gamma)^t$. Provided γ is close to zero, this number will be close to unity when t is small. Plugging this relationship into preferences and simplifying yields:

$$\begin{aligned}
&\sum_{t=0}^{\infty} \beta^t E_0 a_t^{\frac{1-\sigma}{1-\alpha}} \frac{\left(\frac{c_t}{a_t^{\frac{1}{1-\alpha}}}\right)^{1-\sigma} - 1}{1-\sigma} \\
&\sum_{t=0}^{\infty} \beta^t E_0 a_t^{\frac{1-\sigma}{1-\alpha}} \frac{(\widehat{c}_t)^{1-\sigma} - 1}{1-\sigma} \\
&\sum_{t=0}^{\infty} \beta^t \exp(\gamma)^{\frac{(1-\sigma)t}{1-\alpha}} \frac{(\widehat{c}_t)^{1-\sigma} - 1}{1-\sigma} \\
&\sum_{t=0}^{\infty} b^t \frac{(\widehat{c}_t)^{1-\sigma} - 1}{1-\sigma} \\
b \;&=\; \beta \exp(\gamma)^{\frac{(1-\sigma)}{1-\alpha}}
\end{aligned}$$

Effectively, including trend growth requires a (slight) downward adjustment in the discount factor, provided $\sigma > 1$. This downward adjustment will not typically be very large. Suppose $\sigma = 2$, $\alpha = 0.33$, and $\gamma = 0.003$, the adjustment factor will be 0.9955. If the discount factor you use is 0.99, the adjusted discount factor needs to be 0.9855.

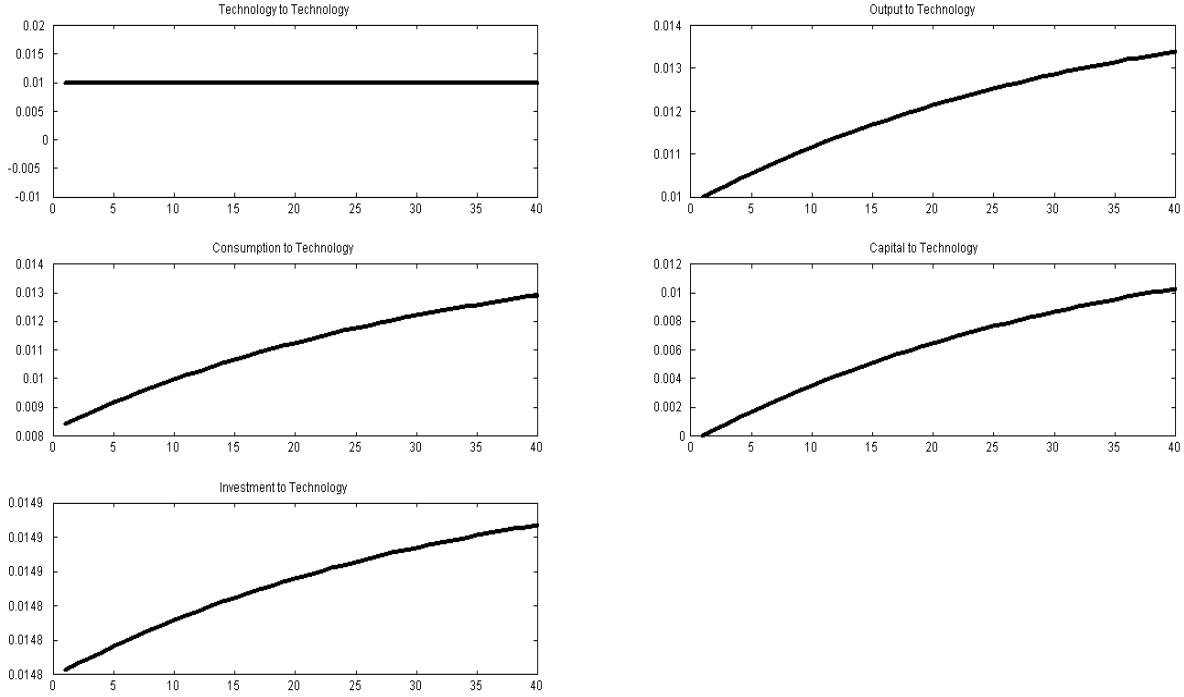
The code is similar to the deterministic case, but there are important differences. Most importantly, there will be permanent responses of the variables in the model to the shocks (although the transformed variables will not respond permanently). The model will compute impulse responses of the transformed variables; you'll want to adjust these, since the shock is to the variable by which we are dividing.

The first step will be to cumulate the effects of the shock on technology itself. Then you'll add this back into the responses of the transformed variables to form the impulse responses of the actual variables. As above, you'll need to "lag" the capital stock response by one period in order to get things right. My full code is here below:

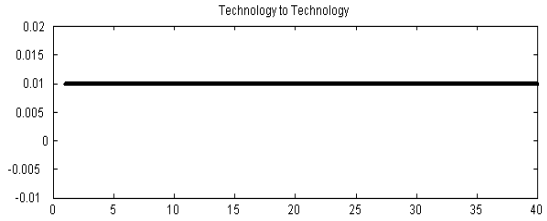
```
var y i k a c;
varexo e;
parameters alpha beta delta rho sigma b gamma sigmae;
% Calibration
alpha = 0.33;
beta = 0.99;
gamma = 0.0033;
delta = 0.025;
sigma = 2;
sigmae = 0.01;
b = beta*exp(gamma)^((1-sigma)/(1-alpha));
model;
exp(c)^(-sigma) = b*exp(c(+1))^(-sigma)*(alpha*(exp(a(+1))^(-alpha/(1-alpha)))*exp(k)^(alph
1) + (1-delta));
exp(y) = (exp(a)^(-alpha/(1-alpha)))*exp(k(-1))^(alpha);
exp(k) = (exp(a)^(-alpha/(1-alpha)))*exp(k(-1))^(alpha) - exp(c) + (1-delta)*(exp(a)^(-
1/(1-alpha)))*exp(k(-1));
a = gamma + e;
exp(i) = exp(y) - exp(c);
end;
initval;
k = log(29);
y = log(3);
a = 1;
c = log(2.5);
i = log(1.5);
end;
shocks;
var e = sigmae^2;
end;
steady;
```

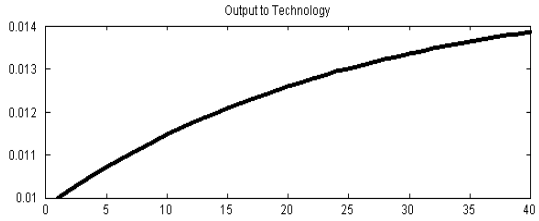
```
H = 40;
stoch_simul(hp_filter = 1600, order = 1, irf = 40);
a_e = cumsum(a_e);
c_e = c_e + a_e*(1/(1-alpha));
y_e = y_e + a_e*(1/(1-alpha));
k_e = k_e + a_e*(1/(1-alpha));
i_e = i_e + a_e*(1/(1-alpha));
k_e2(1,1) = 0;
for j = 2:H
k_e2(j,1) = k_e(j-1,1);
end
```

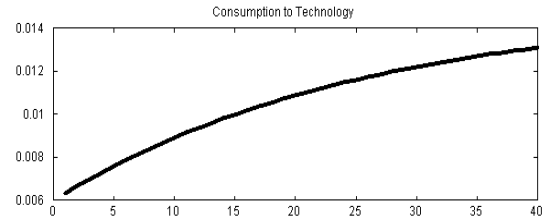
The impulse responses of the actual variables are here:

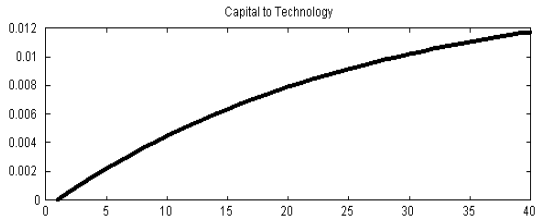


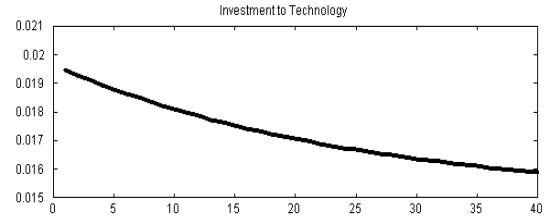
For this parameterization, investment basically just jumps up straight to the new steady state and sits there (although you have to look at the scale closely to realize that). That need not be the case, however. If I go to the case of log utility I would get:



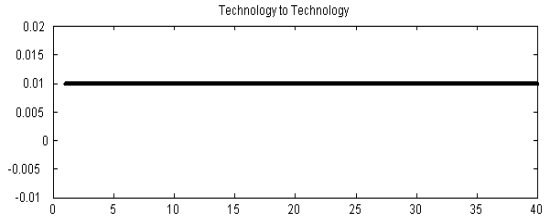


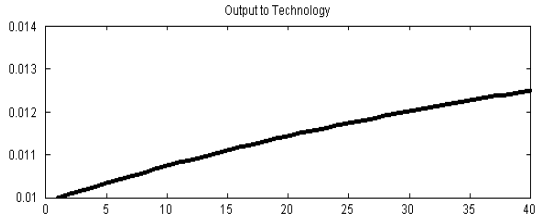


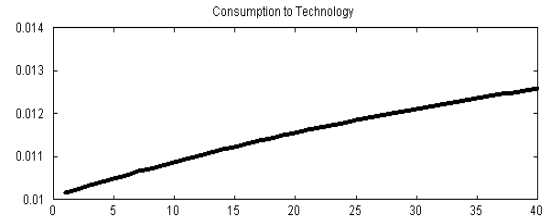


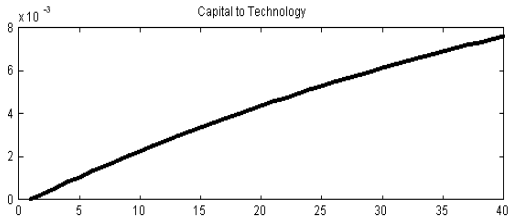


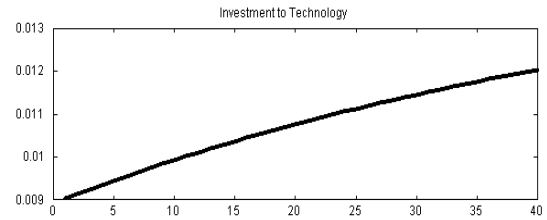
If I make the consumer very risk-averse (say, $\sigma = 8$), then I get:





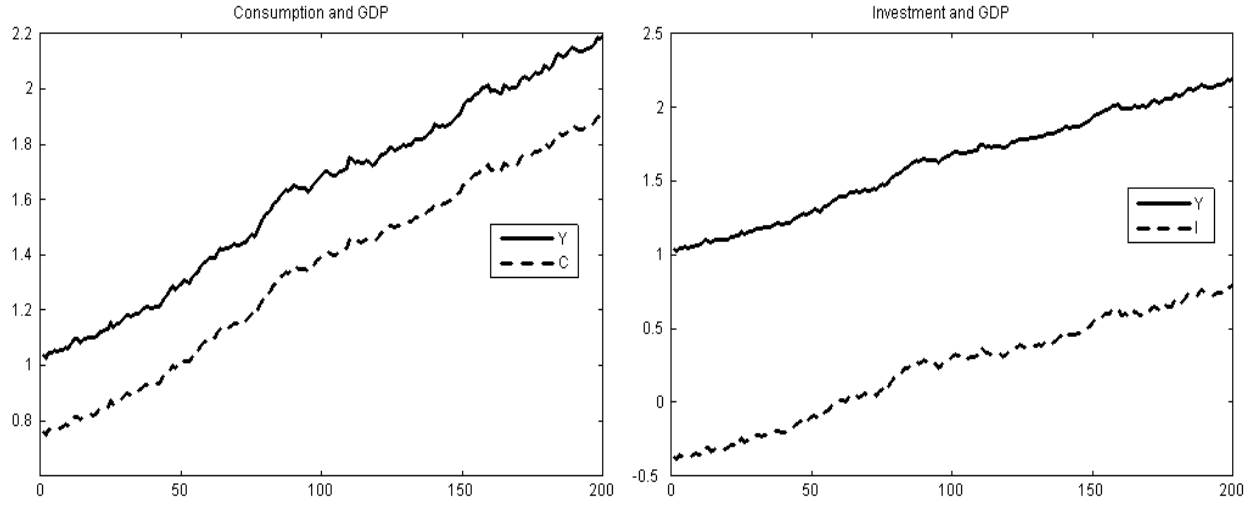






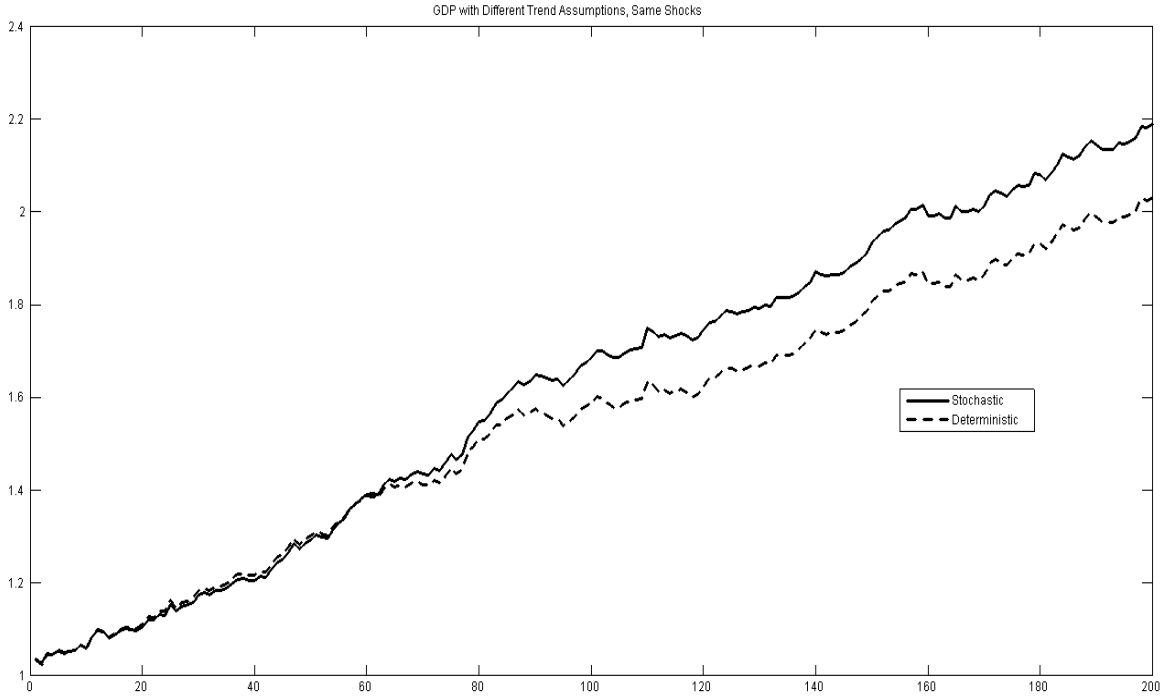
How these IRFs change (in particular how the consumption and investment responses change) as the coefficient of relative risk aversion changes is instructive. When households are very risk averse, they really want to smooth consumption, so consumption responds a lot to permanent changes in productivity. Thus, by default (since the output response on impact is unaffected by the value of σ), investment must not go up by as much, and in this case undershoots its new steady state. If households are not very risk averse (log case), then consumption doesn't jump up by much on impact (households are willing to intertemporally substitute), and so investment jumps up by a lot. At $\sigma = 2$, it just so turns out that investment goes up on impact just about all the way to its new steady state value.

As before, I can simulate my own data from this model. Note that you'll want to do the simulations on your own here; Dynare will do unconditional moments of the transformed series, which don't have an empirical counterpart, and so you'll need to simulate using the policy functions from the transformed data and then you have to add back in trend growth. Below are plots of the simulated series using the benchmark parameterization.



For the above pictures, I use the same seed as I did in the deterministic case. Note that here, unlike in the case of deterministic growth, doing unconditional moments depends on whether you add the trend back in or not. Intuitively, that's because the actual trend here is changing, whereas in the deterministic case it's not. See the discussion above.

Below is a plot of the simulated level of output in this case vs. the case with a deterministic trend.



This example serves to highlight how it's very difficult to detect a trend stationary series from stochastic stationary one. These series look very similar and there's no glaring difference between the two, though in one shocks have a permanent effect while in the other they don't.