Monetary Policy Models - PS1

Ricardo Semião

2024-11-08

Setup

This document was made using Jupyter Notebook to present the results of a Dynare/Octave code. Dynare files must be compiled, such that Dynare code can be run in independent cells. Still, I present the components of each `.mod` file here in code cells, and run the full code with the `oct2py`'s `%%octave` cell magic.

For starters, lets load the cell magic and other relevant modules.

```
from io import StringIO
from shutil import move

import pandas as pd

import re

%load_ext oct2py.ipython
```

Now, we can set some initial configurations, like suppress warnings, and set the Dynare path.

```
%%octave

warning('off', 'Octave:shadowed-function');
graphics_toolkit('gnuplot');

addpath C:\dynare\6.2\matlab
```

Lastly, this document is rendered with quarto. Using powershell, one runs:

```
!powershell quarto render ps1_main.ipynb --to pdf
```

Question 1

Item 1.

Code

First, we declare the endogenous variables, exogenous variables, and parameters of the model.

```
var y c k i h r z;
varexo e;
parameters beta psi delta theta rho;
```

Then, we can set the parameters' values, which can be calibrated from macroeconomic data. Lets start with `theta = 0.36`.

```
theta = 0.36; %capital share
beta  = 0.99; %discount factor
delta = 0.025; %depreciation rate
psi   = 1.72; %leasure utility parameter
```

```
rho   = 0.95; %AR(1) productivity parameter
sigma = (0.000010299)^(1/2); %SD of productivity shock
```

Then, we can define the model equations. In order:

1. Intertemporal consumption choice.
2. Income-leisure choice.
3. Budget constraint.
4. Production function.
5. Capital accumulation path.
6. Interest rate equation.
7. Stochastic productivity equation.

```
model;
    (1/c) = beta*(1/c(+1))*(1+theta*(k^(theta-1))*(exp(z(+1))*h(+1))^(1-theta)-delta); %(1)
    psi*c/(1-h) = (1-theta)*(k(-1)^theta)*(exp(z)^(1-theta))*(h^(-theta)); %(2)
    c+i = y;                                                                %(3)
    y = (k(-1)^theta)*(exp(z)*h)^(1-theta);                                 %(4)
    i = k-(1-delta)*k(-1);                                                  %(5)
    r = theta*y/k;                                                          %(6)
    z = rho*z(-1)+e;                                                        %(7)
end;
```

Then, we can define the initial position of the economy variables. Usually, a good start is the steady state of the model.

```
initval;
    y = 1.2353;
    k = 12.6695;
    c = 0.9186;
    h = 0.33;
    i = 0.316738;
    z = 0;
    e = 0;
    r = 0.0351;
end;
```

Finally, we define the shocks. In this case, we only have the productivity one.

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

Now, we can calculate the steady state, and simulate IRFs. We save them as `.eps`, which can be presented below. We want the graphics as `.eps`, and only to be saved on disk, not displayed.

```
steady;
check;

stoch_simul(hp_filter = 1600, order = 1, nodisplay, graph_format = PDF) y c i;
```

Results

Now, we can run the file that contains all the code above. Then, move the pictures to the correct location.

```
%%octave

cd dynare_scripts/rbc/
dynare rbc
```

```
move(
    'dynare_scripts/rbc/rbc/graphs/rbc_IRF_e.eps',
```

```
    'figures/rbc_IRF_e_033.eps'
)
```

All the outputs that Dynare generates are important to understand if the model is making sense. But here, we are more interested in the IRFs. I'll present them only in the Item 3. section.

Item 2.

We could create a completely new `.mod` file, and change only the `theta` value. But, I'd rather do a more programmatic approach. I will use the `re` module to find the `theta = 0.36` line in the existing file, and replace it with `theta = 0.5`.

```
rbc_path = 'dynare_scripts/rbc/rbc.mod'

with open(rbc_path, 'r', encoding='utf-8') as file:
    rbc_lines = file.readlines()

rbc_lines = [
    re.sub(r'theta   = [0-9.]+', 'theta   = 0.50', x)
    for x in mod_lines
]

with open(rbc_path, 'w', encoding='utf-8') as file_out:
    file_out.writelines(rbc_lines)
```

Now we can run the file again.

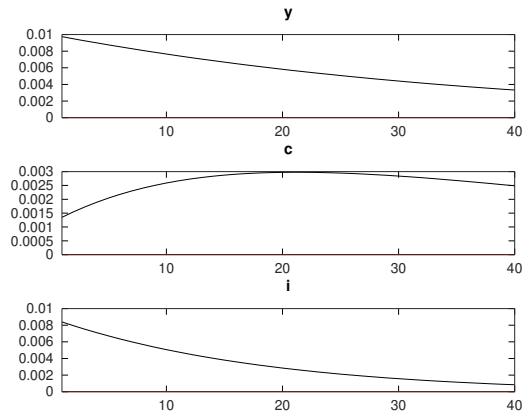
```
%%octave

dynare rbc
```

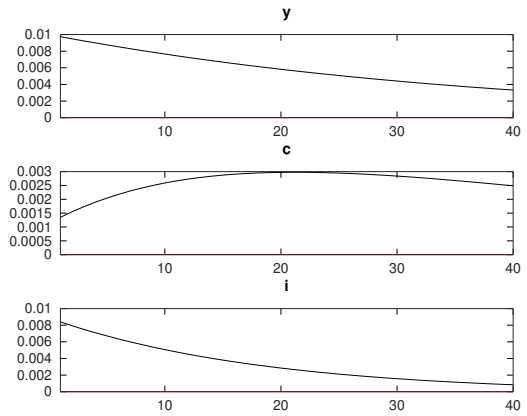
```
move(
    'dynare_scripts/rbc/rbc/graphs/rbc_IRF_e.eps',
    'figures/rbc_IRF_e_050.eps'
)
```

Now, we can compare the IRFs of the two models in the Item 3. section.

Item 3.

The IRFs are presented below.





We can see that all the shocks' effects are larger in the $\theta = 0.5$ case (see the y-axis values). We know that θ is associated with the capital share in the production function. A higher θ yields a more balanced share, which means that the economy is more sensitive to productivity shocks. For example, with higher

productivity, the marginal productivity of capital increases, which implies that the optimal level of investment is higher.

Question 2

Item 1.

No, since the second model remove tendencies in productivity, such that the main endogenous variables x_t are stationary. On the other hand, the original model makes no such transformation, and the X_t variables are always growing (in the long run) given the technology shock.

Item 2.

Now, the transformation is, in some sense, "more constant". The variables are written in terms of the steady state, then applied to a monotonic transformation (log).

Thus, the variables are different, and will have different trajectories (in the level), which can be seen as the null steady-state by construction on the model 3 case. But, again, it is a "more constant" transformation, the dynamics are still the same.

Item 3.

First, lets adjust the variables by the productivity:

$$1 = \beta E\left[(\frac{c_{t+1}}{c_t})^{-\tau}\frac{A_t}{A_{t+1}}\frac{R_t}{\pi_{t+1}}\right]$$

We need to deal with the term $\frac{A_t}{A_{t+1}}$. Lets use equation (8):

$$\ln A_t = \ln\gamma + \ln A_{t-1} + \ln z_t$$

$$e^{\ln A_t} = e^{(}\gamma \ln A_{t-1} \ln z_t)$$

$$A_{t+1} = \gamma A_t z_{t+1}$$

$$\frac{A_t}{A_{t+1}} = \frac{1}{\gamma z_{t+1}}$$

Then, we get equation (12):

$$1 = \beta E\left[(\frac{c_{t+1}}{c_t})^{-\tau}\frac{1}{\gamma z_{t+1}}\frac{R_t}{\pi_{t+1}}\right]$$

Item 4.

Code

Let's first define the model equations. In order:

1. Equation (12): Modified Euler condition
2. Equation (13): Calvo equation for inflation
3. Equation (14): Resource constraint
4. Equation (15): Taylor rule for interest rate
5. Equation (16): Natural interest rate
6. Equation (17): Stochastic process for productivity
7. Equation (18): Stochastic process for public spending

8. Equation (19): Adjusted natural output

```
model;
    1 = beta * ( (c(+1)/c)^(-tau) ) * (1 / (gamma * exp(z(+1)))) * (R / pi(+1));      %(1)
    1 = phi * (pi - pi_star) * (1 - (1 / (2 * nu)) * pi + (pi_star / (2 * nu)))
      - phi * beta * ( (c(+1)/c)^(-tau) ) * (y(+1) / y) * (pi(+1) - pi_star) * pi(+1)
      + (1 / nu) * (1 - c^tau);                                                       %(2)
    y = c + ((g - 1) / g) * y + (phi / 2) * (pi - pi_star)^2 * y;                     %(3)
    R = R_star^(1 - rho_R) * R(-1)^rho_R * exp(eR);                                   %(4)
    R_star = r * pi_star * (pi / pi_star)^psi_1 * (y / y_star)^psi_2;                 %(5)
    z = rho_z * z(-1) + ez;                                                           %(6)
     g = (1 - rho_g) * g_bar + rho_g * g(-1) + eg;                                    %(7)
    y_star = g * c;                                                                   %(8)
end;
```

And then, the steady state model:

```
steady_state_model;
    c       = (1 - nu)^(1 / tau);
    y_star  = g_star * c;
    y       = y_star;
    pi      = pi_star;
    R       = r * pi_star;
    R_star  = R;
    g       = g_star;
    z       = 0;
end;
```

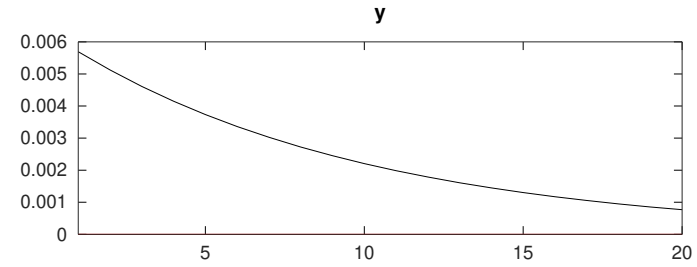
Results

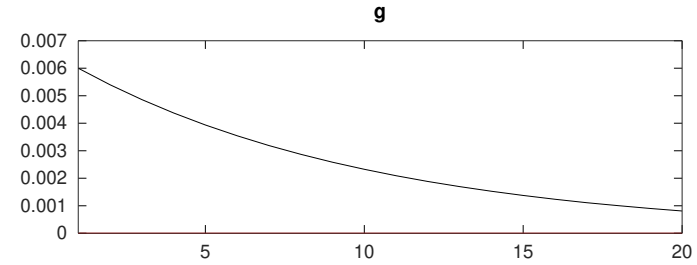
Lets run the file, and move the results to the correct folder.

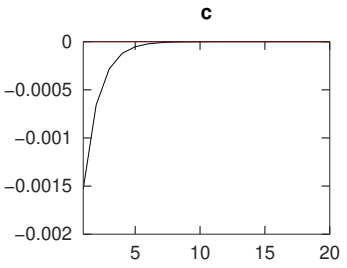
```
%%octave

cd ../as1/
dynare as1
```

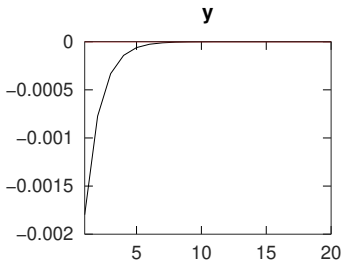
```
for e in ['eg', 'eR', 'ez']:
    move(
        f'dynare_scripts/as1/as1/graphs/as1_IRF_{e}.eps',
        f'figures/as1_IRF_{e}.eps'
    )
```

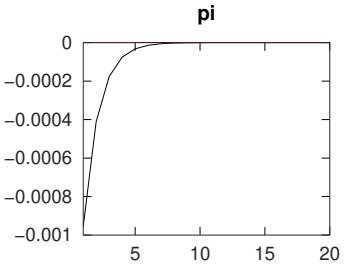
The IRFs are presented below.

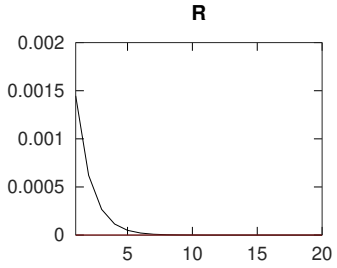

y
0.006
0.005
0.004
0.003
0.002
0.001
0
5
10
15
20

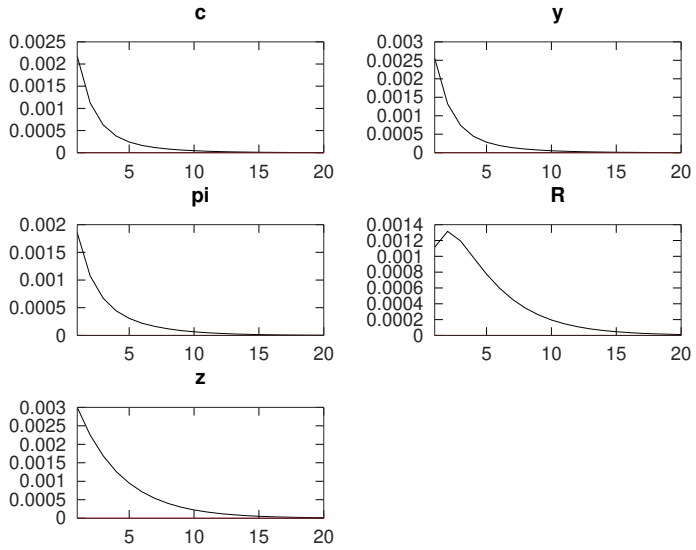

g
0.007
0.006
0.005
0.004
0.003
0.002
0.001
0
5
10
15
20


c
0
−0.0005
−0.001
−0.0015
−0.002
5
10
15
20


y
0
−0.0005
−0.001
−0.0015
−0.002
5
10
15
20


pi
0
−0.0002
−0.0004
−0.0006
−0.0008
−0.001
5
10
15
20


R
0.002
0.0015
0.001
0.0005
0
5
10
15
20



Item 5.

By definition, the steady state π^* is the solution to $\pi_{t+1} = \pi_t \ \forall t$, and when there is no shocks present, that is, from (17), $z = 1$.

Moving forward, transforming the endogenous variables on equation (12) in terms of the steady state, we find:

$$1 = \beta \left[1^{-\tau} \frac{1}{\gamma} \frac{R}{\pi^*} \right]$$

$$1 = \frac{1}{\gamma} \frac{\beta R}{\pi^*}$$

$$R = \frac{\pi^* \gamma}{\beta}$$

We can further simplify this equation using the relation from γ and β from (21), and we get the equation (22):

$$R = \frac{\pi^* \gamma}{\frac{r}{\pi^*}} = r\pi^*$$

Item 6.

We can transform $\hat{x}_t = \ln(x_t/x)$ the variables endogenous in equation (12):

$$1 = \beta E \left[(\frac{c_{t+1}}{c_t})^{-\tau} \frac{1}{\gamma z_{t+1}} \frac{R_t}{\pi_{t+1}} \right]$$

Lets also use the transformation $R = (\gamma \pi^*/\beta)$

$$1 = E \left[e^{r\hat{e}_{t+1} - \tau \hat{c}_t - \hat{z}_t + \hat{R}_t - \pi_{t+1}} \right]$$

Item 7.

First, lets run the file, and move the results to the correct folder.

```
%%octave

cd ../as2/
dynare as2
```

```
for e in ['eg', 'eR', 'ez']:
    os.rename(
        f'dynare_scripts/as2/as2/graphs/as1_IRF_{e}.eps',
        f'figures/as2_IRF_{e}.eps'
    )
```

Now, lets get the averages from the `.log` files. Below there is a function that does it by finding a match of `'THEORETICAL MOMENTS'` in the file, and parsing it as a `pandas.DataFrame`.

```
def read_log_moments(path, n_vars):
    with open(path, 'r', encoding='utf-8') as file:
        lines = file.readlines()

    table_index = -1
    while not re.search(r'THEORETICAL MOMENTS', lines[table_index]):
        table_index += 1

    table_index += 2
    table = ''.join(lines[(table_index):(table_index + n_vars)])

    df = pd.read_csv(
        StringIO(table),
        sep = r'\s+',
        header = None,
        names = ['Variable', 'Mean', 'SE', None],
        usecols = ['Variable', 'Mean', 'SE']
    )

    return df
```

Now, we can apply the function to each file, and get the result as a merged dataframe.

```
n_vars = 6

moments = (
    read_log_moments(path, n_vars)
    for path in ['dynare_scripts/as1/as1.log', 'dynare_scripts/as2/as2.log']
)

pd.merge(*moments, on = 'Variable', suffixes = (' (as1)', ' (as2)'))
```

	Variable	Mean (as1)	SE (as1)	Mean (as2)	SE (as2)
0	c	0.9487	0.0031	0.0	0.0029
1	y	1.1161	0.0136	0.0	0.0141
2	pi	1.0080	0.0026	0.0	0.0022
3	R	1.0105	0.0031	0.0	0.0030
4	z	0.0000	0.0045	0.0	0.0045
5	g	1.1765	0.0138	0.0	0.0138

As expected, all the means in `as2` are zero, by construction of the transformation based on the relation to the steady state. There shouldn't be deviations, specially given the linear approximation. But,

importantly, as said before, the dynamics are the same, such that the SEs are still the same.