ECON 6130 - Problem Set # 3

Julien Manuel Neves

December 3, 2017

Part (1)

We use the following FRED series for our data:

(i) Consumer Price Index of All Items in United States (USACPIALLQINMEI)

(ii) Real Gross Domestic Product (GDPC1)

(iii) Immediate Rates: Less than 24 Hours: Federal Funds Rate for the United States (IRSTFR01USQ156N)

(iv) Unemployment Rate: Aged 15-64: All Persons for the United States (LRUN64TTUSQ156S)

The data are log-linearized the series and detrend with the Hodrick–Prescott filter.

It is important to note that the Consumer Price Index represents the price level. Therefore, after taking the log of the CPI, it is important to differentiate the series before detrending them since

$$\pi_t = \ln(P_t) - \ln(P_{t-1})$$

where P_t is the price level (CPI) at time t.

Moreover, for a measure of labor inputs, we need the following

$$n_t = \ln\left(1 - \frac{U_t}{100}\right)$$

where U_t is the unemployment rate in at time t in per cent.

The detrended series are plotted in Fig.1.

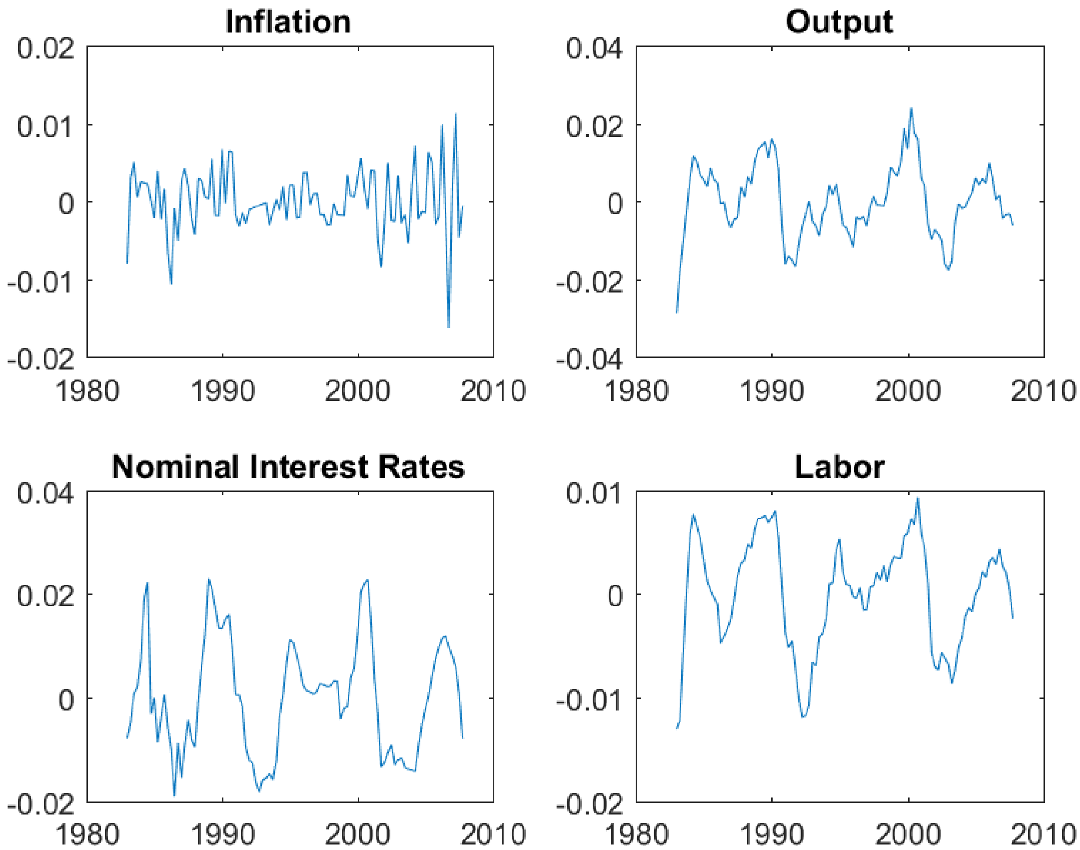

Figure 1: FRED Detrended Data

Part (2)

Note that the equations for the New Keynesian business cycle can be listed as follows

$$
\begin{aligned}
\kappa\tilde{y}_t - \kappa x_t - u_t &= 0 \\
\pi_t - \beta\mathbb{E}_t(\pi_{t+1}) - \kappa x_t - u_t &= 0 \\
x_t - \mathbb{E}_t(x_{t+1}) + \frac{1}{\sigma}[(i_t - \rho) - \mathbb{E}_t(\pi_{t+1}) - (r_t^e - \rho)] &= 0 \\
(r_t^e - \rho) + \sigma(1-\rho_a)\psi_{ya}a_t - (1-\rho_z)z_t &= 0 \\
(i_t - \rho) - \phi_\pi\pi_t - \phi_y\tilde{y}_t - \phi_y\psi_{ya}a_t - v_t &= 0 \\
v_t &= \rho_v v_{t-1} + \epsilon_t^v \\
a_t &= \rho_a a_{t-1} + \epsilon_t^a \\
z_t &= \rho_z z_{t-1} + \epsilon_t^z \\
u_t &= \rho_u u_{t-1} + \epsilon_t^u \\
x_t &= \mathbb{E}_{t-1}(x_t) + \eta_t^{x_t} \\
\pi_t &= \mathbb{E}_{t-1}(\pi_t) + \eta_t^{\tilde{n}}
\end{aligned}
$$

where η_t are the forecast errors.

Thus, we define the following states for our model

$$S_t = [\tilde{y}_t, x_t, \pi_t, r_t^e - \rho, i_t - \rho, v_t, a_t, z_t, u_t, \mathbb{E}_t(x_{t+1}), \mathbb{E}_t(\pi_{t+1})]$$

This implies the following parameters are need to be estimated/calibrated in order solve the model

$$\Theta = \left[\sigma, \beta, \phi, \epsilon, \phi_\pi, \phi_y, \theta, \alpha, \rho_v, \rho_a, \rho_z, \rho_u, \sigma_v, \sigma_a, \sigma_z, \sigma_u\right]$$

where the different parameters are defined as in Gali's book. In total, there is 16 parameters in our model.

Part (3)

Note that our code is based on gensys.m which solves the following linear equation

$$\Gamma_0 S_t = \Gamma_1 S_{t-1} + B + \Psi\epsilon_t + \Pi\eta_t$$

Therefore, by rewriting the set of equations for the New Keynesian business cycle in matrix form, we get

$$\Gamma_0 = \begin{bmatrix} \kappa & -\kappa & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & -\kappa & 1 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & -\beta \\ 0 & 1 & 0 & -\frac{1}{\sigma} & \frac{1}{\sigma} & 0 & 0 & 0 & 0 & -1 & -\frac{1}{\sigma} \\ -\phi_y & 0 & -\phi_\pi & 0 & 1 & -1 & -\phi_y\psi_{ya} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & \sigma(1-\rho_a)\psi_{ya} & -(1-\rho_z) & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\Gamma_1 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \rho_v & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \rho_a & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \rho_z & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \rho_u & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\Psi = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix}'$$

$$\Pi = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}'$$

$$B = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}'$$

Using theses matrices, gensys.m will ouput the coefficients of an $VAR(1)$ process of the following form

$$S_t = AS_{t-1} + C\epsilon_t$$

where $\epsilon_t \sim N(0, I)$.

Finally, to represents our model in state space form, we also need an equation that links the states to our observations. Since we have four exogenous shocks and four variables, we can write inflation, output, nominal interest rates, and labor in the following matrix form

$$Z(t) = DX(t)$$

where

$$D = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & \psi_{ya} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ \frac{1}{1-\alpha} & 0 & 0 & 0 & 0 & 0 & -\frac{1-\psi_{ya}}{1-\alpha} & 0 & 0 & 0 & 0 \end{bmatrix}$$

See the appendix for the code.

Part (4)

Note that the likelihood function for a state space model is given by

$$L(\Theta \mid Z) = -\frac{pT}{2}\ln(2\pi) - \frac{1}{2}\sum_{t=1}^{T}\left[\log(|\Omega_t|) + \tilde{Z}_t'\Omega_t^{-1}\tilde{Z}_t\right]$$

where p is the number of observations and

$$\begin{aligned} \tilde{Z}_t &= Z_t - DAX_{t-1|t-1} \\ X_{t|t} &= AX_{t-1|t-1} + K_t\tilde{Z}_t \\ \Omega_t &= DP_{t|t-1}D' + \Sigma_{vv} \end{aligned}$$

See the appendix for the code.

Part (5)

The results for the maximum likelihood estimate of Θ are given in Tab.1.

	$\hat{\Theta}_{MLE}$
σ	2.4596
β	0.81704
ϕ	3.0245
ϵ	5.2968
ϕ_π	1.5695
ϕ_y	0.64808
θ	0.65917
α	0.17379
ρ_v	0.55141
ρ_a	0.80006
ρ_z	0.62
ρ_u	0.38658
σ_v	0.0085544
σ_a	0.0038708
σ_z	0.12524
σ_u	0.022965

Table 1: Simulated annealing estimation of New Keynesian model

Note that for Tab.1, we use the simulated code implemented in MATLAB. The results from the simulated annealing code used in class are reported in table Tab.2.

While in both cases the impulses responses follow the same shape, the estimate are completely different. In fact, using the code shown in class, our estimates $\hat{\Theta}_{MLE}$ hits the upper bound that we set arbitrarily for β and ϕ_π. This is usually a bad sign and as such we discard these results. I would investigate this situation more deeply if I had more time.

	$\hat{\Theta}_{MLE}$
σ	0.35582
β	1
ϕ	3.3888
ϵ	6.5416
ϕ_π	5
ϕ_y	0.19053
θ	0.72868
α	0.23639
ρ_v	0.95112
ρ_a	0.83334
ρ_z	0.88805
ρ_u	0.60757
σ_v	0.026778
σ_a	0.0039169
σ_z	0.052927
σ_u	0.0071002

Table 2: Simulated annealing estimation of New Keynesian model (simannb.m)

Part (6)

The impulse responses are plotted in Fig.2, Fig.3, Fig.4 and Fig.5.

Note that the impulse response are plotted with respect to shocks of one unit in lieu of one standard deviation, i.e. we do not use C for the impulse responses, but instead C scaled down with the respective standard deviations of the shocks.

Moreover, for the estimation of output gap, we need the following measurement matrix in our state space model

$$D = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & \psi_{ya} & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ \frac{1}{1-\alpha} & 0 & 0 & 0 & 0 & 0 & -\frac{1-\psi_{ya}}{1-\alpha} & 0 & 0 & 0 & 0 \end{bmatrix}$$

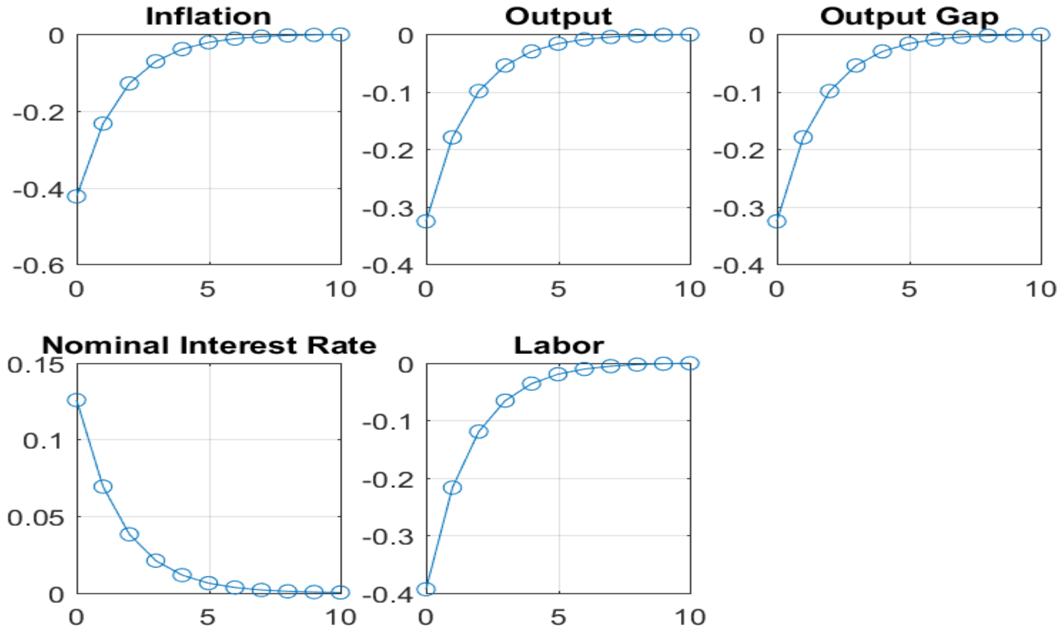

Figure 2: Impulse response to monetary shocks

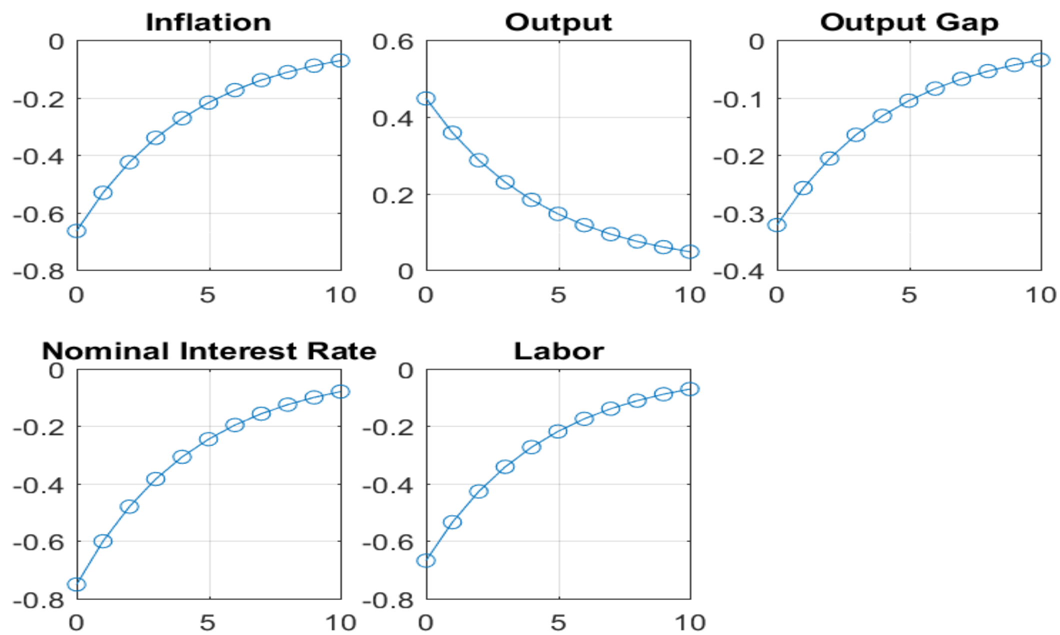

Figure 3: Impulse response to productivity shocks

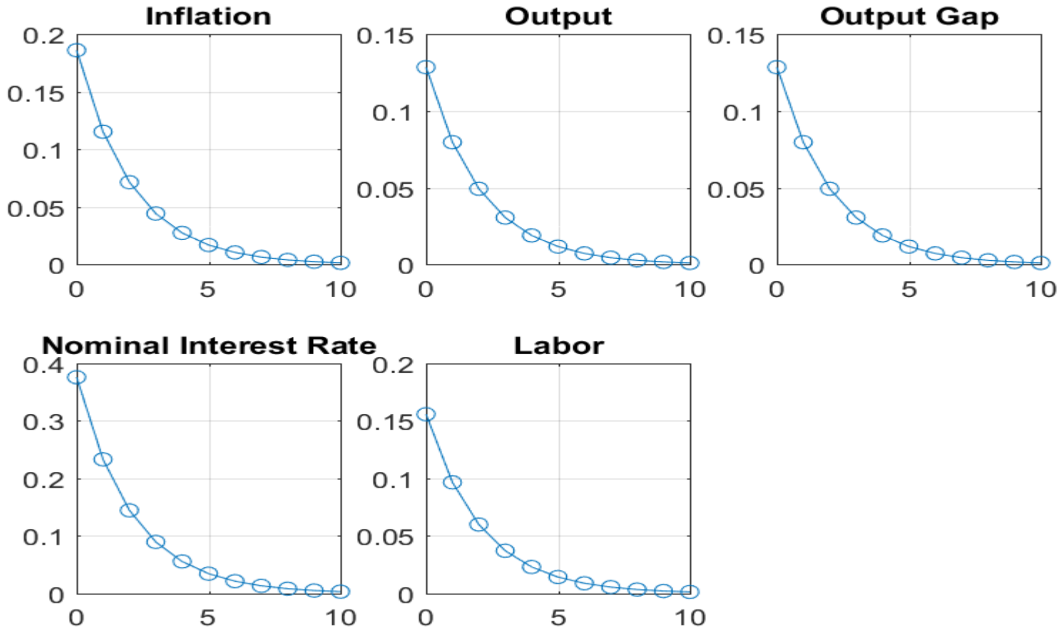

Figure 4: Impulse response to demand shocks

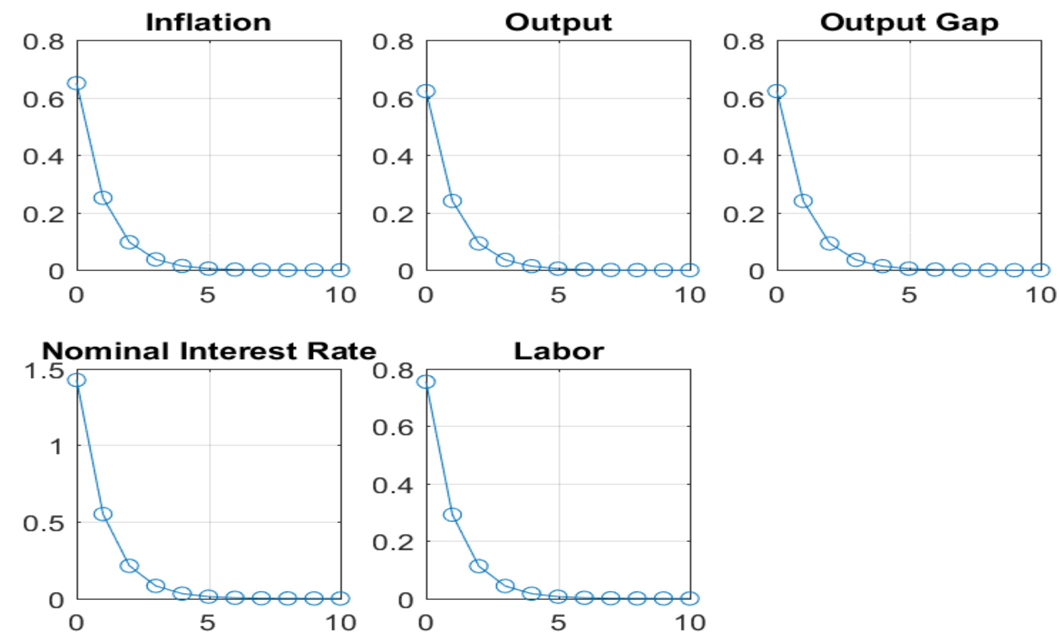

Figure 5: Impulse response to cost-push shocks

Part (7)

Recall that we can write our model in the following state space form

$$S_t = AS_{t-1} + C\epsilon_t$$
$$Z(t) = DX(t)$$

where A and C are defined as previously, $\epsilon_t \sim N(0, I)$, and

$$D = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & \psi_{ya} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ \frac{1}{1-\alpha} & 0 & 0 & 0 & 0 & 0 & -\frac{1-\psi_{ya}}{1-\alpha} & 0 & 0 & 0 & 0 \end{bmatrix}$$

Fig.6 reports both the ouput and the estimated output gap.

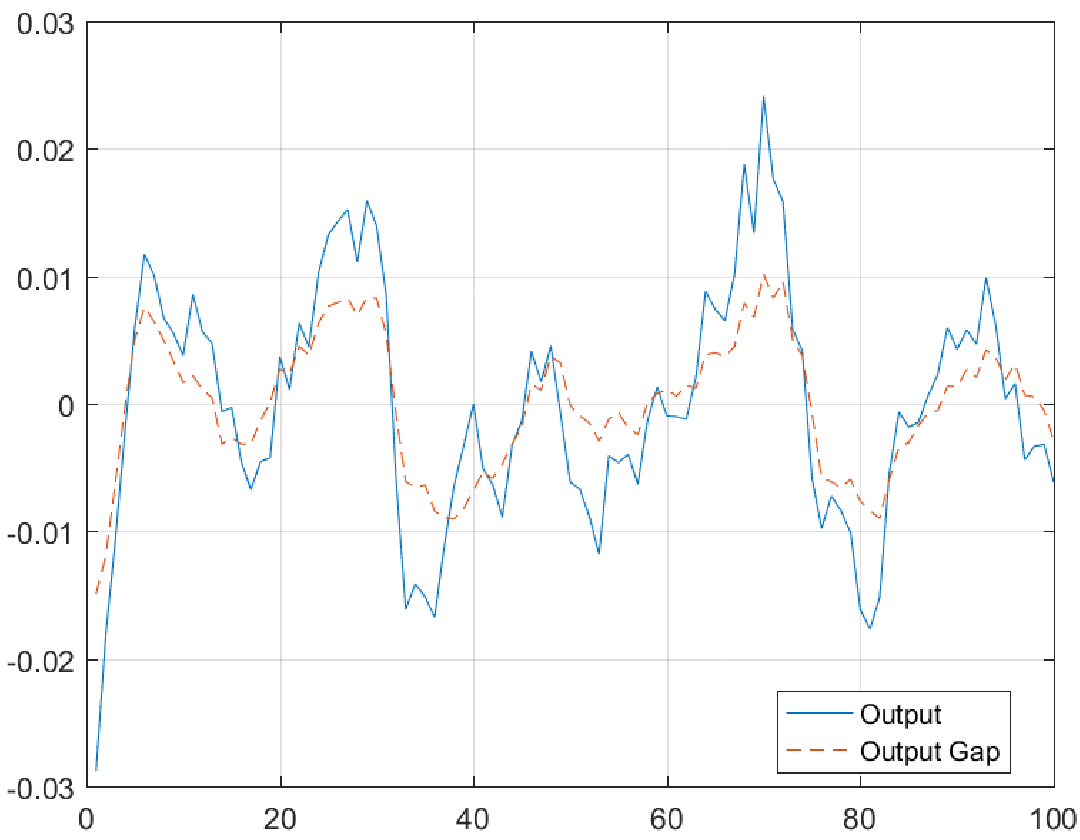

Figure 6: Kalman Filter estimate of output and output gap

Code

```
%% New Keynesian Model - Simulated Annealing
% Based off Tilahun Emiru's and Kris Nimark's codes.
% By Julien Neves

%% Housekeeping
close all;
clear all;
warning off all;

%% Part (1)
global Z
% Set handle for FRED data
url = 'https://fred.stlouisfed.org/';
c = fred(url);

% Set dates for sample period
startdate = '01/01/1983';
enddate = '12/01/2007';

CPI = fetch(c,'USACPIALLQINMEI','09/01/1982',enddate); % fetch CPI
    from FRED
GDP = fetch(c,'GDPC1',startdate,enddate); % fetch GDP from FRED
INT = fetch(c,'IRSTFR01USQ156N',startdate,enddate); % fetch rate
    from FRED
UNR = fetch(c,'LRUN64TTUSQ156S',startdate,enddate); % fetch rate
    from FRED

pi_data = diff(log(CPI.Data(:,2))); % log[price(t)] - log[price(t
    -1)]
gdp_data = log(GDP.Data(:,2)); % log of GDP
i_data = log(1+INT.Data(:,2)/100); % log of nominal interest rate
n_data = log(1-UNR.Data(:,2)/100); % log of employment rate

[~, pi_data] = hpfilter(pi_data,1600); % extract cyclical
    component of pi
[~,gdp_data] = hpfilter(gdp_data,1600); % extract cyclical
    component of y
[~,i_data] = hpfilter(i_data,1600); % extract cyclical component
    of i
[~,n_data] = hpfilter(n_data,1600); % extract cyclical component
    of n

% Combine data
Z = [pi_data';gdp_data'; i_data'; n_data'];

```

```
% Plot FRED detrended data
figure(5)
subplot(2,2,1); plot(INT.Data(:,1),Z(1,:)); datetick('x','yyyy');
    title('Inflation');
subplot(2,2,2); plot(INT.Data(:,1),Z(2,:)); datetick('x','yyyy');
    title('Output');
subplot(2,2,3); plot(INT.Data(:,1),Z(3,:)); datetick('x','yyyy');
    title('Nominal Interest Rates');
subplot(2,2,4); plot(INT.Data(:,1),Z(4,:)); datetick('x','yyyy');
    title('Labor');
saveas(gcf, 'data.png')

%% Part (2)
% Calibration
sigma     = 2; % CRRA parameter.
beta      = 0.99; % discount factor
phi       = 3; % inverse of elastiicity of labor supply
eps       = 5; % elasticity of substitution between goods i and j
phi_pi    = 1.5; % taylor rule parameter
phi_y     = 0.5; % taylor rule parameter
theta     = 0.75; % degree of price stickiness
alpha     = 0.3; % production function parameter
rho_v     = 0.5; % persistence parameter
rho_a     = 0.8; % persistence parameter
rho_z     = 0.7; % persistence parameter
rho_u     = 0.5; % persistence parameter
sigma_v   = 0.01; % standard deviation
sigma_a   = 0.008; % standard deviation
sigma_z   = 0.03; % standard deviation
sigma_u   = 0.01; % standard deviation

%% Part (5)
% Set starting value
THETA = [sigma; beta; phi; eps; phi_pi; phi_y; theta; alpha;
    rho_v; rho_a; rho_z; rho_u; sigma_v; sigma_a; sigma_z; sigma_u
        ];

LB= [0  0 1  1  1 0 0 0 0 0 0 0 0 0 0 0]'; % lower bound
UB= [10 1 10 25 5 5 1 1 1 1 1 1 10 10 10 10]'; % upper bound

% Simulated Annealing - Matlab
options = optimoptions(@simulannealbnd, 'Display', 'diagnose','
    PlotFcn', @saplotx, 'MaxTime', 600, 'FunctionTolerance', 1e-2,
    'MaxStallIterations',1000);
logL = @loglikelihood_DSGE;
```

```
xhat = simulannealbnd(logL, THETA, LB, UB, options);

% % Simulated Annealing - Class
% sa_t= 5;
% sa_rt=.3;
% sa_nt=5;
% sa_ns=5;
% xhat = simannb( 'loglikelihood_DSGE', THETA, LB, UB, sa_t, sa_rt, sa_nt, sa_ns, 1);

thetalabel = ['sigma    '; 'beta     '; 'phi      '; 'eps      '; 'phi_pi   '; 'phi_y    '; 'theta    '; 'alpha    ';
    'rho_v    '; 'rho_a    '; 'rho_z    '; 'rho_u    '; 'sigma_v  '; 'sigma_a  '; 'sigma_z  '; 'sigma_u  '];
disp('ML estimate of THETA')
disp([thetalabel, num2str(xhat)])

%% Part (6)

% Compute Neq Keynesian Model with estimated theta
[A, C, D ,~, T] = nkbc_model(xhat, 'impulse');

% Compute impulse responses
time = 0:10;   % set time horizon
col = T;   % start impulse matrix
for j=1:length(time)
    resp(:,:,j)=D*col; % compute observations
    col=A*col;  % compute next period states
end

for i = 1:4
    % Extract impulse responses for observations
    resp_pi(:,i)=squeeze(resp(1,i,:));
    resp_y(:,i)=squeeze(resp(2,i,:));
    resp_yg(:,i)=squeeze(resp(3,i,:));
    resp_i(:,i)=squeeze(resp(4,i,:));
    resp_n(:,i)=squeeze(resp(5,i,:));

    % Plot Impulse Responses
    figure(i)
    subplot(2,3,1); plot(time,resp_pi(:,i),'--O'); title('Inflation'); grid on;
    subplot(2,3,2); plot(time,resp_y(:,i),'--O'); title('Output'); grid on;
    subplot(2,3,3); plot(time,resp_yg(:,i),'--O'); title('Output
```

```
    Gap'); grid on;
    subplot(2,3,4); plot(time,resp_i(:,i),'-O'); title('Nominal
    Interest Rate'); grid on;
    subplot(2,3,5); plot(time,resp_n(:,i),'-O'); title('Labor');
    grid on;
end

% Print Impulse Responses - Monetary Shock
figure(1)
saveas(gcf, 'impulse_monetary.png')
% Print Impulse Responses - Productivity Shock
figure(2)
saveas(gcf, 'impulse_prod.png')
% Print Impulse Responses - Demand Shock
figure(3)
saveas(gcf, 'impulse_demand.png')
% Print Impulse Responses - Cost-push Shock
figure(4)
saveas(gcf, 'impulse_cost.png')

%% Part (7)
% Solve New Keynesian Model
[A,C,D] = nkbc_model( xhat, 'data');

% Set starting values
X0 = zeros(size(A,2),1);    % set starting X
P0 = dlyap(A,C*C'); % set starting for the variance

% Compute the Kalman Filter
[ X_post, P_post, X_prior, Z_tilde, Omega] = kfilter(Z, A, C, D,
  0, X0, P0 );

figure(6)
plot(Z(2,:));
hold
plot(X_post(1,:),'--');
legend('Output','Output Gap', 'Location', 'best'); grid on;
saveas(gcf, 'kalman.png')
```

New Keynesian Model (nkbc_model.m)

```
function [A, C, D, eu, R] = nkbc_model( THETA, type)
%nkbc_model New Keynesian Model with cost-push shocks
%   State space model:
%   X(t) = A*X(t-1) + C*eps(t)
```

```
%   Z(t) = D*X(t)
%   By Julien Neves

%% Matrix A and C
% Calibration
sigma      = THETA(1); % CRRA parameter.
beta       = THETA(2); % discount factor
phi        = THETA(3); % inverse of elastiicity of labor supply
eps        = THETA(4); % elasticity of substitution between goods i
    and j
phi_pi     = THETA(5); % taylor rule parameter
phi_y      = THETA(6); % taylor rule parameter
theta      = THETA(7); % degree of price stickiness
alpha      = THETA(8); % production function parameter
rho_v      = THETA(9); % persistence parameter
rho_a      = THETA(10); % persistence parameter
rho_z      = THETA(11); % persistence parameter
rho_u      = THETA(12); % persistence parameter
sigma_v    = THETA(13); % standard deviation
sigma_a    = THETA(14); % standard deviation
sigma_z    = THETA(15); % standard deviation
sigma_u    = THETA(16); % standard deviation

% Compute the coefficients
rho = -log(beta);
lambda = (1-theta)*(1-beta*theta)*(1-alpha)/(theta*(1-alpha+alpha*
    eps));
kappa = lambda*(sigma + (phi+alpha)/(1-alpha));
psi_ya = (1+phi)/(sigma*(1-alpha)+phi+alpha);

% State: 'y','x'; 'pi'; 'r^e'; 'i'; 'v'; 'a'; 'z'; 'u'; 'E(x)'; 'E
    (pi)'
Gamma0 = [kappa -kappa 0 0 0 0 0 0 -1 0 0;
    0 -kappa 1 0 0 0 0 0 -1 0 -beta;
    0 1 0 -1/sigma 1/sigma 0 0 0 0 -1 -1/sigma;
    -phi_y 0 -phi_pi 0 1 -1 -phi_y*psi_ya 0 0 0 0;
    0 0 0 1 0 0 sigma*(1-rho_a)*psi_ya -(1-rho_z) 0 0 0;
    0 0 0 0 0 1 0 0 0 0 0;
    0 0 0 0 0 0 1 0 0 0 0;
    0 0 0 0 0 0 0 1 0 0 0;
    0 0 0 0 0 0 0 0 1 0 0;
    0 1 0 0 0 0 0 0 0 0 0;
    0 0 1 0 0 0 0 0 0 0 0];

Gamma1 = [0 0 0 0 0 0 0 0 0 0 0;
```

```
           0 0 0 0 0 0 0 0 0 0 0;
           0 0 0 0 0 0 0 0 0 0 0;
           0 0 0 0 0 0 0 0 0 0 0;
           0 0 0 0 0 0 0 0 0 0 0;
           0 0 0 0 0 rho_v 0 0 0 0 0;
           0 0 0 0 0 0 rho_a 0 0 0 0;
           0 0 0 0 0 0 0 rho_z 0 0 0;
           0 0 0 0 0 0 0 0 rho_u 0 0;
           0 0 0 0 0 0 0 0 0 1 0;
           0 0 0 0 0 0 0 0 0 0 1];

Psi = [ 0 0 0 0 0 1 0 0 0 0 0;
        0 0 0 0 0 0 1 0 0 0 0;
        0 0 0 0 0 0 0 1 0 0 0;
        0 0 0 0 0 0 0 0 1 0 0;]';

Pi = [0 0 0 0 0 0 0 0 0 1 0;
      0 0 0 0 0 0 0 0 0 0 1]';

Cons = [0 0 0 0 0 0 0 0 0 0 0]';

% Solve New Keynesian Model
[A,~,R,~,~,~,~,eu,~]=gensys(Gamma0,Gamma1,Cons,Psi,Pi);

C = R*[sigma_v 0 0 0;
       0 sigma_a 0 0;
       0 0 sigma_z 0;
       0 0 0 sigma_u];

%% Matrix D
% Set up measurement matrix Z(t)=D*S(t)
if strcmp(type, 'data')
    % State: 'y','x'; 'pi'; 'r^e'; 'i'; 'v'; 'a'; 'z'; 'u'; 'E(x)'; 'E(pi)'
    D = [0 0 1 0 0 0 0 0 0 0 0; % inflation
         1 0 0 0 0 0 psi_ya 0 0 0 0; % output
         0 0 0 0 1 0 0 0 0 0 0;  % nominal interest rate
         1/(1-alpha) 0 0 0 0 0 -(1-psi_ya)/(1-alpha) 0 0 0 0]; % labor

elseif strcmp(type, 'impulse')
    % State: 'y','x'; 'pi'; 'r^e'; 'i'; 'v'; 'a'; 'z'; 'u'; 'E(x)'; 'E(pi)'
    D = [0 0 1 0 0 0 0 0 0 0 0; % inflation
         1 0 0 0 0 0 psi_ya 0 0 0 0; % output
```

```
            1 0 0 0 0 0 0 0 0 0 0;  % output gap
            0 0 0 0 1 0 0 0 0 0 0;  % nominal interest rate
            1/(1-alpha) 0 0 0 0 0 -(1-psi_ya)/(1-alpha) 0 0 0 0]; % labor
else
    warning('Measurement matrix missing')
end

end
```

Loglikelihood (loglikelihood_DSGE.m)

```
function [ logL ] = loglikelihood_DSGE( THETA )
%loglikelihood_DSGE Likelihood function for State space model
%   State space model:
%   X(t) = A*X(t-1) + C*eps(t)
%   Z(t) = D*X(t)
%   By Julien Neves
global Z

% Solve New Keynesian Model
[A,C,D,eu] = nkbc_model( THETA, 'data');

% Set starting values
X0 = zeros(size(A,2),1);     % set starting X
P0 = dlyap(A,C*C'); % set starting for the variance

% Compute the Kalman Filter
[~, ~, ~, Z_tilde, Omega] = kfilter(Z, A, C, D, 0, X0, P0 );

% Initialize loglikelihood
T = length(Z);
logL = -T/2*log(2*pi)*size(Z,1);

for t = 1:T
    % Update loglikelihood
    logL = logL - 1/2*log(det(Omega{t})) - 1/2* Z_tilde{t}'/Omega{t}* Z_tilde{t};
end
% if imaginary parts or not identified model set likelihood to small value
if (imag(logL)~=0)||(min(eu)==0)
    logL=-9e+200;
end
logL=-logL; % take the negative loglikelihood since SA minimizes
```

```
end
```

Kalman Filter (kfilter.m)

```
function [ X_post, P_post, X_prior, Z_tilde, Omega] = kfilter(Z, A
    , C, D, S_vv, X0, P0 )
%KFILTER Compute the Kalman Filter for a VAR(1) process
%         X[t+1] = AX[t] + Cu[t+1]
%    Z[t+1] = DX[t+1] + v[t+1]
%    By Julien Neves

% Get size of observations
T = length(Z);

% Allocate space for estimates
X_prior = cell(1,T);
X_post = cell(1,T+1);
K = cell(1,T);
Z_tilde = cell(1,T);
Omega = cell(1,T);
P_post = cell(1,T+1);

% Set starting values
X_post{1} = X0;
P_post{1} = P0;

for t = 1:T
    % Compute X_{t+1|t}
    X_prior{t+1}  = A*X_post{t} ;

    % Compute P_{t+1|t}
    P_prior = A * P_post{t} * A' + C * C';

    % Innovations
    Z_tilde{t} = Z(:,t) - D*X_prior{t+1};
    Omega{t} = D*P_prior'*D'+S_vv ;

    % Compute K_{t+1}
    K{t} = P_prior'*D'/(Omega{t});

    % Compute X_{t+1|t+1}
    X_post{t+1}  = X_prior{t+1} + K{t}  * Z_tilde{t};

    % Compute P_{t+1|t+1}
    P_post{t+1} = P_prior- K{t}*D*P_prior;
```

```
end

% Convert post and prior estimates to matrices
X_prior = cell2mat(X_prior);
X_prior(:,1) = [];
X_post = cell2mat(X_post);
X_post(:,1) = [];
end
```