14E026 Advanced Econometric Methods III

Empirical Problem 2

Suleman Dawood, Bjarni Einarsson, Adam Lee & Robertson Wang

1 Data Exploration

1.1 Monthly Data

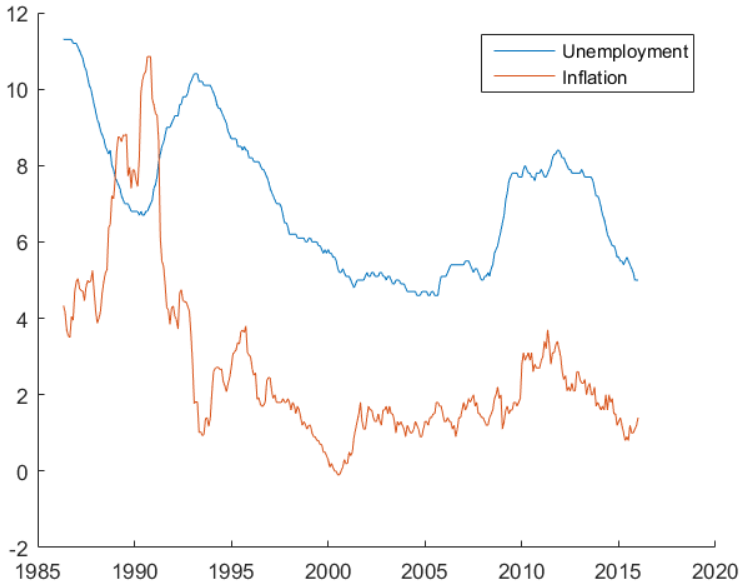

Figure 1: Monthly Time Series of Unemployment and Inflation

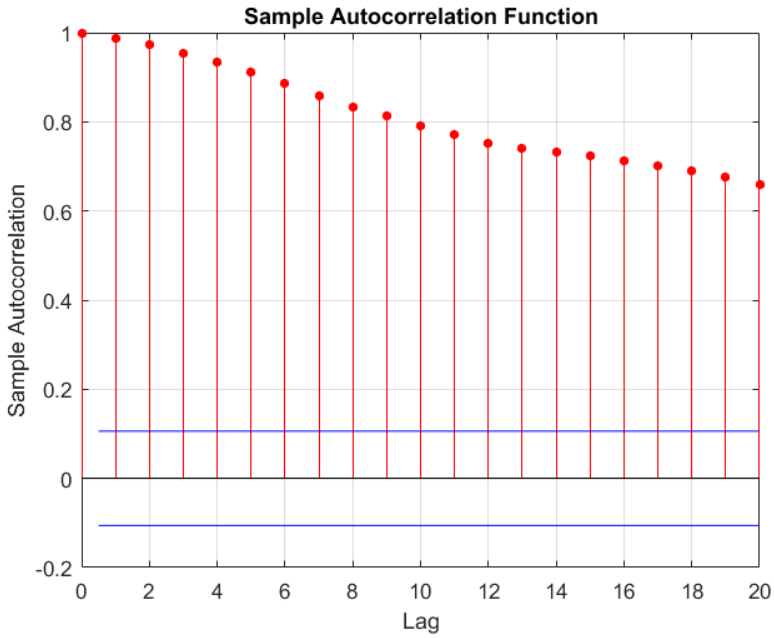

Figure 2: Sample Autocorrelation of Inflation

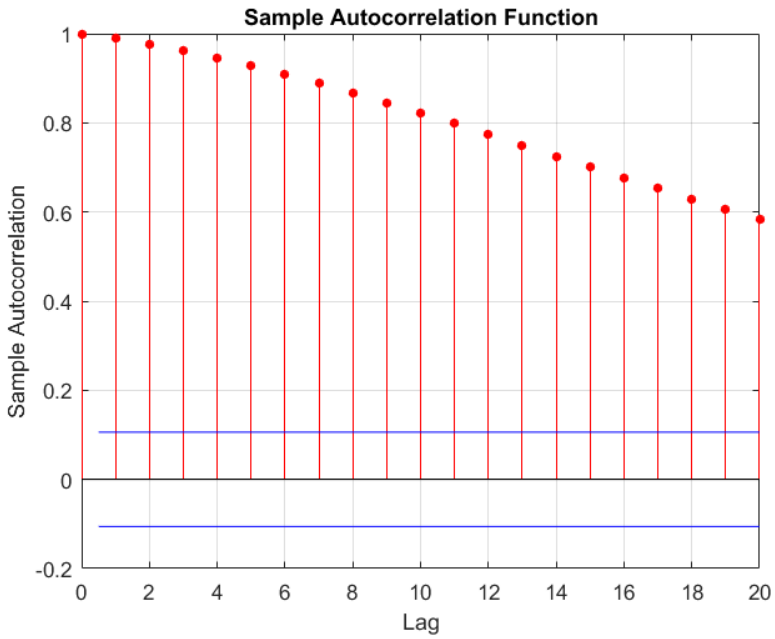

Figure 3: Sample Autocorrelation of Unemployment

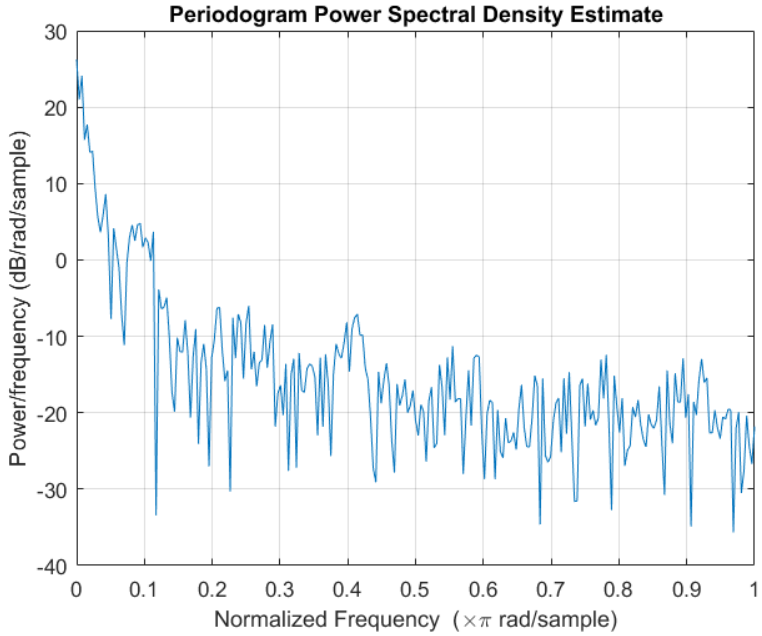

Figure 4: Periodogram of Inflation

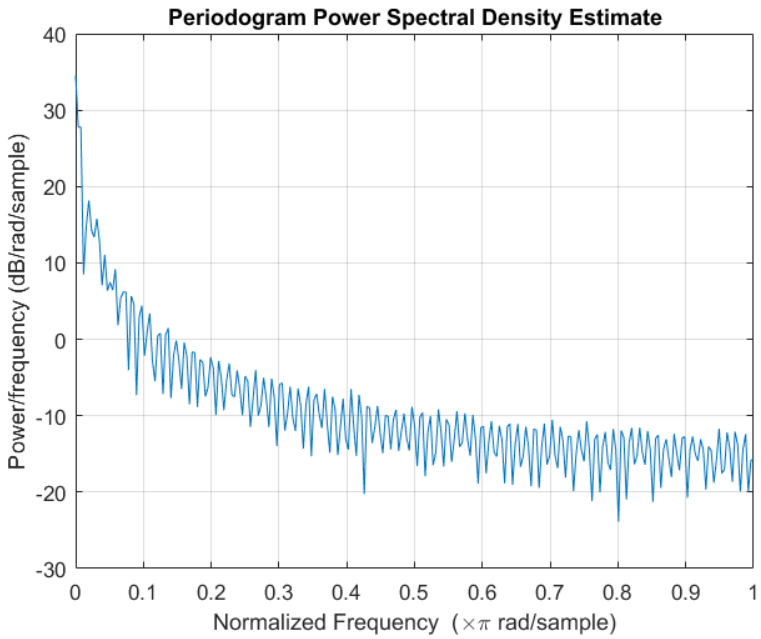

Figure 5: Periodogram of Unemployment

The first figure, wherein both series are plotted, suggests that there may exist some downward trend in unemployment while inflation appears to exhibit stationarity starting from the early 1990s. Theoretically, unemployment must be stationary, since it is a bounded series, and must therefore have a finite mean and variance asymptotically.

Both periodograms suggest that a large amount of the variance can be attributed to low frequency cycles. As the frequency increases, in each case, the variance attributable to this frequency drops off. The periodogram of inflation remains more volatile than that of unemployment throughout. These charts also suggest that the time series may be non-stationary, as the lower the frequency, the more variance can be attributed to it. Moreover the dominant frequency is 0, suggesting an infinite cycle length.

The autocorrelation functions for these series both suggest that there is a fairly similar drop off in autocorrelation per lag. The sample autocorrelation of inflation drops more quickly than that of unemployment. The high sample autocorrelation at long lag lengths indicates that the series are highly persistent. The slow decline in autocorrelation suggests that these time series may be non-stationary.

1.2 Quarterly Data

Quarterly time series were constructed from the original monthly series, by extracting the data points corresponding to February, May, August and November.

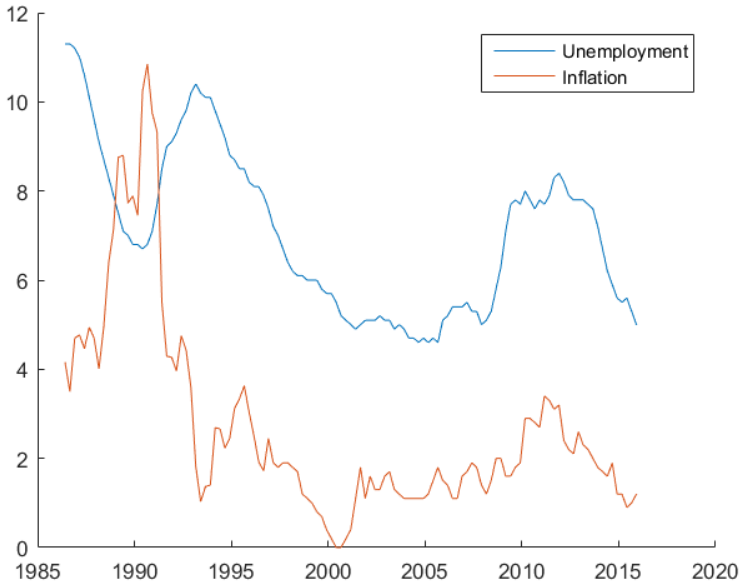

Figure 6: Quarterly Time Series of Unemployment and Inflation

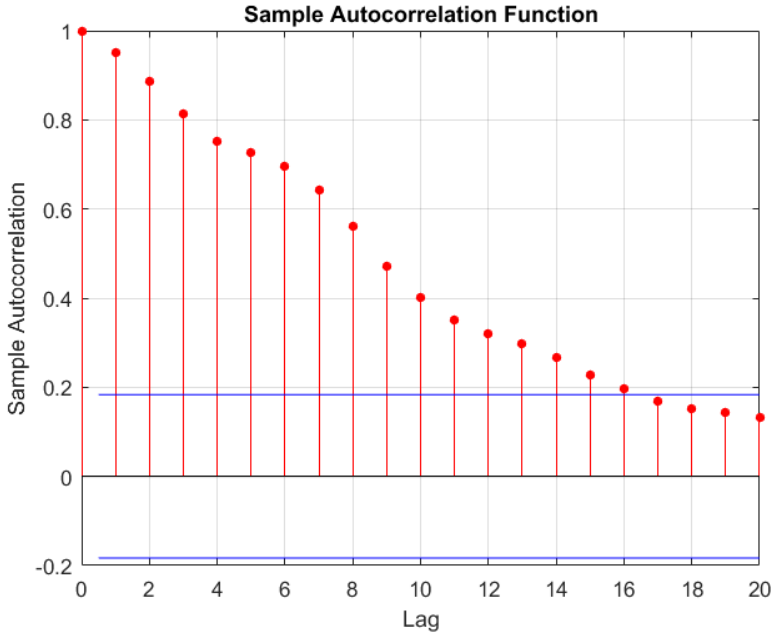

Figure 7: Sample Autocorrelation of Inflation

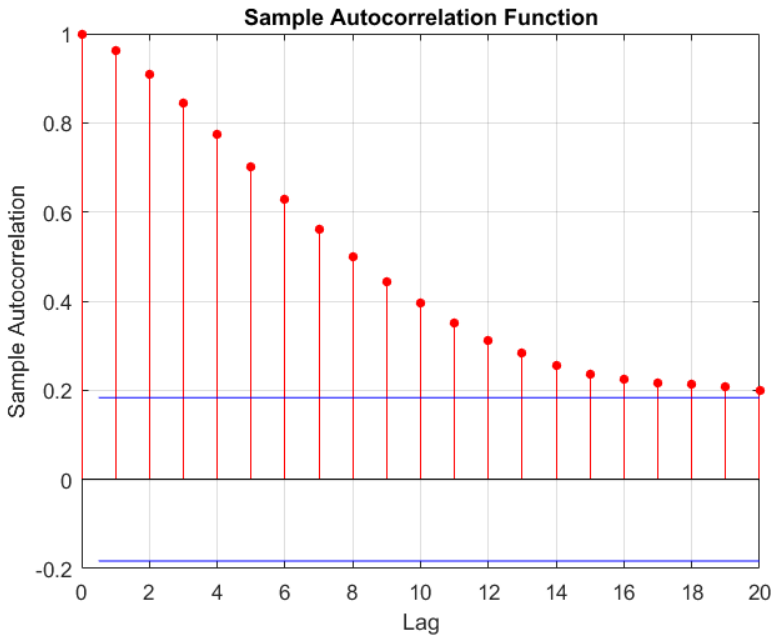

Figure 8: Sample Autocorrelation of Unemployment

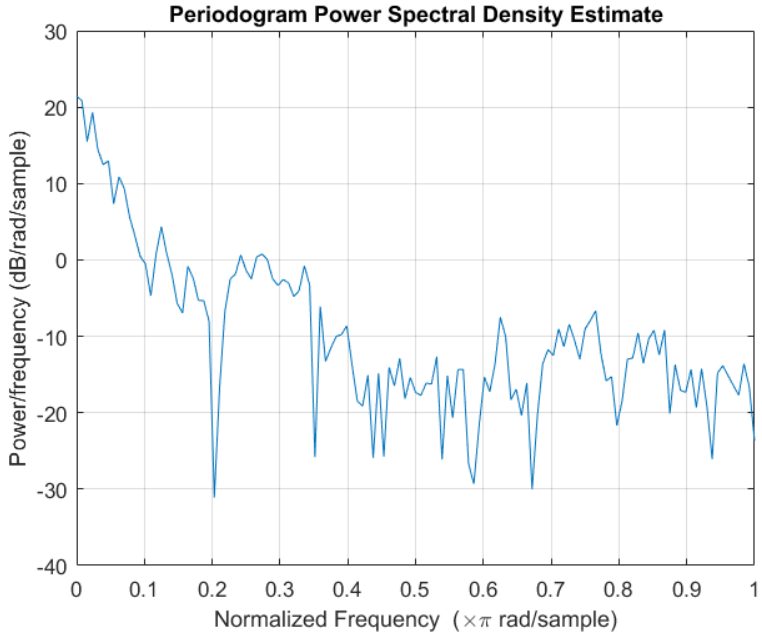

Figure 9: Periodogram of Inflation

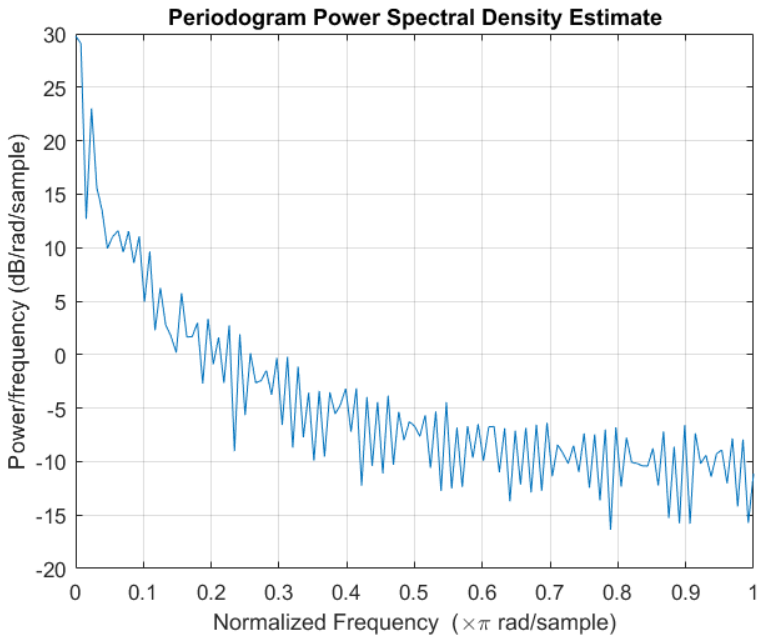

Figure 10: Periodogram of Unemployment

We observe broadly the same pattern in the time series with quarterly observations. Inflation begins to look stationary from the early 1990s, onwards, while unemployment seems to have a slightly negative trend.

The sample autocorrelations still diminish slowly (though faster than the monthly series, for obvious reasons), providing some evidence of possibly non-stationary data.

The periodograms follow the same general pattern as in the monthly case, with low frequency cycles dominating, hinting at an infinite cycle length, and so possibly non-stationary data.

2 Unit Root Testing

We ran unit root tests, using Case 2 (as defined in Hamilton), because these series do not appear to be zero mean (and so a constant should be included in the regression), and a trend-stationary model does not seem appropriate for either series as the unemployment rate is bounded, and inflation does not appear to have a linear-time trend (additional, if this was the case, it would likely result in some strange outcomes for the economy). Therefore, for Phillips-Perron, our estimated model is:

$$Y_t = \alpha + \rho Y_{t-1} + u_t. \tag{1}$$

For the augmented-Dickey Fuller tests, our estimated model is:

$$Y_t = \alpha + \rho Y_{t-1} + \eta_1 \Delta Y_{t-1} + \eta_2 \Delta Y_{t-2} + \cdots + \eta_{p-1} \Delta Y_{t-p+1} + u_t, \tag{2}$$

where the lag length was chosen (as suggested in the lecture slides), as the first lag length in which all the η coefficients were jointly significant (using an F-test). In both cases, our null hypothesis is $H_0 : \ \rho = 1, \ \alpha = 0$. All tests below were performed at a 5% significance level. The Newey-West estimator used in the adjusted PP t-statistic was estimated with 4 as the highest order sample autocovariance for the monthly data, and 3 for the quarterly data.

Table 1: Phillips-Perron Test for unit root

	Reject H_0?	PP t-Stat	Critical Value
Inflation, M	No	-1.3915	-2.87
Inflation, Q	No	-2.3638	-2.88
Unemployment, M	No	-1.6231	-2.87
Unemployment, Q	No	-2.0979	-2.88

Table 2: ADF Test for unit root

	Reject H_0?	ADF t-Stat	Critical Value	Lags	F Stat	Critical Value
Inflation, M	No	-1.6632	-2.87	1	7.0668	3.0214
Inflation, Q	No	-1.9923	-2.88	1	3.4889	3.0759
Unemployment, M	No	-2.0628	-2.87	1	58.36	3.0214
Unemployment, Q	No	-2.4825	-2.88	1	55.81	3.0759

As can be seen in the tables above, for each of these series, we cannot reject the null (using either test) of a unit root, at the 5% significance level.

3 Descriptive VAR Analysis

3.1 Lag Selection

We consider the following VAR(p) model, where x_t is the unemployment rate at time t and z_t is the inflation rate at time t.

$$\begin{bmatrix} x_t \\ z_t \end{bmatrix} = \boldsymbol{\mu} + \boldsymbol{\Phi}_1 \begin{bmatrix} x_{t-1} \\ z_{t-1} \end{bmatrix} + \cdots + \boldsymbol{\Phi}_p \begin{bmatrix} x_{t-p} \\ z_{t-p} \end{bmatrix} + \begin{bmatrix} \epsilon_{1t} \\ \epsilon_{2t} \end{bmatrix}. \tag{3}$$

Results from the BIC criterion for lag selection for lag length 1-4 are reported below:

Table 3: BIC

p	BIC
1	-3.4694
2	-4.0687
3	-3.9201
4	-3.7688

Based on this table, we choose a lag-length of $p = 2$ for the rest of this section. That is, our model is:

$$\begin{bmatrix} x_t \\ z_t \end{bmatrix} = \boldsymbol{\mu} + \boldsymbol{\Phi}_1 \begin{bmatrix} x_{t-1} \\ z_{t-1} \end{bmatrix} + \boldsymbol{\Phi}_2 \begin{bmatrix} x_{t-2} \\ z_{t-2} \end{bmatrix} + \begin{bmatrix} \epsilon_{1t} \\ \epsilon_{2t} \end{bmatrix}. \tag{4}$$

We estimate this model by OLS, obtaining parameter estimates:

$$\hat{\boldsymbol{\mu}} = \begin{bmatrix} 0.1761 \\ -0.0604 \end{bmatrix}, \quad \hat{\boldsymbol{\Phi}}_1 = \begin{bmatrix} 1.6430 & 0.0188 \\ -0.4910 & 1.1267 \end{bmatrix}, \quad \hat{\boldsymbol{\Phi}}_2 = \begin{bmatrix} -0.6799 & 0.0054 \\ 0.5162 & -0.1852 \end{bmatrix}. \tag{5}$$

3.2 Granger-Causality

We next run Granger Causality tests for inflation on unemployment and vice versa. That is, writing out the two equations our VAR(2) captures,

$$x_t = \mu_1 + \phi_{1,11}x_{t-1} + \phi_{1,12}z_{t-1} + \phi_{2,11}x_{t-2} + \phi_{2,12}z_{t-2} + \epsilon_{1t} \tag{6}$$

$$z_t = \mu_2 + \phi_{1,21}x_{t-1} + \phi_{1,22}z_{t-1} + \phi_{2,21}x_{t-2} + \phi_{2,22}z_{t-2} + \epsilon_{2t} \tag{7}$$

we test the following restrictions:

(a) $H_0 : \ \phi_{1,12} = 0, \quad \phi_{2,12} = 0$. That is our null hypothesis is that inflation does not Granger-cause unemployment.

(b) $H_0 : \ \phi_{1,21} = 0, \quad \phi_{2,21} = 0$. That is our null hypothesis is that unemployment does not Granger-cause inflation.

we use an F-test to check these restrictions. The results of this test are tabulated below. In both cases, the null of each series *not* Granger-causing the other is rejected at the 5% significance level.

Table 4: Test for Granger-Causality

Test	Outcome	F-statistic	Critical Value
(a)	Null rejected	8.599	3.926
(b)	Null rejected	5.994	3.926

3.3 Impulse Response Functions

To calculate the Impulse Response Functions, we first write our model into companion form. That is, letting $Y_t = [x_t, z_t]'$, our model can be written as:

$$\begin{bmatrix} \tilde{Y}_t \\ \tilde{Y}_{t-1} \end{bmatrix} = \begin{bmatrix} \boldsymbol{\Phi}_1 & \boldsymbol{\Phi}_2 \\ \boldsymbol{I}_{2\times 2} & \boldsymbol{0}_{2\times 2} \end{bmatrix} \begin{bmatrix} \tilde{Y}_{t-1} \\ \tilde{Y}_{t-2} \end{bmatrix} + \begin{bmatrix} Q \\ \boldsymbol{0}_{2\times 2} \end{bmatrix} \boldsymbol{\eta}_t. \tag{8}$$

where $\tilde{Y}$ denote demeaned series, $\boldsymbol{\eta}_t$ is a vector of orthonormal shocks, and Q is a lower-triangular matrix, such that $QQ' = \Sigma$ is the Cholesky decomposition of $\Sigma = \mathbb{E}(\epsilon_t \epsilon_t')$. We note that, in this case, $\boldsymbol{\eta}_t = Q^{-1}\boldsymbol{\epsilon}_t$. Replacing everything here with our estimated model, where we note that $\hat{\Sigma} = \begin{bmatrix} 0.0326 & -0.0164 \\ -0.0164 & 0.3957 \end{bmatrix}$, we then have that this model can be further re-written as:

$$\boldsymbol{Y}_t = \boldsymbol{A}\boldsymbol{Y}_{t-1} + \boldsymbol{C}\boldsymbol{\eta}_t, \tag{9}$$

where

$$\boldsymbol{Y}_t = \begin{bmatrix} \tilde{Y}_t \\ \tilde{Y}_{t-1} \end{bmatrix}, \quad , \boldsymbol{A} = \begin{bmatrix} \boldsymbol{\Phi}_1 & \boldsymbol{\Phi}_2 \\ \boldsymbol{I}_{2\times 2} & \boldsymbol{0}_{2\times 2} \end{bmatrix}, \quad \boldsymbol{C} = \begin{bmatrix} Q \\ \boldsymbol{0}_{2\times 2} \end{bmatrix}.$$

From this, and given a initial (orthonormal) shock $\boldsymbol{\eta}_1$ (where all other $\boldsymbol{\eta}_s$ are zero), our impulse responses can be easily calculated, with the sequence of impulse responses given by:

$$\boldsymbol{C\eta_1},\ \boldsymbol{AC\eta_1},\ \boldsymbol{A^2C\eta_1}, \ldots$$

We consider a unit shock to unemployment, and then a unit shock to inflation : $\boldsymbol{\eta_1} = [1, 0]'$, and then $\boldsymbol{\eta_1} = [0, 1]'$ respectively. These impulse-responses are shown in the figures below.

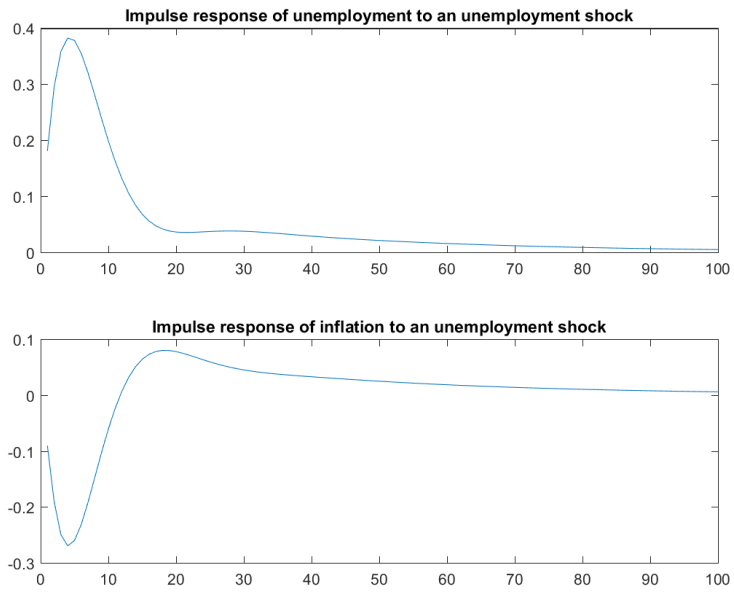

Figure 11: Impulse Responses of Unemployment and Inflation to an Unemployment Shock

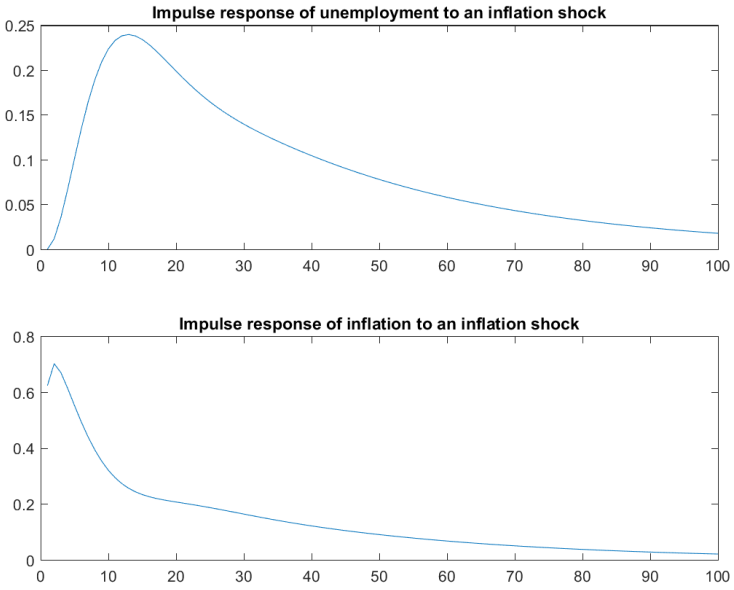

Figure 12: Impulse Responses of Unemployment and Inflation to an Inflation Shock

3.4 Forecast Error Variance Decomposition

We now compute the forecast error variance decomposition. The graphs below show the proportion of the forecast error variance (FEV) attributable to each shock.

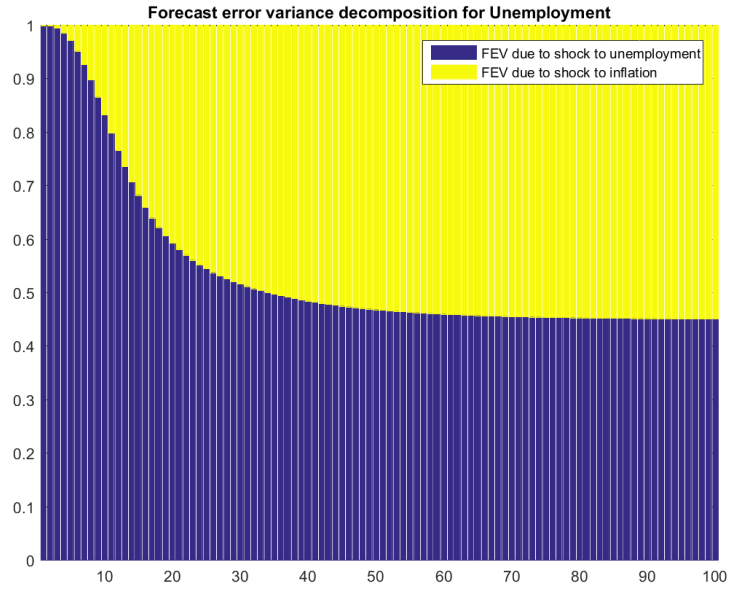

Figure 13: Forecast Error Variance Decomposition for Unemployment

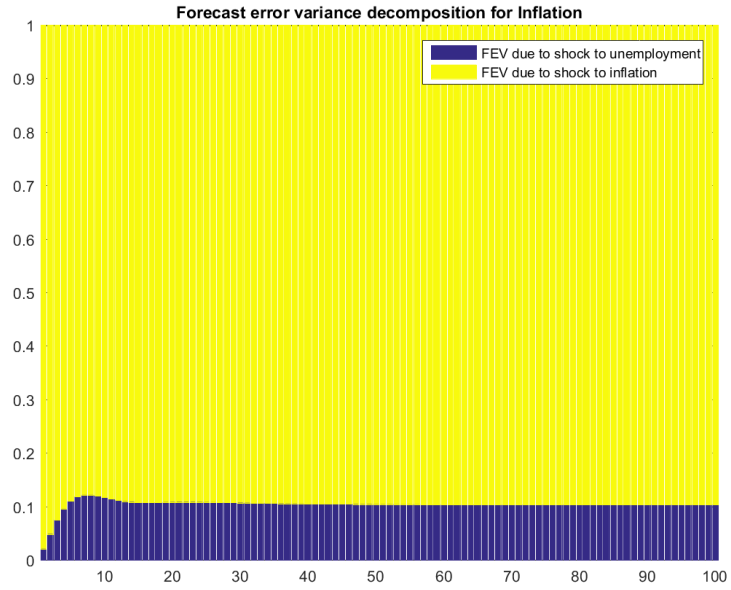

Figure 14: Forecast Error Variance Decomposition for Inflation

4 Time-Varying Parameter Model

We employ the following state space model

$$Y_t = A'X_t + Z_t\alpha_t + \varepsilon_t, \varepsilon \sim \mathrm{N}(0, H_t)$$
$$\alpha_{t+1} = T_t\alpha_t + R_t\eta_t, \eta \sim \mathrm{N}(0, Q_t)$$

where we define $Y_t = \pi_t, A = \phi, X_t = \pi_{t-1}, Z_t = UN_{t-1}, \alpha_t = \beta_t, H_t = \sigma_\varepsilon^2, T_t = 1, R_t = 1$ and $Q_t = \sigma_\eta^2$. The only change this implies to the Kalman filter presented on the slides is that now

$$v_t = Y_t - A'X_t - Z_t a_{t|t-1}.$$

We estimate the model parameters, $\phi, \sigma_\varepsilon^2$ and σ_η^2 by maximum likelihood. The results of the parameter estimation along with their standard errors are presented in table 5. Figure 15 presents the filtered estimates of β_t along with 95% confidence bounds and figure 16 does the same for the smoothed estimates of β_t.

Table 5: MLE estimation

	Parameter	Standard error
ϕ	0.9928	0.0025
σ_ε^2	0.0062	0.0000
σ_η^2	0.0001	0.0000

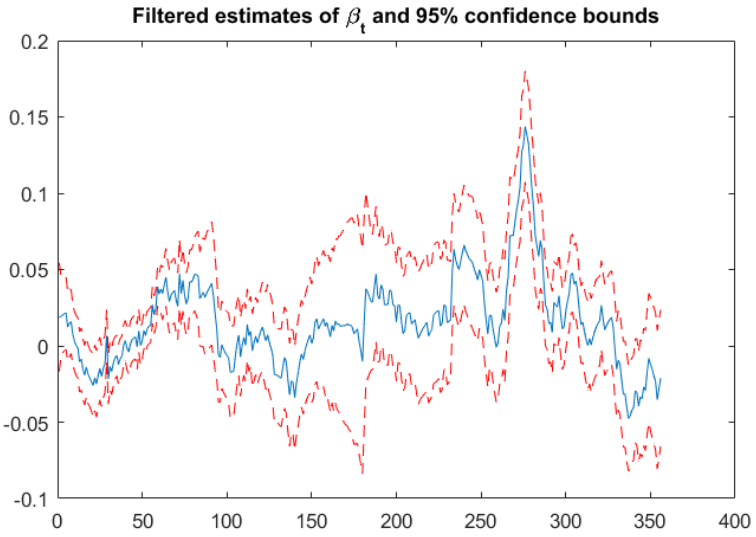

Figure 15: Filtered estimates of β_t

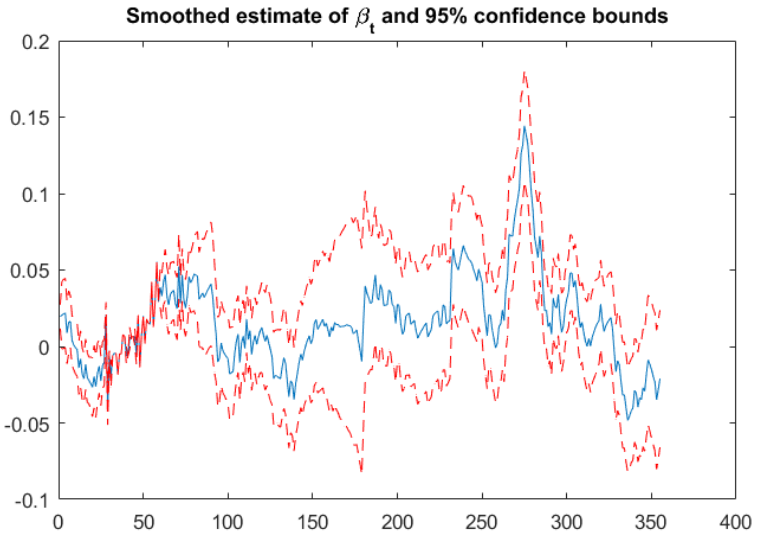

Figure 16: Smoothed estimates of β_t

A Appendix

The code used in this analysis is pasted below.

A.1 Main Block

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Adv. Econometric Methods III                                                   %
%      Empirical Homework 2                                                      %
%                                                                                %
%                                                                                %
% Team 3:                                                                        %
% Suleman Dawood, Bjarni Einarsson, Adam Lee & Robertson Wang                    %
%                                                                                %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%% Housekeeping
clear all
close all

clc
%% Data prep - Quarterly

[data0, names] = xlsread('AllData.xlsx'); % load data
[data1, names] = xlsread('AllDataInflation.xls');
ur = data0(:,5);    % UK unemployment rate
inf = data1(:,5);    % UK inflation rate
year = data0(:,1);
month = data0(:,2);


T = length(ur);    % Number of quarters
%TT_inf=(month_inf(1)/12)+year_inf(1):(1/12):(month_inf(end)/12)+year_inf(end);
%
TT=(month(1)/12)+year(1):(1/12):(month(end)/12)+year(end);


%% Data Prep - Quarterly

k=zeros(T,1);
for i=1:T
    if (month(i) == 2)|(month(i) == 5)|(month(i) == 8)|(month(i) == 11);
        k(i)=1;
end
end

ur_q = ur(k==1);
inf_q = inf(k==1);
year_q=year(k==1);
month_q=month(k==1);
T_q=length(ur_q);
TT_q=(month_q(1)/12)+year_q(1):(3/12):(month_q(end)/12)+year_q(end);
```

```
%% Data Exploration - Monthly
% 1. Plot of the time series, inflation vs. unemployment
figure; hold on
a1 = plot(TT,[ur inf]);
legend('Unemployment','Inflation')

% 2a. Plot of the autocorrelation function for inflation
figure
autocorr(inf)
% 2b. Plot of the autocorrelation function for unemployment
figure
autocorr(ur)

% 3a. Plot of the periodogram for inflation
figure
periodogram(inf)
% 3b. Plot of the periodogram function for unemployment
figure
periodogram(ur)


%% Data Exploration - Quarterly
% 5. Plot of the time series, inflation vs. unemployment
figure; hold on
a1 = plot(TT_q,[ur_q inf_q]);
legend('Unemployment','Inflation')

% 2a. Plot of the autocorrelation function for inflation
figure
autocorr(inf_q)
% 2b. Plot of the autocorrelation function for unemployment
figure
autocorr(ur_q)

% 3a. Plot of the periodogram for inflation
figure
periodogram(inf_q)
% 3b. Plot of the periodogram function for unemployment
figure
periodogram(ur_q)

%% 2a - Unit Root Testing, DF

% a) Dickey-Fuller for Inflation, Monthly

J=0;
lags=1;
while J==0

[h_inf_m,t_inf_m,c_inf_m,f_inf_m,c2_inf_m,J,b_inf_m] = DFTest(inf,lags,2);

lags=lags+1;

end
```

```
lags_inf_m = lags-1;

% b) Dickey-Fuller for UR, Monthly

J=0;
lags=1;
while J==0

[h_ur_m,t_ur_m,c_ur_m,f_ur_m,c2_ur_m,J,b_ur_m] = DFTest(ur,lags,2);

lags=lags+1;

end
lags_ur_m = lags-1;

% c) Dickey-Fuller for Inflation, Quarterly

J=0;
lags=1;
while J==0

[h_inf_q,t_inf_q,c_inf_q,f_inf_q,c2_inf_q,J,b_inf_q] = DFTest(inf_q,lags,2);

lags=lags+1;

end
lags_inf_q = lags-1;

% d) Dickey-Fuller for UR, Quarterly

J=0;
lags=1;
while J==0

[h_ur_q,t_ur_q,c_ur_q,f_ur_q,c2_ur_q,J,b_ur_q] = DFTest(ur_q,lags,2);

lags=lags+1;

end
lags_ur_q = lags-1;

%% 2b - Unit Root Testing, PP

h_m=4; %Lag length for NW estimator, monthly
h_q=3; %Lag length for NW estimator, quarterly

% a) Phillips - Perron for Inflation, Monthly
[h_inf_m_pp,t_inf_m_pp,c_inf_m_pp,b_inf_m_pp,SE_inf_m_pp] = PPtest(inf,h_m);
% b) Phillips - Perron for Unemployment, Monthly
[h_ur_m_pp,t_ur_m_pp,c_ur_m_pp,b_ur_m_pp,SE_ur_m_pp] = PPtest(ur,h_m);
% c) Phillips - Perron for Inflation, Quarterly
[h_inf_q_pp,t_inf_q_pp,c_inf_q_pp,b_inf_q_pp,SE_inf_q_pp] = PPtest(inf_q,h_q);
% d) Phillips - Perron for Unemployment, Quarterly
[h_ur_q_pp,t_ur_q_pp,c_ur_q_pp,b_ur_q_pp,SE_ur_q_pp] = PPtest(ur_q,h_q);
```

```
%% 3 - Descriptive VAR analysis
% - Lag selection using BIC
L = 4;              % Lag length
BIC = NaN(L,1);
Y = [ur_q inf_q];
n = size(Y,2);
for j = 1:L
    X = [ones(size(Y,1),1)];
    for i = 1:j
        X= [X lagger(Y,i)];
    end
    Y = Y(j+1:end,:);
    X = X(j+1:end,:);
    Ts = size(Y,1);

    bols = inv(X'*X)*(X'*Y);
    Sigma = ((Y-X*bols)'*(Y-X*bols))/Ts;
    BIC(j,1) = log(det(Sigma)) + log(Ts)/Ts*j*n^2;
end

% - Estimate preferred model
minbic = min(BIC);
BIC_min = BIC==minbic;
L=find(BIC_min);
Y = [ur_q inf_q];
X = [ones(size(Y,1),1)];
for i = 1:L
    X= [X lagger(Y,i)];
end
Y = Y(L+1:end,:);
X = X(L+1:end,:);
Ts = size(Y,1);
k=size(X,2);
bols = inv(X'*X)*(X'*Y);
Sigma = ((Y-X*bols)'*(Y-X*bols))/Ts;


%% Granger Causality


% Equation for ur
R1 = [0 0 1 0 0;0 0 0 0 1];
F_ur = (R1*bols(:,1))'*inv(R1*inv(X'*X)*R1')*(R1*bols(:,1))/(Sigma(1,1)); %#R is 1


% Equation for inf
R2 = [0 1 0 0 0;0 0 0 1 0];
F_inf = (R2*bols(:,2))'*inv(R2*inv(X'*X)*R2')*(R2*bols(:,2))/(Sigma(2,2)); %#R is 1

% Critical Value

Granger_cValue = finv(0.95,1,Ts-k);
```

```
if F_ur>Granger_cValue
    Granger_F_ur = 1; %Alternative
else Granger_F_ur=0; %Null - inf does not Granger-Cause ur
end

if F_inf>Granger_cValue
    Granger_F_inf = 1; %Alternative
else Granger_F_inf=0; %Null - ur does not Granger-Cause inf
end



%% Impulse response functions
% We note that the preferred model has p = 2.
% Transform AR(2) into AR(1) companion form

Phi1 = [bols(2,1),bols(3,1); bols(2,2), bols(3,2)];
Phi2 = [bols(4,1),bols(5,1); bols(4,2), bols(5,2)];

A = [Phi1, Phi2; eye(2,2), zeros(2,2)];
mu = [bols(1,1);bols(1,2)];
Q = chol(Sigma, 'lower');

C = [Q ; zeros(2,2)];


h=100;

   % h period IRF based on a shock to ur
   shock_ur = [1;0];
   C1 = C*shock_ur;


   IRF_ur  = zeros(4,h);

   for i=1:h
       IRF_ur(:,(i-1)+1) = A^(i-1)*C*shock_ur;
   end
   ur_irf_urshock = IRF_ur(1,:);
   inf_irf_urshock = IRF_ur(2,:);

   figure
   subplot(2,1,1)
   plot(ur_irf_urshock)
   title('Impulse response of unemployment to an unemployment shock')
   subplot(2,1,2)
   plot(inf_irf_urshock)
   title('Impulse response of inflation to an unemployment shock')
   %saveas(gcf,'irf_urshock.png')

   % h period IRF based on a shock to inf
   shock_inf = [0;1];
   C1 = C*shock_inf;
```

```
   IRF_inf  = zeros(4,h);

   for i=1:h
       IRF_inf(:,(i-1)+1) = A^(i-1)*C*shock_inf;
   end
   ur_irf_infshock = IRF_inf(1,:);
   inf_irf_infshock = IRF_inf(2,:);

   figure
   subplot(2,1,1)
   plot(ur_irf_infshock)
   title('Impulse response of unemployment to an inflation shock')
   subplot(2,1,2)
   plot(inf_irf_infshock)
   title('Impulse response of inflation to an inflation shock')
   %saveas(gcf,'irf_infshock.png')



   %% Variance Decomposition

   VD_ur=zeros(h,2);
   for i=1:h
       VD_ur(i,1)=FEVD(i,1,1,Sigma,A);
       VD_ur(i,2)=FEVD(i,1,2,Sigma,A);
   end

   VD_inf=zeros(h,2);
   for i=1:h
       VD_inf(i,1)=FEVD(i,2,1,Sigma,A);
       VD_inf(i,2)=FEVD(i,2,2,Sigma,A);
   end

   figure
   bar(VD_ur,'stacked')
   axis('tight')
   legend('FEV due to shock to unemployment ', 'FEV due to shock to inflation')
   title('Forecast error variance decomposition for Unemployment')
   %saveas(gcf,'FEVD_ur.png')
   figure
   bar(VD_inf,'stacked')
   axis('tight')
   legend('FEV due to shock to unemployment', 'FEV due to shock to inflation')
   title('Forecast error variance decomposition for Inflation')
   %saveas(gcf,'FEVD_inf.png')


%% 4 - Time Varying Paramters
    %% Estimation
options = optimset('PlotFcns',@optimplotx,'DISPLAY','iter');
AA = [0,-1,0;0,0,-1];
BB= [-0.000000000001;-0.000000000001];
[param,~,~,~,~,~,hess] = fmincon(@(param) kfloglik(param,[ur inf]), [0.1 0.1 0.1],AA,BB,[],[],-Inf,Inf,
separam = sqrt(diag(inv(hess)));
```

```
    %% Kalman output
[ap, pp, af, pf, as, ps,r, N] = kalmansmooth(param,[ur inf]);


    %% Figures
figure
plot([af, af+1.96*sqrt(pf), af-1.96*sqrt(pf)])

figure
plot([as(2:end), as(2:end)+1.96*real(sqrt(ps(2:end))), as(2:end)-1.96*real(sqrt(ps(2:end)))])
```

A.2 Called User-Defined Functions

A.2.1 Augmented Dickey-Fuller Test

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Adv. Econometric Methods III                                                %
%      Dickey-Fuller test                                                     %
%  h - result of ADF test. 0: Null, 1: Alternative
%  t - t-stat
%  c - critical value
%  F - F stat for joint sig of all lags
%  c_value - critical value for the F test
%   Joint_Sig -  0: null (insignificant)
%
% This function perfoms a Dickey-Fuller/ADF test of whether or not a time
% series has a unit root. Outputs are as above. The inputs are:
%
% y - time series
% lags - number of lagged differences used; 0 for DF
% model - Case 1, 2 or 3, in the terminology used in Hamilton.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%



function[h,t,c,F,c_value,Joint_Sig,b]= DFTest(y,lags,model)


T=length(y);
L = lagmatrix(y,[0:lags+1]);


if model == 1

% Case 1 according to Hamilton. See pg. 528 -  critical value at 5% is
% always -1.95.


%Generate data matrix
X=zeros(T,lags+1);
X(:,1) = L(:,2);
for j=1:lags
```

```
    X(:,j+1)=L(:,j+1)-L(:,j+2);
end
k = length(X(1,:));
start_row=find(isnan(X(:,end)),1,'last')+1; %drop rows with NaN
X=X(start_row:end,:);
y=y(start_row:end);
T=length(y);
%OLS
b=regress(y,X);

resid = y-X*b;

s_sq = (1/(T-k))*resid'*resid;

V = s_sq*inv(X'*X);

SE = diag(sqrt(V));

% t - statt for coefficient on AR(1) coefficient
t=(b(1)-1)/SE(1);

% F - test for joint significance of all lags.
if lags > 0

I_k = eye(k);
R=I_k(2:end,:);

F = ((R*b)'*inv(R*inv(X'*X)*R')*(R*b))/((k-1)*s_sq);
c_value = finv(0.95,k-1,T-k);

Joint_Sig = 0; %null of joint insignificance

if abs(F)>c_value
    Joint_Sig=1; %H_1 for F_test
end

else

 F = (b(1)-0)/SE(1); %t test is same as F test for one restriction
    c_value = tinv(0.975,T-k);

    Joint_Sig = 0; %null of joint insignificance

if abs(F)>c_value
    Joint_Sig=1; %H_1
end
end


c=-1.95;
h = 0; %can't reject the null (ADF stat)
if t<c
    h = 1; %can reject the null of a unit root
end
```

```
elseif model == 2

    % This is for case 2 in Hamilton's terminology.

%generate data matrix
X=ones(T,lags+2);
X(:,1) = L(:,2);
for j=1:lags
    X(:,j+1)=L(:,j+1)-L(:,j+2);
end
k = length(X(1,:));
start_row=find(isnan(X(:,end-1)),1,'last')+1; % drop NaN rows
X=X(start_row:end,:);
y=y(start_row:end);
T=length(y);
%OLS
b=regress(y,X);

resid = y-X*b;

s_sq = (1/(T-k))*resid'*resid;

V = s_sq*inv(X'*X);

SE = diag(sqrt(V));

% t - stat for coefficient on AR(1) coefficient
t=(b(1)-1)/SE(1);

% F - test for joint significance of all lags.
if lags > 0

I_k = eye(k);
R=I_k(2:end-1,:);

F = ((R*b)'*inv(R*inv(X'*X)*R')*(R*b))/((k-1)*s_sq);
c_value = finv(0.95,k-1,T-k);

Joint_Sig = 0; %null of joint insignificance

if abs(F)>c_value
    Joint_Sig=1; %H_1
end

else %If lags  = 0

 F = (b(1)-0)/SE(1); %t test is same as F test for one restriction
    c_value = tinv(0.975,T-k);

    Joint_Sig = 0; %null of joint insignificance

if abs(F)>c_value
    Joint_Sig=1; %H_1
```

```
end
end

c= DF_Case2_cValue(T);

h = 0; %can't reject the null (ADF stat)

if t<c
    h = 1; %Can reject the null of a unit root
end



elseif model == 3  % The following is for case 3, in Hamilton's definition.
    % that is, the true process is y_t = \alpha + y_{t-1} +u_t, with
    % \alpha \neq 0.
    % Hamilton p. 487 gives that in this case, the usual t and F tests
    % have the usual t and F distributions.


%generate data matrix
X=ones(T,lags+2);
X(:,1) = L(:,2);
for j=1:lags
    X(:,j+1)=L(:,j+1)-L(:,j+2);
end
k = length(X(1,:));
start_row=find(isnan(X(:,end-1)),1,'last')+1; %drop NaN rows
X=X(start_row:end,:);
y=y(start_row:end);
T=length(y);
%OLS
b=regress(y,X);

resid = y-X*b;

s_sq = (1/(T-k))*resid'*resid;

V = s_sq*inv(X'*X);

SE = diag(sqrt(V));

% t - stat for coefficient on AR(1) coefficient
t=(b(1)-1)/SE(1);

% F - test for joint significance of all lags.
if lags > 0

I_k = eye(k);
R=I_k(2:end-1,:);

F = ((R*b)'*inv(R*inv(X'*X)*R')*(R*b))/((k-1)*s_sq);
c_value = finv(0.95,k-1,T-k);
```

```
Joint_Sig = 0; %null of joint insignificance

if abs(F)>c_value
    Joint_Sig=1; % H_1
end

else

 F = (b(1)-0)/SE(1); %t test is same as F test for one restriction
    c_value = tinv(0.975,T-k);

    Joint_Sig = 0; %null of joint insignificance

if abs(F)>c_value
    Joint_Sig=1; % H_1
end
end

c= tinv(0.975, T-k);

h = 0; %can't reject the null (ADF stat)

if abs(t)>c
    h = 1; % Can reject the null of a unit root.
end

end
```

A.2.2 Phillips-Perron Test

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Adv. Econometric Methods III                                                  %
%      Phillips-Perron test for case 2                                             %
%  h - result of ADF test. 0: Null, 1: Alternative
%  t - adjusted t-stat
%  c - critical value
%
% This function performs a Phillips-Perron test of whether or not a time
% series has a unit root. Outputs are as above. The inputs are:
% y - time series
% q - highest order of sample autocovariance used in calculating NW estimator
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function[h,t,c,b,SE]=PPtest(y,q)

% y - time series
% q - # lags for NW estimator

% Generate data matrix

T=length(y);

L=lagmatrix(y,1);
```

```
X=ones(T, 2);

X(:,1)=L;

X=X(2:end,:);
y=y(2:end);
k=length(X(1,:));
T=length(y);

%perform regression

b=regress(y,X);

resid = y-X*b;

s_sq = (1/(T-k))*resid'*resid;

V = s_sq*inv(X'*X);

SE = diag(sqrt(V));

% calculate \gamma_0
gamma_zero = (1/T)*resid'*resid;

%recursive calculation of Newey-West estimator.
NW =gamma_zero;

for i = 1:q

sum=0;
for j=i+1:T
   sum = resid(j)*resid(j-i);
end

NW = NW+(2)*(1-j/(q+1))*(sum/T);

end

%Calculation of modified t-stat
ols_t = (b(1)-1)/SE(1);
t = sqrt((gamma_zero/NW))*ols_t - (1/2)*((NW-gamma_zero)/sqrt(NW))*(T*SE(1))/sqrt(s_sq);

c= DF_Case2_cValue(T);

h = 0; %can't reject the null (ADF stat)

if t<c
    h = 1; % Reject null of unit root
end
```

A.2.3 Calculation of Critical Value for Case 2

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Adv. Econometric Methods III                                          %
%      Critical Values for Dickey-Fuller test  in case 2                %
%                                                                       %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


function c = DF_Case2_cValue(T)

if T <= 25
    c = -3;
elseif T<=50
    c = -2.93;
elseif T<=100
    c=-2.89;
elseif T<=250
    c=-2.88;
elseif T<=500
    c=-2.87;
else c=-2.86;
end
```

A.2.4 Lagger

```
% Lags variable p periods and places zeros in the rows corresponding to
% first p periods

function out = lagger(y,p)

[R,C] = size(y);
y1 = y(1:(R-p),:);
out = [zeros(p,C); y1];
```

A.2.5 Forecast Error Variance Decomposition

```
function[FEVD] = FEVD(h,j,k,Sigma,A)

    % h - # periods ahead forecast
    % j - error variance of variable ... j\in \{1,2\}
    % k - shock to variable ... k\in \{1,2\}


   Psi =zeros(2,2);
   SUM =zeros(2,2);
   J=[eye(2), zeros(2,2)];
   for i = 0:h-1

       Psi=J*A^i*J';
```

```
        SUM = SUM + Psi*Sigma*Psi';
    end
    mse = SUM(j,j);

    FEVD=0;

    e_j = zeros(2,1);
    e_k = zeros(2,1);
    e_j(j,:)=1;
    e_k(k,:)=1;
    ete=0;
    Q=chol(Sigma,'lower');
    for i=0:h-1
        Psi=J*A^i*J';
        Theta = Psi*Q;
        ete = ete+(e_j'*Theta*e_k)^2;
    end
    FEVD = ete/mse;
```

A.2.6 Kalman Filter & Log Likelihood

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Adv. Econometric Methods III                                           %
%      Empirical Homework 2                                              %
%                                                                        %
% Team 3:                                                                %
% Suleman Dawood, Bjarni Einarsson, Adam Lee & Robertson Wang            %
%                                                                        %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function ll = kfloglik(param,data)

phi = param(1);
H = param(2);
Q = param(3);

y = data(2:end,1);
n = size(y,1);
Z = lagger(data,1);
yl = Z(2:end,1);
Z = Z(2:end,2);
% Initialisation
ll = 0;
s = zeros(n,1);
s2 = zeros(n,1);
a10 = zeros(1,1);
P10 = eye(1)*10^7;
% Matrices
T = eye(1);
R = [1];

for i = 1:n
```

```
    v   = y(i,:)-phi*yl(i,:)-Z(i,:)*a10;
    F   = Z(i,:)*P10*Z(i,:)' + H;
    a11 = a10 + P10*Z(i,:)'*inv(F)*v;
    P11 = P10 - P10*Z(i,:)'*inv(F)*Z(i,:)*P10;
    a10 = T*a11;
    P10 = T*P10*(T-T*P10*Z(i,:)'*inv(F)*Z(i,:))'+R*Q*R';
    % Calculate negative log likelihood
    if i > 1
        ll = ll + 0.5*(log(2*pi) + log(F)+v^2/F);
    end

    s(i,:) = a11;
    s2(i,:,:) = P11;
end
```

A.2.7 Kalman Smoother

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Adv. Econometric Methods III                                                   %
%      Empirical Homework 2                                                      %
%                                                                                %
% Team 3:                                                                        %
% Suleman Dawood, Bjarni Einarsson, Adam Lee & Robertson Wang                    %
%                                                                                %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function [ap, pp, af, pf, as, ps,r,N] = kalmansmooth(param,data)
phi = param(1);
H = param(2);
Q = param(3);

y = data(2:end,1);
n = size(y,1);
Z = lagger(data,1);
yl = Z(2:end,1);
Z = Z(2:end,2);

ap = zeros(n,1);
pp = zeros(n,1);
af = zeros(n,1);
pf = zeros(n,1);
as = zeros(n,1);
ps = zeros(n,1);
vv = zeros(n,1);

%% Run Kalman filter
a10 = zeros(1,1);
P10 = eye(1)*10^7;
% Matrices
T = eye(1);
R = [1];
```

```
for i = 1:n
    % Store predicted states
    ap(i,:) = a10;
    pp(i,:) = P10;
    % Kalman filter
    v   = y(i,:)-phi*yl(i,:)-Z(i,:)*a10;
    F   = Z(i,:)*P10*Z(i,:)' + H;
    a11 = a10 + P10*Z(i,:)'*inv(F)*v;
    P11 = P10 - P10*Z(i,:)'*inv(F)*Z(i,:)*P10;
    a10 = T*a11;
    P10 = T*P10*(T-T*P10*Z(i,:)'*inv(F)*Z(i,:))'+R*Q*R';
    % Store filtered states
    af(i,:) = a11;
    pf(i,:) = P11;
    % store stuff for smoother
    vv(i,:) = v;
end


%% State smoothing recursion
r1 = 0;
N1 = 0;
r = zeros(n,1);
N = zeros(n,1);
for i = n:-1:1
    r0 = Z(i,:)'*inv(F)*vv(i) + (T-T*pp(i,:)*Z(i,:)'*inv(F)*Z(i,:))'*r1;
    N0 = Z(i,:)'*inv(F)*Z(i,:) + (T-T*pp(i,:)*Z(i,:)'*inv(F)*Z(i,:))'*N1*(T-T*pp(i,:)*Z(i,:)'*inv(F)*Z(
    as(i,:) = ap(i,:) + pp(i,:)*r0;
    ps(i,:) = pp(i,:) - pp(i,:)*N0*pp(i,:);
    r(i,:) = r0;
    N(i,:) = N0;
end
```