% ========================================================================

%

% Simulating a model of heteroskedasticity

%

% ========================================================================

clear all;

clc;

RandStream.setDefaultStream( RandStream('mt19937ar','seed',123456) )

% Simulate the model

t = 5000;

alpha = 1;

beta = 2;

delta = 1;

gam = 0.5;

x = randn(t,1);

w = (0.0:0.1:0.1\*(t-1))'; % wt is a time trend

u = sqrt(delta + gam\*w).\*randn(t,1);

y = alpha + beta\*x + u;

%\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

%\*\*\*

%\*\*\* Generate graphs

%\*\*\*

%\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

% Switch off TeX interpreter and clear figure

set(0,'defaulttextinterpreter','none');

figure(1);

clf;

%--------------------------------------------------------%

% Panel (a)

subplot(2,2,1);

plot(x,y,'-k');

title('(a)');

xlabel('$x\_t$');

ylabel('$y\_t$');

box off;

%--------------------------------------------------------%

% Panel (b)

subplot(2,2,2);

plot(w,y,'-k');

title('(b)');

xlabel('$w\_t$');

ylabel('$y\_t$');

xlim([0,500]);

box off;

%--------------------------------------------------------%

% Panel (c)

subplot(2,2,3);

plot(w,y.^2,'-k');

title('(c)');

xlabel('$w\_t$');

ylabel('$y^{2}\_{t}$');

xlim([0,500]);

box off;

%--------------------------------------------------------%

% Panel (d)

bols = [ones(t,1) x]\y;

e = y - [ones(t,1) x]\*bols;

subplot(2,2,4);

plot(w,e.^2,'-k');

title('(d)');

xlabel('$x\_t$');

ylabel('$e^{2}\_{t}$');

xlim([0,500]);

box off;

%laprint(1,'heterosim','options','factory');

hetero estimate

% ========================================================================

%

% Estimating a model of heteroskedasticity

%

% ========================================================================

function hetero\_estimate( )

clear all;

clc;

RandStream.setDefaultStream( RandStream('mt19937ar','seed',123456) )

% Simulate the model

t = 500;

[y,x,w] = simulatedata( t );

% Estimate the unconstrained model

theta = [1; 2; 0.1; 0.1];

[theta1,l1,~,~,~,H1] = fminunc(@(b) neglog1(b,y,x,w),theta);

disp(['Log-likelihood (unconstrained) = ', num2str(-l1) ]);

disp( 'Unconstrained parameter estimates ' );

disp( [theta1 (1/t)\*sqrt( diag(inv(H1) ) ) ] );

% Estimate the constrained model

theta = [1; 2; 0.1 ];

[theta0,l0,~,~,~,H0] = fminunc(@(b) neglog0(b,y,x,w),theta);

disp(['Log-likelihood (constrained) = ', num2str(-l0) ]);

disp( 'Constrained parameter estimates ' );

disp( [theta0 (1/t)\*sqrt( diag(inv(H0) ) ) ] );

end

%

% -------------------------- Functions ---------------------------------

%

%-----------------------------------------------------------------------

% Simulate the data

%-----------------------------------------------------------------------

function [ y,x,w ] = simulatedata( t )

beta0 = 1;

beta1 = 2;

gam0 = 0.1;

gam1 = 0.1;

x = randn(t,1);

w = (0.0:0.1:0.1\*(t-1))';

u = sqrt(exp(gam0 + gam1\*w)).\*randn(t,1);

y = beta0 + beta1\*x + u;

end

%-----------------------------------------------------------------------

% Negative unconstrained log-likelihood

%-----------------------------------------------------------------------

function lf = neglog1(theta,y,x,w)

lf = -mean( lnlt1(theta,y,x,w) );

end

%-----------------------------------------------------------------------

% Unconstrained log-likelihood function at each observation

%-----------------------------------------------------------------------

function lnl = lnlt1(b,y,x,w)

mu = b(1) + b(2)\*x;

sig = sqrt( exp(b(3) + b(4)\*w) );

lnl = -(1/2)\*log(2\*pi\*sig.^2) - (y - mu).^2 ./(2\*sig.^2);

end

%-----------------------------------------------------------------------

% Negative constrained log-likelihood

%-----------------------------------------------------------------------

function lf = neglog0(theta,y,x,w)

lf = -mean( lnlt0(theta,y,x,w) );

end

%-----------------------------------------------------------------------

% Constrained log-likelihood function at each observation

%-----------------------------------------------------------------------

function lnl = lnlt0(b,y,x,w)

mu = b(1) + b(2)\*x;

sig = sqrt( exp(b(3) + 0\*w) );

lnl = -(1/2)\*log(2\*pi\*sig.^2) - (y - mu).^2 ./(2\*sig.^2);

end

hetero test

% ========================================================================

%

% Testing a model of heteroskedasticity

%

% ========================================================================

function hetero\_test( )

clear all;

clc;

RandStream.setDefaultStream( RandStream('mt19937ar','seed',123456) )

% Simulate the model

t = 500;

[y,x,w] = simulatedata( t );

% Estimate the unconstrained model

theta = [1; 2; 0.1; 0.1];

[theta1,l1,~,~,~,H1] = fminunc(@(b) neglog1(b,y,x,w),theta);

l1 = -l1;

disp(['Log-likelihood (unconstrained) = ', num2str(l1) ]);

disp( 'Unconstrained parameter estimates ' );

disp( [theta1 (1/t)\*sqrt( diag(inv(H1) ) ) ] );

% Estimate the constrained model

theta = [1; 2; 0.1 ];

[theta0,l0,~,~,~,H0] = fminunc(@(b) neglog0(b,y,x,w),theta);

l0 = -l0;

disp(['Log-likelihood (constrained) = ', num2str(l0) ]);

disp( 'Constrained parameter estimates ' );

disp( [theta0 (1/t)\*sqrt( diag(inv(H0) ) ) ] );

% LR test

lr = -2\*t\*(l0 - l1);

disp(['LR statistic = ',num2str(lr) ]);

disp(['p-value = ',num2str(1-cdf('chi2',lr,1)) ]);

% Wald test

r = [0 , 0 , 0 , 1];

q = 0;

wd = t\*(r\*theta1 - q)'\*inv(r\*inv(H1)\*r')\*(r\*theta1 - q);

disp(['Wald statistic = ',num2str(wd) ]);

disp(['p-value = ',num2str(1-cdf('chi2',wd,1)) ]);

% LM test

theta = [theta0 ; 0];

gmat = numgrad(@lnlt1,theta,y,x,w);

g = mean(gmat)';

j = gmat'\*gmat/t;

lm = t\*g'\*inv(j)\*g;

disp('Gradient evaluated at contrained estimates');

disp( g );

disp('Outer product of gradients matrix');

disp( j );

disp(['LM statistic = ',num2str(lm) ]);

disp(['p-value = ',num2str(1-cdf('chi2',lm,1)) ]);

% LM test (regression form)

x = [ones(t,1) , x];

% Stage 1 regression

b = x\y;

u = y - x\*b;

w = [ones(t,1) , w];

v = u.^2;

% Stage 2 regression

b = w\v;

e = v - w\*b;

r2 = 1 - sum(e.^2)/sum( (v-mean(v)).^2 );

lm = t\*r2;

disp(['LM statistic (regression) = ',num2str(lm) ]);

disp(['p-value = ',num2str(1-cdf('chi2',lm,1)) ]);

end

%

% -------------------------- Functions ---------------------------------

%

%-----------------------------------------------------------------------

% Simulate the data

%-----------------------------------------------------------------------

function [ y,x,w ] = simulatedata( t )

beta0 = 1;

beta1 = 2;

gam0 = 0.1;

gam1 = 0.1;

x = randn(t,1);

w = (0.0:0.1:0.1\*(t-1))';

u = sqrt(exp(gam0 + gam1\*w)).\*randn(t,1);

y = beta0 + beta1\*x + u;

end

%-----------------------------------------------------------------------

% Negative unconstrained log-likelihood

%-----------------------------------------------------------------------

function lf = neglog1(theta,y,x,w)

lf = -mean( lnlt1(theta,y,x,w) );

end

%-----------------------------------------------------------------------

% Unconstrained log-likelihood function at each observation

%-----------------------------------------------------------------------

function lnl = lnlt1(b,y,x,w)

mu = b(1) + b(2)\*x;

sig = sqrt( exp(b(3) + b(4)\*w) );

lnl = -(1/2)\*log(2\*pi\*sig.^2) - (y - mu).^2 ./(2\*sig.^2);

end

%-----------------------------------------------------------------------

% Negative constrained log-likelihood

%-----------------------------------------------------------------------

function lf = neglog0(theta,y,x,w)

lf = -mean( lnlt0(theta,y,x,w) );

end

%-----------------------------------------------------------------------

% Constrained log-likelihood function at each observation

%-----------------------------------------------------------------------

function lnl = lnlt0(b,y,x,w)

mu = b(1) + b(2)\*x;

sig = sqrt( exp(b(3) + 0\*w) );

lnl = -(1/2)\*log(2\*pi\*sig.^2) - (y - mu).^2 ./(2\*sig.^2);

end

% ========================================================================

%

% Program to estimate heteroskedastic models of the business cycle

%

% ========================================================================

function hetero\_g7( )

% Set country

country = 1; % 1 for Canada

% 2 for France

% 3 for Germany

% 4 for Italy

% 5 for Japan

% 6 for UK

% 7 for US

% Get the data

data = getData( country );

y = data(:,1); % Growth rate of GDP

x = data(:,2); % Spread

% Mean variables: lagged gdp growth and lagged interest spread

x = trimr([y x],0,1);

% Variance variables: lagged gdp growth and lagged interest spread

w = x;

y = trimr(y,1,0);

t = length(y);

% Estimate unrestriced model

start = 0.1\*ones(6,1);

ops = optimset('LargeScale', 'off', 'Display', 'final');

[bhat1,lf1] = fminunc(@(p) neglog(p,y,x,w),start,ops);

lf1 = -lf1;

% Estimate restriced model

start = 0.1\*ones(4,1);

[bhat0,lf0] = fminunc(@(p) neglog0(p,y,x),start,ops);

lf0 = -lf0;

% LR test

lr = -2\*t\*(lf0 - lf1);

dof = length(bhat1) - length(bhat0);

disp(' ')

disp(['Log-likelihood function (unrestricted) = ',num2str(lf1) ]);

disp(['Log-likelihood function (restricted) = ',num2str(lf0) ]);

disp(['LR statistic = ',num2str(lr) ]);

disp(['p-value = ',num2str(1-chi2cdf(lr,dof)) ]);

end

%

%--------------------------- Functions -----------------------------------

%

%-------------------------------------------------------------------------

% Unrestricted likelihood function

%-------------------------------------------------------------------------

function lf = neglog(p,y,x,w)

m = p(1) + p(2)\*x(:,1) + p(3)\*x(:,2);

s2 = exp(p(4) + p(5)\*w(:,1) + p(6)\*w(:,2));

lt = -0.5\*log(2\*pi) - 0.5\*log(s2) - 0.5\*((y - m).^2)./s2;

lf = - mean( lt );

end

%-------------------------------------------------------------------------

% Restricted likelihood function

%-------------------------------------------------------------------------

function lf = neglog0(p,y,x)

m = p(1) + p(2)\*x(:,1) + p(3)\*x(:,2);

s2 = exp(p(4));

lt = -0.5\*log(2\*pi) - 0.5\*log(s2) - 0.5\*((y - m).^2)./s2;

lf = - mean( lt );

end

%-------------------------------------------------------------------------

% Gets the data for a specfic country and change dates in specific cases

%-------------------------------------------------------------------------

function y = getData( flag )

if flag == 1 % Canada

y = xlsread('G7Data.xlsx','B8:C162');

y = trimr(y,0,19); % Change Canada dates: June 1961 to December 1999

elseif flag == 2 % France

y = xlsread('G7Data.xlsx','D43:E157');

elseif flag == 3 % Germany

y = xlsread('G7Data.xlsx','F4:G162');

elseif flag == 4 % Italy

y = xlsread('G7Data.xlsx','H48:I162');

elseif flag == 5 % Japan

y = xlsread('G7Data.xlsx','J43:K160');

elseif flag == 6 % UK

y = xlsread('G7Data.xlsx','L4:M162');

y = trimr(y,0,20); % Change UK dates: June 1960 to December 1999

else

y = xlsread('G7Data.xlsx','N4:O162');

y = trimr(y,0,20); % Change US dates: June 1960 to December 1999

end

end

%=========================================================================

%

% Program to estimate a simultaneous model with vector heteroskedasticity

% The system is defined as yt\*b + xt\*a = u

% where

% u ~ N(0,V)

% and

% yt is a (1xn) set of dependent variables at time t

% xt is a (1xm) set of mean explanatory variables at time t

% wt is a (1xs) set of variance explanatory variables at time t

%

%=========================================================================

function hetero\_system( )

clear all;

clc;

RandStream.setDefaultStream( RandStream('mt19937ar','seed',123456) )

% Simulate the model

t = 2000;

[y,x,w] = simulatedata( t );

% Estimate the model

theta0 = [ 0.6; 0.4; 0.2;-0.5; 1.0; 0.5; 0.5; 0.2; 2.0; 0.2 ];

[theta,a,~,~,~,H] = fminunc(@(theta) neglog(theta,y,x,w),theta0);

vcov = inv(H);

disp(['Log-likelihood function = ',num2str(-a) ]);

disp( 'Parameter estimates and standard errors' );

disp([ theta sqrt(diag(vcov)) ]);

% Wald test of no vector heteroskedasticty

r = [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 ];

q = [0 ; 0 ; 0 ];

wd = t\*(r\*theta - q)'\*inv(r\*vcov\*r')\*(r\*theta - q);

dof = size(r,1);

disp(['Wald statistic = ',num2str(wd) ]);

disp(['p-value = ',num2str(1-cdf('chi2',wd,dof)) ]);

end

%

% -------------------------- Functions ---------------------------------

%

%-----------------------------------------------------------------------

% Simulate the data

%-----------------------------------------------------------------------

function [ y,x,w ] = simulatedata( t )

beta1 = 0.6;

alpha1 = 0.4;

beta2 = 0.2;

alpha2 = -0.5;

c11 = 1.0;

c21 = 0.5;

c22 = 2.0;

d11 = 0.5;

d21 = 0.2;

d22 = 0.2;

b = [ 1 , -beta2 ;

-beta1 , 1 ];

a = [-alpha1 , 0 ;

0 , -alpha2 ];

c = [c11 , 0 ;

c21 , c22 ];

d = [d11 , 0 ;

d21 , d22 ] ;

% Exogenous variables

x = [10\*rand(t,1) , 3\*randn(t,1)];

w = rand(t,1);

% Disturbances

u = zeros(t,2);

for i = 1:t

l = c + d\*w(i);

u(i,:) = randn(1,2)\*l';

end

% Simulate the reduced form

y = zeros(t,2);

for i = 1:t

y(i,:) = -x(i,:)\*a\*inv(b) + u(i,:)\*inv(b);

end

end

%-----------------------------------------------------------------------

% Negative unconstrained log-likelihood

%-----------------------------------------------------------------------

function lf = neglog(theta,y,x,w)

lf = -mean( lnlt(theta,y,x,w) );

end

%-----------------------------------------------------------------------

% Unconstrained log-likelihood function at each observation

%-----------------------------------------------------------------------

function lnl = lnlt(theta,y,x,w)

t = size(y,1);

n = size(y,2);

b = [ 1 , -theta(3) ;

-theta(1) , 1 ];

a = [-theta(2) , 0 ;

0 , -theta(4)] ;

c = [theta(5) , 0 ;

theta(7) , theta(9) ];

d = [theta(6) , 0 ;

theta(8) , theta(10)];

u = zeros(t,n);

lnl = zeros(t,1);

for i=1:t

u(i,:) = y(i,:)\*b + x(i,:)\*a;

l = c + d\*w(i);

V = l\*l';

lnl(i) = - n\*0.5\*log(2\*pi) + log(abs(det(b))) - 0.5\*log(det(V)) ...

- 0.5\*u(i,:)\*inv(V)\*u(i,:)';

end

end

%=========================================================================

%

% A heteroskedastic model of money shocks in US asset markets

%

%=========================================================================

function hetero\_event( )

clear all

clc

% Load data

load usasset.mat

% 1. tcm3m (from FRB webpage)

% 2. tcm6m

% 3. tcm1y

% 4. tcm2y

% 5. tcm3y

% 6. tcm5y

% 7. tcm7y

% 8. tcm10y

% 9. edm1m

% 10. xrate ($US/$AU)

% 11. frb target in US

% 12. change in target variable US

% 13. dummy variable corresponding to event dates US (taken from Poole and Kuttner)

% 14. value-weighted returns in US (from CRSP)

% 15. value-weighted index US (from CRSP)

% Different to Poole dating

ydata(1274,13) = 1.0;

ydata(1275,13) = 0.0;

% Define variables

yield = ydata(:,[1 2 3 4 5 6 7 8]); % 3m, 6m 1y, 2y, 3y, 5y, 7y, 10y as percentages

euro = ydata(:,9); % 1m euro rate as percentage

event = ydata(:,13); % dummy variable on meeting dates

nyse = ydata(:,15); % NYSE equity price index

% Dependent variables

dyield = (trimr(yield,1,0) - trimr(yield,0,1));

deuro = trimr(euro,1,0) - trimr(euro,0,1);

rnyse = 100\*(trimr(log(nyse),1,0) - trimr(log(nyse),0,1));

y = dyield(:,1);

d = trimr(event,1,0); % Event day

x = [ones(length(y),1) d];

t = length(y);

% Compute descriptive statistics on non-event and event days

y\_nonevent = y(d == 0);

y\_event = y(d == 1);

disp(['Sample mean (event) = ',num2str(mean(y\_event)) ]);

disp(['Sample mean (non-event) = ',num2str(mean(y\_nonevent)) ]);

disp(['Sample variance (event) = ',num2str(mean((y\_event-mean(y\_event)).^2)) ]);

disp(['Sample variance (non-event) = ',num2str(mean((y\_nonevent-mean(y\_nonevent)).^2)) ]);

% Estimate the unrestricted model by MLE with starting values based on OLS

bols = x\y;

u = y - x\*bols;

s2 = mean(u.^2);

start = [ bols ; s2 ; 0 ];

ops = optimset('LargeScale','off','Display','iter');

[bhat1,lf1,~,~,~,hess] = fminunc(@(b) neglog1(b,y,d),start,ops);

lf1 = -lf1;

vc1 = (1/t)\*inv(hess);

disp(' ');

disp('Unrestricted model')

disp(['MLE estimate of the mean (event days) = ',num2str((bhat1(1) + bhat1(2))) ]);

disp(['MLE estimate of the mean (non-event days) = ',num2str(bhat1(1)) ]);

disp(['MLE estimate of the volatility (event days) = ',num2str(exp(bhat1(3) + bhat1(4))) ]);

disp(['MLE estimate of the volatility (non-event days) = ',num2str(exp(bhat1(3))) ]);

disp(' ');

% Estimate the restricted model by MLE with starting values based on OLS

start = [ bols ; s2 ];

ops = optimset('LargeScale','off','Display','iter');

[bhat0,lf0,~,~,~,hess] = fminunc(@(b) neglog0(b,y,d),start,ops);

lf0 = -lf0;

vc0 = (1/t)\*inv(hess);

disp(' ');

disp('Restricted model')

disp(['MLE estimate of the mean (event days) = ',num2str((bhat0(1) + bhat0(2))) ]);

disp(['MLE estimate of the mean (non-event days) = ',num2str(bhat0(1)) ]);

disp(['MLE estimate of the volatility (event days) = ',num2str(exp(bhat0(3))) ]);

disp(['MLE estimate of the volatility (non-event days) = ',num2str(exp(bhat0(3))) ]);

disp(' ');

% LR test

lr = -2\*t\*(lf0 - lf1);

disp(['LR statistic = ',num2str(lr) ]);

disp(['p-value = ',num2str(1-cdf('chi2',lr,1)) ]);

% Wald test

r = [0 , 0 , 0 , 1];

q = 0;

wd = (r\*bhat1 - q)'\*inv(r\*vc1\*r')\*(r\*bhat1 - q);

disp(['Wald statistic = ',num2str(wd) ]);

disp(['p-value = ',num2str(1-cdf('chi2',wd,1)) ]);

% LM test (regression form)

% Stage 1 regression

b = x\y;

u = y - x\*b;

w = x;

v = u.^2;

% Stage 2 regression

b = w\v;

e = v - w\*b;

r2 = 1 - sum(e.^2)/sum( (v-mean(v)).^2 );

lm = t\*r2;

disp(['LM statistic (regression) = ',num2str(lm) ]);

disp(['p-value = ',num2str(1-cdf('chi2',lm,1)) ]);

end

%--------------------------- Functions -----------------------------------

%

%-------------------------------------------------------------------------

% Unrestricted log-likelihood function

%-------------------------------------------------------------------------

function lf = neglog1(b,y,d)

m = b(1) + b(2)\*d; % Mean

s2 = exp(b(3) + b(4)\*d); % Variance

lt = -0.5\*log(2\*pi) -0.5\*log(s2) - 0.5\*((y - m).^2)./s2;

lf = -mean(lt);

end

%-------------------------------------------------------------------------

% Restricted log-likelihood function

%-------------------------------------------------------------------------

function lf = neglog0(b,y,d)

m = b(1) + b(2)\*d; % Mean

s2 = exp(b(3)); % Variance

lt = -0.5\*log(2\*pi) -0.5\*log(s2) - 0.5\*((y - m).^2)./s2;

lf = -mean(lt);

end

%

%

%

%

% /\*\* Define the log of the likelihood at each observation \*\*/

%

%

%

%

%=========================================================================

%

% Program to estimate a simultaneous model with vector hetero-

% skedasticity and first order autocorrelation

% The system is defined as yt\*b + xt\*a = u

% where

% u = ru(-1) + v

% and yt is a (1xn) set of dependent variables at time t

% xt is a (1xk) set of explanatory variables at time t

% wt is a (1xs) set of variance explanatory variables at time t

%

%=========================================================================

function hetero\_general()

clear all;

clc;

RandStream.setDefaultStream( RandStream('mt19937ar','seed',123456) )

% Simulate the model

t = 2000;

[y,x,w] = simulatedata( t );

% Estimate the model

theta0 = [ 0.6; 0.4; 0.2;-0.5; ...

1.0; 0.5; 0.5; ...

0.2; 2.0; 0.2; ...

0.8; 0.1;-0.2; 0.6];

[theta,a,~,~,~,H] = fminunc(@(theta) neglog(theta,y,x,w),theta0);

vcov = inv(H);

disp(['Log-likelihood function = ',num2str(-a) ]);

disp( 'Parameter estimates and standard errors' );

disp([ theta sqrt(diag(vcov)) ]);

% Wald test of no vector heteroskedasticty and no autocorrelation

r = [0 , 0 , 0 , 0 , 0 , 1 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 ;

0 , 0 , 0 , 0 , 0 , 0 , 0 , 1 , 0 , 0 , 0 , 0 , 0 , 0 ;

0 , 0 , 0 , 0 , 0 , 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 , 0 , 0 , 0 , 0 , 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 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 1 ];

q = [0 ; 0 ; 0 ; 0 ; 0 ; 0 ; 0 ];

wd = t\*(r\*theta - q)'\*inv(r\*vcov\*r')\*(r\*theta - q);

dof = size(r,1);

disp(['Wald statistic = ',num2str(wd) ]);

disp(['p-value = ',num2str(1-cdf('chi2',wd,dof)) ]);

end

%

% -------------------------- Functions ---------------------------------

%

%-----------------------------------------------------------------------

% Simulate the data

%-----------------------------------------------------------------------

function [ y,x,w ] = simulatedata( t )

beta1 = 0.6;

alpha1 = 0.4;

beta2 = 0.2;

alpha2 = -0.5;

c11 = 1.0;

c21 = 0.5;

c22 = 2.0;

d11 = 0.5;

d21 = 0.2;

d22 = 0.2;

rho11 = 0.8;

rho12 = 0.1;

rho21 = -0.2;

rho22 = 0.6;

b = [ 1 , -beta2 ;

-beta1 , 1 ];

a = [-alpha1 , 0 ;

0 , -alpha2 ];

c = [c11 , 0 ;

c21 , c22 ];

d = [d11 , 0 ;

d21 , d22 ] ;

% Exogenous variables

x = [10\*rand(t,1) , 3\*randn(t,1)];

w = rand(t,1);

% Disturbances

u = zeros(t,2);

v = zeros(t,2);

for i = 2:t

l = c + d\*w(i);

v(i,:) = randn(1,2)\*l';

u(i,1) = rho11\*u(i-1,1) + rho12\*u(i-1,2) + v(i,1);

u(i,2) = rho21\*u(i-1,1) + rho22\*u(i-1,2) + v(i,2);

end

% Simulate the reduced form

y = zeros(t,2);

for i = 1:t

y(i,:) = -x(i,:)\*a\*inv(b) + u(i,:)\*inv(b);

end

end

%-----------------------------------------------------------------------

% Negative unconstrained log-likelihood

%-----------------------------------------------------------------------

function lf = neglog(theta,y,x,w)

lf = -mean( lnlt(theta,y,x,w) );

end

%-----------------------------------------------------------------------

% Unconstrained log-likelihood function at each observation

%-----------------------------------------------------------------------

function lf = lnlt(theta,y,x,w)

t = size(y,1);

n = size(y,2);

b = [ 1 , -theta(3) ;

-theta(1) , 1 ];

a = [-theta(2) , 0 ;

0 , -theta(4)] ;

c = [theta(5) , 0 ;

theta(7) , theta(9) ];

d = [theta(6) , 0 ;

theta(8) , theta(10)];

rho = [theta(11) , theta(13) ;

theta(12) , theta(14) ];

u = zeros(t,n);

v = zeros(t,n);

lnl = zeros(t,1);

for i=2:t

u(i,:) = y(i,:)\*b + x(i,:)\*a;

v(i,:) = u(i,:) - u(i-1,:)\*rho;

l = c + d\*w(i);

V = l\*l';

lnl(i) = - n\*0.5\*log(2\*pi) + log(abs(det(b))) - 0.5\*log(det(V)) ...

- 0.5\*v(i,:)\*inv(V)\*v(i,:)';

end

lf = trimr(lnl,1,0);

end

%=========================================================================

%

% Program to test the "Great Moderation" hypothesis, using

% real annual US GDP data per capita from 1946 to 2006.

%

%=========================================================================

function hetero\_moderation( )

clear all;

clc;

gdp = [ 11241;

10924;

11206;

10957;

11717;

12412;

12668;

13032;

12719;

13389;

13410;

13435;

13088;

13783;

13840;

13933;

14552;

14971;

15624;

16420;

17290;

17532;

18196;

18573;

18392;

18771;

19555;

20485;

20195;

19961;

20822;

21565;

22526;

22982;

22666;

23007;

22347;

23146;

24593;

25382;

26024;

26664;

27514;

28221;

28429;

28007;

28556;

28941;

29741;

30128;

30880;

31886;

32833;

33904;

34755;

34645;

34837;

35361;

36300;

37052;

37752 ];

% Compute growth rate

y = 100\*( log(gdp(2:end)) - log(gdp(1:end-1)) );

%\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

%\*\*

%\*\* Generate graph

%\*\*

%\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

% Switch off TeX interpreter and clear figure

set(0,'defaulttextinterpreter','none');

figure(1);

clf;

plot(1947:1:2006,y,'-k')

xlabel('Years');

ylabel('Growth Rate of US Per Capital GDP');

axis tight;

box off;

% Generate descriptive statistics

disp(['Sample mean (1947-1983) = ', num2str(mean(y(1:37))) ]);

disp(['Sample mean (1984-2006) = ', num2str(mean(y(38:60))) ]);

disp(['Sample variance (1984-2006) = ', num2str(mean((y(1:37)-mean(y(1:37))).^2)) ]);

disp(['Sample variance (1984-2006) = ', num2str(mean((y(38:60)-mean(y(38:60))).^2)) ]);

d = [zeros(37,1) ; ones(23,1)]; % Construct dummy variable

t = length(y);

% Estimate the unconstrained model

theta = 0.1\*ones(4,1);

flag = 1;

[theta1,a1,a,a,a,h1] = fminunc(@(b) neglog(b,y,d,flag),theta);

omega1 = inv(h1);

lnl1 = -a1;

disp(['ML estimate of the variance (1946-1983) = ',num2str(exp(theta1(3))) ]);

disp(['ML estimate of the variance (1984-2006) = ',num2str(exp(theta1(3) + theta1(4))) ]);

% Estimate the constrained model

theta = 0.1\*ones(3,1);

[aaa,a0] = fminunc(@(b) neglog0(b,y,d,flag),theta);

lnl0=-a0;

% LR test for heteroskedasticity

lr = -2\*t\*(lnl0 - lnl1);

disp(['LR statistic = ',num2str(lr) ]);

disp(['p-value = ',num2str(1-cdf('chi2',lr,1)) ]);

% Wald test for heteroskedasticity

r = [0 , 0 , 0 , 1];

q = 0;

wd = t\*(r\*theta1 - q)'\*inv(r\*omega1\*r')\*(r\*theta1 - q);

disp(['Wald statistic = ',num2str(wd) ]);

disp(['p-value = ',num2str(1-cdf('chi2',wd,1)) ]);

% LM test (regression form)

x = [ones(t,1) , d];

% Stage 1 regression

b = x\y;

u = y - x\*b;

w = [ones(t,1) , d];

v = u.^2;

% Stage 2 regression

b = w\v;

e = v - w\*b;

r2 = 1 - sum(e.^2)/sum( (v-mean(v)).^2 );

lm = t\*r2;

disp(['LM statistic = ',num2str(lm) ]);

disp(['p-value = ',num2str(1-cdf('chi2',lm,1)) ]);

% Wald test that beta1 = 0

r = [0 , 1 , 0 , 0];

q = 0;

wd = (r\*theta1 - q)'\*inv(r\*omega1\*r')\*(r\*theta1 - q);

disp(' ')

disp(['Wald statistic of beta1 = 0 = ',num2str(wd) ]);

disp(['p-value = ',num2str(1 - cdf('chi2',wd,1)) ]);

%-----------------------------------------------------------------------

% Tests based on the assumption that beta1 = 0.

%-----------------------------------------------------------------------

% Estimate the unconstrained model but with beta1 = 0

theta = [0.1 ; 0.1 ; 0.1];

flag = 0;

[theta1,a1,a,a,a,h1] = fminunc(@(b) neglog(b,y,d,flag),theta);

omega1 = inv(h1);

lnl1 = -a1;

% Estimate the constrained model but with beta1 = 0

theta = [0.1 ; 0.1 ];

[aa,a0] = fminunc(@(b) neglog0(b,y,d,flag),theta);

omega1 = inv(h1);

lnl0=-a0;

% LR test for heteroskedasticity

lr = -2\*t\*(lnl0 - lnl1);

disp(['LR statistic = ',num2str(lr) ]);

disp(['p-value = ',num2str(1-cdf('chi2',lr,1)) ]);

% Wald test

r = [0 , 0 , 1];

q = 0;

wd = t\*(r\*theta1 - q)'\*inv(r\*omega1\*r')\*(r\*theta1 - q);

disp(['Wald statistic = ',num2str(wd) ]);

disp(['p-value = ',num2str(1-cdf('chi2',wd,1)) ]);

% LM test (regression form)

x = ones(t,1);

% Stage 1 regression

b = x\y;

u = y - x\*b;

w = [ones(t,1) , d];

v = u.^2;

% Stage 2 regression

b = w\v;

e = v - w\*b;

r2 = 1 - sum(e.^2)/sum( (v-mean(v)).^2 );

lm = t\*r2;

disp(['LM statistic = ',num2str(lm) ]);

disp(['p-value = ',num2str(1-cdf('chi2',lm,1)) ]);

end

%

%-------------------------Functions------------------------------------

%

%-----------------------------------------------------------------------

% Negative log-likelihood function (unconstrained)

%-----------------------------------------------------------------------

function lf = neglog( b,y,x,flag )

if flag

mu = b(1) + b(2)\*x;

sig2 = exp(b(3) + b(4)\*x);

else

mu = b(1);

sig2 = exp(b(2) + b(3)\*x);

end

lnl = -(1/2)\*log(2\*pi\*sig2) - (y - mu).^2 ./(2\*sig2);

lf = -mean( lnl );

end

%-----------------------------------------------------------------------

% Negative log-likelihood function (constrained)

%-----------------------------------------------------------------------

function lf = neglog0( b,y,x,flag )

if flag

mu = b(1) + b(2)\*x;

sig2 = exp(b(3) + 0\*x);

else

mu = b(1);

sig2 = exp(b(2) + 0\*x);

end

lnl = -(1/2)\*log(2\*pi\*sig2) - (y - mu).^2 ./(2\*sig2);

lf = -mean( lnl );

end

% Define the log of the likelihood for model with beta1 = 0 (unconstrained)

function lnl\_t = lnltbo(b,d,y)

muet = b(1) + 0\*d;

sigt = sqrt(exp(b(3) + b(4)\*d));

lnl\_t = -(1/2)\*log(2\*pi\*sigt.^2) - (y - muet).^2 ./(2\*sigt.^2); % Log like at each obs

end

% Define the constrained log of the likelihood for model with beta1 = 0

%

function lnl\_t = lnltbo0(b,d,y);

muet = b(1) + 0\*d;

sigt = sqrt(exp(b(3) + 0\*d));

lnl\_t = -(1/2)\*log(2\*pi\*sigt.^2) - (y - muet).^2 ./(2\*sigt.^2); % Log like at each obs

end