Problem 1

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

%

% Generate simulated data by simulating the reduced form.

% The set of equations is given by the bivariate system

% y1t = beta1\*y2t + alpha1\*x1t + u1t

% y2t = beta2\*y1t + alpha2\*x2t + u2t

% where E[ut'ut] = omega

%

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

clear all;

clc;

RandStream.setDefaultStream( RandStream('mt19937ar','seed',123457) )

t = 500;

beta1 = 0.6;

beta2 = 0.2;

alpha1 = 0.4;

alpha2 = -0.5;

omega = [1 0.5; 0.5 1.0];

% Construct population parameter matrices

B = [1 -beta2; -beta1 1];

A = [-alpha1 0; 0 -alpha2];

% Construct exogenous variables

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

% Construct structural disturbances

u = randn(t,2)\*chol(omega);

% Construct reduced form parameters

invB = inv(B);

phi = -A\*invB;

disp('Inverse of b');

disp(invB);

disp('Reduced form parameter matrix');

disp(phi);

% Construct reduced form disturbances

v = u\*invB;

% Simulate the model by simulating the reduced form

y = zeros(t,2);

for i=1:t

y(i,:) = -x(i,:)\*A\*invB + v(i,:);

end

% \*\*\* Generate graphs \*\*\*

% Switch off TeX interpreter and clear figure

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

figure(1);

clf;

t = 1:1:500;

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

% Panel (a)

subplot(2,2,1);

plot(t,y(:,1),'-k');

title('(a)');

xlabel('$t$');

ylabel('$y\_{1,t}$');

set(gca,'XTick',0:100:500);

set(gca,'YTick',-10:5:10);

xlim([0,500]);

ylim([-10,10]);

box off;

%set(gca,'LineWidth',1);

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

% Panel (b)

subplot(2,2,2);

plot(t,y(:,2),'-k');

title('(b)');

xlabel('$x\_t$');

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

set(gca,'XTick',0:100:500);

set(gca,'YTick',-10:5:10);

xlim([0,500]);

ylim([-10,10]);

box off;

%set(gca,'LineWidth',1);

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

% Panel (c)

subplot(2,2,3);

plot3(x(:,1),y(:,1),y(:,2),'.k','MarkerSize',5);

title('(c)');

xlabel('$x\_{1,t}$');

ylabel('$y\_{1,t}$');

zlabel('$y\_{2,t}$');

set(gca,'XTick',-10:5:10);

set(gca,'YTick',-10:5:10);

xlim([-10,10]);

ylim([-10,10]);

%set(gca,'LineWidth',1);

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

% Panel (d)

subplot(2,2,4);

plot3(x(:,2),y(:,2),y(:,1),'.k','MarkerSize',5);

title('(d)');

xlabel('$x\_{2,t}$');

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

zlabel('$y\_{1,t}$');

set(gca,'XTick',-10:5:10);

set(gca,'YTick',-10:5:10);

xlim([-10,10]);

ylim([-10,10]);

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

Problem 2

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

%

% Program to estimate linear regression by maximum likelihood.

%

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

function linear\_estimate( )

clear all;

clc;

RandStream.setDefaultStream( RandStream('mt19937ar','seed',123) )

t = 200;

% Parameter values

beta0 = 1.0;

beta1 = 0.7;

beta2 = 0.3;

sig = sqrt(4.0);

% Simulate the data

x1 = randn(t,1);

x2 = randn(t,1);

u = sig\*randn(t,1);

y = beta0 + beta1\*x1 + beta2\*x2 + u;

x = [ones(t,1) x1 x2];

data = [y x1 x2 ];

% Estimate the model

theta\_0 = rand(4,1); % Initial guess

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

theta = fminunc(@(theta) neglog(theta,y,x1,x2),theta\_0,options);

disp( 'Estimated Parameters')

disp( theta )

ht = numhess(@neglog,theta,y,x1,x2);

disp( 'Estimated covariance matrix');

disp((1/t)\*inv(ht));

% Estimate the concentrated model

theta = fminunc(@(theta) neglogc(theta,y,x1,x2),rand(3,1));

disp( 'Estimated Parameters (Concentrated)')

disp( theta )

htc = numhess(@neglogc,theta,y,x1,x2);

disp( 'Estimated covariance matrix (concentrated)');

disp((1/t)\*inv(htc));

%Compute OLS estimates

theta = x\y;

disp('OLS Parameter estimates')

disp( theta );

% Compute the covariance matrices

e = y - x\*theta;

sig2hat = e'\*e/t;

vcov = sig2hat\*inv(x'\*x);

disp('Covariance matrix (OLS)');

disp( vcov );

disp(['Variance estimate (OLS) = ', num2str(sig2hat) ] );

disp(['Variance estimate standard error = ', num2str(2\*sig2hat^2/t) ]);

end

%

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

%

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

% Log-likelihood function

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

function lf = neglog(theta,y,x1,x2)

lf = -mean( lnlt(theta,y,x1,x2) );

end

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

% Log-likelihood function concentrated

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

function lf = neglogc(theta,y,x1,x2)

lf = -mean( lnltc(theta,y,x1,x2) );

end

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

% Log-likelihood function at each observation

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

function lf = lnlt(theta,y,x1,x2)

m = theta(1) + theta(2)\*x1 + theta(3)\*x2;

s2 = theta(4);

z = (y - m)/sqrt(s2);

lf = -0.5\*log(2\*pi) - 0.5\*log(s2) - 0.5\*z.^2;

end

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

% Concentrated Log-likelihood function at each observation

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

function lf = lnltc(theta,y,x1,x2)

m = theta(1) + theta(2)\*x1 + theta(3)\*x2;

u = y - m;

s2 = u'\*u/length(y);

z = (y - m)/sqrt(s2);

lf = -0.5\*log(2\*pi) - 0.5\*log(s2) - 0.5\*z.^2;

end

Problem 3

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

%

% Program to estimate model by full information maximum likelihood.

%

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

function linear\_fiml( )

clear all;

clc;

t = 500;

%flag = 1; % 1 = simulate data

flag = 0; % 0 = use GAUSS data to reproduce text

if flag

[ y,x ] = simulatedata( t );

else

% Aternatively load the GAUSS data

load linear\_fimldata.dat

y = linear\_fimldata(:,[1 2]);

x = linear\_fimldata(:,[3 4]);

end

% Estimate the model with random starting values

theta = fminunc(@(theta) neglog(theta,y,x),rand(4,1));

ht = numhess(@neglog,theta,y,x);

cov = (1/t)\*inv(ht);

u = zeros(t,2);

for i=1:t

u(i,:) = y(i,:)\*[1 -theta(3); -theta(1) 1] + x(i,:)\*[-theta(2) 0; 0 -theta(4)];

end

disp(['Beta 1 and se = ', num2str(theta(1)),' ', num2str(sqrt(cov(1,1))) ]);

disp(['Beta 2 and se = ', num2str(theta(2)),' ', num2str(sqrt(cov(2,2)))]);

disp(['Beta 3 and se = ', num2str(theta(3)),' ', num2str(sqrt(cov(3,3)))]);

disp(['Beta 4 and se = ', num2str(theta(4)),' ', num2str(sqrt(cov(4,4)))]);

disp('Residual variance-covariance matrix')

disp(u'\*u/t);

% Instrumental variable estimation

beta1\_iv = iv( y(:,1), [y(:,2) x(:,1)], x(:,[1 2]) );

disp(['IV estimate of beta1 = ', num2str(beta1\_iv(1))]);

disp(['IV estimate of alpha1 = ', num2str(beta1\_iv(2))]);

iv( y(:,2), [y(:,1) x(:,2)], x(:,[1 2]) );

beta2\_iv = iv( y(:,2), [y(:,1) x(:,2)], x(:,[1 2]) );

disp(['IV estimate of beta2 = ', num2str(beta2\_iv(1))]);

disp(['IV estimate of alpha2 = ', num2str(beta2\_iv(2))]);

end

%

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

%

% Simulate data

function [ y,x ] = simulatedata( t )

RandStream.setDefaultStream( RandStream('mt19937ar','seed',123457) );

beta1 = 0.6;

alpha1 = 0.4;

beta2 = 0.2;

alpha2 = -0.5;

omega = [1 0.5; 0.5 1.0];

% Construct population parameter matrices

b = [1 -beta2; -beta1 1];

a = [-alpha1 0; 0 -alpha2];

% Construct exogenous variables

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

% Construct disturbances

u = randn(t,2)\*chol(omega);

% Simulate the model by simulating the reduced form

y = zeros(t,2);

for i=1:t

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

end

end

% Log-likelihood function

function lf = neglog(theta,y,x)

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

end

% Log-likelihood function at each observation

function rval = lnlt(theta,y,x)

[t n] = size(y);

b = [1 -theta(3); -theta(1) 1];

a = [-theta(2) 0; 0 -theta(4)];

u = zeros(t,n);

for i=1:t

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

end

omega = u'\*u/t; % Concentrate out resid var-covar matrix

lnl = zeros(t,1);

for i=1:t

lnl(i) = -n\*0.5\*log(2\*pi) + log(abs(det(b))) - 0.5\*log(det(omega)) - 0.5\*u(i,:)\*inv(omega)\*u(i,:)';

end

rval = lnl;

end

% Instrumental variable estimation

function b = iv(y,w,x)

% IV estimates

b = inv(w'\*x\*inv(x'\*x)\*x'\*w)\*(w'\*x\*inv(x'\*x)\*x'\*y);

% % Standard error of regression

e = y - w\*b;

k = size(w,2);

t = size(y,1);

sigma = sqrt( e'\*e/t );

% Variance-covariance matrix

vcov = sigma^2\*inv(w'\*x\*inv(x'\*x)\*x'\*w);

sterr = sqrt( diag(vcov) );

tstats = b./sterr;

end

Problem 4

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

%\*\*

%\*\* Program to generate Figure 1a of Stock, Wright and Yogo

%\*\* weak instrument paper (JBES, 2002)

%\*\*

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

clear all;

clc;

state = 1234;

rand('twister', state);

randn('state', state);

N = 10000; % Number of replications

T = 5; % Sample size

beta = 0.0; % Parameter values

phi = 0.25;

sig11 = 1.0;

sig22 = 1.0;

sig12 = 0.99;

omega = [sig11 sig12 ; sig12 sig22];

rho = sig12/sqrt(sig11\*sig22);

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

%\*\*

%\*\* Generate instruments (scaled) and do Monte Carlo replications

%\*\*

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

x = rand(T,1);

%x = x/sqrt(x'\*x);

biv = zeros(N,1);

for i = 1:N

%u = mvnrnd([0 0],omega,T);

u = randn(T,length(omega))\*chol(omega);

w = x\*phi + u(:,1); % Reduced form equation

y = w\*beta + u(:,2); % Structural equation

tmp = inv(x'\*x);

tmp1 = x\*tmp\*x';

tmp2 = w'\*tmp1;

biv(i) = inv(tmp2\*w)\*(tmp2\*y); % IV estimates

end

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

%\*\*

%\*\* Generate graph

%\*\*

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

xi = -2.0:0.001:2.0;

f = ksdensity(biv,xi);

% Switch off TeX interpreter and clear figure

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

figure(1);

clf;

plot(xi,f,'-k');

xlabel('$\beta\_{\text{IV}}$');

ylabel('$f(\beta\_{\text{IV}})$');

box off;

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

Problem 5

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

%

% Program to demonstrate the distribution of the

% estimates of a gamma regression model

%

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

clear all

clc

ndraws = 5000;

% Parameters

b0 = [ 1 2 ];

rho = 0.25;

alpha = 0.1;

% For the sample size of T = 10

RandStream.setDefaultStream( RandStream('mt19937ar','seed',1) );

t = 10;

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

z1 = zeros(ndraws,1);

z2 = zeros(ndraws,1);

for i = 1:ndraws

u = alpha\*gamrnd(rho,1,[t 1]) - rho\*alpha;

y = x\*b0' + u;

b = x\y;

e = y - x\*b;

s2 = e'\*e/t;

vc = s2\*inv(x'\*x);

z1(i) = (b(1) - b0(1))/sqrt(vc(1,1));

z2(i) = (b(2) - b0(2))/sqrt(vc(2,2));

end

% For the sample size of T = 100

RandStream.setDefaultStream( RandStream('mt19937ar','seed',1) );

t = 100;

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

z3 = zeros(ndraws,1);

z4 = zeros(ndraws,1);

for i = 1:ndraws

u = alpha\*gamrnd(rho,1,[t 1]) - rho\*alpha;

y = x\*b0' + u;

b = x\y;

e = y - x\*b;

s2 = e'\*e/t;

vc = s2\*inv(x'\*x);

z3(i) = (b(1) - b0(1))/sqrt(vc(1,1));

z4(i) = (b(2) - b0(2))/sqrt(vc(2,2));

end

% Plot the results

bin = -5.25:0.5:4.75;

figure(1)

subplot(2,2,1)

hist(z1,bin);

title('(a) T=10, $\beta\_0$')

axis([-5 5 0 1200])

subplot(2,2,2)

hist(z2,bin);

title('(a) T=10, $\beta\_1$')

subplot(2,2,3)

hist(z3,bin);

title('(a) T=100, $\beta\_0$')

subplot(2,2,4)

hist(z4,bin);

title('(a) T=100, $\beta\_1$')

Problem 6

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

%

% Program to estimate a model by full information maximum likelihood

% and instrumental variables. The set of equations is given by the

% bivariate system

%

% y1t = beta\*y2t + u1t

% y2t = gam\*y1t + alpha\*xt + u2t

%

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

function linear\_iv( )

clear all;

clc;

% Get data

t = 500;

[ y,x ] = simulatedata( t );

% Estimate the model using random starting values

theta = fminunc( @(theta) neglog(theta,y,x),rand(3,1) );

disp(' MLE Estimates ' );

disp( theta );

% Construct residuals and estimate covariance matrix

[t n] = size(y);

b = [1 -theta(2); -theta(1) 1];

a = [0 -theta(3)];

e = zeros(t,n);

for i=1:t

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

end

me2 = mean(e.^2);

drv = eye(n);

omega = diag( me2 );

disp('Covariance matrix');

disp( omega );

% FIML analytical solution

beta\_fiml = sum(y(:,1).\*x) / sum(y(:,2).\*x);

e1 = y(:,1) - beta\_fiml\*y(:,2);

num = sum(y(:,2).\*e1)\*sum(x.^2) - sum(x.\*e1)\*sum(y(:,2).\*x);

den = sum(y(:,1).\*e1)\*sum(x.^2) - sum(x.\*e1)\*sum(y(:,1).\*x);

gam\_fiml = num/den;

num = sum(y(:,1).\*e1)\*sum(y(:,2).\*x) - sum(y(:,1).\*x)\*sum(y(:,2).\*e1);

alpha\_fiml = num/den;

disp('Closed form FIML estimates')

disp( [beta\_fiml; gam\_fiml; alpha\_fiml ] );

% Instrumental variable estimation

beta1\_iv = iv(y(:,1), y(:,2), x);

beta2\_iv = iv(y(:,2), [y(:,1) x], [e1 x]);

disp('Instrumental variables estimates')

disp( [beta1\_iv; beta2\_iv(1); beta2\_iv(2) ] );

end

%

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

%

% Simulate data

function [ y,x ] = simulatedata( t )

RandStream.setDefaultStream( RandStream('mt19937ar','seed',123457) );

beta = 0.6;

gam = 0.4;

alpha = -0.5;

omega = [2.0 0.0; 0.0 1.0];

% Construct population parameter matrices

b = [1 -gam; -beta 1];

a = [0 -alpha];

% Construct exogenous variables

x = 10\*randn(t,1);

% Construct disturbances

u = randn(t,2)\*chol(omega);

% Simulate the model by simulating 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 log-likelihood function

function lf = neglog( theta,y,x )

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

end

% Log-likelihood at each observation

function lnl = lnlt(theta,y,x)

[t n] = size(y);

b = [1 -theta(2); -theta(1) 1];

a = [0 -theta(3)];

e = zeros(t,n);

% Construct residuals

for i=1:t

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

end

% Concentrate out resid var-covar matrix and restrict it to be diagonal

me2 = mean(e.^2);

omega = eye(n);

omega = diag( me2 );

lnl = zeros(t,1);

for i=1:t

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

- 0.5\*e(i,:)\*inv(omega)\*e(i,:)';

end

end

% Instrumental variable estimation

function b = iv(y,w,x)

% IV estimates

b = inv(w'\*x\*inv(x'\*x)\*x'\*w)\*(w'\*x\*inv(x'\*x)\*x'\*y);

% % Standard error of regression

e = y - w\*b;

k = size(w,2);

t = size(y,1);

sigma = sqrt( e'\*e/t );

% Variance-covariance matrix

vcov = sigma^2\*inv(w'\*x\*inv(x'\*x)\*x'\*w);

sterr = sqrt( diag(vcov) );

tstats = b./sterr;

end

Problem 7

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

%

% Program to estimate a recursive system to demonstrate the

% equivalence of OLS and MLE in this special case of the SUR model.

%

% The model is

% y1t = + alpha1\*x1t + u1t

% y2t = beta1\*y1t + alpha2\*x2t + u2t

% y3t = beta2\*y1t + beta3\*y2t + alpha3\*x3t + u3t

%

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

function linear\_recursive( )

clear all;

clc;

% Simulate data

t = 200;

[ y,x ] = simulatedata( t );

% Estimate the model with random starting values

theta = fminunc( @(theta) neglog(theta,y,x),rand(6,1) );

% Estimate parameters by OLS

eq1 = x(:,1)\y(:,1);

eq2 = [y(:,1) x(:,1)]\ y(:,2);

eq3 = [y(:,[1 2]) x(:,1)]\y(:,3);

true = [0.4; 0.6; -0.5; 0.2; 1.0; 0.2];

thetaols = [eq1; eq2; eq3 ];

disp('Comparing true and estimated parameter values')

disp(' Actual MLE OLS')

disp( [ true theta thetaols] );

% Compute covariance matrix

u = zeros(t,3);

b = [1 -theta(2) -theta(4); 0 1 -theta(5); 0 0 1];

a = [-theta(1) -theta(3) -theta(6); 0 0 0; 0 0 0];

for i=1:t

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

end

tmp = diag(u'\*u/t);

omegah = eye(3);

omegah = diag( tmp );

omega = [2 0 0; 0 1 0; 0 0 5];

disp('Comparison of true and estimated elements of omega');

disp( [ omega(:) omegah(:) ] );

end

%

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

%

% Simulate data

function [ y,x ] = simulatedata( t )

RandStream.setDefaultStream( RandStream('mt19937ar','seed',123457) );

alpha1 = 0.4;

beta1 = 0.6;

beta2 = 0.2;

alpha2 = -0.5;

beta3 = 1.0;

alpha3 = 0.2;

omega = [2 0 0; 0 1 0; 0 0 5];

% Construct population parameter matrices

b = [1 -beta1 -beta2; 0 1 -beta3; 0 0 1];

a = [-alpha1 -alpha2 -alpha3; 0 0 0; 0 0 0];

% Construct exogenous variables

x = [randn(t,1) 2\*randn(t,1) 3\*randn(t,1)];

% Construct disturbances

u = randn(t,3)\*chol(omega);

% Simulate the model by simulating the reduced form

y = zeros(t,3);

for i=1:t

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

end

end

% Negative log-likelihood function

function lf = neglog( theta,y,x )

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

end

% Log-likelihood at each observation

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

[t n] = size(y);

b = [1 -theta(2) -theta(4); 0 1 -theta(5); 0 0 1];

a = [-theta(1) -theta(3) -theta(6); 0 0 0; 0 0 0];

% Construct residuals and concentrate diagonal covariance matrix

u = zeros(t,n);

for i=1:t

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

end

tmp = diag(u'\*u/t);

omega = eye(n);

omega = diag( tmp );

lnl = zeros(t,1);

for i=1:t

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

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

end

end

Problem 8

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

%

% Program to estimate a SUR system

%

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

function linear\_sur( )

clear all;

clc;

RandStream.setDefaultStream( RandStream('mt19937ar','seed',123) )

t = 500;

alpha1 = 0.4;

alpha2 = -0.5;

alpha3 = 1.0;

omega = [ 1 0.5 -0.1;

0.5 1.0 0.2;

-0.1 0.2 1.0];

b = eye(3);

a = [ -alpha1 0 0;

0 -alpha2 0;

0 0 -alpha3];

% Exogenous variables

x = [randn(t,1) 2\*randn(t,1) 3\*randn(t,1)];

% Disturbances

u = randn(t,3)\*chol(omega);

% Simulate the model by simulating the reduced form

% Note that the SUR system is the reduced form

y = zeros(t,3);

for i=1:t

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

end

% Estimate the model

theta0 = rand(3,1);

theta = fminunc(@(theta) neglog(theta,y,x),theta0);

% Compute OLS estimates for comparative purposes

alpha1ols = x(:,1)\y(:,1);

alpha2ols = x(:,2)\y(:,2);

alpha3ols = x(:,3)\y(:,3);

alpha = [alpha1; alpha2; alpha3];

alphaols = [alpha1ols; alpha2ols; alpha3ols];

disp('Comparing true and estimated parameter values')

disp(' Actual MLE OLS')

disp( [ alpha theta alphaols] );

% Compute residuals at optimal parameters

a = [ -theta(1) 0 0;

0 -theta(2) 0;

0 0 -theta(3) ];

for i=1:t;

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

end

omegahat = u'\*u/t;

disp('Comparing true and estimated elements of Omega')

disp( [ omega(:) omegahat(:) ] );

end

%

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

%

% Log-likelihood function

function lf = neglog( theta,y,x )

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

end

% Log-likelihood function at each observation

function lnl = lnlt( theta,y,x )

[t n] = size(y);

b = eye(n);

a = [ -theta(1) 0 0;

0 -theta(2) 0;

0 0 -theta(3) ];

u = zeros(t,n);

% Compute residuals

for i=1:t;

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

end

% Concentrate residual var-covar matrix

omega = u'\*u/t;

% Log-likelihood function

lnl = zeros(t,1);

for i=1:t

lnl(i) = -n\*0.5\*log(2\*pi) + log(abs(det(b))) - 0.5\*log(det(omega)) - 0.5\*u(i,:)\*inv(omega)\*u(i,:)';

end

end

Prblem 9

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

%

% Program to estimate parameters of a Taylor Rule

%

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

function linear\_taylor( )

clear all;

clc;

load taylor;

% Choose data 1987:Q1 to 1999:Q4

infl = taylor.Data(104:end,1);

ygap = taylor.Data(104:end,2);

ffr = taylor.Data(104:end,3);

t = length(ffr);

tmp = [ t, sum(infl), sum(ygap), ;

sum(infl), sum(infl.\*infl), sum(infl.\*ygap) ;

sum(ygap), sum(infl.\*ygap), sum(ygap.\*ygap) ];

disp( tmp );

tmp1 = [ sum(ffr) ;

sum(ffr.\*infl) ;

sum(ffr.\*ygap) ];

disp( tmp1 )

% OLS estimates long-hand

betahat1 = inv(tmp)\*tmp1;

% OLS estimates

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

y = ffr;

betahat = x\y;

disp( 'ML/OLS estimates - both methods' );

disp( [betahat1 betahat ] );

e = y - x\*betahat;

sig2hat = e'\*e/t;

disp( 'ML/OLS estimate of variance' );

disp( sig2hat );

vcov = sig2hat\*inv(x'\*x);

disp( 'Covariance matrix of parameters' )

disp( vcov );

% Wald test of restrictions b(2)=1.5 and b(3)=0.5

R = [0 1 0 ;

0 0 1];

Q = [ 1.5 ;

0.5 ];

W = (R\*betahat - Q)'\*inv(R\*vcov\*R')\*(R\*betahat - Q);

p = 1 - chi2cdf(W,2);

disp('Wald test results')

disp(['Wald statistic = ' num2str(W) ]);

disp(['p value = ' num2str(p) ]);

end

Problem 10

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

%

% Program to estimate Klein's macroeconomic model by full

% information maximum likelihood.

%

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

function linear\_klein( )

clear all;

clc;

t = 22;

% Read in the data

klein\_data = load('klein.dat', '-ASCII' );

c = klein\_data(:,1);

p = klein\_data(:,2);

pw = klein\_data(:,3);

i = klein\_data(:,4);

klag = klein\_data(:,5);

d = klein\_data(:,6);

income = klein\_data(:,7);

gw = klein\_data(:,8);

g = klein\_data(:,9);

tax = klein\_data(:,10);

trend = (-11:1:-11+t-1)';

% Estimate consumption function by OLS

y = c(2:end,:);

pwgw = pw + gw;

x = [ones(t-1,1) p(2:end,:) p(1:end-1) pwgw(2:end,:)];

alpha\_ols = x\y;

disp('OLS parameter estimates of the consumption function:');

disp( alpha\_ols );

% Estimate investment function by OLS

y = i(2:end,:);

x = [ones(t-1,1) p(2:end,:) p(1:end-1) klag(2:end,:)];

beta\_ols = x\y;

disp('OLS parameter estimates of the investment function:');

disp( beta\_ols );

% Estimate wage function by OLS

y = pw(2:end,:);

x = [ones(t-1,1) d(2:end,:) d(1:end-1) trend(2:end,:)];

gam\_ols = ((x'\*y)'/(x'\*x));

disp('OLS parameter estimates of the wage function:');

disp( gam\_ols );

% Define the endogenous variables for the system

cipw = [c i pw];

y = cipw(2:end,:);

% Define exogenous/predetermined variables (instruments) for the system

x = [ones(t-1,1) g(2:end,:) tax(2:end,:) gw(2:end,:) trend(2:end,:) p(1:end-1,:) d(1:end-1,:) klag(2:end,:)];

% Estimate consumption function by IV

pwgw = pw + gw;

alpha\_iv = iv(c(2:end,:), [ones(t-1,1) p(2:end,:) p(1:end-1,:) pwgw(2:end,:)], x);

disp('IV estimates of the consumption function');

disp( alpha\_iv );

% Estimate investment function by IV

beta\_iv = iv(i(2:end,:), [ones(t-1,1) p(2:end,:) p(1:end-1,:) klag(2:end,:)], x);

disp('IV estimates of the investment function');

disp( beta\_iv );

% Estimate wage function by IV

gam\_iv = iv(pw(2:end,:), [ones(t-1,1) d(2:end,:) d(1:end-1) trend(2:end,:)], x);

disp('IV estimates of the wage function');

disp( gam\_iv );

% Estimate the model by FIML using IV starting values

theta0 = [alpha\_iv; beta\_iv; gam\_iv;];

[theta,a0] = fminunc(@(theta) neglog(theta,y,x),theta0);

disp('FIML estimates of the consumption function');

disp( theta(1:4) );

disp('FIML estimates of the investment function');

disp( theta(5:8) );

disp('FIML estimates of the wage function');

disp( theta(9:12) );

disp( ' ' );

disp( ['Log-likelihood = ', num2str(-a0) ] );

[t n] = size(y);

b = [1-theta(2) -theta(6) -theta(10);

-theta(2) 1-theta(6) -theta(10);

theta(2)-theta(4) -theta(6) 1];

a = [-theta(1) -theta(5) -theta(9);

-theta(2) -theta(6) -theta(10);

theta(2) theta(6) 0;

-theta(4) 0 0;

0 0 -theta(12);

-theta(3) -theta(7) 0;

0 0 -theta(11);

0 -theta(8) 0];

% Compute residuals and covariance matrix

u = zeros(t,n);

for i=1:t

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

end

rvc = u'\*u/t;

disp('Estimated covariance matrix');

disp( rvc );

end

%

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

%

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

% Log-likelihood function

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

function lf = neglog( theta,y,x )

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

end

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

% Log-likelihood function at each observation

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

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

[t n] = size(y);

b = [1-theta(2) -theta(6) -theta(10);

-theta(2) 1-theta(6) -theta(10);

theta(2)-theta(4) -theta(6) 1];

a = [-theta(1) -theta(5) -theta(9);

-theta(2) -theta(6) -theta(10);

theta(2) theta(6) 0;

-theta(4) 0 0;

0 0 -theta(12);

-theta(3) -theta(7) 0;

0 0 -theta(11);

0 -theta(8) 0];

% Construct residuals and covariance matrix

u = zeros(t,n);

for i=1:t

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

end

omega = u'\*u/t;

lnl = zeros(t,1);

for i=1:t

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

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

end

end

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

% Instrumental variable estimation

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

function b = iv(y,w,x)

% IV estimates

b = inv(w'\*x\*inv(x'\*x)\*x'\*w)\*(w'\*x\*inv(x'\*x)\*x'\*y);

% % Standard error of regression

e = y - w\*b;

k = size(w,2);

t = size(y,1);

sigma = sqrt( e'\*e/t );

% Variance-covariance matrix

vcov = sigma^2\*inv(w'\*x\*inv(x'\*x)\*x'\*w);

sterr = sqrt( diag(vcov) );

tstats = b./sterr;

end