

```
% This is the Matlab program for the Matching UE and Equipment K paper
% Solution method following UHLIG undetermined coefficients with
% sensitivity to the structure as described in Marimon (2001)
% COPYRIGHT by O. MIKHAIL (2002)
% Feel free to copy, modify and use at your own risk.
% However, you are not allowed to sell this code or otherwise impinge
% on its free distribution.
clear all; % clear all variables in memory
diary matchingue-diary.txt replace; % create the diary file
format compact;
disp('*****');
disp('*****');
disp('MATCHING UE AND DELAY IN EQUIPMENT CAPITAL: EAST GERMANY');
disp('Calibration from BURDA and HUNT (2001)');
disp('Written by O. Mikhail');
disp('Sep 2002');
disp('*****');
disp('*****');
% Just to set up the final results of Impulse in a matrix
HORIZON = 36; % how long impulse response should be
% 5 points for sensitivity analysis
% 5 points x 2 shocks x 11 variables = 110 obs
resp_all = zeros(HORIZON,110);
% Prepare for the loop over all 5 points
ii = 1; jj = 22;
% NOTE 5 POINTS ONLY FOR SENSITIVITY ANALYSIS
psivec = [0.25 0.3 0.35 0.40 0.45]
[notimportant,numberpoints]=size(psivec);
for i=1:1:5
psi = psivec(i);
disp('*****');
disp('*****');
disp('***** LOOP OVER PARAMTER PSI *****');
disp('*****');
disp(sprintf('***** PSI = %3.3f *****',psi))
disp('*****');
disp('*****');
disp('*****');
% PARAMETERS
alpha = 0.35; % Capital Share
beta = 0.99; % Subjective Probabiblity of Surviving
delta = 0.094; % Depreciation rate for capital from BURDA and HUNT (2001)
%psi = 0.25; % delay paramter from BURDA and HUNT (2001)
% invest in structures =approx 3 times invest in equipme✓
nt % therefore the psi parameter equals 0.25
s = 0.08; % Exogenous Separation Rate from LANGOT (1995)
theta = 0.10; % Labor market tightness from LANGOT (1995)
```

```
rhoR    = 0.95;    % AR(1) parameter of the matching shock
rhoA    = 0.95;    % AR(1) parameter of the technology shock

% RATIOS AND PROBABILITIES
lambda  = 0.75;    % worker bargaining power
                    % note that LANGOT (1995) for France used 0.76
                    % FONSECA and MUNOZ (1999) for Spain used 0.85
gamma= 1-lambda;  % HOSIOS Efficiency Condition
                    % gamma is the elasticity of matching with respect to Vacancies
Nbar    = 0.65;    % From Burda and Wyplosz (1994) for Germany = 0.27
                    % mean employment rate (Table 1, BURDA&HUNT(2001) last column)
GammaN  = 0.95;    % Disutility from work
GammaU  = 0.10;    % Disutility from being unemployed
omega   = 0.32;    % cost to fill a job
%Abar = 1;        % Tech normalization - FORGET IT - Let the StSt compute it
Vbar = theta*(1-Nbar); % Make sure that the degree of tightness is 0.1044

% Create the PAR vector of parameters and ratios
% to pass it to the routine FSOLVE
par(1)=alpha; par(2)=beta; par(3)=delta; par(4)=psi; par(5)=s;
par(6)=theta; par(7)=rhoR; par(8)=rhoA; par(9)=lambda; par(10)=gamma;
par(11)=Nbar; par(12)=GammaN; par(13)=GammaU; par(14)=omega;

% Calculate the steady State using FSOLVE
ss0 = ones(8,1); % create initial points to iterate from
% options for FSOLVE function
zizo=optimset('Display','final','MaxFunEvals',9000,'MaxIter',9000);
% The function steadystate.m contains the system of StSt equations
ss = fsolve(@steadystate,ss0,zizo,par); % Solve the system for the ss
% Make sure that K/Y ratio is 4.58 as in BURDA and HUNT(2001, p. 69)
Kbar = ss(2) * 4.58;
% disp('');
% disp('Starting Point for StSt Iterations');
% ss0;
% Now back to the main program - get the results from ss
Qbar = ss(1); Ybar = ss(2); Ibar = ss(3); Rbar= ss(4); Abar = ss(5);
Wbar = ss(6); Cnbar = ss(7); Cubar = ss(8);
disp('');
disp('The Steady State');
disp(sprintf('Qbar    = %3.3f ',ss(1)))
disp(sprintf('Ybar    = %3.3f ',ss(2)))
disp(sprintf('Ibar    = %3.3f ',ss(3)))
disp(sprintf('Rbar    = %3.3f ',ss(4)))
disp(sprintf('Abar    = %3.3f ',ss(5)))
disp(sprintf('Wbar    = %3.3f ',ss(6)))
disp(sprintf('Cnbar   = %3.3f ',ss(7)))
disp(sprintf('Cubar   = %3.3f ',ss(8)))
```

```

disp(sprintf('Vbar    = %3.3f ',Vbar))
disp(sprintf('Kbar    = %3.3f ',Kbar))
disp(sprintf('Capital Output  ratio = %3.3f ',(Kbar/ss(2))))
disp(sprintf('Output  Labor  ratio = %3.3f ',(Ybar/Nbar)))
disp(sprintf('Output  Capital ratio = %3.3f ',(Ybar/Kbar)))
disp(sprintf('Degree of Tightness    = %3.3f ',(Vbar/(1-Nbar))))
disp(sprintf('Cost to fill a job    = %3.3f ',omega))

disp('');
pause;

% Another Way to Calculate the Steady State
% Ybar = A^(1/(1-alpha))*(0.218)^(-alpha/(1-alpha))*Nbar
% Kbar=Ybar/0.218
% w=(lambda*A*(1-alpha)*(Ybar/Nbar))+((1-lambda)*(GammaN-GammaU))
% Vbar=0.035 % So that V/U = 0.10
% Cn=((1-delta)*Kbar)+Ybar-(omega*Vbar)-((1-Nbar)*(GammaU-GammaN))-Kbar
% Cu=Cn-GammaN+GammaU
% q=(R*(Vbar^(gamma-1))*(1-Nbar)^(1-gamma))
% Ibar = Kbar-((1-delta)*Kbar)
% VERIFY, Next two lines should be equal
%omega/(Qbar*beta)
%((1-alpha)*Ybar/Nbar)-Wbar+((1-s)*omega/Qbar)

% Declaring the matrices.
VARNAMES = ['capital    ',
            'labor      ',
            'invest     ', % Investment
            'lambda1    ', % Lagrange Multiplier
            'vacancy    ',
            'wage       ',
            'consU     ', % Consumption when unemployed
            'consN     ', % Consumption when employed
            'output    ', % NOTE that GDP INDEX IS 9
            'Matching  ', % Matching Shock
            'Technology']; % Tech Shock

% Endogenous state variables "x(t)": k(t), n(t), i(t), l(t), v(t), w(t), cu✓
(t) m=7
% Endogenous other variables "y(t)": cn(t), y(t) n=2
% Exogenous state variables "z(t)": R(t), A(t).
% Switch to that notation. Find matrices for format
% 0 = AA x(t) + BB x(t-1) + CC y(t) + DD z(t)
% 0 = E_t [ FF x(t+1) + GG x(t) + HH x(t-1) + JJ y(t+1) + KK y(t) + LL z(t+✓
1) + MM z(t) ]
% z(t+1) = NN z(t) + epsilon(t+1) with E_t [ epsilon(t+1) ] = 0,

% DETERMINISTIC EQUATIONS:
tt = (1-alpha)*(alpha)*lambda*Ybar/Nbar;
    
```

ttt = (1-alpha)\*lambda\*Ybar/Nbar;

```
%      k(t)      n(t)      i(t)  l(t)  v(t)  w(t)  cu(t)
AA = [ 0          0          0    0    0    0    -Cubar
      tt          -tt          0    0    0    -1    0
      alpha      (1-alpha)  0    0    0    0    0]
```

```
%      k(t-1)  n(t-1)  i(t-1)  l(t-1)  v(t-1)  w(t-1)  cu(t-1)
BB = [ 0        0        0        0        0        0        0
      0        0        0        0        0        0        0
      0        0        0        0        0        0        0]
```

```
%      cn(t)  y(t)
CC = [Cnbar  0
      0      0
      0     -1 ]
```

```
%      R(t)  A(t)
DD = [0      0
      0     ttt
      0      1 ]
```

% EXPECTATIONAL EQUATIONS:

```
ooo = beta*(1-alpha)*Ybar/Nbar;
oo = beta*(1-s)*omega*Rbar^(-1)*Vbar^(1-gamma)*(1-Nbar)^(gamma-1);
oooo = omega*Rbar^(-1)*Vbar^(1-gamma)*(1-Nbar)^(gamma-1);
Mbar = Rbar*(Vbar^gamma)*((1-Nbar)^(1-gamma));
```

```
%      k(t+1),      n(t+1),      i(t+1),      l(t+1)      ✓
      v(t+1)      w(t+1)      cu(t+1)
FF=[-Kbar          0          0          0          ✓
    0              0          0          0          ✓
   -Kbar          0          0          0          ✓
    0              0          0          0          ✓
    0              0          -Nbar      0          ✓
    0              0          0          (1-psi)*psi  (1-psi) ✓
    0              0          0          0          ✓
   -beta*alpha*Ybar/Kbar  0          0          beta*(1-delt ✓
a)  0              0          0          0          ✓
    0              -ooo-oo*(gamma-1)  0          0          ✓
   oo*(1-gamma)  -beta*Wbar  0]
```

```
%      k(t),      n(t),      i(t),      ✓
      l(t)  v(t)      w(t)  cu(t)
GG = [ (1-delta)*Kbar  0          Ibar*psi      ✓
      0          0          0          0
      (1-delta)*Kbar  Nbar*(Cubar-Cnbar)  0          ✓
      0          -omega*Vbar  0          (Nbar*Cubar)-Cubar
```

```

0          ((1-s)*Nbar)-Mbar(1-gamma) 0
0      Mbar*gamma      0      0
0          0          0          ((-(1-psi)*psi)+(psi*(psi
-1))) psi      0          0      0
0          0          0          0
-1      0          0      0
0          oooo*(gamma-1)          0
0      -oooo*(1-gamma) 0      0]

```

```

%      k(t-1),          n(t-1),      i(t-1),          l(t-1)      v(t-1)      w(t-1) c
u(t-1)
HH = [ 0          0          Ibar*(1-psi)      0          0          0          0
0          0          0          0          0          0          0
0          0          0          0          0          0          0
0          0          psi*(1-psi)      0          0          0          0
0          0          0          0          0          0          0
0          0          0          0          0          0          0]
]

```

```

%      cn(t+1)      y(t+1),
JJ = [ 0          0
0          0
0          0
0          0
0          beta*alpha*Ybar/Kbar
0          beta*(1-alpha)*Ybar/Nbar ]

```

```

%      cn(t)          y(t),
KK = [ 0          0
-Nbar*Cnbar      Ybar
0          0
0          0
0          0
0          0 ]

```

```

% For R(t+1)          A(t+1)
LL = [ 0          0
0          0
0          0
0          0
0          0
-oo          beta*(1-alpha)*Abar*Ybar/Nbar ]

```

```

% For R(t)          A(t)
MM = [ 0          0
0          0
Mbar          0
0          0

```

```
0 0
oooo 0]

% AUTOREGRESSIVE MATRIX FOR z(t)
%NN = [psi];
%Sigma = [ sigma_eps^2 ];
NN = [rhoR 0
      0 rhoA]

Sigma = [ 0.01 0
          0 0.01]

%% FROM NOW IS OK
%% Setting the options:

[l_equ,m_states] = size(AA);
[l_equ,n_endog ] = size(CC);
[l_equ,k_exog ] = size(DD);

DISPLAY_IMMEDIATELY = 1;
MANUAL_ROOTS = 0;
TOL = .000001;
% Starting the calculations:
% Run the Uhlig Solution
[l_equ,m_states] = size(AA);
[l_equ,n_endog ] = size(CC);
[l_equ,k_exog ] = size(DD);
message = '
      ';
warnings = [];
options;
% Here I picked the roots manually
ssolve; % use ssolve.m instead of solve.m
sol_out;
    impresp;
    simul;
    sim_out;
    moments;
    mom_out;
% form the impulse response all matrix after every iteration on the parameter
resp_all(:,ii:jj) = Resp_mat';
nvar = m_states+n_endog+k_exog; % number of the variables in the model
ii = ii + (nvar*k_exog) % 22 is the number of variables twice because 2 shows
cks for each var
jj = jj + (nvar*k_exog)
end;

save matching-ue-work % saving all workspace
```

```
save zresponse.txt resp_all -ASCII -TABS % saving response matrix in ASCII ✓  
file  
save zresponse.mat resp_all % saving response matrix in Matlab ✓  
file  
  
% Response to shock in matching  
capitalm = [ resp_all(:,1) resp_all(:,23) resp_all(:,45) resp_all(:,67) res✓  
p_all(:,89) ]  
laborm = [ resp_all(:,2) resp_all(:,24) resp_all(:,46) resp_all(:,68) res✓  
p_all(:,90) ]  
vacancym = [ resp_all(:,5) resp_all(:,27) resp_all(:,49) resp_all(:,71) res✓  
p_all(:,93) ]  
consum = [ resp_all(:,7) resp_all(:,29) resp_all(:,51) resp_all(:,73) res✓  
p_all(:,95) ]  
consm = [ resp_all(:,8) resp_all(:,30) resp_all(:,52) resp_all(:,74) res✓  
p_all(:,96) ]  
outputm = [ resp_all(:,9) resp_all(:,31) resp_all(:,53) resp_all(:,75) res✓  
p_all(:,97) ]  
  
% Response to technology  
capitalt = [ resp_all(:,12) resp_all(:,34) resp_all(:,56) resp_all(:,78) re✓  
sp_all(:,100) ]  
labort = [ resp_all(:,13) resp_all(:,35) resp_all(:,57) resp_all(:,79) re✓  
sp_all(:,101) ]  
vacancyt = [ resp_all(:,16) resp_all(:,38) resp_all(:,60) resp_all(:,82) re✓  
sp_all(:,104) ]  
const = [ resp_all(:,18) resp_all(:,40) resp_all(:,62) resp_all(:,84) re✓  
sp_all(:,106) ]  
const = [ resp_all(:,19) resp_all(:,41) resp_all(:,63) resp_all(:,85) re✓  
sp_all(:,107) ]  
outputt = [ resp_all(:,20) resp_all(:,42) resp_all(:,64) resp_all(:,86) re✓  
sp_all(:,108) ]  
  
% Making Graph for Impulse Response  
disp('Inspect figure. Hit key when ready...');  
pause;  
  
%newgraph = plot(Time_axis,0*Time_axis,Time_axis,labort);  
figure('DefaultAxesColorOrder',[0 0 0],'DefaultAxesLineStyleOrder','-|:|--|✓  
-.'');  
ThinLineStyle=1; % change it to =2 if you want to save the graph  
set(gcf,'Renderer','zbuffer');  
newgraph = plot(Time_axis+1,labort);  
set(newgraph(2:max(size(newgraph))), 'LineWidth',0.75); % Drop first obs  
%axis([0 3 0 0.5]) % XMIN XMAX YMIN YMAX  
title('\bfImpulse responses for Labor to a shock in Technology','FontSize',✓  
14);  
xlabel('\bfYears after shock');  
ylabel('\bf% deviation from steady state');
```

```
for psiindex = 1:1:numberpoints
text(Time_axis(min([10,HORIZON/2])), ...
      labort(min([10,HORIZON/2]),psiindex),...
      ['\bf\psi = ', num2str(psivec(:,psiindex))]);
end

pause;

figure('DefaultAxesColorOrder',[0 0 0],'DefaultAxesLineStyleOrder','-|:|--|↙
-.'');
ThinLineStyle=1; % change it to =2 if you want to save the graph
set(gcf,'Renderer','zbuffer');
newgraph = plot(Time_axis,laborm);
set(newgraph(2:max(size(newgraph))), 'LineWidth',0.75); % Drop first obs
%axis([0 3 -0.035 0.02]) % XMIN XMAX YMIN YMAX
title('\bfImpulse responses for Labor to a shock in Matching','FontSize',14↙
);
xlabel('\bfYears after shock');
ylabel('\bf% deviation from steady state');
for psiindex = 1:1:numberpoints
text(Time_axis(min([10,HORIZON/2])), ...
      laborm(min([10,HORIZON/2]),psiindex),...
      ['\bf\psi = ', num2str(psivec(:,psiindex))]);
end

pause;

figure('DefaultAxesColorOrder',[0 0 0],'DefaultAxesLineStyleOrder','-|:|--|↙
-.'');
ThinLineStyle=1; % change it to =2 if you want to save the graph
set(gcf,'Renderer','zbuffer');
newgraph = plot(Time_axis,vacancyt);
set(newgraph(2:max(size(newgraph))), 'LineWidth',0.75); % Drop first obs
%axis([0 3 0.0 1.5]) % XMIN XMAX YMIN YMAX
title('\bfImpulse responses for Vacancy to a shock in Technology','FontSize↙
',14);
xlabel('\bfYears after shock');
ylabel('\bf% deviation from steady state');
for psiindex = 1:1:numberpoints
text(Time_axis(min([10,HORIZON/2])), ...
      vacancyt(min([10,HORIZON/2]),psiindex),...
      ['\bf\psi = ', num2str(psivec(:,psiindex))]);
end

pause;

figure('DefaultAxesColorOrder',[0 0 0],'DefaultAxesLineStyleOrder','-|:|--|↙
-.'');
ThinLineStyle=1; % comment out if you want to save
```

```
set(gcf,'Renderer','zbuffer');
newgraph = plot(Time_axis,vacancym);
set(newgraph(2:max(size(newgraph))), 'LineWidth',0.75); % Drop first obs, change to 2
%axis([0 3 -0.15 0.02]) % XMIN XMAX YMIN YMAX
title('\bfImpulse responses for Vacancy to a shock in Matching','FontSize',14);
xlabel('\bfYears after shock');
ylabel('\bf% deviation from steady state');
    for psiindex = 1:1:numberpoints
        text(Time_axis(min([10,HORIZON/2])), ...
            vacancym(min([10,HORIZON/2]),psiindex),...
            ['\bf\psi = ', num2str(psivec(:,psiindex))]);
    end

pause;

figure('DefaultAxesColorOrder',[0 0 0],'DefaultAxesLineStyleOrder','-|:|--|^-.-');
ThinLineStyles=2; % change it to =2 if you want to save the graph
set(gcf,'Renderer','zbuffer');
subplot(2,2,1)
newgraph1 = plot(Time_axis,labort);
set(newgraph1(2:max(size(newgraph1))), 'LineWidth',0.75); % Drop first obs
%axis([0 9 0 0.03]) % XMIN XMAX YMIN YMAX
title('Technology Shock','FontSize',14);
xlabel('Years after shock','FontSize',8);
ylabel('% deviation in LABOR from stst','FontSize',12);
    for psiindex = 1:1:numberpoints
        text(Time_axis(min([10,HORIZON/2])), ...
            labort(min([10,HORIZON/2]),psiindex),...
            ['\psi = ', num2str(psivec(:,psiindex))]);
    end

subplot(2,2,2)
newgraph2 = plot(Time_axis,laborm);
set(newgraph2(2:max(size(newgraph2))), 'LineWidth',0.75); % Drop first obs
%axis([0 9 0.0009 0.003]) % XMIN XMAX YMIN YMAX
title('Matching Shock','FontSize',14);
xlabel('Years after shock','FontSize',8);
ylabel('% deviation in LABOR from stst','FontSize',12);
%     for psiindex = 1:1:numberpoints
%         text(Time_axis(min([8,HORIZON/2])), ...
%             laborm(min([8,HORIZON/2]),psiindex),...
%             ['\psi = ', num2str(psivec(:,psiindex))]);
%     end

subplot(2,2,3)
newgraph3 = plot(Time_axis,vacancym);
```

```
set(newgraph3(2:max(size(newgraph3))), 'LineWidth', 0.75); % Drop first obs
%axis([0 9 0.0 0.35]) % XMIN XMAX YMIN YMAX
title('Technology Shock', 'FontSize', 14);
xlabel('Years after shock', 'FontSize', 8);
ylabel('% deviation in VACANCY from stst', 'FontSize', 12);
    for psiindex = 1:1:numberpoints
        text(Time_axis(min([5, HORIZON/2])), ...
            vacancyt(min([5, HORIZON/2]), psiindex), ...
            ['\psi = ', num2str(psivec(:, psiindex))]);
    end

subplot(2, 2, 4)
newgraph4 = plot(Time_axis, vacancym);
set(newgraph4(2:max(size(newgraph4))), 'LineWidth', 0.75); % Drop first obs, ✓
change to 2
%axis([0 9 -0.0055 -0.001]) % XMIN XMAX YMIN YMAX
title('Matching Shock', 'FontSize', 14);
xlabel('Years after shock', 'FontSize', 8);
ylabel('% deviation in VACANCY from stst', 'FontSize', 12);
    for psiindex = 1:1:numberpoints
        text(Time_axis(min([4, HORIZON/2])), ...
            vacancym(min([4, HORIZON/2]), psiindex), ...
            ['\psi = ', num2str(psivec(:, psiindex))]);
    end

% FIND OUT IF RESPONSE IS GOOD using DLSIM.M
% DLSIM Simulation of discrete-time linear systems.
% Y = DLSIM(A,B,C,D,U) calculates the time response of the system:
%     x[n+1] = Ax[n] + Bu[n]
%     y[n]   = Cx[n] + Du[n]
% to input sequence U. Matrix U must have as many columns as there are
% inputs, u. Each row of U corresponds to a new time point. DLSIM
% returns a matrix Y with as many columns as there are outputs y, and
% with LENGTH(U) rows.
% [Y,X] = DLSIM(A,B,C,D,U) also returns the state time history.
% DLSIM(A,B,C,D,U,X0) can be used if initial conditions exist.
% Uncomment the following three lines
%ANN = PP; BNN = QQ; CNN = RR; DNN = SS; UNN = Resp_mat(10:11,:);

%X0NN = Resp_mat(1:7,1)
%[YNN,XNN] = DLSIM(ANN,BNN,CNN,DNN,UNN,X0NN)
% now XNN should be the same as Resp_mat(1:7,:)', i.e., zzz should equal ze✓
ro
%zzz = XNN - Resp_mat(1:7,:)'
% Slightly different because of the lag
% DLSIM does not have a lag in the second equation
% so using RR for C is not good.
```

diary off;