```
var L Ld C mrs R Pi Lam Pir x1 x2 pw Y w A wr f1 f2 vp YW vw y pi ld r;

varexo eA eR;

parameters beta epsip epsiw phip phiw sigma chi beta psi rhoA rhoR thetapi sA sR;

load param_dnk_wage;
set_param_value('beta',beta);
set_param_value('epsip',epsip);
set_param_value('epsiw',epsiw);
set_param_value('sigma',sigma);
set_param_value('chi',chi);
set_param_value('psi',psi);
set_param_value('phip',phip);
set_param_value('phiw',phiw);
set_param_value('thetapi',thetapi);
set_param_value('rhoA',rhoA);
set_param_value('rhoR',rhoR);
set_param_value('sA',sA);
set_param_value('sR',sR);

model;

% (1) Labor supply
psi*L^(chi) = C^(-sigma)*mrs;

% (2) Euler equation bonds
1 = R*Lam(+1)*Pi(+1)^(-1);

% (3) SDF
Lam = beta*(C/C(-1))^(-sigma);

% (4) Reset inflation
Pir = (epsip/(epsip-1))*x1/x2;

% (5) x1
x1 = pw*Y + phip*Lam(+1)*Pi(+1)^(epsip)*x1(+1);

% (6) x2
x2 = Y + phip*Lam(+1)*Pi(+1)^(epsip-1)*x2(+1);

% (7) Labor demand goods
w = pw*A;

% (8) Reset wage
wr = (epsiw/(epsiw-1))*f1/f2;

% (9) f1
f1 = mrs*w^(epsiw)*Ld + phiw*Lam(+1)*Pi(+1)^(epsiw)*f1(+1);

% (10) f2
f2 = w^(epsiw)*Ld + phiw*Lam(+1)*Pi(+1)^(epsiw-1)*f2(+1);

% (11) Price evolution
1 = (1-phip)*Pir^(1-epsip) + phip*Pi^(epsip-1);

% (12) Wage evolution
w^(1-epsiw) = (1-phiw)*wr^(1-epsiw) + phiw*Pi^(epsiw-1)*w(-1)^(1-epsiw);

% (13) Resource constraint
Y = C;

% (14) Production function
Y*vp = YW;

% (15) Wholesale production
YW = A*Ld;

% (16) Labor supply / aggregate labor demand
```

```
L = Ld*vw;

% (17) Wage dispersion
vw = (1-phiw)*(wr/w)^(-epsiw) + phiw*Pi^(epsiw)*(w/w(-1))^(epsiw)*vw(-1);

% (18) Price dispersion
vp = (1-phip)*Pir^(-epsip) + phip*Pi^(epsip)*vp(-1);

% (19) Process for A
log(A) = rhoA*log(A(-1)) + sA*eA;

% (20) Taylor rule
log(R) = (1-rhoR)*log(1/beta) + rhoR*log(R(-1)) + (1-rhoR)*thetapi*log(Pi) + sR*eR;

y = log(Y);

pi = log(Pi);

r = log(R);

ld = log(Ld);

end;

initval;
A = 1;
L = 1;
Ld = 1;
pw = (epsip-1)/epsip;
w = (epsip-1)/epsip;
mrs = (epsiw-1)/epsiw * (epsip-1)/epsip;
Y = 1;
C = 1;
R = 1/beta;
Lam = beta;
vp = 1;
vw = 1;
Pi = 1;
Pir = 1;
wr = (epsip-1)/epsip;
YW = 1;
x1 = (1*(epsip-1)/epsip)/(1-phip*beta);
x2 = 1/(1-phip*beta);
f1 = (epsiw-1)/epsiw * (epsip-1)/epsip * ((epsip-1)/epsip)^(epsiw)/(1-phiw*beta);
f2 = ((epsip-1)/epsip)^(epsiw)/(1-phiw*beta);
y = 0;
pi = 0;
r = 0;
ld = 0;
end;

shocks;
var eA = 1;
var eR = 1;
end;

stoch_simul(order=1,irf=20,nograph,ar=1);
```