```
var y c i n k b d phip T r R xi z m p w mu logy logc logi logn logp eterm;

varexo ez ex;

parameters alpha beta theta delta tau xis kappa sz sx rhox rhoz rs Rs mus ws ns ks cs ys is bs ds Ts;

load param_jq;
set_param_value('beta',beta);
set_param_value('alpha',alpha);
set_param_value('tau',tau);
set_param_value('theta',theta);
set_param_value('delta',delta);
set_param_value('xis',xis);
set_param_value('kappa',kappa);
set_param_value('sz',sz);
set_param_value('sx',sx);
set_param_value('rhox',rhox);
set_param_value('rhoz',rhoz);
set_param_value('rs',rs);
set_param_value('Rs',Rs);
set_param_value('ys',ys);
set_param_value('cs',cs);
set_param_value('ks',ks);
set_param_value('ns',ns);
set_param_value('is',is);
set_param_value('mus',mus);
set_param_value('bs',bs);
set_param_value('ds',ds);
set_param_value('Ts',Ts);
set_param_value('ws',ws);


model;

% (1) Labor supply
alpha/(1-n) = w/c;

% (2) Euler equation
1 = m(+1)*(1+r);

% (3) Price of shares
p = m(+1)*(d(+1) + p(+1));

% (4) sdf
m = beta*c(-1)/c;

% (5) labor demand
w = (1-mu*phip)*(1-theta)*z*k(-1)^(theta)*n^(-theta);

% (6) bond euler equation firm
1 = mu*xi*phip*R/(1+r) + m(+1)*R*phip/phip(+1);

% (7) capital Euler equation
1 = mu*xi*phip + m(+1)*(phip/phip(+1))*(1-delta + (1-mu(+1)*phip(+1))*theta*z(+1)*k^(theta-1)*n(+1)^(1-theta));

% (8) houehold constraint
c + b/(1+r) = w*n + b(-1) + d - T;

% (9) firm budget constraint
d + kappa*(d - ds)^(2) = z*k(-1)^(theta)*n^(1-theta) -w*n - k + (1-delta)*k(-1) - b(-1) + b/R;

% (10) Borrowing constraint
xi*(k - b/(1+r)) = z*k(-1)^(theta)*n^(1-theta);

% (11) capital accumulation
k = i + (1-delta)*k(-1);
```

```
% (12) production function
y = z*k(-1)^(theta)*n^(1-theta);

% (13) process for z
log(z) = rhoz*log(z(-1)) + sz*ez;

% (14) process for xi
log(xi) = (1-rhox)*log(xis) + rhox*log(xi(-1)) + sx*ex;

% (15) Relationship between R and r
R = 1 + r*(1-tau);

% (16) Derivative of adjustmetn cost
phip = 1 + (kappa/2)*(d - ds);

% (17) Tax
T = b*(1/R - 1/(1+r));

% (18) log(y)
logy = log(y);

% (19) log(c)
logc = log(c);

% (20) log(i)
logi = log(i);

% (21) log(n)
logn = log(n);

% (22) log(p)
logp = log(p);

% (23) eterm
eterm = (p)/(k - b);

end;

initval;
z = 1;
xi = xis;
k = ks;
c = cs;
mu = mus;
b = bs;
T = Ts;
d = ds;
R = Rs;
r = rs;
w = ws;
n = ns;
i = delta*ks;
m = beta;
phip = 1;
y = ys;
logy = log(ys);
logc = log(cs);
logi = log(is);
logn = log(ns);
p = (beta/(1-beta))*ds;
logp = log((beta/(1-beta))*ds);
end;

steady;

shocks;
var ez = 1;
var ex = 1;
```

```
end;

stoch_simul(order=1,irf=20,nograph,ar=0);
```