Data

Data used in Maccini, Moore, and Schaller. “The Interest Rate, Learning, and Inventory Investment”, forthcoming in the **American Economic Review**, is contained in the Excel spreadsheet titled, "RevDataFeb-04Monthly.xls".

Code for the Markov Switching Model Regressions

Below is the code used to estimate the Markov switching model described in section IV. This code is based on code from the web site of Professor James Hamilton (University of California at San Diego) with small alterations for this specific application.

```
@ This GAUSS file reads in data, sets options, and calls numerical
   optimization routine for numerical estimation of switching model @


        output file=s3outp.txt reset;



        /*=============================================
        EXAMPLE 1:

           @ The following lines read in the raw data, and convert to
             100 times the first difference of the log @

           capt = 135;        @ capt is the sample size @
           load bbbb[capt+1,1] = gnp82.dat;
           bbbb = bbbb[1:capt,1]~bbbb[2:capt+1,1];
           y = 100*ln(bbbb[.,2]./bbbb[.,1]);

          @  Adjust any of the following to control specification
desired @
           ns = 2;           @ ns is the number of primitive states @
           ps = 4;           @ ps is the number of lagged states that
matter
                               for y;  use ps = 0 if only the current
state
                               matters @
           pphi = 4;         @ pphi is the number of lags in
autoregression
                                for y; pphi should be greater than or
equal
                                to ps @
           isig = 1;         @ isig = 1 for constant variances,
                               isig =ns for changing variances @
           ipm = 1;          @ ipm specifies way in which transition
probs
                               are parameterized
                                  ipm = 1 implies p11 and p22 estimated
                                  ipm = 2 implies pij for i=1,..,n
j=1,..,n-1
```

```
                                   ipm = 3 user input code @
          a=0;b=0;c=0;       @ These parameters control the Bayesian
prior
                               as suggested in Hamilton, "A Quasi-
Bayesian
                               Approach to Estimating Parameters for
Mixtures
                               of Normal Distributions", Journal of
Business
                               and Economic Statistics, January 1991.
If
                               no Bayesian adjustment is desired, set
all
                               parameters to zero @

          /* Input initial values for parameters
          The order in which variables are represented is as follows
             first ns elements:    means for states
             next pphi elements:   autoregressive coefficients
             next isig elements:   variances for states when izz =1
                                       std. deviations for states when
izz =2
             next elements:        when izz =1 these are the transition
probs
                                   when izz =2 these are params v(i,j)
such that
                                       p(i,j) = v(i,j)^2 / sum j
v(i,j)^2
                                   elements are ordered as p11, p22
when ns =2
                                       ordered as
p(1,1),p(2,1),...,p(ns,1),
                                       p(2,1),...,p(ns-1,ns) when ns >
2 */

                   nth = 9;      @ nth is the number of params to be
estimated@
         mu1 = 1; mu2 = 0; phi1 = .1; phi2 = 0; phi3 = 0; phi4 = 0;
         sig = 1; p11 = 1.5; p22 = 1.5;
                   th = mu1|mu2|phi1|phi2|phi3|phi4|sig|p11|p22;

          ============================================
         EXAMPLE 2: */


           capt = 494;
           load w[capt,2] = yrrtb00.txt;
           y = w[.,2];
           ns = 3;
           ps = 0;
           pphi = 0;
           isig = 3;
           ipm = 3;
           i=1;
           a=0;b=0;c=0;

           nth = 10;
```

```
           mu1 = -2.11; mu2 = 1.3; mu3 = 5; sig1 = 1.9;
           sig2 = .9; sig3 = 2.2; p11=16.3 ; p12=2; p22=14; p23=.2;



           th=mu1|mu2|mu3|sig1|sig2|sig3|p11|p12|p22|p23;



/*
================================================================================
*/
  @ In general no parts of this section should be changed @

proc startval;     @ This defines starting value for iteration to be th
@
  retp(th); endp;
nk = pphi+1;       @ nk is the first observation for which the
                            likelihood will be evaluated @
izz = 1;           @ izz = 1 when params read in so as to assure
inequality
                    constraints; izz = 2 for final reporting of
results @
n = ns^(ps+1);     @ n is the dimension of the state vector @
kc = 1;            @ kc = 2 to echo parameter values @
ks = 1;            @ ks = 2 if smoothed probs are to be calculated @
captst = capt - nk +1; @ captst is the effective sample size @
skif = zeros(captst,n); @ skif is the matrix of filtered probs @
skis = zeros(captst,n); @ skis is the matrix of smoothed probs @
id = eye(ns);           @ used in certain calculations below @

   "Bayesian prior used";
    "a=";;a;;"b=";;b;;"c=";;c;;


proc pattern1; @ This proc returns a (ps+1)*ns x n matrix.  The ith
                column contains a one in row j if st = j, contains a
                one in row ns+j if st-1 = j, and so on @
     local i1,ix,iq,na;
     na = n/ns;
     ix = eye(ns).*.ones(1,na);
     i1 = 1;
     do until i1 > ps;
       na = na/ns;
       iq = ones(1,ns^i1).*.(eye(ns).*.ones(1,na));
       ix = iq|ix;
     i1 = i1+1;
     endo;
retp(ix);
endp;

hp = pattern1;
#include procs;
/* ============================================================== */
```

```
/*
==============================================================================
*/
proc matpm(xth);  @This proc defines the user's conventions for reading
                  elements of Markov transition probabilities from
                  parameter vector @
   local pm,ixth;
   ixth = rows(xth);
   pm = zeros(ns,ns);
     if ipm == 1;  @ for ns =2 this option has parameters as p11 and
p22 @
          if izz == 1;
             pm[1,1] = xth[1,1]^2/(1 +xth[1,1]^2);
             pm[2,2] = xth[2,1]^2/(1 + xth[2,1]^2);
          else;
             pm[1,1] = xth[1,1];
             pm[2,2] = xth[2,1];
          endif;
          pm[2,1] = 1 - pm[1,1];
          pm[1,2] = 1 - pm[2,2];
     elseif ipm == 2;  @ general case has parameters pij for i =
1,...,n and
                           j = 1,...,n-1 @
        pm[1:ns-1,.] = reshape(xth[1:ixth,1],ns-1,ns);
        if izz == 1;
           pm[ns,.] = ones(1,ns);
           pm = pm^2;
           pm = pm./(sumc(pm)');
        else;
           pm[ns,.] = (1 - sumc(pm))';
           pm[1,3] = 0;
        endif;
     elseif ipm == 3;  @ This section can be rewritten by user to
impose zeros
                           and ones where desired @
        if izz == 1;
             pm[1,1] = xth[1,1]^2/(1 + xth[1,1]^2);
             pm[1,2] = xth[2,1]^2/(1 + xth[2,1]^2 + xth[3,1]^2);
             pm[2,2] = xth[3,1]^2/(1 + xth[2,1]^2 + xth[3,1]^2);
             pm[2,3] = xth[4,1]^2/(1 + xth[4,1]^2);
        elseif izz == 2;
             pm[1,1] = xth[1,1];
             pm[1,2] = xth[2,1];
             pm[2,2] = xth[3,1];
             pm[2,3] = xth[4,1];
        endif;
        pm[2,1] = 1 - pm[1,1];
        pm[3,2] = 1 - pm[2,2] - pm[1,2];
        pm[3,3] = 1 - pm[2,3];
    endif;
retp(pm);
endp;
/*
==============================================================================
*/
```

```
/*
=============================================================================*/
@ Set parameters to use Gauss numerical optimizer @

     library optmum;
     #include optmum.ext;
      __btol = 1.e-06; @ This controls convergence criterion for
coefficients@
      __gtol = 1.e-06; @ This controls convergence criterion for
gradient @
      __algr = 1;     @ This chooses BFGS optimization  @
      _opalgr = 2;    @ This chooses BFGS optimization for newer
versions of
                         GAUSS @
      __miter = 150;  @ This controls the maximum number of iterations
@
      __output = 1;   @ This causes extra output to be displayed @
      __covp = 0;     @ This speeds up return from OPTMUM; note that
the
                        program makes a reparameterization to calculate
                        std. errors @

@ Next call  the GAUSS numerical optimizer @
        output off;
        {x,f,g,h} =optmum(&ofn,startval);
        output file=s3outp.txt on;

"";"";"MLE as parameterized for numerical optimization ";
"Coefficients:";x';
"";"Value of log likelihood:";;-f;
"";"Gradient vector:";g';
/*
=============================================================================
*/

/*=============================================================================
=== */
@ In general no parts of this section need be changed @

@ Reparameterize for reporting final results @
   izz = 2;
@   x = th;  @
"";"Vector is reparameterized to report final results as follows";
   "Means for each state:";x[1:ns,1]';
   If pphi > 0;
      "Autoregressive coefficients:";x[ns+1:ns+pphi,1]';
   endif;
   ncount = ns + pphi ;
   x[ncount+1:ncount+isig,1] = x[ncount+1:ncount+isig,1]^2;
   "Variances:";x[ncount+1:ncount+isig,1];
   ncount = ncount + isig;
/*
=============================================================================
*/
```

```
/*
==============================================================================
*/
proc pmth; @ This proc converts the last elements of parameter vector
                  (which relates to transition probabilities) from the
                   th[i,j]^2/{sum j th[i,j]^2} form that is used for
                   numerical estimation into the p[i,j] form that is
used
                   to calculate standard errors @
local pm;
if ipm == 1;
        x[ncount+1,1] = x[ncount+1,1]^2/(1 + x[ncount+1,1]^2);
        x[ncount+2,1] = x[ncount+2,1]^2/(1 + x[ncount+2,1]^2);
elseif ipm == 2;
        pm = zeros(ns,ns);
        pm[1:ns-1,.] = reshape(th[ncount+1:nth],ns-1,ns);
        pm[ns,.] =  ones(1,ns);
        pm = pm^2;
        pm = pm./(sumc(pm)');
        x[ncount+1:nth,1] = reshape(pm[1:ns-1,.],ns*(ns-1),1);
  elseif ipm == 3;    @ User may want to alter these next lines @
        x[ncount+1,1] = (x[ncount+1,1]^2)/(1 + x[ncount+1,1]^2);
        x[ncount+2,1] = (x[ncount+2,1]^2)/(1 + x[ncount+2,1]^2 +
x[ncount+3,1]^2);
        x[ncount+3,1] = (x[ncount+3,1]^2)/(1 + x[ncount+2,1]^2 +
x[ncount+3,1]^2);
        x[ncount+4,1] = (x[ncount+4,1]^2)/(1 + x[ncount+4,1]^2);
 endif;
retp(x);
endp;
/*
==============================================================================
*/

/*
==============================================================================
*/
@ In general no changes are necessary from here out @

call pmth;
h = (hessp(&ofn,x));
   va = eigrs(h);
kc = 2;
ks = 2;
call ofn(x);
   if minc(eigrs(h)) <= 0;
        "Negative of Hessian is not positive definite";
        "Either you have not found local maximum, or else estimates are
up "
        "against boundary condition.  In latter case, impose the
restricted "
        "params rather than estimate them to calculate standard
errors";
    else;
       h = invpd(h);
       std = diag(h)^.5;
       "For vector of coefficients parameterized as follows,";x';
```

```
       "the standard errors are";std';
    endif;

"";"------------------------------";"";
"Probabilities for primitive states";
"filtered probabilities";format /rd 1,0;
"Obs ";;
t = 0;
do until t > ps;
  i = 1;
    do until i == ns;
       "P(st-";;t;;"=";;i;;") ";;
     i = i+1;
     endo;
  t = t+1;
endo;"";
format /rd 9,4;

 skif = (skif*hp')*(eye(ps+1).*.id[.,1:ns-1]);
 skif =  seqa(nk,1,captst)~skif;skif;

 "";"smoothed probabilities";
format /rd 1,0;
"Obs ";;
i = 1;
   do until i > ns;
      "P(st = ";;i;;") ";;
   i = i+1;
   endo;
format /rd 9,4;
 skis = skis*hp';
 skis = seqa(nk,1,captst)~skis[.,1:ns];skis;

/*i = i+1;
endo;*/

/*=====================================================================
===*/
```

Code for the Simulations Reported in Table 8

Below is the code used to generate Table 8 in Maccini, Moore, and Schaller.

There are four programs, each originally written as a separate file: costx.txt, costax.txt, transx.txt, and transax.txt. Each of these files was written in MS Word, but saved as a plain text file. All of the files are written to run under the GAUSS programming language.

The files perform the following calculations:

Costx.txt calculates the cost of not responding to a regime change for all of the parameter settings for which there are no adjustment costs, that is, where $\gamma = 0$. These are rows 1 through 5 and rows 8 and 9 in table 8 of the paper. Costx.txt writes output to c:\mmscode\Costo.txt.

Transx.txt calculates the cost of not responding to a transitory shock for all of the parameter settings for which there are no adjustment costs, that is, where $\gamma = 0$. These are rows 1 through 5 and rows 8 and 9 in table 8 of the paper. Transx.txt writes output to c:\mmscode\Transo.txt.

Costax.txt calculates the cost of not responding to a regime change for all of the parameter settings for which there are adjustment costs, that is, where $\gamma > 0$. These are rows 6, 7, and 10 of the paper. Costax.txt writes output to c:\mmscode\Costao.txt.

Transax.txt calculates the cost of not responding to a transitory shock for all of the parameter settings for which there are adjustment costs, that is, where $\gamma > 0$. These are rows 6, 7, and 10 of the paper. Transax.txt writes output to c:\mmscode\Transao.txt.

The numbers reported in table 8 of the paper are obtained by taking the cost of not responding to a transitory shock, as reported in the transo.txt or transao.txt file, as a ratio to the cost of not responding to a regime change, as reported in the costo.txt or costao.txt file. (Specifically, the cost of not responding to a transitory shock for a particular parameter setting and at a particular horizon as a ratio to the cost of not responding to a regime change for the same parameter setting and at the same horizon.)

1. costx.txt.

```
/* c:\MMScode\costx.txt: to simulate the cost function and measure cost
of not responding to a persistent interest rate shock. This file is for
parameter settings in which there is no adjustment cost (i.e. gamma
=0). */

New;
Prcsn 80;

Horiz = 4;               /* horizon in years */
Repmax = 10000;
init = 26;              /* two years + two lags*/
ttmax = (Horiz*12)+ init;      /* # years times 12 + init */

sddelx = 0;
sddelw = 0;
/* Sample means as used in regressions */
ybar = 107723.68360; xbar = 107617.23788; nbar = 53305.91455; wbar1 =
110.45917;

/* Estimates of coefficients, cointegrating regression (levels) Table 7
row 1 */
```

```
coxhat = 0.237; cowhat = -301.177; copi1hat = 3845.719; copi3hat = -
5263.93;

/* parameters for MSM of real rate. PP matrix ordered as in Hamilton
(1994)*/
p11 = .982;
p12 = .018;
p13 = 0;
p21 = .007;
p22 = .985;
p23 = .008;
p31 = 0;
p32 = .042;
p33 = .958;
PP = p11~p21~p31|p12~p22~p32|p13~p23~p33;
rrv1 = -1.713|1.615|5.149;                          /* annualized r X
100 */
rrv2 = rrv1/100;
rr1m = -1 + (1+rrv2[1,1])^(1/12);                   /* monthly rates
*/
rr2m = -1 + (1+rrv2[2,1])^(1/12);
rr3m = -1 + (1+rrv2[3,1])^(1/12);
rrv = rr1m|rr2m|rr3m;
sigv1 = sqrt(3.5976)|sqrt(0.6395)|sqrt(3.8248);      /* annualized r X
100 */
sigv2 = sigv1/100;
sigv = (rrv./rrv2).*sigv2;                           /* monthly */
parmat = rrv~sigv~seqa(1,1,3);

/* Unconditional probabilities.  See Hamilton (1994) p.684 */
aax = eye(3)-pp;
aa = aax|ones(1,3);
aamat = (inv(aa'aa))*aa';
puncon = aamat[.,4];
rbar = puncon'rrv;
betabaru = inv(1+rbar);

gam1 = p11*rr1m + p12*rr2m + p13*rr3m;
gam2 = p21*rr1m + p22*rr2m + p23*rr3m;
gam3 = p31*rr1m + p32*rr2m + p33*rr3m;

/* Set parameters of the cost function */
parset = 1;               /* 1 , 2, 3, 4, 5, 6, or 7  */
do while parset <= 7;

if parset ==1;      /* Baseline */
alpha = 1;
delta = 1;
Xi = (-cowhat)*(betabaru*delta/(1-betabaru));
Theta = (alpha-coxhat)*(betabaru*delta/(1-betabaru));
gam = 0;
eta = betabaru*(theta*ybar+xi*wbar1);

elseif parset ==2;   /* Low theta, theta = thetabase/2 */
alpha = 1;
delta = 1;
Xi = (-cowhat)*(betabaru*delta/(1-betabaru));
```

```
Thetbase = (alpha-coxhat)*(betabaru*delta/(1-betabaru));
Theta = thetbase/2;
gam = 0;
eta = betabaru*(theta*ybar+xi*wbar1);

elseif parset ==3;  /* High theta, theta = 2*thetabase */
alpha = 1;
delta = 1;
Xi = (-cowhat)*(betabaru*delta/(1-betabaru));
Thetbase = (alpha-coxhat)*(betabaru*delta/(1-betabaru));
Theta = 2*thetbase;
gam = 0;
eta = betabaru*(theta*ybar+xi*wbar1);

elseif parset ==4; /* Low Xi, Xi = Xibase/2 */
alpha = 1;
delta = 1;
Xibase = (-cowhat)*(betabaru*delta/(1-betabaru));
Xi = xibase/2;
Theta = (alpha-coxhat)*(betabaru*delta/(1-betabaru));
gam = 0;
eta = betabaru*(theta*ybar+xi*wbar1);

elseif parset ==5;  /* High Xi, Xi = 2*Xibase */
alpha = 1;
delta = 1;
Xibase = (-cowhat)*(betabaru*delta/(1-betabaru));
Xi = 2*xibase;
Theta = (alpha-coxhat)*(betabaru*delta/(1-betabaru));
gam = 0;
eta = betabaru*(theta*ybar+xi*wbar1);

elseif parset ==6;    /* Ramey and West, Table 10, row (1)  */
alpha = 0.55;
theta = 0.43;
gam = 0;
delta = 0.15;
xi = 0;
eta = betabaru*(theta*ybar+xi*wbar1);

elseif parset ==7;   /* Ramey and West, Table 10, row (2)  */
alpha = 1.15;
theta = 0.21;
gam = 0;
delta = 0.58;
xi = 0;
eta = betabaru*(theta*ybar+xi*wbar1);

endif;

/* Composite parameters */
azero = -betabaru*theta;
discl = sqrt( (1+(delta/theta) + (1/betabaru))^2 - (4/betabaru) );
lamb1 = ( (1+(delta/theta) + (1/betabaru)) - discl)/2;
lamb2 = ( (1+(delta/theta) + (1/betabaru)) + discl)/2;

cfxnum =(theta - betabaru*(theta+alpha*delta));
```

```
cfxden = azero*lamb2*(1-inv(lamb2));
coefx = cfxnum/cfxden;

coefwa = -lamb1*xi/theta;
coefwb = (1-betabaru)/(1-betabaru*lamb1);
coefw = coefwa*coefwb;

copia = eta/(azero*lamb2);
copib = inv(eye(3) - inv(lamb2)*PP);
coefpi = copia*rrv'copib*PP;

coefpi1 = coefpi[1,1] - coefpi[1,2];
coefpi3 = coefpi[1,3] - coefpi[1,2];

cc = -rbar*betabaru*(theta*ybar+xi*wbar1);
bgam0 = coefpi[1,2] + cc/cfxden;

output off;
closeall;

/* Print settings */
if parset ==1;
output file = c:\MMScode\costo.txt reset;
elseif parset > 1;
output file = c:\MMScode\costo.txt on;
endif;

screen on;
if parset == 1;
print "                                                    ";
print " rbar (uncond mean)      betabar          S.D. Del X      S.D.
Del W ";
print rbar~betabaru~sddelx~sddelw;
print "                                                    ";
print "Sample means";
print "         Nbar             Xbar              ybar
wbar";
print nbar~xbar~ybar~wbar1;
print "                                                    ";
print "                                                    ";
print "***************  First set: Baseline ***********" ;
print "                                                    ";
print "        alpha             theta          gamma          Xi";
print alpha~theta~gam~Xi;
print " Coefficients in Decision Rule   ";
print "      Coef on X      Coef on pi1       Coef on pi3        Coef
on W";
print coefx~coefpi1~coefpi3~coefw;
print "      eta         ";
print eta;
print "                                                        ";


elseif parset ==2;
print "                                                    ";
print "***************  Second set: theta = thetabase/2  ***********" ;
print "                                                    ";
```

```
print "        alpha             theta          gamma          Xi";
print alpha~theta~gam~Xi;
print " Coefficients in Decision Rule   ";
print "      Coef on X      Coef on pi1       Coef on pi3        Coef
on W";
print coefx~coefpi1~coefpi3~coefw;
print "      eta       ";
print eta;
print "                                                          ";

elseif parset ==3;
print "                                                          ";
print "***************  Third set: theta = 2*thetabase ***********" ;
print "                                                          ";
print "        alpha             theta          gamma          Xi";
print alpha~theta~gam~Xi;
print " Coefficients in Decision Rule   ";
print "      Coef on X      Coef on pi1       Coef on pi3        Coef
on W";
print coefx~coefpi1~coefpi3~coefw;
print "      eta        ";
print eta;
print "                                                          ";

elseif parset ==4;
print "                                                          ";
print "***************  Fourth set: Xi = xibase/2  ***********" ;
print "                                                          ";
print "        alpha             theta          gamma          Xi";
print alpha~theta~gam~Xi;
print " Coefficients in Decision Rule   ";
print "      Coef on X      Coef on pi1       Coef on pi3        Coef
on W";
print coefx~coefpi1~coefpi3~coefw;
print "      eta         ";
print eta;
print "                                                          ";

elseif parset ==5;
print "                                                          ";
print "***************  Fifth set: Xi = 2*xibase  ***********" ;
print "                                                          ";
print "        alpha             theta          gamma          Xi";
print alpha~theta~gam~Xi;
print " Coefficients in Decision Rule   ";
print "      Coef on X      Coef on pi1       Coef on pi3        Coef
on W";
print coefx~coefpi1~coefpi3~coefw;
print "      eta        ";
print eta;
print "                                                          ";

elseif parset ==6;
print "                                                          ";
print "*************** Sixth set: R+W row1  ***********" ;
print "                                                          ";
print "        alpha             theta          gamma          Xi";
```

```
print alpha~theta~gam~Xi;
print " Coefficients in Decision Rule   ";
print "      Coef on X      Coef on pi1       Coef on pi3        Coef
on W";
print coefx~coefpi1~coefpi3~coefw;
print "      eta   ";
print eta;
print "                                                          ";

elseif parset ==7;
print "                                                          ";
print "************** Seventh set: R+W row 2  **********" ;
print "                                                          ";
print "        alpha             theta          gamma          Xi";
print alpha~theta~gam~Xi;
print " Coefficients in Decision Rule   ";
print "      Coef on X      Coef on pi1       Coef on pi3        Coef
on W";
print coefx~coefpi1~coefpi3~coefw;
print "      eta ";
print eta;
print "                                                          ";

endif;
screen off;

/************** REPETITIONS  do loop  ***************/
rep = 1;
do while rep <= repmax;
/* initialize variables: state, xx(sales), ww(P-inputs),
nn(inventories), yy (output), costs */
pit = 1|0|0;                 /* begin in low state*/
xx = xbar;
ww = wbar1;
nnt=(coefx*xx+coefw*ww +coefpi1*pit[1,1]+coefpi3*pit[3,1]+bgam0)/(1-
lamb1);
nntm1 = nnt;
nntm2 = nnt;
nnsub = nnt;
nnsm1 = nnt;
nnsm2 = nnt;

yy = xx+nnt-nntm1;
yytm1 = yy;
yysub =yy;
yysm1 = yy;

costop = xi*ww*yy + (theta/2)*(yy^2)  + (gam/2)*((yy-yytm1)^2) +
(delta/2)*((nntm1-alpha*xx)^2);

cop1 = (theta/2)*(yy^2);
cop2 = (gam/2)*((yy-yytm1)^2);
cop3 = (delta/2)*((nntm1-alpha*xx)^2);

costsub = xi*ww*yysub + (theta/2)*(yysub^2) + (gam/2)*((yysub-yysm1)^2)
+ (delta/2)*((nnsm1-alpha*xx)^2);
```

```
csub1 = (theta/2)*(yysub^2);
csub2 = (gam/2)*((yysub-yysm1)^2);
csub3 = (delta/2)*((nnsm1-alpha*xx)^2);

betat = 1;
betaprod = 1;
pdvcost = betaprod*(costop~costsub);
costsum = 0~0;

tt=1;
obs1v = nnt~costop~xx~yy;
obs2v = nnsub~costsub~yysub~(costop-costsub);
obs3v = pit'~tt;
obs4v = cop1~cop2~cop3;
obs5v = csub1~csub2~csub3;
obs6v = pdvcost;

/* tt do loop */
tt = 2;
do while tt <= ttmax;

if (tt-init) > 0;
pit = 0|0|1;
endif;

ww = ww + sddelw*rndn(1,1);
saleshoc = sddelx*rndn(1,1);

/* DECISION RULE:  nt = f( nt-1, nt-2, xt-1, wt, pit, saleshoc */
nnt = lamb1*nntm1 + coefx*xx + coefw*ww +
coefpi1*pit[1,1] + coefpi3*pit[3,1] +  bgam0 - saleshoc;

nnsub = lamb1*nnsm1 + coefx*xx + coefw*ww +
coefpi1 +  bgam0 - saleshoc;


/* generate xt and yt and calculate costs */
xx = xx + saleshoc;
yytm1 = yy;
yy = xx+nnt-nntm1;
yysm1 = yysub;
yysub = xx + nnsub-nnsm1;

costop = xi*ww*yy + (theta/2)*(yy^2) + (gam/2)*((yy-yytm1)^2) +
(delta/2)*((nntm1-alpha*xx)^2);

cop1 = (theta/2)*(yy^2);
cop2 = (gam/2)*((yy-yytm1)^2);
cop3 = (delta/2)*((nntm1-alpha*xx)^2);

costsub = xi*ww*yysub + (theta/2)*(yysub^2) + (gam/2)*((yysub-yysm1)^2)
+ (delta/2)*((nnsm1-alpha*xx)^2);

csub1 = (theta/2)*(yysub^2);
csub2 = (gam/2)*((yysub-yysm1)^2);
csub3 = (delta/2)*((nnsm1-alpha*xx)^2);
```

```
nntm2 = nntm1;
nntm1 = nnt;

nnsm2 = nnsm1;
nnsm1 = nnsub;

/* Present discounted value of costs */
betaprod = betat*betaprod;
pdvcost = betaprod*(costop~costsub);
costsum = pdvcost + costsum;
if (tt-init)<=0;
rrt = rr1m + (sigv[1,1])*rndn(1,1);
elseif (tt-init)>0;
rrt = rr3m + (sigv[3,1])*rndn(1,1);
endif;
betat = inv(1+rrt);



/* record observations */
obs1v = obs1v|nnt~costop~xx~yy;
obs2v = obs2v|nnsub~costsub~yysub~(costop-costsub);
obs3v = obs3v|pit'~tt;
obs4v = obs4v|cop1~cop2~cop3;
obs5v = obs5v|csub1~csub2~csub3;
obs6v = obs6v|pdvcost;

if (tt-init) == 0;
cost0 = costsum;
betapr0 = betaprod;
endif;

if (tt-init) == 6;
cost6 = (costsum-cost0)/betapr0;
endif;

if (tt-init) == 12;
cost12 = (costsum-cost0)/betapr0;
endif;

if (tt-init) == 24;
cost24 = (costsum-cost0)/betapr0;
endif;

tt = tt + 1;
endo;
obsv = obs1v~obs2v~obs3v~obs4v~obs5v~obs6v;
costsumv = cost6|cost12|cost24|(costsum-cost0)/betapr0;
if rep == 1;
/*        1    2     3  4          5      6      7        8     */
/*obs1v=nnt~costop~xx~yy;obs2v=nnsub~costsub~yysub~(costop-costsub);*/
/*    9,10,11  12           13   14   15             16    17    18  */
/* obs3v=pit'~tt;  obs4v = cop1~cop2~cop3; obs5v = csub1~csub2~csub3;*/
/*             19      20          */
/* obs6v = pvcostop~pvcstsub;      */
obsav = obsv/repmax;
pvcstav = costsumv/repmax;
```

```
elseif rep > 1;
obsav = obsav + (obsv/repmax);
pvcstav = pvcstav + costsumv/repmax;
endif;
rep = rep + 1;
endo;

screen on;
print " pdv rspnd Cost    pdv no rsp Cost     Col1 - Col2
Col3/Col1 ";
pctdif6 = ((pvcstav[1,1]-pvcstav[1,2])/pvcstav[1,1]);
Print pvcstav[1,.]~(pvcstav[1,1] - pvcstav[1,2])~pctdif6;
Pctdif12 = ((pvcstav[2,1]-pvcstav[2,2])/pvcstav[2,1]);
Print pvcstav[2,.]~(pvcstav[2,1] - pvcstav[2,2])~pctdif12;
Pctdif24 = ((pvcstav[3,1]-pvcstav[3,2])/pvcstav[3,1]);
Print pvcstav[3,.]~(pvcstav[3,1] - pvcstav[3,2])~pctdif24;
Pctdif48 = ((pvcstav[4,1]-pvcstav[4,2])/pvcstav[4,1]);
Print pvcstav[4,.]~(pvcstav[4,1] - pvcstav[4,2])~pctdif48;
Screen off;
parset = parset+1;
endo;
print timestr(0);
end;
```

2. costax.txt.

```
/* c:\MMScode\costax.txt: to simulate the cost function and measure
cost of not responding a persistent interest rate shock. This file is
for parameter settings in which there are adjustment costs (i.e. gamma
>0). */

New;
Prcsn 80;

Horiz = 4;               /* horizon in years */
Repmax = 10000;
init = 26;              /* two years + two lags*/
ttmax = (Horiz*12)+ init;      /* # years times 12 + init */

sddelx = 0;
sddelw = 0;

/* Sample means as used in regressions */
ybar = 107723.68360; xbar = 107617.23788; nbar = 53305.91455; wbar1 =
110.45917;

/* Estimates of coefficients, cointegrating regression (levels) Table 7
row 1*/
coxhat = 0.237; cowhat = -301.177; copi1hat = 3845.719; copi3hat = -
5263.93;
```

```
/* parameters for MSM of real rate. PP matrix ordered as in Hamilton
(1994) */
p11 = .982;
p12 = .018;
p13 = 0;
p21 = .007;
p22 = .985;
p23 = .008;
p31 = 0;
p32 = .042;
p33 = .958;
PP = p11~p21~p31|p12~p22~p32|p13~p23~p33;
rrv1 = -1.713|1.615|5.149;                          /* annualized r X
100 */
rrv2 = rrv1/100;
rr1m = -1 + (1+rrv2[1,1])^(1/12);                   /* monthly rates
*/
rr2m = -1 + (1+rrv2[2,1])^(1/12);
rr3m = -1 + (1+rrv2[3,1])^(1/12);
rrv = rr1m|rr2m|rr3m;
sigv1 = sqrt(3.5976)|sqrt(0.6395)|sqrt(3.8248);      /* annualized r X
100 */
sigv2 = sigv1/100;
sigv = (rrv./rrv2).*sigv2;                           /* monthly */
parmat = rrv~sigv~seqa(1,1,3);

/* Unconditional probabilities.  See Hamilton (1994) p.684 */
aax = eye(3)-pp;
aa = aax|ones(1,3);
aamat = (inv(aa'aa))*aa';
puncon = aamat[.,4];
rbar = puncon'rrv;
betabaru = inv(1+rbar);

gam1 = p11*rr1m + p12*rr2m + p13*rr3m;
gam2 = p21*rr1m + p22*rr2m + p23*rr3m;
gam3 = p31*rr1m + p32*rr2m + p33*rr3m;


/** Set parameters of the cost function **/
parset = 1;                        /* 1 , 2, or 3 */
do while parset <= 3;

if parset ==1;           /* low gamma (gamma=theta/7) */
alpha = 1;
delta = 1;
Xi = (-cowhat)*(betabaru*delta/(1-betabaru));
Theta = (alpha-coxhat)*(betabaru*delta/(1-betabaru));
gam = theta/7;
eta = betabaru*(theta*ybar+xi*wbar1);

elseif parset ==2;           /* high gamma (gamma=theta) */
alpha = 1;
delta = 1;
Xi = (-cowhat)*(betabaru*delta/(1-betabaru));
Theta = (alpha-coxhat)*(betabaru*delta/(1-betabaru));
gam = theta;
```

```
eta = betabaru*(theta*ybar+xi*wbar1);

elseif parset ==3;    /* Ramey and West, Table 10, row (6)  */
alpha = 1.12;
theta = 0.34;
gam = 0.05;
delta = .01;
xi = 0;
eta = betabaru*(theta*ybar+xi*wbar1);

endif;


/* Composite parameters */
aa1 = (theta + gam*(2+betabaru))/(gam*betabaru);
aa0a =(theta + gam*(1+2*betabaru) - delta*betabaru*alpha );
aa0 = aa0a/(gam*betabaru^2);

   /* roots of fourth order d.e. */
bb1 =(theta + 2*(1+betabaru)*gam)/(gam*betabaru);
bb2a =(theta*(1+betabaru) + gam*(1+4*betabaru+betabaru^2) +
delta*betabaru );
bb2 = (-1)*bb2a/(gam*betabaru^2);
bb3 = bb1/betabaru;
bb4 = (-1)/(betabaru^2);
brow1 = bb1~bb2~bb3~bb4;
brow2 = 1~0~0~0;
brow3 = 0~1~0~0;
brow4 = 0~0~1~0;
bbmat = brow1|brow2|brow3|brow4;
eiva = eig(bbmat);
eisort = sortc(eiva,1);

lamb1 = eisort[1,1];
lamb2 = eisort[2,1];
imlamb = imag(lamb1);
if imlamb == 0;
lamb3 = eisort[3,1];
lamb4 = eisort[4,1];
elseif abs(imlamb) > 0;
lamb4 = eisort[3,1];
lamb3 = eisort[4,1];
endif;

relamb1 = real(lamb1);
relamb2 = real(lamb2);
hh = sqrt(relamb1^2 + imlamb^2);
sinom = imlamb/hh;
omang = arcsin(sinom);
siomtest = sin(omang);

/* Calculate the coefficients for the Decision rule */
zeta1 = real(lamb1+lamb2);
zeta2 = -real(lamb1*lamb2);

if imlamb == 0;
/* Calculate the coefficients when the roots are real */
```

```
    /*  copia = -eta*lamb1*lamb2/((gam*betabaru)*(lamb1-lamb2));*/
copia = eta*zeta2/((gam*betabaru)*(relamb1-relamb2));
copib = betabaru*(rrv');
copi1c = relamb1*inv(eye(3) - (betabaru*relamb1)*PP);
copi2c = relamb2*inv(eye(3) - (betabaru*relamb2)*PP);
coefpi = copia*copib*(copi1c-copi2c)*PP;
coefpi1 = coefpi[1,1]-coefpi[1,2];
coefpi3 = coefpi[1,3]-coefpi[1,2];
bgam22 = coefpi[1,2];

elseif abs(imlamb) > 0;
/* Calculate the coefficients when the roots are complex */
bigs1 = rrv'pp;
s1indx = 1;
bs1new = rrv'pp;


do while abs(maxc(bs1new')) >= 10^(-15);
bs1newa = ((betabaru*hh)^s1indx)/(sin(omang));
bs1newb = (sin(omang*(s1indx+1)))*rrv'(pp^(s1indx+1));
bs1new = bs1newa*bs1newb;
bigs1 = bigs1 + bs1new;
s1indx = s1indx +1;
endo;

coefpi1 = (-eta/gam)*(hh^2)*(bigs1[1,1]-bigs1[1,2]);
coefpi3 = (-eta/gam)*(hh^2)*(bigs1[1,3]-bigs1[1,2]);
bgam22 = (-eta/gam)*(hh^2)*bigs1[1,2];

endif;

cfx1 = -(betabaru^2)*zeta2;
cfx2 = 1 - betabaru*zeta1 + (betabaru^2)*(-zeta2);
coefx = (cfx1/cfx2)*(-1 + aa1 - aa0 + betabaru^(-2));

coefwa = zeta2*xi/gam;
coefwb = (1-betabaru)/(1-betabaru*zeta1-(betabaru^2)*zeta2);
coefw = coefwa*coefwb;

cc = -rbar*betabaru*(theta*ybar+xi*wbar1);
bgam0 = bgam22 + zeta2*cc/(gam*(1-betabaru*zeta1-(betabaru^2)*zeta2));

output off;
closeall;

/* Print settings */
if parset ==1;
output file = c:\MMScode\costao.txt reset;
elseif parset > 1;
output file = c:\MMScode\costao.txt on;
endif;

screen on;

if parset == 1;
print "                                                    ";
```

```
print " rbar (uncond mean)      betabar          S.D. Del X      S.D.
Del W ";
print rbar~betabaru~sddelx~sddelw;
print "                                                    ";
print "Sample means";
print "         Nbar             Xbar              ybar
wbar";
print nbar~xbar~ybar~wbar1;
print "                                                    ";
print "                                                    ";
print "***************  First set: low gamma  (gamma=theta/7)
***********" ;
print "                                                    ";
print "        alpha             theta          gamma          Xi";
print alpha~theta~gam~Xi;
print " Coefficients in Decision Rule (and eta)    ";
print "      Coef on X      Coef on pi1       Coef on pi3        Coef
on W";
print coefx~coefpi1~coefpi3~coefw;
print "      eta          ";
print eta;
print "                                                          ";

elseif parset == 2;
print "                                                          ";
print "***************  Second set: high gamma  (gamma=theta)
***********" ;
print "                                                    ";
print "        alpha             theta          gamma          Xi";
print alpha~theta~gam~Xi;
print " Coefficients in Decision Rule (and eta)    ";
print "      Coef on X      Coef on pi1       Coef on pi3        Coef
on W";
print coefx~coefpi1~coefpi3~coefw;
print "      eta         ";
print eta;
print "                                                    ";

elseif parset ==3;
print "                                                    ";
print "****** Third set: Ramey and West row 6 ***********";
print "                                                     ";
print "        alpha             theta          gamma          Xi";
print alpha~theta~gam~Xi;
print " Coefficients in Decision Rule (and eta)    ";
print "      Coef on X      Coef on pi1       Coef on pi3        Coef
on W";
print coefx~coefpi1~coefpi3~coefw;
print "      eta            ";
print eta;
print "                                                    ";

endif;
screen off;

/************** REPETITIONS  do loop  ***************/
rep = 1;
```

```
do while rep <= repmax;
/* initialize variables: state, xx(sales), ww(P-inputs),
nn(inventories), yy (output), costs */
pit = 1|0|0;                 /* begin in low state*/
xx = xbar;
ww = wbar1;
nnt=(coefx*xx+coefw*ww +coefpi1*pit[1,1]+coefpi3*pit[3,1]+bgam0)/(1-
zeta1-zeta2);
nntm1 = nnt;
nntm2 = nnt;
nnsub = nnt;
nnsm1 = nnt;
nnsm2 = nnt;

yy = xx+nnt-nntm1;
yytm1 = yy;
yysub =yy;
yysm1 = yy;

costop = xi*ww*yy + (theta/2)*(yy^2)  + (gam/2)*((yy-yytm1)^2) +
(delta/2)*((nntm1-alpha*xx)^2);

cop1 = (theta/2)*(yy^2);
cop2 = (gam/2)*((yy-yytm1)^2);
cop3 = (delta/2)*((nntm1-alpha*xx)^2);

costsub = xi*ww*yysub + (theta/2)*(yysub^2) + (gam/2)*((yysub-yysm1)^2)
+ (delta/2)*((nnsm1-alpha*xx)^2);

csub1 = (theta/2)*(yysub^2);
csub2 = (gam/2)*((yysub-yysm1)^2);
csub3 = (delta/2)*((nnsm1-alpha*xx)^2);

betat = 1;
betaprod = 1;
pdvcost = betaprod*(costop~costsub);
costsum = 0~0;

tt=1;
obs1v = nnt~costop~xx~yy;
obs2v = nnsub~costsub~yysub~(costop-costsub);
obs3v = pit'~tt;
obs4v = cop1~cop2~cop3;
obs5v = csub1~csub2~csub3;
obs6v = pdvcost;

/* tt do loop */
tt = 2;
do while tt <= ttmax;

if (tt-init) > 0;
pit = 0|0|1;
endif;

ww = ww + sddelw*rndn(1,1);
saleshoc = sddelx*rndn(1,1);
```

```
/* DECISION RULE:  nt = f( nt-1, nt-2, xt-1, wt, pit, saleshoc) */
nnt = zeta1*nntm1 + zeta2*nntm2 + coefx*xx + coefw*ww +
coefpi1*pit[1,1] + coefpi3*pit[3,1] +  bgam0 - saleshoc;

nnsub = zeta1*nnsm1 + zeta2*nnsm2 + coefx*xx + coefw*ww +
coefpi1 +  bgam0 - saleshoc;


/* generate xt and yt and calculate costs */
xx = xx + saleshoc;
yytm1 = yy;
yy = xx+nnt-nntm1;
yysm1 = yysub;
yysub = xx + nnsub-nnsm1;

costop = xi*ww*yy + (theta/2)*(yy^2) + (gam/2)*((yy-yytm1)^2) +
(delta/2)*((nntm1-alpha*xx)^2);

cop1 = (theta/2)*(yy^2);
cop2 = (gam/2)*((yy-yytm1)^2);
cop3 = (delta/2)*((nntm1-alpha*xx)^2);

costsub = xi*ww*yysub + (theta/2)*(yysub^2) + (gam/2)*((yysub-yysm1)^2)
+ (delta/2)*((nnsm1-alpha*xx)^2);

csub1 = (theta/2)*(yysub^2);
csub2 = (gam/2)*((yysub-yysm1)^2);
csub3 = (delta/2)*((nnsm1-alpha*xx)^2);

nntm2 = nntm1;
nntm1 = nnt;

nnsm2 = nnsm1;
nnsm1 = nnsub;

/* Present discounted value of costs */
betaprod = betat*betaprod;
pdvcost = betaprod*(costop~costsub);
costsum = pdvcost + costsum;
if (tt-init)<=0;
rrt = rr1m + (sigv[1,1])*rndn(1,1);
elseif (tt-init)>0;
rrt = rr3m + (sigv[3,1])*rndn(1,1);
endif;
betat = inv(1+rrt);



/* record observations */
obs1v = obs1v|nnt~costop~xx~yy;
obs2v = obs2v|nnsub~costsub~yysub~(costop-costsub);
obs3v = obs3v|pit'~tt;
obs4v = obs4v|cop1~cop2~cop3;
obs5v = obs5v|csub1~csub2~csub3;
obs6v = obs6v|pdvcost;

if (tt-init) == 0;
```

```
cost0 = costsum;
betapr0 = betaprod;
endif;

if (tt-init) == 6;
cost6 = (costsum-cost0)/betapr0;
endif;

if (tt-init) == 12;
cost12 = (costsum-cost0)/betapr0;
endif;

if (tt-init) == 24;
cost24 = (costsum-cost0)/betapr0;
endif;

tt = tt + 1;
endo;
obsv = obs1v~obs2v~obs3v~obs4v~obs5v~obs6v;
costsumv = cost6|cost12|cost24|(costsum-cost0)/betapr0;
if rep == 1;
/*        1    2     3  4          5      6      7        8     */
/*obs1v=nnt~costop~xx~yy;obs2v=nnsub~costsub~yysub~(costop-costsub);*/
/*    9,10,11  12           13   14   15             16    17    18  */
/* obs3v=pit'~tt;  obs4v = cop1~cop2~cop3; obs5v = csub1~csub2~csub3;*/
/*             19      20          */
/* obs6v = pvcostop~pvcstsub;      */
obsav = obsv/repmax;
pvcstav = costsumv/repmax;
elseif rep > 1;
obsav = obsav + (obsv/repmax);
pvcstav = pvcstav + costsumv/repmax;
endif;
rep = rep + 1;
endo;

screen on;
print " pdv rspnd Cost   pdv no rsp Cost    Col1 - Col2       Col3/Col1
";
pctdif6 = ((pvcstav[1,1]-pvcstav[1,2])/pvcstav[1,1]);
Print pvcstav[1,.]~(pvcstav[1,1] - pvcstav[1,2])~pctdif6;
Pctdif12 = ((pvcstav[2,1]-pvcstav[2,2])/pvcstav[2,1]);
Print pvcstav[2,.]~(pvcstav[2,1] - pvcstav[2,2])~pctdif12;
Pctdif24 = ((pvcstav[3,1]-pvcstav[3,2])/pvcstav[3,1]);
Print pvcstav[3,.]~(pvcstav[3,1] - pvcstav[3,2])~pctdif24;
Pctdif48 = ((pvcstav[4,1]-pvcstav[4,2])/pvcstav[4,1]);
Print pvcstav[4,.]~(pvcstav[4,1] - pvcstav[4,2])~pctdif48;
parset = parset+1;
endo;
print timestr(0);
end;
```

3. transx.txt.

```
/* c:\MMScode\transx.txt: to simulate the cost function and  measure
cost of not responding to a transitory interest rate shock. This file
is for parameter settings in which there are no adjustment costs (i.e.
gamma = 0). */

New;
Prcsn 80;

Horiz = 4;               /* horizon in years */
Repmax = 10000;
init = 26;              /* two years + two lags*/

ttmax = (Horiz*12)+ init;      /* # years times 12 + init */

sddelx = 0;
sddelw = 0;

/* Sample means as used in regressions */
ybar = 107723.68360; xbar = 107617.23788; nbar = 53305.91455; wbar1 =
110.45917;

/* Estimates of coefficients, cointegrating regression (levels) Table 7
row 1 */
coxhat = 0.237; cowhat = -301.177; copi1hat = 3845.719; copi3hat = -
5263.93;

/*parameters for MSM of real rate. PP matrix ordered as in Hamilton
(1994)*/
p11 = .982;
p12 = .018;
p13 = 0;
p21 = .007;
p22 = .985;
p23 = .008;
p31 = 0;
p32 = .042;
p33 = .958;
PP = p11~p21~p31|p12~p22~p32|p13~p23~p33;
rrv1 = -1.713|1.615|5.149;                          /* annualized r X
100 */
rrv2 = rrv1/100;
rr1m = -1 + (1+rrv2[1,1])^(1/12);                   /* monthly rates
*/
rr2m = -1 + (1+rrv2[2,1])^(1/12);
rr3m = -1 + (1+rrv2[3,1])^(1/12);
rrv = rr1m|rr2m|rr3m;
sigv1 = sqrt(3.5976)|sqrt(0.6395)|sqrt(3.8248);      /* annualized r X
100 */
sigv2 = sigv1/100;
sigv = (rrv./rrv2).*sigv2;                           /* monthly */
parmat = rrv~sigv~seqa(1,1,3);

/* Unconditional probabilities.  See Hamilton (1994) p.684 */
aax = eye(3)-pp;
```

```
aa = aax|ones(1,3);
aamat = (inv(aa'aa))*aa';
puncon = aamat[.,4];
rbar = puncon'rrv;
betabaru = inv(1+rbar);

gam1 = p11*rr1m + p12*rr2m + p13*rr3m;
gam2 = p21*rr1m + p22*rr2m + p23*rr3m;
gam3 = p31*rr1m + p32*rr2m + p33*rr3m;

/* Set parameters of the cost function */
parset = 1;               /* 1 , 2, 3, 4, 5, 6, or 7  */
do while parset <= 7;

if parset ==1;      /* Baseline */
alpha = 1;
delta = 1;
Xi = (-cowhat)*(betabaru*delta/(1-betabaru));
Theta = (alpha-coxhat)*(betabaru*delta/(1-betabaru));
gam = 0;
eta = betabaru*(theta*ybar+xi*wbar1);

elseif parset ==2;   /* Low theta, theta = thetabase/2 */
alpha = 1;
delta = 1;
Xi = (-cowhat)*(betabaru*delta/(1-betabaru));
Thetbase = (alpha-coxhat)*(betabaru*delta/(1-betabaru));
Theta = thetbase/2;
gam = 0;
eta = betabaru*(theta*ybar+xi*wbar1);

elseif parset ==3;  /* High theta, theta = 2*thetabase */
alpha = 1;
delta = 1;
Xi = (-cowhat)*(betabaru*delta/(1-betabaru));
Thetbase = (alpha-coxhat)*(betabaru*delta/(1-betabaru));
Theta = 2*thetbase;
gam = 0;
eta = betabaru*(theta*ybar+xi*wbar1);

elseif parset ==4; /* Low Xi, Xi = Xibase/2 */
alpha = 1;
delta = 1;
Xibase = (-cowhat)*(betabaru*delta/(1-betabaru));
Xi = xibase/2;
Theta = (alpha-coxhat)*(betabaru*delta/(1-betabaru));
gam = 0;
eta = betabaru*(theta*ybar+xi*wbar1);

elseif parset ==5;  /* High Xi, Xi = 2*Xibase */
alpha = 1;
delta = 1;
Xibase = (-cowhat)*(betabaru*delta/(1-betabaru));
Xi = 2*xibase;
Theta = (alpha-coxhat)*(betabaru*delta/(1-betabaru));
gam = 0;
eta = betabaru*(theta*ybar+xi*wbar1);
```

```
elseif parset ==6;    /* Ramey and West, Table 10, row (1)  */
alpha = 0.55;
theta = 0.43;
gam = 0;
delta = 0.15;
xi = 0;
eta = betabaru*(theta*ybar+xi*wbar1);

elseif parset ==7;   /* Ramey and West, Table 10, row (2)  */
alpha = 1.15;
theta = 0.21;
gam = 0;
delta = 0.58;
xi = 0;
eta = betabaru*(theta*ybar+xi*wbar1);

endif;


/* Composite parameters */
azero = -betabaru*theta;
discl = sqrt( (1+(delta/theta) + (1/betabaru))^2 - (4/betabaru) );
lamb1 = ( (1+(delta/theta) + (1/betabaru)) - discl)/2;
lamb2 = ( (1+(delta/theta) + (1/betabaru)) + discl)/2;

cfxnum =(theta - betabaru*(theta+alpha*delta));
cfxden = azero*lamb2*(1-inv(lamb2));
coefx = cfxnum/cfxden;

coefwa = -lamb1*xi/theta;
coefwb = (1-betabaru)/(1-betabaru*lamb1);
coefw = coefwa*coefwb;

copia = eta/(azero*lamb2);
copib = inv(eye(3) - inv(lamb2)*PP);
coefpi = copia*rrv'copib*PP;

coefpi1 = coefpi[1,1] - coefpi[1,2];
coefpi3 = coefpi[1,3] - coefpi[1,2];

cc = -rbar*betabaru*(theta*ybar+xi*wbar1);
bgam0 = coefpi[1,2] + cc/cfxden;

output off;
closeall;

/* Print settings */
if parset ==1;
output file = c:\MMScode\transo.txt reset;
elseif parset > 1;
output file = c:\MMScode\transo.txt on;
endif;

screen on;
if parset == 1;
print "                                                    ";
```

```
print " rbar (uncond mean)      betabar          S.D. Del X      S.D.
Del W ";
print rbar~betabaru~sddelx~sddelw;
print "                                                  ";
print "Sample means";
print "         Nbar             Xbar              ybar
wbar";
print nbar~xbar~ybar~wbar1;
print "                                                  ";
print "                                                  ";
print "***************  First set: Baseline ***********" ;
print "                                                  ";
print "        alpha             theta          gamma          Xi";
print alpha~theta~gam~Xi;
print " Coefficients in Decision Rule   ";
print "      Coef on X      Coef on pi1       Coef on pi3        Coef
on W";
print coefx~coefpi1~coefpi3~coefw;
print "      eta          ";
print eta;
print "                                                        ";


elseif parset ==2;
print "                                                  ";
print "***************  Second set: theta = thetabase/2  ***********" ;
print "                                                  ";
print "        alpha             theta          gamma          Xi";
print alpha~theta~gam~Xi;
print " Coefficients in Decision Rule   ";
print "      Coef on X      Coef on pi1       Coef on pi3        Coef
on W";
print coefx~coefpi1~coefpi3~coefw;
print "      eta      ";
print eta;
print "                                                        ";

elseif parset ==3;
print "                                                  ";
print "***************  Third set: theta = 2*thetabase ***********" ;
print "                                                  ";
print "        alpha             theta          gamma          Xi";
print alpha~theta~gam~Xi;
print " Coefficients in Decision Rule   ";
print "      Coef on X      Coef on pi1       Coef on pi3        Coef
on W";
print coefx~coefpi1~coefpi3~coefw;
print "      eta          ";
print eta;
print "                                                        ";

elseif parset ==4;
print "                                                  ";
print "***************  Fourth set: Xi = xibase/2  ***********" ;
print "                                                  ";
print "        alpha             theta          gamma          Xi";
print alpha~theta~gam~Xi;
```

```
print " Coefficients in Decision Rule   ";
print "      Coef on X      Coef on pi1       Coef on pi3        Coef
on W";
print coefx~coefpi1~coefpi3~coefw;
print "      eta         ";
print eta;
print "                                                          ";

elseif parset ==5;
print "                                                          ";
print "***************  Fifth set: Xi = 2*xibase  ************" ;
print "                                                          ";
print "        alpha             theta          gamma          Xi";
print alpha~theta~gam~Xi;
print " Coefficients in Decision Rule   ";
print "      Coef on X      Coef on pi1       Coef on pi3        Coef
on W";
print coefx~coefpi1~coefpi3~coefw;
print "      eta        ";
print eta;
print "                                                          ";

elseif parset ==6;
print "                                                          ";
print "*************** Sixth set: R+W row1  ************" ;
print "                                                          ";
print "        alpha             theta          gamma          Xi";
print alpha~theta~gam~Xi;
print " Coefficients in Decision Rule   ";
print "      Coef on X      Coef on pi1       Coef on pi3        Coef
on W";
print coefx~coefpi1~coefpi3~coefw;
print "      eta   ";
print eta;
print "                                                          ";

elseif parset ==7;
print "                                                          ";
print "**************** Seventh set: R+W row 2  ************" ;
print "                                                          ";
print "        alpha             theta          gamma          Xi";
print alpha~theta~gam~Xi;
print " Coefficients in Decision Rule   ";
print "      Coef on X      Coef on pi1       Coef on pi3        Coef
on W";
print coefx~coefpi1~coefpi3~coefw;
print "      eta ";
print eta;
print "                                                          ";

endif;
screen off;


/************** REPETITIONS  do loop  ***************/
rep = 1;
do while rep <= repmax;
```

```
/* initialize variables: state, xx(sales), ww(P-inputs),
nn(inventories), yy (output), costs */
stvec = 1|0|0;                  /* begin in low state*/
pit = .974|.026|0;
pitctm1 = pit;
pis = pit;
pitcsm1 = pit;

xx = xbar;
ww = wbar1;
nnt=(coefx*xx+coefw*ww +coefpi1*pit[1,1]+coefpi3*pit[3,1]+bgam0)/(1-
lamb1);
nntm1 = nnt;
nntm2 = nnt;
nnsub = nnt;
nnsm1 = nnt;
nnsm2 = nnt;

yy = xx+nnt-nntm1;
yytm1 = yy;
yysub =yy;
yysm1 = yy;

costop = xi*ww*yy + (theta/2)*(yy^2)  + (gam/2)*((yy-yytm1)^2) +
(delta/2)*((nntm1-alpha*xx)^2);

cop1 = (theta/2)*(yy^2);
cop2 = (gam/2)*((yy-yytm1)^2);
cop3 = (delta/2)*((nntm1-alpha*xx)^2);

costsub = xi*ww*yysub + (theta/2)*(yysub^2) + (gam/2)*((yysub-yysm1)^2)
+ (delta/2)*((nnsm1-alpha*xx)^2);

csub1 = (theta/2)*(yysub^2);
csub2 = (gam/2)*((yysub-yysm1)^2);
csub3 = (delta/2)*((nnsm1-alpha*xx)^2);

betat = 1;
betaprod = 1;
pdvcost = betaprod*(costop~costsub);
costsum = 0~0;

tt=1;
obs1v = nnt~costop~xx~yy;
obs2v = nnsub~costsub~yysub~(costop-costsub);
obs3v = pit'~tt;
obs4v = cop1~cop2~cop3;
obs5v = csub1~csub2~csub3;
obs6v = pdvcost;

/* tt do loop */
tt = 2;
do while tt <= ttmax;

/* draw shock to (constant mean) interest rate */
rrs = rr1m + (sigv[1,1])*rndn(1,1);
rshock = 0;
```

```
if (tt-init) == 1;
rshock = sigv[1,1];
endif;
rrt = rrs + rshock;

/* update probabilities (respond) */
f1den = inv(sigv[1,1]*sqrt(2*pi));
frc1 = f1den*exp(-((rrt - rrv[1,1])^2) /(2*sigv[1,1]^2) );
f2den = inv(sigv[2,1]*sqrt(2*pi));
frc2 = f2den*exp(-((rrt - rrv[2,1])^2) /(2*sigv[2,1]^2) );
f3den = inv(sigv[3,1]*sqrt(2*pi));
frc3 = f3den*exp(-((rrt - rrv[3,1])^2) /(2*sigv[3,1]^2) );
frv = frc1|frc2|frc3;
dent = pitctm1'frv;
p1t = pitctm1[1,1]*frc1/dent;
p2t = pitctm1[2,1]*frc2/dent;
p3t = pitctm1[3,1]*frc3/dent;
pit = p1t|p2t|p3t;
pitctm1 = pp*pit;

/* update probabilities (not respond) */
frc1s = f1den*exp(-((rrs - rrv[1,1])^2) /(2*sigv[1,1]^2) );
frc2s = f2den*exp(-((rrs - rrv[2,1])^2) /(2*sigv[2,1]^2) );
frc3s = f3den*exp(-((rrs - rrv[3,1])^2) /(2*sigv[3,1]^2) );
frvs = frc1s|frc2s|frc3s;
dent = pitcsm1'frvs;
p1s = pitcsm1[1,1]*frc1s/dent;
p2s = pitcsm1[2,1]*frc2s/dent;
p3s = pitcsm1[3,1]*frc3s/dent;
pis = p1s|p2s|p3s;
pitcsm1 = pp*pis;

ww = ww + sddelw*rndn(1,1);
saleshoc = sddelx*rndn(1,1);

/* DECISION RULE:  nt = f( nt-1, nt-2, xt-1, wt, pit, saleshoc)  */
nnt = lamb1*nntm1 + coefx*xx + coefw*ww +
coefpi1*pit[1,1] + coefpi3*pit[3,1] +  bgam0 - saleshoc;

nnsub = lamb1*nnsm1 + coefx*xx + coefw*ww +
coefpi1*pis[1,1] + coefpi3*pis[3,1] + bgam0 - saleshoc;


/* generate xt and yt and calculate costs */
xx = xx + saleshoc;
yytm1 = yy;
yy = xx+nnt-nntm1;
yysm1 = yysub;
yysub = xx + nnsub-nnsm1;

costop = xi*ww*yy + (theta/2)*(yy^2) + (gam/2)*((yy-yytm1)^2) +
(delta/2)*((nntm1-alpha*xx)^2);

cop1 = (theta/2)*(yy^2);
cop2 = (gam/2)*((yy-yytm1)^2);
cop3 = (delta/2)*((nntm1-alpha*xx)^2);
```

```
costsub = xi*ww*yysub + (theta/2)*(yysub^2) + (gam/2)*((yysub-yysm1)^2)
+ (delta/2)*((nnsm1-alpha*xx)^2);

csub1 = (theta/2)*(yysub^2);
csub2 = (gam/2)*((yysub-yysm1)^2);
csub3 = (delta/2)*((nnsm1-alpha*xx)^2);

nntm2 = nntm1;
nntm1 = nnt;

nnsm2 = nnsm1;
nnsm1 = nnsub;

/* Present discounted value of costs     */
betaprod = betat*betaprod;
pdvcost = betaprod*(costop~costsub);
costsum = pdvcost + costsum;
betat = inv(1+rrt);



/* record observations */
obs1v = obs1v|nnt~costop~xx~yy;
obs2v = obs2v|nnsub~costsub~yysub~(costop-costsub);
obs3v = obs3v|pit'~tt;
obs4v = obs4v|cop1~cop2~cop3;
obs5v = obs5v|csub1~csub2~csub3;
obs6v = obs6v|pdvcost;

if (tt-init) == 0;
cost0 = costsum;
betapr0 = betaprod;
endif;

if (tt-init) == 6;
cost6 = (costsum-cost0)/betapr0;
endif;

if (tt-init) == 12;
cost12 = (costsum-cost0)/betapr0;
endif;

if (tt-init) == 24;
cost24 = (costsum-cost0)/betapr0;
endif;

tt = tt + 1;
endo;
obsv = obs1v~obs2v~obs3v~obs4v~obs5v~obs6v;
costsumv = cost6|cost12|cost24|(costsum-cost0)/betapr0;
if rep == 1;
/*        1    2     3  4          5      6      7        8     */
/*obs1v=nnt~costop~xx~yy;obs2v=nnsub~costsub~yysub~(costop-costsub);*/
/*    9,10,11  12           13   14   15             16    17    18  */
/* obs3v=pit'~tt;  obs4v = cop1~cop2~cop3; obs5v = csub1~csub2~csub3;*/
/*             19      20          */
```

```
/* obs6v = pvcostop~pvcstsub;      */
obsav = obsv/repmax;
pvcstav = costsumv/repmax;
elseif rep > 1;
obsav = obsav + (obsv/repmax);
pvcstav = pvcstav + costsumv/repmax;
endif;
rep = rep + 1;
endo;


screen on;
print "    pdv respond       pdv not respond    Col1 - Col2
Col3/Col1 ";
pctdif6 = ((pvcstav[1,1]-pvcstav[1,2])/pvcstav[1,1]);
Print pvcstav[1,.]~(pvcstav[1,1] - pvcstav[1,2])~pctdif6;
Pctdif12 = ((pvcstav[2,1]-pvcstav[2,2])/pvcstav[2,1]);
Print pvcstav[2,.]~(pvcstav[2,1] - pvcstav[2,2])~pctdif12;
Pctdif24 = ((pvcstav[3,1]-pvcstav[3,2])/pvcstav[3,1]);
Print pvcstav[3,.]~(pvcstav[3,1] - pvcstav[3,2])~pctdif24;
Pctdif48 = ((pvcstav[4,1]-pvcstav[4,2])/pvcstav[4,1]);
Print pvcstav[4,.]~(pvcstav[4,1] - pvcstav[4,2])~pctdif48;
parset = parset+1;
endo;
print timestr(0);
end;
```

4. transax.txt.

```
/* c:\MMScode\transax.txt: to simulate the cost function and  measure
cost of not responding to a transitory interest rate shock. This file
is for parameter settings in which there are adjustment costs (i.e.
gamma > 0). */

New;
Prcsn 80;

Horiz = 4;               /* horizon in years */
Repmax = 10000;
init = 26;              /* two years + two lags*/

ttmax = (Horiz*12)+ init;      /* # years times 12 + init */

sddelx = 0;
sddelw = 0;

/* Sample means as used in regressions */
ybar = 107723.68360; xbar = 107617.23788; nbar = 53305.91455; wbar1 =
110.45917;

/* Estimates of coefficients, cointegrating regression (levels) table 7
row 1 */
```

```
coxhat = 0.237; cowhat = -301.177; copi1hat = 3845.719; copi3hat = -
5263.93;

/* parameters for MSM of real rate. PP matrix ordered as in Hamilton
(1994) */
p11 = .982;
p12 = .018;
p13 = 0;
p21 = .007;
p22 = .985;
p23 = .008;
p31 = 0;
p32 = .042;
p33 = .958;
PP = p11~p21~p31|p12~p22~p32|p13~p23~p33;
rrv1 = -1.713|1.615|5.149;                          /* annualized r X
100 */
rrv2 = rrv1/100;
rr1m = -1 + (1+rrv2[1,1])^(1/12);                   /* monthly rates
*/
rr2m = -1 + (1+rrv2[2,1])^(1/12);
rr3m = -1 + (1+rrv2[3,1])^(1/12);
rrv = rr1m|rr2m|rr3m;
sigv1 = sqrt(3.5976)|sqrt(0.6395)|sqrt(3.8248);      /* annualized r X
100 */
sigv2 = sigv1/100;
sigv = (rrv./rrv2).*sigv2;                           /* monthly */
parmat = rrv~sigv~seqa(1,1,3);

/* Unconditional probabilities.  See Hamilton (1994) p.684 */
aax = eye(3)-pp;
aa = aax|ones(1,3);
aamat = (inv(aa'aa))*aa';
puncon = aamat[.,4];
rbar = puncon'rrv;
betabaru = inv(1+rbar);

gam1 = p11*rr1m + p12*rr2m + p13*rr3m;
gam2 = p21*rr1m + p22*rr2m + p23*rr3m;
gam3 = p31*rr1m + p32*rr2m + p33*rr3m;

/* Set parameters of the cost function */
parset = 1;                        /* 1 , 2, or 3 */
do while parset <= 3;

if parset ==1;           /* low gamma (gamma=theta/7) */
alpha = 1;
delta = 1;
Xi = (-cowhat)*(betabaru*delta/(1-betabaru));
Theta = (alpha-coxhat)*(betabaru*delta/(1-betabaru));
gam = theta/7;
eta = betabaru*(theta*ybar+xi*wbar1);

elseif parset ==2;           /* high gamma (gamma=theta) */
alpha = 1;
delta = 1;
Xi = (-cowhat)*(betabaru*delta/(1-betabaru));
```

```
Theta = (alpha-coxhat)*(betabaru*delta/(1-betabaru));
gam = theta;
eta = betabaru*(theta*ybar+xi*wbar1);

elseif parset ==3;    /* Ramey and West, Table 10, row (6)  */
alpha = 1.12;
theta = 0.34;
gam = 0.05;
delta = .01;
xi = 0;
eta = betabaru*(theta*ybar+xi*wbar1);

endif;


/* Composite parameters */
aa1 = (theta + gam*(2+betabaru))/(gam*betabaru);
aa0a =(theta + gam*(1+2*betabaru) - delta*betabaru*alpha );
aa0 = aa0a/(gam*betabaru^2);

   /* roots of fourth order d.e. */
bb1 =(theta + 2*(1+betabaru)*gam)/(gam*betabaru);
bb2a =(theta*(1+betabaru) + gam*(1+4*betabaru+betabaru^2) +
delta*betabaru );
bb2 = (-1)*bb2a/(gam*betabaru^2);
bb3 = bb1/betabaru;
bb4 = (-1)/(betabaru^2);
brow1 = bb1~bb2~bb3~bb4;
brow2 = 1~0~0~0;
brow3 = 0~1~0~0;
brow4 = 0~0~1~0;
bbmat = brow1|brow2|brow3|brow4;
eiva = eig(bbmat);
eisort = sortc(eiva,1);

lamb1 = eisort[1,1];
lamb2 = eisort[2,1];
imlamb = imag(lamb1);
if imlamb == 0;
lamb3 = eisort[3,1];
lamb4 = eisort[4,1];
elseif abs(imlamb) > 0;
lamb4 = eisort[3,1];
lamb3 = eisort[4,1];
endif;

relamb1 = real(lamb1);
relamb2 = real(lamb2);
hh = sqrt(relamb1^2 + imlamb^2);
sinom = imlamb/hh;
omang = arcsin(sinom);
siomtest = sin(omang);

/* Calculate the coefficients for the Decision rule */
zeta1 = real(lamb1+lamb2);
zeta2 = -real(lamb1*lamb2);
```

```
if imlamb == 0;
/* Calculate the coefficients when the roots are real */
    /*  copia = -eta*lamb1*lamb2/((gam*betabaru)*(lamb1-lamb2));*/
copia = eta*zeta2/((gam*betabaru)*(relamb1-relamb2));
copib = betabaru*(rrv');
copi1c = relamb1*inv(eye(3) - (betabaru*relamb1)*PP);
copi2c = relamb2*inv(eye(3) - (betabaru*relamb2)*PP);
coefpi = copia*copib*(copi1c-copi2c)*PP;
coefpi1 = coefpi[1,1]-coefpi[1,2];
coefpi3 = coefpi[1,3]-coefpi[1,2];
bgam22 = coefpi[1,2];

elseif abs(imlamb) > 0;
/* Calculate the coefficients when the roots are complex */
bigs1 = rrv'pp;
s1indx = 1;
bs1new = rrv'pp;


do while abs(maxc(bs1new')) >= 10^(-15);
bs1newa = ((betabaru*hh)^s1indx)/(sin(omang));
bs1newb = (sin(omang*(s1indx+1)))*rrv'(pp^(s1indx+1));
bs1new = bs1newa*bs1newb;
bigs1 = bigs1 + bs1new;
s1indx = s1indx +1;
endo;

coefpi1 = (-eta/gam)*(hh^2)*(bigs1[1,1]-bigs1[1,2]);
coefpi3 = (-eta/gam)*(hh^2)*(bigs1[1,3]-bigs1[1,2]);
bgam22 = (-eta/gam)*(hh^2)*bigs1[1,2];

endif;

cfx1 = -(betabaru^2)*zeta2;
cfx2 = 1 - betabaru*zeta1 + (betabaru^2)*(-zeta2);
coefx = (cfx1/cfx2)*(-1 + aa1 - aa0 + betabaru^(-2));

coefwa = zeta2*xi/gam;
coefwb = (1-betabaru)/(1-betabaru*zeta1-(betabaru^2)*zeta2);
coefw = coefwa*coefwb;

cc = -rbar*betabaru*(theta*ybar+xi*wbar1);
bgam0 = bgam22 + zeta2*cc/(gam*(1-betabaru*zeta1-(betabaru^2)*zeta2));

output off;
closeall;

/** Print settings **/

if parset ==1;
output file = c:\MMScode\transao.txt reset;
elseif parset > 1;
output file = c:\MMScode\transao.txt on;
endif;

screen on;
if parset == 1;
```

```
print "                                                    ";
print " rbar (uncond mean)      betabar          S.D. Del X      S.D.
Del W ";
print rbar~betabaru~sddelx~sddelw;
print "                                                    ";
print "Sample means";
print "         Nbar             Xbar              ybar
wbar";
print nbar~xbar~ybar~wbar1;
print "                                                    ";
print "                                                    ";
print "***************  First set: low gamma  (gamma=theta/7)
***********" ;
print "                                                    ";
print "        alpha             theta          gamma          Xi";
print alpha~theta~gam~Xi;
print " Coefficients in Decision Rule (and eta)    ";
print "      Coef on X      Coef on pi1       Coef on pi3        Coef
on W";
print coefx~coefpi1~coefpi3~coefw;
print "      eta         ";
print eta;
print "                                                  ";

elseif parset == 2;
print "                                                  ";
print "***************  Second set: high gamma  (gamma=theta)
***********" ;
print "                                                    ";
print "        alpha             theta          gamma          Xi";
print alpha~theta~gam~Xi;
print " Coefficients in Decision Rule (and eta)    ";
print "      Coef on X      Coef on pi1       Coef on pi3        Coef
on W";
print coefx~coefpi1~coefpi3~coefw;
print "      eta         ";
print eta;
print "                                                    ";

elseif parset ==3;
print "                                                    ";
print "****** Third set: Ramey and West row 6 ***********";
print "                                                     ";
print "        alpha             theta          gamma          Xi";
print alpha~theta~gam~Xi;
print " Coefficients in Decision Rule (and eta)    ";
print "      Coef on X      Coef on pi1       Coef on pi3        Coef
on W";
print coefx~coefpi1~coefpi3~coefw;
print "      eta           ";
print eta;
print "                                                    ";

endif;
screen off;
```

```
/************** REPETITIONS  do loop  ****************/
rep = 1;
do while rep <= repmax;
/* initialize variables: state, xx(sales), ww(P-inputs),
nn(inventories), yy (output), costs */
stvec = 1|0|0;                  /* begin in low state*/
pit = .974|.026|0;
pitctm1 = pit;           /* pit cond St=1, found numerically */
pis = pit;
pitcsm1 = pit;

xx = xbar;
ww = wbar1;
nnt=(coefx*xx+coefw*ww +coefpi1*pit[1,1]+coefpi3*pit[3,1]+bgam0)/(1-
zeta1-zeta2);
nntm1 = nnt;
nntm2 = nnt;
nnsub = nnt;
nnsm1 = nnt;
nnsm2 = nnt;

yy = xx+nnt-nntm1;
yytm1 = yy;
yysub =yy;
yysm1 = yy;

costop = xi*ww*yy + (theta/2)*(yy^2)  + (gam/2)*((yy-yytm1)^2) +
(delta/2)*((nntm1-alpha*xx)^2);

cop1 = (theta/2)*(yy^2);
cop2 = (gam/2)*((yy-yytm1)^2);
cop3 = (delta/2)*((nntm1-alpha*xx)^2);

costsub = xi*ww*yysub + (theta/2)*(yysub^2) + (gam/2)*((yysub-yysm1)^2)
+ (delta/2)*((nnsm1-alpha*xx)^2);

csub1 = (theta/2)*(yysub^2);
csub2 = (gam/2)*((yysub-yysm1)^2);
csub3 = (delta/2)*((nnsm1-alpha*xx)^2);

betat = 1;
betaprod = 1;
pdvcost = betaprod*(costop~costsub);
costsum = 0~0;

tt=1;
obs1v = nnt~costop~xx~yy;
obs2v = nnsub~costsub~yysub~(costop-costsub);
obs3v = pit'~tt;
obs4v = cop1~cop2~cop3;
obs5v = csub1~csub2~csub3;
obs6v = pdvcost;

/* tt do loop */
tt = 2;
do while tt <= ttmax;
```

```
/* draw shock to (constant mean) interest rate */
rrs = rr1m + (sigv[1,1])*rndn(1,1);
rshock = 0;

if (tt-init) == 1;
rshock = sigv[1,1];
endif;
rrt = rrs + rshock;

/* update probabilities (respond) */
f1den = inv(sigv[1,1]*sqrt(2*pi));
frc1 = f1den*exp(-((rrt - rrv[1,1])^2) /(2*sigv[1,1]^2) );
f2den = inv(sigv[2,1]*sqrt(2*pi));
frc2 = f2den*exp(-((rrt - rrv[2,1])^2) /(2*sigv[2,1]^2) );
f3den = inv(sigv[3,1]*sqrt(2*pi));
frc3 = f3den*exp(-((rrt - rrv[3,1])^2) /(2*sigv[3,1]^2) );
frv = frc1|frc2|frc3;
dent = pitctm1'frv;
p1t = pitctm1[1,1]*frc1/dent;
p2t = pitctm1[2,1]*frc2/dent;
p3t = pitctm1[3,1]*frc3/dent;
pit = p1t|p2t|p3t;
pitctm1 = pp*pit;

/* update probabilities (not respond) */
frc1s = f1den*exp(-((rrs - rrv[1,1])^2) /(2*sigv[1,1]^2) );
frc2s = f2den*exp(-((rrs - rrv[2,1])^2) /(2*sigv[2,1]^2) );
frc3s = f3den*exp(-((rrs - rrv[3,1])^2) /(2*sigv[3,1]^2) );
frvs = frc1s|frc2s|frc3s;
dent = pitcsm1'frvs;
p1s = pitcsm1[1,1]*frc1s/dent;
p2s = pitcsm1[2,1]*frc2s/dent;
p3s = pitcsm1[3,1]*frc3s/dent;
pis = p1s|p2s|p3s;
pitcsm1 = pp*pis;

ww = ww + sddelw*rndn(1,1);
saleshoc = sddelx*rndn(1,1);

/* DECISION RULE:  nt = f( nt-1, nt-2, xt-1, wt, pit, saleshoc)  */
nnt = zeta1*nntm1 + zeta2*nntm2 + coefx*xx + coefw*ww +
coefpi1*pit[1,1] + coefpi3*pit[3,1] +  bgam0 - saleshoc;

nnsub = zeta1*nnsm1 + zeta2*nnsm2 + coefx*xx + coefw*ww +
coefpi1*pis[1,1] + coefpi3*pis[3,1] + bgam0 - saleshoc;


/* generate xt and yt and calculate costs */
xx = xx + saleshoc;
yytm1 = yy;
yy = xx+nnt-nntm1;
yysm1 = yysub;
yysub = xx + nnsub-nnsm1;

costop = xi*ww*yy + (theta/2)*(yy^2) + (gam/2)*((yy-yytm1)^2) +
(delta/2)*((nntm1-alpha*xx)^2);
```

```
cop1 = (theta/2)*(yy^2);
cop2 = (gam/2)*((yy-yytm1)^2);
cop3 = (delta/2)*((nntm1-alpha*xx)^2);

costsub = xi*ww*yysub + (theta/2)*(yysub^2) + (gam/2)*((yysub-yysm1)^2)
+ (delta/2)*((nnsm1-alpha*xx)^2);

csub1 = (theta/2)*(yysub^2);
csub2 = (gam/2)*((yysub-yysm1)^2);
csub3 = (delta/2)*((nnsm1-alpha*xx)^2);

nntm2 = nntm1;
nntm1 = nnt;

nnsm2 = nnsm1;
nnsm1 = nnsub;

/* Present discounted value of costs */
betaprod = betat*betaprod;
pdvcost = betaprod*(costop~costsub);
costsum = pdvcost + costsum;
betat = inv(1+rrt);

/* record observations */
obs1v = obs1v|nnt~costop~xx~yy;
obs2v = obs2v|nnsub~costsub~yysub~(costop-costsub);
obs3v = obs3v|pit'~tt;
obs4v = obs4v|cop1~cop2~cop3;
obs5v = obs5v|csub1~csub2~csub3;
obs6v = obs6v|pdvcost;

if (tt-init) == 0;
cost0 = costsum;
betapr0 = betaprod;
endif;

if (tt-init) == 6;
cost6 = (costsum-cost0)/betapr0;
endif;

if (tt-init) == 12;
cost12 = (costsum-cost0)/betapr0;
endif;

if (tt-init) == 24;
cost24 = (costsum-cost0)/betapr0;
endif;

tt = tt + 1;
endo;
obsv = obs1v~obs2v~obs3v~obs4v~obs5v~obs6v;
costsumv = cost6|cost12|cost24|(costsum-cost0)/betapr0;
if rep == 1;
/*        1    2     3  4          5      6      7        8     */
/*obs1v=nnt~costop~xx~yy;obs2v=nnsub~costsub~yysub~(costop-costsub);*/
/*    9,10,11  12           13   14   15             16    17    18  */
```

```
/* obs3v=pit'~tt;  obs4v = cop1~cop2~cop3; obs5v = csub1~csub2~csub3;*/
/*             19      20          */
/* obs6v = pvcostop~pvcstsub;      */
obsav = obsv/repmax;
pvcstav = costsumv/repmax;
elseif rep > 1;
obsav = obsav + (obsv/repmax);
pvcstav = pvcstav + costsumv/repmax;
endif;
rep = rep + 1;
endo;

screen on;
print "    pdv respond       pdv not respond    Col1 - Col2
Col3/Col1 ";
pctdif6 = ((pvcstav[1,1]-pvcstav[1,2])/pvcstav[1,1]);
Print pvcstav[1,.]~(pvcstav[1,1] - pvcstav[1,2])~pctdif6;
Pctdif12 = ((pvcstav[2,1]-pvcstav[2,2])/pvcstav[2,1]);
Print pvcstav[2,.]~(pvcstav[2,1] - pvcstav[2,2])~pctdif12;
Pctdif24 = ((pvcstav[3,1]-pvcstav[3,2])/pvcstav[3,1]);
Print pvcstav[3,.]~(pvcstav[3,1] - pvcstav[3,2])~pctdif24;
Pctdif48 = ((pvcstav[4,1]-pvcstav[4,2])/pvcstav[4,1]);
Print pvcstav[4,.]~(pvcstav[4,1] - pvcstav[4,2])~pctdif48;
parset = parset+1;
endo;
print timestr(0);
end;
```