```
/* oosgce12.prg                          Norm Swanson - 02/09/04          */
/* Consistent Generic ICM Prediction Test  with new boot                  */
/*                                                                        */
/* main empirical program for new recur paper with Valentina              */
/*                                                                        */
/* as oosgc2-4.prg, but revised to construct multiple tests               */
/* including CS, CCS, DM, CM, and F                                       */

output file=c:\oosgce12.out reset; outwidth 255; output on;
format /MA1 /LD 12,9;

/*  */
/* input dataset */
/*  */

/* raw data are from 1954:7-2004:12 (606 obs) */
/* vars are CPI, FFR, tbill tbond Unem IP */

load datin[606,6]=c:\alldat1.txt;

examp=4;
Rtype=3;

/* create dataset to use for the program */
/* ultimately, dataset will have 600 observations */
/* so first R will be 400, say, if P=200, say */

if examp==1; /* CPI inflation and Unem example */
   yy1=ln(datin[2:606,1])-ln(datin[1:605,1]);
   xx1=datin[.,5];
elseif examp==2; /* CPI inflation and delta Unem */
   yy1=ln(datin[2:606,1])-ln(datin[1:605,1]);
   xx1=datin[2:606,5]-datin[1:605,5];
elseif examp==3; /* IP and spread */
   yy1=ln(datin[2:606,6])-ln(datin[1:605,6]);
   xx1=datin[.,4]-datin[.,2];
elseif examp==4; /* IP and R rate - short */
   yy1=ln(datin[2:606,6])-ln(datin[1:605,6]);
   xx1=datin[.,3];
else;
   print "oops";
endif;

/* total sample used henceforth is 600, from 1955:1-2004:12 */

yyy1=yy1[rows(yy1)-600+1:rows(yy1),.];
xxx1=xx1[rows(xx1)-600+1:rows(xx1),.];

/*  */
/* end input dataset */
/*  */

/*  */
/* numsamps */
/*  */

/* figure out number of samples to run */

if Rtype==1;      /* fix R=10 years ... first R=1955:1-1964:12, last
R=1985:1-1994:12, move up by year */
                  /* thus, there are many statistics, one for each new R, or 31 of
them */
                  /* here, let P carry through to the end of the sample in all cases
```

```
   */
   numsamps= 31;
elseif Rtype==2; /* let R= grow ... first R=1955:1-1964:12, last R=1955:1-1994:12,
move up by year */
                 /* thus, there are many statistics, one for each new R, or 31 of
them */
                 /* here, let P carry through to the end of the sample in all cases
  */
   numsamps= 31;
elseif Rtype==3; /* fix R= 10 years ... first R=1955:1-1964:12, last
R=1985:1-1994:12, move up by year */
                 /* thus, there are many statistics, one for each new R, or 31 of
them */
                 /* here, fix P to be 10 years in length */
   numsamps= 31;
else;
   print "oops";
endif;

/*  */
/* end numsamps */
/*  */

/* set other parameters */

ls1={2,3,4,6,12};
Bootnum=100;

gam=0.0; i=1;
do while i<=5;
  gam=gam|((i-1)/2)+0.5;
  i=i+1;
endo;

gamgam=zeros(1,2);
i=1;
do while i<=5;
  j=1;
  do while j<=5;
    gamgam=gamgam|((((j-1)/2)+0.5)~gam[i]);
    j=j+1;
  endo;
  i=i+1;
endo;

gam=gam[2:6,.];        /* 10x1 vector  */
gamgam=gamgam[2:26,.]; /* 100x2 matrix */

/* %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% */
/* %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% */
/*                                                                                */
/* PROCEDURES                                                                     */
/*                                                                                */
/* %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% */
/* %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% */


/* %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% */
/* %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% */

/* model prediction */
/* Pbig is the P for this loop */
/* yy is the data, xx is the extra variable */
```

```
/* gams is a 1x2 vector with a gamma pair, with the first element changing */
/* each time */

proc (9) = pred(yy,xx,Pbig,gamgam);
local ii,T,cnst,X,Y,N,Qu0,QmY,QmEXTRA,QsY,QsEXTRA,Qbet,adjA,adjAA,
Qf1,Qf1A,Qg,QgA,mp1,bstatad,bstatadA,fullerrA,fullerrB,
mp2,qf1B,d1,d1bar,d1sderr,dmstat,c1,c1bar,c1sderr,cmstat,Fstat,Qbetout,mses;

Qf1A={};  Qf1B={};  QgA={};  adjAA={}; bstatadA={}; Qbetout={};

T=rows(yy);

/* Construct 1-step ahead predictions */

ii=1;
do while ii<=Pbig;

/* prediction and main statistic parameter estimation */

  N=T-Pbig-1+ii; cnst=ones(T-Pbig-1+ii-1,1);
  X=cnst~yy[1:N-1]; Y=yy[2:N];
  Qbet=inv(X'*X)*X'*Y;
  Qbetout=Qbetout~Qbet; /* output actual beta estimators, size= k x Pbig */

  /*Qbet=0.0|0.0;*/
  /*print "Qbet" Qbet'~ii;*/

  /* adjustment terms for boot parameter estimation */
  cnst=ones(T-1,1); X=cnst~yy[1:T-1]; Y=yy[2:T];
  adjA= (meanc(X.*(Y-X*Qbet)))';
  adjAA=adjAA|adjA; /* each row is a beta estimator recenting mean of k elements
long */
                    /* and there are Pbig rows in total for Pbig predictions */

  /* I. prediction errors and pieces for generically comprehensive CS statistic */

  QmY=meanc(yy[1:N]); QmEXTRA=meanc(xx[1:N]);
  QsY=stdc(yy[1:N]);  QsEXTRA=stdc(xx[1:N]);

  Qu0=yy[N+1]-(1~yy[N])*Qbet; /* 1x1 */ /* prediction error */
  Qf1=Qu0;                    /* 1x1 */ /* f', quad loss assumed for stat construct
*/
  Qf1A=Qf1A|Qf1;              /* Pbigx1 vector of prediction errors */
  Qg=exp( ((atan((yy[N]-QmY)/(2*QsY)))~(atan((xx[N]-QmEXTRA)/(2*QsEXTRA))))*gamgam'
);
                              /* 1x100, as gamgam is 100x2 */
  QgA=QgA|Qg;

  /* full sample residuals */

  if ii==Pbig;
      fullerrA=Y-X*Qbet;
  endif;

  /* adjusted statistic for use in boot statistic */
  QmY=meanc(yy[1:T-1]); QmEXTRA=meanc(xx[1:T-1]);
  QsY=stdc(yy[1:T-1]);  QsEXTRA=stdc(xx[1:T-1]);
  Qg=exp(
((atan((yy[1:T-1]-QmY)/(2*QsY)))~(atan((xx[1:T-1]-QmEXTRA)/(2*QsEXTRA))))*gamgam' );
                                /* 1x100, as gamgam is 100x2 */

  bstatad=(meanc( (2*(Y-X*Qbet)).*Qg ))';
  bstatadA=bstatadA|bstatad;
```

```
  /* II. construct prediction errors from the bigger alternative model for use in */
  /*     the DM, ENC-T and F-test statistics                                      */

  N=T-Pbig-1+ii; cnst=ones(T-Pbig-1+ii-1,1);
  X=cnst~yy[1:N-1]~xx[1:N-1]; Y=yy[2:N];
  Qbet=inv(X'*X)*X'*Y;
  Qu0=yy[N+1]-(1~yy[N]~xx[N])*Qbet; /* 1x1 */ /* prediction error */
  Qf1=Qu0;
  Qf1B=Qf1B|Qf1; /* Pbigx1 vector of prediction errors */

  /* full sample residuals */

  if ii==Pbig;
     fullerrB=Y-X*Qbet;
  endif;

  ii=ii+1;
endo;

mses=(meanc(Qf1A^2~Qf1B^2))';

/* main statistic - generically comprehensive CS statistics */

mp1=((1/sqrt(Pbig))*sumc((Qf1A*2).*QgA)); /* a vector of rows same as number of
gammas        */
                                        /* = ( mp(gam1) mp(gam2) ... )' =numgammasx1
vector */

/* main statistic - non generically comprehensive CCS statistics */

mp2=(((1/sqrt(Pbig))*sumc((Qf1A).*xx[T-Pbig:T-1,1])))^2; /* 1x1 final CCS statistic
*/

/* DM statistic -- h=1 so no NW estimator used in variance estiamate */

d1= Qf1A^2-Qf1B^2; d1bar= meanc(d1); d1sderr= stdc(d1);
dmstat=sqrt(Pbig)*d1bar/d1sderr;

/* ENC-T statistic -- h=1 so no NW estimator used in variance estiamate */

c1=Qf1A.*(Qf1A-Qf1B); c1bar=meanc(c1); c1sderr=stdc(c1);
cmstat=sqrt(Pbig)*c1bar/c1sderr;

/* F-test -- Chi square version - - 1 dof as there is only 1 extra variable */

Fstat=rows(fullerrB)*((sumc(fullerrA^2)-sumc(fullerrB^2))/sumc(fullerrB^2));

clear X,Y,cnst;
retp(mp1,mp2,dmstat,cmstat,Fstat,adjAA,bstatadA,Qbetout,mses);
endp;

/* %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% */
/* %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% */

/* bootstrap statistics prediction loop used for construction of bootstrap CS tests
*/
/* use modified likelihood function in LS estimation to recenter boot statistic
automatically */

proc (5) = predb(yy,xx,Pbig,gamgam,recterm,betnoPEE);
local
ii,T,cnst,X,Y,N,Qu0,QmY,QmEXTRA,QsY,QsEXTRA,Qbet,Qf1,Qf1A,Qg,QgA,Qf1B,Qf1C,Bmp2;
```

```
Qf1A={}; Qf1B={}; Qf1C={};  QgA={};

T=rows(yy);

/* Construct 1-step ahead predictions */

ii=1;
do while ii<=Pbig;

/* quadratic */

  N=T-Pbig-1+ii; cnst=ones(T-Pbig-1+ii-1,1);
  X=cnst~yy[1:N-1]; Y=yy[2:N];
  Qbet=inv(X'*X)*((X'*Y)-((N-1)*recterm[ii,.]'));
  /*Qbet=0.0|0.0;*/

  /* I. prediction errors and pieces for generically comprehensive CS statistic */

  QmY=meanc(yy[1:N]); QmEXTRA=meanc(xx[1:N]);
  QsY=stdc(yy[1:N]);  QsEXTRA=stdc(xx[1:N]);

  Qu0=yy[N+1]-(1~yy[N])*Qbet; /* 1x1 */ /* prediction error */
  Qf1=Qu0;                    /* 1x1 */ /* f', quad loss assumed for stat construct
*/
  Qf1A=Qf1A|Qf1;              /* Pbigx1 */

  Qg=exp( ((atan((yy[N]-QmY)/(2*QsY)))~(atan((xx[N]-QmEXTRA)/(2*QsEXTRA))))*gamgam'
);
                                      /* 1x100, as gamgam is 100x2 */
  QgA=QgA|Qg;

  /* II. additional pieces: case where assume no PEE and use theta hat from original
stat */

  Qu0=yy[N+1]-(1~yy[N])*betnoPEE[.,ii]; /* 1x1 */ /* prediction error */
  Qf1=Qu0;                    /* 1x1 */ /* f', quad loss assumed for stat construct
*/
  Qf1B=Qf1B|Qf1;              /* Pbigx1 */

  /* III. additional pieces: case where use naiive no adjust boot statistic */

  Qbet=inv(X'*X)*X'*Y;
  Qu0=yy[N+1]-(1~yy[N])*Qbet; /* 1x1 */ /* prediction error */
  Qf1=Qu0;                    /* 1x1 */ /* f', quad loss assumed for stat construct
*/
  Qf1C=Qf1C|Qf1;              /* Pbigx1 */

  ii=ii+1;
endo;

/* main statistic - non generically comprehensive CCS statistics */

Bmp2=(((1/sqrt(Pbig))*sumc((Qf1A).*xx[T-Pbig:T-1,1])))^2; /* 1x1 final CCS statistic
*/

clear X,Y,cnst;
retp(Qf1A,Qf1B,Qf1c,QgA,Bmp2);
endp;

/* %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% */
/* %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% */
```

```
/* block bootstrap data sets from entire sample */

proc (1) = bootblok(dat1,lval);
local N,num_uns,undraw1,x1,ib;

N=rows(dat1);
num_uns=N/lval;

/* draw uniforms U[0,T-l+1] */

undraw1=round((N-lval)*rndu(num_uns,1));

x1={};
ib=1;
do while ib<=num_uns;
  x1=x1|dat1[undraw1[ib]+1:undraw1[ib]+lval,.];
  ib=ib+1;
endo;

retp(x1);
endp;

/* %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% */
/* %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% */

/* Killian VAR parametric method bootstrap */

proc (2) = bootkill(yy,xx);
local ii,T,cnst,X,Y,N,Qbet1,Qbet2,res1,res2,num_uns,undraw1,undraw2,yya,xxa;

T=rows(yy);

/* Generate boostrap data for use in bootstrap statistics construction */

/* 1. fit VAR model to the two series, null imposed */

  N=T-1; cnst=ones(T-2,1); /* assume 1 lag in first equation */
  X=cnst~yy[1:N-1]; Y=yy[2:N];
  Qbet1=inv(X'*X)*X'*Y;
  res1=Y-X*Qbet1;
  X=cnst~yy[1:N-1]~xx[1:N-1]; Y=xx[2:N];
  Qbet2=inv(X'*X)*X'*Y;
  res2=Y-X*Qbet2;

  num_uns=rows(res1);

/* 2. draw uniformly integer values in range (1,...,numres), and draw numres of
these */

undraw1=(round((num_uns-1)*rndu(T,1)))+1;

/* 3. do again, but one value for starting value of x,y in data construction */

undraw2=(round((rows(yy)-1)*rndu(1,1)))+1;

/* build up bootstrap datasets */

yya=zeros(T,1);
xxa=zeros(T,1);

yya[1]=(1~yy[undraw2])*Qbet1+res1[undraw1[1]];
xxa[1]=(1~yy[undraw2]~xx[undraw2])*Qbet2+res2[undraw1[1]];
```

```
ii=2;
do while ii<=rows(yy);
    yya[ii]=(1~yy[ii-1])*Qbet1+res1[undraw1[ii]];
    xxa[ii]=(1~yy[ii-1]~xx[ii-1])*Qbet2+res2[undraw1[ii]];
    ii=ii+1;
endo;

clear X,Y,cnst;
retp(yya,xxa);
endp;

/* %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% */
/* %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% */

/* CS statistics */
/* mp(gamma) in, final mp stats out, aggregated over the gamma */

proc (1) = mpstat(Qmpz);
local sout1,Qmpp1,Qmpp2,Qmpp3;

Qmpp1=meanc(Qmpz^2);
Qmpp2=maxc(abs(Qmpz));
Qmpp3=meanc(abs(Qmpz));
sout1=Qmpp1|Qmpp2|Qmpp3;

retp(sout1);
endp;


/* %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% */
/* %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% */
/* MAIN PROGRAM -                                                       */
/* %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% */
/* %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% */

/* MAIN MONTE CARLO EXPERIMENTS */

  tabrows=numsamps;

  Qtab10A=zeros(tabrows,3*rows(ls1));

  Qtab10D=zeros(tabrows,rows(ls1));

  Qtab10E=zeros(tabrows,3);

  Qtab10F=zeros(tabrows,3);

  Qtab10G=zeros(tabrows,3);

  Qtab10H=zeros(tabrows,4);

  msesmbg=zeros(tabrows,2);

  stats=zeros(tabrows,7);

  cvcs10=zeros(tabrows,3*rows(ls1));

  cvccs10=zeros(tabrows,rows(ls1));

  cvcsk10=zeros(tabrows,3);

  cvrest10=zeros(tabrows,4);
```

```
    dg=1;
    dgnum=numsamps;
    do while dg<=dgnum;

/*
if Rtype==1;      /* fix R=10 years ... first R=1955:1-1964:12, last
R=1985:1-1994:12, move up by year */
                  /* thus, there are many statistics, one for each new R, or 31 of
them */
                  /* here, let P carry through to the end of the sample in all cases
   */
   numsamps= 31;
elseif Rtype==2; /* let R= grow ... first R=1955:1-1964:12, last R=1955:1-1994:12,
move up by year */
                 /* thus, there are many statistics, one for each new R, or 31 of
them */
                 /* here, let P carry through to the end of the sample in all cases
  */
   numsamps= 31;
elseif Rtype==3; /* fix R= 10 years ... first R=1955:1-1964:12, last
R=1985:1-1994:12, move up by year */
                 /* thus, there are many statistics, one for each new R, or 31 of
them */
                 /* here, fix P to be 10 years in length */
   numsamps= 31;
*/

      if Rtype==1;
         yy1=yyy1[((dg-1)*12)+1:rows(yyy1),.];
         xx1=xxx1[((dg-1)*12)+1:rows(xxx1),.];
         Pbig=600-((dg-1)*12)-120;
      elseif Rtype==2;
         yy1=yyy1;
         xx1=xxx1;
         Pbig=600-((dg-1)*12)-120;
      elseif Rtype==3;
         yy1=yyy1[((dg-1)*12)+1:((dg-1)*12)+120+120,.];
         xx1=xxx1[((dg-1)*12)+1:((dg-1)*12)+120+120,.];
         Pbig=120;
      else;
         print "oops";
      endif;

      {mp,ccs,dm,cm,ftest,oadjA,bstatadj,Qbet1,mseout}=pred(yy1,xx1,Pbig,gamgam);

      msesmbg[dg,.]=mseout;

      /* mp is actual statistics for All Gammas */
      /* these are 100x1 of the mp(gamma), with 100 gammas */

      /* all other tests are 1x1 as they are not constructed for each gamma! */

      /* mp is the cs test; ccs is the Chao, Corradi and Swanson (2001) */
      /* simple version of the CS test; */
      /* dm is the Deibold-Mariano test, called the MSE-T or OOS-T by Clark */
      /* and McCracken (2001 - called MSE-T) and discussed in McCracken (2004 -
called OOS-T); */
      /* cm is the ENC-T from Clark and McCracken (2001); */
      /* ftest is the chisquared version of the F-test */

      /* oadjA is adj term for boot estimator construction in bpred */

      /* bstatadj is as mp, but uses all T and replaces mp in */
```

```
      /* construction of final bootstrap statistics Bmp */

      /* Calculate Actual Final sup, average, etc. mp statistics */
      /* 3x1 vectors with 3 rows, one for each stat */

      {mpstats}=mpstat(mp); /* this simply aggregates over the gamma */

      stats[dg,.]=(mpstats'~ccs~dm~cm~ftest);

      /* NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN */
      /* NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN */
      /* NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN */

      /* ================================== */
      /* Now, bootstrap the critical values!!! */
      /* ================================== */

      /* NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN */
      /* NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN */
      /* NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN */

      /* part A -- bootstrap CS and CCS and use tabulated critical values for the
rest */
      /* these values used in Part A Organization of Results */


      BcritA={}; BcritB={}; BcritC={}; BcritD={};

      booti=1;
      do while booti <= bootnum;

        allstatA={}; allstatB={}; allstatC={}; allstatD={};
        lli=1;
        do while lli<=rows(ls1);

          l_val=ls1[lli];

          {sdataa} = bootblok(yy1~xx1,l_val);


{BoQf1A,BoQf1B,BoQf1C,BoQgA,Bmp2a}=predb(sdataa[.,1],sdataa[.,2],Pbig,gamgam,oadjA,Q
bet1);

          /* Calculate Boot Statistics for All Gammas */
          /* these are 100x1 of the mp(gamma), with 100 gammas */

          /* with regard to BmpA-BmpD below, note the following: */
          /* the first statistic is our regular statistic, with both adjustment
terms (i.e. */
          /* the recentered parameter estimator + the bias adjustment term in the
actual final */
          /* boot statistic. */
          /* the second has no adjust at all and sets theta tilde star = theta hat
*/
          /* the third also has no adjust at all, but at least calculates a theta
star */
          /* the fourth statistic is the simple CCS test bootstrap statistic */


          BmpA=(1/sqrt(Pbig))*sumc(((BoQf1A*2).*BoQgA)-bstatadj);
          BmpB=((1/sqrt(Pbig))*sumc((BoQf1B*2).*BoQgA))-mp;
          BmpC=((1/sqrt(Pbig))*sumc((BoQf1C*2).*BoQgA))-mp;
```

```
          BmpD=Bmp2a-ccs;

          /* Calculate Actual Final sup, average, etc. mp statistics */
          /* 3x1 vectors with 3 rows, one for each stat */

          {BmpstatA}=mpstat(BmpA);
          {BmpstatB}=mpstat(BmpB);
          {BmpstatC}=mpstat(BmpC);

          /* build up a row with all 3 stats times # of lvals in the columns of the
row */
          /* do this for each alternative CV approach for CS test */
          allstatA=allstatA~BmpstatA';
          allstatB=allstatB~BmpstatB';
          allstatC=allstatC~BmpstatC';

          allstatD=allstatD~BmpD; /* a row with the single CCS boot stat times # of
lvals */

          lli=lli+1;
        endo;

        BcritA=BcritA|allstatA;
        BcritB=BcritB|allstatB;
        BcritC=BcritC|allstatC;

        BcritD=BcritD|allstatD;

        booti=booti+1;
      endo;

      /* NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN */
      /* NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN */
      /* NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN */

      /* part B -- Killian bootstrap used for CVs for all tests */
      /* these values used in Part B Organization of Results */

      CSboot={}; CCSboot={}; DMboot={}; CMboot={}; Fboot={};

      booti=1;
      do while booti <= 100;

        {yyb,xxb} = bootkill(yy1,xx1);

        {mpb,ccsb,dmb,cmb,ftestb,junkA,junkb,junkc,junkd}=pred(yyb,xxb,Pbig,gamgam);

        {mpstatsb}=mpstat(mpb); /* this simply aggregates over the gamma */

        CSboot=CSboot|mpstatsb';
        CCSboot=CCSboot|ccsb;
        DMboot=DMboot|dmb;
        CMboot=CMboot|cmb;
        Fboot=Fboot|ftestb;


        booti=booti+1;
      endo;

      /* NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN */
      /* NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN */
      /* NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN */
```

```
      /* ==================================== */
      /* END!! bootstrap the critical values!!! */
      /* ==================================== */

      /* NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN */
      /* NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN */
      /* NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN */

      /* Orgainzation of Results */

      /* NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN */
      /* NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN */
      /* NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN */

      /* Part A Organization of Results -- Block Bootstrap and Standard Distribution
Results */


      /* for the 3 CS tests, construct cv's for each of the 3 version of the
bootstrap statistic and tabulate results*/

      /* I. results based on BmpA */

      Bcrit=BcritA;

      cv10={}; cv5={};
      cvi=1;
      do while cvi<=cols(Bcrit);
        temp=sortc(Bcrit[.,cvi],1);
        cv5=cv5|temp[trunc(0.95*rows(Bcrit))];
        cv10=cv10|temp[trunc(0.90*rows(Bcrit))];
        cvi=cvi+1;
      endo;

      cvcs10[dg,.]=cv10';

      /* put together tabulated results */
      /* the table has each row as a DGP, and then the columns are 3 stats for the
first lval, */
      /* and then 3 stats for the second lval, etc. */

      tabi=1;
      do while tabi<=rows(ls1);
        Qmatlval=mpstats .gt cv10[(tabi-1)*3+1:(tabi-1)*3+3,.];
        if dg==1;
           Qtab10A[dg,(tabi-1)*3+1:(tabi-1)*3+3]=Qmatlval';
        else;

Qtab10A[dg,(tabi-1)*3+1:(tabi-1)*3+3]=Qtab10A[dg-1,(tabi-1)*3+1:(tabi-1)*3+3]+Qmatlv
al';
        endif;
        tabi=tabi+1;
      endo;

      /* IV. results based on CCS test */

      Bcrit=BcritD;

      cv10={}; cv5={};
      cvi=1;
      do while cvi<=cols(Bcrit);
        temp=sortc(Bcrit[.,cvi],1);
        cv5=cv5|temp[trunc(0.95*rows(Bcrit))];
```

```
        cv10=cv10|temp[trunc(0.90*rows(Bcrit))];
        cvi=cvi+1;
      endo;

      cvccs10[dg,.]=cv10';

      /* put together tabulated results */
      /* the table has each row as a DGP, and then the columns are 1 stats for the
first lval, */
      /* and then 1 stats for the second lval, etc. */

      tabi=1;
      do while tabi<=rows(ls1);
        Qmatlval=ccs .gt cv10[(tabi-1)*1+1:(tabi-1)*1+1,.];
        if dg==1;
           Qtab10D[dg,(tabi-1)*1+1:(tabi-1)*1+1]=Qmatlval';
        else;

Qtab10D[dg,(tabi-1)*1+1:(tabi-1)*1+1]=Qtab10D[dg-1,(tabi-1)*1+1:(tabi-1)*1+1]+Qmatlv
al';
        endif;
        tabi=tabi+1;
      endo;

      /* V. results based on Ftest, DM, CM tests, assume pi=0 */

      thestats=dm|cm|ftest;

      cv5=1.645|1.645|3.84;
      cv10=1.280|1.280|2.71;

      /* put together tabulated results */
      /* the table has each row as a DGP, and then the columns are the 3 stats */
        Qmatlval=thestats .gt cv10;
        if dg==1;
          Qtab10E[dg,1:3]=Qmatlval';
        else;
          Qtab10E[dg,1:3]=Qtab10E[dg-1,1:3]+Qmatlval';
        endif;

      /* VI. results based on Ftest, DM, CM tests, assume pi not equal 0 */

      thestats=dm|cm|ftest;

      cv5=0.771|1.331|3.84;   /* values for P/R = 1, i.e. P=0.5T */
      cv10=0.443|1.005|2.71;

      /* put together tabulated results */
      /* the table has each row as a DGP, and then the columns are the 3 stats */
        Qmatlval=thestats .gt cv10;
        if dg==1;
          Qtab10F[dg,1:3]=Qmatlval';
        else;
          Qtab10F[dg,1:3]=Qtab10F[dg-1,1:3]+Qmatlval';
        endif;


      /* NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN */
      /* NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN */
      /* NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN */

      /* Part B Organization of Results -- Killian Bootstrap Results*/
```

```
      /* for the 3 CS tests, construct cv's for the Killian version of the bootstrap
statistic and tabulate results*/

      /* I. results based on CSboot */

      Bcrit=CSboot;

      cv10={}; cv5={};
      cvi=1;
      do while cvi<=cols(Bcrit);
        temp=sortc(Bcrit[.,cvi],1);
        cv5=cv5|temp[trunc(0.95*rows(Bcrit))];
        cv10=cv10|temp[trunc(0.90*rows(Bcrit))];
        cvi=cvi+1;
      endo;

      cvcsk10[dg,.]=cv10';

      /* put together tabulated results */
      /* the table has each row as a DGP, and then the columns are 3 stats */

        Qmatlval=mpstats .gt cv10;
        if dg==1;
           Qtab10G[dg,1:3]=Qmatlval';
        else;
           Qtab10G[dg,1:3]=Qtab10G[dg-1,1:3]+Qmatlval';
        endif;

      /* for the rest of the tests construct cv's for the Killian version of the
bootstrap statistic and tabulate results*/

      /* II. results based on CSboot */

      Bcrit=CCSboot~DMboot~CMboot~Fboot;

      thestats= ccs|dm|cm|ftest;

      cv10={}; cv5={};
      cvi=1;
      do while cvi<=cols(Bcrit);
        temp=sortc(Bcrit[.,cvi],1);
        cv5=cv5|temp[trunc(0.95*rows(Bcrit))];
        cv10=cv10|temp[trunc(0.90*rows(Bcrit))];
        cvi=cvi+1;
      endo;

      cvrest10[dg,.]=cv10';


      /* put together tabulated results */
      /* the table has each row as a DGP, and then the columns are 3 stats */

        Qmatlval=thestats .gt cv10;
        if dg==1;
           Qtab10H[dg,1:4]=Qmatlval';
        else;
           Qtab10H[dg,1:4]=Qtab10H[dg-1,1:4]+Qmatlval';
        endif;

      print "Sample Number " dg " is done!!";

      dg=dg+1;
```

```
    endo;

  print "/* NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN */";
  print "/* NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN */";
  print "/* NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN */";

  print "*********************************************************************";
  print "FINAL OUTPUT ********************************************************";
  print "*********************************************************************";



  print "=======================================================";
  print "=======================================================";
  print "====Block Boot and standard distribution Results=======";
  print "=======================================================";
  print "=======================================================";
  print " ";
  print " ";

  print " ";
  print "=======================================================";
  print "CV and # rejections OUTPUT for  CS test full modified boot used";
  print "Output for Bierens Type Test, Quadratic Loss";
  print "Rtype is " Rtype;
  print "Example is " examp;
  print " ";
  print "rows are different samples";
  print "if examp==1; CPI inflation and Unem example";
  print "elseif examp==2; CPI inflation and delta Unem";
  print "elseif examp==3; IP and spread";
  print "elseif examp==4; IP and R rate - short";
  print " ";
  print "Rtype is as:";
  print "if Rtype==1; fix R=10 years ... first R=1955:1-1964:12, last 
R=1985:1-1994:12, move up by year";
  print "thus, there are many statistics, one for each new R, or 31 of them";
  print "here, let P carry through to the end of the sample in all cases";
  print " numsamps= 31";
  print "elseif Rtype==2; let R= grow ... first R=1955:1-1964:12, last 
R=1955:1-1994:12, move up by year";
  print "thus, there are many statistics, one for each new R, or 31 of them";
  print "here, let P carry through to the end of the sample in all cases";
  print " numsamps= 31";
  print "elseif Rtype==3; fix R= 10 years ... first R=1955:1-1964:12, last 
R=1985:1-1994:12, move up by year";
  print "thus, there are many statistics, one for each new R, or 31 of them";
  print "here, fix P to be 10 years in length";
  print "numsamps= 31";

  print " ";
  print "CV used, various block lengths = 2,3,4,6,12";
  print "for each blocklength there are 3 cvs across 3 cols, 1 for each type of 
mpstat, mean of squ, sup, mean of abs";
  print " ";
  print cvcs10;
  print " ";
  print "graphing output -- number of rejections, cummulative down the columns";
  print " ";
  print Qtab10A;
  print " ";
  print "=======================================================";
  print " ";
```

```
  print "=========================================================";
  print "CV and # rejections OUTPUT for Simple CCS test full modified boot used as
in CS test ";
  print "Output for Bierens Type Test, Quadratic Loss";
  print "Rtype is " Rtype;
  print "Example is " examp;
  print " ";
  print "rows are different samples";
  print "if examp==1; CPI inflation and Unem example";
  print "elseif examp==2; CPI inflation and delta Unem";
  print "elseif examp==3; IP and spread";
  print "elseif examp==4; IP and R rate - short";
  print " ";
  print "Rtype is as:";
  print "if Rtype==1; fix R=10 years ... first R=1955:1-1964:12, last
R=1985:1-1994:12, move up by year";
  print "thus, there are many statistics, one for each new R, or 31 of them";
  print "here, let P carry through to the end of the sample in all cases";
  print " numsamps= 31";
  print "elseif Rtype==2; let R= grow ... first R=1955:1-1964:12, last
R=1955:1-1994:12, move up by year";
  print "thus, there are many statistics, one for each new R, or 31 of them";
  print "here, let P carry through to the end of the sample in all cases";
  print " numsamps= 31";
  print "elseif Rtype==3; fix R= 10 years ... first R=1955:1-1964:12, last
R=1985:1-1994:12, move up by year";
  print "thus, there are many statistics, one for each new R, or 31 of them";
  print "here, fix P to be 10 years in length";
  print "numsamps= 31";

  print " ";
  print "CV used, various block lengths = 2,3,4,6,12";
  print " ";
  print cvccs10;
  print " ";
  print "graphing output -- number of rejections, cummulative down the columns";
  print " ";
  print Qtab10D;
  print " ";
  print "=========================================================";
  print " ";


  print "=========================================================";
  print "# rejections OUTPUT for DM, CM, F tests -- PI equal 0 ";
  print "Rtype is " Rtype;
  print "Example is " examp;
  print " ";
  print "rows are different samples";
  print "if examp==1; CPI inflation and Unem example";
  print "elseif examp==2; CPI inflation and delta Unem";
  print "elseif examp==3; IP and spread";
  print "elseif examp==4; IP and R rate - short";
  print " ";
  print "Rtype is as:";
  print "if Rtype==1; fix R=10 years ... first R=1955:1-1964:12, last
R=1985:1-1994:12, move up by year";
  print "thus, there are many statistics, one for each new R, or 31 of them";
  print "here, let P carry through to the end of the sample in all cases";
```

```
  print " numsamps= 31";
  print "elseif Rtype==2; let R= grow ... first R=1955:1-1964:12, last
R=1955:1-1994:12, move up by year";
  print "thus, there are many statistics, one for each new R, or 31 of them";
  print "here, let P carry through to the end of the sample in all cases";
  print " numsamps= 31";
  print "elseif Rtype==3; fix R= 10 years ... first R=1955:1-1964:12, last
R=1985:1-1994:12, move up by year";
  print "thus, there are many statistics, one for each new R, or 31 of them";
  print "here, fix P to be 10 years in length";
  print "numsamps= 31";

  print " ";
  print "CV used are standard tabulated";
  print " ";
  print "graphing output -- number of rejections, cummulative down the columns";
  print " ";
  print Qtab10E;
  print " ";
  print "=======================================================";
  print " ";

  print "=======================================================";
  print "# rejections OUTPUT for DM, CM, F tests -- PI not equal 0, assume pi=1 ";
  print "Rtype is " Rtype;
  print "Example is " examp;
  print " ";
  print "rows are different samples";
  print "if examp==1; CPI inflation and Unem example";
  print "elseif examp==2; CPI inflation and delta Unem";
  print "elseif examp==3; IP and spread";
  print "elseif examp==4; IP and R rate - short";
  print " ";
  print "Rtype is as:";
  print "if Rtype==1; fix R=10 years ... first R=1955:1-1964:12, last
R=1985:1-1994:12, move up by year";
  print "thus, there are many statistics, one for each new R, or 31 of them";
  print "here, let P carry through to the end of the sample in all cases";
  print " numsamps= 31";
  print "elseif Rtype==2; let R= grow ... first R=1955:1-1964:12, last
R=1955:1-1994:12, move up by year";
  print "thus, there are many statistics, one for each new R, or 31 of them";
  print "here, let P carry through to the end of the sample in all cases";
  print " numsamps= 31";
  print "elseif Rtype==3; fix R= 10 years ... first R=1955:1-1964:12, last
R=1985:1-1994:12, move up by year";
  print "thus, there are many statistics, one for each new R, or 31 of them";
  print "here, fix P to be 10 years in length";
  print "numsamps= 31";

  print " ";
  print "CV used are standard tabulated";
  print " ";
  print "graphing output -- number of rejections, cummulative down the columns";
  print " ";
  print Qtab10F;
  print " ";
  print "=======================================================";
  print " ";

  print "=======================================================";
  print "=======================================================";
  print "====Killiank Boot Results=======";
```

```
  print "=======================================================";
  print "=======================================================";


  print " ";
  print "=======================================================";
  print "CV and # rejections OUTPUT for  CS test killian boot";
  print "Output for Bierens Type Test, Quadratic Loss";
  print "Rtype is " Rtype;
  print "Example is " examp;
  print " ";
  print "rows are different samples";
  print "if examp==1; CPI inflation and Unem example";
  print "elseif examp==2; CPI inflation and delta Unem";
  print "elseif examp==3; IP and spread";
  print "elseif examp==4; IP and R rate - short";
  print " ";
  print "Rtype is as:";
  print "if Rtype==1; fix R=10 years ... first R=1955:1-1964:12, last
R=1985:1-1994:12, move up by year";
  print "thus, there are many statistics, one for each new R, or 31 of them";
  print "here, let P carry through to the end of the sample in all cases";
  print " numsamps= 31";
  print "elseif Rtype==2; let R= grow ... first R=1955:1-1964:12, last
R=1955:1-1994:12, move up by year";
  print "thus, there are many statistics, one for each new R, or 31 of them";
  print "here, let P carry through to the end of the sample in all cases";
  print " numsamps= 31";
  print "elseif Rtype==3; fix R= 10 years ... first R=1955:1-1964:12, last
R=1985:1-1994:12, move up by year";
  print "thus, there are many statistics, one for each new R, or 31 of them";
  print "here, fix P to be 10 years in length";
  print "numsamps= 31";

  print " ";
  print "there are 3 cvs across 3 cols, 1 for each type of mpstat, mean of squ, sup,
mean of abs";
  print " ";
  print cvcsk10;
  print " ";
  print "graphing output -- number of rejections, cummulative down the columns";
  print " ";
  print Qtab10G;
  print " ";
  print "=======================================================";
  print " ";


  print " ";
  print "=======================================================";
  print "CV and # rejections OUTPUT for CCS, DM, CM, and F test, killian boot";
  print "Output for Bierens Type Test, Quadratic Loss";
  print "Rtype is " Rtype;
  print "Example is " examp;
  print " ";
  print "rows are different samples";
  print "if examp==1; CPI inflation and Unem example";
  print "elseif examp==2; CPI inflation and delta Unem";
  print "elseif examp==3; IP and spread";
  print "elseif examp==4; IP and R rate - short";
  print " ";
  print "Rtype is as:";
  print "if Rtype==1; fix R=10 years ... first R=1955:1-1964:12, last
```

```
R=1985:1-1994:12, move up by year";
  print "thus, there are many statistics, one for each new R, or 31 of them";
  print "here, let P carry through to the end of the sample in all cases";
  print " numsamps= 31";
  print "elseif Rtype==2; let R= grow ... first R=1955:1-1964:12, last
R=1955:1-1994:12, move up by year";
  print "thus, there are many statistics, one for each new R, or 31 of them";
  print "here, let P carry through to the end of the sample in all cases";
  print " numsamps= 31";
  print "elseif Rtype==3; fix R= 10 years ... first R=1955:1-1964:12, last
R=1985:1-1994:12, move up by year";
  print "thus, there are many statistics, one for each new R, or 31 of them";
  print "here, fix P to be 10 years in length";
  print "numsamps= 31";

  print " ";
  print " ";
  print cvrest10;
  print " ";
  print "graphing output -- number of rejections, cummulative down the columns";
  print " ";
  print Qtab10H;
  print " ";
  print "============================================================";
  print "  ";
  print "  ";
  print "FINALLY ... print actual statistics and mses of smaller and then biggers
model";
  print "  ";
  print " ";
  print "statistics -- 3 CS stats, CCS, DM, CM, and F test";
  print stats;
  print "  ";
  print " ";
  print "mse, smaller and larger models";
  print msesmbg;

output off;
```