

```

                                rokr1c. PRG
/* Rokko -
/* rokr1c. prg - jul 2003
/* revisi on example - triple comparison
/* as rok2. prg
/* related to mul ti 2a. prg from Andrews paper
/* thi s versi on consi ders bi vari ate joi nt di stri buti ons
/* empi ri cal example
/*

format /MA1/LD 7, 4;
outwidth 255;
output file = c:\papers\v_rokko\revisi on\rokr1c. out reset;
output on;

/* Parameters to Set Before Carryng out analysi s
clearg bet, del t, n, gamm, thet, a2, A11, A22, ve1, ve2, cove1e2, pc_K1, pc_H1, pc_Y1, l d_pc_Y;

bet= 0. 988 ; /* beta */
del t= 0. 025 ; /* del ta */
n= 1. 012 ; /* n */
gamm= 0. 0028 ; /* gamma */
thet= 0. 50 ; /* theta */
a2= 0. 0044 ; /* a_2 */
A11= 0. 95 ; /* A_11 */
A22= -0. 10 ; /* A_22 */
ve1= 0. 039 ; /* var(e1) */
ve2= 0. 077 ; /* var(e2) */
cove1e2= -0. 030; /* cov(e1, e2) */

l s={2, 5, 7};
unum=20; /* densi ty of U grid */
bootnum=100; /* number of boot repli cations for CV calc */
simtot=1;

/* read in data for empi ri cal and simul ati on exerci ses */
/* data assumed to be in log di fferen ces immedi ately at thi s stage */
/* for empi ri cal need Y and K */

/* data are from 1929-2002 (74 obs) */
/* vars are K, GDP, and N */
/* last K obs for 2002 is a dummy val ue, but as l agged K used, poses no */
/* problem */

load dati n[74, 4]=c:\papers\v_rokko\revisi on\annUSrev. dat;

Yi n=dati n[. , 2]*1000000000;
Yi n=dati n[. , 2];
Ki n=dati n[. , 1];
Ni n=dati n[. , 3]*1000;
Ni n=dati n[. , 3];
Hi n=dati n[. , 4];

pc_Y=Yi n./Ni n;
pc_K=Ki n./Ni n;
pc_H=Hi n;

pc_Y=pc_Y[5: rows(pc_Y), . ];
pc_K=pc_K[5: rows(pc_K), . ];
pc_H=pc_H[5: rows(pc_H), . ];

print "Annual Data wi th obs = " rows(Yi n);
pri nt Yi n~Ki n~Ni n~Hi n;

```

rokr1c. PRG

```

TT=74; /* define length of actual data sample */
/* log difference levels series */
ld_Y=ln(Yi n[2: rows(Yi n), . ]) - ln(Yi n[1: rows(Yi n)-1, . ]);
ld_K=ln(Ki n[2: rows(Ki n), . ]) - ln(Ki n[1: rows(Ki n)-1, . ]);
ld_N=ln(Ni n[2: rows(Ni n), . ]) - ln(Ni n[1: rows(Ni n)-1, . ]);
ld_H=ln(Hi n[2: rows(Hi n), . ]) - ln(Hi n[1: rows(Hi n)-1, . ]);
/* log difference per capita GDP */
ld_pc_Ya=ld_Y-ld_N;
/* log difference per capita hours */
ld_pc_Ha=ld_H;
/* make this sample 70 long */
ld_Y=ld_Y[4: rows(ld_Y), . ];
ld_K=ld_K[4: rows(ld_K), . ];
ld_N=ld_N[4: rows(ld_N), . ];
ld_H=ld_H[4: rows(ld_H), . ];
ld_pc_Y=ld_pc_Ya[4: rows(ld_pc_Ya), . ]; /* pare down to 70 obs */
ld_pc_Yl=ld_pc_Ya[3: rows(ld_pc_Ya)-1, . ];
ld_pc_H=ld_pc_Ha[4: rows(ld_pc_Ha), . ]; /* pare down to 70 obs */
ld_pc_Hl=ld_pc_Ha[3: rows(ld_pc_Ha)-1, . ];
TT=rows(ld_pc_Y); /* define length of actual data sample */
/*
/* in addition to using ld_pc_Y, may use quarterly y data for the actual GDP */
/* in the final comparison */
load dati n2[224, 2]=c:\papers\v_rokko\qrtUS.dat; /* 1947:1 -2002:4 */
ld_Y1=ln(dati n2[2: rows(dati n2), 1]) - ln(dati n2[1: rows(dati n2)-1, 1]);
ld_N1=ln(dati n2[2: rows(dati n2), 2]) - ln(dati n2[1: rows(dati n2)-1, 2]);
ld_pc_Y1=ld_Y1-ld_N1;
ld_pc_Q=ld_pc_Y1[4: rows(ld_pc_Y1), . ]; /* pare down to 220 obs */
ld_pc_Ql=ld_pc_Y1[3: rows(ld_pc_Y1)-1, . ];
*/
PROC i student(p, v);
  RETP(CDFTCI (1-p, v));
ENDP;
PROC r student(r, c, v);
  RETP(i student(RNDU(r, c), v));
ENDP;

```

rokr1c. PRG

```

/* ----- */
/* PROCEDURE - simulate data */
/* ----- */

/* data generation */

proc (2) = sim2(smpl);
local ux, z, ee, dgi, cova,
      w, phi_, h_, ktil_, ytil_, ctil_, phi0_, phi1_, sstar_, dstar_, B_,
      r_1, r_7, r_9, r_8, r_5, r_6, r_2, r_11, r_18, r_81, r_19, r_91, r_17, r_71,
      r_98, r_89, r_88, r_68, r_86, r_78, r_87, r_69, r_96, r_99, r_97, r_79, r_66, r_67, r_76, r_77,
      r_22, r_25, r_52, r_55, r_12, r_21, r_15, r_51, r_82, r_28, r_29, r_92, r_95, r_59,
      r_62, r_65, r_72, r_75, r_16, r_61, r_85,
      R_0_, F_ c1_, c2_, VQI ast, VQ_ vi, Qbar_, Fbar_, K1_, VSI ast, VS_ c1bar_,
      KO_, PI0_, PI1_, PI2_, PI3_, logk, logh, logy, logc, logka, logha, logya, logca,
      lz, xt, e1t, e2t, covvv, dati, ei, sume1, ci1, ci2, bone, boneb, varY, ve1b, ve2b;

/* Loop for different parameters, etc., to make 2 or three versions of y, k, h, c */

logk={};
logh={};
logy={};
logc={};

dati=1;
do while dati <= 3;

/* put in looping parameter values here */

/*
if dati==1; bet=0.988;
elseif dati==2; bet=0.90;
else; bet=0.95; endif;
*/

/* generate all of the parameters to use in final equations */
/* below, h is really htil */

w = (eye(2)-((A11-0)|(0-A22)))*(0|a2);
phi_ =
(((n^thet)*(1-bet*(1-delt))*exp(-w[2,1]/n)*exp(((2*thet)-1)*w[2,1]))/(bet*thet))^((-1)/(1-thet));
h_ = (exp(w[1,1])/gamm)*(1-thet)*
(((n^(-thet))*exp(-2*thet*w[2,1]))/(((n^(-thet))*exp(-2*thet*w[2,1])*phi_^thet)-
(phi_*exp(-w[2,1]))*(1-((1-delt)*exp(-w[2,1])/n)))));
ktil_ = phi_*h_;
ytil_ = h_*(n^(-thet))*exp(-2*thet*w[2,1])*(phi_^thet);
ctil_ = (h_*(n^(-thet))*exp(-2*thet*w[2,1])*(phi_^thet)) -
(phi_*exp(-w[2,1]))*(1-((1-delt)*exp(-w[2,1])/n));
phi0_ = (0|0|a2|0|0);
phi1_ = eye(5); phi1_[1,1]=0; phi1_[2,2]=A11; phi1_[3,3]=A22; phi1_[4,4]=0;
phi1_[5,5]=0; phi1_[4,2]=1; phi1_[5,3]=1;
sstar_ = (ln(ktil_) | w | w);
dstar_ = (ln(ktil_) | ln(h_));
B_ = zeros(5,2); B_[1,1]=1;

r_1 = -(exp(w[1,1])*exp(-w[2,1]))/ctil_;
r_7 = -(exp(w[1,1])/ctil_)*((thet*ytil_)+(((1-delt)/n)*exp(-2*w[2,1])*ktil_));
r_9 = (exp(w[1,1])/ctil_)*(-(thet*ytil_)+(exp(-w[2,1])*ktil_*(1-
((1-delt)/n)*exp(-w[2,1])))-(exp(-w[2,1])/n));

```

```

r_8 = exp(w[1, 1])*ln(cti l _);
r_5=0;
r_6=0;
r_2 = -(exp(w[1, 1])/cti l _)*exp(-w[2, 1]);

r_11 = -((exp(w[1, 1])*exp(-2*w[2, 1]))/(cti l _*bet^2)) -
((thet*(1-thet)*exp(w[1, 1])*yti l _)/cti l _);
r_18 = r_1;
r_81 = r_18;
r_19 = -(r_1*r_9*exp(-w[1, 1])) - ( (exp(-w[1, 1])/cti l _)*( ((thet^2)*yti l _) +
(((1-del t)/n)*exp(-2*w[2, 1])) ) );
r_91=r_19;
r_16=0;
r_61=r_16;
r_17 = -(r_1*r_7*exp(-w[1, 1])) - ( (exp(-w[1, 1])/cti l _)*( ((thet^2)*yti l _) +
(((1-del t)/n)*exp(-2*w[2, 1])) ) );
r_71=r_17;
r_98=r_9;
r_89=r_98;
r_88=r_8;
r_68=0;
r_86=r_68;
r_78=r_7;
r_87=r_78;
r_69=0;
r_96=r_69;
r_99=-((r_9^2)*exp(-w[1, 1])) + ( (exp(w[1, 1])/cti l _)*(
((thet^2)*yti l _)-(exp(-w[2, 1])*kti l _)+(((1-del t)/n)*exp(-2*w[2, 1])*kti l _) ) );
r_79=-((r_9*r_7)*exp(-w[1, 1])) + ( (exp(w[1, 1])/cti l _)*(
((thet^2)*yti l _)-(exp(-w[2, 1])*kti l _)+(((1-del t)/n)*exp(-2*w[2, 1])*kti l _) ) );
r_97=r_79;
r_66=0;
r_67=0;
r_76=r_67;
r_77 = -(r_7^2)*exp(-w[1, 1]) + (
((thet^2)*yti l _)+(((1-del t)/n)*exp(-2*w[2, 1])*kti l _) );
r_22=-((r_2^2)*exp(-w[1, 1]));
r_25 = -r_2*gamm*exp(-w[1, 1]);
r_52=r_25;
r_55 = -(gamm^2)*exp(-w[1, 1]) - ( (thet*(1-thet))*(exp(w[1, 1])/cti l _)*(yti l _/(h_^2))
);
r_12 = -r_1*r_2*exp(-w[1, 1]);
r_21=r_12;
r_15 = -(r_1*gamm) + ( (exp(w[1, 1])/cti l _)*(thet*(1-thet))*(yti l _/kti l _) );
r_51=r_15;
r_82=r_2;
r_28=r_82;
r_85=gamm;
r_29 = -(r_2*r_9*exp(-w[1, 1])) + ( (exp(w[1, 1])/cti l _)*exp(-w[2, 1]) );
r_92=r_29;
r_95 = -(r_9*gamm*exp(-w[1, 1])) + ( (exp(w[1, 1])/cti l _)*(thet*(1-thet))*(yti l _/h_)
);
r_59=r_95;
r_62=0;
r_65=0;
r_72 = -(r_2*r_7*exp(-w[1, 1]));
r_75 = ( (gamm/cti l _)*( ((thet)*yti l _)+(((1-del t)/n)*exp(-2*w[2, 1])*kti l _) ) ) - (
(exp(w[1, 1])/cti l _)*(thet*(1-thet))*(yti l _/h_) );

R_ = 0.5 * (
(r_11|r_81|r_91|r_61|r_71)~(r_18|r_88|r_98|r_68|r_78)~(r_19|r_89|r_99|r_69|r_79)~(r_
16|r_86|r_96|r_66|r_76)~(r_17|r_87|r_97|r_67|r_77) );
Q_ = 0.5 * ( (r_22|r_52)~(r_25|r_55) );

```

```

                                rokr1c. PRG
F_ = 0.5 * ( (r_12|r_82|r_92|r_62|r_72)~(r_15|r_85|r_95|r_65|r_75) );
c1_ = (r_2|r_5) - (2*sstar_'*F_)' - (2*dstar_'*Q_)' ;
c2_ = (r_1|r_6|r_7|r_8|r_9) - (2*dstar_'*F_)' - (2*sstar_'*R_)' ;

/* now, determine VQ and VS in order to get final parameters for DGPs, using Riccati
equations */

/* VQ */

VQI ast=eye(5)*-0.00;

vi=1;
do while vi <= 1000000;
    VQ_ = R_ + (bet*phi 1_'*VQI ast*phi 1_) - ((bet*phi 1_'*VQI ast*B_) + F_)*i nv(Q_ +
bet*B_'*VQI ast*B_)*((bet*B_'*VQI ast*phi 1_) + F_');
    if sumc(sumc((VQ_-VQI ast)^2)) .le 0.00000001;
        goto vend1;
    el se;
        VQI ast=VQ_;
        vi=vi+1;
    endi f;
endo;
vend1:
/*print "for VQ, use x sims, x= " vi;*/
if vi==1000000; print "VQ di dn't converge, and VQ is " VQ_; endi f;

Qbar_ = Q_ + bet*B_'*VQ_*B_;
Fbar_ = F_ + bet*phi 1_'*VQ_*B_;

K1_ = -Qbar_*Fbar_';
/*
print "K1";
print K1_;
*/
VSI ast=ones(5, 1)*-0.0;

vi=1;
do while vi <= 1000000;
    VS_ = i nv(eye(5) - (bet*(K1_'*B_' + phi 1_')))* ( (K1_'*(c1_ +
2*bet*B_'*VQ_*phi 0_) + c2_ + (2*bet*phi 1_'*VQ_*phi 0_) );
    if sumc(sumc((VS_-VSI ast)^2)) .le 0.00000001;
        goto vend2;
    el se;
        VSI ast=VS_;
        vi=vi+1;
    endi f;
endo;
vend2:
/*print "for VS, use x sims, x= " vi;*/
if vi==1000000; print "VQ di dn't converge, and VQ is " VQ_; endi f;

c1bar_ = ( c1_' + (bet*VS_'*B_) + (2*bet*phi 0_'*VQ_*B_) )';

K0_ = -0.5*i nv(Qbar_)*c1bar_;
/*
print "K0";
print K0_;
*/
/*K0_[1, 1]=0.0;
K0_[2, 1]=0.0;*/

PI 0_ = ( (K0_[1]+K1_[1, 3]*a2) | (K0_[2]+K1_[2, 3]*a2) );
PI 1_ = K1_[1, 1]|K1_[2, 1];

```

```

                                rokr1c. PRG
PI 2_ = ( ( (K1_[1, 2]*A11+K1_[1, 4])|(K1_[2, 2]*A11+K1_[2, 4]) ) ~ (
(K1_[1, 3]*A22+K1_[1, 5])|(K1_[2, 3]*A22+K1_[2, 5]) ) );
PI 3_ = ( (K1_[1, 2]|K1_[2, 2])~(K1_[1, 3]|K1_[2, 3]) );
/*
print "VS" VS_;
print "VQ" VQ_;
*/

/* estimate ve1, ve2, and cove1e2 */

/* generate (u x) */

ci 1=1;
ci 2=1;
boneb=1000000000000;
ve1=0.00005; ve2=0.0015; cove1e2=0.0;
do while ci 1<=14;
  ve1=ve1*2;
  do while ci 2<=14;
    ve2=ve2+0.0005;
    ux=zeros(smpl+100, 2);
    z=zeros(smpl+100, 1);
    z[1]=1;

    if dati==1;
      ee=sqrt(1/3)*rstudent(smpl+100, 2, 3);
    endi f;

    if dati==2;
      ve1=0.039; ve2=0.077;
      ee= rdn(smpl+100, 2); /* normal RVs */
      cova=(ve1~cove1e2)|(cove1e2~ve2);
      ee=(chol(cova)*ee)';
    endi f;

    if dati==3;
      ee= rdn(smpl+100, 2); /* normal RVs */
      cova=(ve1~cove1e2)|(cove1e2~ve2);
      ee=(chol(cova)*ee)';
    endi f;

    dgi =2;
    do while dgi <=smpl+100;
      ux[dgi, .]= ( (0|a2)+((A11~0)|(0~A22))* (ux[dgi-1, .]'+ee[dgi, .]') );
      z[dgi, .]=z[dgi-1, .]*exp(ux[dgi, 2]);
      dgi =dgi+1;
    endo;

    /* generate logk, logh, logy, logc */

    logka=zeros(smpl+100, 1);
    logha=zeros(smpl+100, 1);
    logya=zeros(smpl+100, 1);
    logca=zeros(smpl+100, 1);

    dgi =3;

```

```

                                rokr1c. PRG
do while dgi <=smp1+100;
    logka[dgi, 1]=ln(z[dgi -1, .]) + PI0_[1, 1] +
PI1_[1, 1]*(logka[dgi -1, .]-ln(z[dgi -2, .])) + PI2_[1, .]*ux[dgi -1, .]' +
PI3_[1, .]*ee[dgi, .]';
    logha[dgi, 1]=
PI1_[2, 1]*(logka[dgi -1, .]-ln(z[dgi -2, .])) + PI2_[2, .]*ux[dgi -1, .]' +
PI3_[2, .]*ee[dgi, .]';
    dgi =dgi +1;
endo;

dgi =3;
do while dgi <=smp1+100;
    logya[dgi, 1]=(1-thet)*ln(z[dgi, .]) - thet*ln(n) + (1-thet)*logha[dgi, 1] +
thet*logka[dgi -1, 1];
    logca[dgi, 1]=ln( exp(logya[dgi, 1]) - exp(logka[dgi, 1]) +
((1-del t)/n)*exp(logka[dgi -1, 1]) );
    dgi =dgi +1;
endo;
varY=vcx(logya[101: smp1+100, .]-logya[100: smp1+99, .]);
bone=abs(varY-vcx(ld_pc_Y));
if bone .le boneb;
    boneb=bone;
    ve1b=ve1;
    ve2b=ve2;
endif;

ci 2=ci 2+1;
endo;
ci 1=ci 1+1;
endo;
ve1=ve1b;
ve2=ve2b;
print "Best si geps: " ((ve1~ve2));

/* old sigma epsi lon cal cs
lz= (ln(pc_Y1[2: rows(pc_Y1), .]) + thet*ln(n) - (1-thet)*ln(pc_H1[2: rows(pc_H1), .]) -
thet*ln(pc_K1[1: rows(pc_K1)-1, .]))/(1-thet);
xt=lz[2: rows(lz), .]-lz[1: rows(lz)-1];
e2t = xt[2: rows(xt), .]-a2-A22*xt[1: rows(xt)-1, .];

e2t=e2t-meanc(e2t);
/*
print "e2t";
print e2t;
print "mean" meanc(e2t);
*/
e1t=zeros(rows(lz)-2, 1); /* = rows(e2t) = rows(pc_Y)-3 */
ei =1;
do while ei <=rows(e1t);
    if ei ==1;
        sume1=0;
    else;
        sume1=sume1*A11 + e1t[ei -1, 1];
    endif;
    e1t[ei] = (1/K1_[1, 2])*( ln(pc_K1[ei+3, .]) - lz[ei+1, .] - (K0_[1]+K1_[1, 3]*a2) -
(K1_[1, 1]*(ln(pc_K1[ei+2, .]) - lz[ei, .]))
- ((K1_[1, 5]+K1_[1, 3]*A22)*xt[ei, .]) -
((K1_[1, 4]+K1_[1, 2]*A11)*sume1) - (K1_[1, 3]*e2t[ei, .]) );
    ei =ei +1;
endo;

```

```

e1t=e1t-meanc(e1t);
/*
print "e1t";
print e1t;
print "mean" meanc(e1t);
*/
covvv=vcx(e1t~e2t);
ve1=covvv[1, 1];
ve2=covvv[2, 2];
cove1e2=covvv[1, 2];

print "Epsilon calcs are " (ve1~ve2~cove1e2);

ve1=0.001;
ve2=0.001;
cove1e2=-0.0;

*/

/* end cov calcs */

/* generate all of the data!!! */

/* generate (u x) */

ux=zeros(smpl+100, 2);
z=zeros(smpl+100, 1);
z[1]=1;

if dati==1;
    ee=sqrt(1/3)*rstudent(smpl+100, 2, 3);
endif;

if dati==2;
    ee= rndn(smpl+100, 2); /* normal RVs */
    cov=(ve1~cove1e2)|(cove1e2~ve2);
    ee=(chol(cov)*ee')';
endif;

if dati==3;
    ee= rndn(smpl+100, 2); /* normal RVs */
    cov=(ve1~cove1e2)|(cove1e2~ve2);
    ee=(chol(cov)*ee')';
endif;

dgi=2;
do while dgi<=smpl+100;
    ux[dgi, .]= ( (0|a2)+((A11~0)|(0~A22))*(ux[dgi-1, .]'+ee[dgi, .]')' );
    z[dgi, .]=z[dgi-1, .]*exp(ux[dgi, 2]);
    dgi=dgi+1;
enddo;

/* generate logk, logh, logy, logc */

logka=zeros(smpl+100, 1);
logha=zeros(smpl+100, 1);
logya=zeros(smpl+100, 1);

```



```

logca=zeros(smpl +100, 1);

dgi =3;
do while dgi <=smpl +100;
    logka[dgi , 1]=ln(z[dgi -1, .]) + PI0_[1, 1] +
    PI1_[1, 1]*(logka[dgi -1, .]-ln(z[dgi -2, .])) + PI2_[1, .]*ux[dgi -1, .]' +
    PI3_[1, .]*ee[dgi , .]';
    logha[dgi , 1]=
    PI0_[2, 1] +
    PI1_[2, 1]*(logka[dgi -1, .]-ln(z[dgi -2, .])) + PI2_[2, .]*ux[dgi -1, .]' +
    PI3_[2, .]*ee[dgi , .]';
    dgi =dgi +1;
endo;

dgi =3;
do while dgi <=smpl +100;
    logya[dgi , 1]=(1-thet)*ln(z[dgi , .]) - thet*ln(n) + (1-thet)*logha[dgi , 1] +
    thet*logka[dgi -1, 1];
    logca[dgi , 1]=ln( exp(logya[dgi , 1]) - exp(logka[dgi , 1]) +
    ((1-delt)/n)*exp(logka[dgi -1, 1]) );
    dgi =dgi +1;
endo;

/* add to final data */

logk=logk~logka;
logh=logh~logha;
logy=logy~logya;
logc=logc~logca;

dati =dati +1;

endo;

/* return two or three different growth rate of Y series, one for each of two
approximation methods */
/* with everything for a given alpha, beta, tau */
/* also, return the growth rates and secondly their lags */

retp( (logy[101: smpl +100, .]-logy[100: smpl +99, .]),
      (logh[101: smpl +100, .]-logh[100: smpl +99, .]) );
endp;

/* */
/* */
/* */
/* calculate the main statistic for a given actual data series and simulated data
series */
/* one statistic for each benchmark-alternative combination
*/
/* i.e. calculate Z_{j, T, S} statistics
*/
/* */
/* UNIVARIATE or MARGINAL DISTRIBUTION CASE */
/* */

proc (1) = zee1(Yd, Yds, uuone);
local zi, zN, simFben, simFalt, piceben, outu;

/* for a given data series (Yd), there are NS columns of simulated series, */
/* the first for the benchmark model and the rest for the alternative models */

/* */

```

rokr1c. PRG

```

outu={};
zN=sqrt(rows(Yd));

simFben=(1/rows(Yds[., 1]))*sumc(Yds[., 1] .le uuone);

piceben=(1/zN)*sumc( ((Yd[., .] .le uuone)-simFben)^2 );

zi=1;
do while zi <=cols(Yds)-1;

    simFalt=(1/rows(Yds[., 1]))*sumc(Yds[., zi+1] .le uuone);
    outu=outu~(piceben- ((1/zN)*sumc( ((Yd[., .] .le uuone)-simFalt)^2 )));
    zi=zi+1;
endo;

/* outu is a row of Z_{j, T, S} statistics, one for each bench versus alternative
comparison */
/* Thus, Yds, the simulated data, must have at least two columns to allow for the */
/* construction of one bench and at least one alternative */

/* this output is for a single u value */

retp(outu);
endp;

/* */
/* */
/* */
/* calculate the main statistic for a given actual data series and simulated data
series */
/* one statistic for each benchmark-alternative combination
*/
/* i.e. calculate Z_{j, T, S} statistics
*/
/* */
/* BIVARIATE or JOINT DISTRIBUTION CASE */
/* */

proc (1) = zee2(Yd, Ydl, Yds, Ydsl, uuone, uutwo);
local zi, zN, simFben, simFalt, piceben, outu;

/* for a given data series (Yd), there are NS columns of simulated series, */
/* the first for the benchmark model and the rest for the alternative models */

/* */

outu={};
zN=sqrt(rows(Yd));

simFben=(1/rows(Yds[., 1]))*sumc((Yds[., 1] .le uuone) .and (Ydsl[., 1] .le uutwo));
piceben=(1/zN)*sumc( ( (Yd[., 1] .le uuone) .and (Ydl[., 1] .le uutwo) )
-simFben)^2 );

/*print "simFben " simFben;
print "piceben " piceben; */

zi=1;
do while zi <=cols(Yds)-1;

    simFalt=(1/rows(Yds[., 1]))*sumc((Yds[., zi+1] .le uuone) .and (Ydsl[., zi+1] .le
uutwo));

```

```

/*pri nt "si mFal t " si mFal t; */
    outu=outu~(pi eceben- ((1/zN)*sumc( ( ( Yd[.,1] .le uuone) .and (Ydl [.,1] .le
outwo) ) -si mFal t)^2 )));
    zi =zi +1;
endo;

/* outu is a row of Z_{j,T,S} statistics, one for each bench versus al ternative
compari son */
/* Thus, Yds, the simulated data, must have at least two columns to allow for the */
/* constructi on of one bench and at least one al ternative */

/* this output is for a single u value */

/*pri nt "outu is " outu; */

retp(outu);
endp;

/* */
/* */
/* */
/* actual max stats */
/* */
/* */
/* */

proc (1) = stats(ouAv);
local ZT;

ZT={};

ZT=ZT|maxc(ouAv[1: rows(ouAv)]);

/* so one statistic outputed here */
/* if doing simulati ons, may do max across various subsets of those indi vi dual
elements of ouAv, thus */
/* generati ng size and power statistics, etc. */

retp(ZT);
endp;

/* */
/* */
/* */
/* bootstrap data sets */
/* */
/* */
/* */

proc (1) = bootk(dat1,l val);
local N, num_ uns, undraw1, x1, i b;

N=rows(dat1);
num_ uns=N/l val ;

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

undraw1=round((N-l val ) *rndu(num_ uns, 1));

x1={};

```

rokr1c. PRG

```

i b=1;
do while i b<=num_uns;
  x1=x1|dat1[undraw1[i b]+1: undraw1[i b]+l val , . ];
  i b=i b+1;
endo;

retp(x1);
endp;

/* ----- */
/* ----- */
/* ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^ MAIN PROGRAM ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^ */
/* ----- */
/* ----- */

/* for use in simulation */

/*{Udat}=sim1(1000);
Umin=(minc(Udat[. , 1: 8]));
Umax=(maxc(Udat[. , 1: 8]));
Uinc=(Umax-Umin)/Unum;
uone=meanc(udat); */

/* for empirical application */

Umin=(minc(minc(ld_pc_Y~ld_pc_H)));
Umax=(maxc(maxc(ld_pc_Y~ld_pc_H)));
Uinc=(Umax-Umin)/Unum;

outzaa={};

/* raw data assumed already to be in log differences */

/* define length of the simulated samples, given TT, where TT is actual data sample
used */

outza=zeros(12, 23);

tsi=1;
do while tsi <= 4;
  if tsi == 1; SS=TT; else if tsi==2; SS=2*TT; else if tsi==3; SS=5*TT; else;
SS=10*TT; endif;

  li=1;
  do while li<= 3;
    l_val=ls[li]; /* l_val for actual data */

    /* now, make l_val s for simulated data */
    if tsi==1; l_val S=ls[li]; else if tsi==2; l_val S=2*ls[li]; else;
l_val S=5*ls[li]; endif;
/* assume that the "l_val" values are for the actual data -- for the
simulated, multiply them */
/* by the factor 1, 2, or 5, as SS is made by the same factors multiplying TT
*/

    si=1;
    do while si<=simtot;

      /* simulate data according to some different models, where TT is the length
of the simulated sample */

```

rokr1c. PRG

```

/* set parameter estimation data equal to raw data */

pc_Y1=pc_Y;
pc_K1=pc_K;
pc_H1=pc_H;

{Ydats,Hdats}=sim2(SS); /* this is the simulated data and its lag, */
/* given the actual data and any calculations done */
/* to estimate parameters, form the actual data, etc. */

/* now, calculate all stats over U range */

outttu={};
ui=1; uuui=Umin-Uinc;
do while ui <=Unum+1;
    uuui=uuui+uinc;
    uj=1; uuuj=Umin-Uinc;
    do while uj <=Unum+1;
        uuuj=uuuj+uinc;
        {outtu}=zee2(ld_pc_Y,ld_pc_H,YdatS,HdatS,uuui,uuuj);
        uj=uj+1;
        outttu=outttu|outtu;
    endo;
    ui=ui+1;
endo;

/* average up across the u */

oboy=meanc(outttu);

/*print "oboy"; print oboy; */

/* calculate statistics */

{ZZ}=stats(oboy);

/*
print "stats ";
print ZZ;
*/

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

ZZZb1={}; ZZZb2={};

booti=1;
do while booti <= bootnum;

/*
print "Boot sim " booti;
*/

    {bdatAct} = bootk((pc_Y~pc_K~pc_H~ld_pc_Y~ld_pc_H), l_val);

/* change actual data used in parameter estimation in simulation routine
*/

    pc_Y1=bdatAct[., 1];
    pc_K1=bdatAct[., 2];
    pc_H1=bdatAct[., 3];

    {YdatS_b,HdatS_b}=sim2(SS);

    {bdatSim} = bootk((YdatS_b~HdatS_b), l_val S);

/* I carry out the usual boot approach */

```

rokr1c. PRG

```

    outttu={};
    ui =1; uuui =Umi n-Ui nc;
    do whi le ui <=Unum+1;
        uuui =uuui +ui nc;
        uj =1; uuuj =Umi n-Ui nc;
        do whi le uj <=Unum+1;
            uuuj =uuuj +ui nc;
/*pri nt bdatAct[. , 4]~bdatAct[. , 5]~bdatSi m[. , 1]~bdatSi m[. , 2]; */

{outtu}=zee2(bdatAct[. , 4], bdatAct[. , 5], bdatSi m[. , 1: col s(YdatS_b)], bdatSi m[. , col s(Yda
tS_b)+1: 2*col s(YdatS_b)], uuui , uuuj );
/*pri nt "outtu " outtu; */
        outttu=outttu|outtu;
        uj =uj +1;
    endo;
    ui =ui +1;
endo;
/*pri nt "outttu"; */
/*pri nt outttu; */
    boboy1=meanc(outttu)-oboy; /* these are the usual boot stats wi th the
booted mi nus the actual */
/*pri nt "boboy1"; */
/*pri nt boboy1; */

    /* II carry out the al ternative boot for S growi ng faster than T */

    outttu={};
    ui =1; uuui =Umi n-Ui nc;
    do whi le ui <=Unum+1;
        uuui =uuui +ui nc;
        uj =1; uuuj =Umi n-Ui nc;
        do whi le uj <=Unum+1;
            uuuj =uuuj +ui nc;
            {outtu}=zee2(bdatAct[. , 4], bdatAct[. , 5], YdatS_b, HdatS_b, uuui , uuuj );
            uj =uj +1;
            outttu=outttu|outtu;
        endo;
        ui =ui +1;
    endo;
    boboy2=meanc(outttu)-oboy; /* these are the alt boot stats wi th the
booted mi nus the actual , ut where */
/* only the actual data are resampled, and not
the booted, i . e. S grow faster T */

    /* cal cul ate stati sti cs */

    {ZZb1}=stats(boboy1);
    ZZZb1=ZZZb1|ZZb1';

    {ZZb2}=stats(boboy2);
    ZZZb2=ZZZb2|ZZb2';

    booti =booti +1;
    endo;
/*
pri nt "Boot Stats";
pri nt ZZZb1~ZZZb2;
*/
    /* construct cri ti cal val ues */

    cv1_10={}; cv1_5={};
    cvi =1;

```

```

                                rokr1c. PRG
do while cvi <= col s(ZZZb1);
    temp=sortc(ZZZb1[. , cvi ], 1);
    cv1_10=cv1_10|temp[trunc(0.90*rows(ZZZb1))];
    cv1_5=cv1_5|temp[trunc(0.95*rows(ZZZb1))];
    cvi =cvi +1;
endo;

cv2_10={}; cv2_5={};
cvi =1;
do while cvi <= col s(ZZZb2);
    temp=sortc(ZZZb2[. , cvi ], 1);
    cv2_10=cv2_10|temp[trunc(0.90*rows(ZZZb2))];
    cv2_5=cv2_5|temp[trunc(0.95*rows(ZZZb2))];
    cvi =cvi +1;
endo;

print " _____ ";
print "sim " si " done - {li , tsi } = " (li ~tsi);
print "stats CV 5%,10% ";
print ZZ~cv1_5~cv1_10~cv2_5~cv2_10;

/* empirical collecting of results */

outza[li+(tsi -1)*3, 1]=ZZ;
outza[li+(tsi -1)*3, 2:5]=cv1_5~cv1_10~cv2_5~cv2_10;
outza[li+(tsi -1)*3, 6:8]=(sumc(YdatS . l t 0))' /rows(YdatS);
outza[li+(tsi -1)*3, 9:11]=mi nc(YdatS)' ;
outza[li+(tsi -1)*3, 12:14]=maxc(YdatS)' ;
outza[li+(tsi -1)*3, 15:17]=meanc(YdatS)' ;
outza[li+(tsi -1)*3, 18:20]=stdc(YdatS)' ;
corr1=corr(x(Ydats[. , 1]~Hdats[. , 1]));
corr1a=corr1[2, 1];
corr2=corr(x(Ydats[. , 2]~Hdats[. , 2]));
corr2a=corr2[2, 1];
corr3=corr(x(Ydats[. , 3]~Hdats[. , 3]));
corr3a=corr3[2, 1];
outza[li+(tsi -1)*3, 21:23]=corr1a~corr2a~corr3a;

/* simulation collecting of results */

/*outza[tsi , 1]=ZZ; */
/*outza[tsi , (li -1)*4+2: (li -1)*4+5]=outza[tsi , (li -1)*4+2: (li -1)*4+5]+(ZZ . ge
(cv1_5~cv1_10~cv2_5~cv2_10)); */

/* */

si =si +1;
endo;
print "-----";
print " ";
print "Total Output for li and tsi = " (li ~tsi);
print " ";
print "5%, 10% nominal rejection rates";
print outza[. , . ]/si mtot;
print " ";
print "Actual and Simulated Growth Rate (per Capi ta)";

i ss=1; YdatsSRT={};
do while i ss<= col s(Ydats);
    temp=sortc(Ydats, i ss);
    YdatsSRT=YdatsSRT~temp[. , i ss];
    i ss=i ss+1;

```

```

endo;

if rows(YdatS) > rows(I d_pc_Y);
  ext=rows(YdatS)-rows(I d_pc_Y);
  extr=1*ones(ext, cols(I d_pc_Y));
  print (sortc((I d_pc_Y|extr), 1))~YdatsSRT;
else;
  print (sortc((I d_pc_Y), 1))~YdatsSRT;
endif;
print "total s " rows(sortc(I d_pc_Y, 1))~rows(YdatsSRT);
print "-----";

  li=li+1;
endo;

tsi=tsi+1;
endo;

print "_____FINAL OUTPUT _____";
print "_____Rokko Paper _____";
print "All output for sample = " TT;
print "Column";
print "Zstat boot1 5% 10% boot2 5% 10% %<0(each y vbl) mi n max mean stdc
corr(y1, h1)";
print "Row";
print "ti=1 ls=1";
print "ti=1 ls=2";
print "ti=1 ls=3";
print "ti=2 ls=1 ... ";
print " ";
print (outza[. , 1: 5]~(outza[. , 6: 8]*100)~outza[. , 9: 23])/si mtot;

print "Actual s for %<0, mi n, max, mean, stderr, corr(y1, h1)";
  corr1=corr(x(I d_pc_Y~I d_pc_H);
  corr1a=corr1[2, 1];
print ((sumc(I d_pc_Y . I t
0))' /rows(I d_pc_Y))~mi nc(I d_pc_Y)~maxc(I d_pc_Y)~meanc(I d_pc_Y)~stdc(I d_pc_Y)~corr1a;

output off;

```