[6] GMM Programs

Program: gmm1.prg

```
/*
** GMM FOR SINGLE EQUATION
** Using Newey-West and Andrews Covariance Matrix
*/

new ;

@ LOADING DATA @

    load dat[923,17] = mwemp.db ; @ Data on employed married women @

@ OPEN OUTPUT FILE @

    output file = gmm_1.out reset ;

@ DEFINE VARIABLES @

    lrate   = dat[.,1]  ;
    edu     = dat[.,2]  ;
    urb     = dat[.,3]  ;
    minor   = dat[.,4]  ;
    age     = dat[.,5]  ;
    tenure  = dat[.,6]  ;
    expp    = dat[.,7]  ;
    regs    = dat[.,8]  ;
    occw    = dat[.,9]  ;
    occb    = dat[.,10] ;
    indumg  = dat[.,11] ;
    indumn  = dat[.,12] ;
    unionn  = dat[.,13] ;
    unempr  = dat[.,14] ;
    lofinc  = dat[.,15] ;
    hwork   = dat[.,16] ;
    kids5   = dat[.,17] ;
    lhwork  = ln(hwork+1) ;

@ Define # of observations @

   n = rows(dat) ;

@  DEFINE EQUATION @

proc eqq(b)  ;
local equ    ;

   equ = lrate - b[1]*lhwork - b[2]*ones(n,1) - b[3]*edu
               - b[4]*edu^2 - b[5]*expp - b[6]*expp^2
               - b[7]*age - b[8]*age^2 - b[9]*occw - b[10]*occb
               - b[11]*unempr - b[12]*regs - b[13]*minor
               - b[14]*indumg - b[15]*unionn - b[16]*urb ;

retp(equ) ; endp      ;
```

```
@   DEFINE INSTRUMENTAL VARIABLES  @

    iv = ones(n,1)~edu~(edu^2)~age~(age^2)~expp~(expp^2);
    iv = iv~occw~occb~unempr~regs~minor~indumg~unionn~urb~kids5~lofinc;

@   DEFINE INITIAL VALUE @

    bb = zeros(16,1) ;

@   Choose np = 1 for Newey-West;
           np = 0 for Andrews        @

    np = 1 ;

@   Specify Bandwidth @

    bandw = 0 ; @ if bandw = 0, White-Correction @

@   iter = 0 for two-step GMM ; iter = 1 for iterative GMM @

    iter = 1 ;

/*
**  DO NOT CHANGE FROM HERE
*/

@   Procedure for Newey-West   @

proc nw(uv,l) ;
local j,omeo,n,w,ome  ;

    n    = rows(uv)  ;
    omeo = uv'uv/n   ;

    j = 1; do while j <= l  ;

        w    = 1 - j/(l+1)       ;
        ome  = uv[j+1:n,.]'uv[1:n-j,.]/n ;
        omeo = omeo + w*(ome+ome')       ;

    j = j + 1 ; endo ;

retp(omeo) ;
endp       ;

@   Procedure for Andrews (1991) @

proc quadk(z) ;
local rr ;

  rr = ( 3/(6*pi*z/5)^2 )
       *( 5*sin(6*pi*z/5)/(6*pi*z) - cos(6*pi*z/5) ) ;

retp(rr) ;
endp     ;
```

```
proc andrews(uv,l) ;
local j,omeo,n,w,ome  ;

    n    = rows(uv)  ;
    omeo = uv'uv/n   ;

    j = 1; do while j <= n-1 ;

        w    = j/(l+1)                    ;
        ome  = uv[j+1:n,.]'uv[1:n-j,.]/n  ;
        omeo = omeo + quadk(w)*(ome+ome') ;

    j = j + 1 ; endo ;

retp(omeo) ;
endp       ;


library optmum;
#include optmum.ext;
optset ;

/* FIRST ROUND */

@   Creating weighting matrix @

    w = invpd(iv'iv) ;

proc  f(b) ;
local m    ;

    m = iv'eqq(b) ;

RETP(  m'w*m  ) ;
ENDP  ;

@ starting values @

b0 = bb  ;

__title = "FIRST-STAGE GMM ";

_opgtol = 1e-4;
_opstmth = "BFGS HALF";
__output = 0 ;

{b,func,grad,retcode} = optprt(optmum(&f,b0)) ;

initb = b ;

/* SECOND ROUND */

cre = 10 ; do while cre >= 0.1e-4 ;
initb = initb ;
```

```
@ CREATING WEIGHTING MATRICES @

    if np == 1 ; gosub weight1 ;
    else       ; gosub weight2 ;
    endif      ;

library optmum;
#include optmum.ext;
optset ;

/* Second Round GMM */

proc  f1(b) ;
local m    ;

    m = iv'eqq(b) ;

RETP(  m'w*m ) ;
ENDP  ;

@ starting values @

B0 = initb ;

__title = "SECOND-STAGE GMM ";
_opgtol = 1e-4;
_opstmth = "BFGS HALF";
__output = 0 ;

{b,func,grad,retcode} = optprt(optmum(&f1,b0)) ;

cre = iter*(initb-b)'(initb-b) ;
initb = b ;
endo      ;


/* CONPUTING STANDARD ERRORS */

format /rd 12,4 ;

proc mom(b)     ;
retp(iv'eqq(b)) ;
endp            ;

gra = gradp(&mom,b) ;
cov = invpd(gra'w*gra) ;
se  = sqrt(diag(cov))  ;
tst = b./se  ;

"" ;
"GMM RESULTS" ;
"" ;
"      coeff.      std. err.      t-st " ;
b~se~tst     ;
```

```
/*  HANSEN TEST */

df = cols(iv) - rows(b) ;
"" ;
"J TEST, DF, P-Val =" func  df  cdfchic(func,df) ;

end ; @ This means no longer using subroutines  @

output off ;

weight1:
    w  = invpd( n*nw(iv.*eqq(initb),bandw) ) ;
    return ;

weight2:
    w  = invpd( n*andrews(iv.*eqq(initb),bandw) ) ;
    return ;
```

Outcome in gmm1.out

```
================================================================================
                              FIRST-STAGE GMM
================================================================================
 OPTMUM Version 3.1.4                                        5/15/02  12:53 am
================================================================================

return code =    0
normal convergence

Value of objective function        4.529132

Parameters    Estimates    Gradient
-----------------------------------------
P01              0.3297        0.0000
P02              0.7490        0.0000
P03             -0.1137        0.0000
P04              0.0072        0.0000
P05              0.0273        0.0000
P06             -0.0005        0.0001
P07             -0.0080        0.0000
P08              0.0000        0.0004
P09              0.1759        0.0000
P10              0.0213        0.0000
P11             -0.3140        0.0000
P12             -0.0382        0.0000
P13             -0.0850        0.0000
P14              0.1592        0.0000
P15              0.1382        0.0000
P16              0.1737        0.0000

Number of iterations      119
Minutes to convergence     0.35900
```

```
================================================================================
                              SECOND-STAGE GMM
================================================================================
 OPTMUM Version 3.1.4                                          5/15/02  12:55 am
================================================================================

return code =    0
normal convergence

Value of objective function       35.083812

Parameters    Estimates    Gradient
----------------------------------------
P01              0.2958        0.0000
P02              0.8512        0.0000
P03             -0.1202        0.0000
P04              0.0072        0.0003
P05              0.0273        0.0000
P06             -0.0005       -0.0001
P07             -0.0046        0.0001
P08              0.0000        0.0017
P09              0.1725        0.0000
P10             -0.0087        0.0000
P11             -0.1895       -0.0000
P12             -0.0310        0.0000
P13             -0.1075       -0.0000
P14              0.1489        0.0000
P15              0.1490       -0.0000
P16              0.1870       -0.0000

Number of iterations      116
Minutes to convergence     0.36817

GMM RESULTS

      coeff.      std. err.      t-st

      0.2958       0.1918       1.5426
      0.8512       0.4038       2.1080
     -0.1202       0.0425      -2.8306
      0.0072       0.0017       4.1746
      0.0273       0.0086       3.1670
     -0.0005       0.0003      -1.9513
     -0.0046       0.0158      -0.2911
      0.0000       0.0002       0.0186
      0.1725       0.0405       4.2531
     -0.0087       0.0477      -0.1824
     -0.1895       0.5524      -0.3431
     -0.0310       0.0285      -1.0902
     -0.1075       0.0315      -3.4132
      0.1489       0.0333       4.4660
      0.1490       0.0335       4.4434
      0.1870       0.0283       6.6106

J TEST, DF, P-Val =     35.0838       1.0000       0.0000
```

[7] Estimation of Dynamic Panel Data Models

Program: dynam.prg

```
    /*************************************************
    ** ESTIMATION OF SYMPLE DYNAMIC PANEL DATA MODELS **
    **              WRITTEN BY S. C. AHN              **
    *************************************************/

@ CLEAN MEMMORY @
    new  ;

@ OPEN OUTPUT FILE @
    output file = dynam.out reset ;

@ FORMAT OUTPUT FILE @
    format /rd 10,3 ;

@ LOADING DATA @
      load dat[500,10] = pandy1.db ;

@ Define yy @
      yy = dat;

@ Define N and T @
      nn = 500;
      tt = 9;

/* From here, do not make any change */

@ CREATING DIFFERENCED DATA: DYY INCLUDES (Y0-Y1),...,(YT-1-YT) @
    dyy = yy[.,2]-yy[.,1] ;
    j  = 2  ; do while j <= tt ;
        dyy = dyy~(yy[.,j+1]-yy[.,j]) ;
    j  = j+1; endo ;

@ STACKING DATA @
    depy  = vec(yy[.,2:tt+1]')   ; @ Y @
    lagy  = vec(yy[.,1:tt]')     ; @ Y_{-1} @
    pdepy = meanc(yy[.,2:tt+1]') .*. ones(tt,1) ; @ P_v Y @
    plagy = meanc(yy[.,1:tt]' ) .*. ones(tt,1) ;  @ P_v Y_{-1} @
    qdepy = depy - pdepy ;  @ Q_v Y @
    qlagy = lagy - plagy ;  @ Q_v Y_{-1} @
    ddepy = vec(dyy[.,2:tt]');
    dlagy = vec(dyy[.,1:tt-1]');
    abwdy = ddepy ;
    abwly = vec(yy[.,2:tt]') ;
    abwdly= dlagy            ;
```

```
@ CREATING ARELLANO AND BOND, ARELLANO-BOVER IVS @

@ CREATING L(T,P) MATRIX: NEEDED TO CREATE AB IVS @
    proc ll(t,p) ;
    local i,u    ;

        u = zeros(t,p) ;
        i = 1 ; do while i <= p ;
            u[t-p+i-1,i] = - 1  ; u[t-p+i,i] = 1  ;
        i = i + 1 ; endo        ;

    retp(u) ; endp ;

@ CREATING AHLL(T,P) MATRIX: NEEDED TO CREATE DIAGONAL ANDERSON-HSIAO IVS @
    proc ahll(t,p) ;
   local i,u ;

        u = zeros(t,1) ;
        u[p+2] = - 1  ; u[p+1] = 1  ;

    retp(u) ; endp ;

@ CREATING AB INSTRUMENTS @
    abiv = zeros(tt*nn,1) ;
    j = 1     ; do while j <= tt-1 ;
        abiv = abiv~( yy[.,j] .*. ll(tt,tt-j) ) ;
    j = j + 1 ; endo ;

    pp   = tt*(tt-1)/2    ;
    abiv = abiv[.,2:pp+1] ;

@ CREATING SUB-AB (Diagonal form of Andersen-Hsiao) INSTRUMENTS @
    ahiv = zeros(tt*nn,1) ;
    j = 1     ; do while j <= tt-1 ;
        ahiv = ahiv~( yy[.,j] .*. ahll(tt,j-1) ) ;
    j = j + 1 ; endo ;

    ahpp=tt-1 ;
    ahiv = ahiv[.,2:ahpp+1]  ;

@ CREATING ABo INSTRUMENTS @
    aboiv = zeros(tt*nn,1) ;
    j = 1     ; do while j <= tt-1 ;
      mmmm = zeros(tt,1) ;
      mmmm[j+1] = 1      ;
      aboiv = aboiv~( dyy[.,j] .*. mmmm ) ;
    j = j + 1 ; endo ;

    aboiv = aboiv[.,2:cols(aboiv)]  ;

@ CREATING AB-Type ABo (AABo) INSTRUMENTS @
    aaboiv = zeros(tt*nn,1) ;
    j = 1     ; do while j <= tt-1 ;
      mmmm = zeros(tt,1) ;
      mmmm[j+1] = 1      ;
      aaboiv = aaboiv~( dyy[.,1:j] .*. mmmm ) ;
    j = j + 1 ; endo ;

    aaboiv = aaboiv[.,2:cols(aaboiv)] ;
```

```
@ CREATING DIAGONAL LAGGED Y @
    diagly = zeros(tt*nn,1) ;
    j = 1 ; do while j <= tt ;
        mmmm = zeros(tt,1) ;
        mmmm[j] = 1        ;
        diagly = diagly~(yy[.,j] .*. mmmm) ;
    j = j + 1 ; endo  ;
        g1pp   = cols(diagly) ;
        diagly = diagly[.,2:g1pp] ;

@ # OF AB INSTRUMENTS @
    nabiv  = pp ;

@ # OF AH INSTRUMENTS @
    nahiv  = cols(ahiv) ;

/**********************
** ARELLANO-BOND GMM **
**********************/

@ TSLS ESTIMATOR @

    tscov = invpd( (lagy'abiv)*invpd(abiv'abiv)*(abiv'lagy) ) ;
    tsb   = tscov*(lagy'abiv)*invpd(abiv'abiv)*(abiv'depy)    ;
    qres  = qdepy - qlagy*tsb                                 ;
    tssige2 = (qres'qres/(nn*tt-tt-1)) ;
    tscov   = tssige2*tscov           ;

res  = depy - lagy*tsb  ;

opivab = zeros(1,cols(abiv)) ;
j = 0 ; do while j <= nn-1 ;
    zz = res[tt*j+1:tt*j+tt,.]'abiv[tt*j+1:tt*j+tt,.] ;
    opivab = opivab|zz ;
j = j + 1 ; endo ; opivab = opivab[2:nn+1,.] ;

optwab = invpd(opivab'opivab)    ;


abv = INVPD( (lagy'abiv)*(optwab)*(abiv'lagy) ) ;
abb = abv*(lagy'abiv)*(optwab)*(abiv'depy)      ;

abe  = depy - lagy*abb ;
abj  = (abe'abiv)*optwab*(abiv'abe);
abdf = cols(abiv) - rows(abb);


/**********************
** ARELLANO-BOVER GMM **
**********************/

iv = abiv~aboiv ;

@ TSLS ESTIMATOR @

    tscov = invpd( (lagy'iv)*invpd(iv'iv)*(iv'lagy) ) ;
    tsb   = tscov*(lagy'iv)*invpd(iv'iv)*(iv'depy)    ;
    qres  = qdepy - qlagy*tsb                         ;
    tssige2 = (qres'qres/(nn*tt-tt-1)) ;
    tscov   = tssige2*tscov           ;
```

```
res  = depy - lagy*tsb  ;

opivab = zeros(1,cols(iv)) ;
j = 0 ; do while j <= nn-1 ;
    zz = res[tt*j+1:tt*j+tt,.]'iv[tt*j+1:tt*j+tt,.] ;
    opivab = opivab|zz ;
j = j + 1 ; endo ; opivab = opivab[2:nn+1,.] ;

optwab = invpd(opivab'opivab)    ;

abov = invpd( (lagy'iv)*(optwab)*(iv'lagy) ) ;
abob = abov*(lagy'iv)*(optwab)*(iv'depy)    ;

aboe = depy - lagy*abob ;
aboj = (aboe'iv)*optwab*(iv'aboe);
abodf = cols(iv) - rows(abob) ;

/*********************
** AHN-SCHMIDT GMM1 **
*********************/

/*
**  First step GMM
*/

@ define some global variables for gmm1 @

    well = ll(tt,tt-1)                ;
    well = well[1:tt,1:tt-2]          ;
    m11  = abiv'depy ;
    m12  = abiv'lagy ;
    m21  = (yy[.,tt+1].*.well)'depy ;
    m22  = (yy[.,tt].*.well)'depy + (yy[.,tt+1].*.well)'lagy  ;
    m23  = (yy[.,tt].*.well)'lagy ;

library optmum;
#include optmum.ext;
optset ;

@ Finding initial value @

@ step 1 for gmm1 @

    m11  = abiv'depy ;
    m12  = abiv'lagy ;
    m21  = (yy[.,tt+1].*.well)'depy ;
    m22  = (yy[.,tt].*.well)'depy + (yy[.,tt+1].*.well)'lagy  ;
    m23  = (yy[.,tt].*.well)'lagy ;
    wei1 = invpd((abiv'abiv)/nn)~zeros(cols(abiv),tt-2) ;
    wei2 = zeros(tt-2,cols(abiv))~eye(tt-2) ;
    wei  = wei1|wei2 ;
```

```
                          /* GRADIENT */

proc grad1(b) ;
local gra, m  ;

    m    = (m11-m12*b)|(m21-m22*b+m23*b^2) ;
    gra  = m12|(m22-2*m23*b) ;

retp( - 2*m'wei*gra
    ) ;
endp  ;

                                /* GMM */

proc  f1(b) ;
local m    ;

    m    = (m11-m12*b)|(m21-m22*b+m23*b^2) ;

retp(  m'wei*m
    ) ;
ENDP  ;

_opgdprc = &grad1 ;

@ starting values @

b0 = -1 ;

__title = "GMM11";

_opgtol = 1e-4 ;
_opstmth = "bfgs brent";
__output = 0 ;
{b,func,grad,retcode} = optmum(&f1,b0) ;

g = m12|(m22-2*m23*b) ;
v = invpd(g'g) ;
bb1 = func~b~v ;

@ step 2 @

                                /* GMM */

proc  f2(b) ;
local m    ;

    m    = (m11-m12*b)|(m21-m22*b+m23*b^2) ;

retp(  m'wei*m
    ) ;
ENDP  ;

_opgdprc = &grad1 ;

@ starting values @

b0 = 1  ;

__title = "GMM12";
```

```
_opgtol = 1e-4 ;
_opstmth = "bfgs brent";
__output = 0 ;
{b,func,grad,retcode} = optmum(&f2,b0) ;

g = m12|(m22-2*m23*b) ;
v = invpd(g'g) ;
bb2 = func~b~v  ;
bb = bb1|bb2 ;
bb = sorthc(bb,1) ;

tsbb = bb[1,2] ;

@ define some global variables for gmm1 @

REST     = YY[.,TT+1] - YY[.,TT]*tsbb ;
IVB1     = REST .*. WELL              ;
RES      = DEPY - LAGY*tsbb           ;
AS1IV    = ABIV~IVB1           ;

opivas1 = zeros(1,cols(as1iv)) ;
j = 0 ; do while j <= nn-1 ;
    zz = res[tt*j+1:tt*j+tt,.]'as1iv[tt*j+1:tt*j+tt,.] ;
    opivas1 = opivas1|zz ;
j = j + 1 ; endo ; opivas1 = opivas1[2:nn,.] ;

opwas1   = invpd(opivas1'opivas1) ;


@ step 1 for gmm1 @

                          /* GRADIENT */

proc grad2(b) ;
local gra, m  ;

    m    = (m11-m12*b)|(m21-m22*b+m23*b^2) ;
    gra  = m12|(m22-2*m23*b) ;

retp( - 2*m'opwas1*gra
    ) ;
endp  ;

                                /* GMM */

proc  f3(b) ;
local m    ;

    m    = (m11-m12*b)|(m21-m22*b+m23*b^2) ;

retp(  m'opwas1*m
    ) ;
ENDP  ;

_opgdprc = &grad2 ;
```

```
@ starting values @

b0 = -1 ;

__title = "GMM11";

_opgtol = 1e-4 ;
_opstmth = "bfgs brent";
__output = 0 ;
{b,func,grad,retcode} = optmum(&f3,b0) ;

g = m12|(m22-2*m23*b) ;
v = invpd(g'opwas1*g) ;
bb1 = func~b~v        ;

@ step 2 @

                                /* GMM */

proc  f4(b) ;
local m    ;

    m    = (m11-m12*b)|(m21-m22*b+m23*b^2) ;

retp(  m'opwas1*m
    ) ;
ENDP  ;

_opgdprc = &grad2 ;

@ starting values @

b0 = 1  ;

__title = "GMM12";

_opgtol = 1e-4 ;
_opstmth = "bfgs brent";
__output = 0 ;
{b,func,grad,retcode} = optmum(&f4,b0) ;

g = m12|(m22-2*m23*b) ;
v = invpd(g'opwas1*g) ;
bb2 = func~b~v  ;

bb = bb1|bb2 ;
bb = sorthc(bb,1) ;

as1j = bb[1,1] ;
as1b = bb[1,2] ;
as1v = bb[1,3] ;
as1df = rows(g) - rows(as1b) ;


"" ;
"AB GMM RESULTS:" ;
"  estimates    st. err.   t-stat" ;
abb~sqrt(abv)~(abb/sqrt(abv));
"";
" J-test statistic, df, pval:"  abj abdf cdfchic(abj,abdf) ;
```

```
"";
"ABo GMM RESULTS:" ;
"  estimates    st. err.   t-stat";
abob~sqrt(abov)~(abob/sqrt(abov));
"";
" J-test statistic, df, pval:"  aboj abodf cdfchic(aboj,abodf) ;
"";
"AS1 GMM RESULTS:" ;
"  estimates    st. err.   t-stat";
as1b~sqrt(as1v)~(as1b/sqrt(as1v));
"";
" J-test statistic, df, pval:"  as1j as1df cdfchic(as1j,as1df) ;
"";

OUTPUT OFF
```

Outcome in dynam.out

```
AB GMM RESULTS:
  estimates    st. err.   t-stat
     0.718      0.043     16.623

J-test statistic, df, pval:    35.979     35.000      0.423

ABo GMM RESULTS:
  estimates    st. err.   t-stat
     0.867      0.013     64.521

J-test statistic, df, pval:    53.621     43.000      0.129

AS1 GMM RESULTS:
  estimates    st. err.   t-stat
     0.836      0.014     60.403

J-test statistic, df, pval:    45.362     42.000      0.334
```

Program: dynamic1.prg

```
    /*********************************************************
    ** MONTE CARLO EXPERIMENTS FOR DYNAMIC PANEL DATA MODELS **
    **               WRITTEN BY S. C. AHN                    **
    *********************************************************/

@ CLEAN MEMMORY @
    new  ;

@ OPEN OUTPUT FILE @
    output file = dynamic1.out reset ;

@ FORMAT OUTPUT FILE @
    format /rd 8,3 ;

/***************************************
**  Starting comments on the output file **
***************************************/

" MODEL SPECIFICATION: " ;
"       y_t = d*y_{t-1} + a + e_t ;" ;
"       y_0 = gamma*a + e_0        " ;
"       Stationary y:  gamma = 1/(1-d)" ;

/*****************************
** Data Generating Process **
*****************************/

@ For the size of N @
    j1 = 2           ;
    let jj[4,1] = 50 100 300 500   ;
    do while j1 <= 2 ;
    jj1 = jj[j1,1]   ;

@ For the size of T @
    j2 = 1           ;
    let jjj[3,1] = 4 7 11  ;
    do while j2 <= 1 ;
    jj2 = jjj[J2,1]  ;

@ For the size of delta @
    j3 = 1           ;
    let jjjj[3,1] = 0.9 0.8 0.5 ;
    do while j3 <= 3 ;
    jj3 = jjjj[J3,1] ;

@ For siga @
    j4 = 3           ;
    let jjjjj[3,1] = 0 0.5 1 ;
    do while j4 <= 3 ;
    jj4 =jjjjj[J4,1] ;
```

```
@ Values of major parameters @
    ddd    = 1     ; @ SEED @
    iter   = 100   ; @ # OF ITERATIONS @
    tt     = jj2   ; @ TIME HORIZON @
    nn     = jj1   ; @ # OF CROSS-SECTIONAL UNITS @
    dd     = jj3   ; @ DELTA  @
    se     = 1     ; @ SE(E) @
    sa     = jj4   ; @ SE(A) @
    rr     = 1/(1-dd) ;  @ GAMMA @
    s0     = sqrt(se^2/(1-dd^2))  ; @ SE(Y0) @

@ Printing out the parameter values in the output file @
    "" ;
    "============================" ;
    "VALUES OF MAJOR PARAMETERS"   ;
    "SEED      =   " ddd     ;
    "ITER      =   " iter    ;
    "T         =   " tt      ;
    "N         =   " nn      ;
    "DELTA     =   " dd      ;
    "SIGA      =   " sa      ;
    "SIGE      =   " se      ;
    "GAMMA     =   " rr      ;
    "SIG0      =   " s0      ;

@ Vectors to store estimators from each iterations @


    @  ARELLANO-BOND GMM @
    abb = zeros(iter,1) ; @ AB ESTIMATE @
    abv = zeros(iter,1) ; @ AB VARIANCE @
    abss= 0             ; @ SIZE OF T-TEST @

    @  ARELLANO-BOVER GMM @
    abob = zeros(iter,1) ; @ ABo ESTIMATE @
    abov = zeros(iter,1) ; @ ABo VARIANCE @
    abos = 0             ; @ SIZE OF T-TEST @


    @  AHN-SCHMIDT GMM1 @
    as1b = zeros(iter,1) ; @ AS1 ESTIMATE @
    as1v = zeros(iter,1) ; @ AS1 VARIANCE @
    as1s = 0             ; @ SIZE OF T-TEST @
    as1s1= 0             ; @ SIZE OF AHN-TEST @

    @  AHN-SCHMIDT GMM1 WITH AS1 MOMENT CONDITIONS ONLY @
    as1ob = zeros(iter,1) ; @ AS1 ESTIMATE @
    as1ov = zeros(iter,1) ; @ AS1 VARIANCE @
    as1os = 0             ; @ SIZE OF T-TEST @
    as1os1= 0             ; @ SIZE OF AHN-TEST @
```

```
/*******************
** BEGING THE LOOP **
*******************/

i = 1;
do while i <= iter ;

/***********************
** CREATING VARIABLES **
***********************/

@ UPDATING SEED FOR EACH ITERATION @
    ss1  = (ddd+i) ;

@ CREATING INDIVIDUAL EFFECTS  @
    aaa  = sa*rndns(nn,1,ss1) ;

@ CREATING DATA: YY INCLUDES Y0,Y1,...,YT @
    yy = rr*aaa + s0*rndns(nn,1,ss1)  ;
    j  = 1  ; do while j <= tt ;
        yy = yy~(dd*yy[.,j]+aaa+se*rndns(nn,1,ss1)) ;
    j  = j+1; endo ;

@ CREATING DIFFERENCED DATA: DYY INCLUDES (Y0-Y1),...,(YT-1-YT) @
    dyy = yy[.,2]-yy[.,1] ;
    j  = 2  ; do while j <= tt ;
        dyy = dyy~(yy[.,j+1]-yy[.,j]) ;
    j  = j+1; endo ;


@ STACKING DATA @
    depy  = vec(yy[.,2:tt+1]')   ; @ Y @
    lagy  = vec(yy[.,1:tt]')     ; @ Y_{-1} @
    pdepy = meanc(yy[.,2:tt+1]') .*. ones(tt,1) ; @ P_v Y @
    plagy = meanc(yy[.,1:tt]' ) .*. ones(tt,1) ;  @ P_v Y_{-1} @
    qdepy = depy - pdepy ;  @ Q_v Y @
    qlagy = lagy - plagy ;  @ Q_v Y_{-1} @
    ddepy = vec(dyy[.,2:tt]');
    dlagy = vec(dyy[.,1:tt-1]');
    abwdy = ddepy ;
    abwly = vec(yy[.,2:tt]') ;
    abwdly= dlagy            ;

@ CREATING ARELLANO AND BOND, ARELLANO-BOVER IVS @

@ CREATING L(T,P) MATRIX: NEEDED TO CREATE AB IVS @
    proc ll(t,p) ;
    local i,u    ;

        u = zeros(t,p) ;
        i = 1 ; do while i <= p ;
            u[t-p+i-1,i] = - 1  ; u[t-p+i,i] = 1  ;
        i = i + 1 ; endo        ;

    retp(u) ; endp ;
```

```
@ CREATING AHLL(T,P) MATRIX: NEEDED TO CREATE DIAGONAL ANDERSON-HSIAO IVS @
    proc ahll(t,p) ;
    local i,u ;

        u = zeros(t,1) ;
        u[p+2] = - 1  ; u[p+1] = 1  ;

    retp(u) ; endp ;

@ CREATING AB INSTRUMENTS @
    abiv = zeros(tt*nn,1) ;
    j = 1     ; do while j <= tt-1 ;
        abiv = abiv~( yy[.,j] .*. ll(tt,tt-j) ) ;
    j = j + 1 ; endo ;

    pp   = tt*(tt-1)/2    ;
    abiv = abiv[.,2:pp+1] ;

@ CREATING SUB-AB (Diagonal form of Andersen-Hsiao) INSTRUMENTS @
    ahiv = zeros(tt*nn,1) ;
    j = 1     ; do while j <= tt-1 ;
        ahiv = ahiv~( yy[.,j] .*. ahll(tt,j-1) ) ;
    j = j + 1 ; endo ;

    ahpp=tt-1 ;
    ahiv = ahiv[.,2:ahpp+1]  ;

@ CREATING ABo INSTRUMENTS @
    aboiv = zeros(tt*nn,1) ;
    j = 1     ; do while j <= tt-1 ;
      mmmm = zeros(tt,1) ;
      mmmm[j+1] = 1      ;
      aboiv = aboiv~( dyy[.,j] .*. mmmm ) ;
    j = j + 1 ; endo ;

    aboiv = aboiv[.,2:cols(aboiv)]  ;

@ CREATING AB-Type ABo (AABo) INSTRUMENTS @
    aaboiv = zeros(tt*nn,1) ;
    j = 1     ; do while j <= tt-1 ;
      mmmm = zeros(tt,1) ;
      mmmm[j+1] = 1      ;
      aaboiv = aaboiv~( dyy[.,1:j] .*. mmmm ) ;
    j = j + 1 ; endo ;

    aaboiv = aaboiv[.,2:cols(aaboiv)] ;

@ CREATING DIAGONAL LAGGED Y @
    diagly = zeros(tt*nn,1) ;
    j = 1 ; do while j <= tt ;
        mmmm = zeros(tt,1) ;
        mmmm[j] = 1        ;
        diagly = diagly~(yy[.,j] .*. mmmm) ;
    j = j + 1 ; endo  ;
        g1pp   = cols(diagly) ;
        diagly = diagly[.,2:g1pp] ;
```

```
@ # OF AB INSTRUMENTS @
    nabiv  = pp ;

@ # OF AH INSTRUMENTS @
    nahiv  = cols(ahiv) ;

/**********************
** ARELLANO-BOND GMM **
**********************/

@ TSLS ESTIMATOR @

    tscov = invpd( (lagy'abiv)*invpd(abiv'abiv)*(abiv'lagy) ) ;
    tsb   = tscov*(lagy'abiv)*invpd(abiv'abiv)*(abiv'depy)    ;
    qres  = qdepy - qlagy*tsb                                 ;
    tssige2 = (qres'qres/(nn*tt-tt-1)) ;
    tscov   = tssige2*tscov           ;

res  = depy - lagy*tsb  ;

opivab = zeros(1,cols(abiv)) ;
j = 0 ; do while j <= nn-1 ;
    zz = res[tt*j+1:tt*j+tt,.]'abiv[tt*j+1:tt*j+tt,.] ;
    opivab = opivab|zz ;
j = j + 1 ; endo ; opivab = opivab[2:nn+1,.] ;

optwab = invpd(opivab'opivab)    ;


VARAB1 = INVPD( (LAGY'ABIV)*(OPTWAB)*(ABIV'LAGY) ) ;
ABB1   = VARAB1*(LAGY'ABIV)*(OPTWAB)*(ABIV'DEPY)   ;

abv[I,1]  = VARAB1 ;
abb[I,1]  = ABB1   ;
wald      = (ABB1-dd)^2/VARAB1 ;
pval      = cdfchic(wald,1) ;

@ Computing rejection rate  @

    if pval < 0.05 ; rej05 = 1 ;
    else           ; rej05 = 0 ;
    endif          ;

abss = rej05 + abss ;

/**********************
** ARELLANO-BOVER GMM **
**********************/

iv = abiv~aboiv ;

@ TSLS ESTIMATOR @

    tscov = invpd( (lagy'iv)*invpd(iv'iv)*(iv'lagy) ) ;
    tsb   = tscov*(lagy'iv)*invpd(iv'iv)*(iv'depy)    ;
    qres  = qdepy - qlagy*tsb                         ;
    tssige2 = (qres'qres/(nn*tt-tt-1)) ;
    tscov   = tssige2*tscov           ;
```

```
res  = depy - lagy*tsb  ;

opivab = zeros(1,cols(iv)) ;
j = 0 ; do while j <= nn-1 ;
    zz = res[tt*j+1:tt*j+tt,.]'iv[tt*j+1:tt*j+tt,.] ;
    opivab = opivab|zz ;
j = j + 1 ; endo ; opivab = opivab[2:nn+1,.] ;

optwab = invpd(opivab'opivab)    ;


abov1 = INVPD( (LAGY'IV)*(OPTWAB)*(IV'LAGY) ) ;
abob1 = abov1*(LAGY'IV)*(OPTWAB)*(IV'DEPY)    ;

abov[I,1]  = abov1 ;
abob[I,1]  = abob1 ;
wald      = (abob1-dd)^2/abov1 ;
pval      = cdfchic(wald,1) ;

@ Computing rejection rate  @

    if pval < 0.05 ; rej05 = 1 ;
    else           ; rej05 = 0 ;
    endif          ;

abos = rej05 + abos ;


/*********************
** AHN-SCHMIDT GMM1 **
*********************/

/*
**  First step GMM
*/

@ define some global variables for gmm1 @

    WELL     = LL(TT,TT-1)                ;
    well     = well[1:tt,1:tt-2]          ;
    m11  = abiv'depy ;
    m12  = abiv'lagy ;
    m21  = (yy[.,tt+1].*.well)'depy ;
    m22  = (yy[.,tt].*.well)'depy + (yy[.,tt+1].*.well)'lagy  ;
    m23  = (yy[.,tt].*.well)'lagy ;

library optmum;
#include optmum.ext;
optset ;

@ Finding initial value @
```

```
@ step 1 for gmm1 @

    m11  = abiv'depy ;
    m12  = abiv'lagy ;
    m21  = (yy[.,tt+1].*.well)'depy ;
    m22  = (yy[.,tt].*.well)'depy + (yy[.,tt+1].*.well)'lagy  ;
    m23  = (yy[.,tt].*.well)'lagy ;
    wei1 = invpd((abiv'abiv)/nn)~zeros(cols(abiv),tt-2) ;
    wei2 = zeros(tt-2,cols(abiv))~eye(tt-2) ;
    wei  = wei1|wei2 ;


                          /* GRADIENT */

proc grad1(b) ;
local gra, m  ;

    m    = (m11-m12*b)|(m21-m22*b+m23*b^2) ;
    gra  = m12|(m22-2*m23*b) ;

retp( - 2*m'wei*gra
    ) ;
endp  ;

                                /* GMM */

proc  f1(b) ;
local m    ;

    m    = (m11-m12*b)|(m21-m22*b+m23*b^2) ;

retp(  m'wei*m
    ) ;
ENDP  ;

_opgdprc = &grad1 ;

@ starting values @

b0 = -1 ;

__title = "GMM11";

_opgtol = 1e-4 ;
_opstmth = "bfgs brent";
__output = 0 ;
{b,func,grad,retcode} = optmum(&f1,b0) ;

g = m12|(m22-2*m23*b) ;
v = invpd(g'g) ;
s = (b-dd)^2/v     ;
bb1 = func~b~v~s      ;

@ step 2 @
```

```
                                /* GMM */

proc  f2(b) ;
local m    ;

    m    = (m11-m12*b)|(m21-m22*b+m23*b^2) ;

retp(  m'wei*m
    ) ;
ENDP  ;

_opgdprc = &grad1 ;


@ starting values @

b0 = 1  ;

__title = "GMM12";

_opgtol = 1e-4 ;
_opstmth = "bfgs brent";
__output = 0 ;
{b,func,grad,retcode} = optmum(&f2,b0) ;

g = m12|(m22-2*m23*b) ;
v = invpd(g'g) ;
s = (b-dd)^2/v     ;
bb2 = func~b~v~s  ;

bb = bb1|bb2 ;
bb = sorthc(bb,1) ;

tsbb = bb[1,2] ;

@ define some global variables for gmm1 @

REST     = YY[.,TT+1] - YY[.,TT]*tsbb ;
IVB1     = REST .*. WELL              ;
RES      = DEPY - LAGY*tsbb           ;
AS1IV    = ABIV~IVB1           ;

opivas1 = zeros(1,cols(as1iv)) ;
j = 0 ; do while j <= nn-1 ;
    zz = res[tt*j+1:tt*j+tt,.]'as1iv[tt*j+1:tt*j+tt,.] ;
    opivas1 = opivas1|zz ;
j = j + 1 ; endo ; opivas1 = opivas1[2:nn,.] ;

opwas1   = invpd(opivas1'opivas1) ;


@ step 1 for gmm1 @
```

```
                          /* GRADIENT */

proc grad2(b) ;
local gra, m  ;

    m    = (m11-m12*b)|(m21-m22*b+m23*b^2) ;
    gra  = m12|(m22-2*m23*b) ;

retp( - 2*m'opwas1*gra
    ) ;
endp  ;

                                /* GMM */

proc  f3(b) ;
local m    ;

    m    = (m11-m12*b)|(m21-m22*b+m23*b^2) ;

retp(  m'opwas1*m
    ) ;
ENDP  ;

_opgdprc = &grad2 ;

@ starting values @

b0 = -1 ;

__title = "GMM11";

_opgtol = 1e-4 ;
_opstmth = "bfgs brent";
__output = 0 ;
{b,func,grad,retcode} = optmum(&f3,b0) ;

g = m12|(m22-2*m23*b) ;
v = invpd(g'opwas1*g) ;
s = (b-dd)^2/v     ;
bb1 = func~b~v~s      ;

@ step 2 @
```

```
                                /* GMM */

proc  f4(b) ;
local m    ;

    m    = (m11-m12*b)|(m21-m22*b+m23*b^2) ;

retp(  m'opwas1*m
    ) ;
ENDP  ;

_opgdprc = &grad2 ;

@ starting values @

b0 = 1  ;

__title = "GMM12";

_opgtol = 1e-4 ;
_opstmth = "bfgs brent";
__output = 0 ;
{b,func,grad,retcode} = optmum(&f4,b0) ;

g = m12|(m22-2*m23*b) ;
v = invpd(g'opwas1*g) ;
s = (b-dd)^2/v    ;
bb2 = func~b~v~s  ;

bb = bb1|bb2 ;
bb = sorthc(bb,1) ;

as1b[i,1] = bb[1,2] ;
as1v[i,1] = bb[1,3] ;

  wald = bb[1,4] ;
  pval = cdfchic(wald,1) ;

@ Computing rejection rate  @

    if pval < 0.05 ; rej05 = 1 ;
    else           ; rej05 = 0 ;
    endif          ;

as1s  = as1s + rej05 ;

  wald = f3(dd) - bb[1,1] ;
  pval = cdfchic(wald,1) ;
```

```
@ Computing rejection rate  @

    if pval < 0.05 ; rej05 = 1 ;
    else           ; rej05 = 0 ;
    endif          ;

as1s1 = as1s1 + rej05 ;

    i = i + 1 ; endo ;
    i = i - 1 ;

"" ;
"AB GMM RESULTS:" ;
"    MEAN OF ESTIMATES         :"  meanc(abb) ;
"    MSE                       :"  meanc((abb-dd)^2) ;
"    MEAN OF ASYM. VAR         :"  meanc(abv) ;
"    SIZE OF T-TEST            :"  abss/iter  ;
"";
"ABo GMM RESULTS:" ;
"    MEAN OF ESTIMATES         :"  meanc(abob) ;
"    MSE                       :"  meanc((abob-dd)^2) ;
"    MEAN OF ASYM. VAR         :"  meanc(abov) ;
"    SIZE OF T-TEST            :"  abos/iter   ;
"";
"";
"AS1 GMM RESULTS:" ;
"    MEAN OF ESTIMATES         :"  meanc(as1b) ;
"    MSE                       :"  meanc((as1b-dd)^2) ;
"    MEAN OF ASYM. VAR         :"  meanc(as1v) ;
"    SIZE OF T-TEST            :"  as1s/iter   ;
"    SIZE OF AHN TEST          :"  as1s1/iter   ;

j4 = j4 + 1 ; endo ;
j3 = j3 + 1 ; endo ;
j2 = j2 + 1 ; endo ;
j1 = j1 + 1 ; endo ;

OUTPUT OFF
```

Outcome in dynamic1.out

```
 MODEL SPECIFICATION:
       y_t = d*y_{t-1} + a + e_t ;
       y_0 = gamma*a + e_0
       Stationary y:  gamma = 1/(1-d)

============================
VALUES OF MAJOR PARAMETERS
SEED      =      1.000
ITER      =    100.000
T         =      4.000
N         =    100.000
DELTA     =      0.900
SIGA      =      1.000
SIGE      =      1.000
GAMMA     =     10.000
SIG0      =      2.294

AB GMM RESULTS:
    MEAN OF ESTIMATES         :   0.360
    MSE                       :   0.517
    MEAN OF ASYM. VAR         :   0.185
    SIZE OF T-TEST            :   0.340

ABo GMM RESULTS:
    MEAN OF ESTIMATES         :   0.947
    MSE                       :   0.011
    MEAN OF ASYM. VAR         :   0.003
    SIZE OF T-TEST            :   0.590


AS1 GMM RESULTS:
    MEAN OF ESTIMATES         :   0.964
    MSE                       :   0.036
    MEAN OF ASYM. VAR         :   0.010
    SIZE OF T-TEST            :   0.560
    SIZE OF AHN TEST          :   0.530
```

```
============================
VALUES OF MAJOR PARAMETERS
SEED      =      1.000
ITER      =    100.000
T         =      4.000
N         =    100.000
DELTA     =      0.800
SIGA      =      1.000
SIGE      =      1.000
GAMMA     =      5.000
SIG0      =      1.667

AB GMM RESULTS:
    MEAN OF ESTIMATES         :   0.555
    MSE                       :   0.191
    MEAN OF ASYM. VAR         :   0.126
    SIZE OF T-TEST            :   0.210

ABo GMM RESULTS:
    MEAN OF ESTIMATES         :   0.833
    MSE                       :   0.016
    MEAN OF ASYM. VAR         :   0.007
    SIZE OF T-TEST            :   0.400


AS1 GMM RESULTS:
    MEAN OF ESTIMATES         :   0.892
    MSE                       :   0.055
    MEAN OF ASYM. VAR         :   0.016
    SIZE OF T-TEST            :   0.500
    SIZE OF AHN TEST          :   0.430
```

```
============================
VALUES OF MAJOR PARAMETERS
SEED      =      1.000
ITER      =    100.000
T         =      4.000
N         =    100.000
DELTA     =      0.500
SIGA      =      1.000
SIGE      =      1.000
GAMMA     =      2.000
SIG0      =      1.155

AB GMM RESULTS:
    MEAN OF ESTIMATES         :   0.483
    MSE                       :   0.036
    MEAN OF ASYM. VAR         :   0.026
    SIZE OF T-TEST            :   0.130

ABo GMM RESULTS:
    MEAN OF ESTIMATES         :   0.522
    MSE                       :   0.014
    MEAN OF ASYM. VAR         :   0.006
    SIZE OF T-TEST            :   0.240


AS1 GMM RESULTS:
    MEAN OF ESTIMATES         :   0.535
    MSE                       :   0.035
    MEAN OF ASYM. VAR         :   0.011
    SIZE OF T-TEST            :   0.210
    SIZE OF AHN TEST          :   0.200
```

Outcome in dynamic2.out

```
 MODEL SPECIFICATION:
       y_t = d*y_{t-1} + a + e_t ;
       y_0 = gamma*a + e_0
       Non-Stationary y:  gamma = 1/(1-d)+1

============================
VALUES OF MAJOR PARAMETERS
SEED             =      1.000
ITER             =    100.000
T                =      4.000
N                =    100.000
DELTA            =      0.900
SIGA             =      1.000
SIGE             =      1.000
GAMMA(1/(1-d)+1) =     11.000
SIG0             =      2.294

AB GMM RESULTS:
    MEAN OF ESTIMATES         :   0.677
    MSE                       :   0.172
    MEAN OF ASYM. VAR         :   0.110
    SIZE OF T-TEST            :   0.200

ABo GMM RESULTS:
    MEAN OF ESTIMATES         :   1.002
    MSE                       :   0.015
    MEAN OF ASYM. VAR         :   0.002
    SIZE OF T-TEST            :   0.720


AS1 GMM RESULTS:
    MEAN OF ESTIMATES         :   0.956
    MSE                       :   0.033
    MEAN OF ASYM. VAR         :   0.004
    SIZE OF T-TEST            :   0.530
    SIZE OF AHN TEST          :   0.510
```

```
============================
VALUES OF MAJOR PARAMETERS
SEED              =      1.000
ITER              =    100.000
T                 =      4.000
N                 =    100.000
DELTA             =      0.800
SIGA              =      1.000
SIGE              =      1.000
GAMMA (1/(1-d)+1) =      6.000
SIG0              =      1.667

AB GMM RESULTS:
    MEAN OF ESTIMATES         :   0.776
    MSE                       :   0.053
    MEAN OF ASYM. VAR         :   0.035
    SIZE OF T-TEST            :   0.170

ABo GMM RESULTS:
    MEAN OF ESTIMATES         :   1.007
    MSE                       :   0.049
    MEAN OF ASYM. VAR         :   0.005
    SIZE OF T-TEST            :   0.820


AS1 GMM RESULTS:
    MEAN OF ESTIMATES         :   0.874
    MSE                       :   0.031
    MEAN OF ASYM. VAR         :   0.009
    SIZE OF T-TEST            :   0.400
    SIZE OF AHN TEST          :   0.380
```

```
============================
VALUES OF MAJOR PARAMETERS
SEED              =      1.000
ITER              =    100.000
T                 =      4.000
N                 =    100.000
DELTA             =      0.500
SIGA              =      1.000
SIGE              =      1.000
GAMMA (1/(1-d)+1) =      3.000
SIG0              =      1.155

AB GMM RESULTS:
    MEAN OF ESTIMATES         :   0.505
    MSE                       :   0.013
    MEAN OF ASYM. VAR         :   0.008
    SIZE OF T-TEST            :   0.130

ABo GMM RESULTS:
    MEAN OF ESTIMATES         :   0.895
    MSE                       :   0.163
    MEAN OF ASYM. VAR         :   0.003
    SIZE OF T-TEST            :   1.000


AS1 GMM RESULTS:
    MEAN OF ESTIMATES         :   0.519
    MSE                       :   0.013
    MEAN OF ASYM. VAR         :   0.006
    SIZE OF T-TEST            :   0.200
    SIZE OF AHN TEST          :   0.180
```