Munich Personal RePEc Archive

R-Codes to Calculate GMM Estimations for Dynamic Panel Data Models

Abonazel, Mohamed R.

Department of Applied Statistics and Econometrics, Institute of Statistical Studies and Research, Cairo University, Egypt

December 2015

Online at https://mpra.ub.uni-muenchen.de/70627/
MPRA Paper No. 70627, posted 11 Apr 2016 05:22 UTC

R-Codes to Calculate GMM Estimations for Dynamic Panel Data Models*

Mohamed Reda Abonazel
Department of Applied Statistics and Econometrics, Institute of Statistical Studies and Research, Cairo University, Egypt
mabonazel@hotmail.com
2015

Abstract

These codes presented three functions for calculating three important estimators in dynamic panel data (DPD) models; these estimators are Arellano-Bond (1991), Arellano-Bover (1995), and Blundell-Bond (1998). All functions here need to the following variables: yit_1: dependent variable for DPD model; phi: the value of autoregressive coefficient; D.T_D.T: first-difference operator matrix of Arellano-Bond estimator; HD: instrumental variables of Arellano-Bond estimator; HL: instrumental variables of Arellano-Bover estimator; W: weighting matrix of Blundell-Bond estimator; HS: instrumental variables of Blundell-Bond estimator. Also, they need to the following R libraries: simex; plm; dlm. For more details about the theoretical bases and the developments of that estimators, see, e.g., Youssef *et al*. (2014a,b) and Youssef and Abonazel (2015). Moreover, these codes have been designed to enable the user to make a simulation study in this topic, such as the simulation study in Youssef et al. (2014b).

Keywords: Dynamic panel data models; Generalized method of moments (GMM); Monte Carlo simulation; Two-step GMM estimations.

1. R-Code to Calculate Arellano-Bond Estimator

```
Arellano.Bond<- function (yit_1, phi , D.T_D.T,  HD)
{
N<-ncol(yit_1)
T<-nrow(yit_1)
```

* These codes have been recorded in Appendix C in my Ph.D. thesis (Abonazel, 2014).

```
#### calculate Δy
delta_y<- apply(yit_1 [(2:T),],2,diff) ; dim(delta_y)<-c((T-2)*N,1)
delta_y_1<- apply(yit_1 [(1:T-1),],2,diff) ; dim(delta_y_1)<-c((T-2)*N,1)

####    calculate D using my function
D<- kronecker (diag(1,N), D.T_D.T   )

####    calculate A
AD1<- t(HD) %*%  D  %*% HD
####    calculate φ̂  step (1)
part1<- ginv (t(delta_y_1) %*% HD %*% ginv(AD1) %*% t(HD) %*% delta_y_1)
part2<- t(delta_y_1) %*% HD %*% ginv(AD1) %*% t(HD) %*% delta_y
phi.hat.step1<- part1 %*% part2
dim(phi.hat.step1)<- NULL
Bias.ABond.step1 <- phi.hat.step1 - phi
Bias.Square.phi.hat.step1<- (Bias.ABond.step1 ^2)
SE.phi.hat.step1<-  sqrt (diag(part1))
 phi.hat.Square.step1<- (phi.hat.step1^2)

#### step 2
 #### calculate φ̂   step (2)
        #### calculate residual u.hat
delta_y<- apply(yit_1 [(2:T),],2,diff)
delta_y_1<- apply(yit_1 [(1:T-1),],2,diff)
delta_yit.hat <- phi.hat.step1* delta_y_1

#### calculate Δ u.hat
delta_u.hat.i  <- delta_y- delta_yit.hat

####    calculate  sigma.2.epsilon.hat   for weight BB estimator
sum_delta_u.hat <-   sum(delta_u.hat.i^2)
sigma.2.epsilon.hat <-   sum_delta_u.hat/(2*N*(T-2))
f<-matrix (0,nrow= N+1,ncol=1)
for (j in 1 : N)     f[j+1,1]<- j* (T-2)
A_sum<-0
for (j in 1 : N){
A_sum<-  A_sum + ( t(HD[(f[j,1]+1)  :  f[j+1,1],])    %*%  delta_u.hat.i[,j] %*% t(
delta_u.hat.i[,j]) %*% (HD[(f[j,1]+1)  :  f[j+1,1],])    )
 }
        AD2<- A_sum

#### calculate  φ̂  step 2
dim(delta_y)<-c((T-2)*N,1)
dim(delta_y_1)<-c((T-2)*N,1)
part1_step2<- ginv (t(delta_y_1) %*% HD %*% ginv(AD2) %*% t(HD) %*% delta_y_1)
part2_step2<- t(delta_y_1) %*% HD %*% ginv(AD2) %*% t(HD) %*% delta_y
phi.hat.step2<- part1_step2 %*% part2_step2
dim(phi.hat.step2)<- NULL
Bias.ABond.step2 <- phi.hat.step2- phi
Bias.Square.phi.hat.step2<- Bias.ABond.step2 ^2
```

```
SE.phi.hat.step2<-  sqrt (diag(part1_step2))
phi.hat.Square.step2<- phi.hat.step2^2

##### the function results
values_step2  <- list(phi.hat.step2=phi.hat.step2,   Bias.ABond.step2= Bias.ABond.step2,
Bias.Square.phi.hat.step2= Bias.Square.phi.hat.step2,
SE.phi.hat.step2     =  SE.phi.hat.step2, phi.hat.Square.step2= phi.hat.Square.step2,
sigma.2.epsilon.hat = sigma.2.epsilon.hat)
values_step1  <- list(phi.hat.step1=phi.hat.step1,   Bias.ABond.step1= Bias.ABond.step1,
Bias.Square.phi.hat.step1= Bias.Square.phi.hat.step1,
SE.phi.hat.step1     =  SE.phi.hat.step1, phi.hat.Square.step1= phi.hat.Square.step1 )
result<-list(values_step1=values_step1,values_step2=values_step2)
return(result) }
```

2. R-Code to Calculate Arellano-Bover Estimator

```
Arellano.Bover <- function (yit_1, phi , HL){
N<-ncol(yit_1)
T<-nrow(yit_1)
delta_y_1<- apply(yit_1 [(1:T-1),],2,diff)

#### calculate  y_1 and y
 y_1<-  yit_1 [(2:(T-1)),]  ; dim(y_1)<-c((T-2)*N,1)
y<- yit_1 [(3:T),]     ; dim(y)<-c((T-2)*N,1)

####    calculate  AL1
AL1<- t(HL) %*% HL

####    calculate φ̂  step (1)
part1_Bover_step1<- ginv (t(y_1) %*% HL %*% ginv(AL1) %*% t(HL) %*% y_1)
part2_Bover_step1<- t(y_1) %*% HL %*% ginv(AL1) %*% t(HL) %*% y
phi.hat.Bover.step1<- part1_Bover_step1%*% part2_Bover_step1
dim(phi.hat.Bover.step1)<- NULL
Bias.ABover.step1 <- phi.hat.Bover.step1  - phi
Bias.Square.phi.hat.Bover.step1<- Bias.ABover.step1^2
SE.phi.hat.Bover.step1<-  sqrt (diag(part1_Bover_step1))
phi.hat.Square.Bover.step1<- phi.hat.Bover.step1^2

####  step 2
#### calculate φ̂   step (2)
 y_1<-  yit_1 [(2:(T-1)),]
y<- yit_1 [(3:T),]
yit.hat_Bover <- phi.hat.Bover.step1 * y_1
u.hat.i_Bover  <- y - yit.hat_Bover

#### calculate  AL2
f<-matrix (0,nrow= N+1,ncol=1)
for (j in 1 : N)     f[j+1,1]<- j* (T-2)
AL2<-0       ;   for (j in 1 : N){
```

```
AL2<-  AL2 +    ( t(HL[(f[j,1]+1)  :  f[j+1,1],])    %*%  u.hat.i_Bover [,j] %*% t(
u.hat.i_Bover [,j]) %*% (HL[(f[j,1]+1)  :  f[j+1,1],])    )    }

#### calculate  φ̂  step (2)
dim(y_1)<-c((T-2)*N,1)
dim(y)<-c((T-2)*N,1)
part1_Bover_step2<- ginv (t(y_1) %*% HL %*% ginv(AL2) %*% t(HL) %*% y_1)
part2_Bover_step2<- t(y_1) %*% HL %*% ginv(AL2) %*% t(HL) %*% y
phi.hat.Bover.step2<- part1_Bover_step2%*% part2_Bover_step2
dim(phi.hat.Bover.step2)<- NULL
Bias.ABover.step2 <-   phi.hat.Bover.step2 - phi
Bias.Square.phi.hat.Bover.step2<- Bias.ABover.step2^2
SE.phi.hat.Bover.step2<-  sqrt (diag(part1_Bover_step2))
phi.hat.Square.Bover.step2<- phi.hat.Bover.step2^2

#### the function results
values_step2 <- list(phi.hat.Bover.step2= phi.hat.Bover.step2,   Bias.ABover.step2 =
Bias.ABover.step2,  Bias.Square.phi.hat.Bover.step2= Bias.Square.phi.hat.Bover.step2,
SE.phi.hat.Bover.step2=  SE.phi.hat.Bover.step2, phi.hat.Square.Bover.step2=
phi.hat.Square.Bover.step2)
values_step1 <- list(phi.hat.Bover.step1= phi.hat.Bover.step1,   Bias.ABover.step1 =
Bias.ABover.step1,  Bias.Square.phi.hat.Bover.step1= Bias.Square.phi.hat.Bover.step1,
SE.phi.hat.Bover.step1=  SE.phi.hat.Bover.step1, phi.hat.Square.Bover.step1=
phi.hat.Square.Bover.step1)
result<-list(values_step1=values_step1,values_step2=values_step2)
return(result)}
```

3. R-Code to Calculate Blundell-Bond Estimator

```
BB <- function (yit_1, phi , W= "G" , D.T_D.T, HD,HL,HS){
N<-ncol(yit_1)
T<-nrow(yit_1)

#### calculate  y_s and y_s_1
y<- yit_1 [(3:T),]
y_1<-  yit_1 [(2:(T-1)),]
delta_y_1<- apply(yit_1 [(1:T-1),],2,diff)
delta_y<- apply(yit_1 [(2:T),],2,diff) ;
y_s <- rbind(delta_y,y)   ; dim(y_s)<-c( 2*(T-2)*N,1)
y_s_1<- rbind(delta_y_1,y_1)    ; dim(y_s_1)<-c(2*(T-2)*N,1)

#### calculate G
 G_T<- bdiag (D.T_D.T  , diag(1,T-2))
G<-  kronecker(diag(1,N), G_T)

#### calculate AS1
AS1<- t(HS) %*% G%*%  HS

#### Case of I
if (W=="I") AS1<- t(HS) %*%   HS
```

```
####calculate φ̂  step (1)
part1_BB_step1<- ginv (t(y_s_1) %*% HS  %*% ginv(AS1) %*% t(HS) %*% y_s_1)
part2_BB_step1<- t(y_s_1) %*% HS  %*% ginv(AS1) %*% t(HS) %*% y_s
phi.hat.BB.step1<- part1_BB_step1%*% part2_BB_step1
dim(phi.hat.BB.step1)<- NULL
Bias.BB.step1 <-   phi.hat.BB.step1 - phi
Bias.Square.phi.hat.BB.step1<- Bias.BB.step1^2


#### step 2
#### calculate φ̂   step (2)
          #### calculate residual u.hat.i_S
y_s <- rbind(delta_y,y)
y_s_1<- rbind(delta_y_1,y_1)
 yit.hat_BB <- phi.hat.BB.step1 * y_s_1
u.hat.i_S<- y_s - yit.hat_BB

 ####calculate  sigma.2.mu.hat for weight BB estimator
delta_u.hat.i_BB  <- u.hat.i_S[1:(T-2),]
u.hat.i_BB  <- u.hat.i_S[(T-1):  (2*(T-2)),]
sum_UU<-0
 for (i in 1:N) {
sum_UU <- sum_UU +    sum(u.hat.i_BB  [,i]^2) - (sum( delta_u.hat.i_BB  [,i]^2)/2)  }
sigma.2.mu.hat<-  sum_UU/(N*(T-2))
f<-matrix (0,nrow= N+1,ncol=1)
for (j in 1 : N)     f[j+1,1]<- j* (T-2)

#### calculate  AS2
hs_list<-list()   ; AS2<-0
  for (j in 1 : N){
hs_list [[1]]<- HD[(f[j,1]+1)  :  f[j+1,1],]
hs_list [[2]]<- HL[(f[j,1]+1)  :  f[j+1,1],]
HSi <-diag.block (hs_list)

AS2<-  AS2 +     ( t(HSi)    %*%  u.hat.i_S [,j] %*% t(  u.hat.i_S [,j]) %*%    HSi  )
}

####calculate φ̂  step (2)
dim(y_s_1)<-c(2*(T-2)*N,1)
dim(y_s)<-c( 2*(T-2)*N,1)
part1_BB_step2<- ginv (t(y_s_1) %*% HS  %*% ginv(AS2) %*% t(HS) %*% y_s_1)
part2_BB_step2<- t(y_s_1) %*% HS  %*% ginv(AS2) %*% t(HS) %*% y_s
phi.hat.BB.step2<- part1_BB_step2%*% part2_BB_step2
dim(phi.hat.BB.step2)<- NULL
Bias.BB.step2 <-   phi.hat.BB.step2 - phi
Bias.Square.phi.hat.BB.step2<- Bias.BB.step2^2

#### the function results
```

```
values_step2  <- list(phi.hat.BB.step2= phi.hat.BB.step2,   Bias.BB.step2 = Bias.BB.step2,
Bias.Square.phi.hat.BB.step2= Bias.Square.phi.hat.BB.step2,
sigma.2.mu.hat = sigma.2.mu.hat  )
values_step1  <- list(phi.hat.BB.step1= phi.hat.BB.step1,   Bias.BB.step1 = Bias.BB.step1,
Bias.Square.phi.hat.BB.step1= Bias.Square.phi.hat.BB.step1)
result<-list(values_step1=values_step1,values_step2=values_step2)
return(result)  }
```

References

Abonazel, M. R. (2014). Some estimation methods for dynamic panel data models. Ph.D. thesis. Institute of Statistical Studies and Research. Cairo University.

Arellano, M., Bond, S. (1991). Some tests of specification for panel data: Monte Carlo evidence and an application to employment equations. *Review of Economic Studies* 58:277–98.

Arellano, M., Bover, O. (1995). Another look at the instrumental variable estimation of error-components models. *Journal of Econometrics* 68:29-51.

Blundell, R., Bond, S. (1998). Initial conditions and moment restrictions in dynamic panel data models. *Journal of Econometrics* 87:115–143.

Youssef, A. H., Abonazel, M. R. (2015). Alternative GMM estimators for first-order autoregressive panel model: an improving efficiency approach. *Communications in Statistics-Simulation and Computation* (in press). DOI: 10.1080/03610918.2015.1073307.

Youssef, A., El-sheikh, A., Abonazel, M. (2014a). Improving the efficiency of GMM estimators for dynamic panel models. *Far East Journal of Theoretical Statistics* 47:171–189.

Youssef, A., El-sheikh, A., Abonazel, M. (2014b). New GMM estimators for dynamic panel data models. *International Journal of Innovative Research in Science, Engineering and Technology* 3: 16414–16425.