Test

Thomas Chuffart

20 septembre 2018

Estimation sans prendre en compte l'hétérogénéité

```
library(plm)
```

```
## Loading required package: Formula
```

```
load('panel_psid.RData')
summary(plm(log(wage) ~ educ  + experience , data = df, index = c("pid","year"),
            model = "pooling"))
```

```
## Pooling Model
##
## Call:
## plm(formula = log(wage) ~ educ + experience, data = df, model = "pooling",
##     index = c("pid", "year"))
##
## Balanced Panel: n = 2084, T = 4, N = 8336
##
## Residuals:
##       Min.    1st Qu.     Median    3rd Qu.       Max.
## -10.842813  -0.316410   0.077228   0.454271   4.745900
##
## Coefficients:
##              Estimate Std. Error t-value  Pr(>|t|)
## (Intercept) 8.75821402 0.06090874 143.792 < 2.2e-16 ***
## educ        0.12262350 0.00424355  28.896 < 2.2e-16 ***
## experience  0.02374722 0.00099522  23.861 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Total Sum of Squares:    6765.9
## Residual Sum of Squares: 5790.3
## R-Squared:      0.14419
## Adj. R-Squared: 0.14398
## F-statistic: 701.979 on 2 and 8333 DF, p-value: < 2.22e-16
```

Estimation en prenant en compte l’hétérogénéité

```
summary(plm(log(wage) ~ educ  + experience , data = df, index = c("pid","year"),
            model = "within"))
```

```
## Oneway (individual) effect Within Model
##
## Call:
## plm(formula = log(wage) ~ educ + experience, data = df, model = "within",
##     index = c("pid", "year"))
##
## Balanced Panel: n = 2084, T = 4, N = 8336
##
## Residuals:
##      Min.   1st Qu.    Median   3rd Qu.      Max.
## -6.792246 -0.117163  0.011723  0.149364  2.722799
##
## Coefficients:
##             Estimate Std. Error t-value  Pr(>|t|)
## educ       0.0397879  0.0092493  4.3017 1.721e-05 ***
## experience 0.0187873  0.0012910 14.5525 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Total Sum of Squares:    1502.8
## Residual Sum of Squares: 1448.3
## R-Squared:      0.036252
## Adj. R-Squared: -0.28525
## F-statistic: 117.55 on 2 and 6250 DF, p-value: < 2.22e-16
```

Biais dynamique

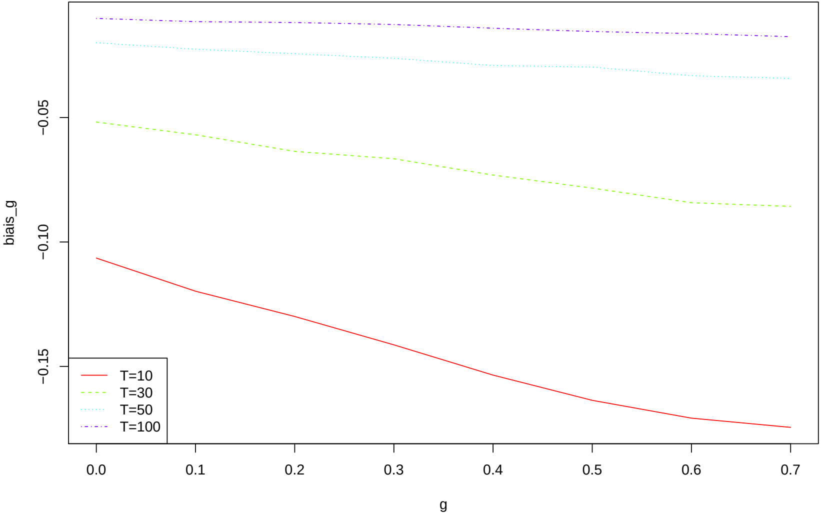
```r
# Fonction permettant de créer l'index temporel
get_year <- function(t,n){
  return(rep(1:t,n))
}

# Fonction permettant de créer l'index individuel
get_id <- function(t,n){
  id <- rep(0,(t*n))
  for (i in 1:n){
    id[(1+(t*(i-1))):(t*i)] <- rep(i,t)
  }
  return(id)
}

# Fonction simulant les données
coeff_lsdv_arsim <- function(t,n,g){
  alpha <- runif(n,-1,1)  # Simulation des paramètres non-observés
  y <- array(rep(0, (t+1)*n), dim=c(t+1, n)) # Initialisation de la variable dépendante
  e <- array(rnorm((t+1)*n), dim=c(t+1, n)) # Simulation des erreurs
  for (t in 2:(t+1)){ # On simule la variable expliqué
    y[t,] <- alpha + g*y[t-1,] + e[t,]
  }
  y0 <- y[2:t,] # y0 est la variable dépendante
  y1 <- y[1:(t-1),] # y1 est le lag de la variable dépendante
  y0 <- c(y0)
  y1 <- c(y1)
  df <- data.frame(id,year,y0,y1) # Construction du dataframe
  # Estimateur LSDV
  lsdv <- plm(y0 ~ y1, index = c("id","year"), data = df, model = "within")
  gam_hat <- lsdv$coefficients
  return(gam_hat)
}
```

Biais dynamique

```r
g = (0:7)/10
t = c(10,20,50,100)
R = 500 ## Nombre de réplications
biais_g <- matrix(0, nrow = length(g), ncol = length(t))
biais_gam_hat <- rep(NA,R)
for (l in 1:length(t)){
  for (k in 1:length(g)){
    G <- g[k] ## Paramètre autorégressif
    T <- t[l]  ## Nombre de périodes
    N <- 100 ## Nombre d'individus
    year <- get_year(T,N) ## Construction de l'index temporel
    id <- get_id(T,N)
    for (r in 1:R){ ## Boucle sur les réplications
      biais_gam_hat[r] <- coeff_lsdv_arsim(T,N,G) - G
    }
    biais_g[k,l] <- mean(biais_gam_hat)
  }
}
```

Biais Dynamique

```
col_set <- rainbow(4)
matplot(g,biais_g, type = 'l', col = col_set)
legend("bottomleft", c("T=10", "T=30","T=50", "T=100"), col
```


biais_g
g
T=10
T=30
T=50
T=100