Example 2-2

September 12, 2020

[ ]:
```r
# first install the following packages and library
install.packages("pder")
install.packages("plm")
library("plm")

# import the data, frame the data, and create the model Qeq
data("TobinQ", package = "pder")
pTobinQ <- pdata.frame(TobinQ)
Qeq <- ikn ~ qn
```

[7]:
```r
##---------------------------------Block 1---------------------------------

#### Example 2-2 ####

## ------------------------------------------------------------------------

# Q.swar is an estimate of the model developed by Swamy and Arora (1972).
# This is a random effects model and by setting the random.method option
# to "swar", will estimate the variances used in Swamy and Arora.
Q.swar <- plm(Qeq, pTobinQ, model = "random", random.method = "swar")

# using the random.models option is an alternative to the random.method.
# it is a character vector of length 1 or 2 that indicates which preliminary
# estimatons are preformed to estimate the variances.
# "within" uses the within residuals to estimate sigma_v
# "between" uses the between residuals to estimate sigma_l
# "pooling" can also be used to estimate both variances

# random.dfcor indicates the denominator of the 2 quadratic forms
# 0 is used when the # of observations used is (NT,N)
# 1 is used when the numerators of the theoretical formuals are used (N(T-1),N)
# 2 is used when the number of estimated parameters are deduced (N(T-1)-K,N-K-1)

Q.swar2 <- plm(Qeq, pTobinQ, model = "random",
               random.models = c("within", "between"),
               random.dfcor = c(2, 2))
summary(Q.swar)
```

```
Oneway (individual) effect Random Effect Model
   (Swamy-Arora's transformation)

Call:
plm(formula = Qeq, data = pTobinQ, model = "random", random.method = "swar")

Balanced Panel: n = 188, T = 35, N = 6580

Effects:
                   var  std.dev share
idiosyncratic 0.005333 0.073028 0.725
individual    0.002019 0.044930 0.275
theta: 0.7351

Residuals:
     Min.   1st Qu.    Median   3rd Qu.      Max.
-0.233027 -0.047509 -0.010278  0.033615  0.621113

Coefficients:
              Estimate Std. Error z-value  Pr(>|z|)
(Intercept) 0.15932695 0.00342490  46.520 < 2.2e-16 ***
qn          0.00386220 0.00016826  22.953 < 2.2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Total Sum of Squares:    37.904
Residual Sum of Squares: 35.093
R-Squared:      0.074154
Adj. R-Squared: 0.074013
Chisq: 526.854 on 1 DF, p-value: < 2.22e-16
```

[3]:
```
##---------------------------------Block 2---------------------------------

# ercomp() function is an alternative to estimating the error components.
# it can either be applied to the GLS fitted model or using a formula as an␣
 ↪input
ercomp(Qeq, pTobinQ)
ercomp(Q.swar)
```

```
                   var  std.dev share
idiosyncratic 0.005333 0.073028 0.725
individual    0.002019 0.044930 0.275
theta: 0.7351


                   var  std.dev share
idiosyncratic 0.005333 0.073028 0.725
```

```
individual    0.002019 0.044930 0.275
theta: 0.7351
```

[6]:
```r
##---------------------------------Block 3---------------------------------

# walhus, amemiya, and nerlove are options for the estimation of the variances␣
 ↪as done in
# Wallace and Hussain (1969), Amemiya (1971), and Nerlove (1971)
Q.walhus <- update(Q.swar, random.method = "swar")
Q.amemiya <- update(Q.swar, random.method = "amemiya")
Q.nerlove <- update(Q.swar, random.method = "nerlove")
Q.models <- list(swar = Q.swar, walhus = Q.walhus,
                 amemiya = Q.amemiya, nerlove = Q.nerlove)

# the sapply() comman d extracts the ercomp() object and the theta element,␣
 ↪which
# indicates the proportion of the individual mean that is removed from the␣
 ↪variables
# it aslo has the option of removing the coefficients for all the models
sapply(Q.models, function(x) ercomp(x)$theta)
sapply(Q.models, coef)
```

swar.id 0.735077121189867 **walhus.id** 0.735077121189867 **amemiya.id** 0.736118550549931 **nerlove.id** 0.748917664807631

	swar	walhus	amemiya	nerlove
(Intercept)	0.159326945	0.159326945	0.159328257	0.159344040
qn	0.003862202	0.003862202	0.003861678	0.003855378