hw3-YifanZheng-8

Yifan Zheng

3/4/2020

8, King and Roberts (2015) gave three examples where the EHW standard errors differ from the OLS standard error. I have replicated one example in Section 4.4. Replicate another one.

```r
library(lmtest)
```

```
## Loading required package: zoo
```

```
##
## Attaching package: 'zoo'
```

```
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```r
library(car)
```

```
## Loading required package: carData
```

```r
library(sandwich)
# load data
load("DreherandJensenJLEreplication.RData")

# check original data
head(x)
```

```
##    un_per_l country  year sum id  gdp_r_wk sum_ii sum_i pc sb pa t19974
## 1 0.4013605 Albania 19974  NA  1        NA     NA    NA NA NA NA      1
## 2 0.3971963 Albania 19981  NA  1 0.3281510     NA    NA NA NA NA      0
## 3 0.3971963 Albania 19982  NA  1 0.3311584     NA    NA NA NA NA      0
## 4 0.3953488 Albania 19983  NA  1 0.3672308     NA    NA NA NA NA      0
## 5 0.3935185 Albania 19984  30  1 0.4263837      7    10  6 19  5      0
## 6 0.3890909 Albania 19991  NA  1 0.4470208     NA    NA NA NA NA      0
##   t19981 t19982 t19983 t19984 t19991 t19992 t19993 t19994 t20001 t20002
## 1      0      0      0      0      0      0      0      0      0      0
## 2      1      0      0      0      0      0      0      0      0      0
## 3      0      1      0      0      0      0      0      0      0      0
## 4      0      0      1      0      0      0      0      0      0      0
## 5      0      0      0      1      0      0      0      0      0      0
## 6      0      0      0      0      1      0      0      0      0      0
##   t20003 t20004 t20011 t20012 t20013 t20014 t20021 t20022 t20023 t20024
## 1      0      0      0      0      0      0      0      0      0      0
## 2      0      0      0      0      0      0      0      0      0      0
## 3      0      0      0      0      0      0      0      0      0      0
## 4      0      0      0      0      0      0      0      0      0      0
## 5      0      0      0      0      0      0      0      0      0      0
## 6      0      0      0      0      0      0      0      0      0      0
##   t20031 t20032       gg_l   mg_l  def_gdp_l gge_gdp_l sum_ii_l sum_i_l
## 1      0      0         NA     NA         NA        NA       NA      NA
## 2      0      0         NA   1.40 -12.013600  11.44472       NA      NA
## 3      0      0   1.116944  -7.63 -10.912080  11.05133       NA      NA
## 4      0      0   5.594948 -20.27  -9.973802  10.71624       NA      NA
## 5      0      0  10.142180 -12.46  -9.164987  10.42739       NA      NA
## 6      0      0   4.595073  -8.66  -8.460567  10.17582        7      10
##   sum_l libor_l    res_c_l gg_oecd_l  cab_gdp_l imf_c_liab_l sum_tra
## 1    NA      NA         NA        NA         NA           NA      NA
## 2    NA    5.92  11.911620 0.7763006 -25.816320        22.56      NA
## 3    NA    5.67   2.991416 0.7030583  -8.225795        -1.21      NA
## 4    NA    5.79  27.128390 0.2538804  -7.838735        -1.22      NA
## 5    NA    5.60   6.201855 0.6152081   2.747653         0.00      30
## 6    30    5.28 -14.509710 0.9617339   1.751430         0.00      NA
##   sum_not_tra election_av un_el_av  cum_ca_l  cum_uk_l  cum_fr_l  cum_ge_l
## 1          NA           0        0 0.6802721 0.5986394 0.6054422 0.6530612
## 2          NA           0        0 0.6728972 0.5934579 0.5981308 0.6495327
## 3          NA           0        0 0.6728972 0.5934579 0.5981308 0.6495327
## 4          NA           0        0 0.6697674 0.5906976 0.5953488 0.6465116
## 5          NA           0        0 0.6666667 0.5879630 0.5925926 0.6435185
## 6          NA           0        0 0.6763636 0.5963637 0.5963637 0.6581818
##    cum_it_l  cum_ja_l ca_el_av uk_el_av fr_el_av ge_el_av it_el_av
## 1 0.6734694 0.6394558        0        0        0        0        0
## 2 0.6682243 0.6168224        0        0        0        0        0
## 3 0.6682243 0.6168224        0        0        0        0        0
## 4 0.6651162 0.6139535        0        0        0        0        0
## 5 0.6620370 0.6111111        0        0        0        0        0
## 6 0.6763636 0.6181818        0        0        0        0        0
##   ja_el_av
## 1        0
## 2        0
## 3        0
## 4        0
## 5        0
## 6        0
```

```r
colnames(x)
```

```
##  [1] "un_per_l"     "country"      "year"         "sum"         
##  [5] "id"           "gdp_r_wk"     "sum_ii"       "sum_i"       
##  [9] "pc"           "sb"           "pa"           "t19974"      
## [13] "t19981"       "t19982"       "t19983"       "t19984"      
## [17] "t19991"       "t19992"       "t19993"       "t19994"      
## [21] "t20001"       "t20002"       "t20003"       "t20004"      
## [25] "t20011"       "t20012"       "t20013"       "t20014"      
## [29] "t20021"       "t20022"       "t20023"       "t20024"      
## [33] "t20031"       "t20032"       "gg_l"         "mg_l"        
## [37] "def_gdp_l"    "gge_gdp_l"    "sum_ii_l"     "sum_i_l"     
## [41] "sum_l"        "libor_l"      "res_c_l"      "gg_oecd_l"   
## [45] "cab_gdp_l"    "imf_c_liab_l" "sum_tra"      "sum_not_tra"
## [49] "election_av"  "un_el_av"     "cum_ca_l"     "cum_uk_l"
## [53] "cum_fr_l"     "cum_ge_l"     "cum_it_l"     "cum_ja_l"
## [57] "ca_el_av"     "uk_el_av"     "fr_el_av"     "ge_el_av"
## [61] "it_el_av"     "ja_el_av"
```

```
sum(is.na(x))
```

```
## [1] 29661
```

```
# delete na and keep useful columns
data.0 = na.omit(x[,c("sum","un_per_l", "election_av", "un_el_av", "gdp_r_wk",
                      "gg_oecd_l","libor_l", "mg_l","imf_c_liab_l", "country", "year")])

# replicate glm and get t statistic
model.0 = glm(sum ~ un_per_l*election_av + gdp_r_wk +gg_oecd_l + libor_l + mg_l +
                factor(country) + factor(year), data = data.0, family = "poisson")
hc0.0 = sqrt(diag(vcovHC(model.0, type = "HC0")))
hc1.0 = sqrt(diag(vcovHC(model.0, type = "HC1")))
covar.0 = summary(model.0)$coef
```

```
# show standard errors
se.0 = cbind(covar.0[1:7,2], hc0.0[1:7],hc1.0[1:7])
colnames(se.0) = c("ols","hc0","hc1")
round(se.0[2,],2)
```

```
##  ols  hc0  hc1
## 3.74 6.29 8.08
```

“For their coefficient on U.S.support -9.55, the classical standard error is 3.73, whereas the robust standard error is larger, at 6.28, a difference of substantive importance.” (King and Roberts (2015))

“We fix the first problem by switching from a Poisson to a negative binomial distribution and the second by truncating it.” (King and Roberts (2015))

“The result is a 0-to-4 truncated negative binomial regression model, paralleling our simulation on the effects of changing to a better-fitting distribution” (King and Roberts (2015))

```
library(maxLik)
```

```
## Loading required package: miscTools
```

```
##
## Please cite the 'maxLik' package as:
## Henningsen, Arne and Toomet, Ott (2011). maxLik: A package for maximum likelihood estimation in R. C
##
## If you have questions, suggestions, or comments regarding the 'maxLik' package, please use a forum o
## https://r-forge.r-project.org/projects/maxlik/
```

```
# construct a 0 truncated nb ll
trunc.0.ll <- function(par, y, X, cut){
  end <- (length(par))-1
  theta <- par[1:end]
```

```r
  alpha <- exp(par[length(par)])
  lambda <- exp(drop(X %*% theta))
  zeros <- pnbinom(cut, size = alpha, mu = lambda,
                   lower.tail = F, log.p = T)
  ll <- sum(dnbinom(y, size = alpha, mu = lambda, log = T) - zeros)
  return(ll)
}


# get x and y
X <- cbind(model.matrix(model.0)[,!is.na(as.vector(model.0$coefficients))])
y <- data.0$sum

# optimize
out <- optim(c(rep(0.01, ncol(X)), 0.01), trunc.0.ll, y = y, X = X, cut = 4,
control = list(fnscale = -1, maxit = 10000), method = "BFGS", hessian = T)
vcov <- solve(-out$hessian)
out2 <- apply(cbind(y, X), 1,
              function(x) numericGradient(trunc.0.ll, out$par, y=x[1],
                                          X = x[2:(length(x))], cut = 4))
#sandwich estimator
meat <- out2 %*% t(out2)
bread <- vcov %*% meat %*% vcov

# check standard errors
se.1 = cbind(sqrt(diag(bread))[1:7], sqrt(diag(vcov))[1:7])
colnames(se.1) = c("ols.new","hc0")
round(se.1[2,], 2)
```

```
## ols.new     hc0
##    6.80    6.12
```

In this new model, the robust standard error is nearly what the paper reported, that is, 6.76.