[Example12-4] FGLS Estimation

Kei Sakamoto

Chapter8 で hetero-robust se 使った OLS よりも WLS にした方が efficient だったのと同じ。HACSE 使った OLS よりも、(regressor が strictly exogenous なら)FGLS にした方が efficient で(x の最初の方の情報失ってるので conditional だけど)BLUE。(large sample なら conditional はあまり関係ない。)

wooldridge では FGLS は 2 パターン。

①Prais-winsten method

②Chocrane-Orcutt method

(Stock-Watson の方では Prais-winsten がない代わりに ADL があった)

```
load("~/計量経済学演習/R data sets for 5e/barium.RData")
barium<-data

library(dynlm);library(car)

## Loading required package: zoo

##
## Attaching package: 'zoo'

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

## Loading required package: carData

#install.packages("orcutt")
library(orcutt)

## Loading required package: lmtest

#install.packages("prais")
library(prais)
```

```r
tsdata <- ts(barium, start=c(1978,2), frequency=12)
```

OLS estimation

```r
ols<-dynlm(log(chnimp)~log(chempi)+log(gas)+log(rtwex)+
      befile6+affile6+afdec6, data=tsdata)
library(sandwich)
coeftest(ols,vcovHAC)
```

```
##
## t test of coefficients:
##
##               Estimate Std. Error t value  Pr(>|t|)
## (Intercept) -17.802768  26.497047 -0.6719   0.50291
## log(chempi)   3.117194   0.654191  4.7650 5.188e-06 ***
## log(gas)      0.196341   1.196616  0.1641   0.86994
## log(rtwex)    0.983016   0.450206  2.1835   0.03088 *
## befile6       0.059574   0.153259  0.3887   0.69815
## affile6      -0.032406   0.233788 -0.1386   0.88998
## afdec6       -0.565245   0.249559 -2.2650   0.02525 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

① Prais-Winsten Estimation

```r
summary(prais_winsten(log(chnimp)~log(chempi)+log(gas)+log(rtwex)+
      befile6+affile6+afdec6, data=tsdata))
```

```
## Iteration 0: rho = 0
## Iteration 1: rho = 0.2708
## Iteration 2: rho = 0.291
## Iteration 3: rho = 0.293
## Iteration 4: rho = 0.2932
## Iteration 5: rho = 0.2932
## Iteration 6: rho = 0.2932
## Iteration 7: rho = 0.2932

##
## Call:
## prais_winsten(formula = log(chnimp) ~ log(chempi) + log(gas) +
##     log(rtwex) + befile6 + affile6 + afdec6, data = tsdata)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.99386 -0.32219  0.03747  0.40226  1.50281
##
## AR(1) coefficient rho after 7 Iterations: 0.2932
##
## Coefficients:
```

```
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -37.07742   22.77831  -1.628   0.1061
## log(chempi)   2.94095    0.63284   4.647 8.46e-06 ***
## log(gas)      1.04637    0.97734   1.071   0.2864
## log(rtwex)    1.13279    0.50666   2.236   0.0272 *
## befile6      -0.01648    0.31938  -0.052   0.9589
## affile6      -0.03316    0.32181  -0.103   0.9181
## afdec6       -0.57681    0.34199  -1.687   0.0942 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5733 on 124 degrees of freedom
## Multiple R-squared:  0.2021, Adjusted R-squared:  0.1635
## F-statistic: 5.235 on 6 and 124 DF,  p-value: 7.764e-05
##
## Durbin-Watson statistic (original): 1.458
## Durbin-Watson statistic (transformed): 2.087
```

ρ converged to around 0.29

②Cochrane-Orcutt estimation

```
summary(cochrane.orcutt(ols))
```

```
## Call:
## dynlm(formula = log(chnimp) ~ log(chempi) + log(gas) + log(rtwex) +
##     befile6 + affile6 + afdec6, data = tsdata)
##
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -37.322241  23.221406  -1.607  0.11057
## log(chempi)   2.947434   0.645559   4.566 1.19e-05 ***
## log(gas)      1.054858   0.990903   1.065  0.28917
## log(rtwex)    1.136918   0.513511   2.214  0.02867 *
## befile6      -0.016372   0.320722  -0.051  0.95937
## affile6      -0.033082   0.323152  -0.102  0.91863
## afdec6       -0.577158   0.343454  -1.680  0.09541 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5756 on 123 degrees of freedom
## Multiple R-squared:  0.1924 ,  Adjusted R-squared:  0.153
## F-statistic: 4.9 on 6 and 123 DF,  p-value: < 1.65e-04
##
## Durbin-Watson statistic
## (original):    1.45841 , p-value: 1.688e-04
## (transformed): 2.06330 , p-value: 4.91e-01
```

推定結果自体に大差はないものの、やはり HACse 使った ols よりも後の２つの方が se 小さく efficient なことがわかる。