GLS, WLS, and MLE

(Jing Li, Miami University)

1. The error term in a regression represents unobserved factors. When the error term is homoskedastic and uncorrelated with each other, OLS is best linear unbiased estimator (BLUE) according to Gauss-Markov theorem.

2. This note is concerned with error term that shows heteroskedasticity or correlation with each other. There are two implications—(i) OLS can still be consistent if exogeneity holds, but we have to use robust standard error; (ii) Generalized Least Squares (GLS) is BLUE, while OLS is not. Essentially GLS is OLS applied to a transformed regression in which the error terms are homoskedastic and uncorrelated with each other.

Cochrane-Orcutt estimator

3. We examine the issue of correlated error term first. In cross sectional data, error term can be correlated in the form of (1) spatial correlation—Ohio and Indiana may be correlated because geographically they are neighbors; (2) cluster correlation—kids in the same family are correlated because they share same parents. In time series data, a variable can be correlated with its own past, called serial correlation. This section focuses on serial correlation.

4. Consider a time series regression with error term u that follows a first-order autoregressive or AR(1) process

$$y_t = \beta_1 x_t + u_t, \quad u_t = \rho u_{t-1} + e_t \tag{1}$$

where we assume (i) exogeneity $cov(x_t, u_t) = 0$, and (ii) e_t is serially-uncorrelated and homoskedastic. It is easy to show the error terms u are serially correlated

$$cov(u_t, u_{t-1}) \neq 0$$

In this case we can still use OLS as long as serial-correlation-robust-standard-error is employed (consider *coeftest* function in lmtest package).

5. An alternative approach is to transform the regression so that Gauss-Markov theorem becomes applicable again. Toward that end, replace u_t with $y_t - \beta_1 x_t$ in the AR(1)

regression $u_t = \rho u_{t-1} + e_t$. Algebra rearrangement leads to

$$(y_t - \rho y_{t-1}) = \beta_1(x_t - \rho x_{t-1}) + e_t \tag{2}$$

Notice that the new error term e_t is serially uncorrelated and homoskedastic. Thus, OLS applied to the transformed regression above should produce BLUE result (called Cochrane-Orcutt estimator).

6. Cochrane-Orcutt estimator is just one example of GLS estimator, which is OLS applied to that transformed regression (2) in this case. The transformed variables $y_t - \rho y_{t-1}$ and $x_t - \rho x_{t-1}$ are called quasi-differenced data.

7. In reality, the autoregressive coefficient ρ is unknown. The feasible GLS (FGLS) estimator entails a multiple-step procedure

   - Step 1: estimate original regression (1) with OLS
   - Step 2: save residual $\hat{u}$ and run AR(1) to obtain $\hat{\rho}$
   - Step 3: create quasi-difference $y_t - \hat{\rho} y_{t-1}$ and $x_t - \hat{\rho} x_{t-1}$, then run the transformed regression (2) with OLS
   - After we get better estimates of β, we can obtain better residuals, and better estimate of ρ. In other words, Steps 2 an 3 may be iterated until the change in estimates is close to zero (convergence)

8. We use simulated data as an illustration. By construction, the error term follows an AR(1) process (e in the following codes are actually u in the math equations).

```
> set.seed(12345)
> n = 100; rho = 0.8; beta1=2.5
> e = rep(0,n)
> for (t in 2:n) {
+  e[t]=rho*e[t-1]+rnorm(1)
+ }
> x = rnorm(n)
> y = 1+ beta1*x + e
```

Simulation enables us to know the true values of $\beta_1 = 2.5$ and $\rho = 0.8$.

9. Before running regressions, we create a user-defined lag function

   ```
   > lag = function(x) {
   +  return(c(NA, x[1:length(x)-1]))
   + }
   ```

   Notice that a missing value NA is included

10. The OLS estimate of original model and AR(1) regression are

    ```
    > mols = lm(y˜x)
    > ehat = resid(mols)
    > rho = coef(lm(ehat˜lag(ehat)))[2]
    > cat("OLS estimate of slope is",coef(mols)[2])
    OLS estimate of slope is 2.76691
    > cat("AR(1) estimate of rho is",rho)
    AR(1) estimate of rho is 0.7067112
    ```

    The two estimates 2.76691 and 0.7067112 are close to true values 2.5 and 0.8.

11. Next we obtain FGLS in an iterative fashion

    ```
    > for (j in 1:4) {
    + cat("*********iteration",j,"*********","\n")
    + ys = y - rho*lag(y)
    + xs = x - rho*lag(x)
    + cat("GLS estimate of slope is",coef(lm(ys˜xs))[2],"\n")
    + ehat = y - coef(lm(ys˜xs))[1]-coef(lm(ys˜xs))[2]*x
    + rho = coef(lm(ehat˜lag(ehat)))[2]
    + cat("AR(1) estimate of rho is",rho,"\n")
    + }
    *********iteration 1 *********
    GLS estimate of slope is 2.583202
    AR(1) estimate of rho is 0.7251058
    *********iteration 2 *********
    GLS estimate of slope is 2.581839
    AR(1) estimate of rho is 0.7251657
    ```

```
*********iteration 3 *********
GLS estimate of slope is 2.581835
AR(1) estimate of rho is 0.7251659
*********iteration 4 *********
GLS estimate of slope is 2.581835
AR(1) estimate of rho is 0.7251659
```

Those estimates 2.581835 and 0.7251659 are closer to true values 2.5 and 0.8 than OLS estimates. Thus, FGLS outperforms OLS in this example. This finding is expected because the error term is serially correlated.

12. By the way, *cochrane.orcutt* function in the orcutt package can obtain the same estimates

```
> library(orcutt)
> cochrane.orcutt(mols, convergence = 8, max.iter=100)
Cochrane-orcutt estimation for first order autocorrelation

 number of interaction: 4
 rho 0.725166

Durbin-Watson statistic
(original):    0.58256 , p-value: 4.074e-13
(transformed): 1.92212 , p-value: 3.676e-01

 coefficients:
(Intercept)           x
   2.251843    2.581835
```

The Durbin-Watson statistic implies that the original model (1) suffers serial correlation in error term, while the transformed regression (2) does not (which is also why GLS is better than OLS). Google "Durbin-Watson statistic" to learn more.

WLS

13. This section focuses on the issue of heteroskedasticity (varying variance) in error term.

14. A popular time series model allowing for conditional heteroskedasticity is ARCH model. For example, the simplest first-order ARCH model consists of three equations

$$y_t = \beta_1 x_t + e_t \tag{3}$$
$$e_t = \sqrt{h_t} v_t \tag{4}$$
$$h_t = a_0 + a_1 e_{t-1}^2 \tag{5}$$

where v_t is white nose (but e_t is not).

15. We can show that the conditional variance of the error term is h, which is NOT constant (i.e., heterskedasticity is present):

$$var(e_t|\Omega_t) = h_t = a_0 + a_1 e_{t-1}^2 \neq constant \tag{6}$$

where the information set Ω_t includes lag values of observable variables. Later we call (3) the mean regression; and (5) the variance regression.

16. If a_1 is positive, then a large previous forecasting error e_{t-1} will result in greater volatility in the current period (measured by h_t) than a small previous forecasting error. This phenomenon is called volatility clustering. Read

```
https://www.fsb.miamioh.edu/lij14/672_engle.pdf
```

to learn more about ARCH model

17. Again we use simulated data so that we can compare estimation to truth

```
> set.seed(12345)
> n = 1000; a0 = 0.02; a1 = 0.7; beta1=-0.07
> e = rep(0,n)
> h = rep(0,n)
> h[1]=a0
> for (t in 2:n) {
+  h[t] = a0 + a1*e[t-1]^2
+  v = rnorm(1)
+  e[t]= sqrt(h[t])*v
+  }
```

```
> x = rnorm(n)
> y = 1 + beta1*x+e
```

The still consistent but non-BLUE OLS estimation of mean regression (3) are

```
> mols = lm(y~x)
> cat("OLS estimate of slope is",coef(mols)[2])
OLS estimate of slope is -0.05732881
```

18. For the variance regression $h_t = a_0 + a_1 e_{t-1}^2$, we can apply *garch* function in the tseries package

```
> ehat = resid(mols)
> library(tseries)
> co = coef(garch(ehat, order=c(0,1),trace=F))
> cat("ARCH(1) estimate is",co)
ARCH(1) estimate is 0.01970662 0.7332201
```

Note that the estimates -0.05732881, 0.01970662, and 0.7332201 are close to true values -0.07, 0.02, and 0.7.

19. Similar to the Cochrane-Orcutt estimator, weighted least squares (WLS) is another example of GLS. It is also OLS applied to a transformed regression. But here the transformed regression is obtained by dividing both sides of model (3) by $\sqrt{h_t}$

$$\frac{y_t}{\sqrt{h_t}} = \beta_1 \frac{y_t}{\sqrt{h_t}} + v_t \tag{7}$$

where we use the fact that $\frac{e_t}{\sqrt{h_t}} = v_t$. If the original model has an intercept we need to add $\beta_0 \frac{1}{\sqrt{h_t}}$ on the right hand side of (7). Notice that this transformed regression does NOT include a constant term.

20. The <u>new</u> error term v_t is serially uncorrelated and homoskedastic, so OLS applied to the <u>transformed</u> regression above should produce BLUE result. By definition, WLS is OLS applied to (7). Again, an iterative procedure may be employed to produce WLS estimation

```
> for (j in 1:4) {
+ cat("********iteration",j,"*********","\n")
+ h = rep(0,n)
+ h[1] = co[1]
+ for (t in 2:n) {
+  h[t]=co[1]+co[2]*ehat[t-1]^2
+ }
+ ys = y/sqrt(h)
+ xs = x/sqrt(h)
+ cs = 1/sqrt(h)
+ cat("GLS estimate of slope is",coef(lm(ys~cs+xs-1))[2],"\n")
+ ehat = y - coef(lm(ys~cs+xs-1))[1]-coef(lm(ys~cs+xs-1))[2]*x
+ co = coef(garch(ehat, order=c(0,1),trace=F))
+ cat("ARCH(1) estimate is",co,"\n")
+ }
********iteration 1 *********
GLS estimate of slope is -0.06613155
ARCH(1) estimate is 0.01886773 0.7643652
********iteration 2 *********
GLS estimate of slope is -0.06683258
ARCH(1) estimate is 0.01882921 0.7659824
********iteration 3 *********
GLS estimate of slope is -0.06688262
ARCH(1) estimate is 0.01882676 0.766087
********iteration 4 *********
GLS estimate of slope is -0.06688619
ARCH(1) estimate is 0.01882659 0.7660944
```

Notice that

(a) h is unobservable in reality

(b) we estimate h using the variance regression

(c) we drop the constant term in the transformed regression

21. The estimation of mean regression in iteration 4 can also be obtained using lm function with weight option

```
> mwls=lm(y~x,weight=1/h)
> summary(mwls)
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  1.005907   0.005668  177.48   <2e-16 ***
x           -0.066886   0.005598  -11.95  <2e-16 ***
Multiple R-squared:  0.1251,        Adjusted R-squared:  0.1243
```

22. By comparison, OLS estimation of the mean regression are below

```
> summary(mols)
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  1.013063   0.007764 130.479  < 2e-16 ***
x           -0.057329   0.007702  -7.443 2.12e-13 ***
Multiple R-squared:  0.05259,        Adjusted R-squared:  0.05164
```

In this example, we see that (i) the WLS estimate of slope coefficient -0.066886 is closer to the true value -0.07 than OLS estimate -0.057329; (ii) WLS standard error 0.005598 is less than OLS 0.007702; (iii) WLS R squared 0.1251 is greater than OLS 0.05259. They all imply superiority of WLS over OLS.

MLE

23. Other than estimating the mean and variance regressions sequentially, we can estimate them simultaneously with maximum likelihood estimation (MLE)

24. Toward that end, we assume v_t follows a standard normal distribution, which implies that the conditional distribution of y is a normal distribution with mean value $\beta_0+\beta_1 x_t$ and variance h_t. Mathematically, we can write that distribution as

$$y_t \sim N(\beta_0 + \beta_1 x_t, h_t), \quad (h_t = a_0 + a_1 e_{t-1}^2) \tag{8}$$

where the unknown parameters are $\beta_0, \beta_1, a_0, a_1$

25. MLE is based on a log-likelihood function. First, the likelihood function at period t is

the probability density function of normal distribution

$$likelihood_t = \frac{1}{\sqrt{2\pi h_t}} e^{-\frac{(y_t-\beta_0-\beta_1 x_t)^2}{2h_t}} \tag{9}$$

Then we take log to obtain log-likelihood

$$loglikelihood_t = -\frac{1}{2}\log h_t - \frac{(y_t-\beta_0-\beta_1 x_t)^2}{2h_t} \tag{10}$$

where we ignore the constant $\sqrt{2\pi}$. Finally, the log-likelihood for whole sample is

$$\sum_t loglikelihood_t = \sum_t \left(-\frac{1}{2}\log h_t - \frac{(y_t-\beta_0-\beta_1 x_t)^2}{2h_t}\right) \tag{11}$$

26. In this case, we have to use numerical method to maximize the log likelihood since a closed-form or analytical solution does not exist. R has an optim function that solves optimization problem using numerical method.

27. The R codes of generating log-likelihood are below

```
> loglikf = function(co) {
+ a0 = co[1]
+ a1 = co[2]
+ b0 = co[3]
+ b1 = co[4]
+ n = length(y)
+ h = a0
+ logf = -0.5*log(h)-(y[1]-b0-b1*x[1])**2/(2*h)
+ e = y[1]-b0-b1*x[1]
+ for (i in 2:n) {
+ h = a0 + a1*e^2
+ logf = logf -0.5*log(h)-(y[i]-b0-b1*x[i])**2/(2*h)
+ e = y[i]-b0-b1*x[i]
+ }
+ return(-logf)
+ }
```

28. Next we maximize the log-likelihood with optim function

```
> optim(co <- c(0.1,0.1,0.1,0.1), loglikf, hessian=TRUE)
$par
[1]  0.01855151  0.77693276  1.00211265  -0.07012560
$ convergence
[1]  0
```

The algorithm converges. The estimates -0.07012560, 0.01855151, and 0.77693276 are close to true values -0.07, 0.02, and 0.7. An introduction to maximum likelihood method is below

`https://www.fsb.miamioh.edu/lij14/311r_or_ml.pdf`