Econometrics II TA Session # (F)GLS*

Kan Pang †

November 30, 2020

Contents

*All comments welcome!

†E-mail: member_1363710747@yahoo.co.jp

1 OLS Estimation with HAC Estimator

Brief Background. The data we use today originates from Harrison, D. and Rubinfeld, D.L.(1978)[5] which investigates the methodological problems associated with the use of housing market data to measure the willingness to pay for clean air. It drew a conclusion that marginal air pollution damages are found to increase with the level of air pollution and with household income. But here we just use some variables of the original data to construct a multiple linear regression model focusing on how the attributes of communities affect housing prices(value) of Boston city.

Data. We use an open data which is called as "Boston Neighboorhood Housing Prices Dataset[1]". Although there exist many variables, we just take 4 of them: `value`, `crime`, `industrial`, `distance`. And here come the descriptions.

- `value`: a continuous variable meaning value of owner-occupied homes in $1000's.
- `crime`: a continuous variable representing per capita crime rate by town.
- `industrial`: a continuous variable showing proportion of non-retail business acres per town.
- `distance`: a continuous variable revealing weighted distances to five Boston employment centres.

As I mentioned before, `value` is the dependent variable and the other three are independent variables. Firstly, let's bulid the dataset in R.

```
library(AER) # Including lmtest and sandwich
library(nlme) # Do gls

# Preparations and preliminary tests

dt = read.csv(
file = "./boston.csv",
header = TRUE, sep = ",", row.names = NULL, stringsAsFactors =
    FALSE)
dt = dt[complete.cases(dt),] # complete.cases returns TURE if one
     row doesn't contain NA
dt = dt[,c("value", "crime", "industrial", "distance")]

n = nrow(dt)
head(dt)
```

```
##    value   crime  industrial distance
## 1  24.0  0.00632       2.31    4.0900
## 2  21.6  0.02731       7.07    4.9671
## 3  34.7  0.02729       7.07    4.9671
## 4  33.4  0.03237       2.18    6.0622
## 5  36.2  0.06905       2.18    6.0622
## 6  28.7  0.02985       2.18    6.0622
```

Model. The multiple regression model is easily written as below.

$$value_i = \beta_0 + \beta_1 crime_i + \beta_2 industrial_i + \beta_3 distance_i + u_i \quad \forall i = 1, \ldots, n.$$

And don't forget the assumptions.

[1]data source: `http://biostat.mc.vanderbilt.edu/DataSets`.

Assumptions of GLS

A linear regression model with heteroscedastic and autocorrelative error terms is defined as follows.

$$\underline{Y} = \underline{\boldsymbol{X}}\boldsymbol{\beta} + \underline{u}$$

with assumptions:

GH1 $\mathbb{E}[\underline{u}|\underline{\boldsymbol{X}}] = 0$.

GH2 $Var(\underline{u}|\underline{\boldsymbol{X}}) = \Omega = \Sigma(\underline{\boldsymbol{X}}, \theta) \succ 0$.

GH3 $\underline{\boldsymbol{X}}^T\underline{\boldsymbol{X}} \succ 0$.

1.1 Checking heteroskedasticity and autocorrelation

It is very common for us to apply ols to linear regressions. But with the previous assumptions GH1-GH3, OLS estimators do not own the BLUE property any more. To deal with the relaxations of Guass-Markov assumptions, one can use OLS method with a heteroscedasticity and autocorrelation consistent (HAC) covariance matrix estimator.
Before we talk about HAC estimator, let me show you 2 tests which are helpful in checking heteroskedasticity and autocorrelation, respectively.
Firstly, the Breusch=Pagan=Godfrey(BPG) test for heteroskedasticity.

The BPG Test

Assume that in a linear regression model, the followings hold.

$$y_i = \boldsymbol{X}_i\boldsymbol{\beta} + u_i \quad u_i|\boldsymbol{X}_i \sim \mathcal{N}_{\mathbb{R}}(0, \sigma_i^2)$$
$$\sigma_i^2 = \mathbb{E}[u_i^2|\boldsymbol{X}_i] = \alpha_0 + \alpha_1 X_{1i} + \cdots + \alpha_p X_{pi} + v_i \quad \forall i = 1, \ldots, n.$$

where v_i is a 0-meand and homoscedastic error term which is not correlated with $\boldsymbol{X}_i$, for all i. And then, the null hypothesis is $H_0 : \alpha_1 = \alpha_2 = \cdots = \alpha_p = 0$(means homoscedasticity). To implement the test, one can use the three-step procedure.

step1 Apply OLS in the model

$$y_i = \boldsymbol{X}_i\boldsymbol{\beta} + u_i$$

step2 Compute the regression residuals, $\hat{u}_i$, square them, and estimate the auxiliary regression

$$\hat{u}_i^2 = \alpha_0 + \alpha_1 X_{1i} + \cdots + \alpha_p X_{pi} + v_i$$

It is possible to use different covariates instead of $\boldsymbol{X}_i$.

step3 Multiply the coefficient of determination(R squared) derieved from the auxiliary regression in step 2 by sample size n to obtain the test satistic

$$nR^2 \sim \chi_p^2$$

For more details, please check Breusch & Pagan(1979)[1].

Performing `bptest()` function from `lmtest` package. We have the followings.

```
# OLS
model = value ~ crime + industrial + distance
ols = lm(model, data = dt)
ols_summary = summary(ols)

# Breusch=Pagan=Godfrey test against heteroskedasticity.
```

```
7
8 (bpgtest = bptest(ols, data = dt, studentize = F)) # from lmtest
9 # If studentize is set to TRUE Koenker's studentized version of
    the test statistic will be used.
```

And the result is returned as:

```
##  Breusch-Pagan test
## data:  ols
## BP = 28.757, df = 3, p-value = 2.519e-06
```

Since the p-value is very small, it's known that the null hypothesis is rejected, there exists heteroscedasticity in this model.

Next, let me introduce the Durbin-Watson test to you.

The Durbin-Watson Test

In statistics, the DurbinWatson statistic is a test statistic used to detect the presence of autocorrelation at lag 1 in the residuals (prediction errors) from a regression analysis. It is named after James Durbin and Geoffrey Watson[2][3].
Assume that

$$y_i = \boldsymbol{X}_i\boldsymbol{\beta} + u_i$$
$$u_i = \rho u_{i-1} + \epsilon_i \quad \forall i = 1, \dots, N.$$

where ϵ_i is a 0-meand and homoscedastic error term which is not correlated with u_i, for all i. Durbin-Watson statistic states that null hypothesis: $\rho = 0$, alternative hypothesis $\rho \neq 0$, then if $\hat{u}_i$ is the OLS residual, the test statistic is

$$dw = \frac{\sum_{i=2}^{N}(\hat{u}_i - \hat{u}_{i-1})^2}{\sum_{i=1}^{N}\hat{u}_i^2}$$

where N is the number of observations. And there is a useful approximatate equation, $dw = 2(1 - \hat{\rho})$ where $\hat{\rho}$ is the sample autocorrelation of the residuals. Besides, dw statistic can be interpreted as follows.
With lower and upper critical values given as $dw_{L,\alpha}$ and $dw_{U,\alpha}$, to test for **positive autocorrelation** at significance α,

- If dw < $dw_{L,\alpha}$, there is statistical evidence that the error terms are positively autocorrelated.
- If dw > $dw_{U,\alpha}$, there is **no** statistical evidence that the error terms are positively autocorrelated.
- If $dw_{L,\alpha}$ < dw < $dw_{U,\alpha}$, the test is inconclusive.

And to test for **negative autocorrelation** at significance α,

- If 4-dw < $dw_{L,\alpha}$, there is statistical evidence that the error terms are negatively autocorrelated.
- If 4-dw > $dw_{U,\alpha}$, there is **no** statistical evidence that the error terms are negatively autocorrelated.
- If $dw_{L,\alpha}$ < 4-dw < $dw_{U,\alpha}$, the test is inconclusive.

Performing `dwtest()` function from `lmtest` package. We have the followings.

```r
# Durbin-Watson test for autocorrelation of disturbances.

(dwtest = dwtest(ols, data = dt, alternative = "two.sided")) #
    alternative = c("greater", "two.sided", "less")
```

And the result is:

```
##  Durbin-Watson test
## data:  ols
##  DW = 0.77642, p-value < 2.2e-16
##  alternative hypothesis: true autocorrelation is not 0
```

Obviously, autocorrelation exists.

Let's learn how to read the statistic table in case you may only have the DW statistic sometime. When you are faced with a table of DW test statistics, please pay attention to whether there is an intercept and the number of covariates in the original linear model $y_i = \boldsymbol{X}_i\boldsymbol{\beta} + u_i$. For example, in our case, we should refer to the table which clearly tells that there are 3 covaritates and an intercept. However, a statistic table marked by 3 covariates but no intercept is totally different from the previous one.

Table 1: Durbin-Watson Statistic: 1 Percent Significance Points of dL and dU(with an intercept)

	k' = 3		k' = 5		k' = 10	
n	dL	dU	dL	dU	dL	dU
10	0.340	1.733	0.150	2.690	NA	NA
25	0.906	1.408	0.756	1.645	0.409	2.362
50	1.245	1.491	1.206	1.537	0.955	1.864
100	1.482	1.604	1.441	1.647	1.335	1.765
200	1.643	1.704	1.633	1.715	1.571	1.779

Table 2: Durbin-Watson Statistic: 1 Percent Significance Points of dL and dU(with no intercept)

	K = 3		K = 5		K = 10	
n	dL	dU	dL	dU	dL	dU
10	0.223	2.121	0.070	1.224	NA	NA
25	0.839	2.518	0.693	2.282	0.361	1.572
50	1.208	2.471	1.128	2.374	0.921	2.098
100	1.463	2.377	1.422	2.333	1.317	2.215
200	1.634	2.286	1.613	2.265	1.561	2.211

In the previous 2 tables, both k' and K represent the number of covariates. And in our case, we choose table 1. Because lower bound is monotonically increasing and DW = 0.77642 < 1.643, it's clear that there exists positive autocorrelation at 1% significane level.

1.2 OLS method with HAC covariance matrix estirmator

When it comes to autocorrelation, we can't simply apply ols method with the White estimator. An estimator, overcoming heteroscedasticity and autocorrelation at the same time, is called as HAC(Heteroscedasticity and Autocorrelation Consistent) covariance matrix estirmator. Today, I show you the most famous one, NeweyWest estimator.

NeweyWest estimator

The NeweyWest estimator was devised by Whitney K. Newey and Kenneth D. West in 1987[7], , although there are a number of later variants. The estimator is used to try to overcome autocorrelation (also called serial correlation), and heteroskedasticity in the error terms in the models, often for regressions applied to time series data.
The general approach is to use $\underline{\boldsymbol{X}}$ and $\boldsymbol{e}$ to devise an estimator of Q^*, a matrix of sums of squares and cross products that involves σ_{ij} and the rows of $\underline{\boldsymbol{X}}$. The least squares estimator $\boldsymbol{b}$ is a consistent estimator of $\boldsymbol{\beta}$, which implies that the least squares residuals $\boldsymbol{e_i}$ are "point-wise" consistent estimators of their population counterparts.

$$Q^* = \frac{1}{T}\sum_{t=1}^{T} e_t^2 x_t x_t' + \frac{1}{T}\sum_{\ell=1}^{L}\sum_{t=\ell+1}^{T} \omega_\ell e_t e_{t-\ell}(x_t x_{t-\ell}' + x_{t-\ell}' x_t)$$

$$\omega_\ell = 1 - \frac{\ell}{L+1}$$

ω_ℓ can be thought of as a "weight". Disturbances that are farther apart from each other are given lower weight, while those with equal subscripts are given a weight of 1. This ensures that second term converges (in some appropriate sense) to a finite matrix. This weighting scheme also ensures that the resulting covariance matrix is positive semi-definite.

Because numerical caculation is a little bit difficult, we directly use `sandwich::NeweyWest()` function to obtain Newey-West covariance estimate. Following are the commands and results.

```
# OLS method with HAC covariance matrix estirmator

cov_hac = NeweyWest(ols)
se_hac = sqrt(diag(cov_hac))

t_hac = coef(ols)/se_hac
p_hac = pt(abs(t_hac), df = nrow(dt) - 4, lower.tail = FALSE)*2
```

```
print("NeweyWest covariance matrix estimate:"); cov_hac
```

```
##                [1] "NeweyWest covariance matrix estimate:"
##              (Intercept)   crime       industrial   distance
## (Intercept)  8.90329265 -0.025862361 -0.327823433 -1.012079887
## crime       -0.02586236  0.003067065 -0.002275065  0.005972089
## industrial  -0.32782343 -0.002275065  0.020075383  0.027409192
## distance    -1.01207989  0.005972089  0.027409192  0.140507641
```

```
print("NeweyWest se estimates:"); se_hac
```

```
## [1] "NeweyWest se estimates:"
## (Intercept)   crime    industrial    distance
## 2.98383858  0.05538109  0.14168763  0.37484349
```

```
print("T statistics by NeweyWest covariance:"); t_hac
```

```
## [1] "T statistics by NeweyWest covariance:"
## (Intercept)   crime    industrial    distance
##  11.899262   -4.926366   -5.153366   -2.709985
```

```
print("P values:"); p_hac
```

```
## [1] "P values:"
## (Intercept)    crime     industrial    distance
##  6.200717e-29 1.139249e-06 3.684529e-07 6.958822e-03
```

The previous steps can be easily realized by using `lmtest::coeftest()` function. Here, I give the commands and display the results.

```
# Comparing with lmtest::coeftest
print("Test using NeweyWest estimator:"); coeftest(ols, vcov. =
    NeweyWest)
```

```
## [1] "Test using NeweyWest estimator:"

## t test of coefficients:
##
##             Estimate  Std. Error t value  Pr(>|t|)
## (Intercept) 35.505478   2.983839 11.8993 < 2.2e-16 ***
## crime       -0.272828   0.055381 -4.9264 1.139e-06 ***
## industrial  -0.730168   0.141688 -5.1534 3.685e-07 ***
## distance    -1.015820   0.374843 -2.7100  0.006959 **
## ---
## Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1
```

2 Numerical FGLS

In statistics, generalized least squares (GLS) is a technique for estimating the unknown parameters in a linear regression model when there are a certain degree of correlation between the residuals and(or) heteroscedasticity in a regression model. The estimators are summarized as follows.

Esitmators of GLS Method

With assumptions GH1-GH3, the GLS estimators for the parameters and covariance matrix are denoted like follows.

$$\hat{\boldsymbol{\beta}}_{GLS} = \left(\underline{\boldsymbol{X}}^T \Omega^{-1} \underline{\boldsymbol{X}}\right)^{-1} \underline{\boldsymbol{X}}^T \Omega^{-1} \underline{Y}$$
$$Var[\hat{\boldsymbol{\beta}}_{GLS} | \underline{\boldsymbol{X}}] = \left(\underline{\boldsymbol{X}}^T \Omega^{-1} \underline{\boldsymbol{X}}\right)^{-1}$$

2.1 Simple FGLS

If the covariance of the errors Ω is unknown, one can get a consistent estimate of Ω. say $\widehat{\Omega}$, using an implementable version of GLS known as the feasible generalized least squares (FGLS) estimator. In FGLS, modeling proceeds in two stages:

Step 1 the model is estimated by OLS or another consistent (but inefficient) estimator, and the residuals are used to build a consistent estimator of the errors covariance matrix (to do so, one often needs to add additional constraints).

Step 2 using the consistent estimator of the covariance matrix of the errors, one can implement GLS ideas.

To perfrom the previous steps, the main problem is that how we construct a consisten estimator $\widehat{\Omega}$. For simplicity, we assume that Ω is a diagonal matrix which is $\begin{bmatrix} \sigma_1^2 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \sigma_n^2 \end{bmatrix}$, and replace each diagonal element by squared OLS residuals($\hat{u}_i^2$), which means that $\widehat{\Omega} = \begin{bmatrix} \hat{u}_1^2 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \hat{u}_n^2 \end{bmatrix}$.

Here are commands.

```
# FGLS method by numerical calculation

X = as.matrix(cbind(rep(1, n), dt$crime, dt$industrial, dt$
    distance))
label = c("(Intercept)", "crime", "industrial", "distance")
colnames(X) = label

Y = as.vector(dt[, "value"])

# Using HCCME(HC_0)

cov_hat = diag(resid(ols)^2)


# Deriving estimates by closed-form estimators
b_fgls =  solve(t(X) %*% solve(cov_hat) %*% X) %*% (t(X) %*%
    solve(cov_hat) %*% Y)
cov_fgls = solve(t(X) %*% solve(cov_hat) %*% X)

se_fgls = sqrt(diag(cov_fgls))
t_fgls = b_fgls/se_fgls
p_fgls = pt(abs(t_fgls), df = n - ncol(X), lower.tail = FALSE)*2
```

And we can use the following commands to check the results.

```
print("FGLS covariance matrix estimate:"); cov_fgls
```

```
##                [1] "FGLS covariance matrix estimate:"
##             (Intercept)        crime     industrial     distance
## (Intercept)  4.211767e-03 -8.010524e-05 -1.199272e-04 -8.757138e-04
## crime       -8.010524e-05  5.611015e-05 -1.415095e-06  1.594712e-05
## industrial  -1.199272e-04 -1.415095e-06  6.642517e-06  1.865562e-05
## distance    -8.757138e-04  1.594712e-05  1.865562e-05  2.076158e-04
```

```
print("FGLS se estimates:"); se_fgls
```

```
##           [1] "FGLS se estimates:"
## (Intercept)       crime  industrial    distance
## 0.064898124 0.007490671 0.002577308 0.014408881
```

```
print("T statistics by FGLS covariance:"); t_fgls
```

```
## [1] "T statistics by FGLS covariance:"
##                 [,1]
```

```
## (Intercept)  547.28832
## crime        -36.71753
## industrial  -283.84251
## distance     -70.21385
```

```
print("P values:"); p_fgls
```

```
## [1] "P values:"
##                 [,1]
## (Intercept)  0.000000e+00
## crime       2.666154e-144
## industrial   0.000000e+00
## distance    9.426943e-262
```

2.2 Iterative FGLS Method

Sometimes, in order to improve the accuracy of the estimators in finite samples, we use iteration, i.e. taking the residuals from FGLS to update the errors covariance estimator, and then updating the FGLS estimation, applying the same idea iteratively until the estimators vary less than some tolerance. But this method does not necessarily improve the efficiency of the estimator very much if the original sample was small.

Now let me briefly introduce the key steps to you. The OLS estimator is calculated as usual by

$$\hat{\boldsymbol{\beta}}_{OLS} = \left(\underline{\boldsymbol{X}}^T \underline{\boldsymbol{X}}\right)^{-1} \underline{\boldsymbol{X}}^T \underline{Y}$$

and estimates of the residuals $\hat{u}_j = (\underline{Y} - \underline{\boldsymbol{X}}\hat{\boldsymbol{\beta}}_{OLS})_j$ are constructed.

Assuming diagonal covariance matrix Ω, we build $\widehat{\Omega}_{OLS} = diag(\hat{\sigma}_1^2, \hat{\sigma}_2^2, \ldots, \hat{\sigma}_n^2)$.

Then we estimate $\boldsymbol{\beta}_{FGLS1}$ using $\widehat{\Omega}_{OLS}$. With $\boldsymbol{\beta}_{FGLS1}$, performing the following steps.

setp1 $\hat{\boldsymbol{\beta}}_{FGLS1} = \left(\underline{\boldsymbol{X}}^T \widehat{\Omega}_{OLS}^{-1} \underline{\boldsymbol{X}}\right)^{-1} \underline{\boldsymbol{X}}^T \widehat{\Omega}_{OLS}^{-1} \underline{Y}$

setp2 $\hat{\boldsymbol{u}}_{FGLS1} = \underline{Y} - \underline{\boldsymbol{X}}\hat{\boldsymbol{\beta}}_{FGLS1}$

setp3 $\widehat{\Omega}_{FGLS1} = diag(\hat{\sigma}^2_{FGLS1,1}, \hat{\sigma}^2_{FGLS1,2}, \ldots, \hat{\sigma}^2_{FGLS1,n})$

setp4 $\hat{\boldsymbol{\beta}}_{FGLS2} = \left(\underline{\boldsymbol{X}}^T \widehat{\Omega}_{FGLS1}^{-1} \underline{\boldsymbol{X}}\right)^{-1} \underline{\boldsymbol{X}}^T \widehat{\Omega}_{FGLS1}^{-1} \underline{Y}$

Repeating step2 to step4 until the convergence of $\widehat{\Omega}$.

Let's check the commands.

```
# Iterative numerical method

termination = function(b, omega){
    grad = -2 * t(X) %*% solve(omega) %*% (Y - X %*% b)
    Inner = t(grad) %*% grad
    Inner_sqrt = sqrt(Inner)
    return(Inner_sqrt)
}

b_loop = b_fgls # step1
```

```r
cov_u = cov_hat
n_loop = 0

while (termination(b_loop, cov_u) > 10^-12) { # judge
    u_hat = as.vector(Y - X %*% b_loop) # step2
    cov_u = diag(u_hat^2)    # step3
    cov_b_loop = solve(t(X) %*% solve(cov_u) %*% X)
    b_loop =  solve(t(X) %*% solve(cov_u) %*% X) %*% (t(X) %*%
       solve(cov_u) %*% Y) # step4
    n_loop = n_loop + 1
}


se_loop = sqrt(diag(cov_b_loop))
t_loop = b_loop/se_loop
p_loop = pt(abs(t_loop), df = n - ncol(X), lower.tail = FALSE)*2
```

And the results[2]

```r
print("Number of iterations:"); n_loop
```

```
## [1] "Number of iterations:"
## [1] 2
```

```r
print("FGLS covariance matrix estimate:"); cov_b_loop
```

```
## [1] "FGLS covariance matrix estimate:"
##             (Intercept)         crime    industrial      distance
## (Intercept)  1.036975e-04 -6.721951e-05 -3.911968e-06 -1.146902e-05
## crime       -6.721951e-05  4.795889e-05  1.298768e-06  8.942851e-06
## industrial  -3.911968e-06  1.298768e-06  5.092477e-07 -1.687944e-08
## distance    -1.146902e-05  8.942851e-06 -1.687944e-08  1.832846e-06
```

```r
print("FGLS se estimates:"); se_loop
```

```
## [1] "FGLS se estimates:"
## (Intercept)       crime   industrial     distance
## 0.0101831954 0.0069252360 0.0007136159 0.0013538264
```

```r
print("T statistics by FGLS covariance:"); t_loop
```

```
## [1] "T statistics by FGLS covariance:"
##                  [,1]
## (Intercept)  3488.42343
## crime         -41.94157
## industrial  -1027.18612
## distance     -744.37635
```

```r
print("P values:"); p_loop
```

[2]Indeed, we didn't reach convergence because $\widehat{\Omega}_{FGLS}$ turned out to be computationally sigular after 2 times of iterations.

```
## [1] "P values:"
##                  [,1]
## (Intercept)  0.000000e+00
## crime       3.533637e-166
## industrial   0.000000e+00
## distance     0.000000e+00
```

3 Built-in R Function

3.1 nlme::gls Function

In fact, we don't need to repeat the previous section when estimating because we have `nlme::gls()` function. And to use this function, please pay attention to the following 2 arguments.

- **correlation**: We should assign an argument describing the correlation structure of the error terms. In this case, we use `corAR1(rho)` which provides a AR(1) DGP and rho is derieved from dw test statistic.
- **weights**: We should assign an argument describing the heteroscedasticity structure of the error terms. In this case, just use `varPower()`[3].

Commands and results are displayed as follows.

```
# Estimating by nlme::gls

# Caculating AR(1) estimate from dwtest
rho = 1 - 0.5 * as.numeric(dwtest$statistic)

# Estimating
gls_esti = gls(model, data = dt, correlation = corAR1(rho),
    weights = varPower())

(gls_summary = summary(gls_esti))
```

```
## Generalized least squares fit by REML
## Model: model
## Data: dt
## AIC      BIC      logLik
## 3259.125 3288.655 -1622.562
##
## Correlation Structure: AR(1)
## Formula: ~1
## Parameter estimate(s):
## Phi
## 0.6622399
## Variance function:
## Structure: Power of variance covariate
## Formula: ~fitted(.)
## Parameter estimates:
## power
## 0.4333236
##
## Coefficients:
##                 Value Std.Error   t-value p-value
```

[3] an varFunc object in R and for more details, use help(varPower) and help(varFunc)

```
## (Intercept) 31.310565 2.5770092 12.149962  0.0000
## crime       -0.107077 0.0298874 -3.582697  0.0004
## industrial  -0.588257 0.1110842 -5.295590  0.0000
## distance    -0.514168 0.3808652 -1.349999  0.1776
##
## Correlation:
##             (Intr) crime  indstr
## crime       -0.105
## industrial -0.843 -0.053
## distance    -0.851  0.119  0.585
##
## Standardized residuals:
## Min         Q1        Med         Q3        Max
## -1.6892209 -0.6198956 -0.1628794  0.3536828  4.2750494
##
## Residual standard error: 2.078919
## Degrees of freedom: 506 total; 502 residual
```

3.2 Results of all methods

Use stargazer function to make a table displaying the results of each model.

```r
# Making a table to show each model
library(stargazer)

R2_fgls = 1 - as.numeric(crossprod(Y - X %*% b_fgls)) / sum((Y - mean(Y))^2)
R2_loop = 1 - as.numeric(crossprod(Y - X %*% b_loop)) / sum((Y - mean(Y))^2)
R2_byR  = 1 - as.numeric(crossprod(Y - X %*% coef(gls_esti))) / sum((Y - mean(Y))^2)

se_byR  = coef(gls_summary)[, "Std.Error"]
t_byR   = coef(gls_summary)[, "t-value"]
p_byR   = coef(gls_summary)[, "p-value"]

stargazer(
ols, gls_esti, gls_esti, gls_esti,
coef = list(coef(ols), b_fgls, b_loop, coef(gls_esti)), se = list(se_hac, se_fgls, se_loop, se_byR),
t = list(t_hac, t_fgls, t_loop, t_byR), p = list(p_hac, p_fgls, p_loop, p_byR),
t.auto = FALSE, p.auto = FALSE,
report = "vcstp", keep.stat = c("n"),
add.lines = list(
c("Type", "HA-Roubusted OLS", "fgls", "fgls_loop", "Built-in R gls"),
c("R-Squared", round(ols_summary$r.squared, 4), round(R2_fgls, 4), round(R2_loop, 4), round(R2_byR, 4))),
title = "Results of linear model estimations",
label = "LS",
type = "latex", header = FALSE, font.size = "small",
table.placement = "htb", omit.table.layout = "n"
)
```

For results, check table 3.

3.3 Summary and Which Method to Choose

Looking at table 3, we know that compared with OLS using HAC estimates, numerical gls (2) and (3) obviously return more efficient estimates. And by iteration, accuracy did increase to some extent. But with a more correctly specified structure of errors covariance matrix, `gls()` in R didn't give optimal estiamtes(slightly better than OLSe).

Whereas GLS is more efficient than OLS under heteroscedasticity or autocorrelation, this is not true for FGLS. The feasible estimator is, provided the errors covariance matrix is consistently estimated, asymptotically more efficient, but for a small or medium size sample, it can be actually less efficient than OLS. This is why, some authors prefer to use OLS, and reformulate their inferences by simply considering an alternative estimator for the variance of the estimator robust to heteroscedasticity or serial autocorrelation(HCCME or HACE). But for large samples FGLS is preferred over OLS under heteroskedasticity or serial correlation[4].

[4]A cautionary note is that the FGLS estimator is not always consistent. One case in which FGLS might be inconsistent is if there are individual specific fixed effects.[4]

Table 3: Results of linear model estimations

	Dependent variable:			
	value			
	OLS		*generalized least squares*	
	(1)	(2)	(3)	(4)
crime	−0.273	−0.275	−0.290	−0.107
	(0.055)	(0.007)	(0.007)	(0.030)
	t = −4.926	t = −36.718	t = −41.942	t = −3.583
	p = 0.00001	p = 0.000	p = 0.000	p = 0.0004
industrial	−0.730	−0.732	−0.733	−0.588
	(0.142)	(0.003)	(0.001)	(0.111)
	t = −5.153	t = −283.843	t = −1,027.186	t = −5.296
	p = 0.00000	p = 0.000	p = 0.000	p = 0.00000
distance	−1.016	−1.012	−1.008	−0.514
	(0.375)	(0.014)	(0.001)	(0.381)
	t = −2.710	t = −70.214	t = −744.376	t = −1.350
	p = 0.007	p = 0.000	p = 0.000	p = 0.178
Constant	35.505	35.518	35.523	31.311
	(2.984)	(0.065)	(0.010)	(2.577)
	t = 11.899	t = 547.288	t = 3,488.423	t = 12.150
	p = 0.000	p = 0.000	p = 0.000	p = 0.000
Type	HA-Roubusted OLS	fgls	fgls_loop	Built-in R gls
R-Squared	0.3044	0.3044	0.3041	0.2732
Observations	506	506	506	506

References

[1] Breusch, T. S.; Pagan, A. R. (1979), "A Simple Test for Heteroskedasticity and Random Coefficient Variation," *Econometrica.* 47 (5): 12871294.

[2] Durbin, J.; Watson, G. S. (1950), "Testing for Serial Correlation in Least Squares Regression, I," *Biometrika.* 37 (34): 409428.

[3] Durbin, J.; Watson, G. S. (1951), "Testing for Serial Correlation in Least Squares Regression, II," *Biometrika.* 38 (12): 159179.

[4] Hansen, Christian B. (2007), "Generalized Least Squares Inference in Panel and Multilevel Models with Serial Correlation and Fixed Effects," *Journal of Econometrics.* 140 (2): 670694. doi:10.1016/j.jeconom.2006.07.011 .

[5] Harrison, D. and Rubinfeld, D.L. (1978), "Hedonic prices and the demand for clean air," J. *Environ. Economics & Management,* 5: 81-102.

[6] Hlavac, Marek (2018), "*stargazer: Well-Formatted Regression and Summary Statistics Tables.*"

[7] Newey, Whitney K; West, Kenneth D (1987), "A Simple, Positive Semi-definite, Heteroskedasticity and Autocorrelation Consistent Covariance Matrix," *Econometrica.* Vol. 55, No. 3, pp. 703-708.

[8] Wooldridge, Jeffrey M (2010), "Econometric analysis of cross section and panel data," MIT Press.