Standard Errors in OLS

Luke Sonnet

Contents

This document reviews how we think about and estimate uncertainty in OLS under various conditions. Much of the document is taken directly from these very clear notes, Greene's Econometric Analysis, and slides by Chad Hazlett.

Variance-Covariance of $\hat{\beta}$

Take the classic regression equation

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}$$

where $\mathbf{y}$ is an $n \times 1$ outcome vector, $\mathbf{X}$ is an $n \times p$ matrix of covariates, $\boldsymbol{\beta}$ is an $n \times 1$ vector of coefficients, and $\boldsymbol{\epsilon}$ is an $n \times 1$ vector of noise, or errors. Using OLS, our estimate of $\boldsymbol{\beta}$ is

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}^\top \mathbf{X})^{-1}\mathbf{X}^\top \mathbf{y}$$

This is just an estimate of the coefficients. We also would like to understand the variance of this estimate to quantify our uncertainty and possibly to perform significance testing. We can derive an explicit function that represents the variance of our estimates, $\mathbb{V}[\hat{\boldsymbol{\beta}}|\mathbf{X}]$, given that $\mathbf{X}$ is fixed.

First note that,

$$\begin{aligned}
\hat{\boldsymbol{\beta}} &= (\mathbf{X}^\top \mathbf{X})^{-1}\mathbf{X}^\top \mathbf{y} \\
&= (\mathbf{X}^\top \mathbf{X})^{-1}\mathbf{X}^\top (\mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}) \\
&= \boldsymbol{\beta} + (\mathbf{X}^\top \mathbf{X})^{-1}\mathbf{X}^\top \boldsymbol{\epsilon} \\
\hat{\boldsymbol{\beta}} - \boldsymbol{\beta} &= (\mathbf{X}^\top \mathbf{X})^{-1}\mathbf{X}^\top \boldsymbol{\epsilon}
\end{aligned}$$

Then,

$$\begin{aligned}
\mathbb{V}[\hat{\boldsymbol{\beta}}|\mathbf{X}] &= \mathbb{E}[(\hat{\boldsymbol{\beta}} - \boldsymbol{\beta})(\hat{\boldsymbol{\beta}} - \boldsymbol{\beta})^\top |\mathbf{X}] \\
&= \mathbb{E}[(\mathbf{X}^\top \mathbf{X})^{-1}\mathbf{X}^\top \boldsymbol{\epsilon}((\mathbf{X}^\top \mathbf{X})^{-1}\mathbf{X}^\top \boldsymbol{\epsilon})^\top |\mathbf{X}] \\
&= \mathbb{E}[(\mathbf{X}^\top \mathbf{X})^{-1}\mathbf{X}^\top \boldsymbol{\epsilon}\boldsymbol{\epsilon}^\top \mathbf{X}(\mathbf{X}^\top \mathbf{X})^{-1}|\mathbf{X}] \\
&= (\mathbf{X}^\top \mathbf{X})^{-1}\mathbf{X}^\top \mathbb{E}[\boldsymbol{\epsilon}\boldsymbol{\epsilon}^\top |\mathbf{X}]\mathbf{X}(\mathbf{X}^\top \mathbf{X})^{-1}
\end{aligned}$$

This then is our answer for the variance-covariance matrix of our coefficients $\hat{\boldsymbol{\beta}}$. While we have $\mathbf{X}$, we do not have $\mathbb{E}[\boldsymbol{\epsilon}\boldsymbol{\epsilon}^\top |\mathbf{X}]$, which is the variance-covariance matrix of the errors. Furthermore, this matrix has $n \times n$

unknown parameters that define the variance of each error and the correlation of errors. In theory, we could know the correlation between the error across observations, known as serial correlation, or whether variance of the errors is constant across observations, known as homoskedasticity.

This is a crucial point: we could have very complicated error structures but we cannot estimate the full matrix $\mathbb{E}[\boldsymbol{\epsilon}\boldsymbol{\epsilon}^\top|\mathbf{X}]$. We have to make assumptions about the error structure. Do all observations have errors with the same variance? Then we have homoskedasticity. Google heteroskedasticity for graphical representations of when this is violated. Do observations have errors that are correlated in groups? Then you should build this structure into your estimates of $\mathbb{E}[\boldsymbol{\epsilon}\boldsymbol{\epsilon}^\top|\mathbf{X}]$. In this document, I run through three of the most common cases. The standard case when we assume spherical errors (no serial correlation and no heteroskedasticity), the case where we allow heteroskedasticity, and the case where there is grouped correlation in the errors. In all cases we assume that the conditional mean of the error is 0. Precisely $\mathbb{E}[\epsilon|X] = 0$.

Standard Estimation (Spherical Errors)

Very often we make the standard spherical errors assumption. This means that the variance-covariance structure of the errors is very simple. In fact, we assume that all errors have the same variance and that there is no correlation across errors. This looks like the following:

$$\mathbb{E}[\boldsymbol{\epsilon}\boldsymbol{\epsilon}^\top|\mathbf{X}] = \begin{bmatrix} \sigma^2 & 0 & \cdots & 0 \\ 0 & \sigma^2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \sigma^2 \end{bmatrix} = \sigma^2\mathbf{I}$$

Therefore, all errors have the same variance, some scalar σ^2. Then the variance of our coefficients simplifies,

$$\begin{aligned} \mathbb{V}[\hat{\boldsymbol{\beta}}|\mathbf{X}] &= (\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbb{E}[\boldsymbol{\epsilon}\boldsymbol{\epsilon}^\top|\mathbf{X}]\mathbf{X}(\mathbf{X}^\top\mathbf{X})^{-1} \\ &= (\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\sigma^2\mathbf{I}\mathbf{X}(\mathbf{X}^\top\mathbf{X})^{-1} \\ &= \sigma^2(\mathbf{X}^\top\mathbf{X})^{-1} \end{aligned}$$

Now all we need is an estimate of σ^2 in order to get our estimate for $\mathbb{V}[\hat{\boldsymbol{\beta}}|\mathbf{X}]$. I do not show this here, but an unbiased estimate for σ^2 is,

$$\hat{\sigma^2} = \frac{\mathbf{e}^\top\mathbf{e}}{n-p}$$

where $\mathbf{e} = \hat{\mathbf{y}} - \mathbf{y} = \mathbf{X}\hat{\boldsymbol{\beta}} - \mathbf{y}$ is the vector of residuals, and n is the number of observations and p is the number of covariates.

Thus our estimate of $\mathbb{V}[\hat{\boldsymbol{\beta}}|\mathbf{X}]$ is

$$\widehat{\mathbb{V}[\hat{\boldsymbol{\beta}}|\mathbf{X}]} = \frac{\mathbf{e}^\top\mathbf{e}}{n-p}(\mathbf{X}^\top\mathbf{X})^{-1}$$

The diagonal of this matrix is our estimated variance for each coefficient, the square root of which is the familiar standard error that we often use to construct confidence intervals or perform null hypothesis significance tests.

Let's see this in `R`

```
## Construct simulated data and errors
set.seed(1)
X <- cbind(1, rnorm(100), runif(100))

set.seed(2)
```

```r
eps <- rnorm(100)

beta <- c(1, 2, 3)
y <- X %*% beta + eps

## Manual solutions
## Beta hat
beta_hat <- solve(t(X) %*% X, t(X) %*% y)
beta_hat
```

```
##          [,1]
## [1,] 1.067999
## [2,] 1.806047
## [3,] 2.821665
```

```r
## Residuals
resid <- y - X %*% beta_hat
## Estimate of sigma_2
sigma2_hat <- (t(resid) %*% resid) / (nrow(X) - ncol(X))
sigma2_hat
```

```
##          [,1]
## [1,] 1.338826
```

```r
## Estimate of V[\hat{\bbeta}]
vcov_beta_hat <- c(sigma2_hat) * solve(t(X) %*% X)
vcov_beta_hat
```

```
##               [,1]          [,2]         [,3]
## [1,]  0.0463264144  0.0001312435 -0.075750093
## [2,]  0.0001312435  0.0168795926 -0.004526778
## [3,] -0.0757500928 -0.0045267783  0.175265100
```

```r
## Estimate of standard errors
sqrt(diag(vcov_beta_hat))
```

```
## [1] 0.2152357 0.1299215 0.4186467
```

This leaves us with the following coefficients and standard error estimates:

```r
cbind(beta_hat, sqrt(diag(vcov_beta_hat)))
```

```
##          [,1]      [,2]
## [1,] 1.067999 0.2152357
## [2,] 1.806047 0.1299215
## [3,] 2.821665 0.4186467
```

Let's show the same thing using `lm`.

```r
lm_out <- lm(y ~ 0 + X)
cbind(lm_out$coefficients, coef(summary(lm_out))[, 2])
```

```
##        [,1]      [,2]
## X1 1.067999 0.2152357
## X2 1.806047 0.1299215
## X3 2.821665 0.4186467
```

Looks good!

Robust Estimation (Heteroskedasticity Constistent Errors)

Sometimes, the assumption that our errors are homoskedastic is unrealistic. If we think there is naturally greater variance for values with a higher value of X, for example. A concrete example could be where income is the outcome and age is the explanatory variable. Among young individuals, income is probably less variable than among older individuals and thus the spread of income around the average income is greater for older individuals than for younger individuals. Another way to think of this is that our observations are still independent, but they are not identically distributed because they have different variance.

Heteroskedasticity is not a problem for coefficients, but it does bias our estimates of the standard errors. We can get White's heteroskedasticity consistent standard errors, or robust standard errors, by assuming something else for the variance-covariance of the errors ($\mathbb{E}[\boldsymbol{\epsilon}\boldsymbol{\epsilon}^\top|\mathbf{X}]$) and choosing a different estimator.

Instead of forcing all diagonal elements of $\mathbb{E}[\boldsymbol{\epsilon}\boldsymbol{\epsilon}^\top|\mathbf{X}]$ to be a single scalar, what if we allow them all to be different? This accounts for all kinds of heteroskedasticity, because each error is allowed to have a different variance. Precisely,

$$\mathbb{E}[\boldsymbol{\epsilon}\boldsymbol{\epsilon}^\top|\mathbf{X}] = \begin{bmatrix} \sigma_1^2 & 0 & \cdots & 0 \\ 0 & \sigma_2^2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \sigma_n^2 \end{bmatrix}$$

Thus we now have n different variances, σ_i^2. Then the variance of our coefficients simplifies,

$$\begin{aligned} \mathbb{V}[\hat{\boldsymbol{\beta}}|\mathbf{X}] &= (\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbb{E}[\text{diag}[\sigma_i^2]|\mathbf{X}]\mathbf{X}(\mathbf{X}^\top\mathbf{X})^{-1} \\ &= (\mathbf{X}^\top\mathbf{X})^{-1}\frac{1}{n}\sum_{i=1}^{n}\sigma_i^2\mathbf{x_i}\mathbf{x_i}^\top(\mathbf{X}^\top\mathbf{X})^{-1} \end{aligned}$$

Then, White shows in his often cited 1980 paper, that, $\frac{1}{n}\sum_{i=1}^{n} e_i^2\mathbf{x_i}\mathbf{x_i}^\top$ is a consistent, but biased, estimator for $\frac{1}{n}\sum_{i=1}^{n}\sigma_i^2\mathbf{x_i}\mathbf{x_i}^\top$ where $\mathbf{x_i}$ is the $p\times 1$ vector of covariates for observation i. So $\mathbb{E}[\mathbf{X}^\top\boldsymbol{\epsilon}\boldsymbol{\epsilon}^\top\mathbf{X}|\mathbf{X}]$ is consistently but biasedly estimated by $\frac{1}{n}\sum_{i=1}^{n} e_i^2\mathbf{x_i}\mathbf{x_i}^\top$. Thus, we can write our estimate for the variance as

$$\widehat{\mathbb{V}[\hat{\boldsymbol{\beta}}|\mathbf{X}]}_{HC} = (\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\text{diag}[e_i^2]\mathbf{X}(\mathbf{X}^\top\mathbf{X})^{-1}$$

To be clear $\text{diag}[e_i^2]$ is a diagonal matrix with each element on the diagonal being observation i's residual squared. All of these quantities are all observed, so we can directly compute the heteroskedasticity robust variance covariance matrix and standard errors.

However, it is now standard to use a finite sample correction for the bias in this estimator. While the estimate is consistent, it is biased and thus when the sample is not infinite, a correction can be used to improve the

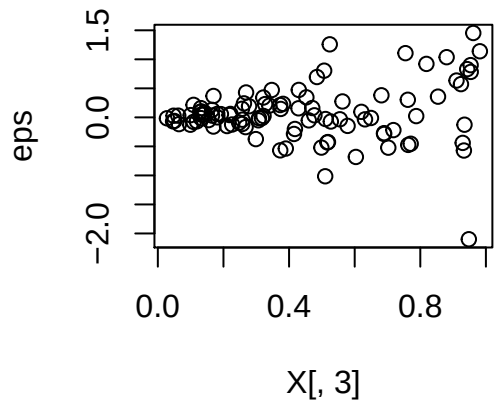
bias. There are several different corrections we can use. A simple one, and the one used by default in Stata, is the `HC1` robust variance covariance matrix. This is simply

$$\widehat{\mathbb{V}[\hat{\boldsymbol{\beta}}|\mathbf{X}]}_{HC1} = \frac{n}{n-p}(\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top \text{diag}[e_i^2]\mathbf{X}(\mathbf{X}^\top\mathbf{X})^{-1}$$

Thus all we are doing is multiplying the elements by $\frac{n}{n-p}$ which will be close to 1 if we have many more observations n than covariates p. However, it is probably preferable to use `HC2` or `HC3`, but I will not go into those here for the sake of simplicity.

Let's do this in `R`:

```
## Noise that is large for higher values of X[, 3]
set.seed(1)
eps <- rnorm(100, 0, sd = X[, 3])
plot(X[, 3], eps)
```




```
y <- X %*% beta + eps

## Manual solutions
## Beta hat
beta_hat <- solve(t(X) %*% X, t(X) %*% y)
beta_hat
```

```
##           [,1]
## [1,] 0.9503923
## [2,] 2.4367714
## [3,] 3.1610179
```

Now let's get the `HC1` robust standard errors.

```
## Residuals
resid <- y - X %*% beta_hat
sigma2_hat <- t(resid) %*% resid / (nrow(X) - ncol(X))
```

```
## Standard, non-robust estimate
vcov_beta_hat <- c(sigma2_hat) * solve(t(X) %*% X)
vcov_beta_hat
```

```
##               [,1]          [,2]          [,3]
## [1,]  2.479749e-03  7.025170e-06 -0.0040547332
## [2,]  7.025170e-06  9.035269e-04 -0.0002423083
## [3,] -4.054733e-03 -2.423083e-04  0.0093815492
```

```
## Robust HC1 stimate of V[\hat{\bbeta}]
vcov_rob_beta_hat <- nrow(X)/(nrow(X) - ncol(X)) *
  solve(t(X) %*% X) %*% t(X) %*% diag(c(resid^2)) %*% X %*% solve(t(X) %*% X)
vcov_rob_beta_hat
```

```
##              [,1]         [,2]         [,3]
## [1,]  0.003743534  0.000355192 -0.008265779
## [2,]  0.000355192  0.003046248 -0.002539765
## [3,] -0.008265779 -0.002539765  0.022678946
```

```
## Display results
outmat <- cbind(beta_hat, sqrt(diag(vcov_beta_hat)),  sqrt(diag(vcov_rob_beta_hat)))
colnames(outmat) <- c("Beta Hat", "Standard SE", "HC1 Robust SE")
outmat
```

```
##       Beta Hat Standard SE HC1 Robust SE
## [1,] 0.9503923  0.04979708    0.06118443
## [2,] 2.4367714  0.03005872    0.05519282
## [3,] 3.1610179  0.09685840    0.15059531
```

We can do this using `lm` and the `sandwich` package.

```
lmout <- lm(y ~ 0 + X)
library(sandwich)
## HC1 Robust
vcov_rob_beta_hat <- vcovHC(lmout, type = "HC1")
## HC2 Robust
vcov_robHC2_beta_hat <- vcovHC(lmout, type = "HC2")
## HC3 Robust
vcov_robHC3_beta_hat <- vcovHC(lmout, type = "HC3")
outmat <- cbind(lmout$coefficients,
                coef(summary(lmout))[, 2],
                sqrt(diag(vcov_rob_beta_hat)),
                sqrt(diag(vcov_robHC2_beta_hat)),
                sqrt(diag(vcov_robHC3_beta_hat)))
colnames(outmat) <- c("Beta Hat",
                      "Standard SE",
                      "HC1 Robust SE",
                      "HC2 Robust SE",
                      "HC3 Robust SE")
outmat
```

```
##     Beta Hat Standard SE HC1 Robust SE HC2 Robust SE HC3 Robust SE
## X1 0.9503923  0.04979708    0.06118443    0.06235143    0.06454567
## X2 2.4367714  0.03005872    0.05519282    0.05704224    0.05989300
## X3 3.1610179  0.09685840    0.15059531    0.15474172    0.16155457
```

The biggest difference is between the regular standard errors and the robust standard errors. The finite corrections are only slightly diferent from one another.

Cluster Robust Estimation

Another problem is that your data may be clustered. You may have groups of observations that belong to certain groups which may mean that there is dependence in the error within groups. We still assume that there is no dependence across groups. For example, imagine that you have survey respondents in 25 countries. The errors are going to be correlated and distributed differently within countries. Or imagine studying the performance of students in different classrooms. Those in the same classroom are likely to receive similar "shocks" or random effects that those in other classrooms will not. We need to account for this clustering in our data.

Again, this is not a problem for our coefficients. However, the variance covariance matrix of the errors now has a clustered structure. Let's imagine we have m groups, and each group has n_m observations. Then we can write the variance covariance matrix of the errors as

$$\mathbb{E}[\boldsymbol{\epsilon}\boldsymbol{\epsilon}^\top|\mathbf{X}] = \begin{bmatrix} \sigma^2_{(1,1)1} & \cdots & \sigma^2_{(1,n_1)1} & 0 & \cdots & 0 & & & \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & & & \\ \sigma^2_{(n_1,1)1} & \cdots & \sigma^2_{(n_1,n_1)1} & 0 & \cdots & 0 & & & \\ 0 & \cdots & 0 & \sigma^2_{(1,1)2} & \cdots & \sigma^2_{(1,n_2)2} & & & \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & & & \\ 0 & \cdots & 0 & \sigma^2_{(n_2,1)2} & \cdots & \sigma^2_{(n_2,n_2)2} & & & \\ & & & & & & \ddots & & \\ & & & & & & & & \\ & & & & & & & & \\ & & & & & & & & \\ & & & & & & & & \end{bmatrix}$$

$$\begin{bmatrix} & & \\ & & \\ & & \\ & & \\ & & \\ & & \\ & & \\ \sigma^2_{(1,1)m} & \cdots & \sigma^2_{(1,n_m)m} \\ \vdots & \ddots & \vdots \\ \sigma^2_{(n_m,n_m)m} & \cdots & \sigma^2_{(n_m,n_m)m} \end{bmatrix}$$

Thus we can write the variance covariance of our coefficients as

$$\mathbb{V}[\hat{\boldsymbol{\beta}}|\mathbf{X}] = (\mathbf{X}^\top\mathbf{X})^{-1}\sum_{g=1}^{m}\mathbf{x_g}^\top\boldsymbol{\epsilon}_g\boldsymbol{\epsilon}_g^\top\mathbf{x_g}(\mathbf{X}^\top\mathbf{X})^{-1}$$

where $\mathbf{x_g}$ is an $n_g \times p$ matrix of all p covariates for the observations in group g and $\boldsymbol{\epsilon}_g$ is an $n_g \times 1$ vector of errors for the n_g observations in group g. So we have this block structure where we have a full variance covariance matrix and we need to estimate the blocks of errors. Without getting into the derivation, we can use $\sum_{g=1}^{m}\mathbf{e}_g\mathbf{e}_g\mathbf{x_g}\mathbf{x_g}^\top$ to estimate $\sum_{g=1}^{m}\boldsymbol{\epsilon}_g\boldsymbol{\epsilon}_g^\top\mathbf{x_g}\mathbf{x_g}^\top$. Thus our estimated variance covariance matrix of the coefficients is

$$\widehat{\mathbb{V}[\hat{\boldsymbol{\beta}}|\mathbf{X}]}_{CR} = (\mathbf{X}^\top\mathbf{X})^{-1}\sum_{g=1}^{m}\mathbf{x_g}^\top\mathbf{e}_g\mathbf{e}_g\mathbf{x_g}(\mathbf{X}^\top\mathbf{X})^{-1}$$

We also apply a finite sample correction to this estimator because it is biased in finite samples. The standard "fancy" corrected estimator that Stata uses is

$$\widehat{\mathbb{V}[\hat{\boldsymbol{\beta}}|\mathbf{X}]}_{CR_{fancy}} = \frac{m}{m-1}\frac{n-1}{n-p}(\mathbf{X}^\top\mathbf{X})^{-1}\sum_{g=1}^{m}\mathbf{x_g}^\top\mathbf{e}_g\mathbf{e}_g\mathbf{x_g}(\mathbf{X}^\top\mathbf{X})^{-1}$$

Again, as m and n go to infinite, the correction will go to 1. This should make it obvious that a small number of clusters will require a bigger correction from the first term.

Let's do this in R.

```
## Generate epsilon from correlated matrix
## 10 groups, same blocks but this is not necessary
library(clusterGeneration)
```

```
## Loading required package: MASS
```

```
library(mvtnorm)
block_eps <- genPositiveDefMat(10)
sigma_eps <- kronecker(diag(10), block_eps$Sigma)
eps <- rmvnorm(1, mean = rep(0, 100), sigma = sigma_eps/4)
groups <- rep(1:10, each = 10)
groups
```

```
##   [1]  1  1  1  1  1  1  1  1  1  1  2  2  2  2  2  2  2  2  2  2  3  3  3
##  [24]  3  3  3  3  3  3  3  4  4  4  4  4  4  4  4  4  4  5  5  5  5  5  5
##  [47]  5  5  5  5  6  6  6  6  6  6  6  6  6  6  7  7  7  7  7  7  7  7  7
##  [70]  7  8  8  8  8  8  8  8  8  8  8  9  9  9  9  9  9  9  9  9  9 10 10
##  [93] 10 10 10 10 10 10 10 10
```

```
y <- X %*% beta + t(eps)

## Manual solutions
## Beta hat
beta_hat <- solve(t(X) %*% X, t(X) %*% y)
beta_hat
```

```
##           [,1]
## [1,] 0.8392765
## [2,] 2.1686256
## [3,] 3.3014213
```

```
## Residuals
resid <- y - X %*% beta_hat
sigma2_hat <- 1/(nrow(X) - ncol(X)) * c(t(resid) %*% resid)
## Standard, non-robust estimate
vcov_beta_hat <- c(sigma2_hat) * solve(t(X) %*% X)
vcov_beta_hat
```

```
##               [,1]          [,2]         [,3]
## [1,]  0.0382446856  0.0001083478 -0.062535349
## [2,]  0.0001083478  0.0139349164 -0.003737073
## [3,] -0.0625353487 -0.0037370734  0.144689779
```

```
## Cluster Robust estimate of V[\hat{\bbeta}]
meat <- matrix(0, nrow = ncol(X), ncol = ncol(X))
for (g in 1:10) {
```

```
  meat <- meat + t(X[groups == g, ]) %*% resid[groups == g] %*%
    t(resid[groups == g]) %*% X[groups == g, ]
}
vcov_crob_beta_hat <- (10/(10-1)) * ((100 - 1)/(100 - 3)) *
  solve(t(X) %*% X) %*% meat %*% solve(t(X) %*% X)
vcov_crob_beta_hat
```

```
##              [,1]         [,2]         [,3]
## [1,]  0.039699729  0.009047246 -0.058446415
## [2,]  0.009047246  0.022368271 -0.005527682
## [3,] -0.058446415 -0.005527682  0.125846996
```

```
## Display results
outmat <- cbind(beta_hat, sqrt(diag(vcov_beta_hat)),  sqrt(diag(vcov_crob_beta_hat)))
colnames(outmat) <- c("Beta Hat", "Standard SE", "Cluster Robust SE")
outmat
```

```
##       Beta Hat Standard SE Cluster Robust SE
## [1,] 0.8392765   0.1955625         0.1992479
## [2,] 2.1686256   0.1180462         0.1495603
## [3,] 3.3014213   0.3803811         0.3547492
```

R does not have a built in function for cluster robust standard errors. But there are scripts online to do this. Let's use one I wrote:

```
## Put data in data.frame
df <- as.data.frame(cbind(y, X, groups))
names(df) <- c("y", "x1", "x2", "x3", "groups")
## Load script for cluster standard errors
source("http://lukesonnet.github.io/teaching/clusterRSE.R")
## Fit model
lmout <- lm(y ~ x2 + x3, data = df)
## Use my custom script for clustered errors, clustered by "groups"
vcov_crob_beta_hat_lm <- clusterRSE(lmout, "groups", df)$vcovCL
```

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

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

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

```
## [1] 1.134021
```

```
## Display results
outmat <- cbind(beta_hat, sqrt(diag(vcov_beta_hat)),  sqrt(diag(vcov_crob_beta_hat_lm)))
colnames(outmat) <- c("Beta Hat", "Standard SE", "Cluster Robust SE")
outmat
```

```
##              Beta Hat Standard SE Cluster Robust SE
## (Intercept) 0.8392765   0.1955625         0.1992479
## x2          2.1686256   0.1180462         0.1495603
## x3          3.3014213   0.3803811         0.3547492
```

Same as above!

Some comments

Why would you use regular standard errors if heteroskedastic standard errors and clustered standard errors both allow for more complicated error structures?

Homoskedasticity is simply a special case of the heteroskedastic error structure; it is simply the case where $\sigma_j = \sigma_i$ for all i and j. So using heteroskedastic standard errors will always handle the case of homoskedasticity and will always be safe in that way. However:

- Regular standard errors do not have finite sample bias. So if we truly believe homoskedasticity to be true, then we can avoid finite sample bias by using the regular standard errors.
- Furthermore, if homoskedasticity actually is true, then our estimates of the standard errors will be more efficient. This means it will approach the true value faster (as the sample size grows), then heteroskedastic standard errors.
- However, we rarely believe that errors actually are homoskedastic, and it is often best to use the heteroskedasticity robust standard errors

Remember, the error structure is not important for unbiasedness of $\hat{\beta}$ as long as it has conditional mean 0

Review your notes for the proof that $\hat{\boldsymbol{\beta}}$ is an unbiased estimator for $\boldsymbol{\beta}$. Never do we use the variance-covariance matrix, $\mathbb{E}[\boldsymbol{\epsilon}\boldsymbol{\epsilon}^\top | X]$. All we use is the conditional mean of $\boldsymbol{\epsilon}$. This whole discussion is about the biasedness of our estimates for $\mathbb{V}[\hat{\boldsymbol{\beta}}]$, which is our estimate of uncertainty and is how we do hypothesis testing.

Normally Distributed Errors

We have been very focused on the variance-covariance of $\boldsymbol{\epsilon}$, but not on how those errors have been distributed. For example, it is often stated that we assume that the errors are **normally distributed**. The normality of the errors is not necessary for unbiasedness of either $\hat{\boldsymbol{\beta}}$ or $\widehat{\mathbb{V}[\hat{\boldsymbol{\beta}}]}$. So why do people make that assumption?

- Normality is somewhat important for significance testing. Specifically, with normal, independent standard errors we can be assured the $\hat{\boldsymbol{\beta}}$ is distributed noormally even in finite samples. This means we can get t-statistic that is actually t distributed. Thus it is important for significance testing, but not for the standard errors themselves. Nonetheless, even without normal errors, $\hat{\boldsymbol{\beta}}$ will still be distributed normally asymptotically, meaning as the sample size goes to ∞. Furthermore, our test statistic will also be normally distributed and thus asymptotically significance testing will also be valid. These are the result of the central limit theorem. This generally means that if you have a very large sample (where "very large" is intentionally vague), then the assumption is not necessary for significance testing. However, in finite samples, the normality assumption guarantees that your confidence intervals and p-values are correct.
- Normality (or perhaps other *specific* distributional assumptions) is necessary for the "best" or minimum variance of the OLS estimator in finite samples.

- Normality of the errors is needed for the standard normal linear model if you fit it using maximum likelihood. You will learn about this later in the sequence; it returns the same coefficients as OLS, but the framework is different. So this is largely about how you conceptualize regression.