200C Section: Standard Errors in OLS

This document is a slight modification of notes from the previous 200C TA, Soonhong Cho, who developed this document based on *Causal Inference in Statistics, Social, and Biomedical Sciences* (2015) by Imbens and Rubin, *Foundations of Agnostic Statistics* (2019) by Aronow and Miller, Kosuke Imai's lecture notes, and Erin Hartman's lecture notes.

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

- Let's start with the variance-covariance matrix of the OLS estimator $\hat{\beta}$.
  - First, note that

$$\begin{aligned}\hat{\beta} &= (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'Y \\ &= (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'(\mathbf{X}\beta+\epsilon) && \text{(model: } Y=\mathbf{X}\beta+\epsilon) \\ &= \underbrace{(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{X}}_{\mathbb{I}_n}\beta + (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\epsilon \\ &= \beta + (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\epsilon\end{aligned}$$

  - Then, we can derive the variance as:

$$\begin{aligned}\mathbb{V}[\hat{\beta}|\mathbf{X}] &= \mathbb{V}[\beta|\mathbf{X}] + \mathbb{V}[(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\epsilon|\mathbf{X}] \\ &= \mathbb{V}[(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\epsilon|\mathbf{X}] && (\beta \text{ is unkown constant}) \\ &= (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbb{V}[\epsilon|\mathbf{X}]((\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}')' && (X \text{ is non-random conditional on } X) \\ &= (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbb{V}[\epsilon|\mathbf{X}]\mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\end{aligned}$$

- It's the general form of the variance of OLS estimator without any assumption. We know everything from data except for $\mathbb{V}[\epsilon|\mathbf{X}]$, so we have to estimate it. This looks like the following:

$$\mathbb{V}[\epsilon|\mathbf{X}] = \begin{bmatrix} \mathbb{V}[\epsilon_1|\mathbf{X}] & \text{Cov}[\epsilon_1,\epsilon_2|X] & \cdots & \text{Cov}[\epsilon_1,\epsilon_n|X] \\ \text{Cov}[\epsilon_2,\epsilon_1|X] & \mathbb{V}[\epsilon_2|\mathbf{X}] & \cdots & \text{Cov}[\epsilon_2,\epsilon_n|X] \\ \vdots & \vdots & \ddots & \vdots \\ \text{Cov}[\epsilon_n,\epsilon_1|X] & \text{Cov}[\epsilon_n,\epsilon_2|X] & \cdots & \mathbb{V}[\epsilon_n|\mathbf{X}] \end{bmatrix}$$

- Why do we need this? We are interested in the error terms because they will tell us something about the uncertainty in our $\hat{\beta}$ parameter estimates. If we are wrong about the error structure, then our estimates will still be accurate based on the data. What will be inaccurate is our certainty about those estimates —- our confidence intervals, p-values, and associated measures of uncertainty. This could be mean that we form conclusions based on noise, which could lead us to accept false conclusions.

1. Standard Variance Estimator under Homoskedasticity

- The standard ("classical") assumption on error structure in OLS is that there are spherical errors. This assumption on the variance-covariance structure includes
  - Homoskedasticity: $\mathbb{E}[\epsilon_i^2|X] = \sigma^2$ (implies $\mathbb{V}(\epsilon_i|X) = \sigma^2$), and
  - No correlation between observations: $\mathbb{E}[\epsilon_i\epsilon_j|X] = 0$ (implies $Cov(\epsilon_i,\epsilon_j) = 0$).

- Under the standard spherical errors assumption, the variance-covariance matrix has the simplest form as following. Let $\sigma_i^2 \equiv \mathbb{V}[\epsilon_i|\mathbf{X}]$ for notational simplicity.

$$\mathbb{V}[\epsilon|\mathbf{X}] = \begin{bmatrix} \mathbb{V}[\epsilon_1|\mathbf{X}] & \text{Cov}[\epsilon_1, \epsilon_2|\mathbf{X}] & \cdots & \text{Cov}[\epsilon_1, \epsilon_n|\mathbf{X}] \\ \text{Cov}[\epsilon_2, \epsilon_1|\mathbf{X}] & \mathbb{V}[\epsilon_2|\mathbf{X}] & \cdots & \text{Cov}[\epsilon_2, \epsilon_n|\mathbf{X}] \\ \vdots & \vdots & \ddots & \vdots \\ \text{Cov}[\epsilon_n, \epsilon_1|\mathbf{X}] & \text{Cov}[\epsilon_n, \epsilon_2|\mathbf{X}] & \cdots & \mathbb{V}[\epsilon_n|\mathbf{X}] \end{bmatrix}$$

$$= \begin{bmatrix} \sigma_1^2 & 0 & \cdots & 0 \\ 0 & \sigma_2^2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \sigma_n^2 \end{bmatrix} \qquad \text{(no correlation between obs: } \text{Cov}(\epsilon_i, \epsilon_j) = 0\text{)}$$

$$= \begin{bmatrix} \sigma^2 & 0 & \cdots & 0 \\ 0 & \sigma^2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \sigma^2 \end{bmatrix} \qquad \text{(homoskedasticity: } \mathbb{V}[\epsilon_i|\mathbf{X}] = \sigma_i^2 = \sigma^2\text{)}$$

$$= \sigma^2 \mathbb{I}_n$$

- Then the variance of $\hat{\beta}$ is:

$$\begin{aligned} \mathbb{V}[\hat{\beta}|\mathbf{X}] &= (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}' \underbrace{\mathbb{V}[\epsilon|\mathbf{X}]}_{\sigma^2\mathbb{I}_n} \mathbf{X}(\mathbf{X}'\mathbf{X})^{-1} \\ &= \sigma^2(\mathbf{X}'\mathbf{X})^{-1} \underbrace{\mathbf{X}'\mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}}_{\mathbb{I}_n} \\ &= \sigma^2(\mathbf{X}'\mathbf{X})^{-1} \end{aligned}$$

- Since we don't know the population variance σ^2, we estimate it with some sample quantities, using the OLS residuals $\hat{\epsilon} = \mathbf{e} = Y - \mathbf{X}\hat{\beta}$. The (unbiased) estimator is:

$$\hat{\sigma}^2 = \frac{\mathbf{e}'\mathbf{e}}{n-k},$$

  where n is the sample size and k is the number of regressors in the model. That is, we are using some kind of average squared error (sum of squared errors divided by the number of observations with a little correction) to estimate the variance. The correction part is just like the bessel correction for sample standard deviation (remember wed divided something by $n-1$ instead of n) where we account for the degrees of freedom the model parameters "ate up."

- Our final estimator for the variance of $\hat{\beta}$ is:

$$\widehat{\mathbb{V}[\hat{\beta}|\mathbf{X}]} = \frac{\mathbf{e}'\mathbf{e}}{n-k}(\mathbf{X}'\mathbf{X})^{-1}.$$

  Taking square root of the diagonal terms of the this matrix will give the standard error estimate.

```
library(tidyverse)
library(knitr)
library(kableExtra)
set.seed(123)
```

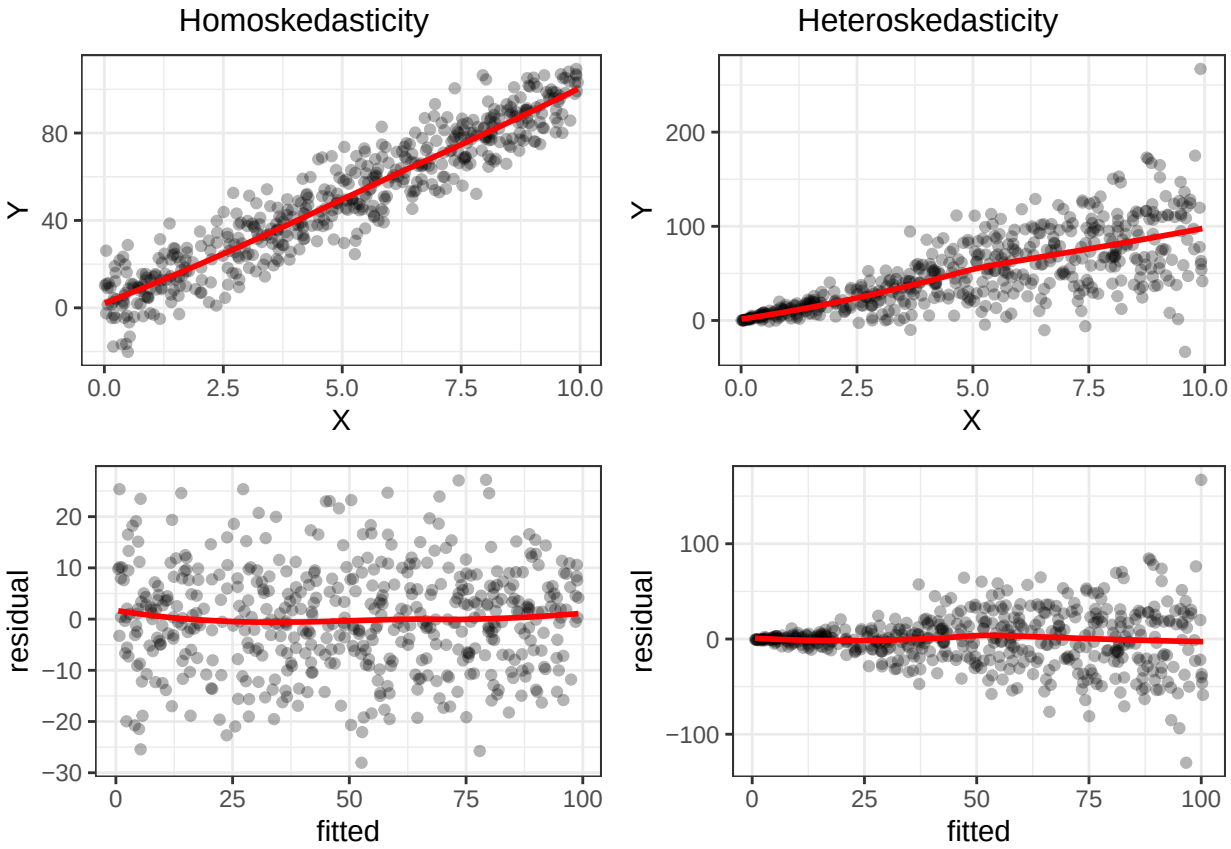
lm_est	lm_se	manual_est	manual_se
0.978	0.181	0.978	0.181
1.868	0.103	1.868	0.103
3.010	0.032	3.010	0.032

```r
#simulated data
X <- cbind(1,
           rnorm(100),
           runif(100, 0, 10))
epsilon <- rnorm(100)
true_beta = c(1, 2, 3)
Y <- X %*% true_beta + epsilon

#R's built-in `lm` function
lm_res <- lm(Y ~ X - 1) #exclude default intercept by `lm` as we have intercept column in X
lm_coef <- summary(lm_res)$coefficients[, 1:2]

#manually
beta_hat <- solve(t(X) %*% X) %*% (t(X) %*% Y)
residuals <- Y - X %*% beta_hat
sigma2_hat <- (t(residuals) %*% residuals) / (nrow(X) - ncol(X)) #degree of freedom correction by (n-k)
vcov_beta_hat <- c(sigma2_hat) * solve(t(X) %*% X) #force `sigma2_hat` to be a vector than matrix
se_hat <- sqrt(diag(vcov_beta_hat))
```

2. Heteroskedasticity-Robust Variance Estimator

- (Example) Homoskedastic errors assumption is too unrealistic. A famous example is the relationship between food expenditures and income: grad student with lower incomes, their food expenditures are often restricted due to their budget constraint (little variance). But, as incomes increase, they can afford to spend more on food (more variance). When their wealth goes beyond some threshold, they have leeway to decide how much they want to spend on food: either stick to their cheap diet or try luxurious restaurants everyday (much more variance). Therefore, there is a greater variance in food expenditures of wealthier people relative to lower-income individuals.
- Here we contrast heteroskedasticity to homoskedasticity: to plot Y against X, or the predicted values (by model) against residuals (you could use some formal tests like Breush-Pagan, Cook-Weisbert, White, etc., to detect heteroskedasticity).

```r
homoskedastic <- tibble(X = runif(n = 500, 0, 10),
                        Y = 10*X + rnorm(n = 500, 0, 10)) %>%
  mutate(fitted = fitted(lm(Y ~ X)), residual = residuals(lm(Y ~ X)))

hetero <- tibble(X=homoskedastic$X,
                 Y = 10*X + rnorm(n = 500, 0, 5*X)) %>%  #sd is a function of X!
  mutate(fitted = fitted(lm(Y ~ X)), residual = residuals(lm(Y ~ X)))
```



- (Problem) Heteroskedasticity does not cause bias in $\hat{\beta}$, but it does cause bias and inconsistency in our estimates of the standard error. Why? Because when we estimate σ^2 using the residuals, we place equal weight on each despite the fact that the more precisely measured or less varying observations should convey more information. Then any statistical inference on $\hat{\beta}$ (e.g., Is it statistically significant?) is incorrect. The degree of this problem depends on how serious the heteroskedasticity is.

- (Solution) First, if you know the exact pattern of heteroskedasticity, it is possible to use **Weighted Least Squares (WLS)** to compensate for it: this is akin to telling OLS to count specific observations for less or more. For example, if heteroskedasticity were known up to a multiplicative constant: $\mathbb{V}[\epsilon_i|\mathbf{X}] = a_i\sigma^2$, where $a_i = a_i(\mathbf{x_i})$ is a positive and known function of $\mathbf{x_i}$. Then WLS is just an OLS with transforming the outcome, multiplying y_i by $\frac{1}{\sqrt{a_i}}$: it rescales errors to $\frac{\epsilon_i}{\sqrt{a_i}}$, so that makes the error variance homoskedastic as

$$\mathbb{V}\left[\frac{1}{\sqrt{a_i}}\epsilon_i \middle| \mathbf{X}\right] = \frac{1}{a_i}\mathbb{V}[\epsilon_i|\mathbf{X}] = \frac{1}{a_i}a_i\sigma^2 = \sigma^2.$$

  But in general you would be misspecifying the weights ($a_i(\curvearrowleft_i)$ is rarely known), and this approach has fallen out of favor.

- **[Robust SEs]** We prefer standard error estimators that are heteroskedasticity-consistent. They give consistent (converging toward the correct answer) estimates under heteroskedasticity. The covariance matrix now looks like:

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

hence we are stuck with this expression for the variance of $\hat{\beta}$ as

$$\mathbb{V}[\hat{\beta}|\mathbf{X}] = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbb{V}[\epsilon|\mathbf{X}]\mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}.$$

- The idea of White (1980) is the plug-in principle: if we can consistently estimate the components of $\mathbb{V}[\epsilon|\mathbf{X}]$, we can directly use this expression by replacing $\mathbb{V}[\epsilon|\mathbf{X}]$ with its estimate $\widehat{\mathbb{V}[\epsilon|\mathbf{X}]}$. And we use the residuals to estimate each error term. How? It seems hopeless estimating all the unknown quantities in $\mathbb{V}[\epsilon|\mathbf{X}]$ at first since it's ridiculous to estimate $(n \times n)$ matrix with only n observations, but the key insight is:

$$\begin{aligned}\mathbb{V}[\hat{\beta}|\mathbf{X}] &= (\mathbf{X}'\mathbf{X})^{-1} \underbrace{\underset{k\times n}{\mathbf{X}'}\ \underset{n\times n}{\mathbb{V}[\epsilon|\mathbf{X}]}\ \underset{n\times k}{\mathbf{X}}}_{k\times k} (\mathbf{X}'\mathbf{X})^{-1} \\ &= (\mathbf{X}'\mathbf{X})^{-1}\frac{1}{n}\sum_{i=1}^{n}\sigma_i^2\mathbf{x}_i\mathbf{x}_i^T(\mathbf{X}'\mathbf{X})^{-1},\end{aligned}$$

  where we need to estimate way fewer parameters with $k << n$! Then we can estimate $\mathbb{V}[\hat{\beta}|\mathbf{X}]$ by replacing σ_i^2 with squared residuals e_i^2 which are all observed after fitting the model.

- Then we have:

$$\widehat{\mathbb{V}[\hat{\beta}|\mathbf{X}]}_{\text{HC0}} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\begin{bmatrix} e_1^2 & 0 & \cdots & 0 \\ 0 & e_2^2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & e_n^2 \end{bmatrix}\mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}$$

  as a consistent estimator of $\mathbb{V}[\hat{\beta}|\mathbf{X}]$ under any form of heteroskedasticity. It is called Heteroskedasticity Consistent (HC) variance estimator or Eicker-Huber-White Robust variance estimator. It is consistent—and thus justified asymptotically—but it's biased so it is good to have an estimator which corrects for the bias in the estimator in finite samples.

- **[Sandwich Estimator]** The so-called "sandwich" estimator formula offers a general formula for various HC estimators, with different finite-sample corrections:

$$\begin{aligned}\underbrace{\widehat{\mathbb{V}(\hat{\beta}|\mathbf{X})}}_{\text{sandwich}} &= \underbrace{(\mathbf{X}'\mathbf{X})^{-1}}_{\text{bread}}\underbrace{\mathbf{X}'\widehat{\mathbb{V}[\epsilon|\mathbf{X}]}\mathbf{X}}_{\text{meat}}\underbrace{(\mathbf{X}'\mathbf{X})^{-1}}_{\text{bread}} \\ &= (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\text{diag}(\hat{\epsilon}_i^2)\mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\end{aligned}$$

  - **HC0**: When the "meat" matrix is filled with squared residuals in diagonal

$$\mathbf{X}'\text{diag}(e_i^2)\mathbf{X}$$

    as White (1980) proposed, it's labeled HC0.

  - **HC1**: The meat for HC1 is

$$\frac{n}{n-k}\mathbf{X}'\text{diag}(e_i^2)\mathbf{X},$$

    where n is size and k is the number of parameters. The more data available relative to the number of parameters estimated, the closer it is to 1.

  - **HC2**: The meat for HC2 is

$$\mathbf{X}'\text{diag}\left(\frac{e_i^2}{1-h_{ii}}\right)\mathbf{X},$$

    where h_{ii} is the diagonal terms of the "projection matrix" (or "hat matrix") $\mathbf{P}_{\mathbf{X}} \equiv \mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'$. We typically interpret these terms as "leverage," how much each individual observation is an outlier or produces leverage.

- **HC3**: Almost same as HC2, but now the meat is

$$\mathbf{X}'\text{diag}\left(\frac{e_i^2}{1-h_{ii}^2}\right)\mathbf{X}.$$

  It has the squared leverage in it so that gives more weights to the highest leverage observations.

- In finite samples, these estimators are valid. The standard for modern empirical work is using HC2 standard errors.

- But don't forget this: they DO NOT, nor do any variance estimators affect the point estimates of $\hat{\beta}$. All they're doing is adjusting the standard errors associated with $\hat{\beta}$, not the estimation of it.

```
## HC0 estimator
#1. manually (using our data from before)
beta_hat_hc0 <- solve(t(X) %*% X) %*% t(X) %*%
  diag(c(residuals^2)) %*% #meat!
  X %*% solve(t(X) %*% X)
se_hc0 <- sqrt(diag(beta_hat_hc0))

#2. traditional(?) procedure using `sandwich` package
library(sandwich)
library(lmtest)
#sqrt(diag(vcovHC(lm_res, type = "HC0"))) #only SEs
coeftest(lm_res, vcov=vcovHC(lm_res, type = "HC0"))
```

```
##
## t test of coefficients:
##
##    Estimate Std. Error  t value  Pr(>|t|)
## X1 0.978112   0.162015   6.0372 2.878e-08 ***
## X2 1.868484   0.087108  21.4502 < 2.2e-16 ***
## X3 3.010065   0.028872 104.2558 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
#3. even more convenient
library(estimatr)
lm_hc0 <- lm_robust(Y ~ X - 1, se_type = "HC0")

## HC1 estimator
#1. manually (using our data from before)
scaling_coefficient = nrow(X) / (nrow(X) - ncol(X))
beta_hat_hc1 <- scaling_coefficient * solve(t(X) %*% X) %*% t(X) %*%
  diag(c(residuals^2)) %*% #meat!
  X %*% solve(t(X) %*% X)
se_hc1 <- sqrt(diag(beta_hat_hc1))

#or `estimatr` package
lm_hc1 <- lm_robust(Y ~ X - 1, se_type = "HC1")

## HC2
lm_hc2 <- lm_robust(Y ~ X - 1, se_type = "HC2") #HC2 is default for `lm_robust` function

## HC3
lm_hc3 <- lm_robust(Y ~ X - 1, se_type = "HC3")
```

HC0	HC1	HC2	HC3
0.1620	0.1645	0.1646	0.1673
0.0871	0.0884	0.0890	0.0909
0.0289	0.0293	0.0293	0.0298

```r
#comparison
bind_cols(lm_hc0$std.error, lm_hc1$std.error, lm_hc2$std.error, lm_hc3$std.error) %>%
  kable(digits=4, booktabs = T, align = rep('c', 4),
        col.names = c("HC0", "HC1", "HC2", "HC3")) %>%
  kable_styling(full_width = F, position = "center", font_size = 15)
```

NOTE: Neyman variance estimator and HC2

- Recall that we use the Neyman variance estimator as a conservative estimator of the standard errors for the Difference-in-Means estimator for ATE:

$$\hat{\mathbb{V}}_{\text{Neyman}}(\hat{\tau}) = \frac{\widehat{\mathbb{V}(Y_{1i})}}{n_1} + \frac{\widehat{\mathbb{V}(Y_{0i})}}{n_0},$$

  where $\widehat{\mathbb{V}(Y_{1i})} = \hat{\sigma}^2_{Y|D_i=1}$ is the sample variance of Y_i for the treated units and $\widehat{\mathbb{V}(Y_{0i})} = \hat{\sigma}^2_{Y|D_i=0}$ is the sample variance of Y_i for the control units. "Conservative" means standard error is always larger than the true standard error ($\widehat{SE}_{\hat{\tau}_{ATE}} \geq SE_{\hat{\tau}_{ATE}}$ of SATE; proof is in Chad's slides) so that we need more evidence (larger effect size estimate) to make a causal claim.

- How does this variance match up with the heteroskedastic SEs? Note that the standard error for the parameter estimate (DiM – the analogue in regression is β) allows for differing variances in treatment and control. With some derivations, it turns out that the "HC2" heteroskedasticity-robust variance estimator is numerically equal to the Neyman variance estimator (Samii and Aronow 2012):

$$\hat{\mathbb{V}}_{HC2}(\hat{\beta}) = \frac{\hat{\sigma}^2_{Y|D_i=1}}{n_1} + \frac{\hat{\sigma}^2_{Y|D_i=0}}{n_0} \equiv \hat{\mathbb{V}}_{\text{Neyman}}(\hat{\tau}_{ATE})$$

  Actually, both 1) allow variance to differ between treatment group and control group ("heteroskedasticity" in regression context or "unequal variance" in t-test context), and 2) assume the two groups are independent (no correlation between units in regression / no covarinace term in Neyman variance estimator). In addition to relative efficiency in large samples and smaller bias in small samples (than HC0 and HC1), this similarity to design-based randomization estimators justifies the use of HC2.

- Summary: thus, in a completely randomized experiment, you can analyze the data simply 1) regression the outcome Y_i on D_i and get the coefficient estimate on D_i (it's our estimate for ATE), 2) obtain the robust standard error (HC2 is default), which relaxes unrealistic homogeneous treatment effect, and 3) conduct statistical inference (t-test, confidence intervals, etc.)

3. Cluster-Robust Variance Estimator

- Back to the "spherical errors" assumption. We reviewed how to relax the first part, homoskedasticity. Then, what if we want to relax the second part, "no correlation between observations"? The violation of iid/random sampling can occur when data have "clusters." With clustering, unit-level errors of within the same cluster are correlated, even if we randomly sample clusters.

- Examples of clustering, where units belong to a cluster:

  – Randomly sample households and randomly assign them to different treatment conditions, but the measurement of voter turnout is at the individual level, then household is cluster.

- Randomly sample districts but measure outcomes among individual schools, then district is cluster.
- Randomly sample classrooms but measure outcomes among every student in classroom, then classroom is cluster.
- Conduct survey across many countries but measure outcomes at individual level, then country is cluster.

- Let's begin with a simple linear model with two error components (assuming homoskedastic errors across clusters, $\sigma_j^2 = \sigma^2$, for simplicity):

$$\begin{aligned} y_{ij} &= \beta_0 + \beta_1 x_{ij} + \epsilon_{ij} \\ &= \beta_0 + \beta_1 x_{ij} + \underbrace{v_j}_{\text{cluster}} + \underbrace{u_{ij}}_{\text{unit}}, \end{aligned}$$

where

$$v_j \overset{iid}{\sim} N(0, \rho\sigma^2) \text{ denotes cluster-level error component}$$

and

$$u_{ij} \overset{iid}{\sim} N(0, (1-\rho)\sigma^2) \text{ denotes unit-level error component.}$$

The two components v_j and u_{ij} are assumed to be independent of each other.

- $\rho \in (0, 1)$ is called the within-cluster correlation (or intracluster correlation coefficient, "ICC"), which measures the degree of clustered dependence of units. When $\rho = 1$, all units within a cluster are considered to be identical. When $\rho = 0$, there is no correlation of units within a cluster, and all observations are considered to be independent of each other, so the standard iid setting.
- σ^2 is the variance of the composite error:

$$\begin{aligned} \mathbb{V}[\epsilon_{ij}] &= \mathbb{V}[v_j + u_{ij}] \\ &= \mathbb{V}[v_j] + \mathbb{V}[u_{ij}] \\ &= \underbrace{(\rho\sigma^2)}_{\text{cluster}} + \underbrace{(1-\rho)\sigma^2}_{\text{unit}} = \sigma^2 \end{aligned}$$

- The covraince between two units i and k in the same cluster j is $\rho\sigma^2$:

$$\text{Cov}[\epsilon_{ij}, \epsilon_{kj}] = \rho\sigma^2$$

- The covraince between two units i and k in different clusters j and m is assumed zero:

$$\text{Cov}[\epsilon_{ij}, \epsilon_{km}] = 0$$

- Also, we could allow heteroskedasticity across clusters: in that case the variance for all units i in cluster j is $\mathbb{V}[\epsilon_{ij}|\mathbf{X}] = \sigma_j^2$
- Under our simple setting, the variance-covariance matrix of the error would be block diagonal:

$$\mathbb{V}[\epsilon|\mathbf{X}] = \begin{bmatrix} \sigma^2 & \cdots & \rho\sigma^2 & 0 & \cdots & 0 & & & \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & & & \\ \rho\sigma^2 & \cdots & \rho\sigma^2 & 0 & \cdots & 0 & & & \\ 0 & \cdots & 0 & \sigma^2 & \cdots & \rho\sigma^2 & & & \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & & & \\ 0 & \cdots & 0 & \rho\sigma^2 & \cdots & \sigma^2 & & & \\ & & & & & & \ddots & & \\ & & & & & & & \sigma^2 & \cdots & \rho\sigma^2 \\ & & & & & & & \vdots & \ddots & \vdots \\ & & & & & & & \rho\sigma^2 & \cdots & \sigma^2, \end{bmatrix}$$

where more general expression could have been possible (e.g., each elements being $\sigma^2_{(a,b)j}$ where a, b are row/column indices of units in cluster j, etc.) but I suppressed every subscripts to focus on the structure.

- [Estimation] The idea is the same as heteroskedasticity-robust estimators. With sandwich formula, filling in meat matrix with observed residuals (based on plug-in principle), and some finite-sample modifications. The estimator in a general form without any finite-sample correction looks like:

$$\widehat{\mathbb{V}_{\text{CR}}[\hat{\beta}|X]} = (\mathbf{X}'\mathbf{X})^{-1} \left(\sum_{j=1}^{J} \mathbf{X}_j' \hat{\epsilon}_j \hat{\epsilon}_j' \mathbf{X}_j \right) (\mathbf{X}'\mathbf{X})^{-1},$$

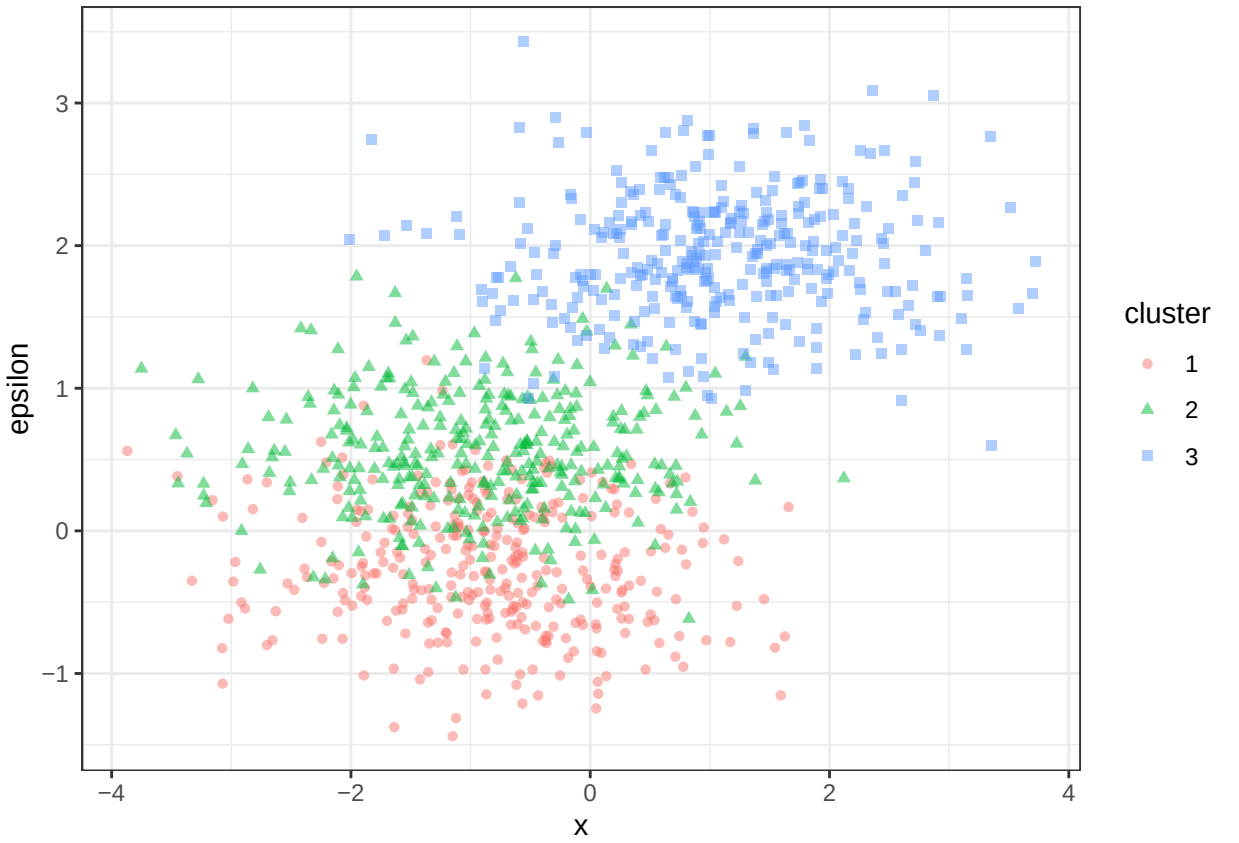
where $\mathbf{X}_j$ is an (#units in cluster $j \times$ # covariates) matrix and $\hat{\epsilon}_j$ is an (#units in cluster $j \times 1$) vector of residuals for the cluster j. Then we could incorporate some small sample bias adjustments using number of units/parameters or hat matrix, like HC series.

I know, it's a huge mess. So let's practice with R coding to get intuition on what's going on. Here we suppose a simple regression model

$$y_i = \beta_0 + \beta_1 X_i + \epsilon_i,$$

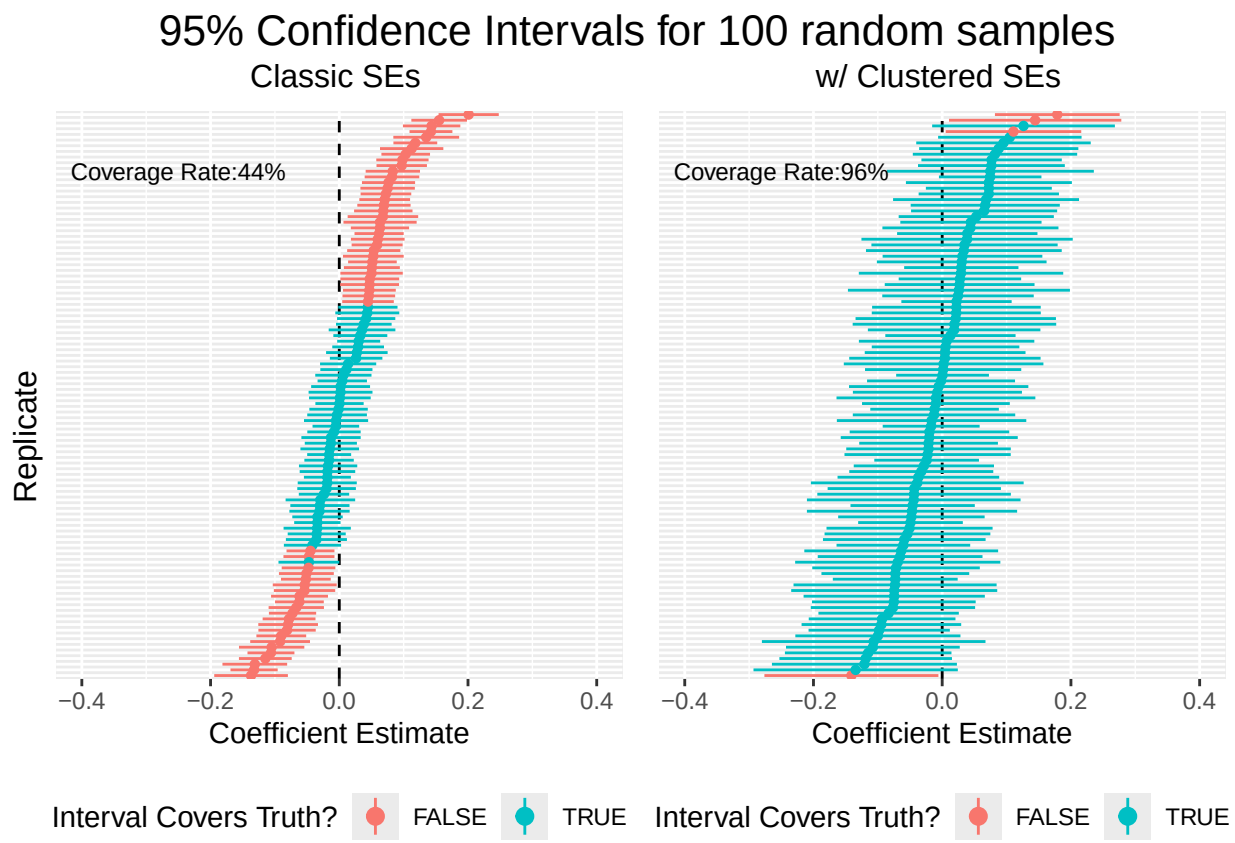
with assuming normal errors.

```
library(mvtnorm) #for Multivariate normal

#make a function to generate clustered data: equal cluster size for simplicity
dgf_cluster <- function(pars = c(-.3, 1), n = 1000, n_cluster = 50, rho = .5){
    #create clusters
    clusters <- rep(1:n_cluster, each = n / n_cluster)

    ##we let cluster/unit errors have two components: for one part to be correlated with
    ##predictor, the other part for pure error term (so use `rmvnorm`)

    #cluster error component
    #draw errors from bivariate normal (x, y): 1st (4th) element of Sigma matrix determines x (y) scale
    v_j <- rmvnorm(n = n_cluster, sigma = matrix(c(1, 0, 0, 1*rho), ncol = 2))

    #unit error component
    u_ij <- rmvnorm(n=n, sigma = matrix(c(1, 0, 0, (1 - rho)), ncol = 2))

    #correlate X with errors
    x <- (u_ij[ , 1] + rep(v_j[ , 1], each = n / n_cluster))

    #epsilon_ij = v_j + u_ij
    epsilon <- u_ij[ , 2] + rep(v_j[ , 2], each = n / n_cluster)

    #make outcome according to linear model
    y <- pars[1] + pars[2]*x + epsilon

    return(tibble(x, y, epsilon, cluster = as_factor(clusters)))
}

#plot example clustered pattern
dgf_cluster(n=999,  n_cluster = 3, rho=.8) %>%
  ggplot(aes(x=x, y=epsilon, col=cluster, shape=cluster)) +
  geom_point(alpha=.5) +
  theme_bw()
```




```
#OLS simulation function w/ or w/o cluster
sim_cluster <- function(rep=100, pars=c(-.3, 1), n=1000, n_cluster=50,
                        rho=.8, clustered=FALSE, cluster_col=cluster){
  replicate(rep, { #use replicate function to do r times
    #create a dataset
    sim_data <- dgf_cluster(pars=pars, n=n , n_cluster=n_cluster, rho=rho)
    if(!clustered){ #default is lm w/o clusterering
       fit <- lm_robust(y ~ x, data=sim_data) #
    }else{
       fit <- lm_robust(y ~ x, data=sim_data, clusters=cluster) #clustered SEs (default: "CR2")
    }
    tibble(est = fit$coefficients["x"], se  = fit$std.error["x"],
           ci_lower = fit$conf.low["x"], ci_upper =fit$conf.high["x"])
  }, simplify = FALSE) %>% bind_rows() %>%
  rownames_to_column(var = "replicate") %>%
  mutate(cover = ci_lower < pars[2] & ci_upper > pars[2]) #coverage of true mean
  #  mutate(coverage = mean(cover))
}

sim_pars <- c(.4, 0) #true: no treatment effect of X on Y
sim_res <- sim_cluster(pars=sim_pars) #w/o cluster
sim_res_cluster <- sim_cluster(pars=sim_pars, clustered=TRUE) #w/ cluster

ci_plot <- function(data, ...){
  ggplot(data, aes(x = reorder(replicate, est), y = est,
                                      ymin = ci_lower, ymax = ci_upper, color = cover)) +
```

```
  geom_hline(yintercept = sim_pars[2], linetype = "dashed") +
  #geom_pointrange(): draws point ranges and a value for point; useful for est. and CIs in one command
  geom_pointrange(fatten = 1) +
  scale_colour_discrete(name  ="Interval Covers Truth?") +
  scale_y_continuous(limits = c(-.4, .4)) + labs(y="Coefficient Estimate") +
  coord_flip() +
  annotate(geom="text", x=90, y=-.25, size=3,
           label=paste0("Coverage Rate:", round(mean(data$cover), 3)*100, "%")) +
  theme(axis.text.y=element_blank(), axis.ticks.y=element_blank(), axis.title.y=element_blank(),
                     legend.position = "bottom")
}

#plot
library(grid)
grid.arrange( arrangeGrob(ci_plot(rep=1000, sim_res %>% sample_n(100)), top="Classic SEs"),
              arrangeGrob(ci_plot(rep=1000, sim_res_cluster %>% sample_n(100)), top="w/ Clustered SEs")
              top = textGrob("95% Confidence Intervals for 100 random samples", gp=gpar(fontsize=16)),
              left = textGrob("Replicate", rot = 90, vjust = 1),
              #labs(x = "Replicate", y = "Coef Est.") +
              ncol=2)
```



As the left plot shows, more than 5% of the CIs (56%!!!) does not include the true values, meaning that we reject the null of no effect more than 5% of the time (more chance of getting "significant" results). That is, if we analyze clustered-data with OLS, we may incorrectly reject the null hypothesis more than we should since the standard error is not clustered. When we cluster the standard errors, the CIs achieve the target coverage rate 95% as the right plot suggests. This contrast warns us that STANDARD ERRORS MATTER.

```r
# clustering at district level
library(survey)
data(api)

# 15 districts
unique(apiclus1$dnum)

# estimate with clustered SEs
# set clusters = dnum
lm_robust(api00 ~ enroll+meals+full, data = apiclus1, clusters = dnum, se_type = "CR2")
```

- Let's wrap up. We've talked about what assumptions go into what SEs and which SEs are appropriate in what circumstance.
    1. **Regular SEs**: No correlation between observations and Homoskedasticity.
    2. **HC Robust SEs**: No correlation between observations but allowing Heteroskedasiticity.
    3. **Cluster Robust SEs**: Allowing within-cluster correlation between units but assuming no correlation across clusters. Typically also allowing heteroskedasticity across clusters.