Robust and Clustered Standard Errors

Molly Roberts

March 6, 2013

Outline

1. An Introduction to Robust and Clustered Standard Errors
    - Linear Regression with Non-constant Variance
    - GLM's and Non-constant Variance
    - Cluster-Robust Standard Errors
2. Replicating in R

Outline

1. An Introduction to Robust and Clustered Standard Errors
   - Linear Regression with Non-constant Variance
   - GLM's and Non-constant Variance
   - Cluster-Robust Standard Errors
2. Replicating in R

Review: Errors and Residuals

Review: Errors and Residuals

Errors are the vertical distances between observations and the unknown Conditional Expectation Function. Therefore, they are unknown.

Review: Errors and Residuals

Errors are the vertical distances between observations and the unknown Conditional Expectation Function. Therefore, they are unknown.

Residuals are the vertical distances between observations and the estimated regression function. Therefore, they are known.

Notation

Errors represent the difference between the outcome and the true mean.

Notation

Errors represent the difference between the outcome and the true mean.

$$\mathbf{y} = \mathbf{X}\beta + \mathbf{u}$$
$$\mathbf{u} = \mathbf{y} - \mathbf{X}\beta$$

Residuals represent the difference between the outcome and the estimated mean.

$$\mathbf{y} = \mathbf{X}\hat{\beta} + \hat{\mathbf{u}}$$
$$\hat{\mathbf{u}} = \mathbf{y} - \mathbf{X}\hat{\beta}$$

Notation

Errors represent the difference between the outcome and the true mean.

$$\mathbf{y} = \mathbf{X}\beta + \mathbf{u}$$
$$\mathbf{u} = \mathbf{y} - \mathbf{X}\beta$$

Notation

Errors represent the difference between the outcome and the true mean.

$$\mathbf{y} = \mathbf{X}\beta + \mathbf{u}$$
$$\mathbf{u} = \mathbf{y} - \mathbf{X}\beta$$

Residuals represent the difference between the outcome and the estimated mean.

Notation

Errors represent the difference between the outcome and the true mean.

$$\mathbf{y} = \mathbf{X}\beta + \mathbf{u}$$
$$\mathbf{u} = \mathbf{y} - \mathbf{X}\beta$$

Residuals represent the difference between the outcome and the estimated mean.

$$\mathbf{y} = \mathbf{X}\hat{\beta} + \hat{\mathbf{u}}$$
$$\hat{\mathbf{u}} = \mathbf{y} - \mathbf{X}\hat{\beta}$$

Variance of $\hat{\beta}$ depends on the errors

$$\begin{aligned}\hat{\beta} &= \left(\mathbf{X}'\mathbf{X}\right)^{-1}\mathbf{X}'\mathbf{y} \\ &= \left(\mathbf{X}'\mathbf{X}\right)^{-1}\mathbf{X}'(\mathbf{X}\beta + \mathbf{u}) \\ &= \beta + \left(\mathbf{X}'\mathbf{X}\right)^{-1}\mathbf{X}'\mathbf{u}\end{aligned}$$

Variance of $\hat{\beta}$ depends on the errors

$$
\begin{aligned}
V[\hat{\beta}] &= V[\beta] + V[(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{u}] \\
&= \mathbf{0} + V[(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{u}] \\
&= E[(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{u}\mathbf{u}'\mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}] - E[(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{u}]E[(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{u}]' \\
&= E[(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{u}\mathbf{u}'\mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}] - \mathbf{0}
\end{aligned}
$$

Variance of $\hat{\beta}$ depends on the errors (continued)

$$
\begin{aligned}
V[\hat{\beta}] &= V[\beta] + V[(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{u}] \\
&= \mathbf{0} + V[(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{u}] \\
&= E[(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{u}\mathbf{u}'\mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}] - E[(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{u}]E[(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{u}]' \\
&= E[(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{u}\mathbf{u}'\mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}] - \mathbf{0} \\
&= (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'E[\mathbf{u}\mathbf{u}']\mathbf{X}(\mathbf{X}'\mathbf{X})^{-1} \\
&= (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{\Sigma}\mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}
\end{aligned}
$$

Constant Error Variance and Dependence

Under standard OLS assumptions,

Constant Error Variance and Dependence

Under standard OLS assumptions,

$$\mathbf{u} \sim N_n(0, \boldsymbol{\Sigma})$$

Constant Error Variance and Dependence

Under standard OLS assumptions,

$$\mathbf{u} \sim N_n(0, \boldsymbol{\Sigma})$$

$$\boldsymbol{\Sigma} = Var(\mathbf{u}) = E[\mathbf{u}\mathbf{u}'] = \begin{bmatrix} \sigma^2 & 0 & 0 & \ldots & 0 \\ 0 & \sigma^2 & 0 & \ldots & 0 \\ & & & \vdots & \\ 0 & 0 & 0 & \ldots & \sigma^2 \end{bmatrix}$$

Constant Error Variance and Dependence

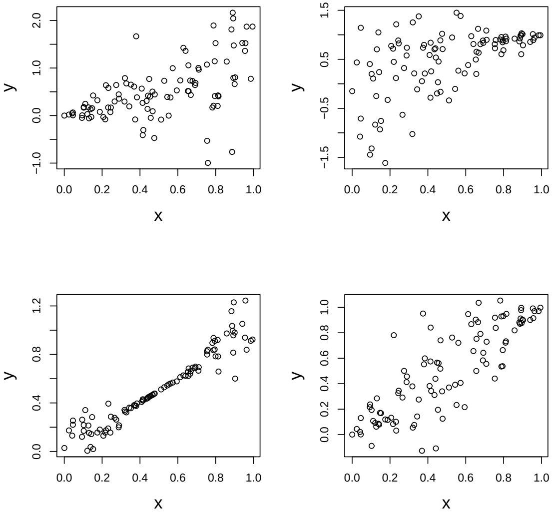
Under standard OLS assumptions,

$$\mathbf{u} \sim N_n(0, \boldsymbol{\Sigma})$$

$$\boldsymbol{\Sigma} = Var(\mathbf{u}) = E[\mathbf{u}\mathbf{u}'] = \begin{bmatrix} \sigma^2 & 0 & 0 & \dots & 0 \\ 0 & \sigma^2 & 0 & \dots & 0 \\ & & & \vdots & \\ 0 & 0 & 0 & \dots & \sigma^2 \end{bmatrix}$$

What does this mean graphically for a CEF with one explanatory variable?

Evidence of Non-constant Error Variance (4 examples)



Notation

The constant error variance assumption sometimes called homoskedasiticity states that

Notation

The constant error variance assumption sometimes called homoskedasiticity states that

$$Var(\mathbf{u}) = E[\mathbf{u}\mathbf{u}'] = \begin{bmatrix} \sigma^2 & 0 & 0 & \dots & 0 \\ 0 & \sigma^2 & 0 & \dots & 0 \\ & & & \vdots & \\ 0 & 0 & 0 & \dots & \sigma^2 \end{bmatrix}$$

Notation

The constant error variance assumption sometimes called homoskedasiticity states that

$$Var(\mathbf{u}) = E[\mathbf{u}\mathbf{u}'] = \begin{bmatrix} \sigma^2 & 0 & 0 & \dots & 0 \\ 0 & \sigma^2 & 0 & \dots & 0 \\ & & & \vdots & \\ 0 & 0 & 0 & \dots & \sigma^2 \end{bmatrix}$$

In this section we will allow violations of this assumption in the following heteroskedastic form.

Notation

The constant error variance assumption sometimes called homoskedasiticity states that

$$Var(\mathbf{u}) = E[\mathbf{u}\mathbf{u}'] = \begin{bmatrix} \sigma^2 & 0 & 0 & \dots & 0 \\ 0 & \sigma^2 & 0 & \dots & 0 \\ & & & \vdots & \\ 0 & 0 & 0 & \dots & \sigma^2 \end{bmatrix}$$

In this section we will allow violations of this assumption in the following heteroskedastic form.

$$Var(\mathbf{u}) = E[\mathbf{u}\mathbf{u}'] = \begin{bmatrix} \sigma_1^2 & 0 & 0 & \dots & 0 \\ 0 & \sigma_2^2 & 0 & \dots & 0 \\ & & & \vdots & \\ 0 & 0 & 0 & \dots & \sigma_n^2 \end{bmatrix}$$

Consequences of non-constant error variance

- The $\hat{\sigma}^2$ will not be unbiased for σ^2.

Consequences of non-constant error variance

- The $\hat{\sigma}^2$ will not be unbiased for σ^2.
- For "α" level tests, probability of Type I error will not be α.

Consequences of non-constant error variance

- The $\hat{\sigma}^2$ will not be unbiased for σ^2.
- For "α" level tests, probability of Type I error will not be α.
- "$1 - \alpha$" confidence intervals will not have $1 - \alpha$ coverage probability.

Consequences of non-constant error variance

- The $\hat{\sigma}^2$ will not be unbiased for σ^2.
- For "α" level tests, probability of Type I error will not be α.
- "$1 - \alpha$" confidence intervals will not have $1 - \alpha$ coverage probability.
- The LS estimator is no longer BLUE.

However,

Consequences of non-constant error variance

- The $\hat{\sigma}^2$ will not be unbiased for σ^2.
- For "α" level tests, probability of Type I error will not be α.
- "$1 - \alpha$" confidence intervals will not have $1 - \alpha$ coverage probability.
- The LS estimator is no longer BLUE.

However,

- The degree of the problem depends on the amount of heteroskedasticity.

Consequences of non-constant error variance

- The $\hat{\sigma}^2$ will not be unbiased for σ^2.
- For "α" level tests, probability of Type I error will not be α.
- "$1-\alpha$" confidence intervals will not have $1-\alpha$ coverage probability.
- The LS estimator is no longer BLUE.

However,

- The degree of the problem depends on the amount of heteroskedasticity.
- $\hat{\boldsymbol{\beta}}$ is still unbiased for $\boldsymbol{\beta}$

Heteroskedasticity Consistent Estimator

Suppose

Heteroskedasticity Consistent Estimator

Suppose

$$V[\mathbf{u}] = \mathbf{\Sigma} = \begin{bmatrix} \sigma_1^2 & 0 & 0 & \dots & 0 \\ 0 & \sigma_2^2 & 0 & \dots & 0 \\ & & & \vdots & \\ 0 & 0 & 0 & \dots & \sigma_n^2 \end{bmatrix}$$

Heteroskedasticity Consistent Estimator

Suppose

$$V[\mathbf{u}] = \boldsymbol{\Sigma} = \begin{bmatrix} \sigma_1^2 & 0 & 0 & \dots & 0 \\ 0 & \sigma_2^2 & 0 & \dots & 0 \\ & & & \vdots & \\ 0 & 0 & 0 & \dots & \sigma_n^2 \end{bmatrix}$$

then $Var(\hat{\beta}) = (\mathbf{X}'\mathbf{X})^{-1}\,\mathbf{X}'\Sigma\mathbf{X}\,(\mathbf{X}'\mathbf{X})^{-1}$ and Huber (and then White) showed that

Heteroskedasticity Consistent Estimator

Suppose

$$V[\mathbf{u}] = \Sigma = \begin{bmatrix} \sigma_1^2 & 0 & 0 & \dots & 0 \\ 0 & \sigma_2^2 & 0 & \dots & 0 \\ & & & \vdots & \\ 0 & 0 & 0 & \dots & \sigma_n^2 \end{bmatrix}$$

then $Var(\hat{\beta}) = (\mathbf{X}'\mathbf{X})^{-1}\,\mathbf{X}'\Sigma\mathbf{X}\,(\mathbf{X}'\mathbf{X})^{-1}$ and Huber (and then White) showed that

$$(\mathbf{X}'\mathbf{X})^{-1}\,\mathbf{X}' \begin{bmatrix} \hat{\mathbf{u}}_1^2 & 0 & 0 & \dots & 0 \\ 0 & \hat{\mathbf{u}}_2^2 & 0 & \dots & 0 \\ & & & \vdots & \\ 0 & 0 & 0 & \dots & \hat{\mathbf{u}}_n^2 \end{bmatrix} \mathbf{X}\,(\mathbf{X}'\mathbf{X})^{-1}$$

Heteroskedasticity Consistent Estimator

Suppose

$$V[\mathbf{u}] = \boldsymbol{\Sigma} = \begin{bmatrix} \sigma_1^2 & 0 & 0 & \dots & 0 \\ 0 & \sigma_2^2 & 0 & \dots & 0 \\ & & & \vdots & \\ 0 & 0 & 0 & \dots & \sigma_n^2 \end{bmatrix}$$

then $Var(\hat{\beta}) = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\boldsymbol{\Sigma}\mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}$ and Huber (and then White) showed that

$$(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}' \begin{bmatrix} \hat{\mathbf{u}}_1^2 & 0 & 0 & \dots & 0 \\ 0 & \hat{\mathbf{u}}_2^2 & 0 & \dots & 0 \\ & & & \vdots & \\ 0 & 0 & 0 & \dots & \hat{\mathbf{u}}_n^2 \end{bmatrix} \mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}$$

is a consistent estimator of $V[\hat{\beta}]$.

Things to note about this approach

1. Requires larger sample size

Things to note about this approach

1. Requires larger sample size
   - large enough for each estimate (e.g., large enough in both treatment and baseline groups or large enough in both runoff and non-runoff groups)

Things to note about this approach

1. Requires larger sample size
   - large enough for each estimate (e.g., large enough in both treatment and baseline groups or large enough in both runoff and non-runoff groups)
   - large enough for consistent estimates (e.g., need $n \geq 250$ for Stata default when highly heteroskedastic (Long and Ervin 2000)).

Things to note about this approach

1. Requires larger sample size
   - large enough for each estimate (e.g., large enough in both treatment and baseline groups or large enough in both runoff and non-runoff groups)
   - large enough for consistent estimates (e.g., need $n \geq 250$ for Stata default when highly heteroskedastic (Long and Ervin 2000)).
2. Doesn't make $\hat{\beta}$ BLUE
3. What are you going to do with predicted probabilities?

Outline

1. An Introduction to Robust and Clustered Standard Errors
   - Linear Regression with Non-constant Variance
   - GLM's and Non-constant Variance
   - Cluster-Robust Standard Errors
2. Replicating in R

What happens when the model is not linear?

Huber (1967) developed a general way to find the standard errors for models *that are specified in the wrong way.*

What happens when the model is not linear?

Huber (1967) developed a general way to find the standard errors for models *that are specified in the wrong way.*

Under certain conditions, you can get the standard errors, even if your model is misspecified.

What happens when the model is not linear?

Huber (1967) developed a general way to find the standard errors for models *that are specified in the wrong way.*

Under certain conditions, you can get the standard errors, even if your model is misspecified.

These are the robust standard errors that scholars now use for other glm's, and that happen to coincide with the linear case.

What does it mean for a non-linear model to have heteroskedasticity?

- Think about the probit model in the latent variable formulation.

What does it mean for a non-linear model to have heteroskedasticity?

- Think about the probit model in the latent variable formulation.
- Pretend that there is heteroskedasticity on the linear model for y^*.

What does it mean for a non-linear model to have heteroskedasticity?

- Think about the probit model in the latent variable formulation.
- Pretend that there is heteroskedasticity on the linear model for y^*.
- Heteroskedasticity in the latent variable formulation will completely change the functional form of $P(y = 1|x)$.

What does it mean for a non-linear model to have heteroskedasticity?

- Think about the probit model in the latent variable formulation.
- Pretend that there is heteroskedasticity on the linear model for y^*.
- Heteroskedasticity in the latent variable formulation will completely change the functional form of $P(y = 1|x)$.
- What does this mean? The $P(y = 1|x) \neq \Phi(\mathbf{x}\beta)$. Your model is wrong.

RSEs for GLMs

To derive robust standard errors in the general case, we assume that

RSEs for GLMs

To derive robust standard errors in the general case, we assume that

$$y \sim f_i(y|\theta)$$

RSEs for GLMs

To derive robust standard errors in the general case, we assume that

$$y \sim f_i(y|\theta)$$

Then our likelihood function is given by

RSEs for GLMs

To derive robust standard errors in the general case, we assume that

$$y \sim f_i(y|\theta)$$

Then our likelihood function is given by

$$\prod_{i=1}^{n} f_i(Y_i|\theta)$$

RSEs for GLMs

To derive robust standard errors in the general case, we assume that

$$y \sim f_i(y|\theta)$$

Then our likelihood function is given by

$$\prod_{i=1}^{n} f_i(Y_i|\theta)$$

and thus the log-likelihood is

RSEs for GLMs

To derive robust standard errors in the general case, we assume that

$$y \sim f_i(y|\theta)$$

Then our likelihood function is given by

$$\prod_{i=1}^{n} f_i(Y_i|\theta)$$

and thus the log-likelihood is

$$L(\theta) = \sum_{i=1}^{n} \log f_i(Y_i|\theta)$$

RSEs for GLMs

We will denote the first and second partial derivatives of L to be:

RSEs for GLMs

We will denote the first and second partial derivatives of L to be:

$$L'(\theta) = \sum_{i=1}^{n} g_i(Y_i|\theta), L''(\theta) = \sum_{i=1}^{n} h_i(Y_i|\theta)$$

RSEs for GLMs

We will denote the first and second partial derivatives of L to be:

$$L'(\theta) = \sum_{i=1}^{n} g_i(Y_i|\theta), L''(\theta) = \sum_{i=1}^{n} h_i(Y_i|\theta)$$

Where

$$g_i(Y_i|\theta) = [logf_i(y|\theta)]' = \frac{\delta}{\delta\theta} \log f_i(y|\theta)$$

RSEs for GLMs

We will denote the first and second partial derivatives of L to be:

$$L'(\theta) = \sum_{i=1}^{n} g_i(Y_i|\theta), L''(\theta) = \sum_{i=1}^{n} h_i(Y_i|\theta)$$

Where

$$g_i(Y_i|\theta) = [log f_i(y|\theta)]' = \frac{\delta}{\delta\theta} \log f_i(y|\theta)$$

RSEs for GLMs

We will denote the first and second partial derivatives of L to be:

$$L'(\theta) = \sum_{i=1}^{n} g_i(Y_i|\theta), L''(\theta) = \sum_{i=1}^{n} h_i(Y_i|\theta)$$

Where

$$g_i(Y_i|\theta) = [logf_i(y|\theta)]' = \frac{\delta}{\delta\theta} \log f_i(y|\theta)$$

and

RSEs for GLMs

We will denote the first and second partial derivatives of L to be:

$$L'(\theta) = \sum_{i=1}^{n} g_i(Y_i|\theta), L''(\theta) = \sum_{i=1}^{n} h_i(Y_i|\theta)$$

Where

$$g_i(Y_i|\theta) = [logf_i(y|\theta)]' = \frac{\delta}{\delta\theta} \log f_i(y|\theta)$$

and

$$h_i(Y_i|\theta) = [logf_i(y|\theta)]'' = \frac{\delta^2}{\delta\theta^2} \log f_i(y|\theta)$$

RSEs for GLMs

This shouldn't be too unfamiliar.

RSEs for GLMs

This shouldn't be too unfamiliar.

Remember, the Fisher information matrix is $-E_{\theta}[h_i(Y_i|\theta)]$.

RSEs for GLMs

- Let's assume the model is correct – there is a true value θ_0 for θ.

RSEs for GLMs

- Let's assume the model is correct – there is a true value θ_0 for θ.
- Then we can use the Taylor approximation for the log-likelihood function to estimate what the likelihood function looks like around θ_0:

RSEs for GLMs

- Let's assume the model is correct – there is a true value θ_0 for θ.
- Then we can use the Taylor approximation for the log-likelihood function to estimate what the likelihood function looks like around θ_0:

$$L(\theta) = L(\theta_0) + L'(\theta_0)(\theta - \theta_0) + \frac{1}{2}(\theta - \theta_0)^T L''(\theta_0)(\theta - \theta_0)$$

RSEs for GLMs

- We can use the Taylor approximation to approximate $\hat{\theta} - \theta_0$ and therefore the variance covariance matrix.

RSEs for GLMs

- We can use the Taylor approximation to approximate $\hat{\theta} - \theta_0$ and therefore the variance covariance matrix.
- We want to find the maximum of the log-likelihood function, so we set $L'(\theta) = 0$:

RSEs for GLMs

- We can use the Taylor approximation to approximate $\hat{\theta} - \theta_0$ and therefore the variance covariance matrix.
- We want to find the maximum of the log-likelihood function, so we set $L'(\theta) = 0$:

$$L'(\theta_0) + (\theta - \theta_0)^T L''(\theta_0) = 0$$

RSEs for GLMs

- We can use the Taylor approximation to approximate $\hat{\theta} - \theta_0$ and therefore the variance covariance matrix.
- We want to find the maximum of the log-likelihood function, so we set $L'(\theta) = 0$:

$$L'(\theta_0) + (\theta - \theta_0)^T L''(\theta_0) = 0$$

$$\hat{\theta} - \theta_0 = [-L''(\theta_0)]^{-1} L'(\theta_0)^T$$

RSEs for GLMs

- We can use the Taylor approximation to approximate $\hat{\theta} - \theta_0$ and therefore the variance covariance matrix.
- We want to find the maximum of the log-likelihood function, so we set $L'(\theta) = 0$:

$$L'(\theta_0) + (\theta - \theta_0)^T L''(\theta_0) = 0$$

$$\hat{\theta} - \theta_0 = [-L''(\theta_0)]^{-1} L'(\theta_0)^T$$

$$Avar(\hat{\theta}) = [-L''(\theta_0)]^{-1} [Cov(L'(\theta_0))] [-L''(\theta_0)]^{-1}$$

RSEs for GLMs

It's the sandwich estimator.

$$Avar(\hat{\theta}) = [-L''(\theta_0)]^{-1}[Cov(L'(\theta_0))][-L''(\theta_0)]^{-1}$$

$$= \left[-\sum_{i=1}^{n} h_i(Y_i|\hat{\theta})\right]^{-1} \left[\sum_{i=1}^{n} g_i(Y_i|\hat{\theta})^T g_i(Y_i|\hat{\theta})\right] \left[-\sum_{i=1}^{n} h_i(Y_i|\hat{\theta})\right]^{-1}$$

RSEs for GLMs

It's the sandwich estimator.

$$= \left[-\sum_{i=1}^{n} h_i(Y_i|\hat{\theta}) \right]^{-1} \left[\sum_{i=1}^{n} g_i(Y_i|\hat{\theta})^T g_i(Y_i|\hat{\theta}) \right] \left[-\sum_{i=1}^{n} h_i(Y_i|\hat{\theta}) \right]^{-1}$$

Bread: $\left[-\sum_{i=1}^{n} h_i(Y_i|\hat{\theta}) \right]^{-1}$ Meat: $\left[\sum_{i=1}^{n} g_i(Y_i|\hat{\theta})^T g_i(Y_i|\hat{\theta}) \right]$

Cluster-Robust Standard Errors

Using clustered-robust standard errors, the meat changes.

Cluster-Robust Standard Errors

Using clustered-robust standard errors, the meat changes.

Instead of summing over each individual, we first sum over the groups.

Cluster-Robust Standard Errors

Using clustered-robust standard errors, the meat changes.

Instead of summing over each individual, we first sum over the groups.

$$\left[\sum_{j=1}^{n}\sum_{i\in c_j} g_i(Y_i|\hat{\theta})^T g_i(Y_i|\hat{\theta})\right]$$

Cluster-Robust Standard Errors

Using clustered-robust standard errors, the meat changes.

Instead of summing over each individual, we first sum over the groups.

$$\left[\sum_{j=1}^{n}\sum_{i\in c_j} g_i(Y_i|\hat{\theta})^T g_i(Y_i|\hat{\theta})\right]$$

Cluster-Robust Standard Errors

Using clustered-robust standard errors, the meat changes.

Instead of summing over each individual, we first sum over the groups.

$$\left[\sum_j \sum_{i \in c_j} g_i(Y_i|\hat{\theta})^T g_i(Y_i|\hat{\theta})\right]$$

Outline

1 An Introduction to Robust and Clustered Standard Errors
- Linear Regression with Non-constant Variance
- GLM's and Non-constant Variance
- Cluster-Robust Standard Errors

2 Replicating in R

Replicating in R

- There are lots of different ways to replicate these standard errors in R.

Replicating in R

- There are lots of different ways to replicate these standard errors in R.
- Sometimes it's difficult to figure out what is going on in Stata.

Replicating in R

- There are lots of different ways to replicate these standard errors in R.
- Sometimes it's difficult to figure out what is going on in Stata.
- But by really understanding what is going on in R, you will be able to replicate once you know the equation for Stata.

Some Data

I'm going to use data from the Gov 2001 Code library.

Some Data

I'm going to use data from the Gov 2001 Code library.

```
load("Gov2001CodeLibrary.RData")
```

Some Data

I'm going to use data from the Gov 2001 Code library.

```
load("Gov2001CodeLibrary.RData")
```

This particular dataset is from the paper "When preferences and commitments collide: The effect of relative partisan shifts on International treaty compliance." in *International Organization* by Joseph Grieco, Christopher Gelpi, and Camber Warren.

Some Data

I'm going to use data from the Gov 2001 Code library.

```
load("Gov2001CodeLibrary.RData")
```

This particular dataset is from the paper "When preferences and commitments collide: The effect of relative partisan shifts on International treaty compliance." in *International Organization* by Joseph Grieco, Christopher Gelpi, and Camber Warren.

Thanks to Michele Margolis and Dan Altman for their contributions to the library!

The Model

First, let's run their model:

The Model

First, let's run their model:

```
fmla <- as.formula(restrict ~ art8 + shift_left + flexible
       regnorm + gdpgrow + resgdp + bopgdp + useimfcr +
       surveil + univers + resvol + totvol + tradedep + mil
       termlim + parli + lastrest + lastrest2 + lastrest3)
```

The Model

First, let's run their model:

```
fmla <- as.formula(restrict ~ art8 + shift_left + flexible
       regnorm + gdpgrow + resgdp + bopgdp + useimfcr +
       surveil + univers + resvol + totvol + tradedep + mil
       termlim + parli + lastrest + lastrest2 + lastrest3)
```

```
  fit <-glm(fmla, data=treaty1,
      family=binomial(link="logit"))
```

The Meat and Bread

First recognize that the bread of the sandwich estimator is just the variance covariance matrix.

The Meat and Bread

First recognize that the bread of the sandwich estimator is just the variance covariance matrix.

```
library(sandwich)
bread <-vcov(fit)
```

The Meat and Bread

First recognize that the bread of the sandwich estimator is just the variance covariance matrix.

```
library(sandwich)
bread <-vcov(fit)
```

For the meat, we are going to use the estimating function to create the matrices first derivative:

```
est.fun <- estfun(fit)
```

Note: if estfun doesn't work for your glm, there is a way to do it using numericGradient().

The Sandwich

So we can create the sandwich

The Sandwich

So we can create the sandwich

```
meat <- t(est.fun)%*%est.fun
sandwich <- bread%*%meat%*%bread
```

The Sandwich

So we can create the sandwich

```
meat <- t(est.fun)%*%est.fun
sandwich <- bread%*%meat%*%bread
```

And put them back in our table

The Sandwich

So we can create the sandwich

```
meat <- t(est.fun)%*%est.fun
sandwich <- bread%*%meat%*%bread
```

And put them back in our table

```
library(lm.test)
coeftest(fit, sandwich)
```

Note: For the linear case, estfun() is doing something a bit different than in the logit, so use:

```
robust <- sandwich(lm.1, meat=crossprod(est.fun)/N)
```

Clustered Standard Errors

Clustered Standard Errors

First, we have to identify our clusters:

Clustered Standard Errors

First, we have to identify our clusters:

```
fc <- treaty1$imf_ccode
m <- length(unique(fc))
k <- length(coef(fit))
```

Clustered Standard Errors

First, we have to identify our clusters:

```
fc <- treaty1$imf_ccode
m <- length(unique(fc))
k <- length(coef(fit))
```

Then, we sum the u's by cluster

Clustered Standard Errors

First, we have to identify our clusters:

```
fc <- treaty1$imf_ccode
m <- length(unique(fc))
k <- length(coef(fit))
```

Then, we sum the u's by cluster

```
u <- estfun(fit)
u.clust <- matrix(NA, nrow=m, ncol=k)
for(j in 1:k){
u.clust[,j] <- tapply(u[,j], fc, sum)
}
```

Last, we can make our cluster robust matrix:

Last, we can make our cluster robust matrix:

```
cl.vcov <- vcov %*% ((m / (m-1)) * t(u.clust)
    %*% (u.clust)) %*% + vcov
```

Last, we can make our cluster robust matrix:

```
cl.vcov <- vcov %*% ((m / (m-1)) * t(u.clust)
    %*% (u.clust)) %*% + vcov
```

And test our coefficients

Last, we can make our cluster robust matrix:

```
cl.vcov <- vcov %*% ((m / (m-1)) * t(u.clust)
    %*% (u.clust)) %*% + vcov
```

And test our coefficients

```
coeftest(fit, cl.vcov)
```

A couple notes

- There are easier ways to do this in R (see for example hccm).

A couple notes

- There are easier ways to do this in R (see for example hccm).
- But it's good to know what is going on, especially when you are replicating.

A couple notes

- There are easier ways to do this in R (see for example hccm).
- But it's good to know what is going on, especially when you are replicating.
- Beware: degrees of freedom corrections.

A couple notes

- There are easier ways to do this in R (see for example hccm).
- But it's good to know what is going on, especially when you are replicating.
- Beware: degrees of freedom corrections.