# ECON 557 – Advanced Data Analysis

Michael T. Sandfort

Department of Economics
Masters in Applied Economics Program
Georgetown University

February 17, 2023

# Review: Maximum Likelihood Estimates of a True Model Are CAN

- Recall a key result from our last class:
  - If $f(\mathbf{y}_i|\mathbf{x}_i, \boldsymbol{\theta})$ is the true density of $\mathbf{y}_i$ given $\mathbf{x}_i$ and $\boldsymbol{\theta}$ (specification), $\boldsymbol{\theta}_o$ uniquely maximizes the expected likelihood of $\mathbf{y}_i$ given $\mathbf{x}_i$ and $\boldsymbol{\theta}$ (identification), $\boldsymbol{\Theta} \subset \mathbb{R}^n$ is closed and bounded, and $f$ and the log-likelihood are differentiable, then $\hat{\boldsymbol{\theta}}_{\text{ML}}$ is a consistent estimator of $\boldsymbol{\theta}_o$.
  - The ML estimate is asymptotically normal, with

$$\sqrt{n}(\hat{\boldsymbol{\theta}}_{\text{ML}} - \boldsymbol{\theta}_o) \xrightarrow{d} \mathcal{N}\left(\mathbf{0}, (-E[\mathbf{H}(\boldsymbol{\theta}_o)])^{-1}\right)$$

    where $\mathbf{H}$ is the Hessian of the log-likelihood.

# Review: Model Misspecification and the "Sandwich" Covariance Estimator

- With a correct conditional mean function and an EDF model, we can still get an analogue to the zero expected score condition like

$$\mathbb{E}_g \left[ \left. \frac{\partial \mathcal{L}}{\partial \boldsymbol{\theta}} \right|_{\boldsymbol{\theta}=\boldsymbol{\theta}^*} \right] = \mathbf{0}$$

- Unfortunately, if we don't use the correct model, the information equality no longer holds, so we need to separately define the observed information

$$\mathbf{A}(\boldsymbol{\theta}) = E_g \left[ \frac{\partial^2 \mathcal{L}(y, \boldsymbol{\theta})}{\partial \boldsymbol{\theta} \partial \boldsymbol{\theta}^\top} \right]$$

  and expected information

$$\mathbf{B}(\boldsymbol{\theta}) = E_g \left[ \frac{\partial \mathcal{L}(y, \boldsymbol{\theta})}{\partial \boldsymbol{\theta}^\top} \frac{\partial \mathcal{L}(y, \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \right]$$

- Even so, our QML estimator is consistent for $\boldsymbol{\theta}^*$ ($\hat{\boldsymbol{\theta}}_{\text{QML}} \to \boldsymbol{\theta}^*$) and asymptotically normal, with

$$\sqrt{N}(\hat{\boldsymbol{\theta}}_{\text{QML}} - \boldsymbol{\theta}^*) \overset{A}{\sim} \mathcal{N}(0, \mathbf{A}^{-1}\mathbf{B}\mathbf{A}^{-1})$$

# Standard Errors for Parameters

- Standard errors for the parameters $\widehat{\boldsymbol{\theta}}$ can be taken from the diagonal of the asymptotic covariance matrix, just as you would in OLS.
- Remember that these covariance matrices are for the estimator, not for the outcome or covariates.
- How close the true, finite-sample distribution of the estimator (sampling distribution) is to the assumed "infinite sample" asymptotic distribution of the estimator depends on, among other things, the size of the sample.
- Using the asymptotic distribution from the first page is more "model trusting" – like relying on homoskedasticity in a linear regression model as we discussed last week.
- Using the asymptotic distribution from the second page is less "model trusting." It allows for the possibility that the distribution family or variance function might be incorrect.

# Standard Errors for Marginal Effects

- Often, though, the true parameter is of less immediate interest than is the marginal effect of a covariate.
- Thinking about the conditional mean function, the marginal effect of $x_j$ flows up through the index $\eta = \mathbf{x}\boldsymbol{\beta}$ and then through the inverse link function $b'^{-1}(\eta)$ (assuming a canonical link, for simplicity), following the chain rule.
- Even if $x_j$ enters alone (no $x_j^2$ or other nonlinear functions are present), the marginal effect will typically be both nonlinear and implicate more parameters than just $\beta_j$.
- Analytical calculation of standard errors for marginal effects resolves this problem by taking a local linear approximation to the marginal effect and then using the standard formula for the variance of a linear function.
- This approach is known as the delta method.

# The Delta Method

- The delta method is quite general – it can be used to calculate standard errors or confidence intervals for any smooth function $h(\cdot)$ of a set of coefficient estimates where an estimate of the covariance matrix for $\widehat{\boldsymbol{\beta}}$ is available.
- To calculate confidence intervals for the marginal effects, we'll take $h()$ to be the marginal effect of $x_j$:

$$h(\boldsymbol{\beta}) = \frac{\partial \mu(\mathbf{x}\boldsymbol{\beta})}{\partial x_j}$$

- Remember that we already have an estimate, $\widehat{\mathbb{V}[\widehat{\boldsymbol{\beta}}]}$, of the sampling variance of $\widehat{\boldsymbol{\beta}}$.
- We're trying to estimate the sampling variance of $h(\widehat{\boldsymbol{\beta}})$.

# The Delta Method

- If $h()$ were linear rather than nonlinear (say $\mathbf{c}^\top \boldsymbol{\beta}$), then our job would be easy.
- Suppose for the moment that we have the correct model.
- We saw before that the sampling variance of the ML estimator is the inverse of the negative expected Hessian

$$\widehat{\mathbb{V}(\widehat{\boldsymbol{\beta}})} \equiv \mathbf{A}^{-1} = \mathbb{E}[-\mathbf{H}(\boldsymbol{\beta})]^{-1}$$

  that is, from

$$\sqrt{n}(\widehat{\boldsymbol{\beta}} - \boldsymbol{\beta}) \xrightarrow{d} \mathcal{N}(\mathbf{0}, \mathbf{A}^{-1}).$$

- We could therefore rely on our rules for variances to say that for some vector $\mathbf{c}$, $\widehat{\mathbb{V}(\mathbf{c}^\top \widehat{\boldsymbol{\beta}})} = \mathbf{c}\mathbf{A}^{-1}\mathbf{c}^\top$ is a good estimator of this linear combination of parameters.
- If this formula is puzzling to you, remember that for scalar random variables $\mathbb{V}[aX] = a^2\mathbb{V}[X]$.

# The Delta Method

- But if $h()$ is differentiable, it's almost (locally) linear in the sense that we can take a Taylor series approximation

$$h(\widehat{\boldsymbol{\beta}}) \approx h(\boldsymbol{\beta}) + (\widehat{\boldsymbol{\beta}} - \boldsymbol{\beta})^\top \frac{\partial h(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}}$$

- Rearranging terms in the above expression gives

$$\sqrt{n}[h(\widehat{\boldsymbol{\beta}}) - h(\boldsymbol{\beta})] \approx \sqrt{n}(\widehat{\boldsymbol{\beta}} - \boldsymbol{\beta})^\top \frac{\partial h(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}}$$

  so

$$h(\widehat{\boldsymbol{\beta}}) \overset{a}{\sim} \mathcal{N}\left(h(\boldsymbol{\beta}), \frac{\partial h(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}^\top} \mathbb{V}(\boldsymbol{\beta}) \frac{\partial h(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}}\right)$$

- We would then propose to estimate the asymptotic variance of $h(\widehat{\boldsymbol{\beta}})$ with

$$\widehat{\mathbb{V}[h(\widehat{\boldsymbol{\beta}})]} = \frac{\partial h(\widehat{\boldsymbol{\beta}})}{\partial \boldsymbol{\beta}^\top} \widehat{\mathbb{V}(\widehat{\boldsymbol{\beta}})} \frac{\partial h(\widehat{\boldsymbol{\beta}})}{\partial \boldsymbol{\beta}}$$

  which operationalizes the estimator.

# ML Standard Errors for Marginal Effects

- Suppose $h()$ is the partial effect of $x_j$, entering only linearly into our probit model. Then it's easy to show that

$$\begin{aligned}\frac{\partial h(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}} &= \begin{bmatrix} \frac{\partial h}{\partial \beta_1} & \frac{\partial h}{\partial \beta_2} & \cdots & \frac{\partial h}{\partial \beta_k} \end{bmatrix} \\ &= \begin{bmatrix} \phi'(\mathbf{x}\boldsymbol{\beta})x_1\beta_j \\ \phi'(\mathbf{x}\boldsymbol{\beta})x_2\beta_j \\ \vdots \\ \phi'(\mathbf{x}\boldsymbol{\beta})x_j\beta_j + \phi(\mathbf{x}\boldsymbol{\beta}) \\ \vdots \\ \phi'(\mathbf{x}\boldsymbol{\beta})x_k\beta_j \end{bmatrix}'\end{aligned}$$

- The delta method is applicable any time you want to find standard errors for a function of your coefficients, i.e., you can use it with OLS, FGLS, GMM or ML.

# Example: ML SE's for Marginal Effects (APE or AME)

- These are the mean marginal effects across all observations in the data.
- They are "model trusting" in the sense that the covariance estimator assumes the information equality holds, which is only guaranteed when not only the mean response, but its variance and the modeling family are correctly specified.
- What is being reported is the effect of another year of education on the predictor for labor force participation.
- The effect is denominated in percentage points.

```
> library(margins)
> summary(margins(res.probit))
   factor     AME     SE       z      p   lower   upper
      age -0.0159 0.0024 -6.7392 0.0000 -0.0205 -0.0113
     educ  0.0394 0.0073  5.4186 0.0000  0.0251  0.0536
    exper  0.0256 0.0022 11.4506 0.0000  0.0212  0.0300
  kidsge6  0.0108 0.0132  0.8189 0.4129 -0.0151  0.0367
  kidslt6 -0.2612 0.0319 -8.1860 0.0000 -0.3237 -0.1986
 nwifeinc -0.0036 0.0015 -2.4604 0.0139 -0.0065 -0.0007
```

# Example: ML Standard Errors for Marginal Effects (PEA, by hand)

- We can also implement the calculation directly by hand.
- This is also a "model-trusting" estimate, but at covariate means.
- You can easily check that this is the same as what is returned by `margins` for the standard error on the education variable (next slide).

```
> X.avg
     iota nwifeinc     educ    exper  expersq      age   kidslt6  kidsge6
[1,]    1 20.12896 12.28685 10.63081 113.0141 42.53785 0.2377158 1.353254
> k <- length(X.avg); k
[1] 8
> beta.j <-  coef(res.probit)[3]
> idx <- (matrix(X.avg,1,k) %*% matrix(coef(res.probit),k,1))[1,1]
> grad.h <- matrix(-idx*dnorm(idx)*X.avg*beta.j,nrow=1,ncol=k)
> grad.h[1,3] <- grad.h[1,3]+dnorm(idx)
> vcov.marginal <- grad.h %*% vcov(res.probit) %*% t(grad.h)
> sqrt(vcov.marginal)
             [,1]
[1,] 0.009660841
```

# Example: ML Standard Errors for Marginal Effects (PEA, by "canned")

- Here is the same estimate using `margins`.
- Note that a year of education might make sense as a unit, but what to make of the value at which `kidslt6` is evaluated?

```
> options(width=150)
> summary(margins(res.probit,at=as.data.frame(X.avg)[,-1]))
   factor nwifeinc    educ   exper  expersq     age kidslt6 kidsge6     AME     SE       z      p   lower   upper
      age  20.1290 12.2869 10.6308 113.0141 42.5378  0.2377  1.3533 -0.0200 0.0032 -6.1635 0.0000 -0.0263 -0.0136
     educ  20.1290 12.2869 10.6308 113.0141 42.5378  0.2377  1.3533  0.0495 0.0097  5.1217 0.0000  0.0305  0.0684
    exper  20.1290 12.2869 10.6308 113.0141 42.5378  0.2377  1.3533  0.0315 0.0031 10.0721 0.0000  0.0253  0.0376
  kidsge6  20.1290 12.2869 10.6308 113.0141 42.5378  0.2377  1.3533  0.0136 0.0166  0.8177 0.4135 -0.0190  0.0462
  kidslt6  20.1290 12.2869 10.6308 113.0141 42.5378  0.2377  1.3533 -0.3282 0.0453 -7.2473 0.0000 -0.4170 -0.2394
 nwifeinc  20.1290 12.2869 10.6308 113.0141 42.5378  0.2377  1.3533 -0.0045 0.0019 -2.4348 0.0149 -0.0082 -0.0009
```

# Review: Estimating the Asymptotic Variance of $\widehat{\boldsymbol{\beta}}_{OLS}$

As we saw last class:

- Under homoskedasticity, $\widehat{\boldsymbol{\beta}}_{OLS}$ is approximately normal with mean $\boldsymbol{\beta}$ and variance
$$\frac{1}{N}\sigma^2 \left(\mathbb{E}[\mathbf{x}'\mathbf{x}]\right)^{-1}.$$
- Without the homoskedasticity, $\widehat{\boldsymbol{\beta}}_{OLS}$ is again approximately normal with mean $\boldsymbol{\beta}$, but with variance
$$\frac{1}{N}\left(\mathbb{E}[\mathbf{x}'\mathbf{x}]\right)^{-1}\mathbb{E}[u^2\mathbf{x}'\mathbf{x}]\left(\mathbb{E}[\mathbf{x}'\mathbf{x}]\right)^{-1}$$

Because both of these expressions are characterized by expectations, an analogy estimator is immediately available.

# Review: Estimating the Asymptotic Variance of $\widehat{\boldsymbol{\beta}}_{OLS}$

Because $\mathbb{V}[u] = \mathbb{E}[u^2] = \sigma^2$ under homoskedasticity, an analogy estimator of $\sigma^2$ is available

$$\widehat{\sigma}^2 = \frac{1}{N}\sum_{i=1}^{N}\widehat{u}_i^2$$

where $\widehat{u}_i$ are the OLS residuals $y_i - \mathbf{x}_i\widehat{\boldsymbol{\beta}}_{OLS}$.

But this ignores the small amount of built-in bias which comes from using $\widehat{\boldsymbol{\beta}}_{OLS}$ rather than $\boldsymbol{\beta}$ to compute the $\widehat{u}_i$. Consequently, it is more common to use

$$s^2 = \frac{1}{N-K}\sum_{i=1}^{N}\widehat{u}_i^2 = \frac{N}{N-K}\widehat{\sigma}^2$$

to estimate $\sigma^2$. This is also known as the degrees-of-freedom (d.o.f.) correction or Bessel's correction.

# Review:Estimating the Asymptotic Variance of $\widehat{\boldsymbol{\beta}}_{OLS}$

For the term $\left(\mathbb{E}[\mathbf{x}'\mathbf{x}]\right)^{-1}$, we choose by analogy

$$\left[\frac{1}{N}\sum_{i=1}^{N}\mathbf{x}_i'\mathbf{x}_i\right]^{-1} = N\left(\mathbf{X}'\mathbf{X}\right)^{-1}$$

which is a sum of $K \times K$ matrices (recall that $\mathbf{x}_i$ is a row vector).

Thus, under homoskedasticity, our estimator for the asymptotic variance of $\widehat{\boldsymbol{\beta}}_{OLS}$ is

$$\widehat{V_{\widehat{\boldsymbol{\beta}}_{OLS}}} = \frac{1}{N}\left[s^2 \cdot N\left(\mathbf{X}'\mathbf{X}\right)^{-1}\right] = s^2(\mathbf{X}'\mathbf{X})^{-1}$$

# Estimating the Asymptotic Variance of $\widehat{\boldsymbol{\beta}}_{OLS}$

Under heteroskedasticity we need an estimator for $\left(\mathbb{E}[u^2\mathbf{x}'\mathbf{x}]\right)^{-1}$. Again, an analogy estimator is readily available:

$$\frac{1}{N}\sum_{i=1}^{N}\widehat{u}_i^2\mathbf{x}_i'\mathbf{x}_i$$

Plugging this in gives a heteroskedasticity-consistent (HC) estimator of the asymptotic variance of $\widehat{\boldsymbol{\beta}}_{OLS}$, known as "HC0":

$$\begin{aligned}\widehat{V^{\text{HC0}}_{\widehat{\boldsymbol{\beta}}_{OLS}}} &= \frac{1}{N}\left[N\left(\mathbf{X}'\mathbf{X}\right)^{-1}\left(\frac{1}{N}\sum_{i=1}^{N}\widehat{u}_i^2\mathbf{x}_i'\mathbf{x}_i\right)N\left(\mathbf{X}'\mathbf{X}\right)^{-1}\right] \\ &= \left(\mathbf{X}'\mathbf{X}\right)^{-1}\left[\sum_{i=1}^{N}\widehat{u}_i^2\mathbf{x}_i'\mathbf{x}_i\right]\left(\mathbf{X}'\mathbf{X}\right)^{-1}\end{aligned}$$

# Estimating the Asymptotic Variance of $\widehat{\boldsymbol{\beta}}_{OLS}$

Applying the degrees-of-freedom correction to "HC0" gives "HC1":

$$\widehat{V^{\text{HC1}}_{\widehat{\boldsymbol{\beta}}_{OLS}}} = \frac{N}{N-K}(\mathbf{X}'\mathbf{X})^{-1}\left[\sum_{i=1}^{N}\widehat{u}_i^2\mathbf{x}_i'\mathbf{x}_i\right](\mathbf{X}'\mathbf{X})^{-1}$$

# Estimating the Asymptotic Variance of $\widehat{\boldsymbol{\beta}}_{OLS}$

Similarly, the HC estimators "HC2" and "HC3" are given by

$$\widehat{V^{\text{HC2}}_{\widehat{\boldsymbol{\beta}}_{OLS}}} = (\mathbf{X}'\mathbf{X})^{-1}\left[\sum_{i=1}^{N}\frac{\widehat{u}_i^2}{1-h_i}\mathbf{x}_i'\mathbf{x}_i\right](\mathbf{X}'\mathbf{X})^{-1}$$

and

$$\widehat{V^{\text{HC3}}_{\widehat{\boldsymbol{\beta}}_{OLS}}} = (\mathbf{X}'\mathbf{X})^{-1}\left[\sum_{i=1}^{N}\frac{\widehat{u}_i^2}{(1-h_i)^2}\mathbf{x}_i'\mathbf{x}_i\right](\mathbf{X}'\mathbf{X})^{-1}.$$

where $h_i$ is the $i$th diagonal element of $P_{\mathbf{X}}$ (the "hat" matrix). These were suggested by MacKinnon and White (1985) to improve performance in small samples.

# Estimating the Asymptotic Variance of $\widehat{\beta}_{OLS}$

In R, the above estimators can all be used through the `lmtest` and `sandwich` packages:

```
> suppressMessages(library(tidyverse))
> suppressMessages(library(lmtest))
> library(sandwich)
> library(haven)
>
> cps <- read_dta(paste0(myDataPath,"cps09mar.dta"))
> cps$wage <- cps$earnings/(cps$hours*cps$week)
> cps$lwage <- log(cps$wage)
> cps$sex <- factor(cps$female,labels=c("Male","Female"))
> cps$exper <- cps$age-cps$education-6
> cps$expersq <- (cps$exper)^2
>
> md <- cps |> filter(female==1,race==2,marital==1,exper==12)
>
> res.lm <- lm(lwage~education,data=md)
```

# Estimating the Asymptotic Variance of $\widehat{\beta}_{OLS}$

While we could use `summary` to view the results with homoskedasticity assumed, we'll use `coeftest` from the `lmtest` library because of its flexibility:

```
> coeftest(res.lm)

t test of coefficients:

            Estimate Std. Error t value Pr(>|t|)
(Intercept) 0.697815   0.706653  0.9875 0.336490
education   0.155039   0.044648  3.4725 0.002718 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

# Estimating the Asymptotic Variance of $\widehat{\beta}_{OLS}$

Suppose I want to compare to Stata output, which uses the "HC1" estimator by default when "robust" errors are specified:

```
> coeftest(res.lm,vcov=vcovHC,type="HC1")

t test of coefficients:

            Estimate Std. Error t value  Pr(>|t|)
(Intercept) 0.697815   0.486106  1.4355    0.1683
education   0.155039   0.030129  5.1458 6.779e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

# Estimating the Asymptotic Variance of $\widehat{\beta}_{OLS}$

If I want to compare a lot of different specifications, using one of the `Xapply` functions can be very helpful:

```
> # How to use lapply to programmatically explore models
> vcov_names <- c("const","HC0","HC1","HC2","HC3")
> vcov_ests <- lapply(vcov_names,
+         function(vname) { vcovHC(res.lm, type=vname) }
+ )
```

The above stores the estimated asymptotic variance-covariance matrix of the $\widehat{\beta}$ in a list, where each list entry corresponds to a different vcov estimator (homoskedastic, "HC0", "HC1", etc.).

# Estimating the Asymptotic Variance of $\widehat{\beta}_{OLS}$

- The code below reproduces the table in BHE, p. 117.
- It's also a nice demonstration of how to use an anonymous (a.k.a. "lambda") function with the `Xapply` family of tools.

```
> se_ests <- sapply(vcov_names, function(vname) { sqrt(diag(vcovHC(res.lm, type=vname))) })
> se_ests <- t(se_ests)
> round(se_ests,3)
      (Intercept) education
const       0.707     0.045
HC0         0.461     0.029
HC1         0.486     0.030
HC2         0.493     0.031
HC3         0.527     0.033
```

# Another Extended Example

Consider example from BHE p.257. This is the same CPS data and model we have been working with in our discussion of heteroskedasticity.

# Basic Calculations

- We are interested in the coefficients, variance estimate, and predicted wage for someone with 16 years of education.
- We are also interested in standard errors for all of these quantities.
- For the first two, the information is right in the summary

```
> summary(res.lm)

Call:
lm(formula = lwage ~ education, data = md)

Residuals:
     Min       1Q   Median       3Q      Max
-0.77197 -0.27430 -0.01919  0.23494  0.82188

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.69782    0.70665   0.987  0.33649
education    0.15504    0.04465   3.472  0.00272 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.3998 on 18 degrees of freedom
Multiple R-squared:  0.4012,Adjusted R-squared:  0.3679
F-statistic: 12.06 on 1 and 18 DF,  p-value: 0.002718
```

# Coefficients and Standard Errors

- The coefficients match, but the standard errors we just computed do not match BHE p. 258.

$$\begin{aligned}\log(wage) = & \underset{(0.031)}{0.155} \quad education + \underset{(0.493)}{0.698} \quad + \widehat{e} \\ \widehat{\sigma}^2 = & \underset{(0.043)}{0.144} \\ n = & \, 20.\end{aligned}$$

- Looking back at our earlier table, we see that the reported standard errors match "HC2" errors (which is a little unusual for Stata).

```
> coeftest(res.lm,vcov=vcovHC(res.lm,type="HC2"))

t test of coefficients:

            Estimate Std. Error t value  Pr(>|t|)
(Intercept) 0.697815   0.492771  1.4161    0.1738
education   0.155039   0.030519  5.0801 7.809e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

- Now the standard errors match match, too.

# The Variance Estimate

- Next compute $\widehat{\sigma}^2$.

```
> n <- nrow(md)
> sigHat2 <- (1/n)*sum(res.lm$residuals^2)
> sigHat2
[1] 0.1438872
```

- Note, this is $\widehat{\sigma}^2 = \frac{(n-k)}{n} s^2$, not $s^2$, which explains the lack of bias correction.
- Define a few variables here to clarify the upcoming calculations.

```
> k <- 2
> X <- matrix(c(rep(1,n),md$education),nrow=n,ncol=2)
> u <- matrix(res.lm$residuals,nrow=n)
> b0 <- res.lm$coefficients[1]
> b1 <- res.lm$coefficients[2]
```

# Standard Error of the Variance Estimate

- To compute the standard error on $\widehat{\sigma}^2$, we use the result from p. 146 (which also holds approximately asymptotically) that $\frac{(n-k)s^2}{\sigma^2} \sim \chi^2_{n-k}$.
- It follows that

$$\begin{aligned}
\mathbb{V}[\widehat{\sigma}^2] &= \mathbb{V}\left[\frac{(n-k)}{n}s^2\right] \\
&= \mathbb{V}\left[\frac{(n-k)}{n}\frac{\sigma^2}{(n-k)}\frac{(n-k)}{\sigma^2}s^2\right] \\
&= \mathbb{V}\left[\frac{\sigma^2}{n}\frac{(n-k)}{\sigma^2}s^2\right] \\
&= \frac{\sigma^4}{n^2}2(n-k)
\end{aligned}$$

  which follows from rules for variances and distribution theory for $\chi^2$ random variables.
- Our analogy estimator is the square root of the variance $\frac{2(n-k)\widehat{\sigma}^2}{n^2}$

```
> vcovSigHat2 <- sqrt( 2*(n-2)*(sigHat2^2)/(n^2) )
> vcovSigHat2
[1] 0.04316615
```

# The Mean Wage Estimate

- We are interested in $\mathbb{E}[\text{wage}|\text{education} = 16]$.
- But note that we estimated using log wages, so we have a little bit of algebra to get to the value we want.

$$\begin{aligned}
\mu &= \mathbb{E}[\text{wage}|\text{education} = 16] \\
&= \mathbb{E}\left[e^{\beta_0 + 16\beta_1 + u}\right] \\
&= e^{\beta_0 + 16\beta_1}\mathbb{E}\left[e^{u}\right] \\
&= e^{\beta_0 + 16\beta_1 + \frac{\sigma^2}{2}}
\end{aligned}$$

- The third step comes from an assumption that $u$ is independent of the level of education.
- The fourth step comes from $u$ distributed normal and the expectation of the normal moment generating function.

# The Mean Wage Estimate

- It follows that $\widehat{\mu} = e^{\widehat{\beta_0} + 16\widehat{\beta_1} + \frac{\widehat{\sigma}^2}{2}}$

```
> exp(b0+16*b1+sigHat2/2)
(Intercept)
   25.80016
```

- This matches what is found in the text

$$\widehat{\mu} = \exp\left(16\widehat{\beta}_1 + \widehat{\beta}_2 + \widehat{\sigma}^2/2\right) = \begin{array}{c} 25.80 \\ (2.29) \end{array}$$

# The Standard Error of the Mean Wage Estimate

- The mean wage estimate involves both $\widehat{\boldsymbol{\beta}}$ and $\widehat{\sigma}^2$, so our estimate of the standard error is going to need to use both.
- Even though we don't need to jointly estimate $\widehat{\boldsymbol{\beta}}$ and $\widehat{\sigma}^2$, here's an instance where we need the score and Hessian for the joint estimator to compute the statistic of interest.
- The two key parts are the derivative of the function of the parameters, $h(\cdot)$, and an estimate of the covariance matrix of the parameters $\widehat{\mathbb{V}(\boldsymbol{\beta}, \sigma^2)}$.
- For the first part, we have $h(\boldsymbol{\beta}, \sigma^2) = e^{\beta_0 + 16\beta_1 + \frac{\sigma^2}{2}}$, so

$$\begin{bmatrix} \frac{\partial h}{\partial \beta_0} \\ \frac{\partial h}{\partial \beta_1} \\ \frac{\partial h}{\partial \sigma^2} \end{bmatrix} = \begin{bmatrix} e^{\beta_0 + 16\beta_1 + \frac{\sigma^2}{2}} \\ 16e^{\beta_0 + 16\beta_1 + \frac{\sigma^2}{2}} \\ \frac{1}{2}e^{\beta_0 + 16\beta_1 + \frac{\sigma^2}{2}} \end{bmatrix}$$

# The Standard Error of the Mean Wage Estimate

- The "model trusting" version of the standard error just uses the estimate of the negative inverse Hessian for the parameter vector $(\boldsymbol{\beta}, \sigma^2)$, which we saw from last class:

$$\begin{aligned}\mathbf{A}^{-1} &\equiv -E[\widehat{H(\boldsymbol{\beta}, \sigma^2)}|\mathbf{x}]^{-1} \\ &= \begin{bmatrix} \widehat{\sigma}^2 \mathbf{X}^\top \mathbf{X}^{-1} & \mathbf{0}_{k \times 1} \\ \mathbf{0}_{1 \times k} & \frac{2\widehat{\sigma}^4}{n} \end{bmatrix}\end{aligned}$$

- This also matches what was found in the text

```
> A <- matrix(0,3,3)
> A[1:2,1:2] <- (1/sigHat2)* crossprod(X)
> A[3,3] <- n/(2*(sigHat2^2))
> VcovEst <- solve(A)
> muHat <- exp(b0 + 16*b1 + sigHat2/2)
> dMuHat_db <- matrix(c(muHat,16*muHat,(1/2)*muHat),ncol=1)
> sqrt(t(dMuHat_db) %*% VcovEst %*% dMuHat_db)
         [,1]
[1,] 2.289307
```

# Comments on the Example

- This example has walked through how to compute standard errors for several different statistics from a "typical" regression.
- First, we did the simple exercise of computing standard errors on coefficients. These standard errors can be used to build Wald-style confidence intervals for the true parameter $\beta_k$ by just going out, e.g., $\pm 1.96 \text{se}_k$ on either side of the estimate $\widehat{\beta_k}$.
- The estimator for the variance of $\widehat{\sigma}^2$ is less common to see, but we saw how a little bit of distribution theory from the normal regression model can be really helpful. A confidence interval for $\widehat{\sigma}^2$ will not be symmetric around the coefficient estimate due to the asymmetry of the $\chi^2$ distribution.

# Comments on the Example

- We saw that even though we don't have to jointly estimate $\boldsymbol{\beta}$ and $\sigma^2$ in a maximum likelihood setting, it can sometimes be useful to have access to the score and Hessian matrices from joint estimation.
- It is a little strange to be reporting misspecification-robust standard errors ("HC2") for the coefficients, but "model-trusting" standard errors for the wage estimate.
- If you want to really check your understanding, you can try to compute the "misspecification robust" version of the estimate of $\widehat{\mu}$, by estimating the outer product of the score $\mathbf{B}$ (which is not block diagonal), and then taking the covariance of $(\widehat{\boldsymbol{\beta}}, \widehat{\sigma}^2)$ to be $\mathbf{A}^{-1}\mathbf{B}\mathbf{A}^{-1}$, which gives a standard error of 2.901.

# Comments on the Example

- While pedagogically useful, this has been a fairly heavy lift for a log-wage regression.
- Before we consider a modern, CPU-cyle intensive, complement for the delta method, let's consider one other alternative.
- Since we think that wages are roughly an exponential in the linear predictor, we could just run a Poisson regression – now that we know how to GLM.
- The resulting prediction for the mean wage for a college graduate (education $= 16$) and its standard error are very close to what we achieved through the log-linear model we previously estimated.
- But because the scale parameter $\sigma^2$ does not appear in the linear predictor for the Poisson, the standard error calculation is dramatically simpler.

# A Poisson Regression

```
> res.glm <- glm(wage~education,family=quasipoisson(link="log"),data=md)
> summary(res.glm)

Call:
glm(formula = wage ~ education, family = quasipoisson(link = "log"),
    data = md)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-3.2691  -1.5297  -0.4488   1.1069   4.9276

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.88161    0.81836   1.077  0.29558
education    0.14805    0.05006   2.957  0.00843 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for quasipoisson family taken to be 4.285092)

    Null deviance: 112.410  on 19  degrees of freedom
Residual deviance:  72.185  on 18  degrees of freedom
AIC: NA

Number of Fisher Scoring iterations: 4
> pred.glm <- predict(res.glm,data.frame(education=16),type="response",se.fit=TRUE)
> pred.glm$fit
      1
25.8019
> pred.glm$se.fit
       1
2.375681
```

# Resampling

- Cheap computing power offers an alternative to, and occasionally an improvement on, the use of asymptotic theory to approximate the sampling distributions of estimators.
- We start with an estimator $T(\mathbf{W})$ and a sample $\mathbf{W} = \{\mathbf{w}_i\}_{i=1}^N$ from our population with distribution $F$. We want to know the distribution of $T(\cdot)$:

$$P(T(\mathbf{W}) \leq t)$$

- As we saw with marginal effects under maximum likelihood, even approximating the variance of this distribution can be an involved process. Generally, the distribution of $T(\cdot)$ is a complicated function of $F$, $t$ and the sample size $N$.

# The Jackknife

- The jackknife is a resampling method most commonly used for variance estimation.
- It is computed from the collection of "leave one out" estimators $\{\boldsymbol{\beta}_{(-i)}\}_{i=1}^{n}$, formed by training an estimator on the $n$ distinct subsets of size $n-1$ that can be taken from $\mathbf{y}$,$\mathbf{X}$.
- It is computed as

$$\widehat{V}^{\text{jack}} = \frac{(n-1)}{n} \sum_{i=1}^{n} (\widehat{\boldsymbol{\beta}}_{(-i)} - \overline{\boldsymbol{\beta}})(\widehat{\boldsymbol{\beta}}_{(-i)} - \overline{\boldsymbol{\beta}})^{\top}$$

  where $\overline{\boldsymbol{\beta}}$ is the mean of the jackknife estimates, $\frac{1}{n}\sum_{i=1}^{n} \widehat{\boldsymbol{\beta}}_{(-i)}$.
- It is a $k \times k$ matrix – an outer product of the coefficient vectors.
- If you can train on $n$ observations, then you can train on $n-1$, so an advantage to the jackknife is that it's always available.
- But it also requires $n$ training runs, which may be costly for complex models.

# The Jackknife

- The jackknife is a kind of double-analogy estimator,
  - The jackknife estimates of the coefficients each replicate a training run analogous to our estimation on the full data set – we compute the jackknife estimates the same way we train on the full data set.
  - Variation in the jackknife estimates around the ensemble estimate $\overline{\boldsymbol{\beta}}$ (observed) is analogous to variation (the sampling distribution) of $\widehat{\boldsymbol{\beta}}$ around $\boldsymbol{\beta}$ (unobserved).
- The jackknife is a complement to standard error estimates based on asymptotics either directly or via the delta method.

# The Jackknife

- The code below implements the jackknife estimator ("by hand") for the small data set from our earlier example.

```
> vJack.iter <- data.frame(b0=rep(NA,n),b1=rep(NA,n),sig2Hat=rep(NA,n),muHat=rep(NA,n))
> res.jack <- lapply(1:nrow(md),
+         function(i){
+         res.temp <- lm(lwage ~ education, data=md[-i,])
+         uHat <- res.temp$residuals
+         sig2Hat = sum(uHat^2)/(nrow(md)-1)
+         vJack.iter$b0[i] <<- b0 <- res.temp$coefficients[1]
+         vJack.iter$b1[i] <<- b1 <- res.temp$coefficients[2]
+         vJack.iter$sig2Hat[i] <<- sig2Hat <- sum(res.temp$residuals^2)/(nrow(md)-1)
+         vJack.iter$muHat[i] <<- exp(b0+16*b1+sig2Hat/2)
+         })
> vJack.means <- vJack.iter |> summarize_all(mean)
> vJack.est <- data.frame(b0=0.698,b1=0.155,sig2Hat=.144,muHat=25.80)
> vJack.asy <- data.frame(b0=0.493,b1=0.031,sig2Hat=0.043,muHat=2.29)
> vJack.vcov <- matrix(0,nrow=4,ncol=4)
> for(i in 1:nrow(md)){
+   vJack.vcov = vJack.vcov +
+     crossprod(matrix(as.numeric(vJack.iter[i,]-vJack.means),nrow=1))
+ }
> vJack.se <- sqrt(((n-1)/n)*diag(vJack.vcov))
> round(vJack.se,3)
[1] 0.514 0.032 0.046 2.392
> vJack.asy
     b0    b1 sig2Hat muHat
1 0.493 0.031   0.043  2.29
```

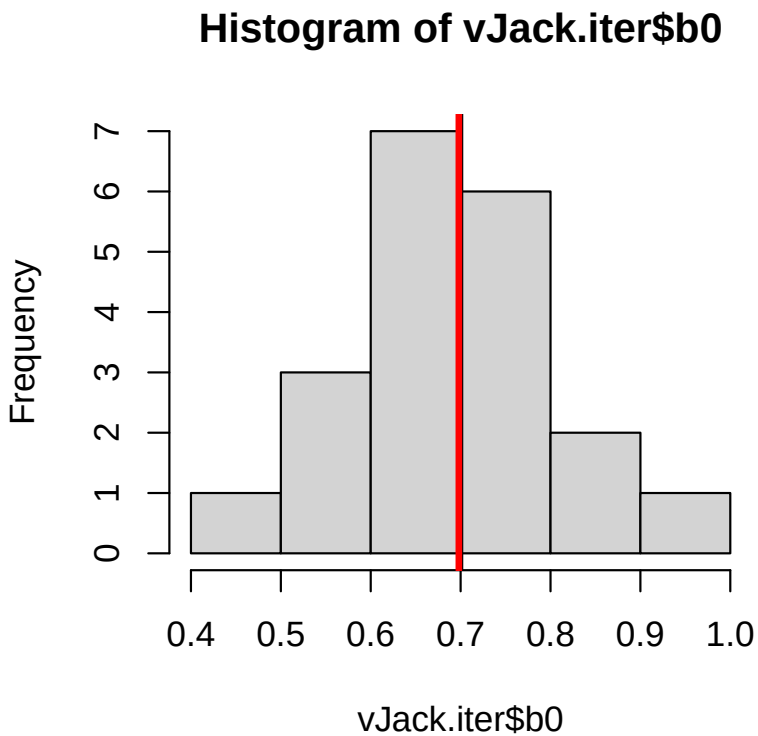
- Note the similarity of `vJack.se` (jackknife) and `vJack.asy` (asymptotic).

# The Jackknife

```
> vJack.iter
          b0        b1   sig2Hat    muHat
1  0.7644407 0.1502651 0.1501709 25.63010
2  0.7982208 0.1478447 0.1485321 25.48276
3  0.7390956 0.1526368 0.1512300 25.96888
4  0.6945454 0.1564402 0.1444519 26.30644
5  0.7007889 0.1537644 0.1456625 25.37717
6  0.6553704 0.1580801 0.1509369 26.05267
7  0.7053750 0.1517989 0.1139930 24.31646
8  0.8220826 0.1461350 0.1469751 25.37412
9  0.5879746 0.1615833 0.1506348 25.75490
10 0.6934291 0.1569186 0.1388494 26.40494
11 0.5102767 0.1684762 0.1412452 26.48341
12 0.6907141 0.1580821 0.1184050 26.55534
13 0.9744205 0.1389433 0.1411248 26.26076
14 0.4511564 0.1686975 0.1314873 24.92970
15 0.8519773 0.1458537 0.1498343 26.06396
16 0.6955408 0.1560136 0.1480695 26.20086
17 0.5133751 0.1652522 0.1402926 25.21795
18 0.6978262 0.1550341 0.1514601 25.89634
19 0.7415532 0.1519050 0.1509046 25.72557
20 0.6974507 0.1551950 0.1513732 25.95223
> vJack.means
         b0       b1   sig2Hat    muHat
1 0.6992807 0.154946 0.1432817 25.79773
> vJack.est
     b0    b1 sig2Hat muHat
1 0.698 0.155   0.144  25.8
```

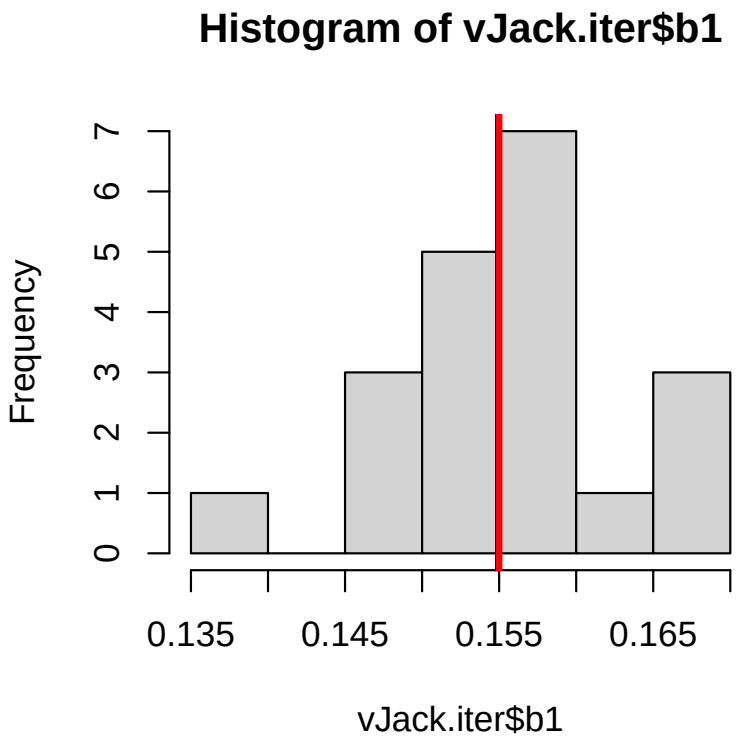
# Jackknife: Sampling Distribution of $\widehat{\beta}_0$

```
> hist(vJack.iter$b0)
> abline(v=vJack.means$b0,lwd=3,col="black")
> abline(v=vJack.est$b0,lwd=3,col="red")
```



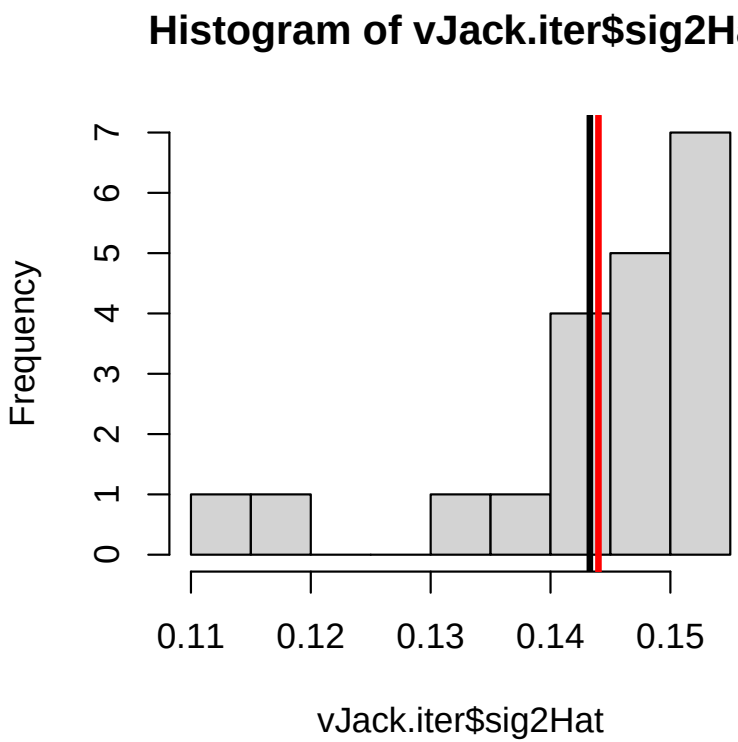
# Jackknife: Sampling Distribution of $\widehat{\beta}_1$

```
> hist(vJack.iter$b1)
> abline(v=vJack.means$b1,lwd=3,col="black")
> abline(v=vJack.est$b1,lwd=3,col="red")
```



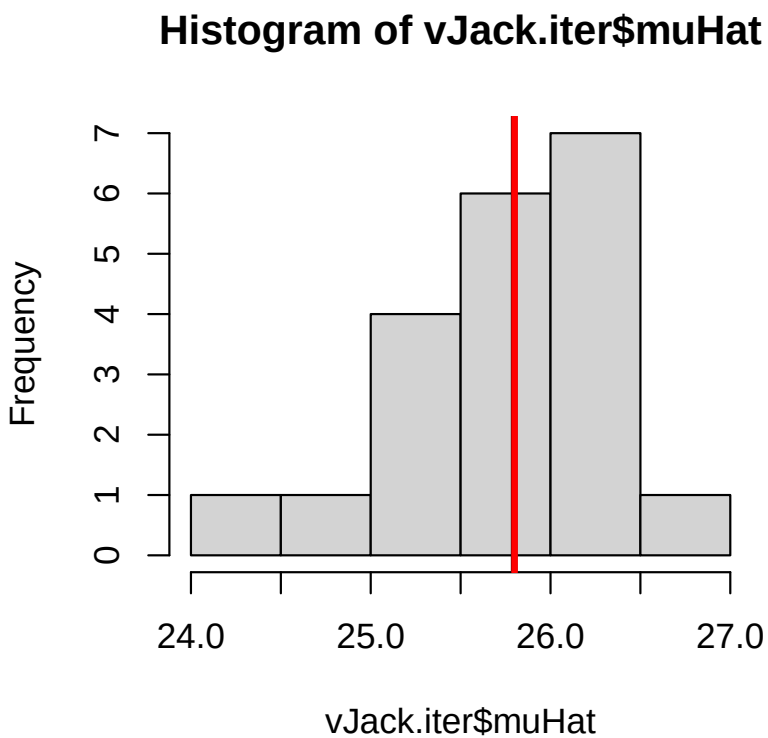
# Jackknife: Sampling Distribution of $\widehat{\sigma}^2$

```
> hist(vJack.iter$sig2Hat)
> abline(v=vJack.means$sig2Hat,lwd=3,col="black")
> abline(v=vJack.est$sig2Hat,lwd=3,col="red")
```



# Jackknife: Sampling Distribution of $\widehat{\mu}$

```
> hist(vJack.iter$muHat)
> abline(v=vJack.means$muHat,lwd=3,col="black")
> abline(v=vJack.est$muHat,lwd=3,col="red")
```



# Bootstrapping

- Our guiding analogy for bootstrapping begins with the notion that our sample is itself a "population" with distribution function $\widehat{F}_N(\mathbf{w})$. Unlike the true population $F$, however, we can draw new samples from $\widehat{F}_N(\mathbf{w})$ at will.
- In large samples, $\widehat{F}_N$ should get close to $F$ (by the Fundamental Theorem of Statistics) so sampling from $\widehat{F}_N$ ought to be a good stand-in for sampling from $F$.
- So if we are interested in the sampling distribution of $T(\mathbf{W})$, we might approximate it by computing its values $T(\mathbf{W}^*)$ across many draws $\mathbf{W}^* = \{\mathbf{w}_i^*\}_{i=1}^N$ from $\widehat{F}_N$. The draws $\{\mathbf{W}_b^*\}_{b=1}^B$ for some large $B$ are called the bootstrap samples, and the process of drawing them is typically called resampling.

# Bootstrapping Regression

- For the regression model

$$y = m(\mathbf{x}, \boldsymbol{\beta}) + u,$$

three types of resampling (i.e., three distinct empirical distributions $\widehat{F}_N$) are commonly used in practice. Having estimated a parametric model, we can then:

  - Resample from the observations $\{\mathbf{w}_i\} = \{(y_i, \mathbf{x}_i)\}$ – the nonparametric or paired bootstrap.
  - Resample from the residuals $\{\widehat{u}_i\}$ – the residual bootstrap.
  - Resample from the estimated model with $y_i \sim F(\mathbf{x}_i, \widehat{\boldsymbol{\theta}})$ – the parametric bootstrap.

- I've ordered them from most-used to least-used, which interestingly also corresponds to ordering them inversely by how much they rely on the correctness of the specified model.

# Bootstrapping: Paired Bootstrap

- The paired bootstrap assumes neither that the model for the conditional mean $m()$ nor that the error specification $u$ is correct.
- This approach simply resamples from the existing sample, so that $\mathbf{w}_i^*$ is simply a row taken at random (i.e., with probability $\frac{1}{N}$ and with replacement) from $\{(y_i, \mathbf{x}_i)\}_{i=1}^N$.
- Replacement is important, since it means that some observations will be sampled more than once in a given bootstrap sample, while others may not be sampled at all.
- The probability that a given observation is in a particular bootstrap sample is roughly $\frac{2}{3}$ (or $\approx 0.63$) in large samples.

# Bootstrapping: Residual Bootstrap

- The residual bootstrap assumes the model for the conditional mean in the DGP is correct, but allows that the distribution of the modeling error $u$ may be misspecified.
- That's why this approach is often described as "intermediate" between the parametric and nonparametric bootstraps.
- Residual bootstrapping can be used without a fully specified model for $u$ – we simply resample from the empirical distribution of the estimated residuals $\widehat{u}$ – so it works particularly well with OLS absent distributional assumptions.
- If $\{u_i^*\}$ is drawn (with replacement) from $\{\widehat{u}_i\}_{i=1}^N$ with probability $\frac{1}{N}$, then $\mathbf{w}_i^* = (m(\mathbf{x}_i, \widehat{\boldsymbol{\beta}}) + u_i^*, \mathbf{x}_i)$.

# Bootstrapping: Parametric Bootstrap

- The parametric bootstrap assumes that we have the population DGP for $y$ correct up to parameters $\boldsymbol{\beta}$.
- Re-run the DGP for $y$ using our parameter estimates via $F(\mathbf{x}_i, \widehat{\boldsymbol{\beta}})$, generating new $y_i^*$.
- In a normal regression model, we would draw $y_i^* \sim N(m(\mathbf{x}_i, \widehat{\boldsymbol{\beta}}), \widehat{\sigma}^2)$, taking $\mathbf{w}_i^* = (y_i^*, \mathbf{x}_i)$.
- This bootstrapping scheme only works if we have a fully specified parametric model for $y$.
- Consistent and asymptotically normal OLS estimates may be available without a fully specified parametric model, in which case the parametric bootstrap cannot be used.

# Bootstrapping

A quick example where we assume a normal regression model applies to

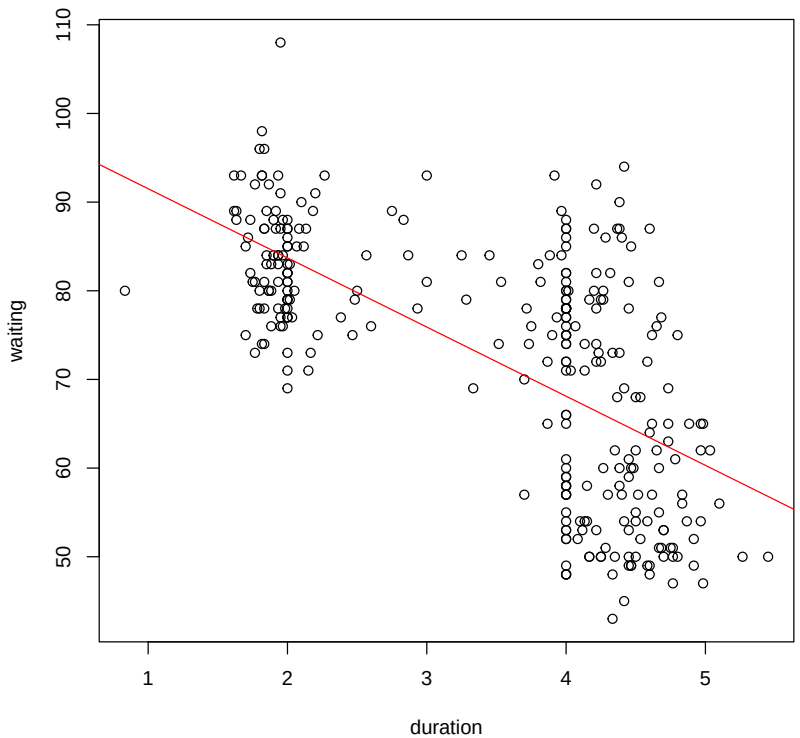
$$y = \beta_0 + \beta_1 x + u$$

where $x$ is the duration of a geyser eruption and $y$ is the waiting time since the last eruption.

```
> library(MASS)

Attaching package:  'MASS'
The following object is masked from 'package:dplyr':
   select
The following object is masked from 'package:wooldridge':
   cement
> data(geyser)
```

# Bootstrapping

```
> plot(waiting~duration,data=geyser)
> geyser.lm <- lm(waiting~duration,data=geyser)
> abline(geyser.lm,col="red")
```

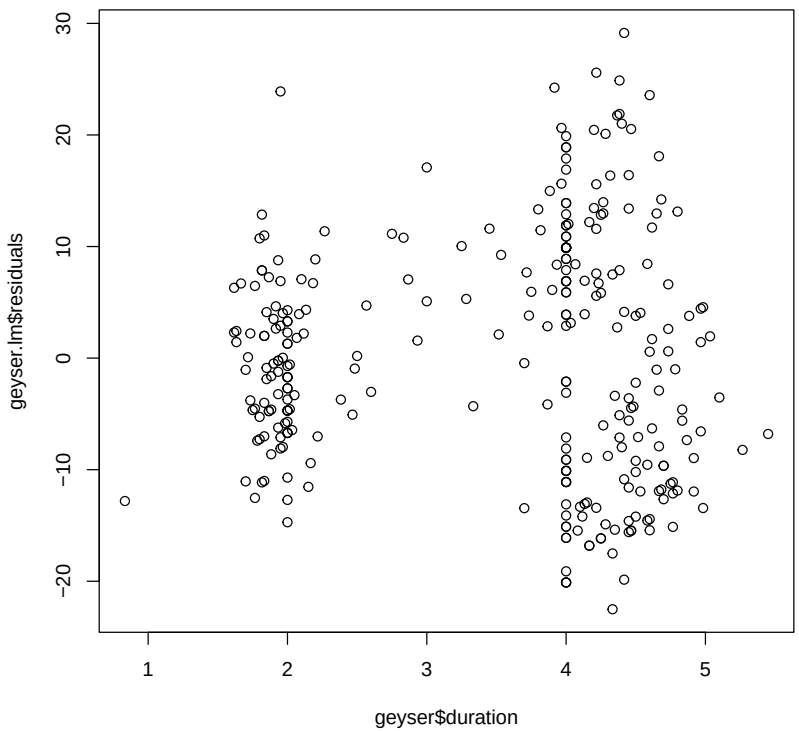


# Bootstrapping

```
> summary(geyser.lm)

Call:
lm(formula = waiting ~ duration, data = geyser)

Residuals:
     Min       1Q   Median       3Q      Max
-22.5084  -8.1683  -0.4892   7.5365  29.1416

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  99.3099     1.9569   50.75   <2e-16 ***
duration     -7.8003     0.5368  -14.53   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 10.64 on 297 degrees of freedom
Multiple R-squared:  0.4155,Adjusted R-squared:  0.4136
F-statistic: 211.2 on 1 and 297 DF,  p-value: < 2.2e-16
```

# Bootstrapping

```
> plot(geyser$duration,geyser.lm$residuals)
```



# Bootstrapping

## A useful function

```
> est.waiting.on.duration <- function(data) {
+     fit <- lm(waiting ~ duration, data=data)
+     return(coefficients(fit))
+ }
```

## Pairs bootstrap of confidence intervals:

```
> resample.pairs <- function(this.data.frame) {
+     idx <- sample(1:nrow(this.data.frame),nrow(this.data.frame),replace=T)
+     return(this.data.frame[idx,])
+ }
> bootstrap.CIs.pairs <- function(B,alpha) {
+     beta.star <- replicate(B,est.waiting.on.duration(resample.pairs(geyser)))
+     low.quantiles <- apply(beta.star,1,quantile,probs=alpha/2)
+     high.quantiles <- apply(beta.star,1,quantile,probs=1-alpha/2)
+     C.l <- 2*coefficients(geyser.lm) - high.quantiles
+     C.u <- 2*coefficients(geyser.lm) - low.quantiles
+     CIs <- rbind(C.l,C.u)
+     return(CIs)
+ }
```

# Bootstrapping

## Residuals bootstrap of confidence intervals:

```
> resample.residuals <- function(lm.results) {
+     new.frame <- lm.results$model
+     new.frame[,1] <- fitted(lm.results) +
+         sample(residuals(lm.results),length(residuals(lm.results)),replace=T)
+     return(new.frame)
+ }
> bootstrap.CIs.residuals <- function(B,alpha) {
+     beta.star <- replicate(B,est.waiting.on.duration(resample.residuals(geyser.lm)))
+     low.quantiles <- apply(beta.star,1,quantile,probs=alpha/2)
+     high.quantiles <- apply(beta.star,1,quantile,probs=1-alpha/2)
+     C.l <- 2*coefficients(geyser.lm) - high.quantiles
+     C.u <- 2*coefficients(geyser.lm) - low.quantiles
+     CIs <- rbind(C.l,C.u)
+     return(CIs)
+ }
```

# Bootstrapping

## Parametric bootstrap of confidence intervals

```
> resample.parametric <- function(lm.results) {
+     new.frame <- lm.results$model
+     uHat <- residuals(lm.results)
+     N <- length(uHat)
+     estVar <- ( uHat^2 )/N
+     new.frame[,1] <- rnorm(N, mean=fitted(lm.results), sd=sqrt(estVar) )
+     return(new.frame)
+ }
> bootstrap.CIs.parametric <- function(B,alpha) {
+     beta.star <- replicate(B,est.waiting.on.duration(resample.parametric(geyser.lm)))
+     low.quantiles <- apply(beta.star,1,quantile,probs=alpha/2)
+     high.quantiles <- apply(beta.star,1,quantile,probs=1-alpha/2)
+     C.l <- 2*coefficients(geyser.lm) - high.quantiles
+     C.u <- 2*coefficients(geyser.lm) - low.quantiles
+     CIs <- rbind(C.l,C.u)
+     return(CIs)
+ }
```

# Bootstrapping

```
> signif(bootstrap.CIs.pairs(B=1e4,alpha=0.05),3)
    (Intercept) duration
C.l        96.6    -8.69
C.u       102.0    -6.93
> signif(bootstrap.CIs.residuals(B=1e4,alpha=0.05),3)
    (Intercept) duration
C.l        95.5    -8.83
C.u       103.0    -6.74
> signif(bootstrap.CIs.parametric(B=1e4,alpha=0.05),3)
    (Intercept) duration
C.l        99.2    -7.85
C.u        99.5    -7.75
```

Are you at all suspicious of the parametric bootstrap results? Why or why not?

# Bootstrapping our Log Wage Regression

- Even with a data set of fairly modest size (our example data has $N = 20$), we can get fairly smooth approximations to the sampling distributions of many estimators of interest.
- Note that the number of distinct bootstrap samples, while very large, is finite. The number of distinct samples of size $s$ (item order irrelevant) taken with replacement from a set of size $N$ is given by

$$C^R(N, s) = \frac{(N + s - 1)!}{s!(N - 1)!}$$

- For bootstrap samples, we take $s = N$, so the total number of distinct bootrap samples is

$$C^R(N, N) = \frac{(2N - 1)!}{N!(N - 1)!}.$$

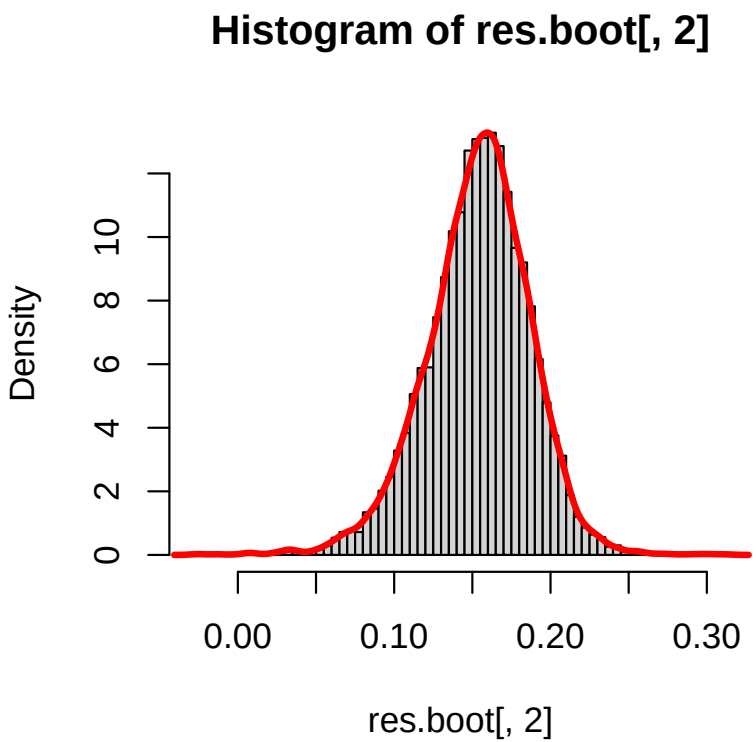
- Even for a sample of size $N = 20$, this is $68,923,264,410$ distinct bootstrap samples.
- From this large pool, we will be sampling $B = 10000$ replicates.

# Bootstrapping our Log Wage Regression

```
> set.seed(12345)
> B <- 10000
> Bn <- nrow(md)
>
> res.boot <- matrix(NA,nrow=B,ncol=length(coef(res.lm))+2,
+     dimnames=list(rep=seq(B), coef=c(names(coef(res.lm)),"sig2Hat","muHat")))
>
> for (i in seq(B)) {
+   boot.data <- md[sample(nrow(md),size=Bn,replace=TRUE),]
+   boot.update <- update(res.lm,data=boot.data)
+   boot.coef <- coef(boot.update)
+   boot.sig2Hat <- sum(residuals(boot.update)^2)/(n-1)
+   boot.muHat <- exp(boot.coef[1]+16*boot.coef[2]+boot.sig2Hat/2)
+   res.boot[i,] <- c(boot.coef,boot.sig2Hat,boot.muHat)
+ }
```

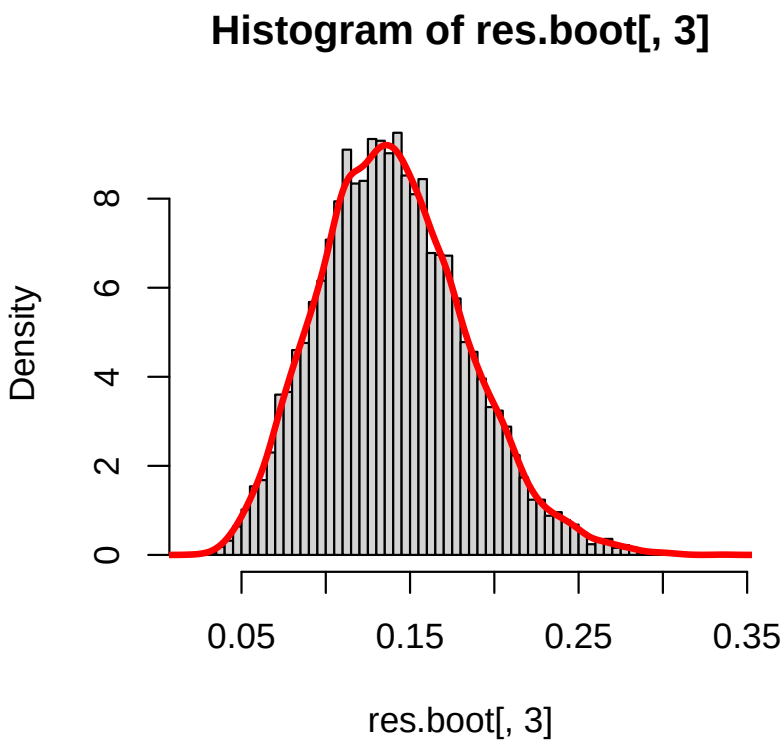
# Bootstrap: Sampling Distribution of $\widehat{\beta}_1$

```
> hist(res.boot[,2],n=100,freq=FALSE)
> lines(density(res.boot[,2]),col="red",lwd=3)
```



# Bootstrap: Sampling Distribution of $\widehat{\mu}$

```
> hist(res.boot[,3],n=100,freq=FALSE)
> lines(density(res.boot[,3]),col="red",lwd=3)
```



# Variance-Covariance Matrix

```
> boot.vcov <- matrix(0,nrow=4,ncol=4)
> boot.means <- res.boot |> as_tibble() |> summarize_all(mean)
> for(i in 1:B){
+   boot.vcov = boot.vcov +
+     crossprod(matrix(as.numeric(res.boot[i,]-boot.means),nrow=1))
+ }
> boot.se <- sqrt(diag((1/(B-1))*boot.vcov))
> boot.se
[1] 0.54188760 0.03336459 0.04322325 2.42883794
```