11/10 Case study with WLS: Jevons gold coins

<u>Context:</u> W. Stanley Jevons collected $n = 274$ British sovereign gold coins in Manchester, England (1868)

⇝ Clean and weigh each coin + record decade of issue

Data:

```
> gold.df <- read.csv("jevons_gold.csv",header=T)
> gold.df
  age sample_size   avg_wt sample_sd min_wt max_wt
1   1         123 7.97315   0.01246  7.922  7.999
2   2          78 7.95033   0.02272  7.892  7.993
3   3          32 7.92756   0.03426  7.848  7.984
4   4          17 7.89671   0.04068  7.827  7.965
5   5          24 7.87333   0.05373  7.757  7.961
```

y (avg_wt), SD (sample_sd)

5 cases total (summary data)

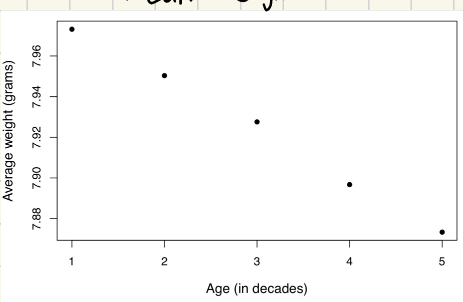
- Age (of coin in decades)
- Sample size = # coins in each age group
- Summary statistics for coin weights (in grams) by age group.

In circulation, coins degrade... older coins are worth slightly less

Newly minted coins supposed to weigh 7.9876 g (legal minimum = 7.9379 g)

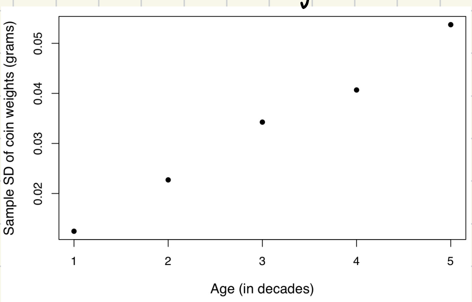
Q: How do the weights of coins change with age?

mean weight




```
plot(gold.df$age,gold.df$avg_wt,pch=16,
     xlab="Age (in decades)",ylab="Average weight (grams)",cex.lab=1.2)
```

SD of weight




```
plot(gold.df$age,gold.df$sample_sd,pch=16,
     xlab="Age (in decades)",
     ylab="Sample SD of coin weights (grams)",cex.lab=1.2)
```

Observations:

1) Sample means decrease linearly with age

2) Spread of weights increases with age

3) Sample sizes decrease with age (mostly)

e.g. 7.87333 is the average of $n_s = 24$ numbers

$\bar{y}_1 = 7.97315$ estimates the average weight of all British sovereign gold coins in Manchester issued between 1858 - 1868

($\bar{y}_1$ = Avg weight of $n_1 = 123$ coins issued during this time)

⇝ We expect this estimate to be more reliable / precise than the previous one (7.87333g)

Model: $\bar{y}_i = \beta_0 + \beta_1 \, age_i + \varepsilon_i$ $\qquad$ $Var(\bar{y}_i)$ is non-constant in i

⇝ WLS

Assume weights of coins of age = i are independent random variables $X_1, \dots, X_{n_i}$ with mean $\mu(i)$ and variance $\sigma(i)^2$, and $\bar{y}_i = \frac{1}{n_i} \sum_{j=1}^{n_i} X_j$ is their average.

Q: What is $Var(\bar{y}_i)$?

$$Var(\bar{y}_i) = \frac{1}{n_i^2} \sum_{j=1}^{n_i} \underbrace{Var(X_j)}_{\sigma(i)^2} = \frac{n_i \, \sigma(i)^2}{n_i^2} = \frac{\sigma(i)^2}{n_i}$$

$\sigma(i)^2$ unobserved $\rightsquigarrow$ plug in $\hat{\sigma}(i)^2 = SD_i^2$ (sample variance)

For WLS, weights are defined $w_i = 1/\mathrm{Var}(\bar{y}_i) \approx n_i/SD_i^2$

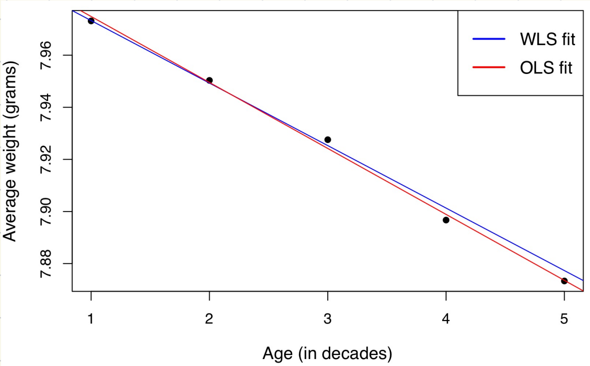
e.g. For age groups 1 & 5 :

$$w_1 = \frac{n_1}{SD_1^2} = \frac{123}{(0.01246)^2} \approx 792,262$$

$$w_5 = \frac{n_5}{SD_5^2} = \frac{24}{(0.05373)^2} \approx 8,313$$

Interpretation: The estimate for the average weight of 1 decade old coins in Manchester during 1868 is about $792.3/8.3 \approx 95$ times more precise than the estimate for 5 decade old coins

```
ols.mod <- lm(avg_wt~age,gold.df)
wls.mod <- lm(avg_wt~age,gold.df,weights=sample_size/sample_sd^2)
```




```
plot(gold.df$age,gold.df$avg_wt,pch=16,
     xlab="Age (in decades)",ylab="Average weight (grams)",
     cex.lab=1.2,main="Jevons' gold coins")
abline(wls.mod,lty=1,col="blue")
abline(ols.mod,lty=1,col="red")
legend("topright",c("WLS fit","OLS fit"),col=c("blue","red"),
       lty=1,lwd=2,cex=1.2)
```

$\rightsquigarrow$ WLS (blue) prioritizes y_1 over y_5

```
summary(ols.mod)
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  8.0001940  0.0026782 2987.16 8.27e-11 ***
age         -0.0253260  0.0008075  -31.36 7.12e-05 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.002554 on 3 degrees of freedom
Multiple R-squared:  0.997,	Adjusted R-squared:  0.9959
F-statistic: 983.7 on 1 and 3 DF,  p-value: 7.122e-05
```

```
summary(wls.mod)
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  7.9973172  0.0010462 7644.43 4.94e-12 ***
age         -0.0239990  0.0007331  -32.73 6.27e-05 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.4706 on 3 degrees of freedom
Multiple R-squared:  0.9972,	Adjusted R-squared:  0.9963
F-statistic:  1072 on 1 and 3 DF,  p-value: 6.266e-05
```

Interpretation (for WLS): With each passing decade, the coin weight decreases on average by 0.024 grams. The estimated weight of a newly minted coin is

$$\widehat{E[\text{weight} \mid \text{age}=0]} = \hat{\beta}_0 + \hat{\beta}_1(0) = 7.9973 \text{ grams}$$

OLS point estimates are very similar $\left(\begin{array}{l} \hat{\beta}_1 = -0.025\,g \\ \hat{\beta}_0 = 8.0002\,g \end{array}\right)$

<u>Remark:</u> In the WLS model $y \sim N(X\beta, \sigma^2 W^{-1})$, the OLS estimator $\hat{\beta}^{OLS} = (X^TX)^{-1}X^Ty$ is still unbiased for β, i.e.

$$E[\hat{\beta}^{OLS}] = (X^TX)^{-1}X^T \underbrace{E[y]}_{X\beta} = \beta$$

, but not as efficient, i.e.

$$Var(\hat{\beta}^{WLS}) \leq Var(\hat{\beta}^{OLS})$$

e.g. the estimated variances are:

```
> vcov(ols.mod)
              (Intercept)           age
(Intercept)  7.172719e-06 -1.956196e-06
age         -1.956196e-06  6.520653e-07
> vcov(wls.mod)
              (Intercept)          age
(Intercept)  1.094456e-06 -6.84043e-07
age         -6.840430e-07  5.37498e-07
```

Diagonal entries represent $\widehat{Var}(\hat{\beta}_0)$ and $\widehat{Var}(\hat{\beta}_1)$ estimated under OLS (top) or WLS (bottom)

$\rightsquigarrow$ CI's for $\hat{\beta}_j$'s computed using WLS will be thinner compared to those computed using OLS method, assuming the WLS model is correct (more efficient inference)

Q: For a randomly sampled 3-decade old coin, what is the predicted weight?

$$\hat{y} = 7.9973 - 0.024\,(3) = 7.925 \text{ grams}$$

Actual weight follows $N(\mu(3), \sigma(3)^2) \approx N(7.925, 0.03426^2)$ ★

95% CI for $\mu(3) = \beta_0 + \beta_1 (3)$

```
> predict(wls.mod,data.frame(age=3),interval="confidence",level=0.95)
       fit       lwr      upr
1 7.92532 7.921018 7.929623
```

Interpretation: 95% confident the average weight among 3-decade old British sovereign coins in Manchester during 1868 was between 7.921g and 7.929g

95% PI for weight:

```
> predict(wls.mod,data.frame(age=3),interval="prediction",
+          level=0.95,weights=1/(gold.df$sample_sd[3]^2))
       fit      lwr      upr
1 7.92532 7.873826 7.976814
```

Q: Why is $w^* = 1/SD(3)^2$? Response variable is average wt

⇝ for a single age 3 coin sampled at random, $n^* = 1$

Interpretation: An age $x = 3$ coin sampled at random has 95% chance to weigh between 7.874g and 7.977g

Q: What fraction of age $x = 3$ coins are below the legal limit of 7.9379g? (Hint: use ★)

11/12 Misspecified variance + Heteroskedastic Consistent (HC) std errors

Example (Gas sniffer data)

Context: Pump gasoline into a tank ⇝ vapor inside tank goes out ⇝ hydro carbon vapor goes into atmosphere (source of pollution)

Goal: Estimate efficiency of a vapor recovery device (to block vapor from reaching atmosphere) based on measurements from a "gas sniffer"

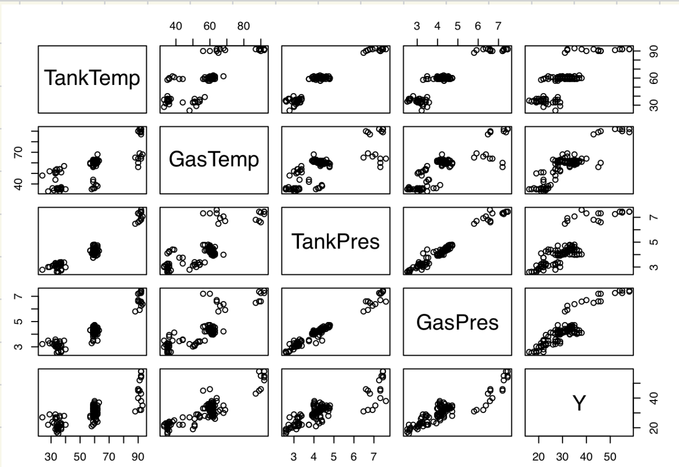
Data:

```
> sniffer.df <- read.csv("sniffer.csv")
> head(sniffer.df)
  TankTemp GasTemp TankPres GasPres  Y
1       28      33     3.00    3.49 22
2       24      48     2.78    3.22 27
3       33      53     3.32    3.42 29
4       29      52     3.00    3.28 29
5       33      44     3.28    3.58 27
6       31      36     3.10    3.26 24
```

Variables:

- Tank temperature (°F)
- Gas temp (°F)
- Initial tank pressure (psi)
- Gas pressure = vapor pressure of dispensed gasoline (psi)
- Y = mass of emitted hydrocarbons (grams)

Pairwise scatterplot matrix:



Obs: 1) Predictors seem ≈ linearly related to y

2) Tank and gas pressures are tightly correlated ⇝ can lead to instability (next week)

3) Variance of y is not obviously constant or non-constant (ambiguous)

Observation 3 + no clear choice of weights $\rightsquigarrow$ proceed provisionally with OLS (assumes constant variance) + be ready to adjust

```
> lmod <- lm(Y~TankTemp+GasTemp+TankPres+GasPres,sniffer.df)
> summary(lmod)
```

```
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.15391    1.03489   0.149   0.8820
TankTemp    -0.08269    0.04857  -1.703   0.0912 .
GasTemp      0.18971    0.04118   4.606 1.03e-05 ***
TankPres    -4.05962    1.58000  -2.569   0.0114 *
GasPres      9.85744    1.62515   6.066 1.57e-08 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 2.758 on 120 degrees of freedom
Multiple R-squared:  0.8933,	Adjusted R-squared:  0.8897
F-statistic: 251.1 on 4 and 120 DF,  p-value: < 2.2e-16
```

Looks reasonable ..

but can these standard errors be trusted?

<u>Claim:</u> If the true model is

$$y_i = \beta^T x_i + \varepsilon_i \quad , \text{ where } \varepsilon_i \sim N(0, \sigma^2/w_i) \text{ independently}$$

then $\mathrm{Var}(\hat{\beta}^{OLS}) \neq \sigma^2 (X^T X)^{-1}$ in general (unless $w_i \equiv 1$)

<u>Key point:</u> If variance is not constant, the OLS estimator $\hat{\beta}^{OLS} = (X^T X)^{-1} X^T y$ is still unbiased; however, $\mathrm{Var}(\hat{\beta}^{OLS})$ will depend on the true variance structure.

<u>Solution:</u> Use the same point estimate but a different (robust) SE

- This approach will give valid SE without assuming constant variance
- Doesn't require us to explicitly model the variance like WLS does

Variance of $\hat{\beta}^{OLS}$ under heteroskedastic noise

Suppose $y \sim N(X\beta, \sigma^2 W^{-1})$

$\hookrightarrow$ means $\text{diag}(w_1^{-1}, \ldots, w_n^{-1})$

Then $\text{Var}(\hat{\beta}^{OLS}) = \text{Var}(My)$ where $M = (X^TX)^{-1}X^T$

$$= M \underbrace{\text{Var}(y)}_{\sigma^2 W^{-1}} M^T$$

$$= (X^TX)^{-1} \underbrace{X^T \sigma^2 W^{-1} X}_{\text{filling}} (X^TX)^{-1} \qquad \text{(Sandwich formula)}$$

bread

Given this sandwich matrix, could deduce $se(\hat{\beta}^{OLS})$ from diagonal entries

Issue: We don't know $\sigma^2 W^{-1}$ $\rightsquigarrow$ estimate it from data

Eicker-Huber-White (EHW) estimator

Replace $\sigma^2 W^{-1}$ in the sandwich formula by:

$$\hat{\Sigma} = \text{diag}(e_1^2, \ldots, e_n^2) = \begin{pmatrix} e_1^2 & & 0 \\ & \ddots & \\ 0 & & e_n^2 \end{pmatrix} \quad \text{where } e_i = y_i - \hat{y}_i$$

Intuition: If $e_i \approx \varepsilon_i \sim N(0, \sigma^2/w_i)$, then $E[e_i^2] \approx \sigma^2/w_i$

so that $E[\hat{\Sigma}] \approx \sigma^2 W^{-1}$

Consistency property: $\hat{\Sigma} \to \sigma^2 W^{-1}$ as $n \to \infty$

⇝ EHW gives (asymptotically) correct standard errors for OLS estimator when variance is non-constant. a.k.a HC estimator
"heteroskedasticity consistent"

An adjustment for small sample settings (HC3 estimator)

$$\hat{\Sigma}^{HC3} = \operatorname{diag}\Big(\underbrace{e_1^2/(1-h_{11})^2}_{\approx \sigma_i^2}, \ldots, e_n^2/(1-h_{nn})^2 \Big)$$

(Divide by $(1-h_{ii})^2$ since $Var(e_i)$ depends on leverage h_{ii})

⇝ Standard errors for $\hat{\beta}_0^{OLS}, \ldots, \hat{\beta}_p^{OLS}$ given by sqrt of diagonal entries in $\widehat{Var}(\hat{\beta}^{OLS}) = (X^TX)^{-1} X^T \hat{\Sigma}^{HC3} X (X^TX)^{-1}$

Manual computation in R for sniffer data:

```
> lmod <- lm(Y~TankTemp+GasTemp+TankPres+GasPres,sniffer.df)
> levs <- hatvalues(lmod)
> X <- model.matrix(lmod)
> ehw <- solve(t(X) %*% X) %*%
+   t(X) %*% diag(residuals(lmod)^2/(1-levs)^2) %*% X %*%
+   solve(t(X) %*% X)
> sqrt(diag(ehw))
(Intercept)    TankTemp     GasTemp    TankPres     GasPres
 1.04734551  0.04444225  0.03380072  1.97239041  2.05585118
```

R function "vcovHC":

```
> library(sandwich)
> var.ols <- vcovHC(lmod,type="HC3")
> sqrt(diag(var.ols))
(Intercept)    TankTemp     GasTemp    TankPres     GasPres
 1.04734551  0.04444225  0.03380072  1.97239041  2.05585118
```

Compare with uncorrected:

```
> summary(lmod)$coefficients[,2]
(Intercept)    TankTemp     GasTemp    TankPres     GasPres
 1.03488948  0.04856801  0.04118370  1.58000429  1.62515157
```

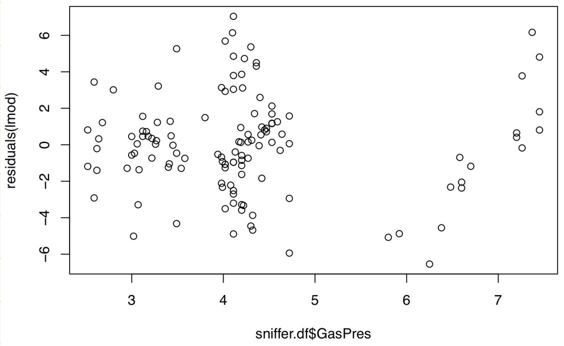
relative % Change: 1.2% -8.5% -17.9% +24.7% +26.5%

Interpretation: HC confidence intervals are adjusted to account for misspecified variance structure (as large as ≈ 27% wider in sniffer data)

e.g. $\frac{1.972 - 1.58}{1.58} = 0.247$

Residuals vs Gas Pressure: One could argue that variance increases with Gas Pressure ⤳ might try

$$Var(y|x) = \sigma^2 \, GasPres$$



& fit WLS with $w_i = 1/GasPres_i$

```
wls.lmod <- lm(Y~TankTemp+GasTemp+TankPres+GasPres,sniffer.df,weights=1/GasPres)
summary(wls.lmod)
```

```
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.50006    1.01707   0.492   0.6239
TankTemp    -0.06057    0.04522  -1.339   0.1830
GasTemp      0.18501    0.04047   4.571 1.19e-05 ***
TankPres    -3.26153    1.54518  -2.111   0.0369 *
GasPres      8.73210    1.60301   5.447 2.76e-07 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.318 on 120 degrees of freedom
Multiple R-squared:  0.8812,	Adjusted R-squared:  0.8772
F-statistic: 222.5 on 4 and 120 DF,  p-value: < 2.2e-16
```

cf. OLS fit

```
Coefficients:
            Estimate Std. Error
(Intercept)  0.15391    1.03489
TankTemp    -0.08269    0.04857
GasTemp      0.18971    0.04118
TankPres    -4.05962    1.58000
GasPres      9.85744    1.62515
```

Takeaways:

- WLS ≈ OLS for this data since heteroskedasticity is mild
- HC3 intervals are valid under heteroskedasticity and can be less efficient, but don't require us to specify weights

Lecture on the Bootstrap

Motivation: Standard intervals for parameter inference,

$$\hat{\theta} \pm 1.96\,\widehat{se}$$

e.g. 1) $\hat{\theta} = \bar{X}_n$, $\widehat{se} = S/\sqrt{n}$, relies on: CLT, $n \to \infty$

2) $\hat{\beta}_1 = \frac{S_{xy}}{S_{xx}}$, $\widehat{se} = \sqrt{\hat{\sigma}^2/S_{xx}}$, relies on: Constant variance, normality

Q: If n is small, and se formula is hard to calculate, or normality doesn't hold, what can we do?

e.g. sample median: no simple formula for standard error

($se = \sigma/\sqrt{n}$ only holds for sample mean)

Thought expt

Suppose $y_1, \ldots, y_n \sim G$ and we know what G is. To assess variability of the estimator $\hat{\theta} = \text{median}(y_1, \ldots, y_n)$,

1) Draw a new sample $y_1^*, \ldots, y_n^* \sim G$
2) Compute and record $\text{median}(y_1^*, \ldots, y_n^*)$
3) Repeat steps 1 and 2, e.g. $B = 1000$ times

From the 1000 sample medians, plot histogram & check spread

$se(\hat{\theta}) \approx$ std dev. of histogram

a.k.a. "sampling distribution of $\hat{\theta}$"

In practice, we don't know the population distribution G.

Bootstrap idea (Efron, 1979): Treat the dataset as our "population".

1) Sample from it (with replacement) n times and record the estimate computed from resampled data

2) Repeat many times & look at the spread of estimates.

$\leadsto$ approximates the sampling distribution of the estimator

Example: transaction data

For each of $n = 261$ branches of an Austrialian bank, we have:

1) t_1 = # of transactions of type 1

2) t_2 = # of transactions of type 2

3) time = number of minutes of labor used by branch (response variable)

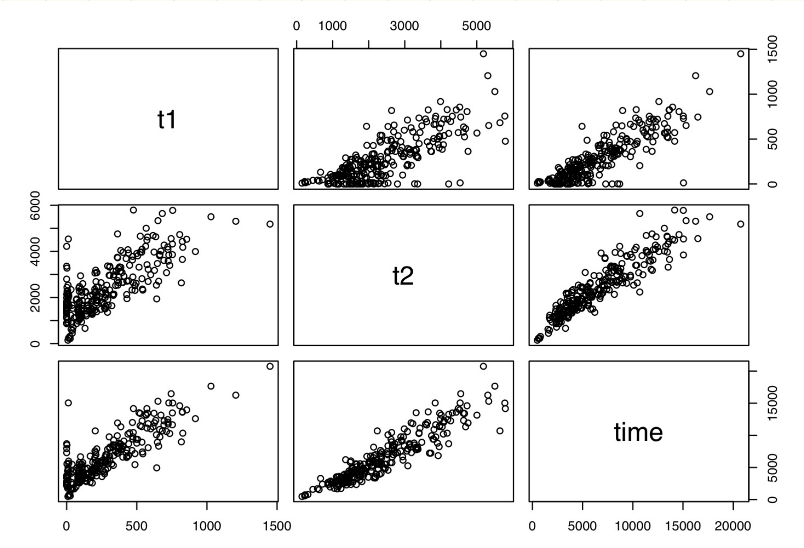
Data:

```
> transaction.df <- read.csv("transaction.csv")
> head(transaction.df)
   t1   t2  time
1   0 1166  2396
2   0 1656  2348
3   0  899  2403
4 516 3315 13518
5 623 3969 13437
6 395 3087  7914
```

$n = 261$

Q: How long do type 1 transactions take to process? How much longer do they take than type 2 transactions?

```
> pairs(transaction.df)
```



$\approx$ linear with variance increasing in # transactions

OLS fit:

```
lmod <- lm(time~t1+t2,transaction.df)
summary(lmod)
```

```
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 144.36944  170.54410   0.847    0.398
t1            5.46206    0.43327  12.607   <2e-16 ***
t2            2.03455    0.09434  21.567   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1143 on 258 degrees of freedom
Multiple R-squared:  0.9091,    Adjusted R-squared:  0.9083
F-statistic:  1289 on 2 and 258 DF,  p-value: < 2.2e-16
```

(Don't trust the Standard errors)

Interpretation: $\hat{\beta}_1 = 5.46$ (mins per type 1 transaction)

$\hat{\beta}_2 = 2.03 \rightsquigarrow$ Type 1 transactions take $\hat{\beta}_1/\hat{\beta}_2 \approx 2.7$ times as long to process as type 2 transactions.

Q: How can we quantify our uncertainty in these point estimates?

For standard errors of $\hat{\beta}$, we could try:

1) Deriving weights for WLS
2) Sandwich estimator (EHW or HC3)

neither of these give $se(\hat{\beta}_1/\hat{\beta}_2)$

3) Bootstrap (more versatile but computationally heavier)

Case resampling Bootstrap

Procedure: 1) Enumerate the rows of our (original) data matrix from 1 to n, and sample (with replacement) n rows

2) Stack the n sampled rows into a new data matrix
[Some of the cases will appear $\geq$ once, some won't appear]
Compute the regression on the new data set & save the values of the fitted coefficients $\hat{\beta}$

3) Repeat steps 1 + 2 many times, e.g. R = 10,000 times

4) Compute the 2.5% and 97.5% quantiles of $\{\hat{\beta}_j^{(b)}\}_{b=1}^{B}$ for each $j = 1, \ldots, p$ (These are the bootstrap CI's)

(percentile bootstrap intervals)

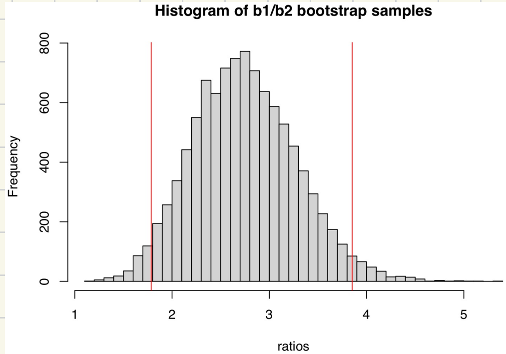
Advantages of bootstrap method:

- No distributional assumptions on errors (e.g. normality)

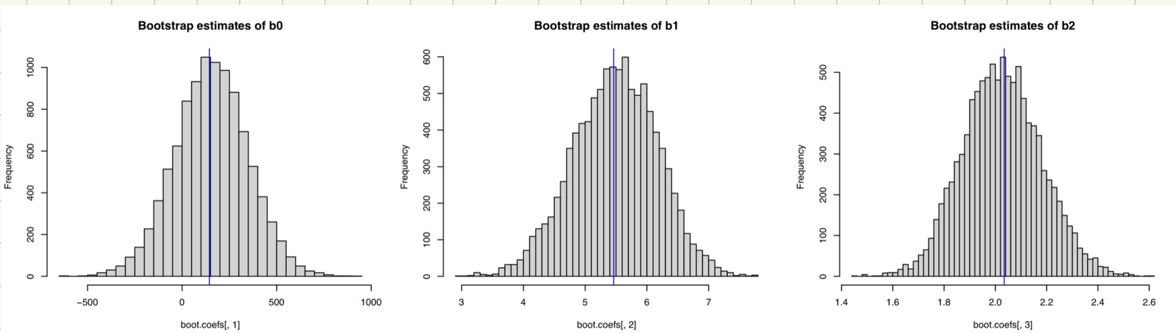
- Valid under non-constant variance (like sandwich SEs)
- Also works for nonlinear functions of β, e.g. β_1/β_2

Example: 95% CI for β_1/β_2 in transaction data

```r
B <- 10000
n <- nrow(transaction.df)
boot.coefs <- matrix(NA,nrow=B,ncol=3)
set.seed(1)
for (b in 1:B) {
  # Resample row indices
  boot.indices <- sample(1:n,size=n,replace=TRUE)
  # Create bootstrap dataset
  df.boot <- transaction.df[boot.indices,]
  # Fit model on bootstrap data
  boot.lmod <- lm(time~t1+t2,data=df.boot)
  # Store bootstrap estimate
  boot.coefs[b,] <- coefficients(boot.lmod)
}
# Compute beta1 / beta2 ratio for each bootstrap sample
ratios <- boot.coefs[,2]/boot.coefs[,3]
hist(ratios,breaks=50,main="Histogram of b1/b2 bootstrap samples")
boot.CI <- quantile(ratios,c(0.025,0.975)); boot.CI
abline(v=boot.CI,col="red")
```



$\rightsquigarrow$ 95% CI for β_1/β_2: $(1.79, 3.85)$




```r
# Plot histograms for beta0, beta1, beta2 estimates
par(mfrow=c(1,3))
hist(boot.coefs[,1],main="Bootstrap estimates of b0",breaks=50)
abline(v=lmod$coefficients[1],col="blue")
hist(boot.coefs[,2],main="Bootstrap estimates of b1",breaks=50)
abline(v=lmod$coefficients[2],col="blue")
hist(boot.coefs[,3],main="Bootstrap estimates of b2",breaks=50)
abline(v=lmod$coefficients[3],col="blue")
```

distributions are $\approx$ normal $\rightsquigarrow$ Could also use, e.g. $\hat{\beta}_2 \pm 2$ boot.se

```
> # Comparison between OLS, bootstrap, and HC3 standard errors
> summary(lmod)$coefficients[,2]
 (Intercept)          t1          t2
170.54410348  0.43326792  0.09433682
> boot.se <- apply(boot.coefs,2,sd); boot.se
[1] 192.8369373   0.6817002   0.1530175
> HC3.se <- sqrt(diag(vcovHC(ols.mod,"HC3"))); HC3.se
(Intercept)          t1          t2
203.1588383   0.7288087   0.1634533
```

Bootstrap & HC3 corrections are similar. HC3 is slightly more conservative

If assumptions 1) const variance & 2) linearity are satisfied then residual bootstrap is more efficient (also works if errors aren't normal or for a non-linear function $f(\beta)$)

Residual bootstrap

1) Fit the model & compute residuals $e = y - \hat{y}$

2) Sample from residual vector with replacement n times

$\rightsquigarrow e_1^*, \dots, e_n^*$

3) Create new responses: $y_i^* = \hat{y}_i + e_i^*$ for $i = 1, \dots, n$ and refit lm on $(x_1, y_1^*), \dots, (x_n, y_n^*)$ $\rightsquigarrow$ save the fitted coefficients

4) Repeat steps 2 + 3, e.g. $R = 10000$ times & compute CIs as done in case resampling bootstrap

```
> # automatic function
> set.seed(1)
> out.case <- Boot(lmod,R=10000,method="case")
> head(out.case$t)
     (Intercept)       t1       t2
[1,]   143.63604 5.266164 2.084432
[2,]   286.56843 6.392418 1.870808
[3,]   390.45745 6.700745 1.761827
[4,]   464.32524 6.361305 1.799856
[5,]   399.35939 6.090188 1.809003
[6,]   -26.42866 5.962041 2.062347
> apply(out.case$t,2,mean)
(Intercept)          t1          t2
 154.935482    5.487804    2.026705
> out.case
```

```
Bootstrap Statistics :
       original         bias     std. error
t1*  144.369443  10.566039662  191.2784466
t2*    5.462057   0.025747607    0.6725316
t3*    2.034549  -0.007843693    0.1510532
```

e.g. $144.37 + 10.57 = 154.94$

Remember to run library(car) before running Boot() in R!

```
> out.res <- Boot(lmod,R=10000,method="residual")
> out.res

ORDINARY NONPARAMETRIC BOOTSTRAP


Call:
boot::boot(data = dd, statistic = boot.f, R = R, .fn = f, parallel = p_type,
    ncpus = ncores, cl = cl2)


Bootstrap Statistics :
       original        bias    std. error
t1* 144.369443 -9.170967e-02 173.74198822
t2*   5.462057  3.254262e-03   0.43849173
t3*   2.034549 -3.078375e-05   0.09618935
```

Similar to OLS std errors

When to use the various methods:

Situation	Method	Why?
Constant variance and 1) normal errors or 2) Large n, e.g. >50	OLS SE	1) $\Rightarrow$ Normal $\hat{\beta}$ (exactly) 2) $\Rightarrow$ CLT applicable ($\approx$)
Heteroskedasticity (and we can model it e.g. $Var(\varepsilon_i) = \sigma^2 g(x_i)$)	WLS SE	More efficient (if our variance model is correct)
Heteroskedasticity and don't know how to model it (more typical)	Sandwich SE	Valid asymptotically ($n \to \infty$) Computationally cheap
Heteroskedasticity or asymmetric error dbn, or need CI for $f(\beta)$, e.g. $f(\beta) = \beta_1/\beta_2$	Case bootstrap $\Rightarrow$ residual bootstrap	Accounts for skew (CI not nec. symmetric) + Simulates sampling dbn of non-linear estimates e.g. $\hat{\beta}_1/\hat{\beta}_2$