Econ 675: HW 1

Erin Markiewitz

September 28, 2018

Contents

1 Simple Linear Regression w/ Measurement Error **3**
- 1.1 . . . 3
- 1.2 . . . 3
- 1.3 . . . 4
- 1.4 . . . 4
- 1.5 . . . 4
- 1.6 . . . 4
- 1.7 . . . 5
- 1.8 . . . 5
- 1.9 . . . 5
- 1.10 . . . 5
- 1.11 . . . 5
- 1.12 . . . 6

2 Implementing Least-Squares Estimators **6**
- 2.1 . . . 6
- 2.2 . . . 6
- 2.3 . . . 6
- 2.4 . . . 6
- 2.5 . . . 7
  - 2.5.1 . . . 7
  - 2.5.2 . . . 7

3 Analysis of Experiments **9**
- 3.1 . . . 9
  - 3.1.1 . . . 9
  - 3.1.2 . . . 9
- 3.2 . . . 9
  - 3.2.1 . . . 10
  - 3.2.2 . . . 10
- 3.3 . . . 11
  - 3.3.1 . . . 11
  - 3.3.2 . . . 11

4 Code Appendix **12**

1 Simple Linear Regression w/ Measurement Error

1.1

By LLN,

$$\begin{aligned}\hat{\beta_{LS}} &= \frac{\tilde{\mathbf{x}}'\mathbf{x}}{\tilde{\mathbf{x}}'\tilde{\mathbf{x}}}\beta + \frac{\tilde{\mathbf{x}}'\epsilon}{\tilde{\mathbf{x}}'\mathbf{x}} = \frac{\tilde{\mathbf{x}}'\mathbf{x}/n}{\tilde{\mathbf{x}}'\tilde{\mathbf{x}}/n}\beta + \frac{\tilde{\mathbf{x}}'\epsilon/n}{\tilde{\mathbf{x}}'\mathbf{x}/n} \\ &\to_p \frac{\mathbb{E}[(x_i+u_i)x_i]}{\mathbb{E}[(x_i+u_i)^2]}\beta = \frac{\sigma_x^2}{\sigma_x^2+\sigma_u^2}\beta = \lambda\beta\end{aligned} \tag{1}$$

As $\sigma_x^2, \sigma_u^2 > 0 \implies \frac{\sigma_x^2}{\sigma_x^2+\sigma_u^2} < 1, \lambda\beta < \beta$. $\hat{\beta}_{LS}$ is biased downward.

1.2

By LLN,

$$\begin{aligned}\hat{\sigma}_\epsilon^2 &= (\mathbf{y} - (\mathbf{x}+\mathbf{u})(\hat{\beta}_{LS})'(\mathbf{y} - (\mathbf{x}+\mathbf{u})(\hat{\beta}_{LS})/n \\ &= (\epsilon - (\hat{\beta}_{LS}-\beta)\mathbf{x} - \mathbf{u}\hat{\beta}_{LS})'(\epsilon - (\hat{\beta}_{LS}-\beta)\mathbf{x} - \mathbf{u}\hat{\beta}_{LS})/n \\ &= \epsilon'\epsilon/n - (\hat{\beta}_{LS}-\beta)\epsilon'\mathbf{x}/n - \epsilon'\mathbf{u}/n \\ &+ (\hat{\beta}_{LS}-\beta)\mathbf{x}'\epsilon/n + (\hat{\beta}_{LS}-\beta)^2\mathbf{x}'\mathbf{x}/n \\ &+ (\hat{\beta}_{LS}-\beta)\hat{\beta}_{LS}\mathbf{x}'\mathbf{u}/n + \hat{\beta}_{LS}\mathbf{x}'\epsilon/n + \beta)\hat{\beta}_{LS}\mathbf{u}'\mathbf{x}/n + \hat{\beta}_{LS}^2\mathbf{u}'\mathbf{u}/n \\ &\to_p \sigma_\epsilon^2 + o_p(1) + o_p(1) + o_p(1) + (1-\lambda)^2\beta^2\sigma_x^2 + o_p(1) + o_p(1) + o_p(1) + \lambda^2\beta^2\sigma_u^2 \\ &= \sigma_\epsilon^2 + (1-\lambda)^2\beta^2\sigma_x^2 + \lambda^2\beta^2\sigma_u^2\end{aligned} \tag{2}$$

which has an upward bias.

Now considering $\sigma_\epsilon^2(\tilde{\mathbf{x}}'\tilde{\mathbf{x}}/n)^{-1}$ using our previous results,

$$\sigma_\epsilon^2(\tilde{\mathbf{x}}'\tilde{\mathbf{x}}/n)^{-1} \to_p \frac{\sigma_\epsilon^2 + (1-\lambda)^2\beta^2\sigma_x^2 + \lambda^2\beta^2\sigma_u^2}{\sigma_x^2+\sigma_u^2} = \lambda\frac{\sigma_\epsilon^2}{\sigma_x^2} + \lambda(1-\lambda)\beta^2 \tag{3}$$

We cannot sign the bias with the information provide.

1.3

By Slutsky and previous results,

$$\frac{\hat{\beta}_{LS}}{\sqrt{\sigma_\epsilon^2(\tilde{\mathbf{x}}'\tilde{\mathbf{x}}/n)^{-1}}} \to_p \frac{\sqrt{\lambda}\beta}{\sqrt{\frac{\sigma_\epsilon^2}{\sigma_x^2} + (1-\lambda)\beta^2}} \tag{4}$$

$$\frac{\sqrt{\lambda}\beta}{\sqrt{\frac{\sigma_\epsilon^2}{\sigma_x^2} + (1-\lambda)\beta^2}} < \frac{\beta}{\sqrt{\frac{\sigma_\epsilon^2}{\sigma_x^2}}} \tag{5}$$

So the estimate is biased downward.

1.4

As the instrument is uncorrelated with u_i and ϵ_i,

$$\hat{\beta}_{IV} = \frac{\check{\mathbf{x}}'\mathbf{y}}{\check{\mathbf{x}}'\tilde{\mathbf{x}}} = \frac{\check{\mathbf{x}}'\mathbf{y}/n}{\check{\mathbf{x}}'\tilde{\mathbf{x}}/n}\beta + \frac{\check{\mathbf{x}}'\epsilon/n}{\check{\mathbf{x}}'\tilde{\mathbf{x}}/n} \to_p \frac{\mathbb{E}[\check{x}_i x_i]}{\mathbb{E}[\check{x}_i(x_i+u_i)]}\beta = \beta \tag{6}$$

1.5

$\mathbf{y} = \mathbf{x}\beta + \epsilon = (\tilde{\mathbf{x}} - \mathbf{u})\beta + \epsilon = \tilde{\mathbf{x}}\beta + (\epsilon - \mathbf{u}\beta)$

$$\begin{aligned}\sqrt{n}(\hat{\beta}_{IV} - \beta) &= \sqrt{n}\left(\frac{\check{\mathbf{x}}'\tilde{\mathbf{x}}}{\check{\mathbf{x}}'\tilde{\mathbf{x}}}\beta + \frac{\check{\mathbf{x}}'(\epsilon - \mathbf{u}\beta)}{\check{\mathbf{x}}'\tilde{\mathbf{x}}} - \beta\right) \\ &= \frac{\check{\mathbf{x}}'(\epsilon - \mathbf{u}\beta)/\sqrt{n}}{\check{\mathbf{x}}'\tilde{\mathbf{x}}/n} = (\mathbb{E}[\check{x}_i x_i])^{-1}\frac{\check{\mathbf{x}}'(\epsilon - \mathbf{u}\beta)}{\sqrt{n}} + o_p(1) \to_d N(0, V_{IV})\end{aligned} \tag{7}$$

where $V_{IV} = \frac{\mathbb{E}[\check{x}_i^2(\epsilon_i - u_i\beta)^2]}{(\mathbb{E}[\check{x}_i x_i])^2}$

1.6

First, estimate heteroskedasticity-consistent standard errors:

$$\begin{aligned}\hat{V}_{IV} = \left(\frac{\check{\mathbf{x}}'\tilde{\mathbf{x}}}{n}\right)^{-1}\left(\frac{1}{n}\sum_{i=1}^{n}\check{x}_i^2(y_i - \check{x}_i\hat{\beta}_{IV})^2\right)\left(\frac{\check{\mathbf{x}}'\tilde{\mathbf{x}}}{n}\right)^{-1} &= \frac{\mathbb{E}[\check{x}_i^2(\epsilon_i - u_i\beta)^2]}{(\mathbb{E}[\check{x}_i x_i])^2} + o_p(1) \\ \to_p V_{IV}\end{aligned} \tag{8}$$

Then, construct the confidence interval

$$CI_{95} = \left[\hat{\beta}_{IV} \pm 1.96 * \sqrt{\frac{\hat{V}_{IV}}{n}}\right] \tag{9}$$

1.7

From 1.1 we know $\hat{\lambda} = \frac{\check{\sigma}^{\mathbf{2}}_{\mathbf{x}}}{\tilde{\mathbf{x}}'\tilde{\mathbf{x}}} \to_p \frac{\sigma_x^2}{\sigma_x^2+\sigma_u^2} = \lambda$, so $\frac{\hat{\beta}_{IV}}{\hat{\lambda}} \to_p \beta$

1.8

Define the function $g(w) = w_1 w_2 / w_3$ where $\mathbf{w} = (w_1, w_2, w_3) \in \mathbb{R}$ such that $\dot{\mathbf{g}}(\mathbf{w}) = (w_2/w_3, w_1/w_3, -w_1 w_2/w_3^2)$. By the delta method.

$$\sqrt{n}\left(g\left(\begin{bmatrix}\hat{\beta}_{LS}\\ \tilde{\mathbf{x}}'\tilde{\mathbf{x}}/n\\ \check{\sigma}^{\mathbf{2}}_{\mathbf{x}}\end{bmatrix}\right) - g\left(\begin{bmatrix}\lambda\beta\\ \sigma^{\mathbf{2}}_{\mathbf{x}}+\sigma^{\mathbf{2}}_{\mathbf{u}}\\ \sigma^{\mathbf{2}}_{\mathbf{x}}\end{bmatrix}\right)\right) = \sqrt{n}\left(\frac{\hat{\beta}_{IV}}{\hat{\lambda}} - \beta\right) \to_d N(0, V_{VS}) \tag{10}$$

where $V_{VS} = \dot{\mathbf{g}}(\mathbf{w_0})'\Sigma\dot{\mathbf{g}}(\mathbf{w_0})$ and $\mathbf{w_0} = (\lambda\beta, \sigma_x^2+\sigma_u^2, \sigma_x^2)$

1.9

Assuming there is an estimator $\hat{\Sigma}$,

$$CI_{95} = \left[\mathbf{g}(\hat{\mathbf{w}}) \pm 1.96 * \sqrt{\frac{\hat{V}_{VS}}{n}}\right] \tag{11}$$

where $V_{VS} = \dot{\mathbf{g}}(\hat{\mathbf{w}})'\hat{\Sigma}\dot{\mathbf{g}}(\hat{\mathbf{w}})$ and $\hat{\mathbf{w}} = (\hat{\beta}_{LS}, \tilde{\mathbf{x}}'\tilde{\mathbf{x}}/n, \check{\sigma}_x^2)$

1.10

The fixed effects estimator is equivalent to the first differences estimator when t=2: $\hat{\beta}_{FE} = \frac{\mathbf{\Delta\tilde{x}\Delta y}}{\mathbf{\Delta\tilde{x}\Delta\tilde{x}}}$ where $\mathbf{\Delta\tilde{x}} = \mathbf{\tilde{x}_2} - \mathbf{\tilde{x}_1}$ and $\mathbf{\Delta y} = \mathbf{y_2} - \mathbf{y_1}$. Note, $\Delta y_i = (\Delta x_i - \Delta u_i)\beta + \Delta\epsilon_i$ where $\epsilon_i = \epsilon_{i2} - \epsilon_{i1}$

$$\hat{\beta}_{FE} = \frac{\mathbf{\Delta\tilde{x}\Delta y}}{\mathbf{\Delta\tilde{x}\Delta\tilde{x}}} \to_p \frac{\mathbb{E}[(\mathbf{\Delta x_i} + \mathbf{\Delta u_i})\mathbf{\Delta x_i}]}{\mathbb{E}[(\mathbf{\Delta x_i} + \mathbf{\Delta u_i})^{\mathbf{2}}]}\beta = \frac{\sigma^2_{\Delta x}}{\sigma^2_{\Delta x} + \sigma^2_{\Delta u}} = \gamma\beta \tag{12}$$

By construction $\gamma < 1$, therefore $\hat{\beta}_{FE}$ is biased downward.

1.11

Using the previous result

$$\begin{aligned}\gamma = \frac{\sigma^2_{\Delta x}}{\sigma^2_{\Delta x} + \sigma^2_{\Delta u}} &= \frac{\mathbb{V}[x_{i2} - x_{i1}]}{\mathbb{V}[x_{i2} - x_{i1}] + \mathbb{V}[u_{i2} - u_{i1}]} = \frac{2\sigma_x^2 - 2\,\mathbb{C}[x_{i2}, x_{i1}]}{2\sigma_x^2 - 2\,\mathbb{C}[x_{i2}, x_{i1}] + 2\sigma_u^2 - 2\,\mathbb{C}[u_{i2}, u_{i1}]}\\ &= \frac{\sigma_x^2(1-\rho_x)}{\sigma_x^2(1-\rho_x) + \sigma_u^2(1-\rho_u)} = \frac{\sigma_x^2}{\sigma_x^2 + \sigma_u^2\frac{(1-\rho_u)}{(1-\rho_x)}}\end{aligned} \tag{13}$$

1.12

If $\rho_x \approx 1$ and $\rho_u \approx 0$, then $\gamma \approx 0$ and $\gamma = \frac{\sigma_x^2}{\sigma_x^2+\sigma_u^2\frac{(1-\rho_u)}{(1-\rho_x)}} < \frac{\sigma_x^2}{\sigma_x^2+\sigma_u^2} = \lambda$. Intuitively, if regressors are highly correlated and measurement error is uncorrelated over time, then fixed-effects estimation will produced more biased estimates than simple linear regression.

2 Implementing Least-Squares Estimators

2.1

Given the Estimator: $\hat{\beta}(\mathbf{W}) = argmin_{\beta \ in \mathbb{R}^d}((\mathbf{y} - \mathbf{X}\beta)'\mathbf{W}(\mathbf{y} - \mathbf{X}\beta)')$ Take the FOC.

$$\begin{aligned} -\mathbf{X}'\mathbf{W}(\mathbf{y} - \mathbf{X}\beta) - (\mathbf{y} - \mathbf{X}\beta)'\mathbf{W}\mathbf{X} &= 0 \\ -\mathbf{X}'\mathbf{W}\mathbf{y} + \mathbf{X}'\mathbf{W}\mathbf{X}\beta + \mathbf{X}'\mathbf{W}\mathbf{X}\beta - \mathbf{X}'\mathbf{W}\mathbf{y} &= 0 \\ \mathbf{2X'WX}\beta &= 2X'Wy \\ \hat{\beta}(\mathbf{W}) &= (\mathbf{X}'\mathbf{W}\mathbf{X})^{-\mathbf{1}}\mathbf{X}'\mathbf{W}\mathbf{y} \end{aligned} \tag{14}$$

2.2

$$\begin{aligned} \sqrt{n}\left(\hat{\beta}(\mathbf{W}) - \beta\right) &= \sqrt{n}\left((\mathbf{X}'\mathbf{W}\mathbf{X})^{-\mathbf{1}}\mathbf{X}'\mathbf{W}(\mathbf{x_i'}\beta + \epsilon) - \beta\right) \\ &= \left(\beta + \frac{\mathbf{X}'\mathbf{W}\epsilon/\sqrt{\mathbf{n}}}{(\mathbf{X}'\mathbf{W}\mathbf{X})/\mathbf{n}} - \beta\right) \\ &= \frac{\mathbf{X}'\mathbf{W}\epsilon/\sqrt{\mathbf{n}}}{(\mathbf{X}'\mathbf{W}\mathbf{X})/\mathbf{n}} \\ &\to_d \mathbb{V}(0, \mathbf{V}(\mathbf{W})) \end{aligned} \tag{15}$$

where $\mathbf{V}(\mathbf{W}) = (\mathbf{X}'\mathbf{W}\mathbf{X})^{-\mathbf{1}}(\mathbf{X}'\mathbf{W}\mathbf{\Sigma}^{-\mathbf{1}}\mathbf{W}\mathbf{X})(\mathbf{X}'\mathbf{W}\mathbf{X})^{-\mathbf{1}}$

Now if $\mathbb{V}[\mathbf{y}|\mathbf{X}, \mathbf{W}] = \sigma^{\mathbf{2}}\mathbf{I_n}$, the sandwich matrix form of $\mathbf{V}(\mathbf{W})$ simplifies:
$\mathbf{V}(\mathbf{W}) = (\mathbf{X}'\mathbf{W}\mathbf{X})^{-\mathbf{1}}(\mathbf{X}'\mathbf{W}\sigma^{\mathbf{2}}\mathbf{W}\mathbf{X})(\mathbf{X}'\mathbf{W}\mathbf{X})^{-\mathbf{1}} = (\mathbf{X}'\mathbf{X})^{-\mathbf{1}}\sigma^{\mathbf{2}}$

2.3

The following estimator is a consistent variance-covariance estimator of $\mathbf{V}(\mathbf{W})$:

$$\hat{\mathbf{V}}(\mathbf{W}) = (\mathbf{X}'\mathbf{W}\mathbf{X})/n)^{-1}(\mathbf{X}'\mathbf{W}\epsilon'\epsilon\mathbf{W}\mathbf{X}/\sqrt{n})(\mathbf{X}'\mathbf{W}\mathbf{X}/n)^{-1} \tag{16}$$

2.4

Stata and R code for question 2.4 is in the code appendix

2.5

2.5.1

R code: Regression Table (matrix)

term	estimate	std.error	statistic	p.value
(Intercept)	6485.553	4513.513	1.437	0.151
treat	1535.482	638.238	2.406	0.017
black	-2592.377	794.999	-3.261	0.001
age	39.341	40.470	0.972	0.332
educ	-740.540	944.679	-0.784	0.434
educ_sq	60.082	53.768	1.117	0.264
earn74	-0.030	0.104	-0.288	0.774
black_earn74	0.175	0.132	1.330	0.184
u74	1316.032	1505.927	0.874	0.383
u75	-1167.688	1275.416	-0.916	0.360

Stata code: Regression Table (matrix)

term	estimate	std.error	statistic	p.value	CI minus	CI plus
treat	1535.4824	638.2380	2.4058	0.0166	281.0688	2789.8961
black	-2592.3766	794.9991	-3.2609	0.0012	-4154.8937	-1029.8595
age	39.3405	40.4701	0.9721	0.3315	-40.2007	118.8817
educ	-740.5400	944.6787	-0.7839	0.4335	-2597.2421	1116.1622
educ_sq	60.0823	53.7684	1.1174	0.2644	-45.5958	165.7604
earn74	-0.0299	0.1037	-0.2879	0.7735	-0.2337	0.1740
black_earn74	0.1754	0.1318	1.3304	0.1841	-0.0837	0.4344
u74	1316.0320	1505.9270	0.8739	0.3827	-1643.7657	4275.8296
u75	-1167.6884	1275.4156	-0.9155	0.3604	-3674.4316	1339.0548
(Intercept)	6485.5531	4513.5125	1.4369	0.1515	-2385.4508	15356.5570

The coefficients match up.

2.5.2

R code: Regression Table

variable	beta	se	t_test	p_value	CI_L	CI_U
const	6485.55	4513.51	1.44	0.15	-2384.89	15356.00
treat	1535.48	638.24	2.41	0.02	281.15	2789.82
black	-2592.38	795.00	-3.26	0.00	-4154.80	-1029.96
age	39.34	40.47	0.97	0.33	-40.20	118.88
educ	-740.54	944.68	-0.78	0.43	-2597.13	1116.05
educ_sq	60.08	53.77	1.12	0.26	-45.59	165.75
earn74	-0.03	0.10	-0.29	0.77	-0.23	0.17
black_earn74	0.18	0.13	1.33	0.18	-0.08	0.43
u74	1316.03	1505.93	0.87	0.38	-1643.58	4275.64
u75	-1167.69	1275.42	-0.92	0.36	-3674.27	1338.90

Stata Code: Regression Table

	(1)
	earn78
treat	1535.5*
	(2.41)
black	-2592.4**
	(-3.26)
age	39.34
	(0.97)
educ	-740.5
	(-0.78)
educ2	60.08
	(1.12)
earn74	-0.0299
	(-0.29)
blacke74	0.175
	(1.33)
u74	1316.0
	(0.87)
u75	-1167.7
	(-0.92)
cons	6485.6
	(1.44)
N	445

t statistics in parentheses

* $p < 0.05$, ** $p < 0.01$, *** $p < 0.001$

The coefficients match up.

3 Analysis of Experiments

3.1

3.1.1

$$
\begin{aligned}
\mathbb{E}[T_{DM}] &= \mathbb{E}[\bar{Y}_1] - \mathbb{E}[\bar{Y}_0] \\
&= \mathbb{E}[\frac{1}{N_1}\sum_{i=1}^{n} D_i(1)Y_i] - \mathbb{E}[\frac{1}{n-N_1}\sum_{i=1}^{n} D_i(0)Y_i] \\
&= \mathbb{E}[\frac{1}{N_1}\sum_{i=1}^{n} D_i(1)Y_i(1)] - \mathbb{E}[\frac{1}{n-N_1}\sum_{i=1}^{n} D_i(0)Y_i(0)] \\
&= \mathbb{E}[\mathbb{E}[\frac{1}{N_1}\sum_{i=1}^{n} D_i(1)Y_i(1)|T_i]] - \mathbb{E}[\mathbb{E}[\frac{1}{n-N_1}\sum_{i=1}^{n} D_i(0)Y_i(0)]|T_i]] \\
&= \mathbb{E}[\frac{1}{N_1}\sum_{i=1}^{n} \left(\frac{N_1}{n}\right) Y_i(1)] - \mathbb{E}[\frac{1}{n-N_1}\sum_{i=1}^{n} \left(\frac{N_1}{n}\right) Y_i(0)] \\
&= \mathbb{E}[\frac{1}{N_1}\sum_{i=1}^{n} \left(\frac{N_1}{n}\right) Y_i(1)] - \mathbb{E}[\frac{1}{n-N_1}\sum_{i=1}^{n} \left(\frac{n-N_1}{n}\right) Y_i(0)] \\
&= \mathbb{E}[\frac{1}{n}\sum_{i=1}^{n} Y_i(1)] - \mathbb{E}[\frac{1}{n}\sum_{i=1}^{n} Y_i(0)] \\
&= \frac{1}{n}\sum_{i=1}^{n} Y_i(1) - \frac{1}{n}\sum_{i=1}^{n} Y_i(0)
\end{aligned} \tag{17}
$$

3.1.2

R code: Asymptotically Conservative 95 CI for average treatment effect

TDM est	Conservative SE	CI Lower	CI Upper
1794.34	671.00	479.21	3109.47

Stata code: Asymptotically Conservative 95 CI for average treatment effect

TDM est	Conservative SE	CI Lower	CI Upper
1794.3431	670.9967	479.2137	3109.4725

The point estimates, standard errors, and confidence intervals match up across platforms.

3.2

3.2.1

R code: Sharp Null Hypothesis of No Treatment Effect

DM P value	KS P value
0.05	0.033

Stata Code: Sharp Null Hypothesis of No Treatment Effect

DM P value	KS P Value
.0045	.034

The p-values line up for the most part. The differences are rounding errors.

3.2.2

Stata Code: Finite Sample valid 95 CI for the treatment effect

	Observed Coef.	Bootstrap Std. Err.	z	P>|z|	Normal-based [95% Conf. Interval]	
diff	1794.343	656.4896	2.73	0.006	507.6472	3081.039

R Code: Finite Sample valid 95 CI for the treatment effect

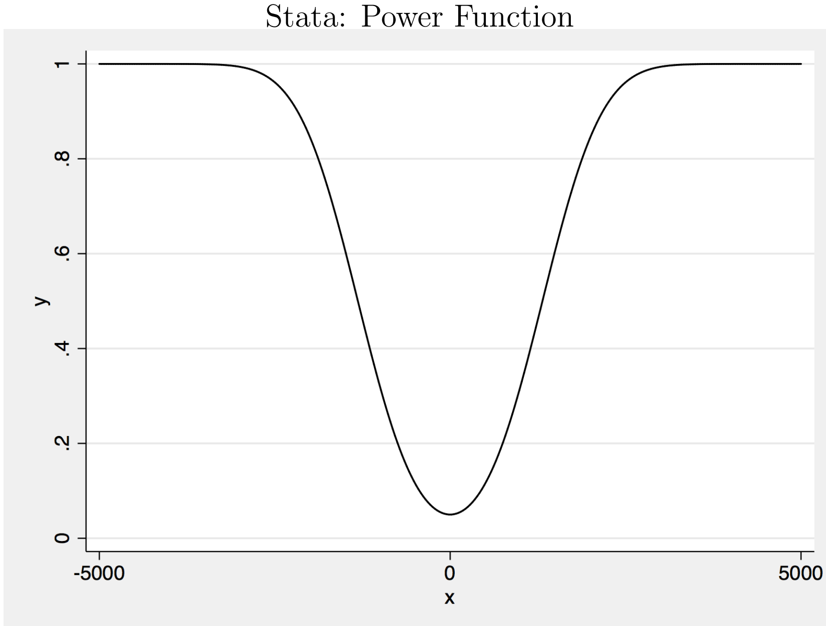
Hypothesized Treatment Effect	p_value
5000.00	0.00
4750.00	0.00
4500.00	0.00
4250.00	0.00
4000.00	0.00
3750.00	0.00
3500.00	0.01
3250.00	0.02
3000.00	0.07
2750.00	0.13
2500.00	0.28
2250.00	0.49
2000.00	0.75
1750.00	0.95
1500.00	0.64
1250.00	0.41
1000.00	0.21
750.00	0.10
500.00	0.04
250.00	0.01
0.00	0.01
-250.00	0.00
-500.00	0.00
-750.00	0.00
-1000.00	0.00
-1250.00	0.00
-1500.00	0.00

The confidence intervals match up across platforms. The Rubin's technique of bootstrapping the p-value was easier to implement the question in R. Looking at the table from R, it is clear that the confidence interval from Stata is congruent.

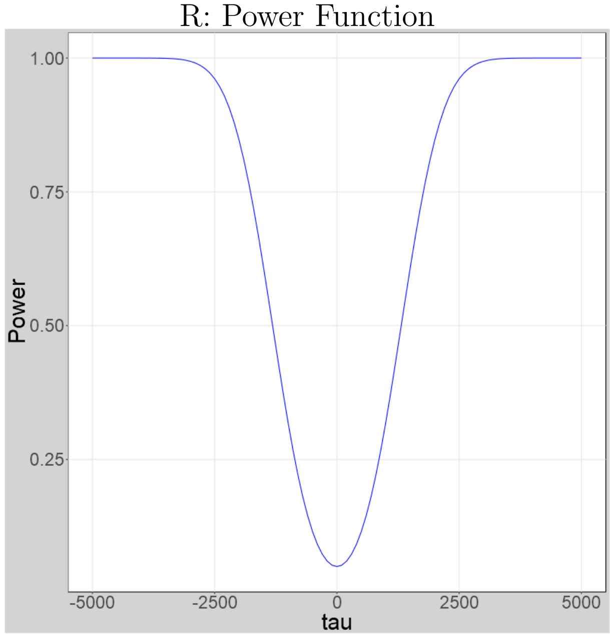
3.3

3.3.1

Stata: Power Function



R: Power Function



3.3.2

Stata computes that the sufficient sample size: 1436.824051
R computes that the sufficient sample size: 1436.824

The difference is clearly a rounding error.

4 Code Appendix

Stata

```
clear all
set more off
global dir "/Users/aaronmarkiewitz/Dropbox/Phd_Coursework/Econ 675"
cd "$dir"
set seed 675
set scheme s2mono

global dir $dir\data
global dir $dir\data
import delimited LaLonde_1986.csv


*Question 2.5

gen cons            = 1
gen educ2           = educ^2
gen blacke74    = black*earn74
local y earn78
local xlist treat black age educ educ2 earn74 blacke74 u74 u75 cons


local alpha .05
local n = _N
local d = wordcount("`xlist'")
*Procedure writen for Question 2.4
mata:
W = I(`d')
st_view(y=., ., "`y'")
st_view(X=., ., tokens("`xlist'"))

XW      = cross(X', W)
XWX     = cross(XW, X)
XWXinv  = invsym(XWX)
XWXinvc = cholinv(XWX)
b  = XWXinv*cross(X, y)
bc = XWXinvc*cross(X, y)

e  = y - X*b
v_e = diag(e:*e)
```

```
n   = rows(X)

dof = `n' - `d'
V   = XWXinv*(X'*v_e*X)*XWXinv*(n/dof)
se = sqrt(diagonal(V))

t_stat = (b):/se
p = 2*ttail(rows(X)-cols(X),abs(t_stat))
cil = b+invt(dof, (`alpha'/2))*se
ciu = b+invt(dof, (1-`alpha'/2))*se
st_matrix("b", b')
st_matrix("V", V')

end

mata: output = (b,se,t_stat,p,cil,ciu)
mmat2tex output using .\mata_out.tex, replace

reg `y' `xlist', nocons robust

esttab using .\q2_5outreg.tex, replace


** Question 3 **
clear all
import delimited LaLonde_1986.csv



*Question 3.1

sum earn78 if treat==0
local N0 = r(N)
local mu0 = r(mean)
local sd0 = r(sd)
local V0 = r(Var)/r(N)
local sig_sq0 = r(Var)

sum earn78 if treat==1
local N1 = r(N)
local mu1 = r(mean)
local sd1 = r(sd)
local V1 = r(Var)/r(N)
local sig_sq1 = r(Var)

```

```
local tau = `mu1'-`mu0'
local v = sqrt(`V1'+`V0')
local t_stat = `tau'/`v'
local p = 2*normal(-abs(`t_stat'))

local mu0 = round(`mu0', .01)
local mu1 = round(`mu1', .0001)
local sd0 = round(`sd0', .01)
local sd1 = round(`sd1', .0001)

di "`tau'"


local cil = `tau' - invnormal(0.975)*`v'
local ciu = `tau' + invnormal(0.975)*`v'

di "`CIlower'"
di "`CIupper'"



mata: output = (`tau',`v',`t_stat',`p',`cil',`ciu')
mmat2tex output using .\31_out_mata.tex, replace



*Question 3.2a

* difference in means estimator
permute treat diffmean=(r(mu_2)-r(mu_1)), reps(1999) nowarn: ttest earn78, by(treat)
matrix pval_dm = r(p)
local p_dm = pval_dm[1,1]
di "DM p-value"
di "DM p-value= `p_dm'"


* KS statistic
permute treat ks=r(D), reps(1999) nowarn: ksmirnov earn78, by(treat)
matrix pval_ks = r(p)
local p_ks = pval_ks[1,1]
di "KS p-value= `p_ks'"
```

```
*Question 3.2b

* Infer missing values under the null of constant treatment effect
gen     Y1_imputed = earn78
replace Y1_imputed = earn78 + `tau' if treat==0

gen     Y0_imputed = earn78
replace Y0_imputed = earn78 - `tau' if treat==1


* Write program to put into bootstrap function
program define meandiff, rclass
        summarize   Y1_imputed if treat==1
        local           tau1 = r(mean)
        sum             Y0_imputed if treat==0
        local           tau0 = r(mean)
        return      scalar meandiff = `tau1' - `tau0'
end

* Run bootstrap function using meandiff program
bootstrap diff = r(meandiff), reps(1999): meandiff

/*


*Question 3.3
twoway function y= 1 - normal(invnormal(0.975)-x/`v') + normal(-invnormal(0.975)-x/`v'), range(-5000 5000)
graph export stata_power.png,replace

mata: mata clear
mata:
 function myfunc(N, s0, s1, p, tau){

   return(1 - normal(invnormal(0.975)-tau/sqrt(1/N*s1*(1/p)+1/N*s0*(1/(1-p)))) +
       normal(-invnormal(0.975)-tau/sqrt(1/N*s1*(1/p)+1/N*s0*(1/(1-p)))) -0.8)

 }
 s0 =  30072466.58373794
 s1 =  61896056.06715253
    p     = 2/3
   tau   = 1000
```

```
  mm_root(x=., &myfunc(), 1000, 1500, 0, 10000, s0,s1, p ,tau)

        x

end
```

R

```r
#Set up

library(data.table)
library(Matrix)
library(lmtest)
library(sandwich)
library(broom)
library(ggplot2)
library(xtable)
rm(list = ls(pos = ".GlobalEnv"), pos = ".GlobalEnv")
options(scipen = 999)
cat("\f")

setwd("/Users/aaronmarkiewitz/Dropbox/Phd_Coursework/Econ 675")
set.seed(675)
#Q2.4
# generate data for test case

# set n_col and n_row
n_col <- 5
n_row <- 100
n_cell <- n_col*n_row

# create random matrices
y_data <- matrix(runif(n_row, 0, 100), nrow = n_row, ncol = 1)
x_data <- matrix(runif(n_cell, 0, 1), nrow = n_row, ncol = n_col)

# function
mat_reg <- function(x = NULL, y = NULL, opt_chol = FALSE, CI_level = .95){

  # get matrix size parameters
  n_col <- ncol(x)
  n_row <- nrow(x)

  # estimate beta
```

```
# check which inverse function to use
if(!opt_chol){

  #  standard
  B <- Matrix::solve(Matrix::crossprod(x, x))%*%(Matrix::crossprod(x, y))
}else{

  # cholesky
  chol_m <- chol(Matrix::crossprod(x, x))
  B<- chol2inv(chol_m)%*%(Matrix::crossprod(x, y))

}

# estimate V

# residuals
my_resid <- y - x%*%B

# grab diagonal of var matrix
M_diag <- diag(as.numeric(my_resid^2*(n_row/(n_row-n_col))),
   nrow = n_row, ncol = n_row)
M <- (t(x) %*% M_diag %*% x)

# asymptotic variance
if(!opt_chol){

  V <- solve(crossprod(x, x)) %*% M  %*% solve(crossprod(x, x))

}else{

  A_inv <-   chol2inv(chol_m) %*% M  %*% chol2inv(chol_m)
  V <- A_inv

}
sqrt(diag(V))

# other stats

# beta and variance
out_dt <- data.table(beta = as.numeric(B), V_hat = diag(V) )

# standard errors
out_dt[, se := sqrt(V_hat)]

```

```
  # t test
  out_dt[, t_test := beta/(se)]

  # p values
  out_dt[, p_value := 2*(1- pt((abs(t_test)), n_row - n_col))]

  # confidence interval
  out_dt[, CI_L := beta - (se) * qt(1-((1-CI_level)/2), n_row )]
  out_dt[, CI_U := beta + (se) * qt(1-((1-CI_level)/2), n_row )]

  # drop v_hat
  out_dt[, V_hat := NULL]


  out_list <- list()

  out_list[["results"]] <- out_dt
  out_list[["varcov"]] <- V

  return(out_list)

}


# run function to check difference between cholesky and symmetric
    inverse
reg_1 <- mat_reg(x = x_data, y = y_data, opt_chol = FALSE)
reg_2 <- mat_reg(x = x_data, y = y_data, opt_chol = TRUE)

# compare coefficients
coeff_diff <- reg_1[["results"]][, beta] - reg_2[["results"]][,
    beta]

# Check
all.equal(reg_1$varcov, reg_2$varcov)
reg_1$varcov -  reg_2$varcov


#Q 2.5

# load data #note paste is so it fits on pdf in markdown
lalonde_dt <- fread(paste0("LaLonde_1986.csv"))

# grab y matrix
y_la <- as.matrix(lalonde_dt[, earn78])
```

```r

# create other vars for regression
lalonde_dt[, educ_sq := educ^2]
lalonde_dt[, black_earn74 := black*earn74]
lalonde_dt[, const := 1]

# grab x vars
x_vars <- c("treat", "black", "age", "educ",
            "educ_sq", "earn74","black_earn74",
            "u74","u75")

# make x matrix
x_la <- as.matrix(lalonde_dt[, c("const", x_vars), with = FALSE])

# run function on this data
lalonde_reg <- mat_reg(x = x_la, y = y_la)

# grab the results
results_2_5_a <- lalonde_reg[["results"]]

# add in coef label
results_2_5_a[, variable := c("const", x_vars)]

# put variables in front
setcolorder(results_2_5_a, c("variable", setdiff( colnames(results_2_5_a), "variable")))


# using lm


# get regression formula
reg_form <- as.formula(paste("earn78~", paste(x_vars, collapse="+")))

# run regression
lalonde_lm <- lm(reg_form, lalonde_dt)

# get summary, NOTE: these are NOT robust standard errors
lalong_lm_dt <- summary(lalonde_lm)$coefficients

# get robust standard errors. I use HC2 to match my math above
# any differnces are floating point errors
lm_robust <- coeftest(lalonde_lm, vcov = vcovHC(lalonde_lm, type="HC1"))
```

```r

results_2_5_b <- data.table(tidy(lm_robust))


# 3.1.a
TDM <- lalonde_dt[treat == 1, mean(earn78)] - lalonde_dt[treat ==
    0, mean(earn78)]

# get variance
s1_sq <- lalonde_dt[treat == 1, var(earn78)]
s0_sq <- lalonde_dt[treat == 0, var(earn78)]

# get V_tdm
V_tdm <- s1_sq/lalonde_dt[treat == 1, .N] + s0_sq/lalonde_dt[treat
    == 0, .N]

#  standard error
se_tdm <- sqrt(V_tdm)

# convidence interval
tdm_CI_L <- TDM - se_tdm * qnorm(.975)
tdm_CI_U <- TDM + se_tdm * qnorm(.975)

# put together resuts
results_3_1_b <- data.table("TDM est" = TDM,
                            "Conservative SE" = se_tdm,
                            "CI Lower" = tdm_CI_L,
                            "CI Upper" =  tdm_CI_U)

# fisher

# write function
fisher_p <- function(in_data       = NULL,
                     y_var         = NULL,
                     treat_var     = NULL,
                     null_hyp      = 0,
                     opt_test_stat = "DM",
                     n_iter        = 1999){

  # check that a test has ben speciies
  if(!opt_test_stat %chin% c("DM", "KS")){
    stop("Specify either DM ot KS test")
  }

  # check for non-zero null under the KS test (function doesn't do
```

```
    that)
  if(opt_test_stat == "KS" & null_hyp != 0){
    stop("The KS test is not compatibe with a non-zero null at the
       moment")
  }

  # copy data so I can create y(0) and y(1) cols without altering
     input data set
  data_c <- copy(in_data)

  # create colums for sharp null treated and untreated y variables
  data_c[get(treat_var) == 1, y_1 := get(y_var) ]
  data_c[get(treat_var) == 0, y_1 := get(y_var) + null_hyp ]
  data_c[get(treat_var) == 0, y_0 := get(y_var) ]
  data_c[get(treat_var) == 1, y_0 := get(y_var) - null_hyp ]

  # create a data.table for the results of bootstrap
  sim_data <- data.table(iteration = c(1:(n_iter+1)))

  # get the number of treated vars
  n_treat <- nrow(data_c[get(treat_var) == 1, ])
  n_row <- nrow(data_c)

  # do actual test
  if(opt_test_stat == "DM"){

    # get mean of treatment
    m_t <- data_c[get(treat_var) == 1, mean(get(y_var))]

    # get mean of untreated
    m_unt <- data_c[get(treat_var) == 0, mean(get(y_var))]

    test_1 <-m_t - m_unt - null_hyp

  }
  if(opt_test_stat == "KS"){
    ksout <- suppressWarnings(ks.test(data_c[get(treat_var) == 1,
       get(y_var)],
                                      data_c[get(treat_var) == 0,
                                         get(y_var)] ))
    test_1 <- ksout$statistic
  }

  # put results of actual data in table
  sim_data[iteration == 1, test := test_1]
```

```
  # for each iteration
  for(i in 2:(n_iter + 1)){

    # create a permutation
    sample_i_1 <- sample.int(n = n_row, size = n_treat)
    sample_i_0 <- setdiff(c(1: n_row), sample_i_1)

    # calculate the averate treatment effect for this given sample
    if(opt_test_stat == "DM"){

      test_i <- data_c[sample_i_1, mean(y_1)] - data_c[sample_i_0, mean(y_0)] - null_hyp
    }
    if(opt_test_stat == "KS"){
      ksout <- suppressWarnings(ks.test(data_c[sample_i_1, y_1], data_c[sample_i_0, y_0] ))
      test_i <- ksout$statistic
    }


    # store this value in the data table
    sim_data[ i, test := test_i]
  }

  # get absolute value and rank of the tests
  sim_data[, abs_test := abs(test)]
  sim_data[, test_rank := frank(abs_test)]

  # get p value
  p_value <- (nrow(sim_data) - sim_data[iteration == 1, test_rank] + 1)/nrow(sim_data)

  return(p_value)


}

# run function on data
results_3_2_a_DM <- fisher_p(in_data      = lalonde_dt,
                             y_var        = "earn78",
                             treat_var    = "treat",
                             null_hyp     = 0,
                             opt_test_stat = "DM",
                             n_iter       = 1999)
```

```r

results_3_2_a_KS <- fisher_p(in_data        = lalonde_dt,
                             y_var          = "earn78",
                             treat_var      = "treat",
                             null_hyp       = 0,
                             opt_test_stat  = "KS",
                             n_iter         = 1999)

# make it fancy for output
results_3_2_a_DM <- data.table("DM P value" =   results_3_2_a_DM )
results_3_2_a_KS <- data.table("KS P value" =   results_3_2_a_KS )


# construct 95% confidence interval


# run functions
grid <- seq(5000,-1500,-250)

dm_p_list <- lapply(grid,
                    fisher_p,
                    in_data= lalonde_dt,
                    y_var = "earn78",
                    treat_var = "treat",
                    opt_test_stat= "DM",
                    n_iter = 999)

results_3_2_b <- data.table(hyp_treat = grid, p_value = dm_p_list)
setnames(results_3_2_b, "hyp_treat", "Hypothesized Treatment
   Effect")


# Power calculations

# plot attributes from EA
plot_attributes <- theme(plot.background = element_rect(fill = "
   lightgrey"),
                         panel.grid.major.x = element_line(color =
                            "gray90"),
                         panel.grid.minor  = element_blank(),
                         panel.background = element_rect(fill = "
                           white",
                                                          colour =
                                                            "black
                                                            ") ,
```

```
                              panel.grid.major.y = element_line(color = "gray90"),
                              text = element_text(size= 30),
                              plot.title = element_text(vjust=0,
                                                        colour = "#0B6357",
                                                        face = "bold",
                                                        size = 30))


# write power function
power_function <- function(x, se= NULL) {
  1 - pnorm(qnorm(0.975)-x/se) + pnorm(-qnorm(0.975)-x/se)
}

# plot function
power_plot <- ggplot(data = data.frame(x = 0), mapping = aes(x = x))
power_plot <- power_plot + stat_function(fun = power_function,
                                         args = list(se= results_3_1_b$`Conservative SE`),
                                         color = "blue")
power_plot <- power_plot + xlim(-5000,5000) + xlab("tau") + ylab("Power")  + plot_attributes
power_plot


# find  n

# Parameterize the equation
p     = 2/3
tau   = 1000

# Write down the power function, which implicitly defines N
Fun <- function(N, s.0 = s0_sq, s.1 = s1_sq){
  -0.8 + 1 - pnorm(qnorm(0.975)-tau/sqrt(1/N*s.1*(1/p)+1/N*s.0*(1/(1-p)))) +
    pnorm(-qnorm(0.975)-tau/sqrt(1/N*s.1*(1/p)+1/N*s.0*(1/(1-p))))
}

# Solve for N
N.sol <- uniroot(Fun,c(0,100000000))$root
```

```r
# save stuff

# save plot
png("power_func_r.png", height = 800, width = 800, type = "cairo")
print(power_plot)
dev.off()

# save results
res_objects <- ls()[grepl("results", ls())]

save_tex_tables <- function(obj_name = NULL){

  table <- get(obj_name)

  print(xtable(table, type = "latex"),
        file = paste0(obj_name, ".tex"),
        include.rownames = FALSE,
        floating = FALSE)

}

lapply(res_objects, save_tex_tables)
```