ECON675 – Assignment 1

Anirudh Yadav

September 28, 2018

Contents

1 Simple linear regression with measurement error

1.1 OLS estimator

We have

$$\begin{aligned}
\hat{\beta}_{LS} &= (\tilde{\boldsymbol{x}}'\tilde{\boldsymbol{x}})^{-1}(\tilde{\boldsymbol{x}}'\boldsymbol{y}) \\
&= (\sum_{i=1}^n \tilde{x}_i\tilde{x}_i')^{-1}(\sum_{i=1}^n \tilde{x}_i y_i) \\
&= (\sum_{i=1}^n \tilde{x}_i^2)^{-1}(\sum_{i=1}^n \tilde{x}_i y_i) \text{ since } \tilde{x}_i \text{ is a scalar} \\
&= (\frac{1}{n}\sum_{i=1}^n \tilde{x}_i^2)^{-1}(\frac{1}{n}\sum_{i=1}^n \tilde{x}_i y_i).
\end{aligned}$$

Now, since $\tilde{x}_i$ and y_i are iid, we can use WLLN:

$$\begin{aligned}
\frac{1}{n}\sum_{i=1}^n \tilde{x}_i^2 &\to_p \mathbb{E}[\tilde{x}_i^2] \\
\implies (\frac{1}{n}\sum_{i=1}^n \tilde{x}_i^2)^{-1} &\to_p (\mathbb{E}[\tilde{x}_i^2])^{-1} \text{ by CMT.}
\end{aligned}$$

And

$$\frac{1}{n}\sum_{i=1}^n \tilde{x}_i y_i \to_p \mathbb{E}[\tilde{x}_i y_i]$$

Then, by CMT

$$\hat{\beta}_{LS} \to_p (\mathbb{E}[\tilde{x}_i^2])^{-1}\mathbb{E}[\tilde{x}_i y_i] \tag{1}$$

Now,

$$\begin{aligned}
\mathbb{E}[\tilde{x}_i^2] &= \mathbb{E}[(x_i + u_i)^2] \\
&= \mathbb{E}[x_i^2 + 2x_iu_i + u_i^2] \\
&= \sigma_x^2 + \sigma_u^2 \qquad (2)
\end{aligned}$$

And similarly,

$$\begin{aligned}
\mathbb{E}[\tilde{x}_i y_i] &= \mathbb{E}[(x_i + u_i)(x_i\beta + \epsilon_i)] \\
&= \mathbb{E}[x_i^2\beta + x_i\epsilon_i + x_iu_i\beta + u_i\epsilon_i] \\
&= \sigma_x^2\beta \qquad (3)
\end{aligned}$$

Combining (1), (2) and (3) gives

$$\hat{\beta}_{LS} \to_p \frac{\sigma_x^2}{\sigma_x^2 + \sigma_u^2}\beta.$$

Thus, $\lambda = \frac{\sigma_x^2}{\sigma_x^2+\sigma_u^2}$ and $\hat{\beta}_{LS}$ is biased downwards.

1.2 Standard errors

We have

$$
\begin{aligned}
\hat{\sigma}_\epsilon^2 &= \frac{1}{n}\sum_{i=1}^n \hat{\epsilon}_i^2 \\
&= \frac{1}{n}\sum_{i=1}^n (y_i - \tilde{x}_i\hat{\beta})^2 \\
&= \frac{1}{n}\sum_{i=1}^n (y_i^2 - 2\tilde{x}_i y_i\hat{\beta} + \tilde{x}_i^2\hat{\beta}^2) \\
&= \frac{1}{n}\sum_{i=1}^n \big((x_i\beta + \epsilon_i)^2 - 2(x_i + u_i)(x_i\beta + \epsilon_i)\hat{\beta} + (x_i + u_i)^2\hat{\beta}^2\big) \\
&= \frac{1}{n}\sum_{i=1}^n \big(x_i^2\beta^2 + 2x_i\epsilon_i\beta + \epsilon_i^2 - 2(x_i^2\beta + x_i\epsilon_i + u_i x_i\beta + u_i\epsilon_i)\hat{\beta} + (x_i^2 + 2x_i u_i + u_i^2)\hat{\beta}^2\big).
\end{aligned}
$$

Next, letting things converge (via WLLN) and using the CMT where required (e.g. $\hat{\beta}^2 \to_p \lambda^2\beta^2$) we get

$$
\begin{aligned}
\hat{\sigma}_\epsilon^2 &\to_p \sigma_x^2\beta^2 + \sigma_\epsilon^2 - 2\sigma_x^2\lambda\beta^2 + \sigma_x^2\lambda^2\beta^2 + \sigma_u^2\lambda^2\beta^2 \\
&= \sigma_\epsilon^2 + \sigma_x^2\beta^2(1 - 2\lambda + \lambda^2) + \sigma_u^2\lambda^2\beta^2 \\
&= \sigma_\epsilon^2 + (1-\lambda)^2\sigma_x^2\beta^2 + \sigma_u^2\lambda^2\beta^2, \qquad (4)
\end{aligned}
$$

as required. Thus, $\hat{\sigma}_\epsilon^2$ is biased upwards. Next, note that

$$
\begin{aligned}
(\tilde{\boldsymbol{x}}'\tilde{\boldsymbol{x}}/n)^{-1} &= \frac{1}{n}(\sum_{i=1}^n \tilde{x}_i^2)^{-1} \\
&\to_p (\sigma_x^2 + \sigma_u^2)
\end{aligned}
$$

using (2). Thus, by CMT:

$$
\hat{\sigma}_\epsilon^2(\tilde{\boldsymbol{x}}'\tilde{\boldsymbol{x}}/n)^{-1} \to_p \frac{\sigma_\epsilon^2 + (1-\lambda)^2\sigma_x^2\beta^2 + \sigma_u^2\lambda^2\beta^2}{\sigma_x^2 + \sigma_u^2}
$$

Noting that $\lambda = \frac{\sigma_x^2}{\sigma_x^2+\sigma_u^2}$, we get

$$
\hat{\sigma}_\epsilon^2(\tilde{\boldsymbol{x}}'\tilde{\boldsymbol{x}}/n)^{-1} \to_p \lambda\frac{\sigma_\epsilon^2}{\sigma_x^2} + (1-\lambda)^2\lambda\beta^2 + \lambda^2\beta^2\frac{\sigma_u^2}{\sigma_x^2 + \sigma_u^2}
$$

Note that $\frac{\sigma_u^2}{\sigma_x^2+\sigma_u^2} = 1 - \lambda$, so

$$
\begin{aligned}
\hat{\sigma}_\epsilon^2(\tilde{\boldsymbol{x}}'\tilde{\boldsymbol{x}}/n)^{-1} &\to_p \lambda\frac{\sigma_\epsilon^2}{\sigma_x^2} + (1-\lambda)^2\lambda\beta^2 + \lambda^2(1-\lambda)\beta^2 \\
&= \lambda\frac{\sigma_\epsilon^2}{\sigma_x^2} + \lambda(1-\lambda)\beta^2, \qquad (5)
\end{aligned}
$$

thus we cannot sign the bias.

1.3 t-test

Applying the CMT to (5):

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

Then, since $\hat{\beta} \to_p \lambda\beta$ we can use the CMT to get the desired result:

$$\frac{\hat{\beta}}{\sqrt{\hat{\sigma}_\epsilon^2(\tilde{\boldsymbol{x}}'\tilde{\boldsymbol{x}}/n)^{-1}}} \to_p \frac{\sqrt{\lambda}\beta}{\sqrt{\frac{\sigma_\epsilon^2}{\sigma_x^2} + (1-\lambda)\beta^2}}.$$

Now, suppose that there was no measurement error so that we observed x_i directly. Then, given homoskedasticity, we would have the usual asymptotic variance

$$\begin{aligned}
\mathbb{V}[\hat{\beta}] &\to_p \sigma_\epsilon^2\mathbb{E}[x_i x_i']^{-1} \\
&= \sigma_\epsilon^2\mathbb{E}[x_i^2]^{-1} \\
&= \sigma_\epsilon^2/\sigma_x^2 \\
\implies \sqrt{\mathbb{V}[\hat{\beta}]} &\to_p \sqrt{\sigma_\epsilon^2/\sigma_x^2}
\end{aligned}$$

Now, with no measurement error, the t-statistic for the test $H_0 : \beta = \beta_0$ is

$$t = \frac{\hat{\beta} - \beta_0}{s.e.(\hat{\beta})} \to_p \frac{\beta - \beta_0}{plim\sqrt{\mathbb{V}[\hat{\beta}]}} = \frac{\beta - \beta_0}{\sqrt{\sigma_\epsilon^2/\sigma_x^2}}$$

Thus, since

$$\frac{\hat{\beta}}{\sqrt{\hat{\sigma}_\epsilon^2(\tilde{\boldsymbol{x}}'\tilde{\boldsymbol{x}}/n)^{-1}}} \to_p \frac{\sqrt{\lambda}\beta}{\sqrt{\frac{\sigma_\epsilon^2}{\sigma_x^2} + (1-\lambda)\beta^2}} < \frac{\beta}{\sqrt{\sigma_\epsilon^2/\sigma_x^2}},$$

with measurement error, the t-statistic is biased downwards.

1.4 Second measurement, consistency

Since $\check{x}$ satisfies the conditions for a valid instrument we can use the usual IV estimator (noting that we're only dealing with scalars):

$$\hat{\beta}_{IV} = (\sum_{i=1}^{n} \check{x}_i x_i)^{-1}(\sum_{i=1}^{n} \check{x}_i y_i)$$

It is easy to show that $\hat{\beta}_{IV}$ is consistent. In matrix notation

$$\begin{aligned}
\hat{\beta}_{IV} &= (\check{\boldsymbol{x}}'\boldsymbol{x})^{-1}(\check{\boldsymbol{x}}'\boldsymbol{y}) \\
&= (\check{\boldsymbol{x}}'\boldsymbol{x})^{-1}(\check{\boldsymbol{x}}'(\boldsymbol{x}\beta + \boldsymbol{\epsilon})) \\
&= \beta + (\check{\boldsymbol{x}}'\boldsymbol{x})^{-1}(\check{\boldsymbol{x}}'\boldsymbol{\epsilon}) \\
&= \beta + (\check{\boldsymbol{x}}'\boldsymbol{x}/n)^{-1}(\check{\boldsymbol{x}}'\boldsymbol{\epsilon}/n) \\
&\to_p \beta + \mathbb{E}[\check{x}_i x_i]\mathbb{E}[\check{x}_i \epsilon_i] \\
&= \beta,
\end{aligned}$$

since $\mathbb{E}[\check{x}_i \epsilon_i] = 0$.

1.5 Second measurement, inference

We want to characterize the asymptotic variance of $\hat{\beta}_{IV}$. From above, we have

$$\begin{aligned}
\hat{\beta}_{IV} &= \beta + (\check{\boldsymbol{x}}'\boldsymbol{x})^{-1}(\check{\boldsymbol{x}}'\boldsymbol{\epsilon}) \\
\implies \hat{\beta}_{IV} - \beta &= (\check{\boldsymbol{x}}'\boldsymbol{x}/n)^{-1}(\check{\boldsymbol{x}}'\boldsymbol{\epsilon}/n) \\
&= (\frac{1}{n}\sum_{i=1}^{n} \check{x}_i x_i)^{-1}(\frac{1}{n}\sum_{i=1}^{n} \check{x}_i e_i) \\
\implies \sqrt{n}(\hat{\beta}_{IV} - \beta) &= (\frac{1}{n}\sum_{i=1}^{n} \check{x}_i x_i)^{-1}(\frac{1}{\sqrt{n}}\sum_{i=1}^{n} \check{x}_i e_i)
\end{aligned}$$

Now, $\check{x}_i e_i$ is a mean-zero iid r.v., so by the CLT:

$$\frac{1}{\sqrt{n}}\sum_{i=1}^{n} \check{x}_i e_i \to_d \mathcal{N}(0, \mathbb{V}[\check{x}_i e_i]) = \mathcal{N}(0, \mathbb{E}[\check{x}_i^2 e_i^2])$$

Next, note that by WLLN

$$\frac{1}{n}\sum_{i=1}^{n} \check{x}_i x_i \to \mathbb{E}[\check{x}_i x_i]$$

Thus, by CMT and Slutsky we have

$$\sqrt{n}(\hat{\beta}_{IV} - \beta) \to_d \mathcal{N}(0, V_{IV})$$

with

$$\begin{aligned}
V_{IV} &= \mathbb{E}[\check{x}_i x_i]^{-1}\mathbb{E}[\check{x}_i^2 e_i^2]\mathbb{E}[\check{x}_i x_i]^{-1} \\
&= \frac{\mathbb{E}[\check{x}_i^2 e_i^2]}{\mathbb{E}[\check{x}_i x_i]^2}.
\end{aligned}$$

1.6 Second measurement, inference

To construct an asymptotically valid CI we just need a consistent estimator for V_{IV}. Since $\hat{\beta}_{IV}$ is consistent, the standard plug-in variance estimator will be consistent too:

$$\begin{aligned}\hat{V}_{IV} &= \frac{\hat{\mathbb{E}}[\check{x}_i^2 e_i^2]}{\hat{\mathbb{E}}[\check{x}_i x_i]^2} \\ &= \frac{\frac{1}{n}\sum_{i=1}^n \check{x}_i^2 (y_i - \check{x}_i \hat{\beta}_{IV})^2}{[\frac{1}{n}\sum_{i=1}^n \check{x}_i x_i]^2}\end{aligned}$$

Then, the 95% confidence interval is given by

$$CI_{95} = \left[\hat{\beta}_{IV} - 1.96\sqrt{\hat{V}_{IV}/n}, \hat{\beta}_{IV} + 1.96\sqrt{\hat{V}_{IV}/n}\right]$$

1.7 Validation sample, consistency

First note that the "true" parameter β is given by

$$\begin{aligned}\beta &= (\mathbb{E}[x_i x_i'])^{-1}\mathbb{E}[x_i y_i] \\ &= (\mathbb{E}[x_i^2])^{-1}\mathbb{E}[x_i y_i] \\ &= \mathbb{E}[x_i y_i]/\sigma_x^2.\end{aligned}$$

Now, we are given a consistent estimator of σ_x^2, so we just need a consistent estimator of $\mathbb{E}[x_i y_i]$ and we're done. Consider the observed sample average $\frac{1}{n}\sum_i \tilde{x}_i y_i$. Now,

$$\begin{aligned}\frac{1}{n}\sum_{i=1}^n \tilde{x}_i y_i &= \frac{1}{n}\sum_{i=1}^n (x_i + u_i) y_i \\ &= \frac{1}{n}\sum_{i=1}^n (x_i y_i + u_i x_i \beta + u_i \epsilon_i) \\ &\to_p \mathbb{E}[x_i y_i],\end{aligned}$$

by WLLN. Thus, we have $\check{\sigma}_x^2 \to_p \sigma_x^2$ and $\frac{1}{n}\sum_i \tilde{x}_i y_i \to_p \mathbb{E}[x_i y_i]$. Then by the CMT we have

$$\hat{\beta}_{VS} = \frac{\frac{1}{n}\sum_i \tilde{x}_i y_i}{\check{\sigma}_x^2} \to_p \frac{\mathbb{E}[x_i y_i]}{\sigma_x^2} = \beta.$$

Note that we can write $\hat{\beta}_{VS}$ in matrix form:

$$\hat{\beta}_{VS} = \frac{(\tilde{\boldsymbol{x}}'\boldsymbol{y})/n}{\check{\sigma}_x^2}.$$

1.8 Validation sample, distribution

First notice that

$$\begin{aligned}
\hat{\beta}_{LS} &= (\tilde{\boldsymbol{x}}'\tilde{\boldsymbol{x}}/n)^{-1}(\tilde{\boldsymbol{x}}'\boldsymbol{y}/n) \\
\implies (\tilde{\boldsymbol{x}}'\tilde{\boldsymbol{x}}/n)\hat{\beta}_{LS} &= (\tilde{\boldsymbol{x}}'\boldsymbol{y}/n) \\
\therefore \frac{(\tilde{\boldsymbol{x}}'\tilde{\boldsymbol{x}}/n)}{\breve{\sigma}_x^2}\hat{\beta}_{LS} &= \hat{\beta}_{VS}
\end{aligned}$$

Now we can use the Delta method. First let $\boldsymbol{v} = (v_1, v_2, v_3) \in \mathbb{R}^3$. And define the vector-valued function $f(\boldsymbol{v}) = v_1 v_2 / v_3$. Then we have

$$\begin{aligned}
\sqrt{n}\left(f\begin{bmatrix}\hat{\beta}_{LS}\\ \tilde{\boldsymbol{x}}'\tilde{\boldsymbol{x}}/n \\ \breve{\sigma}_x^2\end{bmatrix} - f\begin{bmatrix}\lambda\beta \\ \sigma_x^2+\sigma_u^2 \\ \sigma_x^2\end{bmatrix}\right) &= \sqrt{n}\left(\hat{\beta}_{VS} - \beta\right) \\
&\to_d f'(\boldsymbol{v}_0)\mathcal{N}(0, \boldsymbol{\Sigma}),
\end{aligned}$$

by the Delta method. Note that $\boldsymbol{v}_0 = (\lambda\beta, \sigma_x^2 + \sigma_u^2, \sigma_x^2)$ and that

$$f'(\boldsymbol{v}) = \begin{bmatrix} v_2/v_3 \\ v_1/v_3 \\ -v_1v_2/v_3^2\end{bmatrix} \implies f'(\boldsymbol{v}_0) = \begin{bmatrix}1/\lambda \\ \lambda\beta/\sigma_x^2 \\ -\beta/\sigma_x^2\end{bmatrix}$$

since $\lambda = \sigma_x^2/(\sigma_x^2 + \sigma_u^2)$. Thus, we have the final result

$$\sqrt{n}\left(\hat{\beta}_{VS} - \beta\right) \to_d \mathcal{N}(0, V_{VS})$$

where

$$V_{VS} = [f'(\boldsymbol{v}_0)]'\boldsymbol{\Sigma} f'(\boldsymbol{v}_0).$$

1.9 Validation sample, inference

To construct a confidence interval for β we need a consistent estimator for V_{VS}. We are given a consistent estimator for $\boldsymbol{\Sigma}$ so we just need a consistent estimator for $f'(\boldsymbol{v}_0)$.

First note that

$$\hat{\boldsymbol{v}}_0 \equiv \begin{bmatrix}\hat{\beta}_{LS}\\ \tilde{\boldsymbol{x}}'\tilde{\boldsymbol{x}}/n \\ \breve{\sigma}_x^2\end{bmatrix} \to_p \begin{bmatrix}\lambda\beta \\ \sigma_x^2+\sigma_u^2 \\ \sigma_x^2\end{bmatrix} = \boldsymbol{v}_0$$

And of course $f'(\boldsymbol{v}_0)$ is just a continuous transformation of $\boldsymbol{v}_0$. Thus,

$$f'(\hat{\boldsymbol{v}}_0) \to_p f'(\boldsymbol{v}_0)$$

by the CMT. Then we have our consistent variance estimator

$$\hat{V}_{VS} \equiv [f'(\hat{\boldsymbol{v}}_0)]'\hat{\boldsymbol{\Sigma}} f'(\hat{\boldsymbol{v}}_0) \to_p V_{VS}.$$

Accordingly, the 95% CI for β is given by

$$CI_{95} = \left[\hat{\beta}_{VS} - 1.96\sqrt{\hat{V}_{VS}/n}, \hat{\beta}_{VS} + 1.96\sqrt{\hat{V}_{VS}/n}\right].$$

1.10 FE estimator, consistency

First note that since $T = 2$, the FE estimator is equivalent to the first-difference (FD) estimator. That is

$$\begin{aligned}\hat{\beta}_{FE} &= \hat{\beta}_{FD} \\ &= \left(\frac{1}{n}\sum_{i=1}^{n}(\tilde{x}_{i2}-\tilde{x}_{i1})^2\right)^{-1}\left(\frac{1}{n}\sum_{i=1}^{n}(\tilde{x}_{i2}-\tilde{x}_{i1})(y_{i2}-y_{i1})\right).\end{aligned}$$

Now, we can proceed as in earlier derivations by using WLLN:

$$\begin{aligned}\frac{1}{n}\sum_{i=1}^{n}(\tilde{x}_{i2}-\tilde{x}_{i1})^2 \to_p \mathbb{E}[(\tilde{x}_{i2}-\tilde{x}_{i1})^2]& \\ &= \mathbb{E}[(x_{i2}-x_{i1}+u_{i2}-u_{i1})^2] \\ &= \mathbb{E}[(x_{i2}-x_{i1})^2]+\mathbb{E}[(u_{i2}-u_{i1})^2]+2\mathbb{E}[(x_{i2}-x_{i1})(u_{i2}-u_{i1})] \\ &= \sigma^2_{\Delta x}+\sigma^2_{\Delta u}\end{aligned}$$

since $\mathbb{E}[(x_{i2}-x_{i1})] = \mathbb{E}[(u_{i2}-u_{i1})] = 0$ and $\mathbb{E}[x_{it}u_{is}] = 0$ for all $t, s = 1, 2$. And

$$\begin{aligned}\frac{1}{n}\sum_{i=1}^{n}(\tilde{x}_{i2}-\tilde{x}_{i1})(y_{i2}-y_{i1}) \to_p &\mathbb{E}[(\tilde{x}_{i2}-\tilde{x}_{i1})(y_{i2}-y_{i1})] \\ &=\mathbb{E}[(x_{i2}-x_{i1}+u_{i2}-u_{i1})(x_{i2}\beta-x_{i1}\beta+e_{i2}-e_{i1})] \\ &=\mathbb{E}[(x_{i2}-x_{i1})^2]\beta+\mathbb{E}[(x_{i2}-x_{i1})(e_{i2}-e_{i1})]... \\ &\quad+\mathbb{E}[(x_{i2}-x_{i1})(u_{i2}-u_{i1})]\beta+\mathbb{E}[(u_{i2}-u_{i1})(e_{i2}-e_{i1})] \\ &=\mathbb{E}[(x_{i2}-x_{i1})^2]\beta \\ &=\sigma^2_{\Delta x}\beta,\end{aligned}$$

since $\mathbb{E}[x_{it}u_{is}] = \mathbb{E}[e_{it}u_{is}]$ for all $t, s = 1, 2$. Thus, by CMT

$$\hat{\beta}_{FE} \to_p \frac{\sigma^2_{\Delta x}}{\sigma^2_{\Delta x}+\sigma^2_{\Delta u}}\beta$$

as required. Clearly, $\hat{\beta}_{FE}$ is biased downwards.

1.11 FE estimator, time dependence

Note that covariance-stationarity implies $\sigma^2_{x_t} = \sigma^2_x$ and $\sigma^2_{u_t} = \sigma^2_u$ for $t = 1, 2$. Next, note that

$$\begin{aligned}\sigma^2_{\Delta x} = \mathbb{V}[x_{i2}-x_{i1}] &= \mathbb{V}[x_{i2}]+\mathbb{V}[x_{i1}]-2\mathbb{C}\mathrm{ov}[x_{i2},x_{i1}] \\ &= 2\sigma^2_x-2\mathbb{C}\mathrm{ov}[x_{i2},x_{i1}].\end{aligned}$$

Thus, we get

$$\begin{aligned}
\gamma &= \frac{2(\sigma_x^2 - \mathbb{C}\text{ov}[x_{i2}, x_{i1}])}{2(\sigma_x^2 - \mathbb{C}\text{ov}[x_{i2}, x_{i1}] + \sigma_u^2 - \mathbb{C}\text{ov}[u_{i2}, u_{i1}])} \\
&= \frac{\sigma_x^2 - \rho_x \sigma_x^2}{\sigma_x^2 - \rho_x \sigma_x^2 + \sigma_u^2 - \rho_u \sigma_u^2} \\
&= \frac{\sigma_x^2 (1 - \rho_x)}{\sigma_x^2 (1 - \rho_x) + \sigma_u^2 (1 - \rho_u)} \\
&= \frac{\sigma_x^2}{\sigma_x^2 + \sigma_u^2 \frac{1-\rho_u}{1-\rho_x}}
\end{aligned}$$

as required.

1.12 FE estimator, implications

If $\rho_x \approx 1$ and $\rho_u \approx 0$ then

$$\gamma \approx \frac{\sigma_x^2}{\sigma_x^2 + \sigma_u^2 \cdot \infty} = 0.$$

Thus $\hat{\beta}_{FE} \to_p 0$. Thus, in this case the FE estimator is very bad. The intuition is as follows. When the true regressors are almost perfectly correlated over time, and measurement error is uncorrelated over time, then the only variation of the observed regressors over time is due to measurement error. But by assumption, measurement error is uncorrelated with y, so that $\hat{\beta}_{FD}$ equals zero in large samples.

2 Implementing least-squares estimators

2.1

We have

$$\hat{\boldsymbol{\beta}}(\boldsymbol{W}) = \arg\min_{\beta \in \mathbb{R}^d} (\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta})'\boldsymbol{W}(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta})$$

Given that $\boldsymbol{W}$ is symmetric, the FOC is

$$-2\boldsymbol{X}'\boldsymbol{W}(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}) = 0$$

Solving for β gives the desired result:

$$\hat{\boldsymbol{\beta}}(\boldsymbol{W}) = (\boldsymbol{X}'\boldsymbol{W}\boldsymbol{X})^{-1}(\boldsymbol{X}'\boldsymbol{W}\boldsymbol{y}).$$

2.2

Proceeding from above

$$\hat{\boldsymbol{\beta}} = (\boldsymbol{X}'\boldsymbol{W}\boldsymbol{X})^{-1}(\boldsymbol{X}'\boldsymbol{W}(\boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{e})).$$

With the usual algebra we get

$$\sqrt{n}(\hat{\boldsymbol{\beta}} - \boldsymbol{\beta}) = (\frac{1}{n}\boldsymbol{X}'\boldsymbol{W}\boldsymbol{X})^{-1}(\frac{1}{\sqrt{n}}\boldsymbol{X}'\boldsymbol{W}\boldsymbol{e})$$

Using WLLN

$$\frac{1}{n}\boldsymbol{X}'\boldsymbol{W}\boldsymbol{X} \to_p \boldsymbol{A}$$

And using CLT

$$\frac{1}{\sqrt{n}}\boldsymbol{X}'\boldsymbol{W}\boldsymbol{e} \to_d \mathcal{N}(0, \mathbb{V}[\frac{1}{\sqrt{n}}\boldsymbol{X}'\boldsymbol{W}\boldsymbol{e}])$$

Denote $\mathbb{V}[\frac{1}{\sqrt{n}}\boldsymbol{X}'\boldsymbol{W}\boldsymbol{e}] = \boldsymbol{\Sigma}$. Then combining these results via CMT, and Slutsky gives the usual sandwich-form result

$$\sqrt{n}(\hat{\boldsymbol{\beta}} - \boldsymbol{\beta}) \to_d \mathcal{N}(0, \boldsymbol{V(W)})$$

where

$$\boldsymbol{V(W)} = \boldsymbol{A}^{-1}\boldsymbol{\Sigma}\boldsymbol{A}'^{-1}$$

If we assume that $\boldsymbol{W}$ is diagonal, we can go further:

$$\sqrt{n}(\hat{\boldsymbol{\beta}} - \boldsymbol{\beta}) = (\frac{1}{n}\sum_{i=1}^{n} w_i\boldsymbol{x}_i\boldsymbol{x}_i')^{-1}(\frac{1}{\sqrt{n}}\sum_{i=1}^{n} w_i\boldsymbol{x}_i e_i)$$

Then we get the result

$$\sqrt{n}(\hat{\boldsymbol{\beta}} - \boldsymbol{\beta}) \rightarrow_d \mathcal{N}(0, \mathbb{E}[w_i \boldsymbol{x}_i \boldsymbol{x}_i']^{-1} \mathbb{E}[w_i^2 \boldsymbol{x}_i \boldsymbol{x}_i' e_i^2] \mathbb{E}[w_i \boldsymbol{x}_i \boldsymbol{x}_i']^{-1}).$$

Now, suppose that $\mathbb{V}[\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{W}] = \sigma^2 \boldsymbol{I}_n$. Then

$$\begin{aligned}
\boldsymbol{\Sigma} = \mathbb{V}[\frac{1}{\sqrt{n}} \boldsymbol{X}' \boldsymbol{W} \boldsymbol{e}] &= \frac{1}{n} \mathbb{E}[\boldsymbol{X}' \boldsymbol{W} \boldsymbol{e} \boldsymbol{e}' \boldsymbol{W} \boldsymbol{X}] \\
&= \frac{1}{n} \mathbb{E}[\mathbb{E}[\boldsymbol{X}' \boldsymbol{W} \boldsymbol{e} \boldsymbol{e}' \boldsymbol{W} \boldsymbol{X} | \boldsymbol{X}, \boldsymbol{W}]] \\
&= \frac{1}{n} \mathbb{E}[\boldsymbol{X}' \boldsymbol{W} \mathbb{E}[\boldsymbol{e} \boldsymbol{e}' | \boldsymbol{X}, \boldsymbol{W}] \boldsymbol{W} \boldsymbol{X}] \\
&= \frac{\sigma^2}{n} \mathbb{E}[\boldsymbol{X}' \boldsymbol{W} \boldsymbol{W} \boldsymbol{X}]
\end{aligned}$$

And then

$$\boldsymbol{V}(\boldsymbol{W}) = \frac{\sigma^2}{n} \boldsymbol{A}^{-1} \mathbb{E}[\boldsymbol{X}' \boldsymbol{W} \boldsymbol{W} \boldsymbol{X}] \boldsymbol{A}'^{-1}$$

2.3

We can use the plug-in estimator

$$\hat{\boldsymbol{V}}(\boldsymbol{W}) = \frac{s^2}{n} \hat{\boldsymbol{A}}^{-1} \left(\frac{1}{n} \boldsymbol{X}' \boldsymbol{W} \boldsymbol{W} \boldsymbol{X} \right) \hat{\boldsymbol{A}}'^{-1}$$

where

$$s^2 = \frac{1}{n - d} \sum_{i=1}^{n} \hat{\epsilon}_i^2$$

and

$$\hat{\boldsymbol{A}} = \frac{1}{n} \boldsymbol{X}' \boldsymbol{W} \boldsymbol{X}.$$

Thus, we get

$$\hat{\boldsymbol{V}}(\boldsymbol{W}) = s^2 (\boldsymbol{X}' \boldsymbol{W} \boldsymbol{X})^{-1} (\boldsymbol{X}' \boldsymbol{W} \boldsymbol{W} \boldsymbol{X}) (\boldsymbol{X}' \boldsymbol{W} \boldsymbol{X})^{-1}.$$

2.4

I implemented the required inference procedure in `R` and `Mata` (see attached Appendix for the source code.) Below I focus on the results from `R`. To test whether using a symmetric inverse of Cholesky inverse produce different results I first generated some random data. The choice of inverse does not change any results meaningfully.

2.5

(a) The OLS results from my matrix-based implementations are reported in Table 1.

Table 1: **OLS Results from Matrix-based Implementation in R**

	$\hat{\beta}$	s.e.	t-stat	p-value	CI lower	CI upper
_const	6485.553	4462.511	1.453	0.147	-2260.807	15231.913
treat	1535.482	631.026	2.433	0.015	298.694	2772.271
black	-2592.377	786.016	-3.298	0.001	-4132.939	-1051.814
age	39.341	40.013	0.983	0.326	-39.083	117.764
educ	-740.540	934.004	-0.793	0.428	-2571.154	1090.074
educ^2	60.082	53.161	1.130	0.259	-44.111	164.276
earn74	-0.030	0.103	-0.291	0.771	-0.231	0.171
black × earn74	0.175	0.130	1.346	0.179	-0.080	0.431
u74	1316.032	1488.910	0.884	0.377	-1602.179	4234.243
u75	-1167.688	1261.004	-0.926	0.355	-3639.210	1303.833

Notes: column (3) reports standard errors computed using the usual White sandwich form (not adjusted for degrees of freedom); column (4) reports the t-statistic for the null hypothesis $H_0 : \beta_0 = 0$; columns (6) and (7) report the estimated 95% confidence interval.

(b) I ran the same regression using R's `lm` and `sandwich` packages and Stata's `reg` function (with the option `r` for robust standard errors). The results are reported in Tables 2 below (I omit the t-statistics, p-values and confidence intervals for brevity and ease of comparison – after all, these statistics are all just functions of the estimated coefficients and standard errors).

Table 2: **OLS Results from In-built Functions in R and Stata**

	$\hat{\beta}$		s.e.	
	lm (R)	reg (Stata)	lm (R)	reg (Stata)
_const	6485.553	6485.553	4462.511	4513.513
treat	1535.482	1535.482	631.026	638.238
black	-2592.377	-2592.377	786.016	794.9991
age	39.341	39.34052	40.013	40.47006
educ	-740.540	-740.54	934.004	944.6787
educ^2	60.082	60.08233	53.161	53.76838
earn74	-0.030	-.0298577	0.103	.1036976
black × earn74	0.130	.1753547	0.175	.1318064
u74	1316.032	1316.032	1488.910	1505.927
u75	-1167.688	-1167.688	1261.004	1275.416

Notes: standard errors for the R implentation are computed using the usual White sandwich form (not adjusted for degrees of freedom), while the standard errors for the Stata implementation are computed using option r.

The results from R's `lm` (and `sandwich`) package correspond exactly to the results from my matrix implementation. The estimated coefficients from Stata's `reg` function are also exactly the same as those from my matrix implementation. However, the estimated standard errors from `reg` differ slightly from those estimated in my matrix implementation. This is because Stata's `r(obust)` option scales the usual White sandwich form standard errors (which I used in my matrix implementation, and in the `lm` results) by $n/(n-d)$ to adjust for degrees of freedom. Accordingly, the option `"HC1"` in R's `vcovHC` function (contained in the `sandwich` package) produces exactly the same standard errors as the Stata implementation above.

3 Analysis of experiments

3.1 Neyman's approach

(a) We'll follow the derivation in section 6.3 of Imbens and Rubin's textbook. We start with

$$T_{DM} = \frac{1}{N_1}\sum_{i=1}^{n} D_i(1)Y_i - \frac{1}{N_0}\sum_{i=1}^{n} D_i(0)Y_i$$

Next, note that

$$D_i(1)Y_i = D_i(1)[D_i(1)Y_i(1) + (1 - D_i(1))Y_i(0)] = D_i(1)Y_i(1),$$

since $D_i(1)^2 = D_i(1)$. SImilarly,

$$D_i(0)Y_i = D_i(0)Y_i(0),$$

since $D_i(0)D_i(1) = 0$. Thus,

$$\begin{aligned} T_{DM} &= \frac{1}{N_1}\sum_{i=1}^{n} D_i(1)Y_i(1) - \frac{1}{N_0}\sum_{i=1}^{n} D_i(0)Y_i(0) \\ &= \frac{1}{n}\sum_{i=1}\left(\frac{D_i(1)Y_i(1)}{N_1/n} - \frac{D_i(0)Y_i(0)}{N_0/n}\right). \end{aligned}$$

Now, since potential outcomes are non-random, the only component of T_{DM} that is random is $D_i(t)$. Given complete randomization, we know that

$$\mathbb{E}[D_i(1)] = \mathbb{E}[\mathbf{1}(T_i = 1)] = \mathbb{P}[T_i = 1] = N_1/n$$

And,

$$\mathbb{E}[D_i(0)] = 1 - \mathbb{E}[D_i(1)] = N_0/n$$

Thus, we get the desired result

$$\mathbb{E}[T_{DM}] = \frac{1}{n}\sum_{i=1}(Y_i(1) - Y_i(0)) \equiv \tau_{ate},$$

so that T_{DM} is unbiased for τ_{ate}.

Using the `Lalonde_1986.csv` data, I estimated $T_{DM} = \$1794.343$.

(b) With the given information, an asymptotically conservative 95% confidence interval of the average treatment effect is given by:

$$CI_{95} = \left[T_{DM} - 1.96\sqrt{\frac{\bar{S}_0^2}{N_0} + \frac{\bar{S}_1^2}{N_1}}, T_{DM} + 1.96\sqrt{\frac{\bar{S}_0^2}{N_0} + \frac{\bar{S}_1^2}{N_1}}\right]$$

I computed the confidence interval in `R` (see attached code in the Appendix), with the results presented in the Table 3.

Table 3: **Asymptotically conservative 95% confidence interval for T_{DM}**

T_{DM}	$\sqrt{\frac{\bar{S}_0^2}{N_0}+\frac{\bar{S}_1^2}{N_1}}$	CI lower	CI upper
1794.343	670.9967	479.2137	3109.473

Notes:

3.2 Fisher's approach

The key insight of Fisher's approach is that all 'missing' potential outcomes can be inferred from the observed outcomes (see Imbens and Rubin, 2015, pp 60-61). This insight directly follows from the sharp null hypothesis of no treatment effect; that is:

$$H_0 : Y_i(0) = Y_i(1) \text{ for all } i.$$

Here's how Fisher's exact p-value (FEP) approach works in practice. Let the assignment vector be $\boldsymbol{T} = (T_1, ..., T_n)$ where T_i is a treatment dummy, and suppose we have chosen a test statistic T (in our case we will use T_{DM} and T_{KS}). Under the sharp null hypothesis above, we can calculate the value of T for different treatment assignment vectors. Rather than computing T for every possible assignment vector, we simply calculate T for a randomly chosen subset of assignment vectors.[1] For instance, let T^{obs} be the observed value of the test statistic. Then randomly draw an n-dimensional vector with N_1 ones and $n - N_1$ zeroes. Calculate the test statistic for this draw, say T^1. Repeat this process $K - 1$ times, each time drawing a new assignment vector and calculating the statistic T^k. We approximate the p-value for this test statistic by the fraction of the K statistics that are as extreme, or more extreme that T^{obs}:

$$\hat{p} = \frac{1}{K}\sum_{k=1}^{K} \mathbf{1}[T^k \geq T^{obs}]$$

Thus, if $\hat{p}$ is close to zero, this says that the observing T^{obs} would be very unlikely given H_0, which is taken as evidence against the null.

(a) Table 4 reports the approximation of Fisher's p-value for the sharp null of no treatment effect.

Table 4: **P-values for LaLonde Data, for Null Hypothesis of No Treatment Effect**

Test statistic	Observed statistic	P-value
T_{DM}	1794.343	0.0026
T_{KS}	0.132	0.038

Notes: to compute the p-values, I randomly draw 250,000 assignment vectors; the p-value is proportion of draws at least as large as observed statistic.

(b) I follow Imbens & Rubin, 5.7. To construct 95% confidence intervals for the treatment effect we first consider the null hypothesis of a constant additive treatment effect:

$$H_0 : Y_i(1) = Y_i(0) + C$$

[1] In the Lalonde data, there is a population $n = 445$, with $N_1 = 185$ units assigned to treatment. Accordingly, there are $\binom{445}{185}$ different possible assignment vectors (i.e. a very, very large number)!

Just as with the sharp null of no treatment effect, the sharp null of a constant treatment effect also allows us to infer missing potential values for all units. Given this complete knowledge we can again calculate p-values for a range of values of the postulated effect, C. The set of values of C where we get p-values larger than 0.05, provides a 95% "Fisher" interval for a common additive treatment effect.

Table 5 shows the p-value associated with a constant treatment effect C. Note that in this case the test statistic is $T = |\bar{Y}_1 - \bar{Y}_0 - C|$, and the p-value is define in the usual way. The Fisher 95% confidence interval for the treatment effect is $[569, 3028]$.

Table 5: **P-Values for Tests of Constant Treatment Effects**

Hypothesized treatment effect (C)	P-value
4750	0.0000
4500	0.0000
4250	0.0002
4000	0.0003
3750	0.0024
3500	0.0074
3250	0.0219
3028	0.05
3000	0.0596
2750	0.1375
2500	0.2723
2250	0.4829
2000	0.7515
1750	0.9431
1500	0.6479
1250	0.4003
1000	0.2157
750	0.1025
569	0.051
500	0.0430
250	0.0147
0	0.0051
-250	0.0012
-500	0.0003
-750	0.0000
-1000	0.0000

Notes: to compute the p-values, I randomly draw 10,000 assignment vectors; the p-value is proportion of draws at least as large as observed statistic.

Another (I think equivalent) way to compute the 95% confidence interval is via bootstrap. In this case the null is $H_0 = Y_i(1) = Y_i(0) + \hat{\tau}$ where $\hat{\tau}$ is the estimated treatment effect using T_{DM}. Again, given that this is a sharp null we can infer potential values for all units. As above we can easily get the randomized distribution of $\hat{\tau}$. Then we simply extract the 2.5% and 97.5% quantiles.

3.3 Power calculations

(a) We have the following two-sided testing problem:

$$H_0 : \tau_{ate} = \tau_0 \text{ v.s. } H_1 : \tau_{ate} \neq \tau_0.$$

with the test statistic:

$$T(Y) = \frac{T_{DM} - \tau_0}{s.e.(T_{DM})}, \text{ where } s.e.(T_{DM}) = \sqrt{\frac{\sigma_1^2}{N_1} + \frac{\sigma_0^2}{N_0}}$$

We know that under H_0, the test statistic is asymptotically normal. That is,

$$\frac{T_{DM} - \tau_{ate}}{s.e.(T_{DM})} \rightarrow_d \mathcal{N}(0, 1).$$

Denote the true parameter τ_{ate} as τ. Now, the power function is given by

$$\begin{aligned}
\beta(\tau) &= \mathbb{P}[\text{reject } H_0] \\
&= \mathbb{P}[|T(Y)| > c^*], \text{ with } c^* = \Phi^{-1}(1 - \alpha/2) \\
&= 1 - \mathbb{P}[|T(Y)| \leq c^*] \\
&= 1 - \mathbb{P}[-c^* \leq T(Y) \leq c^*] \\
&= 1 - \mathbb{P}\left[-c^* \leq \frac{T_{DM} - \tau_0}{s.e.(T_{DM})} \leq c^*\right] \\
&= 1 - \mathbb{P}\left[-c^* + \frac{\tau_0 - \tau}{s.e.(T_{DM})} \leq \frac{T_{DM} - \tau}{s.e.(T_{DM})} \leq c^* + \frac{\tau_0 - \tau}{s.e.(T_{DM})}\right] \\
&\approx 1 - \left[\Phi\left(c^* + \frac{\tau_0 - \tau}{s.e.(T_{DM})}\right) - \Phi\left(-c^* + \frac{\tau_0 - \tau}{s.e.(T_{DM})}\right)\right]
\end{aligned}$$

Then, the conservative power function is

$$\beta(\tau) = 1 - \Phi\left(c^* + \frac{\tau_0 - \tau}{s.\hat{e}.(T_{DM})}\right) + \Phi\left(-c^* + \frac{\tau_0 - \tau}{s.\hat{e}.(T_{DM})}\right)$$

where

$$s.\hat{e}.(T_{DM}) = \sqrt{\frac{\bar{S}_0^2}{N_0} + \frac{\bar{S}_1^2}{N_1}}$$

I plot $\beta(\tau)$ for $\tau_0 = 0$ below.

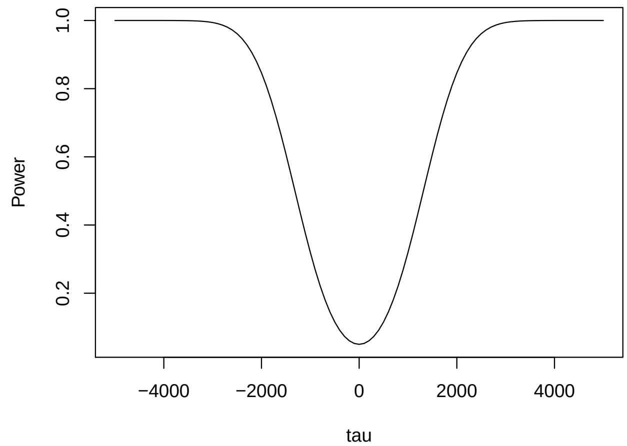

Figure 1: Power function for $\tau_0 = 0$

(b) First note that for $\tau = 0$ we can write the power function as

$$\beta(\tau) = 1 - \Phi\left(c^* - \frac{\tau}{\sqrt{\frac{\sigma_1^2}{Np} + \frac{\sigma_0^2}{N(1-p)}}}\right) + \Phi\left(-c^* - \frac{\tau}{\sqrt{\frac{\sigma_1^2}{Np} + \frac{\sigma_0^2}{N(1-p)}}}\right) \tag{6}$$

where $p = N_1/N$ and N is the total sample size. We want to solve (6) for N given the following parameters: $\tau = 1000, \beta = 0.8$ and $p = 2/3$. Plugging in the sample variances, $\bar{S}_1^2$ and $\bar{S}_0^2$, for σ_1^2 and σ_0^2, respectively gives the solution $N^* = 1437$.

4 Appendix: R and STATA code

4.1 R code

4.1.1 Question 2: implementing least squares

```
## ECON675: ASSIGNMENT 1
## Q2: IMPLEMENTING LEAST SQUARES ESTIMATORS
## Anirudh Yadav
## 8/16/2018

######################################################################
# Load packages, clear workspace
######################################################################
rm(list = ls())             #clear workspace
library('MASS')             #for ginv function
library('sandwich')         #for variance-covargiance estimation
options(scipen = 999)       #forces R to use normal numbers instead of scientific notation


######################################################################
# Generate some random data
######################################################################
# Draw some random numbers from the std norm distribution
# and use these as our data for testing the OLS function below.
# Specifically, we'll use these data to test whether the two
# different types of inverses give the same results.

# Generate covariates
X        <- replicate(3, rnorm(100))

# Generate iid mean-zero errors
U        <- rnorm(100)

# Impose a `true' beta
betatrue <- as.matrix(c(1,2,3))

# Compute the resulting Y
Y        <- X%*%betatrue + U


######################################################################
# Q4: Write function for computing OLS results + related stats
######################################################################

linear_reg <- function(X,Y, cholinv=FALSE, alpha=0.05){

# Compute crossproduct matrix
  if(!cholinv){
      # using symmetric inverse
      M = solve(crossprod(X))

  }else{
      #using Cholesky inverse
      m  = chol(crossprod(X))
      M = chol2inv(m)
  }

      # Compute OLS estimator
      beta = M%*%crossprod(X,Y)

      # Compute residuals vector
      u     <- Y - X%*%beta

      # Construct diagonal matrix of squared residuals
      D     <- diag(as.numeric(u^2))
```

```
      # Compute variance-covarinance matrix
      n <- nrow(X)
      k <- ncol(X)
      V     <- n*M%*%t(X)%*%D%*%X%*%M

      # Compute standard errors
      se    <- as.matrix(sqrt(diag(V)/n))

      # Compute t-stats for testing H0:beta_k=0
      t=matrix()
      for (i in 1:k) {
              t[i] = beta[i]/se[i]
      }
      t <- as.matrix(t)

      # Compute p-values
      p=matrix()
      for (i in 1:k) {
              p[i] = 2*pt(abs(t[i]),df=n-k,lower.tail=FALSE)
      }
      p <- as.matrix(p)

      #Compute lower bounds of the CI
      CIlower=matrix()
      for (i in 1:k) {
               CIlower[i] = beta[i] - qnorm(1-alpha/2)*se[i]
      }

      #Compute upper bounds of the CI
      CIupper=matrix()
      for (i in 1:k) {
              CIupper[i] = beta[i] + qnorm(1-alpha/2)*se[i]
      }


    #Combine results into a dataframe
    results             <- cbind(beta,se,t,p,as.matrix(CIlower),as.matrix(CIupper))
    colnames(results)   <- c("beta","se","t","p","CIlower","CIupper")

  return(results)
}

# Test whether choice of inverse affects results
results1 <- linear_reg(X,Y)
results2 <- linear_reg(X,Y,cholinv=TRUE)
diff     <- results1-results2

######################################################################
# Q5: Input data, create additional covariates
######################################################################

# Get LaLonde data
data <- read.csv('PhD_Coursework/ECON675/HW1/LaLonde_1986.csv')

# Convert to data frame
df <- as.data.frame(data)

# Add a column of ones for intercept in regression
df$ones <- 1

# Create squared education variable
df$educsq <- (df$educ)^2

# Create black-earn74 interaction
df$black_earn74 <- (df$black)*(df$earn74)

# Create X matrix
X <- cbind(df$ones,df$treat,df$black,df$age,df$educ,df$educsq,df$earn74,df$black_earn74,df$u74,df$u75)
```

```
# Create Y vector
Y <- cbind(df$earn78)

######################################################################
# Q5(a): run matrix implementation of OLS
######################################################################

# Run linear_reg function using different inverses
results1 <- linear_reg(X,Y)
results2 <- linear_reg(X,Y,cholinv=TRUE)


######################################################################
# Q5(b): compute OLS results using in-build lm() function
######################################################################

# Fit the linear regression
ols    <- lm(Y ~ X-1)      #nb. the minus 1 is becuase the lm package includes an intercept automatically
                           #    but the X matrix already includes an intercept

# Compute Eiker-White variance-covariance matrix (using sandwhich pkg)
# These match the standard errors from the manual computation above.
# Note that STATA's default when "r" is specified is equivalent to HC1!
V_check     <- vcovHC(ols, type = "HC0")
se_check    <- as.matrix(sqrt(diag(V_check)))
```

4.1.2 Question 3: analysis of experiments

```
## ECON675: ASSIGNMENT 1
## Q3: ANALYSIS OF EXPERIMENTS
## Anirudh Yadav
## 8/16/2018

######################################################################
# Load packages, clear workspace
######################################################################
rm(list = ls())             #clear workspace
library(dplyr)              #for data manipulation
library(ggplot2)            #for pretty plots
library(boot)               #for bootstrapping
options(scipen = 999)       #forces R to use normal numbers instead of scientific notation


######################################################################
# Input data
######################################################################

# Get LaLonde data
data <- read.csv('PhD_Coursework/ECON675/HW1/LaLonde_1986.csv')

# Convert to data frame
data <- as.data.frame(data)

######################################################################
# Q1 (a): difference in means estimator
######################################################################

# Rename variables
Y.obs       = data$earn78
treat       = data$treat

# Compute difference in means estimator
T.obs.dm    = mean(Y.obs[treat==1],na.rm=TRUE)-mean(Y.obs[treat==0],na.rm=TRUE)


######################################################################
```

```
# Q1 (b): conservative confidence intervals
######################################################################
N1 = sum(treat)
N0 = nrow(data)-N1

# Compute "conservative" standard error
s.1               <- sd(Y.obs[treat==1],na.rm=TRUE)^2
s.0               <- sd(Y.obs[treat==0],na.rm=TRUE)^2
se.conserv        <- sqrt(1/N1*s.1 + 1/N0*s.0)

# Compute lower and upper bounds of the interval and store in vector
CI.lower = T.obs.dm - qnorm(0.975)*se.conserv
CI.upper = T.obs.dm + qnorm(0.975)*se.conserv

# Store results
results           <- cbind(T.obs.dm,se.conserv,CI.lower,CI.upper)


######################################################################
# Q2 (a): Fisher Exact P-values
######################################################################

# The FEP function computes Fisher (approximate) p-values for
# sharp null of no treatment effect, for the difference in means statistic
# and the K-S statistic.

# NOTES:
# This function takes ~150 secs to run using the KS statistic with 250k draws!
# Is there a more efficient way to do this?


FEP <- function(K=249999,ks=FALSE){

  # Initialze vector of length K
  T.vec = vector(length=K)

  # Generate K random draws of the assignment vector
  T.MAT = replicate(K,sample(treat))

  if(!ks){

  # Compute observed difference in means
  T.obs    = mean(Y.obs[treat==1],na.rm=TRUE)-mean(Y.obs[treat==0],na.rm=TRUE)

      # Loop through random draws of the assignment vector,
      # compute and store the test statistic
      for (i in 1:K) {
              T.dm    = mean(Y.obs[T.MAT[,i]==1],na.rm=TRUE)-mean(Y.obs[T.MAT[,i]==0],na.rm=TRUE)
              T.vec[i]            <- T.dm
            }

      }else{

    # USE K-S statistic
    options(warn=-1) #turn warnings off

    # Compute observed KS statistic
    T.obs               <- ks.test(Y.obs[treat==1],Y.obs[treat==0])$statistic

    # Loop through random draws of the assignment vector,
    # compute and store the test statistic
    for (i in 1:K) {
      T.ks               <- ks.test(Y.obs[T.MAT[,i]==1],Y.obs[T.MAT[,i]==0])$statistic
      T.vec[i]           <- T.ks
    }

  }
```

```
  options(warn=0) #turn warnings back on!


  # Calculate p-value
  p = 1/K*sum(T.vec>=T.obs)

  return(p)

}


######################################################################
# Q2 (a): Fisher confidence intervals
######################################################################

## First I follow the approach in Imbens & Rubin, s5.7 ##

FisherInterval <- function(K=9999,C.vec=seq(5000,-1500,-250)){

  # Initialize vector of lenght C.vec
  P.vec = vector(length=length(C.vec))

  # Initialze vector of length K
  T.vec = vector(length=K)

  # Generate K random draws of the assignment vector
  T.MAT = replicate(K,sample(treat))

  for (j in 1:length(C.vec)){

    # Compute observed difference in means
    T.obs    = abs(mean(Y.obs[treat==1],na.rm=TRUE)-mean(Y.obs[treat==0],na.rm=TRUE)- C.vec[j])

    # Compute missing potential outcomes under the null
    Y.1 = ifelse(treat==1,Y.obs,Y.obs+C.vec[j])
    Y.0 = ifelse(treat==1,Y.obs-C.vec[j],Y.obs)

      for (i in 1:K) {
          T.dm    = abs(mean(Y.1[T.MAT[,i]==1],na.rm=TRUE)-mean(Y.0[T.MAT[,i]==0],na.rm=TRUE) - C.vec[j])
          T.vec[i]            <- T.dm
          }

    p = 1/K*sum(T.vec>=T.obs)
    P.vec[j] <- p
  }
  return(cbind(C.vec,P.vec))
}

# Run function with 10000 draws
# FisherInterval()


## Another way to compute the CI is using bootstrap ##

  # Compute missing potential outcomes under the null
  Y.1 = ifelse(treat==1,Y.obs,Y.obs+T.obs.dm)
  Y.0 = ifelse(treat==1,Y.obs-T.obs.dm,Y.obs)

  # Specify the statistic that we will compute for different permutations
  T.dm <- function(x, ind) {
      T.k <- mean(Y.1[data$treat[ind]==1]) - mean(Y.0[data$treat[ind]==0])
      return(T.k)
  }

  # Run bootstrap
  boot.result  <- boot(data = data, R = 9999, statistic = T.dm, sim = "permutation", stype = "i")
  boot.CI      <- quantile(boot.result$t, c(0.025, 0.975))
```

```
  # Empirical 95% CI for constant treatment effect = T.obs.dm
  print (boot.CI)

######################################################################
# Q3 (a): Plot power function
######################################################################

PowerFun <- function(x) {
            1 - pnorm(qnorm(0.975)-x/se.conserv) + pnorm(-qnorm(0.975)-x/se.conserv)
            }

# Plot usinging ggplot2
p1   <- ggplot(data.frame(x = c(-5000, 5000)), aes(x = x)) + stat_function(fun = PowerFun)

# Plot using curve
curve(1 - pnorm(qnorm(0.975)-x/se.conserv) +
                pnorm(-qnorm(0.975)-x/se.conserv),-5000,5000,xlab="tau",ylab="Power")


######################################################################
# Q3 (b): Sample size calculation
######################################################################

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

# Write down the power function, which implicitly defines N
# [Note that I use the sample variances to proxy for the population variances]

Fun <- function(N){
        -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
```

4.2 STATA code

4.2.1 Question 2: implementing least squares

```
********************************************************************************
* ECON675: ASSIGNMENT 1
* Q2: IMPLEMENTING LEAST-SQUARES ESTIMATORS
* Anirudh Yadav
* 8/16/2018
********************************************************************************


********************************************************************************
* Preliminaries
********************************************************************************
clear all
set more off

* Set working directory
global dir "/Users/Anirudh/Desktop/GitHub"


********************************************************************************
* Import data, create additional covariates
********************************************************************************

* Import LaLonde data
import delimited using "$dir/PhD_Coursework/ECON675/HW1/LaLonde_1986.csv"

* Generate additional covariates
gen educsq=educ^2
gen black_earn74 = black*earn74
gen ones = 1

********************************************************************************
* Q4: Matrix implementation of OLS
********************************************************************************
mata:

// Create data matricies
X  = st_data(.,("ones", "treat", "black", "age", "educ", "educsq", "earn74","black_earn74", "u74","u75"))
Y  = st_data(.,("earn78"))

// Compute OLS point estimator
M  = invsym(cross(X,X))
betahat = M*cross(X,Y)

// Construct diagonal matrix of squared residuals
U  = Y - X*betahat
D = diag(U*U')

// Compute asymptotic White var-cov matrix
n  = rows(X)
d  = cols(X)

V  = n*M*X'*D*X*M

// Compute standard errors
se  = sqrt(diag(V)/n)
se   = diagonal(se)

// Compute t-statistics (element-wise division)
t       = betahat:/se

// Compute p-values
p       = 2*ttail(n-d, t)

// Compute 95% confidence intervals
```

```
CIlower = betahat - invnormal(0.975)*se
CIupper = betahat + invnormal(0.975)*se

betahat, se, t , p , CIlower, CIupper
end

********************************************************************************
* Q5(b): compute OLS results using reg function
********************************************************************************

reg earn78 treat black age educ educsq earn74 black_earn74 u74 u75, r

* NOTE that the differences in se's is because the "r" option implements the
* d.f adjustment; i.e. se(reg) = n/(n-d)*se(mata).
* I could easily implement the d.f. adjustment in the mata implementation, but
* I think it's nice to see the comparison.
```

4.2.2 Question 3: analysis of experiments

```
********************************************************************************
* ECON675: ASSIGNMENT 1
* Q3: ANALYSIS OF EXPERIMENTS
* Anirudh Yadav
* 8/26/2018
********************************************************************************

********************************************************************************
* Preliminaries
********************************************************************************
clear all
set more off

* Set working directory
global dir "/Users/Anirudh/Desktop/GitHub"


********************************************************************************
* Import data
********************************************************************************

* Import LaLonde data
import delimited using "$dir/PhD_Coursework/ECON675/HW1/LaLonde_1986.csv"


********************************************************************************
* Q1(a): Difference in means estimator
********************************************************************************

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

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

local tau = `mu1'-`mu0'
local v = sqrt(`V1'+`V0')
local T = `tau'/`v'
local pval = 2*normal(-abs(`T'))

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

```
local sd1 = round(`sd1', .0001)

di "`tau'"

********************************************************************************
* Q1(b): Conservative confidence intervals
********************************************************************************

local CIlower = `tau' - invnormal(0.975)*`v'
local CIupper = `tau' + invnormal(0.975)*`v'

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


********************************************************************************
* Q2(a): Fisher exact p-values
********************************************************************************

* Using difference in means estimator
permute treat diffmean=(r(mu_2)-r(mu_1)), reps(1999) nowarn: ttest earn78, by(treat)
matrix pval = r(p)
display "p-val = " pval[1,1]

* Using KS statistic
permute treat ks=r(D), reps(1999) nowarn: ksmirnov earn78, by(treat)
matrix pval = r(p)
display "p-val = " pval[1,1]


********************************************************************************
* Q2(b): Fisher confidence intervals
********************************************************************************

* 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

********************************************************************************
* Q3(a): Plot power function
********************************************************************************
local alpha = 0.05
local z     = invnormal(1-`alpha'/2)

* Plot power function
twoway (function y = 1 - normal(`z'- x/`v') + normal(-`z'-x/`v'), range(-4000 //
4000)), ///
   yline(`alpha', lpattern(dash)) yti("Power") xti("tau")

********************************************************************************
* Q3(b): Sample size calculation
********************************************************************************
mata:
```

```
mata clear

  function mypowerfunc(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)

}

y = st_data(., "earn78")
t = st_data(., "treat")

S1 = variance(select(y,t:==1))
S0 = variance(select(y,t:==0))
    p     = 2/3
tau   = 1000


mm_root(x=., &mypowerfunc(), 1000, 1500, 0, 10000, S0,S1, p ,tau)

x

end
```