STATS 305A H.W. # 1

Adhitya Venkatesh

October 3, 2017

PDF Problems:

1.)

We start with the normal equation that solves for the OLS estimate β:

$$X^T(Y - \beta X) = 0$$

We define $e = Y - \beta X$, resulting in the expression

$$X^T e = 0$$

For the column (say j WLOG) of X, we have a vector of ones, so

$$X_j^T e = 0$$

Equivalently,

$$\sum_{i=1}^{n} e_i = 0$$

Thus, we have proven the sum of residuals is 0 if the matrix X is designed as specified.

2.)

For 3D to 2D projection, we have the following matrices:

$$\mathbf{P_1} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

This projection maps any point (x,y,z) to (x,y,0). Thus the subpsace is simply the xy-plane.

$$\mathbf{P_2} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Similar to the previous example, this projection maps any point (x,y,z) to (0,y,z). Thus the subpsace is simply the yz-plane.

Taking the orthogonal complement (since P_1, P_2 are orthogonal projection matrices) of each of these projections, we have $P_1^c = I - P_1$ and similarly $P_2^c = I - P_2$

$$\mathbf{P_1^c} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

This projection maps any point (x,y,z) to (0,0,z), thus a mapping from $\mathbb{R}^{\not 2} -> \mathbb{R}$.

Finally, from $P_2^c = I - P_2$, we have

$$\mathbf{P_2^c} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

This projection maps any point (x,y,z) to (x,0,0), thus another valid mapping from $\mathbb{R}^{\not\perp} -> \mathbb{R}$.

Weisberg Problems:

1.1.1

The predictor is obviously ppgdp and the response ferility rate.

1.1.2

```
library(alr4) #contains datasets referenced in textbook
```

```
## Loading required package: car
```

```
## Warning: package 'car' was built under R version 3.3.2
```

```
## Loading required package: effects
```

```
## Warning: package 'effects' was built under R version 3.3.2
```

```
## Loading required package: carData
```

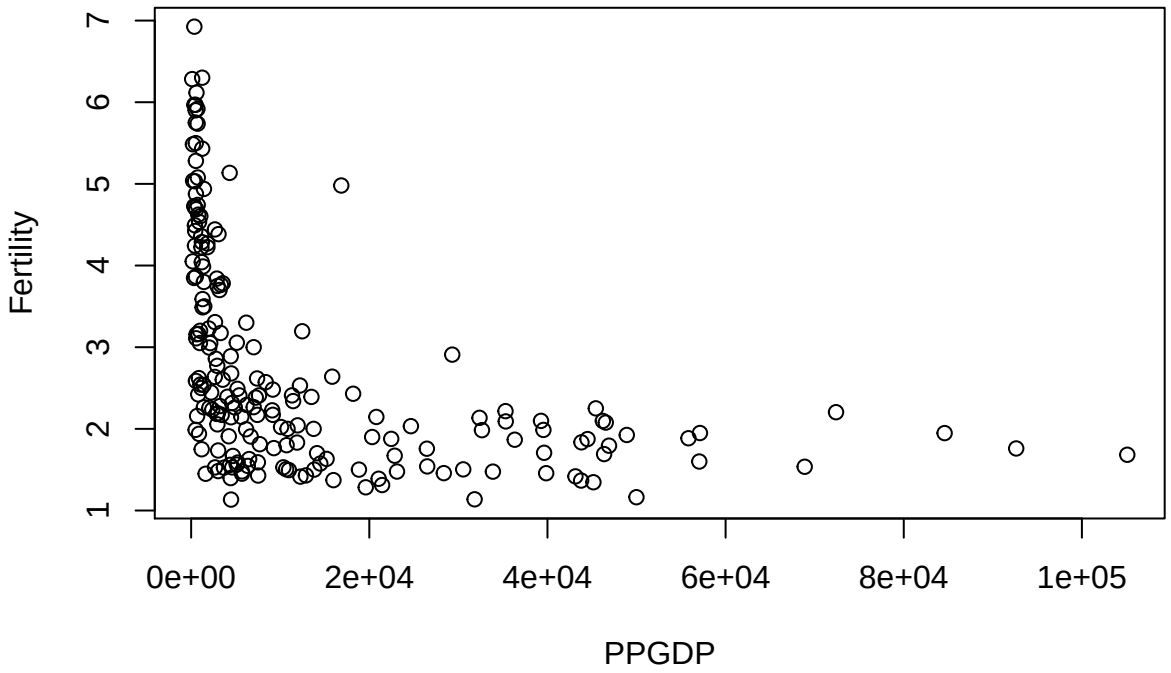
```
## Warning: package 'carData' was built under R version 3.3.2
```

```
##
## Attaching package: 'carData'
```

```
## The following objects are masked from 'package:car':
##
##     Guyer, UN, Vocab
```

```
## lattice theme set by effectsTheme()
## See ?effectsTheme for details.
```

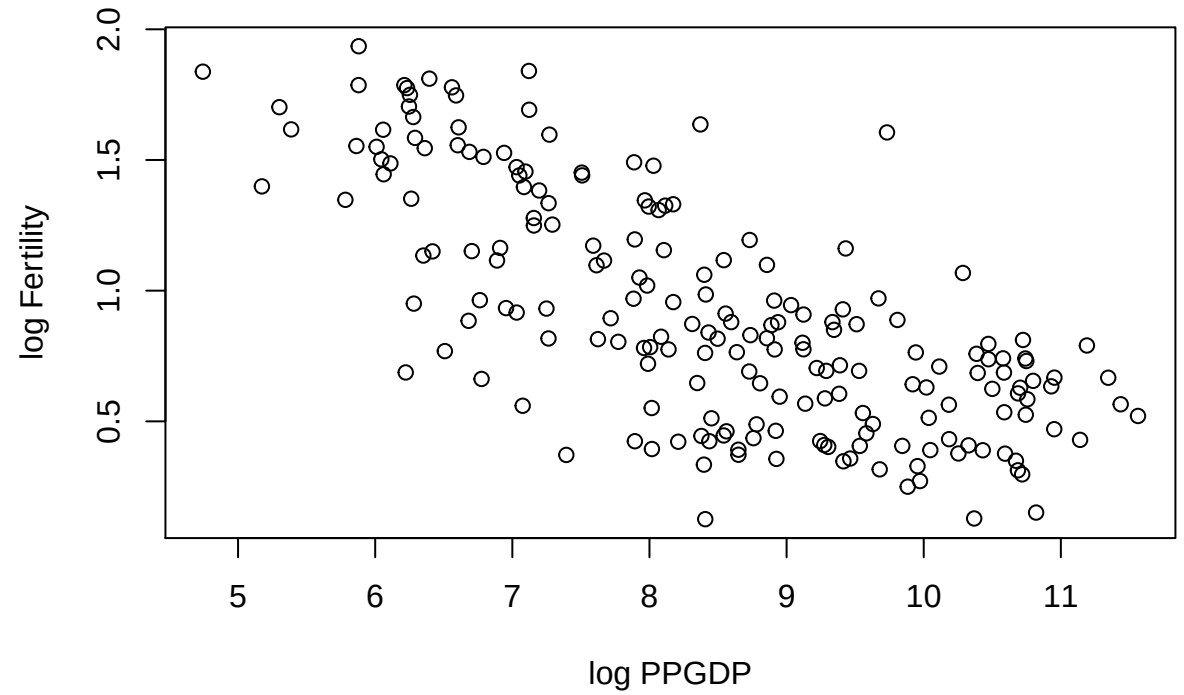
```
plot(UN11$ppgdp, UN11$fertility, xlab = 'PPGDP', ylab = 'Fertility')
```



Upon inspection, the spread of points and the rapid decline (faster than linear) of fertility rate with increasing ppgdg suggest a straight-line mean function will fail to capture the relationship between the variables as depicted in the graph and is thus implausible.

1.1.3

```
plot(log(UN11$ppgdp), log(UN11$fertility), xlab = 'log PPGDP', ylab = 'log Fertility')
```



Eye-test suggests a simple linear regression is quite plausible.

2.2.1

Points above the line y = x represent cities where the price of rice was greater in 2009 than 2003 (likely more

strongly affected by recession) and conversely those below the line represent cities where the price of rice was greater in 2003.

2.2.2

Thus, Largest increase: Vilnius (increase of 51.5) Largest decrease: Mumbai (decrease of 59)

2.2.3

Not necessarily. A slope coefficient of less than 1 indicates that the expected price of rice in 2009 was a fraction of what it was in 2003, that is the overall global average. However, in reality the price of rice is highly dependent on the market in which one buys/sells it.

2.2.4

It's clear from the graph that there are influential outliers such as Mumbai, Nairobi, and Vilnius that heavily skew the model parameters and a linear model assumes independence of sample points (cities in this case), which is clearly violated by the interconnectivity of the various rice markets as well as the fact that the economy of different cities were impacted at varying magnitudes by the recession.

2.3.1

A simple linear model would be more appropriate after taking logs of the prices because the transform brings the influential points closer to the rest of the data and thus will yield a more robust model.

2.3.2

Here β_0 can be interpreted as a multiplicative (proportionality) constant equating the response and exponentiated predictor. Finally, $\beta_1 = \beta$ (say) can be interpreted as follows: if the input is multiplied by a factor k then the output is multiplied by a factor k^β.

4. (2.13) The following is the desired function which will be used for the final problem.

```
linear.Regression = function(x, y, intercept = TRUE, B_0 = NULL)
{
  if(intercept){
      cbind(bias = 1, x)  #Column of ones to integrate bias
  }

  b_fit = solve(crossprod(x), crossprod(x,y))  #Solves Normal Eqn
  y_fitted = x%*%b_fit
  residuals = y - y_fitted
  x = as.matrix(x)  #For covariance operation to work properly

  result = list()

  result$coef = b_fit
  result$fit_vals = y_fitted
  result$resid = residuals
  result$R_SQ = 1 - sum((residuals)^2)/sum((y - mean(y))^2)
  result$covariance = cov(x)

  return(result)
}
```

2.13.1

We have the following results:

Coef: [29.917, 0.542] = [β_0, β_1] StdErr: [1.622, 0.026] R_sq = coef. of determination: 0.240 Variance Estimate = 5.132

2.13.2

Assuming the errors approximately follow a normal distribution, the z-value for the 99th percentile is 2.58, meaning

99% confint:

$$\beta_1 = [0.542 - 2.58(0.026), 0.542 + 2.58(0.026)] = [0.475, 0.609]$$

2.13.3

Using the same z-value from previous part and the variance estimation from 2.13.1, we have $y_d = 29.917 + 0.542(64) = 64.605$

Thus, 99% confidence interval is

$$y_d : [64.605 - 2.265 * 2.58, 64.605 + 2.265 * 2.58] = [58.76, 70.45]$$