DATA 609 HW 3

Oluwakemi Omotunde

February 22, 2018

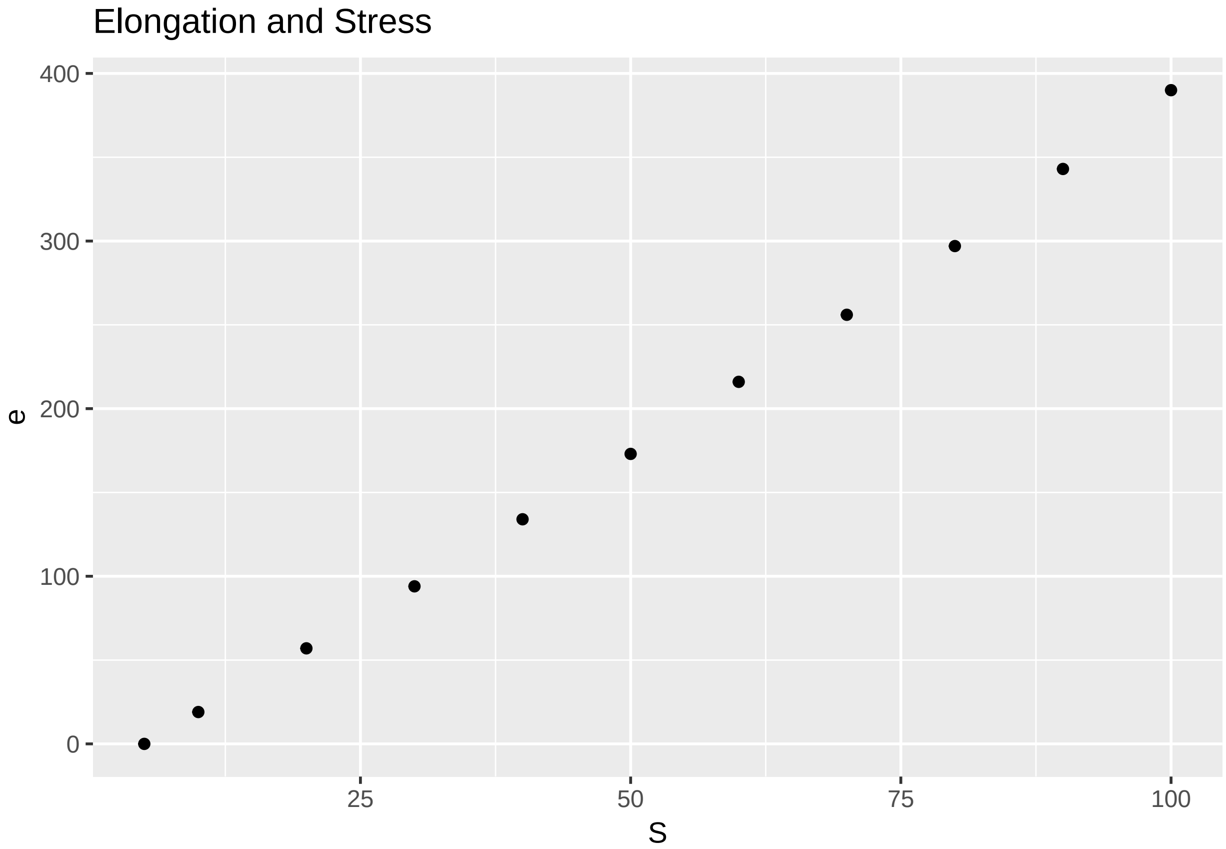
Page 113 # 2

The following table gives the elongation(e) in inches per inch(in./in/) for a given stress(S) on a steel wire measure in pounds per square inch(lb/in^2). Test the model e = c1*S by plotting the data. Estimate c1 graphically.

```
S <- c(5, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100)
e <- c(0, 19, 57, 94, 134, 173, 216, 256, 297, 343, 390)

df <- data.frame(S, e)

#plot(df)

library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 3.4.1
```

```
ggplot(df, aes(x = S, y = e)) + labs(title = "Elongation and Stress") + geom_point()
```



I would estimate c1 to be about 4.

Page 127 #10

Fit the data with the models given, using least squares.

```
b <- c("Mercury", "Venus", "Earth", "Mars", "Jupiter", "Saturn", "Uranus", "Neptune")
p <- c(7.60e6, 1.94e7, 3.16e7, 5.94e7, 3.74e8, 9.35e8, 2.64e9, 5.22e9)
d <- c(5.579e10, 1.08e11, 1.5e11, 2.28e11, 7.79e11, 1.43e12, 2.87e12, 4.5e12)
df_2 <- data.frame(b,p,d)
df_2
```

```
##         b        p         d
## 1 Mercury 7.60e+06 5.579e+10
## 2   Venus 1.94e+07 1.080e+11
## 3   Earth 3.16e+07 1.500e+11
## 4    Mars 5.94e+07 2.280e+11
## 5 Jupiter 3.74e+08 7.790e+11
## 6  Saturn 9.35e+08 1.430e+12
## 7  Uranus 2.64e+09 2.870e+12
## 8 Neptune 5.22e+09 4.500e+12
```

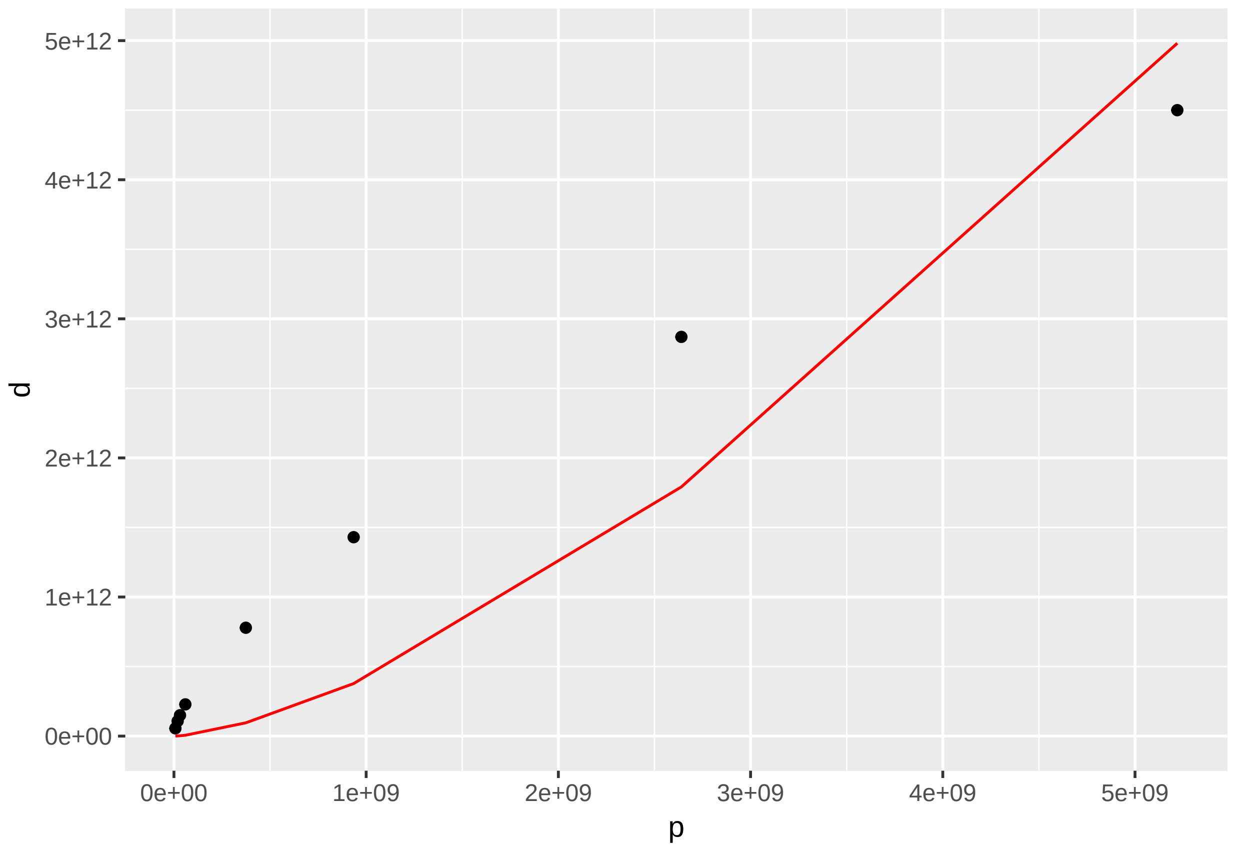
We are trying to fit to model y = ax^(3/2). We will solve for a first.

```
n <- sum(df_2$p^(3/2) * df_2$d)
den <- sum(df_2$p^3)

a <- n/den
a
```

```
## [1] 0.01320756
```

From above, our least squares model is: y = .013x^3/2. Lastly, lets go ahead and plot the actual points and the calculated points.

```
ggplot(df_2, aes(x = p, y = d)) +geom_point() + geom_line(aes(x = p, y = (a * p^(3/2))), color ="red")
```



Page 221 #2

Use a smooth polynoial to fit the data in Table 5.18 to obtain arrivals and unloading times. Compare results to those in Table 5.19 and 5.20.

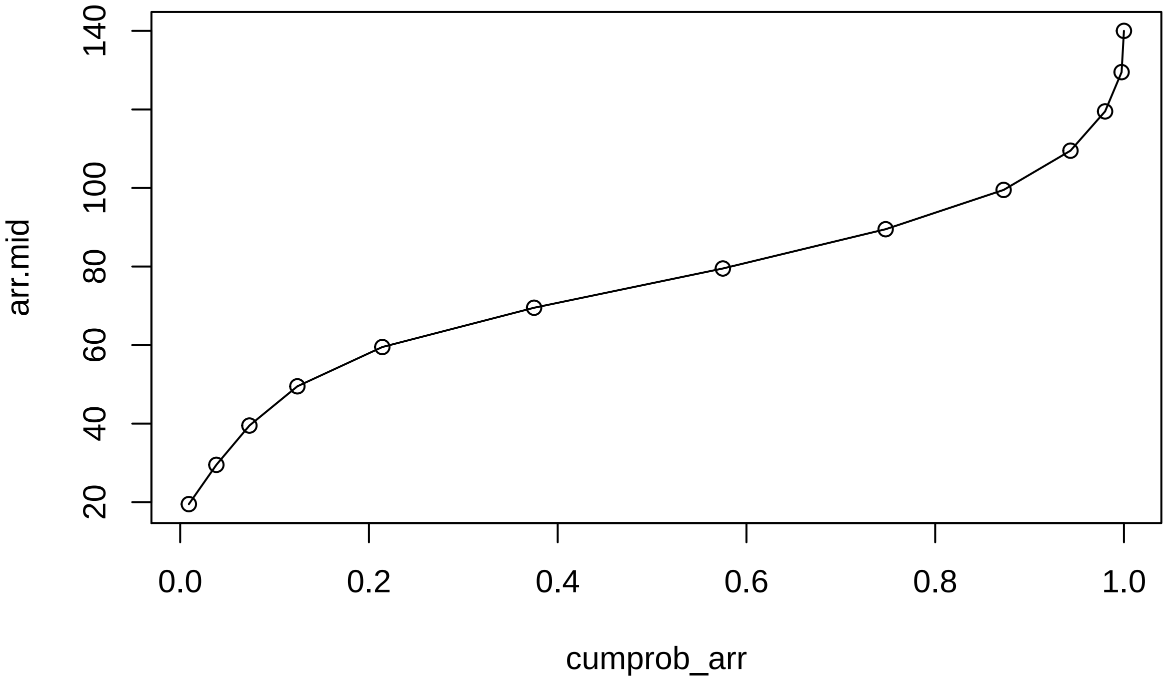
```
arr.lb <- 0:12
arr.lb <- arr.lb * 10 +15

arr.ub <- arr.lb + 9
arr.ub[length(arr.ub)] <- 145

arr.mid <- (arr.ub + arr.lb)/2
arr.occr <- c(11, 35, 42, 61, 108, 193, 240, 207, 150, 85, 44, 21, 3)
```

```
unload.lb <- 0:8
unload.lb <- unload.lb * 5 + 45

unload.ub <- unload.lb + 4
unload.ub[length(unload.ub)] <- 90

unload.mid <- (unload.lb + unload.ub)/ 2
unload.occr <- c(20, 54, 114, 103, 156, 223, 250, 171, 109)
```

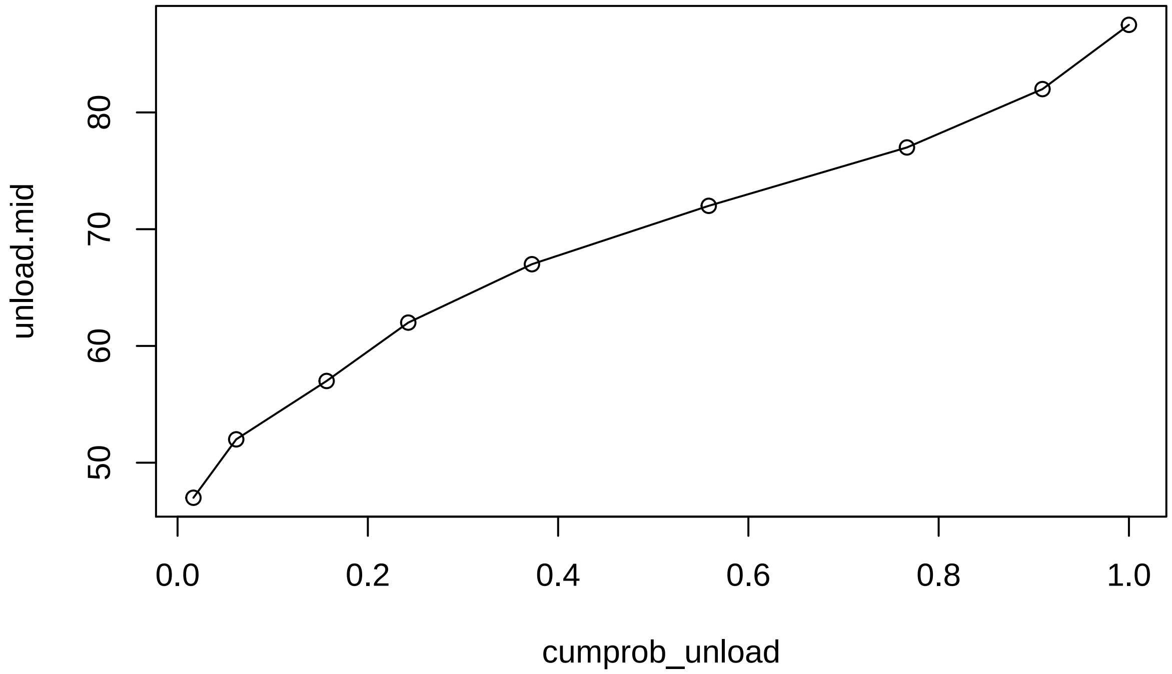
```
arr.prob <- arr.occr / sum(arr.occr)
cumprob_arr <- cumsum(arr.prob)

unload.prob <- unload.occr / sum(unload.occr)
cumprob_unload <- cumsum(unload.prob)
```

```
lmod.arr <- approxfun(cumprob_arr, arr.mid, method = "linear")
lmod.unload <- approxfun(cumprob_unload, unload.mid, method = "linear")
```

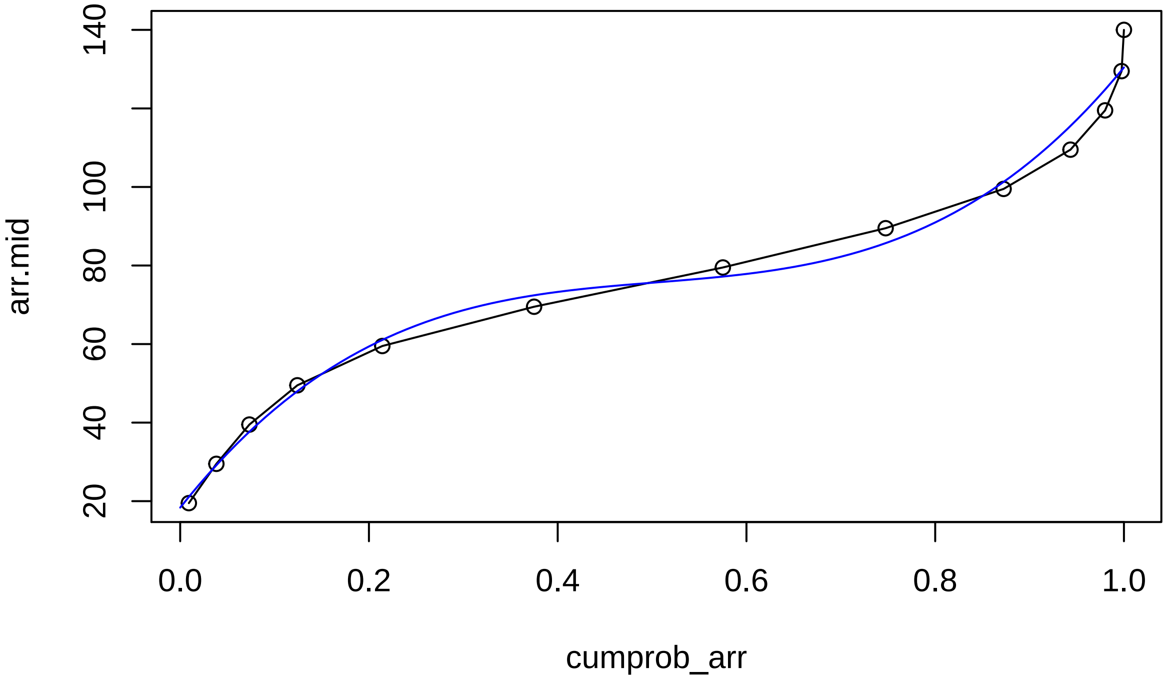
```
plot(cumprob_arr, arr.mid)
points(cumprob_arr, lmod.arr(cumprob_arr), type = "l")
```



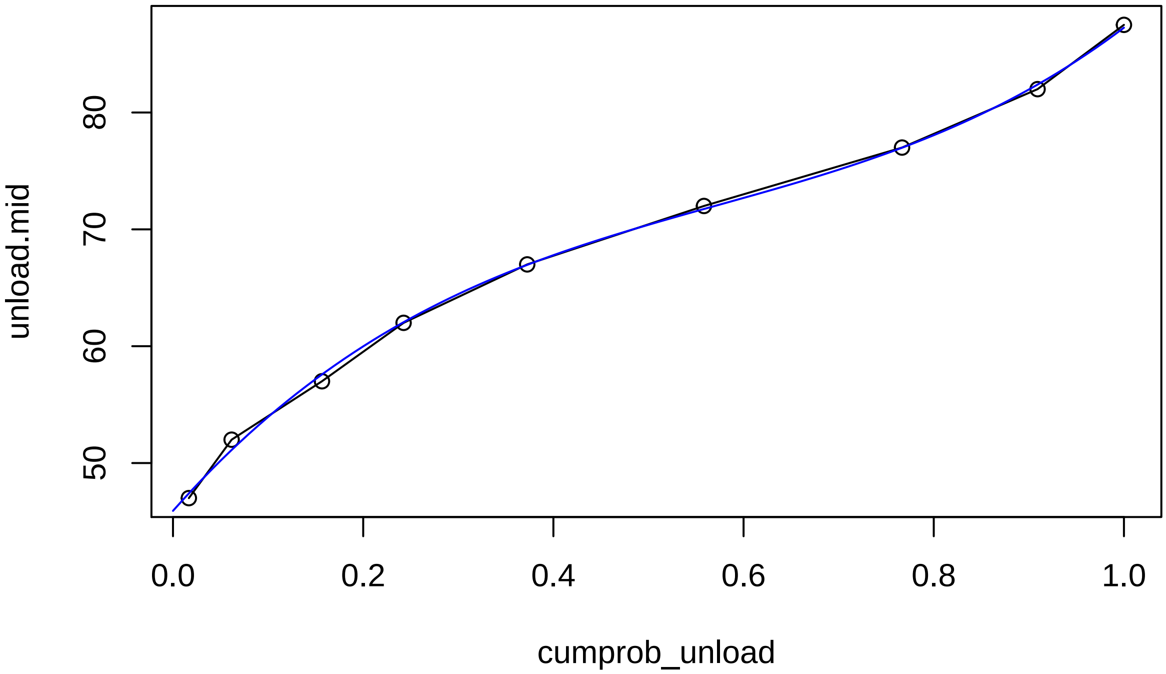
```
plot(cumprob_unload, unload.mid)
points(cumprob_unload, lmod.unload(cumprob_unload), type = "l")
```



```
arr.lm <- lm(arr.mid ~ poly(cumprob_arr, 3))
unload.lm <- lm(unload.mid ~ poly(cumprob_unload, 3))
plot(cumprob_arr, arr.mid)
points(cumprob_arr, lmod.arr(cumprob_arr), type = "l")

df_3 <- data.frame(cumprob_arr = seq(from = 0, to = 1, by = .01))
y <- predict(arr.lm, newdata = df_3)
points(df_3$cumprob_arr, y, type = "l", col = "blue")
```



```
plot(cumprob_unload, unload.mid)
points(cumprob_unload, lmod.unload(cumprob_unload), type = "l")

df_4 <- data.frame(cumprob_unload = seq(from = 0, to = 1, by = .01))
y <- predict(unload.lm, newdata = df_4)
points(df_4$cumprob_unload, y, type = "l", col = "blue")
```



It seems as though both models are good reps of the data.