HW3

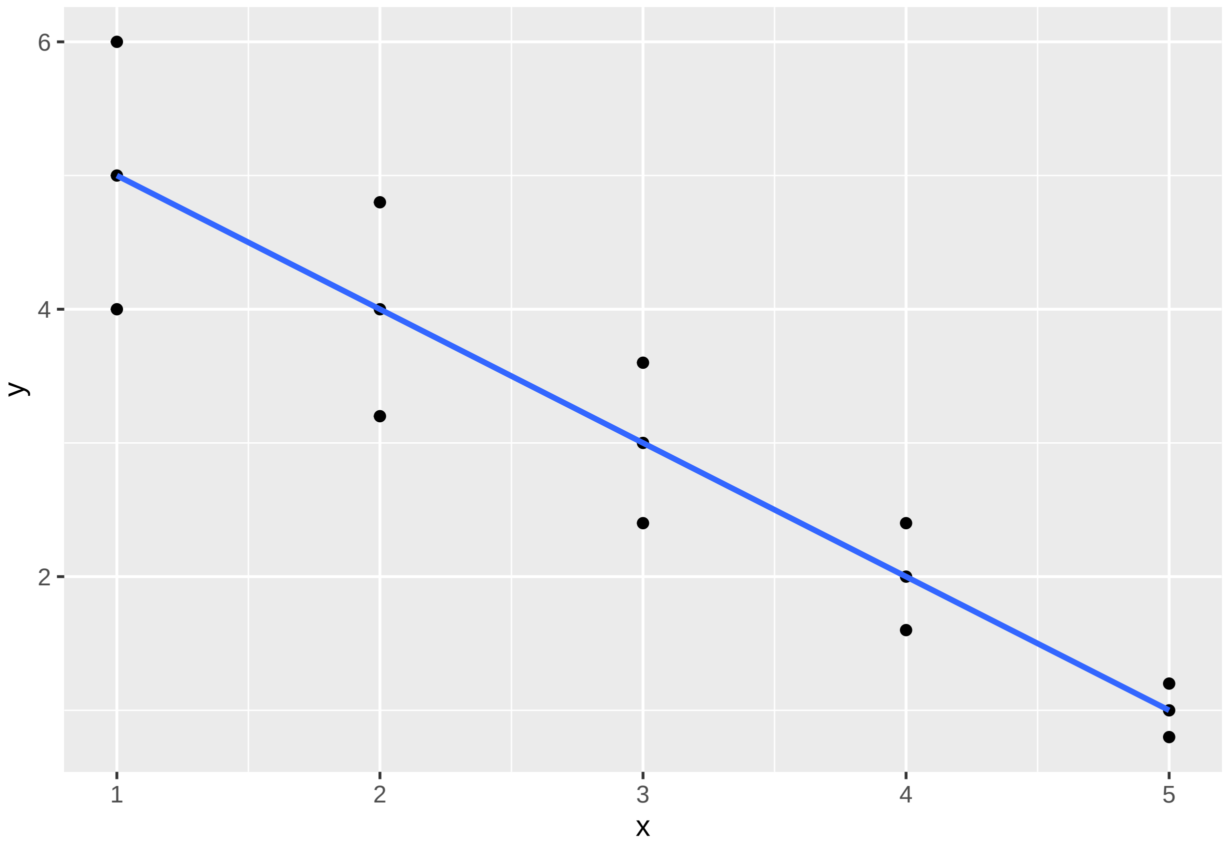
Shay Lebovitz

1/29/2021

3.2

```
x <- c(1, 1, 1, 2, 2, 2, 3, 3, 3, 4, 4, 4, 5, 5, 5)
y <- c(0.8, 1, 1.2, 1.6, 2, 2.4, 2.4, 3, 3.6, 3.2, 4, 4.8, 4, 5, 6)
y <- rev(y)
xy <- tibble(x, y)
ggplot(data = xy, aes(x, y)) +
  geom_point() +
  geom_smooth(method = 'lm', se = F)
```

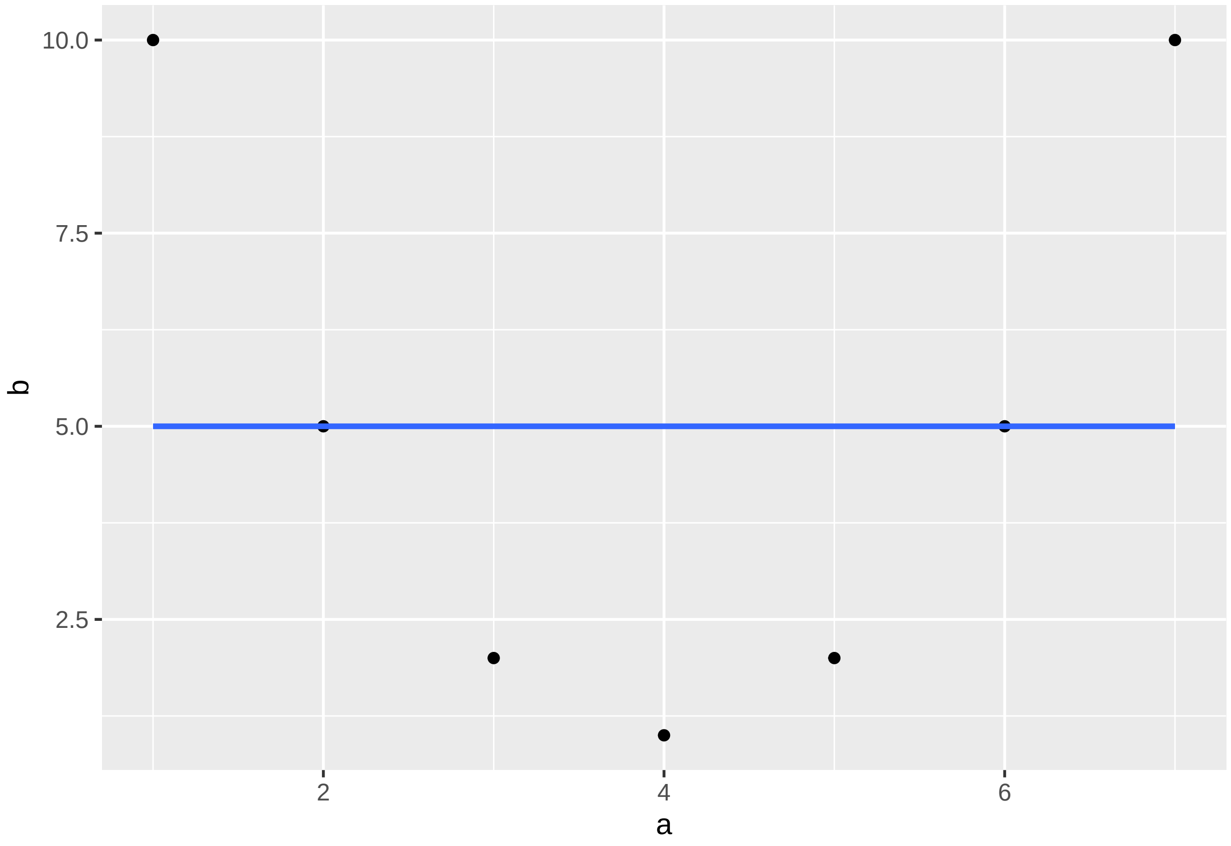
```
## `geom_smooth()` using formula 'y ~ x'
```



```
a <- c(1, 2, 3, 4, 5, 6, 7)
b <- c(10, 5, 2, 1, 2, 5, 10)
ab <- tibble(a, b)
```

```r
ggplot(data = ab, aes(a, b)) +
  geom_point() +
  geom_smooth(method = 'lm', se = F)
```

```
## `geom_smooth()` using formula 'y ~ x'
```

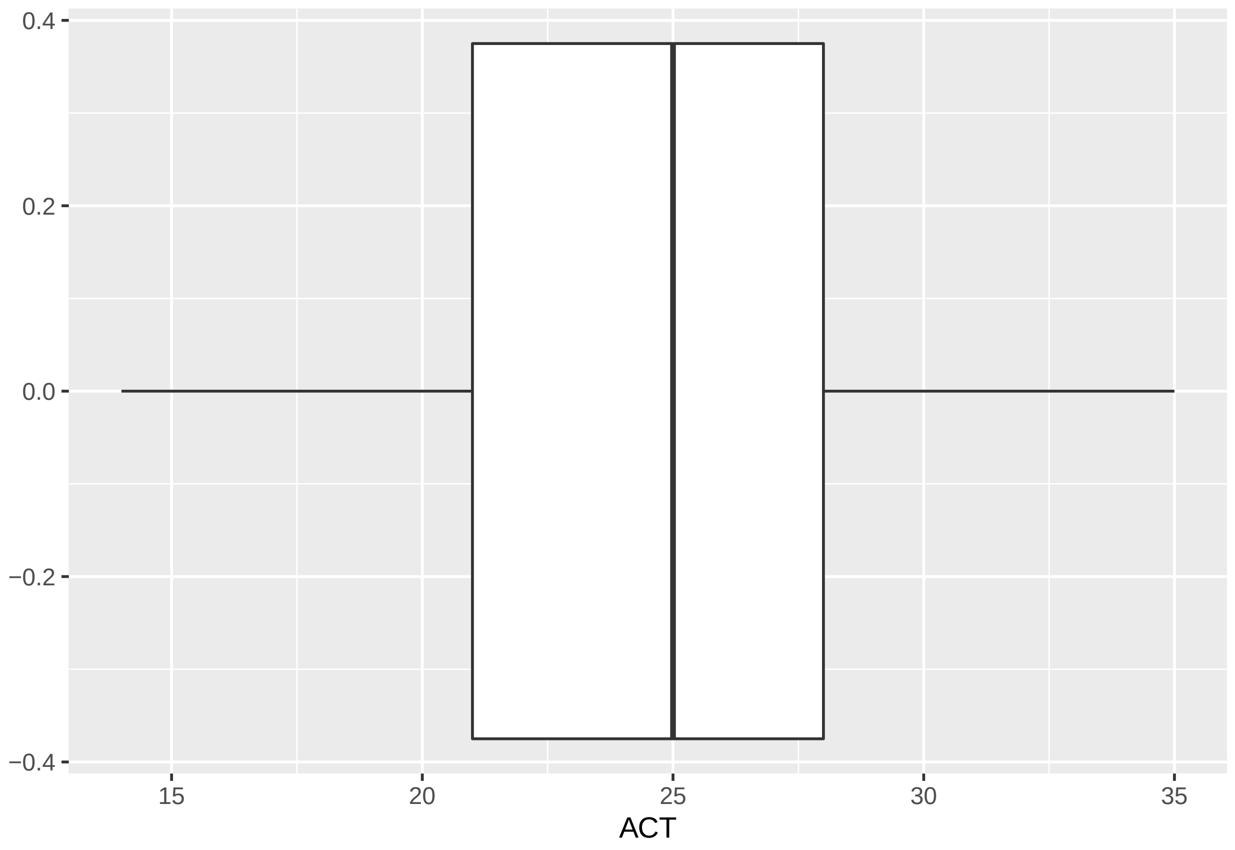


3.3

```r
gpa <- read.table('data/CH01PR19.txt')
gpa <- tibble(gpa)
gpa <- gpa %>%
  rename(ACT = V2, GPA = V1)
```

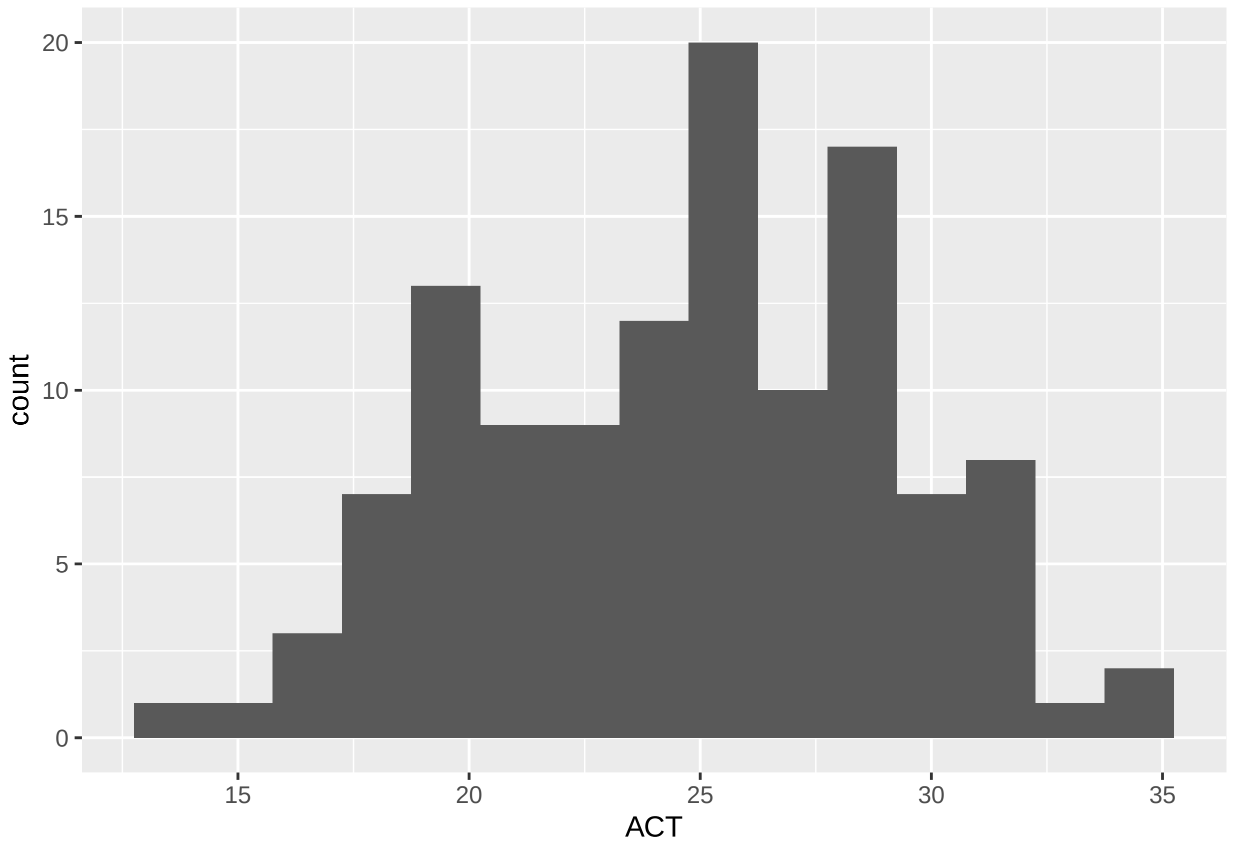
a)

```r
gpa %>%
  ggplot(aes(x = ACT)) +
  geom_boxplot()
```



I wouldn't say there are any noteworthy features of this plot. It looks fairly evenly distributed. The median is slightly closer to the third quartile, suggesting the data is very slightly left-skewed, but nothing to worry about for regression.

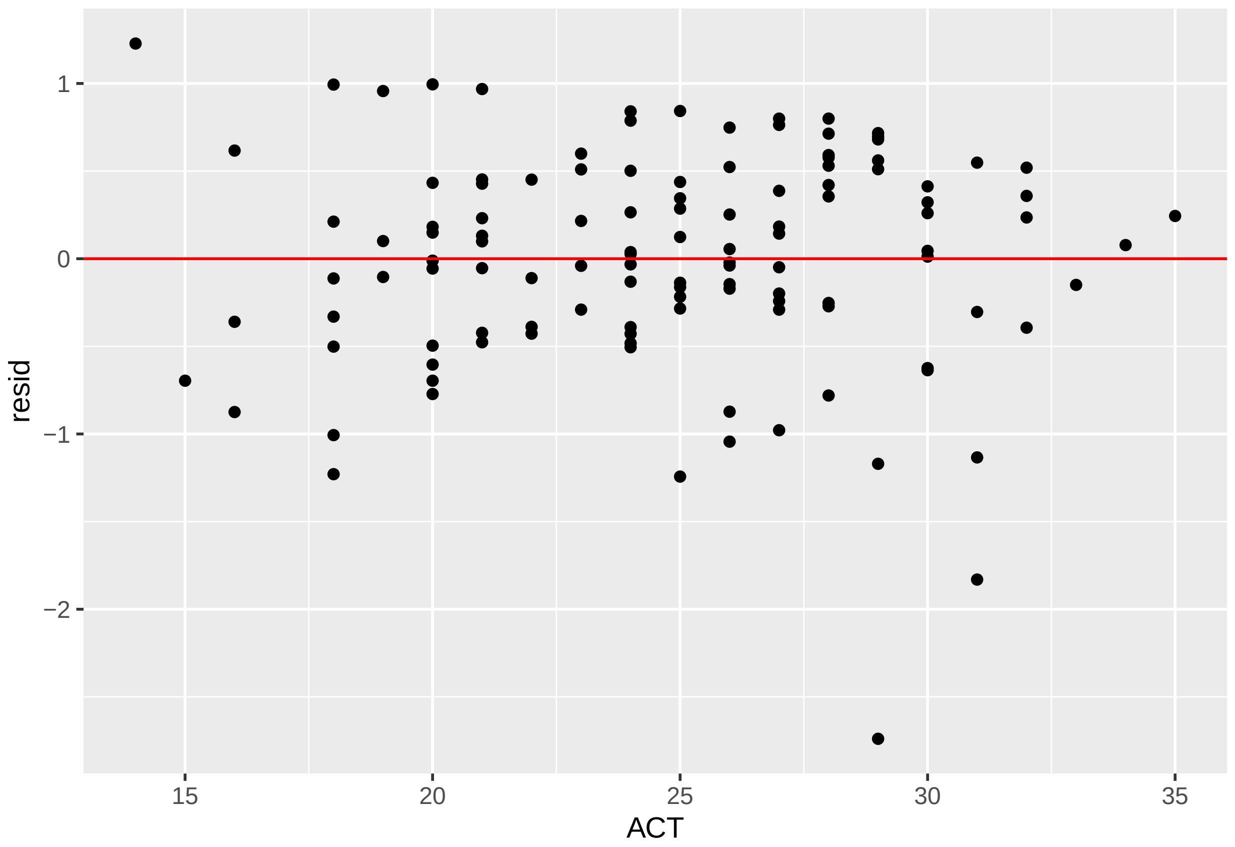
```
gpa %>%
  ggplot(aes(x = ACT)) +
  geom_histogram(bins = 15)
```



We see the slight left skew from the histogram. Overall, the data is approximately normal.

b)

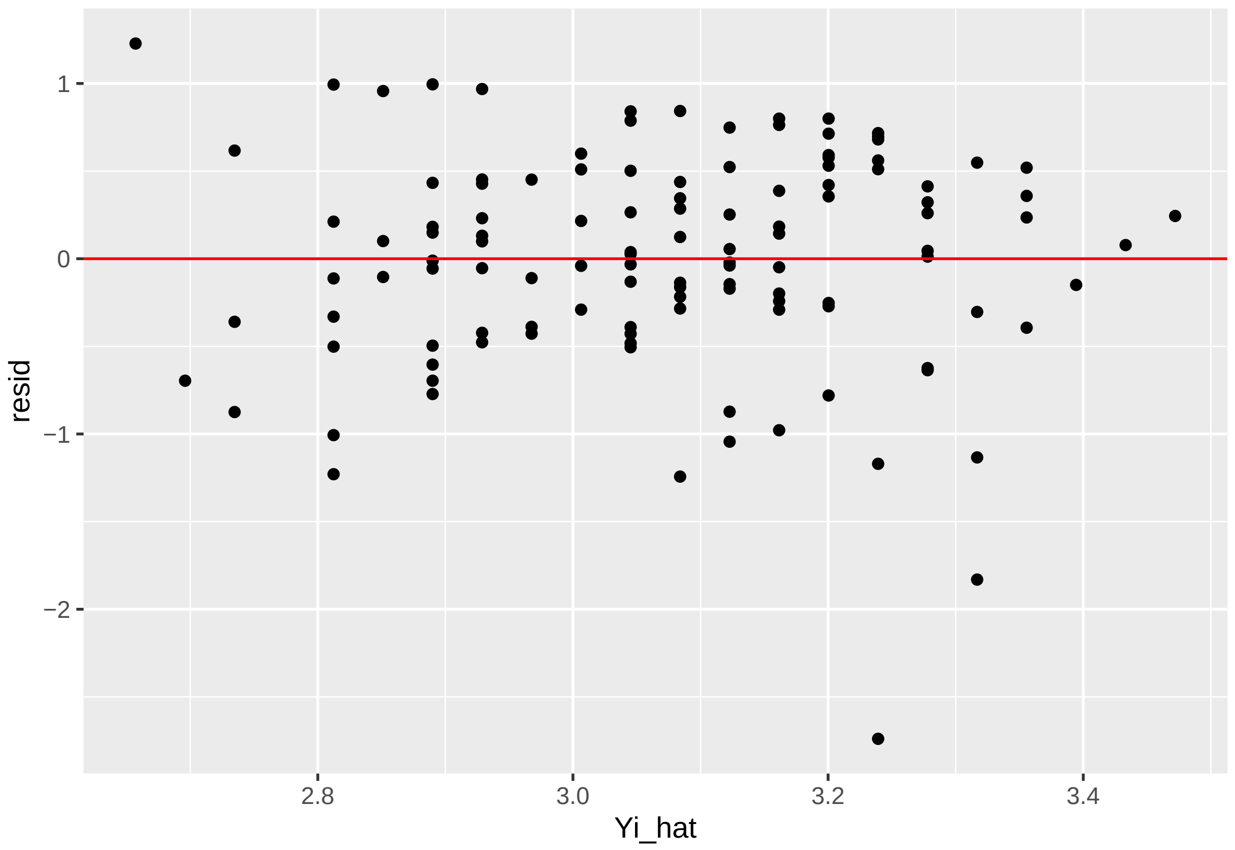
```
gpa_fit <- lm(GPA ~ ACT, data = gpa)

gpa <- gpa %>%
  mutate(Yi_hat = 2.114 + 0.0388 * ACT) %>%
  mutate(resid = GPA - Yi_hat)

gpa %>%
  ggplot(aes(x = ACT, y = resid)) +
  geom_point() +
  geom_abline(slope = 0, intercept = 0, color = 'red')
```



The residuals don't show any obvious pattern, and thus we can assume they are uncorrelated with ACT, as required for linear regression. There seems to be a few outliers on the negative side around `ACT = 30`. This causes the variance of the residuals to change with X. However, no clear pattern as to how the variance changes can be made.

c)

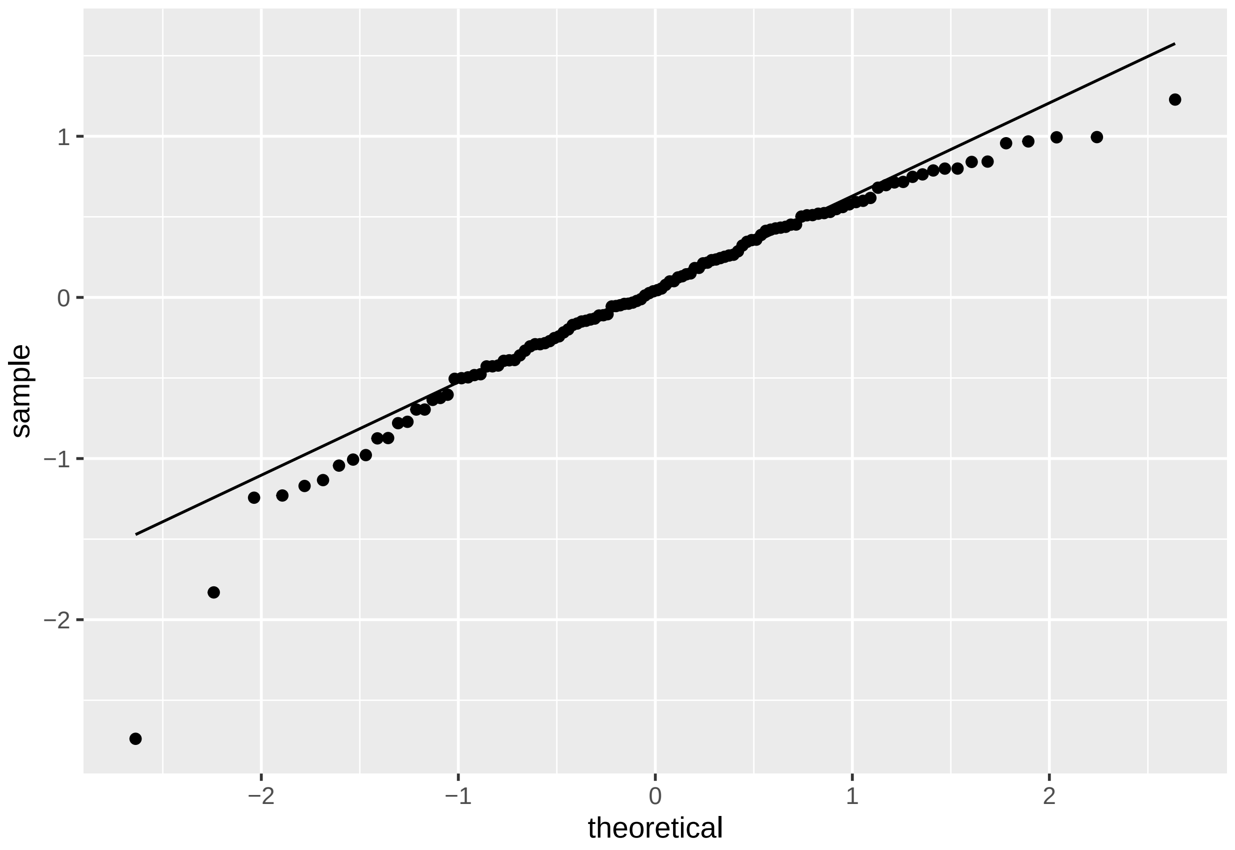
```
gpa %>%
  ggplot(aes(x = Yi_hat, y = resid)) +
  geom_point() +
  geom_abline(slope = 0, intercept = 0, color = 'red')
```



This is the exact same plot as shown in (b), and thus the exact same analysis can be carried out.

d)

```
gpa %>%
  ggplot(aes(sample = resid)) +
  geom_qq() +
  geom_qq_line()
```



We see lower than expected residuals at the edges, suggesting the residuals are left-skewed.

```
res <- gpa_fit$residuals
MSE <- anova(gpa_fit)[2,3]

# sort the residuals from the smallest to largest
res.sorted <- sort(res)
n <- 120
seq <- c(1:n)
Percentile <- (seq - 0.375)/(n + 0.25)
res.expected <- sqrt(MSE) * qnorm(Percentile)
cor(res.sorted, res.expected)
```

```
## [1] 0.9737275
```

Based on table B.6 and alpha = 0.05, the critical value is 0.987 based on n = 120. With a correlation of 0.973, we cannot conclude that the error terms are reasonably normally distributed.

e)

```
#Step 1. Break the residuals into two groups.
Group1 <- res[gpa$ACT < 26]
Group2 <- res[gpa$ACT >= 26]

#Step 2. Obtain the median of each group, using the commands:
M1 <- median(Group1)
M2 <- median(Group2)
```

```r
#Step 3-6. can be done by doing two-sample t-test
t.test(abs(Group1-M1), abs(Group2-M2), var.equal = TRUE)
```

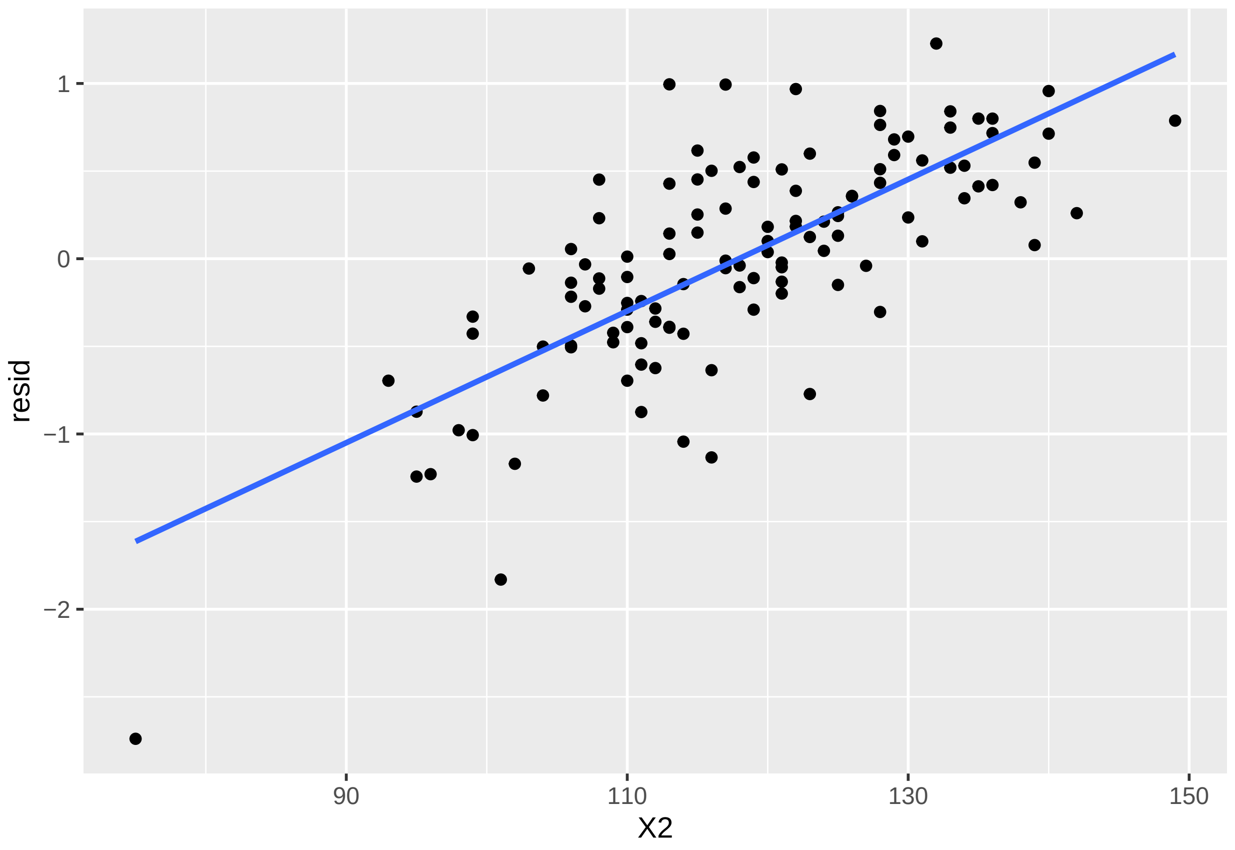
```
##
##  Two Sample t-test
##
## data:  abs(Group1 - M1) and abs(Group2 - M2)
## t = -0.89674, df = 118, p-value = 0.3717
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.21994668  0.08283519
## sample estimates:
## mean of x mean of y
## 0.4379603 0.5065161
```

We get a test statistic of $t = -0.8967$ on 118 degrees of freedom. With a critical value of $t^* = 1.98$, we can conclude that this is no difference in residual variance between the two groups, and thus error variance does not vary with X. In (c), it looked like variance did vary a little across X, but the t-test shows that it does not vary enough to be statistically significant.

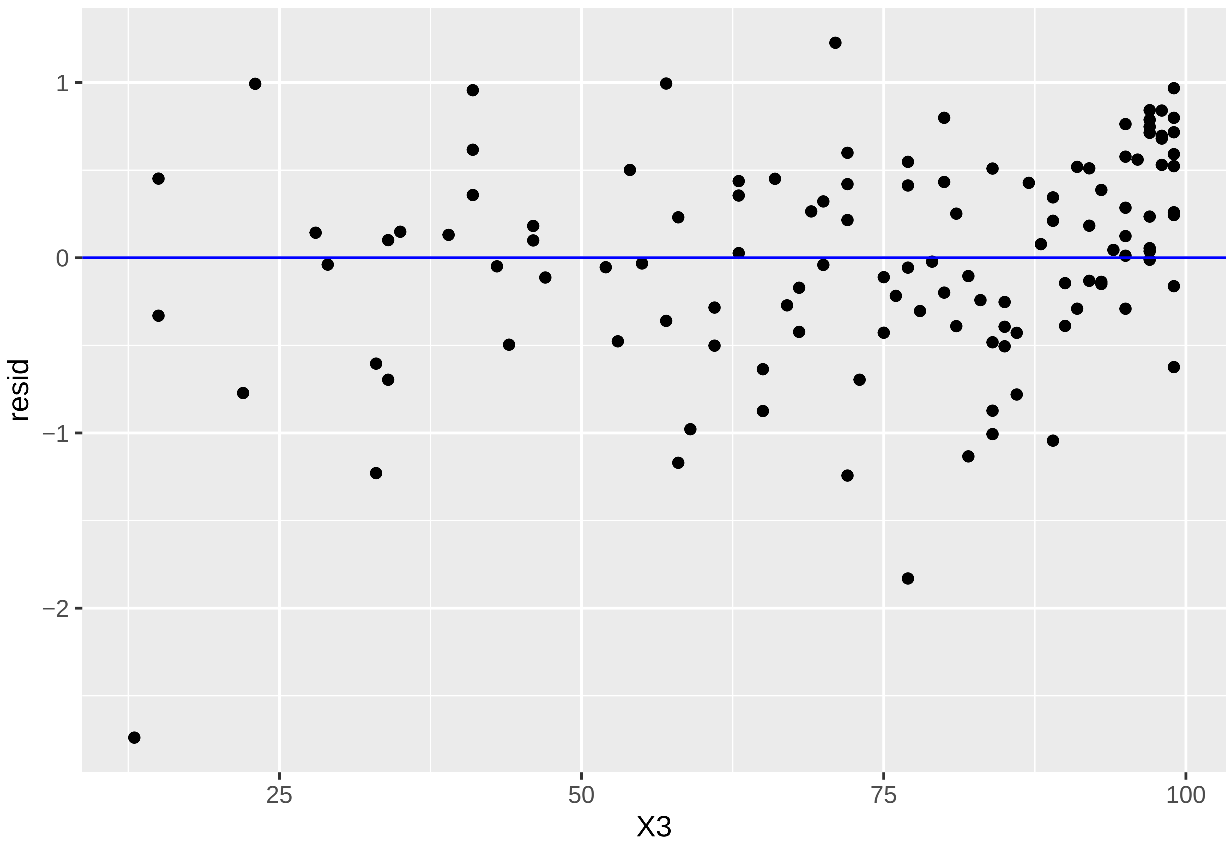
f)

```r
gpa_plus <- read.table('data/CH03PR03.txt', header = F)
gpa_plus <- tibble(gpa_plus)
gpa_plus <- gpa_plus %>%
  rename(ACT = V2, GPA = V1, X2 = V3, X3 = V4)

gpa_plus <- gpa_plus %>%
  mutate(Yi_hat = 2.114 + 0.0388 * ACT) %>%
  mutate(resid = GPA - Yi_hat)

par(mfrow = c(1, 2))
ggplot(data = gpa_plus, aes(x = X2, y = resid)) +
  geom_point() +
  geom_smooth(method = 'lm', se = F)
```

```
## `geom_smooth()` using formula 'y ~ x'
```



```r
ggplot(data = gpa_plus, aes(x = X3, y = resid)) +
  geom_point() +
  geom_abline(slope = 0, intercept = 0, color = 'blue')
```



There is a clear association of the residuals with X2, or intelligence test score. Thus, leaving this variable out causes omitted variable bias. There does not seem to be any relation between residuals and X3, so leaving this variable out would be ok. To conclude whether including either or both of these variables is important, one could conduct an F test of the full (including X2 and/or X3) vs. reduced models (excluding X2 and X3).

3.15

a)

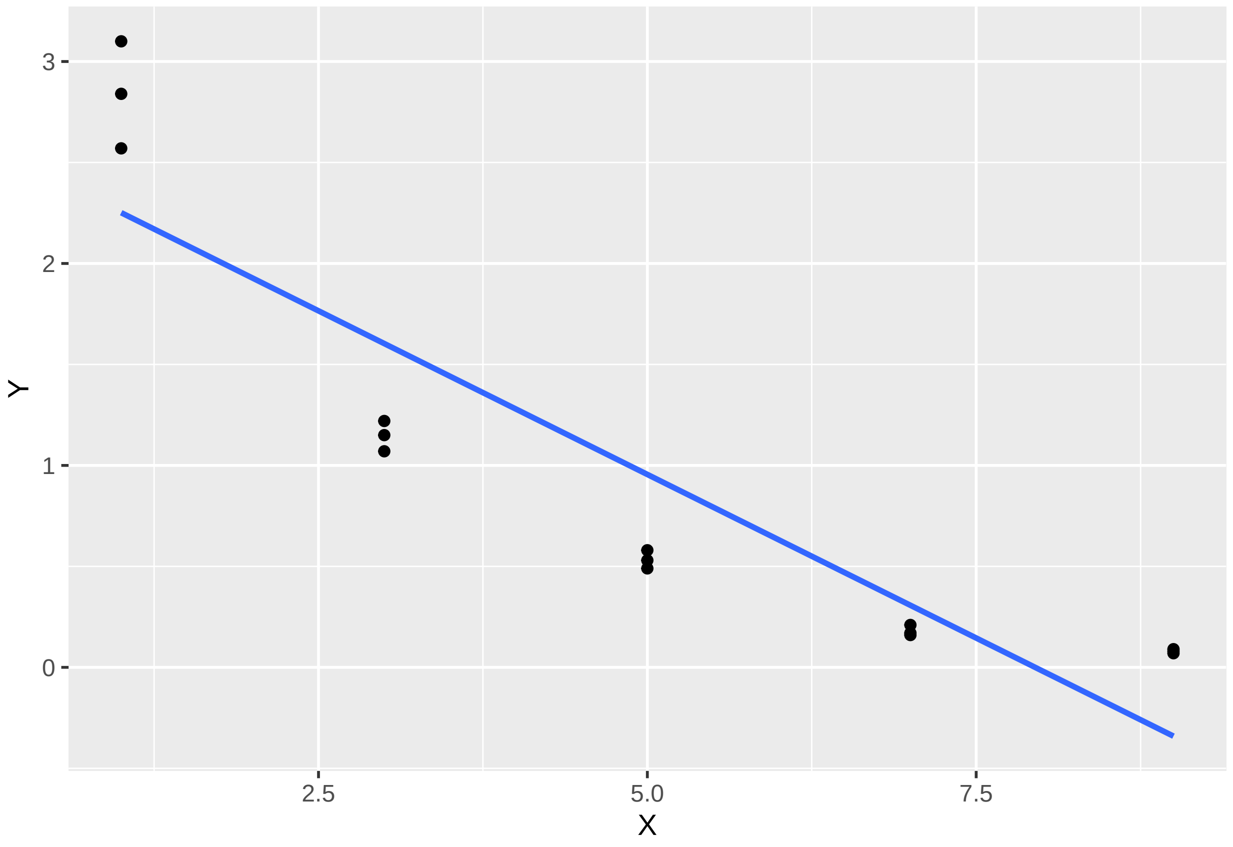
```
sol <- read.table('data/CH03PR15.txt', header = F)
sol <- tibble(sol)
sol <- sol %>%
  rename(X = V2, Y = V1)

sol_fit <- lm(Y ~ X, data = sol)
summary(sol_fit)
```

```
##
## Call:
## lm(formula = Y ~ X, data = sol)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.5333 -0.4043 -0.1373  0.4157  0.8487
##
```

```
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept)   2.5753     0.2487  10.354 1.20e-07 ***
## X            -0.3240     0.0433  -7.483 4.61e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4743 on 13 degrees of freedom
## Multiple R-squared:  0.8116, Adjusted R-squared:  0.7971 
## F-statistic: 55.99 on 1 and 13 DF,  p-value: 4.611e-06
```

```
ggplot (data = sol, aes(X, Y)) +
  geom_point() +
  geom_smooth(method = 'lm', se = F)
```

```
## `geom_smooth()` using formula 'y ~ x'
```



b)

```
anova(sol_fit)
```

```
## Analysis of Variance Table
##
## Response: Y
##           Df  Sum Sq Mean Sq F value    Pr(>F)    
## X          1 12.5971  12.597  55.994 4.611e-06 ***
## Residuals 13  2.9247   0.225
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

$H_0 : E[Y] = B_0 + B_1 * X$ vs. $H_a : E[Y \neq B_0 + B_1 * X$

$c = 5, n_j = 3, j = 1, ..., 5$

$df_F = n - c = 15 - 5 = 10$, $df_R = n - 2 = 15 - 3 = 13$

$F^* = (SSLF/(c - 2))/(SSPE/(n - c)$

```
sol <- sol %>%
  group_by(X) %>%
  mutate(pure_error = Y - mean(Y))
SSPE <- sum((sol$pure_error)^2)
SSFL <- 2.9247 - SSPE
f_star <- (SSFL/3)/(SSPE/10)
```

$F(0.975, 3, 10) = 4.8256$

$F^* = 58.6044 > 4.8256$

Therefore we reject the null hypothesis in favor of the alternative hypothesis, and conclude that the regression function is not linear. By eye, the function looks to be on the order of $1/X$.

c)

The F test performed in (b) only indicates that there is no *linear* association. However, it looks like the association between Y and X follows a $1/X$ pattern. I do not believe that the test in (b) will tell us which regression function is most appropriate, however.

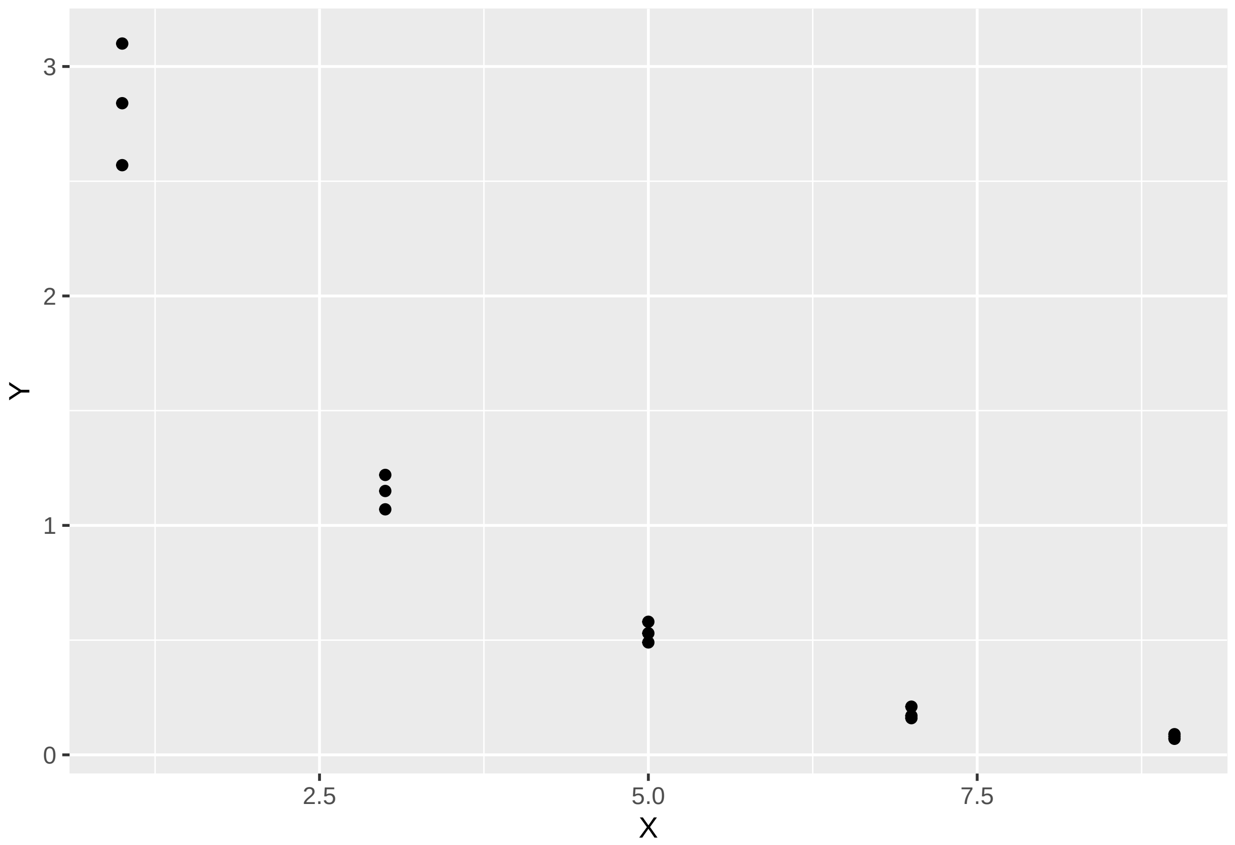
## 3.16

a)

#3.16)

#a)

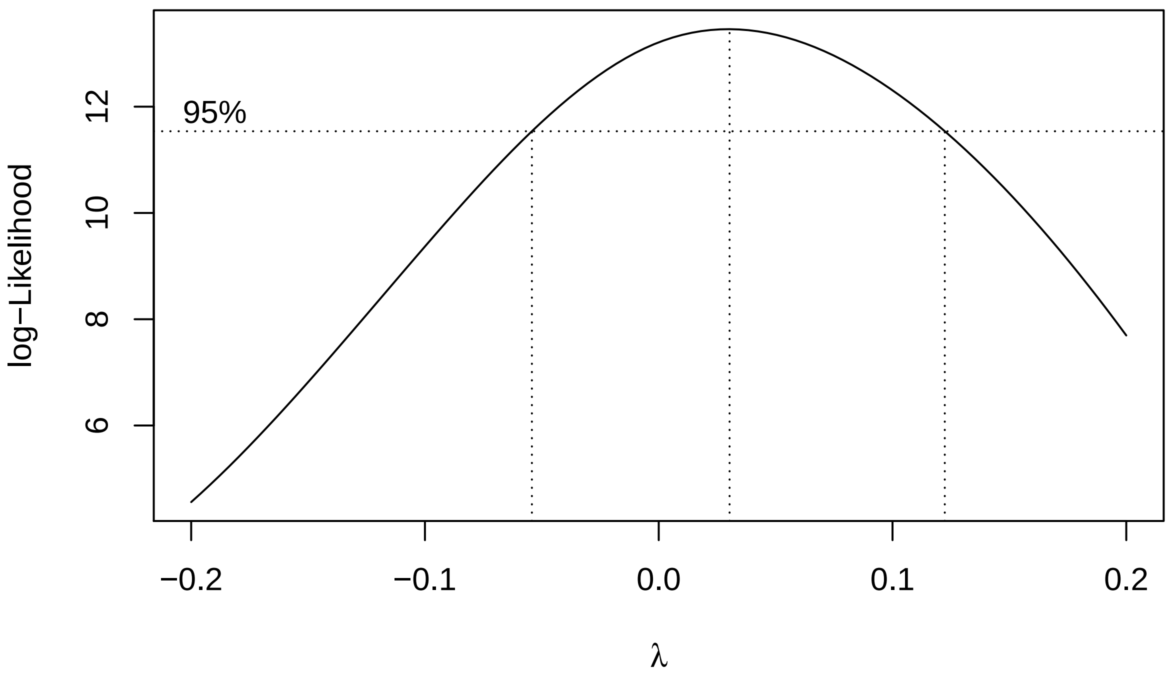
```
ggplot (data = sol, aes(X, Y)) +
  geom_point()
```



We see that the variance of the error terms decreases with increasing X. Therefore, a transformation of X will not be appropriate, because it won't affect the variance of error terms. Instead, we must transform Y. I believe $Y' = log_{10}Y$ will be appropriate.

b)

```
my_boxcox <- boxcox(sol_fit, lambda = seq(-0.2, 0.2, 0.1))
```



```
lambda <- my_boxcox$x
log_like <- my_boxcox$y
bc <- cbind(lambda, log_like)
sorted_bc <- bc[order(-log_like), ]
head(sorted_bc)
```

```
##          lambda log_like
## [1,] 0.03030303 13.45848
## [2,] 0.02626263 13.45511
## [3,] 0.03434343 13.45323
## [4,] 0.02222222 13.44293
## [5,] 0.03838384 13.43953
## [6,] 0.01818182 13.42179
```

We get the highest log-likelihood and thus lowest SSE at lambda = 0.03. Thus a transformation of $Y' = Y^{0.03}$ is suggested. However, we see that 0 is in the 95% confidence interval, and will be a much easier number to work with. Thus, I suggest that the transformation should be $Y' = ln(Y)$.

c)

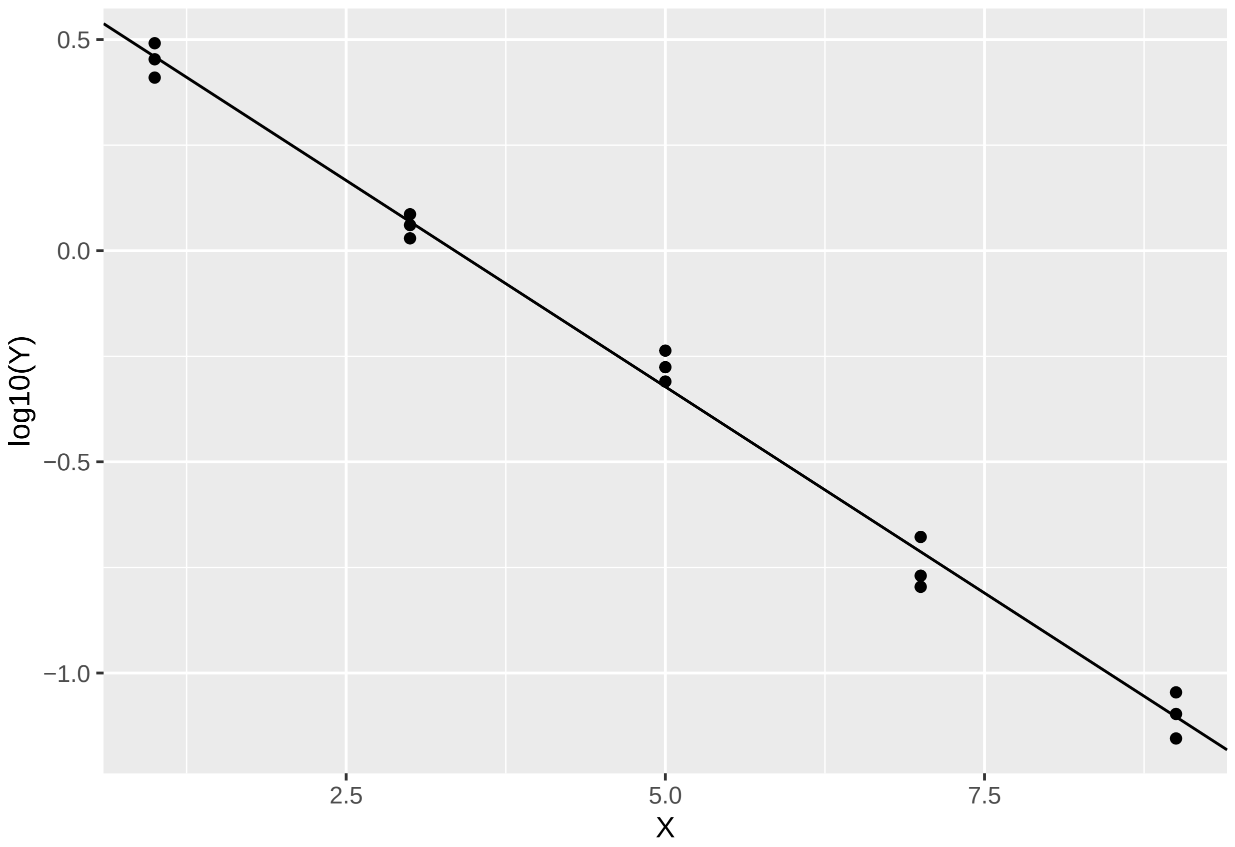
```
sol_fit_log <- lm(log10(Y) ~ X, data = sol)
summary(sol_fit_log)
```

```
##
## Call:
## lm(formula = log10(Y) ~ X, data = sol)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.082958 -0.044421  0.006813  0.033512  0.085550
##
```

```
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)    
## (Intercept)  0.654880   0.026181   25.01 2.22e-12 ***
## X           -0.195400   0.004557  -42.88 2.19e-15 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 0.04992 on 13 degrees of freedom
## Multiple R-squared:  0.993,  Adjusted R-squared:  0.9924 
## F-statistic:  1838 on 1 and 13 DF,  p-value: 2.188e-15
```

d)

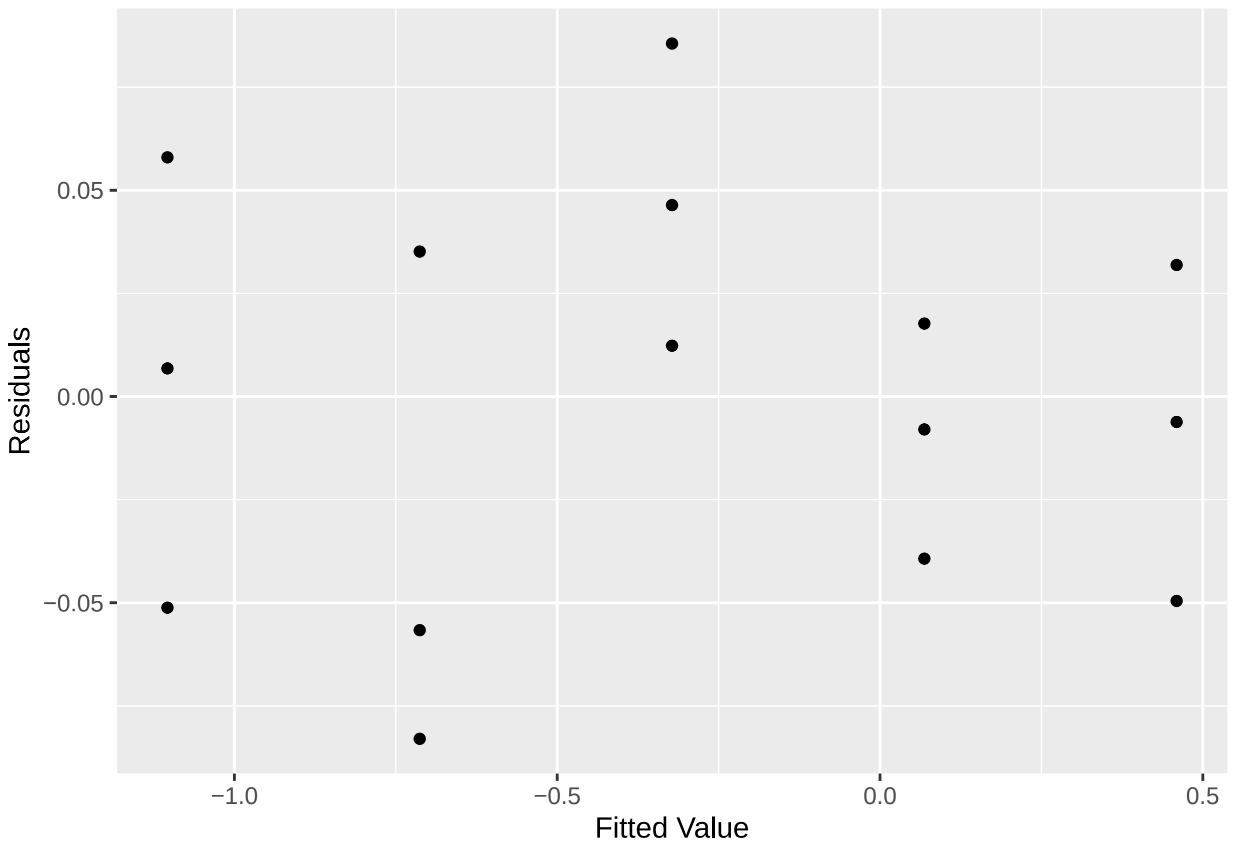
```r
ggplot(data = sol) +
  geom_point(aes(X, log10(Y))) +
  geom_abline(slope = sol_fit_log$coeff[2], intercept = sol_fit_log$coeff[1])
```



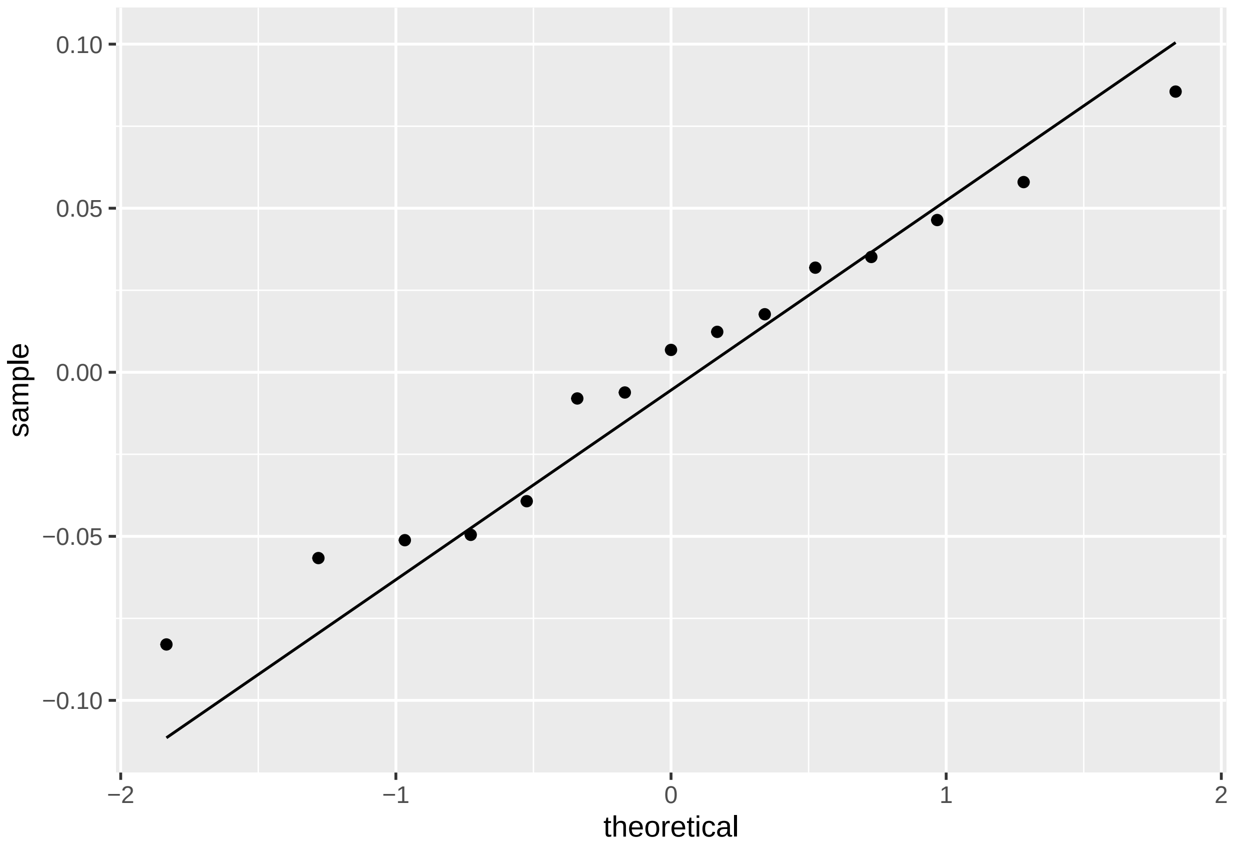
Yes, the log transformed Y variable regression does look to be a great fit.

e)

```r
ggplot(data = sol) +
  geom_point(aes(sol_fit_log$coeff[1] + sol_fit_log$coeff[2] * X, sol_fit_log$residuals)) +
  labs(x = 'Fitted Value', y = 'Residuals')
```



```
sol %>%
  ggplot(aes(sample = sol_fit_log$residuals)) +
  geom_qq() +
  geom_qq_line()
```



We see that the log10 transformation of Y not only achieves constant and normal variance of error terms, but it also linearizes the data almost perfectly. The normal probability plot may suggest heavy tails similar to a t-distribution. Overall, the residuals appear to be more normalized than the original data, and thus the transformation can be deemed a success.

f)

To get back to the original units, just exponentiate both sides:

$y = 10^{0.6549-0.1954*X}$