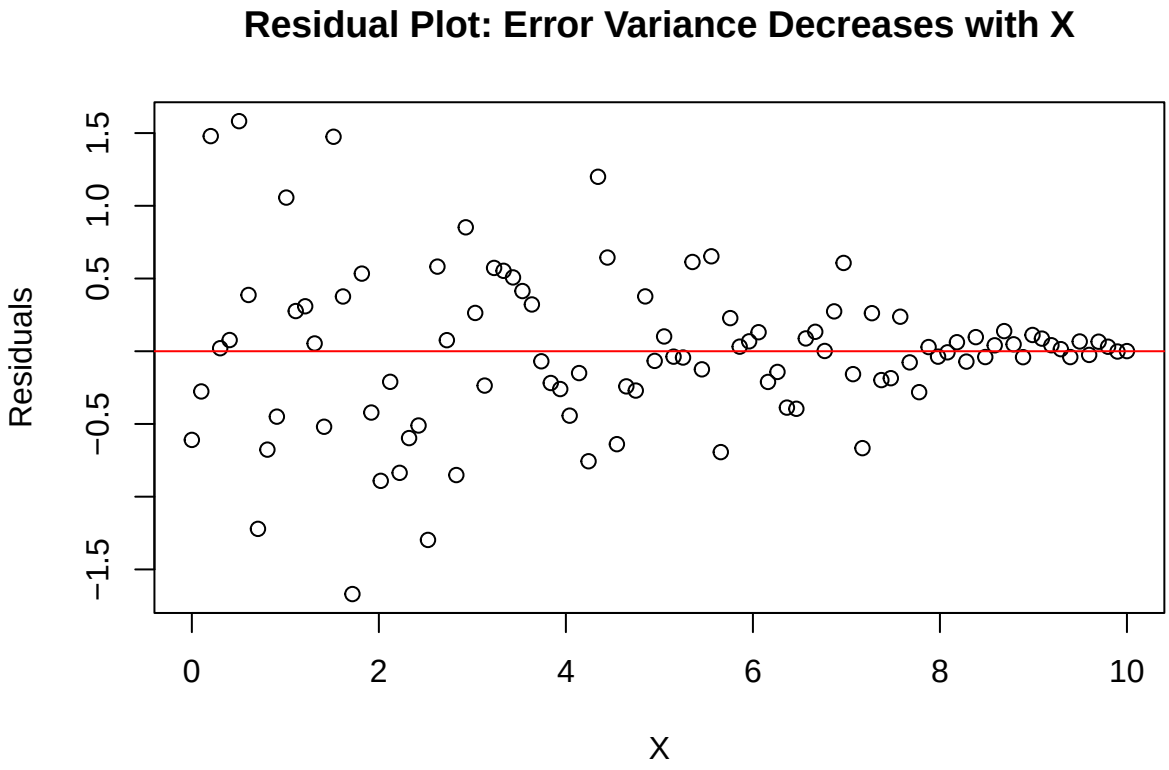
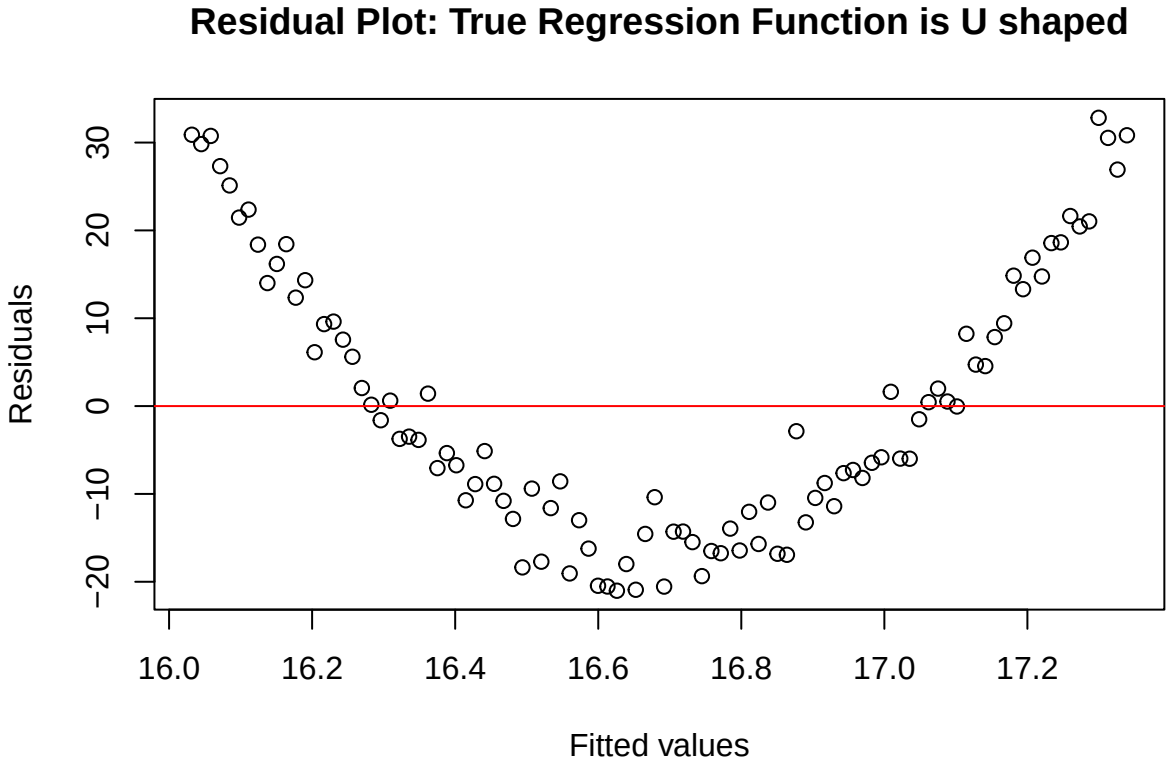
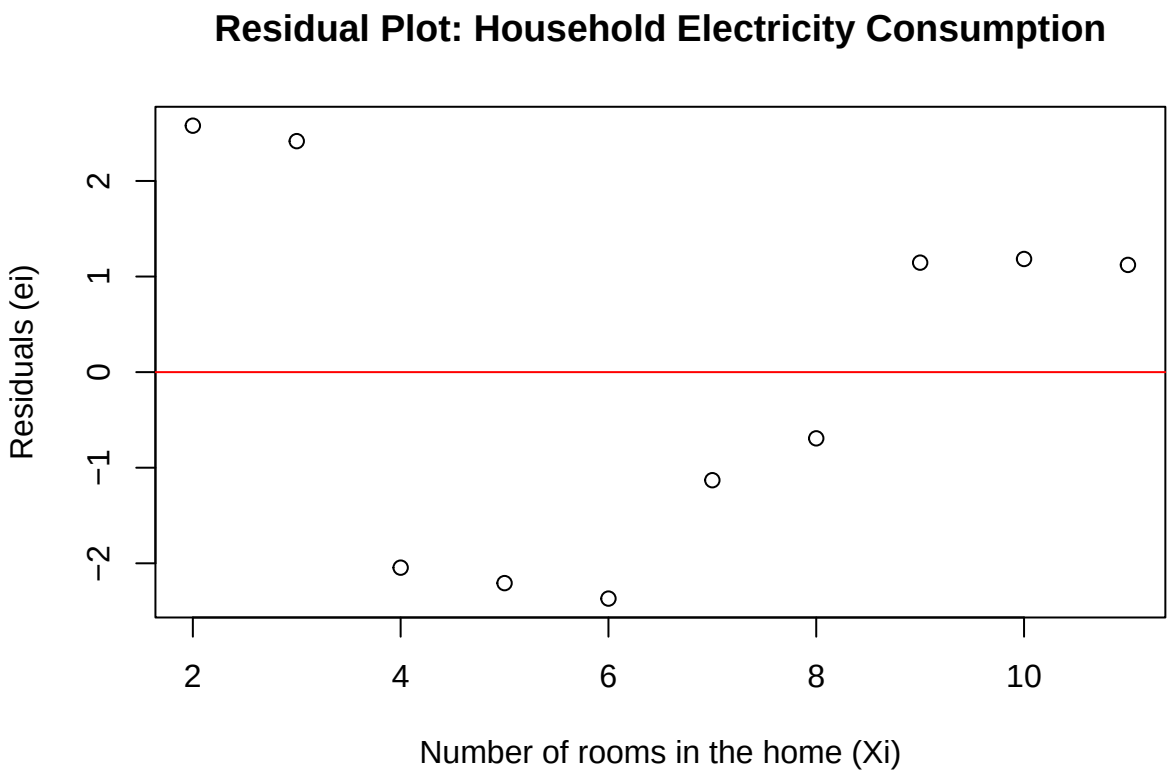
3.2


Residual Plot: Error Variance Decreases with X
Residuals
−1.5
−0.5
0.5
1.0
1.5
0
2
4
6
8
10
X


Residual Plot: True Regression Function is U shaped
Residuals
−20
−10
0
10
20
30
16.0
16.2
16.4
16.6
16.8
17.0
17.2
Fitted values

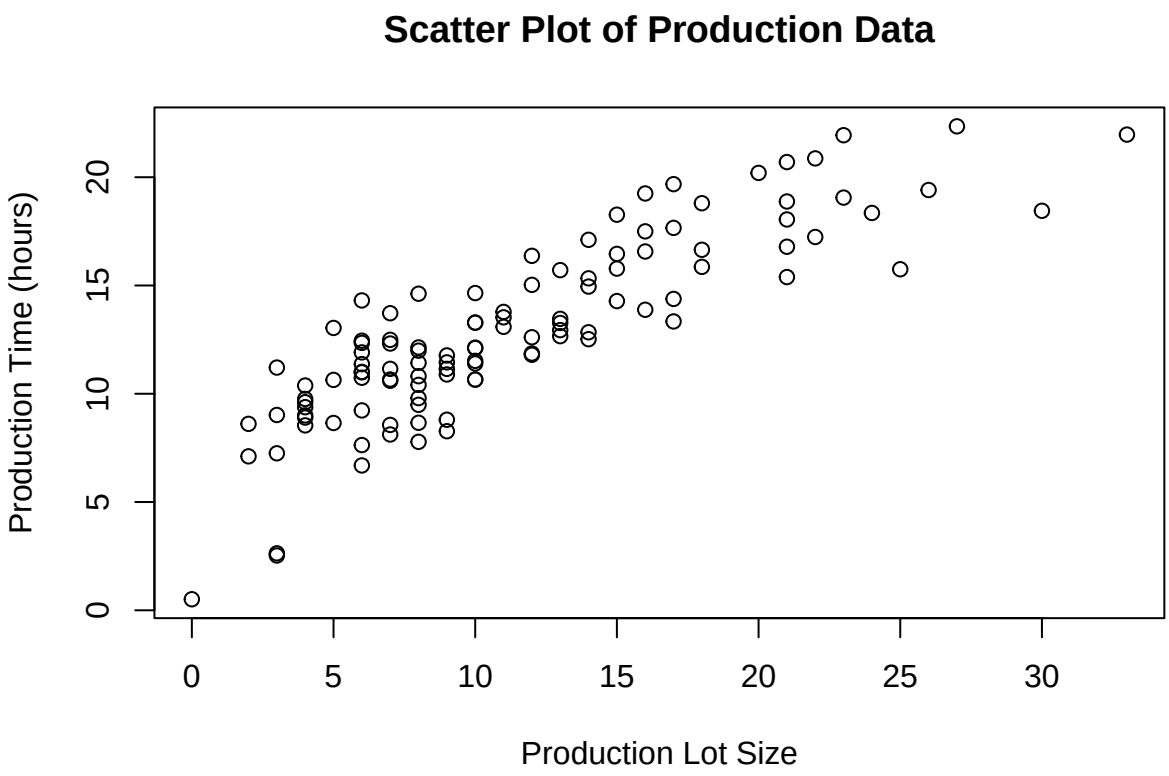
3.9

Based on the plot, there does not seem to be a linear relationship between the datapoints. A transformation might alleviate the problem, as it can linearize non-linear relationships.



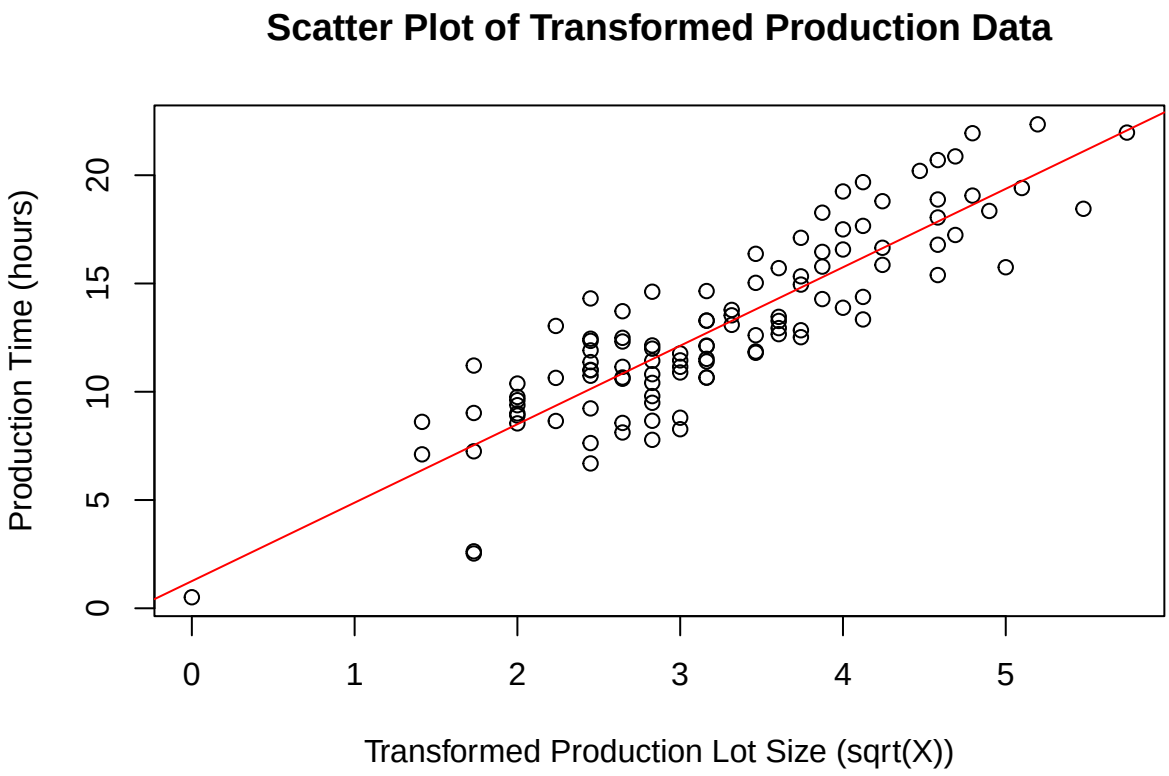
3.18

a. A linear relationship on the original data is not adequate because the relationship does not look linear but rather square root. We would like to do a transformation on X so that ϵ_{ij} is still normally distributed afterwards.

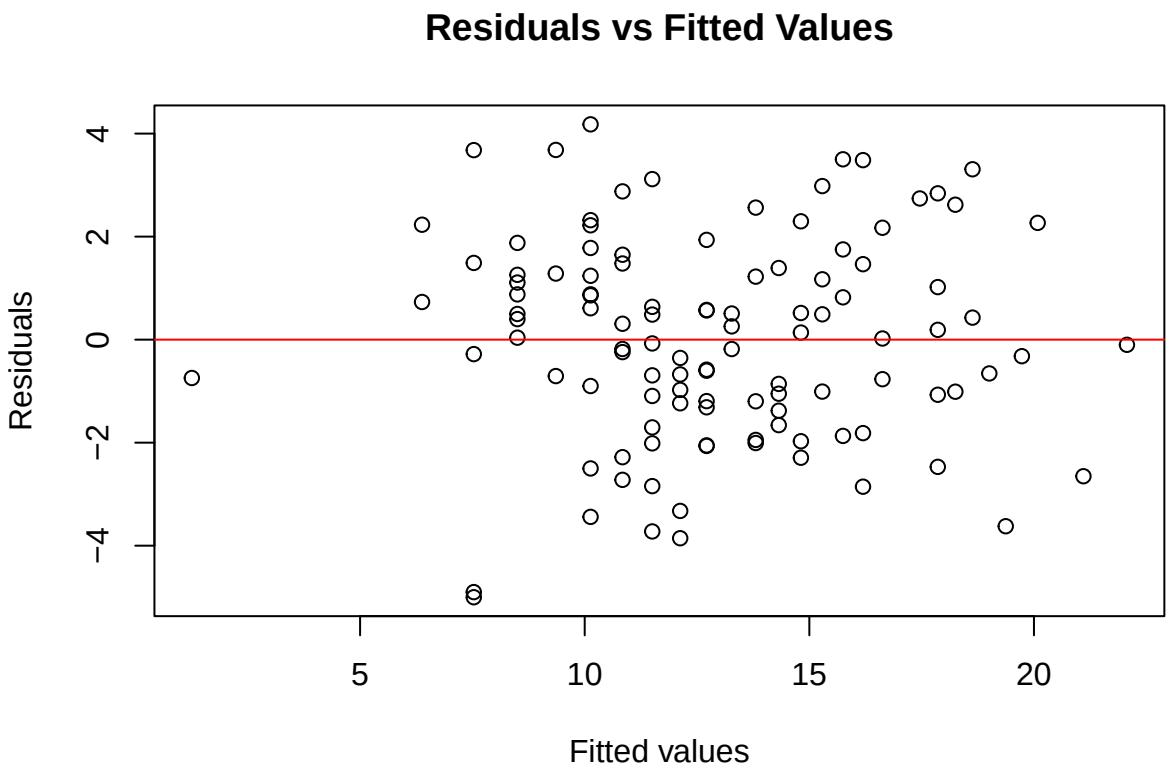


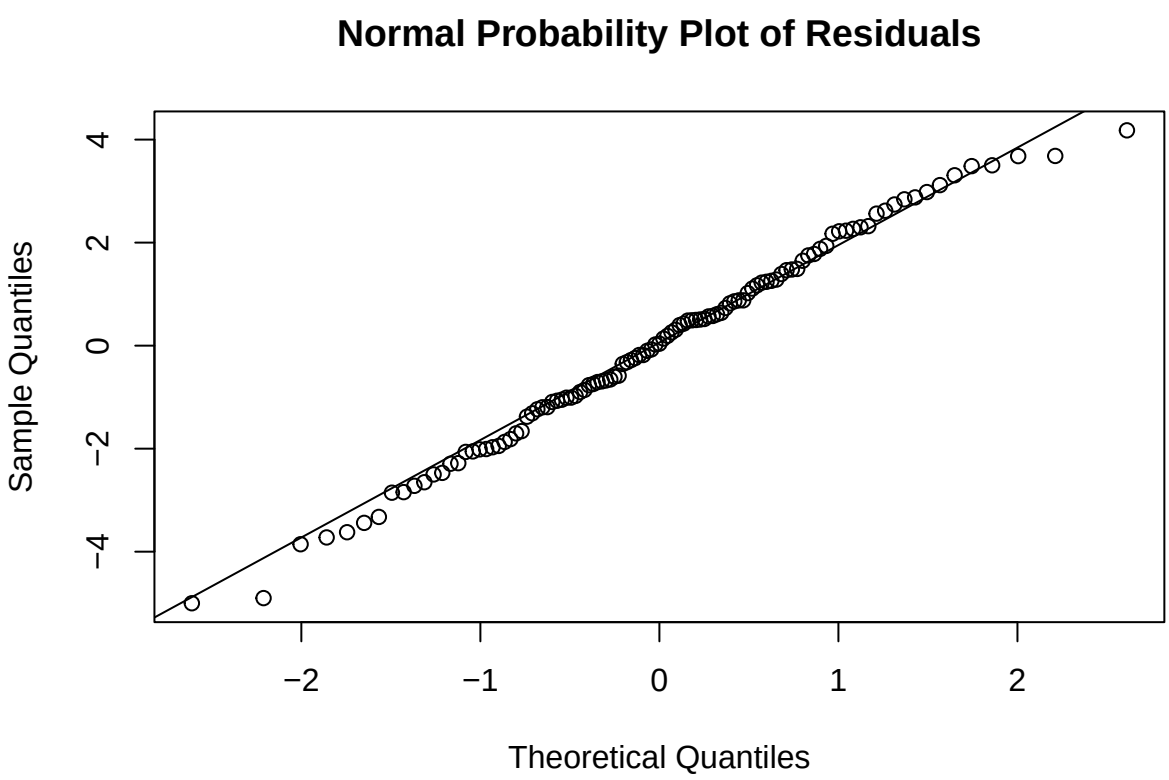
b. $\hat{Y}_i = 1.2547 + 3.6235x_i'$

c. The regression line appears to be a good fit of the transformed data.



d. My plots show that there is constant variance (because the dots in the residuals plot are spread out approximately equally for different fitted values) and the distribution is normal (because the dots in the normality plot conform to the line).


Residuals vs Fitted Values
Residuals
4
2
0
−2
−4
5
10
15
20
Fitted values



e. $\hat{Y}_i = 1.2547 + 3.6235\sqrt{x_i}$

3.19

This difference can arise because $\hat{Y}_i = b_0 + b_1 x_i$ does not depend on ϵ_i as it is the predictor, while $Y_i = \beta_0 + \beta_1 x_i + \epsilon_i$ depends on ϵ_i. The residual is dependent on ϵ_i, as this represents the variance. The plot with residual against Y_i is more meaningful and useful for making interpretations.

3.20

The transformation $X' = 1/X$ maintains a linear regression relationship, so the error terms are still independent and $\epsilon_{ij} \sim N(0, \sigma^2)$. However, the same does not hold after the transformation $Y' = 1/Y$, as this transformation modifies the error values.

3.23

The full model is $Y_{ij} = \mu_j + \epsilon_{ij}$

$df_F = n - c = 20 - 10 = 10$

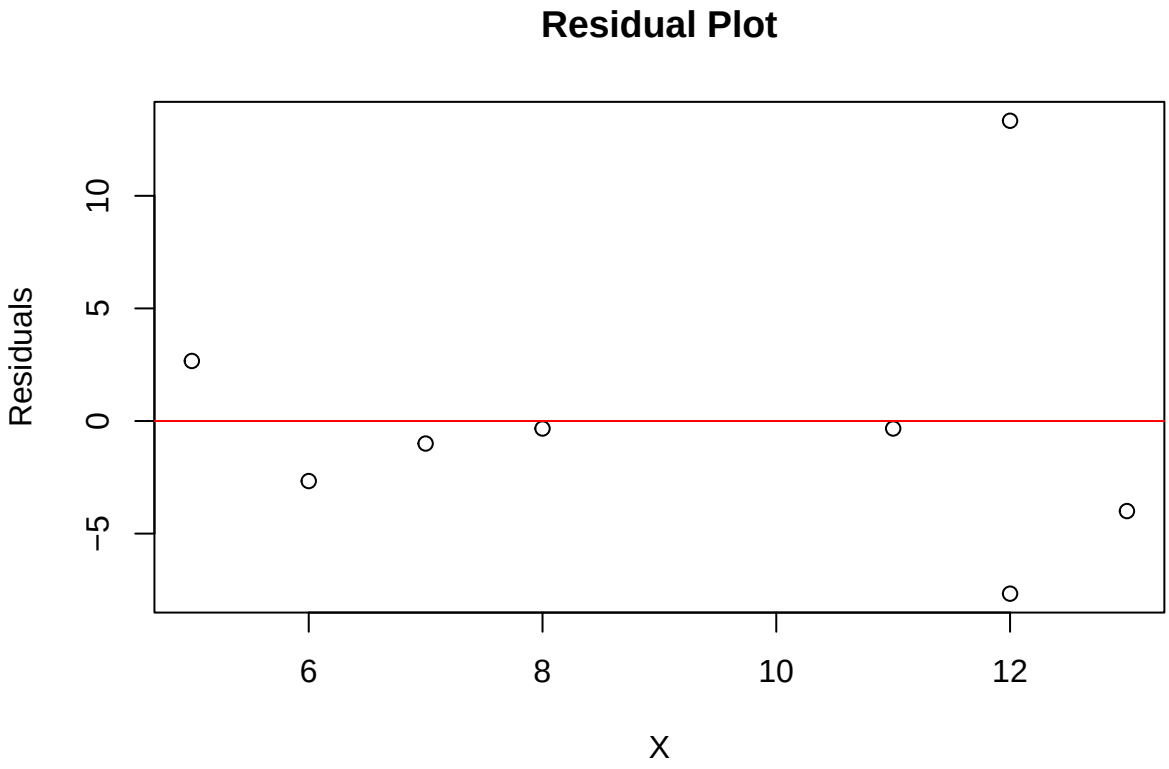
The reduced model is $Y_{ij} = \beta_1 x_j + \epsilon_{ij}$

$df_R = n - 2 = 20 - 2 = 18$

3.24

a. $\hat{Y}_i = 48.6667 + 2.3333x_i$

The residual plot shows that there is an outlier at 12. Most residuals lie between ± 5, but this outlier is beyond 10.



b. $\hat{Y}_i = 53.068 + 1.6214x_i$

We can conclude that this outlier had a huge effect on the estimated coefficients of our model.

c. $\hat{Y}_i(x_i = 12) \in [60.3127, 84.7359]$

The observation Y_7 falls outside of the prediction interval. This means that Y_7 is an outlier.

Appendix

3.2

```
# Case 1: Error variance decreases with X
# Generate synthetic data
set.seed(123)
X <- seq(0, 10, length.out = 100)
Y <- 2*X + rnorm(100, mean = 0, sd = 1 - X/10)  # Decreasing error variance
```

```r
# Fit linear regression model
model <- lm(Y ~ X)

# Get residuals
residuals <- residuals(model)

# Plot residuals against X
plot(X, residuals, main = "Residual Plot: Error Variance Decreases with X",
     xlab = "X", ylab = "Residuals")
abline(h = 0, col = "red")

# Case 2: True regression function is U shaped
# but a linear regression function is fitted
# Generate synthetic data
X2 <- seq(-5, 5, length.out = 100)
# True regression function is U shaped
Y2 <- 2*X2^2 + rnorm(100, mean = 0, sd = 3)

# Fit linear regression model
model2 <- lm(Y2 ~ X2)

# Residuals for Case 2
residuals2 <- resid(model2)

# Plot residual vs. fitted values
plot(fitted(model2), residuals2, xlab = "Fitted values", ylab = "Residuals",
     main = "Residual Plot: True Regression Function is U shaped")

# Add a horizontal line at y = 0 for reference
abline(h = 0, col = "red")
```

3.9

```r
# Provided data
x_i <- c(2, 3, 4, 5, 6, 7, 8, 9, 10, 11)
e_i <- c(3.2, 2.9, -1.7, -2.0, -2.3, -1.2, -0.9, 0.8, 0.7, 0.5)

# Fit linear regression model
model <- lm(e_i ~ x_i)

# Get residuals
residuals <- resid(model)

# Plot residuals against Xi
plot(x_i, residuals, xlab = "Number of rooms in the home (Xi)",
     ylab = "Residuals (ei)",
     main = "Residual Plot: Household Electricity Consumption")

# Add a horizontal line at y = 0 for reference
abline(h = 0, col = "red")
```

3.18

```r
# Read the txt file into a dataframe
production_data <- read.table("CH03PR18.txt")
colnames(production_data) <- c("Y", "X")
```

3.18-a

```r
# Do plot
plot(production_data$X, production_data$Y,
     xlab = "Production Lot Size",
     ylab = "Production Time (hours)",
     main = "Scatter Plot of Production Data")
```

3.18-b

```r
# Transform the data
transformed_X <- sqrt(production_data$X)

# Perform linear regression on the transformed data
lm_result <- lm(production_data$Y ~ transformed_X)

# Extract estimated coefficients
intercept <- coef(lm_result)[1]
coef_sqrtX <- coef(lm_result)[2]
```

3.18-c

```r
# Plot the transformed data
plot(sqrt(production_data$X), production_data$Y,
     xlab = "Transformed Production Lot Size (sqrt(X))",
     ylab = "Production Time (hours)",
     main = "Scatter Plot of Transformed Production Data")

# Add the estimated regression line
abline(lm_result, col = "red")
```

3.18-d

```r
# Obtain the residuals
residuals <- resid(lm_result)

# Plot residuals against fitted values
plot(fitted(lm_result), residuals,
```

```
     xlab = "Fitted values", ylab = "Residuals",
     main = "Residuals vs Fitted Values")

# Add a horizontal line at y = 0
abline(h = 0, col = "red")

# Prepare a normal probability plot
qqnorm(residuals, main = "Normal Probability Plot of Residuals")
qqline(residuals)
```

3.24-a1

```
# Get all the data
x <- c(5, 8, 11, 7, 13, 12, 12, 6)
y <- c(63, 67, 74, 64, 75, 69, 90, 60)

# Fit a linear regression model
model <- lm(y ~ x)

# Extract coefficients
intercept <- coef(model)[1]
slope <- coef(model)[2]
```

3.24-a2

```
# Calculate residuals
residuals <- resid(model)

# Plot residuals against Xi
plot(x, residuals, xlab = "X", ylab = "Residuals", main = "Residual Plot")
abline(h = 0, col = "red")  # Add a horizontal line at 0
```

3.24-b

```
# Omit case 7
x <- c(5, 8, 11, 7, 13, 12, 6)
y <- c(63, 67, 74, 64, 75, 69, 60)

# Fit a linear regression model
model <- lm(y ~ x)

# Extract coefficients
intercept <- coef(model)[1]
slope <- coef(model)[2]
```

3.24-c

```r
# Predict Y for a new observation at X = 12
new_x <- 12
predicted_y <- predict(model, newdata = data.frame(x = new_x),
                       interval = "prediction", level = 0.99)

# Get prediction interval
lower <- min(predicted_y)
upper <- max(predicted_y)
```