Homework 2.

Theory question

Question 1. Derive the normal equations and the $\hat{\beta}$ from the multivariate version of the multiple regression likelihood

$$\frac{\partial \ell(\beta, \sigma^2)}{\partial \beta} = \frac{\partial}{\partial \beta}\left[-\frac{n}{2}\log(2\pi\sigma^2) - \frac{1}{2\sigma^2}(Y - X\beta)'(Y - X\beta)\right]$$

Since $-\frac{n}{2}\log(2\pi\sigma^2)$ is constant with respect to β:

$$\frac{\partial \ell}{\partial \beta} = 0 - \frac{1}{2\sigma^2}\frac{\partial}{\partial \beta}\left[(Y - X\beta)'(Y - X\beta)\right]$$

Let $Q = (Y - X\beta)'(Y - X\beta) = Y'Y - 2\beta'X'Y + \beta'X'X\beta$. Then:

$$\frac{\partial Q}{\partial \beta} = -2X'Y + 2X'X\beta$$

Set the derivative to zero:

$$\begin{aligned}
\frac{1}{\sigma^2}(2X'Y - 2X'X\beta) &= 0 \\
X'X\beta &= X'Y \\
\hat{\beta}_{MLE} &= (X'X)^{-1}X'Y
\end{aligned}$$

Question 2. Demonstrate that $\hat{\beta}$ is an unbiased estimator of β

$$
\begin{aligned}
Y &= X\beta + \epsilon \\
\hat{\beta} &= (X'X)^{-1}X'Y \\
\hat{\beta} &= (X'X)^{-1}X'(X\beta + \epsilon) \\
\hat{\beta} &= (X'X)^{-1}X'(X\beta) + (X'X)^{-1}X'(\epsilon) \\
\hat{\beta} &= (X'X)^{-1}(X'X)\beta + (X'X)^{-1}X'(\epsilon) \\
\hat{\beta} &= \beta + (X'X)^{-1}X'(\epsilon) \\
\mathrm{E}(X) &= \beta + \mathrm{E}((X'X)^{-1}X'(\epsilon)) \\
\mathrm{E}(X) &= \beta + (X'X)^{-1}X'\,\mathrm{E}(\epsilon) \\
\mathrm{E}(\epsilon) &= 0 \qquad\qquad \text{(by assumption)}
\end{aligned}
$$

$$
\boxed{\mathrm{E}(X) = \beta}
$$

Question 3. Demonstrate that $\hat{\beta}$ follows a multivariate normal distribution

$$
\begin{aligned}
\mathrm{Cov}(\hat{\beta}) &= \mathrm{Cov}((X'X)^{-1}X'\varepsilon) \\
&= (X'X)^{-1}X' \cdot \mathrm{Cov}(\varepsilon) \cdot [(X'X)^{-1}X']' \\
&= (X'X)^{-1}X' \cdot (\sigma^2 I_n) \cdot X(X'X)^{-1} \\
&= (\sigma^2 I_n)(X'X)^{-1}X'X(X'X)^{-1} \\
&= (\sigma^2 I_n)(X'X)^{-1}
\end{aligned}
$$

$$
\mathrm{E}(X) = \beta \qquad \mathrm{Cov}(\hat{\beta}) = (\sigma^2 I_n)(X'X)^{-1}
$$

A **linear transformation** of a multivariate normal is also a **multivariate normal**, and

$$
\hat{\beta} = A + C(\epsilon)
$$

$$
\hat{\beta} = \beta + (X'X)^{-1}X'\epsilon
$$

Therefore

Since $\epsilon \sim N(0, \sigma^2 I_n)$ and $\hat{\beta}$ is a linear transformation of ϵ, it follows that:

$$
\hat{\beta} \sim N(\beta, \sigma^2 (X'X)^{-1})
$$

US Judge ratings

Regression Analysis for USJudgeRatings

This will be split into 5 step, 1. Preliminary data exploration to understand relationship between target variable `RTEN` and `PREP` and pltting the 2 variables against each other. 2. Simple Linear Relationship: Building a simple linear regression between `RTEN` and `PREP` and Plotting residuals against trial preparation variable to see if the relationship is linear 3. Variable selection: Running an F test to see if additional variables are needed 4. Multiple Linear Relationship: using an algorithmic approach to see if other variables are needed 5. Building final model and interpretation

STEP 1. DATA PREPARATION

```
# 1. Basic structure & missingness
cat("Missing values:", sum(is.na(USJudgeRatings)), "\n")
```

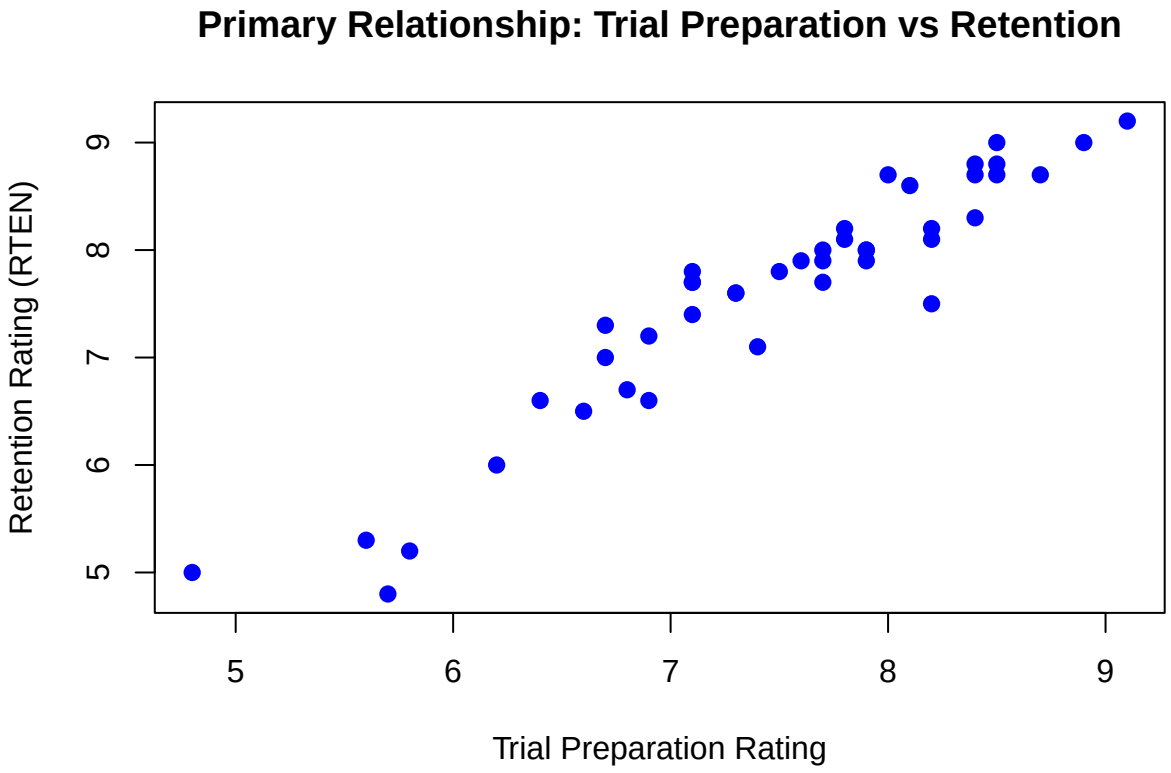
```
## Missing values: 0
```

```
cat("\nNo missing values -- ok to proceed with whole dataset")
```

```
##
## No missing values -- ok to proceed with whole dataset
```

Zooming into the trial preparation vs retention, we see that there is a linear relationship between the 2 variables - this justifies our selection for a simple linear regression between `RTEN` and `PREP`

```
# Focused plot for Trial Preparation vs Retention Rating
plot(USJudgeRatings$PREP, USJudgeRatings$RTEN,
     xlab = "Trial Preparation Rating",
     ylab = "Retention Rating (RTEN)",
     main = "Primary Relationship: Trial Preparation vs Retention",
     pch = 16, col = "blue", cex = 1.2)
```



STEP 2. SIMPLE LINEAR MODEL OF RTEN AGAINST PREP

We see that a 1 unit increase in the preparedness of the judge increases their retention by 1.09 points. This model also has a high adjusted

$$R^2$$

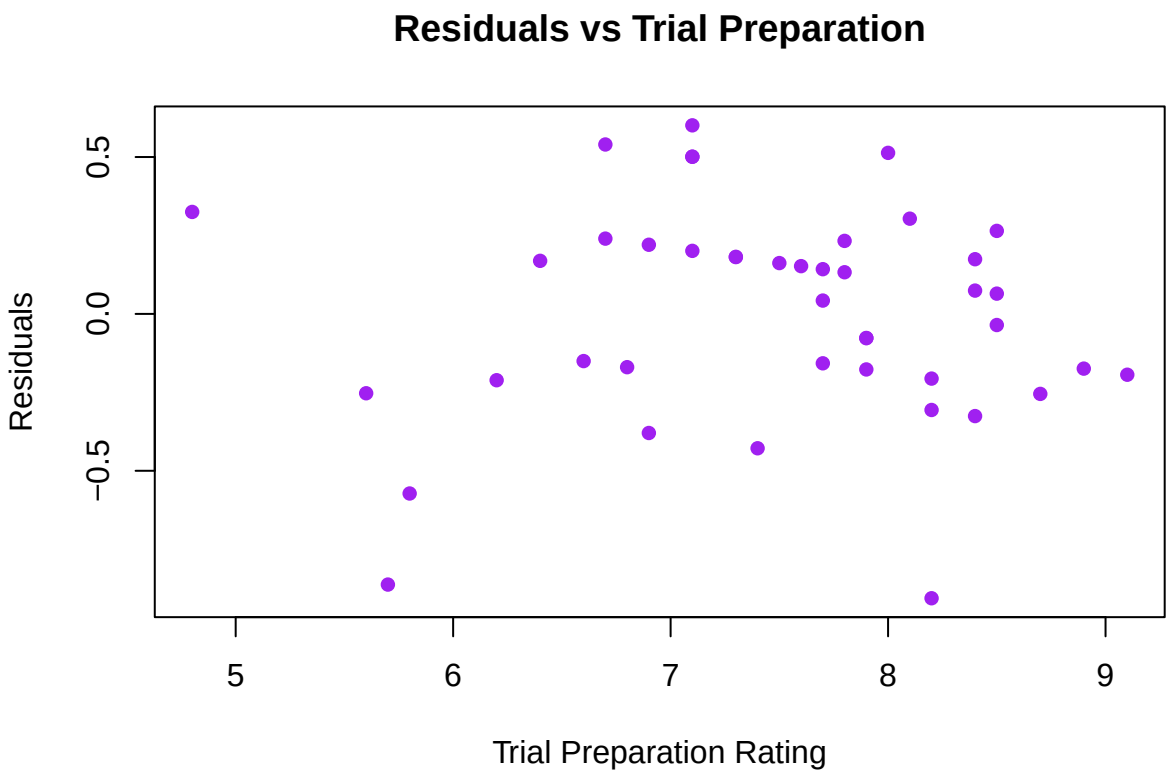
of 0.90, indicating that over 90% of the variation in judge retention can be explained by the judge preparation variable alone.

```
model_prep <- lm(RTEN ~ PREP, data = USJudgeRatings)
summary(model_prep)
```

```
##
## Call:
## lm(formula = RTEN ~ PREP, data = USJudgeRatings)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.90625 -0.20008  0.06453  0.21065  0.60091
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.59257    0.42265  -1.402    0.168
## PREP         1.09742    0.05615  19.543   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.347 on 41 degrees of freedom
## Multiple R-squared:  0.9031, Adjusted R-squared:  0.9007
## F-statistic: 381.9 on 1 and 41 DF,  p-value: < 2.2e-16
```

We see that the plot between the residuals and the trial preparation variable resemble a random scatter, suggesting that there is in fact a linear relationship between trial preparation and judge retention.

```
plot(USJudgeRatings$PREP, residuals(model_prep),
     xlab = "Trial Preparation Rating",
     ylab = "Residuals",
     main = "Residuals vs Trial Preparation",
     pch = 16, col = "purple")
```


Residuals vs Trial Preparation
Residuals
0.5
0.0
−0.5
5
6
7
8
9
Trial Preparation Rating

STEP 3. CHECKING IF OTHER COVARIATES ARE NEEDED

Running an F test against all the other variables to see if the other variables are needed.

For the F test, the null hypothesis states that all coefficients for the additional judicial rating variables (besides PREP) are equal to zero, meaning they provide no additional explanatory power for predicting retention ratings beyond what is already captured by trial preparation alone.

We see that the P value of the F test is near zero, and statistically significant at the 0.1% level, which presents strong evidence against rejecting the null; this indicates that other variables are needed in the model.

```
# Complex model: PREP + all other variables
model_complex <- lm(RTEN ~ ., data = USJudgeRatings)  # . means all other variable
# F-test to compare nested models
f_test_result <- anova(model_prep, model_complex, test='F')
print(f_test_result)
```

```
## Analysis of Variance Table
##
## Model 1: RTEN ~ PREP
## Model 2: RTEN ~ CONT + INTG + DMNR + DILG + CFMG + DECI + PREP + FAMI +
##     ORAL + WRIT + PHYS
##   Res.Df    RSS Df Sum of Sq      F    Pr(>F)
## 1     41 4.9354
## 2     31 0.4274 10     4.508 32.696 1.052e-13 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

STEP 4. MODEL SELECTION AND VARIABLE SELECTION

Using the leaps package, we see that the model with both a high R^2 and a more parsimonious model, with only 5 covariates.

```
library(leaps)

##### Adjusted R^2 for Judicial Ratings #####
best_adjr2 <- leaps(x = USJudgeRatings[,-which(names(USJudgeRatings) == "RTEN")],
                    y = USJudgeRatings$RTEN,
                    method = 'adjr2')

# Create readable output
models <- best_adjr2$which
adjR2 <- cbind(models, round(best_adjr2$adjr2, digits = 5))
colnames(adjR2) <- c(names(USJudgeRatings)[names(USJudgeRatings) != "RTEN"], 'adjR2')

# Display models sorted by adjusted R² (highest first)
top_10_adjr2 <- adjR2[order(best_adjr2$adjr2, decreasing = TRUE),][1:10,]
print(top_10_adjr2)
```

```
##   CONT INTG DMNR DILG CFMG DECI PREP FAMI ORAL WRIT PHYS   adjR2
## 7    0    1    1    0    1    1    0    1    1    0    1 0.98972
## 7    0    1    1    0    1    1    0    0    1    1    1 0.98969
## 5    0    1    1    0    0    1    0    0    1    0    1 0.98967
## 6    0    1    1    0    1    1    0    0    1    0    1 0.98962
## 6    0    1    1    0    0    1    0    1    1    0    1 0.98961
## 6    0    1    1    0    0    1    0    0    1    1    1 0.98960
## 6    0    1    0    0    1    1    0    1    1    0    1 0.98952
## 7    1    1    1    0    1    1    0    0    1    0    1 0.98951
## 8    0    1    1    1    1    1    0    1    1    0    1 0.98950
## 8    1    1    1    0    1    1    0    1    1    0    1 0.98948
```

Based on this R^2 selection criteria, lets build RTEN ~ INTG, DMNR, DECI, ORAL, PHYS

```
model_r2 = lm(RTEN ~ INTG+ DMNR+ DECI+ ORAL+ PHYS, data=USJudgeRatings)
summary(model_r2)
```

```
##
## Call:
## lm(formula = RTEN ~ INTG + DMNR + DECI + ORAL + PHYS, data = USJudgeRatings)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.240656 -0.069026 -0.009474  0.068961  0.246402
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -2.20433    0.43611  -5.055 1.19e-05 ***
## INTG         0.37785    0.10559   3.579 0.000986 ***
## DMNR         0.15199    0.06354   2.392 0.021957 *
## DECI         0.16672    0.07702   2.165 0.036928 *
## ORAL         0.29169    0.10191   2.862 0.006887 **
## PHYS         0.28292    0.04678   6.048 5.40e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Residual standard error: 0.1119 on 37 degrees of freedom
## Multiple R-squared:  0.9909, Adjusted R-squared:  0.9897
## F-statistic: 806.1 on 5 and 37 DF,  p-value: < 2.2e-16
```

Lets also check the BIC – we see as expected, the model with the lowest BIC is the model with 5 covariates- it is also of note that the preparation variable is not included.

```
# Get the top 10 models by adjusted R²
top_10_adjr2 <- adjR2[order(best_adjr2$adjr2, decreasing = TRUE),][1:10,]

# Calculate BIC for each of the top 10 models
n <- nrow(USJudgeRatings)
bic_values <- numeric(10)

for(i in 1:10) {
  # Get which variables are in this model
  model_vars <- names(which(top_10_adjr2[i, 1:11] == 1))  # First 11 columns are variables

  # Build the formula
  if(length(model_vars) > 0) {
    formula <- as.formula(paste("RTEN ~", paste(model_vars, collapse = " + ")))
    model <- lm(formula, data = USJudgeRatings)
    bic_values[i] <- BIC(model)
  } else {
    # Intercept-only model
    model <- lm(RTEN ~ 1, data = USJudgeRatings)
    bic_values[i] <- BIC(model)
  }
}

# Create final results table - convert to data frame first
top_10_with_bic <- data.frame(top_10_adjr2, BIC = bic_values)

# Sort by BIC (lower is better)
top_10_with_bic[order(top_10_with_bic$BIC), ]
```

```
##      CONT INTG DMNR DILG CFMG DECI PREP FAMI ORAL WRIT PHYS   adjR2       BIC
## X5      0    1    1    0    0    1    0    0    1    0    1 0.98967 -46.47553
## X6      0    1    1    0    1    1    0    0    1    0    1 0.98962 -43.65629
## X6.1    0    1    1    0    0    1    0    1    1    0    1 0.98961 -43.60802
## X6.2    0    1    1    0    0    1    0    0    1    1    1 0.98960 -43.59449
## X6.3    0    1    0    0    1    1    0    1    1    0    1 0.98952 -43.23883
## X7      0    1    1    0    1    1    0    1    1    0    1 0.98972 -41.53361
## X7.1    0    1    1    0    1    1    0    0    1    1    1 0.98969 -41.39501
## X7.2    1    1    1    0    1    1    0    0    1    0    1 0.98951 -40.65669
## X8      0    1    1    1    1    1    0    1    1    0    1 0.98950 -38.11787
## X8.1    1    1    1    0    1    1    0    1    1    0    1 0.98948 -38.04348
```

Step 5. building the final model and interpretation

```
summary(model_r2)
```

```
##
## Call:
## lm(formula = RTEN ~ INTG + DMNR + DECI + ORAL + PHYS, data = USJudgeRatings)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.240656 -0.069026 -0.009474  0.068961  0.246402
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -2.20433    0.43611  -5.055 1.19e-05 ***
## INTG         0.37785    0.10559   3.579 0.000986 ***
## DMNR         0.15199    0.06354   2.392 0.021957 *
## DECI         0.16672    0.07702   2.165 0.036928 *
## ORAL         0.29169    0.10191   2.862 0.006887 **
## PHYS         0.28292    0.04678   6.048 5.40e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1119 on 37 degrees of freedom
## Multiple R-squared:  0.9909, Adjusted R-squared:  0.9897
## F-statistic: 806.1 on 5 and 37 DF,  p-value: < 2.2e-16
```

Overall Model Significance F-statistic = 806.1 with p-value $<$ 2.2e-16 Adjusted R-squared: 0.9897

Extremely significant - the model as a whole explains retention ratings

Interpretation of Coefficients:

Judicial Integrity (INTG) - For each 1-point increase in integrity rating, retention rating increases by 0.378 points, holding other judicial qualities constant

Demeanor (DMNR) - Each 1-point increase in demeanor rating corresponds to a 0.152 point increase in retention rating, adjusting for other factors

Decisiveness (DECI)- More decisive judges receive 0.167 point higher retention ratings per 1-point decisiveness increase

Oral Ruling Quality (ORAL) - Superior oral rulings boost retention ratings by 0.292 points per 1-point quality increase

Physical Ability (PHYS) - Physical ability has the strongest impact: 0.283 point retention increase per 1-point ability increase

Model Performance: $R^2 = 0.9909$: 99.09% of retention rating variance explained

Adjusted $R^2 = 0.9897$: Exceptional fit even after penalizing for 5 predictors

Key Insight: Physical ability and judicial integrity emerge as the most powerful predictors, while your original focus on trial preparation (PREP) was correctly excluded. Lawyers appear to value overall judicial competence and presence more than specific trial skills.

US Judge ratings

Regression Analysis for ROCK

step 1. data preparation step 2. variable transformation step 3. building the model and variable selection step 4. testing assumptions

```
data <- rock
head(data)
```

```
##   area    peri     shape perm
## 1 4990 2791.90 0.0903296  6.3
## 2 7002 3892.60 0.1486220  6.3
## 3 7558 3930.66 0.1833120  6.3
## 4 7352 3869.32 0.1170630  6.3
## 5 7943 3948.54 0.1224170 17.1
## 6 7979 4010.15 0.1670450 17.1
```

STEP 1. DATA PREPARATION

```r
# 1. Basic structure & missingness
cat("Missing values:", sum(is.na(data)), "\n")
```

```
## Missing values: 0
```

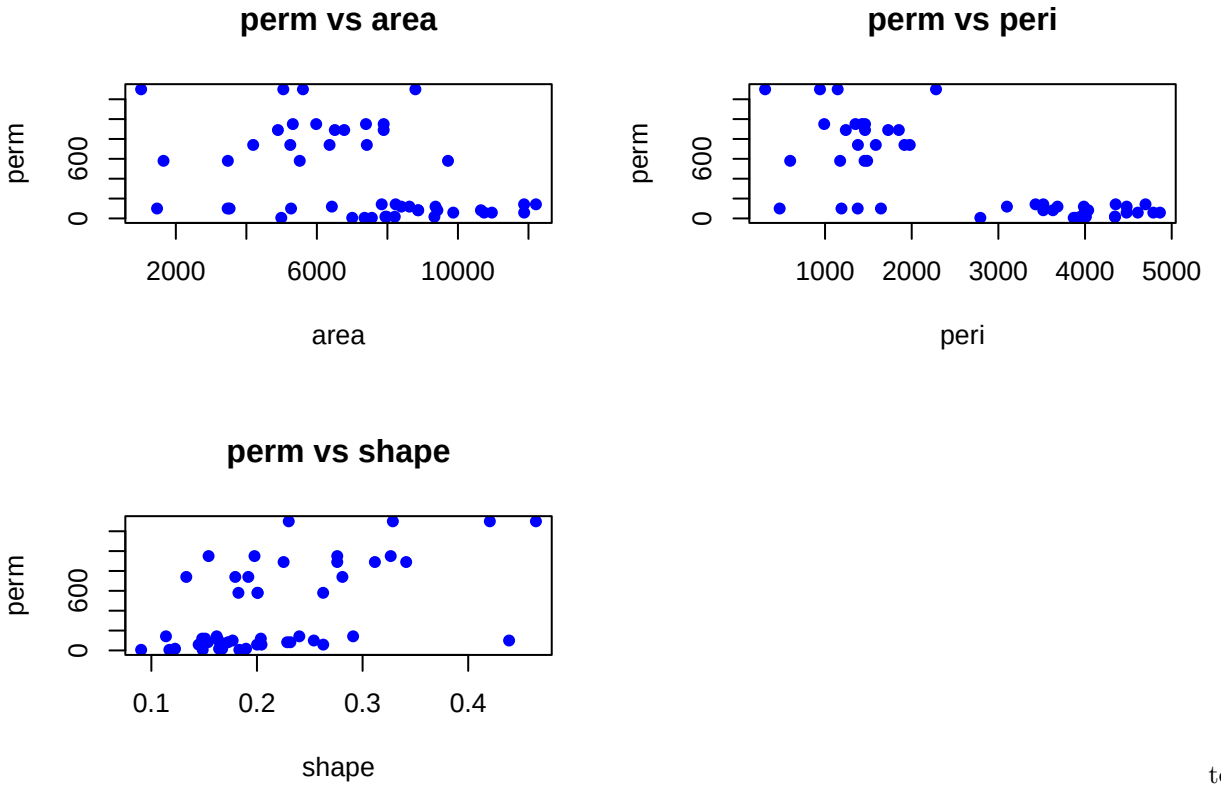
```r
cat("\nNo missing values -- ok to proceed with whole dataset")
```

```
##
## No missing values -- ok to proceed with whole dataset
```

STEP 2. VARIABLE TRANSFORMATION

lets first plot the variables against the target, we don't see a clear linear relationship

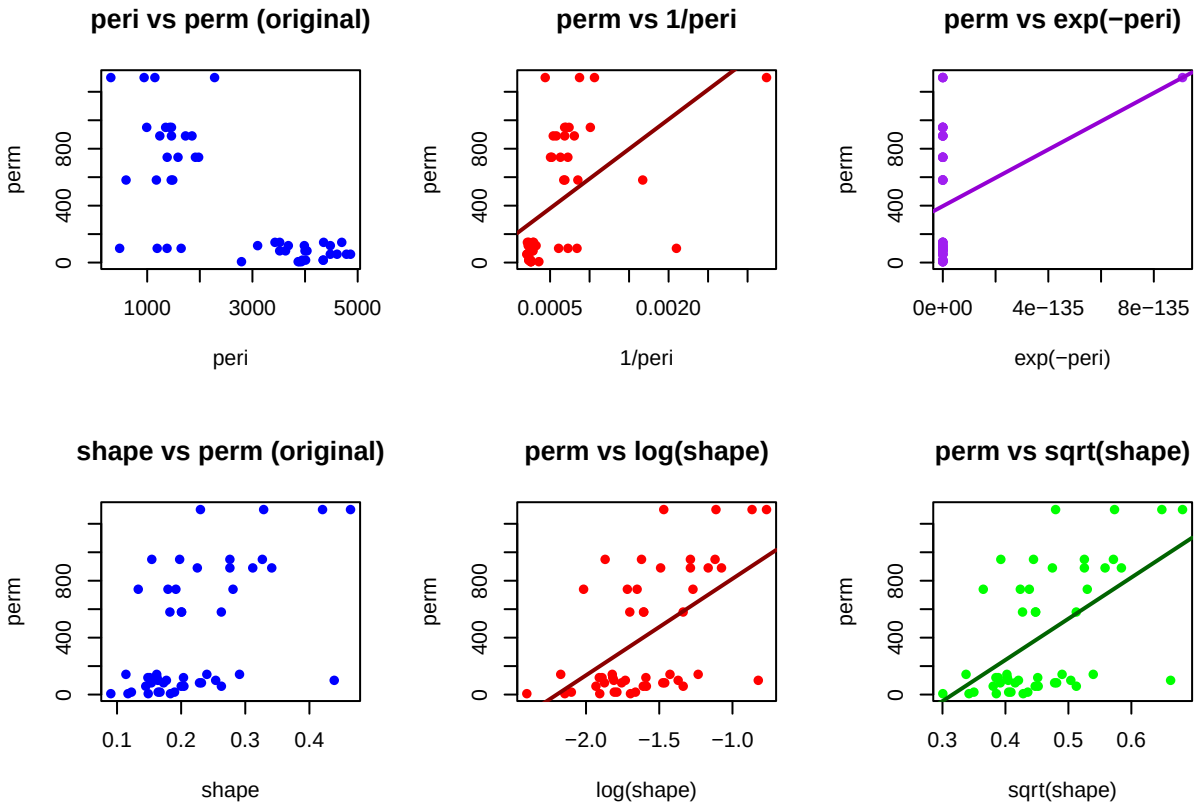
```
# Get predictor names (exclude 'perm')
predictors <- names(data)[names(data) != "perm"]

# Set up plotting layout
par(mfrow = c(2, 2))

# Loop through each predictor and create plots
for (predictor in predictors) {
  # Scatter plot with linear fit
  plot(data[[predictor]], data$perm,
       xlab = predictor, ylab = "perm",
       main = paste("perm vs", predictor),
       pch = 16, col = "blue")

}
```



testing some e^-x or ln(x) doesn;t really help either

```
# Test exponential decay transformation for peri
par(mfrow = c(2, 3))

# Original peri vs perm
plot(data$peri, data$perm, main = "peri vs perm (original)",
     xlab = "peri", ylab = "perm", pch = 16, col = "blue")

# Exponential decay transformation: perm ~ exp(-peri) or perm ~ 1/peri
plot(1/data$peri, data$perm, main = "perm vs 1/peri",
```

```
     xlab = "1/peri", ylab = "perm", pch = 16, col = "red")
abline(lm(perm ~ I(1/peri), data = data), col = "darkred", lwd = 2)

plot(exp(-data$peri), data$perm, main = "perm vs exp(-peri)",
     xlab = "exp(-peri)", ylab = "perm", pch = 16, col = "purple")
abline(lm(perm ~ I(exp(-peri)), data = data), col = "darkviolet", lwd = 2)

# Test logarithmic transformation for shape
# Original shape vs perm
plot(data$shape, data$perm, main = "shape vs perm (original)",
     xlab = "shape", ylab = "perm", pch = 16, col = "blue")

# Log transformation
plot(log(data$shape), data$perm, main = "perm vs log(shape)",
     xlab = "log(shape)", ylab = "perm", pch = 16, col = "red")
abline(lm(perm ~ I(log(shape)), data = data), col = "darkred", lwd = 2)

# Square root transformation (alternative)
plot(sqrt(data$shape), data$perm, main = "perm vs sqrt(shape)",
     xlab = "sqrt(shape)", ylab = "perm", pch = 16, col = "green")
abline(lm(perm ~ I(sqrt(shape)), data = data), col = "darkgreen", lwd = 2)
```

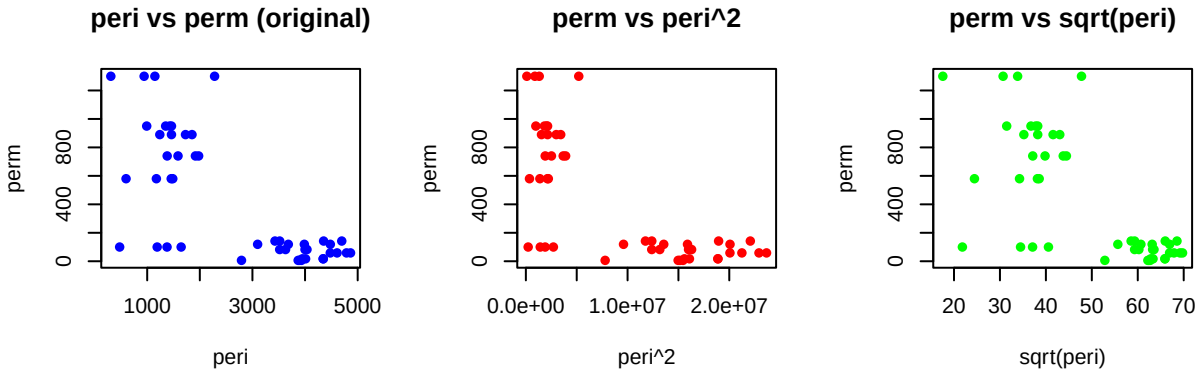


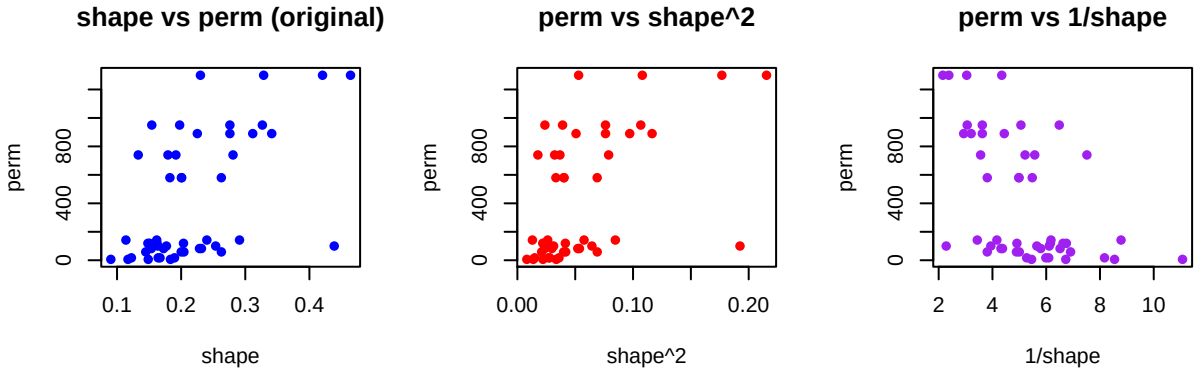

```
par(mfrow = c(1, 1))
```

Higher order terms don't really help make the relationship linear

```
# Try alternative transformations
par(mfrow = c(2, 3))
```

```r
# For peri - try polynomial and other transformations
plot(data$peri, data$perm, main = "peri vs perm (original)",
     xlab = "peri", ylab = "perm", pch = 16, col = "blue")

# Try polynomial for peri
plot(data$peri^2, data$perm, main = "perm vs peri^2",
     xlab = "peri^2", ylab = "perm", pch = 16, col = "red")

# Try sqrt for peri (since it might be scale-related)
plot(sqrt(data$peri), data$perm, main = "perm vs sqrt(peri)",
     xlab = "sqrt(peri)", ylab = "perm", pch = 16, col = "green")

# For shape - try polynomial and reciprocal
plot(data$shape, data$perm, main = "shape vs perm (original)",
     xlab = "shape", ylab = "perm", pch = 16, col = "blue")

# Try polynomial for shape
plot(data$shape^2, data$perm, main = "perm vs shape^2",
     xlab = "shape^2", ylab = "perm", pch = 16, col = "red")

# Try reciprocal for shape
plot(1/data$shape, data$perm, main = "perm vs 1/shape",
     xlab = "1/shape", ylab = "perm", pch = 16, col = "purple")
```

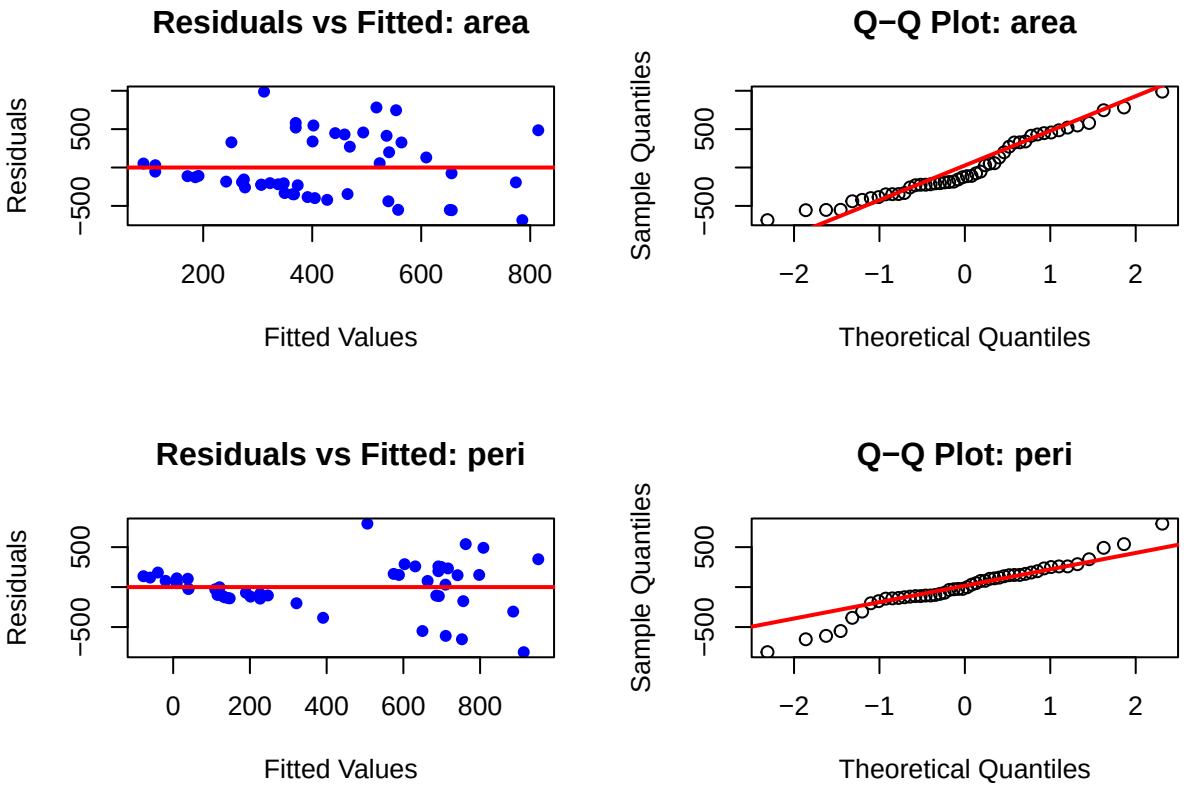




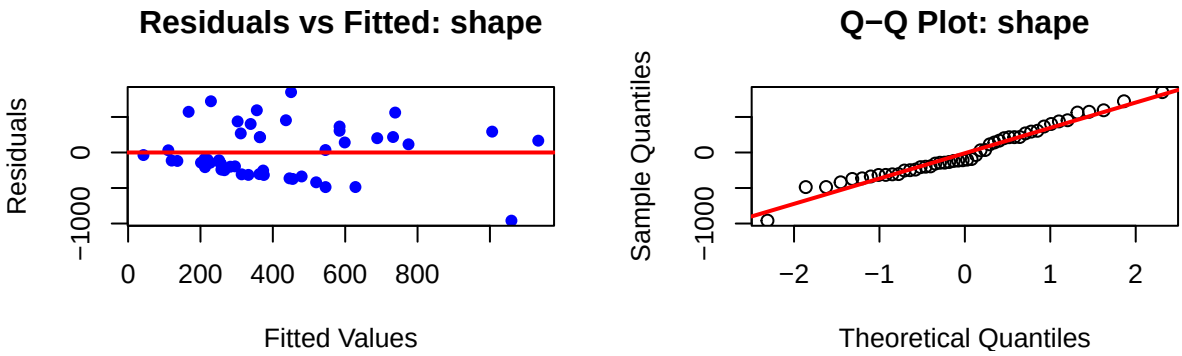

```r
par(mfrow = c(1, 1))
```

```r
# Get predictor names (exclude 'perm')
predictors <- names(data)[names(data) != "perm"]
```

```r
# Set up plotting layout: 2 plots per predictor
par(mfrow = c(2, 2))

# Loop through each predictor
for (predictor in predictors) {
  # Fit simple linear model
  formula <- as.formula(paste("perm ~", predictor))
  model <- lm(formula, data = data)

  # Calculate residuals and fitted values
  residuals <- resid(model)
  fitted <- fitted(model)

  # Plot 1: Residuals vs Fitted Values
  plot(fitted, residuals,
       main = paste("Residuals vs Fitted:", predictor),
       xlab = "Fitted Values", ylab = "Residuals",
       pch = 16, col = "blue")
  abline(h = 0, col = "red", lwd = 2)

  # Plot 2: Q-Q Plot
  qqnorm(residuals, main = paste("Q-Q Plot:", predictor))
  qqline(residuals, col = "red", lwd = 2)
}
```

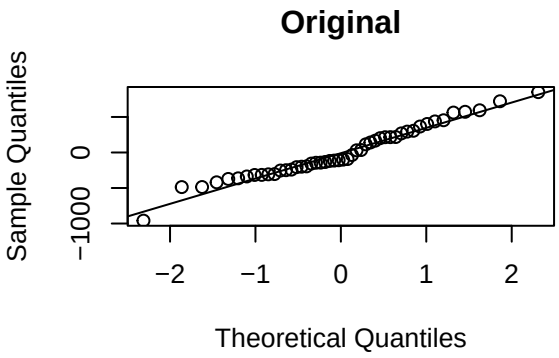


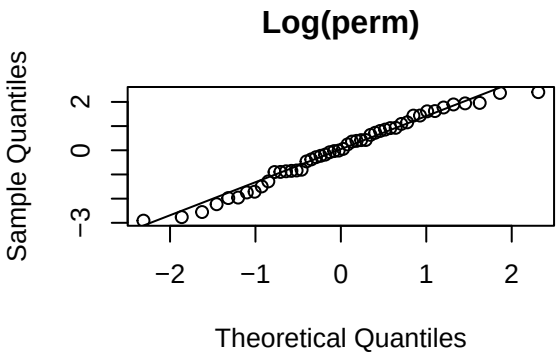

```r
# Reset plotting layout
par(mfrow = c(1, 1))
```

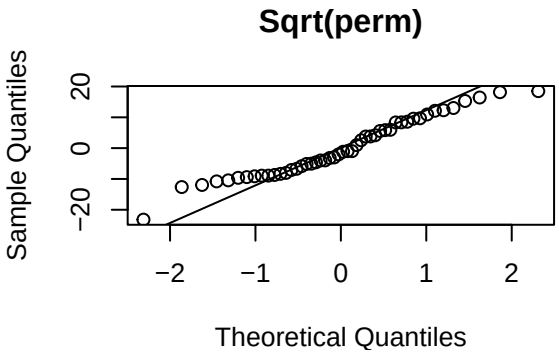


There looks like non-normality in the data according to the QQ plot, lets test some transformations on the Y variable, `perm` ln(perm) looks like a good transformation

```
# You might want to check if transformations help:
# Try transforming the response variable 'perm'
par(mfrow = c(2, 2))

# Original model
model_original <- lm(perm ~ shape, data = data)
qqnorm(resid(model_original), main = "Original")
qqline(resid(model_original))

# Log transformation
model_log <- lm(log(perm) ~ shape, data = data)
qqnorm(resid(model_log), main = "Log(perm)")
qqline(resid(model_log))

# Square root transformation
model_sqrt <- lm(sqrt(perm) ~ shape, data = data)
qqnorm(resid(model_sqrt), main = "Sqrt(perm)")
qqline(resid(model_sqrt))

# Box-Cox transformation (if needed)
# library(MASS)
# boxcox(perm ~ shape, data = data)
```

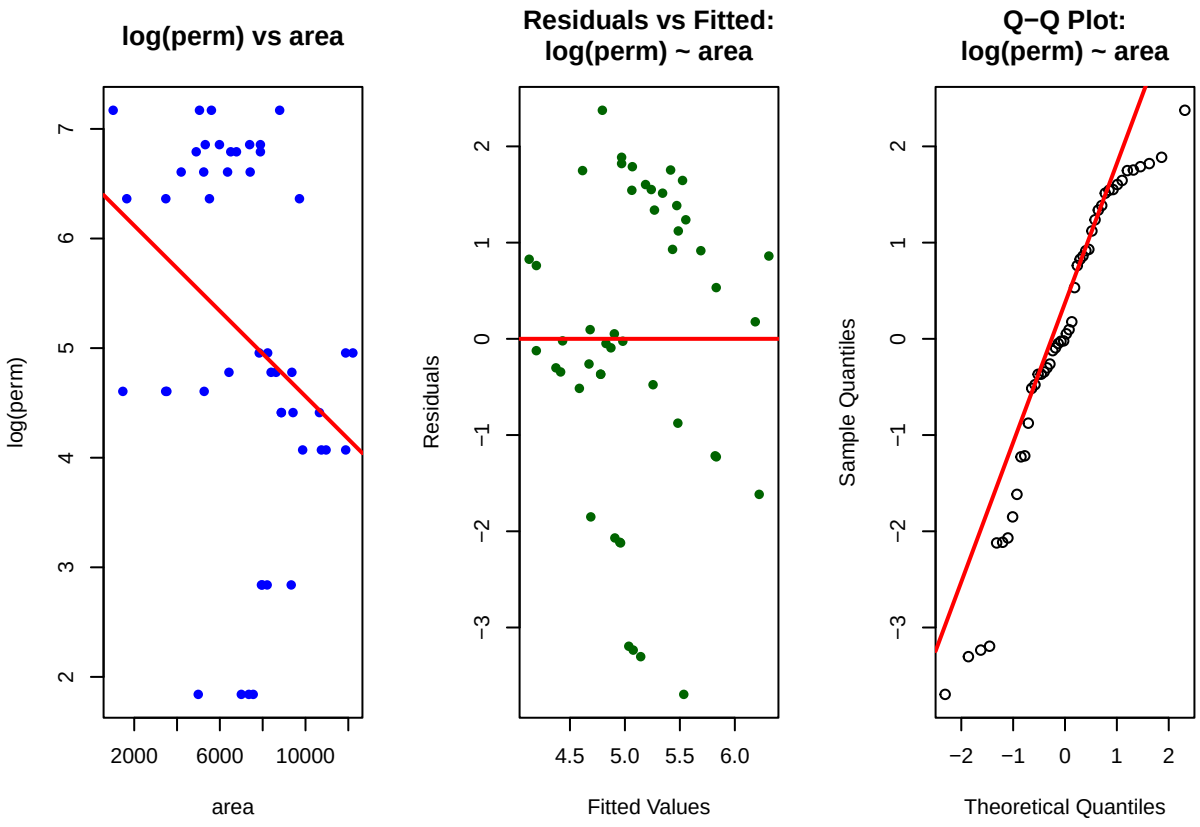






Lets plot `area` against the ln(perm)

```
data$log_perm <- log(data$perm)

# Set up plotting for area
par(mfrow = c(1, 3))

# Fit model with log(perm)
model_area <- lm(log_perm ~ area, data = data)
residuals <- resid(model_area)
fitted <- fitted(model_area)

# Plot 1: log(perm) vs area
plot(data$area, data$log_perm,
     main = "log(perm) vs area",
     xlab = "area", ylab = "log(perm)",
     pch = 16, col = "blue")
abline(model_area, col = "red", lwd = 2)

# Plot 2: Residuals vs Fitted
plot(fitted, residuals,
     main = "Residuals vs Fitted:\nlog(perm) ~ area",
     xlab = "Fitted Values", ylab = "Residuals",
     pch = 16, col = "darkgreen")
abline(h = 0, col = "red", lwd = 2)

# Plot 3: Q-Q Plot
qqnorm(residuals, main = "Q-Q Plot:\nlog(perm) ~ area")
qqline(residuals, col = "red", lwd = 2)
```

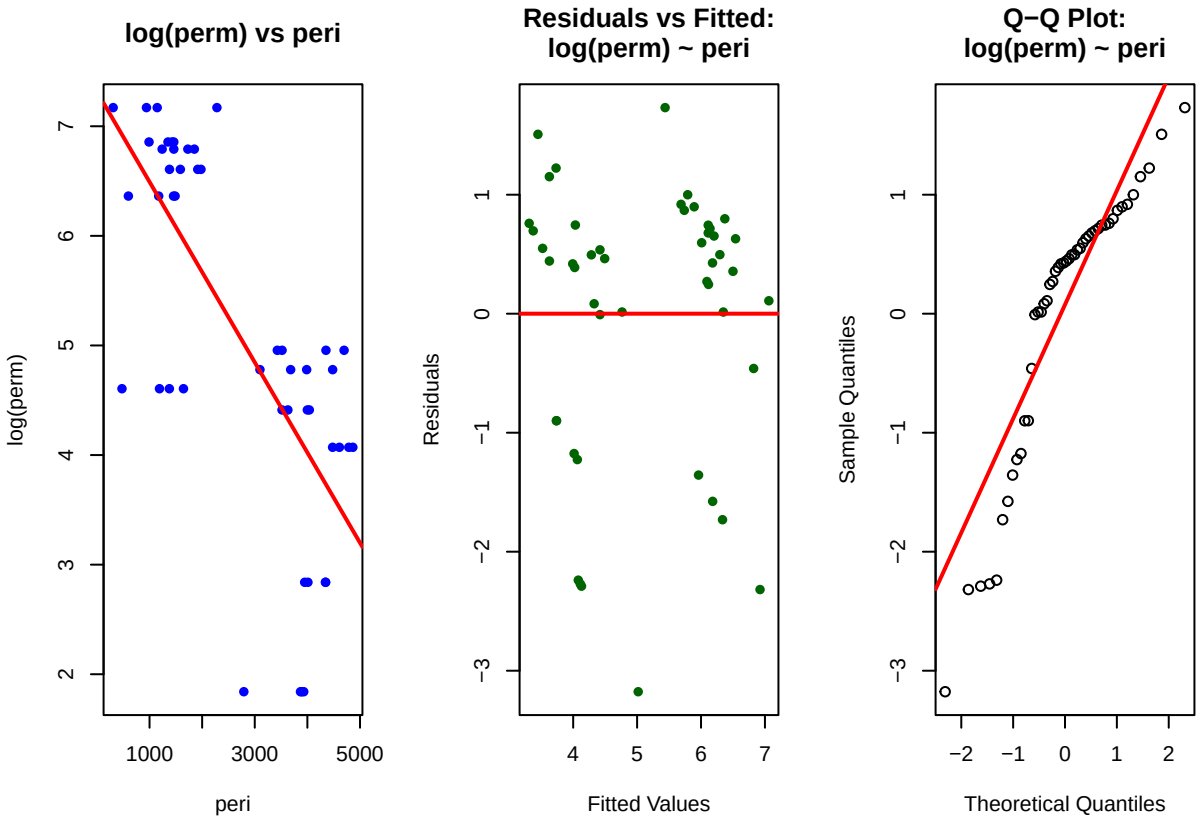



```
par(mfrow = c(1, 1))
```

Lets plot `peri` against ln(perm)

```
# Set up plotting for peri
par(mfrow = c(1, 3))

# Fit model with log(perm)
model_peri <- lm(log_perm ~ peri, data = data)
residuals <- resid(model_peri)
fitted <- fitted(model_peri)

# Plot 1: log(perm) vs peri
plot(data$peri, data$log_perm,
     main = "log(perm) vs peri",
     xlab = "peri", ylab = "log(perm)",
     pch = 16, col = "blue")
abline(model_peri, col = "red", lwd = 2)

# Plot 2: Residuals vs Fitted
plot(fitted, residuals,
     main = "Residuals vs Fitted:\nlog(perm) ~ peri",
     xlab = "Fitted Values", ylab = "Residuals",
     pch = 16, col = "darkgreen")
abline(h = 0, col = "red", lwd = 2)

# Plot 3: Q-Q Plot
qqnorm(residuals, main = "Q-Q Plot:\nlog(perm) ~ peri")
```

```r
qqline(residuals, col = "red", lwd = 2)
```

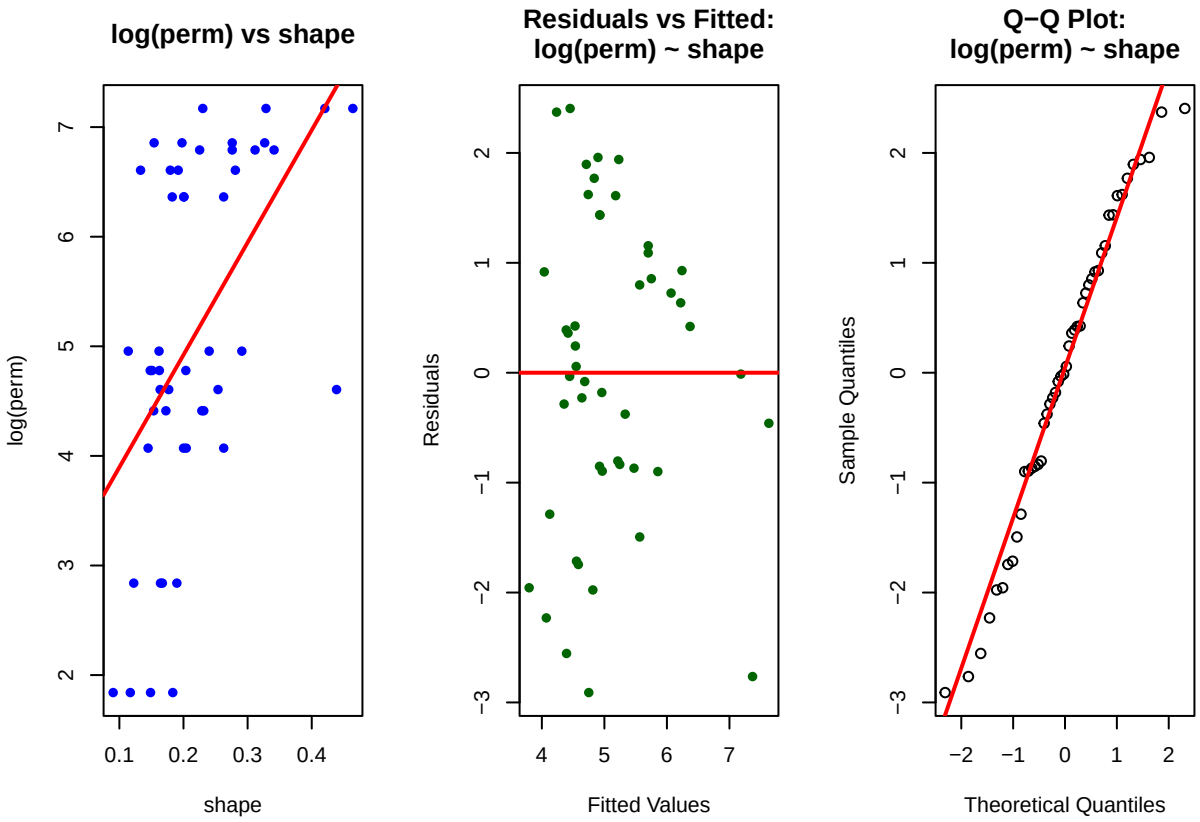



```r
par(mfrow = c(1, 1))
```

Lets plot **shape** against ln(perm)

```r
# Set up plotting for shape
par(mfrow = c(1, 3))

# Fit model with log(perm)
model_shape <- lm(log_perm ~ shape, data = data)
residuals <- resid(model_shape)
fitted <- fitted(model_shape)

# Plot 1: log(perm) vs shape
plot(data$shape, data$log_perm,
     main = "log(perm) vs shape",
     xlab = "shape", ylab = "log(perm)",
     pch = 16, col = "blue")
abline(model_shape, col = "red", lwd = 2)

# Plot 2: Residuals vs Fitted
plot(fitted, residuals,
     main = "Residuals vs Fitted:\nlog(perm) ~ shape",
     xlab = "Fitted Values", ylab = "Residuals",
     pch = 16, col = "darkgreen")
abline(h = 0, col = "red", lwd = 2)
```

```r
# Plot 3: Q-Q Plot
qqnorm(residuals, main = "Q-Q Plot:\nlog(perm) ~ shape")
qqline(residuals, col = "red", lwd = 2)
```



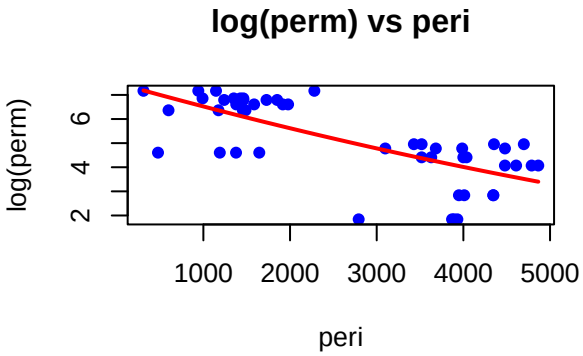

```r
par(mfrow = c(1, 1))
```

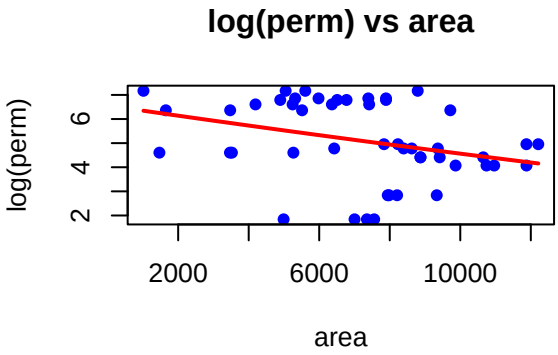
`Shape` variable looks good, lets look at `peri` and `area`

`peri` seems to have a quadratic form with perm

```r
# Test polynomial relationships
par(mfrow = c(2, 2))

# Peri polynomials
plot(data$peri, data$log_perm, main = "log(perm) vs peri",
     xlab = "peri", ylab = "log(perm)", pch = 16, col = "blue")
peri_quad <- lm(log_perm ~ poly(peri, 2), data = data)
x_seq <- seq(min(data$peri), max(data$peri), length = 100)
y_pred <- predict(peri_quad, newdata = data.frame(peri = x_seq))
lines(x_seq, y_pred, col = "red", lwd = 2)

# Area polynomials
plot(data$area, data$log_perm, main = "log(perm) vs area",
     xlab = "area", ylab = "log(perm)", pch = 16, col = "blue")
area_quad <- lm(log_perm ~ poly(area, 2), data = data)
x_seq <- seq(min(data$area), max(data$area), length = 100)
y_pred <- predict(area_quad, newdata = data.frame(area = x_seq))
lines(x_seq, y_pred, col = "red", lwd = 2)
```

```r
par(mfrow = c(1, 1))
```





STEP 3. fitting the model - lets test our transformed variables against the standard linear model as a baseline

we see that the adjusted R^2 of the standard linear model is still higher than the transformed model despite the transformed model having a lower AIC and BIC. This suggests that the `area` variable has some explanatory power.

```
# Model 1: Linear model with original perm
model1 <- lm(perm ~ peri + area + shape, data = data)

# Model 2: Log model with quadratic peri term
model2 <- lm(log(perm) ~ peri + I(peri^2) + shape, data = data)

# Calculate metrics for both models
results <- data.frame(
  Model = c("perm ~ peri + area + shape", "log(perm) ~ peri + peri^2 + shape"),
  R_squared = c(summary(model1)$r.squared, summary(model2)$r.squared),
  Adj_R_squared = c(summary(model1)$adj.r.squared, summary(model2)$adj.r.squared),
  AIC = c(AIC(model1), AIC(model2)),
  BIC = c(BIC(model1), BIC(model2))
)
results
```

```
##                               Model R_squared Adj_R_squared      AIC      BIC
## 1        perm ~ peri + area + shape 0.7044103     0.6842564 670.5591 679.9151
## 2 log(perm) ~ peri + peri^2 + shape 0.5707354     0.5414674 152.3031 161.6591
```

F test to see if we need the area variable, we reject the null at the 0.1% level, suggesting that we need the area variable as well.

```
# Reduced model (without area)
reduced_model <- lm(log(perm) ~ peri + I(peri^2) + shape, data = data)

# Full model (with area)
full_model <- lm(log(perm) ~ peri + I(peri^2) + area + shape, data = data)

# F-test to compare models
f_test <- anova(reduced_model, full_model)
print("=== F-test for including area ===")
```

```
## [1] "=== F-test for including area ==="
```

```
print(f_test)
```

```
## Analysis of Variance Table
##
## Model 1: log(perm) ~ peri + I(peri^2) + shape
## Model 2: log(perm) ~ peri + I(peri^2) + area + shape
##   Res.Df    RSS Df Sum of Sq     F    Pr(>F)
## 1     44 54.487
## 2     43 31.310  1    23.177 31.83 1.211e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

It seems like model 3 has the lowest AIC and BIC with the highest R^2

```
# Model 1: Linear model with original perm
model1 <- lm(perm ~ peri + area + shape, data = data)
```

```r
# Model 2: Log model with quadratic peri term
model2 <- lm(log(perm) ~ peri + I(peri^2) + shape, data = data)

# Model 3: Log model with quadratic peri term + area
model3 <- lm(log(perm) ~ peri + I(peri^2) + shape + area, data = data)

# Calculate metrics for all models
results <- data.frame(
  Model = c("perm ~ peri + area + shape",
            "log(perm) ~ peri + peri^2 + shape",
            "log(perm) ~ peri + peri^2 + shape + area"),  # Fixed the typo here
  R_squared = c(summary(model1)$r.squared, summary(model2)$r.squared, summary(model3)$r.squared),
  Adj_R_squared = c(summary(model1)$adj.r.squared, summary(model2)$adj.r.squared, summary(model3)$adj.r
  AIC = c(AIC(model1), AIC(model2), AIC(model3)),
  BIC = c(BIC(model1), BIC(model2), BIC(model3))
)

print(results)
```

```
##                                     Model R_squared Adj_R_squared      AIC
## 1               perm ~ peri + area + shape 0.7044103     0.6842564 670.5591
## 2        log(perm) ~ peri + peri^2 + shape 0.5707354     0.5414674 152.3031
## 3 log(perm) ~ peri + peri^2 + shape + area 0.7533303     0.7303843 127.7099
##        BIC
## 1 679.9151
## 2 161.6591
## 3 138.9371
```

final model selected

```r
summary(model3)
```

```
##
## Call:
## lm(formula = log(perm) ~ peri + I(peri^2) + shape + area, data = data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.8462 -0.5182  0.1548  0.6108  1.3679
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  5.952e+00  8.595e-01   6.925 1.65e-08 ***
## peri        -2.089e-03  6.260e-04  -3.337  0.00176 **
## I(peri^2)    9.465e-08  1.011e-07   0.936  0.35430
## shape        1.101e+00  1.892e+00   0.582  0.56360
## area         5.074e-04  8.993e-05   5.642 1.21e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.8533 on 43 degrees of freedom
## Multiple R-squared:  0.7533, Adjusted R-squared:  0.7304
## F-statistic: 32.83 on 4 and 43 DF,  p-value: 1.465e-12
```

STEP 4. testing assumptions – residuals look pretty randomly scattered, and the QQ plot, while not

completely hugging the line look decent.

Residuals vs Fitted Plot (if it's random noise): What it tells us:

Linearity assumption is satisfied - the relationship between predictors and response is properly captured

Constant variance (homoscedasticity) - the spread of residuals is consistent across all fitted values

No pattern means no systematic bias in the model

The model specification is correct - you've included the right terms (like the quadratic peri term)

Q-Q Plot (if it hugs the line): What it tells us:

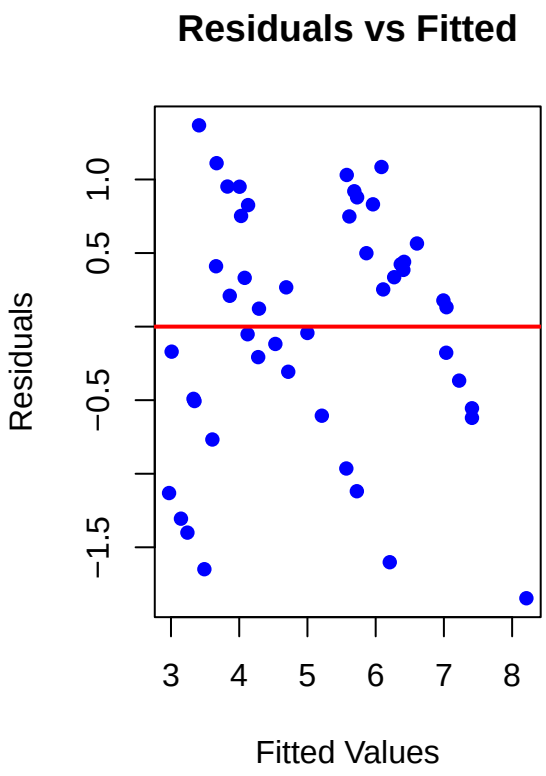
Normality assumption is satisfied - the residuals follow a normal distribution

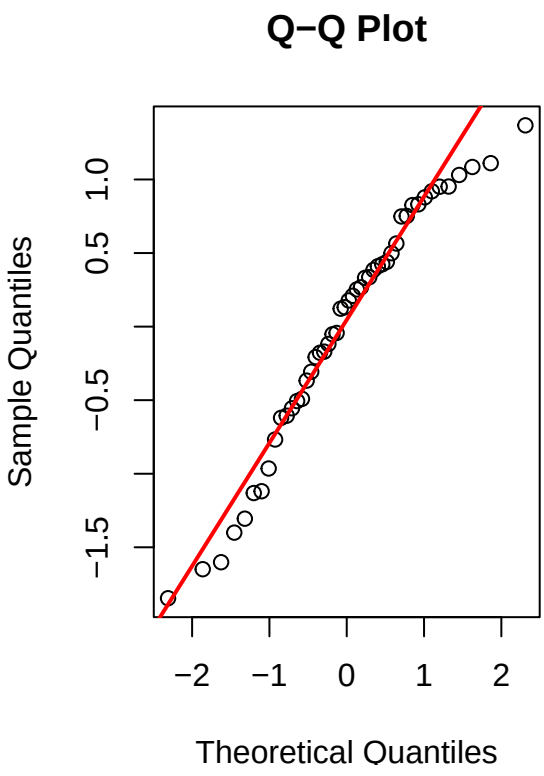
Reliable p-values and confidence intervals - statistical inference is valid

No extreme outliers that could distort the results

The error distribution is well-behaved

```r
# Fit the best model
model3 <- lm(log(perm) ~ peri + I(peri^2) + shape + area, data = data)

# Set up plotting area
par(mfrow = c(1, 2))

# 1. Residuals vs Fitted (Linearity & Constant Variance)
plot(fitted(model3), resid(model3),
     main = "Residuals vs Fitted",
     xlab = "Fitted Values", ylab = "Residuals",
     pch = 16, col = "blue")
abline(h = 0, col = "red", lwd = 2)

# 2. Q-Q Plot (Normality)
qqnorm(resid(model3), main = "Q-Q Plot")
qqline(resid(model3), col = "red", lwd = 2)
```






```
par(mfrow = c(1, 1))
```