MLB Standings

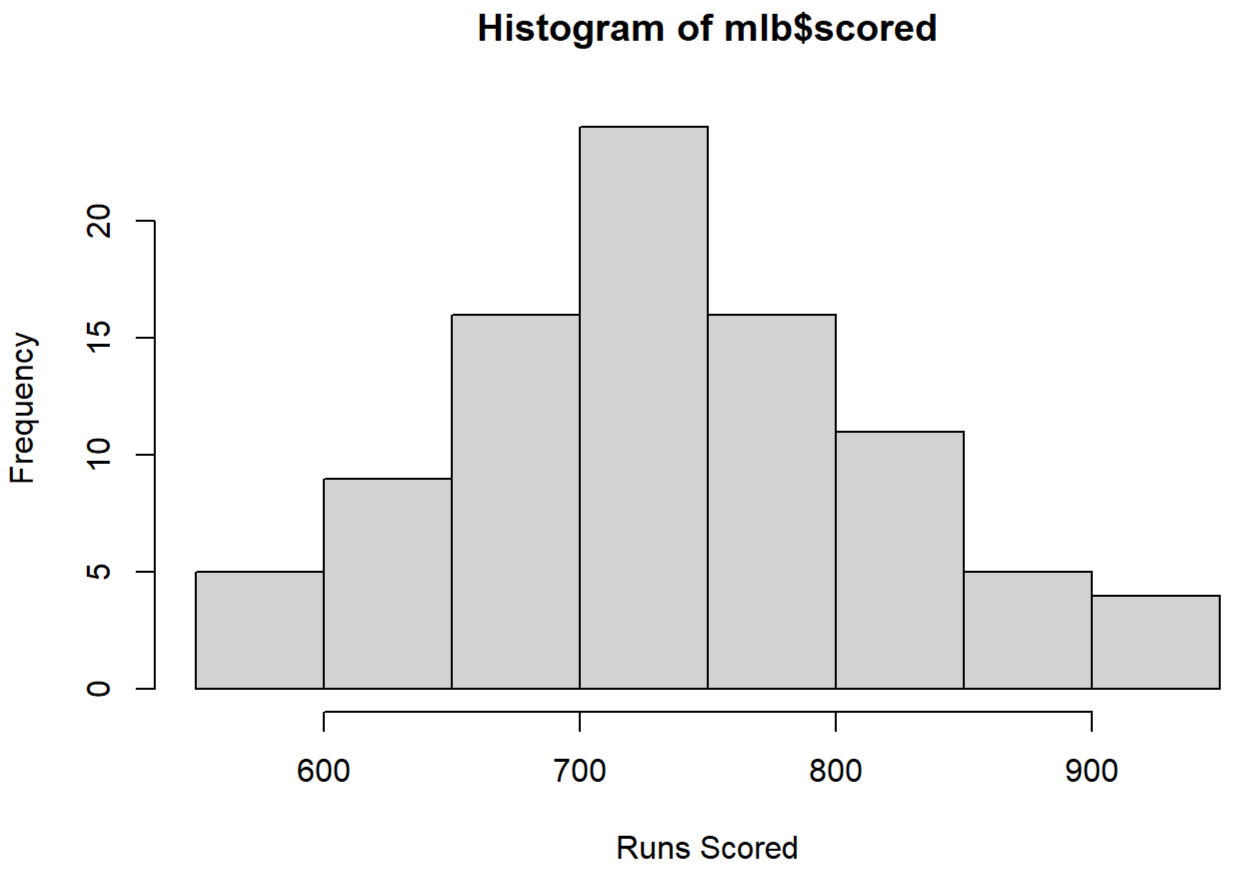
Data Exploration

Read in data

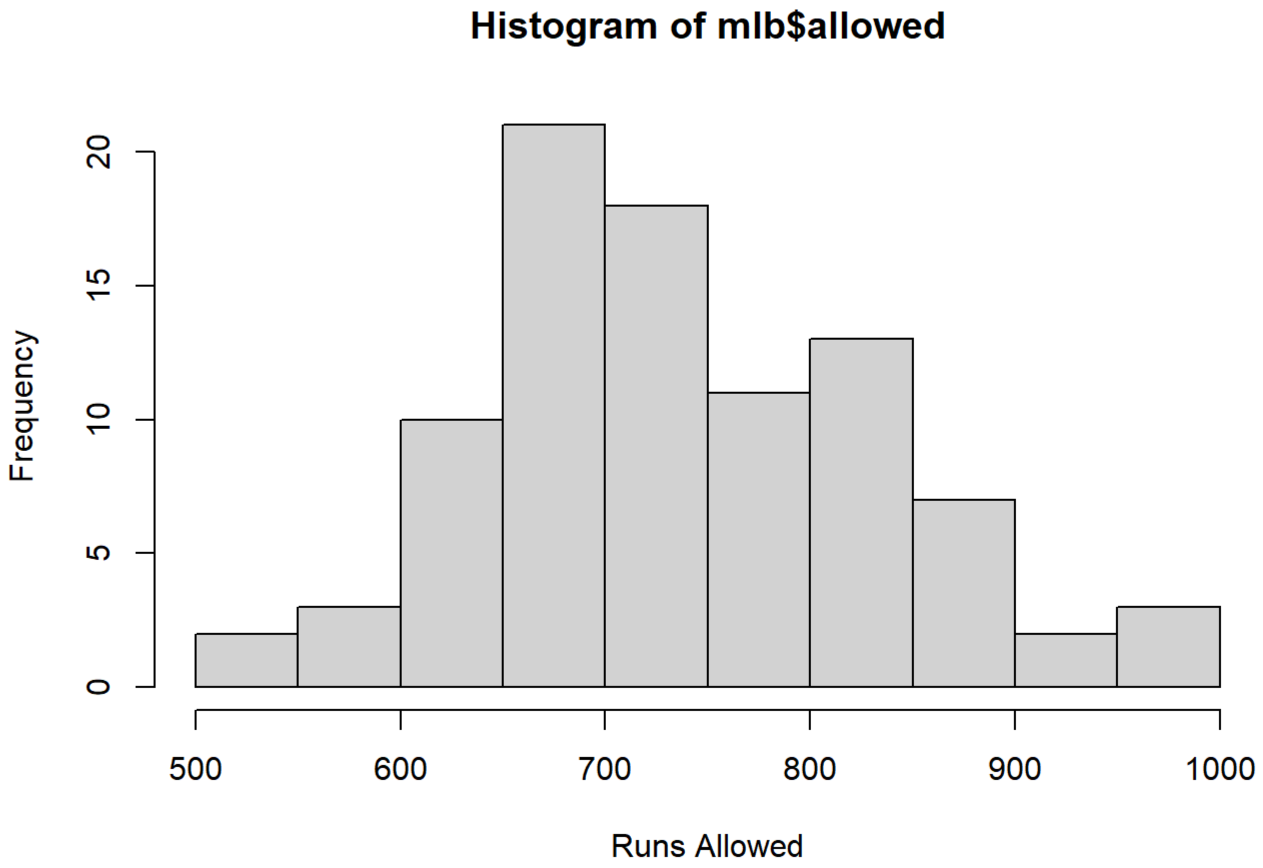
```
mlb = read.table("mlb_standings.csv", header = TRUE, sep = ",")
```

Histograms of each variable

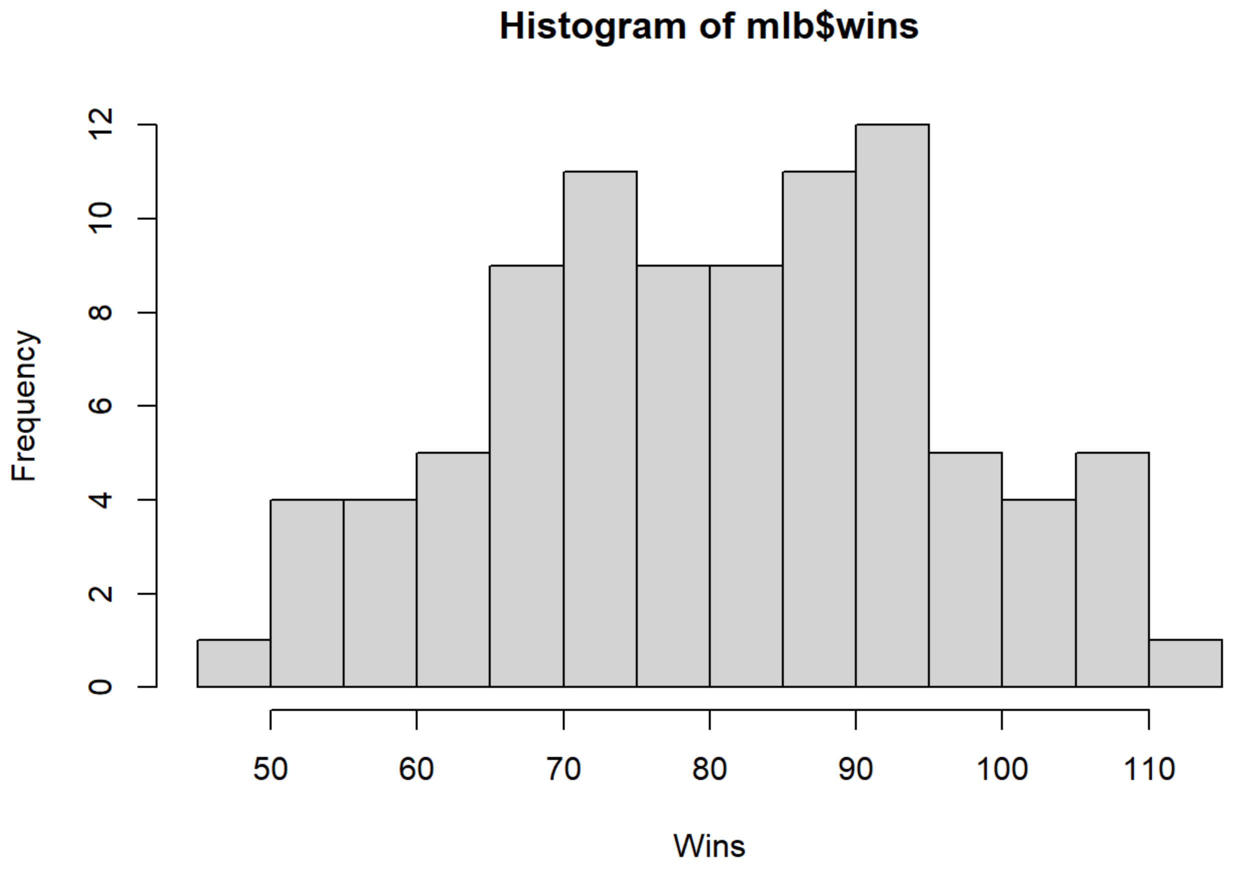
```
hist(mlb$scored, xlab = "Runs Scored")      # Runs scored
```



```
hist(mlb$allowed, xlab = "Runs Allowed")    # Runs allowed
```

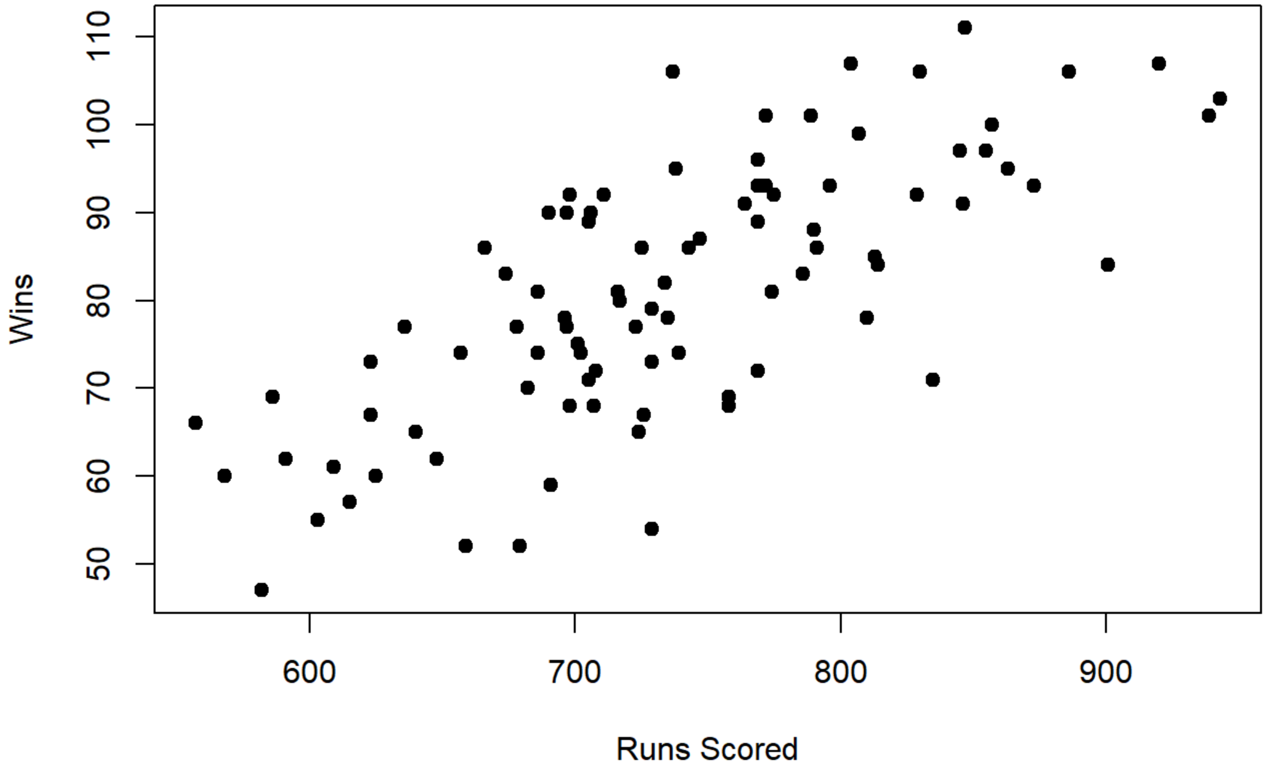


```
hist(mlb$wins, breaks = 12, xlab = "Wins")  # Wins
```

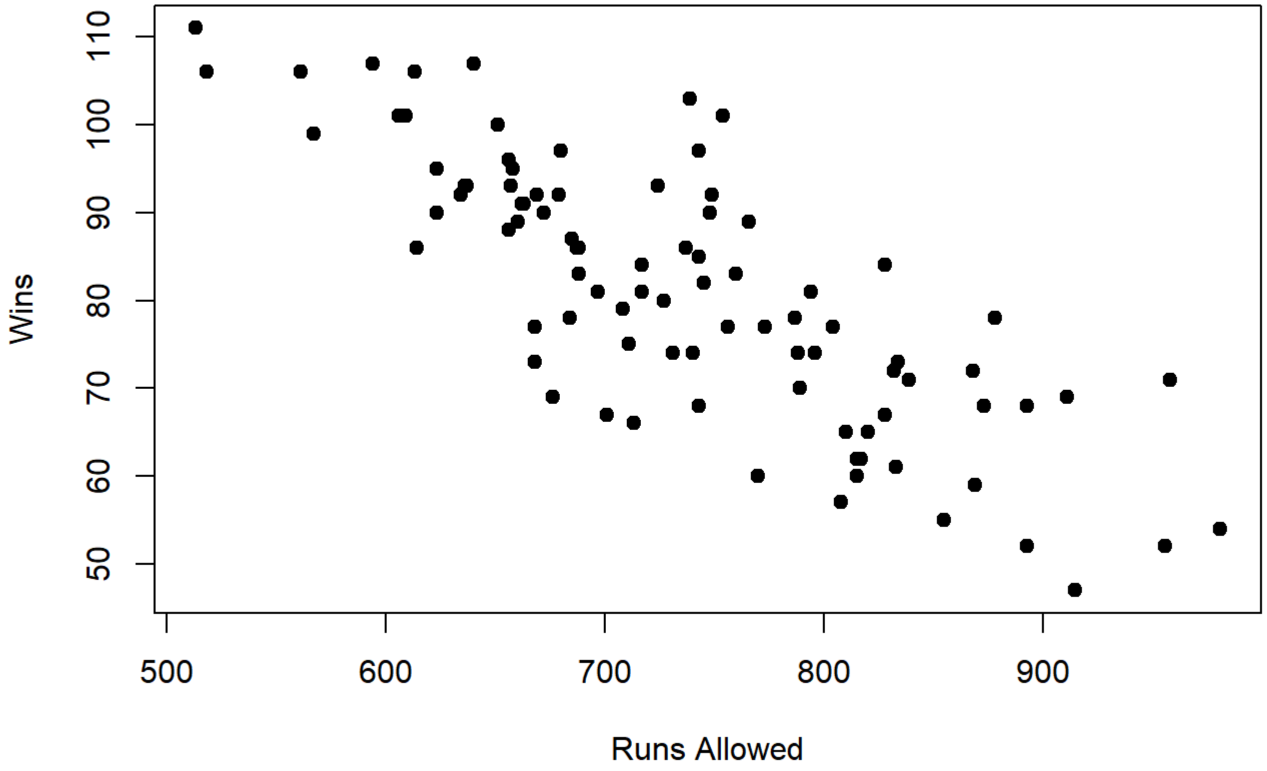


Scatterplots of each pair of variables

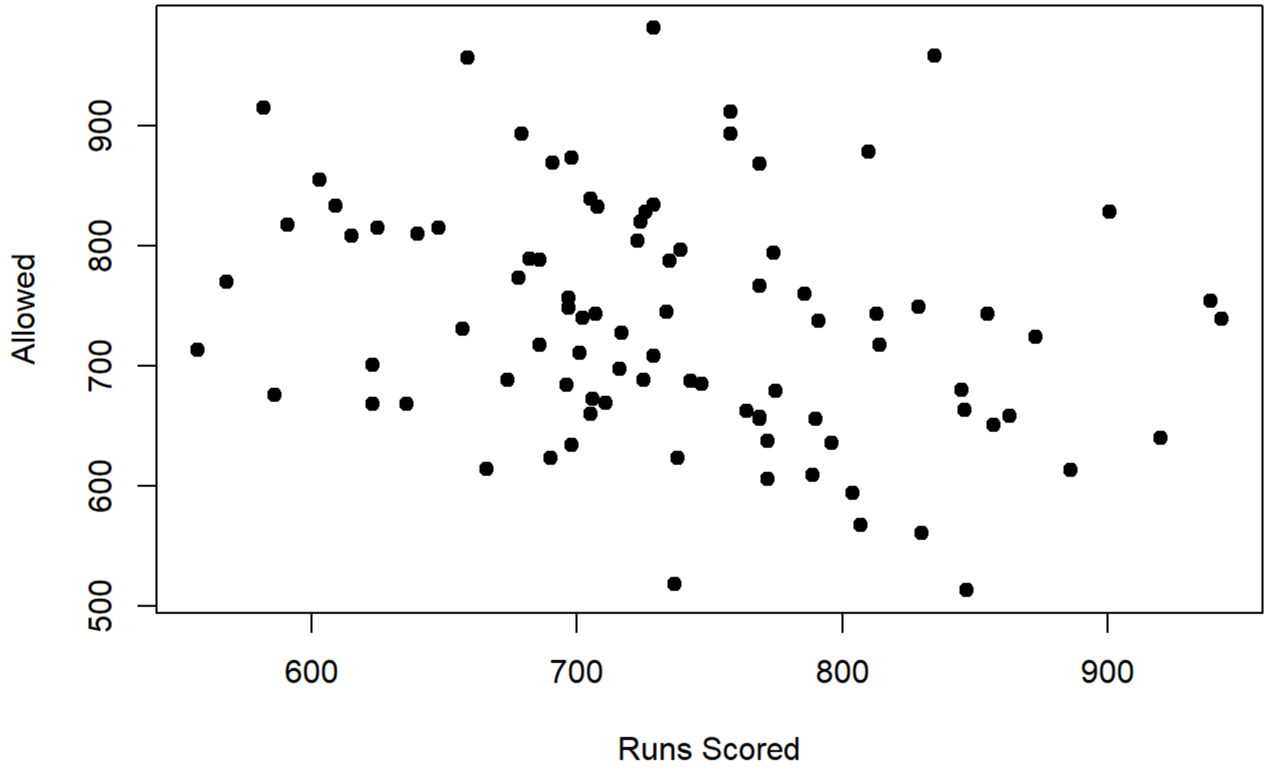
```
plot(mlb$scored, mlb$wins, pch = 19, xlab = "Runs Scored", ylab = "Wins")        # Wins vs. runs scored
```



```
plot(mlb$allowed, mlb$wins, pch = 19, xlab = "Runs Allowed", ylab = "Wins")      # Wins vs. runs
allowed
```



```
plot(mlb$scored, mlb$allowed, pch = 19, xlab = "Runs Scored", ylab = "Allowed")  # Wins vs. runs
allowed
```



Model

Fit model

```
model = lm(wins ~ scored + allowed, data = mlb)
```

Calculate and save predictions, pure residuals, and all three types of other residuals (standardized, studentized, and jackknife)

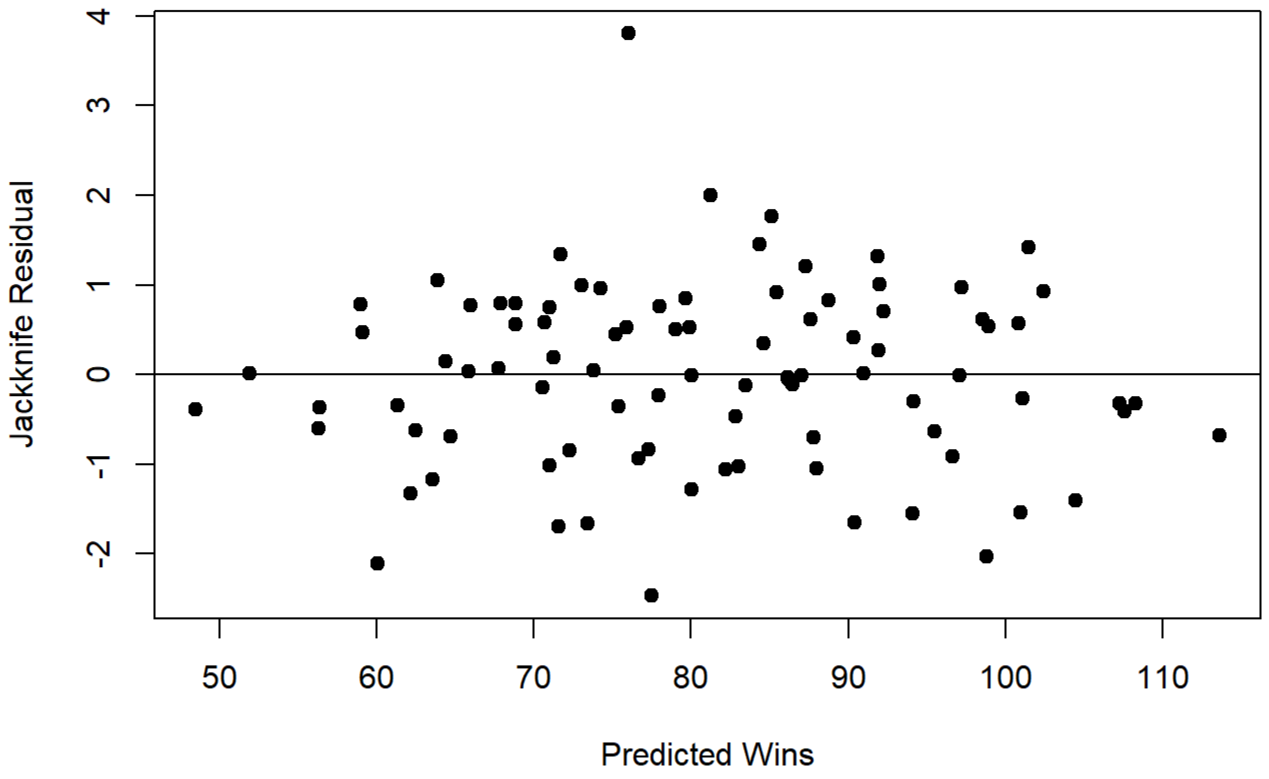
```
require(MASS)
```

```
## Loading required package: MASS
```

```
# Predictions and residuals
mlb$pred = predict(model) # Predictions
mlb$residuals = residuals(model) # Residuals

# Standardized residuals
mlb$stand_res = residuals(model)/sigma(model) # Divide residual by residual standard error

# Studentized residuals
mlb$stud_res = stdres(model)

# Jackknife residuals
mlb$jackknife = rstudent(model)
```

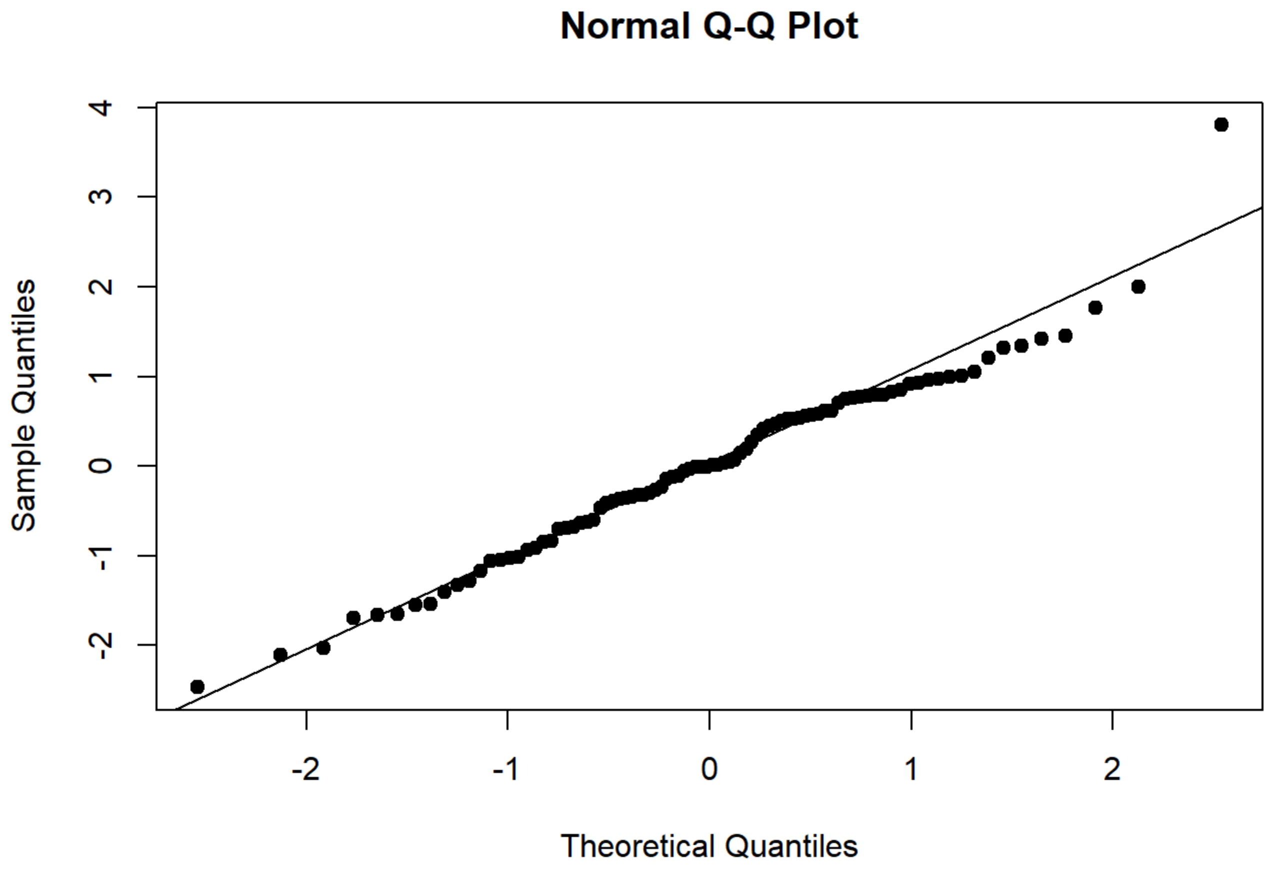
Create residual plot of jackknife residuals against predicted values.

```
mlb$pred = predict(model)
plot(mlb$pred, mlb$jackknife, pch = 19, xlab = "Predicted Wins", ylab = "Jackknife Residual")
abline(a=0, b=0)
```



Create normal probability plot of jackknife residuals.

```
qqnorm(mlb$jackknife, pch = 19)
qqline(mlb$jackknife)  # Puts the diagonal line on the plot
```



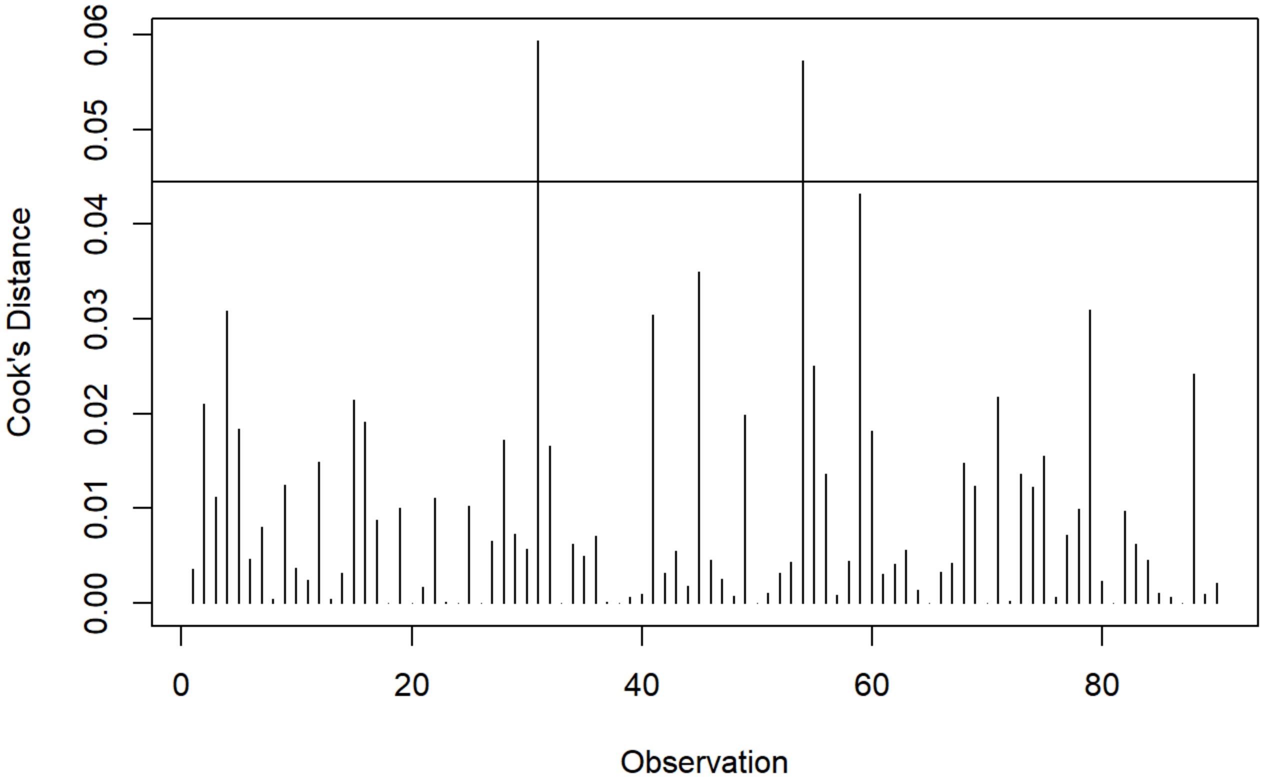
Run Shapiro-Wilk test

```
shapiro.test(mlb$jackknife)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  mlb$jackknife
## W = 0.97688, p-value = 0.1089
```

Calculate and save Cook’s distance values to the data frame

```
mlb$cook = cooks.distance(model)
plot(mlb$cook, xlab = "Observation", ylab = "Cook's Distance", type = "h")
abline(a = 4/90, b = 0)
```



Remove influential points and fit model on remaining data

```
mlb$inf_pt = ifelse(mlb$cook > 4/90, 1, 0)
s = mlb[which(mlb$inf_pt == 0), ]
model2 = lm(wins ~ scored + allowed, data = s)
summary(model2)
```

```
##
## Call:
## lm(formula = wins ~ scored + allowed, data = s)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9.4030 -2.4892  0.0895  2.8649  7.7582
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 80.010055   5.132124   15.59   <2e-16 ***
## scored       0.097709   0.004684   20.86   <2e-16 ***
## allowed     -0.096484   0.004067  -23.73   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.622 on 85 degrees of freedom
## Multiple R-squared:  0.9405, Adjusted R-squared:  0.9391
## F-statistic: 671.3 on 2 and 85 DF,  p-value: < 2.2e-16
```