4.2.1: R - Variations on Ordinary Least Squares (Weighted Least Squares, Robust Regression, Nonlinear Regression)

Stat 5100: Dr. Bean

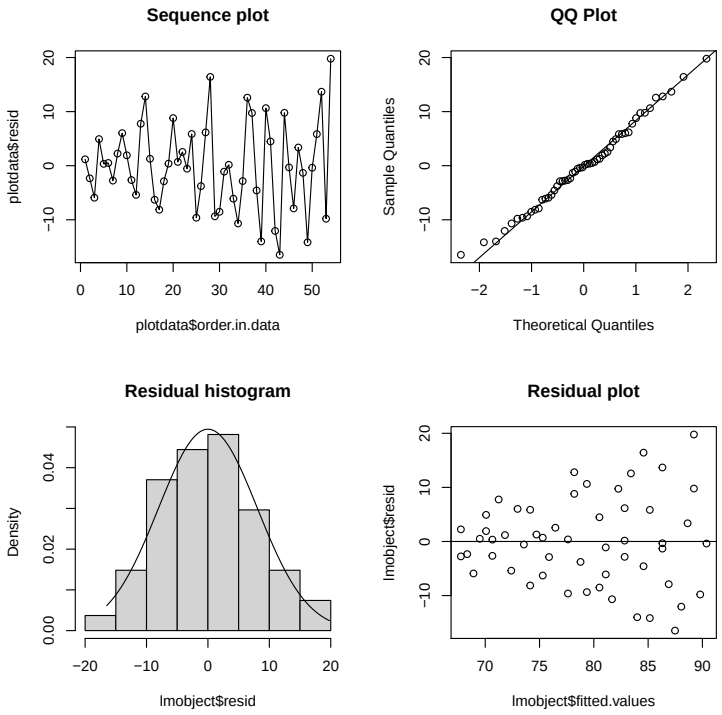
Example 1: Weighted least squares

A health researcher is interested in studying the relationship between diastolic blood pressure (bp) and age in adult women. Data are reported on 54 healthy adult women.

```
library(stat5100)
data(bpexample)

head(bpexample)

##   age bp
## 1  27 73
## 2  21 66
## 3  22 63
## 4  24 75
## 5  25 71
## 6  23 70
```

First, let's try fitting an OLS model and checking assumptions:

```
bp_lm <- lm(bp ~ age, data = bpexample)
summary(bp_lm)

##
## Call:
## lm(formula = bp ~ age, data = bpexample)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -16.4786  -5.7877  -0.0784   5.6117  19.7813
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 56.15693    3.99367  14.061  < 2e-16 ***
## age          0.58003    0.09695   5.983 2.05e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.146 on 52 degrees of freedom
## Multiple R-squared:  0.4077,Adjusted R-squared:  0.3963
## F-statistic: 35.79 on 1 and 52 DF,  p-value: 2.05e-07

stat5100::visual_assumptions(bp_lm)
```



Our summary tells us that our OLS model would be:

$$\hat{Y} = 0.58003X + 56.15693$$

where X is our age.

```
stat5100::brown_forsythe_lm(bp_lm)

## [1] "Brown-forsythe test for constant variance in the residuals:"
## [1] "T-statistic: -2.7855, p-value: 0.0074"

stat5100::cor_normality_lm(bp_lm)

## Correlation test of normality:
##                   resid expected_norm
## resid         1.0000000     0.9962478
## expected_norm 0.9962478     1.0000000
##
## Total observations: 54
## Make sure to consult with table B.6 for your final result.
```

Very clearly, it looks like there is nonconstant variance in the residuals. To resolve this, we will use weighted least squares. What we do is we estimate the absolute value of the residual at each value of X (age) with another regression model, and then using those estimates we can weight another model that predicts bp from age.

```
absolute_residuals <- abs(bp_lm$residuals)
abs_resid_lm <- lm(absolute_residuals ~ age,
                   data = data.frame(cbind(absolute_residuals,
                                           age = bpexample$age)))
predicted_abs_resid <- abs_resid_lm$fitted.values

# Fit a new linear model with the weights obtained form predicted_abs_resid:
```

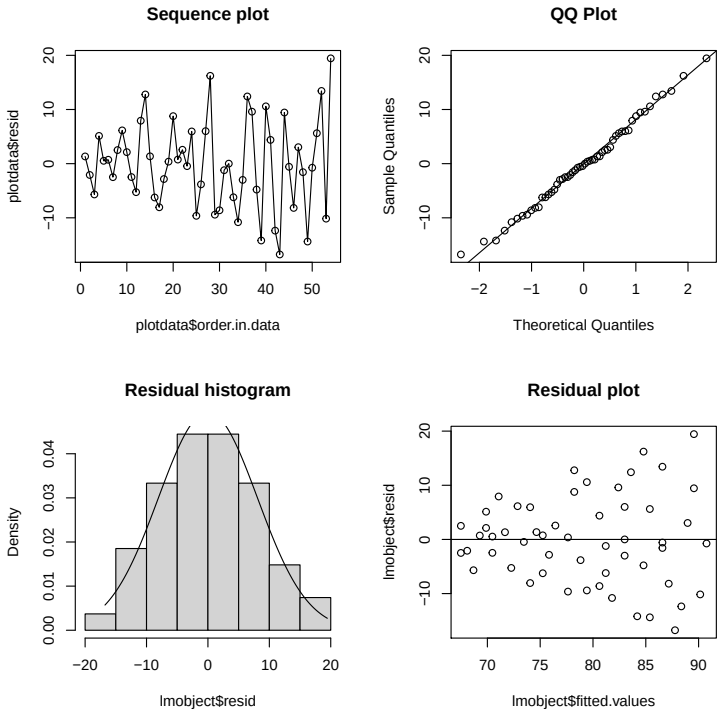
```r
weighted_bp_lm <- lm(bp ~ age, data = bpexample,
                     weights = 1 / predicted_abs_resid^2)
summary(weighted_bp_lm)

##
## Call:
## lm(formula = bp ~ age, data = bpexample, weights = 1/predicted_abs_resid^2)
##
## Weighted Residuals:
##     Min      1Q  Median      3Q     Max
## -2.0230 -0.9939 -0.0327  0.9250  2.2008
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 55.56577    2.52092  22.042  < 2e-16 ***
## age          0.59634    0.07924   7.526 7.19e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.213 on 52 degrees of freedom
## Multiple R-squared:  0.5214,Adjusted R-squared:  0.5122
## F-statistic: 56.64 on 1 and 52 DF,  p-value: 7.187e-10
```

Based upon the above result, our weighted least squares model would be:

$$\hat{Y} = 0.59634X + 55.56577$$

where X is the age. Notice below that if we look at the visual assumptions of our weighted model, it will look much the same as the original model. In this case, even though weighted least squares is "better" because our data violates the assumption of nonconstant variance in the residuals, the model ends up being incredibly similar to the OLS model, both in terms of visual assumption checks and in the β coefficient estimates.

```r
stat5100::visual_assumptions(weighted_bp_lm)
```



Example 2: Iteratively reweighted least squares (IRLS) on Toluca dataset

As part of a cost improvement program, the Toluca company wished to better understand the relationship between the lot size (X) and the total work hours (Y).

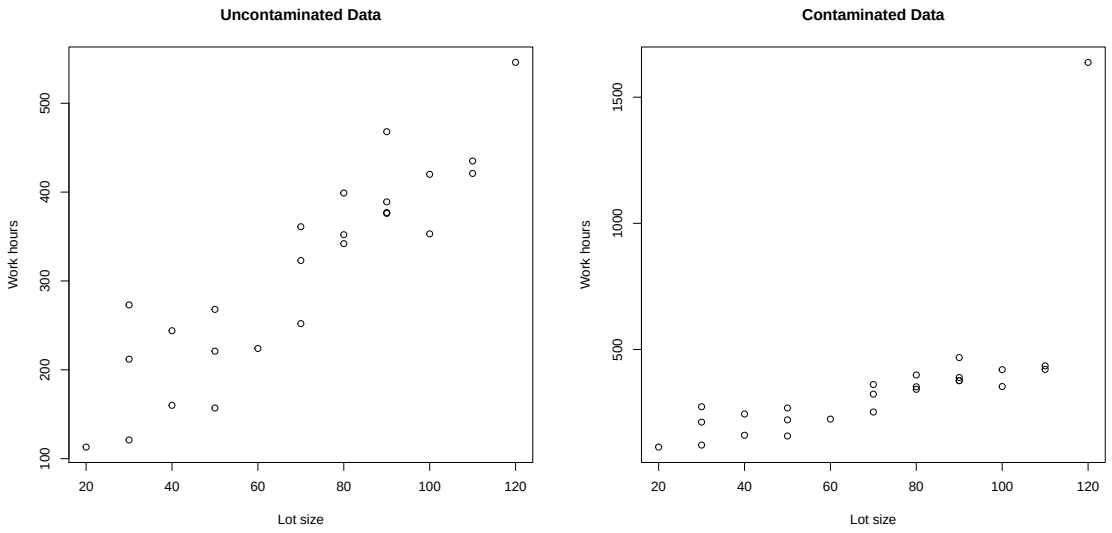
```
data(toluca)
head(toluca)

##   lotsize workhours
## 1      80       399
## 2      30       121
## 3      50       221
## 4      90       376
## 5      70       361
## 6      60       224
```

Previously, OLS regression worked nicely on this dataset. However, to illustrate the usefulness of robust regression, we will intentionally contaminate the Toluca dataset with some outliers. We will eventually notice that fitting an OLS model on the contaminated data gives a very different model, but fitting a robust regression model on the contaminated data will give a very similar model to the OLS model fitted on the clean dataset.

```
# Contaminate the data by tripling all the workhours if workhours is above 500
toluca_c <- toluca
toluca_c$workhours[toluca_c$workhours > 500] <-
  3*toluca_c$workhours[toluca_c$workhours > 500]
```

Let's look at scatterplots of the contaminated and uncontaminated data:

```
plot(toluca$lotsize, toluca$workhours, xlab = "Lot size", ylab = "Work hours",
     main = "Uncontaminated Data")
plot(toluca_c$lotsize, toluca_c$workhours, xlab = "Lot size", ylab = "Work hours",
     main = "Contaminated Data")
```

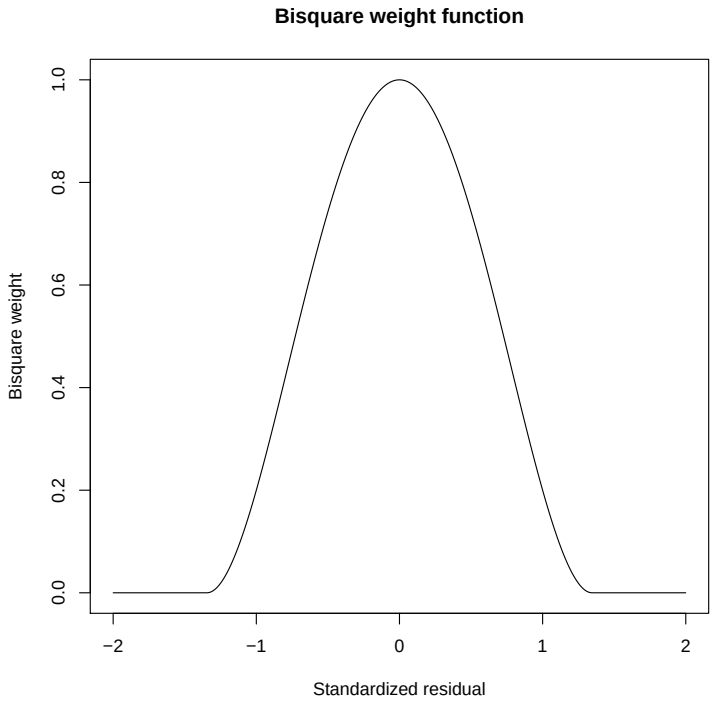


Now, we are going to use robust regression using Tukey's Bisquare weight. From our notes, the bisquare weighting function looks like

$$w(u) = \begin{cases} \left(1 - \left(\frac{u}{c}\right)^2\right)^2 & \text{if } |u| \leq c \\ 0 & \text{otherwise} \end{cases}$$

where u is a standardized residual. Let's take a look at what the bisquare weight looks like for $c = 1.345$:

```r
c <- 1.345
u <- seq(-2, 2, by = 0.01)
w <- (1 - (u/c)^2)^2
w[abs(u) >= c] <- 0

plot(u, w, type = "l", main = "Bisquare weight function",
     xlab = "Standardized residual", ylab = "Bisquare weight")
```



Now, let's fit the following models:

1. OLS regression on original data
2. OLS regression on contaminated data
3. Robust regression with bisquare weight on contaminated data

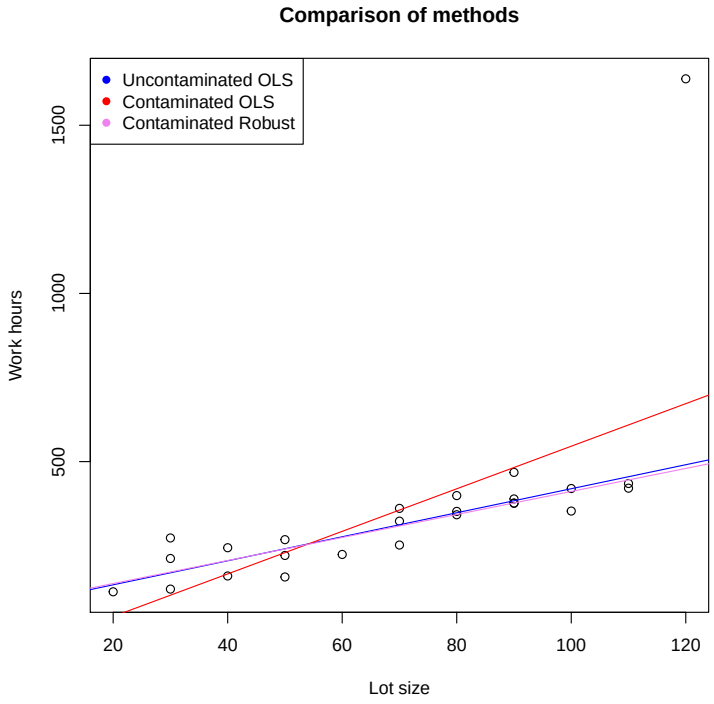
```r
# Model 1 (OLS on original data):
toluca_ols <- lm(workhours ~ lotsize, data = toluca)
summary(toluca_ols)

##
## Call:
## lm(formula = workhours ~ lotsize, data = toluca)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -83.876 -34.088  -5.982  38.826 103.528
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   62.366     26.177   2.382   0.0259 *
## lotsize        3.570      0.347  10.290 4.45e-10 ***
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 48.82 on 23 degrees of freedom
## Multiple R-squared:  0.8215,Adjusted R-squared:  0.8138
## F-statistic: 105.9 on 1 and 23 DF,  p-value: 4.449e-10

# Model 2 (OLS on contaminated data):
toluca_c_ols <- lm(workhours ~ lotsize, data = toluca_c)
summary(toluca_c_ols)

##
## Call:
## lm(formula = workhours ~ lotsize, data = toluca_c)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -192.79 -103.96  -32.96   18.15  965.65
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -86.984    120.908  -0.719 0.479126
## lotsize        6.328      1.603   3.948 0.000639 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 225.5 on 23 degrees of freedom
## Multiple R-squared:  0.404,Adjusted R-squared:  0.3781
## F-statistic: 15.59 on 1 and 23 DF,  p-value: 0.0006393

# Model 3 (Robust regression on contaminated data)
toluca_c_robust <- MASS::rlm(workhours ~ lotsize, data = toluca_c,
                             method = c("MM"))
summary(toluca_c_robust)

##
## Call: rlm(formula = workhours ~ lotsize, data = toluca_c, method = c("MM"))
## Residuals:
##        Min         1Q     Median         3Q        Max
##  -83.19265  -24.53354   -0.07146   38.02706 1158.26944
##
## Coefficients:
##             Value   Std. Error t value
## (Intercept) 69.0941 27.4773     2.5146
## lotsize      3.4220  0.3642     9.3958
##
## Residual standard error: 54.72 on 23 degrees of freedom
```

Let us now plot all three regression lines on the contaminated dataset.

```
# First, get coefficient estimates from the models.
orig_coef <- toluca_ols$coefficients
contam_coef <- toluca_c_ols$coefficients
robust_coef <- toluca_c_robust$coefficients

plot(toluca_c$lotsize, toluca_c$workhours, main = "Comparison of methods",
     xlab = "Lot size", ylab = "Work hours")
abline(a = orig_coef[1], b = orig_coef[2], col = "blue")
```

```r
abline(a = contam_coef[1], b = contam_coef[2], col = "red")
abline(a = robust_coef[1], b = robust_coef[2], col = "violet")
legend("topleft", legend = c("Uncontaminated OLS", "Contaminated OLS", "Contaminated Robust")
       , col = c("blue", "red", "violet"), pch = 16)
```



In the above plot, note that the contaminated robust line is essentially the same as the uncontaminated OLS line. The contaminated OLS line is clearly different, primarily because it is getting pulled upward so much by the outlier point in the contaminated dataset.

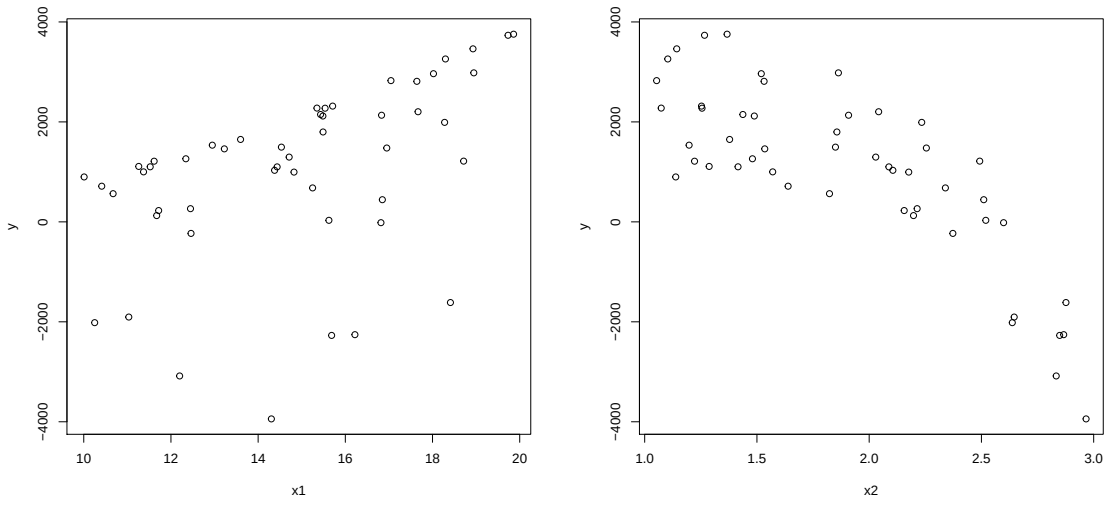
Example 3.1: Nonlinear regression

Here, we will assume a model form of the following:

$$Y = \beta_0 + \beta_1 X_1^{\beta_2} - \beta_3 e^{\beta_4 X_2} + \epsilon$$

We will try our nonlinear regression model on some randomly generated data:

```
# Set a random seed
set.seed(141414)

x1 <- 10 + 10*runif(n = 50)
x2 <- 1 + 2*runif(n = 50)
error <- 10*rnorm(n = 50)

# Define a true relationship with Y: (beta0 = 50, beta1 = 10, beta2 = 2,
# beta3 = 16, beta4 = 2)
y <- 50 + 10*x1^2 - 16*exp(2*x2) + error

# Now look at scatterplots of X1 and Y and X2 and Y:
plot(x1, y)
plot(x2, y)
```




```
# Train our nonlinear regression model
simulated_nls <- nls(y ~ b0 + b1*x1^b2 - b3*exp(b4*x2),
                     start = list(b0 = 100, b1 = 8, b2 = 3, b3 = -20, b4 = 4))
summary(simulated_nls)

##
## Formula: y ~ b0 + b1 * x1^b2 - b3 * exp(b4 * x2)
##
## Parameters:
##     Estimate Std. Error t value Pr(>|t|)
## b0 68.362945  22.559272    3.03  0.00404 **
## b1  9.506840   0.652230   14.58  < 2e-16 ***
## b2  2.014894   0.021366   94.30  < 2e-16 ***
## b3 15.773176   0.234294   67.32  < 2e-16 ***
## b4  2.004962   0.005068  395.60  < 2e-16 ***
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 10.66 on 45 degrees of freedom
##
## Number of iterations to convergence: 17
## Achieved convergence tolerance: 8.145e-08
```

Based upon the above summary, our final function guess would be the following:

$$\hat{Y} = 68.363 + 9.507X_1^{2.015} - 15.773e^{2.005X_2}$$

Compare this with the true equation form:

$$Y = 50 + 10X_1^2 - 16e^{2X_2} + \epsilon$$