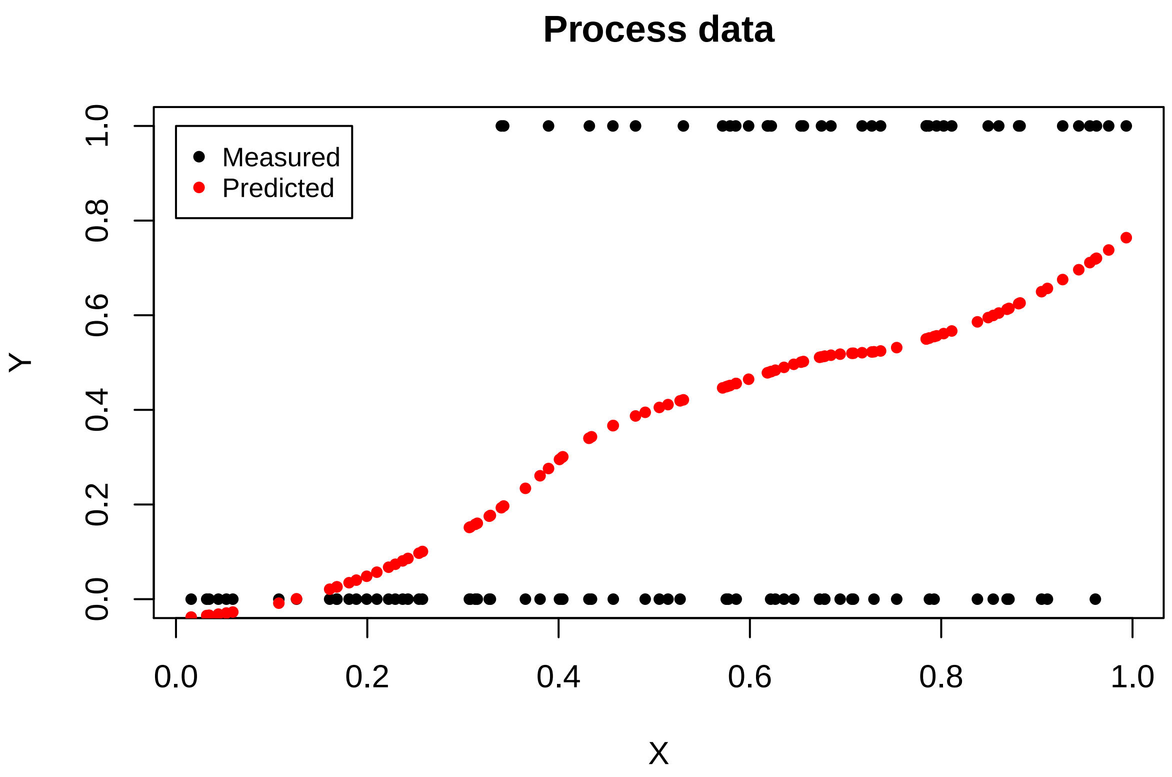
732A90 - Exam - March 2016 - Task 2

Assignment 2

2.1 LOESS model

As we can see in the plot we are

```r
process <- read.csv2("../data/process.csv")

# Fit loess model and predict the data points
fit <- loess(Y ~ X, data = process)
predicted_values <- predict(fit, newdata = process$X)

# Plot the actual data and the predicted points
plot(process$X, process$Y,
     pch = 16, cex = 0.8,
     xlab = "X", ylab = "Y",
     main = "Process data")
points(process$X, predicted_values,
       pch = 16, cex = 0.8,
       col = "red")
legend(x = 0, y = 1,
       legend = c("Measured", "Predicted"),
       col = c("black", "red"),
       pch = c(16, 16),
       cex = 0.8)
```



2.2 BFGS method

First we attach an additional column with the values of $Z = x^2$ to the data set:

```r
library(dplyr)
process %>%
  mutate(Z = X^2) ->
process
```

The likelihood and the log-likehood can be derived by:

$$L(X \mid p, Y) = \prod_{i+1}^{n} p_i^{Y_i} (1 - p_i)^{1-Y_i} \tag{1}$$

$$LL(X \mid p, Y) = \log\left(\prod_{i+1}^{n} p_i^{Y_i} (1 - p_i)^{1-Y_i}\right) \tag{2}$$

$$= \sum_{i=1}^{n} Y_i \log(p_i) + (1 - Y_i) \log(1 - p_i) \tag{3}$$

So the negative log-likelihood is:

$$-LL(X \mid p, Y) = -\sum_{i=1}^{n} Y_i \log(p_i) + (1 - Y_i) \log(1 - p_i) \tag{4}$$

with p_i:

$$p_i = max(0.01, min(0.99, w_1 X_i + w_2 X_i^2)), \quad with\ X_i^2 = Z_i \tag{5}$$

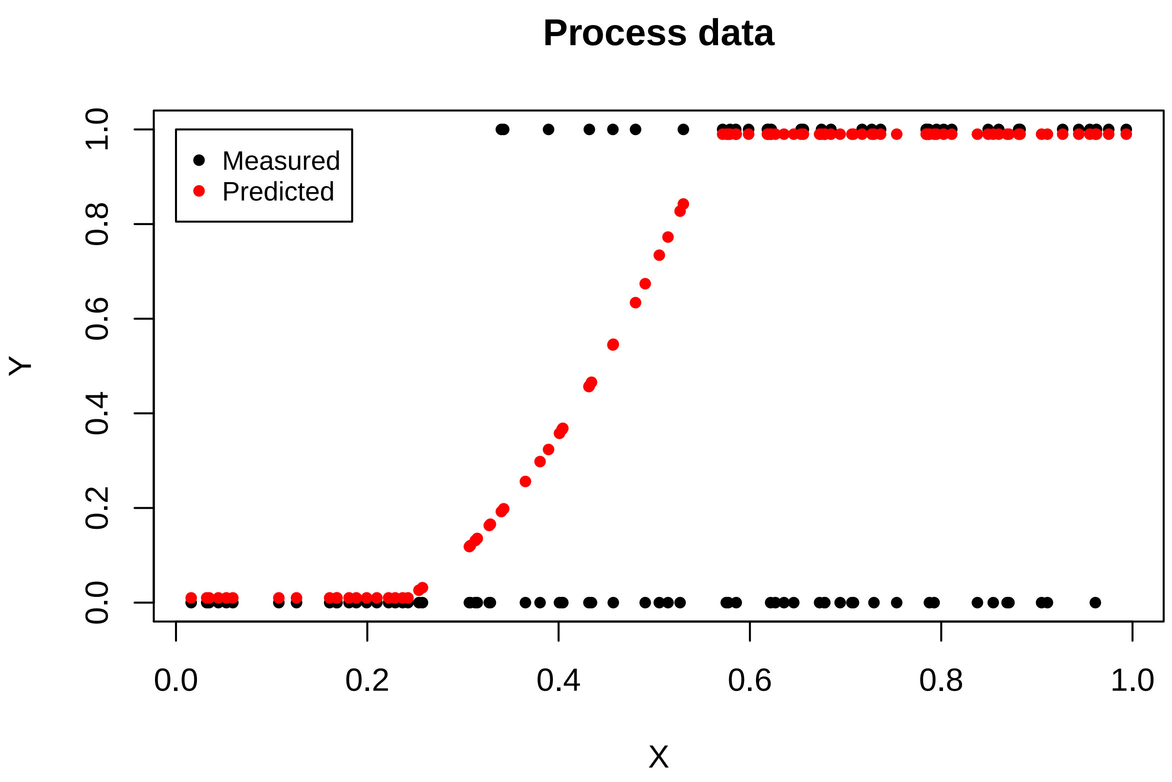
```r
minus_ll <- function(theta, X, Y, Z) {
  w1 <- theta[1]
  w2 <- theta[2]
  n <- length(X)

  p <- vector("numeric", length = n)
  for (i in 1:n) {
    p[i] <- max(0.01, min(0.99, w1*X[i] + w2*Z[i]))
  }

  ll <- sum(Y * log(p) + (1-Y) * (1-p))

  return(-ll)
}

X <- process$X
Y <- process$Y
Z <- process$Z

bfgs_a <- optim(par = c(0.1, 0.1),
                fn = minus_ll,
                method = "BFGS",
                X = X, Y = Y, Z = Z)
bfgs_b <- optim(par = c(0.3, 0.3),
                fn = minus_ll,
                method = "BFGS",
                X = X, Y = Y, Z = Z)
bfgs_c <- optim(par = c(0.9, 0.1),
                fn = minus_ll,
                method = "BFGS",
                X = X, Y = Y, Z = Z)
```

The value of the convergence attribute is 0 in all cases which indicates that the algorithm converged.

Initializing the BFGS algorithm wit option b and c will give us a value of the negative log-likellihood of ≈ -25.32. This is far better than the value of option a which is ≈ -0.26. In this specific case one would favour option c because fewer iterations were needed to get the same result. Supplying a gradient might even lead to better results / fewer iterations.

2.3 Plot the data with optimal parameters

```
w1 <- bfgs_c$par[1]
w2 <- bfgs_c$par[2]

fitted_values <- function(x, z, w1, w2) {
  n <- length(x)
  result <- vector("numeric", length = n)

  for (i in 1:n) {
    result[i] <- max(0.01, min(0.99, w1*x[i] + w2*z[i]))
  }

  return(result)
}

y_hat <- fitted_values(process$X, process$Z, w1, w2)

# Plot the actual data and the predicted points
plot(process$X, process$Y,
     pch = 16, cex = 0.8,
     xlab = "X", ylab = "Y",
     main = "Process data")
points(process$X, y_hat,
       pch = 16, cex = 0.8,
       col = "red")
legend(x = 0, y = 1,
       legend = c("Measured", "Predicted"),
       col = c("black", "red"),
       pch = c(16, 16),
       cex = 0.8)
```



2.4 Permutation test

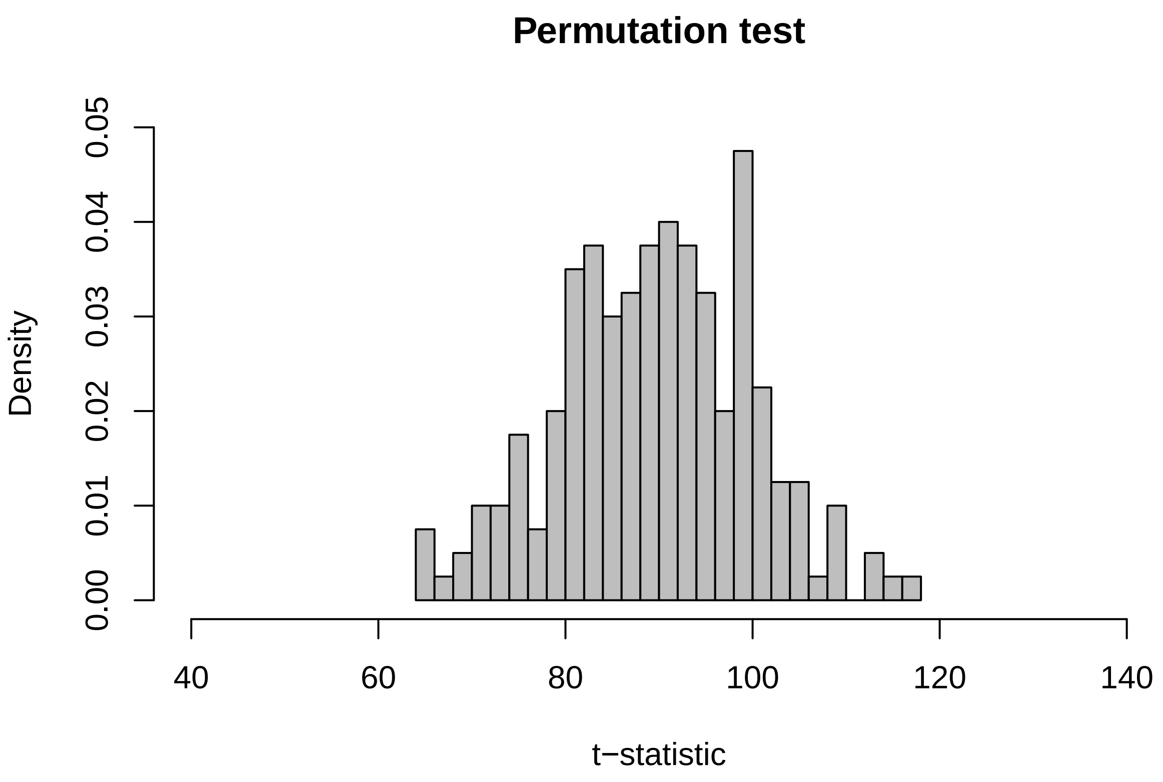
We implement a permutation test by permuting the Z feature 200 times.

I AM ABSOLUTELY NOT SURE ABOUT THIS!!!: With the permutation test we want to test if the feature Z should be included in our model or not. If w_2 is set to 0 the effect of the feature Z is 0. By permuting the order of the Z feature we want to test the effect of the permutations on the model. Accepting H_0 means that Z does not affect the results.

The true value of the test statistic is 133.3071572. No t-statistic of the permuted samples is bigger / more extreme than the true test statistic. Hence the p-value of the test is 0.

```r
B <- 200
n <- nrow(process)
result <- vector("numeric", length = n)

for (i in 1:B) {
  idx <- sample(1:n, n, replace = FALSE)
  data <- data.frame(Y = process$Y,
                     X = process$X,
                     Z = process[idx, "Z"])

  result[i] <- optim(par = c(0.1, 0.1),
                     fn = minus_ll,
                     method = "BFGS",
                     X = data$X,
                     Y = data$Y,
                     Z = data$Z)$par[2]

}
```

```r
hist(result, breaks = 20, freq = FALSE,
     col = "grey",
     xlim = c(40, 140), ylim = c(0, 0.05),
     main = "Permutation test",
     xlab = "t-statistic", ylab = "Density")
```



```r
sum(result > bfgs_a$par[2])
```

```
## [1] 0
```