Grp 5217b stat handin 2

1. Fit a first and a second order polynomial to the data and plot both in the same plot.

```r
data <- data.frame(x = 1:10, y = c(5.52127567436017,
                                   6.85340682052011,
                                   10.0201614384284,
                                   13.3112793929856,
                                   14.2698633593385,
                                   12.6786166068488,
                                   18.5461642188697,
                                   18.0289396423979,
                                   23.0282117434266,
                                   21.8451069217528))

mod0 <- lm(y ~ 1, data);summary(mod0)
```

```
##
## Call:
## lm(formula = y ~ 1, data = data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.8890 -3.7255 -0.6197  4.0066  8.6179
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   14.410      1.884    7.65 3.16e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5.957 on 9 degrees of freedom
```

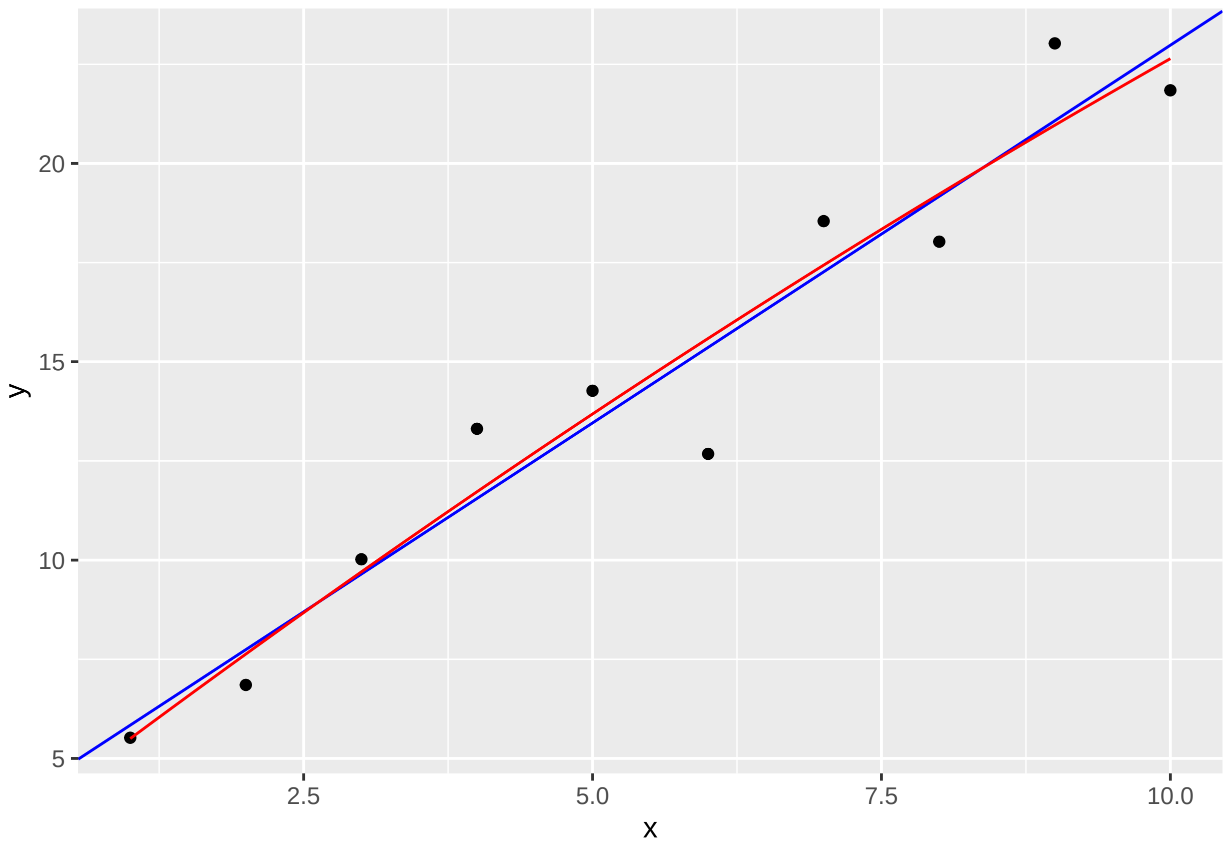
```r
mod1 <- lm(y ~ x, data); summary(mod1)
```

```
##
## Call:
## lm(formula = y ~ x, data = data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.68410 -1.07517  0.02729  1.16197  1.95104
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   3.9338     1.0809   3.639  0.00659 **
## x             1.9048     0.1742  10.935 4.34e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.582 on 8 degrees of freedom
## Multiple R-squared:  0.9373, Adjusted R-squared:  0.9295
## F-statistic: 119.6 on 1 and 8 DF,  p-value: 4.339e-06
```

```
mod2 <- lm(y ~ x + I(x^2), data); summary(mod2)
```

```
##
## Call:
## lm(formula = y ~ x + I(x^2), data = data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.9100 -0.7928  0.1685  0.9784  2.0640
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.31264    1.96841   1.683   0.1363
## x            2.21540    0.82209   2.695   0.0309 *
## I(x^2)      -0.02823    0.07283  -0.388   0.7098
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.674 on 7 degrees of freedom
## Multiple R-squared:  0.9386, Adjusted R-squared:  0.9211
## F-statistic: 53.51 on 2 and 7 DF,  p-value: 5.733e-05
```

```
mod2_fun <- function(x) {
  mod2$coefficients[1] + mod2$coefficients[2] * x + mod2$coefficients[3] * x^2
}
ggplot(data, aes(x,y)) +
  geom_point() +
  geom_abline(intercept = mod1$coefficients[1], slope = mod1$coefficients[2], color = "blue") +
  stat_function(fun = mod2_fun, color = "red")
```



2. Extract the estimates of beta for both models, and test for each parameter whether it can be omitted from the model (formulate precise hypotheses H0 and H1). Is the additional parameter in the second order polynomial relevant?

```
#estimat for beta i linear model
mod1$coefficients[2]
```

```
##        x
## 1.904821
```

```
#estimat for beta i polynomial model
mod2$coefficients[2:3]
```

```
##           x      I(x^2)
##  2.21539649 -0.02823416
```

kan model 2 reduceres til model 1? : H0: beta_2 = 0, H1: beta_2 |= 0, ved alpha = 0.05

```
anova(mod0,mod1)
```

```
## Analysis of Variance Table
##
## Model 1: y ~ 1
## Model 2: y ~ x
##   Res.Df    RSS Df Sum of Sq      F    Pr(>F)
## 1      9 319.37
## 2      8  20.03  1    299.34 119.57 4.339e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
anova(mod1,mod2)
```

```
## Analysis of Variance Table
##
## Model 1: y ~ x
## Model 2: y ~ x + I(x^2)
##   Res.Df    RSS Df Sum of Sq      F Pr(>F)
## 1      8 20.027
## 2      7 19.607  1    0.4209 0.1503 0.7098
```

da Pr(>F) > 0.05 accepteres H0 ved 5% signifikans niveau.

3. For both models calculate both the coefficient of determination and the adjusted coefficient of determination. Interpret all four values.

```r
summary(mod1)$r.squared # R^2 og R_adj for mod 1
```

```
## [1] 0.9372898
```

```r
summary(mod1)$adj.r.squared
```

```
## [1] 0.9294511
```

```r
summary(mod2)$r.squared # R^2 og R_adj for mod 2
```

```
## [1] 0.9386078
```

```r
summary(mod2)$adj.r.squared
```

```
## [1] 0.9210671
```

4. Make confidence intervals for both of the parameters in beta in the first order polynomial. Relate these to the hypotheses specified in Exercise 2.2. (Notice: from this exercise and onwards we are skipping the second order polynomial.)

```r
confint(mod1)
```

```
##                2.5 %   97.5 %
## (Intercept) 1.441309 6.426268
## x           1.503121 2.306520
```

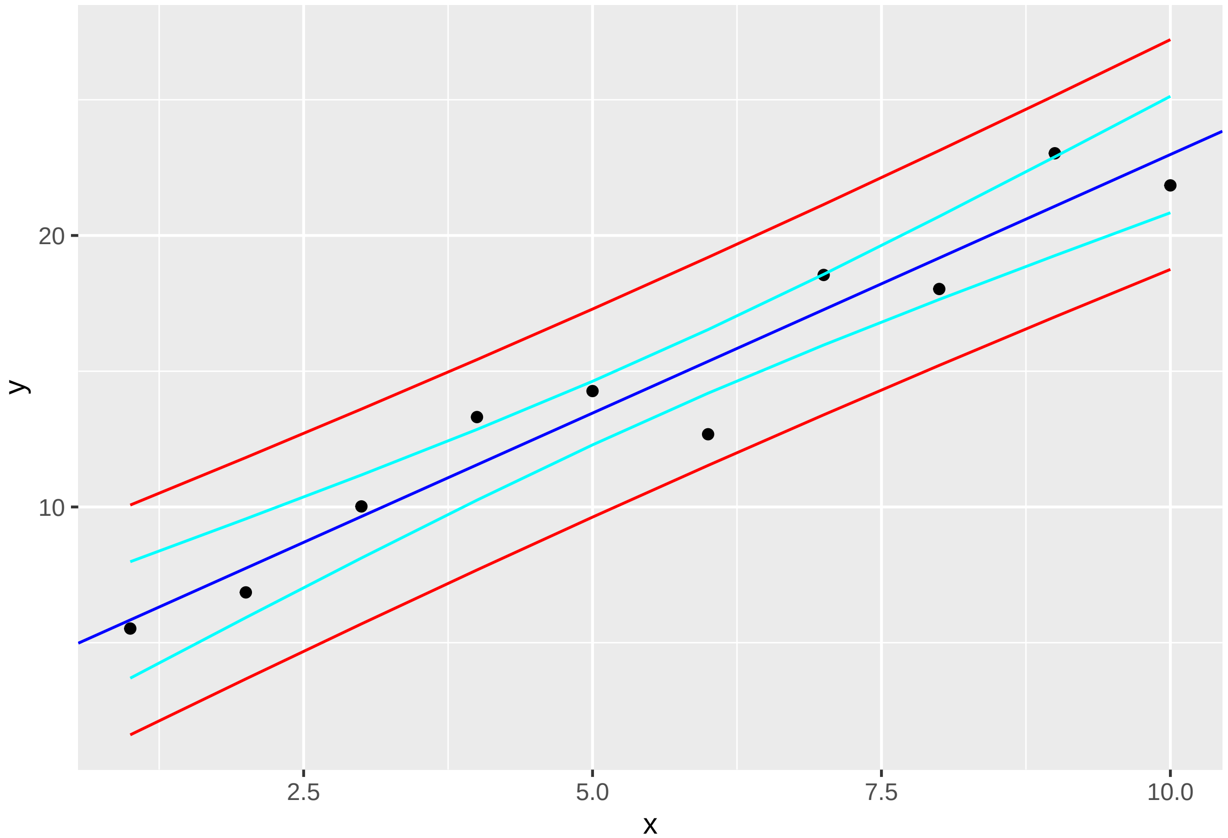
```r
confint(mod2)
```

```
##                  2.5 %    97.5 %
## (Intercept) -1.3419096 7.9671839
## x            0.2714615 4.1593314
## I(x^2)      -0.2004595 0.1439912
```

da konfidens intervallet for beta_2 i mod 2 ligger omkring 0 støtter det hypotesen fra tidligere om at beta_2 = 0 da konfidens intervallet for beta_1 i mod 1 ikke ligger omkring 0, så viser det at beta_1 ikke er lig 0 og altså er mod1 bedre end mod0

5. Make confidence and prediction bounds for a self-chosen range of x-values in the first order polynomial. Plot these on top of the data, and interpret both bounds

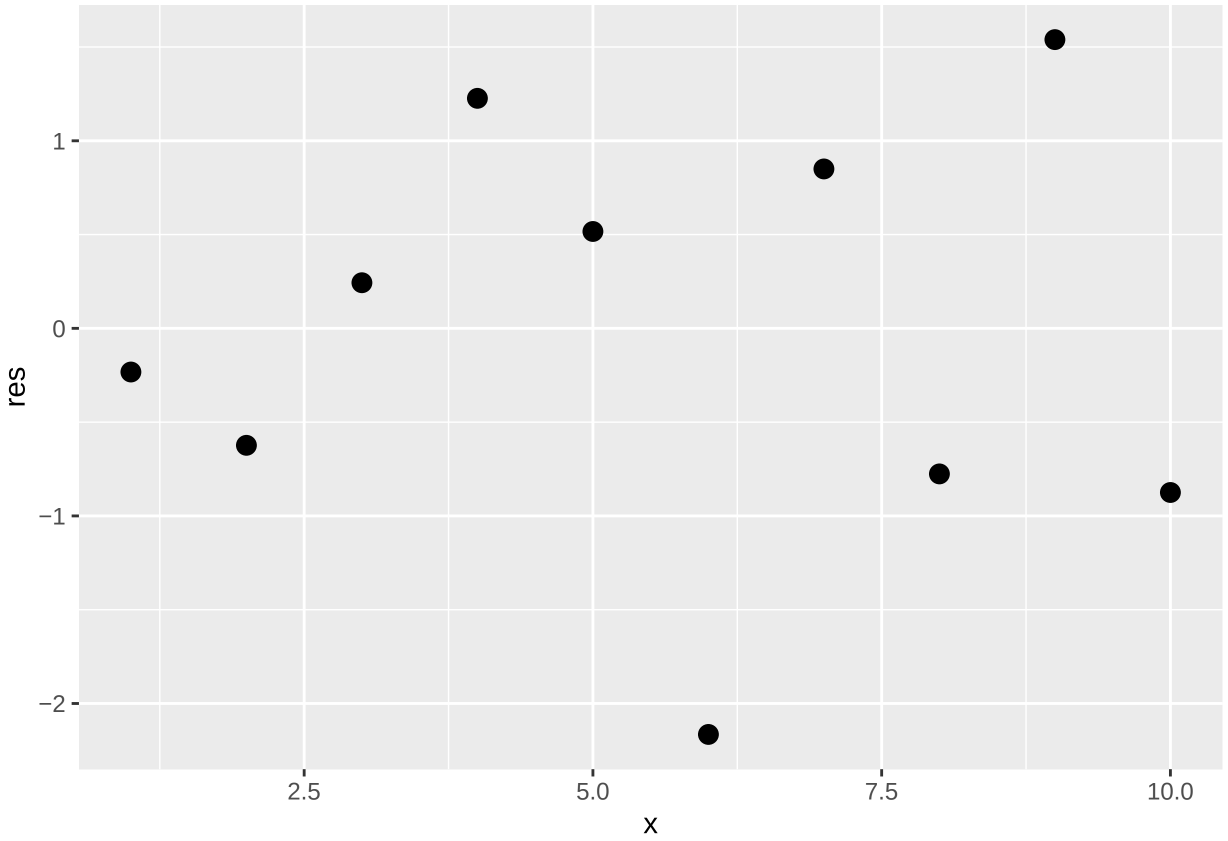
```r
x <- data.frame(x = 1:10)
yc <- predict(mod1, x, interval = "confidence")
ypi <- predict(mod1, x, interval = "predict")

ggplot(data, aes(x,y)) +
  geom_point() +
  geom_abline(intercept = mod1$coefficients[1], slope = mod1$coefficients[2], color = "blue") +
  geom_line(aes(x, yc[,2]), color = "cyan") +
  geom_line(aes(x, yc[,3]), color = "cyan") +
  geom_line(aes(x, ypi[,2]), color = "red") +
  geom_line(aes(x, ypi[,3]), color = "red")
```



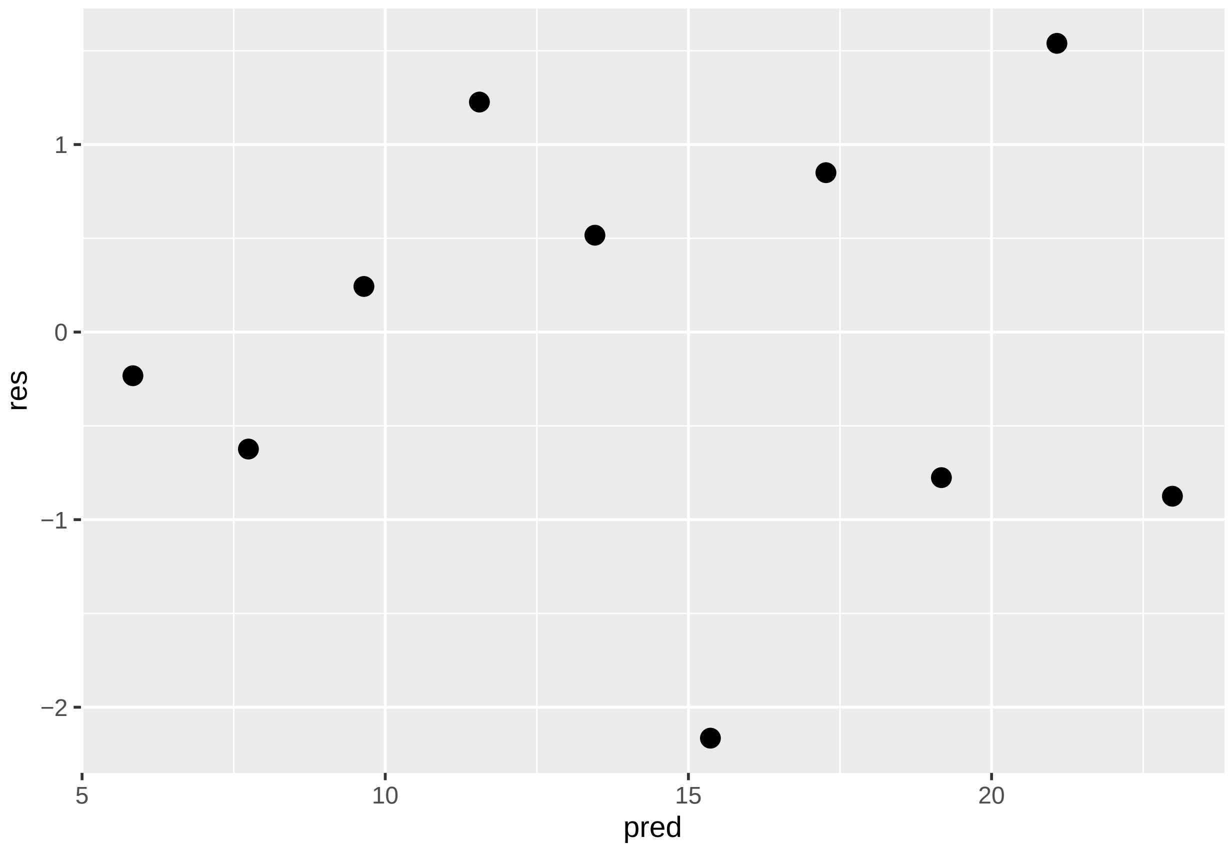
Da n= 10 gælder den asymptotiske opførsel for konfidensintervallet ikke, hvilket også afspejles i plottet ved den lyseblå linje, da den ikke er tæt ved den regresseret model. Som forventet er prædiktions intervallet større end confidens intervallet.

6. Calculate residuals for the first order model (you may choose which kind of residuals). Create residual plots, and use these to check the assumptions behind using a general linear model on this data.

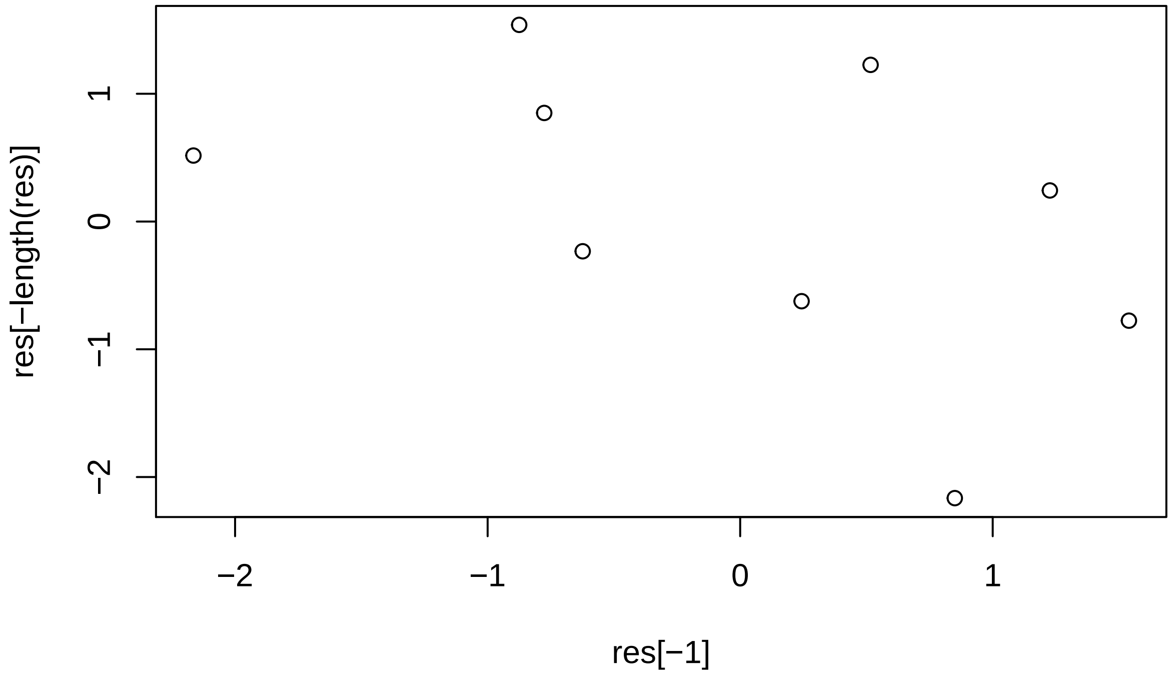
```
res <- rstudent(mod1)
pred <- predict(mod1)
y <- data$y

ggplot(cbind(data, res), aes(x, res)) +
  geom_point(size = 3)
```



```
ggplot(cbind(data, pred), aes(pred, res)) +
  geom_point(size = 3)
```



```r
plot(res[-1],res[-length(res)]) #tjek for uafhængighed af residualer
```

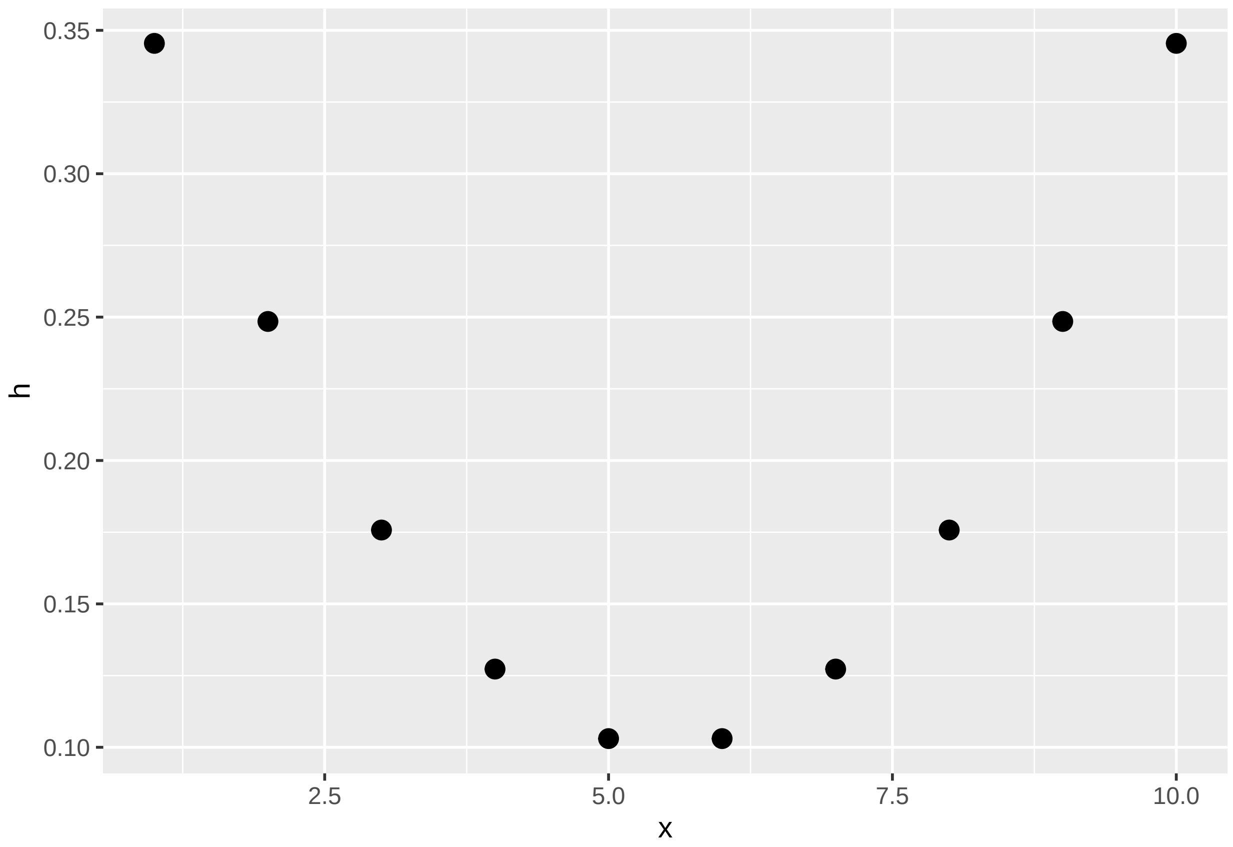


Antallet af data punkter gør at de to plots ikke viser noget særligt tydeligt, de skulle gerne vise punkterne vilkårligt fordelt omkring 0, dog kan det både ligne trumpet form(stigende varians), eller at de stiger affint(stigende mean). Hvis man mente det lignte en trumpet form så ville man kunne transformere Y evt. ved en logaritme.

Plottet som viser sammenhængen mellem residualet og det forrige residuale viser ikke at residualerne er afhængige.

7. Calculate leverages for the first order model. For which xi do the measurements yi influence the estimates the most?

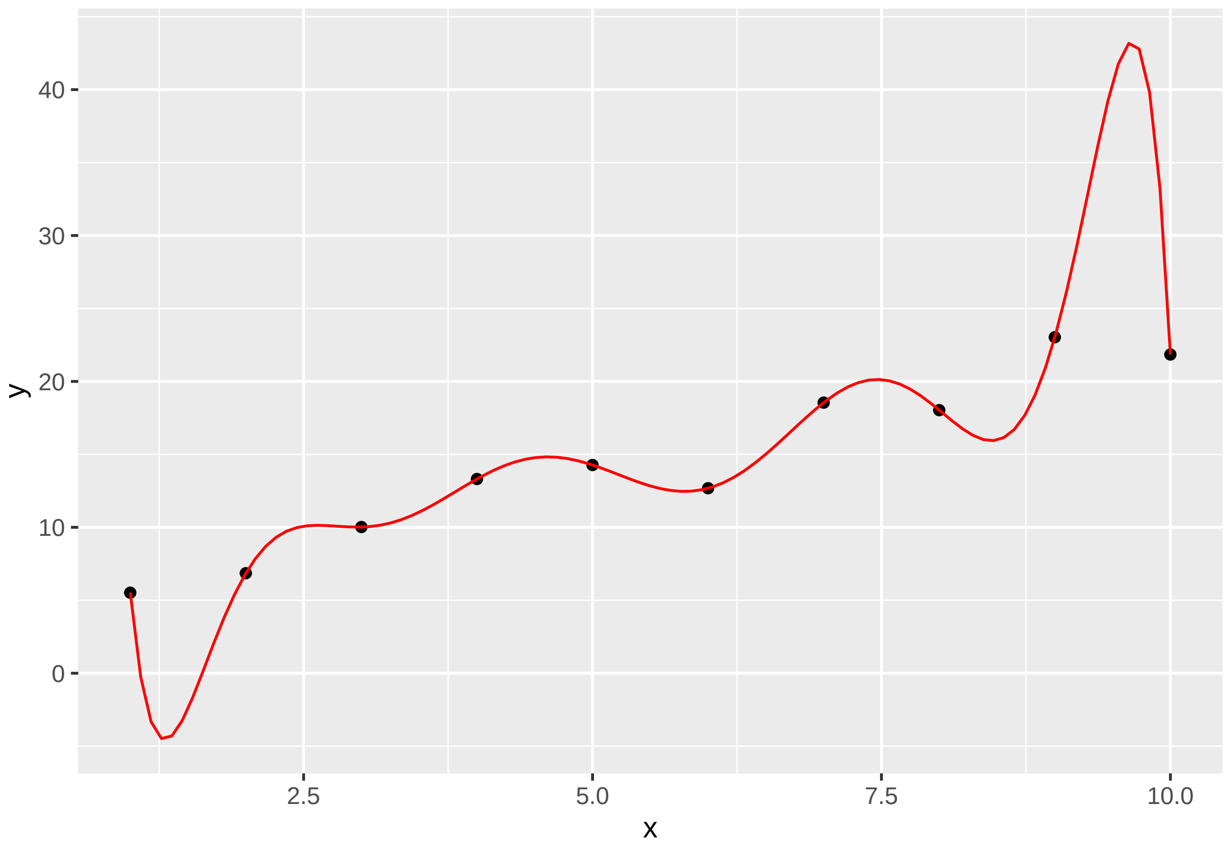
```
h <- hatvalues(mod1)

ggplot(cbind(data, h), aes(x, h)) +
  geom_point(size = 3)
```



```
# leveragen er hoejst i starten og slutningen
```

8. Fit a ninth order polynomial to the data and plot this on top of the data. What happens? Calculate the coefficient of determination. Is this model better at describing the data than the first and second order polynomials?

```
mod9 <- lm(y ~ poly(x, 9, raw = TRUE), data)
cof9 <- coef(mod9)
mod9_fun <- function(x) {
  cof9[1] + x * cof9[2] + x^2 * cof9[3] + x^3 * cof9[4] +
    x^4 * cof9[5] + x^5 * cof9[6]+   x^6 * cof9[7] + x^7 *
    cof9[8] + x^8 * cof9[9] + x^9 * cof9[10]
}

ggplot(data, aes(x,y)) +
  geom_point() +
  stat_function(fun = mod9_fun, color = "red")
```



```r
summary(mod9)$r.squared
```

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

```r
summary(mod9)$adj.r.squared
```

```
## [1] NaN
```

```r
#tjek prædikation
mod9_fun(11)
```

```
## (Intercept)
##   -1141.241
```

modellen rammer alle punkterne fuldstændig, men ville ikke være god til at prædikere nye værdier som vist hvor 11 er sat in i funktionen og giver -1141.241. Yderligere giver justeret R ikke noget fordi vi har samme antal observationer som parametre.