We first initialize the `tidyverse` library that contains `dplyr` and `ggplot2`.

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --
```

```
## v ggplot2 3.3.6     v purrr   0.3.4
## v tibble  3.1.7     v dplyr   1.0.9
## v tidyr   1.2.0     v stringr 1.4.0
## v readr   2.1.2     v forcats 0.5.1
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

Let's import the data.

```
setwd("/Users/mathieu/Lab/company-electricity-statistical-analysis/src/")
df <- read.csv2(file = "electricity_consumption_dataset.txt", sep = ";", dec = ".")
head(df)
```

```
##           Y         X
## 1 16.290510  4.667310
## 2 11.572919  5.048043
## 3 20.772148  6.120283
## 4 20.732848 13.202527
## 5  9.426162  4.476845
## 6 21.584711 12.548479
```

This data contains 2 variables: **X: Electricity consumption (MWh)** and **Y = Productivity (1000€ / day)**.
We rename the variables accordingly.

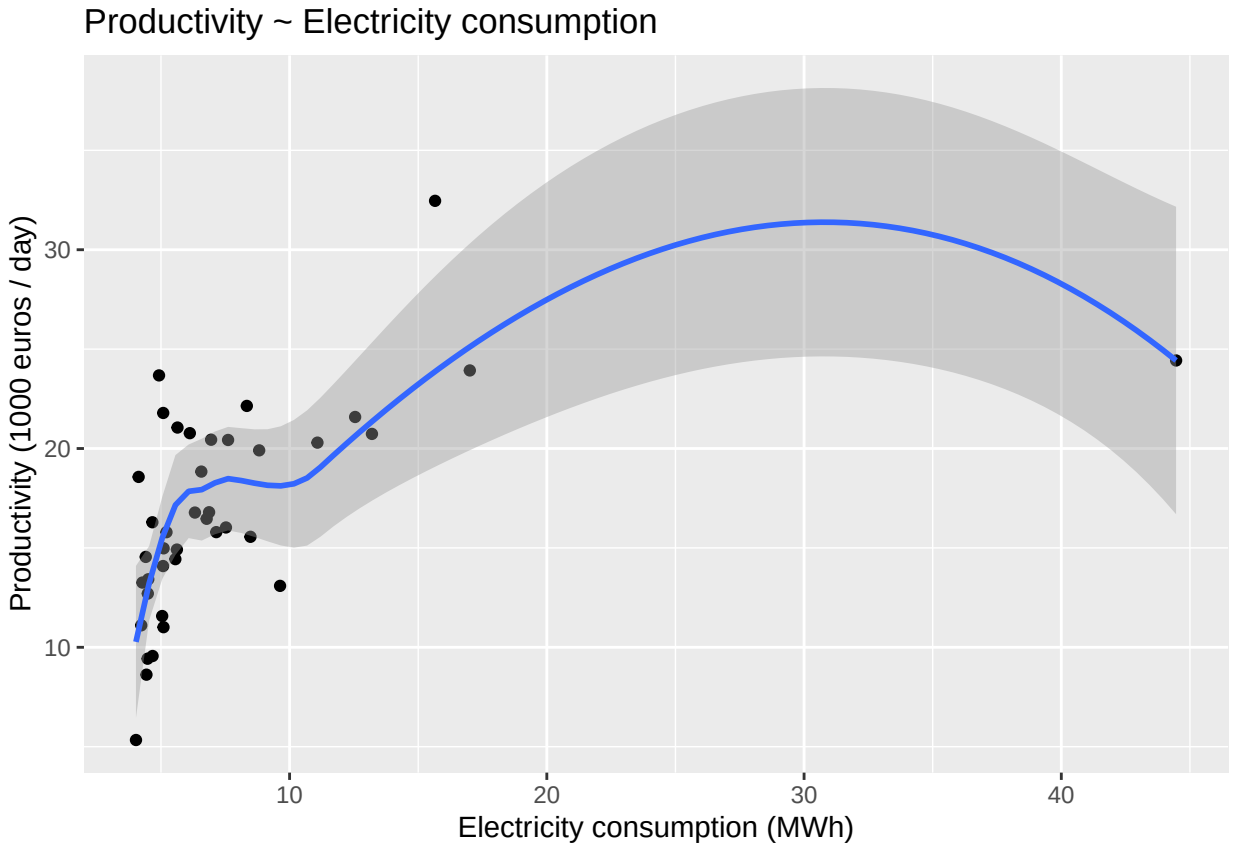
```
df <- rename(df, electricity_consumption = X, productivity = Y)
```

1. Fitting naively a linear model.

We want to fit a linear model on our data. Is there a linear relationship between the **electricity consumption** and the **productivity** ? We can plot a *scatter plot* in order to visualize the relationship between these 2 variables.

```
ggplot(df, aes(x = electricity_consumption, y = productivity)) +
  geom_jitter() +
  geom_smooth() +
  ggtitle("Productivity ~ Electricity consumption") +
  xlab("Electricity consumption (MWh)") +
  ylab("Productivity (1000 euros / day)")
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



As we can see, this does not seem linear at all. We can still try to fit a linear model.
We use the **ordinary least square (OLS)** method that consists in minimising the sum of the square of the error:

$$\text{SSE} = \sum_{i=0}^{+n} (y_i - \hat{y}_i)^2$$

We can do that easily in r

```
simple_lm <- lm(productivity ~ electricity_consumption, data = df)
summary(simple_lm)
```

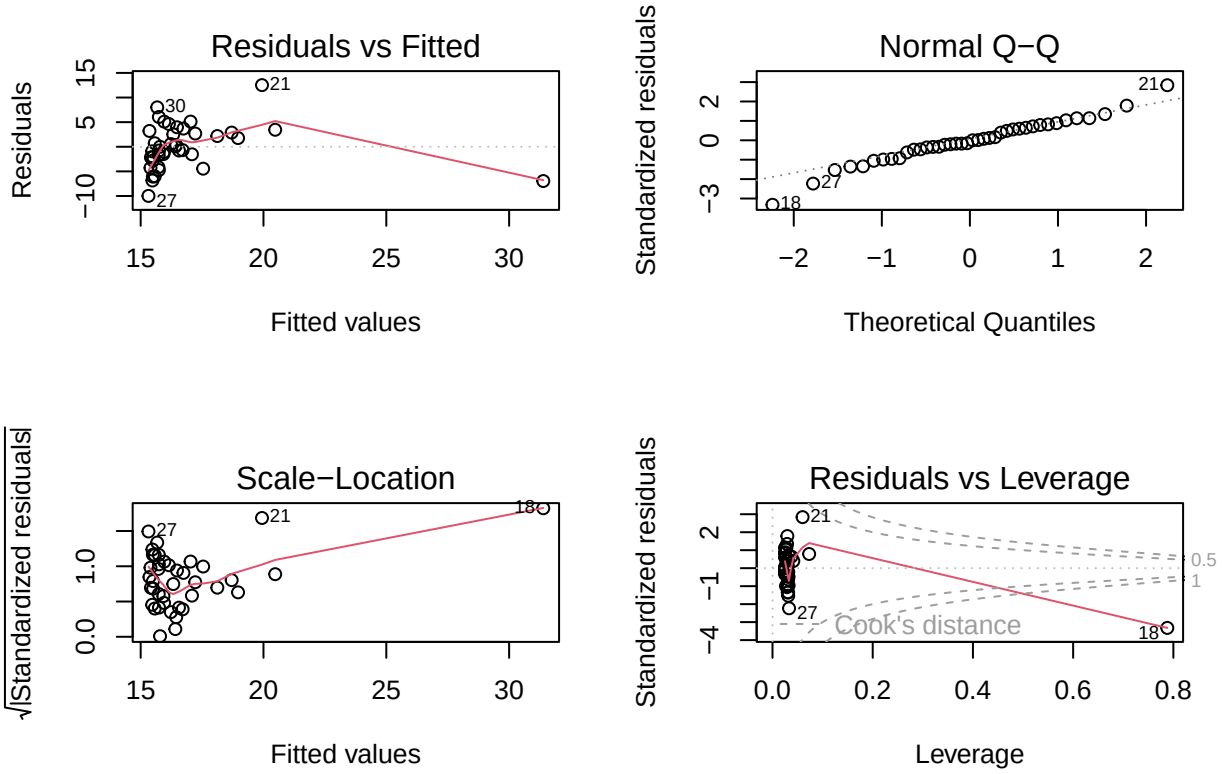
```
##
## Call:
## lm(formula = productivity ~ electricity_consumption, data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9.9816 -2.3189 -0.3405  2.9626 12.5203
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)               13.7173     1.1097  12.362 6.91e-15 ***
## electricity_consumption    0.3975     0.1084   3.667 0.000747 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.55 on 38 degrees of freedom
## Multiple R-squared:  0.2614, Adjusted R-squared:  0.242
```

```
## F-statistic: 13.45 on 1 and 38 DF,  p-value: 0.0007472
```

This method is based on **4 assumptions** :
- **Linearity**: the relationship between the variables X and Y is linear.
- **Independence**: each observation is independent.
- **Homoscedasticity**: the residuals have constant variance at every level.
- **Normality**: the residuals are normaly distributed.

Let's check that out.

```
par(mfrow = c(2,2))
plot(simple_lm)
```



1.1. Linearity

We can plot a *scatter plot* of the **residuals** again the **fitted values**.
If there is a linear relationship, the points should be randomly distributed around the horizontal line $y = 0$.

As we can see in the **residual vs fitted** plot, the mean of the residuals is not O.

1.2. Independence

We want to make sure the is no dependance between some variables (i.e. each observation is collected independently). We can check the *correlation coefficient* of 2 variables in r.

1.3. Homoescedasticity

We want to make sure the variance of the error term ϵ of the model is constant. The **Null Hypothesis** H_0 is that our model is _homoscedastic. We don't want to reject that. We want to check that the **p-value** of the **Breush-Pagan test** is above the 5% level otherwise we should add more independant variables to our model that could explain why the error is larger in some cases and not others.

1.4. Normality

We want to make sure the residuals are normaly distributed so that our model is not consistently under- or over-predicting the values.
In order to visualize that in r, we can plot an *histogram* of the residuals or a *qqplot*

We can see in the **qqplot** that the high and low values are pulling away from the dashed line.

Model evaluation

We want to check if our model is reliable as a tool for understanding the phenomema we're studying.

R-Squared

The **R-Squared** is a number ranging from 0 to 1. It is commonly interpreted as indicating the **percentage of share of the variance in the dependent variable**.

Higher value of the **R-Squared** means that we explained more of the variation in our dependent variable. It should be noted that **R-Squared** can only increase as we add more variables but adding more variables does not automatically improve our model. Therefore, **Adjusted R-Squared** is a better metric in case of multiple linear regression as it decreases if the added variables do not improve the model.

```
summary(simple_lm)
```

```
##
## Call:
## lm(formula = productivity ~ electricity_consumption, data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9.9816 -2.3189 -0.3405  2.9626 12.5203
##
## Coefficients:
##                         Estimate Std. Error t value Pr(>|t|)
## (Intercept)              13.7173     1.1097  12.362 6.91e-15 ***
## electricity_consumption   0.3975     0.1084   3.667 0.000747 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.55 on 38 degrees of freedom
## Multiple R-squared:  0.2614, Adjusted R-squared:  0.242
## F-statistic: 13.45 on 1 and 38 DF,  p-value: 0.0007472
```

The **R-Squared** is 0.2614 so 26 of the variation of the **productivity** is explained by the **electricity consumption**.

2. Transforming the variables

2.1. Square transformation

We first try improving our model by adding a square term.

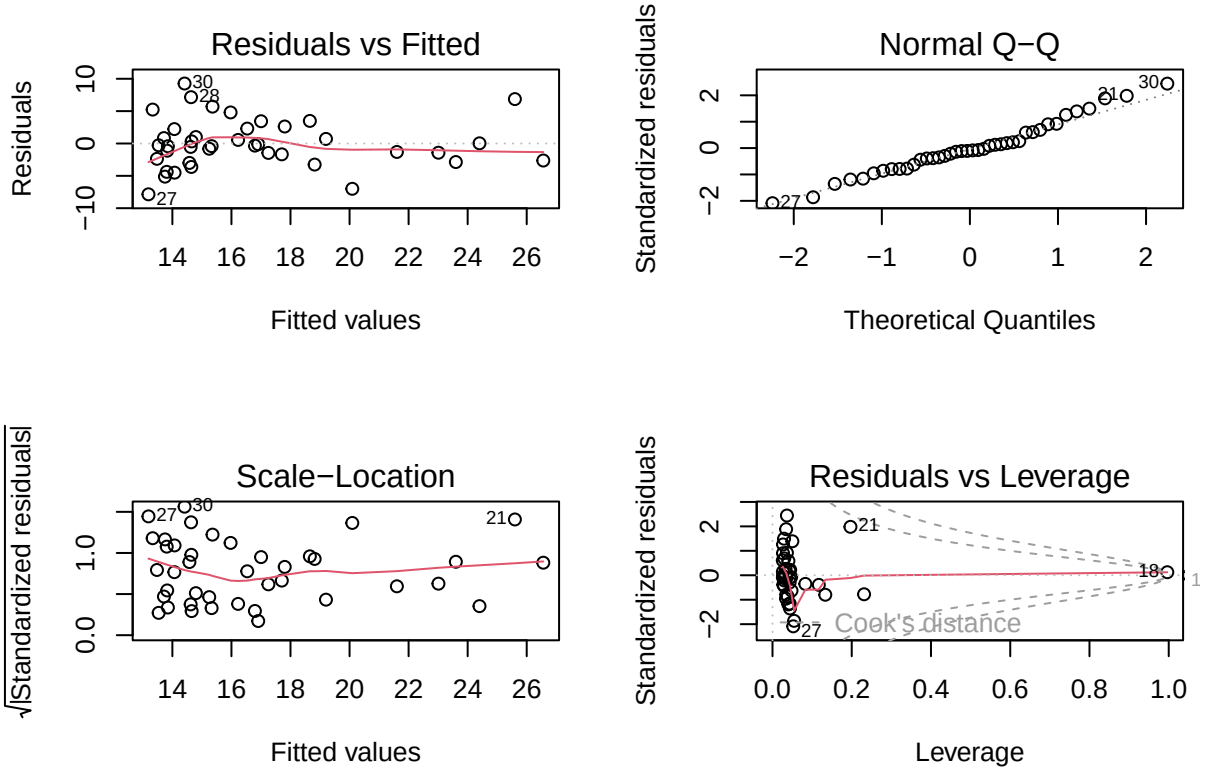
```
square_lm <- lm(productivity ~ electricity_consumption + I(electricity_consumption^2), data = df)
summary(square_lm)
```

```
##
## Call:
## lm(formula = productivity ~ electricity_consumption + I(electricity_consumption^2),
```

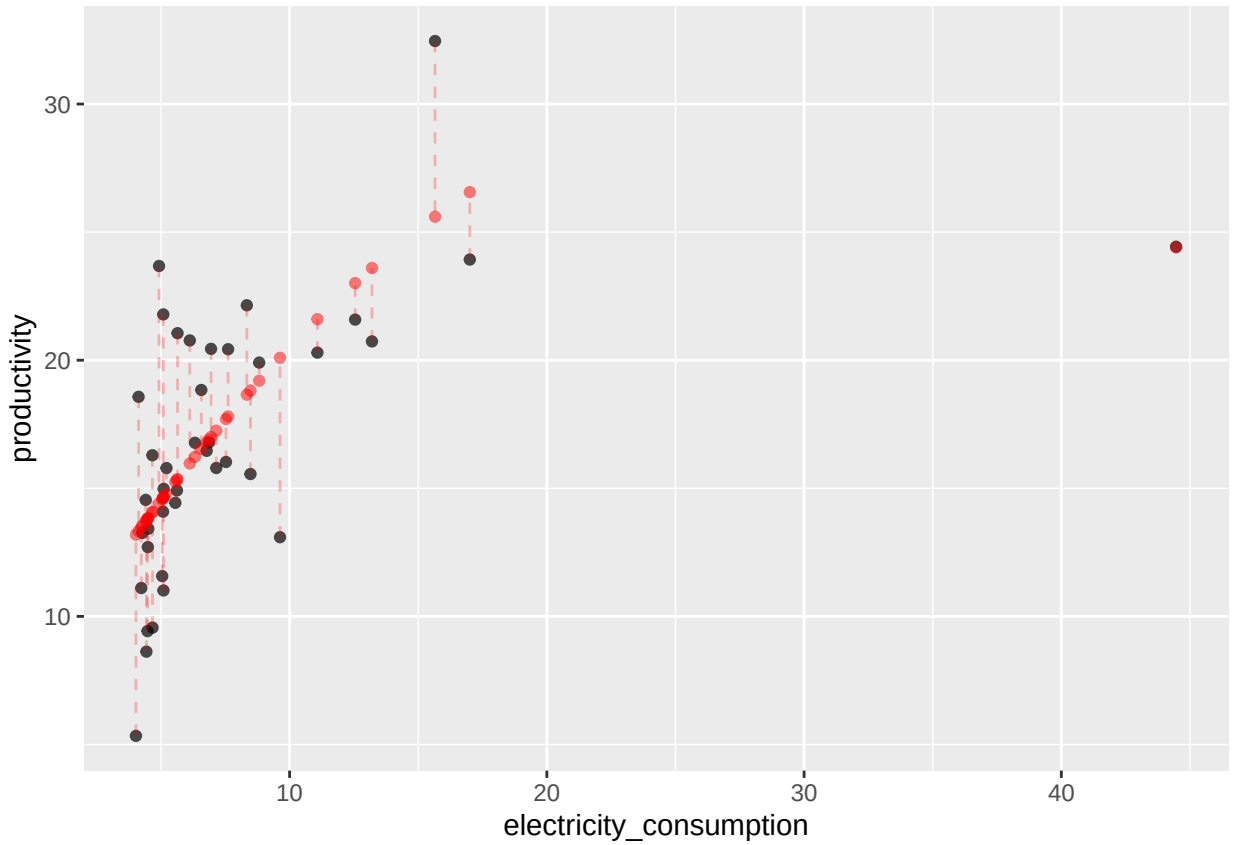
```
##     data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.8536 -2.4357 -0.3714  2.2440  9.2655
##
## Coefficients:
##                                Estimate Std. Error t value Pr(>|t|)
## (Intercept)                    7.162903   1.903969   3.762 0.000583 ***
## electricity_consumption        1.607098   0.318895   5.040 1.25e-05 ***
## I(electricity_consumption^2) -0.027426   0.006923  -3.962 0.000326 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.864 on 37 degrees of freedom
## Multiple R-squared:  0.4814, Adjusted R-squared:  0.4534
## F-statistic: 17.17 on 2 and 37 DF,  p-value: 5.301e-06
```

```
par(mfrow = c(2,2))
plot(square_lm)
```

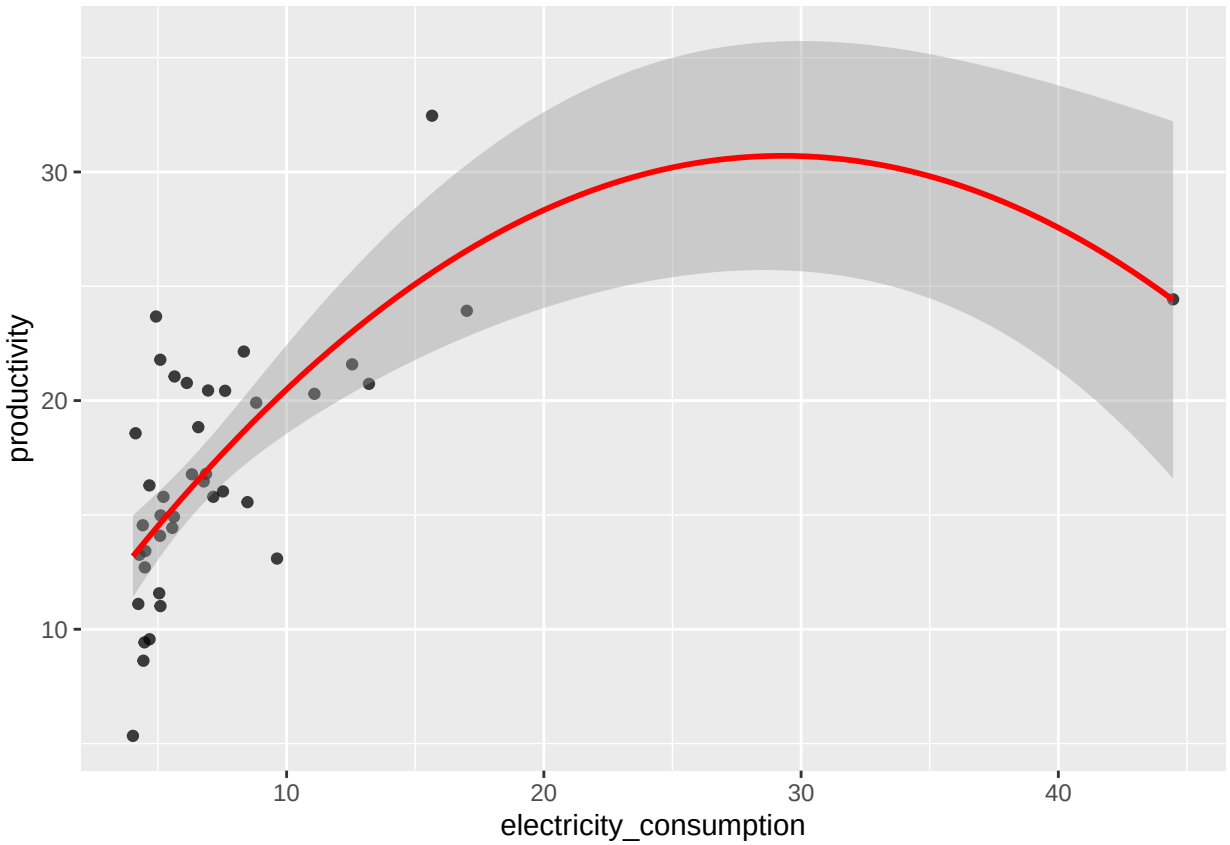
```
## Warning in sqrt(crit * p * (1 - hh)/hh): Production de NaN

## Warning in sqrt(crit * p * (1 - hh)/hh): Production de NaN
```




```
ggplot(df, aes(x = electricity_consumption, y = productivity)) +
  geom_point(alpha = 0.7) +
  geom_point(aes(x = electricity_consumption, y = square_lm$fitted.values), color = "red", alpha = 0.5) +
  geom_segment(aes(xend = electricity_consumption, yend = square_lm$fitted.values), color = "red", alpha
```




```
ggplot(df, aes(x = electricity_consumption, y = productivity)) +
  geom_point(alpha = 0.75) +
  geom_smooth(method = "lm", formula = y ~ x + I(x^2), color = "red")
```

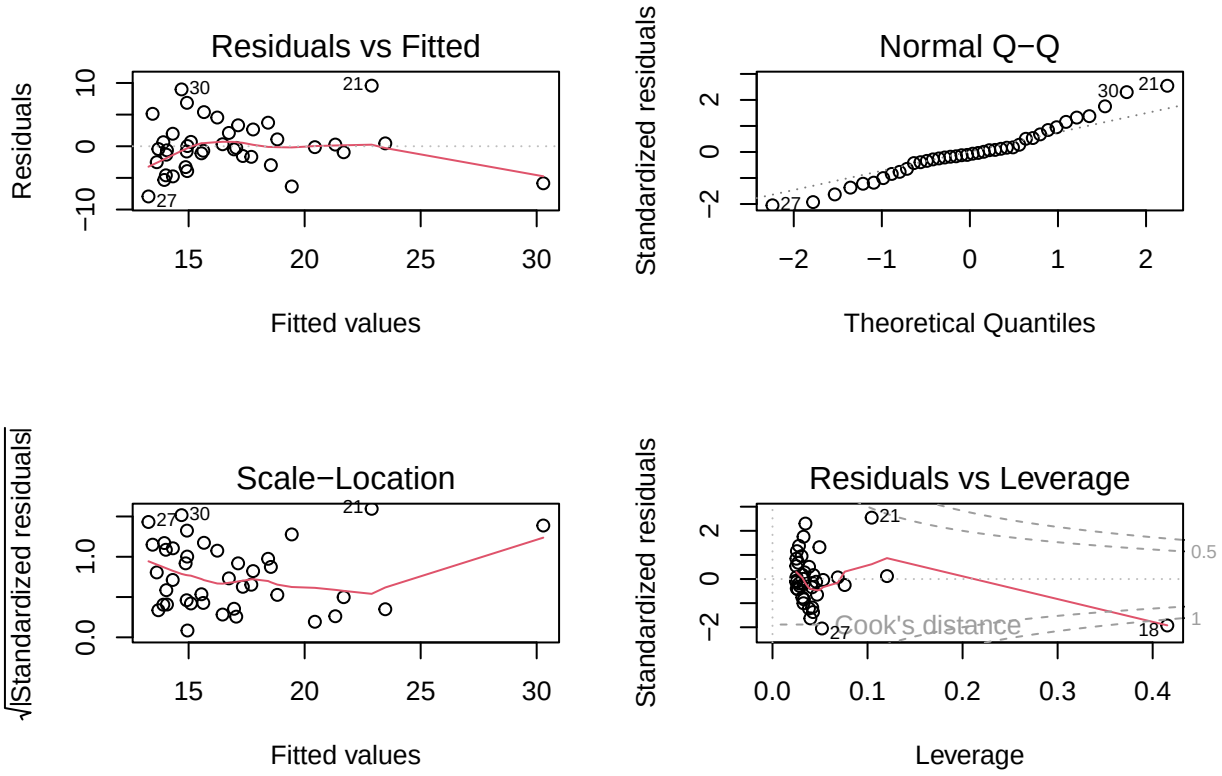


2.2. Log transformation

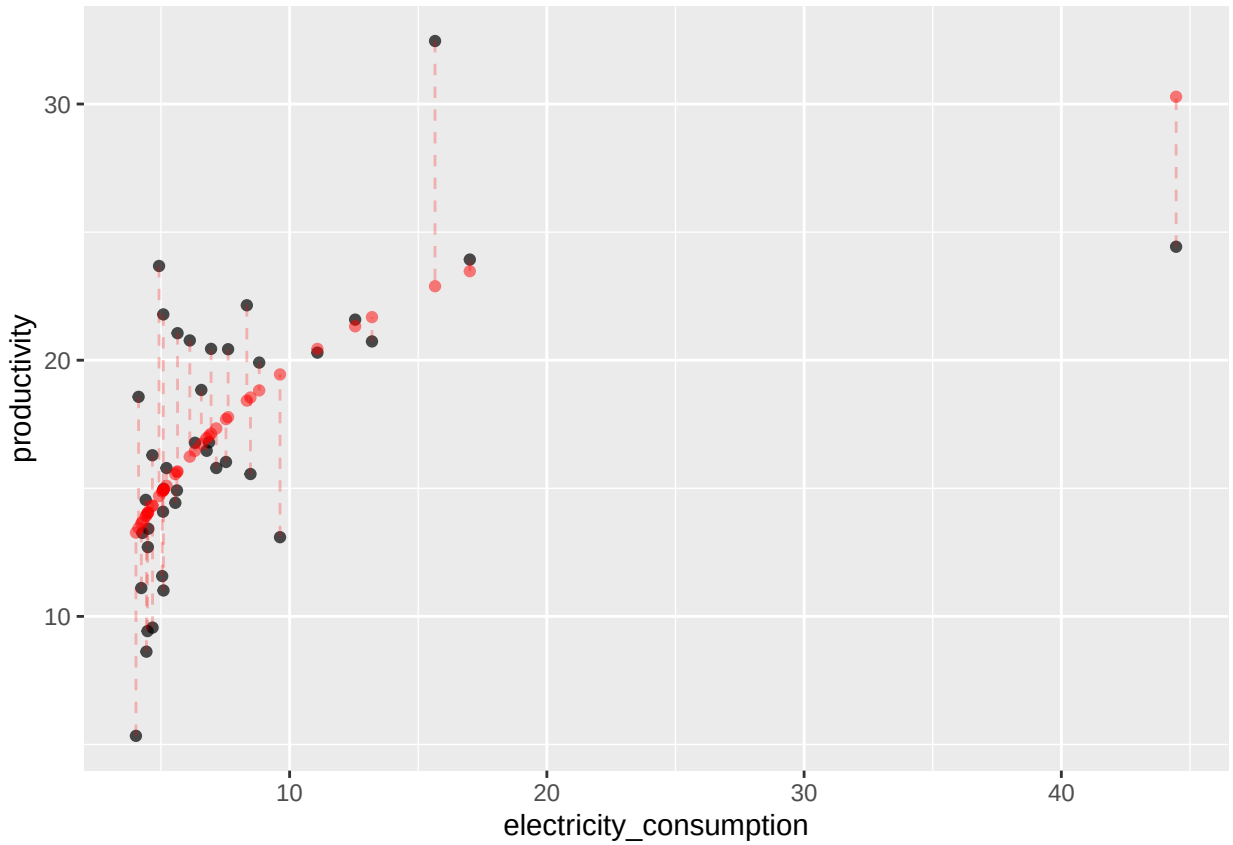
```
log_lm <- lm(productivity ~  log(electricity_consumption), data = df)
summary(log_lm)
```

```
##
## Call:
## lm(formula = productivity ~ log(electricity_consumption), data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.9331 -1.8846 -0.3487  2.0075  9.5712
##
## Coefficients:
##                              Estimate Std. Error t value Pr(>|t|)
## (Intercept)                     3.400      2.552   1.332    0.191
## log(electricity_consumption)    7.085      1.306   5.423 3.52e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.975 on 38 degrees of freedom
## Multiple R-squared:  0.4363, Adjusted R-squared:  0.4215
## F-statistic: 29.41 on 1 and 38 DF,  p-value: 3.518e-06
```

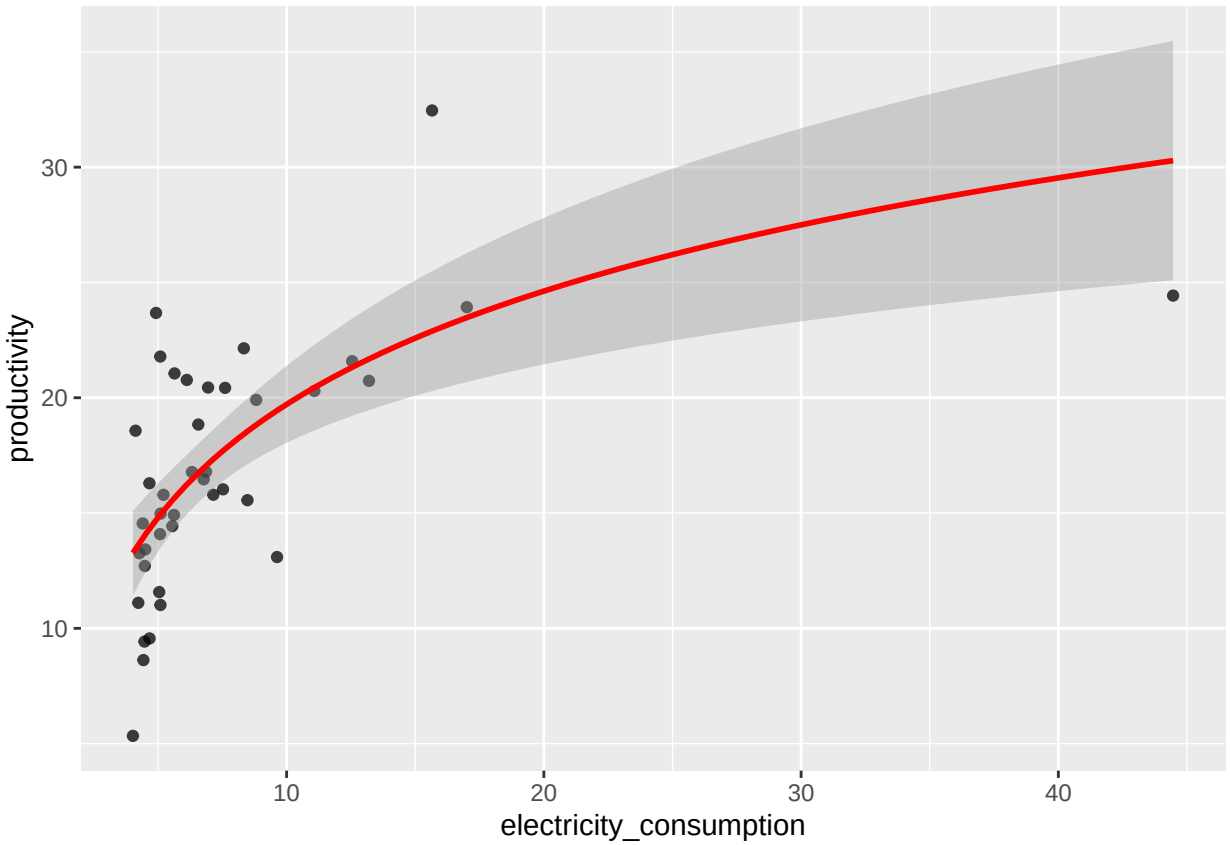
```
par(mfrow = c(2,2))
plot(log_lm)
```




```
ggplot(df, aes(x = electricity_consumption, y = productivity)) +
  geom_point(alpha = 0.7) +
  geom_point(aes(x = electricity_consumption, y = log_lm$fitted.values), color = "red", alpha = 0.5) +
  geom_segment(aes(xend = electricity_consumption, yend = log_lm$fitted.values), color = "red", alpha =
```




```
ggplot(df, aes(x = electricity_consumption, y = productivity)) +
  geom_point(alpha = 0.75) +
  geom_smooth(method = "lm", formula = y ~ log(x), color = "red")
```

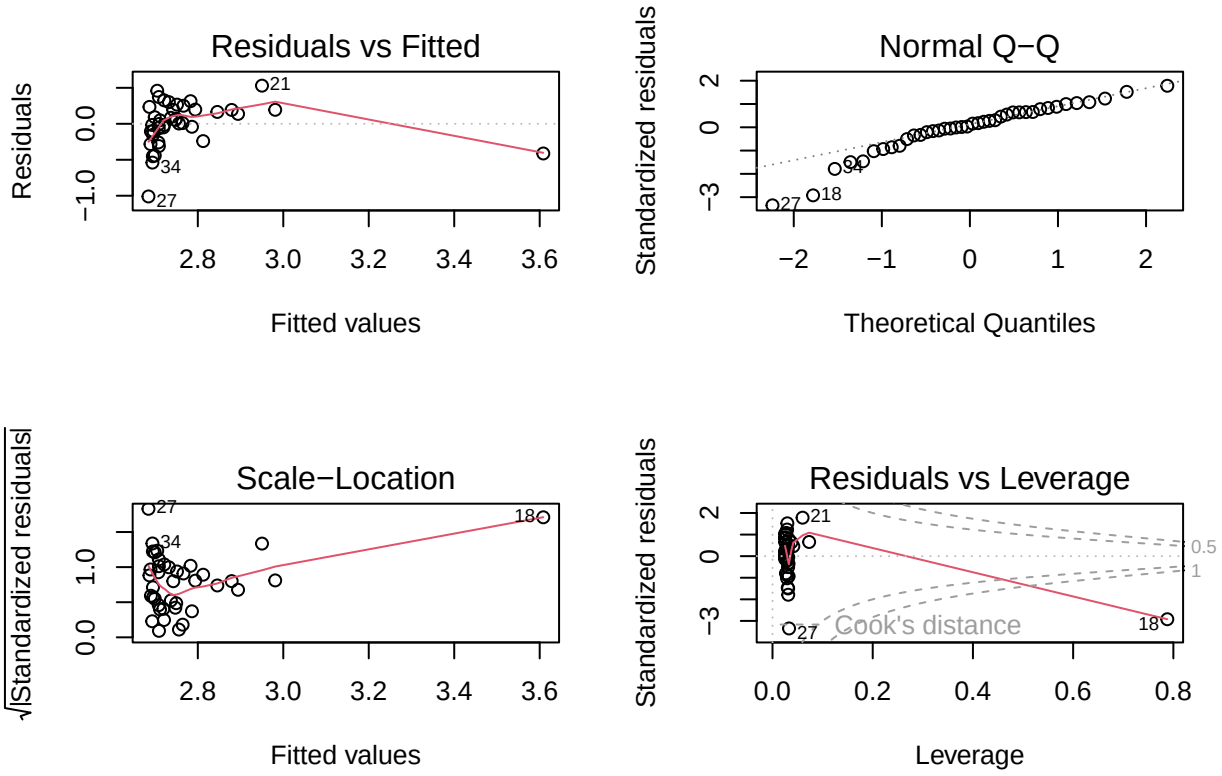


2.2. Other Log transformation

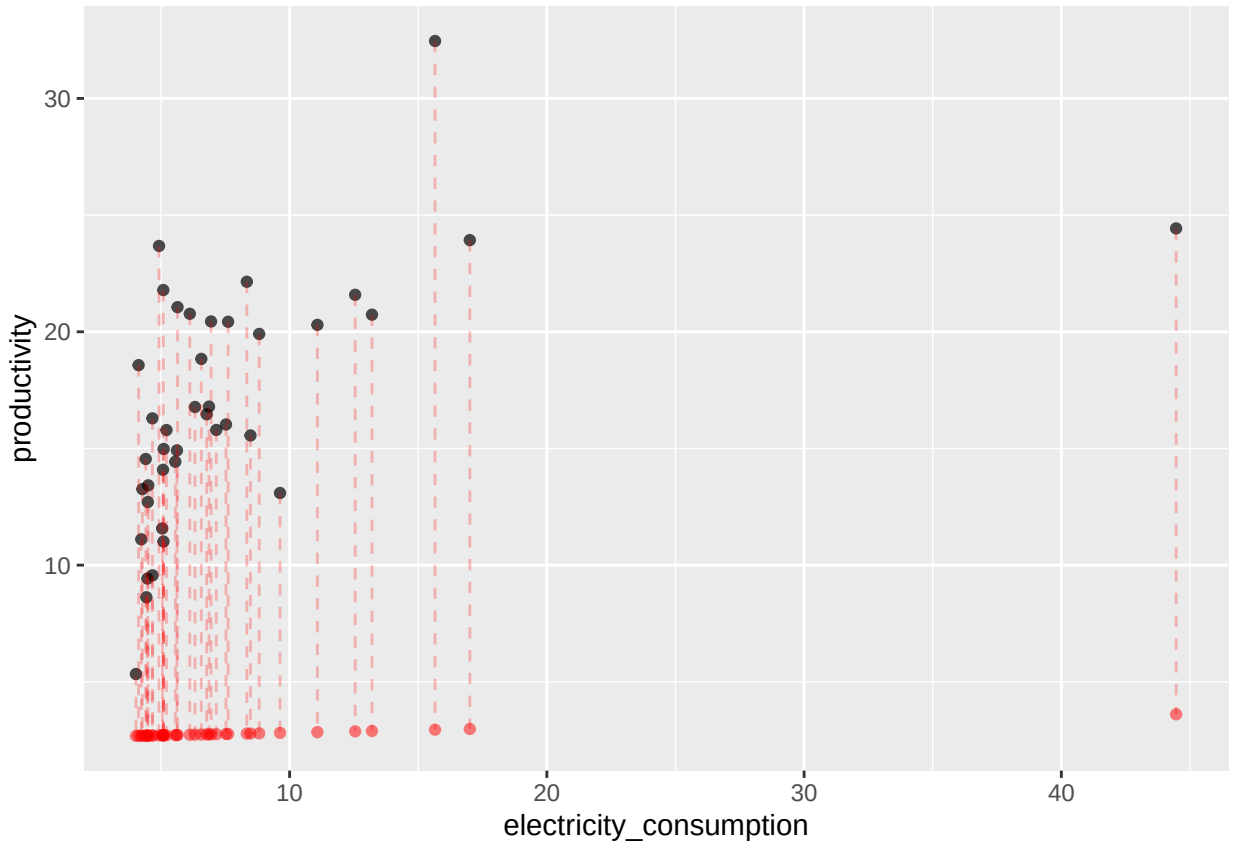
```
other_log_lm <- lm(log(productivity) ~ electricity_consumption, data = df)
summary(log_lm)
```

```
##
## Call:
## lm(formula = productivity ~ log(electricity_consumption), data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.9331 -1.8846 -0.3487  2.0075  9.5712
##
## Coefficients:
##                              Estimate Std. Error t value Pr(>|t|)
## (Intercept)                     3.400      2.552   1.332    0.191
## log(electricity_consumption)    7.085      1.306   5.423 3.52e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.975 on 38 degrees of freedom
## Multiple R-squared:  0.4363, Adjusted R-squared:  0.4215
## F-statistic: 29.41 on 1 and 38 DF,  p-value: 3.518e-06
```

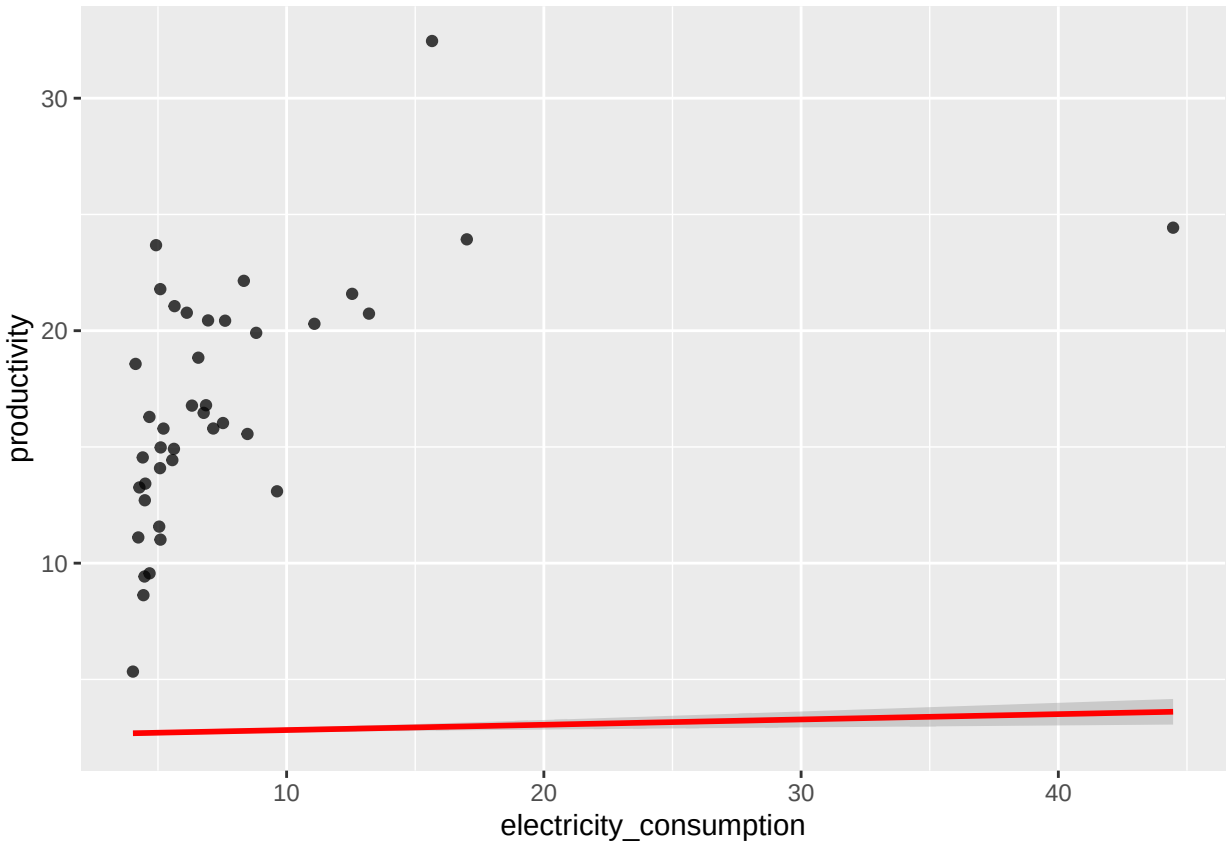
```
par(mfrow = c(2,2))
plot(other_log_lm)
```




```
ggplot(df, aes(x = electricity_consumption, y = productivity)) +
  geom_point(alpha = 0.7) +
  geom_point(aes(x = electricity_consumption, y = other_log_lm$fitted.values), color = "red", alpha = 0
  geom_segment(aes(xend = electricity_consumption, yend = other_log_lm$fitted.values), color = "red", al
```




```
ggplot(df, aes(x = electricity_consumption, y = productivity)) +
  geom_point(alpha = 0.75) +
  geom_smooth(method = "lm", formula = log(y) ~ x, color = "red")
```


30
20
10
productivity
10
20
30
40
electricity_consumption