The Munich Procedure

Calculating and applying coefficient corrections following the Munich Procedure

Michaela Schauer

2024-04-02

Contents

1 Introduction 4

2 About this code 4

3 Packages 5

4 Set working directory 5

5 Preparing the spreadsheet 6

5.1 Coefcors for p-XRF and laboratory methods 6
5.2 Device-internal coefcors 7

6 Example of developing a coefcor: Aluminium in coefcor I 8

6.1 First iteration 8
6.1.1 First impressions 8
6.1.1.1 *Scatter plot* 8
6.1.1.2 *Correlation* 9
6.1.1.3 *More key values* 9
6.1.2 Ordinary Linear Regression (OLR) 10
6.1.2.1 *Calculating the OLR* 10
6.1.2.2 *Calculating the Root Mean Squared error (RMS)* 11
6.1.2.3 *Calculating the relative SEE (rSEE)* 11
6.1.2.4 *Checking residuals* 12
6.1.2.5 *Checking for significant outliers* 12
6.1.2.6 *Testing for robustness - confidence intervals* 13
6.1.3 Regression Trough Origin (RTO) 14
6.1.3.1 *Calculating the RTO* 14
6.1.3.2 *Calculating the Root Mean Squared error (RMS)* 15
6.1.3.3 *Calculating the relative SEE (rSEE)* 15
6.1.3.4 *Calculating the coefficient of determination (r^2)* 16
6.1.3.5 *Checking residuals* 16
6.1.3.6 *Checking for significant outliers* 17
6.1.3.7 *Testing for robustness - confidence intervals* 18
6.1.4 Result 19

6.2 Second iteration 19
6.2.1 First impressions 20
6.2.2 Ordinary Linear Regression (OLR) 20
6.2.3 Regression Trough Origin (RTO) 23
6.2.4 Result 25
6.3 Third iteration 26
6.3.1 First impressions 26
6.3.2 Ordinary Linear Regression (OLR) 27
6.3.3 Regression Trough Origin (RTO) 29
6.3.4 Result 31
6.4 Fourth iteration 32
6.4.1 First impressions 32
6.4.2 Ordinary Linear Regression (OLR) 33
6.4.3 Regression Trough Origin (RTO) 35
6.4.4 Result 37
6.5 Fifth iteration 38
6.5.1 First impressions 38
6.5.2 Ordinary Linear Regression (OLR) 39
6.5.3 Regression Trough Origin (RTO) 41
6.5.4 Result 43
6.6 Sixth iteration 44
6.6.1 First impressions 44
6.6.2 Ordinary Linear Regression (OLR) 45
6.6.3 Regression Trough Origin (RTO) 47
6.6.4 Result 49
6.7 Seventh iteration 50
6.7.1 First impressions 50
6.7.2 Ordinary Linear Regression (OLR) 51
6.7.3 Regression Trough Origin (RTO) 54
6.7.4 Result 56
6.8 Eighth iteration 57
6.8.1 First impressions 57
6.8.2 Ordinary Linear Regression (OLR) 58
6.8.3 Regression Trough Origin (RTO) 60
6.8.4 Result 62
6.9 Ninth iteration 63
6.9.1 First impressions 63
6.9.2 Ordinary Linear Regression (OLR) 64
6.9.3 Regression Trough Origin (RTO) 66
6.9.4 Result 69
6.10 Tenth iteration 69
6.10.1 First impressions 69
6.10.2 Ordinary Linear Regression (OLR) 70
6.10.3 Regression Trough Origin (RTO) 73
6.10.4 Result 75
6.11 Eleventh iteration 76
6.11.1 First impressions 76
6.11.2 Ordinary Linear Regression (OLR) 77
6.11.3 Regression Trough Origin (RTO) 79
6.11.4 Result - RTO 82

7 Example of extracting criteria and factors 82
7.1 Extracting from OLR . . . 83
7.2 Extracting from RTO . . . 84
7.3 Exporting coefcor criteria (_criteria.csv) . . . 86
7.4 Exporting coefcor factors - slope and intercept (_factors.csv) . . . 86

8 Example of graphics export 87
8.1 Exporting all relevant descriptive graphics for OLR and RTO . . . 87
8.2 Exporting the traditionally presented scatter plots . . . 92

9 Example of applying coefcors obtained from p-XRF and laboratory values to a dataset 94
9.1 Loading and formating the data . . . 94
9.2 Performing the calculations . . . 94
9.3 Compiling and exporting the processed data (_cor_data.csv) . . . 97

10 Example of applying device-internal coefcors to a dataset 98
10.1 Loading and formating the data . . . 98
10.2 Performing the calculations . . . 99
10.3 Compiling and exporting the processed data (_cor_data.csv) . . . 101

1 Introduction

This script **presents the Munich Procedure** which is **based on the Frankfurt Procedure** developed by Dr Markus Helfert and **takes this approach a step further** (Schauer 2023 & Schauer 2024): The Frankfurt Procedure compares values measured by p-XRF (x-values or independent variable) with values of the same set of samples obtained by a laboratory method - in this case WD-RFA (y-values or dependent variable). A coefficient correction (coefcor) is applied by using the calculated slope and, if appropriate, intercept to correct the p-XRF values to match those of the laboratory method. In addition, the same approach can be used to compare data obtained from the same instrument before and after a change of set-up by the manufacturer. The Munich Procedure additionally provides the calculations as R-scripts and adds quality criteria such as rSEE, RMS and CI, as well as the associated graphical output.

This code example shows the development of a coefcor for the element **aluminium** by **comparing p-XRF and WD-RFA values** from measurements of the **Frankfurt pottery sample set**. The p-XRF data used are from the first data set produced for the Niton XL3t No 97390 - namely **coefcor I**.

As the following code is an example for the application on aluminium - for all other elements the term 'Al' can simply be replaced by the desired element (Ti, Fe etc.).

2 About this code

The code was written using RGUI version 4.3.1, RCMDR version 2.8-0 and RStudio version 2023.06.1 with Quarto version 1.3.433. It is executable, i.e. it is fully functional when handled according to the instructions:

- In order for the code to run without errors, the 'Chunk Output line' must be checked under 'Edit the R-Markdown format options for the current file' (small cogwheel to the right of the 'Render' button).
- To set the working directory, the checkbox under 'Workspace Panes' (the four-square icon above the Render button) must be set to 'Show all panes'. Then there is a panel at the bottom right of the screen, whose display in this panel's toolbar must be set to 'Files'. There you will find a blue cogwheel (More File Commands), select this and a drop down menu will appear. Select 'set as working directory' - this will set the working directory to the folder where the project is located. This only works if the entire project folder has been saved unchanged!

The sources used to create the code for each chapter can be found under 'source' and are only presented when the code is first introduced.

Schauer, Michaela 2024: Coefficient corrections for portable X-ray fluorescence data of the Niton XL3t No. 97390 (coefcor I-IV) developed according to the Munich procedure. Data in Brief 53. 109914. https://doi.org/10.1016/j.dib.2023.109914

Schauer, Michaela 2023: R-scripts and data of coefficient corrections developed since 2017 for the Niton XL3t No. 97390 following the Munich Procedure. 12. September 2023. Open Data LMU. https://doi.org/10.5282/ubm/data.405

3 Packages

source

Instructions following https://www.projectpro.io/recipes/load-package-r and https://www.dataquest.io/blog/install-package-r/

install packages

To use this script you have to install the necessary packages by using this code

```
install.packages(c('car','cowplot','dplyr','ggpmisc','ggplot2','ggpubr','grid',
                   'tibble','Rcmdr','RcmdrMisc'))
```

The below named packages must be loaded actively at every restart (regardless of whether R-Gui or R-Studio is used to run the code).

```
library(car)
library(cowplot)
library(dplyr)
library(ggpmisc)
library(ggplot2)
library(ggpubr)
library(grid)
library(tibble)
library(Rcmdr)
library(RcmdrMisc)
```

4 Set working directory

source

Instructions following https://martinctc.github.io/blog/rstudio-projects-and-working-directories-a-beginner%27s-guide/, https://www.grainge.org/pages/authoring/relative_paths/relative_paths.htm and https://r4ds.had.co.nz/workflow-projects.html

Next, the file path - i.e. the location where the file that R is supposed to work with can be found - must be defined. In R-terminology this is call working directory. We are working with a relative path which is defined in and by the location of the R-project-file (**MunichProcedure.Rproj**).

To adjust the working directory to the position of the files on your computer, click on 'Files' in the task bar of the lower right pane and select 'Munich Procedure' from the terms in the header written in blue. You should see the folder structure of this project in the field below. Now under 'More' (blue cogwheel) select 'Set As Working Directory'.

```
knitr::opts_knit$set(root.dir = "./")
```

5 Preparing the spreadsheet

5.1 Coefcors for p-XRF and laboratory methods

source

Instructions following https://www.digitalocean.com/community/tutorials/r-read-csv-file-into-data-frame, https://sparkbyexamples.com/r-programming/r-select-function-from-dplyr/, https://dplyr.tidyverse.org/reference/mutate.html, https://www.digitalocean.com/community/tutorials/replace-in-r, https://search.r-project.org/R/refmans/base/html/Round.html, https://www.statology.org/transpose-data-frame-in-r/, https://stackoverflow.com/questions/40947288/how-to-calculate-mean-of-all-columns-by-group, https://www.rdocumentation.org/packages/base/versions/3.6.2/topics/merge and https://datatofish.com/export-dataframe-to-csv-in-r/

The source file must be prepared in the same way as the **_data.csv** files(see e.g. **coefcorI_example_data.csv**). The most important requirements are the headers of the columns - p-XRF data have headers that only give the elements (Si, Ti etc.), while the header for laboratory analyses also contains information about the analysis type - for example WD-RFA (Si.WDXRF, Ti.WDRFA etc.).

The file format is created from the analytical data in files named **_data.csv** (in this example **coefcorI_adata.csv**). Loading is only possible if the required file is stored in a folder that can be found in the defined working directory.

```
dataset<- read.csv("../data_analytical//coefcorI_adata.csv")
```

Next we filter for the information we need.

```
dataset_pXRF<-select(dataset,"Sample","Si","Ti","Al","Fe","Mn","Mg","Ca","K","P",
"S","Cl","Sc","V","Cr","Co","Ni","Cu","Zn","As","Se","Rb","Sr","Y","Zr","Nb","Mo",
"Pd","Ag","Cd","Sn","Sb","Te","Cs","Ba","La","Ce","Hf","Ta","W","Re","Au","Hg",
"Pb","Bi","Th","U")
```

Then we replace the information that a certain value was below the limit of detection (<LOD) with 0 and define all measurement values as numeric characters.

```
dataset_pXRF <- mutate_all(dataset_pXRF, ~gsub("<LOD", "0", .))

dataset_pXRF[, 2:47] <- lapply(dataset_pXRF[, 2:47], as.numeric)
```

Following that, we calculate the mean for each sample and element, then we round those values to whole numbers.

```
dataset_pXRF_mean<-summarise(dataset_pXRF, across(everything(), mean), .by = c(Sample))

dataset_pXRF_mean[, 2:47] <- round(dataset_pXRF_mean[, 2:47])
```

Next, we load the file containing the WDXRF measurements of Dr. Markus Helfert (**MHelfert_FrankfurtProcedur**

```r
dataset_WDXRF<- read.csv("../data_analytical//MHelfert_FrankfurtProcedure_WDXRFdata.csv")
```

And combine both files by sample number.

```r
data<- merge(dataset_pXRF_mean, dataset_WDXRF, by = "Sample", all = FALSE)
```

Then, we export the file as **_data.csv** (in our case **coefcorI_data.csv**)

```r
write.csv(data,"../data_analytical/coefcorI_data.csv",row.names=TRUE)
```

Now we are ready to go and can start to develop coefcor I.

5.2 Device-internal coefcors

> **source**
>
> Instructions following https://www.digitalocean.com/community/tutorials/r-read-csv-file-into-data-frame, https://sparkbyexamples.com/r-programming/r-select-function-from-dplyr/, https://www.statology.org/r-add-suffix-to-column-names/ and https://datatofish.com/export-dataframe-to-csv-in-r/

If the _data.csv file is to be prepared for internal coefcors such as for example coefcor ItoII of the Niton XL3t No 97390 (Schauer 2023 & Schauer 2024), the files prepared in accordance with section 5.1 form the basis.

The first step is to load the file whose data are to be fitted - i.e. they correspond to the independent variable (x-axis), ergo the p-XRF data from before. Then we select the required variables. In our example this dataset corresponds to coefcorI_data.csv.

```r
dataset_pXRF_I<- read.csv("../data_analytical//coefcorI_data.csv")

dataset_pXRF_I<-select(dataset_pXRF_I,"Sample","Si","Ti","Al","Fe","Mn","Mg","Ca","K","P",
"S","Cl","Sc","V","Cr","Co","Ni","Cu","Zn","As","Se","Rb","Sr","Y","Zr","Nb","Mo",
"Pd","Ag","Cd","Sn","Sb","Te","Cs","Ba","La","Ce","Hf","Ta","W","Re","Au","Hg","Pb",
"Bi","Th","U")
```

In the second step, the file containing the values to be fitted is loaded - i.e. they correspond to the dependent variable (y-axis), ergo the WDXRF values from before. Again, the desired variables are selected. In our example this dataset corresponds to coefcorII_data.csv.

```r
dataset_pXRF_II<- read.csv("../data_analytical//coefcorII_data.csv")

dataset_pXRF_II<-select(dataset_pXRF_II,"Sample","Si","Ti","Al","Fe","Mn","Mg","Ca","K","P",
"S","Cl","Sc","V","Cr","Co","Ni","Cu","Zn","As","Se","Rb","Sr","Y","Zr","Nb","Mo",
"Pd","Ag","Cd","Sn","Sb","Te","Cs","Ba","La","Ce","Hf","Ta","W","Re","Au","Hg","Pb",
"Bi","Th","U")
```

As before, an abbreviation is added to the column headings of the independent variables, namely the Roman numeral of the coefcor - in our example '.II'.

```
colnames(dataset_pXRF_II)[-1] <- paste(colnames(dataset_pXRF_II)[-1], ".II", sep = "")
```

And combine both files by sample number.

```
data<- merge(dataset_pXRF_II, dataset_pXRF_II, by = "Sample", all = FALSE)
```

Then, we export the file as **_data.csv** (in our case **coefcorItoII_data.csv**)

```
write.csv(data,"../data_analytical//coefcorItoII_data.csv",row.names=TRUE)
```

Now we are ready to go and could start to develop coefcor ItoII - yet the following example uses the data for coefcor I.

6 Example of developing a coefcor: Aluminium in coefcor I

The following code is used to compute coefcors from a **_data.csv**-file for the element aluminium based on the yet to load data set (see section 6.1) of coefcor I (**coefcorI_data.csv**).

> **Device-interal coefcors**
>
> If an device-internal coefcor is to be developed, only '.WDXRF' needs to be replaced by the respective Roman number of the independent variables (i.e. their suffix). Thus, in the case of coefcor ItoII, '.WDXRF' would be replaced by '.II'.

6.1 First iteration

For the first iteration, the entire data set containing only the values of p-XRF and WDXRF-measurements is loaded (**coefcorI_data.csv**).

```
dataset<- read.csv("../data_analytical//coefcorI_data.csv")
```

This data is now used to calculate linear regressions which can give us the coefcors we are looking for.

6.1.1 First impressions

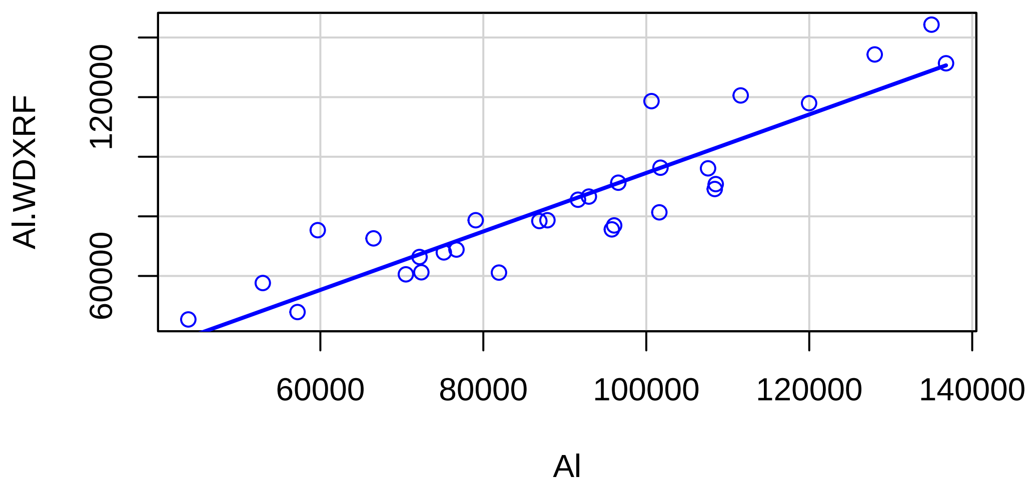
The following information gives a first impression of the data with regard to the already existing agreement between p-XRF values and those of laboratory analysis.

6.1.1.1 *Scatter plot*

> **source**
>
> Instructions following http://www.sthda.com/english/wiki/scatter-plots-r-base-graphs

The scatter plot allows an intuitive interpretation of the agreement of p-XRF- and WD-RFA-values based on the graph.

```
scatterplot(Al.WDXRF~Al, regLine=TRUE, smooth=FALSE, boxplots=FALSE, data=dataset)
```



In this case, the agreement does look quite okay already.

6.1.1.2 *Correlation*

> **source**
>
> Instructions following https://www.statology.org/r-cor-function/

Complementing the graph, the value of the pearson correlation coefficient gives an impression of how good the agreement of the values is. The correlation coefficient can have values between 0 and 1, whereby results close to 1 are desirable.

```
cor(dataset$Al,dataset$Al.WDXRF)
```

```
[1] 0.9105113
```

Also this figure is good - still it is likely that some iterations will be necessary to get a linear regression which complies with all the criteria defined in Schauer 2024 and shortly described below.

6.1.1.3 *More key values*

> **source**
>
> Instructions following https://www.rdocumentation.org/packages/Rcmdr/versions/2.0-4/topics/numSummary

It is also worth taking a look at other key values such as mean value, standard deviation and the values of the quartiles. From this table, lower (0%) and upper limit (100%) are taken after completion of the analysis. They define the range of values for which the selected linear regression and thus the associated coefcors are valid.

```
numSummary(dataset[,c("Al", "Al.WDXRF"), drop=FALSE],
           statistics=c("mean", "sd", "quantiles"),quantiles=c(0,.25,.5,.75,1))
```

```
             mean       sd    0%   25%     50%      75%   100%  n
Al       90520.87 23619.45 43795 73085 92284.5 106119.8 136788 30
Al.WDXRF 85260.67 25454.23 45409 68140 78698.0  94906.0 144324 30
```

So in this case, p-XRF values of the element aluminium which range between 43795 ppm (0% - lower limit) and 136788 ppm (100% - upper limit) could be fitted with the coefcors generated in this iteration.

6.1.2 Ordinary Linear Regression (OLR)

First, an ordinary linear regression - i.e. a calculation with slope and intercept - is performed. The characteristic values generated here are then compared with those of a Regression Trough Origin (RTO) calculated subsequently.

6.1.2.1 *Calculating the OLR*

> **source**
>
> Instructions following https://www.statology.org/OLR-regression-in-r/ and https://search.r-project.org/R/refmans/stats/html/lm.html

These lines of code give the most important information of the linear regression. Of particular interest are the spread of the residuals, the estimated values of intercept and slope (Al), the residual standard error (SEE) and the coefficient of determination (r^2). With regard to the latter, the 'multiple R-squared' value is relevant.

```
OLRAl<-lm(Al.WDXRF~Al, data=dataset)
summary(OLRAl)
```

```
Call:
lm(formula = Al.WDXRF ~ Al, data = dataset)

Residuals:
     Min       1Q   Median       3Q      Max
-15694.1  -5683.2   -948.6   5667.8  23468.0

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept) -3.562e+03  7.870e+03  -0.453    0.654
Al           9.812e-01  8.421e-02  11.652 2.97e-12 ***
```

```
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 10710 on 28 degrees of freedom
Multiple R-squared:  0.829, Adjusted R-squared:  0.8229
F-statistic: 135.8 on 1 and 28 DF,  p-value: 2.972e-12
```

The residuals show a very large scatter, the necessary adjustment of the intercept is very high, but the slope correction is acceptable (the closer to 1 the better). The SEE is high, while r^2 is to low.

6.1.2.2 *Calculating the Root Mean Squared error (RMS)*

> **source**
>
> Instructions following https://statisticsglobe.com/extract-fitted-values-from-regression-model-r and https://sparkbyexamples.com/r-programming/add-column-to-dataframe-in-r/

With the root mean squared error (RMS), another relevant value of the linear regression is calculated. It allows an estimation of the deviation of the p-XRF-values from their expected position (i.e. the value of the WD-XRF measurement). It is not used as a discriminating criterion here but is shown in the factors table (_factors.csv).

To calculate the RMS, first the fitted values are extracted from the linear regression model and added as new columns to the dataset (now datasetb), then the RMS is computed.

```
fitted.OLRAl <- fitted(OLRAl)
datasetb<-cbind(dataset,fitted.OLRAl)
RMS<-sqrt(mean((datasetb$Al-datasetb$fitted.OLRAl)^2)/nrow(datasetb))
RMS
```

```
[1] 963.6646
```

6.1.2.3 *Calculating the relative SEE (rSEE)*

> **source**
>
> Instructions following https://www.rdocumentation.org/packages/stats/versions/3.6.2/topics/sigma and https://www.statology.org/r-mean-of-column/

Of much higher importance to the Munich Procedure is the rSEE. To calculate this factor, first the SEE must be obtained from the previously generated data; the mean of the p-XRF-aluminium-values computed.

```
SEE<-sigma(OLRAl)
SEE
```

```
[1] 10711.22
```

Then the formula to calculate the relative residual standard error is defined and the rSEE for aluminium (Al) calculated.

```
mean<-mean(dataset$Al)
rSEE<-(SEE/mean)*100
rSEE
```

```
[1] 11.83288
```

The value of the rSEE in this case is above 10% and thus higher then the threshold.

6.1.2.4 *Checking residuals*

source

Instructions following https://www.rdocumentation.org/packages/stats/versions/3.6.2/topics/plot.lm and https://stat.ethz.ch/R-manual/R-devel/library/stats/html/plot.lm.html

Residuals provide information on how much the measured values differ from the values estimated by linear regression. In practice, it turns out that the Q-Q plot, in which the residuals should lie on a straight line as much as possible, gives a good first impression of whether significant outliers might be present.

```
oldparOLRAl <- par(oma=c(0,0,3,0), mfrow=c(2,2))
plot(OLRAl)
```

In the Q-Q-plot, the pair of measurements in row 25 of the table (the header counts as row '0') could be an outlier. Therefore, removing this sample might significantly lower the rSEE.

6.1.2.5 *Checking for significant outliers*

source

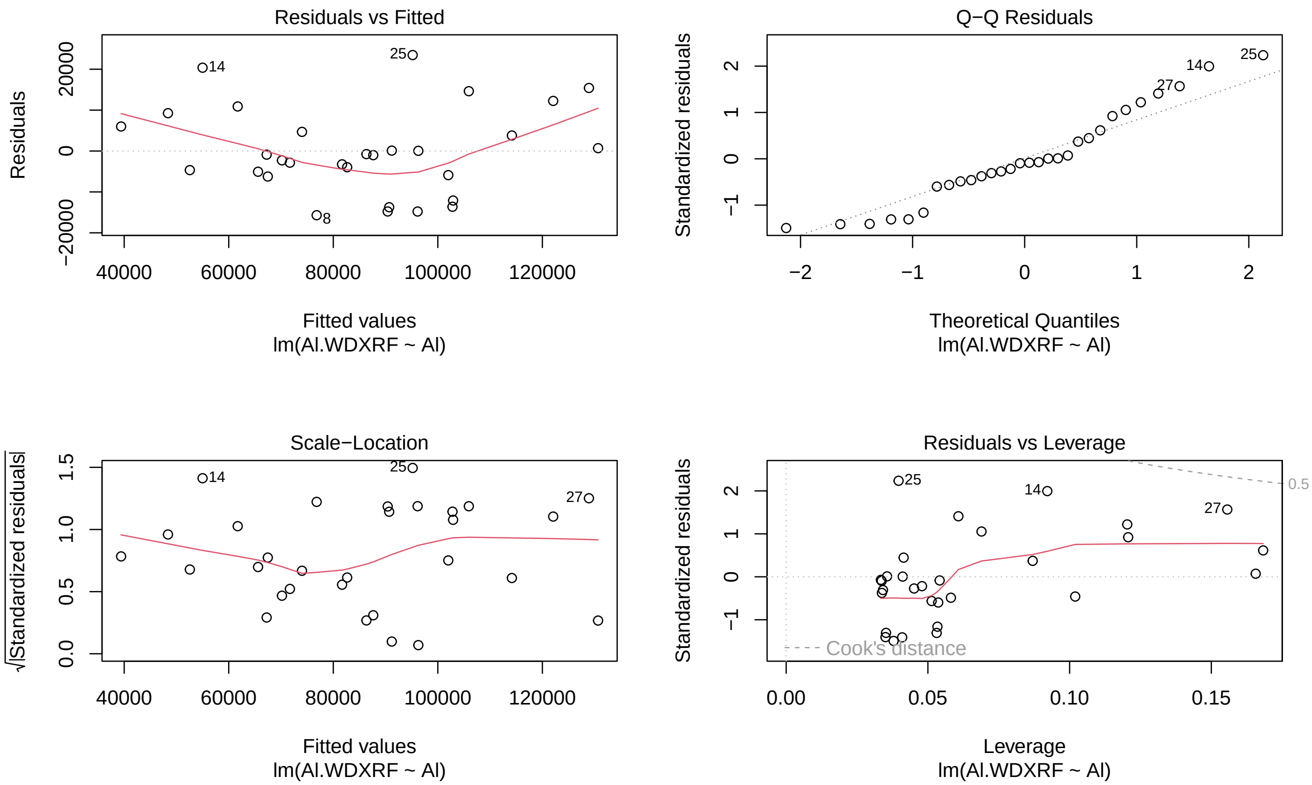
Instructions following https://rdrr.io/cran/car/src/R/outlierTest.R

To check, if significant outliers (Bonferroni p<0.05) are present, a Bonferroni test for outlier (with studentized residuals) is performed.

```
outlierTest(OLRAl)
```

```
No Studentized residuals with Bonferroni p < 0.05
Largest |rstudent|:
   rstudent unadjusted p-value Bonferroni p
25 2.422307           0.022409      0.67227
```

This analysis confirms - as already suspected - that the sample in line 25 is a outlier. But, the p-value of the Bonferroni p shows this outlier is not significant. Due to the criteria defined in Schauer 2024, the sample does not have to be excluded, yet it might become necessary if this sample is also an outlier for the RTO.




```
NULL
```

6.1.2.6 *Testing for robustness - confidence intervals*

source

Instructions following https://www.rdocumentation.org/packages/stats/versions/3.6.2/topics/confint, https://www.rdocumentation.org/packages/boot/versions/1.3-28.1/topics/boot, https://www.rdocumentation.org/packages/boot/versions/1.3-28.1/topics/plot.boot and https://stopsack.github.io/risks/reference/confint.margstd_boot.html

To test the robustness of the linear regression, the values of the 0.5 and 0.95 confidence levels of the slope (Al) estimated from all p-XRF aluminium values are compared with those calculated by a bootstrap algorithm. The results of both calculations should always be as close as possible to each other. First we calculate regular CI-values.

```
Confint(OLRAl, level=0.90)
```

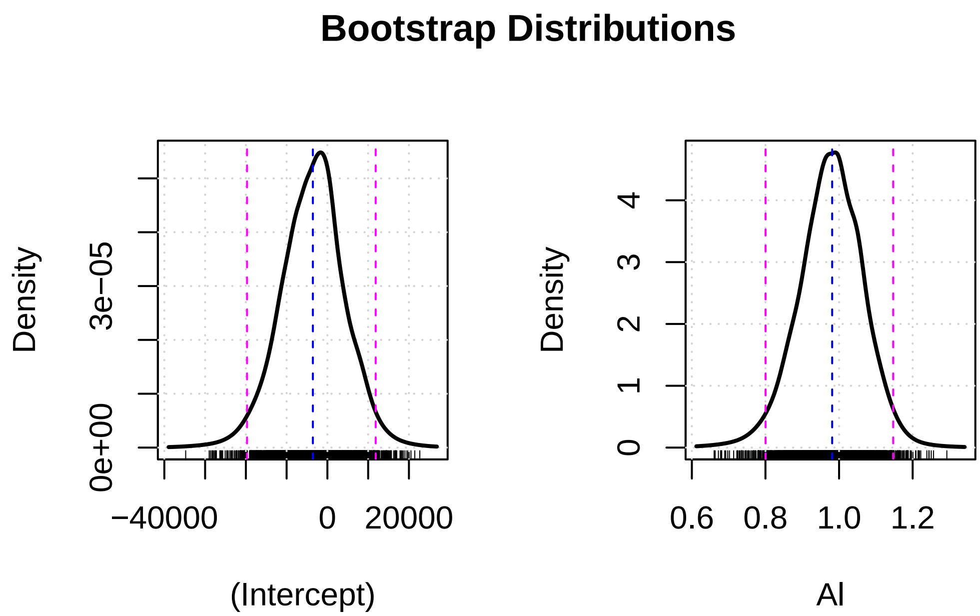
```
                 Estimate          5 %        95 %
(Intercept) -3562.0894024 -1.694952e+04 9825.344971
Al              0.9812407  8.379863e-01    1.124495
```

Then we apply the bootstrap algorithm

```
.bs.samplesOLRAl<- Boot(OLRAl, R=2500, method="case")
confint(.bs.samplesOLRAl, level=0.9, type="bca")
```

```
Bootstrap bca confidence intervals

                         5 %        95 %
(Intercept) -1.709252e+04 9095.108698
Al             8.345235e-01    1.123719
```

```
plotBoot(.bs.samplesOLRAl)
```



According to the criteria defined in Schauer 2024, the difference in the 0.5 confidence level of the slope (Al) values are close enough to call the linear regression robust. Also the density distribution of the slope looks not too bad.

Now we are done with all the necessary calculation for the OLR - let's start with the RTO.

6.1.3 Regression Trough Origin (RTO)

The procedure for calculating RTO and the associated parameters follows almost exactly the procedure for computing OLR. The main difference is that r^2 has to be calculated manually. Explanation for this can be found in Schauer 2024.

6.1.3.1 *Calculating the RTO*

source

Instructions following https://search.r-project.org/R/refmans/stats/html/lm.html

These lines of code give the most important information of the linear regression. Of particular interest are the spread of the residuals, the estimated values of intercept and slope (Al), the residual standard error (SEE) and the coefficient of determination (r^2). With regard to the latter, the 'multiple R-squared' value is relevant.

```
RTOAl<-lm(Al.WDXRF~Al-1, data=dataset)
summary(RTOAl)
```

```
Call:
lm(formula = Al.WDXRF ~ Al - 1, data = dataset)

Residuals:
   Min     1Q Median     3Q    Max
-16232  -6094  -1462   4515  23622

Coefficients:
   Estimate Std. Error t value Pr(>|t|)
Al  0.94432    0.02064   45.76   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 10560 on 29 degrees of freedom
Multiple R-squared:  0.9863,    Adjusted R-squared:  0.9859
F-statistic:  2094 on 1 and 29 DF,  p-value: < 2.2e-16
```

The residuals - as before for the OLR - show a large scatter. Again, the slope correction is acceptable, the SEE is quite high, while r^2 as calculated by the formula can't be used right away (see below).

6.1.3.2 *Calculating the Root Mean Squared error (RMS)*

source

Instructions following https://statisticsglobe.com/extract-fitted-values-from-regression-model-r and https://sparkbyexamples.com/r-programming/add-column-to-dataframe-in-r/

Again, in the sense of completeness, the RMS is calculated.

```
fitted.RTOAl <- fitted(RTOAl)
datasetc<-cbind(dataset,fitted.RTOAl)
RMS<-sqrt(mean((datasetc$Al-datasetc$fitted.RTOAl)^2)/nrow(datasetc))
RMS
```

```
[1] 950.017
```

6.1.3.3 *Calculating the relative SEE (rSEE)*

source

Instructions following https://www.rdocumentation.org/packages/stats/versions/3.6.2/topics/sigma and https://www.statology.org/r-mean-of-column/

Same for the rSEE - with the difference that there is no need to calculate the mean as it was already defined when computing the rSEE or the OLR (see above)

```
SEE<-sigma(RTOAl)
SEE
```

```
[1] 10563.36
```

Then the formula to calculate the relative residual standard error is defined and the rSEE for aluminium (Al) calculated.

```
mean<-mean(dataset$Al)
rSEE<-(SEE/mean)*100
rSEE
```

```
[1] 11.66953
```

Also in this case, the value of the rSEE is higher then 10% and thus above the threshold defined in Schauer 2024.

6.1.3.4 *Calculating the coefficient of determination (r^2)*

> **source**
>
> Instructions following https://pubs.cif-ifc.org/doi/pdf/10.5558/tfc71326-3 and https://onlinelibrary.wiley.com/doi/10.1111/1467-9639.00136

To calculate the corrected r^2 (cr2) that correctly describes the RTO, other variables have to be determined first. We already have the first one - the SEE (see above). Te next are the residual sum of squares - SSR - followed by the corrected total sum of squares - cSST. Using these three variables, cr2 can then be calculated. For the export of the graphics ([section -#sec-exgrapic]), every cr2 hast do be named after the element it belong to - e.g. Alcr2.

```
SSR<-(nrow(dataset)-1)*(SEE^2)
cSST <-sum(((dataset$Al.WDXRF)-mean(dataset$Al.WDXRF))^2)
Alcr2<-1-(SSR/cSST)
Alcr2
```

```
[1] 0.8277798
```

As for the OLR, r^2 is to low to be accepted.

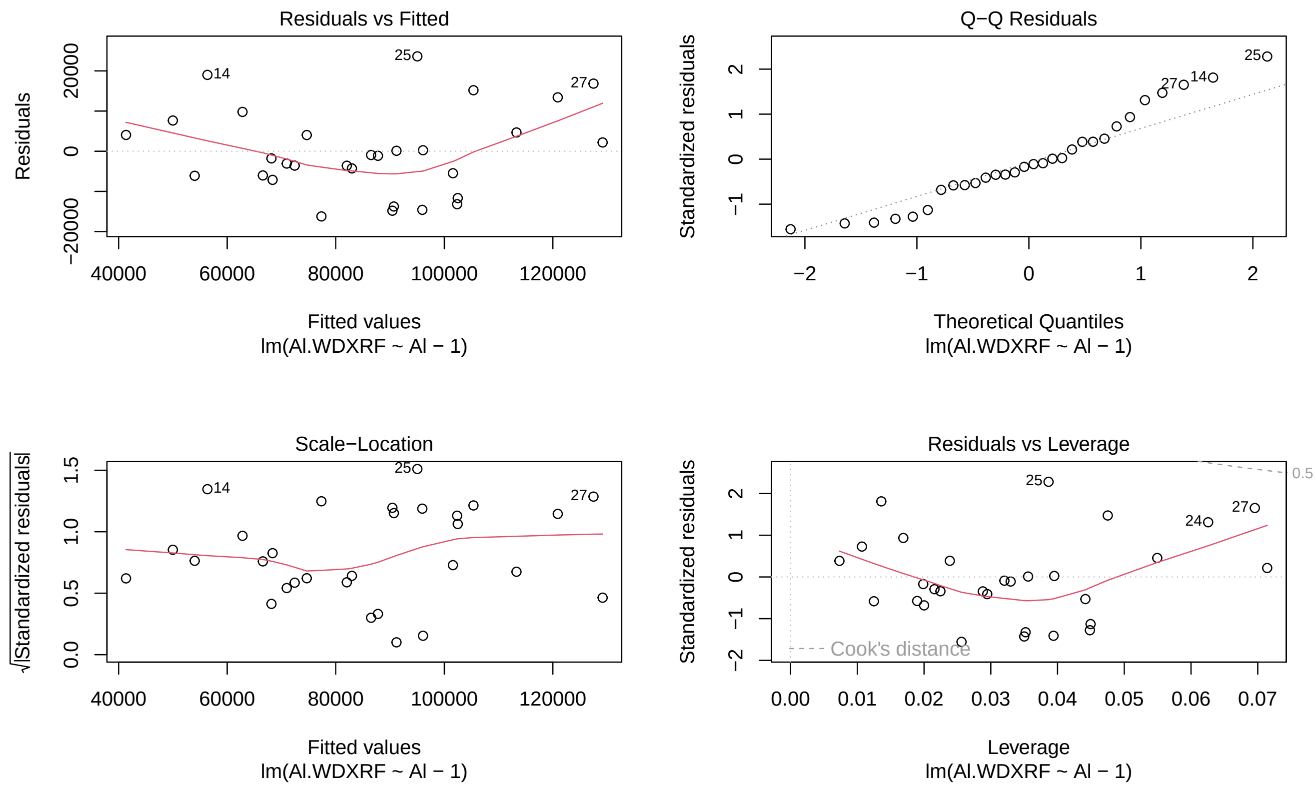
6.1.3.5 *Checking residuals*

> **source**

Instructions following https://www.rdocumentation.org/packages/stats/versions/3.6.2/topics/plot.lm

Again we also check the residuals.

```
oldparRTOAl <- par(oma=c(0,0,3,0), mfrow=c(2,2))
plot(RTOAl)
```




```
NULL
```

As before, in the Q-Q-plot, the sample in row 25 might be an outlier. So we have to check if it is by performing a Bonferroni test.

6.1.3.6 *Checking for significant outliers*

source

Instructions following https://rdrr.io/cran/car/src/R/outlierTest.R

Still, we perform the Bonferroni test.

```
outlierTest(RTOAl)
```

```
No Studentized residuals with Bonferroni p < 0.05
Largest |rstudent|:
```

```
   rstudent unadjusted p-value Bonferroni p
25 2.473844           0.019696      0.59089
```

As before, the sample in row 25 is a (non significant) outlier. According to the criteria defined in Schauer 2024 this means we will have to perform a second iteration.

6.1.3.7 *Testing for robustness - confidence intervals*

> **source**
>
> Instructions following https://www.rdocumentation.org/packages/stats/versions/3.6.2/topics/confint, https://www.rdocumentation.org/packages/boot/versions/1.3-28.1/topics/boot, https://www.rdocumentation.org/packages/boot/versions/1.3-28.1/topics/plot.boot and https://stopsack.github.io/risks/reference/confint.margstd_boot.html

Even, as we are clear how to proceed, checking the robustness of the RTO is always a good idea.

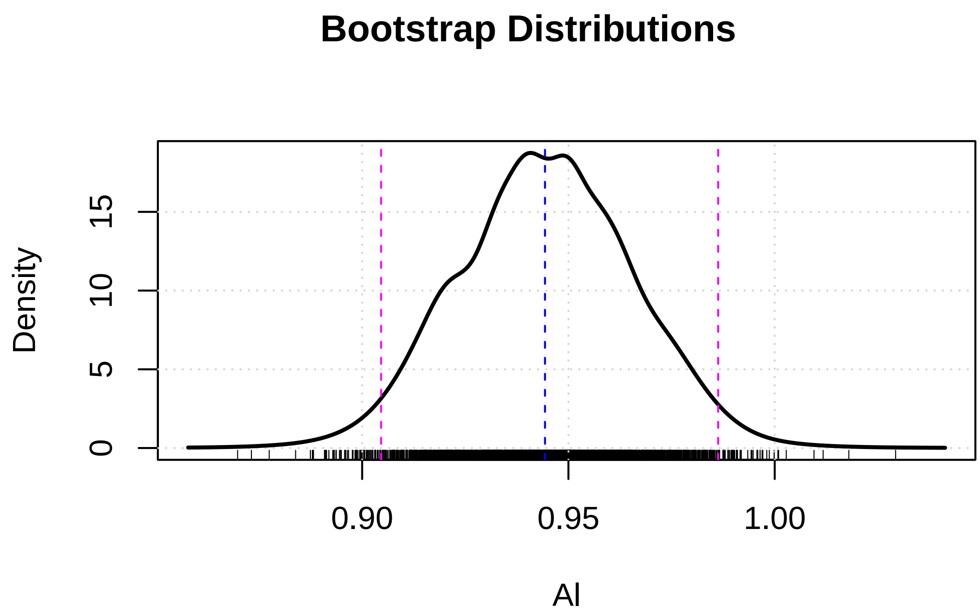
```
Confint(RTOAl, level=0.90)
```

```
    Estimate       5 %      95 %
Al 0.9443196 0.9092542 0.9793849
```

```
.bs.samplesRTOAl<- Boot(RTOAl, R=2500, method="case")
confint(.bs.samplesRTOAl, level=0.9, type="bca")
```

```
Bootstrap bca confidence intervals

         5 %      95 %
Al 0.9101073 0.9803524
```

```
plotBoot(.bs.samplesRTOAl)
```



According to the criteria defined in Schauer 2024, the difference in the slope (Al) values are small enough to call the linear regression robust. Also, the density distribution of the slope looks not too bad.

6.1.4 Result

The first iteration showed that the same outlier (sample in row 25) is visible in OLR as well as in RTO. Moreover, rSEE and r^2 for the two linear regressions are not within acceptable limits. Yet, robustness is given for both. Since three of the relevant criteria are not fulfilled, we carry out a second iteration of the analysis.

6.2 Second iteration

source

Instructions following https://www.digitalocean.com/community/tutorials/r-read-csv-file-into-data-frame and https://sparkbyexamples.com/r-programming/drop-dataframe-rows-in-r/

Since in our case we can improve the linear regression by removing the outlier - i.e. line 25 - we will do this for our second iteration.

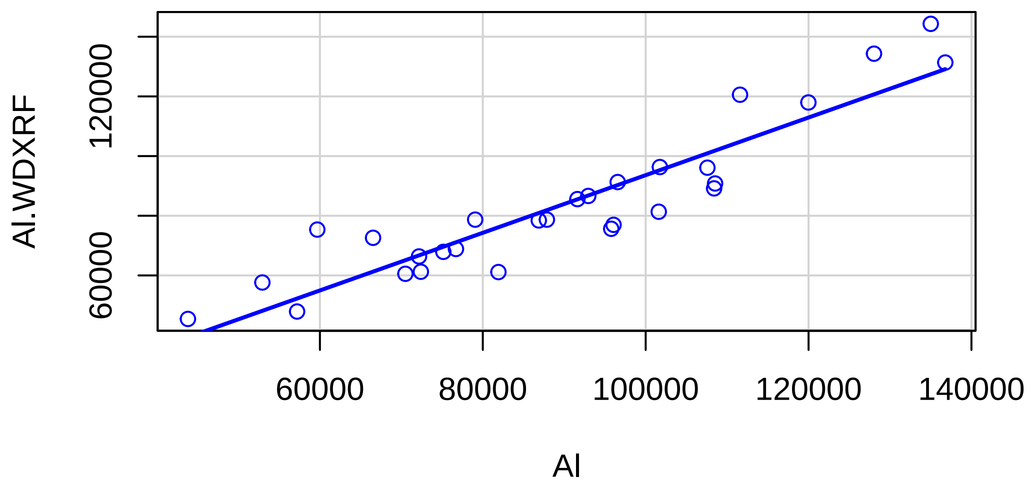
```
dataset1<- read.csv("../data_analytical//coefcorI_data.csv")
dataset<- dataset1[-c(25), ]
```

Thus, the data set we are now working with contains all samples except for the one in line 25. I.e. now all calculations for OLR and RTO are carried out again and checked for compliance with the criteria mentioned in Schauer 2024. Based on this result, it is then decided whether a further iteration is necessary or if one of the two linear regressions can be selected and the coefcors accepted.

calculations

6.2.1 First impressions

```
scatterplot(Al.WDXRF~Al, regLine=TRUE, smooth=FALSE, boxplots=FALSE, data=dataset)
```




```
cor(dataset$Al,dataset$Al.WDXRF)
```

```
[1] 0.9221522
```

```
numSummary(dataset[,c("Al", "Al.WDXRF"), drop=FALSE],
           statistics=c("mean", "sd", "quantiles"),quantiles=c(0,.25,.5,.75,1))
```

```
             mean       sd    0%   25%   50%    75%   100%  n
Al        90172.0 23958.74 43795 72395 91616 107579 136788 29
Al.WDXRF  84109.1 25096.90 45409 67902 78698  91294 144324 29
```

6.2.2 Ordinary Linear Regression (OLR)

```
OLRAl<-lm(Al.WDXRF~Al, data=dataset)
summary(OLRAl)
```

```
Call:
lm(formula = Al.WDXRF ~ Al, data = dataset)

Residuals:
```

```
     Min       1Q   Median       3Q      Max
-15011.0  -4813.6   -352.9   5317.8  20710.9

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept) -2.993e+03  7.267e+03  -0.412    0.684
Al           9.660e-01  7.798e-02  12.387  1.2e-12 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 9886 on 27 degrees of freedom
Multiple R-squared:  0.8504,    Adjusted R-squared:  0.8448
F-statistic: 153.4 on 1 and 27 DF,  p-value: 1.196e-12
```

```r
fitted.OLRAl <- fitted(OLRAl)
datasetb<-cbind(dataset,fitted.OLRAl)
RMS<-sqrt(mean((datasetb$Al-datasetb$fitted.OLRAl)^2)/nrow(datasetb))
RMS
```

```
[1] 1135.644
```

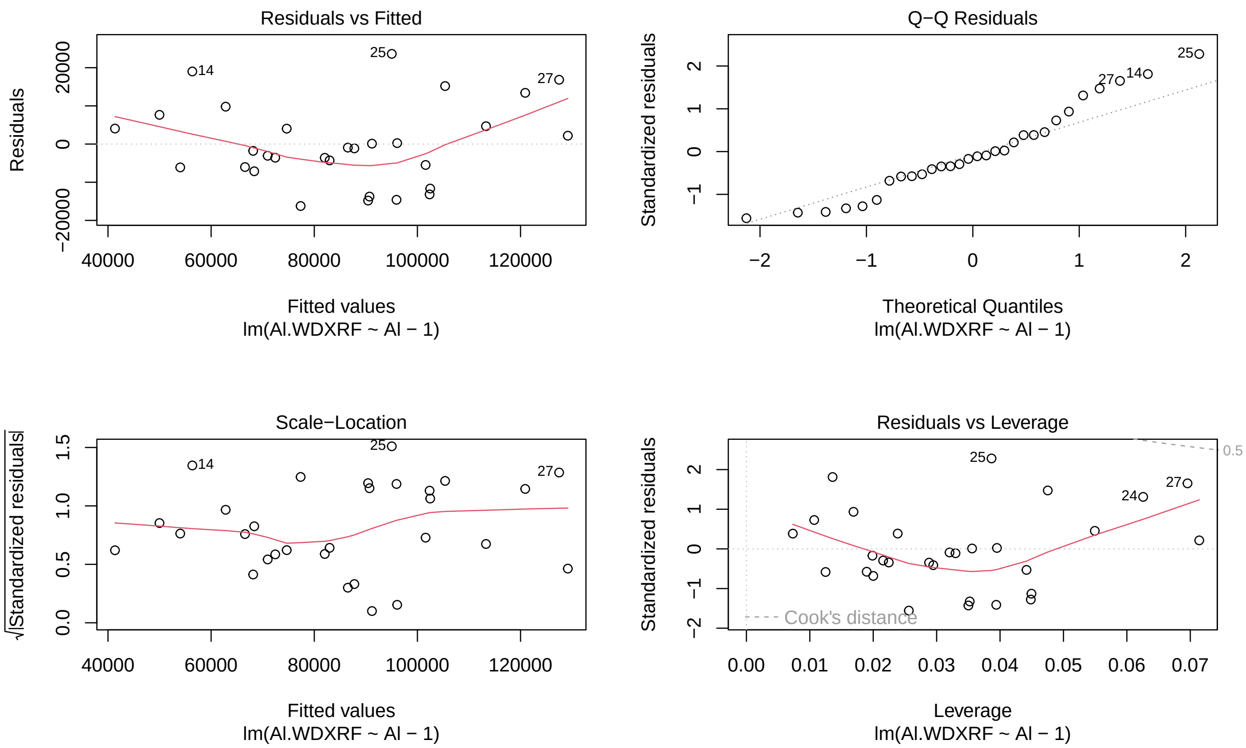
```r
SEE<-sigma(OLRAl)
SEE
```

```
[1] 9886.31
```

```r
mean<-mean(dataset$Al)
rSEE<-(SEE/mean)*100
rSEE
```

```
[1] 10.96384
```

```r
oldparOLRAl <- par(oma=c(0,0,3,0), mfrow=c(2,2))
plot(OLRAl)
```




```
NULL
```

```
outlierTest(OLRAl)
```

```
No Studentized residuals with Bonferroni p < 0.05
Largest |rstudent|:
   rstudent unadjusted p-value Bonferroni p
14 2.381525           0.024849      0.72061
```

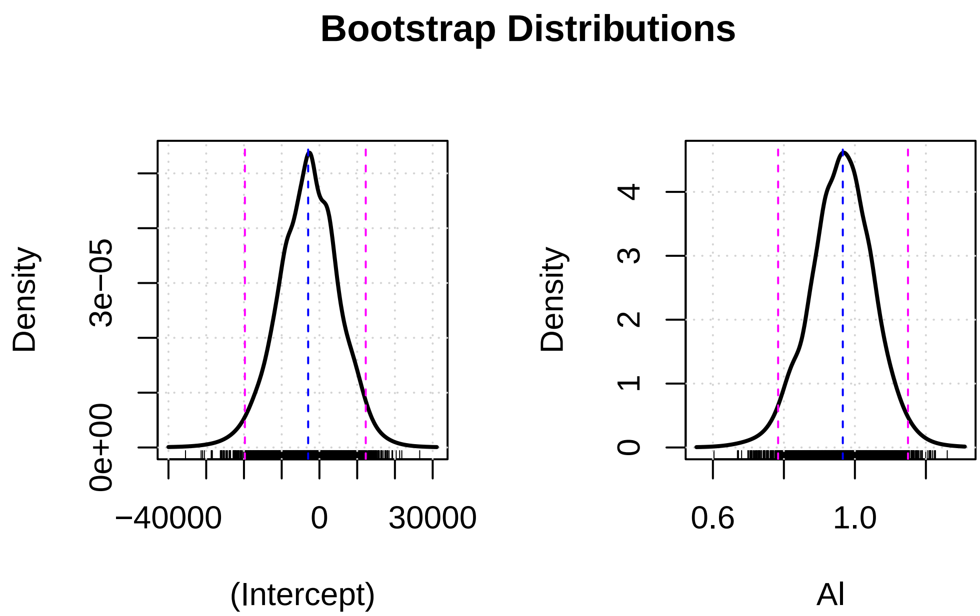
```
Confint(OLRAl, level=0.90)
```

```
                Estimate           5 %        95 %
(Intercept) -2993.351979 -1.537190e+04 9385.198412
Al              0.965959  8.331341e-01    1.098784
```

```
.bs.samplesOLRAl<- Boot(OLRAl, R=2500, method="case")
confint(.bs.samplesOLRAl, level=0.9, type="bca")
```

```
Bootstrap bca confidence intervals

                      5 %         95 %
(Intercept) -1.690438e+04 10106.961297
Al           8.135558e-01     1.116781
```

```
plotBoot(.bs.samplesOLRAl)
```



6.2.3 Regression Trough Origin (RTO)

```
RTOAl<-lm(Al.WDXRF~Al-1, data=dataset)
summary(RTOAl)
```

```
Call:
lm(formula = Al.WDXRF ~ Al - 1, data = dataset)

Residuals:
   Min     1Q Median     3Q    Max 
-15458  -5576  -1103   4782  19572 

Coefficients:
   Estimate Std. Error t value Pr(>|t|)    
Al   0.9349     0.0194   48.18   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 9739 on 28 degrees of freedom
Multiple R-squared:  0.9881,    Adjusted R-squared:  0.9877 
F-statistic:  2321 on 1 and 28 DF,  p-value: < 2.2e-16
```

```
fitted.RTOAl <- fitted(RTOAl)
datasetc<-cbind(dataset,fitted.RTOAl)
RMS<-sqrt(mean((datasetc$Al-datasetc$fitted.RTOAl)^2)/nrow(datasetc))
RMS
```

```
[1] 1126.93
```

```r
SEE<-sigma(RTOAl)
SEE
```

```
[1] 9738.615
```

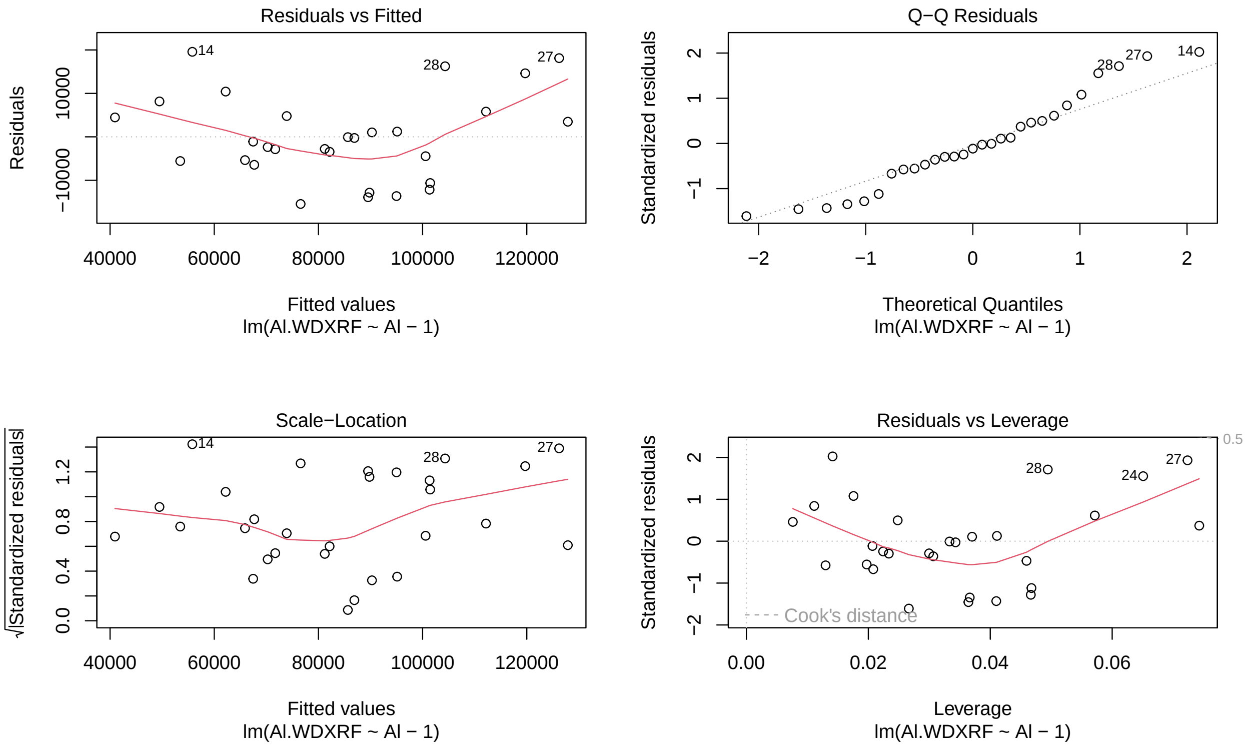
```r
mean<-mean(dataset$Al)
rSEE<-(SEE/mean)*100
rSEE
```

```
[1] 10.80004
```

```r
SSR<-(nrow(dataset)-1)*(SEE^2)
cSST <-sum(((dataset$Al.WDXRF)-mean(dataset$Al.WDXRF))^2)
Alcr2<-1-(SSR/cSST)
Alcr2
```

```
[1] 0.8494245
```

```r
oldparRTOAl <- par(oma=c(0,0,3,0), mfrow=c(2,2))
plot(RTOAl)
```




```
NULL
```

```r
outlierTest(RTOAl)
```

```
No Studentized residuals with Bonferroni p < 0.05
Largest |rstudent|:
   rstudent unadjusted p-value Bonferroni p
14 2.151244           0.040564           NA
```

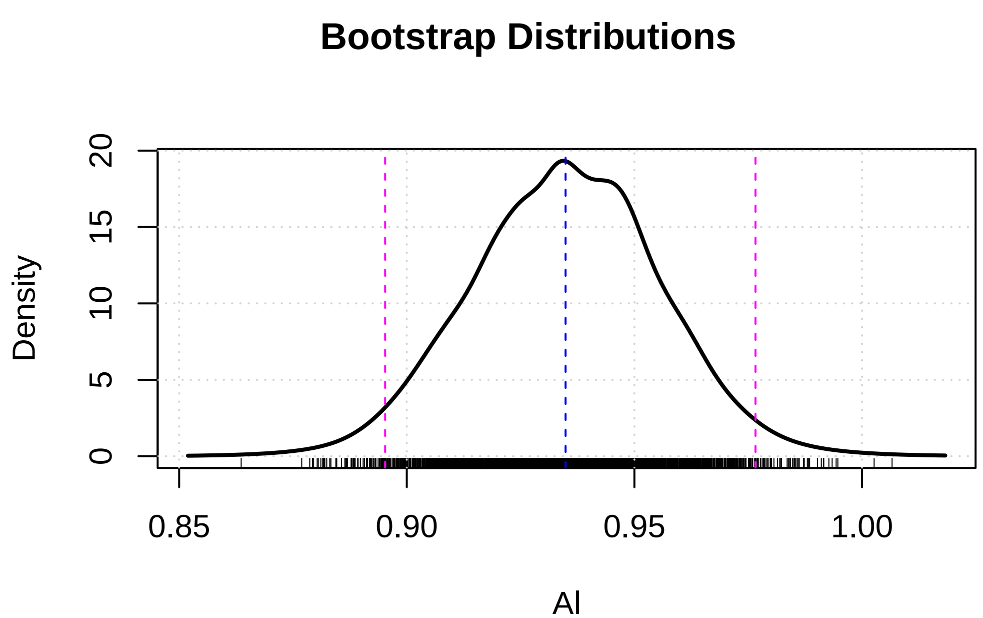
```
Confint(RTOAl, level=0.90)
```

```
    Estimate       5 %      95 %
Al 0.9348813 0.9018713 0.9678913
```

```
.bs.samplesRTOAl<- Boot(RTOAl, R=2500, method="case")
confint(.bs.samplesRTOAl, level=0.9, type="bca")
```

```
Bootstrap bca confidence intervals

         5 %      95 %
Al 0.9007452 0.9698718
```

```
plotBoot(.bs.samplesRTOAl)
```



6.2.4 Result

Both linear regressions give the sample in row 14 as a (non-significant) outlier. In addition, r^2 is too low, rSEE still too high for OLR and RTO, while robustness is fine. So, having seen that, we know that we need to do a third iteration.

6.3 Third iteration

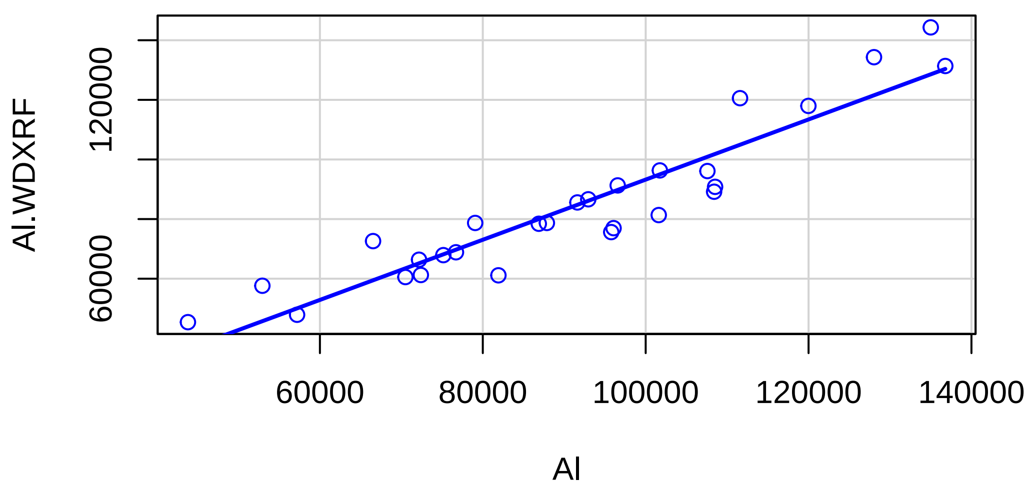
With this knowledge, for the third iteration, we now remove the sample in row 14 in addition to the one in row 25.

```
dataset1<- read.csv("../data_analytical//coefcorI_data.csv")
dataset<- dataset1[-c(25,14), ]
```

calculations

6.3.1 First impressions

```
scatterplot(Al.WDXRF~Al, regLine=TRUE, smooth=FALSE, boxplots=FALSE, data=dataset)
```




```
cor(dataset$Al,dataset$Al.WDXRF)
```

```
[1] 0.9362728
```

```
numSummary(dataset[,c("Al", "Al.WDXRF"), drop=FALSE],
           statistics=c("mean", "sd", "quantiles"),quantiles=c(0,.25,.5,.75,1))
```

```
             mean       sd    0%      25%     50%    75%   100%  n
Al       91261.07 23656.10 43795 74465.00 92284.5 107785 136788 28
Al.WDXRF 84421.43 25499.97 45409 67518.25 78698.0  92498 144324 28
```

6.3.2 Ordinary Linear Regression (OLR)

```
OLRA1<-lm(A1.WDXRF~A1, data=dataset)
summary(OLRA1)
```

```
Call:
lm(formula = A1.WDXRF ~ A1, data = dataset)

Residuals:
     Min       1Q   Median       3Q      Max
-13866.9  -4306.0    121.8   5068.6  15757.0

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept) -7.684e+03  6.993e+03  -1.099    0.282
A1           1.009e+00  7.426e-02  13.591 2.53e-13 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 9128 on 26 degrees of freedom
Multiple R-squared:  0.8766,    Adjusted R-squared:  0.8719
F-statistic: 184.7 on 1 and 26 DF,  p-value: 2.532e-13
```

```
fitted.OLRA1 <- fitted(OLRA1)
datasetb<-cbind(dataset,fitted.OLRA1)
RMS<-sqrt(mean((datasetb$A1-datasetb$fitted.OLRA1)^2)/nrow(datasetb))
RMS
```

```
[1] 1293.209
```

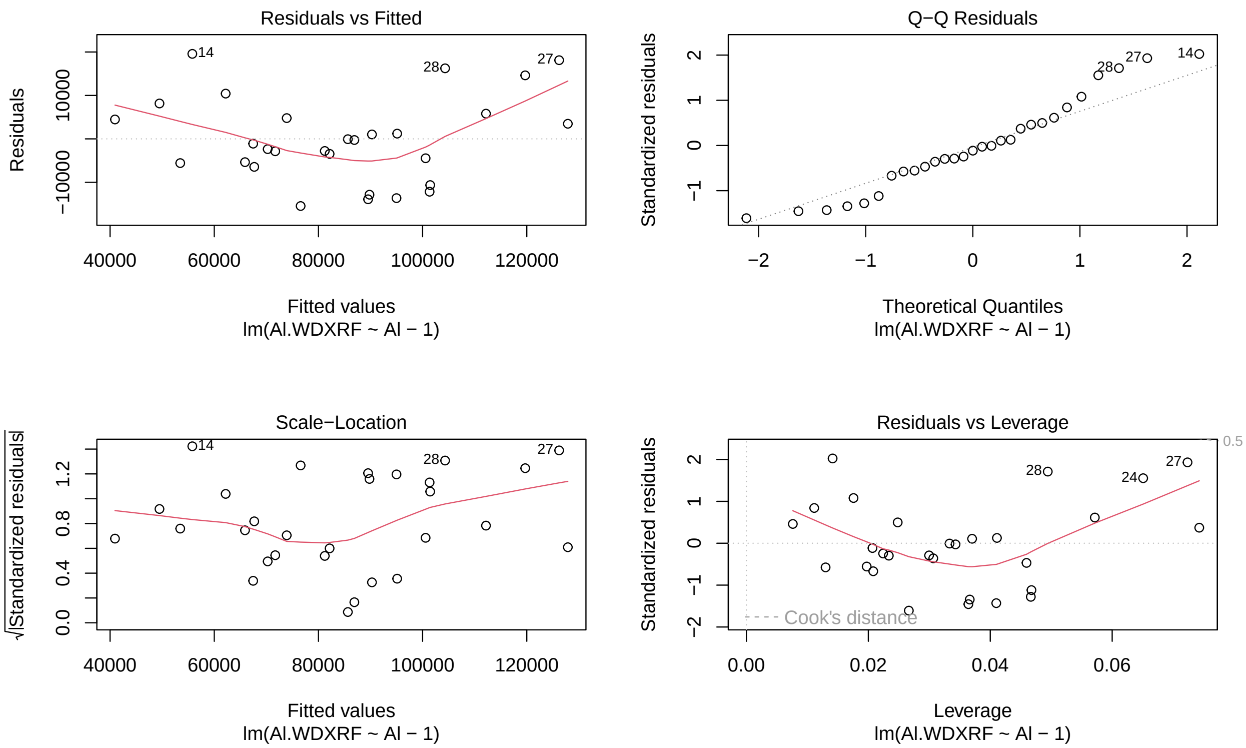
```
SEE<-sigma(OLRA1)
SEE
```

```
[1] 9128.106
```

```
mean<-mean(dataset$A1)
rSEE<-(SEE/mean)*100
rSEE
```

```
[1] 10.00219
```

```
oldparOLRA1 <- par(oma=c(0,0,3,0), mfrow=c(2,2))
plot(OLRA1)
```




```
NULL
```

```
outlierTest(OLRAl)
```

```
No Studentized residuals with Bonferroni p < 0.05
Largest |rstudent|:
   rstudent unadjusted p-value Bonferroni p
27 1.990638           0.057559           NA
```

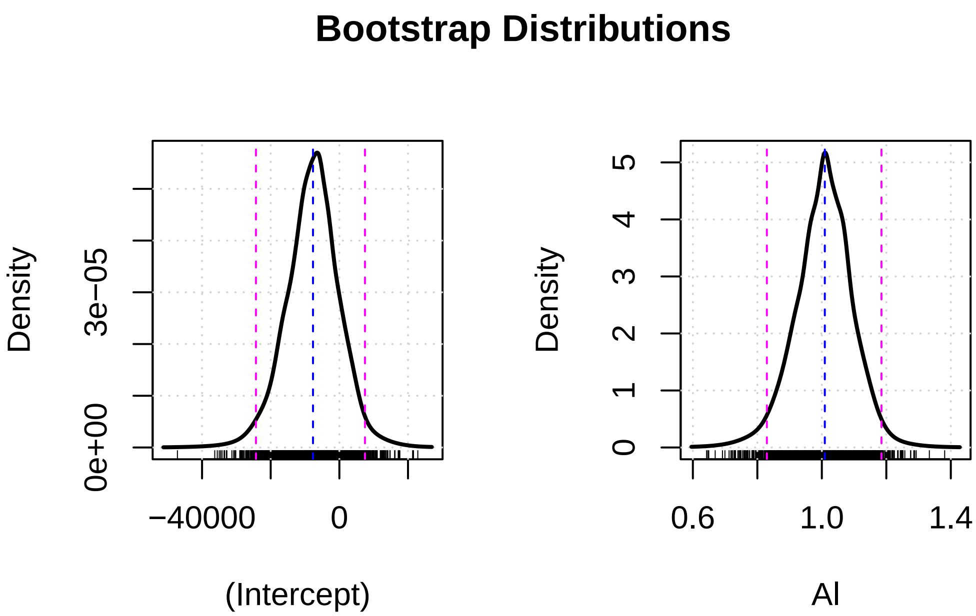
```
Confint(OLRAl, level=0.90)
```

```
               Estimate           5 %       95 %
(Intercept) -7683.85069 -1.961151e+04 4243.81165
Al              1.00925  8.825911e-01    1.13591
```

```
.bs.samplesOLRAl<- Boot(OLRAl, R=2500, method="case")
confint(.bs.samplesOLRAl, level=0.9, type="bca")
```

```
Bootstrap bca confidence intervals

                      5 %        95 %
(Intercept) -2.143322e+04 4490.556135
Al           8.596707e-01    1.156139
```

```
plotBoot(.bs.samplesOLRAl)
```



6.3.3 Regression Trough Origin (RTO)

```
RTOAl<-lm(Al.WDXRF~Al-1, data=dataset)
summary(RTOAl)
```

```
Call:
lm(formula = Al.WDXRF ~ Al - 1, data = dataset)

Residuals:
   Min     1Q Median     3Q    Max
-15073  -5507  -1385   4792  18748

Coefficients:
   Estimate Std. Error t value Pr(>|t|)
Al  0.93018    0.01839   50.59   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 9163 on 27 degrees of freedom
Multiple R-squared:  0.9896,    Adjusted R-squared:  0.9892
F-statistic:  2559 on 1 and 27 DF,  p-value: < 2.2e-16
```

```
fitted.RTOAl <- fitted(RTOAl)
datasetc<-cbind(dataset,fitted.RTOAl)
RMS<-sqrt(mean((datasetc$Al-datasetc$fitted.RTOAl)^2)/nrow(datasetc))
RMS
```

```
[1] 1242.61
```

```
SEE<-sigma(RTOAl)
SEE
```

```
[1] 9163.078
```

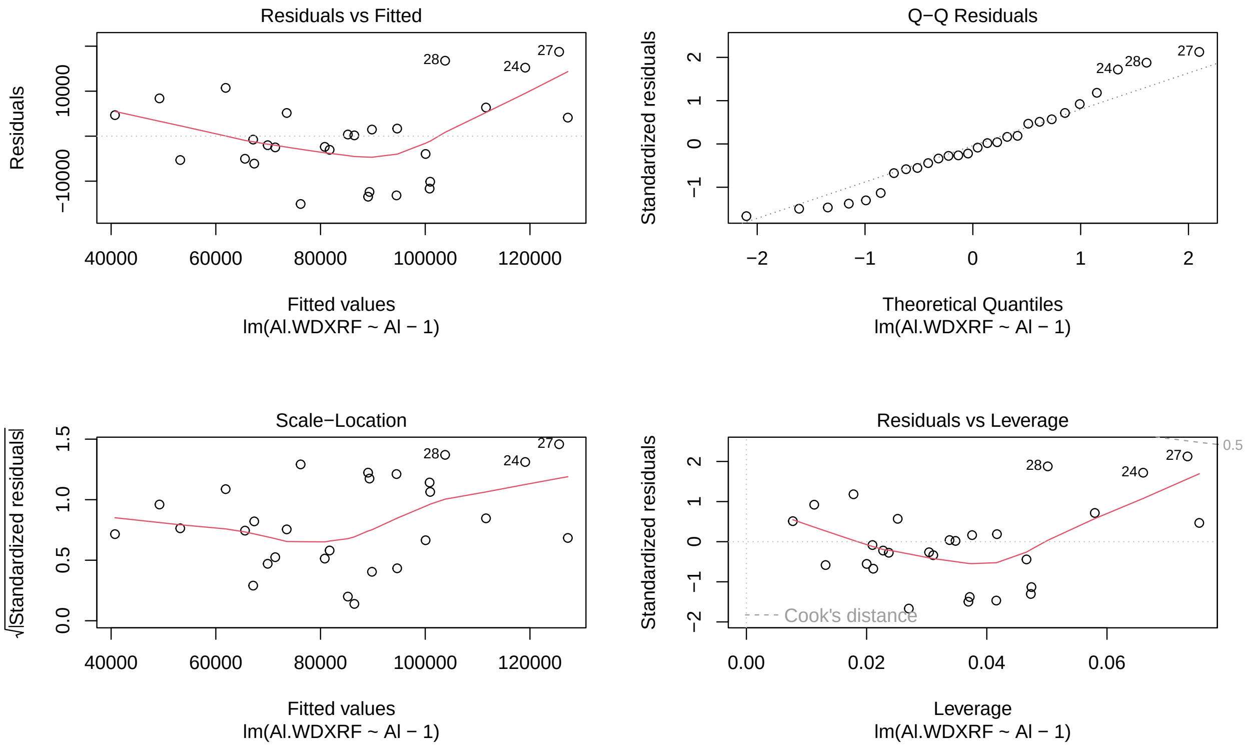
```
mean<-mean(dataset$Al)
rSEE<-(SEE/mean)*100
rSEE
```

```
[1] 10.04051
```

```
SSR<-(nrow(dataset)-1)*(SEE^2)
cSST <-sum(((dataset$Al.WDXRF)-mean(dataset$Al.WDXRF))^2)
Alcr2<-1-(SSR/cSST)
Alcr2
```

```
[1] 0.8708771
```

```
oldparRTOAl <- par(oma=c(0,0,3,0), mfrow=c(2,2))
plot(RTOAl)
```




```
NULL
```

```
outlierTest(RTOAl)
```

```
No Studentized residuals with Bonferroni p < 0.05
Largest |rstudent|:
   rstudent unadjusted p-value Bonferroni p
27 2.285811           0.030661       0.8585
```

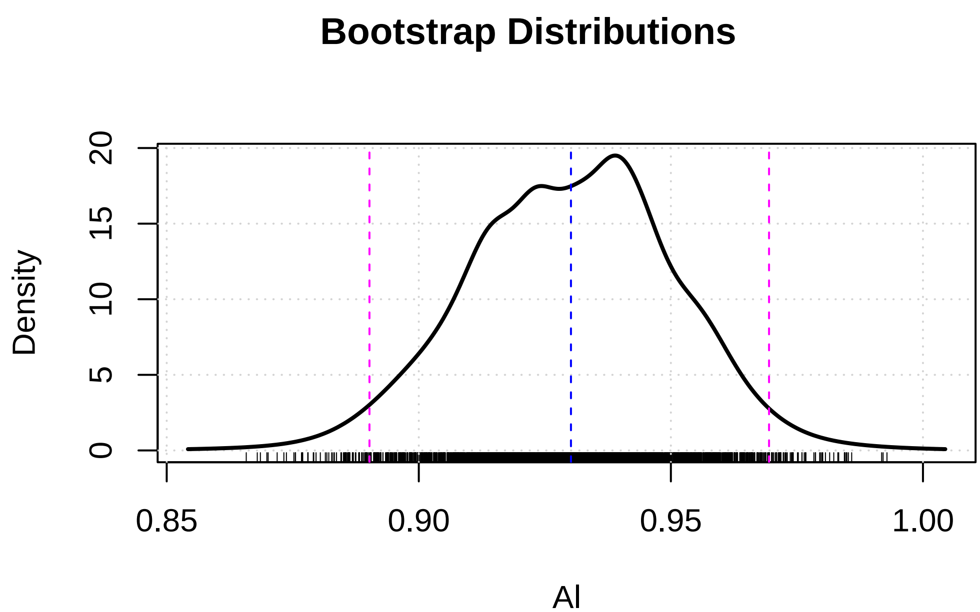
```r
Confint(RTOAl, level=0.90)
```

```
    Estimate       5 %      95 %
Al 0.9301774 0.8988566 0.9614982
```

```r
.bs.samplesRTOAl<- Boot(RTOAl, R=2500, method="case")
confint(.bs.samplesRTOAl, level=0.9, type="bca")
```

```
Bootstrap bca confidence intervals

         5 %      95 %
Al 0.8954861 0.9631428
```

```r
plotBoot(.bs.samplesRTOAl)
```



6.3.4 Result

Both linear regressions give the sample in row 27 as a (non-significant) outlier. Again, r^2 is too low, rSEE still too high. Robustness is fine for OLR and RTO. So on to iteration four.

6.4 Fourth iteration

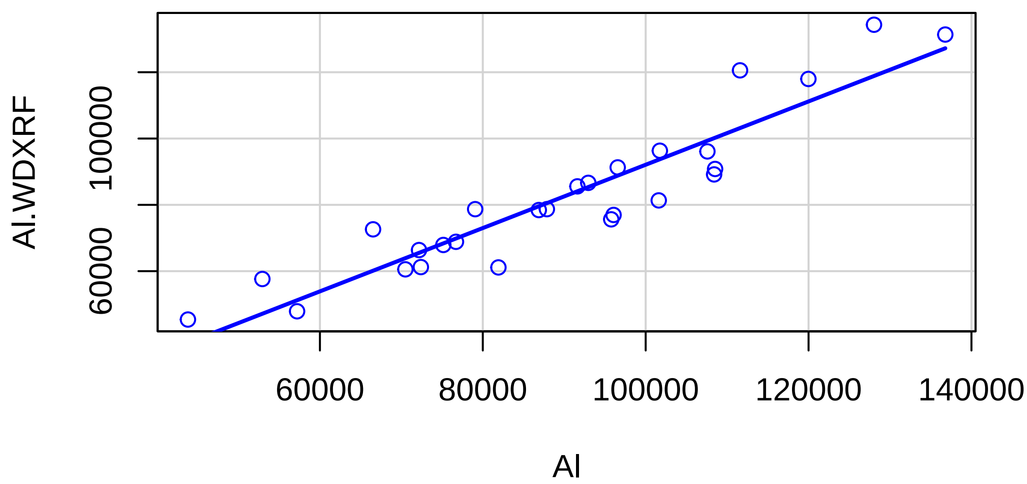
We additionally remove row 27 from the data set.

```
dataset1<- read.csv("../data_analytical//coefcorI_data.csv")
dataset<- dataset1[-c(25,14,27), ]
```

calculations

6.4.1 First impressions

```
scatterplot(Al.WDXRF~Al, regLine=TRUE, smooth=FALSE, boxplots=FALSE, data=dataset)
```




```
cor(dataset$Al,dataset$Al.WDXRF)
```

```
[1] 0.9299688
```

```
numSummary(dataset[,c("Al", "Al.WDXRF"), drop=FALSE],
           statistics=c("mean", "sd", "quantiles"),quantiles=c(0,.25,.5,.75,1))
```

```
             mean       sd    0%     25%   50%      75%   100%  n
Al       89641.04 22468.24 43795 73775.0 91616 104660.5 136788 27
Al.WDXRF 82202.81 23068.04 45409 67134.5 78698  91056.0 134321 27
```

6.4.2 Ordinary Linear Regression (OLR)

```
OLRAl<-lm(Al.WDXRF~Al, data=dataset)
summary(OLRAl)
```

```
Call:
lm(formula = Al.WDXRF ~ Al, data = dataset)

Residuals:
     Min       1Q   Median       3Q      Max 
-13703.8  -3940.8   -469.6   5366.3  17405.3 

Coefficients:
              Estimate Std. Error t value Pr(>|t|)    
(Intercept) -3.386e+03  6.969e+03  -0.486    0.631    
Al           9.548e-01  7.549e-02  12.648 2.31e-12 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 8649 on 25 degrees of freedom
Multiple R-squared:  0.8648,	Adjusted R-squared:  0.8594 
F-statistic:   160 on 1 and 25 DF,  p-value: 2.308e-12
```

```
fitted.OLRAl <- fitted(OLRAl)
datasetb<-cbind(dataset,fitted.OLRAl)
RMS<-sqrt(mean((datasetb$Al-datasetb$fitted.OLRAl)^2)/nrow(datasetb))
RMS
```

```
[1] 1444.281
```

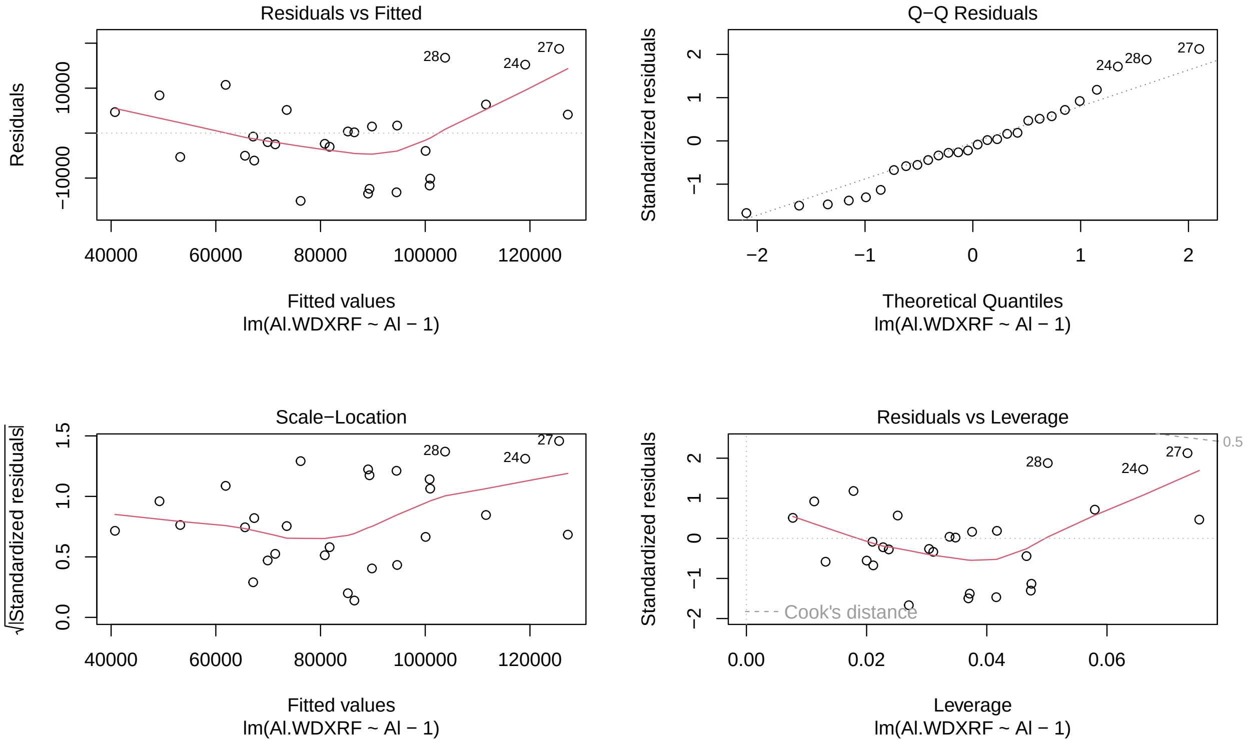
```
SEE<-sigma(OLRAl)
SEE
```

```
[1] 8648.649
```

```
mean<-mean(dataset$Al)
rSEE<-(SEE/mean)*100
rSEE
```

```
[1] 9.648091
```

```
oldparOLRAl <- par(oma=c(0,0,3,0), mfrow=c(2,2))
plot(OLRAl)
```




```
NULL
```

```r
outlierTest(OLRAl)
```

```
No Studentized residuals with Bonferroni p < 0.05
Largest |rstudent|:
   rstudent unadjusted p-value Bonferroni p
28  2.25551           0.033493      0.90431
```

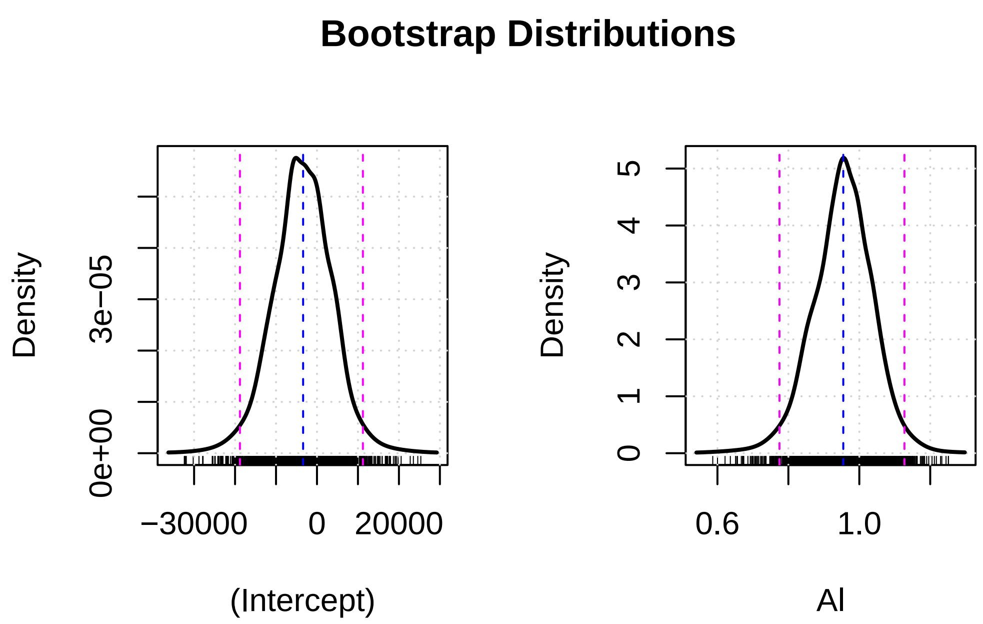
```r
Confint(OLRAl, level=0.90)
```

```
                 Estimate          5 %        95 %
(Intercept) -3385.9613614 -1.528955e+04 8517.624376
Al              0.9547946  8.258461e-01    1.083743
```

```r
.bs.samplesOLRAl<- Boot(OLRAl, R=2500, method="case")
confint(.bs.samplesOLRAl, level=0.9, type="bca")
```

```
Bootstrap bca confidence intervals

                      5 %        95 %
(Intercept) -1.546008e+04 8537.623131
Al           8.116031e-01    1.091759
```

```r
plotBoot(.bs.samplesOLRAl)
```



6.4.3 Regression Trough Origin (RTO)

```
RTOAl<-lm(Al.WDXRF~Al-1, data=dataset)
summary(RTOAl)
```

```
Call:
lm(formula = Al.WDXRF ~ Al - 1, data = dataset)

Residuals:
   Min     1Q Median     3Q    Max
-14172  -4994  -1179   5390  17994

Coefficients:
   Estimate Std. Error t value Pr(>|t|)
Al  0.91918    0.01776   51.74   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 8521 on 26 degrees of freedom
Multiple R-squared:  0.9904,    Adjusted R-squared:   0.99
F-statistic:  2678 on 1 and 26 DF,  p-value: < 2.2e-16
```

```
fitted.RTOAl <- fitted(RTOAl)
datasetc<-cbind(dataset,fitted.RTOAl)
RMS<-sqrt(mean((datasetc$Al-datasetc$fitted.RTOAl)^2)/nrow(datasetc))
RMS
```

```
[1] 1435.87
```

```
SEE<-sigma(RTOAl)
SEE
```

```
[1] 8520.646
```

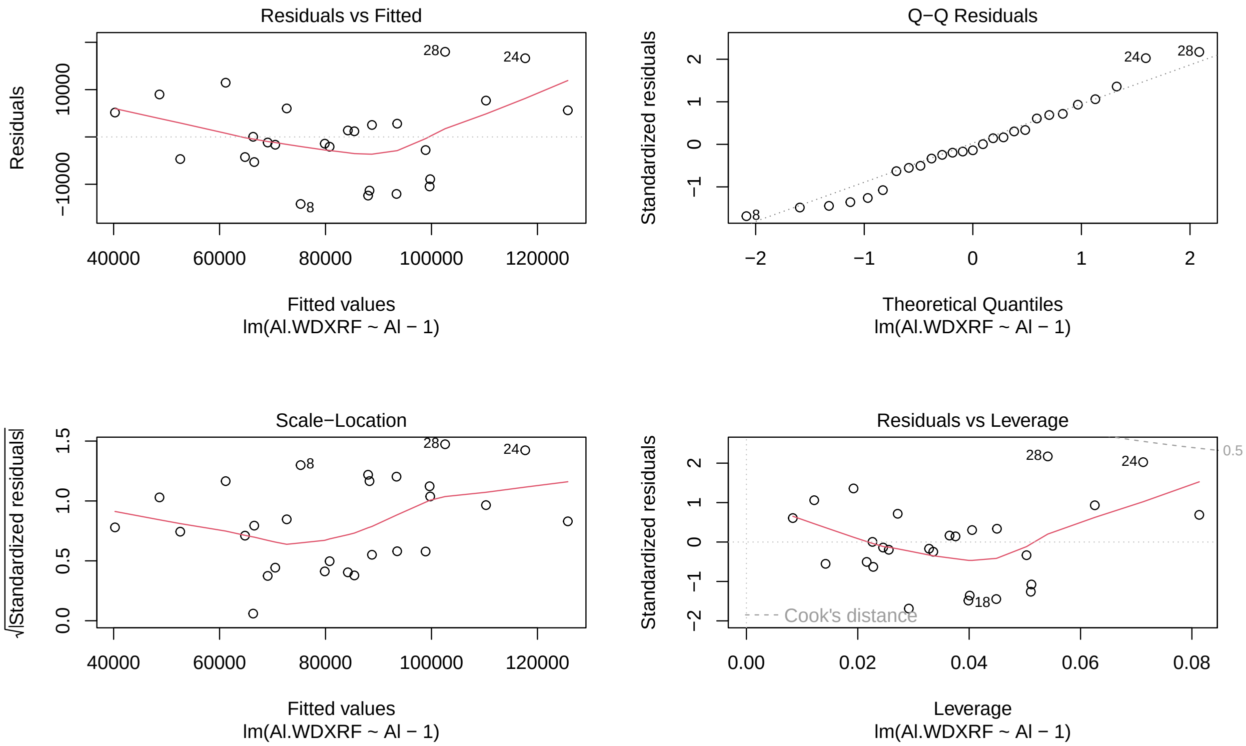
```
mean<-mean(dataset$Al)
rSEE<-(SEE/mean)*100
rSEE
```

```
[1] 9.505296
```

```
SSR<-(nrow(dataset)-1)*(SEE^2)
cSST <-sum(((dataset$Al.WDXRF)-mean(dataset$Al.WDXRF))^2)
Alcr2<-1-(SSR/cSST)
Alcr2
```

```
[1] 0.8635656
```

```
oldparRTOAl <- par(oma=c(0,0,3,0), mfrow=c(2,2))
plot(RTOAl)
```




```
NULL
```

```
outlierTest(RTOAl)
```

```
No Studentized residuals with Bonferroni p < 0.05
Largest |rstudent|:
   rstudent unadjusted p-value Bonferroni p
28  2.35321           0.026784      0.72315
```

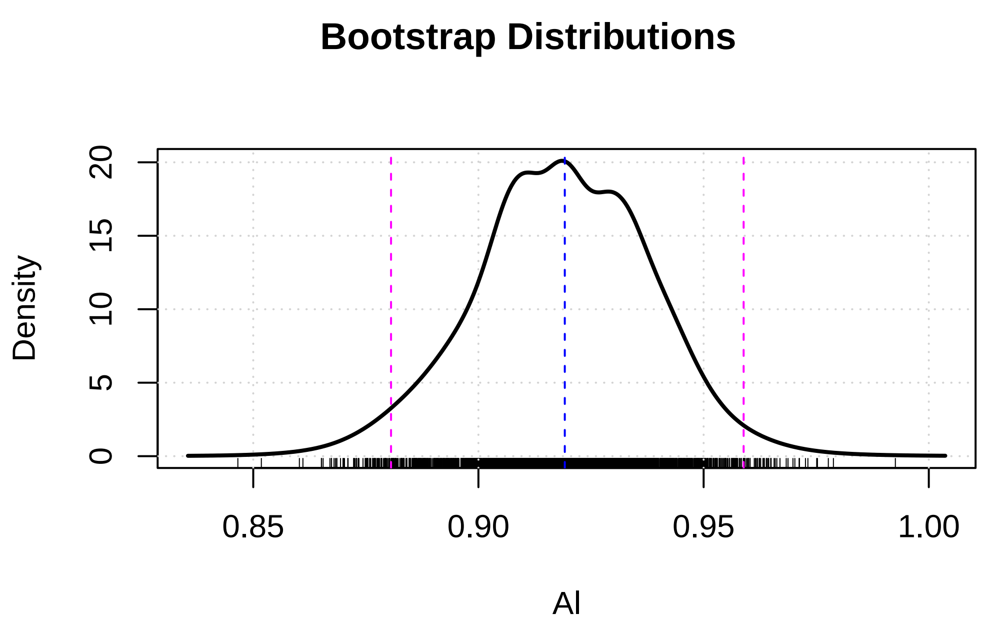
```
Confint(RTOAl, level=0.90)
```

```
    Estimate       5 %      95 %
Al 0.9191769 0.8888791 0.9494747
```

```
.bs.samplesRTOAl<- Boot(RTOAl, R=2500, method="case")
confint(.bs.samplesRTOAl, level=0.9, type="bca")
```

```
Bootstrap bca confidence intervals

         5 %     95 %
Al 0.8866721 0.951634
```

```
plotBoot(.bs.samplesRTOAl)
```



6.4.4 Result

Both linear regressions give the sample in row 28 as a (non-significant) outlier. Again, r^2 is too low, but rSEE and robustness are fine for OLR and RTO. So on to iteration fife.

6.5 Fifth iteration

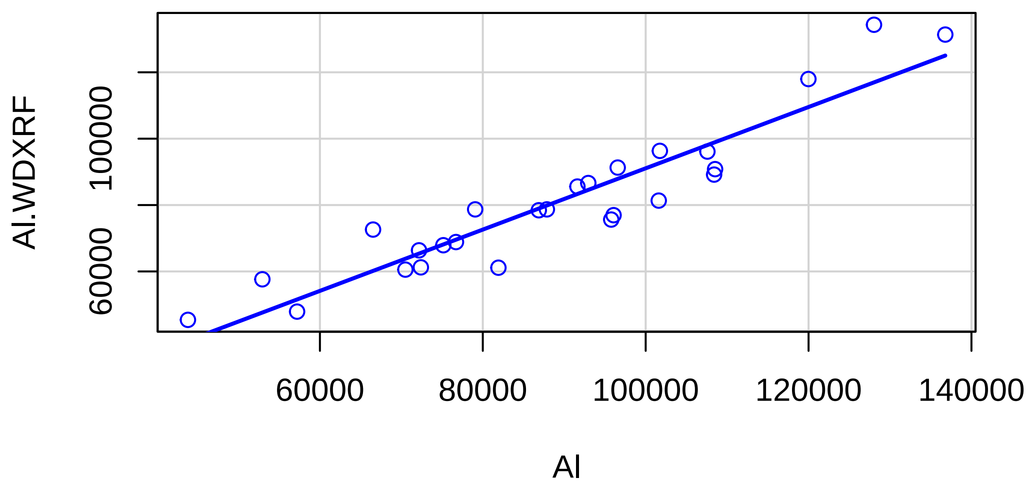
For the fifth iteration we additionally exclude row 28.

```
dataset1<- read.csv("../data_analytical//coefcorI_data.csv")
dataset<- dataset1[-c(25,14,27,28), ]
```

calculations

6.5.1 First impressions

```
scatterplot(Al.WDXRF~Al, regLine=TRUE, smooth=FALSE, boxplots=FALSE, data=dataset)
```




```
cor(dataset$Al,dataset$Al.WDXRF)
```

```
[1] 0.9352198
```

```
numSummary(dataset[,c("Al", "Al.WDXRF"), drop=FALSE],
           statistics=c("mean", "sd", "quantiles"),quantiles=c(0,.25,.5,.75,1))
```

```
            mean       sd    0%      25%     50%       75%   100%  n
Al       88797.0 22472.44 43795 73085.00 89743.0 101707.50 136788 26
Al.WDXRF 80727.5 22187.88 45409 66750.75 78565.5  90407.75 134321 26
```

6.5.2 Ordinary Linear Regression (OLR)

```
OLRAl<-lm(Al.WDXRF~Al, data=dataset)
summary(OLRAl)
```

```
Call:
lm(formula = Al.WDXRF ~ Al, data = dataset)

Residuals:
     Min       1Q   Median       3Q      Max 
-13250.4  -4175.1   -375.2   5586.8  17360.2 

Coefficients:
              Estimate Std. Error t value Pr(>|t|)    
(Intercept) -1.266e+03  6.529e+03  -0.194    0.848    
Al           9.234e-01  7.136e-02  12.940 2.58e-12 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 8018 on 24 degrees of freedom
Multiple R-squared:  0.8746,    Adjusted R-squared:  0.8694 
F-statistic: 167.4 on 1 and 24 DF,  p-value: 2.583e-12
```

```
fitted.OLRAl <- fitted(OLRAl)
datasetb<-cbind(dataset,fitted.OLRAl)
RMS<-sqrt(mean((datasetb$Al-datasetb$fitted.OLRAl)^2)/nrow(datasetb))
RMS
```

```
[1] 1616.831
```

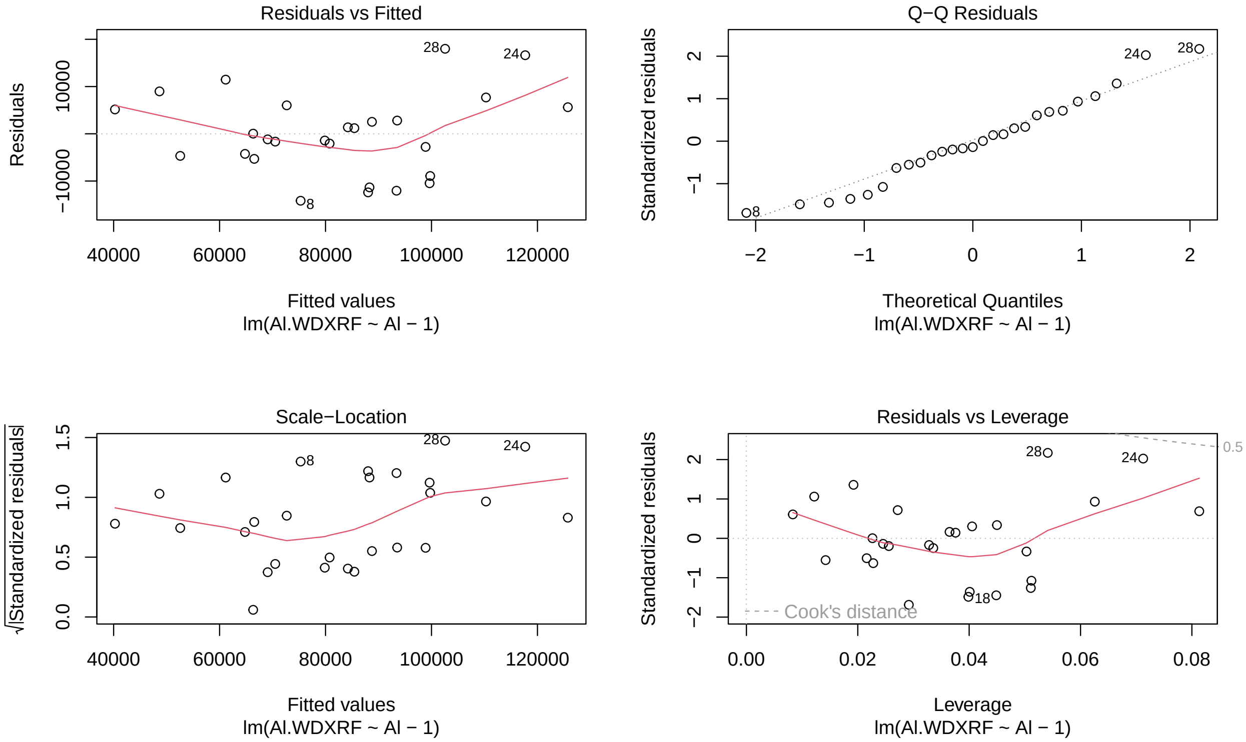
```
SEE<-sigma(OLRAl)
SEE
```

```
[1] 8018.006
```

```
mean<-mean(dataset$Al)
rSEE<-(SEE/mean)*100
rSEE
```

```
[1] 9.029591
```

```
oldparOLRAl <- par(oma=c(0,0,3,0), mfrow=c(2,2))
plot(OLRAl)
```




```
NULL
```

```r
outlierTest(OLRAl)
```

```
No Studentized residuals with Bonferroni p < 0.05
Largest |rstudent|:
   rstudent unadjusted p-value Bonferroni p
24 2.640695           0.014612      0.37991
```

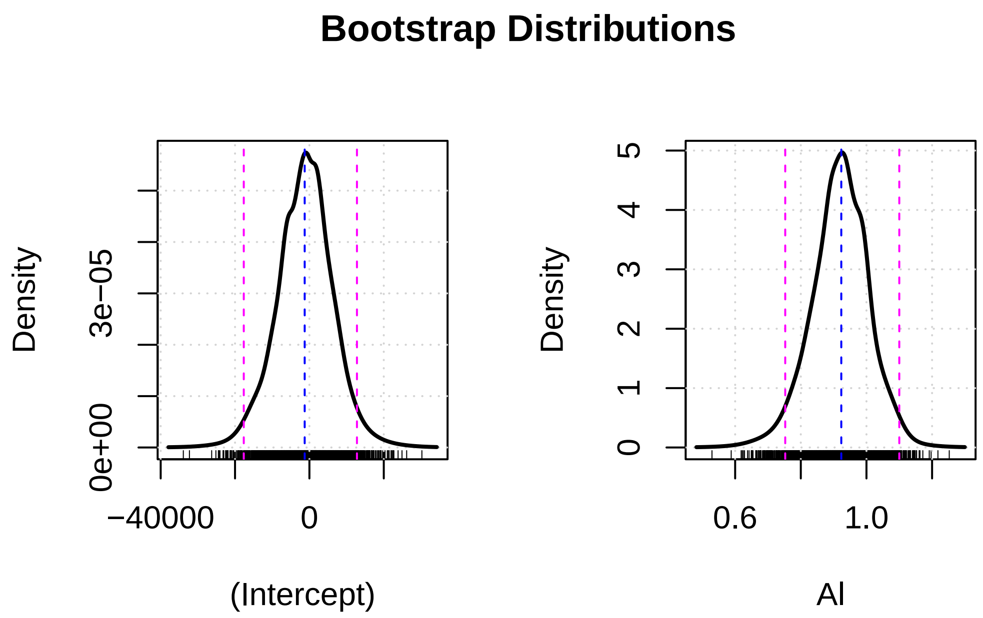
```r
Confint(OLRAl, level=0.90)
```

```
                 Estimate           5 %        95 %
(Intercept) -1265.6557394 -1.243536e+04 9904.052039
Al              0.9233775  8.012914e-01    1.045464
```

```r
.bs.samplesOLRAl<- Boot(OLRAl, R=2500, method="case")
confint(.bs.samplesOLRAl, level=0.9, type="bca")
```

```
Bootstrap bca confidence intervals

                     5 %         95 %
(Intercept) -1.531969e+04 10203.353624
Al           7.814904e-01     1.076988
```

```r
plotBoot(.bs.samplesOLRAl)
```



6.5.3 Regression Trough Origin (RTO)

```
RTOAl<-lm(Al.WDXRF~Al-1, data=dataset)
summary(RTOAl)
```

```
Call:
lm(formula = Al.WDXRF ~ Al - 1, data = dataset)

Residuals:
     Min       1Q   Median       3Q      Max
-13416.2  -4519.8   -553.1   5103.7  17813.6

Coefficients:
   Estimate Std. Error t value Pr(>|t|)
Al  0.90995    0.01685   53.99   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 7862 on 25 degrees of freedom
Multiple R-squared:  0.9915,	Adjusted R-squared:  0.9912
F-statistic:  2915 on 1 and 25 DF,  p-value: < 2.2e-16
```

```
fitted.RTOAl <- fitted(RTOAl)
datasetc<-cbind(dataset,fitted.RTOAl)
RMS<-sqrt(mean((datasetc$Al-datasetc$fitted.RTOAl)^2)/nrow(datasetc))
RMS
```

```
[1] 1615.725
```

```r
SEE<-sigma(RTOAl)
SEE
```

```
[1] 7862.158
```

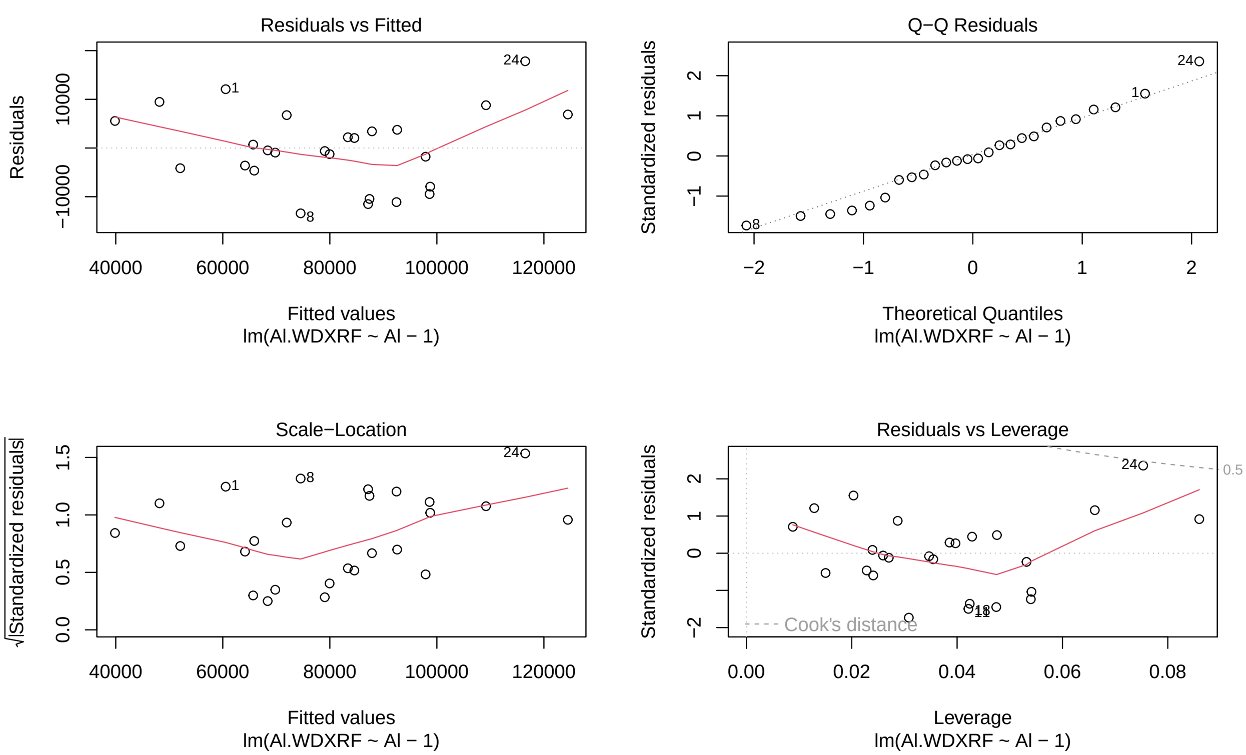
```r
mean<-mean(dataset$Al)
rSEE<-(SEE/mean)*100
rSEE
```

```
[1] 8.854081
```

```r
SSR<-(nrow(dataset)-1)*(SEE^2)
cSST <-sum(((dataset$Al.WDXRF)-mean(dataset$Al.WDXRF))^2)
Alcr2<-1-(SSR/cSST)
Alcr2
```

```
[1] 0.8744398
```

```r
oldparRTOAl <- par(oma=c(0,0,3,0), mfrow=c(2,2))
plot(RTOAl)
```




```
NULL
```

```r
outlierTest(RTOAl)
```

```
No Studentized residuals with Bonferroni p < 0.05
Largest |rstudent|:
   rstudent unadjusted p-value Bonferroni p
24 2.617467           0.015096      0.39249
```

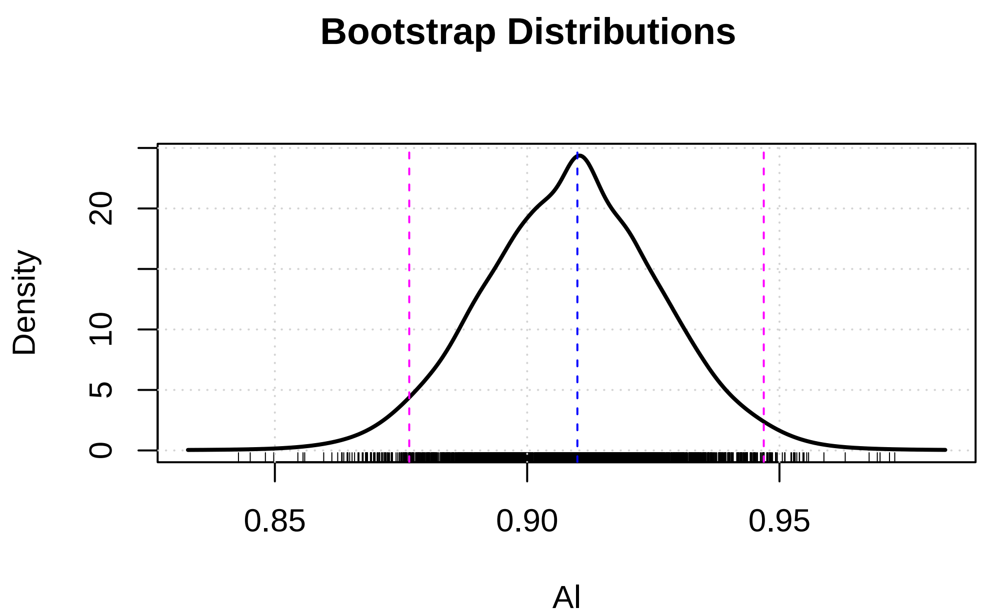
```
Confint(RTOAl, level=0.90)
```

```
   Estimate       5 %      95 %
Al 0.909951 0.8811636 0.9387385
```

```
.bs.samplesRTOAl<- Boot(RTOAl, R=2500, method="case")
confint(.bs.samplesRTOAl, level=0.9, type="bca")
```

```
Bootstrap bca confidence intervals

         5 %      95 %
Al 0.8815718 0.9421537
```

```
plotBoot(.bs.samplesRTOAl)
```



6.5.4 Result

In the fifth iteration, r^2 is still too low for OLR and RTO, but rSEE is within the required range and robustness is also given. Both linear regressions show the sample in row 24 as an outlier, which is why it is removed for the sixth iteration.

6.6 Sixth iteration

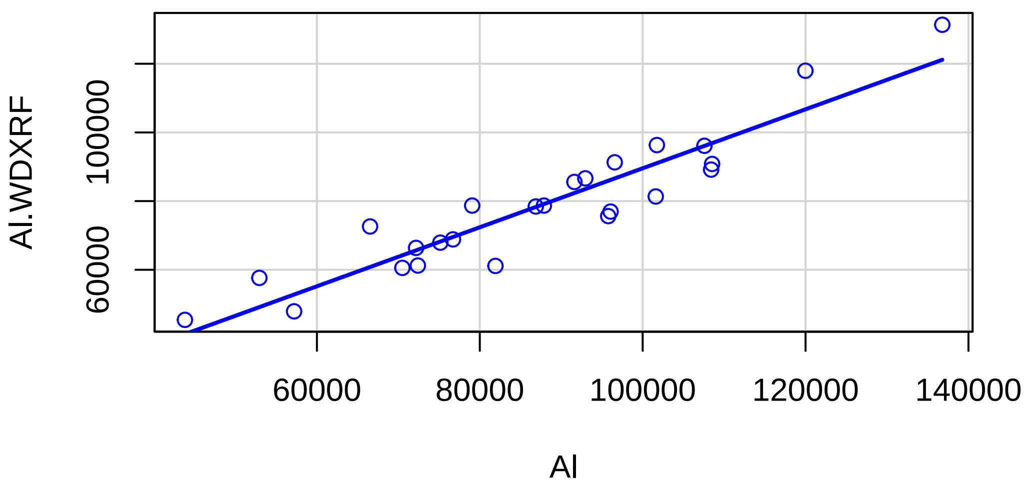
Now we exclude row 24.

```
dataset1<- read.csv("../data_analytical//coefcorI_data.csv")
dataset<- dataset1[-c(25,14,27,28,24), ]
```

calculations

6.6.1 First impressions

```
scatterplot(Al.WDXRF~Al, regLine=TRUE, smooth=FALSE, boxplots=FALSE, data=dataset)
```




```
cor(dataset$Al,dataset$Al.WDXRF)
```

```
[1] 0.9343291
```

```
numSummary(dataset[,c("Al", "Al.WDXRF"), drop=FALSE],
           statistics=c("mean", "sd", "quantiles"),quantiles=c(0,.25,.5,.75,1))
```

```
             mean       sd    0%   25%   50%    75%   100%  n
Al       87227.40 21431.96 43795 72395 87870 101604 136788 25
Al.WDXRF 78583.76 19706.59 45409 66367 78433  89177 131358 25
```

6.6.2 Ordinary Linear Regression (OLR)

```
OLRAl<-lm(Al.WDXRF~Al, data=dataset)
summary(OLRAl)
```

```
Call:
lm(formula = Al.WDXRF ~ Al, data = dataset)

Residuals:
     Min       1Q   Median       3Q      Max 
-12897.1  -4888.3     41.9   4687.3  11813.1 

Coefficients:
             Estimate Std. Error t value Pr(>|t|)    
(Intercept) 3.646e+03  6.131e+03   0.595    0.558    
Al          8.591e-01  6.833e-02  12.572 8.67e-12 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 7175 on 23 degrees of freedom
Multiple R-squared:  0.873,	Adjusted R-squared:  0.8674 
F-statistic: 158.1 on 1 and 23 DF,  p-value: 8.675e-12
```

```
fitted.OLRAl <- fitted(OLRAl)
datasetb<-cbind(dataset,fitted.OLRAl)
RMS<-sqrt(mean((datasetb$Al-datasetb$fitted.OLRAl)^2)/nrow(datasetb))
RMS
```

```
[1] 1827.187
```

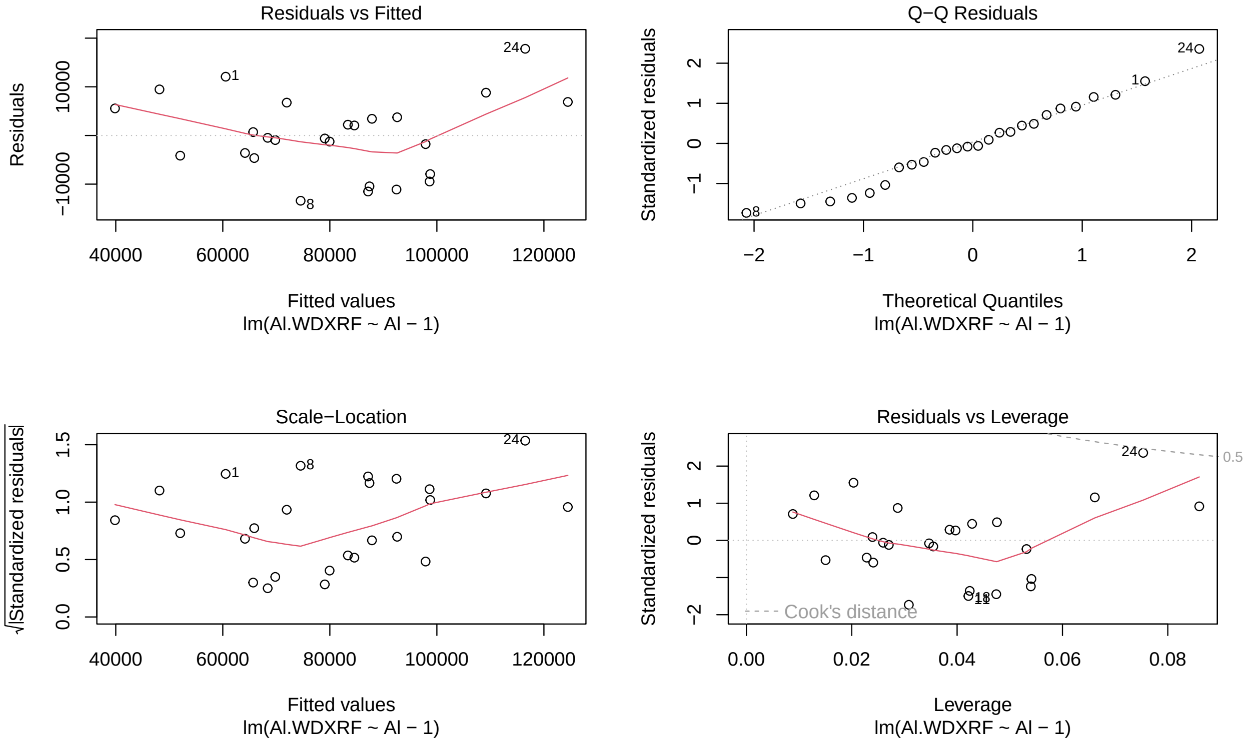
```
SEE<-sigma(OLRAl)
SEE
```

```
[1] 7174.72
```

```
mean<-mean(dataset$Al)
rSEE<-(SEE/mean)*100
rSEE
```

```
[1] 8.225305
```

```
oldparOLRAl <- par(oma=c(0,0,3,0), mfrow=c(2,2))
plot(OLRAl)
```




```
NULL
```

```r
outlierTest(OLRAl)
```

```
No Studentized residuals with Bonferroni p < 0.05
Largest |rstudent|:
   rstudent unadjusted p-value Bonferroni p
8 -1.945074           0.064657           NA
```

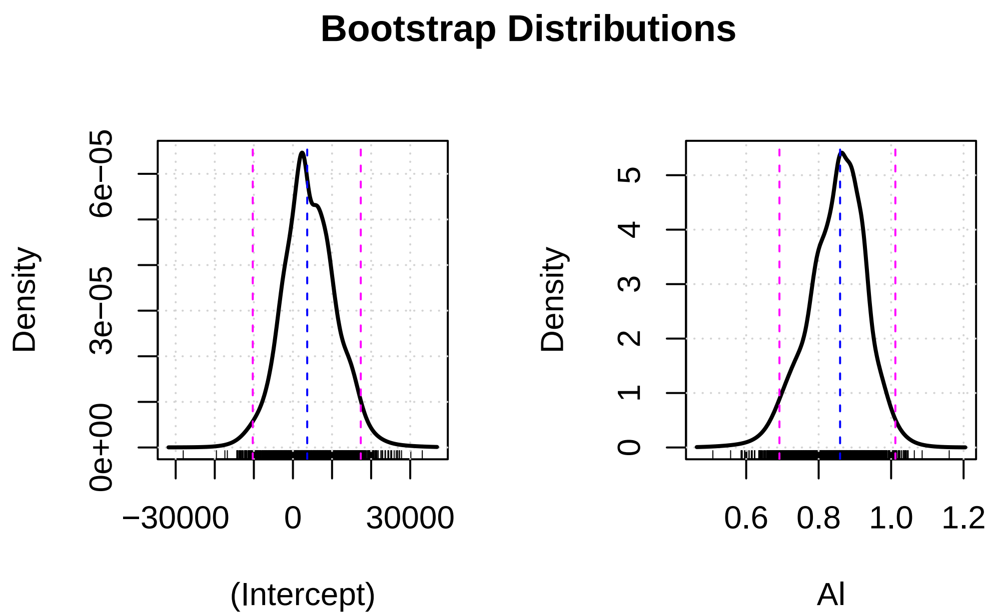
```r
Confint(OLRAl, level=0.90)
```

```
                Estimate            5 %         95 %
(Intercept) 3645.6923078 -6861.8747902 1.415326e+04
Al             0.8591116     0.7419957 9.762274e-01
```

```r
.bs.samplesOLRAl<- Boot(OLRAl, R=2500, method="case")
confint(.bs.samplesOLRAl, level=0.9, type="bca")
```

```
Bootstrap bca confidence intervals

                      5 %         95 %
(Intercept) -7726.5410289 1.537624e+04
Al              0.7166626 9.857368e-01
```

```r
plotBoot(.bs.samplesOLRAl)
```



6.6.3 Regression Trough Origin (RTO)

```
RTOAl<-lm(Al.WDXRF~Al-1, data=dataset)
summary(RTOAl)
```

```
Call:
lm(formula = Al.WDXRF ~ Al - 1, data = dataset)

Residuals:
     Min       1Q   Median       3Q      Max
-12487.7  -3822.4    363.8   4894.9  12830.6

Coefficients:
   Estimate Std. Error t value Pr(>|t|)
Al  0.89862    0.01578   56.96   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 7077 on 24 degrees of freedom
Multiple R-squared:  0.9927,    Adjusted R-squared:  0.9924
F-statistic:  3244 on 1 and 24 DF,  p-value: < 2.2e-16
```

```
fitted.RTOAl <- fitted(RTOAl)
datasetc<-cbind(dataset,fitted.RTOAl)
RMS<-sqrt(mean((datasetc$Al-datasetc$fitted.RTOAl)^2)/nrow(datasetc))
RMS
```

```
[1] 1819.2
```

```
SEE<-sigma(RTOAl)
SEE
```

```
[1] 7077.441
```

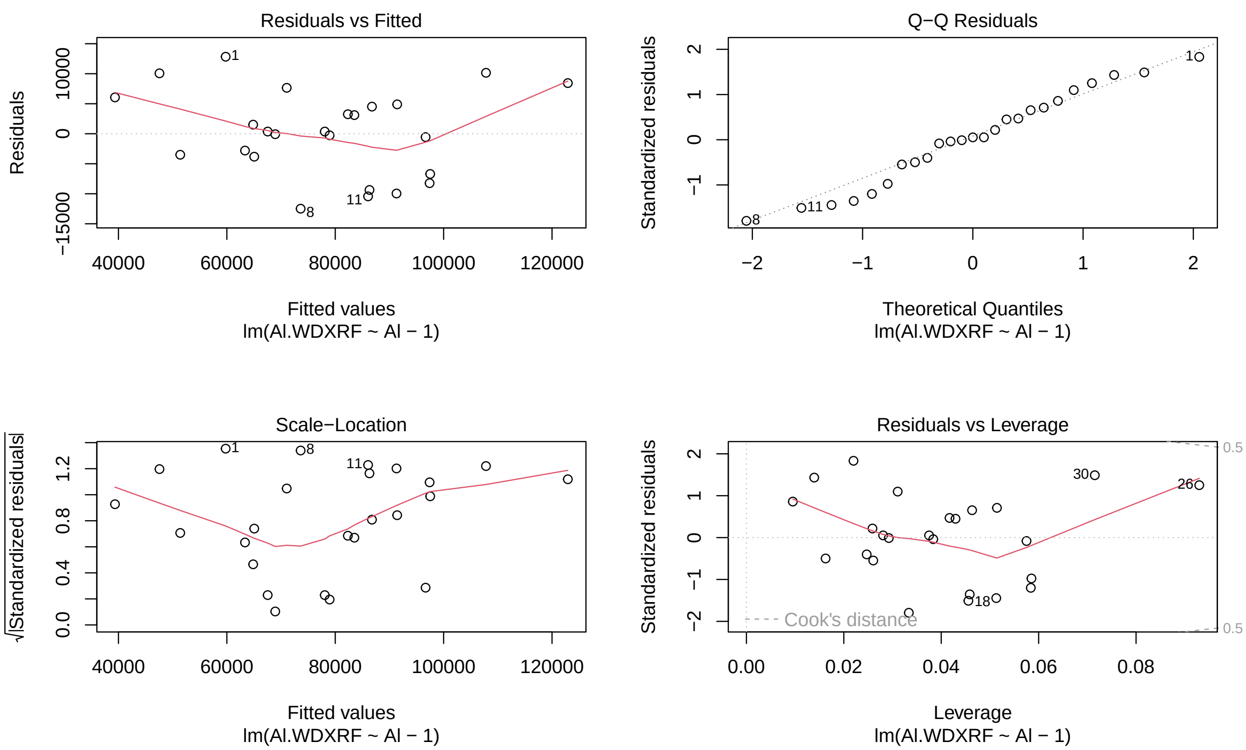
```
mean<-mean(dataset$Al)
rSEE<-(SEE/mean)*100
rSEE
```

```
[1] 8.113782
```

```
SSR<-(nrow(dataset)-1)*(SEE^2)
cSST <-sum(((dataset$Al.WDXRF)-mean(dataset$Al.WDXRF))^2)
Alcr2<-1-(SSR/cSST)
Alcr2
```

```
[1] 0.8710179
```

```
oldparRTOAl <- par(oma=c(0,0,3,0), mfrow=c(2,2))
plot(RTOAl)
```




```
NULL
```

```
outlierTest(RTOAl)
```

```
No Studentized residuals with Bonferroni p < 0.05
Largest |rstudent|:
  rstudent unadjusted p-value Bonferroni p
1 1.935143           0.065363           NA
```

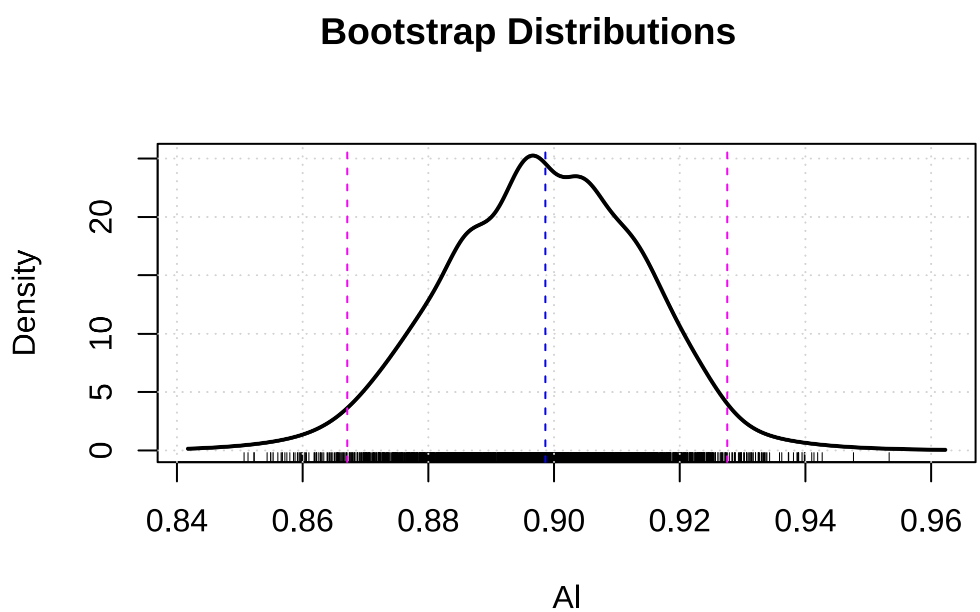
```
Confint(RTOAl, level=0.90)
```

```
    Estimate      5 %      95 %
Al 0.8986173 0.871625 0.9256096
```

```
.bs.samplesRTOAl<- Boot(RTOAl, R=2500, method="case")
confint(.bs.samplesRTOAl, level=0.9, type="bca")
```

```
Bootstrap bca confidence intervals

         5 %      95 %
Al 0.8721035 0.9235699
```

```
plotBoot(.bs.samplesRTOAl)
```



6.6.4 Result

In this run we see different results: r^2 is past the required limits, robustness and rSEE are fine for both linear regressions. Yet, the sample in row 8 qualifies as a (non-significant) outlier for the OLR, sample 1 for RTO. A direct comparison of the characteristic values reveals the following:

criteria	OLR	RTO
r^2	0.873	0.8710
rSEE	8.23	8.11
CI 0.05	0.7420 - 0.7164	0.8716 - 0.8707
CI 0.95	0.9762 - 0.9920	0.9256 - 0.9231

The direct comparison shows very clearly that the RTO has the better values or proportions of values in all criteria. For this reason - and because r^2 is still too low - the sample in row 1 is excluded for the next iteration.

Different values by the bootstraph?

If you do the calculations yourself, the values of CIBS5 and CIBS95 will most likely differ slightly from those in the table. Don't worry - this is completely normal! As long as the differences between your values and those in the table are not too great (i.e. differences of more than 0.05) everything is fine.

6.7 Seventh iteration

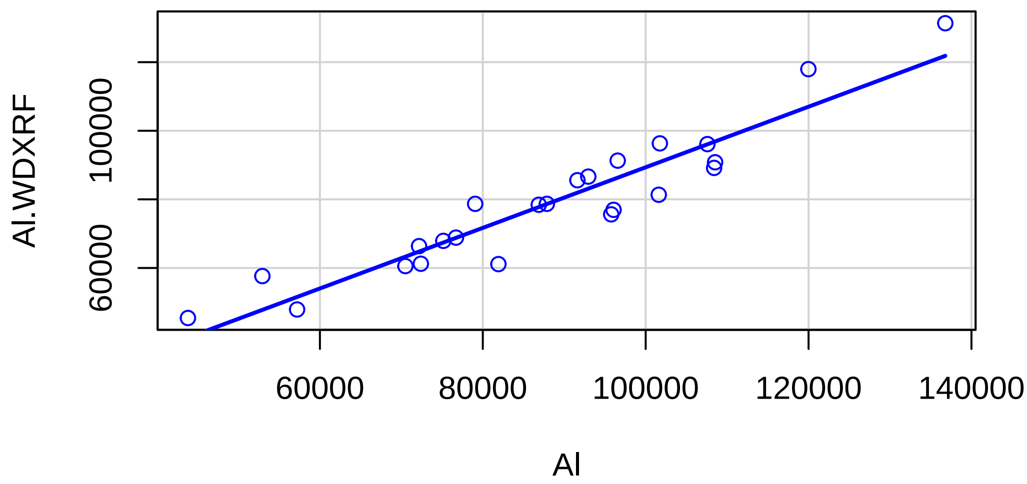
And out with row 1.

```
dataset1<- read.csv("../data_analytical//coefcorI_data.csv")
dataset<- dataset1[-c(25,14,27,28,24,1), ]
```

calculations

6.7.1 First impressions

```
scatterplot(Al.WDXRF~Al, regLine=TRUE, smooth=FALSE, boxplots=FALSE, data=dataset)
```




```
cor(dataset$Al,dataset$Al.WDXRF)
```

```
[1] 0.9427524
```

```
numSummary(dataset[,c("Al", "Al.WDXRF"), drop=FALSE],
           statistics=c("mean", "sd", "quantiles"),quantiles=c(0,.25,.5,.75,1))
```

```
             mean       sd    0%     25%     50%       75%   100%  n
Al       88089.96 21445.07 43795 74465.0 89743.0 101638.50 136788 24
Al.WDXRF 78832.58 20090.28 45409 65083.5 78565.5  89587.25 131358 24
```

6.7.2 Ordinary Linear Regression (OLR)

```
OLRAl<-lm(Al.WDXRF~Al, data=dataset)
summary(OLRAl)
```

```
Call:
lm(formula = Al.WDXRF ~ Al, data = dataset)

Residuals:
   Min     1Q Median     3Q    Max
-12256  -4319    285   5090  10971

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 1.032e+03  6.032e+03   0.171    0.866
Al          8.832e-01  6.661e-02  13.259 5.72e-12 ***
---
```

```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 6851 on 22 degrees of freedom
Multiple R-squared:  0.8888,    Adjusted R-squared:  0.8837
F-statistic: 175.8 on 1 and 22 DF,  p-value: 5.716e-12
```

```r
fitted.OLRAl <- fitted(OLRAl)
datasetb<-cbind(dataset,fitted.OLRAl)
RMS<-sqrt(mean((datasetb$Al-datasetb$fitted.OLRAl)^2)/nrow(datasetb))
RMS
```

```
[1] 1954.824
```

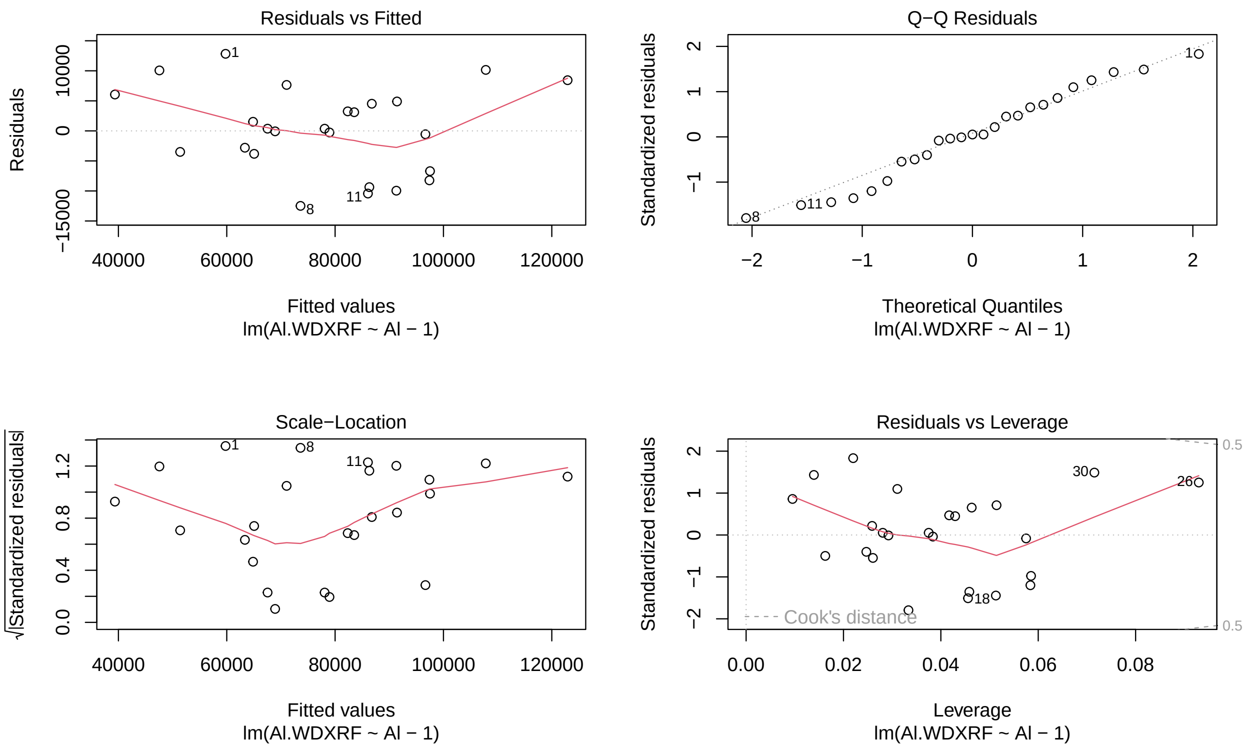
```r
SEE<-sigma(OLRAl)
SEE
```

```
[1] 6850.562
```

```r
mean<-mean(dataset$Al)
rSEE<-(SEE/mean)*100
rSEE
```

```
[1] 7.77678
```

```r
oldparOLRAl <- par(oma=c(0,0,3,0), mfrow=c(2,2))
plot(OLRAl)
```




```
NULL
```

```r
outlierTest(OLRAl)
```

```
No Studentized residuals with Bonferroni p < 0.05
Largest |rstudent|:
   rstudent unadjusted p-value Bonferroni p
8 -1.943091           0.065531           NA
```

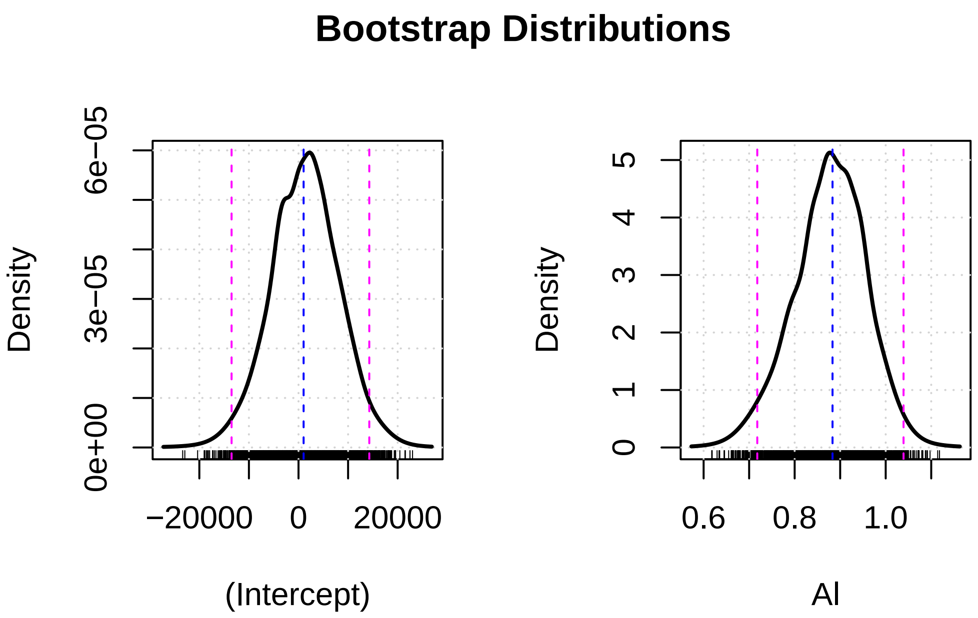
```r
Confint(OLRAl, level=0.90)
```

```
                Estimate          5 %         95 %
(Intercept) 1032.0240723 -9325.6851601 1.138973e+04
Al             0.8831944     0.7688166 9.975722e-01
```

```r
.bs.samplesOLRAl<- Boot(OLRAl, R=2500, method="case")
confint(.bs.samplesOLRAl, level=0.9, type="bca")
```

```
Bootstrap bca confidence intervals

                      5 %         95 %
(Intercept) -1.126613e+04 11856.108768
Al           7.472384e-01     1.017284
```

```r
plotBoot(.bs.samplesOLRAl)
```



6.7.3 Regression Trough Origin (RTO)

```
RTOAl<-lm(Al.WDXRF~Al-1, data=dataset)
summary(RTOAl)
```

```
Call:
lm(formula = Al.WDXRF ~ Al - 1, data = dataset)

Residuals:
     Min       1Q   Median       3Q      Max
-12132.4  -4189.3    475.3   5036.7  10673.1

Coefficients:
   Estimate Std. Error t value Pr(>|t|)
Al  0.89428    0.01511   59.17   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 6704 on 23 degrees of freedom
Multiple R-squared:  0.9935,    Adjusted R-squared:  0.9932
F-statistic:  3502 on 1 and 23 DF,  p-value: < 2.2e-16
```

```
fitted.RTOAl <- fitted(RTOAl)
datasetc<-cbind(dataset,fitted.RTOAl)
RMS<-sqrt(mean((datasetc$Al-datasetc$fitted.RTOAl)^2)/nrow(datasetc))
RMS
```

```
[1] 1954.214
```

```
SEE<-sigma(RTOAl)
SEE
```

```
[1] 6704.438
```

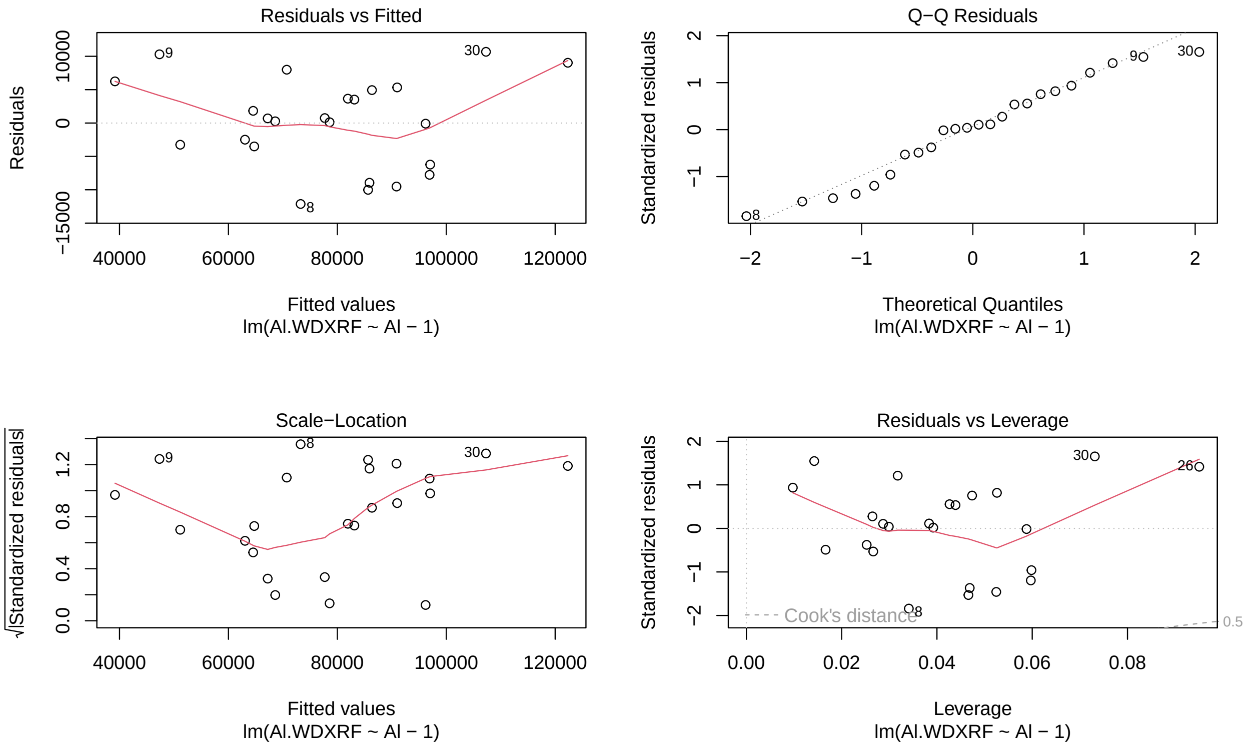
```
mean<-mean(dataset$Al)
rSEE<-(SEE/mean)*100
rSEE
```

```
[1] 7.610899
```

```
SSR<-(nrow(dataset)-1)*(SEE^2)
cSST <-sum(((dataset$Al.WDXRF)-mean(dataset$Al.WDXRF))^2)
Alcr2<-1-(SSR/cSST)
Alcr2
```

```
[1] 0.888634
```

```
oldparRTOAl <- par(oma=c(0,0,3,0), mfrow=c(2,2))
plot(RTOAl)
```




```
NULL
```

```
outlierTest(RTOAl)
```

```
No Studentized residuals with Bonferroni p < 0.05
Largest |rstudent|:
   rstudent unadjusted p-value Bonferroni p
8 -1.950273           0.063999           NA
```

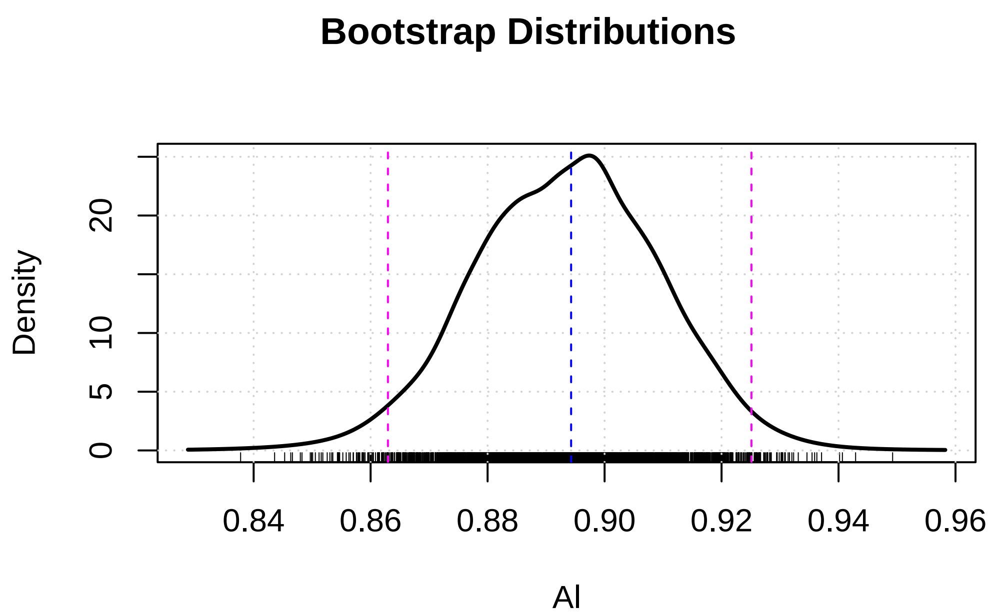
```
Confint(RTOAl, level=0.90)
```

```
    Estimate       5 %      95 %
Al 0.8942803 0.8683796 0.9201811
```

```
.bs.samplesRTOAl<- Boot(RTOAl, R=2500, method="case")
confint(.bs.samplesRTOAl, level=0.9, type="bca")
```

```
Bootstrap bca confidence intervals

         5 %      95 %
Al 0.8677046 0.9202462
```

```
plotBoot(.bs.samplesRTOAl)
```



6.7.4 Result

Both linear regressions give the sample in row 8 as a (non-significant) outlier. Again, r^2 is too low, but rSEE and robustness are fine for OLR and RTO.

6.8 Eighth iteration

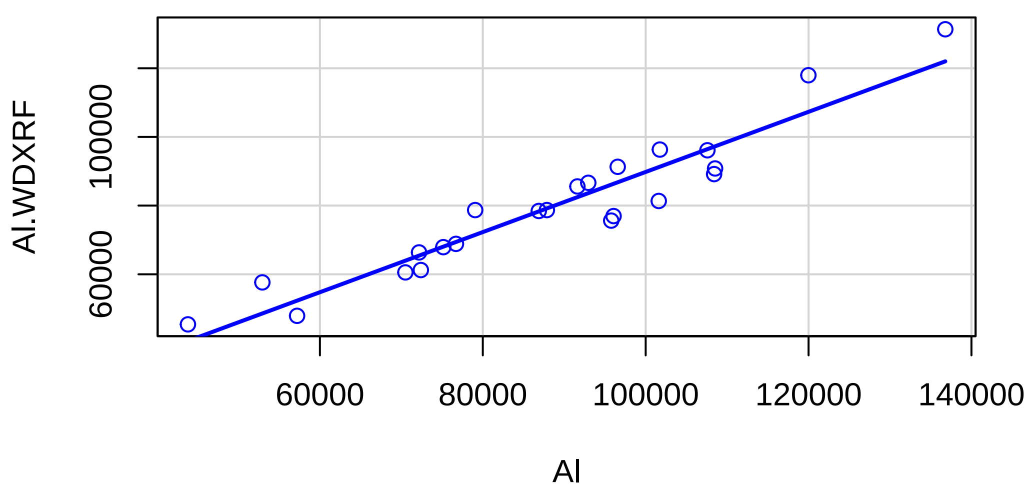
Lets exclude row 8, too.

```
dataset1<- read.csv("../data_analytical//coefcorI_data.csv")
dataset<- dataset1[-c(25,14,27,28,24,1,8), ]
```

calculations

6.8.1 First impressions

```
scatterplot(Al.WDXRF~Al, regLine=TRUE, smooth=FALSE, boxplots=FALSE, data=dataset)
```




```
cor(dataset$Al,dataset$Al.WDXRF)
```

```
[1] 0.9498882
```

```
numSummary(dataset[,c("Al", "Al.WDXRF"), drop=FALSE],
           statistics=c("mean", "sd", "quantiles"),quantiles=c(0,.25,.5,.75,1))
```

```
             mean       sd    0%     25%   50%      75%   100%  n
Al       88358.22 21885.83 43795 73775.0 91616 101673.0 136788 23
Al.WDXRF 79602.39 20176.64 45409 67134.5 78698  89997.5 131358 23
```

6.8.2 Ordinary Linear Regression (OLR)

```
OLRAl<-lm(Al.WDXRF~Al, data=dataset)
summary(OLRAl)
```

```
Call:
lm(formula = Al.WDXRF ~ Al, data = dataset)

Residuals:
     Min       1Q   Median       3Q      Max 
-10464.9  -4404.3   -138.3   4667.4  10675.1 

Coefficients:
             Estimate Std. Error t value Pr(>|t|)    
(Intercept) 2.227e+03  5.717e+03   0.389    0.701    
Al          8.757e-01  6.289e-02  13.925 4.47e-12 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 6455 on 21 degrees of freedom
Multiple R-squared:  0.9023,    Adjusted R-squared:  0.8976 
F-statistic: 193.9 on 1 and 21 DF,  p-value: 4.471e-12
```

```
fitted.OLRAl <- fitted(OLRAl)
datasetb<-cbind(dataset,fitted.OLRAl)
RMS<-sqrt(mean((datasetb$Al-datasetb$fitted.OLRAl)^2)/nrow(datasetb))
RMS
```

```
[1] 1908.136
```

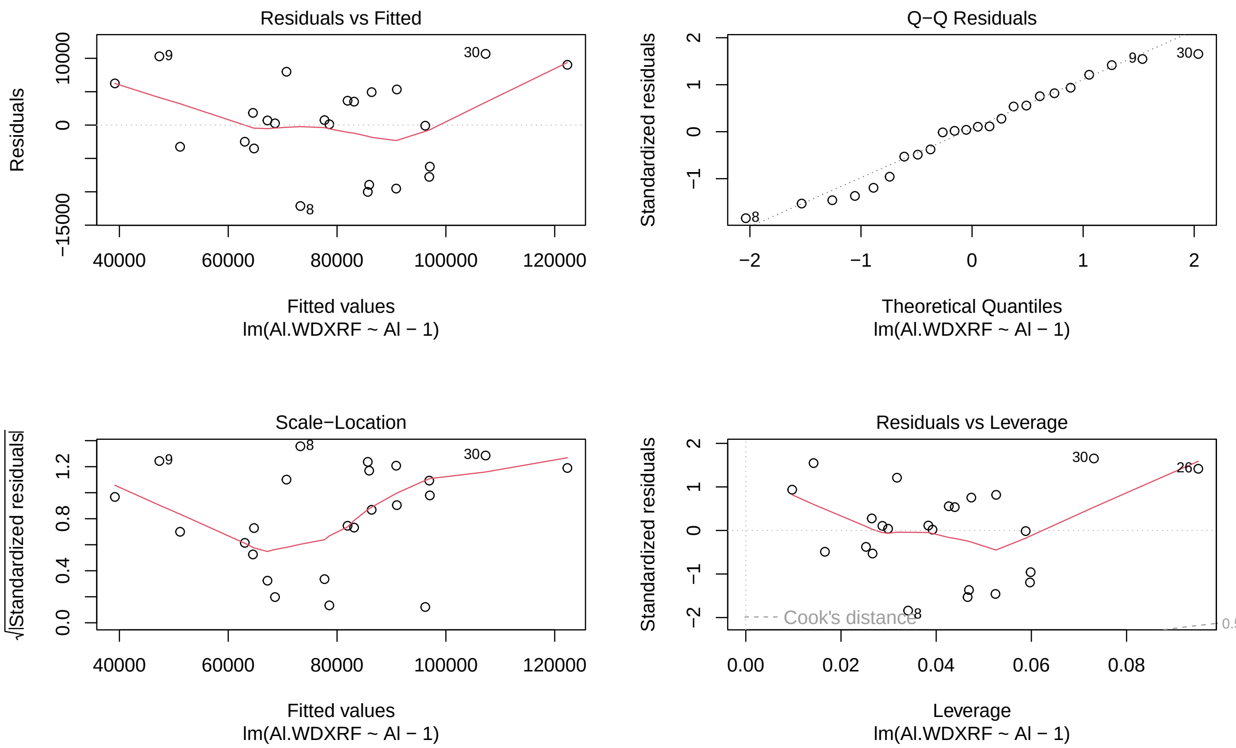
```
SEE<-sigma(OLRAl)
SEE
```

```
[1] 6455.434
```

```
mean<-mean(dataset$Al)
rSEE<-(SEE/mean)*100
rSEE
```

```
[1] 7.30598
```

```
oldparOLRAl <- par(oma=c(0,0,3,0), mfrow=c(2,2))
plot(OLRAl)
```




```
NULL
```

```r
outlierTest(OLRAl)
```

```
No Studentized residuals with Bonferroni p < 0.05
Largest |rstudent|:
   rstudent unadjusted p-value Bonferroni p
30 1.886986           0.073764           NA
```

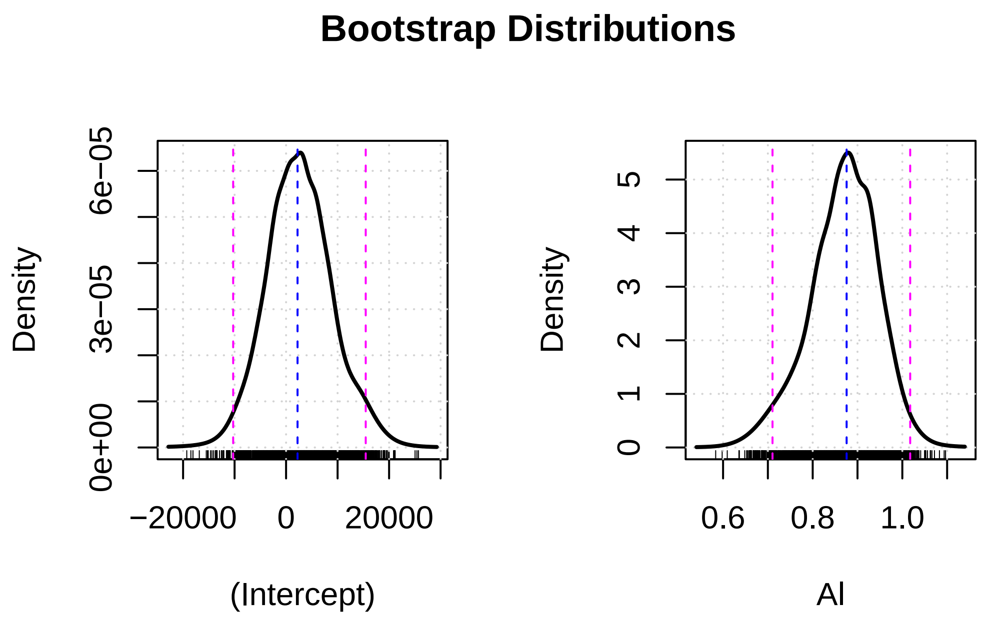
```r
Confint(OLRAl, level=0.90)
```

```
                Estimate          5 %         95 %
(Intercept) 2226.5502167 -7611.2380481 1.206434e+04
Al             0.8757062     0.7674963 9.839162e-01
```

```r
.bs.samplesOLRAl<- Boot(OLRAl, R=2500, method="case")
confint(.bs.samplesOLRAl, level=0.9, type="bca")
```

```
Bootstrap bca confidence intervals

                      5 %         95 %
(Intercept) -8691.0294715 1.338129e+04
Al              0.7391363 9.949074e-01
```

```r
plotBoot(.bs.samplesOLRAl)
```



6.8.3 Regression Trough Origin (RTO)

```
RTOAl<-lm(Al.WDXRF~Al-1, data=dataset)
summary(RTOAl)
```

```
Call:
lm(formula = Al.WDXRF ~ Al - 1, data = dataset)

Residuals:
     Min       1Q   Median       3Q      Max
-10517.9  -3720.1    286.4   4618.1  10045.9

Coefficients:
   Estimate Std. Error t value Pr(>|t|)
Al  0.89951    0.01452   61.96   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 6330 on 22 degrees of freedom
Multiple R-squared:  0.9943,    Adjusted R-squared:  0.994
F-statistic:  3839 on 1 and 22 DF,  p-value: < 2.2e-16
```

```
fitted.RTOAl <- fitted(RTOAl)
datasetc<-cbind(dataset,fitted.RTOAl)
RMS<-sqrt(mean((datasetc$Al-datasetc$fitted.RTOAl)^2)/nrow(datasetc))
RMS
```

```
[1] 1905.003
```

```r
SEE<-sigma(RTOAl)
SEE
```

```
[1] 6329.748
```

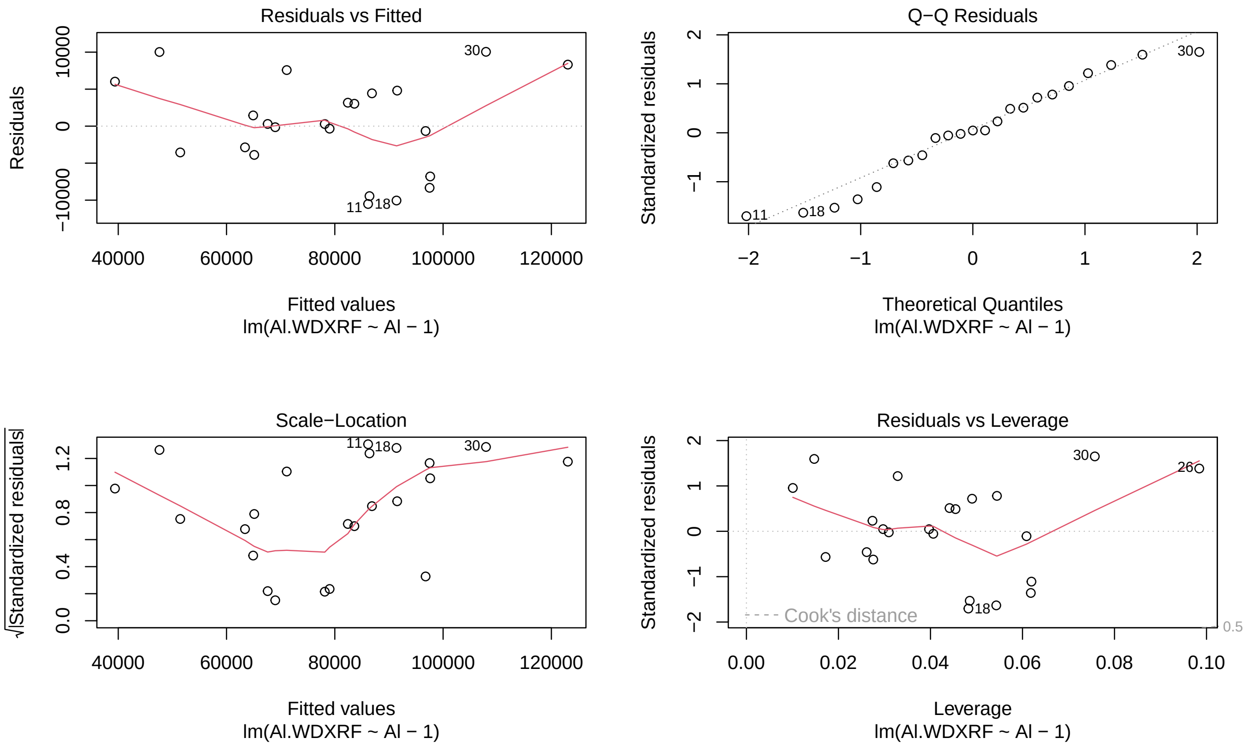
```r
mean<-mean(dataset$Al)
rSEE<-(SEE/mean)*100
rSEE
```

```
[1] 7.163735
```

```r
SSR<-(nrow(dataset)-1)*(SEE^2)
cSST <-sum(((dataset$Al.WDXRF)-mean(dataset$Al.WDXRF))^2)
Alcr2<-1-(SSR/cSST)
Alcr2
```

```
[1] 0.9015819
```

```r
oldparRTOAl <- par(oma=c(0,0,3,0), mfrow=c(2,2))
plot(RTOAl)
```




```
NULL
```

```r
outlierTest(RTOAl)
```

```
No Studentized residuals with Bonferroni p < 0.05
Largest |rstudent|:
    rstudent unadjusted p-value Bonferroni p
11 -1.786024           0.088544           NA
```

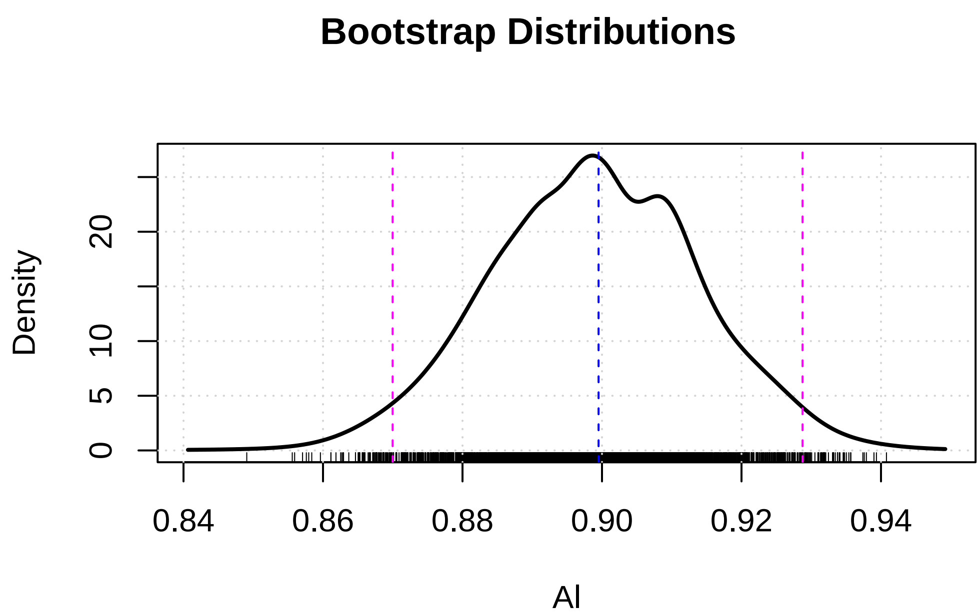
```
Confint(RTOAl, level=0.90)
```

```
    Estimate        5 %      95 %
Al 0.8995085 0.8745799 0.9244372
```

```
.bs.samplesRTOAl<- Boot(RTOAl, R=2500, method="case")
confint(.bs.samplesRTOAl, level=0.9, type="bca")
```

```
Bootstrap bca confidence intervals

         5 %      95 %
Al 0.8750248 0.925148
```

```
plotBoot(.bs.samplesRTOAl)
```



6.8.4 Result

Now we are getting somewhere: r^2, robustness and rSEE are fine for both linear regressions. The sample in row 30 qualifies as a (non-significant) outlier for the OLR, sample 11 for RTO. A direct comparison of the characteristic values reveals the following:

criteria	OLR	RTO
r^2	0.9023	0.9016
rSEE	7.31	7.16
CI 0.05	0.7675 - 0.7450	0.8746 - 0.8723
CI 0.95	0.9839 - 0.9982	0.9244 - 0.9238

The direct comparison shows again that the RTO has the better values or proportions of values in all criteria except for r^2 which is very slightly better for the OLR. Still, the RTO is deemed better. As r^2 is very close to the benchmark the sample in row 11 is excluded for the next iteration to see if the values are getting better.

6.9 Ninth iteration

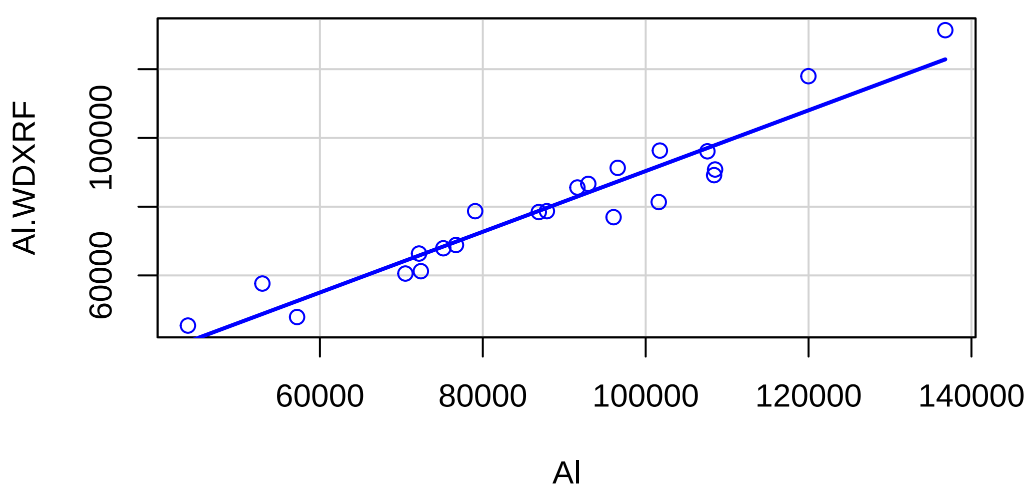
Proceeding with excluding row 11.

```
dataset1<- read.csv("../data_analytical//coefcorI_data.csv")
dataset<- dataset1[-c(25,14,27,28,24,1,8,11), ]
```

calculations

6.9.1 First impressions

```
scatterplot(Al.WDXRF~Al, regLine=TRUE, smooth=FALSE, boxplots=FALSE, data=dataset)
```




```
cor(dataset$Al,dataset$Al.WDXRF)
```

```
[1] 0.9565483
```

```
numSummary(dataset[,c("Al", "Al.WDXRF"), drop=FALSE],
           statistics=c("mean", "sd", "quantiles"),quantiles=c(0,.25,.5,.75,1))
```

```
             mean       sd    0%      25%   50%       75%   100%  n
Al       88021.32 22339.73 43795 73085.00 89743 101707.50 136788 22
Al.WDXRF 79783.05 20632.40 45409 66750.75 78698  90407.75 131358 22
```

6.9.2 Ordinary Linear Regression (OLR)

```
OLRAl<-lm(Al.WDXRF~Al, data=dataset)
summary(OLRAl)
```

```
Call:
lm(formula = Al.WDXRF ~ Al, data = dataset)

Residuals:
     Min       1Q   Median       3Q      Max 
-10438.6  -4430.5   -426.7   4303.7   9952.1 

Coefficients:
             Estimate Std. Error t value Pr(>|t|)    
(Intercept) 2.021e+03  5.461e+03    0.37    0.715    
Al          8.834e-01  6.022e-02   14.67 3.62e-12 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 6164 on 20 degrees of freedom
Multiple R-squared:  0.915,	Adjusted R-squared:  0.9107 
F-statistic: 215.3 on 1 and 20 DF,  p-value: 3.618e-12
```

```
fitted.OLRAl <- fitted(OLRAl)
datasetb<-cbind(dataset,fitted.OLRAl)
RMS<-sqrt(mean((datasetb$Al-datasetb$fitted.OLRAl)^2)/nrow(datasetb))
RMS
```

```
[1] 1838.242
```

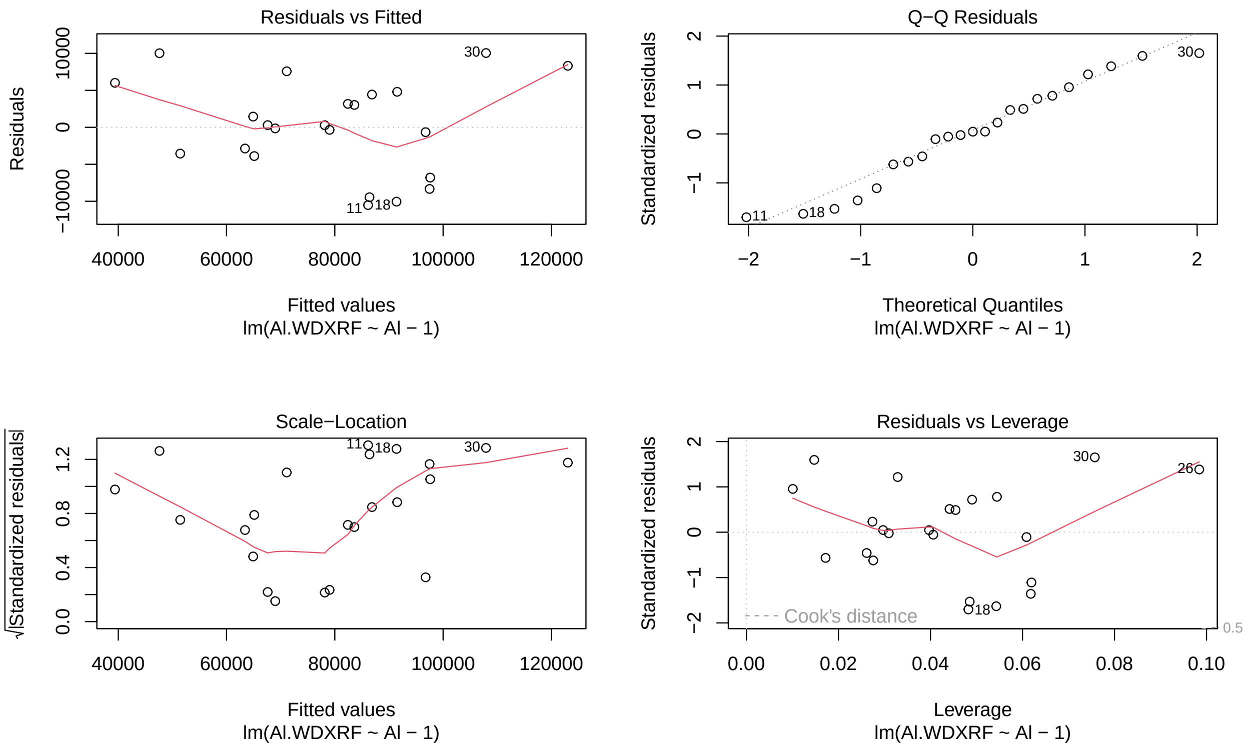
```
SEE<-sigma(OLRAl)
SEE
```

```
[1] 6164.434
```

```
mean<-mean(dataset$Al)
rSEE<-(SEE/mean)*100
rSEE
```

```
[1] 7.003342
```

```
oldparOLRAl <- par(oma=c(0,0,3,0), mfrow=c(2,2))
plot(OLRAl)
```




```
NULL
```

```
outlierTest(OLRAl)
```

```
No Studentized residuals with Bonferroni p < 0.05
Largest |rstudent|:
    rstudent unadjusted p-value Bonferroni p
18 -1.852753           0.079517           NA
```

```
Confint(OLRAl, level=0.90)
```

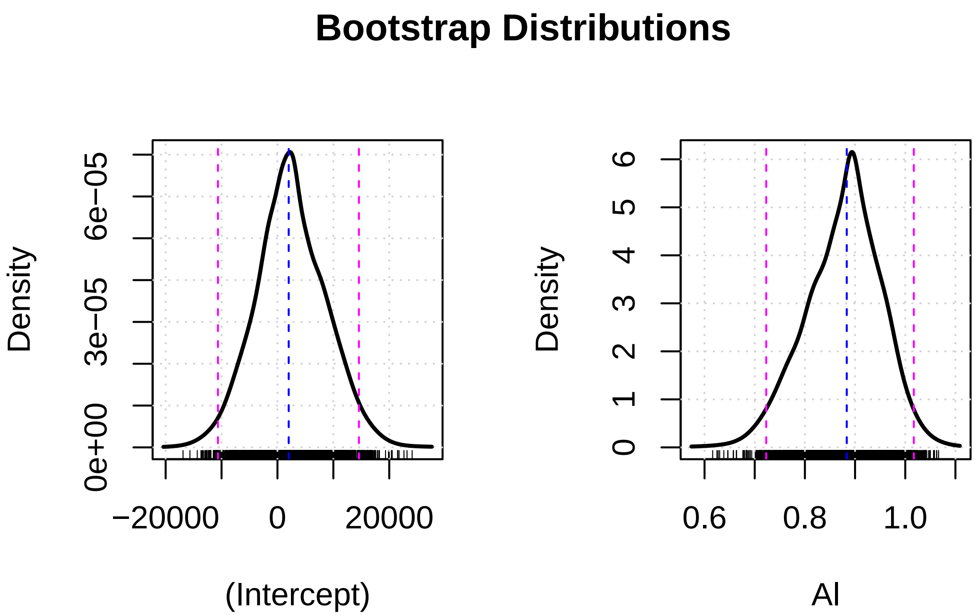
```
                Estimate          5 %         95 %
(Intercept) 2021.1820924 -7397.0375539 1.143940e+04
Al             0.8834435     0.7795894 9.872977e-01
```

```
.bs.samplesOLRAl<- Boot(OLRAl, R=2500, method="case")
confint(.bs.samplesOLRAl, level=0.9, type="bca")
```

```
Bootstrap bca confidence intervals
```

```
                        5 %         95 %
(Intercept) -8522.0471526 1.241959e+04
Al              0.7466253 9.959083e-01
```

```
plotBoot(.bs.samplesOLRAl)
```



6.9.3 Regression Trough Origin (RTO)

```
RTOAl<-lm(Al.WDXRF~Al-1, data=dataset)
summary(RTOAl)
```

```
Call:
lm(formula = Al.WDXRF ~ Al - 1, data = dataset)

Residuals:
     Min       1Q   Median       3Q      Max
-10615.3  -3718.5   -158.1   4151.9   9725.6

Coefficients:
   Estimate Std. Error t value Pr(>|t|)
Al  0.90508    0.01419   63.78   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 6036 on 21 degrees of freedom
Multiple R-squared:  0.9949,    Adjusted R-squared:  0.9946
F-statistic:  4067 on 1 and 21 DF,  p-value: < 2.2e-16
```

```r
fitted.RTOAl <- fitted(RTOAl)
datasetc<-cbind(dataset,fitted.RTOAl)
RMS<-sqrt(mean((datasetc$Al-datasetc$fitted.RTOAl)^2)/nrow(datasetc))
RMS
```

```
[1] 1835.314
```

```r
SEE<-sigma(RTOAl)
SEE
```

```
[1] 6036.44
```

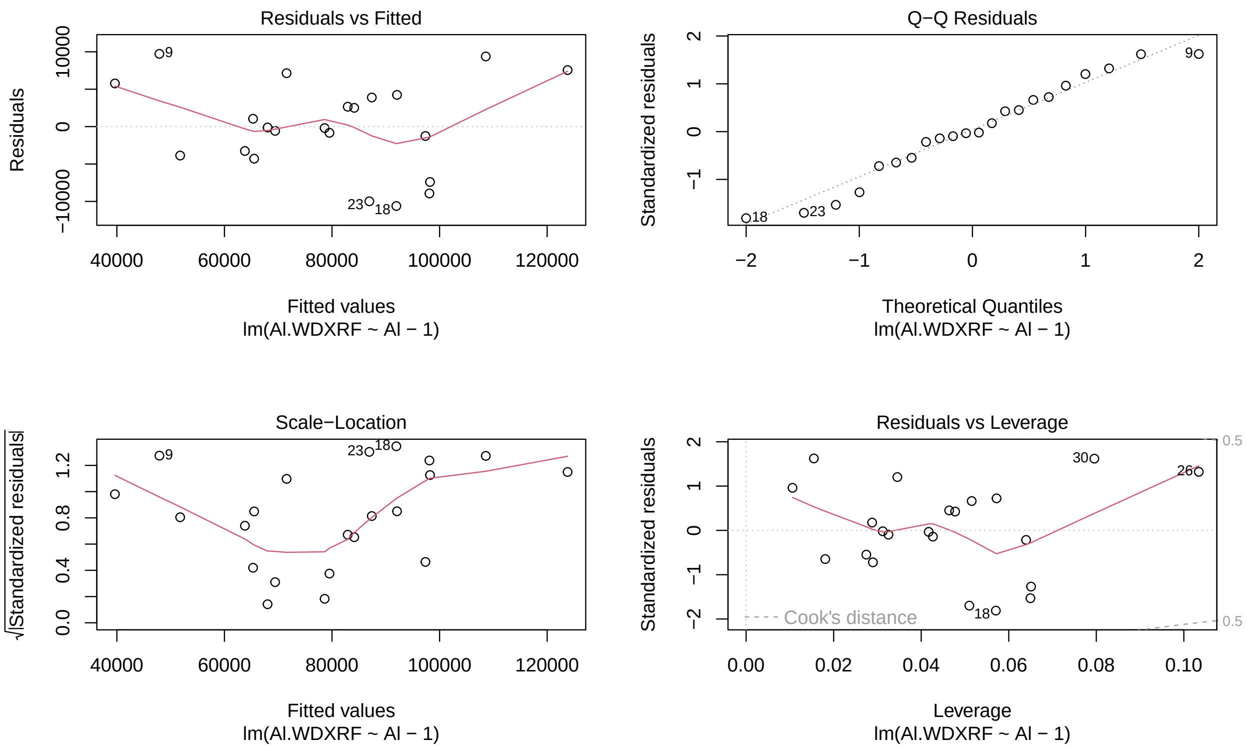
```r
mean<-mean(dataset$Al)
rSEE<-(SEE/mean)*100
rSEE
```

```
[1] 6.85793
```

```r
SSR<-(nrow(dataset)-1)*(SEE^2)
cSST <-sum(((dataset$Al.WDXRF)-mean(dataset$Al.WDXRF))^2)
Alcr2<-1-(SSR/cSST)
Alcr2
```

```
[1] 0.9144023
```

```r
oldparRTOAl <- par(oma=c(0,0,3,0), mfrow=c(2,2))
plot(RTOAl)
```




```
NULL
```

```r
outlierTest(RTOAl)
```

```
No Studentized residuals with Bonferroni p < 0.05
Largest |rstudent|:
    rstudent unadjusted p-value Bonferroni p
18 -1.923924           0.068713           NA
```

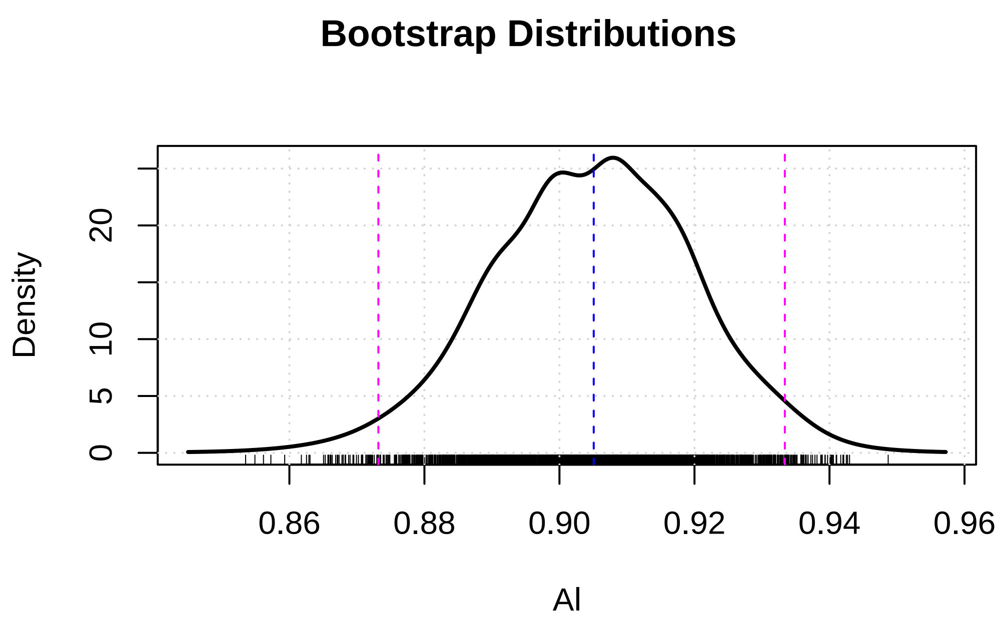
```r
Confint(RTOAl, level=0.90)
```

```
    Estimate        5 %      95 %
Al 0.9050759 0.8806561 0.9294956
```

```r
.bs.samplesRTOAl<- Boot(RTOAl, R=2500, method="case")
confint(.bs.samplesRTOAl, level=0.9, type="bca")
```

```
Bootstrap bca confidence intervals

          5 %      95 %
Al 0.8783731 0.9296875
```

```r
plotBoot(.bs.samplesRTOAl)
```



6.9.4 Result

In direct comparison, the values of the criteria of the RTO remain better. Also, they increased by excluding sample 11. As both linear regressions give the sample in row 18 as outlier, we are going to exclude this one, too.

6.10 Tenth iteration

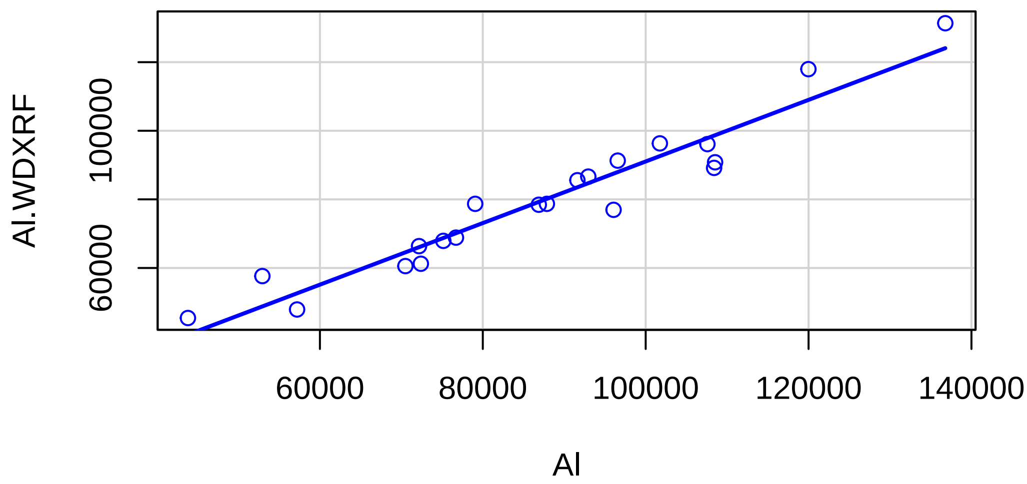
Now excluding row 18.

```
dataset1<- read.csv("../data_analytical//coefcorI_data.csv")
dataset<- dataset1[-c(25,14,27,28,24,1,8,11,18), ]
```

calculations

6.10.1 First impressions

```
scatterplot(Al.WDXRF~Al, regLine=TRUE, smooth=FALSE, boxplots=FALSE, data=dataset)
```




```
cor(dataset$Al,dataset$Al.WDXRF)
```

```
[1] 0.9633137
```

```
numSummary(dataset[,c("Al", "Al.WDXRF"), drop=FALSE],
           statistics=c("mean", "sd", "quantiles"),quantiles=c(0,.25,.5,.75,1))
```

```
             mean       sd    0%   25%   50%    75%   100%  n
Al       87374.52 22679.35 43795 72395 87870 101742 136788 21
Al.WDXRF 79708.71 21138.90 45409 66367 78698  90818 131358 21
```

6.10.2 Ordinary Linear Regression (OLR)

```
OLRAl<-lm(Al.WDXRF~Al, data=dataset)
summary(OLRAl)
```

```
Call:
lm(formula = Al.WDXRF ~ Al, data = dataset)

Residuals:
   Min     1Q Median     3Q    Max
-10554  -4009   -829   3713   8984

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 1.257e+03  5.173e+03   0.243    0.811
Al          8.979e-01  5.739e-02  15.646 2.62e-12 ***
---
```

```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 5821 on 19 degrees of freedom
Multiple R-squared:  0.928, Adjusted R-squared:  0.9242
F-statistic: 244.8 on 1 and 19 DF,  p-value: 2.616e-12
```

```r
fitted.OLRA1 <- fitted(OLRA1)
datasetb<-cbind(dataset,fitted.OLRA1)
RMS<-sqrt(mean((datasetb$A1-datasetb$fitted.OLRA1)^2)/nrow(datasetb))
RMS
```

```
[1] 1744.009
```

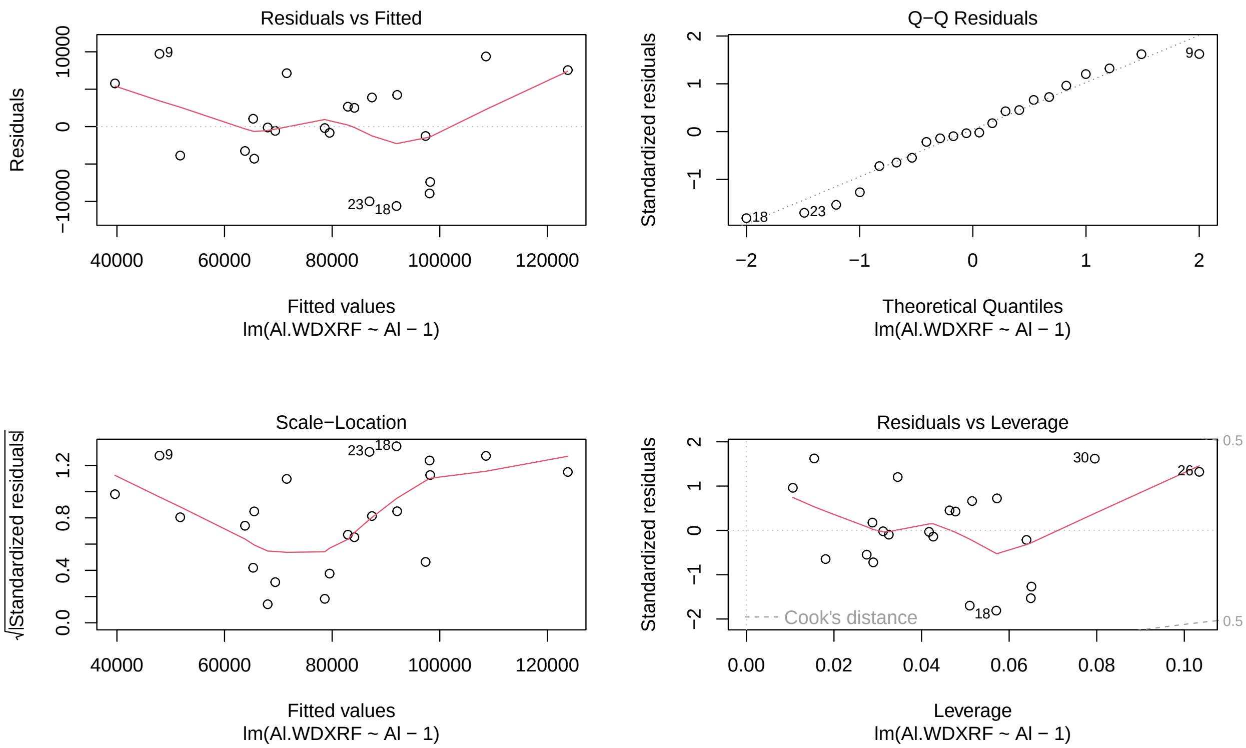
```r
SEE<-sigma(OLRA1)
SEE
```

```
[1] 5820.597
```

```r
mean<-mean(dataset$A1)
rSEE<-(SEE/mean)*100
rSEE
```

```
[1] 6.661664
```

```r
oldparOLRA1 <- par(oma=c(0,0,3,0), mfrow=c(2,2))
plot(OLRA1)
```




```
NULL
```

```
outlierTest(OLRAl)
```

```
No Studentized residuals with Bonferroni p < 0.05
Largest |rstudent|:
   rstudent unadjusted p-value Bonferroni p
23 -2.00849           0.059836           NA
```

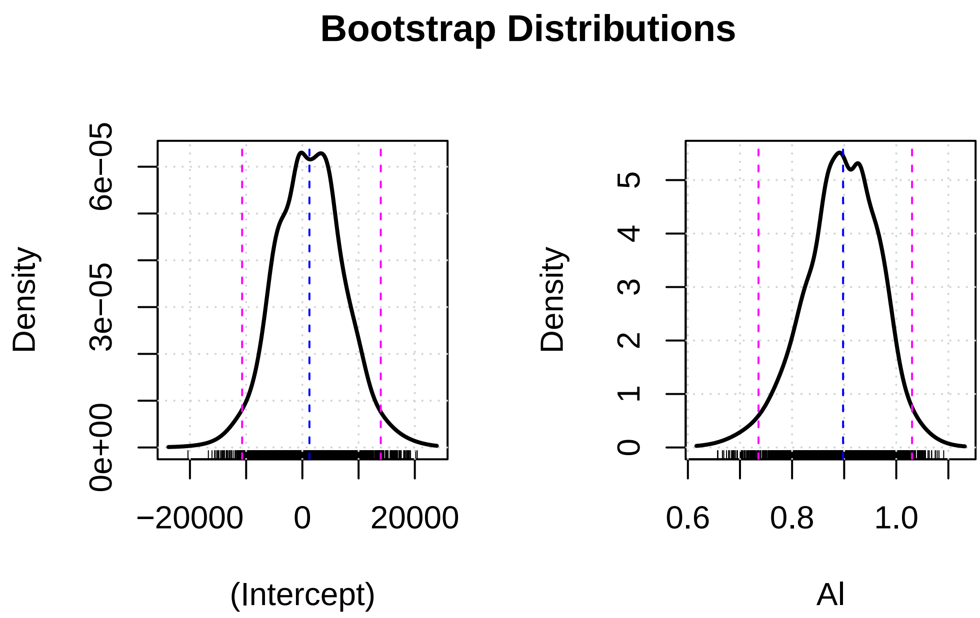
```
Confint(OLRAl, level=0.90)
```

```
                Estimate           5 %         95 %
(Intercept) 1256.6511447 -7687.5108511 1.020081e+04
Al             0.8978826     0.7986509 9.971142e-01
```

```
.bs.samplesOLRAl<- Boot(OLRAl, R=2500, method="case")
confint(.bs.samplesOLRAl, level=0.9, type="bca")
```

```
Bootstrap bca confidence intervals

                     5 %         95 %
(Intercept) -8682.3453843 11549.284151
Al              0.7672977     1.007969
```

```
plotBoot(.bs.samplesOLRAl)
```



6.10.3 Regression Trough Origin (RTO)

```
RTOAl<-lm(Al.WDXRF~Al-1, data=dataset)
summary(RTOAl)
```

```
Call:
lm(formula = Al.WDXRF ~ Al - 1, data = dataset)

Residuals:
     Min       1Q   Median       3Q      Max
-10595.0  -3704.9   -594.1   3594.6   9391.0

Coefficients:
   Estimate Std. Error t value Pr(>|t|)
Al  0.91140    0.01376   66.25   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 5682 on 20 degrees of freedom
Multiple R-squared:  0.9955,    Adjusted R-squared:  0.9952
F-statistic:  4389 on 1 and 20 DF,  p-value: < 2.2e-16
```

```
fitted.RTOAl <- fitted(RTOAl)
datasetc<-cbind(dataset,fitted.RTOAl)
RMS<-sqrt(mean((datasetc$Al-datasetc$fitted.RTOAl)^2)/nrow(datasetc))
RMS
```

```
[1] 1742.708
```

```
SEE<-sigma(RTOAl)
SEE
```

```
[1] 5682.021
```

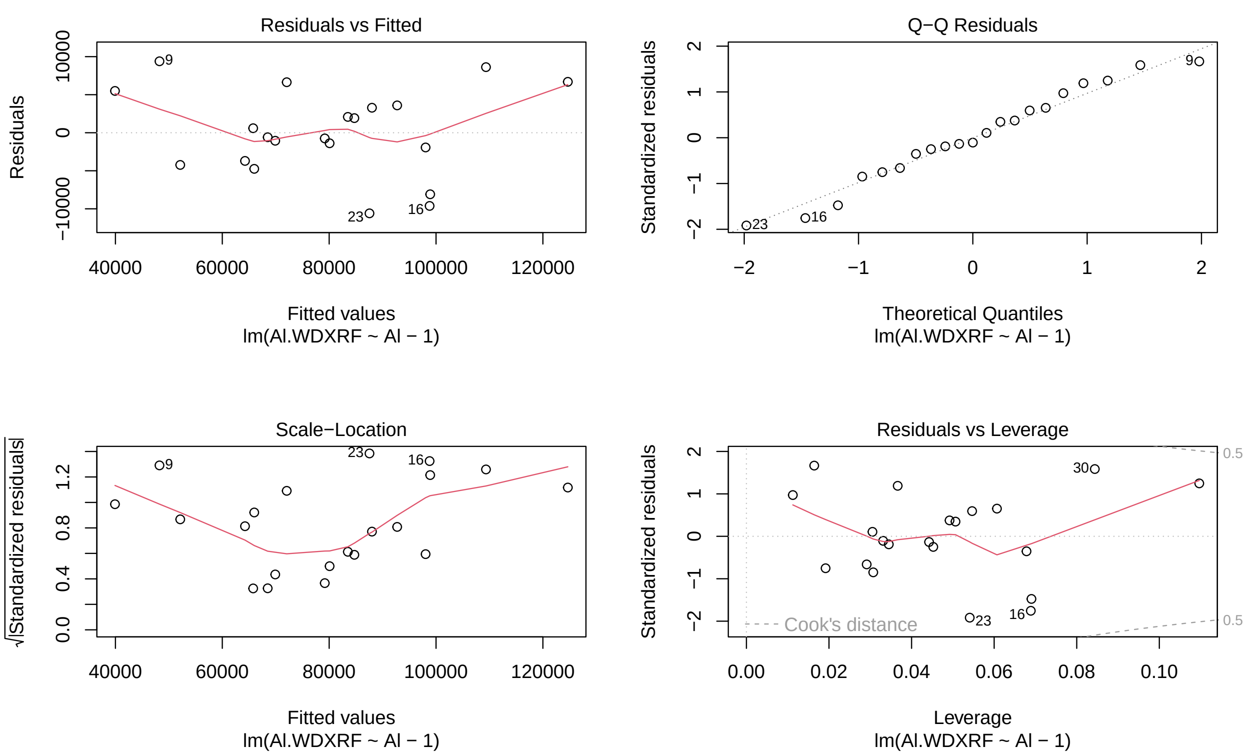
```
mean<-mean(dataset$Al)
rSEE<-(SEE/mean)*100
rSEE
```

```
[1] 6.503063
```

```
SSR<-(nrow(dataset)-1)*(SEE^2)
cSST <-sum(((dataset$Al.WDXRF)-mean(dataset$Al.WDXRF))^2)
Alcr2<-1-(SSR/cSST)
Alcr2
```

```
[1] 0.9277495
```

```
oldparRTOAl <- par(oma=c(0,0,3,0), mfrow=c(2,2))
plot(RTOAl)
```




```
NULL
```

```
outlierTest(RTOAl)
```

```
No Studentized residuals with Bonferroni p < 0.05
Largest |rstudent|:
    rstudent unadjusted p-value Bonferroni p
23 -2.068397            0.05249           NA
```

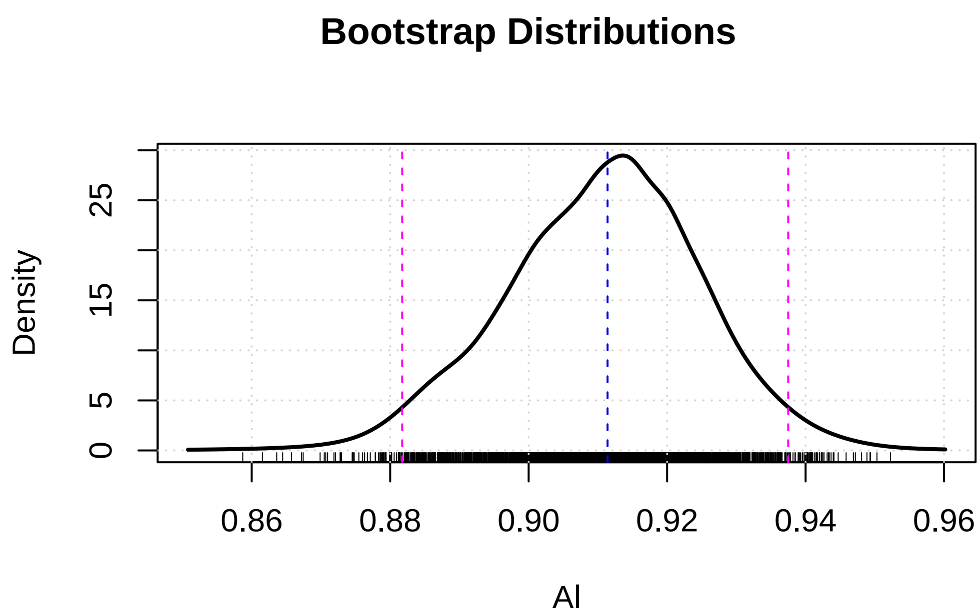
```
Confint(RTOAl, level=0.90)
```

```
    Estimate       5 %      95 %
Al 0.9113977 0.8876719 0.9351236
```

```
.bs.samplesRTOAl<- Boot(RTOAl, R=2500, method="case")
confint(.bs.samplesRTOAl, level=0.9, type="bca")
```

```
Bootstrap bca confidence intervals

         5 %      95 %
Al 0.8854027 0.9335376
```

```
plotBoot(.bs.samplesRTOAl)
```



6.10.4 Result

By excluding row 18, something interesting is happening:

criteria	OLR	RTO
r^2	0.915 *0.928*	0.9144 *0.9277*

criteria	OLR	RTO
rSEE	7.00 *6.66*	6.86 *6.50*
CI 0.05	0.7796 - 0.7521 *0.7597 - 0.7987*	0.8807 - 0.8723 *0.8877 - 0.8855*
CI 0.95	0.9873 - 0.9982 *1.0037 - 0.9971*	0.9244 - 0.9273 *0.9351 - 0.9348*

While the criteria of the RTO are getting better, the agreement of CILR5 and CIBS5 of the OLR are getting worse. Also, the range of values is increasing. This means, the robustness less good for this OLR. As RTO is still better, this does not affect the next step but it is worth noticing. As both linear regressions are giving the sample in line 23 as outlier, we are excluding it.

6.11 Eleventh iteration

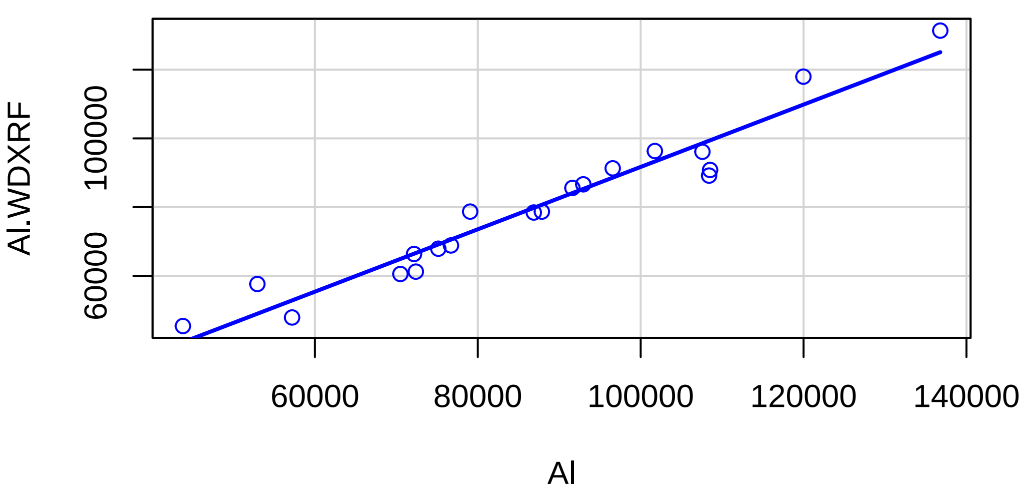
Now excluding row 23.

```
dataset1<- read.csv("../data_analytical//coefcorI_data.csv")
dataset<- dataset1[-c(25,14,27,28,24,1,8,11,18,23), ]
```

calculations

6.11.1 First impressions

```
scatterplot(Al.WDXRF~Al, regLine=TRUE, smooth=FALSE, boxplots=FALSE, data=dataset)
```




```
cor(dataset$Al,dataset$Al.WDXRF)
```

```
[1] 0.9701069
```

```r
numSummary(dataset[,c("Al", "Al.WDXRF"), drop=FALSE],
           statistics=c("mean", "sd", "quantiles"),quantiles=c(0,.25,.5,.75,1))
```

```
             mean       sd    0%      25%     50%      75%   100%  n
Al       86940.35 23178.81 43795 72338.75 87373.5 103201.2 136788 20
Al.WDXRF 79846.55 21678.37 45409 65083.50 78698.0  90937.0 131358 20
```

6.11.2 Ordinary Linear Regression (OLR)

```r
OLRAl<-lm(Al.WDXRF~Al, data=dataset)
summary(OLRAl)
```

```
Call:
lm(formula = Al.WDXRF ~ Al, data = dataset)

Residuals:
     Min       1Q   Median       3Q      Max 
-10142.8  -2942.0   -664.9   3461.5   8642.6 

Coefficients:
             Estimate Std. Error t value Pr(>|t|)    
(Intercept)  964.7972  4805.5128   0.201    0.843    
Al             0.9073     0.0535  16.960 1.62e-12 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 5405 on 18 degrees of freedom
Multiple R-squared:  0.9411,	Adjusted R-squared:  0.9378 
F-statistic: 287.6 on 1 and 18 DF,  p-value: 1.624e-12
```

```r
fitted.OLRAl <- fitted(OLRAl)
datasetb<-cbind(dataset,fitted.OLRAl)
RMS<-sqrt(mean((datasetb$Al-datasetb$fitted.OLRAl)^2)/nrow(datasetb))
RMS
```

```
[1] 1653.891
```

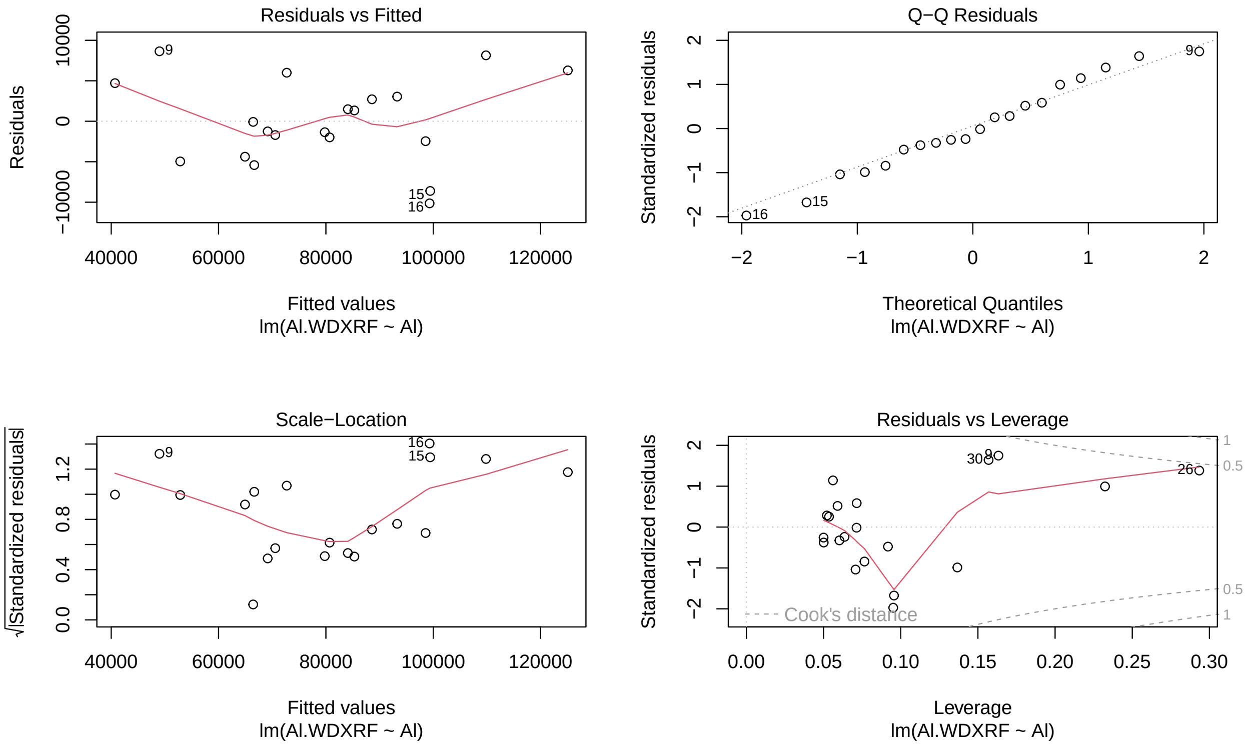
```r
SEE<-sigma(OLRAl)
SEE
```

```
[1] 5405.02
```

```r
mean<-mean(dataset$Al)
rSEE<-(SEE/mean)*100
rSEE
```

```
[1] 6.216929
```

```
oldparOLRAl <- par(oma=c(0,0,3,0), mfrow=c(2,2))
plot(OLRAl)
```




```
NULL
```

```
outlierTest(OLRAl)
```

```
No Studentized residuals with Bonferroni p < 0.05
Largest |rstudent|:
    rstudent unadjusted p-value Bonferroni p
16 -2.165474           0.044863      0.89726
```

```
Confint(OLRAl, level=0.90)
```

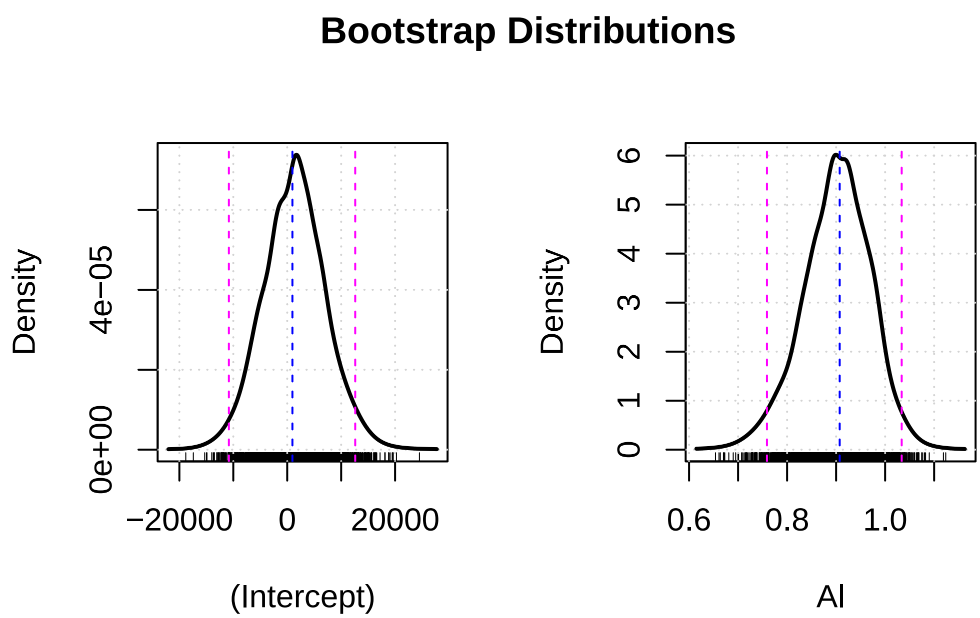
```
               Estimate          5 %        95 %
(Intercept) 964.7971979 -7368.2676888 9297.862085
Al            0.9073089     0.8145417    1.000076
```

```
.bs.samplesOLRAl<- Boot(OLRAl, R=2500, method="case")
confint(.bs.samplesOLRAl, level=0.9, type="bca")
```

```
Bootstrap bca confidence intervals

                      5 %         95 %
```

```
(Intercept) -9079.6990773 10404.183608
Al              0.7874974     1.017895
```

```
plotBoot(.bs.samplesOLRAl)
```



6.11.3 Regression Trough Origin (RTO)

```
RTOAl<-lm(Al.WDXRF~Al-1, data=dataset)
summary(RTOAl)
```

```
Call:
lm(formula = Al.WDXRF ~ Al - 1, data = dataset)

Residuals:
     Min       1Q   Median       3Q      Max 
-10304.9  -2999.1   -465.9   3519.2   9057.2 

Coefficients:
   Estimate Std. Error t value Pr(>|t|)    
Al  0.91770    0.01311      70   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 5267 on 19 degrees of freedom
Multiple R-squared:  0.9961,    Adjusted R-squared:  0.9959 
F-statistic:  4900 on 1 and 19 DF,  p-value: < 2.2e-16
```

```r
fitted.RTOAl <- fitted(RTOAl)
datasetc<-cbind(dataset,fitted.RTOAl)
RMS<-sqrt(mean((datasetc$Al-datasetc$fitted.RTOAl)^2)/nrow(datasetc))
RMS
```

```
[1] 1653.001
```

```r
SEE<-sigma(RTOAl)
SEE
```

```
[1] 5266.747
```

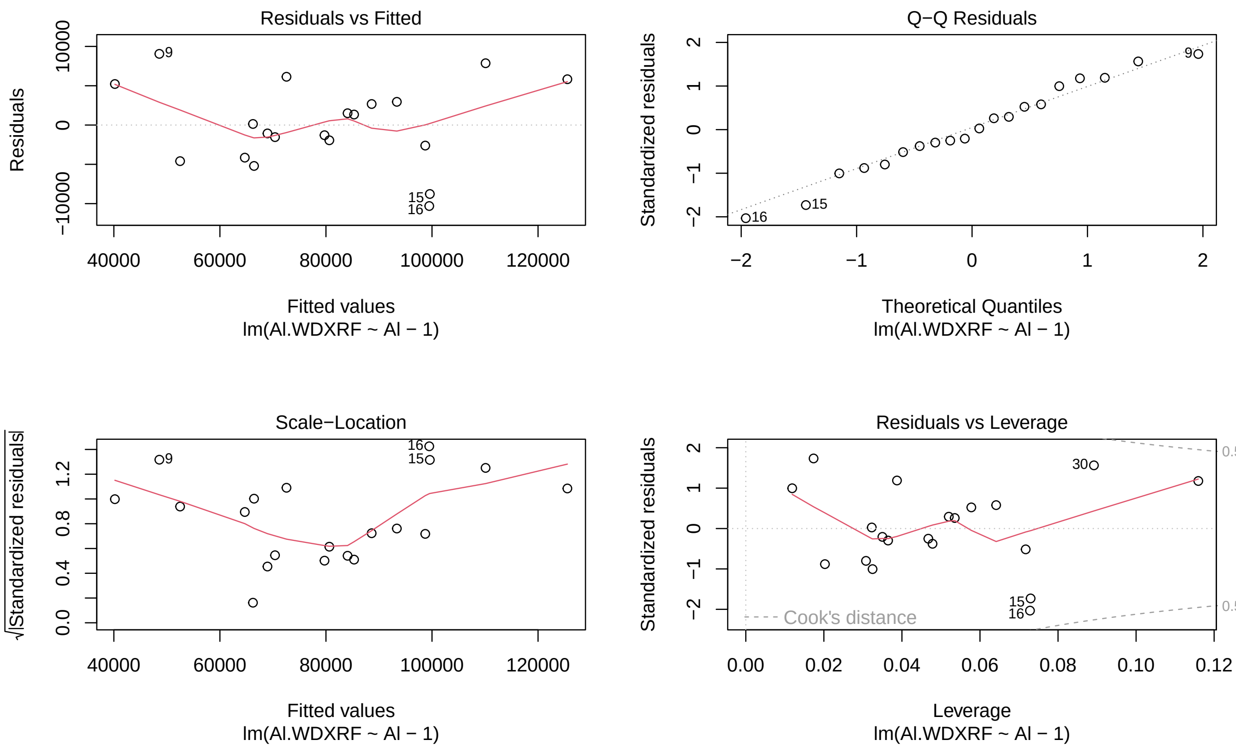
```r
mean<-mean(dataset$Al)
rSEE<-(SEE/mean)*100
rSEE
```

```
[1] 6.057886
```

```r
SSR<-(nrow(dataset)-1)*(SEE^2)
cSST <-sum(((dataset$Al.WDXRF)-mean(dataset$Al.WDXRF))^2)
Alcr2<-1-(SSR/cSST)
Alcr2
```

```
[1] 0.9409756
```

```r
oldparRTOAl <- par(oma=c(0,0,3,0), mfrow=c(2,2))
plot(RTOAl)
```




```
NULL
```

```
outlierTest(RTOAl)
```

```
No Studentized residuals with Bonferroni p < 0.05
Largest |rstudent|:
    rstudent unadjusted p-value Bonferroni p
16 -2.235542            0.03829       0.7658
```

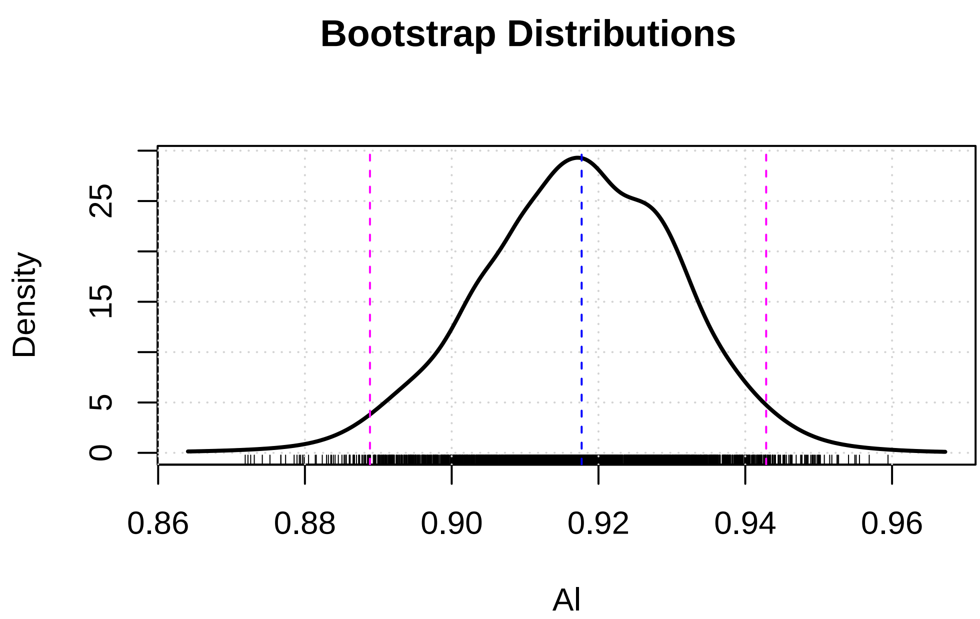
```
Confint(RTOAl, level=0.90)
```

```
    Estimate       5 %      95 %
Al 0.9177042 0.8950345 0.9403739
```

```
.bs.samplesRTOAl<- Boot(RTOAl, R=2500, method="case")
confint(.bs.samplesRTOAl, level=0.9, type="bca")
```

```
Bootstrap bca confidence intervals

         5 %      95 %
Al 0.8932434 0.9392519
```

```
plotBoot(.bs.samplesRTOAl)
```



6.11.4 Result - RTO

With this step, also the robustness of the RTO starts to decrease:

criteria	OLR	RTO
r^2	0.928 *0.9411*	0.9277 *0.9410*
rSEE	6.66 *6.22*	6.50 *6.06*
CI 0.05	0.7597 - 0.7987 *0.8145 - 0.7791*	0.8877 - 0.8855 *0.8950 - 0.8932*
CI 0.95	1.0037 - 0.9971 *1.0000 - 1.0113*	0.9351 - 0.9348 *0.9404 - 0.9390*

Therefore, excluding row 23 is not purposeful and the RTO of iteration ten is determined as the final linear regression to extract the coefcors from.

7 Example of extracting criteria and factors

source

Instructions following https://stackoverflow.com/questions/66771929/intercept-and-slope-functions-in-r, https://www.digitalocean.com/community/tutorials/rbind-function-r,https://www.statology.org/number-of-rows-in-r/, https://statisticsglobe.com/r-max-min-function/, https://www.statology.org/extract-r-squared-from-lm-in-r/, https://sparkbyexamples.com/r-programming/select-columns-by-index-position-in-r-2/, https://search.r-project.org/R/refmans/base/html/Round.html, https://www.statology.org/transpose-data-frame-in-r/ and https://datatofish.com/export-dataframe-to-csv-in-r/

In order to compile all information and factors of aluminium for coefcor I in a table, the relevant criteria must first be extracted from the previous calculations. The procedure differs somewhat for OLR and RTO. For this reason, both are described below by way of example, even though only the factors of the RTO are decisive for aluminium. To get the right coefcor, we first have to re-run again some parts the calculation of iteration ten. In the normal workflow, this step is omitted because the last iteration calculated is also the one from which the coefcors are extracted.

7.1 Extracting from OLR

First, the important factors slope (a) and intercept (b) are extracted.

```
cf <- Confint(OLRAl, level=0.90)
a<-cf[2]
a<-round(a,4)
b<-cf[1]
b<-round(b,0)
```

Next, r^2, rSEE, SEE and RMS are selected. Since the last three terms are used for OLR and RTO, they must be specified here.

```
r2<-summary(OLRAl)$r.squared
r2<-round(r2,4)

SEE<-sigma(OLRAl)
mean<-mean(dataset$Al)
rSEE<-(SEE/mean)*100
SEE<-round(SEE,0)
rSEE<-round(rSEE,2)

fitted.OLRAl <- fitted(OLRAl)
datasetb<-cbind(dataset,fitted.OLRAl)
RMS<-sqrt(mean((datasetb$Al-datasetb$fitted.OLRAl)^2)/nrow(datasetb))
RMS<-round(RMS,0)
```

Then the range of values for which the coefcor applies is defined. For this, the lowest (start) and highest measured value (end) are extracted.

```
start<-min(dataset$Al)
start<-round(start,0)
end<-max(dataset$Al)
end<-round(end,0)
```

Additionally, the values of the confidence levels (0.05 and 0.95) of the slope calculated from the filtered, whole data set (CILR) and by the bootstrap (CIBS) are to be provided.

```
CILR5<-cf[4]
CILR5<-round(CILR5,4)
CILR95<-cf[6]
```

```
CILR95<-round(CILR95,4)

cfb<-confint(.bs.samplesOLRAl, level=0.9, type="bca")
CIBS5<-cfb[2]
CIBS5<-round(CIBS5,4)
CIBS95<-cfb[4]
CIBS95<-round(CIBS95,4)
```

Last but not least, we extract percentage (outpct) and specification of the rows of the outliers (out). The number of samples in the data set (here 30) can be taken manually from the entire data set (first iteration - first impressions table).

```
outpct<-(100/30)*(30-nrow(dataset))
outpct<-round(outpct,2)
out<-'25-14-27-28-24-1-8-11-18'
```

Now all variables with their respective values are merged into one table, the column header is labelled with the element name and this column is explicitly selected and defined as its own short table.In this way, all elements can be summarised together in an overview table termed **_criteria.csv** (e.g. **coefcorI_criteria.csv**) at the end of the analysis when all coefcors for all elements are defined (see below).

```
AlOLR_tab<-rbind(a,b,r2,rSEE,SEE,RMS,start,end,CILR5,CIBS5,CILR95,CIBS95,out,outpct)
colnames(AlOLR_tab)[1] ="AlOLR"
AlOLR_criteria<-t(AlOLR_tab)
AlOLR_criteria
```

```
      a        b      r2      rSEE   SEE    RMS    start   end      CILR5
AlOLR "0.8979" "1257" "0.928" "6.66" "5821" "1744" "43795" "136788" "0.7987"
      CIBS5    CILR95   CIBS95   out                         outpct
AlOLR "0.7657" "0.9971" "1.0093" "25-14-27-28-24-1-8-11-18" "30"
```

7.2 Extracting from RTO

The compilation of the important criteria of the RTO works according to the same principle - only that now the RTO is selected to serve as input. And r^2 and b have to be extracted in a different way. In the case of the RTO, b is "0".

```
cf <- Confint(RTOAl, level=0.90)
a<-cf[1]
a<-round(a,4)
b='0'

r2<-Alcr2
r2<-round(r2,4)

SEE<-sigma(RTOAl)
mean<-mean(dataset$Al)
rSEE<-(SEE/mean)*100
```

```
SEE<-round(SEE,0)
rSEE<-round(rSEE,2)

fitted.RTOAl <- fitted(RTOAl)
datasetc<-cbind(dataset,fitted.RTOAl)
RMS<-sqrt(mean((datasetc$Al-datasetc$fitted.RTOAl)^2)/nrow(datasetc))
RMS<-round(RMS,0)

start<-min(dataset$Al)
start<-round(start,0)
end<-max(dataset$Al)
end<-round(end,0)

CILR5<-cf[2]
CILR5<-round(CILR5,4)
CILR95<-cf[3]
CILR95<-round(CILR95,4)

cfb<-confint(.bs.samplesRTOAl, level=0.9, type="bca")
CIBS5<-cfb[1]
CIBS5<-round(CIBS5,4)
CIBS95<-cfb[2]
CIBS95<-round(CIBS95,4)

outpct<-(100/30)*(30-nrow(dataset))
outpct<-round(outpct,2)
out<-'25-14-27-28-24-1-8-11-18'

Al_tab<-rbind(a,b,r2,rSEE,SEE,RMS,start,end,CILR5,CIBS5,CILR95,CIBS95,out,outpct)
colnames(Al_tab)[1] ="Al"
Al_criteria<-t(Al_tab)
Al_criteria
```

```
   a        b   r2       rSEE  SEE    RMS    start   end      CILR5    CIBS5
Al "0.9114" "0" "0.9277" "6.5" "5682" "1743" "43795" "136788" "0.8877" "0.8856"
   CILR95   CIBS95   out                        outpct
Al "0.9351" "0.9346" "25-14-27-28-24-1-8-11-18" "30"
```

Export Al-criteria as table

If desired, the aluminium criteria table can be exported as a .csv file:

```
write.csv(Al_criteria,"../data_processed//coefcorI_example_criteria_Al.csv",
          row.names=FALSE)
```

7.3 Exporting coefcor criteria (_criteria.csv)

> **source**
>
> Instructions following https://datatofish.com/export-dataframe-to-csv-in-r/

And finally, we can combine and export the criteria of the coefcors for Al and AlOLR in a joint table (**_criteria.csv**) which for this example is called **coefcorI_example_criteria.csv**.

```
criteria<- rbind(Al_criteria,AlOLR_criteria)
criteria
```

```
      a        b      r2       rSEE   SEE    RMS    start   end      CILR5
Al    "0.9114" "0"    "0.9277" "6.5"  "5682" "1743" "43795" "136788" "0.8877"
AlOLR "0.8979" "1257" "0.928"  "6.66" "5821" "1744" "43795" "136788" "0.7987"
      CIBS5    CILR95   CIBS95   out                        outpct
Al    "0.8856" "0.9351" "0.9346" "25-14-27-28-24-1-8-11-18" "30"
AlOLR "0.7657" "0.9971" "1.0093" "25-14-27-28-24-1-8-11-18" "30"
```

```
write.csv(criteria,"../data_processed//coefcorI_example_criteria.csv",row.names=TRUE)
```

And - tada! - we are done. Well, almost…

7.4 Exporting coefcor factors - slope and intercept (_factors.csv)

> **source**
>
> Instructions following https://campus.datacamp.com/courses/model-a-quantitative-trading-strategy-in-r/chapter-1-introduction-to-r-for-trading?ex=4 and https://rdrr.io/r/base/cbind.html

In order to apply the coefcor factors of slope (a) and intercept (b) to a data set using the R-script developed for the Munich Procedure, they have to be compiled in a certain way (**_factors.csv**). For this purpose we need to load the **_criteria.csv-file**, in our case **coefcorI_example_criteria.csv**.

```
dataset2<- read.csv("../data_processed//coefcorI_example_criteria.csv")
```

Then we can extract the necessary factors for slope and intercept from Al and Al_OLR.

```
Al_a<-dataset2[1,2]
Al_b<-dataset2[1,3]
AlOLR_a<-dataset2[2,2]
AlOLR_b<-dataset2[2,3]
```

We bind them together and export a table containing only the relevant coefcor factors in a spreadsheet called **_factors.csv** - in our example **coefcorI_example_factors.csv**. An example of factors of a complete coefcor would be **coefcorI_factors.csv**

```
s_i<- cbind(Al_a,Al_b,AlOLR_a,AlOLR_b)
s_i
```

```
       Al_a Al_b AlOLR_a AlOLR_b
[1,] 0.9114    0  0.8979    1257
```

```
write.csv(s_i,"../data_processed//coefcorI_example_factors.csv",row.names=TRUE)
```

8 Example of graphics export

source

Instructions following http://www.sthda.com/english/wiki/ggplot2-scatter-plots-quick-start-guide-r-software-and-data-visualization#basic-scatter-plots, http://www.sthda.com/english/wiki/ggplot2-title-main-axis-and-legend-titles, https://ggplot2.tidyverse.org/reference/geom_smooth.html, https://github.com/kassambara/ggpubr/issues/312, https://quarto.org/docs/computations/execution-options.html, https://stat.ethz.ch/R-manual/R-devel/library/graphics/html/hist.html,https://cran.r-project.org/web/packages/ggplotify/vignettes/ggplotify.html, https://www.rdocumentation.org/packages/cowplot/versions/1.1.1/topics/plot_grid, https://www.statology.org/plot-title-in-r/ and https://ggplot2.tidyverse.org/reference/ggsave.html

8.1 Exporting all relevant descriptive graphics for OLR and RTO

If you wish, you can compile and export all relevant graphics of the final run. For this purpose, the graphics must first be defined as variables and then combined in a joint export of OLR (**_OLR_allfig.eps**) and RTO (**_RTO_allfig.eps**) of the given element - in our case Al of coefcor I (**coefcorI_example_Al_OLR_allfig.eps/coefcorI_example_Al_RTO_allfig.eps**). The figure will include:

- scatter plots of p-XRF values (Al) to WDXRF values (Al.WDXRF), showing respectively the linear regression of OLR (Al_scatter_OLR) or RTO (Al_scatter_RTO).The algorithm used here is rounding the intercept values to the nearest 100, R^2 to two decimal digits. The otherwise usual formula y=ax+b is given in the form y=b+ax.

Attention - we have to take r^2 of the RTO explicitly from the previously performed calculations because it is calculated incorrectly by the algorithm section 6.1.3.4.

```
Al_scatter_OLR<-ggplot(dataset, aes(x=Al, y=Al.WDXRF))+geom_point()+
  geom_smooth(method=lm,se=FALSE,formula = y ~ x)+ ggtitle("OLR")+
  theme(plot.title = element_text(color="black", size=9,face="bold"))+
  stat_poly_eq(formula = y ~ x,aes(label = paste(..eq.label.., ..rr.label..,
        sep = "~~~")),parse = TRUE, coef.digits = 4, f.digits = 4,rr.digits=4)

Alcr2<-round(Alcr2,4)
```

```
Al_scatter_RTO<-ggplot(dataset, aes(x=Al, y=Al.WDXRF))+geom_point()+
  geom_smooth(method=lm,se=FALSE,formula = y ~ x-1)+ ggtitle("RTO")+
  theme(plot.title = element_text(color="black", size=9,face="bold"))+
  stat_poly_eq(formula = y ~ x - 1,aes(label = paste(..eq.label..,
                sep = "~~", "R2==", Alcr2)),parse = TRUE, coef.digits = 4)
```

- scatter plots of WDXRF values (Al.WDXRF) to fitted values. The latter term refers to the recalculated p-XRF values based on the specified values for slope (RTO - scatter_Al_RTO_WDXRF_fitted) and intercept (ORL - scatter_Al_OLR_WDXRF_fitted).The values shown in the graph for the linear regression model chosen above (in our case the RTO - see section 7.2) should for slope and r^2 be as near to 1 as possible, intercept close to 0. The better these criteria are met, the higher the quality of the recalculation of p-XRF measurements made possible by coefcors.

Here we can use the r^2 (Alcr2) calculated above in section 7.2 and used for Al_scatter_RTO as only the scaling and origin of the data have been changed. However, the relationship between the data points and therefore r^2 remains the same as the linear function is still the best fit to the data.

```
Al_scatter_OLR_WDXRF_fitted<-ggplot(datasetb, aes(x=fitted.OLRAl, y=Al.WDXRF))+
  geom_point()+geom_smooth(method=lm,se=FALSE,formula = y ~ x)+ ggtitle("OLR")+
  theme(plot.title = element_text(color="black", size=9,face="bold"))+
  stat_poly_eq(formula = y ~ x,aes(label = paste(..eq.label.., ..rr.label..,
         sep = "~~~")),parse = TRUE, coef.digits = 4, f.digits = 4,rr.digits=4)

Al_scatter_RTO_WDXRF_fitted<-ggplot(datasetc, aes(x=fitted.RTOAl, y=Al.WDXRF))+
  geom_point()+geom_smooth(method=lm,se=FALSE,formula = y ~ x-1)+ ggtitle("RTO")+
  theme(plot.title = element_text(color="black", size=9,face="bold"))+
   stat_poly_eq(formula = y ~ x - 1,aes(label = paste(..eq.label..,
                sep = "~~", "R2==", Alcr2)),parse = TRUE, coef.digits = 4)
```

- the graphs of the diagnostics of the linear regressions (now compiled as OLRAl_1_2/3_5 and RTOAl_1_2/3_5) we already know from our steps in section 6.1.2.4 and section 6.1.3.5.

```
par(mar = c(0.1, 0.1, 0.1,0.1))

OLRAl1<-as_grob(~plot(OLRAl,1))
OLRAl2<-as_grob(~plot(OLRAl,2))
OLRAl3<-as_grob(~plot(OLRAl,3))
OLRAl5<-as_grob(~plot(OLRAl,5))

OLRAl_1_2<-plot_grid(OLRAl1,OLRAl2)
OLRAl_3_5<-plot_grid(OLRAl3,OLRAl5)

RTOAl1<-as_grob(~plot(RTOAl,1))
RTOAl2<-as_grob(~plot(RTOAl,2))
RTOAl3<-as_grob(~plot(RTOAl,3))
RTOAl5<-as_grob(~plot(RTOAl,5))
```

```
RTOAl_1_2<-plot_grid(RTOAl1,RTOAl2)
RTOAl_3_5<-plot_grid(RTOAl3,RTOAl5)
```

- histograms of the distribution of the residuals for OLR (Al_hist_OLR) and RTO (Al_hist_RTO). To do this, the values of the residuals must first be extracted from the linear regression calculations. This is done in the same way as in section 6.1.2.2 and section 6.1.3.2.

```
residuals.OLRAl <- residuals(OLRAl)
datasetb<-cbind(datasetb,residuals.OLRAl)

Al_hist_OLR<-as_grob(~Hist(datasetb$residuals.OLRAl, scale="frequency",
                           breaks="Sturges", col="darkgray",xlab="OLR residuals"))

residuals.RTOAl <- residuals(RTOAl)
datasetc<-cbind(datasetc,residuals.RTOAl)

Al_hist_RTO<-as_grob(~Hist(datasetc$residuals.RTOAl, scale="frequency",
                           breaks="Sturges", col="darkgray",xlab="RTO residuals"))
```

- finally, we assign the density distributions of the bootstrap known from steps section 6.1.2.6 and section 6.1.3.7 for each linear regression (Al_boot_OLR/Al_boot_RTO) to separate variables.

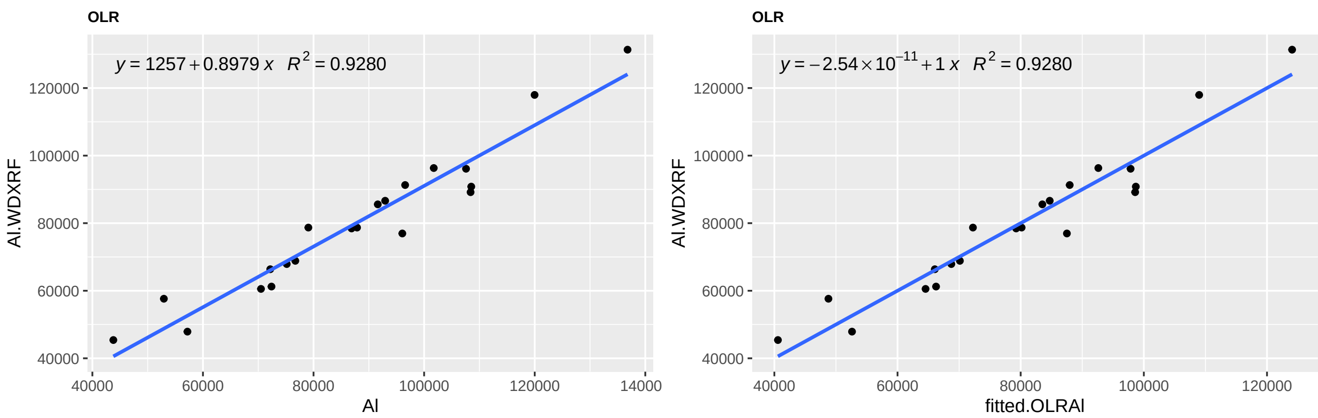
```
Al_boot_OLR<-as_grob(~plotBoot(.bs.samplesOLRAl)+title('OLR'))
Al_boot_RTO<-as_grob(~plotBoot(.bs.samplesRTOAl)+title('RTO'))
```

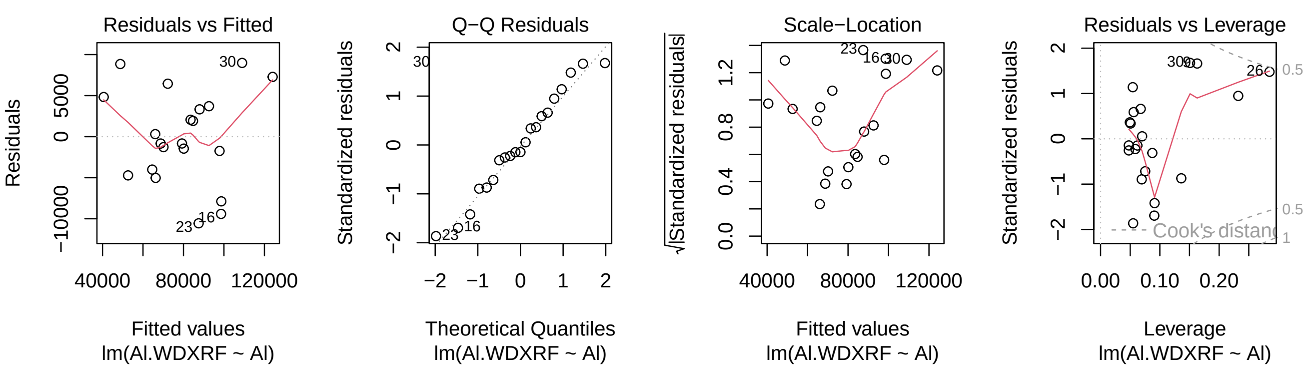
Now that all the graphics are defined we can put them together as **coefcorI_example_Al_OLR_allfig.eps**.

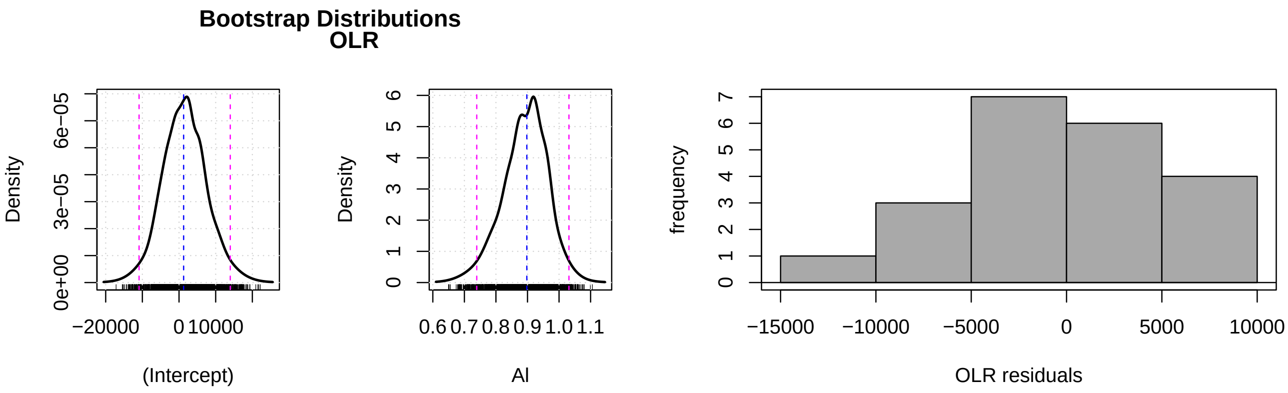
```
coefcorI_example_Al_OLR_allfig<-plot_grid(Al_scatter_OLR,Al_scatter_OLR_WDXRF_fitted,
                                          OLRAl_1_2,OLRAl_3_5,Al_boot_OLR,Al_hist_OLR,
                                          ncol=2,nrow=3,rel_heights=c(1,1.5,1))

ggsave("coefcorI_example_Al_OLR_allfig.eps",path=("..//graphics"),plot=last_plot(),
       device="eps",height=25,width=30,unit=c("cm"))
```

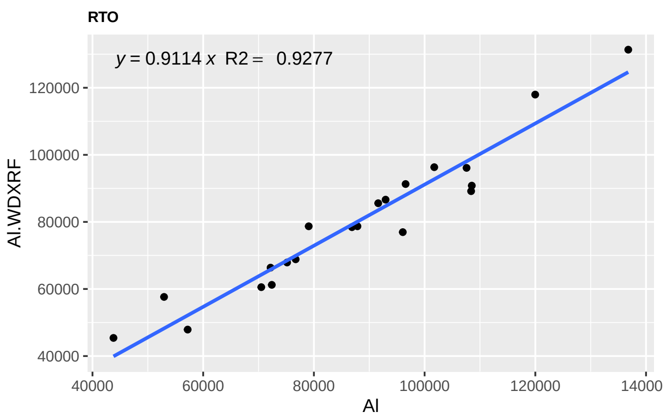
Same procedure for the RTO graphics named **coefcorI_example_Al_RTO_allfig.eps**.

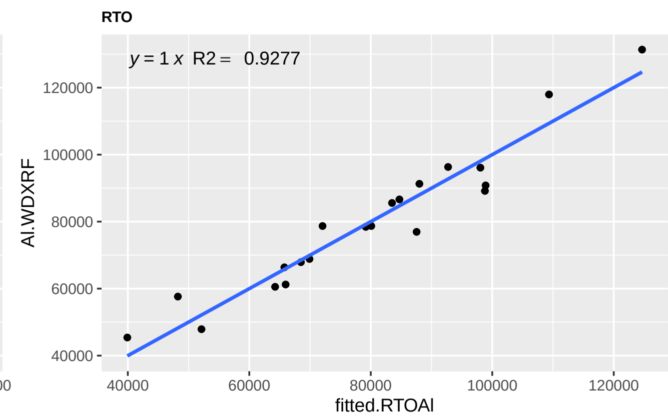
```
coefcorI_example_Al_RTO_allfig<-plot_grid(Al_scatter_RTO,Al_scatter_RTO_WDXRF_fitted,
                                          RTOAl_1_2,RTOAl_3_5,Al_boot_RTO,Al_hist_RTO,
                                          ncol=2,nrow=3,rel_heights=c(1,1.5,1))

ggsave("coefcorI_example_Al_RTO_allfig.eps",path=("..//graphics"),plot=last_plot(),
       device="eps",height=25,width=30,unit=c("cm"))
```

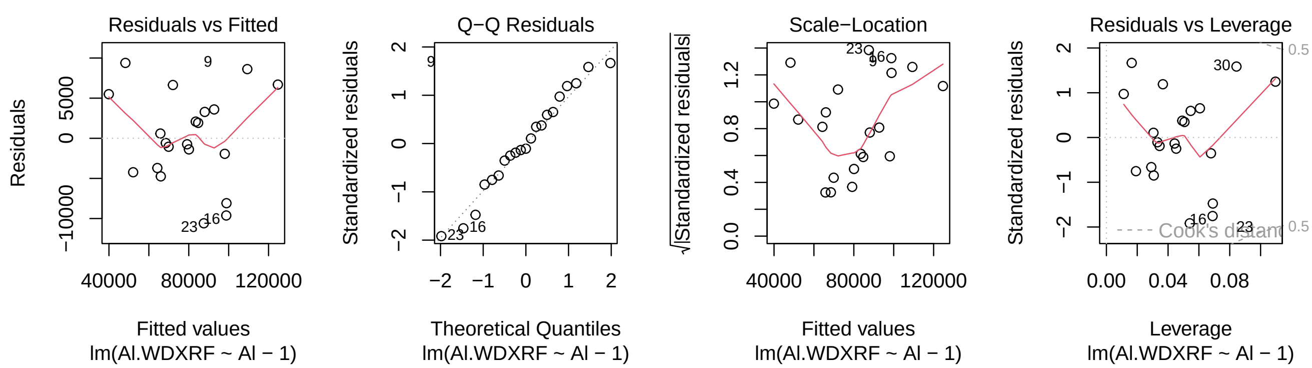

OLR
$y = 1257 + 0.8979\,x \quad R^2 = 0.9280$
Al.WDXRF
Al
OLR
$y = -2.54 \times 10^{-11} + 1\,x \quad R^2 = 0.9280$
Al.WDXRF
fitted.OLRAl

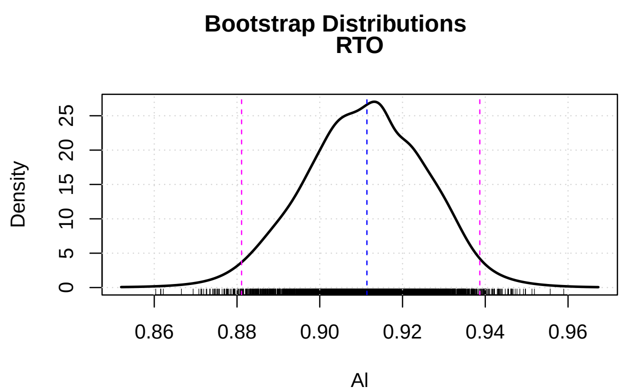

Residuals vs Fitted
Residuals
Fitted values
lm(Al.WDXRF ~ Al)
Q−Q Residuals
Standardized residuals
Theoretical Quantiles
lm(Al.WDXRF ~ Al)
Scale−Location
√|Standardized residuals|
Fitted values
lm(Al.WDXRF ~ Al)
Residuals vs Leverage
Standardized residuals
Cook's distance
Leverage
lm(Al.WDXRF ~ Al)

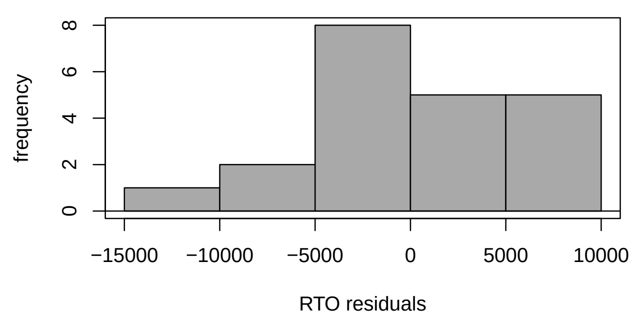

Bootstrap Distributions
OLR
Density
(Intercept)
Density
Al
frequency
OLR residuals


RTO
y = 0.9114 x R2 = 0.9277
Al.WDXRF
Al


RTO
y = 1 x R2 = 0.9277
Al.WDXRF
fitted.RTOAl


Residuals vs Fitted
Residuals
Fitted values
lm(Al.WDXRF ~ Al − 1)
Q−Q Residuals
Standardized residuals
Theoretical Quantiles
lm(Al.WDXRF ~ Al − 1)
Scale−Location
√|Standardized residuals|
Fitted values
lm(Al.WDXRF ~ Al − 1)
Residuals vs Leverage
Standardized residuals
Leverage
lm(Al.WDXRF ~ Al − 1)


Bootstrap Distributions
RTO
Density
Al


frequency
RTO residuals

8.2 Exporting the traditionally presented scatter plots

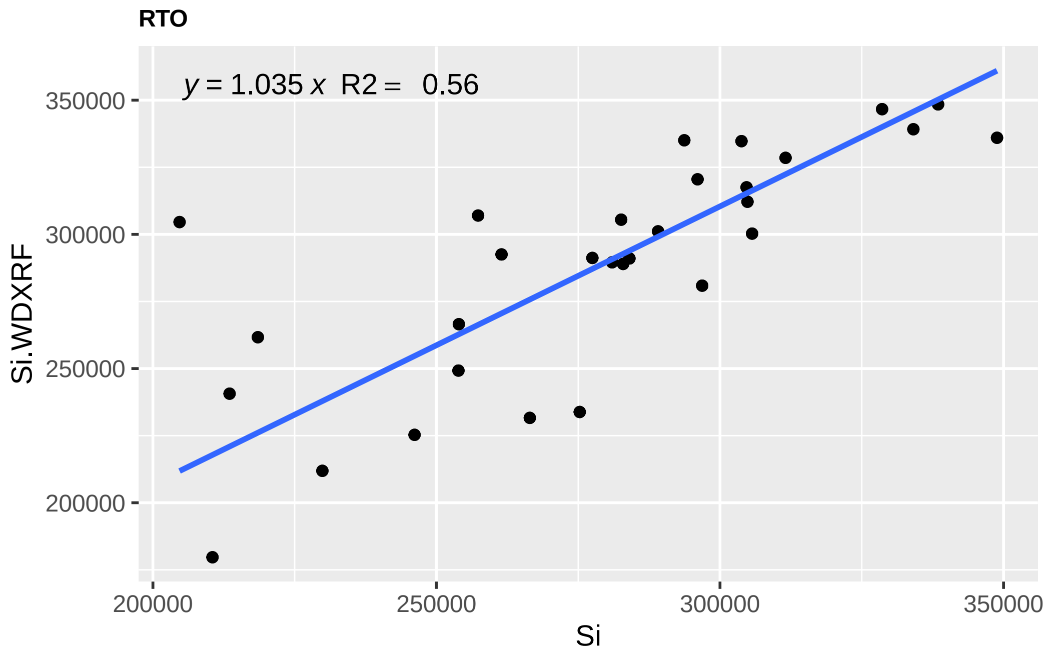
source

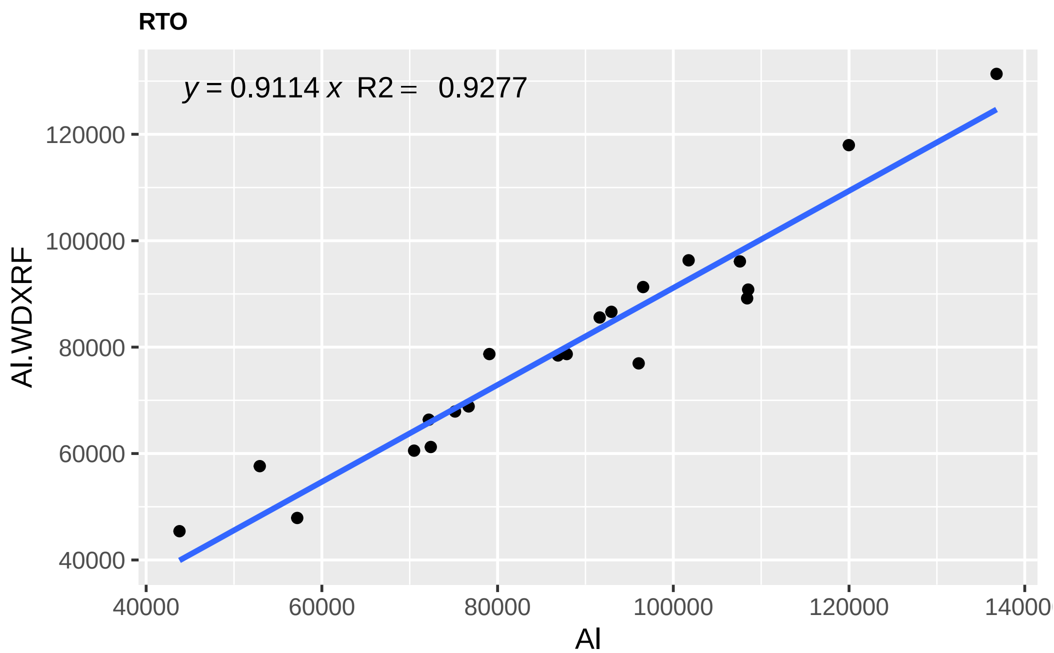
Instructions following https://www.digitalocean.com/community/tutorials/r-read-csv-file-into-data-frame, https://ggplot2.tidyverse.org/reference/ggsave.html,http://www.sthda.com/english/wiki/ggplot2-scatter-plots-quick-start-guide-r-software-and-data-visualization#basic-scatter-plots, https://www.rdocumentation.org/packages/cowplot/versions/1.1.1/topics/plot_grid, https://www.statology.org/plot-title-in-r/, https://datatofish.com/export-dataframe-to-csv-in-r/

In addition to the named graphics which show the criteria of the selected linear regression of a particular element, you can of course create other graphs. Here we create the combined display of scatter plots, functions and r^2 values of different chemical elements that are traditionally used to present coefcors or calibrations.

In order to create such an example, a diagram for p-XRF values and WDXRF values of silicium (without any calculations to find the best linear regression but displaying a RTO of this data) is created using the data set **coefcorI_data.csv** and displayed together with the final scatter plot we defined above for aluminium. This graphics export is called **_traditional_scatterplots.eps** (in our example therefore **coefcorI_example_traditional_scatterplots.eps**)

```
dataset<- read.csv("../data_analytical//coefcorI_data.csv")

RTOSi<-lm(Si.WDXRF~Si-1, data=dataset)
SEE_Si<-sigma(RTOSi)
SSR_Si<-(nrow(dataset)-1)*(SEE_Si^2)
cSST_Si <-sum(((dataset$Si.WDXRF)-mean(dataset$Si.WDXRF))^2)
Sicr2<-1-(SSR_Si/cSST_Si)
Sicr2<-round(Sicr2,2)

Si_scatter_RTO<-ggplot(dataset, aes(x=Si, y=Si.WDXRF))+geom_point()+
  geom_point()+geom_smooth(method=lm,se=FALSE,formula = y ~ x-1)+ ggtitle("RTO")+
  theme(plot.title = element_text(color="black", size=9,face="bold"))+
  stat_poly_eq(formula = y ~ x - 1,aes(label = paste(..eq.label..,
               sep = "~~", "R2==", Sicr2)),parse = TRUE, coef.digits = 4)

coefcorI_example_traditional_scatterplots<-plot_grid(Si_scatter_RTO,
      Al_scatter_RTO,ncol=2,nrow=1)

ggsave("coefcorI_example_traditional_scatterplots.eps",path=("..//graphics"),
       plot=last_plot(),device="eps",height=8,width=15.3,unit=c("cm"))
```


RTO
y = 1.035 x R2 = 0.56
350000
300000
250000
200000
Si.WDXRF
200000
250000
300000
350000
Si


RTO
y = 0.9114 x R2 = 0.9277
120000
100000
80000
60000
40000
Al.WDXRF
40000
60000
80000
100000
120000
14000
Al

9 Example of applying coefcors obtained from p-XRF and laboratory values to a dataset

The **_factor.csv** of a coefcor is only valid in the respective time period and is therefore used to apply the coefcors to a p-XRF data set produced during that time. In this example we use **coefcorI_factors.csv** which is valid for data of the Niton XL3t No 97390 collected between July 2017 and August 2018 (Schauer 2024).

9.1 Loading and formating the data

> **source**
>
> Instructions following https://www.digitalocean.com/community/tutorials/r-read-csv-file-into-data-frame and https://sparkbyexamples.com/r-programming/r-select-function-from-dplyr/

In the following, the application of the correction factors (_factors.csv) - in this case from coefcor I (**coefcorI_factors.csv**) - to a data set generated by p-XRF (e.g. measurements on ancient pottery) is demonstrated. coefcorI_data.csv is used as an example data set. First we load both files:

```
coefcor<- read.csv("../data_processed//coefcorI_factors.csv")
dataset<- read.csv("../data_analytical//coefcorI_data.csv")
```

Then we create a new dataset containing only the columns with p-XRF values and the sample names because we only want to recalculate those. We also define ‘sample’ as a variable to be able to add it to the recalculated data.

```
dataset<-select(dataset,"Sample","Si","Ti","Al","Fe","Mn","Mg","Ca","K","P","S",
                "Cl","Sc","V","Cr","Co","Ni","Cu","Zn","As","Se","Rb","Sr","Y",
                "Zr","Nb","Mo","Pd","Ag","Cd","Sn","Sb","Te","Cs","Ba","La","Ce",
                "Hf","Ta","W","Re","Au","Hg","Pb","Bi","Th","U")

Sample<-dataset$Sample
```

9.2 Performing the calculations

> **source**
>
> Instructions following https://cran.r-project.org/web/packages/dplyr/vignettes/rowwise.htmlr and https://dplyr.tidyverse.org/reference/mutate.html

From this data, the necessary variables are extracted: new values to be calculated for the respective element (e.g. Si) as well as the slope _a (e.g. Si_a) and intercept _b (e.g. Si_b) to be applied.

```
Si<-dataset$Si
Si_a<-coefcor$Si_a
Si_b<-coefcor$Si_b
```

The recalculation is performed using these variables. It is then necessary to decide whether the values should be given in ppm (e.g. Si_ppm), weight percent (e.g. Sipct) or, in the case of the major elements Si, Ti, Al, Fe, Mn, Mg, Ca, K and P, in oxidation percentage (e.g. SiO2).

```
Sippm<-Si_a*Si+Si_b
Sippm
```

```
 [1] 282702.9 258373.1 276851.7 303960.0 316729.8 315289.7 245772.0 341986.0
 [9] 249612.5 334382.8 286544.2 295039.9 309324.5 238982.8 309950.6 349988.8
[17] 315978.1 316104.4 338631.4 276907.9 270892.6 299213.1 293321.0 321274.1
[25] 279523.2 300073.3 297725.2 298952.8 243451.7 307523.2
```

```
Sipct<-(Si_a*Si+Si_b)*0.0001
Sipct
```

```
 [1] 28.27029 25.83731 27.68517 30.39600 31.67298 31.52897 24.57720 34.19860
 [9] 24.96125 33.43828 28.65442 29.50399 30.93245 23.89828 30.99506 34.99888
[17] 31.59781 31.61044 33.86314 27.69079 27.08926 29.92131 29.33210 32.12741
[25] 27.95232 30.00733 29.77252 29.89528 24.34517 30.75232
```

```
SiO2<-(Si_a*Si+Si_b)*0.00021393
SiO2
```

```
 [1] 60.47863 55.27376 59.22688 65.02616 67.75800 67.44992 52.57800 73.16106
 [9] 53.39960 71.53451 61.30039 63.11789 66.17379 51.12559 66.30773 74.87311
[17] 67.59721 67.62422 72.44342 59.23891 57.95206 64.01065 62.75017 68.73017
[25] 59.79839 64.19468 63.69236 63.95497 52.08162 65.78844
```

According to the Munich Procedure, the major elements are given in oxidation percentage.

```
Ti<-dataset$Ti
Ti_a<-coefcor$Ti_a
Ti_b<-coefcor$Ti_b
TiO2<-(Ti_a*Ti+Ti_b)*0.0001668

Al<-dataset$Al
Al_a<-coefcor$Al_a
Al_b<-coefcor$Al_b
Al2O3<-(Al_a*Al+Al_b)*0.00018895

Fe<-dataset$Fe
Fe_a<-coefcor$Fe_a
Fe_b<-coefcor$Fe_b
Fe2O3<-(Fe_a*Fe+Fe_b)*0.000143

Mn<-dataset$Mn
```

```
Mn_a<-coefcor$Mn_a
Mn_b<-coefcor$Mn_b
MnO<-(Mn_a*Mn+Mn_b)*0.00012912

Mg<-dataset$Mg
Mg_a<-coefcor$Mg_a
Mg_b<-coefcor$Mg_b
MgO<-(Mg_a*Mg+Mg_b)*0.00016583

Ca<-dataset$Ca
Ca_a<-coefcor$Ca_a
Ca_b<-coefcor$Ca_b
CaO<-(Ca_a*Ca+Ca_b)*0.00013992

K<-dataset$K
K_a<-coefcor$K_a
K_b<-coefcor$K_b
K2O<-(K_a*K+K_b)*0.00012046

P<-dataset$P
P_a<-coefcor$P_a
P_b<-coefcor$P_b
P2O5<-(P_a*P+P_b)*0.00022914
```

For sample-based normalisation, the created variables are combined into a new data set (data_norm) and supplemented with the sum of the main elements (sum) calculated line by line (data_norm_withsum). This new column is used to compute the percentage deviation from 100% of the major elements per row i.e. sample (sumpct).

```
data_norm<-data.frame(SiO2,TiO2,Al2O3,Fe2O3,MnO,MgO,CaO,K2O,P2O5)

data_norm_withsum<-data_norm %>% rowwise() %>%  mutate(sum = sum(c(SiO2,TiO2,Al2O3,
                                                      Fe2O3,MnO,MgO,CaO,K2O,P2O5)))

sumpct<-100/data_norm_withsum$sum
```

The factor sumpct will now be used for the recalculation of the main elements.

```
SiO2<-SiO2*sumpct
TiO2<-TiO2*sumpct
Al2O3<-Al2O3*sumpct
Fe2O3<-Fe2O3*sumpct
MnO<-MnO*sumpct
MgO<-MgO*sumpct
CaO<-CaO*sumpct
K2O<-K2O*sumpct
P2O5<-P2O5*sumpct
```

Once the main elements have been recalculated, the cofcor factors (here coefcor I) are applied to the trace elements. These are given in ppm. As examples, the calculations for rubidium and strontium are shown.

```
Rb<-dataset$Rb
Rb_a<-coefcor$Rb_a
Rb_b<-coefcor$Rb_b
Rb<-Rb_a*Rb+Rb_b

Sr<-dataset$Sr
Sr_a<-coefcor$Sr_a
Sr_b<-coefcor$Sr_b
Sr<-Sr_a*Sr+Sr_b
```

9.3 Compiling and exporting the processed data (_cor_data.csv)

> **source**
>
> Instructions following https://search.r-project.org/R/refmans/base/html/Round.html, https://www.statology.org/transpose-data-frame-in-r/ and https://datatofish.com/export-dataframe-to-csv-in-r/

Finally, all the newly calculated variables are combined with the sample number in a new table called **_cor_data.csv** (in our case **coefcorI_exsample_cor_data.csv**). Major element values are rounded to one decimal place, trace element values to whole digits. This table is then exported. It contains quantitative, processed data and forms the basis for further analysis.

```
coefcorI_exsample_cor_data<-data.frame(Sample,SiO2,TiO2,Al2O3,Fe2O3,MnO,MgO,CaO,K2O,
                                       P2O5,Rb,Sr)

coefcorI_exsample_cor_data[, c("SiO2","TiO2","Al2O3","Fe2O3","MnO","MgO","CaO",
"K2O","P2O5")] <- round(coefcorI_exsample_cor_data[, c("SiO2","TiO2","Al2O3",
                                  "Fe2O3","MnO","MgO","CaO","K2O","P2O5")], 1)

coefcorI_exsample_cor_data[, c("Rb","Sr")] <- round(coefcorI_exsample_cor_data[, c("Rb","Sr")],

coefcorI_exsample_cor_data
```

```
   Sample SiO2 TiO2 Al2O3 Fe2O3 MnO MgO  CaO K2O P2O5  Rb  Sr
1    H006 70.1  0.7  13.3   5.9 0.1 1.9  3.8 3.4  0.8 117 163
2    H045 56.6  0.5  10.1   3.8 0.1 2.8 24.3 1.4  0.4 141 295
3    H047 60.7  0.6  16.2   4.9 0.1 2.5 12.0 2.8  0.2 138 658
4    H052 69.0  0.7  15.9   6.1 0.2 2.3  2.4 3.1  0.4 138 135
5    H053 70.6  0.7  15.8   5.5 0.1 2.0  2.1 3.0  0.1 141 134
6    H056 70.8  1.3  16.8   7.3 0.1 0.7  1.0 1.9  0.1 118  78
7    H057 60.1  0.5   8.6   3.4 0.1 1.4 23.4 2.1  0.5  77 264
8    H058 72.1  0.6  13.9   4.4 0.0 2.5  3.9 2.4  0.1 107 109
9    H060 66.0  0.6  11.3   4.1 0.1 1.3 13.6 2.9  0.2  89 225
10   H066 75.5  0.7  13.2   5.0 0.0 1.3  2.1 2.2  0.1 100  97
```

```
11   H076 56.6  0.5  15.2   5.0 0.1 2.7 16.8 2.8  0.3 125 438
12   H077 70.3  0.7  15.2   6.7 0.2 1.5  2.5 2.2  0.8 135 189
13   H078 74.6  0.6  14.0   4.3 0.1 1.4  2.2 2.4  0.4 117 143
14   H079 71.5  0.8  14.4   7.5 0.1 1.1  0.9 3.2  0.5 104  71
15   H087 63.2  0.7  17.8   7.2 0.1 3.0  3.7 3.8  0.4 156 142
16   H088 71.4  0.7  17.8   3.5 0.0 1.8  1.0 3.6  0.1 280  99
17   H089 68.3  0.6  18.7   4.1 0.0 1.5  2.7 3.6  0.4 225 140
18   H092 66.3  0.6  17.2   3.9 0.1 3.9  4.9 2.9  0.2 144 170
19   H095 74.3  1.7  17.1   3.9 0.0 0.6  0.9 1.3  0.2  82 105
20   H100 63.9  0.7  14.2   5.1 0.1 1.9 11.1 2.6  0.4 112 480
21   H102 59.5  0.5  13.3   4.8 0.1 2.8 16.4 2.5  0.1 108 339
22   H111 63.8  0.8  17.5   6.9 0.1 3.2  4.1 3.5  0.2 172 138
23   H112 59.5  0.7  15.7   5.9 0.1 3.1 13.2 1.6  0.3 162 429
24   H114 70.2  1.1  22.5   1.5 0.0 0.5  2.4 1.7  0.2 115 117
25   H124 70.2  2.1  20.4   3.2 0.0 0.7  1.2 1.9  0.4  72  95
26   H125 66.6  1.2  24.4   3.1 0.0 1.1  1.0 2.4  0.2 155 147
27   H126 66.6  1.3  24.3   3.1 0.0 1.5  0.8 2.2  0.2 175 141
28   H127 68.7  2.2  20.6   4.7 0.1 0.6  1.0 1.9  0.2  97 294
29   H138 54.1  0.5  12.6   4.3 0.1 3.8 21.8 2.5  0.2  83 317
30   H139 71.1  1.1  22.3   3.5 0.0 0.3  0.5 1.1  0.1  63  77
```

```r
write.csv(coefcorI_exsample_cor_data,
          "../data_processed//coefcorI_example_cor_data.csv",row.names=TRUE)
```

10 Example of applying device-internal coefcors to a dataset

The **_factor.csv** needed to correct values from the time period in which they were taken (for example from coefcor I) to fit the data of different time period (for example coefcor II; see Schauer 2023 & Schauer 2024) has to be chosen. In this example we use **coefcorItoII_factors.csv**.

10.1 Loading and formating the data

> **source**
>
> Instructions following https://www.digitalocean.com/community/tutorials/r-read-csv-file-into-data-frame and https://sparkbyexamples.com/r-programming/r-select-function-from-dplyr/

In the following, the application of the correction factors (_factors.csv) - in this case from coefcor ItoII (**coefcorItoII_factors.csv**) - to a data set generated by p-XRF (e.g. measurements on ancient pottery) is demonstrated. As before, we load both files and create the new dataset and variable.

```r
coefcor<- read.csv("../data_processed//coefcorItoII_factors.csv")
dataset<- read.csv("../data_analytical//coefcorI_data.csv")

dataset<-select(dataset,"Sample","Si","Ti","Al","Fe","Mn","Mg","Ca","K","P","S",
                "Cl","Sc","V","Cr","Co","Ni","Cu","Zn","As","Se","Rb","Sr","Y",
```

```
                "Zr","Nb","Mo","Pd","Ag","Cd","Sn","Sb","Te","Cs","Ba","La","Ce",
                "Hf","Ta","W","Re","Au","Hg","Pb","Bi","Th","U")

Sample<-dataset$Sample
```

10.2 Performing the calculations

As this calculation is performed to fit the data before applying the coefcor used in the period to which the data are correct (in our case this would be coefcor II), no calculation of oxidation percentage or normalisation needs to be performed. For all elements the corrections are applied and expressed in ppm.

```
Si<-dataset$Si
Si_a<-coefcor$Si_a
Si_b<-coefcor$Si_b
Si<-Si_a*Si+Si_b

Ti<-dataset$Ti
Ti_a<-coefcor$Ti_a
Ti_b<-coefcor$Ti_b
Ti<-Ti_a*Ti+Ti_b

Al<-dataset$Al
Al_a<-coefcor$Al_a
Al_b<-coefcor$Al_b
Al<-Al_a*Al+Al_b

Fe<-dataset$Fe
Fe_a<-coefcor$Fe_a
Fe_b<-coefcor$Fe_b
Fe<-Fe_a*Fe+Fe_b

Mn<-dataset$Mn
Mn_a<-coefcor$Mn_a
Mn_b<-coefcor$Mn_b
Mn<-Mn_a*Mn+Mn_b

Mg<-dataset$Mg
Mg_a<-coefcor$Mg_a
Mg_b<-coefcor$Mg_b
M<-Mg_a*Mg+Mg_b

Ca<-dataset$Ca
Ca_a<-coefcor$Ca_a
Ca_b<-coefcor$Ca_b
C<-Ca_a*Ca+Ca_b
```

```
K<-dataset$K
K_a<-coefcor$K_a
K_b<-coefcor$K_b
K<-K_a*K+K_b

P<-dataset$P
P_a<-coefcor$P_a
P_b<-coefcor$P_b
P<-P_a*P+P_b

Cl<-dataset$Cl
Cl_a<-coefcor$Cl_a
Cl_b<-coefcor$Cl_b
Cl<-Cl_a*Cl+Cl_b

V<-dataset$V
V_a<-coefcor$V_a
V_b<-coefcor$V_b
V<-V_a*V+V_b

Cr<-dataset$Cr
Cr_a<-coefcor$Cr_a
Cr_b<-coefcor$Cr_b
Cr<-Cr_a*Cr+Cr_b

Ni<-dataset$Ni
Ni_a<-coefcor$Ni_a
Ni_b<-coefcor$Ni_b
Ni<-Ni_a*Ni+Ni_b

Zn<-dataset$Zn
Zn_a<-coefcor$Zn_a
Zn_b<-coefcor$Zn_b
Zn<-Zn_a*Zn+Zn_b

As<-dataset$As
As_a<-coefcor$As_a
As_b<-coefcor$As_b
As<-As_a*As+As_b

Rb<-dataset$Rb
Rb_a<-coefcor$Rb_a
Rb_b<-coefcor$Rb_b
Rb<-Rb_a*Rb+Rb_b

Sr<-dataset$Sr
Sr_a<-coefcor$Sr_a
Sr_b<-coefcor$Sr_b
```

```
Sr<-Sr_a*Sr+Sr_b

Y<-dataset$Y
Y_a<-coefcor$Y_a
Y_b<-coefcor$Y_b
Y<-Y_a*Y+Y_b

Zr<-dataset$Zr
Zr_a<-coefcor$Zr_a
Zr_b<-coefcor$Zr_b
Zr<-Zr_a*Zr+Zr_b

Nb<-dataset$Nb
Nb_a<-coefcor$Nb_a
Nb_b<-coefcor$Nb_b
Nb<-Nb_a*Nb+Nb_b

Ba<-dataset$Ba
Ba_a<-coefcor$Ba_a
Ba_b<-coefcor$Ba_b
Ba<-Ba_a*Ba+Ba_b

Pb<-dataset$Pb
Pb_a<-coefcor$Pb_a
Pb_b<-coefcor$Pb_b
Pb<-Pb_a*Pb+Pb_b
```

10.3 Compiling and exporting the processed data (_cor_data.csv)

source

Instructions following https://search.r-project.org/R/refmans/base/html/Round.html, https://www.statology.org/transpose-data-frame-in-r/ and https://datatofish.com/export-dataframe-to-csv-in-r/

As before, all newly calculated variables are combined with the sample number in a new table called **_cor_data.csv** (in our case **coefcorItoII_example_cor_data.csv**). All element values are rounded to whole digits. This table is then exported. It contains quantitative, processed data and forms the basis for applying the coefcor of the given time period (here coefcor II).

```
coefcorItoII_cor_data<-data.frame(Sample,Si,Ti,Al,Fe,Mn,Mg,Ca,
                                K,P,Cl,V,Cr,Ni,Zn,As,Rb,Sr,Y,Zr,Nb,Ba,Pb)

coefcorItoII_cor_data[, c("Si","Ti","Al","Fe","Mn","Mg","Ca","K","P","Cl","V",
                            "Cr","Ni","Zn","As","Rb","Sr","Y","Zr","Nb",
            "Ba","Pb")] <- round(coefcorItoII_cor_data[, c("Si","Ti","Al","Fe",
            "Mn","Mg","Ca","K","P","Cl","V","Cr","Ni","Zn","As","Rb","Sr",
            "Y","Zr","Nb","Ba","Pb")], 0)
```

```r
write.csv(coefcorItoII_cor_data,
          "../data_processed//coefcorItoII_example_cor_data.csv",row.names=TRUE)
```