Loading libraries

##Anscombe's quartet This data set illustrates how variables even with identical numerical statistical properties when plotted are very different, proving the necessity of also plotting data.

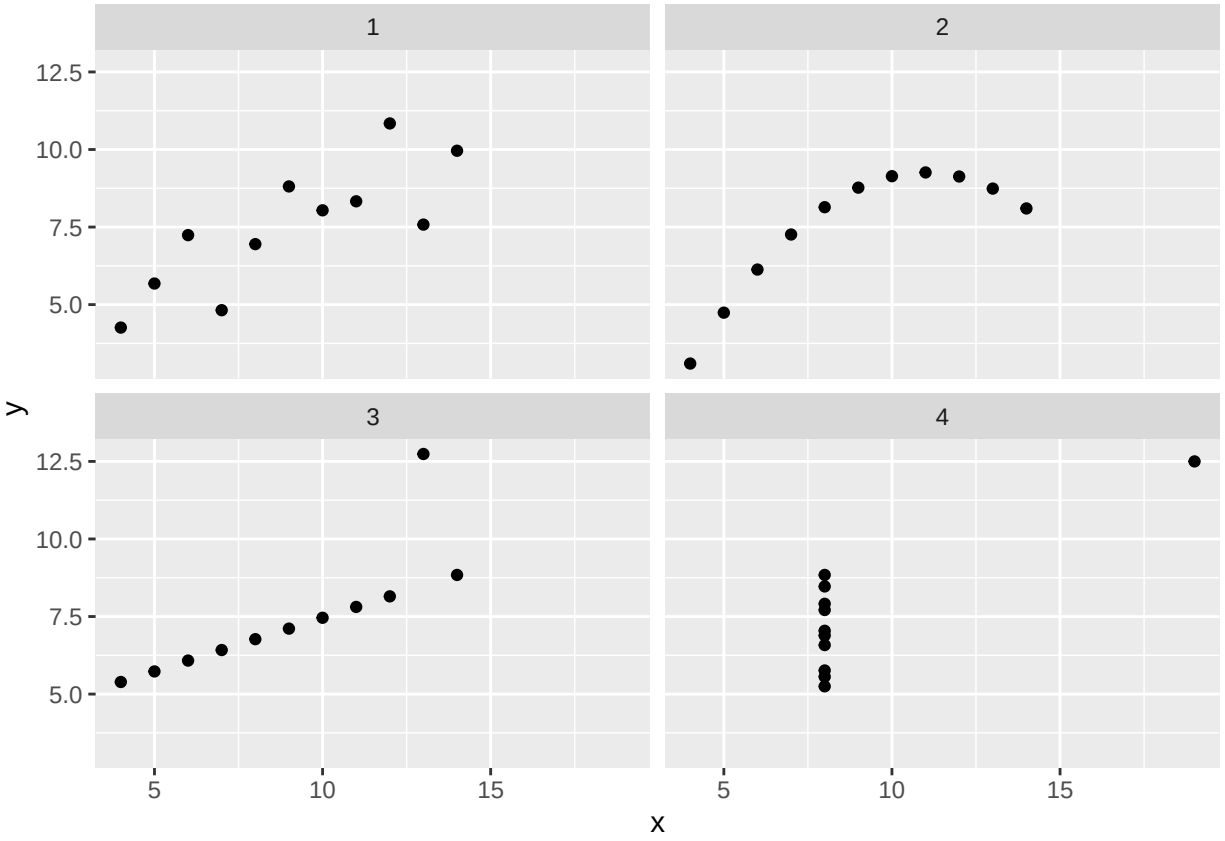
```
df <- anscombe
colnames(df)
```

```
## [1] "x1" "x2" "x3" "x4" "y1" "y2" "y3" "y4"
```

```
df_combined <- rbind(data.frame(x=df$x1, y=df$y1, set="1"),
                     data.frame(x=df$x2, y=df$y2, set="2"),
                     data.frame(x=df$x3, y=df$y3, set="3"),
                     data.frame(x=df$x4, y=df$y4, set="4"))
#groups columns into pairs so that they can be facceted, set column is a factor
```

Scatter plot facceted by set

```
df_combined %>% ggplot(aes(x, y)) + geom_point() + facet_wrap(~set)
```



Means and standard deviations by set

```
df_combined %>%
  group_by(set) %>%
  summarise(mean_x = mean(x), mean_y = mean(y), sd_x = sd(x), sd_y = sd(y))
```

```
## # A tibble: 4 x 5
##   set   mean_x mean_y  sd_x  sd_y
##   <fct>  <dbl>  <dbl> <dbl> <dbl>
## 1 1          9   7.50  3.32  2.03
## 2 2          9   7.50  3.32  2.03
```

```
## 3 3          9   7.5   3.32  2.03
## 4 4          9   7.50  3.32  2.03
```

Pearsons, Kendalls and Spearmans test results by set and their respective p-values

```
df_combined %>%
  group_by(set) %>%
  summarise(Pearson = cor(x,y, method = "pearson"), pValue_Pearson = cor.test(x,y, method = "pearson")$p
            Kendall = cor(x,y, method = "kendall"), pValue_Kendall = cor.test(x,y, method = "kendall")$p
            Spearman = cor(x,y, method = "spearman"), pValue_Spearman = cor.test(x,y, method = "spearma
            )
```

```
## Warning in cor.test.default(x, y, method = "kendall"): Cannot compute exact p-
## value with ties
```

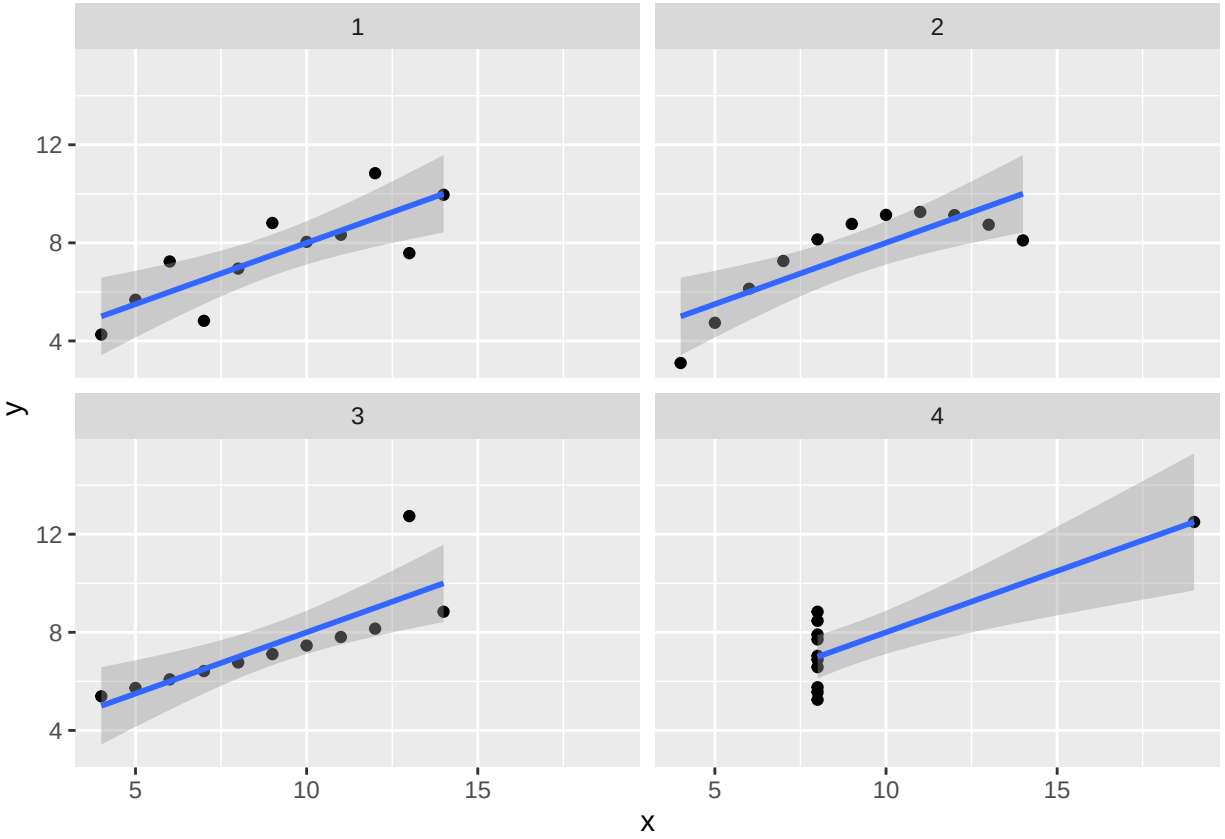
```
## Warning in cor.test.default(x, y, method = "spearman"): Cannot compute exact p-
## value with ties
```

```
## # A tibble: 4 x 7
##   set   Pearson pValue_Pearson Kendall pValue_Kendall Spearman pValue_Spearman
##   <fct>   <dbl>          <dbl>   <dbl>          <dbl>    <dbl>           <dbl>
## 1 1       0.816        0.00217   0.636    0.00571        0.818         0.00373
## 2 2       0.816        0.00218   0.564    0.0165         0.691         0.0231
## 3 3       0.816        0.00218   0.964    0.000000551    0.991         0
## 4 4       0.817        0.00216   0.426    0.114          0.5           0.117
```

Added geom smooth with method lm

```
df_combined %>% ggplot(aes(x, y)) + geom_point() + facet_wrap(~set) + geom_smooth(method = lm)
```

```
## `geom_smooth()` using formula 'y ~ x'
```



##Air quality data set Data Set Information:

The dataset contains 9358 instances of hourly averaged responses from an array of 5 metal oxide chemical sensors embedded in an Air Quality Chemical Multisensor Device. The device was located on the field in a significantly polluted area, at road level,within an Italian city. Data were recorded from March 2004 to February 2005 (one year)representing the longest freely available recordings of on field deployed air quality chemical sensor devices responses. Ground Truth hourly averaged concentrations for CO, Non Metanic Hydrocarbons, Benzene, Total Nitrogen Oxides (NOx) and Nitrogen Dioxide (NO2) and were provided by a co-located reference certified analyzer. Evidences of cross-sensitivities as well as both concept and sensor drifts are present as described in De Vito et al., Sens. And Act. B, Vol. 129,2,2008 (citation required) eventually affecting sensors concentration estimation capabilities. Missing values are tagged with -200 value. This dataset can be used exclusively for research purposes. Commercial purposes are fully excluded.

Attribute Information:

0 Date (DD/MM/YYYY) 1 Time (HH.MM.SS) 2 True hourly averaged concentration CO in mg/mˆ3 (reference analyzer) 3 PT08.S1 (tin oxide) hourly averaged sensor response (nominally CO targeted) 4 True hourly averaged overall Non Metanic HydroCarbons concentration in microg/mˆ3 (reference analyzer) 5 True hourly averaged Benzene concentration in microg/mˆ3 (reference analyzer) 6 PT08.S2 (titania) hourly averaged sensor response (nominally NMHC targeted) 7 True hourly averaged NOx concentration in ppb (reference analyzer) 8 PT08.S3 (tungsten oxide) hourly averaged sensor response (nominally NOx targeted) 9 True hourly averaged NO2 concentration in microg/mˆ3 (reference analyzer) 10 PT08.S4 (tungsten oxide) hourly averaged sensor response (nominally NO2 targeted) 11 PT08.S5 (indium oxide) hourly averaged sensor response (nominally O3 targeted) 12 Temperature in Â°C 13 Relative Humidity (%) 14 AH Absolute Humidity

Loading csv file

```
AQ_raw <- read.csv("/home/aleksandar/Desktop/R statistics 2/hw_1_Rstat2/AirQualityUCI/AirQualityUCI.csv
```

Removing NAs Since missing values are denoted with -200 in this data set, first step would be to change all -200 values to NA and then remove NAs

```
AQ_NAs <- AQ_raw %>% na_if(-200) %>% na_if("-200,0")
#removing empty columns
toDrop <- c("X", "X.1")
AQ_NAs <- AQ_NAs[,!(names(AQ_NAs)) %in% toDrop]
```

Exploring NAs, after visually checking data set it seems most NAs are from NMHC.GT column

```
sapply(AQ_NAs, function(x) sum(is.na(x)))
```

```
##          Date          Time       CO.GT.   PT08.S1.CO.      NMHC.GT.
##             0             0         1683          480          8557
##      C6H6.GT. PT08.S2.NMHC.       NOx.GT.  PT08.S3.NOx.       NO2.GT.
##           366           480         1753          480          1756
##  PT08.S4.NO2.   PT08.S5.O3.            T           RH           AH
##           480           480          366          366          366
```

Since NMHC.GT has a lot of NAs using complete cases would drop about 90% of cases. I think its a good option to remove NMHC.GT from the main analyses.

```
toDrop <- c("NMHC.GT.", "Date", "Time") #also dropped Date and Time colums since I wont be using them
AQ_NAs <- AQ_NAs[,!(names(AQ_NAs)) %in% toDrop]
complete_AQrows <- complete.cases(AQ_NAs)
AQ <- AQ_NAs[complete_AQrows,]
```

Checking for non numeric variables

```
sapply(AQ, function(x)is.numeric(x))
```

```
##         CO.GT.    PT08.S1.CO.       C6H6.GT. PT08.S2.NMHC.        NOx.GT.
##          FALSE           TRUE          FALSE           TRUE           TRUE
##  PT08.S3.NOx.        NO2.GT.  PT08.S4.NO2.    PT08.S5.O3.              T
##           TRUE           TRUE           TRUE           TRUE          FALSE
##             RH             AH
##          FALSE          FALSE
```

```
AQnum <- data.frame(sapply(AQ, function(x) as.double(gsub(",",".",x)))) #converting to double

sapply(AQnum, function(x)is.numeric(x))
```

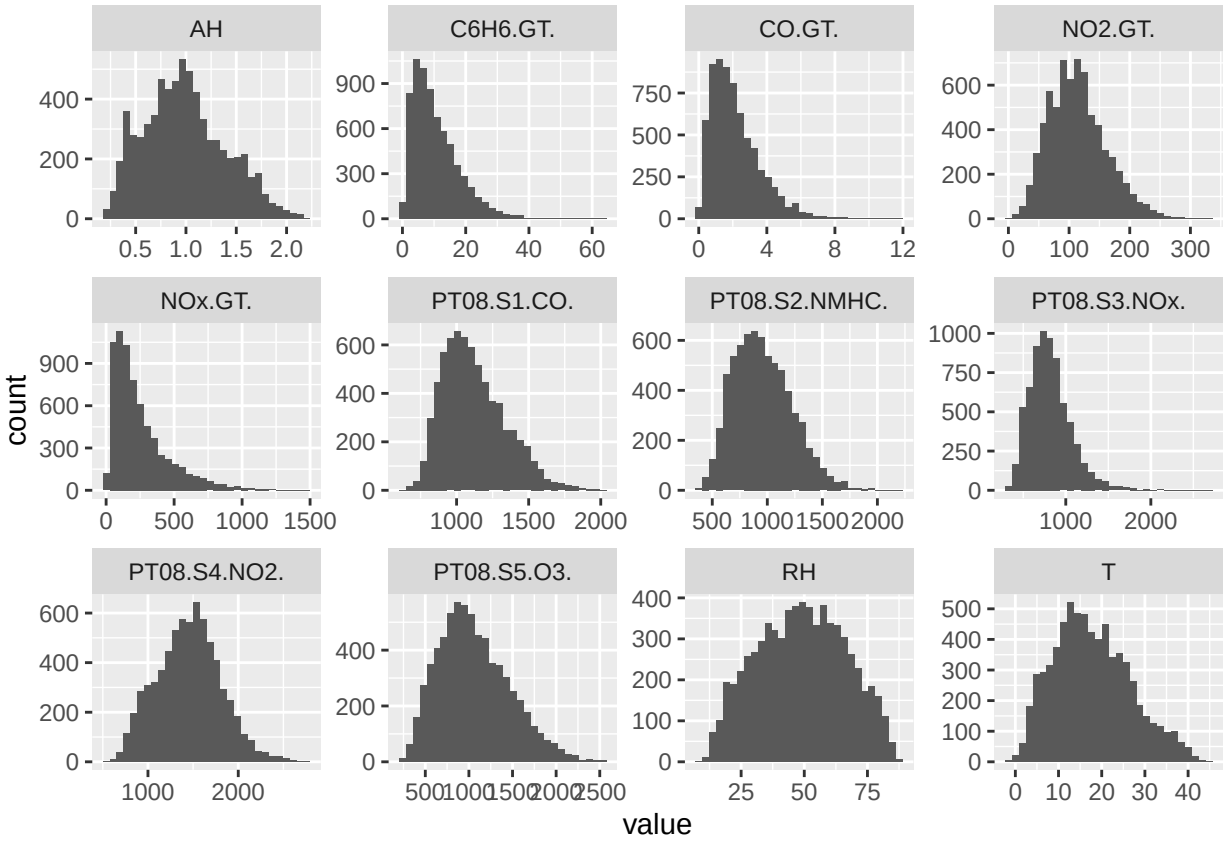
```
##         CO.GT.    PT08.S1.CO.       C6H6.GT. PT08.S2.NMHC.        NOx.GT.
##           TRUE           TRUE           TRUE           TRUE           TRUE
##  PT08.S3.NOx.        NO2.GT.  PT08.S4.NO2.    PT08.S5.O3.              T
##           TRUE           TRUE           TRUE           TRUE           TRUE
##             RH             AH
##           TRUE           TRUE
```

Exploring the data individually

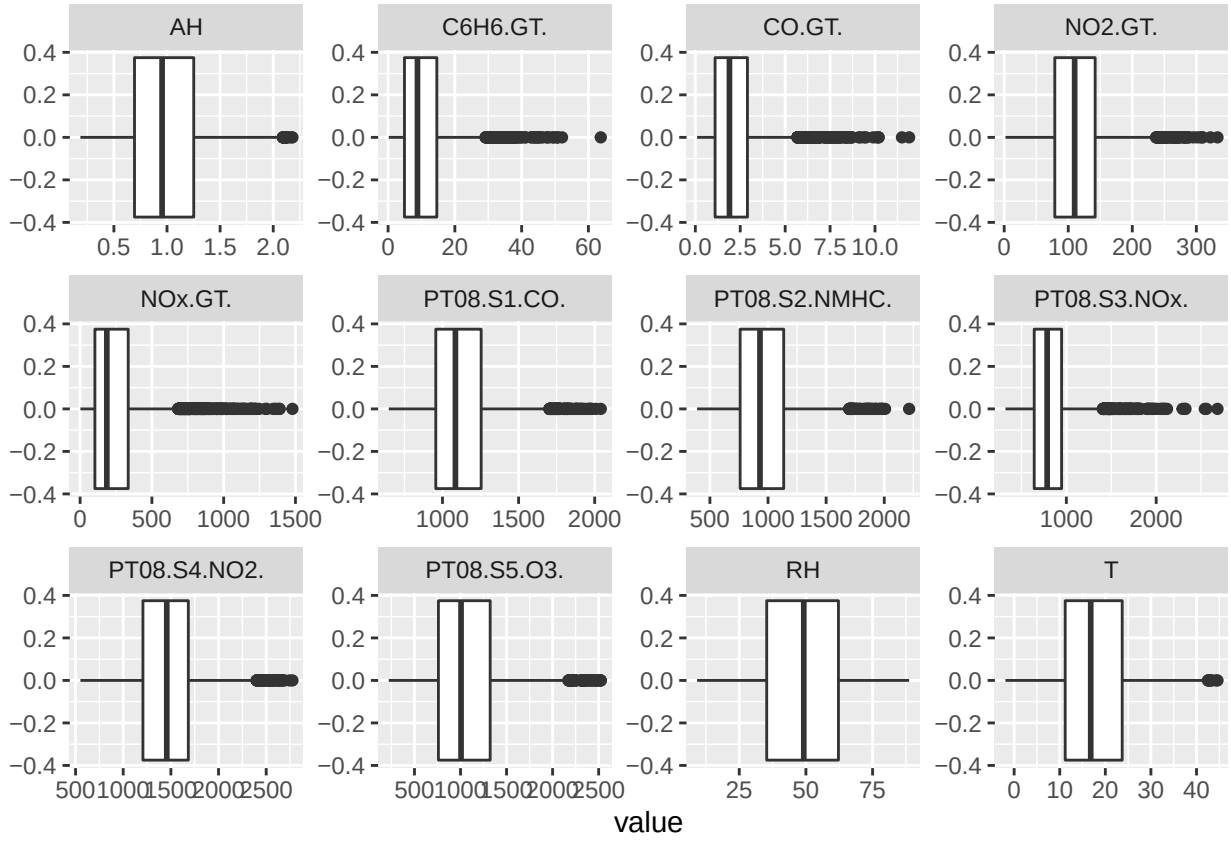
```
AQlong <- gather(AQnum)

ggplot(AQlong, aes(value)) + geom_histogram() + facet_wrap(~key, scales = "free")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



```
ggplot(AQlong, aes(value)) + geom_boxplot() + facet_wrap(~key, scales = "free")
```



Finding pairwise correlations for data set Response for further analyses is C6H6.GT.

```
sapply(AQnum, function(x) sapply(AQnum, function(y) cor(x,y)))
```

```
##                    CO.GT. PT08.S1.CO.    C6H6.GT. PT08.S2.NMHC.    NOx.GT.
## CO.GT.          1.00000000  0.87701378  0.93000844    0.91431022  0.7864556
## PT08.S1.CO.     0.87701378  1.00000000  0.87743017    0.88606831  0.7077047
## C6H6.GT.        0.93000844  0.87743017  1.00000000    0.98270456  0.7183438
## PT08.S2.NMHC.   0.91431022  0.88606831  0.98270456    1.00000000  0.7053592
## NOx.GT.         0.78645556  0.70770473  0.71834383    0.70535922  1.0000000
## PT08.S3.NOx.   -0.70103781 -0.76289451 -0.72572179   -0.78163029 -0.6621663
## NO2.GT.         0.67383957  0.62826276  0.60324096    0.63330954  0.7570291
## PT08.S4.NO2.    0.63083404  0.67590960  0.76180525    0.77428754  0.2337926
## PT08.S5.O3.     0.85348004  0.89716636  0.86115359    0.87677728  0.7885502
## T               0.01833425  0.02827677  0.18900256    0.22833308 -0.2759984
## RH              0.06475315  0.16923408 -0.02159153   -0.04608374  0.2322552
## AH              0.05934617  0.14975239  0.18707157    0.20559028 -0.1441860
##                PT08.S3.NOx.    NO2.GT. PT08.S4.NO2. PT08.S5.O3.           T
## CO.GT.         -0.70103781  0.6738396  0.630834042  0.85348004  0.01833425
## PT08.S1.CO.    -0.76289451  0.6282628  0.675909601  0.89716636  0.02827677
## C6H6.GT.       -0.72572179  0.6032410  0.761805247  0.86115359  0.18900256
## PT08.S2.NMHC.  -0.78163029  0.6333095  0.774287538  0.87677728  0.22833308
## NOx.GT.        -0.66216631  0.7570291  0.233792602  0.78855025 -0.27599835
## PT08.S3.NOx.    1.00000000 -0.6413769 -0.511223442 -0.79336411 -0.09949483
## NO2.GT.        -0.64137694  1.0000000  0.142612010  0.70252411 -0.21432470
## PT08.S4.NO2.   -0.51122344  0.1426120  1.000000000  0.57424200  0.56658556
```

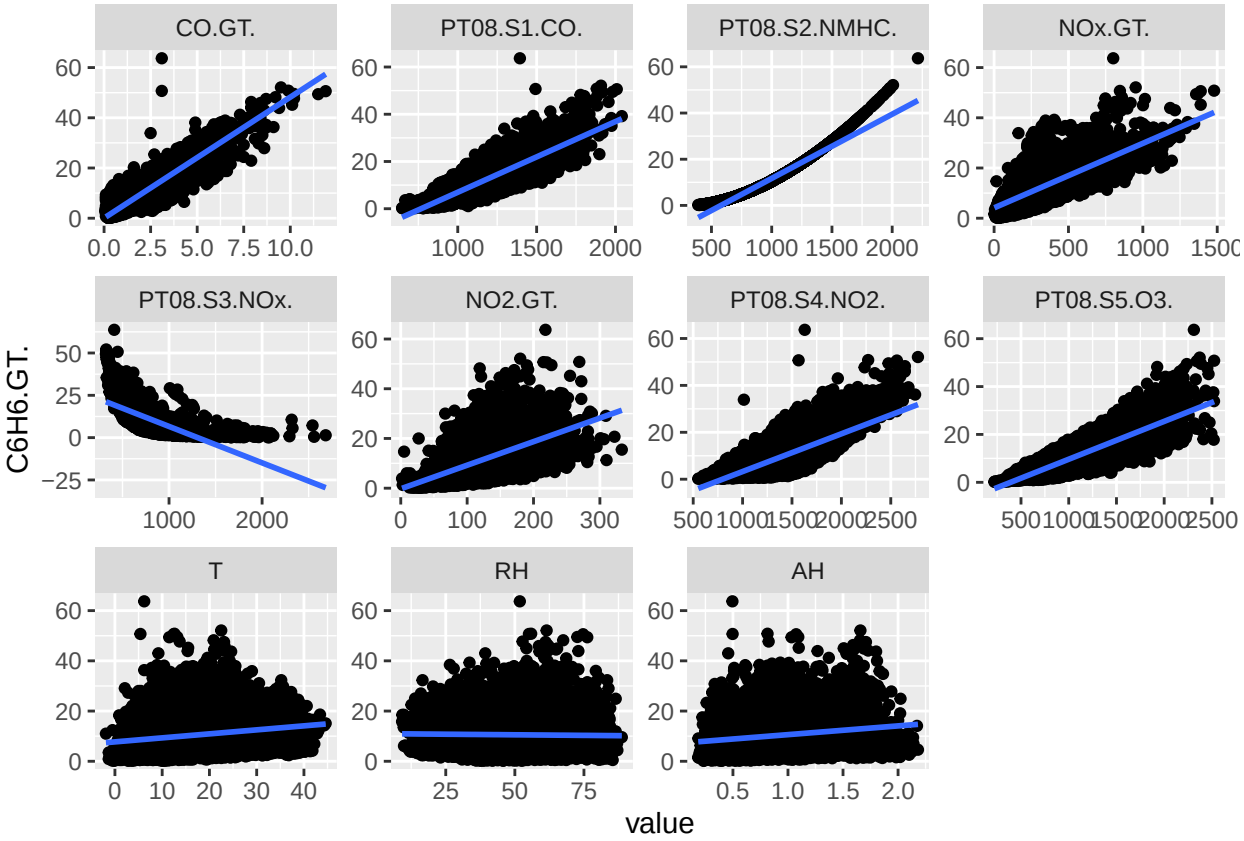
```
## PT08.S5.O3.     -0.79336411  0.7025241  0.574242002  1.00000000 -0.04614601
## T               -0.09949483 -0.2143247  0.566585555 -0.04614601  1.00000000
## RH              -0.11647947 -0.0753329 -0.009160077  0.16482121 -0.56390892
## AH              -0.22338099 -0.3496460  0.646390126  0.07580693  0.66063806
##                          RH          AH
## CO.GT.          0.064753147  0.05934617
## PT08.S1.CO.     0.169234080  0.14975239
## C6H6.GT.       -0.021591530  0.18707157
## PT08.S2.NMHC. -0.046083742  0.20559028
## NOx.GT.         0.232255190 -0.14418603
## PT08.S3.NOx.   -0.116479467 -0.22338099
## NO2.GT.        -0.075332896 -0.34964601
## PT08.S4.NO2.   -0.009160077  0.64639013
## PT08.S5.O3.     0.164821208  0.07580693
## T              -0.563908917  0.66063806
## RH              1.000000000  0.17957592
## AH              0.179575919  1.00000000
```

```
#pairs(AQnum, )
```

##Visualising and inspecting some pairs of columns more closely

```
d <- melt(AQnum, id.vars = "C6H6.GT.")
ggplot(d, aes(x = value, y = C6H6.GT.)) + geom_point() + geom_smooth(method = lm) + facet_wrap(~variable
```

```
## `geom_smooth()` using formula 'y ~ x'
```



All models with response C6H6

```r
depVarList = setdiff(colnames(AQnum), c("C6H6.GT."))

allModels = lapply(depVarList, function(x){
  lm(formula= paste0("C6H6.GT. ~ `", x, "`"), data= AQnum)})

names(allModels) = depVarList
print(lapply(allModels, function(x) summary(x)))
```

```
## $CO.GT.
##
## Call:
## lm(formula = paste0("C6H6.GT. ~ `", x, "`"), data = AQnum)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -15.198  -1.585  -0.158   1.497  48.725
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.04054    0.05977   0.678    0.498
## CO.GT.       4.81746    0.02286 210.782   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.744 on 6939 degrees of freedom
## Multiple R-squared:  0.8649, Adjusted R-squared:  0.8649
## F-statistic: 4.443e+04 on 1 and 6939 DF,  p-value: < 2.2e-16
##
##
## $PT08.S1.CO.
##
## Call:
## lm(formula = paste0("C6H6.GT. ~ `", x, "`"), data = AQnum)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -11.955  -2.260  -0.183   1.955  44.938
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -2.298e+01  2.243e-01  -102.5   <2e-16 ***
## PT08.S1.CO.  2.995e-02  1.965e-04   152.4   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.581 on 6939 degrees of freedom
## Multiple R-squared:  0.7699, Adjusted R-squared:  0.7699
## F-statistic: 2.322e+04 on 1 and 6939 DF,  p-value: < 2.2e-16
##
##
## $PT08.S2.NMHC.
##
## Call:
## lm(formula = paste0("C6H6.GT. ~ `", x, "`"), data = AQnum)
```

```
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.1674 -0.9614 -0.4998  0.5063 18.2658
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)   -1.608e+01  6.249e-02  -257.3   <2e-16 ***
## PT08.S2.NMHC.  2.778e-02  6.285e-05   442.1   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.383 on 6939 degrees of freedom
## Multiple R-squared:  0.9657, Adjusted R-squared:  0.9657
## F-statistic: 1.954e+05 on 1 and 6939 DF,  p-value: < 2.2e-16
##
##
## $NOx.GT.
##
## Call:
## lm(formula = paste0("C6H6.GT. ~ `", x, "`"), data = AQnum)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -12.702  -3.973  -1.113   2.790  38.999
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 4.1106722  0.0974625   42.18   <2e-16 ***
## NOx.GT.     0.0257062  0.0002989   86.01   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5.194 on 6939 degrees of freedom
## Multiple R-squared:  0.516,  Adjusted R-squared:  0.5159
## F-statistic:  7398 on 1 and 6939 DF,  p-value: < 2.2e-16
##
##
## $PT08.S3.NOx.
##
## Call:
## lm(formula = paste0("C6H6.GT. ~ `", x, "`"), data = AQnum)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -9.071 -3.698 -0.810  2.306 44.416
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)  28.1238172  0.2092372  134.41   <2e-16 ***
## PT08.S3.NOx. -0.0215075  0.0002448  -87.87   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 5.136 on 6939 degrees of freedom
## Multiple R-squared:  0.5267, Adjusted R-squared:  0.5266
## F-statistic:  7721 on 1 and 6939 DF,  p-value: < 2.2e-16
##
##
## $NO2.GT.
##
## Call:
## lm(formula = paste0("C6H6.GT. ~ `", x, "`"), data = AQnum)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -17.858  -3.572  -0.613   2.585  43.268
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.247248   0.185743  -1.331    0.183
## NO2.GT.      0.094857   0.001506  63.005   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5.954 on 6939 degrees of freedom
## Multiple R-squared:  0.3639, Adjusted R-squared:  0.3638
## F-statistic:  3970 on 1 and 6939 DF,  p-value: < 2.2e-16
##
##
## $PT08.S4.NO2.
##
## Call:
## lm(formula = paste0("C6H6.GT. ~ `", x, "`"), data = AQnum)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -9.941 -3.579 -0.460  2.919 50.307
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -1.283e+01  2.457e-01  -52.22   <2e-16 ***
## PT08.S4.NO2.  1.610e-02  1.643e-04   97.96   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.836 on 6939 degrees of freedom
## Multiple R-squared:  0.5803, Adjusted R-squared:  0.5803
## F-statistic:  9596 on 1 and 6939 DF,  p-value: < 2.2e-16
##
##
## $PT08.S5.O3.
##
## Call:
## lm(formula = paste0("C6H6.GT. ~ `", x, "`"), data = AQnum)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
```

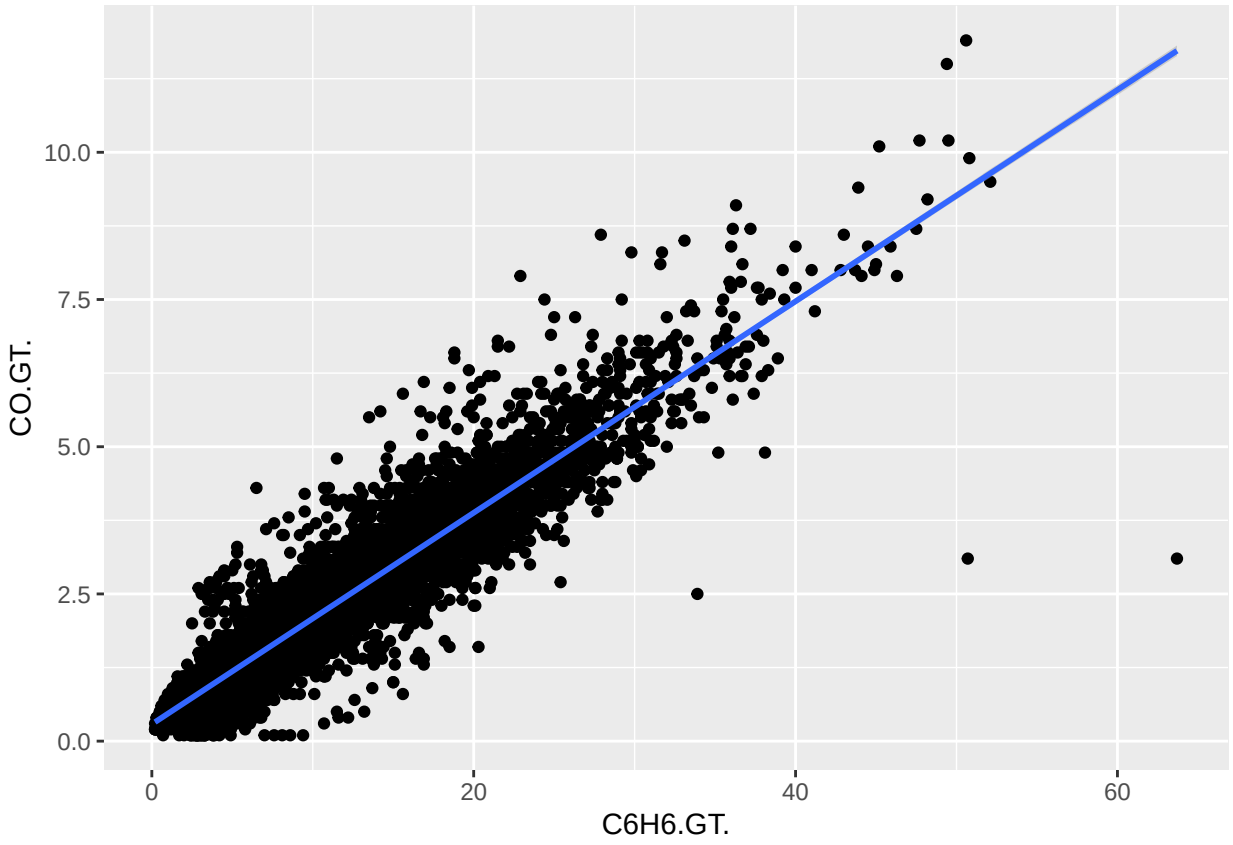
```
## -15.863  -2.333  -0.038   2.112  33.326
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -6.1732848  0.1269922  -48.61   <2e-16 ***
## PT08.S5.O3.  0.0158144  0.0001121  141.11   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.795 on 6939 degrees of freedom
## Multiple R-squared:  0.7416, Adjusted R-squared:  0.7415
## F-statistic: 1.991e+04 on 1 and 6939 DF,  p-value: < 2.2e-16
##
##
## $T
##
## Call:
## lm(formula = paste0("C6H6.GT. ~ `", x, "`"), data = AQnum)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -11.193  -5.400  -1.790   3.677  54.989
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  7.72215    0.19736   39.13   <2e-16 ***
## T            0.15952    0.00995   16.03   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.331 on 6939 degrees of freedom
## Multiple R-squared:  0.03572,    Adjusted R-squared:  0.03558
## F-statistic: 257.1 on 1 and 6939 DF,  p-value: < 2.2e-16
##
##
## $RH
##
## Call:
## lm(formula = paste0("C6H6.GT. ~ `", x, "`"), data = AQnum)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -10.460  -5.624  -1.763   3.973  53.172
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 11.006445   0.266723  41.266   <2e-16 ***
## RH          -0.009246   0.005139  -1.799   0.0721 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.464 on 6939 degrees of freedom
## Multiple R-squared:  0.0004662,  Adjusted R-squared:  0.0003221
## F-statistic: 3.236 on 1 and 6939 DF,  p-value: 0.07206
```

```
##
##
## $AH
##
## Call:
## lm(formula = paste0("C6H6.GT. ~ `", x, "`"), data = AQnum)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -12.451  -5.553  -1.702   3.861  54.853
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   7.1229     0.2335   30.50   <2e-16 ***
## AH            3.4818     0.2195   15.86   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.334 on 6939 degrees of freedom
## Multiple R-squared:  0.035,  Adjusted R-squared:  0.03486
## F-statistic: 251.6 on 1 and 6939 DF,  p-value: < 2.2e-16
```

I decided to look into CO.GT as the most suitable predictor for C6H6

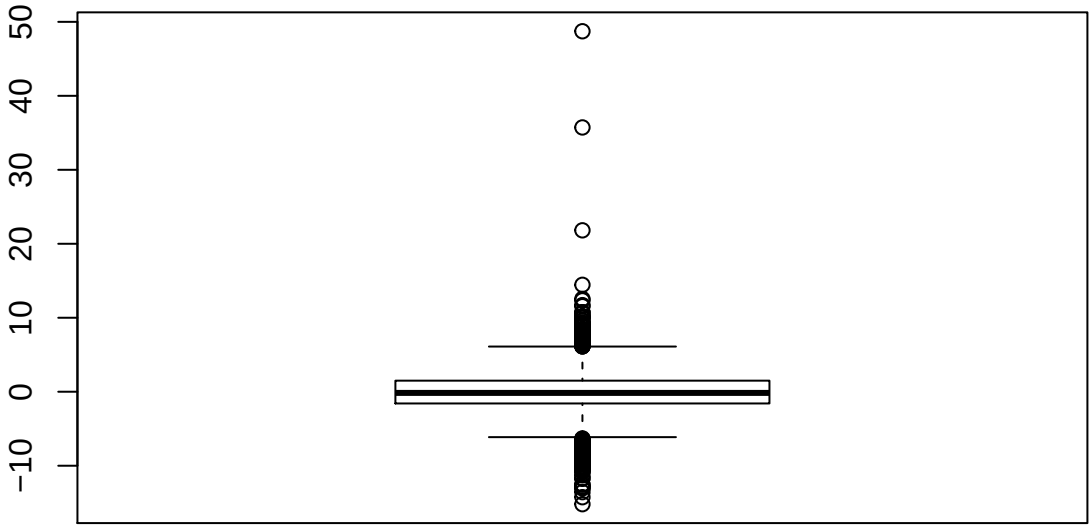
```
ggplot(AQnum, aes( C6H6.GT., CO.GT.) ) + geom_point() + geom_smooth(method = lm)
```

```
## `geom_smooth()` using formula 'y ~ x'
```



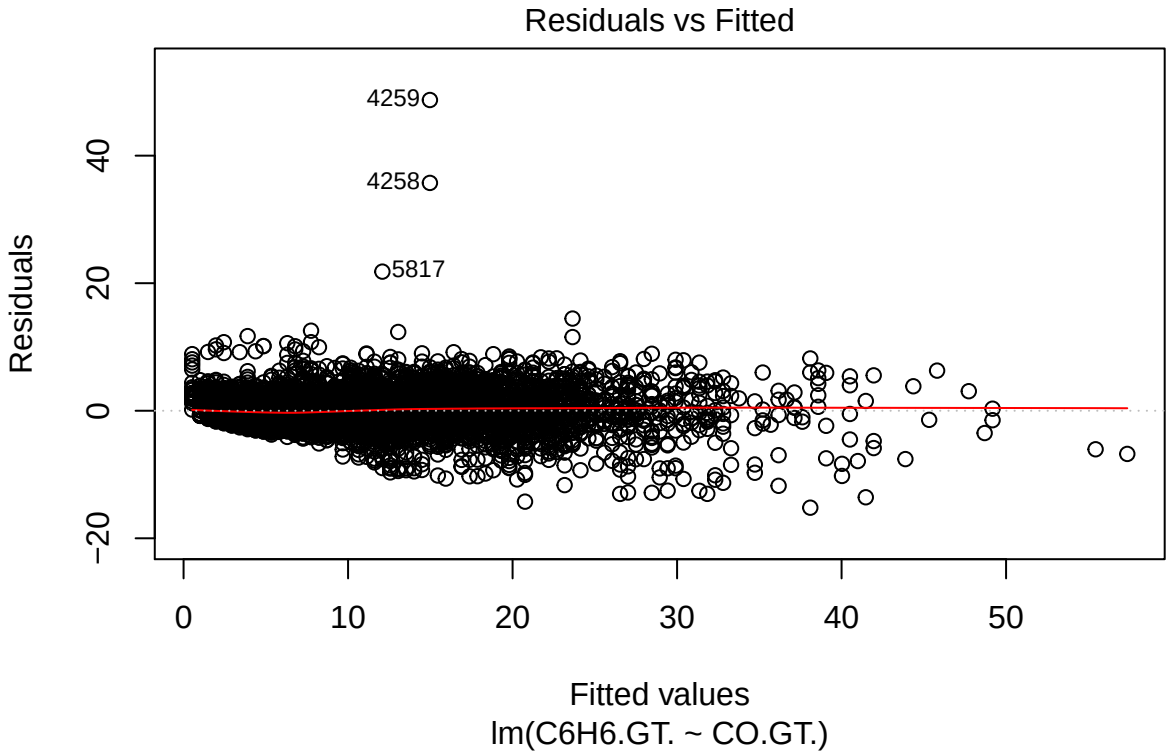
Residuals are symetrical and mean is around zero, this means

```r
modCO <-lm(formula = C6H6.GT. ~ CO.GT., data = AQnum)
boxplot(modCO$residuals)
```



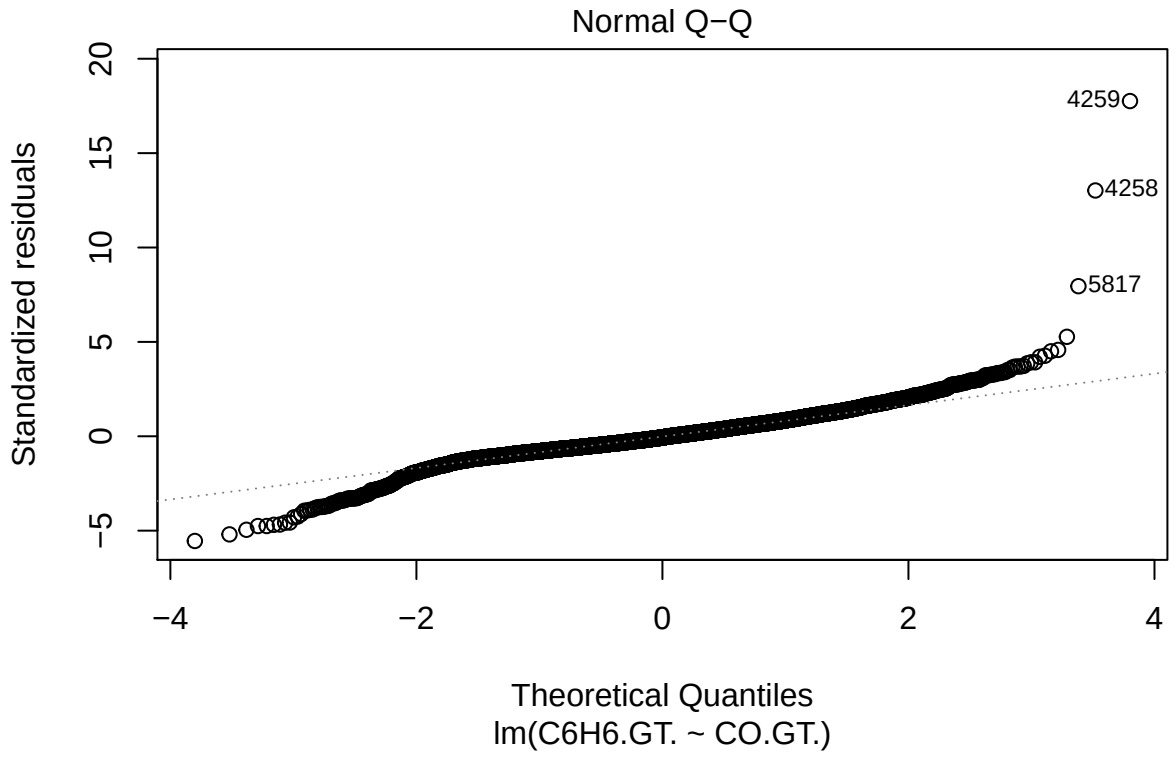
Residuals vs fitted - how well do fitted value align to residuals - it means residuals are euqally spread around the data

```r
plot(modCO, which = 1)
```



QQplot - are theoretical quantiles the same as actual ones - if they are on the line distribution is close to normal

```r
plot(modCO, which = 2)
```



Perfoming train vs test data LM analysis

```r
set.seed(42)
sample <- sample.int(n = nrow(AQnum), size = floor(0.75 * nrow(AQnum)))
train <- AQnum[sample,]
test <- AQnum[-sample,]

new_modCO <- lm(data = train, formula = C6H6.GT. ~ CO.GT.)
summary(new_modCO)
```

```
##
## Call:
## lm(formula = C6H6.GT. ~ CO.GT., data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -14.194  -1.592  -0.178   1.505  35.758
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.08072    0.06845   1.179    0.238
## CO.GT.       4.79383    0.02625 182.621   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.722 on 5203 degrees of freedom
## Multiple R-squared:  0.865,  Adjusted R-squared:  0.865
## F-statistic: 3.335e+04 on 1 and 5203 DF,  p-value: < 2.2e-16
```

```
ggplot(data = train,aes( C6H6.GT., CO.GT.) ) + geom_point() + geom_smooth(method = lm)
```

```
## `geom_smooth()` using formula 'y ~ x'
```

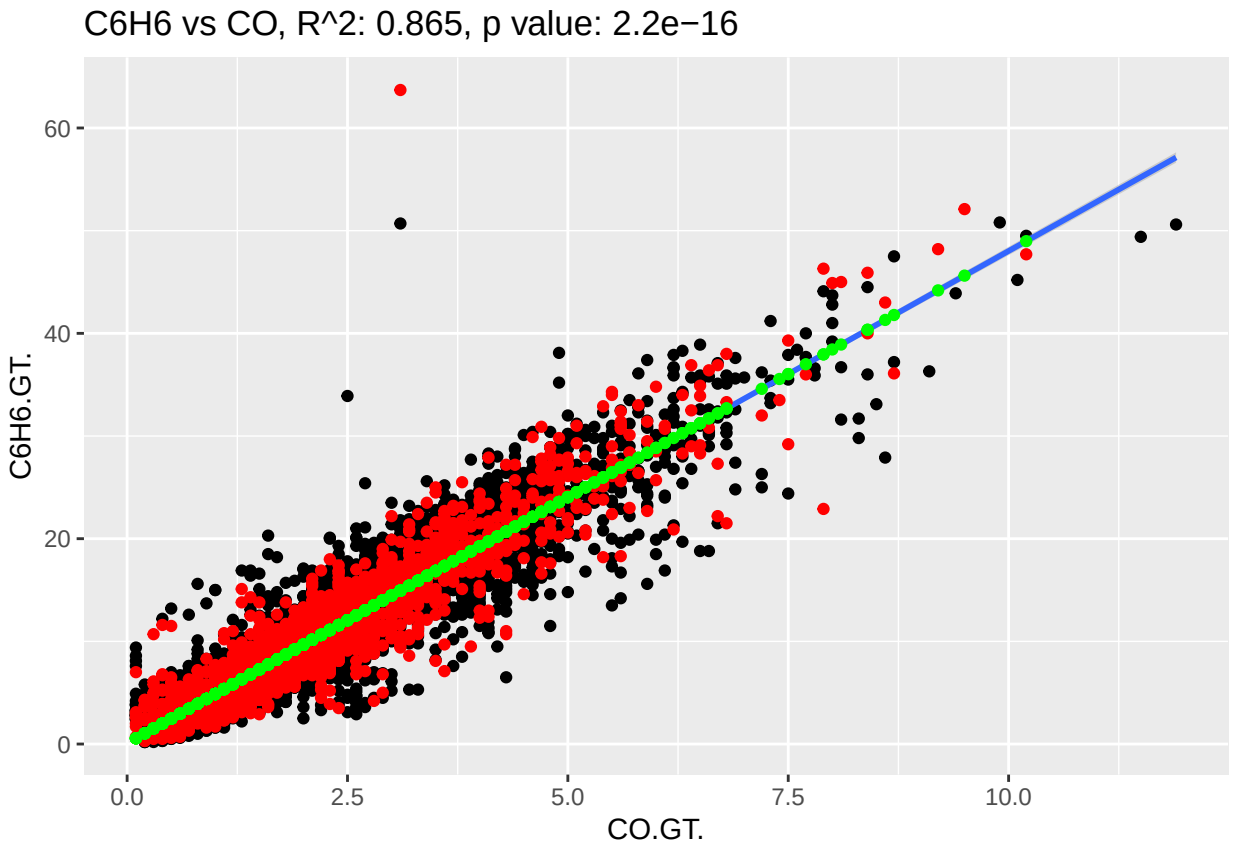
Prediction using trained model

```
pred <- predict(new_modCO, newdata = test)
test$CO.GT._predicted <- pred
```

Constructing plot

```
ggplot() + geom_point(data = train, aes(x= CO.GT., y = C6H6.GT.)) + geom_smooth(data =train, aes(x=CO.G
```

```
## `geom_smooth()` using formula 'y ~ x'
```


C6H6 vs CO, R^2: 0.865, p value: 2.2e−16
C6H6.GT.
CO.GT.
0
20
40
60
0.0
2.5
5.0
7.5
10.0