Statistical analysis of an algorithm performances

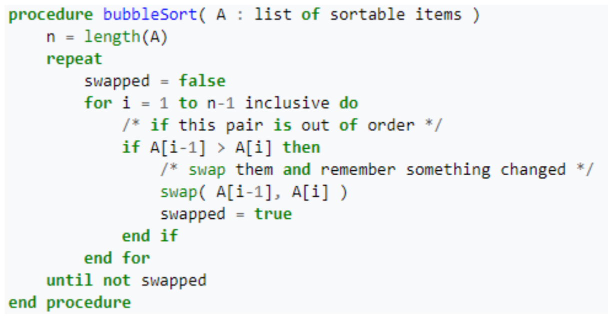
1st FULCO GIAMMARIO
MAT. 191071

***Abstract*—This report provides a statistical analysis on a factorial experiment made to test the execution time of a sorting algorithm, in particular the BubbleSort algorithm, performing on the same vectors under different conditions. The statistical analysis and the results reported are made with RStudio.**

I. EXPERIMENT DESCRIPTION

A. Introduction

The experiment studied consists on putting the algorithm under several different conditions measuring the time elapsed for the sorting of the algorithm. The algorithm used is a standard BubbleSort algorithm with an average computational complexity of $O(n^2)$ and, for completeness of description, its score is depicted on the figure below. Note that Bubblesort is not the best performing algorithm to sort vectors of numbers but it is probably the most famous one.



```
procedure bubbleSort( A : list of sortable items )
    n = length(A)
    repeat
        swapped = false
        for i = 1 to n-1 inclusive do
            /* if this pair is out of order */
            if A[i-1] > A[i] then
                /* swap them and remember something changed */
                swap( A[i-1], A[i] )
                swapped = true
            end if
        end for
    until not swapped
end procedure
```

Figure 1: *BubbleSort Pseudocode*

B. Factorial Design

In terms of factorial design, the experiment must be evaluated under certain conditions: indeed it's necessary to find the factors and their levels; speak of these, the experiment has been performed as a 2^5 factorial experiment, meaning there are five factors and each of them has exactly two levels (considered high = 1 and low = 0). Let's now explain and put the factors into the experiment context:

Factor A: The CPU on which the algorithm is performed; it varies between I7-6500U @ 2.5Ghz (-1) and I5-6400 @ 2.7Ghz (1).

Factor B: The programming language in which the algorithm is performed, meaning that it can be C++(-1) or Matlab Programming Language(1).

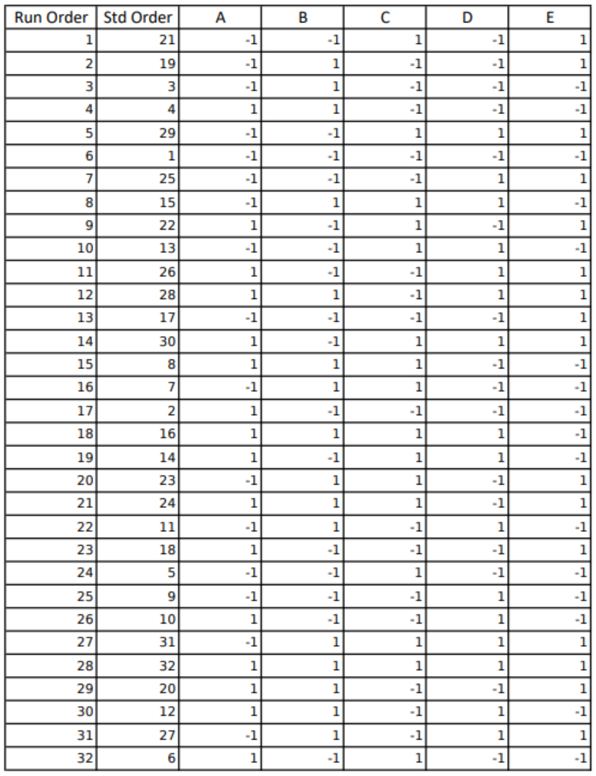
Factor C: It describes the lenght of the vector that has to be sorted; so this factor can oscillates between a Small vector of 1k elements (-1) and a Huge vector of 100k elements (1).

Factor D: It specifies how ”big” are the elements of the array; it can be small numbers from 0 to 100 (-1) or huge numbers $\geq$ 1 million (1).

Factor E: The data type of the elements in the array; two levels has been chosen, so they are integers (-1) and double precision floating points (1).

C. Design Matrix

As previously described, the algorithm is played just once (meaning no repetitions were made and producing a total value of 32 experiments) on random vectors under the variations of these combining factors while the elapsed time of the algorithm has been recorded and stored as the measure of the experiment. So in this way a design matrix of the experiment has been constructed, making sure to randomize the sequential sorting of each execution in order to minimize the external uncontrollable and undesired effects.



Run Order	Std Order	A	B	C	D	E
1	21	-1	-1	1	-1	1
2	19	-1	1	-1	-1	1
3	3	-1	1	-1	-1	-1
4	4	1	1	-1	-1	-1
5	29	-1	-1	1	1	1
6	1	-1	-1	-1	-1	-1
7	25	-1	-1	-1	1	1
8	15	-1	1	1	1	-1
9	22	1	-1	1	-1	1
10	13	-1	-1	1	1	-1
11	26	1	-1	-1	1	1
12	28	1	1	-1	1	1
13	17	-1	-1	-1	-1	1
14	30	1	-1	1	1	1
15	8	1	1	1	-1	-1
16	7	-1	1	1	-1	-1
17	2	1	-1	-1	-1	-1
18	16	1	1	1	1	-1
19	14	1	-1	1	1	-1
20	23	-1	1	1	-1	1
21	24	1	1	1	-1	1
22	11	-1	1	-1	1	-1
23	18	1	-1	-1	-1	1
24	5	-1	-1	1	-1	-1
25	9	-1	-1	-1	1	-1
26	10	1	-1	-1	1	-1
27	31	-1	1	1	1	1
28	32	1	1	1	1	1
29	20	1	1	-1	-1	1
30	12	1	1	-1	1	-1
31	27	-1	1	-1	1	1
32	6	1	-1	1	-1	-1

Figure 2: *Run Order*

II. Statistical Analysis

After the collection of the data and the actual performance of the experiment it is possibile to proceed with the statistical analysis using RStudio.

A. *Daniel's Method*

Since the used factorial design didn't comprehend repetitions (unreplicated FP) of every treatment combination, it's not possible to proceed immediately with the ANOVA analysis, considering that the system variability can not be assessed; but by studying the effects of the model, it is possible to neglect the effects and, by so, the factors that barely don't influence the behaviour of the system, using the data collected for the treatments that involved those effects as repetitions for the others (Daniel's method). So, considering the first linear model as:

lm Elapsed : ~A*B*C*D*E

It is possible to build the effects Quantile-Quantile plot and then study it.

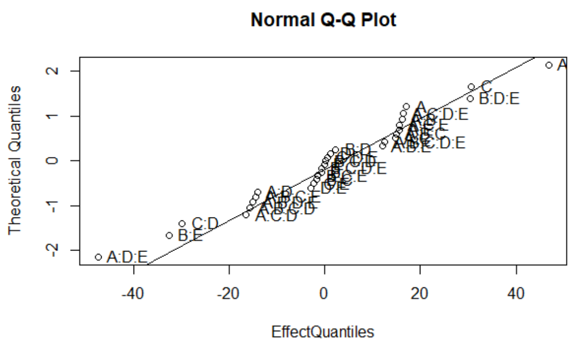


Figure 3: *Normal Quantile Plot*

As it can be seen from this plot, the E factor is closer to the line with respect to the others, as for every treatment in which this factor is involved, meaning that this factor can be taken out of the model without having a loss of information.

B. *First linear model analysis*

Let's now consider the new linear model, excluding the E factor:

lm2 : Elapsed ~A*B*C*D

Since Daniel's method has been applied, it's possible now to perform the ANOVA analysis, but first we have to check the model adequacy to verify if it has to be changed considering that it could be not adequate:

The Shapiro-Wilk test is pretty self-explainatory since the p-value < 0.05, that is the condition of rejecting the Null hypotesis of normality of the distribution of the residuals; so the model isn't adequate, even the histogram shows a non bell-shaped behaviour. Moreover, the Shapiro-Wilk test is confirmed by the Q-Q plot, since the positions of the residuals are not on the line of normality. Let's perform ANOVA

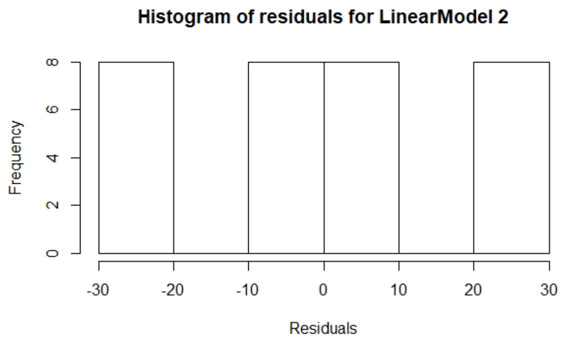


Figure 4: *Histogram of model 2*

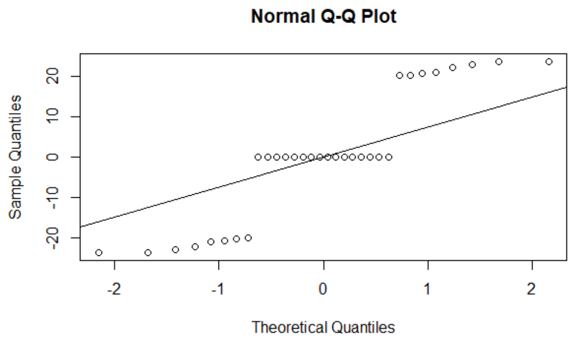
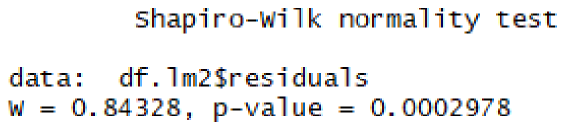


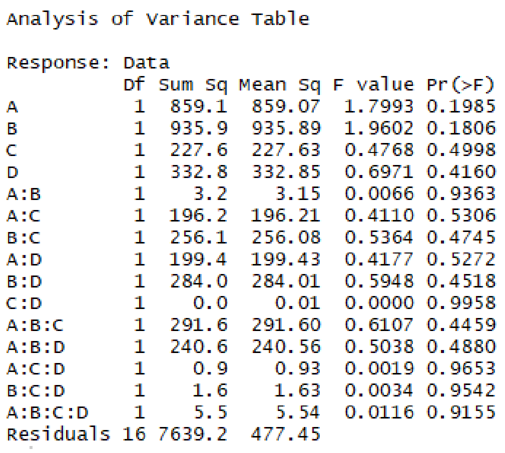
Figure 5: *Normal Quantile Plot of model 2*



```
        Shapiro-Wilk normality test

data:  df.lm2$residuals
W = 0.84328, p-value = 0.0002978
```

Figure 6: *Shapiro test of model 2*

analysis to see how can we subsequently improve this situation with another model:



```
Analysis of Variance Table

Response: Data
          Df Sum Sq Mean Sq F value Pr(>F)
A          1  859.1  859.07  1.7993 0.1985
B          1  935.9  935.89  1.9602 0.1806
C          1  227.6  227.63  0.4768 0.4998
D          1  332.8  332.85  0.6971 0.4160
A:B        1    3.2    3.15  0.0066 0.9363
A:C        1  196.2  196.21  0.4110 0.5306
B:C        1  256.1  256.08  0.5364 0.4745
A:D        1  199.4  199.43  0.4177 0.5272
B:D        1  284.0  284.01  0.5948 0.4518
C:D        1    0.0    0.01  0.0000 0.9958
A:B:C      1  291.6  291.60  0.6107 0.4459
A:B:D      1  240.6  240.56  0.5038 0.4880
A:C:D      1    0.9    0.93  0.0019 0.9653
B:C:D      1    1.6    1.63  0.0034 0.9542
A:B:C:D    1    5.5    5.54  0.0116 0.9155
Residuals 16 7639.2  477.45
```

Figure 7: *Anova test of model 2*

From this ANOVA Test we can assert that the combinations of the factors C with others are not significant to the results of the experiment so it is possible to find a better model excluding those combinations.

C. Second linear model analysis

Let's now consider the new linear model, focusing on excluding the combinations with the factor C from the previous model:

lm3 : Elapsed ~A*B*D + C

A model constructed like this will consider all the single effects and all the combinations of them excluding those with the C factor. At first, let's check this model adequacy as we've done for the previous one:

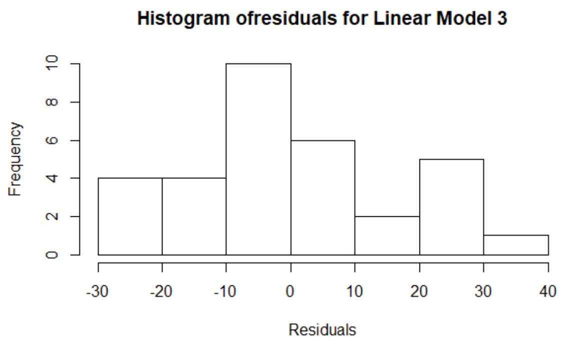


Figure 8: *Histogram of model 3*

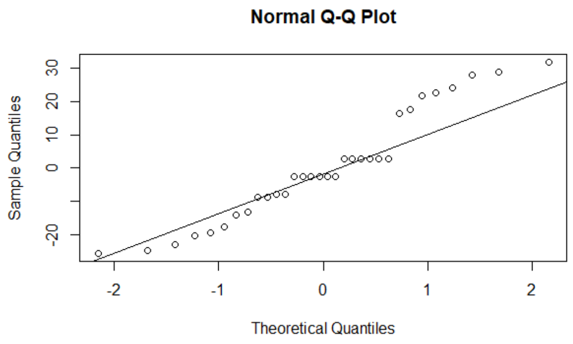
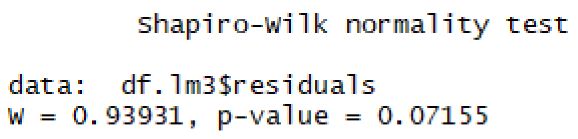


Figure 9: *Q-Q plot of model 3*



```
       Shapiro-Wilk normality test

data:  df.lm3$residuals
W = 0.93931, p-value = 0.07155
```

Figure 10: *Shapiro test of model 3*

As before the Shapiro-Wilk test tells us how much the distribution of residuals is normal and this time it has a p-value > 0.05, in particular p-value of 0.07155, pointing out that the model is this time normal but very close to the limit. This could be also noticed by looking at the shape of the histogram and from the Q-Q plot, that this time presents a little more "aligned" distribution of the quantiles; These conclusions are also confirmed by this other graph that provides us other statistics useful in order to analyze the normality distribution of the residuals, using the standardized residuals that are shown in Figure 11.

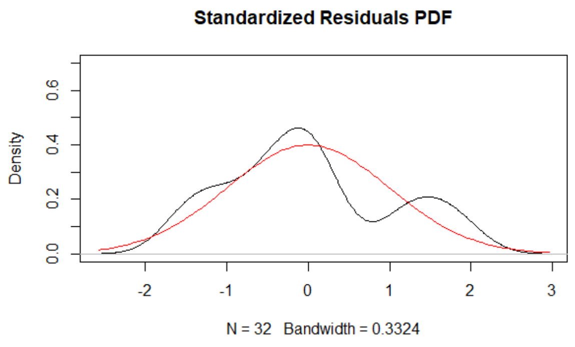
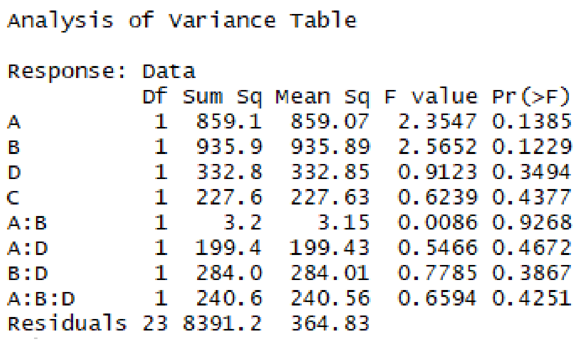


Figure 11: *Residual PDF of model 3*

The red line on the right graph represents a normal PDF, showing its classic bell shape centered in 0: as we can see the black line, representing the standardized residuals PDF, that doesn't overlap it. Performing an ANOVA analysis on this new and adequate model we obtain:



```
Analysis of Variance Table

Response: Data
          Df Sum Sq Mean Sq F value Pr(>F)
A          1  859.1  859.07  2.3547 0.1385
B          1  935.9  935.89  2.5652 0.1229
D          1  332.8  332.85  0.9123 0.3494
C          1  227.6  227.63  0.6239 0.4377
A:B        1    3.2    3.15  0.0086 0.9268
A:D        1  199.4  199.43  0.5466 0.4672
B:D        1  284.0  284.01  0.7785 0.3867
A:B:D      1  240.6  240.56  0.6594 0.4251
Residuals 23 8391.2  364.83
```

Figure 12: *Anova test of model 3*

Even if the model is normal according to the Shapiro test, the results are not so good, we can try to improve the model once again.

D. Third linear model analysis

The previous model was not sufficiently adequate, so let's try to improve it one more time to see if there is a stronger one; so let's now consider this model, that excludes the worst couple from the previous anova test

lm4 : Elapsed ~A*B*D + C - A*B +A+B

As before, it's necessary to check the model adequacy performing the same previous analysis:

This time the Shapiro-Wilk test provides us a fairly good p-value of 0.5624, that is sufficient to verify the Null-hypotesis,

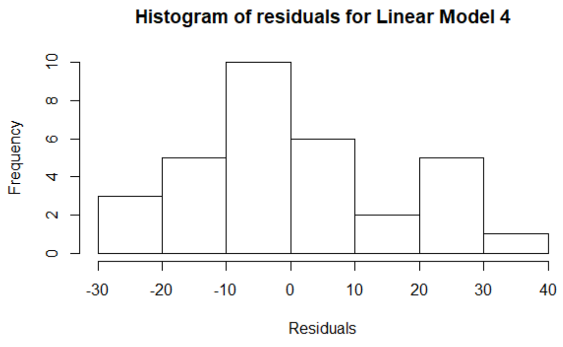


Figure 13: *Histogram of the model 4*

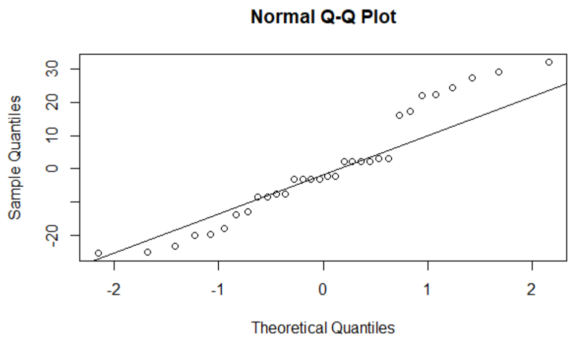
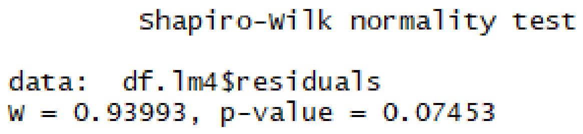


Figure 14: *Normal Q-Q of model 4*



```
         Shapiro-Wilk normality test

data:  df.lm4$residuals
W = 0.93993, p-value = 0.07453
```

Figure 15: *Shapiro test of model 4*

from which we can infere that this model residuals distribution has a pretty good match with a normal distribution; so this model is surely more adequate than all the others. Analyzing also the other plots we can surely say that the histogram presents a noticeable bell-like shape, as the quantile distribution follows better than before the normal line; As before, performing also the analysis on the standardized residuals confirms us the assumption by which we consider this model very adequate, since the PDF of the standardized residuals follow better the red line of the normal PDF, this is shown in Figure 16

Let's perform the ANOVA analysis to bring the conclusion about this model: Figure 17

In the end, even excluding the worst couple didn't give a better model with respect to the previous one, so let's now build a BoxCox model to get a better results from the model of the previous step instead

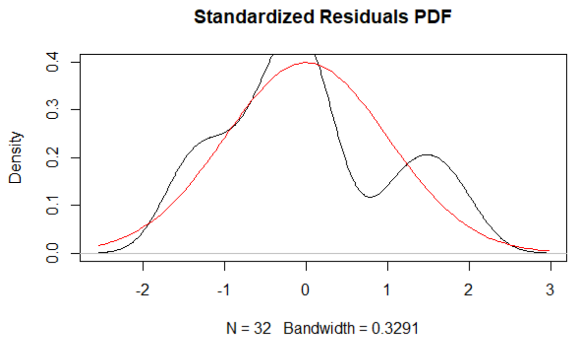
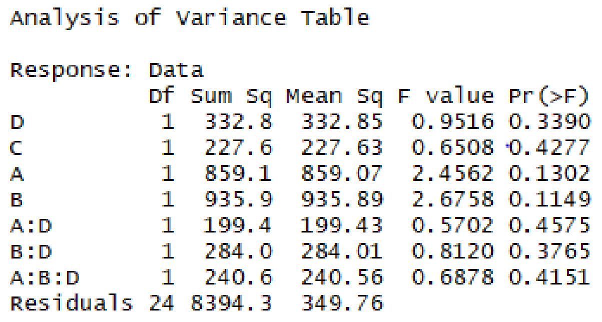


Figure 16: *Residual PDF of model 4*



```
Analysis of Variance Table

Response: Data
          Df Sum Sq Mean Sq F value Pr(>F)
D          1  332.8  332.85  0.9516 0.3390
C          1  227.6  227.63  0.6508 0.4277
A          1  859.1  859.07  2.4562 0.1302
B          1  935.9  935.89  2.6758 0.1149
A:D        1  199.4  199.43  0.5702 0.4575
B:D        1  284.0  284.01  0.8120 0.3765
A:B:D      1  240.6  240.56  0.6878 0.4151
Residuals 24 8394.3  349.76
```

Figure 17: *Anova test of model 4*

E. Box Cox Model

Finally we try to improve the results obtained by our model applying the BoxCox test. Which should give the best exponent for the output to optimize the normality of the model. Likelihood function is shown in Figure 18.

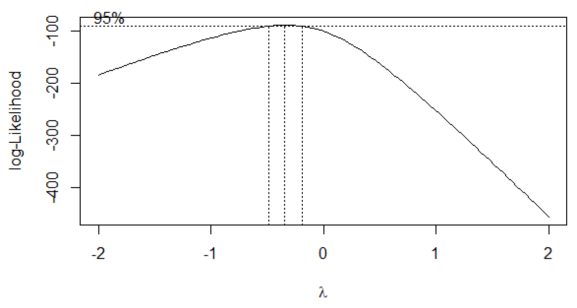


Figure 18: *Likelihood Function of the boxcox test*

The test provide us with a $\Lambda = -0.343434$ so we can write the non linear model as:

$$\text{lm4} : Elapsed^{0.3434} \sim \text{A*B*D} + \text{C}$$

In the end we get a much better result than the linear model we used before, looking at the Histograms and the result of the Shapiro test it's pretty straightforward that the normality of the model is much better after the boxcox test is performed:

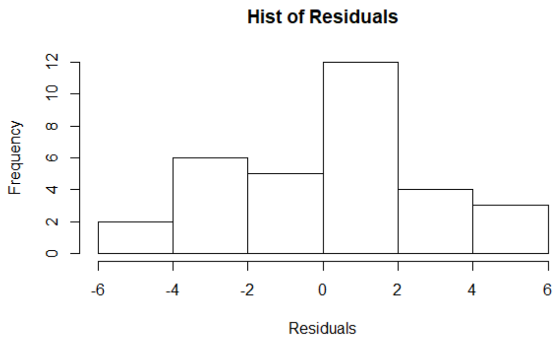
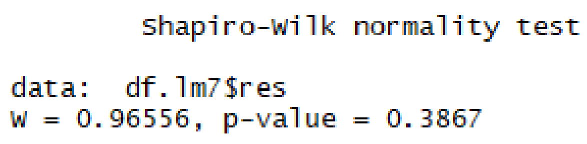


Figure 19: *Histogram after BoxCox*

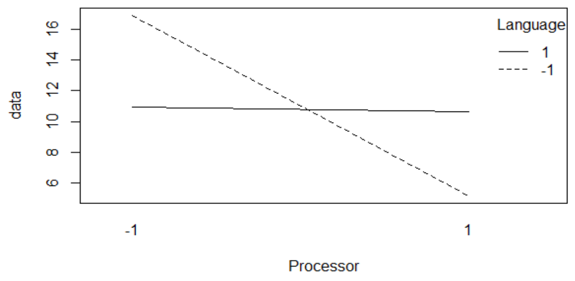


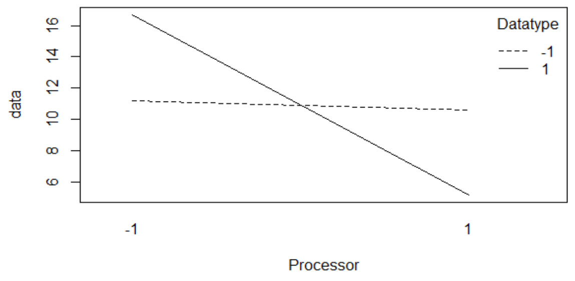
```
          Shapiro-Wilk normality test

data:  df.lm7$res
W = 0.96556, p-value = 0.3867
```

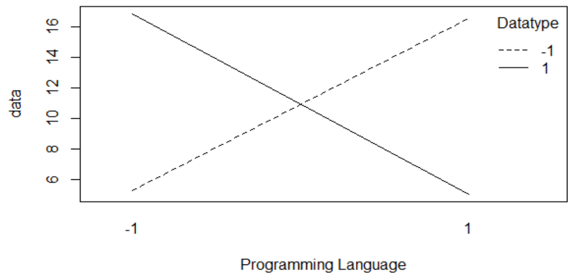
Figure 20: *Histogram after BoxCox*

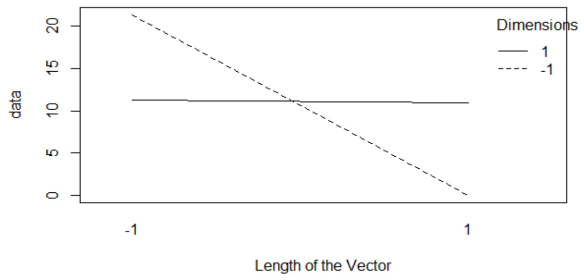
III. Conclusion

So, this statistical analysis offers us interesting data concerning the experiment performed: indeed, the Daniel's method at the beginning of the experiment design phase happens to neglect a factor which gives us the opportunity to perform an ANOVA test, in particular, the factor E which has been excluded was expected to be the weakest factor because the data type of the numbers doesn't make any difference in swapping memory location for bit strings, even more in modern CPUs architecture which have powerful hardware tools at their disposal. Moreover, the ANOVA analysis and the model found highlight how the B parameter carry more information with respect to the other 3 parameters which was expected since MATLAB is by far more effcient in working with vectors with respect to the C++ language another significant parameters when evaluating a sorting algorithm performances is of the course the CPU computation capability and this of course stands as the second most important factor. It's surprising that the factor C results to be the less relevant because it is well known that the length of the vector has great influence in those kind of algorithm performances, this can be explained of course, by the presence in the test of the MATLAB programming language which gives almost identical results even with bigger vectors, that should give a less significance to this factor which will result a lot more significant if another programming language is tested i.e. Java lowering the value of the factor B. Finally, notice that since the last model found was not so finely tuned and presented an low normality of the residuals, it has been chosen to perform a Box-Cox non-linear transformation of the model. Concluding with some interaction plots that depicted and intuitively shows the behaviour described by the model and the ANOVA analysis:





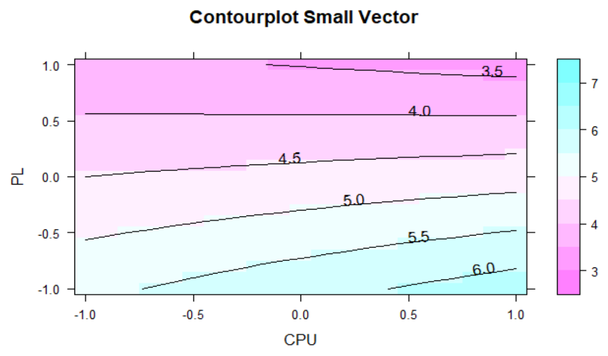


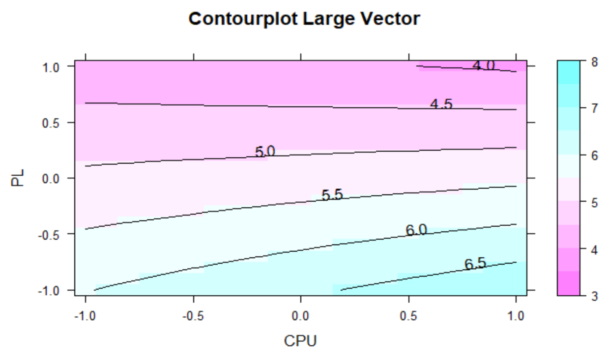


A. Contour Plots

Supporting what have been just said, let's briefly check the contour plots: indeed it is possible, with this tool, to abstract the values in between the two low and high levels of the so called quantitative factors (so those factor that have a real

”continuous” behaviour in the reality, and the discretization of it onto two levels was just a way to simplify the problem by study the extremes or, more precisely, two valuable values of that variable) using the model just found, predicting so the behaviour of the system in those intermediate levels. These quantitative factors are often analyzed with respect of a qualitative factor, for which have no sense to talk about intermediate values. Considering this and considering the last ANOVA analysis, the contour plots here below face the CPU model with the dimension of the vector (the two most influencing quantitative factors, A and B), with respect of the length of the vector. Of course,these are not proper quantitative functions since they are discrete but we can imagine, for example, that the test has been made on various CPUs that grows in computational capacity from -1 to the most powerful in 1. The Dimension also can be seen as a continuous variable because the length of the vector can be streched from a minimum of 1000 to a max of 100k elements passing all the intermediate integer values. The programmming language is a bit strange though cause the first one C++ works with Integer numbers or floating points numbers as basic data type while matlab works mostly with floating points numbers and the basic datatype is a Matrix and it’s a bit difficult to imagine programming languages that works with a setup that is intermediate between the two.





IV. R-CODE

.

```
###################
# analysis.r      #
# Giammario Fulco #
###################

#setwd ("~/Desktop/Homework/analysis")


library(MASS)
###### CREATING THE DATA FRAME FOR INPUT DATA #######


lvl<-c(-1,1)
dm<-expand.grid (A=lvl,B=lvl,C=lvl,D=lvl,E=lvl)
dm<-data.frame(StdOrder=c(1:nrow (dm)),dm)
dm<-data.frame(RunOrder=sample(nrow(dm)),dm)
dm<-dm[order(dm$RunOrder),]
dm$StdOrder =
c(21,19,3,4,29,1,25,15,22,13,26,28,17,30,8,7,2,16,14,23,24,11,18,5,9,10,31,32,20,1
2,27,6)
data<-
c(40.606,0.003,0.003,0.003,41.304,0.004,0.006,0.004,40.25,47.286,42.002,0.002,0.00
4,0.01,47.303,0.004,0.002,0.01,0.004,0.003,0.005,0.003,0.002,45.935,0.009,0.003,0.
004,0.005,0.003,0.003,0.003,44.252)
df<-data.frame(dm,Data=data)
#write.table(df,"~/Desktop/Homework/analysis/data_.dat",row=F)

##########################################################
## ANALYSIS                                             ##
##########################################################

############# MODEL 1 #################
df.lm<-lm(Data~A*B*C*D*E,d=df)

# Daniel s method : pl o t the e f f e c t s
effect<-as.vector(df.lm$effect)[2:length(df.lm$eff)]
qn<-qqnorm(effect,datax=T,ylab="EffectQuantiles")
qqline(effect,datax = T)
text(qn$x,qn$y,lab=names(df.lm$eff)[2:length(df.lm$eff)],pos=4)

anova(df.lm)

# ############# MODEL 2 ###############

# First model, after Daniel' s method
df.lm2<-lm(Data~ A*B*C*D, d=df)
anova (df.lm2)

hist(df.lm2$residuals,main="Histogram of residuals for LinearModel
2",xlab="Residuals")
```

```
shapiro.test(df.lm2$residuals) #<0.05  not normal

qqnorm(df.lm2$residuals)
qqline(df.lm2$residuals)

plot(jitter(df.lm2$fit),df.lm2$residuals,xlab="Fitted
Values",ylab="Residuals",main="Fitted Value Pattern")

# ############# MODEL 3 ################

df.lm3<-lm(Data~(A*B*D)+C,d=df)
anova(df.lm3)

hist(df.lm3$residuals,main="Histogram ofresiduals for Linear Model 3" , xlab =
"Residuals")

shapiro.test(df.lm3$residuals)

qqnorm(df.lm3$residuals)
qqline(df.lm3$residuals)

plot(jitter(df.lm3$fit),df.lm3$residuals,xlab="Fitted Values" ,ylab="Residuals" ,
main=" Fitted Value Pattern")

stdres<-rstandard(df.lm3)
plot(stdres,ylim=c(-3,3), ylab="Standardized Residuals" , main="Standardized
residuals distribution")
##1 sigma
abline(h=1, col="green")
abline(h=-1,col="green")
##2 sigma
abline(h=2,col="orange")
abline(h=-2,col="orange")
##3 sigma
abline(h=3,col="red")
abline(h=-3,col="red" )

plot(density(stdres),ylim=c(0,0.7),main="Standardized Residuals PDF" )
curve(dnorm(x),add=T,col="red")

########### MODEL 4 ###################
# Fourth model.
df.lm4<-lm(Data~ A*B*D+C-A*B+A+B,d=df)
anova(df.lm4)

hist(df.lm4$residuals,main="Histogram of residuals for Linear Model 4" , xlab =
"Residuals")

shapiro.test(df.lm4$residuals)

qqnorm(df.lm4$residuals)
qqline(df.lm4$residuals)
```

```r
plot(jitter(df.lm4$fit),df.lm4$residuals,xlab="Fitted Values
",ylab="Residuals",main="Fitted Value Pattern")

stdres4<-rstandard(df.lm4)
plot(stdres4,ylim=c(-3,3),ylab="Standardized Residuals",main="Standardized
residuals distribution")
#1 sigma
abline(h=1,col="green")
abline(h=-1,col="green")
#2 sigma
abline(h=2,col="orange")
abline(h=-2,col="orange")
#3 sigma
abline(h=3,col="red")
abline(h=-3,col="red")

plot(density(stdres4),ylim=c(0,0.4),main="Standardized Residuals PDF" )
curve(dnorm(x),add=T,col="red")

# BoxCox
A<-df[1:32,1]
B<-df[1:32,2]
C<-df[1:32,3]
D<-df[1:32,4]
E<-df[1:32,5]
y<-data
x<-A*B*D+C
#run the box-cox transformation
bc<-boxcox(y~x)

(lambda<-bc$x[which.max(bc$y)])

## [1] -0.3434343

#final model
df.lm7<-lm(Data^(lambda)~A*B*D+C, d=df)
#NORMALITY CHECKs (MAC)
# Shapiro milk test to check the normality
hist(df.lm7$res, xlab="Residuals",main ="Hist of Residuals")

shapiro.test(df.lm7$res)

# add noise
plot(jitter(df.lm7$fitted),df.lm7$res)

#QQ plot to check the normality of the residuals is fair
qqnorm(df.lm7$residuals)
qqline(df.lm7$residuals)

#anova
anova(df.lm7)
```

```r
interaction.plot(df$A,df$B,data, xlab="Processor", ylab=
"data",trace.label="Language")

interaction.plot(df$A,df$E,data, xlab="Processor", ylab=
"data",trace.label="Datatype")

interaction.plot(df$B,df$E,data, xlab="Programming Language", ylab=
"data",trace.label="Datatype")

interaction.plot(df$C,df$D,data, xlab="Length of the Vector", ylab=
"data",trace.label="Dimensions")


########### CONTOUR PLOTS ###################
library(lattice)

CPU<-seq(-1,1,0.1)
PL<-seq(-1,1,0.1)
g <- expand.grid(A = CPU,B=PL, C =df$C, D = df$D)
g$data <- predict(df.lm7,g)

contourplot(data~A*B, data = g[g$C==1,], cuts = 10, region = T, xlab = "CPU", ylab
= "PL", main = "Contourplot Small Vector")

contourplot(data~A*B, data = g[g$C==-1,], cuts = 10, region = T, xlab = "CPU",
ylab = "PL", main = "Contourplot Large Vector")
```