Problem Set 2

Anna Bellaver 918299, Maria Carta 917645, Roberto Di Cunto 894190, Martina Mattocci 1089750, Davide Valentini 1088760

Table of contents

Exercise 1

The dataset "psych" contains 24 psychological tests administered to 301 students (with ages ranging from 11 to 16) in a suburb of Chicago: a group of 156 students (74 boys, 82 girls) from the Pasteur School and a group of 145 students (72 boys, 73 girls) from the Grant-White School.

```
rm(list=ls())
psych<-read.table("data/psych.txt",header=T)
dim(psych)
```

```
[1] 301  28
```

```
head(psych)
```

```
  Case Sex  Age V1 V2 V3 V4 V5 V6 V7 V8 V9 V10 V11 V12 V13 V14 V15 V16 V17 V18
1    1   M 13.1 20 31 12  3 40  7 23 22  9  78  74 115 229 170  86  96   6   9
2    2   F 13.6 32 21 12 17 34  5 12 22  9  87  84 125 285 184  85 100  12  12
3    3   F 13.1 27 21 12 15 20  3  7 12  3  75  49  78 159 170  85  95   1   5
4    4   M 13.2 32 31 16 24 42  8 18 21 17  69  65 106 175 181  80  91   5   3
5    5   F 12.2 29 19 12  7 37  8 16 25 18  85  63 126 213 187  99 104  15  14
6    6   F 14.1 32 20 11 18 31  3 12 25  6 100  92 133 270 164  84 104   6   6
  V19 V20 V21 V22 V23 V24   group
1  16   3  14  34   5  24 PASTEUR
2  10  -3  13  21   1  12 PASTEUR
3   6  -3   9  18   7  20 PASTEUR
4  10  -2  10  22   6  19 PASTEUR
5  14  29  15  19   4  20 PASTEUR
6  14   9   2  16  10  22 PASTEUR
```

```
with(psych,table(group))
```

```
group
  GRANT PASTEUR
    145     156
```

"Sex" is a factor with levels "F" and "M"; "Age" is a numeric vector; "group" is a factor with levels "GRANT" and "PASTEUR". The 24 psychological test scores are named "V1" to "V24" and represent:

- V1 visual perception,
- V2 cubes,
- V3 paper form board,
- V4 flags,
- V5 general information,
- V6 paragraph comprehension,
- V7 sentence completion,
- V8 word classification,
- V9 word meaning,
- V10 addition,
- V11 code,
- V12 counting dots,
- V13 straight-curved capitals,
- V14 word recognition,
- V15 number recognition,
- V16 figure recognition,
- V17 object-number,
- V18 number-figure,
- V19 figure-word,
- V20 deduction,
- V21 numerical puzzles,
- V22 problem reasoning,
- V23 series completion,
- V24 arithmetic problems.

Point 1

Use the Grant-White students data. Obtain the maximum likelihood solution for (m = 5) and (m = 6) factors and compute the proportion of total sample variance due to each factor. List the specific variances, and assess the accuracy of the approximation of the correlation matrix. Compare the results. Which choice of m do you prefer? Why?

Factor Analysis

We perform Factor Analysis in order to identify *"m<p"* unobserved latent sources (called "Factors") that we use to represent our Data. Indeed, these Factors are Random Variables $F = (F_1, ..., F_m)$ that can express the original $X = (X_1, ..., X_p)$ by a linear combination:

$$X - \mu = LF + \epsilon$$

where:

1* $\mu = (\mu_1, ..., \mu_p)$: Is the Population Mean Vector of X,

2* L: Is the $m \times p$ Matrix of Factor Loadings, which contains the coefficients of the linear combination and is the object estimated in the model,

3* $\epsilon = (\epsilon_1, ..., \epsilon_p)$: Vector of the Errors.

We estimate the Matrix of Loadings using the maximum Likelyhood approach and we choose the "Varimax" rotation in order to obtain a set of coefficients that allow us to interpret easily the underlying Factors. Data will be standardized, so we'll deal with Loadings scaled by the standard deviations "σ_j" of the variable X_j they're referred to (so in that case, the Sample Covariance Matrix S is equal to the Sample Correlation Matrix R).

Choice of the number of Factors

To choose the best number of Factors to retain in the Model, we wish both to reach a satisfactory dimensionality reduction and to explain well the original Data. To evaluate those aspects, we'll perform Factor Analysis on *m=5* and *m=6* Factors and we'll look at:

- **Proportion of Total Sample Variance:** It's the amount of the total variation explained by the m factors (sum of the squared columns of L) considered with respect to the overall variance of data (equal to $trace(R)$, a good percentage is considered to be around the 80
- **Approximation of R:** Since the Factor Analysis Model aims to give a good approximation of the Sample Correlation Matrix $R = \hat{L}\hat{L}^T + \hat{\Psi}$ (where $\hat{\Psi}$ is the diagonal matrix that contains the "Specific Variances", so the (j, j)-th element of this matrix is

the portion of variance of the variable X_j not explained by the Factors). We'll compute the squared Frobenius Norm of the "Residual Matrix" ($\|R - (\hat{L}\hat{L}^T + \hat{\Psi})\|_F^2 = trace((R - (\hat{L}\hat{L}^T + \hat{\Psi}))(R - (\hat{L}\hat{L}^T + \hat{\Psi}))^T)$) which is equal to the sum of the squares of all the elements of the residual matrix and allows us to understand how better R is approximated by the model.

- **Communalities and Specific Variances:** We'll see which are the variables that are explained better by the Factors chosen and which of them may require an additional Factor (since the communalities are defined as the portion of variance of the variable X_j explained by all the factors considered, and they're computed as the sum of the squares of the j-th row of $\widehat{L}$).
- **Interpretability of the Factors:** Considering all the previous element, we'll evaluate the number of Factors also considering how much they're able to give a clear interpretation of the groups of variables. To do so, we'll look at the Loading Matrix and notice if there're variables that express a sufficiently high coefficient with respect to a certain factor (an acceptable threshold for the absolute value of the coefficient is around 0.6 or higher).

```
Grant_obs<-which(psych$group=="GRANT")
Grant<-psych[Grant_obs,-c(1,2,3,28)]
names(Grant)<-c( "visual perception", "cubes", "paper form board", "flags",
                "general information", "paragraph comprehension",
                "sentence completion", "word classification", "word meaning",
                "addition", "code", "counting dots", "straight curved capitals",
                "word recognition", "number recognition", "figure recognition",
                "object-number", "number-figure", "figure-word", "deduction",
                "numerical puzzles", "problem reasoning", "series completion",
                "arithmentic problems")
R_1<-cor(Grant)
```

We remove the first two columns and the last one since they are categorical variables and not numerical, we also remove the variable "Age", despite being numerical, since we want to study the 24 psychological tests without being influenced by the age of the students who did them.

We show the results obtained setting the number of Factors m=5, starting from the Matrix of Loadings $\widehat{L}$:

```
Grant.fa5<-factanal(covmat=R_1,factors=5)
Grant.fa5$loadings
```

```
Loadings:
                         Factor1 Factor2 Factor3 Factor4 Factor5
visual perception         0.165   0.655   0.125   0.181   0.207
cubes                     0.108   0.442
paper form board          0.134   0.559           0.112
flags                     0.230   0.533
general information       0.738   0.189   0.192   0.149
paragraph comprehension   0.772   0.187           0.248   0.124
sentence completion       0.798   0.214   0.143
word classification       0.571   0.343   0.239   0.128
word meaning              0.808   0.202           0.219
addition                  0.181  -0.108   0.845   0.180
code                      0.195           0.423   0.436   0.418
counting dots                     0.232   0.694   0.102   0.129
straight curved capitals  0.186   0.433   0.479           0.538
word recognition          0.185                   0.552
number recognition        0.104   0.122           0.509
figure recognition                0.406           0.509
object-number             0.154           0.210   0.595
number-figure                     0.300   0.322   0.458
figure-word               0.156   0.221   0.144   0.378
deduction                 0.373   0.461   0.127   0.293  -0.194
numerical puzzles         0.172   0.398   0.431   0.238
problem reasoning         0.364   0.423   0.114   0.320
series completion         0.362   0.542   0.248   0.231  -0.115
arithmentic problems      0.368   0.179   0.495   0.321

               Factor1 Factor2 Factor3 Factor4 Factor5
SS loadings      3.640   2.957   2.454   2.386   0.628
Proportion Var   0.152   0.123   0.102   0.099   0.026
Cumulative Var   0.152   0.275   0.377   0.477   0.503
```

Now, we consider the Model with *m=6* Factors, again starting by the Loading Matrix:

```
Grant.fa6<-factanal(covmat=R_1,factors=6)
Grant.fa6$loadings
```

```
Loadings:
                         Factor1 Factor2 Factor3 Factor4 Factor5 Factor6
visual perception         0.178   0.181   0.126   0.573   0.319   0.205
cubes                     0.112                   0.297   0.420
```

```
paper form board          0.145   0.153           0.612
flags                     0.244   0.123           0.487   0.182
general information       0.741   0.129   0.182   0.102   0.184   0.102
paragraph comprehension   0.774   0.229           0.152           0.151
sentence completion       0.815           0.160   0.221
word classification       0.588   0.151   0.263   0.365
word meaning              0.811   0.222                   0.181
addition                  0.177   0.168   0.835  -0.152
code                      0.188   0.400   0.413           0.104   0.559
counting dots                     0.131   0.704   0.221
straight curved capitals  0.194           0.501   0.472           0.438
word recognition          0.194   0.502                           0.106
number recognition        0.112   0.474                   0.154
figure recognition                0.477           0.317   0.257
object-number             0.151   0.733   0.215          -0.176
number-figure                     0.482   0.316   0.226   0.172
figure-word               0.162   0.384   0.141   0.187
deduction                 0.388   0.303   0.117   0.313   0.332  -0.152
numerical puzzles         0.181   0.207   0.418   0.232   0.409
problem reasoning         0.372   0.333           0.257   0.357
series completion         0.372   0.259   0.237   0.399   0.348
arithmentic problems      0.372   0.313   0.484           0.193

               Factor1 Factor2 Factor3 Factor4 Factor5 Factor6
SS loadings      3.753   2.468   2.435   2.133   1.132   0.669
Proportion Var   0.156   0.103   0.101   0.089   0.047   0.028
Cumulative Var   0.156   0.259   0.361   0.449   0.497   0.525
```

```r
p<-dim(Grant)[2]
diag(crossprod(Grant.fa5$loadings))/p
```

```
   Factor1    Factor2    Factor3    Factor4    Factor5 
0.15166898 0.12322568 0.10225064 0.09942214 0.02616646 
```

```r
diag(crossprod(Grant.fa6$loadings))/p
```

```
   Factor1    Factor2    Factor3    Factor4    Factor5    Factor6 
0.15635453 0.10281636 0.10145071 0.08887333 0.04716391 0.02786255 
```

```r
L5_G<-Grant.fa5$loadings
Residual5<-R_1-(L5_G%*%t(L5_G)+diag(Grant.fa5$unique))
sum(Residual5^2)
```

```
[1] 0.7335059
```

```r
L6_G<-Grant.fa6$loadings
Residual6<-R_1-(L6_G%*%t(L6_G)+diag(Grant.fa6$unique))
sum(Residual6^2)
```

```
[1] 0.6020222
```

We observe a small difference looking at the cumulative proportion of the total variance explained by the factors (for m=*5* is *0.503* instead for m=*6* is *0.525*), but there is not a negligible gap between the sums of the squared entries of the residual matrix (for m=*5* is *0.7335* instead for m=*6* is *0.602*). Moreover, factor 6 is not relevant in the cluster discrimination. For this reasons we choose m=*5*, since with m=*6* we have a better approximation of the correlation matrix, but it is not relevant for the discrimination of the clusters.

Point 2

Give an interpretation to the common factors in the (m = 5) solution with varimax rotation.

Checking the Loadings matrix for Grant students with $m=5$ we can distinguish 5 different groups:

Loadings:

	Factor1	Factor2	Factor3	Factor4	Factor5
visual perception	0.165	0.655	0.125	0.181	0.207
cubes	0.108	0.442			
paper form board	0.134	0.559		0.112	
flags	0.230	0.533			
general information	0.738	0.189	0.192	0.149	
paragraph comprehension	0.772	0.187		0.248	0.124
sentence completion	0.798	0.214	0.143		
word classification	0.571	0.343	0.239	0.128	
word meaning	0.808	0.202		0.219	
addition	0.181	-0.108	0.845	0.180	
code	0.195		0.423	0.436	0.418
counting dots		0.232	0.694	0.102	0.129
straight curved capitals	0.186	0.433	0.479		0.538
word recognition	0.185			0.552	
number recognition	0.104	0.122		0.509	
figure recognition		0.406		0.509	
object-number	0.154		0.210	0.595	
number-figure		0.300	0.322	0.458	
figure-word	0.156	0.221	0.144	0.378	
deduction	0.373	0.461	0.127	0.293	-0.194
numerical puzzles	0.172	0.398	0.431	0.238	
problem reasoning	0.364	0.423	0.114	0.320	
series completion	0.362	0.542	0.248	0.231	-0.115
arithmentic problems	0.368	0.179	0.495	0.321	

	Factor1	Factor2	Factor3	Factor4	Factor5
SS loadings	3.640	2.957	2.454	2.386	0.628
Proportion Var	0.152	0.123	0.102	0.099	0.026
Cumulative Var	0.152	0.275	0.377	0.477	0.503

We can interpret the factors as:

- factor 1 as reading and comprehension ability since it groups text and words related variables,

- factor 2 as logical reasoning and spatial processing since it relates logical skills, like deduction and problem reasoning, and logical tests like cubes,
- factor 3 as mathematical abilities because considers counting abilities,
- factor 4 as visual pattern recognition since it's about recognizing words, numbers, figures and objects,
- factor 5 represents only the variable straight-curved capitals.

Point 3

Make a scatter plot of the first two factor scores for m = 5 obtained by the regression method. Is their correlation equal to zero? Should we expect so? Comment

"Factor Scores" $f_i = (f_{i1}, ..., f_{im})$, $i = 1, ..., n$ represent estimates of the values hired by the factors estimated in the model. They're unobserved quantities estimated by the so called "Regression Method" through the formula:

$$f_i = \widehat{L}^T S^{-1}(x_i - \bar{x})$$

And in our case they can computed and plotted for diagnostic pourposes about the estimated Factors. In particular we can check if the assumptions made on the model are sufficiently satisfied by the factors extracted:

- **Uncorrelation between Factors**
- **Gaussianity of the Factors**
- **Mean zero and unitary Variance**

We start by computing the Factor Scores and displaying the first values:

```r
Grant.fa.reg<-factanal(x=Grant,factors=5,scores = "regression")

scores<-Grant.fa.reg$scores
```

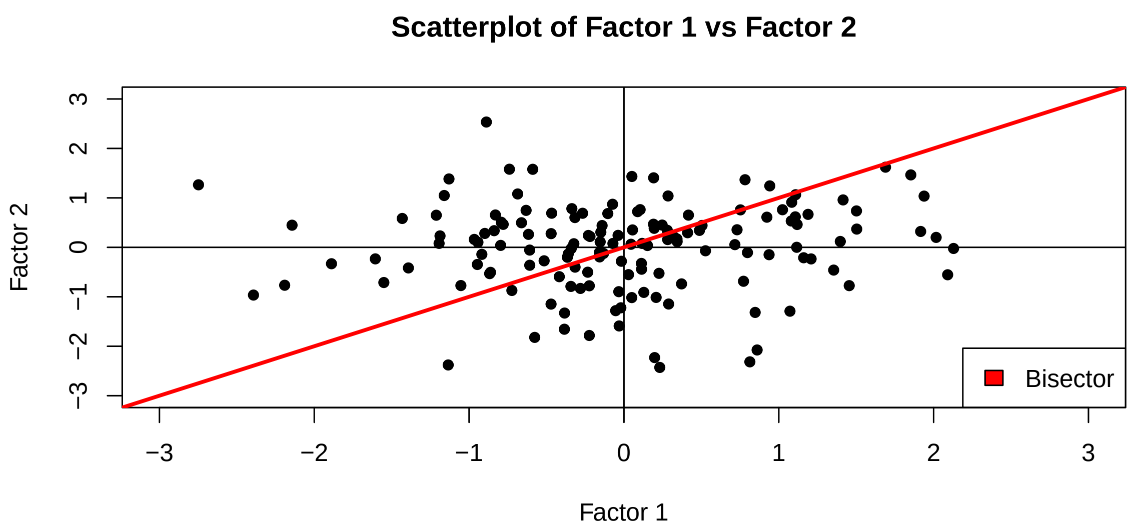
Then, we can rapidly check if the basic assumptions on Factors are satisfied:

```r
f_cor = cor(scores)[1:2, 1:2]
f_cov = cov(scores)[1:2, 1:2]
f_bar = colMeans(scores)[1:2]

summary = cbind(f_bar, diag(f_cov), rep(f_cor[1, 2], 2))
rownames(summary) = c("Factor 1", "Factor 2")
colnames(summary) = c("Expectations", "Variances", "Correlation")
summary
```

```
          Expectations Variances Correlation
Factor 1  6.094263e-17 0.8673433  0.07425218
Factor 2 -6.494147e-17 0.7712268  0.07425218
```

The assumptions are almost perfectly satisfied by the factors since the expectations are equal to "0" and their variances are very close to "1". About the correlation coefficient, it's nearly close to "0" ($\simeq 0.07$) and so we can conclude that the two factors are uncorrelated. We could expect such a result since uncorrelation between factors is one of the assumptions made in the model (as we said before). We can detect this aspect also by looking at the scatterplot of the factors scores:

```
plot(x = scores[, 1], y = scores[, 2],
     main = "Scatterplot of Factor 1 vs Factor 2", xlab = "Factor 1",
     ylab = "Factor 2", pch = 16, ylim = c(-3, 3), xlim = c(-3, 3))
abline(h = 0, v = 0)
abline(a = 0, b = 1, col = "red", lwd = 2.5)
legend("bottomright", fill = "red", legend = "Bisector")
```

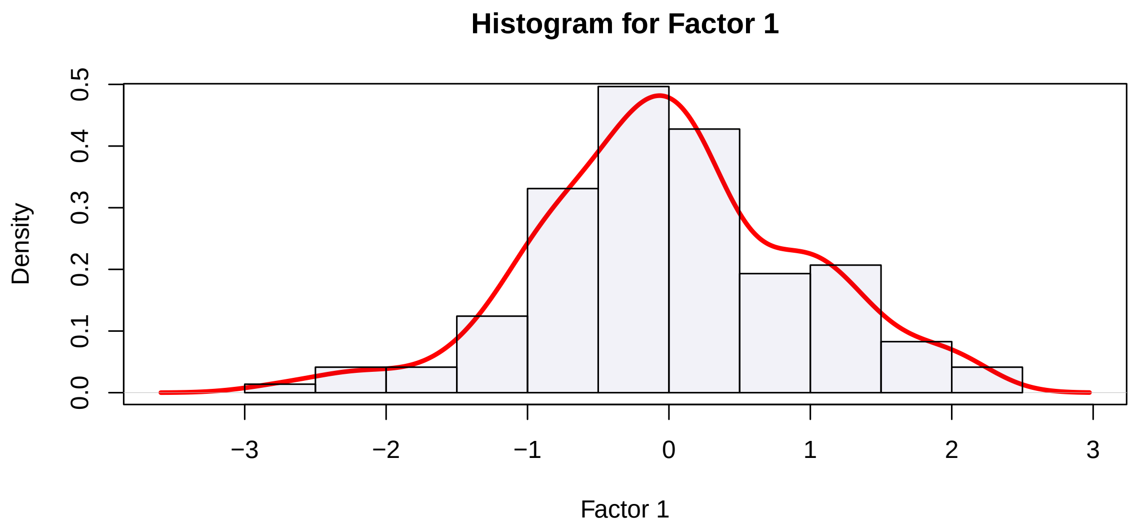


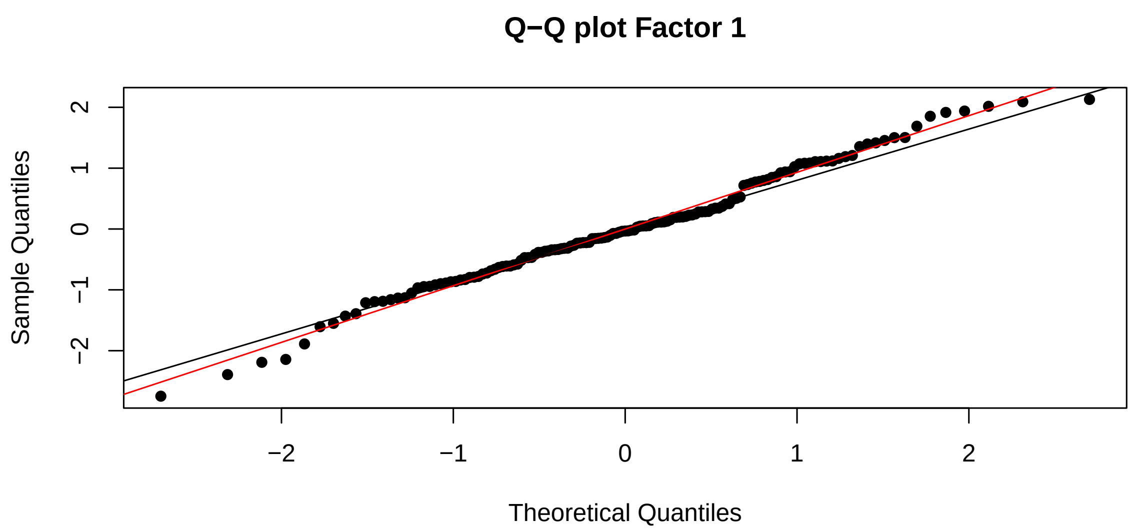
This plot graphically confirms what we saw before by computing the correlation coefficient, since the points seems to be scattered around the plane without a specific relationship, indeed they don't follow the bisector of the 1^{st} and 3^{rd} quadrant at all.

Then, we can check if normality assumption is satisfied plotting histograms and qq-plots of the factors:

Factor 1:

```
par(mfrow = c(2,1))

plot(density(scores[, 1]), lwd = 3, col = "red", main =  "Histogram for Factor 1",
     xlab = "Factor 1") #, ylim = c(0, 0.4)
color = rgb(red = 0, green = 0, blue = 0.5, alpha = 0.05)
hist(scores[, 1], probability = T, add = T, col = color)

sample_quantiles = sort(scores[, 1])
sample_mean = mean(scores[, 1])
sample_sd = sd(scores[, 1])

probabilities = ppoints(dim(scores)[1])
theoretical_quantiles = qnorm(probabilities)

plot(y = sample_quantiles, x = theoretical_quantiles, pch = 16,
     xlab = "Theoretical Quantiles", ylab = "Sample Quantiles",
     main = "Q-Q plot Factor 1")
abline(a = sample_mean, b = sample_sd, col = "red")
qqline(scores[, 1])
```



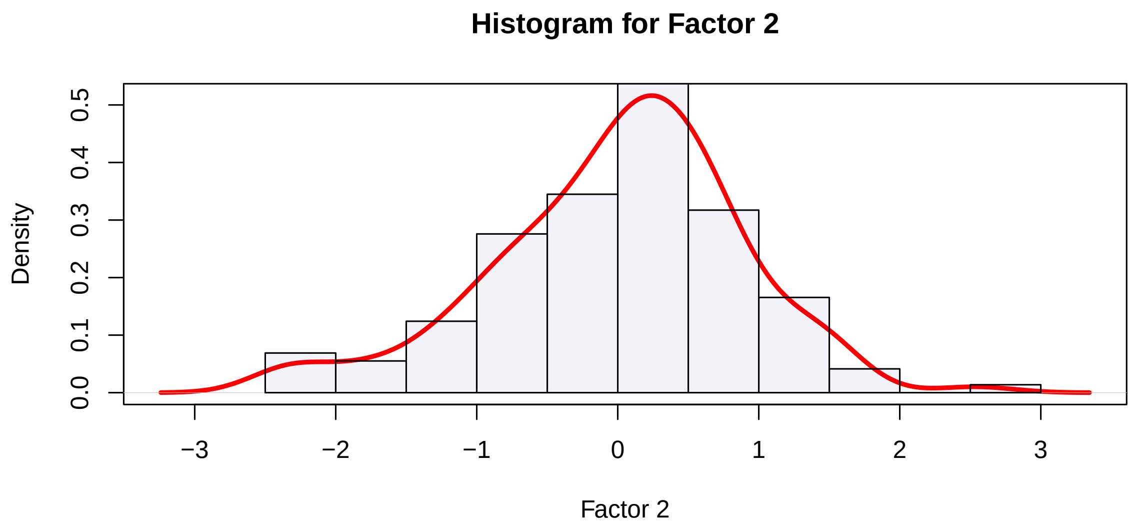


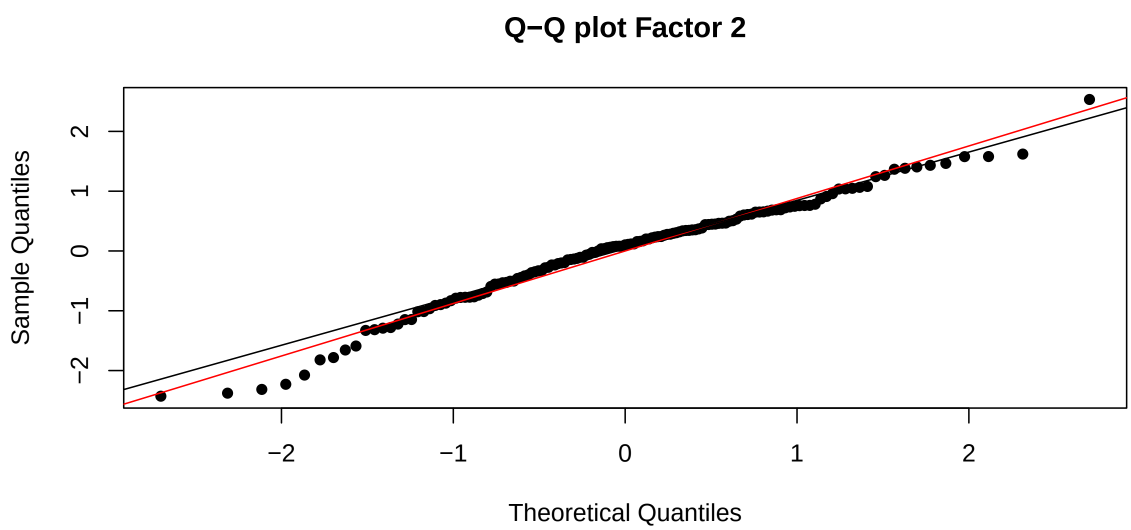
Both the histogram and the qq-plot clearly confirms that we can't consider F_1 to be normally distributed since its density function shows 2 different peaks and its associated quantiles deviates from the theoretical ones it should have under gaussian assumption.

Factor 2:

```
par(mfrow = c(2,1))
```

```
plot(density(scores[, 2]), lwd = 3, col = "red", main =  "Histogram for Factor 2",
     xlab = "Factor 2")
color = rgb(red = 0, green = 0, blue = 0.5, alpha = 0.05)
hist(scores[, 2], probability = T, add = T, col = color)

sample_quantiles = sort(scores[, 2])
sample_mean = mean(scores[, 2])
sample_sd = sd(scores[, 2])

probabilities = ppoints(dim(scores)[1])
theoretical_quantiles = qnorm(probabilities)

plot(y = sample_quantiles, x = theoretical_quantiles, pch = 16,
     xlab = "Theoretical Quantiles", ylab = "Sample Quantiles",
     main = "Q-Q plot Factor 2")
abline(a = sample_mean, b = sample_sd, col = "red")
qqline(scores[, 2])
```

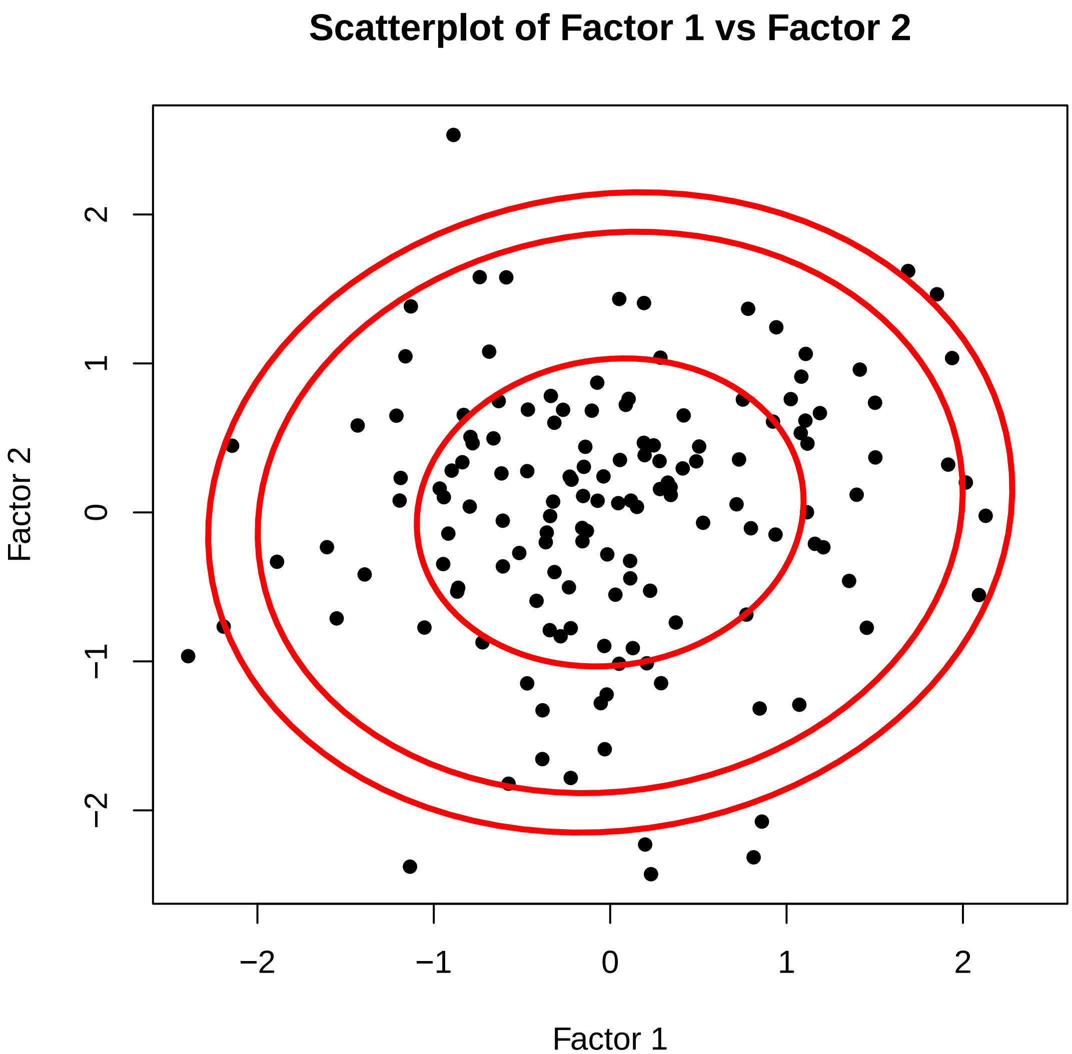




The situation is a bit better if we consider F_2, indeed its histogram looks more like a Normal Distribution, except for two local peaks of density we detect for the lowest and highest values of the variable. Indeed, also the qq-plot shows that the quantiles of F_2 deviates from the theoretical quantiles of a Normal Distribution just for the initial and the final quantiles.

We can finally make use of the scatterplots of the factors to see if the cloud of points shows an elliptical shape (which means that $F = (F_1, F_2) \sim \mathcal{N}_2(\begin{bmatrix} 0, & 0 \end{bmatrix}, I)$):

```
f_cov = cov(scores)
f_bar = colMeans(scores)
plot(x = scores[, 1], y = scores[, 2], main = "Scatterplot of Factor 1 vs Factor 2",
     xlab = "Factor 1", ylab = "Factor 2", pch = 16, xlim = c(-2.4, 2.4))
lines(ellipse(x = f_cov, centre = f_bar, level = 0.5), col = "red", lwd = 3)
lines(ellipse(x = f_cov, centre = f_bar, level = 0.9), col = "red", lwd = 3)
lines(ellipse(x = f_cov, centre = f_bar, level = 0.95), col = "red", lwd = 3)
```



The bivariate plot of factor scores shows that the cloud of points seems to have an elliptical shape just for the low/middle quantiles, while for high quantiles we can't confirm that gaussianity assumption is satisfied.

Point 4

Obtain the maximum likelihood solution with varimax rotation for (m = 5) factors by using the Pasteur students data. Is the interpretation to the common factors similar to that of Grant–White students?

```r
Pasteur_obs<-which(psych$group=="PASTEUR")
Pasteur<-psych[Pasteur_obs,-c(1,2,3,28)]
names(Pasteur)<-c("visual perception", "cubes", "paper form board", "flags",
                "general information", "paragraph comprehension",
                "sentence completion", "word classification", "word meaning",
                "addition", "code", "counting dots", "straight curved capitals",
                "word recognition", "number recognition", "figure recognition",
                "object-number", "number-figure", "figure-word", "deduction",
                "numerical puzzles", "problem reasoning", "series completion",
                "arithmentic problems")
R_2<-cor(Pasteur)

Pasteur.fa5<-factanal(covmat=R_2,factors=5)
Pasteur.fa5$loadings
```

```
Loadings:
                         Factor1 Factor2 Factor3 Factor4 Factor5
visual perception         0.314   0.578   0.138
cubes                             0.517                  -0.144
paper form board                  0.444  -0.177
flags                             0.671   0.190   0.170
general information       0.806                           0.143
paragraph comprehension   0.782   0.157                   0.202
sentence completion       0.904                   0.109
word classification       0.684   0.169   0.139   0.152
word meaning              0.775   0.249           0.102   0.156
addition                  0.141  -0.208   0.116   0.500   0.641
code                      0.349           0.231   0.671
counting dots                                     0.526   0.217
straight curved capitals          0.272           0.544
word recognition                          0.690
number recognition       -0.133   0.125   0.613  -0.110
figure recognition                0.386   0.475   0.176   0.161
object-number                             0.523   0.289
number-figure             0.100           0.465
figure-word                       0.244   0.357   0.241
```

deduction	0.123	0.514	0.189		
numerical puzzles	0.284	0.387	0.141	0.195	0.432
problem reasoning	0.469	0.481		0.152	
series completion	0.357	0.587	0.144		0.299
arithmentic problems	0.218	0.294	0.226	0.240	0.530

	Factor1	Factor2	Factor3	Factor4	Factor5
SS loadings	3.944	2.810	2.018	1.691	1.205
Proportion Var	0.164	0.117	0.084	0.070	0.050
Cumulative Var	0.164	0.281	0.366	0.436	0.486

By checking the Loadings matrix for Pasteur students with *m=5* is possible to distinguish 5 different groups:

Loadings:

	Factor1	Factor2	Factor3	Factor4	Factor5
visual perception	0.314	0.578	0.138		
cubes		0.517			-0.144
paper form board		0.444	-0.177		
flags		0.671	0.190	0.170	
general information	0.806				0.143
paragraph comprehension	0.782	0.157			0.202
sentence completion	0.904			0.109	
word classification	0.684	0.169	0.139	0.152	
word meaning	0.775	0.249		0.102	0.156
addition	0.141	-0.208	0.116	0.500	0.641
code	0.349		0.231	0.671	
counting dots				0.526	0.217
straight curved capitals		0.272		0.544	
word recognition			0.690		
number recognition	-0.133	0.125	0.613	-0.110	
figure recognition		0.386	0.475	0.176	0.161
object-number			0.523	0.289	
number-figure	0.100		0.465		
figure-word		0.244	0.357	0.241	
deduction	0.123	0.514	0.189		
numerical puzzles	0.284	0.387	0.141	0.195	0.432
problem reasoning	0.469	0.481		0.152	
series completion	0.357	0.587	0.144		0.299
arithmentic problems	0.218	0.294	0.226	0.240	0.530

	Factor1	Factor2	Factor3	Factor4	Factor5
SS loadings	3.944	2.810	2.018	1.691	1.205
Proportion Var	0.164	0.117	0.084	0.070	0.050
Cumulative Var	0.164	0.281	0.366	0.436	0.486

It is similar to the Grant data but with some differences:

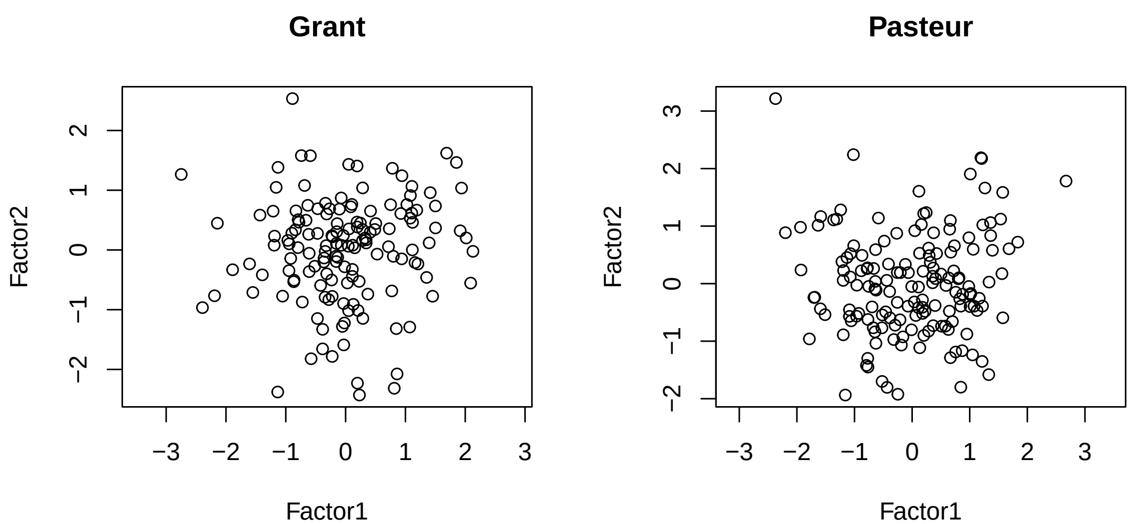
- factor 1 and 2 are the same as before,
- factor 3 is the same as factor 4 of Grant data apart from the ”code” variable which remains in factor 4, so we can now interpret factor 3 as visual pattern recognition,
- factor 4 contains ”code”, ”counting dots” and ”straight-curved capitals” which were before distributed over factors 4, 3 and 5,
- factor 5 is the same as factor 3 of Grant data apart from the ”counting dots” variable which remains in factor 3, so we can now interpret factor 5 as mathematical abilities.

We could explain this differences between Grant and Pasteur school with the presence of additional courses in both schools.

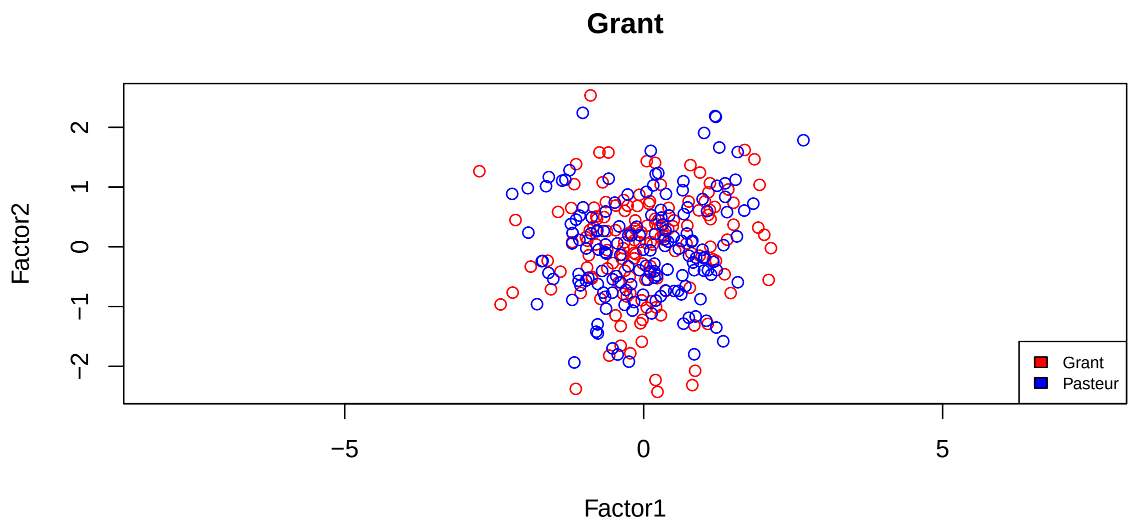
Point 5

Make a scatter plot of the first two factor scores from the rotated MLFA solution for each school. Comment.

```
Pasteur.fa.reg<-factanal(x=Pasteur,factors=5,scores = "regression")

par(mfrow=c(1,2))
plot(Factor2~Factor1, data=Grant.fa.reg$scores, asp=1, main="Grant")
plot(Factor2~Factor1, data=Pasteur.fa.reg$scores, asp=1, main="Pasteur")
```




```
par(mfrow=c(1,1))
plot(Factor2~Factor1, data=Grant.fa.reg$scores, asp=1, main="Grant", col="red")
points(Factor2~Factor1, data=Pasteur.fa.reg$scores, asp=1, main="Pasteur",
       col="blue")
legend("bottomright", fill = c("red", "blue"), legend = c("Grant", "Pasteur"),
       cex = 0.7)
```



There isn't a relevant difference between Grant and Pasteur scatter plots and this is coherent with point 4, where we observed that factors 1 and 2 have the same interpretation since they represent the same variables.

Exercise 2

The pendigits data set was created by collecting 250 samples from 44 writers. These writers were asked to write 250 digits in random order inside boxes of 500 by 500 tablet pixel resolution. The raw data on each of ($n = 10992$) handwritten digits consisted of a sequence, $((x_t,\ y_t),\ t = 1, 2, ..., T)$, of tablet coordinates of the pen at fixed time intervals of 100 milliseconds, where (x_t) and (y_t) were integers in the range 0-500. These data were then normalized to make the representations invariant to translation and scale distortions. The new coordinates were such that the coordinate that had the maximum range varied between 0 and 100. Usually (x_t) stays in this range, because most integers are taller than they are wide. Finally, from the normalized trajectory of each handwritten digit, 8 regularly spaced measurements, $((x_t,\ y_t))$, were chosen by spatial resampling, which gave a total of ($p = 16$) variables. The data includes a class attribute, column digit, coded $(0, 1, ..., 9)$, about the actual digit.

```r
pendigits<-read.table("data/pendigits.txt", sep=",",head=F)
names(pendigits)<-c(paste0(rep(c("x","y"),8),rep(1:8,each=2)),"digit")
dim(pendigits)
```

```
[1] 10992    17
```

```r
head(pendigits)
```

```
   x1  y1 x2  y2  x3  y3  x4  y4 x5 y5  x6 y6  x7 y7  x8 y8 digit
1  47 100 27  81  57  37  26   0  0 23  56 53 100 90  40 98     8
2   0  89 27 100  42  75  29  45 15 15  37  0  69  2 100  6     2
3   0  57 31  68  72  90 100 100 76 75  50 51  28 25  16  0     1
4   0 100  7  92   5  68  19  45 86 34 100 45  74 23  67  0     4
5   0  67 49  83 100 100  81  80 60 60  40 40  33 20  47  0     1
6 100 100 88  99  49  74  17  47  0 16  37  0  73 16  20 20     6
```

```r
lookup<-c("darkgreen",  "brown", "lightblue",  "magenta", "purple",
                        "blue", "red", "lightgreen", "orange", "cyan")
names(lookup)<-as.character(0:9)
digit.col<-lookup[as.character(pendigits$digit)]
```

Point 1

Use linear discriminant analysis (LDA). Display the first two LD variables in a scatterplot, color coding the observations according to variable "digit.col" above. How well do they discriminate the 10 digits? Refer also to theory.

Linear Discriminant Analysis (LDA) is a classification model which can be used to determine the class k that belongs to the discrete set $\zeta = 1, ..., K$ of a categorical target variable G of an observation, basing the evaluation on the values hired by some predictor variables for that observation. The model estimates for the i-th observation the posterior probability of being "class k" given the vector of realization of the predictor variables $x_i = (x_{i1}, ..., x_{ip})$, i. e.:

$$P(G = k|X = x_i)$$

and then it assigns the observation "i" to the class "k" (which is the predicted class $\widehat{G}(x_i)$) related to the highest posterior probability, so:

$$\widehat{G}(x_i) = \underset{k \in \zeta}{\text{argmax}}[P(G = k|X = x_i)]$$

In the contest of LDA, posterior probabilities are estimated making two important assumptions about the distributions of data:

- **Multivariate Gaussian:** We assume that each of the classes come from a Multivariate Gaussian Model $\mathcal{N}_p(\mu_k, \Sigma)$.
- **Equal Covariance Matrix:** We assume that each class has the same Covariance Matrix, i. e. $\Sigma_k = \Sigma$, for $k = 1, ..., K$.

So we fit the model making these assumptions and we use the following estimates for the quantities of interest:

1) **Prior Probabilites:** $\pi_k = \frac{n_k}{n}$, where n_k is the numerosity of the k-th class.
2) **Centroid of class "k":** $\hat{\mu}_k = \frac{1}{n_k} \sum_{x_i \in k} x_i$.
3) **Pooled Sample Covariance Matrix:** $\widehat{\Sigma} = \frac{1}{n-K} \sum_{k=1}^{K} \sum_{g_i = k} (x_i - \hat{\mu}_k)(x_i - \hat{\mu}_k)^T$, which is the common estimate of the Covariance Matrix. The Pooled Sample covariance Matrix of our dataset is the following:

```
n <- nrow(pendigits)
p <- ncol(pendigits)-1
```

```r
k<- length(unique(pendigits$digit))

classes<- c(0:9)
sigma_p<- matrix(0, nrow = p, ncol = p)
sigmas<- list()

for(index in seq(1, k)){
  sigmas[[index]] = cov(pendigits[which(pendigits$digit == classes[index]), ]
                      [, -(p+1)])*(nrow(pendigits[which(pendigits$digit ==
                                                       classes[index]), ]) - 1)
}

for(i in seq(1, k)){
  sigma_p = sigma_p + sigmas[[i]]
}
sigma_p<- (1/(nrow(pendigits) - k))*sigma_p
sigma_p
```

```
           x1         y1          x2          y2           x3           y3
x1  569.62833  82.130323  216.040011  161.112721 -119.8381848   78.0486388
y1   82.13032 148.922870   -5.871096   66.515022  -18.9564816   -7.3049015
x2  216.04001  -5.871096  515.762662  105.135988  137.4589096   99.4180957
y2  161.11272  66.515022  105.135988  214.919560  -86.5412282  153.6001366
x3 -119.83818 -18.956482  137.458910  -86.541228  453.6605080   -0.1910549
y3   78.04864  -7.304901   99.418096  153.600137   -0.1910549  297.7785150
x4  -27.36529 -24.481328 -166.284202  -58.368938  134.9687099  -29.2011398
y4   16.87195 -48.432201   48.053282   31.278471   49.1977115  215.7544874
x5   14.02458 -39.249972  -18.592210   -1.604847  -89.1830336   39.1360503
y5  -76.35149 -59.614333  -26.375446  -93.669338   50.4448317   10.2097151
x6  -93.57750 -16.972312   40.190463   -5.242858   -8.0623544   44.7774463
y6 -161.52629 -43.775485 -128.697930 -168.992693   39.9268111 -180.2330547
x7  -64.88043  10.244017  -44.630328    1.640668  -54.2710975    0.2695008
y7 -146.20005 -12.275067 -165.644958 -150.376559   32.0203246 -218.5493322
x8  -72.68523 -31.610878 -135.497777  -10.168586 -136.8839143  -43.2911781
y8  -62.59829  14.463453 -133.766049  -59.757474   -8.8318376 -140.5540981
           x4         y4          x5          y5          x6          y6
x1  -27.36529   16.87195   14.024581  -76.351488  -93.577501 -161.52629
y1  -24.48133  -48.43220  -39.249972  -59.614333  -16.972312  -43.77549
x2 -166.28420   48.05328  -18.592210  -26.375446   40.190463 -128.69793
y2  -58.36894   31.27847   -1.604847  -93.669338   -5.242858 -168.99269
x3  134.96871   49.19771  -89.183034   50.444832   -8.062354   39.92681
```

```
y3  -29.20114  215.75449   39.136050   10.209715   44.777446 -180.23305
x4  585.48932  103.14164  269.312322  130.040503  -70.389163   83.79976
y4  103.14164  360.31098  145.026911  223.593891  118.382425  -52.34033
x5  269.31232  145.02691  537.854314  160.618907  222.173685   10.29596
y5  130.04050  223.59389  160.618907  355.914576  155.893655  172.62551
x6  -70.38916  118.38242  222.173685  155.893655  592.448697   43.12032
y6   83.79976  -52.34033   10.295956  172.625507   43.120316  342.11339
x7 -164.82079  -18.38913 -108.541737   -1.759184  218.779728   24.62186
y7   37.20243 -213.55300 -108.805085  -36.521224  -87.030404  237.36862
x8 -113.12444 -187.98368 -204.408576 -256.686355 -336.119992  -46.66687
y8   -7.47653 -227.16219 -149.956868 -174.725331 -163.862140   44.07700
             x7         y7         x8          y8
x1  -64.8804349 -146.20005  -72.68523  -62.598291
y1   10.2440174  -12.27507  -31.61088   14.463453
x2  -44.6303284 -165.64496 -135.49778 -133.766049
y2    1.6406681 -150.37656  -10.16859  -59.757474
x3  -54.2710975   32.02032 -136.88391   -8.831838
y3    0.2695008 -218.54933  -43.29118 -140.554098
x4 -164.8207902   37.20243 -113.12444   -7.476530
y4  -18.3891296 -213.55300 -187.98368 -227.162186
x5 -108.5417372 -108.80508 -204.40858 -149.956868
y5   -1.7591842  -36.52122 -256.68636 -174.725331
x6  218.7797276  -87.03040 -336.11999 -163.862140
y6   24.6218557  237.36862  -46.66687   44.076997
x7  393.8819968   31.04184   96.57732   35.370766
y7   31.0418421  359.56867  163.45104  255.816369
x8   96.5773238  163.45104  836.50539  274.038583
y8   35.3707661  255.81637  274.03858  356.211411
```

Model Assumptions:

We can rapidly check if the assumptions of LDA are satisfied by our dataset.

1) *Equal Covariance Matrix:*

We compute the Frobenius Norms=sqrt(trace(XX^*), where X^* is the conjugate transpose, of the approximation between $\widehat{\Sigma}$ and the Sample covariance Matrices of each class Σ_k in order to see if we can consider the different classes of data to have the same Covariance structure:

```r
differences=c()
for(i in seq(1, k)){
  differences[i] = sqrt(tr((sigma_p - sigmas[[i]])%*%t((sigma_p - sigmas[[i]]))))
}
```

```
names(differences)<-classes
differences
```

```
       0        1        2        3        4        5        6        7
 5246379  5529992  1726144  1151598  2801833 13150969  1844132  3478116
       8        9
 6834221  3643434
```

The results show that probably we can't consider the different classes to have the same Covariance Matrix since they return quite different approximations of $\widehat{\Sigma}$.

2) *Multivariate Normality of the K classes:*

To investigate the multivariate normality for each digit class, we have decided to produce a "Gamma plot" (in the x-axis the Chisquare quantile and in the y-axis the squared Mahalanobis distance) for all of them. The Gamma plot is a QQ plot of squared Mahalanobis distances and so if the data is multivariate normal, it is expected that most of the points should fall on the diagonal line.

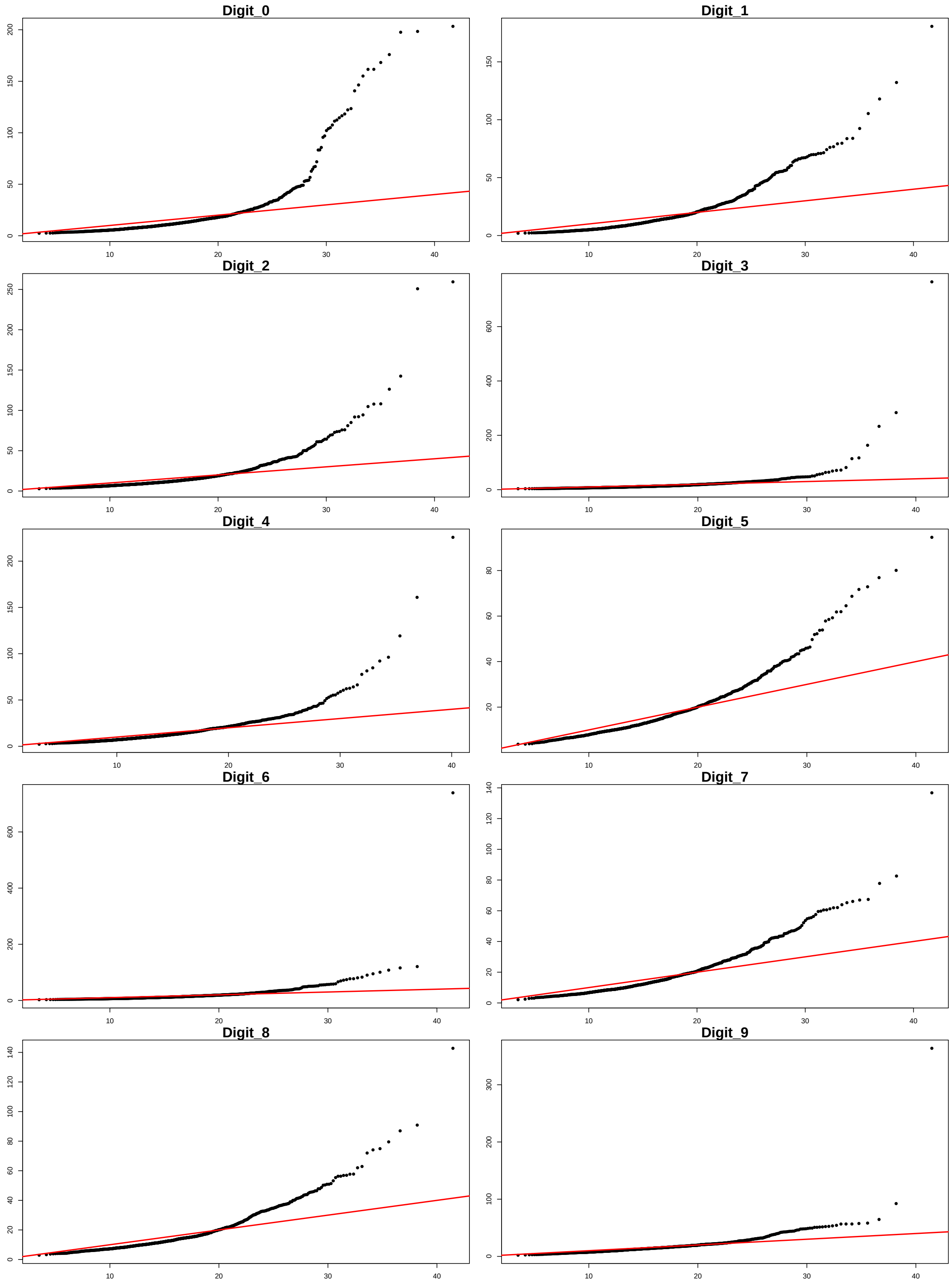
```
data_digit = list()
for (i in seq(1, k)) {
  data_digit[[i]] = pendigits[which(pendigits$digit == classes[i]), ][, -(p+1)]
}

x.bar<- c()
for (i in seq(1, k)) {
  x.bar[[i]] = colMeans(data_digit[[i]])
}

S<- c()
for (i in seq(1, k)) {
  S[[i]] = cov(data_digit[[i]])
}
#S[[1]]

n<- ncol(data_digit[[1]])
nrow(data_digit[[1]])
```

```
[1] 1143
```

```
#Digit 0
par(mfrow=c(5,2), mar = c(1.9, 1.9, 1.9, 1.9))

mdist0<- mahalanobis(data_digit[[1]], center = x.bar[[1]], cov = S[[1]])
plot(qchisq(ppoints(mdist0), df=n), sort(mdist0), pch=16, xlab="Chisquare quantile",
     ylab="Squared Mahalanobis distance")
abline(a=0, b=1, col="red", lwd=2)
title(main =list("Digit_0", cex=2))


#Digit 1
mdist1<- mahalanobis(data_digit[[2]], center = x.bar[[2]], cov = S[[2]])
plot(qchisq(ppoints(mdist1), df=n), sort(mdist1), pch=16, xlab="Chisquare quantile",
     ylab="Squared Mahalanobis distance")
abline(a=0, b=1, col="red", lwd=2)
title(main =list("Digit_1", cex=2))


#Digit 2
mdist2<- mahalanobis(data_digit[[3]], center = x.bar[[3]], cov = S[[3]])
plot(qchisq(ppoints(mdist2), df=n), sort(mdist2), pch=16, xlab="Chisquare quantile",
     ylab="Squared Mahalanobis distance")
abline(a=0, b=1, col="red", lwd=2)
title(main =list("Digit_2", cex=2))


#Digit 3
mdist3<- mahalanobis(data_digit[[4]], center = x.bar[[4]], cov = S[[4]])
plot(qchisq(ppoints(mdist3), df=n), sort(mdist3), pch=16, xlab="Chisquare quantile",
     ylab="Squared Mahalanobis distance")
abline(a=0, b=1, col="red", lwd=2)
title(main =list("Digit_3", cex=2))


#Digit 4
d_4<- data_digit[[5]][,-16]
x.bar4<- colMeans(d_4)
S_4<- cov(d_4)

mdist4<- mahalanobis(d_4, center = x.bar4, cov = S_4)
plot(qchisq(ppoints(mdist4), df=n-1), sort(mdist4), pch=16, xlab="Chisquare quantile",
```

```
     ylab="Squared Mahalanobis distance")
abline(a=0, b=1, col="red", lwd=2)
title(main =list("Digit_4", cex=2))


#Digit 5
mdist5<- mahalanobis(data_digit[[6]], center = x.bar[[6]], cov = S[[6]])
plot(qchisq(ppoints(mdist5), df=n), sort(mdist5), pch=16, xlab="Chisquare quantile",
     ylab="Squared Mahalanobis distance")
abline(a=0, b=1, col="red", lwd=2)
title(main =list("Digit_5", cex=2))


#Digit 6
mdist6<- mahalanobis(data_digit[[7]], center = x.bar[[7]], cov = S[[7]])
plot(qchisq(ppoints(mdist6), df=n), sort(mdist6), pch=16, xlab="Chisquare quantile",
     ylab="Squared Mahalanobis distance")
abline(a=0, b=1, col="red", lwd=2)
title(main =list("Digit_6", cex=2))


#Digit 7
mdist7<- mahalanobis(data_digit[[8]], center = x.bar[[8]], cov = S[[8]])
plot(qchisq(ppoints(mdist7), df=n), sort(mdist7), pch=16, xlab="Chisquare quantile",
     ylab="Squared Mahalanobis distance")
abline(a=0, b=1, col="red", lwd=2)
title(main =list("Digit_7", cex=2))


#Digit 8
mdist8<- mahalanobis(data_digit[[9]], center = x.bar[[9]], cov = S[[9]])
plot(qchisq(ppoints(mdist8), df=n), sort(mdist8), pch=16, xlab="Chisquare quantile",
     ylab="Squared Mahalanobis distance")
abline(a=0, b=1, col="red", lwd=2)
title(main =list("Digit_8", cex=2))


#Digit 9
mdist9<- mahalanobis(data_digit[[10]], center = x.bar[[10]], cov = S[[10]])
plot(qchisq(ppoints(mdist9), df=n), sort(mdist9), pch=16, xlab="Chisquare quantile",
     ylab="Squared Mahalanobis distance")
```

```
abline(a=0, b=1, col="red", lwd=2)
title(main =list("Digit_9", cex=2))
```


Digit_0
Digit_1
Digit_2
Digit_3
Digit_4
Digit_5
Digit_6
Digit_7
Digit_8
Digit_9

This allows us to conclude that the assumption of Multivariate normality for the K classes is clearly not satisfied, since we observe that each class' distribution is characterized by an heavy right tail.

The LDA perfomes well on an amazingly large and diverse set of classification tasks. The reason for which the LDA have such a good track record, is not likely to be that the data are approximately Gaussian and that the covariances are approximately equal. More likely a reason is that the data can only support simple decision boundaries such as linear, and the estimates provided via the Gaussian models are stable. This is a bias-variance trade off, we can put up with the bias of a linear decision boundary because it can be estimated with much lower variance than more exotic alternatives. For all these reasons, using the LDA model with the assumptions of Multivariate Normality and equal Covariance Matrices both for the K classes, could be a good solution but not optimal.

```r
lda.fit<-lda(pendigits$digit~.,data=pendigits)
lda.fit$scaling[,c(1,2)]
```

```
            LD1          LD2
x1  0.017031469  0.010842533
y1  0.010496495  0.030217727
x2  0.006154561 -0.002398809
y2 -0.034121100 -0.038556558
x3 -0.024329618 -0.022268517
y3  0.003846840  0.004680081
x4 -0.006772741  0.005488090
y4  0.015380360 -0.012348136
x5  0.002884508 -0.008763204
y5 -0.011847742 -0.009512370
x6  0.011499150  0.019019773
y6 -0.001588012  0.015872651
x7 -0.002137111 -0.006247144
y7  0.023485579 -0.008138420
x8 -0.020739921  0.017834059
y8  0.032188123 -0.052611219
```

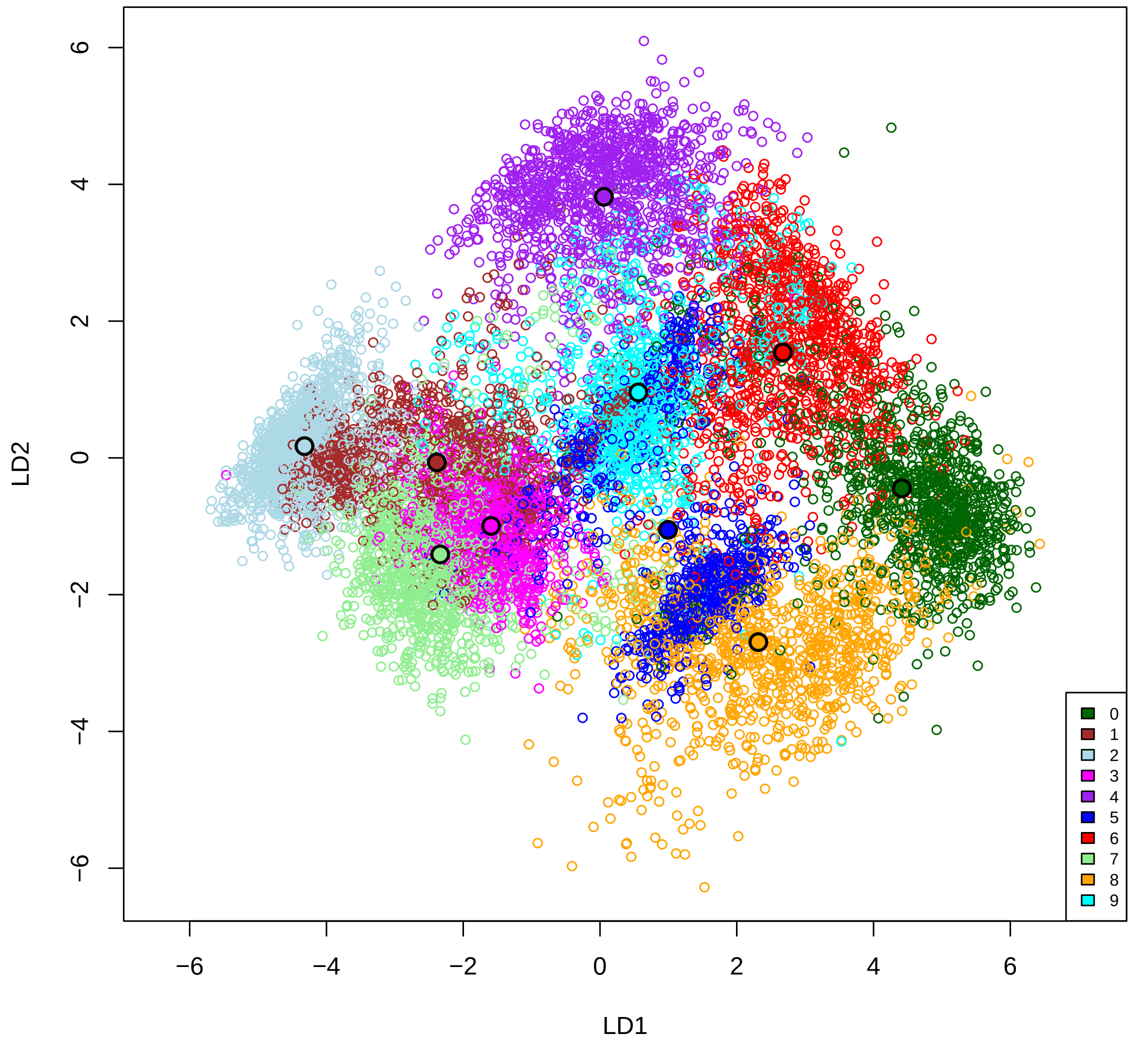
```r
lda.pred<-predict(lda.fit)
names(lda.pred)
```

```
[1] "class"     "posterior" "x"
```

```r
round(lda.pred$x[1:5,],3)
```

```
     LD1    LD2    LD3    LD4    LD5    LD6    LD7    LD8    LD9
1  2.963 -2.900  0.219  0.097 -0.920 -1.548  1.891 -1.170  0.410
2 -3.621  1.060  2.056  1.162 -0.180 -0.552  0.669 -0.015 -1.162
3 -1.437 -0.732 -0.661 -0.767 -0.509  2.785 -1.435  1.297 -0.124
4 -0.808  3.120 -0.953  1.986  1.467 -0.566 -0.252 -0.150  0.123
5 -3.291 -1.005 -1.025 -0.530 -0.299  0.633 -0.937  1.165  0.921
```

```r
means.hat<-aggregate(lda.pred$x,by=list(pendigits$digit),FUN=mean)
means.hat<-means.hat[,-1]

plot(LD2~LD1,data=lda.pred$x,asp=1,pch=1,col=digit.col,cex=0.8)
points(means.hat[,1],means.hat[,2],cex=1.5,bg=lookup,pch=21,lwd=2)
legend("bottomright", fill = lookup, legend = as.character(0:9),cex = 0.7)
```

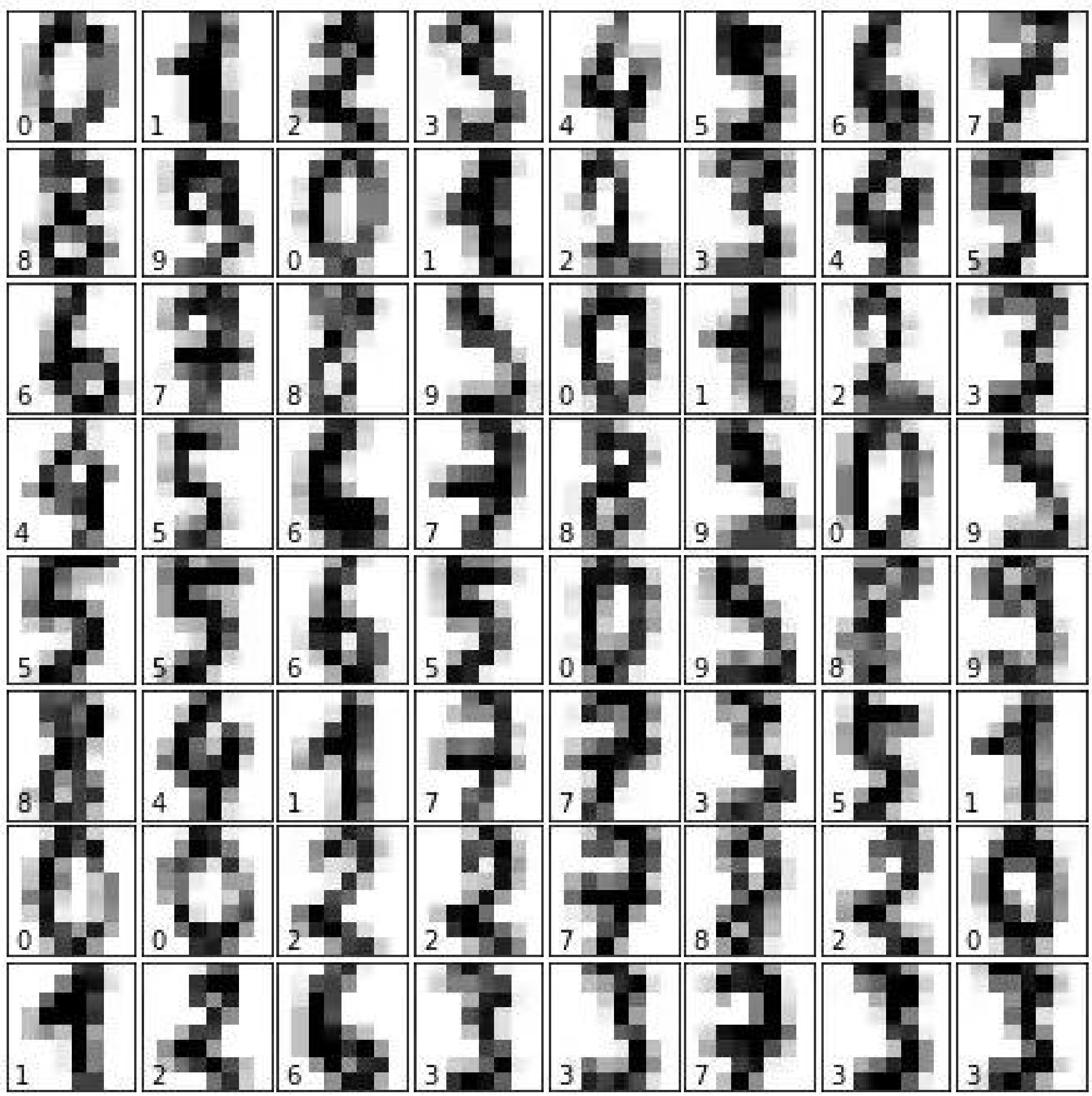


The above plot shows that some centroids are set in a region of the (LD1, LD2) space which is not shared by the other, and this means that the have a clear and distinguishable position that allows the model to produce a good prediction of these classes, while there are some centroids which are closer to each other and present some overlaps for their observations. In particular:

- digit 0 is quite well discriminated, it overlaps a bit with digits 8 and 6,
- digit 1 is not well discriminated, it overlaps with 2, 3 and 7,

- digit 2 is quite well discriminated, it overlaps only a bit with 1,
- digit 3 is quite well discriminated, it overlaps a bit with 1 and 7,
- digit 4 is quite well discriminated, it overlaps only a bit with 9,
- digit 5 is the worst discriminated, it overlaps with 3, 8 and 9,
- digit 6 is quite well discriminated, it doesn't seem to overlap with other digits,
- digit 7 is quite well discriminated, it overlaps a bit with 1 and 3,
- digit 8 is not well discriminates, it overlaps with 0 and 5,
- digit 9 is not well discriminates, it overlaps with 4, 5 and 6.

It is coherent with the pixel representation of the digits, as we can see from this picture



This is because, for example, digits 5 and 3 have the pixels concentrated in the same regions of the cell, as 8 and 9. Instead 1, 2 and 7 have the pixels concentrated in the superior part of the cell. This is the reason of the misclassification.

To show how much the LDA variables count in the discrimination of the digit classes (K=10), relative to the prediction, we can consider the singular values, which give the ratio of the between- and within-group standard deviations on the linear discriminant variables (their squares are the canonical F-statistics).

```r
cumsum(lda.fit$svd^2/sum(lda.fit$svd^2))
```

```
[1] 0.4245951 0.6122662 0.7328315 0.8179385 0.8904452 0.9406798 0.9772083
[8] 0.9968587 1.0000000
```

which means that the first 4 discriminant variables explain most of the variance of the data points.

Point 2

Compute the confusion matrix on the training data. What are the groups more difficult to discriminate from the others? Comment in view of the answer to point 1.

```
(H<-table(fitted=lda.pred$class, true=pendigits$digit))
```

```
      true
fitted    0    1    2    3    4    5    6    7    8    9
     0 1015    0    0    0    0    0    4    1   76    1
     1   10  799   23   19    0    3    0   61   18   64
     2    0  206 1113    1    1    0    0   16    0    0
     3    0   16    1 1020    1   58    0   19   10   10
     4   33    2    0    0 1116    0    3    7    0   20
     5    0   51    0    0    0  714    5    5   57   12
     6    5    7    0    0    4    3 1029    0    6    1
     7    0   29    7   13    1    0    0 1015    2    0
     8   77    0    0    0    0    9   15    6  875    2
     9    3   33    0    2   21  268    0   12   11  945
```

```
mis<-rep(1:10,0)
for (k in c(1:10)) {
  mis[k]<-(sum(H[,k])-H[k,k])/sum(H[,k])*100
}
mis
```

```
 [1] 11.198600 30.096238  2.709790  3.317536  2.447552 32.322275  2.556818
 [8] 11.120841 17.061611 10.426540
```

```
mean(mis)
```

```
[1] 12.32578
```

We can observe, by looking at the confusion matrix, that digit 1 and 5 are the worst discriminated, indeed if we compute the error class by class we observe that digit 1 and 5 have an overall 30% rate of error, which is way above the sample mean error of 12% (even though also digit 8 does not score well, it has an error rate of 17%). Despite that, digits 2, 3, 4 and 6 are optimally fitted, while digits 0, 7 and 9 are just well discriminated.

We can also observe that, as one could expect, the scatterplot matrix was not interpreted optimally, but it succeeded to recognize that digit 1 and 5 are badly fitted in spite of digit 6 which was recognized as one of the well fitted classes.

Point 3

Use leave-one-out cross validation (CV). Compute the confusion matrix and the corresponding CV error. Is it larger than the training error? Why so?

Consider the AER (Actual Error Rate), with $K = 2$ for simplicity of notation,

$$AER = \pi_2 P(X \in \hat{R}_1)|G = 2) + \pi_1 P(X \in \hat{R}_2)|G = 1) = \pi_1 \int_{\hat{R}_2} f_1(x)dx + \pi_2 \int_{\hat{R}_1} f_2(x)dx$$

where π_k are the prior probabilities of class k and $f_k(x)$ are the unknown class-conditional densities of X in class $G = k$. It is the total probability of misclassification in terms of the classification regions $\hat{R}_k$ estimated from the training data. The AER also indicates how the sample classification rule will perform in future samples.

```r
lda.cv.fit<-lda(pendigits$digit~.,data=pendigits, CV=TRUE)

table(lda.cv.fit$class,pendigits$digit)
```

```
       0    1    2    3    4    5    6    7    8    9
0   1013    0    0    0    0    0    4    1   79    1
1     10  798   23   19    1    3    0   61   18   65
2      0  206 1113    1    2    0    0   17    0    0
3      0   16    1 1020    1   58    0   20   10   10
4     34    2    0    0 1115    0    3    7    0   20
5      0   51    0    0    0  712    5    5   57   13
6      5    8    0    0    4    3 1029    0    7    1
7      0   29    7   13    1    0    0 1013    2    0
8     78    0    0    0    0    9   15    6  871    2
9      3   33    0    2   20  270    0   12   11  943
```

```r
1-mean(lda.cv.fit$class==pendigits$digit)
```

```
[1] 0.1241812
```

We find that the CV error rate is slightly bigger than the training error rate. We expect that CV error rate is bigger than the training error, because this last tends to underestimate the AER since the data used to build the classification rule are also used to evaluate it. This evaluation method is vulnerable to over-fitting of the model to the training data, while, by using CV error rate we avoid this problem.

In fact, if we consider the performances of the classification rule on a test set, which is composed by different data with respect to the ones on which it is generated (training set), the evaluation of the model is done on new data. Therefore, with leave-one-out cross validation, if we consider a data set of n observations, we fit (train) the classification rule n times, and at each time the test set is given by the holdout observation, so we test the model for each observation since the aim is to estimate the misclassification error of future observation and not the one that we already have.

Let n^h_{kM} be the number of holdout observations misclassified in group k, then the estimates of the conditional misclassification probabilities $P(X \in R_2|G = 1)$ and $P(X \in R_1|G = 2)$ are given by

$$P(X \in \widehat{R_2|}G = 1) = \frac{n^h_{1M}}{n_1}, \qquad P(X \in \widehat{R_1|}G = 2) = \frac{n^h_{2M}}{n_2}$$

while the AER is estimated by the total proportion of misclassified, or cross-validation error rate (CV), that is

$$CV\ error = \frac{n^h_{1M} + n^h_{2M}}{n_1 + n_2}$$

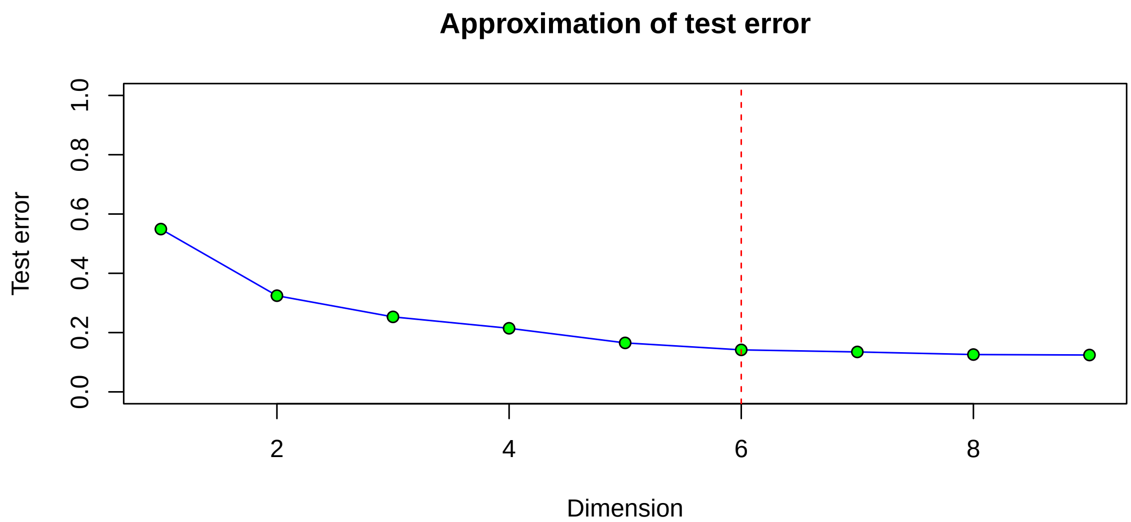
Point 4

Compute the 44-fold cross validation error for each reduced-rank LDA classifier, including full-rank LDA, by using the partition of the observations provided by the variable groupCV below. Plot the error curve against the number of discriminant variables. What classifier do you prefer? Comment.

```
groupCV<-rep(1:44, each=250)
groupCV<-groupCV[1:length(pendigits$digit)]

app.err <- rep(0, 9)
g <- floor(nrow(pendigits)/44) + 1

for(i in 1:44)
{
  test <- pendigits[which(groupCV==i),]
  test <- test[is.na(test[,1]) == FALSE,]
  train <- pendigits[-which(groupCV==i),]
  lda.fit.44 <- lda(digit~., data=train)
  for(j in 1:9)
  {
    lda.pred.44 <- predict(lda.fit.44, test, dimen = j)
    test.err <- 1 - mean(lda.pred.44$class == test$digit)
    app.err[j] <- app.err[j] + 1/44*test.err
  }
}

app.err
```

```
[1] 0.5491300 0.3244313 0.2530075 0.2145289 0.1652261 0.1416476 0.1347355
[8] 0.1259144 0.1242810
```

```
plot(app.err, type = "l", col = "blue", xlab = "Dimension",
     ylab = "Test error",ylim = c(0,1), main = "Approximation of test error")
points(1:9, app.err, col = "black", asp = 1, bg = "green", pch = 21)
abline(v=6, lty=2, col="red")
```



By looking the test error plot, we can observe that we have a significant reduction (in the test error) between 5 and 6 discriminant (≈ 0.5), while between 6 and 7 we only get a reduction of only 0.2 and we also archive a dimensionality reduction, which is one of the objective of LDA.

Moreover, if we consider the other differences, they are even lower, therefore we decide to retain 6 discriminant.

Point 5 (Optional)

Find a classification rule that improves on the CV error rate estimates found before. Feel free to use any classification method, even one not covered in class.

A random forest consists of a large number of individual decision trees that operate as an ensemble, each tree in the forest returns a class prediction and the class with the most votes becomes our model's prediction. The reason for which the random forest model works very well is that a large number of relatively uncorrelated trees (models) operate as a committee which will outperform any of the individual constituent models, since uncorrelated models can produce ensemble predictions that are more accurate than any of the individual predictions and this is because the trees protect each other from their individual errors.

Therefore, while some trees may be wrong, many other trees will be right, so as a group the trees are able to move in the correct direction. Moreover, while decisions trees are very sensitive to the data they are trained on — small changes to the training set can result in significantly different tree structures. Random forest takes advantage of this by allowing each individual tree to randomly sample from the dataset with replacement, resulting in different trees. This process is known as Bagging (Bootstrap Aggregation).

In spite of that we have decided to use a random forest model with a 44-fold cross validation since it will reduce the probability of having a model that generates overfitting.

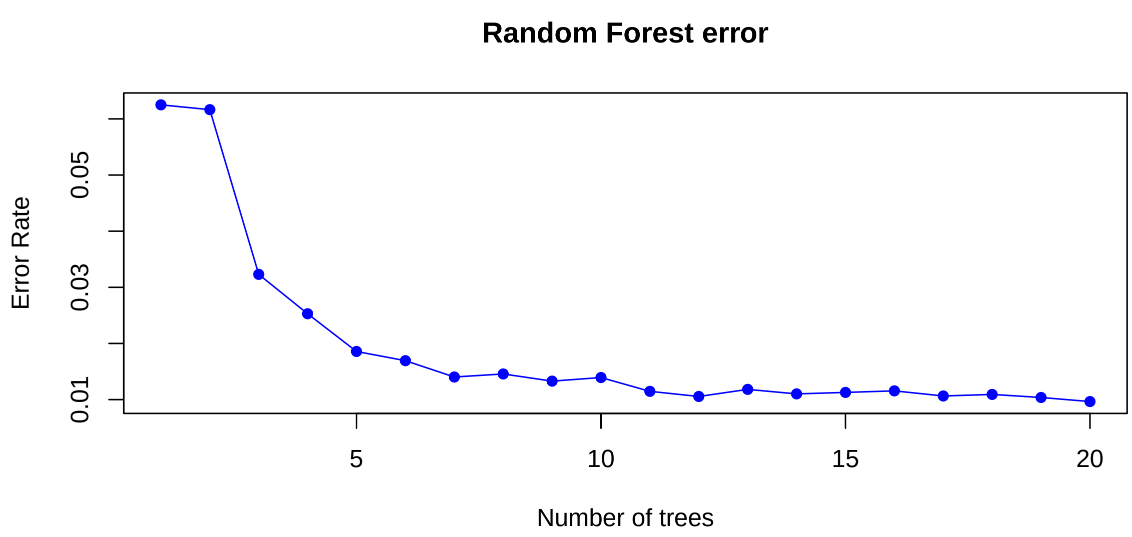
```
set.seed(123)
rf.pendigits<-data.frame(pendigits,groupCV)
f.tree<-20
max.tree<-c(1:f.tree)

err.index<-1
error.rf<-rep(0,44)
RF_Error<-c()

ptm<-rep(0,f.tree)


for (num.tree in max.tree)
{
  ptm[num.tree]<-proc.time()
  for (j in 1:44)
    {
    rf.fit=randomForest(as.factor(digit) ~ ., data=rf.pendigits[which(groupCV!=j),]
                        ,ntree=num.tree)
    prediction=predict(rf.fit, newdata=rf.pendigits[which(groupCV==j),])
```

```
    error.rf[j]=1-mean(prediction==rf.pendigits[which(groupCV==j),17])
  }

ptm[num.tree]<-proc.time()-ptm[num.tree]#costo computazionale

RF_Error[err.index]<-mean(error.rf)


 err.index=err.index+1
}
```

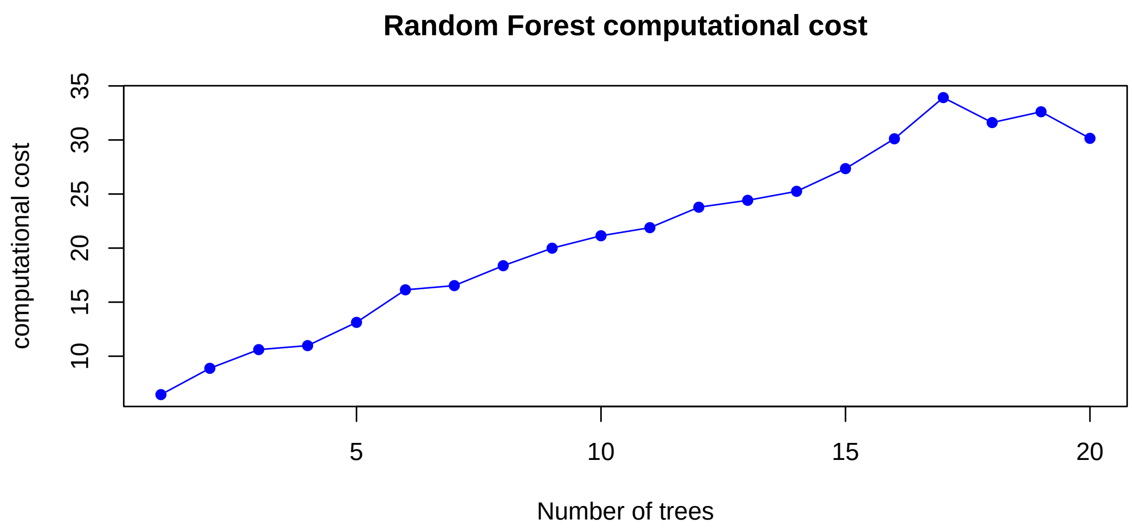
```
RF_Error
```

```
 [1] 0.06250263 0.06164463 0.03230278 0.02529376 0.01857250 0.01693313
 [7] 0.01402705 0.01456649 0.01329376 0.01393013 0.01147258 0.01055748
[13] 0.01182119 0.01101503 0.01128174 0.01155748 0.01064538 0.01092111
[19] 0.01037866 0.00965139
```

```
plot(x=max.tree,y=RF_Error,main="Random Forest error",
     xlab="Number of trees",ylab="Error Rate", pch=16,col='blue')
lines(x=max.tree,y=RF_Error,col="blue")
```



```r
plot(x=max.tree,y=ptm,main="Random Forest computational cost",
     xlab="Number of trees",ylab="computational cost", pch=16,col='blue')
lines(x=max.tree,y=ptm,col="blue")
```



We generated different forests with an increasing number of trees and evaluated their CV error.

What we can observe is that while CV error rate drops drastically by adding more trees to our forest, the computational cost increases linearly, therefore we decided to use a random forest with 10 trees by looking at the CV error plot, computational cost plot and the CV error table.