R Notebook

Code

Importation des données :

Hide

```
library(readr)
library(dplyr)
Data=read_csv("Notes.csv") #On récupère le csv dans la variable Data qui contiendra TOUTES le
s données
```

```

-- Column specification --------------------------------------------------------
------------------------------------------------
cols(
  Eleve = col_character(),
  Math = col_double(),
  Physique = col_double(),
  HistGeo = col_double(),
  Litterature = col_double(),
  EPS = col_double(),
  Genre = col_character()
)
```

Hide

```
typeof(Data$Math)
```

```
[1] "double"
```

Hide

```
NoteGenre=Data[2:7] #On retire seulement le nom des élèves
Notes=Data[2:6] #On retire la première colonne avec le nom des élèves pour ne se focaliser qu
e sur les notes
Data
```

Eleve <chr>	Math <dbl>	Physique <dbl>	HistGeo <dbl>	Litterature <dbl>	EPS <dbl>	Genre <chr>
Eleve 1	15	17	12	12	13	GARCON
Eleve 2	12	10	11	11	14	GARCON
Eleve 3	7	7	13	12	10	GARCON
Eleve 4	10	11	17	16	9	FILLE
Eleve 5	6	8	10	9	17	GARCON
Eleve 6	7	5	8	7	18	GARCON
Eleve 7	15	14	10	12	8	GARCON
Eleve 8	2	3	6	5	17	GARCON

Eleve <chr>	Math <dbl>	Physique <dbl>	HistGeo <dbl>	Litterature <dbl>	EPS <dbl>	Genre <chr>
Eleve 9	15	13	14	14	12	FILLE
Eleve 10	17	18	16	18	14	FILLE

1-10 of 60 rows Previous **1** 2 3 4 5 6 Next

Separation de chaque matière en vecteurs :

Hide

```
NotesMath=as.numeric(unlist(Notes["Math"]))
NotesPhysique=as.numeric(unlist(Notes["Physique"]))
NotesHistGeo=as.numeric(unlist(Notes["HistGeo"]))
NotesLitterature=as.numeric(unlist(Notes["Litterature"]))
NotesEPS=as.numeric(unlist(Notes["EPS"]))
```

1. Matrices de Covariance/Correlation et regression linéaire

Le but est de trouver une correlation entre les notes des differentes matières Affichage du dataset puis de la matrice de covariance

Hide

```
cov(Notes)
```

```
                Math   Physique   HistGeo Litterature        EPS
Math        21.812429 20.7615819  9.639548   8.3661017 -1.0734463
Physique    20.761582 21.6768362 11.063277   9.1457627 -0.5480226
HistGeo      9.639548 11.0632768 13.765819   9.8457627 -1.5522599
Litterature  8.366102  9.1457627  9.845763  10.5525424 -0.6186441
EPS         -1.073446 -0.5480226 -1.552260  -0.6186441 14.0437853
```

On préfèrera la matrice de correlation : proche de -1 = variables opposées, proche de 0 = variables non corrélées, proches de 1 = variables identiques

Hide

```
cor(Notes)
```

```
                 Math    Physique    HistGeo Litterature         EPS
Math         1.00000000  0.95479574  0.5562930  0.55143277 -0.06133187
Physique     0.95479574  1.00000000  0.6404494  0.60470482 -0.03140931
HistGeo      0.55629303  0.64044937  1.0000000  0.81690167 -0.11164038
Litterature  0.55143277  0.60470482  0.8169017  1.00000000 -0.05081831
EPS         -0.06133187 -0.03140931 -0.1116404 -0.05081831  1.00000000
```

On remarque qu'il y a une forte correlation entre les maths et la physique, ainsi qu'entre l'histoire/geo et le français d'une moindre manière. Il n'y a par contre une correlation d'aucune matière avec l'EPS.

Grace à la régression linéaire, on peut tenter d'estimer la note d'EPS par exemple en fonction des autres notes.

Hide

```
NotesregEPS=lm(EPS~Math+Physique+HistGeo+Litterature, data=Data)
summary(NotesregEPS)
```

```

Call:
lm(formula = EPS ~ Math + Physique + HistGeo + Litterature, data = Data)

Residuals:
    Min      1Q  Median      3Q     Max 
-11.663  -1.342   0.400   1.643   6.842 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)  13.0743     1.8368   7.118 2.42e-09 ***
Math         -0.4029     0.3690  -1.092    0.280    
Physique      0.4503     0.3990   1.128    0.264    
HistGeo      -0.3004     0.2510  -1.196    0.237    
Litterature   0.1508     0.2699   0.559    0.579    
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 3.803 on 55 degrees of freedom
Multiple R-squared:  0.03978,   Adjusted R-squared:  -0.03005 
F-statistic: 0.5696 on 4 and 55 DF,  p-value: 0.6857
```

On peut établir une fonction de regression linéaire qui est NoteEPS = 13,0743 - 0,4 x NoteMath+0,4 x NotePhysique - 0,3 x NoteHistGeo + 0,15 x NoteLitterature. Ce résultat montre qu'en fait la note d'EPS est relativement peu dépendante des autres notes. Il est donc difficile de prédire une note d'EPS. Par contre si nous refaisons une regression linéaire pour les maths par exemple :

Hide

```
NotesregMath=lm(Math~EPS+Math+Physique+HistGeo+Litterature, data=Data)
```

```
la r玫锜ponse est apparue dans le membre de droite et y a 玫锜t玫锜 玫锜limin玫锜eprobl玫锆me a
vec le terme 2 dans model.matrix : aucune colonne n'est assign玫锜e
```

Hide

```
summary(NotesregMath)
```

```
Call:
lm(formula = Math ~ EPS + Math + Physique + HistGeo + Litterature,
    data = Data)

Residuals:
    Min      1Q  Median      3Q     Max
-1.9518 -1.0781 -0.4701  1.0166  2.3816

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  1.60516    0.89467   1.794   0.0783 .
EPS         -0.05266    0.04823  -1.092   0.2796
Physique     1.01400    0.05098  19.891   <2e-16 ***
HistGeo     -0.17099    0.08900  -1.921   0.0599 .
Litterature  0.07044    0.09740   0.723   0.4726
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1.375 on 55 degrees of freedom
Multiple R-squared:  0.9192,    Adjusted R-squared:  0.9133
F-statistic: 156.4 on 4 and 55 DF,  p-value: < 2.2e-16
```

Nous remarquons que la correlation est bien là entre les Maths et la Physique, mais aussi dans une moindre mesure avec les autres matières. Est ce que cette regression linéaire nous permet de prédire la note en Maths ?

Hide

```
fitted(NotesregMath) #Nous montre la prédiction pour chaque élève
```

```
       1         2         3         4         5         6         7         8         9
10        11        12        13        14
16.951825  9.901737  6.798856 10.505290  7.745870  4.852325 14.515134  3.078104 12.747392 17.
651811  9.111394  6.300231 13.524735 17.646386
       15        16        17        18        19        20        21        22        23
24        25        26        27        28
 4.886832 17.794831 14.835623 14.417380  8.761840 12.983393 16.881389  9.901737  6.798856 10.
434854  7.745870  4.852325 14.515134  3.078104
       29        30        31        32        33        34        35        36        37
38        39        40        41        42
12.676956 15.523263  9.111394  6.300231 13.595171 17.746941  4.886832 15.766838 13.821626 14.
417380  8.761840 12.983393 16.951825  9.901737
       43        44        45        46        47        48        49        50        51
52        53        54        55        56
 6.798856 10.505290  7.745870  4.957652 14.515134  3.078104 12.676956 17.681930  9.111394  6.
300231 13.595171 15.618392  4.886832 17.794831
       57        58        59        60
14.835623 14.417380  8.832276 12.983393
```

Hide

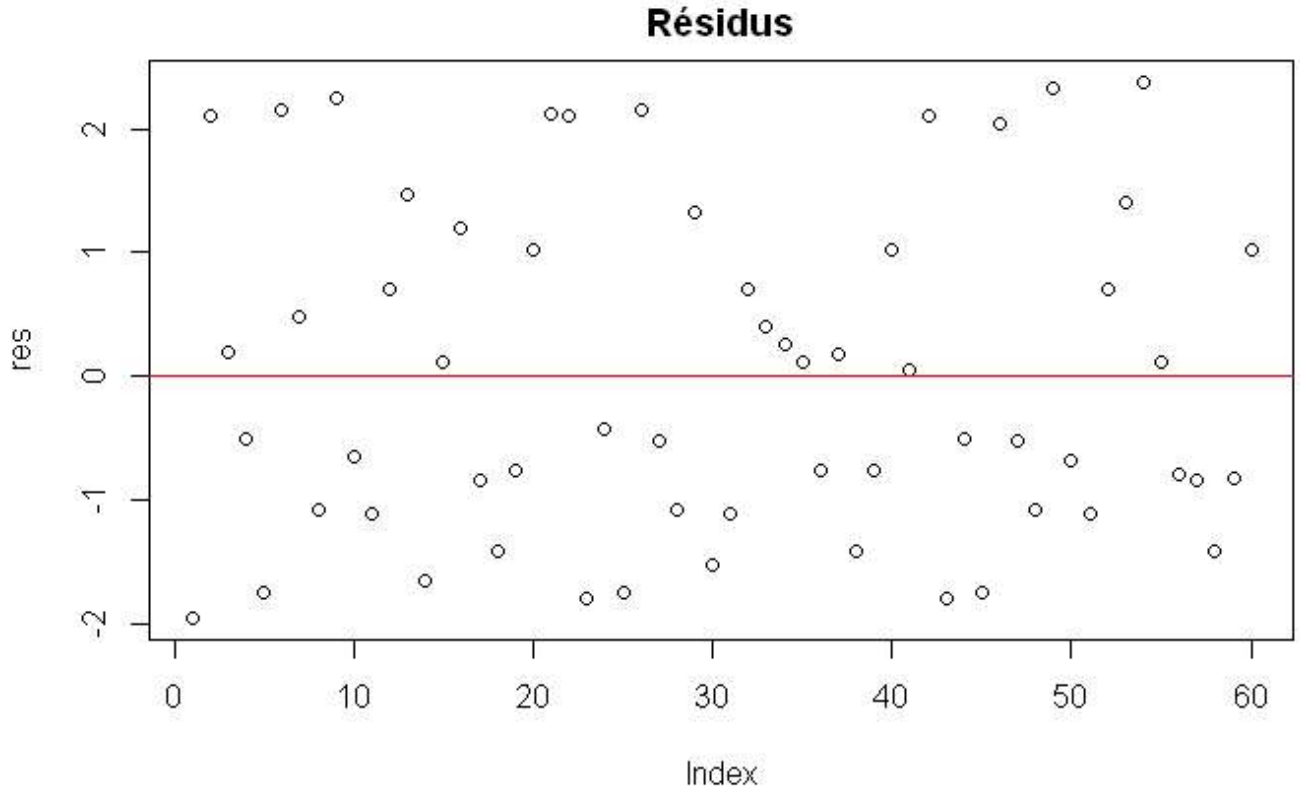
```
resid(NotesregMath) #Nous montre l'erreur pour chaque élève (Note réelle - Note estimée)
```

```
         1           2           3           4           5           6           7
8           9          10          11          12
-1.95182497  2.09826308  0.20114362 -0.50528969 -1.74587012  2.14767543  0.48486565 -1.078103
71  2.25260826 -0.65181139 -1.11139430  0.69976945
         13          14          15          16          17          18          19
20          21          22          23          24
 1.47526453 -1.64638611  0.11316824  1.20516854 -0.83562315 -1.41738023 -0.76183973  1.016607
19  2.11861090  2.09826308 -1.79885638 -0.43485381
         25          26          27          28          29          30          31
32          33          34          35          36
-1.74587012  2.14767543 -0.51513435 -1.07810371  1.32304414 -1.52326284 -1.11139430  0.699769
45  0.40482866  0.25305929  0.11316824 -0.76683752
         37          38          39          40          41          42          43
44          45          46          47          48
 0.17837382 -1.41738023 -0.76183973  1.01660719  0.04817503  2.09826308 -1.79885638 -0.505289
69 -1.74587012  2.04234771 -0.51513435 -1.07810371
         49          50          51          52          53          54          55
56          57          58          59          60
 2.32304414 -0.68193013 -1.11139430  0.69976945  1.40482866  2.38160784  0.11316824 -0.794831
46 -0.83562315 -1.41738023 -0.83227561  1.01660719
```

On obtiens une estimation plutot correcte, à 2 points près. Afin de valider notre estimation, on peut vérifier que les résidus sont répartis de manière homogène autours de 0 :

Hide

```
res<-resid(NotesregMath)
plot(res,main="Résidus")
abline(h=0,col="red")
```



On peut aussi calculer la moyenne des résidus :

Hide

```
mean(res)
```

```
[1] 8.326673e-18
```

On voit qu'elle est très proche de 0 et donc que les résidus sont répartis de manière homogène. D'autre part, summary(NotesregMath) nous montre une p-value très petite, ce qui indique que au test de significativité globale on peut rejeter l'hypothèse H0 et dire que le modèle est globalement significatif. On remarque que c'est moins le cas pour le modèle concernant l'EPS.

2. Visualisations 2.a Histogramme

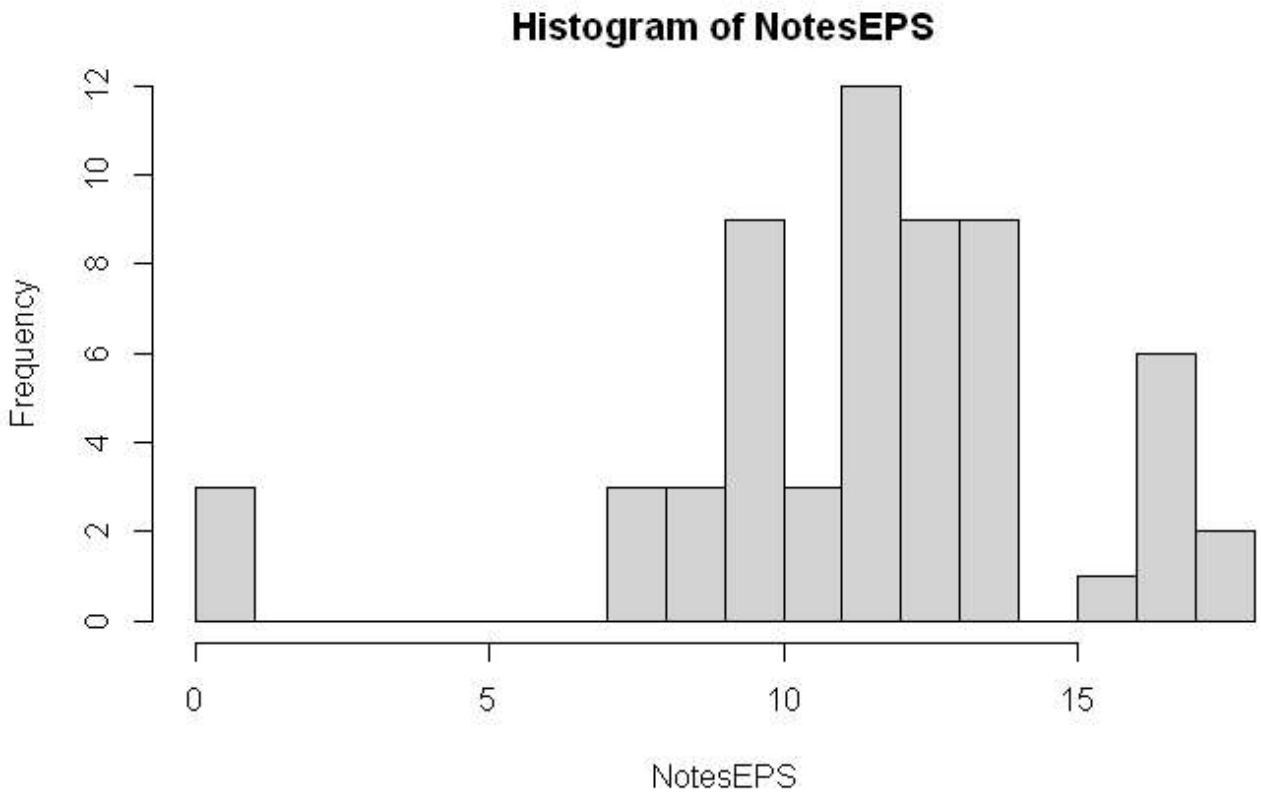
Hide

```
NotesEPS
```

```
 [1] 14 14 14  9 12  9  9 12 12 10 10 12 12 12 10 12 12 12 13 13 13 13 13  8  8  8 10 10 10 1
3 10 10 10 14 14 14 13 13 13 12 12 12 14 14 14 17
[47] 17 17 11 11 11 18 16 18  0  0  0 17 17 17
```

Hide

```
hist(NotesEPS, breaks=20)
```



On peut afficher la PDF (fonction de densité) :

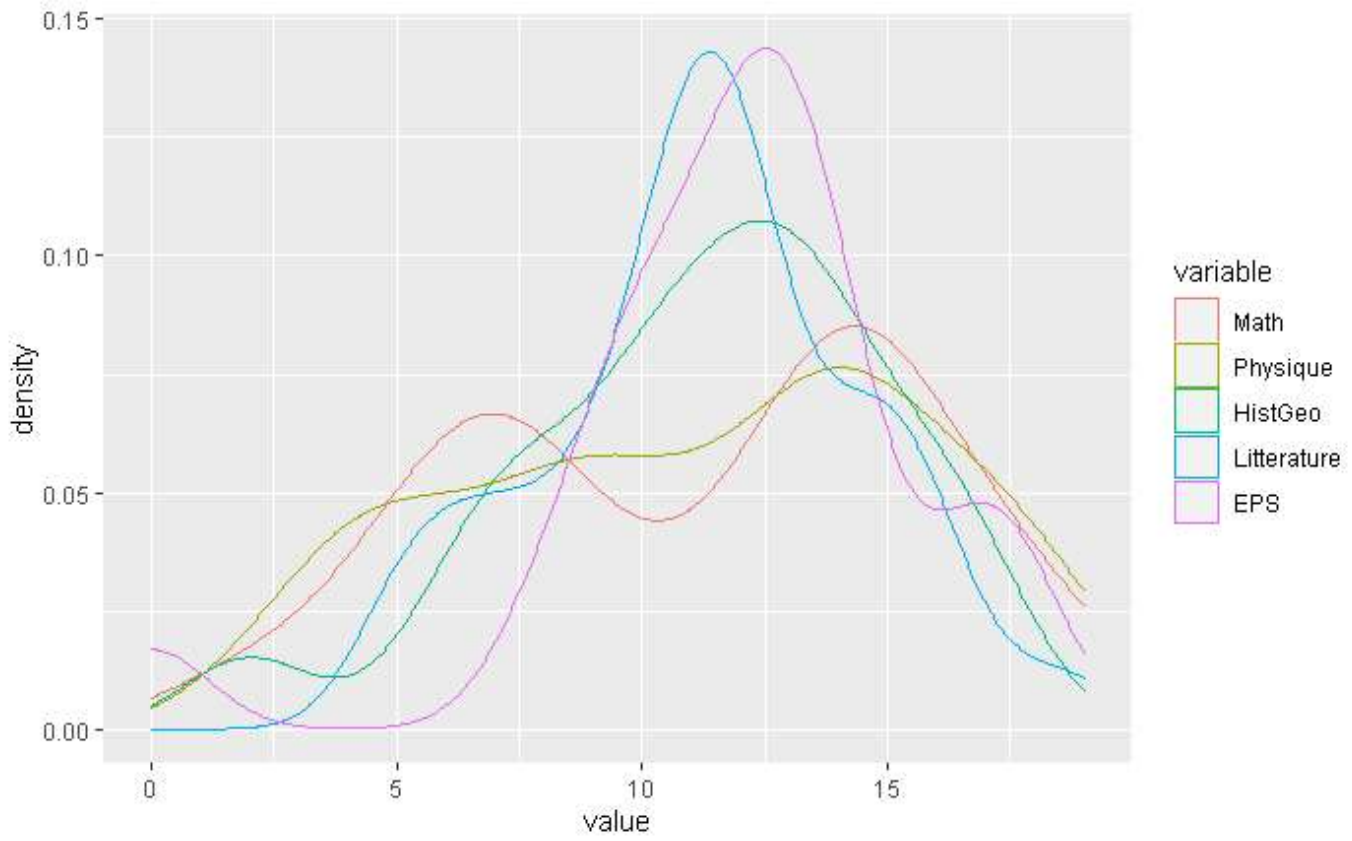
Hide

```
library(ggplot2)
library(reshape2)
Notes.plot = melt(Notes)
```

```
No id variables; using all as measure variables
```

Hide

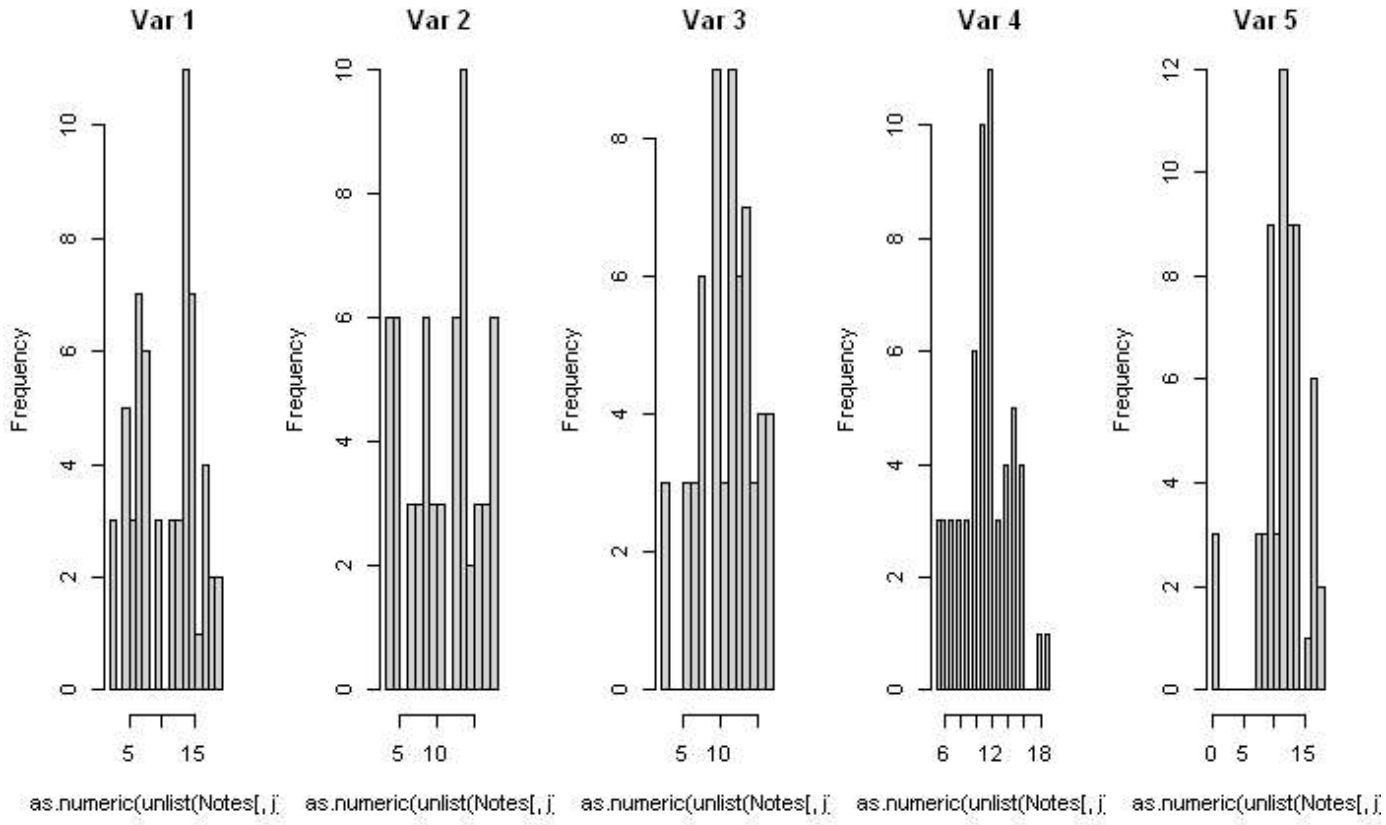
```
p <- ggplot(aes(x=value, colour=variable), data=Notes.plot)
p + geom_density()
```



Pour afficher un ensemble d'histogrammes pour chaque dimension on utilisera le code :

Hide

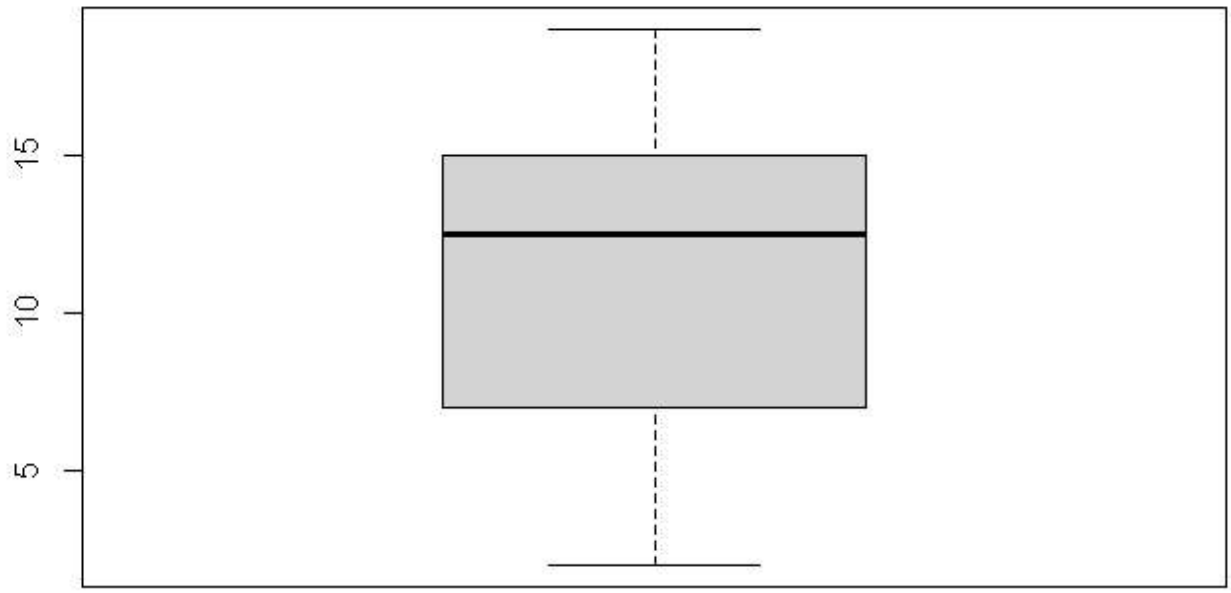
```
par(mfrow=c(1,5))
for (j in 1:5) hist(as.numeric(unlist(Notes[,j])),breaks=20,main = paste("Var",j))
```



2.b Box Plot Sur une seule dimension (une seule variable aléatoire)

Hide

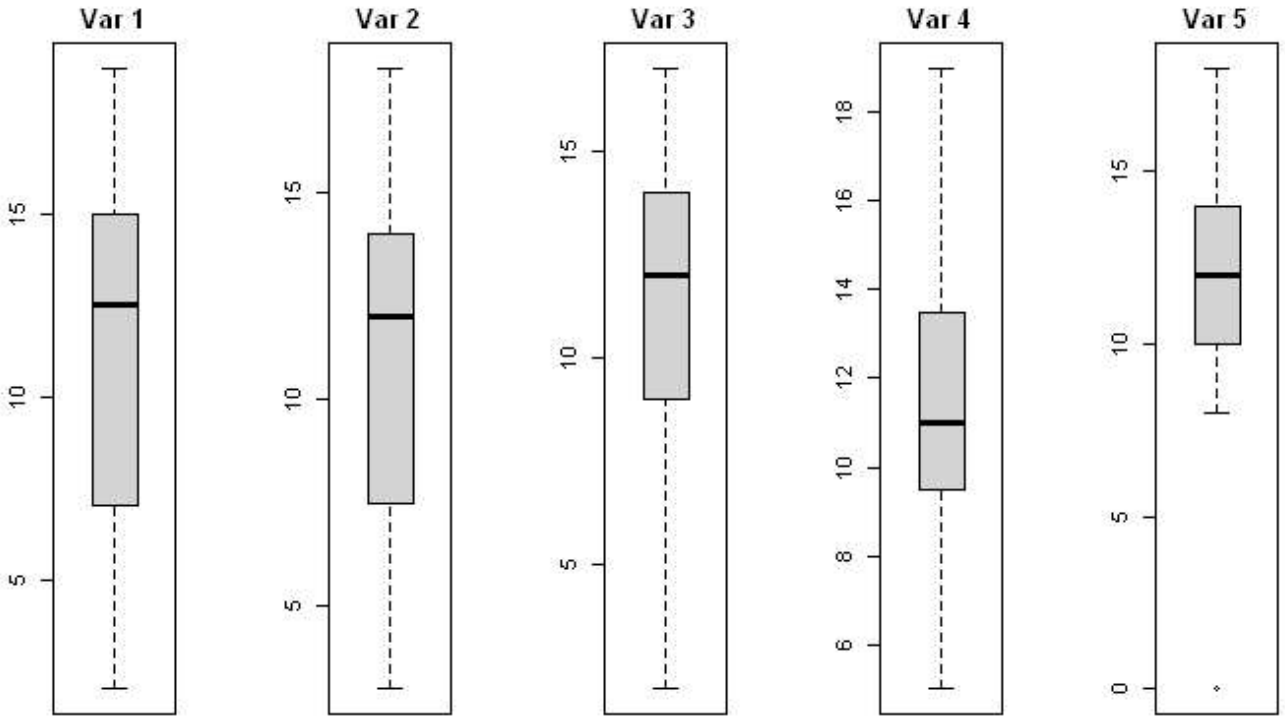
```
boxplot(NotesMath)
```



Sur plusieur dimension, il est possible d'afficher un boxplot par dimension :

Hide

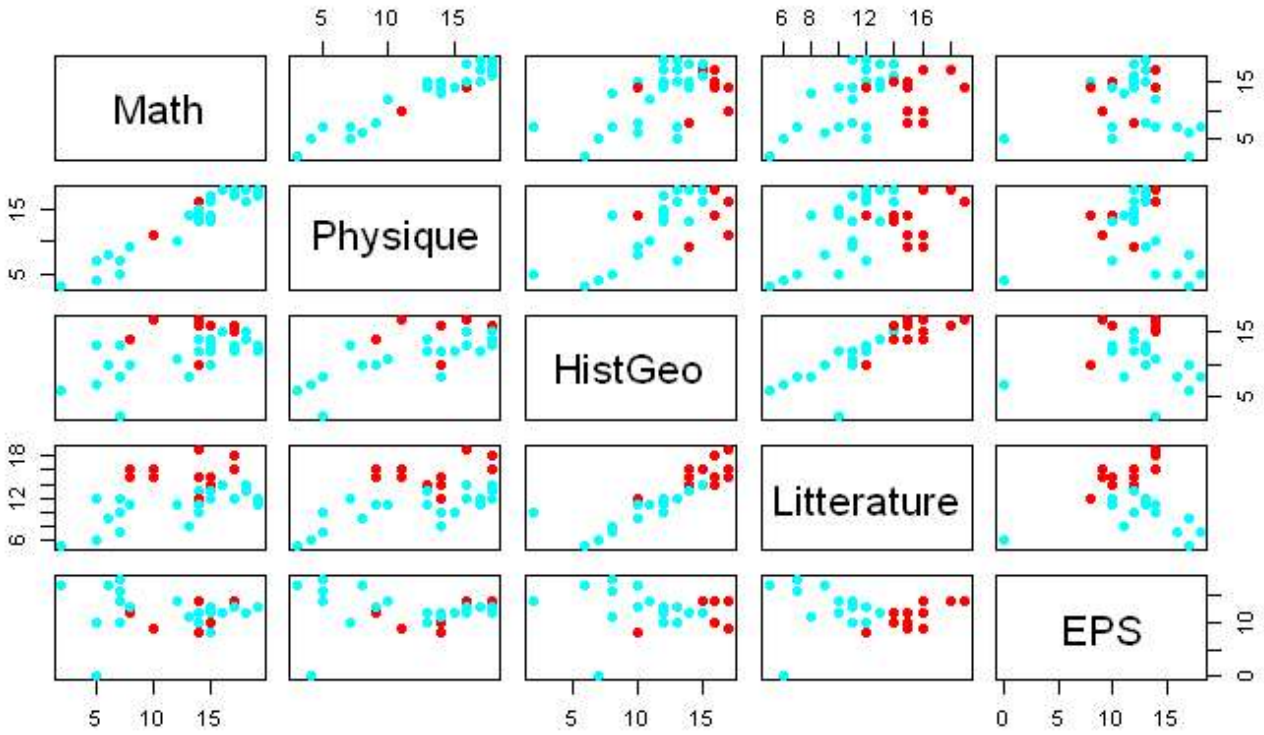
```
par(mfrow=c(1,5))
for (j in 1:5) boxplot(as.numeric(unlist(Notes[,j])), main = paste("Var",j))
```



2. ScatterPlot Permet d'afficher une valeur de colonne en fonction d'une autre. Sur les données sur plusieur dimensions on peut les afficher sous forme matricielle :

Hide

```
zfac <- factor(Data$Genre)
mescouleurs <- rainbow(length(levels(zfac)))
plot(Notes,pch = 19,col=mescouleurs[zfac])
```



On peut retrouver la correlation ici entre les maths et la physique, ainsi qu'entre l'histoire geographie et la litterature

Note : Au cas où nous aurions besoin d'étudier la moyenne en fonction du genre :

Hide

```
moyenne=c()

for (i in 1:length(row.names(Notes))){
moyenne<-c(moyenne,sum(Notes[i,])/length(Notes[i,]))
}

eleves = c(Data["Eleve"])
genre = c(Data["Genre"])
MoyenneNotes = data.frame(eleves,moyenne,genre)
MoyenneNotes
```

Eleve	moyenne	Genre
<chr>	<dbl>	<chr>
Eleve 30	16.0	FILLE
Eleve 10	16.6	FILLE
Eleve 50	16.0	FILLE
Eleve 4	12.6	FILLE
Eleve 59	11.8	FILLE
Eleve 44	12.6	FILLE
Eleve 24	12.4	FILLE

Eleve <chr>	moyenne <dbl>	Genre <chr>
Eleve 19	11.6	FILLE
Eleve 39	11.6	FILLE
Eleve 53	14.0	FILLE

1-10 of 60 rows Previous **1** 2 3 4 5 6 Next

3. Machine Learning

3.1. Procédures de ML supervisé (X et Y sont connus)

3.1.a KNN

Hide

```
library(class)
X = Data[,2:6]
Y = as.data.frame(Data[,7])
learn = sample(1:60,50) # sample 60 indexes of individuals that will be used in the learning
 set
train = X[learn,] # 50 obs for learning
cl = Y[learn,]
test = X[-learn,] # The remaining 10 obs for validation
Yreel = Y[-learn,]
?knn
Prediction = knn(train,test,cl,k = 4)
Resultat=data.frame(Data[-learn,],Prediction)
Resultat
```

Eleve <chr>	Math <dbl>	Physique <dbl>	HistGeo <dbl>	Litterature <dbl>	E… <dbl>	Genre <chr>	Prediction <fctr>
Eleve 30	14	16	17	19	14	FILLE	FILLE
Eleve 13	15	14	16	14	10	FILLE	FILLE
Eleve 56	17	18	13	12	13	GARCON	GARCON
Eleve 47	14	14	10	12	8	GARCON	GARCON
Eleve 20	14	13	12	11	10	GARCON	GARCON
Eleve 51	8	9	10	11	13	GARCON	GARCON
Eleve 57	14	15	12	10	12	GARCON	GARCON
Eleve 38	13	14	8	8	11	GARCON	GARCON
Eleve 35	5	4	7	6	0	GARCON	GARCON
Eleve 48	2	3	6	5	17	GARCON	GARCON

1-10 of 10 rows

Hide

```
NA
```

On remarquera que les erreurs sont dues à un contre exemple. En effet les garçons sont sensés être meilleurs en math et physique et les filles meilleures en histoire et français. Un garçon mauvais en maths/physique et bon en histoire/litterature sera prédit comme fille.

On peut calculer la marge d'erreur :

Hide

```
err = sum(Prediction != Yreel) / length(Yreel)
err
```

```
[1] 0.2
```

Tentons d'améliorer ce résultat en utilisant la méthode V-fold cross validation (Plusieurs periodes d'apprentissage/test avec des échantillons différents)

Hide

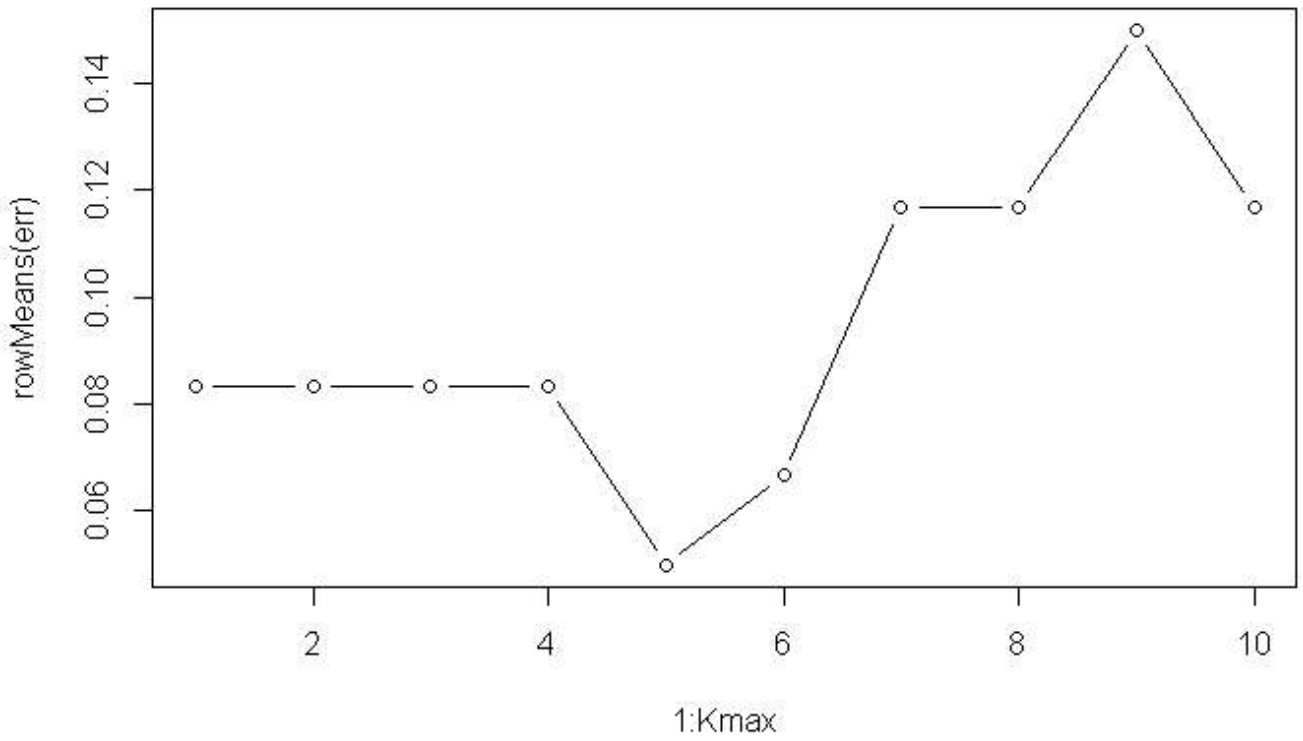
```
V = 20
fold = rep(1:V,nrow(X)/V)
for (v in 1:V){
  learn = which(fold != v)
  Xl = X[learn,] # 100 obs for learning
  Yl = Y[learn,]
  Xv = X[-learn,] # The remaining 50 obs for validation
  Yv = Y[-learn,]
  f = knn(Xl,Xv,Yl,k = 5)
  err[v] = sum(f != Yv) / length(Yv)
}
#err
mean(err)
```

```
[1] 0.09166667
```

Nous calculons ici l'erreur moyenne sur 20 tests differents à K=3. Est ce 3 est le bon paramètre ? Utilisons la validation croisée avec un K de 2 à 10 pour le vérifier.

Hide

```
Kmax = 10
V = 20
err = matrix(NA,Kmax,20)
fold = rep(1:V,nrow(X)/V)
for (v in 1:V){
  learn = which(fold != v)
  Xl = X[learn,]
  Yl = Y[learn,]
  Xv = X[-learn,]
  Yv = Y[-learn,]
  for (k in 1:Kmax){
    f = knn(Xl,Xv,Yl,k)
    err[k,v] = sum(f != Yv) / length(Yv)
  }
}
plot(1:Kmax,rowMeans(err),type='b')
```



Il semble que le K ayant le taux d'erreur le plus bas est de 5. Nous noterons ce parametre K* (ou Kstar), que nous pourrons utiliser pour prédire le genre d'un élève en fonction de ses notes.

Enfin, pour calculer l'estimation statistique de l'erreur globale, on applique la formule erreur moyenne +/- écart type :

Hide

```
V = 20
fold = rep(1:V,nrow(X)/V)
for (v in 1:V){
  learn = which(fold != v)
  Xl = X[learn,] # 100 obs for learning
  Yl = Y[learn,]
  Xv = X[-learn,] # The remaining 50 obs for validation
  Yv = Y[-learn,]
  f = knn(Xl,Xv,Yl,k = 5)
  err[v] = sum(f != Yv) / length(Yv)
}
StatGlobErr=c((mean(err)-sd(err)),(mean(err)),(mean(err)+sd(err)))
StatGlobErr
```

```
[1] -0.05754492  0.09166667  0.24087826
```

3.1.b SVM Tout d'abord on reformatte le data frame Data pour ne garder que 4 dimensions : - La note en Math - La note en Français - Son genre Nous allons étudier le genre en fonction de la note en Math et Français.

Hide

```
library(e1071)
DF = data.frame(Data$Math, Data$Litterature, as.factor(Data$Genre))
DF
```

Data.Math <dbl>	Data.Litterature <dbl>	as.factor.Data.Genre. <fctr>
14	19	FILLE
17	18	FILLE
17	16	FILLE
10	16	FILLE
8	16	FILLE
10	16	FILLE
10	15	FILLE
8	15	FILLE
8	15	FILLE
15	15	FILLE

1-10 of 60 rows Previous **1** 2 3 4 5 6 Next

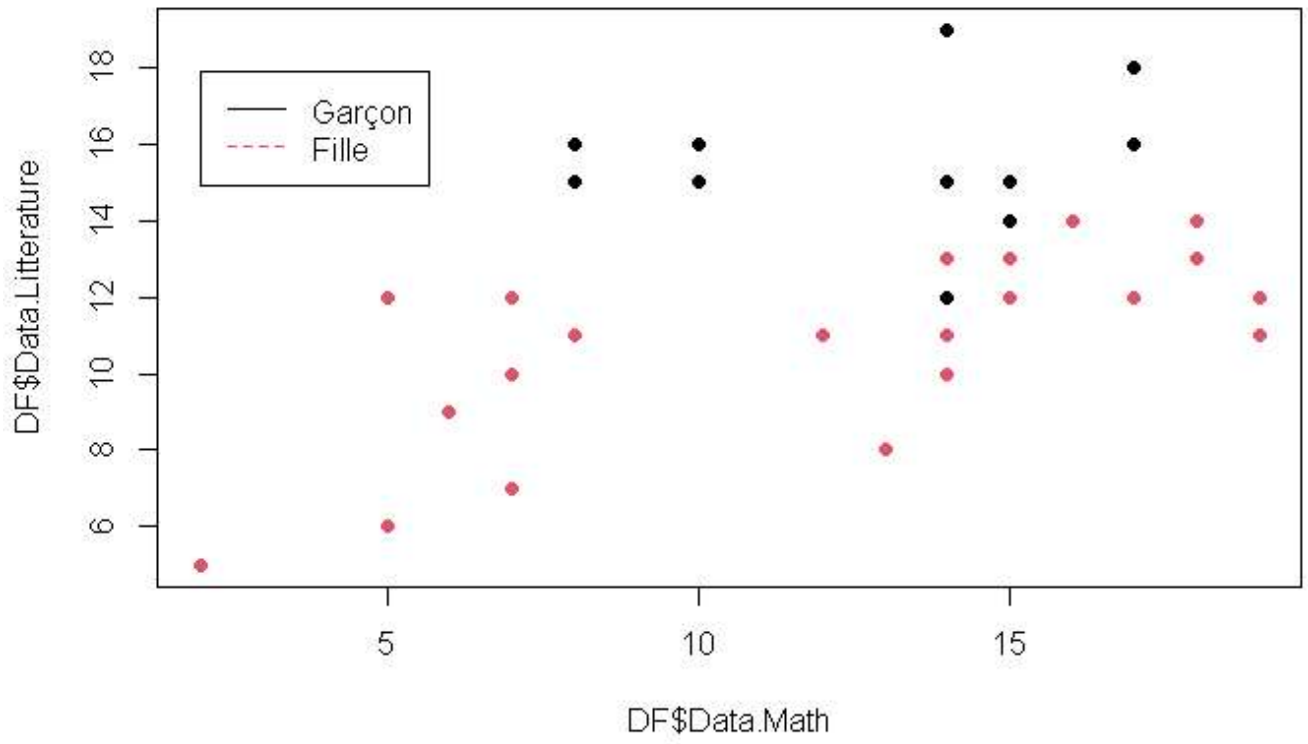
Hide

```
summary(DF)
```

```
  Data.Math     Data.Litterature as.factor.Data.Genre.
 Min.   : 2.00   Min.   : 5.00    FILLE :14
 1st Qu.: 7.00   1st Qu.: 9.75    GARCON:46
 Median :12.50   Median :11.00
 Mean   :11.13   Mean   :11.30
 3rd Qu.:15.00   3rd Qu.:13.25
 Max.   :19.00   Max.   :19.00
```

Hide

```
plot(DF$Data.Math,DF$Data.Litterature, col=DF$as.factor.Data.Genre, pch=16)
legend(2,17.9,legend=c("Garçon","Fille"),col=1:length(DF$as.factor.Data.Genre),lty=1:2)
```



Construction du classificateur SVM mlin :

Hide

```
mlin <- svm(DF$as.factor.Data.Genre ~ DF$Data.Math+DF$Data.Litterature, data=DF, kernel="line
ar",scale=F)
print(mlin)
```

```

Call:
svm(formula = DF$as.factor.Data.Genre ~ DF$Data.Math + DF$Data.Litterature, data = DF, kernel
= "linear", scale = F)


Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  linear
       cost:  1

Number of Support Vectors:  9
```

Mettons en évidence les “points supports” qui détermineront notre hyperplan SVM

Hide

```
print((rownames(df))[mlin$index])
```

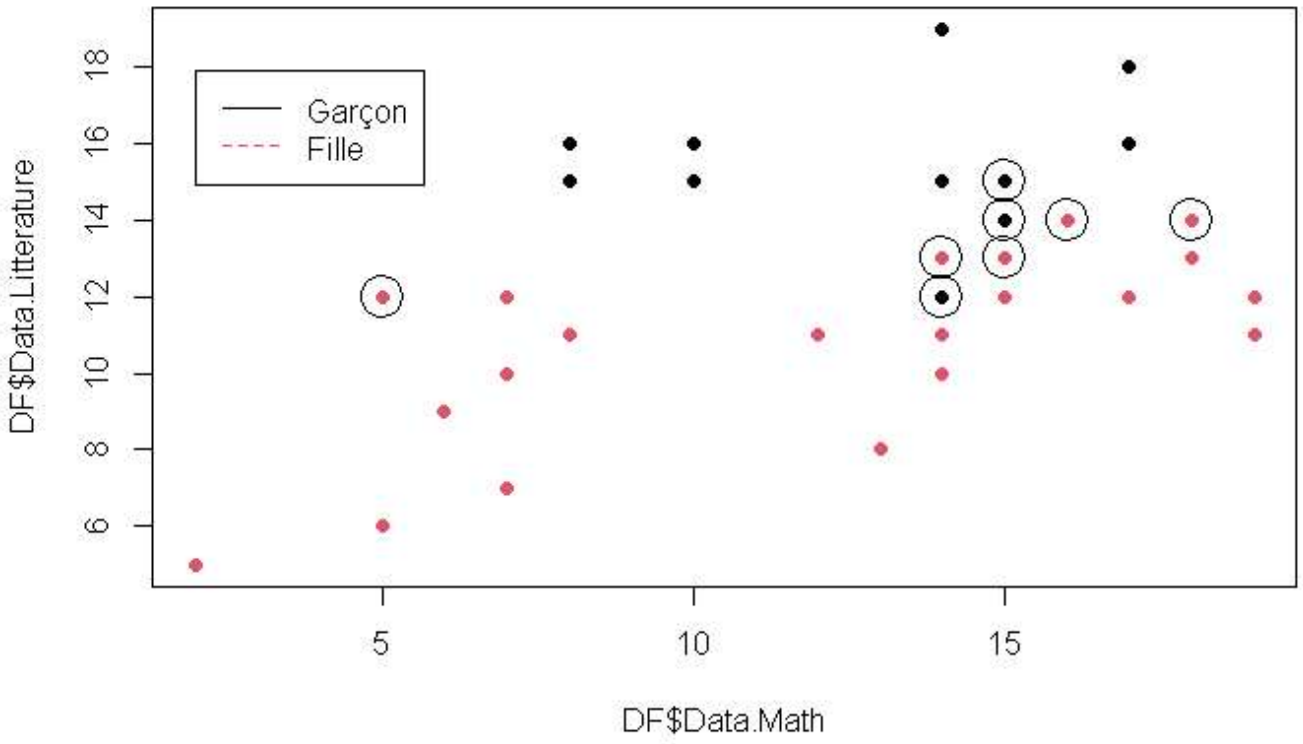
```
NULL
```

Hide

```
plot(DF$Data.Math,DF$Data.Litterature, col=DF$as.factor.Data.Genre, pch=16)
legend(2,17.9,legend=c("Garçon","Fille"),col=1:length(DF$as.factor.Data.Genre),lty=1:2)
```

Hide

```
points(DF$Data.Math[mlin$index],DF$Data.Litterature[mlin$index],cex=3,col=rgb(0,0,0))
```



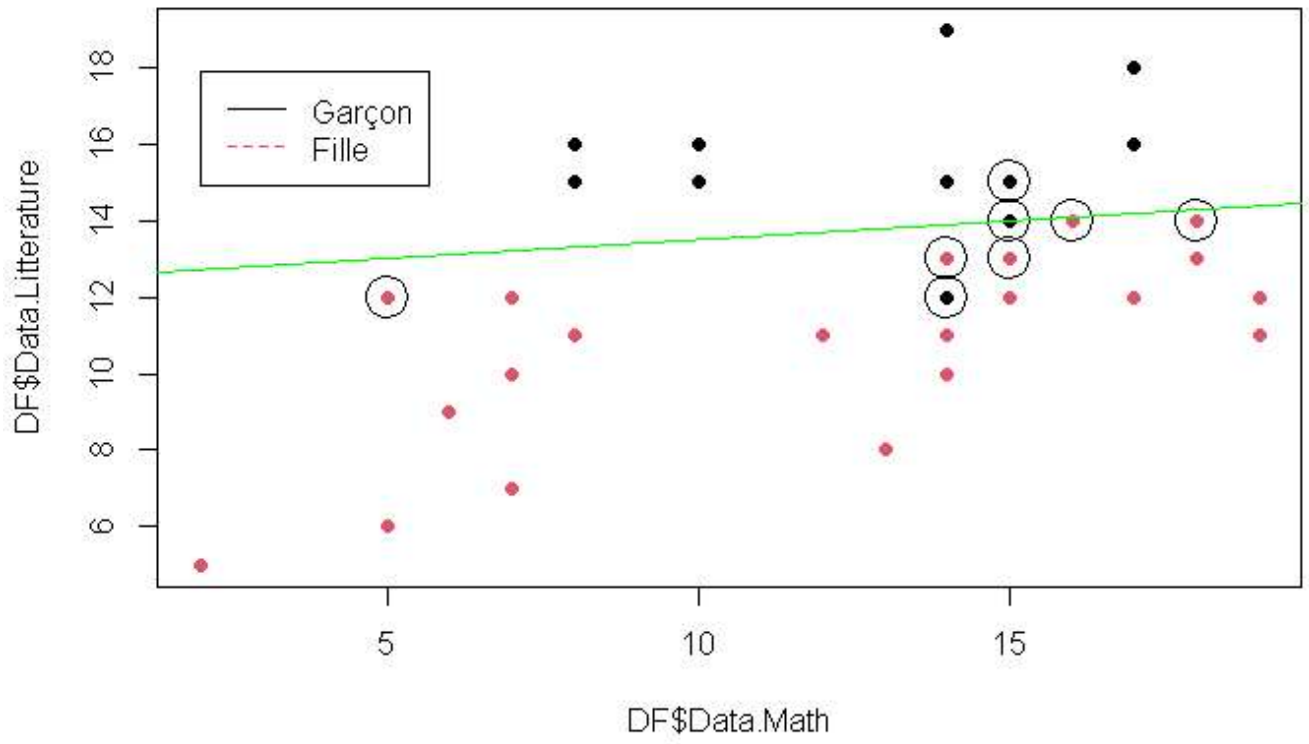
Nous pouvons maintenant représenter les frontières :

Hide

```
beta.0 <- -mlin$rho
beta.1 <- sum(mlin$coefs*DF$Data.Math[mlin$index])
beta.2 <- sum(mlin$coefs*DF$Data.Litterature[mlin$index])
plot(DF$Data.Math,DF$Data.Litterature, col=DF$as.factor.Data.Genre, pch=16)
legend(2,17.9,legend=c("Garçon","Fille"),col=1:length(DF$as.factor.Data.Genre),lty=1:2)
```

Hide

```
points(DF$Data.Math[mlin$index],DF$Data.Litterature[mlin$index],cex=3,col=rgb(0,0,0))
abline(-beta.0/beta.2,-beta.1/beta.2,col="green")
```



3.1.c LDA

Hide

```
library(MASS)
X = Data[,2:6]
Y = as.data.frame(Data[,7])
learn = sample(1:60,50) # sample 60 indexes of individuals that will be used in the learning
 set
learn = sample(1:60,50) # sample 60 indexes of individuals that will be used in the learning
 set
train = X[learn,] # 50 obs for learning
cl = Y[learn,]
test = X[-learn,] # The remaining 10 obs for validation
Yreel = Y[-learn,]
f=lda(train,cl)
f
```

```
Call:
lda(train, cl)

Prior probabilities of groups:
 FILLE GARCON
  0.24   0.76

Group means:
           Math Physique  HistGeo Litterature      EPS
FILLE  13.08333 13.41667 15.16667   15.416667 11.16667
GARCON 10.65789 10.50000 10.18421    9.947368 11.89474

Coefficients of linear discriminants:
                    LD1
Math          0.10867001
Physique      0.03558773
HistGeo      -0.04991321
Litterature -0.50448343
EPS           0.04106295
```

Hide

```
pred=predict(f,test)
sum(pred$class != Yreel) / length(Yreel)
```

```
[1] 0
```

3.2 Unsupervised learning

Le but de l'apprentissage non supervisé est de classifier une variable aléatoire sans avoir la correspondance pour apprentissage. En d'autre termes, on a X mais pas Y.

3.2.a PCA Le but ici est de réduire les dimensions d'une variable aléatoire. Cette méthode utilise la combinaison linéaire (operations matricielles, calculs de eigen values). Tout d'abord il faut "centraliser" les données, c'est à dire déplacer les axes d'origine pour qu'ils se croisent au "centre" des valeurs. Ensuite calculer une matice Sigma en fonction de la variable aléatoire "centralisée" et d'en extraire les premiers eigenvectors. On retiendra les d premiers eigenvectors comme réduction des dimensions de notre data frame.

Appliquons la pas à pas :

Hide

```
Data=read_csv("Notes.csv")
```

```
-- Column specification --------------------------------------------------------
------------------------------------------------
cols(
  Eleve = col_character(),
  Math = col_double(),
  Physique = col_double(),
  HistGeo = col_double(),
  Litterature = col_double(),
  EPS = col_double(),
  Genre = col_character()
)
```

Hide

```
X=Data[,2:5]
X
```

Math <dbl>	Physique <dbl>	HistGeo <dbl>	Litterature <dbl>
14	16	17	19
17	18	16	18
17	18	15	16
10	11	17	16
8	9	14	16
10	11	17	16
10	11	17	15
8	9	14	15
8	9	14	15
15	14	16	15

1-10 of 60 rows Previous **1** 2 3 4 5 6 Next

X contiens seulement les notes mais pas l'indication de groupe 'Garçon' ou 'Fille' La procédure manuelle est la suivante : - Calcul de Xbar grace à la fonction scale(). Xbar contiens la valeur de X-μX. Elle permet de "centrer" les axes - Calcul de Sigma = XbarT*Xbar - Calcul des eigenvalues et eigenvectors de Sigma (choix de la valeur d, le nombre de eigenvector à retenir) - Affichage de Yhat, l'estimateur de classification

Hide

```
# Première étape : calcul de Xbar
Xbar = scale(X,center = TRUE, scale = FALSE)
Xbar
```

```
           Math   Physique    HistGeo Litterature
 [1,]  2.8666667  4.8666667  5.7166667         7.7
 [2,]  5.8666667  6.8666667  4.7166667         6.7
 [3,]  5.8666667  6.8666667  3.7166667         4.7
 [4,] -1.1333333 -0.1333333  5.7166667         4.7
 [5,] -3.1333333 -2.1333333  2.7166667         4.7
 [6,] -1.1333333 -0.1333333  5.7166667         4.7
 [7,] -1.1333333 -0.1333333  5.7166667         3.7
 [8,] -3.1333333 -2.1333333  2.7166667         3.7
 [9,] -3.1333333 -2.1333333  2.7166667         3.7
[10,]  3.8666667  2.8666667  4.7166667         3.7
[11,]  2.8666667  2.8666667  4.7166667         3.7
[12,]  6.8666667  4.8666667  3.7166667         2.7
[13,]  4.8666667  6.8666667  3.7166667         2.7
[14,]  3.8666667  1.8666667  2.7166667         2.7
[15,]  3.8666667  2.8666667  4.7166667         2.7
[16,]  6.8666667  6.8666667  2.7166667         1.7
[17,]  3.8666667  1.8666667  2.7166667         1.7
[18,]  2.8666667  1.8666667  2.7166667         1.7
[19,]  5.8666667  5.8666667  0.7166667         0.7
[20,]  3.8666667  5.8666667  0.7166667         0.7
[21,]  5.8666667  6.8666667  1.7166667         0.7
[22,]  3.8666667  4.8666667  1.7166667         0.7
[23,]  7.8666667  6.8666667  1.7166667         0.7
[24,]  3.8666667  2.8666667 -1.2833333         0.7
[25,]  2.8666667  2.8666667 -1.2833333         0.7
[26,]  2.8666667  2.8666667 -1.2833333         0.7
[27,] -4.1333333 -4.1333333  1.7166667         0.7
[28,] -6.1333333 -4.1333333  1.7166667         0.7
[29,] -6.1333333 -4.1333333  1.7166667         0.7
[30,]  7.8666667  5.8666667  0.7166667        -0.3
[31,]  2.8666667  1.8666667  0.7166667        -0.3
[32,]  2.8666667  1.8666667  0.7166667        -0.3
[33,]  2.8666667  1.8666667  0.7166667        -0.3
[34,]  0.8666667 -1.1333333 -0.2833333        -0.3
[35,]  0.8666667 -1.1333333 -0.2833333        -0.3
[36,]  0.8666667 -1.1333333 -0.2833333        -0.3
[37,] -3.1333333 -2.1333333 -1.2833333        -0.3
[38,] -3.1333333 -2.1333333 -1.2833333        -0.3
[39,] -3.1333333 -2.1333333 -1.2833333        -0.3
[40,]  2.8666667  3.8666667  0.7166667        -1.3
[41,]  2.8666667  2.8666667  0.7166667        -1.3
[42,]  2.8666667  3.8666667  0.7166667        -1.3
[43,] -4.1333333 -6.1333333 -9.2833333        -1.3
[44,] -4.1333333 -6.1333333 -9.2833333        -1.3
[45,] -4.1333333 -6.1333333 -9.2833333        -1.3
[46,] -5.1333333 -3.1333333 -1.2833333        -2.3
[47,] -5.1333333 -3.1333333 -1.2833333        -2.3
[48,] -5.1333333 -3.1333333 -1.2833333        -2.3
[49,]  1.8666667  2.8666667 -3.2833333        -3.3
[50,]  1.8666667  2.8666667 -3.2833333        -3.3
[51,]  1.8666667  2.8666667 -3.2833333        -3.3
[52,] -4.1333333 -6.1333333 -3.2833333        -4.3
[53,] -4.1333333 -6.1333333 -3.2833333        -4.3
[54,] -4.1333333 -6.1333333 -3.2833333        -4.3
[55,] -6.1333333 -7.1333333 -4.2833333        -5.3
[56,] -6.1333333 -7.1333333 -4.2833333        -5.3
```

```
[57,] -6.1333333 -7.1333333 -4.2833333        -5.3
[58,] -9.1333333 -8.1333333 -5.2833333        -6.3
[59,] -9.1333333 -8.1333333 -5.2833333        -6.3
[60,] -9.1333333 -8.1333333 -5.2833333        -6.3
attr(,"scaled:center")
       Math    Physique     HistGeo Litterature
   11.13333    11.13333    11.28333    11.30000
```

Hide

```
# Deuxième étape : calcul de Sigma
Sigma = t(Xbar) %*% Xbar
Sigma
```

```
                Math  Physique  HistGeo Litterature
Math        1286.9333 1224.9333 568.7333       493.6
Physique    1224.9333 1278.9333 652.7333       539.6
HistGeo      568.7333  652.7333 812.1833       580.9
Litterature  493.6000  539.6000 580.9000       622.6
```

Hide

```
# Troisieme étape : Décomposition de Sigma
out = eigen(Sigma)
out
```

```
eigen() decomposition
$values
[1] 3188.18931  631.99720  131.17903   49.28446

$vectors
           [,1]       [,2]       [,3]        [,4]
[1,] -0.5989783  0.4352828  0.1148977  0.66223293
[2,] -0.6128774  0.2910069 -0.1245442 -0.72400618
[3,] -0.3931985 -0.6491107 -0.6259301  0.17961548
[4,] -0.3331684 -0.5518144  0.7612482 -0.07071685
```

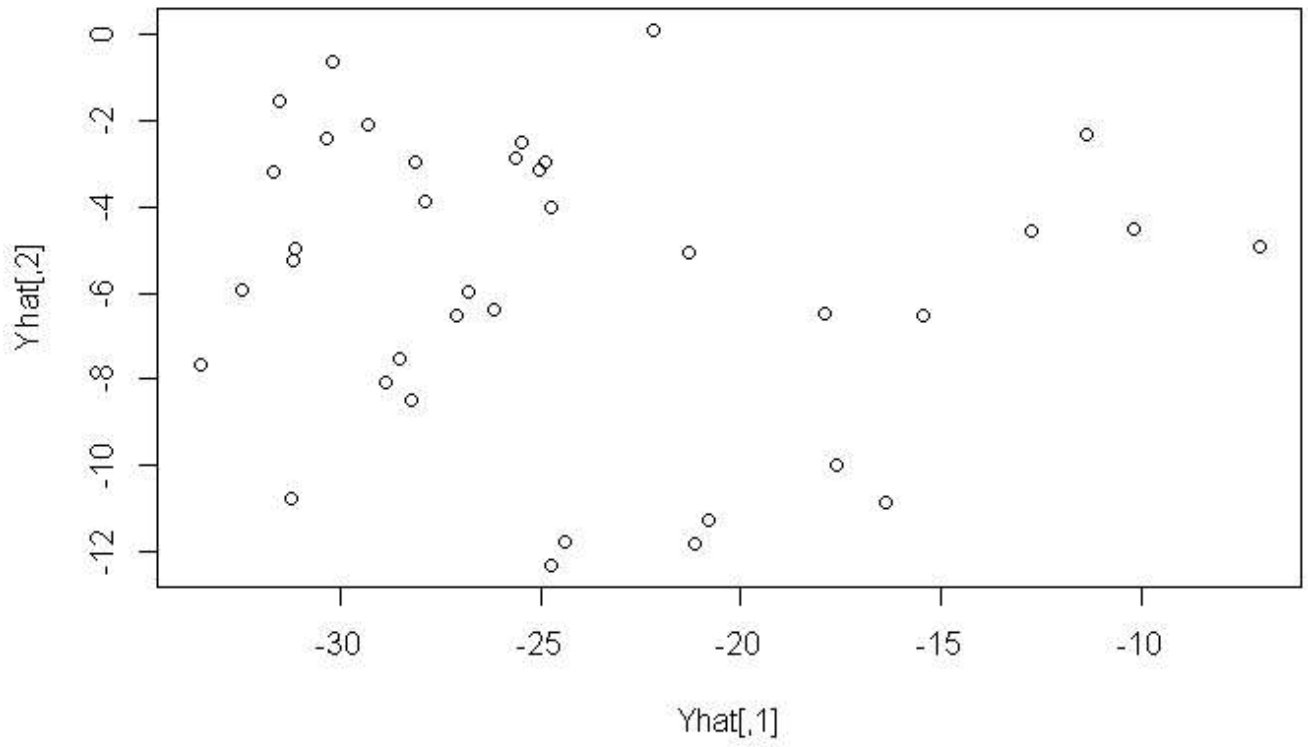
Hide

```
d = 3 #on choisi arbitrairement de garder les deux premiers eigenvector. On étudiera plus tar
d comment choisir le meilleur d
Ustar = out$vect[,1:d]
Ustar
```

```
           [,1]       [,2]       [,3]
[1,] -0.5989783  0.4352828  0.1148977
[2,] -0.6128774  0.2910069 -0.1245442
[3,] -0.3931985 -0.6491107 -0.6259301
[4,] -0.3331684 -0.5518144  0.7612482
```

Hide

```
# Dernière étape : Calcul de la matrice Yhat
Yhat = as.matrix(X) %*% Ustar
plot(Yhat)
```



En R, la fonction princomp permet de calculer le PCA automatiquement :

Hide

```
Data=read_csv("Notes.csv")
```

```
-- Column specification --------------------------------------------------------
-------------------------------------------------
cols(
  Eleve = col_character(),
  Math = col_double(),
  Physique = col_double(),
  HistGeo = col_double(),
  Litterature = col_double(),
  EPS = col_double(),
  Genre = col_character()
)
```

Hide

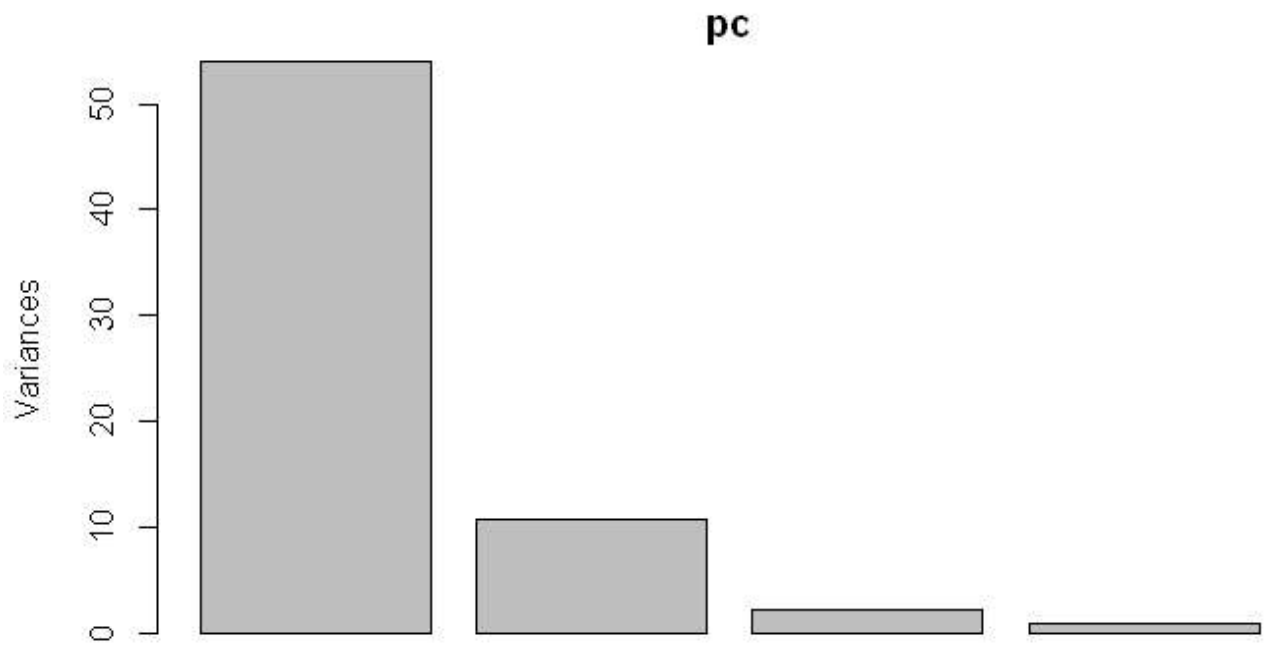
```
X=Data[,2:5]

# Compute PCA
pc = prcomp(X)
# Project the data
Yhat = predict(pc)
Yhat
```

```
             PC1        PC2         PC3         PC4
 [1,]  -9.5129229  5.2956763  2.00663570  1.14281362
 [2,] -11.8092457  2.2068890  1.96692239  0.71302584
 [3,] -10.7497104  0.4541495  1.07035613  0.75120762
 [4,]  -3.0531173  6.8363988 -0.11397877 -0.04043615
 [5,]   0.5501896  6.3416460  1.78310443  0.37486379
 [6,]  -3.0531173  6.8363988 -0.11397877 -0.04043615
 [7,]  -2.7199489  6.2845844 -0.87522695 -0.11115300
 [8,]   0.8833580  5.7898316  1.02185626  0.30414694
 [9,]   0.8833580  5.7898316  1.02185626  0.30414694
[10,]  -7.1602741  2.5860390 -0.04844083 -1.07068360
[11,]  -6.5612958  3.0213218 -0.16333855 -0.40845067
[12,]  -9.4565969 -0.5027483 -0.08815414 -1.50047138
[13,]  -9.4843952 -0.2141966 -0.56703794  1.27200684
[14,]  -5.4278313  1.0270101  0.56671535 -1.50617567
[15,]  -6.8271057  2.0342246 -0.80968901 -1.14140045
[16,]  -9.9559849 -2.2856873 -0.47256060  0.05643962
[17,]  -5.0946628  0.4751957 -0.19453282 -1.57689252
[18,]  -4.4956845  0.9104785 -0.30943054 -0.91465960
[19,]  -7.6245637 -3.4094334  0.02769788  0.28318047
[20,]  -6.4266072 -2.5388679 -0.20209756  1.60764633
[21,]  -8.6306397 -3.0513296 -0.72277640  0.82757118
[22,]  -6.2069282 -1.5987502 -0.70348346  0.70402466
[23,]  -9.8285962 -3.9218952 -0.49298097 -0.49689468
[24,]  -3.8015778 -2.9640686  1.42339518 -0.20514126
[25,]  -3.2025996 -2.5287858  1.30849746  0.45709167
[26,]  -3.2025996 -2.5287858  1.30849746  0.45709167
[27,]   4.1007950  4.5025742 -0.50176753 -0.51416758
[28,]   5.2987515  5.3731398 -0.73156296  0.81029828
[29,]   5.2987515  5.3731398 -0.73156296  0.81029828
[30,]  -8.4893519 -4.8318134 -0.50375487 -1.11200223
[31,]  -3.0429507 -1.4913718 -0.58006671 -0.69686233
[32,]  -3.0429507 -1.4913718 -0.58006671 -0.69686233
[33,]  -3.0429507 -1.4913718 -0.58006671 -0.69686233
[34,]   0.3868367 -0.3968962  0.18970050 -1.36479955
[35,]   0.3868367 -0.3968962  0.18970050 -1.36479955
[36,]   0.3868367 -0.3968962  0.18970050 -1.36479955
[37,]   3.7888258  0.9861310  0.48058391  0.73974146
[38,]   3.7888258  0.9861310  0.48058391  0.73974146
[39,]   3.7888258  0.9861310  0.48058391  0.73974146
[40,]  -3.9355371 -2.6252001 -1.59040325  0.68043318
[41,]  -3.3226597 -2.3341932 -1.46585907 -0.04357300
[42,]  -3.9355371 -2.6252001 -1.59040325  0.68043318
[43,]  10.3180702 -3.1592587  5.11005547 -0.12784334
[44,]  10.3180702 -3.1592587  5.11005547 -0.12784334
[45,]  10.3180702 -3.1592587  5.11005547 -0.12784334
[46,]   6.2659966  1.0440747 -1.14716368  1.19876743
[47,]   6.2659966  1.0440747 -1.14716368  1.19876743
[48,]   6.2659966  1.0440747 -1.14716368  1.19876743
[49,]  -0.4845506 -5.5989821 -0.59953278  1.19568816
[50,]  -0.4845506 -5.5989821 -0.59953278  1.19568816
[51,]  -0.4845506 -5.5989821 -0.59953278  1.19568816
[52,]   8.9583845 -0.9200376 -0.92926959 -1.41768679
[53,]   8.9583845 -0.9200376 -0.92926959 -1.41768679
[54,]   8.9583845 -0.9200376 -0.92926959 -1.41768679
[55,]  11.4955854 -0.9593903 -1.16983892 -0.70832849
[56,]  11.4955854 -0.9593903 -1.16983892 -0.70832849
```

```
[57,]  11.4955854 -0.9593903 -1.16983892 -0.70832849
[58,]  14.6317647 -0.5634602 -1.52530597  0.66326274
[59,]  14.6317647 -0.5634602 -1.52530597  0.66326274
[60,]  14.6317647 -0.5634602 -1.52530597  0.66326274
```

Hide

```
# Select d
screeplot(pc)
```



Hide

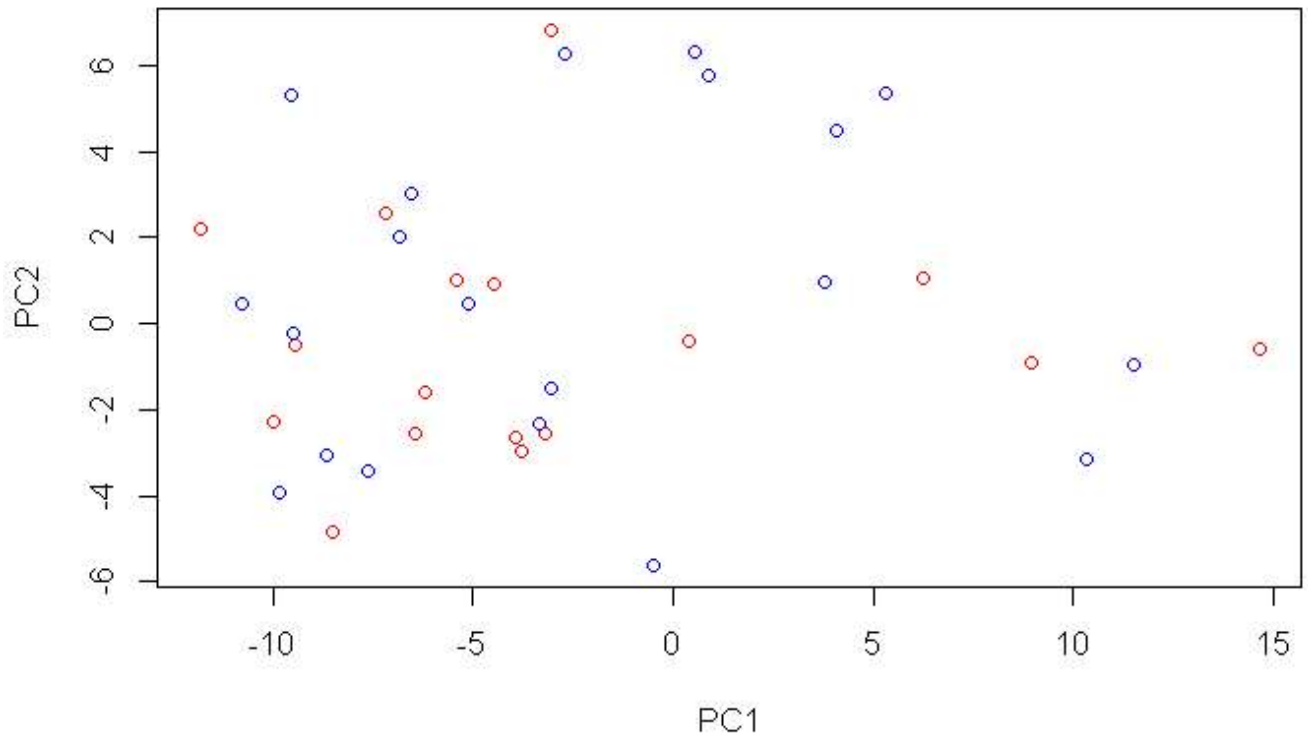
```
100 * pc$sdev^2 / sum(pc$sdev^2)
```

```
[1] 79.691783 15.797363  3.278943  1.231911
```

Ainsi on a 4 composants. Pour réduire les dimensions on ne gardera que les composants les plus pertinants, c'est à dire ayant les variances les plus élevées. En calculant la proportion de variance de chaque composant, on a donc 79% pour le 1, 15% pour le 2, 3% pour le 3 et 1% pour le 4. Il semble donc évident que le 1 et le 2 sont les composant principaux car ils contiennent plus de 95% de l'information, et que 3 et 4 peuvent donc être ignorés. On retiendra donc d=2.

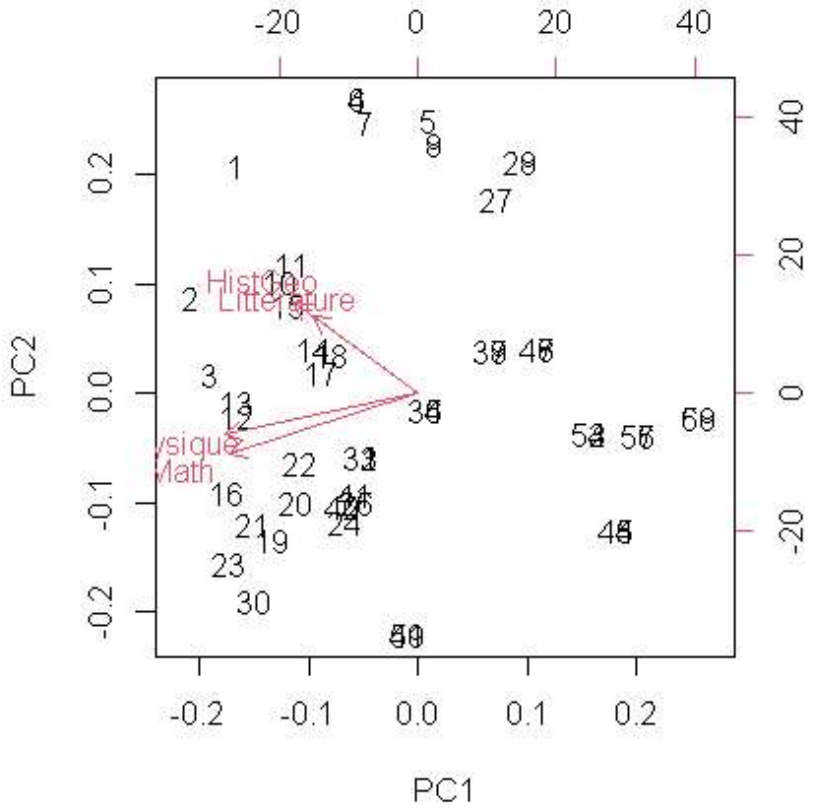
Hide

```
d = 1

# Plot
plot(pc$x[,1:2], col=c('blue','red'))
```



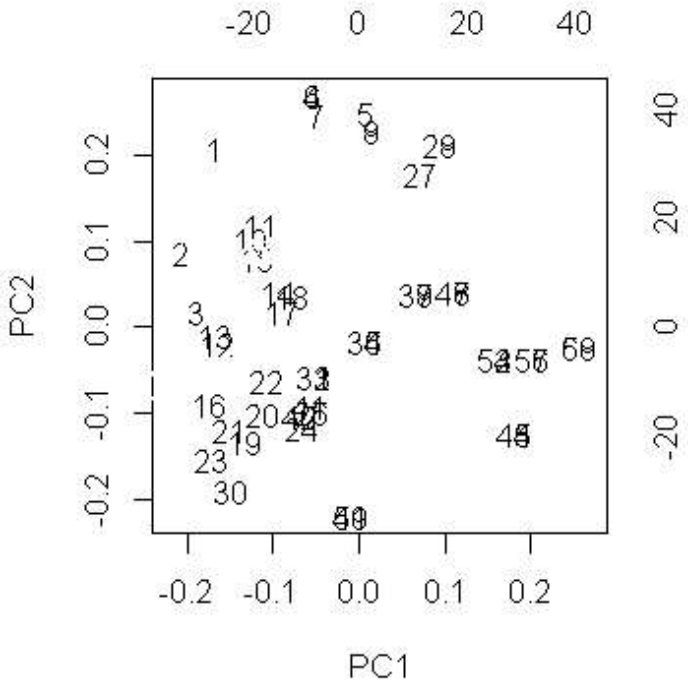
Hide

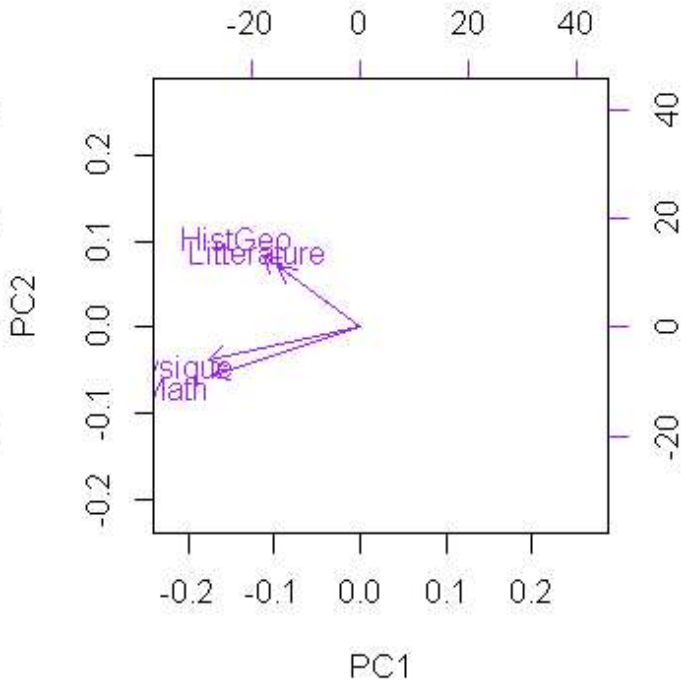
```
biplot(pc)
par(mfrow = c(1,2))
```



Hide

```
biplot(pc,col = c("black",0))
biplot(pc,col = c(0,"purple")); box()
```


-20
0
20
40
0.2
0.1
0.0
-0.1
-0.2
PC2
-0.2
-0.1
0.0
0.1
0.2
PC1


-20
0
20
40
0.2
0.1
0.0
-0.1
-0.2
PC2
HistGeo
Litterature
Math
-0.2
-0.1
0.0
0.1
0.2
PC1