Attention de changer le data.frame en matrice !

`cov` utilise l'estimateur non biaisé en divisant par $n-1$ où n est la taille du jeu de données

```
X<-read.table(text="
math scie fran lati d-m
jean 6.0 6.0 5.0 5.5 8.0
aline 8.0 8.0 8.0 8.0 9.0
annie 6.0 7.0 11.0 9.5 11.0
monique 14.5 14.5 15.5 15.0 8.0
didier 14.0 14.0 12.0 12.5 10.0
andr´e 11.0 10.0 5.5 7.0 13.0
pierre 5.5 7.0 14.0 11.5 10.0
brigitte 13.0 12.5 8.5 9.5 12.0
evelyne 9.0 9.5 12.5 12.0 18.0
")
X<-as.matrix(X)
X<-scale(X, center=TRUE,scale=FALSE)
n<-nrow(X)
print(cov(X))
```

```
          math      scie      fran      lati    d.m
math 12.812500 11.156250  2.989583  5.427083 0.1250
scie 11.156250 10.062500  4.635417  6.166667 0.0625
fran  2.989583  4.635417 13.569444 10.454861 0.4375
lati  5.427083  6.166667 10.454861  8.902778 0.7500
d.m   0.125000  0.062500  0.437500  0.750000 9.7500
```

```
print(S<-1/n*t(X)%*%X)
```

```
           math       scie       fran      lati        d.m
math 11.3888889 9.91666667  2.6574074 4.8240741 0.11111111
scie  9.9166667 8.94444444  4.1203704 5.4814815 0.05555556
fran  2.6574074 4.12037037 12.0617284 9.2932099 0.38888889
lati  4.8240741 5.48148148  9.2932099 7.9135802 0.66666667
d.m   0.1111111 0.05555556  0.3888889 0.6666667 8.66666667
```

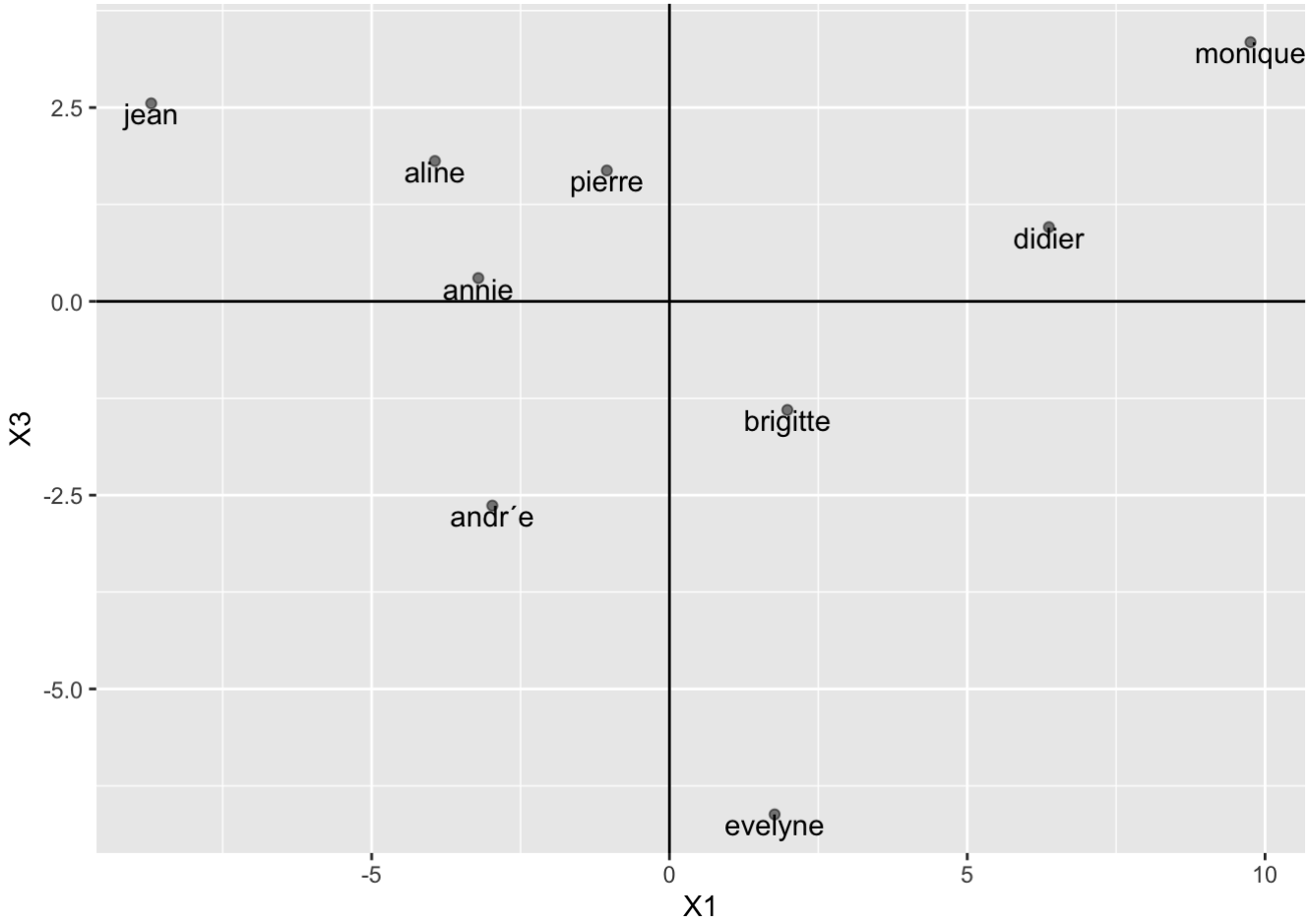
```
my.PCA<-function(X,scale=TRUE,nb.comp=2,plot=TRUE){

  X<-scale(as.matrix(X),center=TRUE,scale=scale)
  S<-1/n*t(X)%*%X
  eigen(S)->res.PCA
  Lambdas<-res.PCA$values[1:nb.comp]
  U<-res.PCA$vectors[,1:nb.comp]
  C=X%*%U
  return(list(C=C,U=U,Lambdas=Lambdas,variance.tot=sum(X^2)/n))
}

my.PCA(X,scale=FALSE,nb.comp=5)->resultat
print(resultat)
```

```
$C
              [,1]       [,2]       [,3]        [,4]        [,5]
jean     -8.700907  1.7027046  2.5539182 -0.14945398 -0.11731596
aline    -3.938596  0.7085441  1.8104644 -0.09068389  0.04349922
annie    -3.209392 -3.4590552  0.3006617  0.17254286  0.01928215
monique   9.755741 -0.2157421  3.3436726 -0.17347137  0.10041455
didier    6.371422  2.1733326  0.9570588  0.07066256 -0.18799232
andr´e   -2.974017  4.6509322 -2.6349457 -0.02321315  0.14809545
pierre   -1.050967 -6.2271742  1.6880636  0.11529582  0.04281219
brigitte  1.980533  4.0685562 -1.4007122  0.24321198  0.01039742
evelyne   1.766183 -3.4020982 -6.6181814 -0.16489082 -0.05919270

$U
           [,1]       [,2]        [,3]        [,4]        [,5]
[1,] 0.51453535  0.5669492 -0.05132308 -0.28874852  0.57254891
[2,] 0.50698853  0.3719958 -0.01445296  0.55305647 -0.54635285
[3,] 0.49235486 -0.6503536  0.10806565  0.39373536  0.40978192
[4,] 0.48462835 -0.3232385  0.02254331 -0.67419539 -0.45343643
[5,] 0.03062778 -0.1128933 -0.99245689  0.03443659  0.01266839

$Lambdas
[1] 28.253249801 12.074723274  8.615733579  0.021732182  0.009869805

$variance.tot
[1] 48.97531
```

```r
print(Q.Eq<-cumsum(resultat$Lambdas)/resultat$variance.tot)
```

```
[1] 0.5768876 0.8234348 0.9993547 0.9997985 1.0000000
```

```r
library(ggplot2)
my.PCA(X,scale=FALSE,nb.comp=5)->student.PCA

C<-data.frame(student.PCA$C)
ggplot(data = C, aes(x=C[,1],y=C[,3],label=rownames(C))) +
  # Add the points
  geom_point(alpha=0.5) +
  geom_text(vjust=1)+
  xlab(names(C)[1])+ylab(names(C)[3])+
  geom_vline(xintercept=0)+geom_hline(yintercept=0)
```



Cercle des corrélations

```
circleFun <- function(center = c(0,0),diameter = 1, npoints = 100){
    r = diameter / 2
    theta<- seq(0,2*pi,length.out = npoints)
    xx <- center[1] + r * cos(theta)
    yy <- center[2] + r * sin(theta)
    return(data.frame(x = xx, y = yy))
}


cor.circle <- circleFun(c(0,0),2,npoints = 100)
colnames(cor.circle)<-c("PC1","PC3")

D<-cor(as.matrix(X),as.matrix(C)) # Coordonnées des anciennes variables par rapport au
D<-data.frame(D[,c(1,3)])
D$label<-rownames(D)
names(D)<-c("PC1","PC3","label")
# Cercle des correlations
plot.correlation.circle<-function(D){
  ggplot() +
  # Draw the circle
  geom_path(data = cor.circle, aes(PC1, PC3)) +
  # Add the arrows
  geom_segment(data = D, aes(x = 0, y = 0, xend = PC1, yend = PC3),
               arrow = arrow(length = unit(0.2, "cm")), lineend = "round") +
```

```
  # Add the labels
  geom_text(data = D, aes(x = PC1, y = PC3, label = label), vjust = -0.5, hjust = 0.5,
  # Set the aspect ratio
  coord_equal() +
  # Set the plot limits
  xlim(-1, 1) +
  ylim(-1, 1)
}
plot.correlation.circle(D)+geom_vline(xintercept=0)+geom_hline(yintercept=0)
```