Working Notes: Data Analysis with R, Python and Rcpp (version 0.999)

P.BADY

Date: 2024-02-26

Contents

- **Contributor:** P.BADY
- **Date:** '2024-02-26'
- **Revision:** Mon Feb 26 14:56:28 2024
- **Abstract:**

The document investigates the evolution/progression of the tumor size (volume) before the the first TMZ treatment in Low grade Glioma (codeleted and non-codeleted IDH mutant patients) from Montpellier cohort. Several methods were used to estimate the estimation of the growth rate such as linear mixed model, non-linear models with bayesian estimation.

- **Keywords:** python, C/C++, pandas, numpy, R, dudi, multifactorial analyses

---

---

1 Motivations

The objective of this document is to propose an implementation of data analysis methods based on duality diagram in python and to develop and expand my skill in python. The document contains information related to the use of Rcpp (R and C++).

2 first test PCA based on R package ade4

2.1 Example from the R package ade4

```
library(ade4)
data(deug)
pca1 <- dudi.pca(deug$tab, center = deug$cent, scale = FALSE, scan = FALSE)
pca2 <- dudi.pca(deug$tab, center = TRUE, scale = TRUE, scan = FALSE)
```

The script can be directly usd in Rstudio and R via the R package reticulate (https://rstudio.github.io/reticulate/).

```
write.csv(deug$tab,file="data/deugtab.csv")
```

2.2 Data importation

see for example: https://www.kaggle.com/code/arnopub/pandas-pr-sentation-des-dataframe

```
import numpy as np
import pandas as pd
deugtab = pd.read_csv('data/deugtab.csv')
deugtab
```

```
##      Unnamed: 0  Algebra  Analysis  Proba  ...  Option1  Option2  English  Sport
## 0             1       40      26.0     26  ...       17     24.0     19.0   11.5
## 1             2       37      34.5     37  ...       24     22.0     26.0   11.5
## 2             3       37      41.0     29  ...       24     27.0     19.6   11.5
## 3             4       63      37.5     57  ...       23     23.0     21.0   14.0
## 4             5       55      31.5     34  ...       19     24.0     24.0   11.5
## ..          ...      ...       ...    ...  ...      ...      ...      ...    ...
## 99          100       60      41.0     18  ...       20     24.0     17.2    0.0
## 100         101       48      44.0     22  ...       22     28.0     19.6    0.0
## 101         102       44      45.0     42  ...       27     22.0     18.4   15.0
## 102         103       47      32.0     26  ...       23     28.0     19.0   11.5
## 103         104       44      32.0     42  ...       28     27.5     23.0   11.5
##
## [104 rows x 10 columns]
```

```
del deugtab['Unnamed: 0']
deugtab
```

```
##      Algebra  Analysis  Proba  Informatic  ...  Option1  Option2  English  Sport
## 0         40      26.0     26        26.0  ...       17     24.0     19.0   11.5
## 1         37      34.5     37        32.0  ...       24     22.0     26.0   11.5
## 2         37      41.0     29        34.5  ...       24     27.0     19.6   11.5
## 3         63      37.5     57        35.5  ...       23     23.0     21.0   14.0
## 4         55      31.5     34        36.0  ...       19     24.0     24.0   11.5
## ..       ...       ...    ...         ...  ...      ...      ...      ...    ...
## 99        60      41.0     18        30.0  ...       20     24.0     17.2    0.0
## 100       48      44.0     22        30.0  ...       22     28.0     19.6    0.0
```

```
## 101       44      45.0     42        35.0  ...       27     22.0     18.4   15.0
## 102       47      32.0     26        21.0  ...       23     28.0     19.0   11.5
## 103       44      32.0     42        26.5  ...       28     27.5     23.0   11.5
##
## [104 rows x 9 columns]
```

3 PCA from scratch in python

PCA from scratch (https://towardsdatascience.com/principal-component-analysis-from-scratch-in-numpy-61843da1f967)

```
# centering = TRUE
X= deugtab - deugtab.mean()
# Normalize
Z = X / X.std(ddof=0)
print('MEAN:')
```

```
## MEAN:
```

```
print(Z.mean())
```

```
## Algebra      -1.024821e-16
## Analysis      3.928481e-16
## Proba        -1.216975e-16
## Informatic   -1.708035e-16
## Economy       4.782499e-16
## Option1      -1.281027e-17
## Option2       1.814788e-16
## English      -9.137990e-16
## Sport         1.665335e-16
## dtype: float64
```

```
print('---'*15)
```

```
## ---------------------------------------------
```

```
print('STD:')
```

```
## STD:
```

```
print(Z.std(ddof=0))
```

```
## Algebra       1.0
## Analysis      1.0
## Proba         1.0
## Informatic    1.0
## Economy       1.0
## Option1       1.0
## Option2       1.0
## English       1.0
## Sport         1.0
## dtype: float64
```

diagonalisation and eigenvectors

```
import numpy as np
len(Z)
```

```
## 104
```

```
ZZ = np.dot(Z.T, Z)/len(Z)
eigenvalues, eigenvectors = np.linalg.eig(ZZ)
D = np.diag(eigenvalues)
P = eigenvectors
Z_new = np.dot(Z, P)
```

valeur propres non ordonnées !!!!

Calculate the proportion of variance explained by each feature

```
sum_eigenvalues = np.sum(eigenvalues)
sum_eigenvalues
```

```
## 8.999999999999996
```

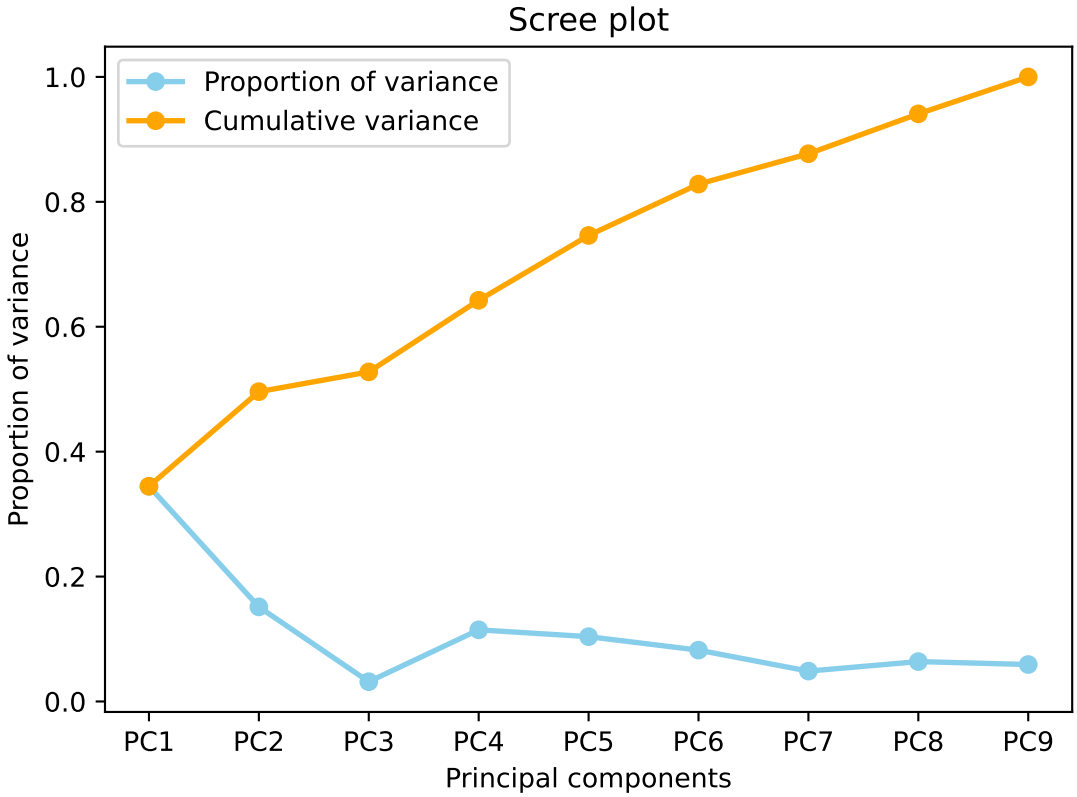
```
prop_var = [i/sum_eigenvalues for i in eigenvalues]
```

Calculate the cumulative variance

```
cum_var = [np.sum(prop_var[:i+1]) for i in range(len(prop_var))]
```

Plot scree plot from PCA

```
import matplotlib.pyplot as plt
x_labels = ['PC{}'.format(i+1) for i in range(len(prop_var))]
plt.plot(x_labels, prop_var, marker='o', markersize=6, color='skyblue',
  linewidth=2, label='Proportion of variance')
plt.plot(x_labels, cum_var, marker='o', color='orange', linewidth=2,
  label="Cumulative variance")
plt.legend()
plt.title('Scree plot')
plt.xlabel('Principal components')
plt.ylabel('Proportion of variance')
plt.show()
```



4 Collaboration R and python

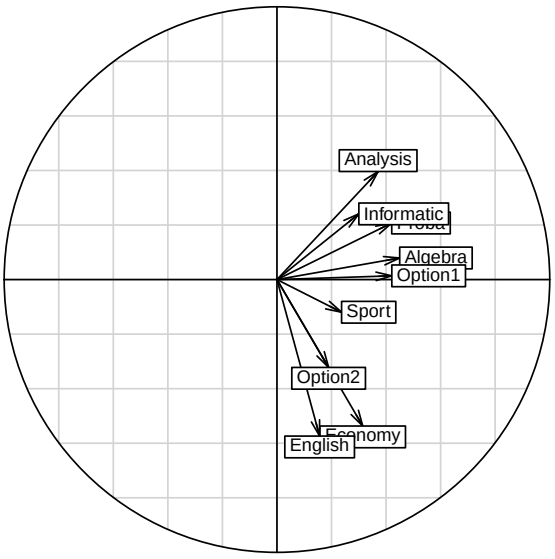
https://rstudio.github.io/reticulate/

```
library(reticulate)
```

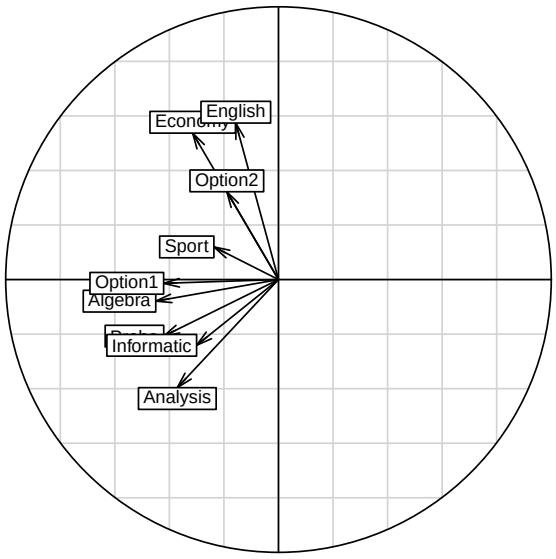
```
##
## Attachement du package : 'reticulate'
```

```
## L'objet suivant est masqué depuis 'package:rtracklayer':
##
##     import
```

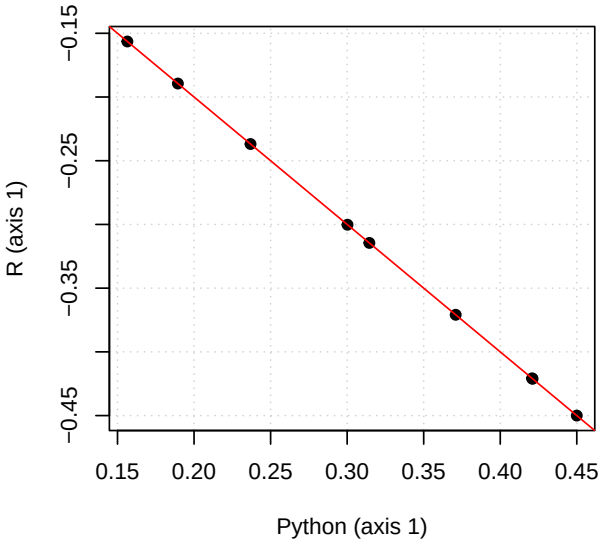
```
library(ade4)
P <- py$P
colnames(P) <- paste("Axis",1:ncol(P),sep="")
rownames(P) <- colnames(py$deugtab)
par(mfrow=c(2,2))
s.corcircle(P,sub="Python version")
s.corcircle(pca2$c1,sub="R version")
plot(P[,1],pca2$c1[,1],panel.first=c(grid()),xlab="Python (axis 1)",
     ylab="R (axis 1)",pch=19);abline(0,-1,col="red")
plot(P[,2],pca2$c1[,2],panel.first=c(grid()),xlab="Python (axis 2)",
     ylab="R (axis 2)",pch=19);abline(0,-1,col="red")
```

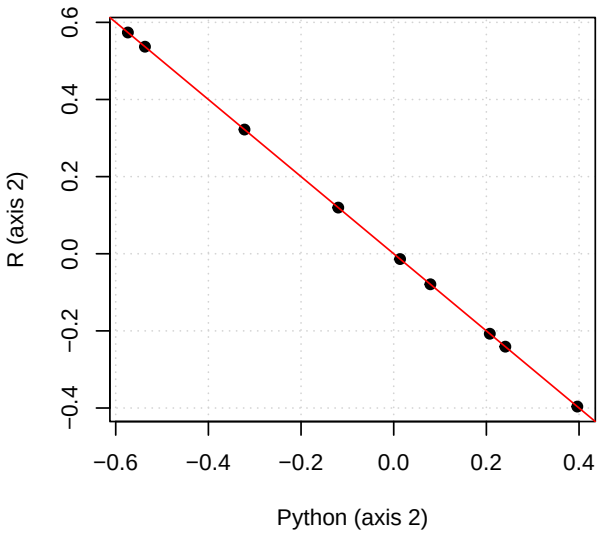


Python version

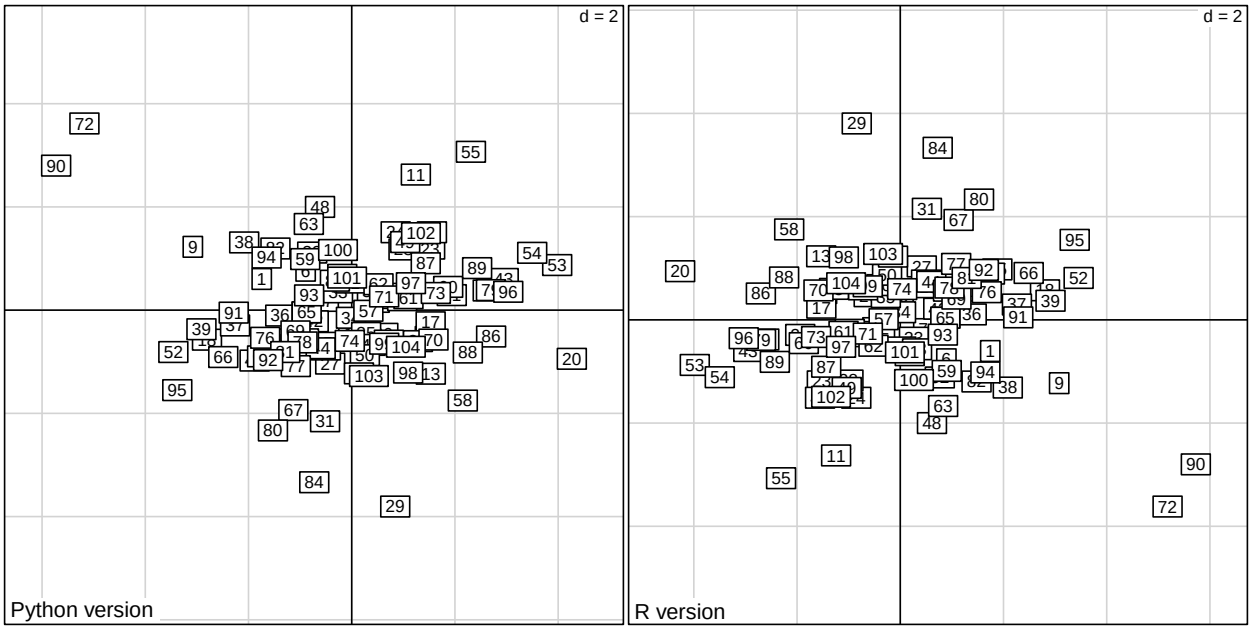


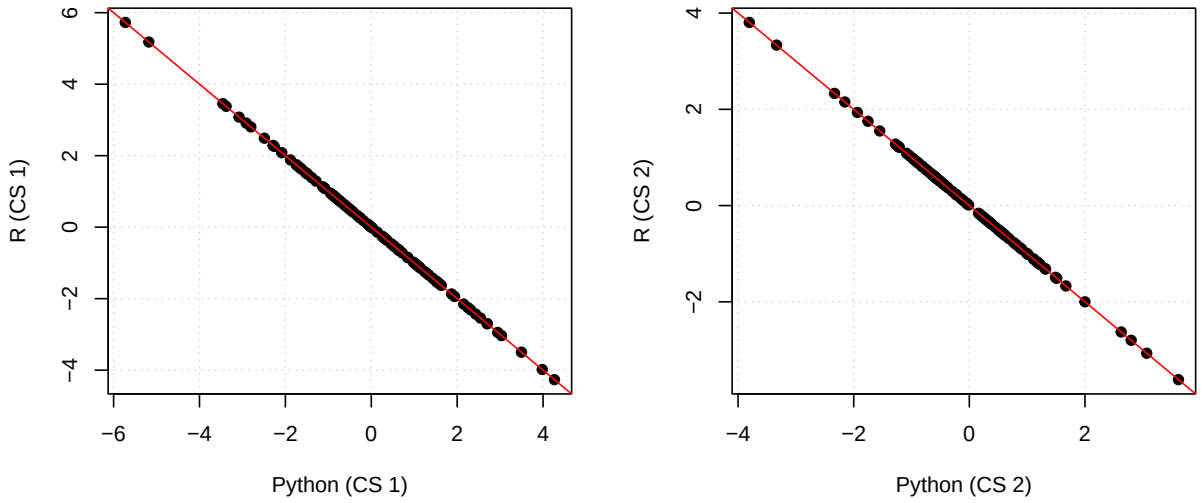
R version



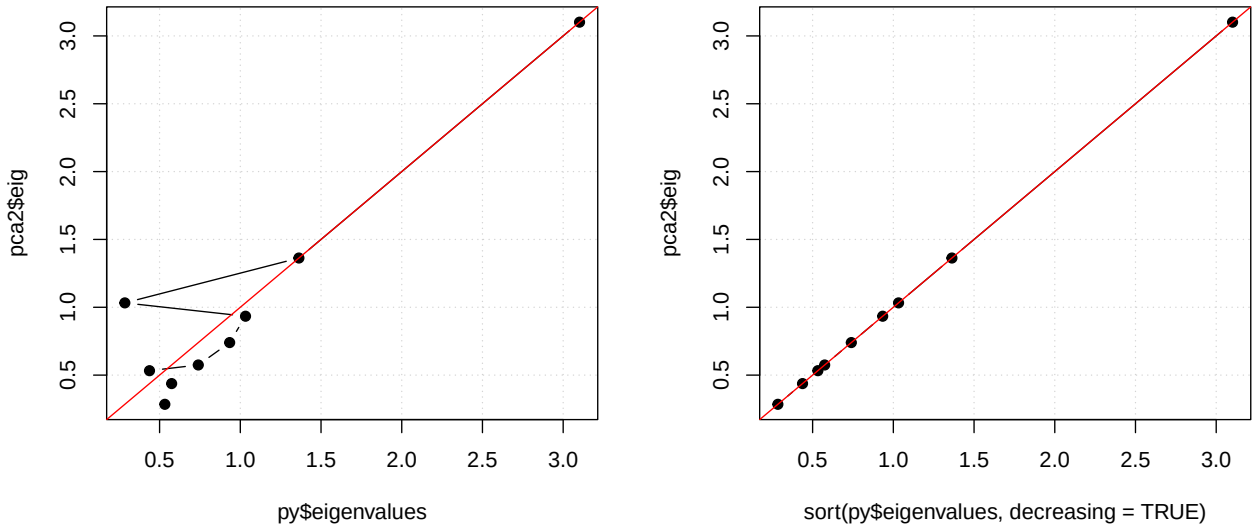



```
coordli <- py$Z_new
colnames(coordli) <- paste("CS",1:ncol(coordli),sep="")
rownames(coordli) <- rownames(py$deugtab)
par(mfrow=c(2,2))
s.label(coordli,sub="Python version")
s.label(pca2$li,sub="R version")
plot(coordli[,1],pca2$li[,1],panel.first=c(grid()),xlab="Python (CS 1)",
     ylab="R (CS 1)",pch=19);abline(0,-1,col="red")
plot(coordli[,2],pca2$li[,2],panel.first=c(grid()),xlab="Python (CS 2)",
     ylab="R (CS 2)",pch=19);abline(0,-1,col="red")
```






```
par(mfrow=c(1,2))
plot(py$eigenvalues,pca2$eig,type="b",panel.first=c(grid()),pch=19)
abline(0,1,col="red")
plot(sort(py$eigenvalues,decreasing = TRUE),pca2$eig,type="b",
     panel.first=c(grid()),pch=19)
abline(0,1,col="red")
```



problem in the order of the eigenvalues !!!

```
pca2$eig
```

```
## [1] 3.1013578 1.3629834 1.0323269 0.9340533 0.7397529 0.5746693 0.5325414 0.4375395
## [9] 0.2847754
```

```
py$D
```

```
##           [,1]     [,2]      [,3]     [,4]      [,5]      [,6]      [,7]      [,8]
##  [1,] 3.101358 0.000000 0.0000000 0.000000 0.0000000 0.0000000 0.0000000 0.0000000
##  [2,] 0.000000 1.362983 0.0000000 0.000000 0.0000000 0.0000000 0.0000000 0.0000000
##  [3,] 0.000000 0.000000 0.2847754 0.000000 0.0000000 0.0000000 0.0000000 0.0000000
##  [4,] 0.000000 0.000000 0.0000000 1.032327 0.0000000 0.0000000 0.0000000 0.0000000
##  [5,] 0.000000 0.000000 0.0000000 0.000000 0.9340533 0.0000000 0.0000000 0.0000000
##  [6,] 0.000000 0.000000 0.0000000 0.000000 0.0000000 0.7397529 0.0000000 0.0000000
##  [7,] 0.000000 0.000000 0.0000000 0.000000 0.0000000 0.0000000 0.4375395 0.0000000
##  [8,] 0.000000 0.000000 0.0000000 0.000000 0.0000000 0.0000000 0.0000000 0.5746693
##  [9,] 0.000000 0.000000 0.0000000 0.000000 0.0000000 0.0000000 0.0000000 0.0000000
##           [,9]
##  [1,] 0.0000000
##  [2,] 0.0000000
##  [3,] 0.0000000
##  [4,] 0.0000000
##  [5,] 0.0000000
##  [6,] 0.0000000
##  [7,] 0.0000000
##  [8,] 0.0000000
##  [9,] 0.5325414
```

```
py$eigenvalues
```

```
## [1] 3.1013578 1.3629834 0.2847754 1.0323269 0.9340533 0.7397529 0.4375395 0.5746693
## [9] 0.5325414
```

problem !!!

test of the pca from scikit-learn

```
import sklearn.decomposition as sd
from sklearn.decomposition import PCA
pca = PCA(n_components=9)
Z2 = Z/np.sqrt(104)
pca.fit(Z2)
```

```
## PCA(n_components=9)
```

```
print(pca.explained_variance_ratio_)
```

```
## [0.34459531 0.1514426  0.11470299 0.1037837  0.08219477 0.06385214
##  0.05917127 0.0486155  0.03164171]
```

```
print(pca.singular_values_)
```

```
## [1.76106724 1.16746882 1.01603489 0.96646433 0.86008891 0.75806943
##  0.72975436 0.66146771 0.5336435 ]
```

```
print(pca.singular_values_*pca.singular_values_)
```

```
## [3.10135782 1.36298344 1.0323269  0.9340533  0.73975293 0.57466926
##  0.53254143 0.43753953 0.28477539]
```

5 Construction of the function "pydudi" (first prototype)

based on the duality diagram (see more details below)

- https://pbil.univ-lyon1.fr/R/pdf/tdr61.pdf
- https://pbil.univ-lyon1.fr/R/pdf/stage3.pdf
- https://pbil.univ-lyon1.fr/R/pdf/bs8.pdf

first test => need to adjust the weighting (test with COA)

```
import os
import string
import re
import pandas as pd
import numpy as np

def pydudi(X,cw,lw,nf):
  dim = X.shape
  n = dim[0]
  p = dim[1]
  nf0 = nf-1
  # n=len(X)
  # p=len(X.columns)
  D = np.diag(np.sqrt(lw))
  Q = np.diag(np.sqrt(cw))

  # XtDXQ => problem  with Q !!!

  XD = np.dot(X.T,D).T
  XD = np.dot(XD,Q)
  XtX = np.dot(XD.T,XD)


  # decomposition
```

```
  eigenvalues, eigenvectors = np.linalg.eig(XtX)
  index = np.argsort(eigenvalues)[::-1]

  # np.nonzero(eigenvalues)[0]
  eigenvalues = eigenvalues.real
  eigenvectors = eigenvectors.real
  eigenvalues=eigenvalues[index]
  eigenvectors=eigenvectors[:,index]
  # results
  rank = len(np.nonzero(eigenvalues)[0])
  C1 = np.dot(np.diag(1/np.sqrt(cw)),eigenvectors[:,0:nf])
  #C1 = eigenvectors[:,0:nf]
  XQ = np.dot(X,np.diag(cw))
  Li = np.dot(XQ, C1)

  # need to adjust the weighting (problem with sqrt)
  L1 = np.dot(Li,np.diag(1/np.sqrt(eigenvalues[0:nf])))
  Co = np.dot(C1,np.diag(np.sqrt(eigenvalues[0:nf])))
  return eigenvalues,rank,Li,L1,Co,C1,nf;

###
```

5.1 Test pydudi with PCA

```
import numpy as np
import pandas as pd
deugtab = pd.read_csv('data/deugtab.csv')
deugtab
```

```
##      Unnamed: 0  Algebra  Analysis  Proba  ...  Option1  Option2  English  Sport
## 0             1       40      26.0     26  ...       17     24.0     19.0   11.5
## 1             2       37      34.5     37  ...       24     22.0     26.0   11.5
## 2             3       37      41.0     29  ...       24     27.0     19.6   11.5
## 3             4       63      37.5     57  ...       23     23.0     21.0   14.0
## 4             5       55      31.5     34  ...       19     24.0     24.0   11.5
## ..          ...      ...       ...    ...  ...      ...      ...      ...    ...
## 99          100       60      41.0     18  ...       20     24.0     17.2    0.0
## 100         101       48      44.0     22  ...       22     28.0     19.6    0.0
## 101         102       44      45.0     42  ...       27     22.0     18.4   15.0
## 102         103       47      32.0     26  ...       23     28.0     19.0   11.5
## 103         104       44      32.0     42  ...       28     27.5     23.0   11.5
##
## [104 rows x 10 columns]
```

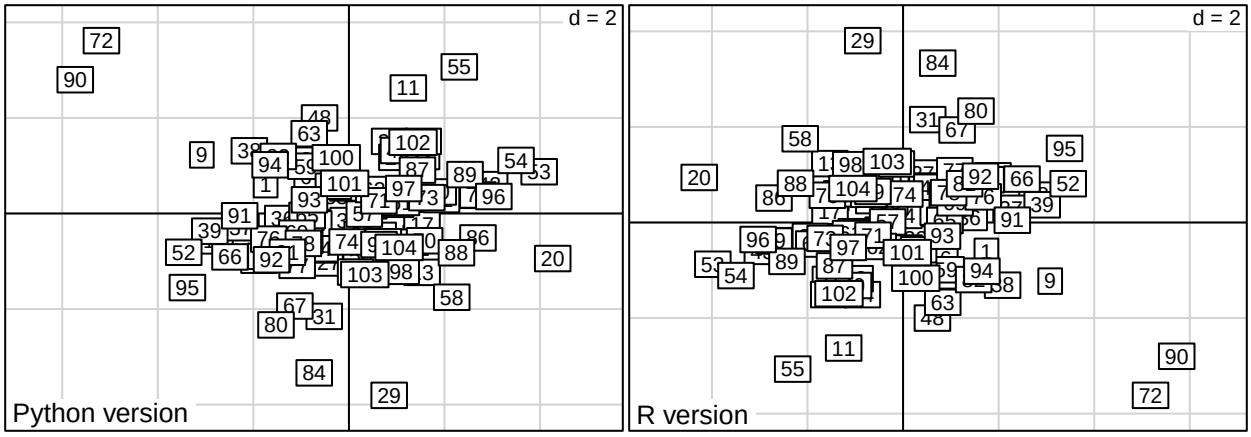
```
del deugtab['Unnamed: 0']
deugtab
```

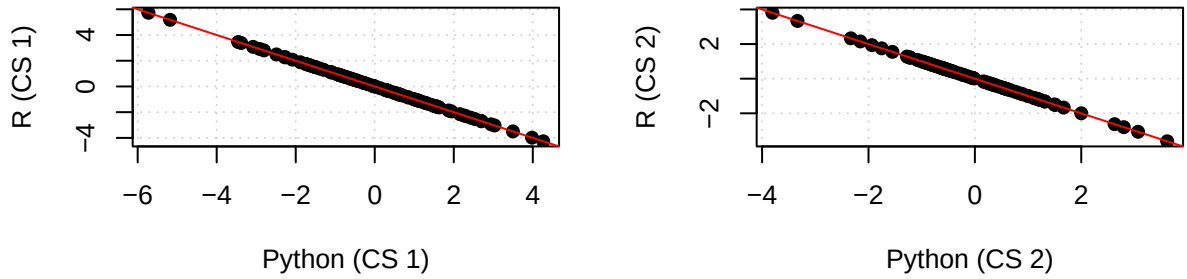
```
##      Algebra  Analysis  Proba  Informatic  ...  Option1  Option2  English  Sport
## 0         40      26.0     26        26.0  ...       17     24.0     19.0   11.5
## 1         37      34.5     37        32.0  ...       24     22.0     26.0   11.5
## 2         37      41.0     29        34.5  ...       24     27.0     19.6   11.5
## 3         63      37.5     57        35.5  ...       23     23.0     21.0   14.0
## 4         55      31.5     34        36.0  ...       19     24.0     24.0   11.5
## ..       ...       ...    ...         ...  ...      ...      ...      ...    ...
```

```
## 99        60      41.0     18        30.0  ...       20     24.0     17.2    0.0
## 100       48      44.0     22        30.0  ...       22     28.0     19.6    0.0
## 101       44      45.0     42        35.0  ...       27     22.0     18.4   15.0
## 102       47      32.0     26        21.0  ...       23     28.0     19.0   11.5
## 103       44      32.0     42        26.5  ...       28     27.5     23.0   11.5
##
## [104 rows x 9 columns]
```

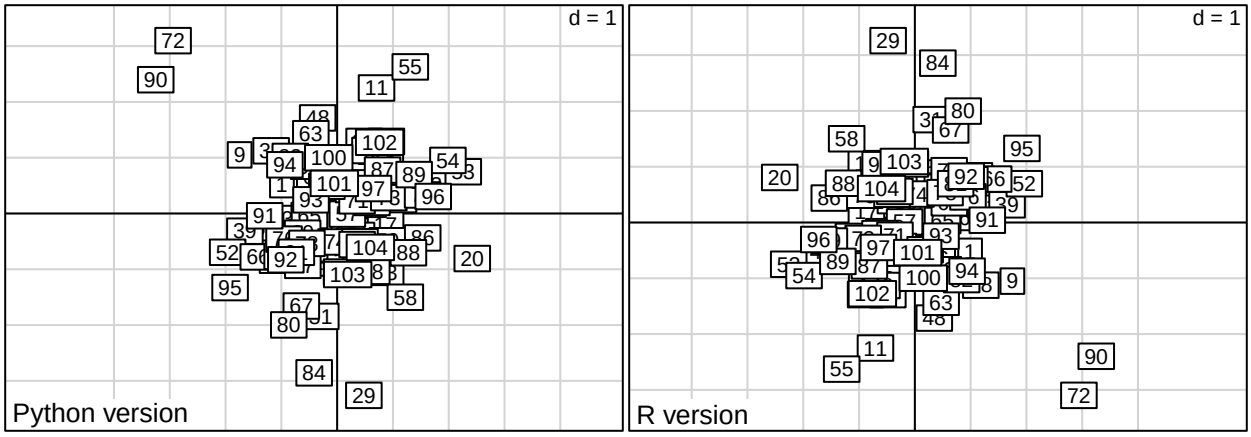
```
X= deugtab - deugtab.mean()
# Normalize
X = X / X.std(ddof=0)
dim = X.shape
n = dim[0]
p = dim[1]
# lw = pd.DataFrame(np.repeat(1/len(X),len(X)))[0]
# cw = pd.DataFrame(np.repeat(1,len(X.columns)))[0]
lw = pd.DataFrame(np.repeat(1/n,n))[0]
cw = pd.DataFrame(np.repeat(1,p))[0]
ted = pydudi(X,cw,lw,2)
```

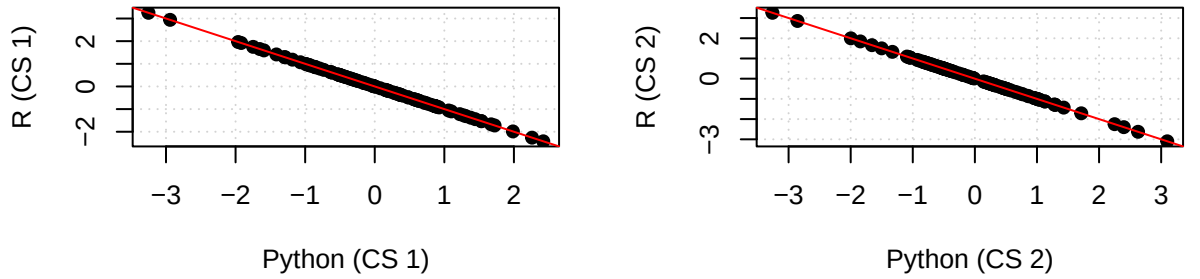
```
library(reticulate)
names(py$ted) <- c("eig","rank","li","l1","co","c1","nf")
coordli <- py$ted$li
par(mfrow=c(2,2))
s.label(coordli,sub="Python version")
s.label(pca2$li,sub="R version")
plot(coordli[,1],pca2$li[,1],panel.first=c(grid()),xlab="Python (CS 1)",
     ylab="R (CS 1)",pch=19);abline(0,-1,col="red")
plot(coordli[,2],pca2$li[,2],panel.first=c(grid()),xlab="Python (CS 2)",
     ylab="R (CS 2)",pch=19);abline(0,-1,col="red")
```



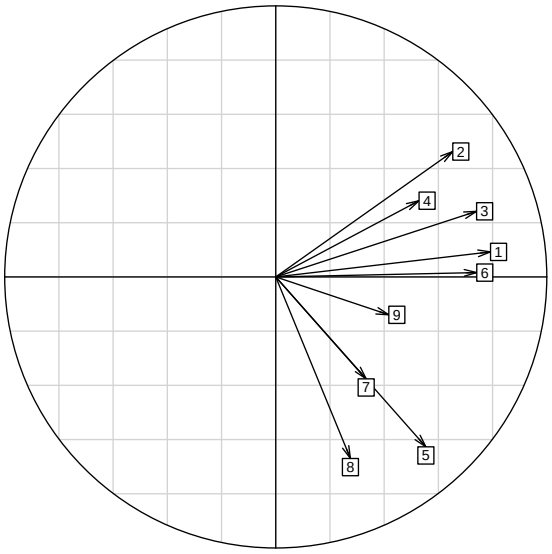



```
coordl1 <- py$ted$l1
par(mfrow=c(2,2))
s.label(coordl1,sub="Python version")
s.label(pca2$l1,sub="R version")
plot(coordl1[,1],pca2$l1[,1],panel.first=c(grid()),xlab="Python (CS 1)",
     ylab="R (CS 1)",pch=19);abline(0,-1,col="red")
plot(coordl1[,2],pca2$l1[,2],panel.first=c(grid()),xlab="Python (CS 2)",
     ylab="R (CS 2)",pch=19);abline(0,-1,col="red")
```

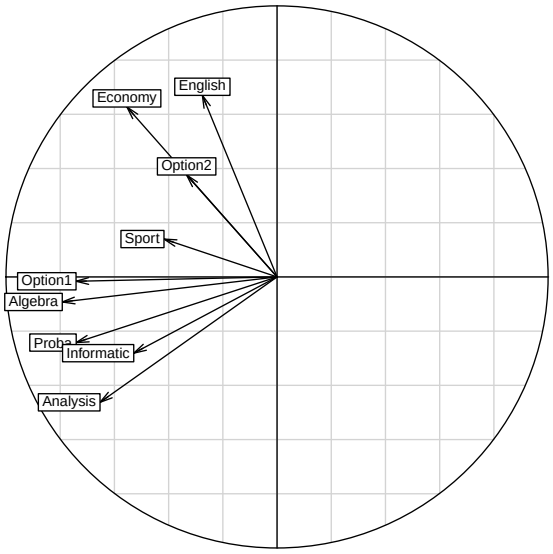




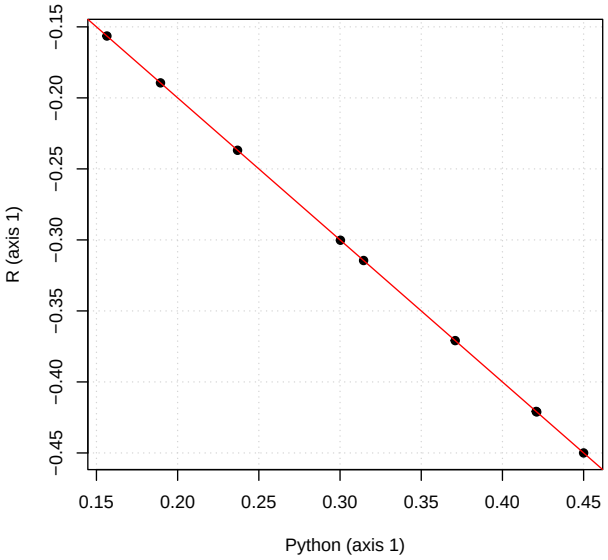

```
par(mfrow=c(2,2))
s.corcircle(py$ted$co,sub="Python version")
s.corcircle(pca2$co,sub="R version")
plot(py$ted$c1[,1],pca2$c1[,1],panel.first=c(grid()),xlab="Python (axis 1)",
     ylab="R (axis 1)",pch=19);abline(0,-1,col="red")
plot(py$ted$c1[,2],pca2$c1[,2],panel.first=c(grid()),xlab="Python (axis 2)",
     ylab="R (axis 2)",pch=19);abline(0,-1,col="red")
```

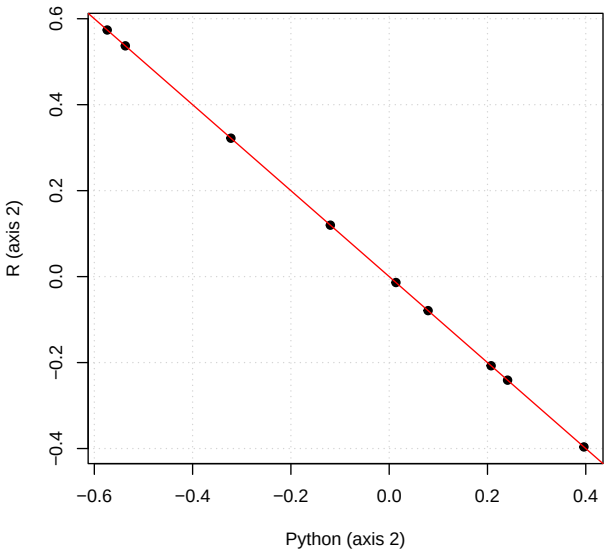


Python version



R version





5.2 Test pydudi with COA

COA: correspondence analysis.

```
 Benzécri, J.P. and Coll. (1973) _L'analyse des données. II
 L'analyse des correspondances_, Bordas, Paris. 1-620.

 Greenacre, M. J. (1984) _Theory and applications of correspondence
 analysis_, Academic Press, London.
```

Si R help from ade4 (dudi.coa) and https://pbil.univ-lyon1.fr/R/pdf/stage4.pdf

```
data(rpjdl)
chisq.test(rpjdl$fau)$statistic
```

```
## Warning in chisq.test(rpjdl$fau): L'approximation du Chi-2 est peut-être incorrecte
```

```
## X-squared
##  7323.597
```

```
rpjdl.coa <- coa1 <- dudi.coa(rpjdl$fau, scannf = FALSE, nf = 4)
sum(rpjdl.coa$eig)*rpjdl.coa$N # the same
```

```
## [1] 7323.597
```

```
write.csv(rpjdl$fau,file="data/fau.csv")
```

```
# import numpy as np
import pandas as pd
fau = pd.read_csv('data/fau.csv')
fau
```

```
##      Unnamed: 0  AR  CP  ST  CC  UE  PV  JT  ...  CN  SS  FC  MC  EC  EH  El  PD
## 0             1   0   0   0   0   0   0   0  ...   1   0   0   0   0   0   0   1
## 1             2   0   0   0   0   0   0   0  ...   0   1   0   0   0   0   0   0
## 2             3   0   0   0   0   1   0   0  ...   1   1   0   0   0   0   1   0
## 3             4   0   0   0   0   0   0   0  ...   1   0   0   0   0   0   0   0
## 4             5   1   0   0   0   0   0   0  ...   0   0   0   0   0   0   0   0
## ..          ...  ..  ..  ..  ..  ..  ..  ..  ...  ..  ..  ..  ..  ..  ..  ..  ..
## 177         178   0   0   0   0   0   0   0  ...   0   0   0   0   0   0   0   0
## 178         179   0   0   0   0   0   0   0  ...   0   0   1   0   0   0   0   0
## 179         180   0   0   0   0   0   1   0  ...   0   1   1   0   0   0   0   0
## 180         181   0   0   1   0   0   0   1  ...   0   1   0   0   0   0   0   0
## 181         182   0   0   0   0   0   0   0  ...   0   1   1   0   1   0   0   0
##
## [182 rows x 52 columns]
```

```
del fau['Unnamed: 0']
fau
```

```
##      AR  CP  ST  CC  UE  PV  JT  GT  LA  ...  CA  CN  SS  FC  MC  EC  EH  El  PD
## 0     0   0   0   0   0   0   0   0   0  ...   1   1   0   0   0   0   0   0   1
## 1     0   0   0   0   0   0   0   1   0  ...   1   0   1   0   0   0   0   0   0
## 2     0   0   0   0   1   0   0   0   0  ...   0   1   1   0   0   0   0   1   0
## 3     0   0   0   0   0   0   0   0   0  ...   0   1   0   0   0   0   0   0   0
## 4     1   0   0   0   0   0   0   1   0  ...   1   0   0   0   0   0   0   0   0
## ..   ..  ..  ..  ..  ..  ..  ..  ..  ..  ...  ..  ..  ..  ..  ..  ..  ..  ..  ..
## 177   0   0   0   0   0   0   0   0   0  ...   0   0   0   0   0   0   0   0   0
## 178   0   0   0   0   0   0   0   0   0  ...   0   0   0   1   0   0   0   0   0
## 179   0   0   0   0   0   1   0   0   0  ...   0   0   1   1   0   0   0   0   0
## 180   0   0   1   0   0   0   1   0   0  ...   0   0   1   0   0   0   0   0   0
## 181   0   0   0   0   0   0   0   0   0  ...   0   0   1   1   0   1   0   0   0
##
## [182 rows x 51 columns]
```

```
import numpy as np

X=fau
sumX = fau.sum().sum()
sumCol =  fau.sum(axis=0)
sumRow = fau.sum(axis=1)

pij = X/sumX
pi = sumRow/sumX
```

```
pj = sumCol/sumX

Dj = np.diag(1/pj)
Di = np.diag(1/pi)

Z = np.dot(Di,pij)
Z= np.dot(Z,Dj)
Z.shape
```

```
## (182, 51)
```

```
Z =Z - 1
Z = np.nan_to_num(Z)

# Normalize
lw = pi
cw = pj
D= np.diag(np.sqrt(pi))
Q= np.diag(np.sqrt(pj))
X =Z
ted = pydudi(Z,cw,lw,2)
```

```
require(reticulate)
names(py$ted) <- c("eig","rank","li","l1","co","c1","nf")
```

```
head(t(t(py$X)%*%py$D))[,1]
```

```
## [1] -0.06986433 -0.06986433 -0.06535210 -0.04940154  0.85002979  0.76588343
```

```
head(py$X*diag(py$D))[,1]
```

```
## [1] -0.06986433 -0.06986433 -0.06535210 -0.04940154  0.85002979  0.76588343
```

```
XD <- t(t(py$X)%*%py$D)
head(XD%*%py$Q)[,1]
```

```
## [1] -0.007717578 -0.007717578 -0.007219133 -0.005457152  0.093898719  0.084603474
```

```
head(sweep(XD,2,diag(py$Q),"*"))[,1]
```

```
## [1] -0.007717578 -0.007717578 -0.007219133 -0.005457152  0.093898719  0.084603474
```

```
coa1$eig
```

```
##  [1] 0.753246079 0.292905714 0.229339077 0.204667043 0.157288711 0.151440858
##  [7] 0.150767524 0.139271459 0.128099632 0.121613888 0.117678929 0.114349277
## [13] 0.111133629 0.108686867 0.104567293 0.098786861 0.093491391 0.089476206
## [19] 0.083038229 0.078627728 0.071855317 0.066171912 0.064569801 0.063747501
## [25] 0.061830129 0.056205831 0.054891709 0.051264371 0.051057282 0.048112526
## [31] 0.047741238 0.045225569 0.042174679 0.041081123 0.039945517 0.037205530
## [37] 0.034420610 0.031713688 0.029256753 0.027162167 0.026386209 0.022092347
## [43] 0.021089820 0.020517411 0.016786030 0.016066325 0.015476722 0.014463573
## [49] 0.012991222 0.008353461
```

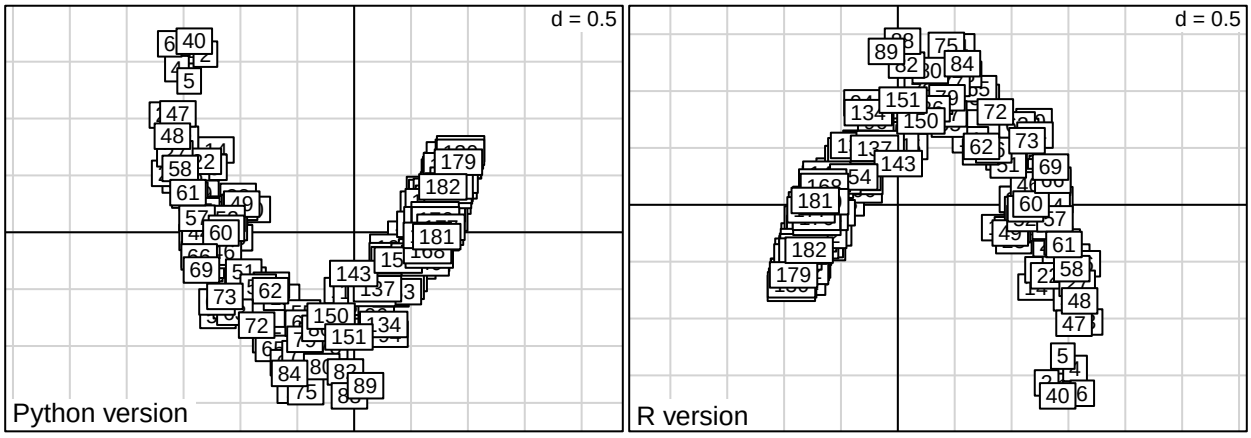
```
py$ted$eig
```

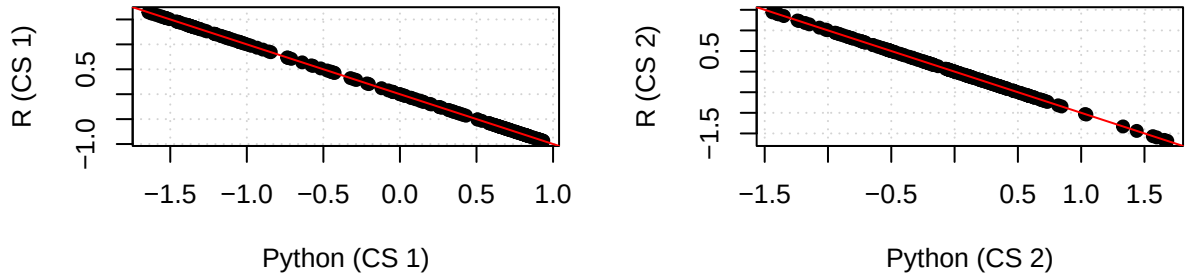
```
##  [1]  7.532461e-01  2.929057e-01  2.293391e-01  2.046670e-01  1.572887e-01
##  [6]  1.514409e-01  1.507675e-01  1.392715e-01  1.280996e-01  1.216139e-01
## [11]  1.176789e-01  1.143493e-01  1.111336e-01  1.086869e-01  1.045673e-01
```

```
## [16]  9.878686e-02  9.349139e-02  8.947621e-02  8.303823e-02  7.862773e-02
## [21]  7.185532e-02  6.617191e-02  6.456980e-02  6.374750e-02  6.183013e-02
## [26]  5.620583e-02  5.489171e-02  5.126437e-02  5.105728e-02  4.811253e-02
## [31]  4.774124e-02  4.522557e-02  4.217468e-02  4.108112e-02  3.994552e-02
## [36]  3.720553e-02  3.442061e-02  3.171369e-02  2.925675e-02  2.716217e-02
## [41]  2.638621e-02  2.209235e-02  2.108982e-02  2.051741e-02  1.678603e-02
## [46]  1.606633e-02  1.547672e-02  1.446357e-02  1.299122e-02  8.353461e-03
## [51] -1.171126e-16
```

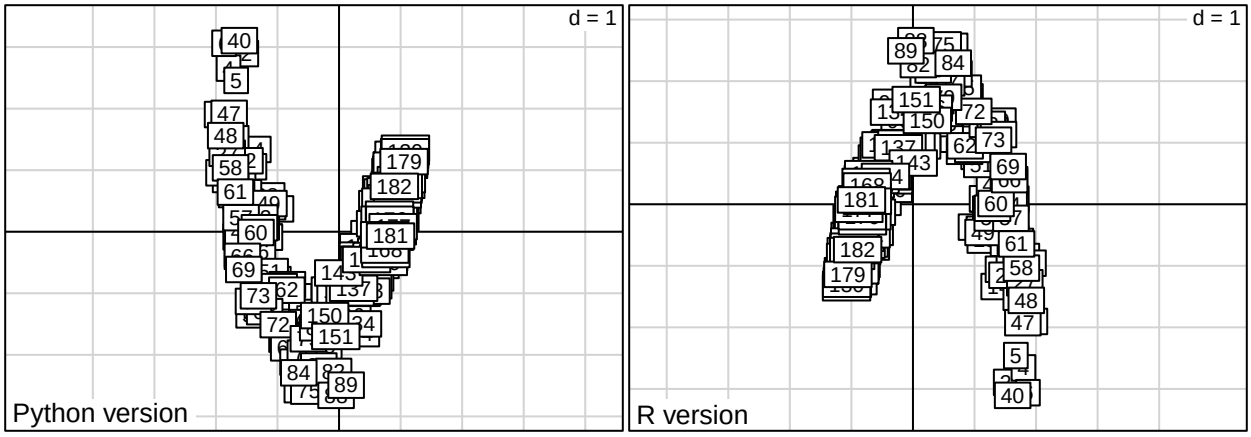
ok for the eigenvalues, probleme dans le calcul des coordonnées ???

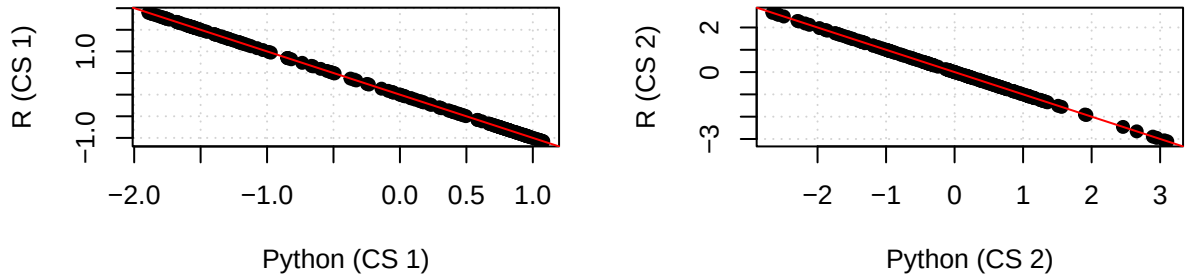
```
coordli <- py$ted$li
par(mfrow=c(2,2))
s.label(coordli,sub="Python version")
s.label(coa1$li,sub="R version")
plot(coordli[,1],coa1$li[,1],panel.first=c(grid()),xlab="Python (CS 1)",
     ylab="R (CS 1)",pch=19);abline(0,-1,col="red")
plot(coordli[,2],coa1$li[,2],panel.first=c(grid()),xlab="Python (CS 2)",
     ylab="R (CS 2)",pch=19);abline(0,-1,col="red")
```



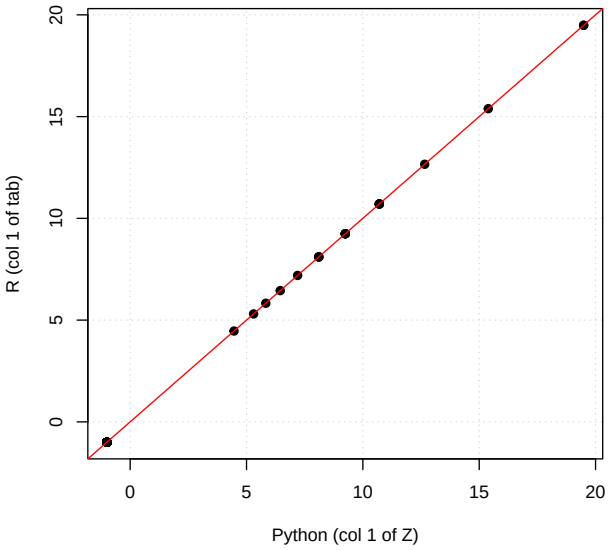


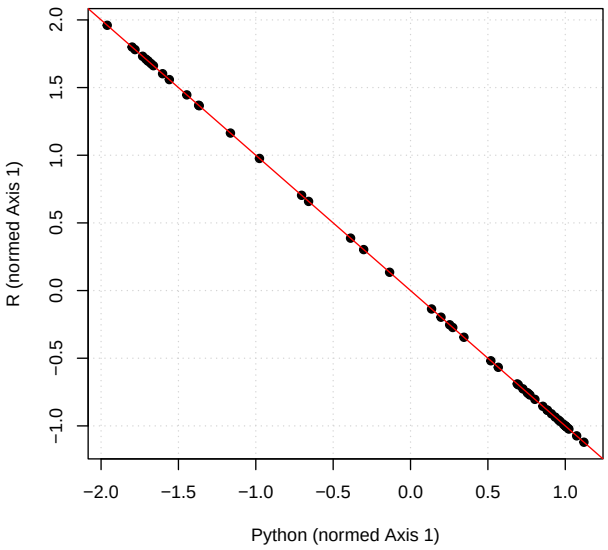

```
coordl1 <- py$ted$l1
par(mfrow=c(2,2))
s.label(coordl1,sub="Python version")
s.label(coa1$l1,sub="R version")
plot(coordl1[,1],coa1$l1[,1],panel.first=c(grid()),xlab="Python (CS 1)",
     ylab="R (CS 1)",pch=19);abline(0,-1,col="red")
plot(coordl1[,2],coa1$l1[,2],panel.first=c(grid()),xlab="Python (CS 2)",
     ylab="R (CS 2)",pch=19);abline(0,-1,col="red")
```

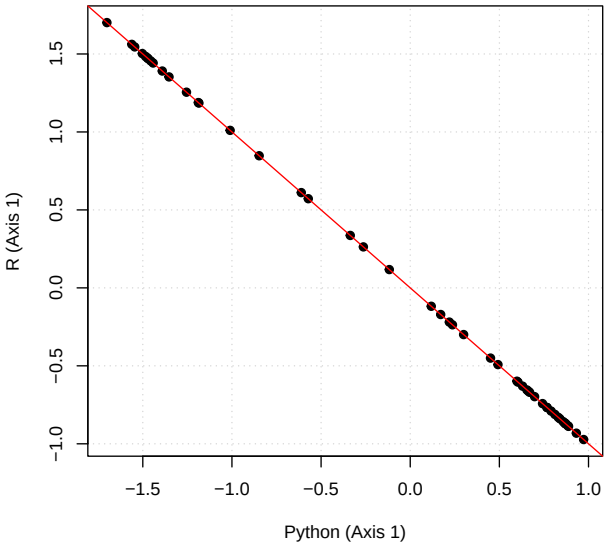





```
par(mfrow=c(2,2))
plot(py$Z[,1],coa1$tab[,1],panel.first=c(grid()),xlab="Python (col 1 of Z)",
     ylab="R (col 1 of tab)",pch=19);abline(0,1,col="red")
plot(py$ted$c1[,1],coa1$c1[,1],panel.first=c(grid()),xlab="Python (normed Axis 1)",
     ylab="R (normed Axis 1)",pch=19);abline(0,-1,col="red")
plot(py$ted$co[,1],coa1$co[,1],panel.first=c(grid()),xlab="Python (Axis 1)",
     ylab="R (Axis 1)",pch=19);abline(0,-1,col="red")
```







=> need to check the weight !!! => Ok !!

5.3 Test pydudi with MCA

MCA: Multiple Correspondence Analysis

```
Tenenhaus, M. & Young, F.W. (1985) An analysis and synthesis of multiple correspondence analysis, optima
Lebart, L., A. Morineau, and M. Piron. 1995. Statistique exploratoire multidimensionnelle. Dunod, Paris
```

Si R help from ade4 (dudi.acm) and https://pbil.univ-lyon1.fr/R/pdf/stage4.pdf. the description of the methods is given below:

- https://pbil.univ-lyon1.fr/R/pdf/tdr521.pdf
- https://pbil.univ-lyon1.fr/R/pdf/tdr52.pdf

duality diagram

```
data(ours)
summary(ours)
```

```
##  altit  deniv  cloiso domain boise  hetra  favor  inexp  citat  depart
##  1: 8   1:13   1:12   1: 9   1:10   1:19   1:15   1:20   1:22   AHP:5
##  2:17   2:14   2: 4   2:13   2:15   2: 5   2:12   2:10   2: 7   AM :4
##  3:13   3:11   3:22   3:16   3:13   3:14   3:11   3: 8   3: 4   D  :5
##                                                      4: 5   HP :8
##                                                             HS :4
##                                                             I  :5
##                                                             S  :7
```

```
acm1 <- dudi.acm(ours, scan = FALSE)
```

```
write.csv(ours,file="data/ours.csv")
```

importation with pandas

```
# import numpy as np
import pandas as pd
ours = pd.read_csv('data/ours.csv')
ours
```

```
##     Unnamed: 0  altit  deniv  cloiso  domain  ...  hetra  favor  inexp  citat  depart
## 0            1      2      3       3       2  ...      3      3      2      1      HS
## 1            2      1      2       1       2  ...      1      2      2      2      HS
## 2            3      3      3       3       2  ...      2      3      3      2      HS
## 3            4      3      3       3       1  ...      3      3      2      3      HS
## 4            5      3      3       1       2  ...      3      2      3      1       S
## 5            6      3      3       3       1  ...      3      3      3      3       S
## 6            7      2      2       3       2  ...      1      2      3      1       S
## 7            8      1      1       2       2  ...      1      3      2      2       S
## 8            9      2      3       1       2  ...      2      3      3      4       S
## 9           10      2      2       3       1  ...      3      2      3      1       S
## 10          11      1      1       1       1  ...      1      2      2      1       S
## 11          12      2      2       3       1  ...      3      3      2      3       I
## 12          13      2      3       3       1  ...      3      3      2      3       I
## 13          14      1      3       2       2  ...      1      1      3      2       I
## 14          15      2      2       1       3  ...      2      2      2      1       I
## 15          16      3      3       3       3  ...      3      3      3      4       I
## 16          17      3      1       3       3  ...      3      3      1      4       D
## 17          18      3      2       3       3  ...      3      2      2      4       D
## 18          19      2      1       1       3  ...      3      3      1      4       D
## 19          20      2      1       1       2  ...      2      1      1      2       D
## 20          21      2      1       1       2  ...      2      1      1      1       D
## 21          22      1      1       1       2  ...      1      1      1      2      HP
## 22          23      2      2       2       2  ...      1      1      1      2      HP
## 23          24      1      1       3       3  ...      1      1      1      1      HP
## 24          25      2      3       2       3  ...      1      1      1      1      HP
## 25          26      2      2       1       1  ...      1      1      1      1      HP
## 26          27      2      2       3       1  ...      1      1      1      1      HP
## 27          28      3      2       1       3  ...      3      2      1      1      HP
## 28          29      2      1       1       2  ...      1      1      1      1      HP
## 29          30      1      1       3       3  ...      1      2      1      1     AHP
## 30          31      3      1       3       3  ...      3      2      1      1     AHP
## 31          32      3      2       3       3  ...      1      2      1      1     AHP
```

```
## 32          33      3      2       3       3  ...      1      1      1      1     AHP
## 33          34      3      1       3       1  ...      3      1      1      1     AHP
## 34          35      1      2       3       3  ...      1      1      1      1      AM
## 35          36      2      2       3       3  ...      1      1      1      1      AM
## 36          37      3      1       3       3  ...      1      1      1      1      AM
## 37          38      2      3       3       3  ...      1      2      2      1      AM
##
## [38 rows x 11 columns]
```

```
del ours['Unnamed: 0']
ours
```

```
##     altit  deniv  cloiso  domain  boise  hetra  favor  inexp  citat depart
## 0       2      3       3       2      2      3      3      2      1     HS
## 1       1      2       1       2      1      1      2      2      2     HS
## 2       3      3       3       2      2      2      3      3      2     HS
## 3       3      3       3       1      3      3      3      2      3     HS
## 4       3      3       1       2      2      3      2      3      1      S
## 5       3      3       3       1      3      3      3      3      3      S
## 6       2      2       3       2      2      1      2      3      1      S
## 7       1      1       2       2      1      1      3      2      2      S
## 8       2      3       1       2      3      2      3      3      4      S
## 9       2      2       3       1      3      3      2      3      1      S
## 10      1      1       1       1      1      1      2      2      1      S
## 11      2      2       3       1      3      3      3      2      3      I
## 12      2      3       3       1      3      3      3      2      3      I
## 13      1      3       2       2      1      1      1      3      2      I
## 14      2      2       1       3      2      2      2      2      1      I
## 15      3      3       3       3      3      3      3      3      4      I
## 16      3      1       3       3      3      3      3      1      4      D
## 17      3      2       3       3      3      3      2      2      4      D
## 18      2      1       1       3      3      3      3      1      4      D
## 19      2      1       1       2      2      2      1      1      2      D
## 20      2      1       1       2      2      2      1      1      1      D
## 21      1      1       1       2      1      1      1      1      2     HP
## 22      2      2       2       2      1      1      1      1      2     HP
## 23      1      1       3       3      1      1      1      1      1     HP
## 24      2      3       2       3      2      1      1      1      1     HP
## 25      2      2       1       1      2      1      1      1      1     HP
## 26      2      2       3       1      1      1      1      1      1     HP
## 27      3      2       1       3      3      3      2      1      1     HP
## 28      2      1       1       2      2      1      1      1      1     HP
## 29      1      1       3       3      1      1      2      1      1    AHP
## 30      3      1       3       3      2      3      2      1      1    AHP
## 31      3      2       3       3      2      1      2      1      1    AHP
## 32      3      2       3       3      2      1      1      1      1    AHP
## 33      3      1       3       1      3      3      1      1      1    AHP
## 34      1      2       3       3      1      1      1      1      1     AM
## 35      2      2       3       3      2      1      1      1      1     AM
## 36      3      1       3       3      3      1      1      1      1     AM
## 37      2      3       3       3      2      1      2      2      1     AM
```

```
def disjonctif(X):
  X_cat = X.astype("category")
  X_dis = pd.get_dummies(X_cat)
```

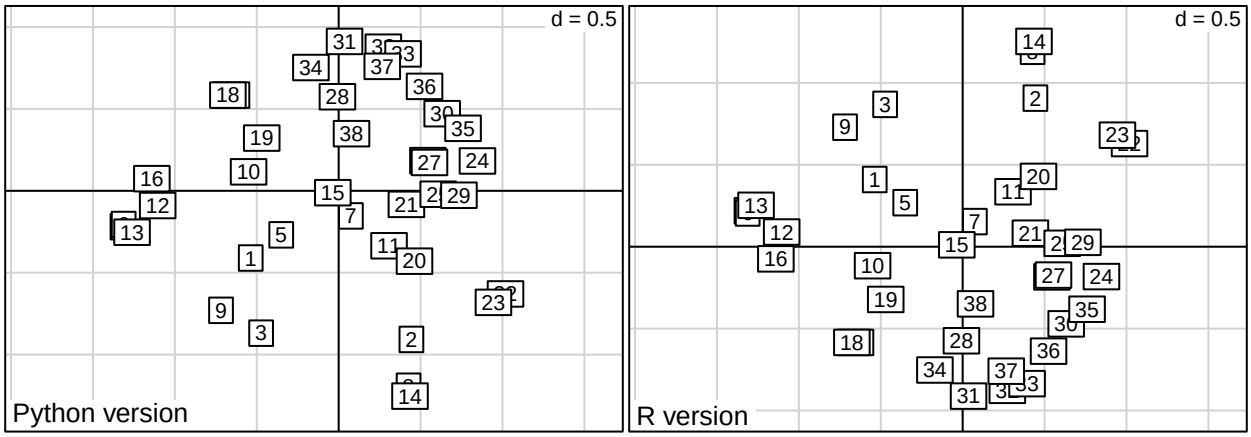
```
  X_dis = X_dis*1
  return X_dis ;
```

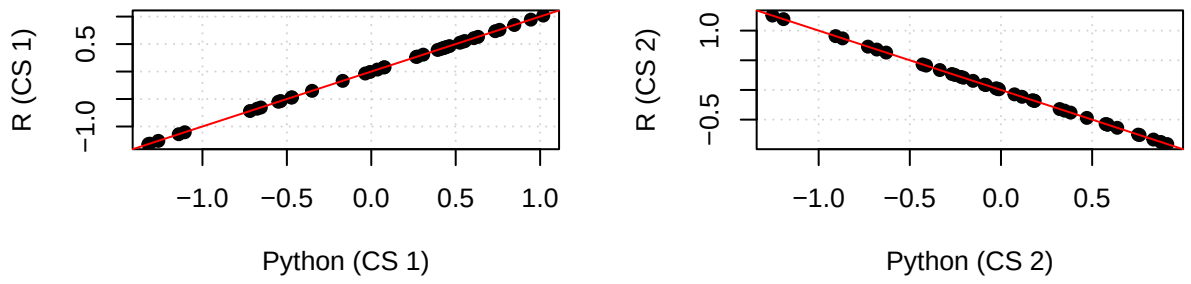
preparation of the triplet

```
Xdis = disjonctif(ours)
m = Xdis.shape[1]
n = Xdis.shape[0]
v = ours.shape[1]
lw = pd.DataFrame(np.repeat(1/n,n))[0]
D = np.diag(lw)
cw = np.dot(np.dot(Xdis.T,D), np.ones(n))
Dm = np.diag(cw)
X = np.dot(Xdis,np.diag(1/cw))-1
cw = cw/v
ted = pydudi(X,cw,lw,2)
```

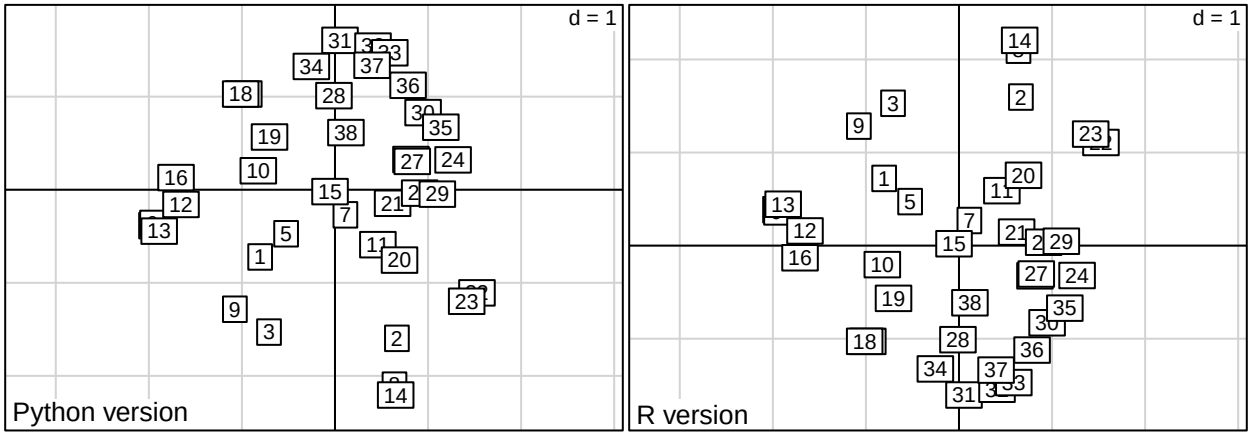
```
require(reticulate)
names(py$ted) <- c("eig","rank","li","l1","co","c1","nf")
```

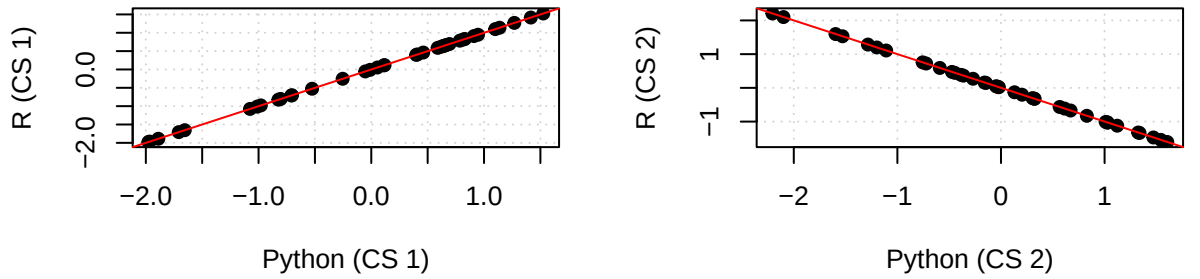
```
coordli <- py$ted$li
par(mfrow=c(2,2))
s.label(coordli,sub="Python version")
s.label(acm1$li,sub="R version")
plot(coordli[,1],acm1$li[,1],panel.first=c(grid()),xlab="Python (CS 1)",
     ylab="R (CS 1)",pch=19);abline(0,1,col="red")
plot(coordli[,2],acm1$li[,2],panel.first=c(grid()),xlab="Python (CS 2)",
     ylab="R (CS 2)",pch=19);abline(0,-1,col="red")
```



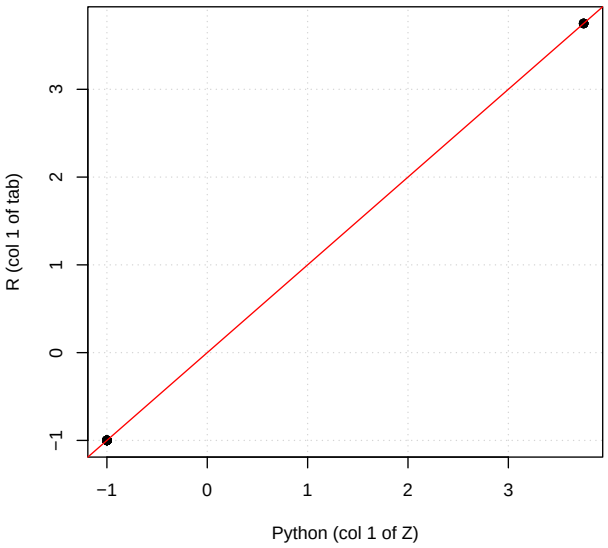


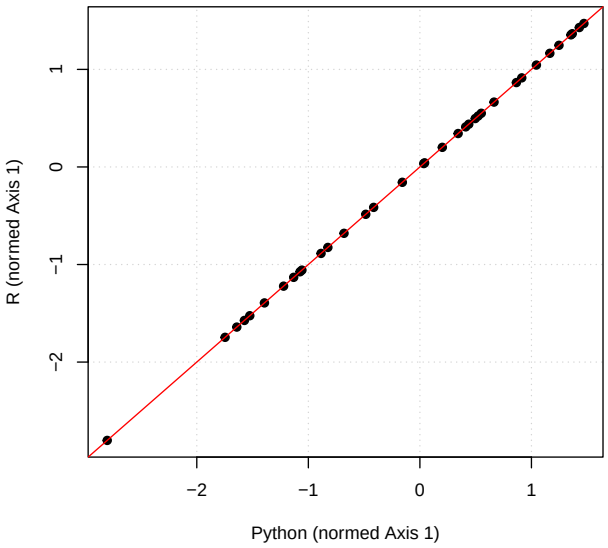
```
coordl1 <- py$ted$l1
par(mfrow=c(2,2))
s.label(coordl1,sub="Python version")
s.label(acm1$l1,sub="R version")
plot(coordl1[,1],acm1$l1[,1],panel.first=c(grid()),xlab="Python (CS 1)",
     ylab="R (CS 1)",pch=19);abline(0,1,col="red")
plot(coordl1[,2],acm1$l1[,2],panel.first=c(grid()),xlab="Python (CS 2)",
     ylab="R (CS 2)",pch=19);abline(0,-1,col="red")
```

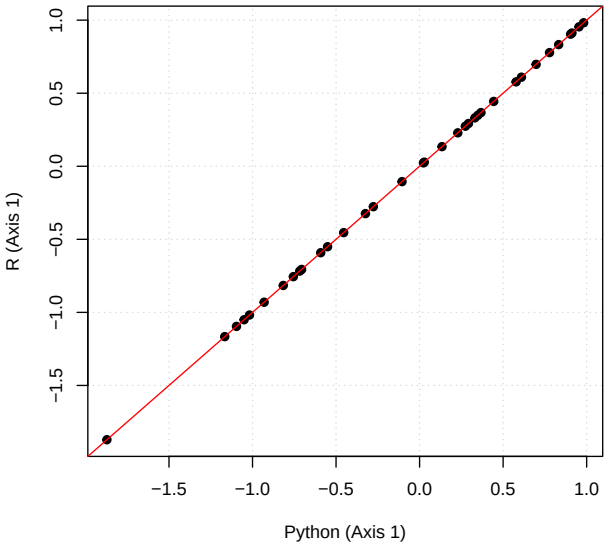





```
par(mfrow=c(2,2))
plot(py$X[,1],acm1$tab[,1],panel.first=c(grid()),xlab="Python (col 1 of Z)",
     ylab="R (col 1 of tab)",pch=19);abline(0,1,col="red")
plot(py$ted$c1[,1],acm1$c1[,1],panel.first=c(grid()),xlab="Python (normed Axis 1)",
     ylab="R (normed Axis 1)",pch=19);abline(0,1,col="red")
plot(py$ted$co[,1],acm1$co[,1],panel.first=c(grid()),xlab="Python (Axis 1)",
     ylab="R (Axis 1)",pch=19);abline(0,1,col="red")
```







6 Class and object dudi

define the structure of the object and description of the elements based on object 'dudi' from R package ade4 (and ADE-4):

- tab
- eig
- rank
- c1
- co
- l1
- li

heritage for the PCA and COA (and other methods):

- tab

- lw
- cw
- dudi
- and specific elements

6.1 Object Dudi

```
class Dudi:
  def __init__(self,tab,cw,lw,eig,rank,nf,c1,co,l1,li):
    self.tab = tab
    self.cw = cw
    self.lw = lw
    self.eig = eig
    self.rank = rank
    self.nf = nf
    self.c1 = c1
    self.co = co
    self.l1 = l1
    self.li = li

def pyDudi(X,cw,lw,nf):
  dim = X.shape
  n = dim[0]
  p = dim[1]
  nf0 = nf-1
  # n=len(X)
  # p=len(X.columns)
  D = np.diag(np.sqrt(lw))
  Q = np.diag(np.sqrt(cw))

  # XtDXQ => problem  with Q !!!
  XD = np.dot(X.T,D).T
  XD = np.dot(XD,Q)
  XtX = np.dot(XD.T,XD)

  # decomposition
  eigenvalues, eigenvectors = np.linalg.eig(XtX)
  index = np.argsort(eigenvalues)[::-1]

  # np.nonzero(eigenvalues)[0]
  eigenvalues = eigenvalues.real
  eigenvectors = eigenvectors.real
  eigenvalues=eigenvalues[index]
  eigenvectors=eigenvectors[:,index]
  # results
  rank = len(np.nonzero(eigenvalues)[0])
  C1 = np.dot(np.diag(1/np.sqrt(cw)),eigenvectors[:,0:nf])
  #C1 = eigenvectors[:,0:nf]
  XQ = np.dot(X,np.diag(cw))
  Li = np.dot(XQ, C1)

  # need to adjust the weighting (problem with sqrt)
  L1 = np.dot(Li,np.diag(1/np.sqrt(eigenvalues[0:nf])))
  Co = np.dot(C1,np.diag(np.sqrt(eigenvalues[0:nf])))
```

```
  return Dudi(X,cw,lw,eigenvalues,rank,nf,C1,Co,L1,Li);
```

6.2 test pour PCA

```
def pyPCA(X,cw=None,lw=None,nf=2,center=True,scale=True):
  dim = X.shape
  n = dim[0]
  p = dim[1]
  if center:
    X = X-X.mean()
  if scale:
    X = X/X.std(ddof=0)
  if lw==None :
    lw = pd.DataFrame(np.repeat(1/n,n))[0]
  if cw==None :
    cw = pd.DataFrame(np.repeat(1,p))[0]
  dudi = pyDudi(X,cw,lw,nf)
  return dudi;
```

```
pca1 = pyPCA(deugtab)
```

6.3 test pour COA

```
def pyCOA(X,nf=2):
  sumX = X.sum().sum()
  sumCol =  X.sum(axis=0)
  sumRow = X.sum(axis=1)
  pij = X/sumX
  pi = sumRow/sumX
  pj = sumCol/sumX
  Dj = np.diag(1/pj)
  Di = np.diag(1/pi)
  Z = np.dot(Di,pij)
  Z = np.dot(Z,Dj)
  Z = Z - 1
  Z = np.nan_to_num(Z)
  dudi = pyDudi(Z,pj,pi,nf)
  return dudi;
```

```
coa1 = pyCOA(fau)
```

6.4 test pour ACM

```
def disjonctif(X):
  X_cat = X.astype("category")
  X_dis = pd.get_dummies(X_cat)
  X_dis = X_dis*1
  return X_dis;

def pyACM(X,lw=None,nf=2):
  Xdis = disjonctif(ours)
  m = Xdis.shape[1]
```

```
  n = Xdis.shape[0]
  v = ours.shape[1]
  if lw==None :
    lw = pd.DataFrame(np.repeat(1/n,n))[0]
  D = np.diag(lw)
  cw = np.dot(np.dot(Xdis.T,D), np.ones(n))
  Dm = np.diag(cw)
  X = np.dot(Xdis,np.diag(1/cw))-1
  cw = cw/v
  dudi = pyDudi(X,cw,lw,nf)
  return dudi;
```

```
mca1 = pyACM(ours)
```

7 Construction of the function dudi in Rcpp

7.1 Test Rcpp: correlation between two variables

```
require(parallel)
require(Rcpp)
```

```
## Le chargement a nécessité le package : Rcpp
```

```
require(RcppArmadillo)
```

```
## Le chargement a nécessité le package : RcppArmadillo
```

```
#Sys.setenv("PKG_CXXFLAGS"="-fopenmp")
#Sys.setenv("PKG_LIBS"="-fopenmp")
sourceCpp(file.path(getwd(),"src","utility.cpp"))
```

correlation coefficient

```
cor(1:10, 2:11)
```

```
## [1] 1
```

```
CORR(1:10, 2:11)
```

```
## [1] 1
```

computation of the area under curve

7.2 Object dudi

dudi obect with Rcpparmadillo

```
sourceCpp(file.path(getwd(),"src","utility.cpp"))
pca2 <- dudi.pca(deug$tab, center = TRUE, scale = TRUE, scan = FALSE)
test <- arc_dudi(as.matrix(pca2$tab),pca2$cw,pca2$lw,2)
test$eig
```

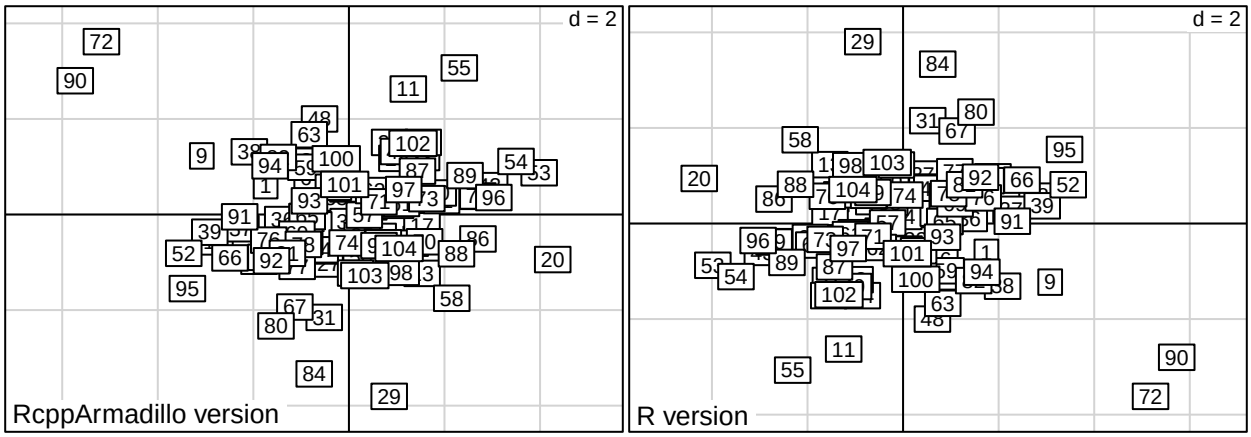
```
##            [,1]
##  [1,] 3.1013578
##  [2,] 1.3629834
##  [3,] 1.0323269
##  [4,] 0.9340533
```

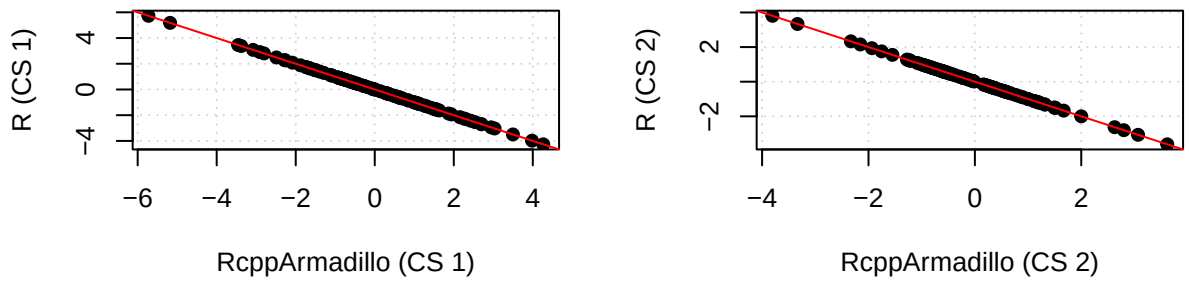
```
##  [5,] 0.7397529
##  [6,] 0.5746693
##  [7,] 0.5325414
##  [8,] 0.4375395
##  [9,] 0.2847754
```

```
pca2$eig
```

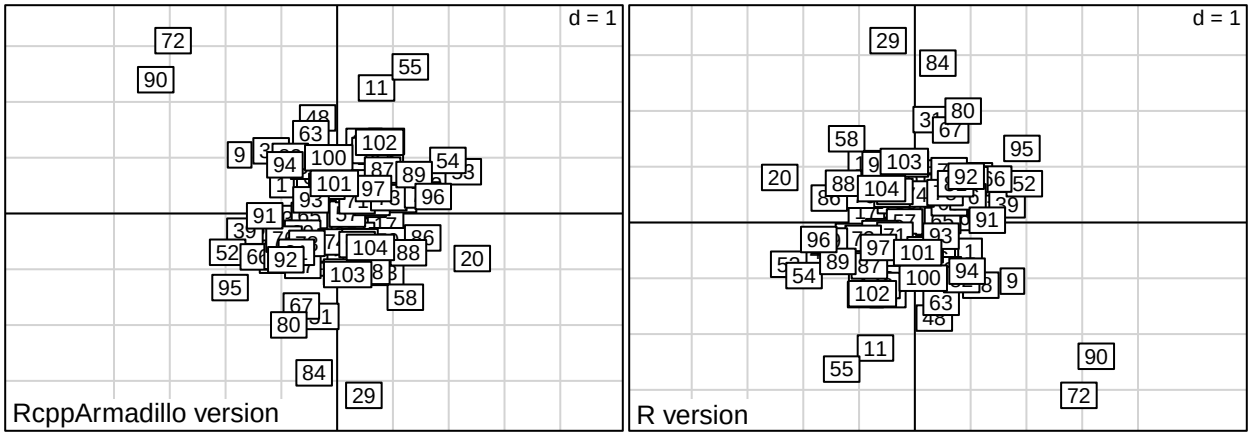
```
## [1] 3.1013578 1.3629834 1.0323269 0.9340533 0.7397529 0.5746693 0.5325414 0.4375395
## [9] 0.2847754
```

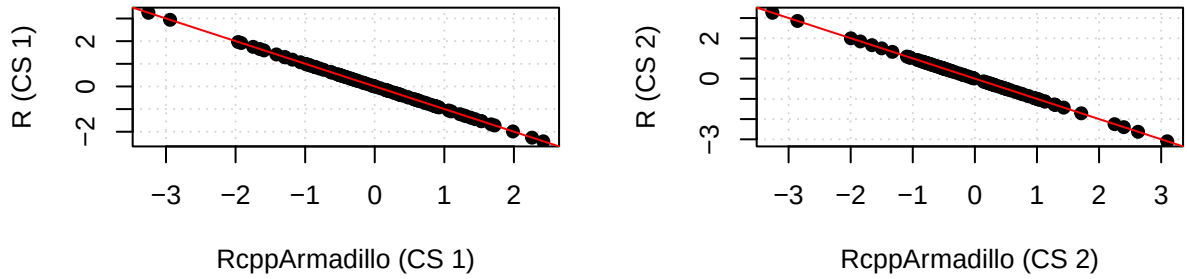
```
coordli <- test$li
par(mfrow=c(2,2))
s.label(coordli,sub="RcppArmadillo version")
s.label(pca2$li,sub="R version")
plot(coordli[,1],pca2$li[,1],panel.first=c(grid()),xlab="RcppArmadillo (CS 1)",
     ylab="R (CS 1)",pch=19);abline(0,-1,col="red")
plot(coordli[,2],pca2$li[,2],panel.first=c(grid()),xlab="RcppArmadillo (CS 2)",
     ylab="R (CS 2)",pch=19);abline(0,-1,col="red")
```



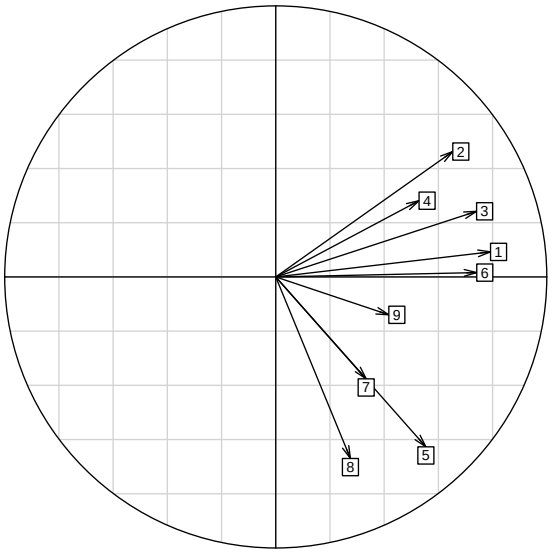



```
coordl1 <- test$l1
par(mfrow=c(2,2))
s.label(coordl1,sub="RcppArmadillo version")
s.label(pca2$l1,sub="R version")
plot(coordl1[,1],pca2$l1[,1],panel.first=c(grid()),xlab="RcppArmadillo (CS 1)",
     ylab="R (CS 1)",pch=19);abline(0,-1,col="red")
plot(coordl1[,2],pca2$l1[,2],panel.first=c(grid()),xlab="RcppArmadillo (CS 2)",
     ylab="R (CS 2)",pch=19);abline(0,-1,col="red")
```

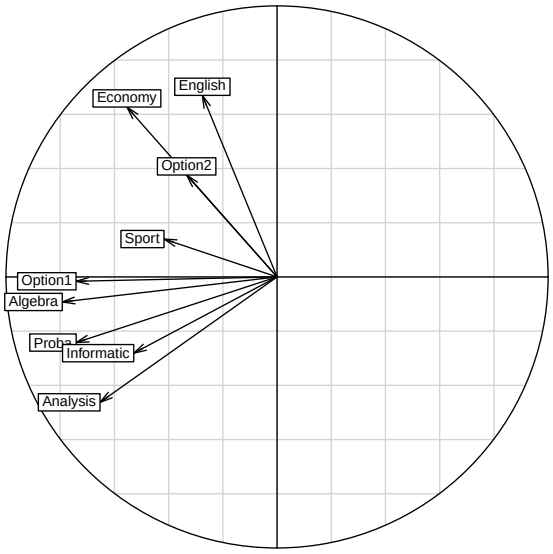




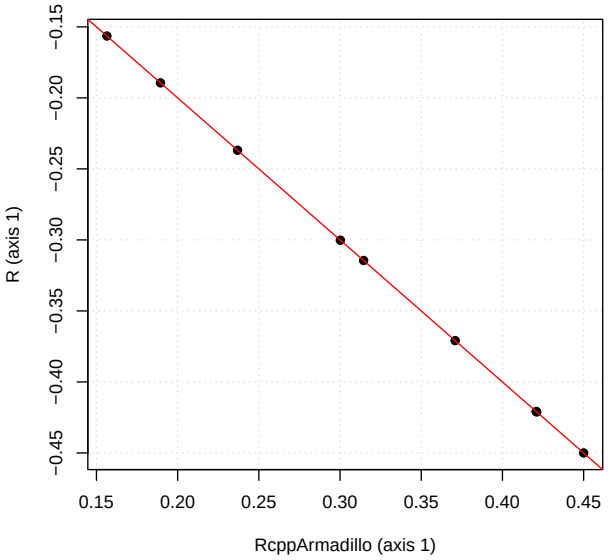

```
par(mfrow=c(2,2))
s.corcircle(test$co,sub="RcppArmadillo version")
s.corcircle(pca2$co,sub="R version")
plot(test$c1[,1],pca2$c1[,1],panel.first=c(grid()),xlab="RcppArmadillo (axis 1)",
     ylab="R (axis 1)",pch=19);abline(0,-1,col="red")
plot(test$c1[,2],pca2$c1[,2],panel.first=c(grid()),xlab="RcppArmadillo (axis 2)",
     ylab="R (axis 2)",pch=19);abline(0,-1,col="red")
```

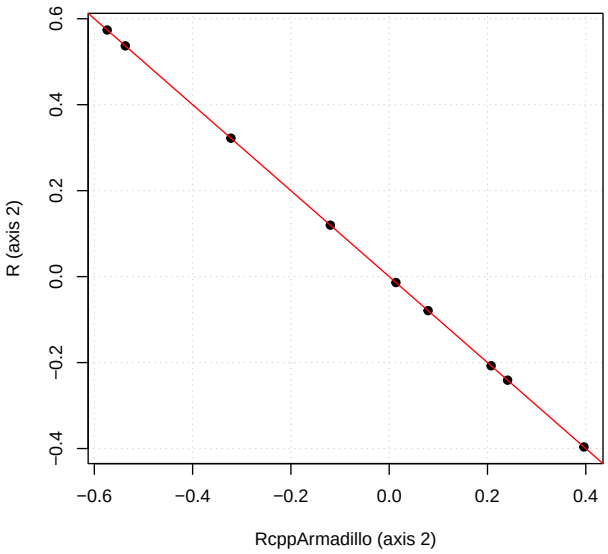


RcppArmadillo version



R version





7.3 Test PCA

argument for the function cpca:

- X: matrix
- cw: numeric vector corresponding to the column weighting
- lw: numeric vector corresponding to the row weighting
- nf: number of selected axes
- center: 1 for TRUE, 0 for FALSE
- scale: 1 for TRUE, 0 for FALSE

```
sourceCpp(file.path(getwd(),"src","utility.cpp"))
test2 <- arc_pca(as.matrix(deug$tab),pca2$cw,pca2$lw,2,1,1)
pca2$eig
```

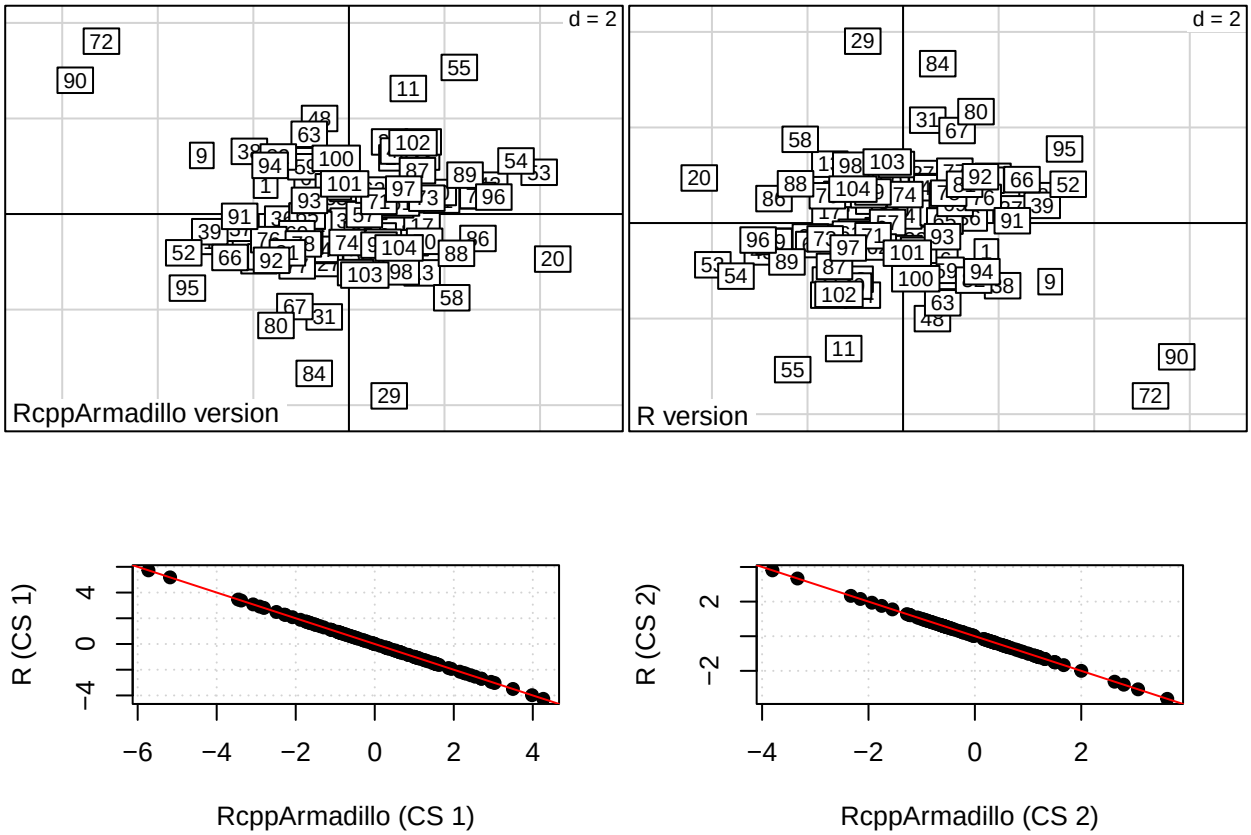
```
## [1] 3.1013578 1.3629834 1.0323269 0.9340533 0.7397529 0.5746693 0.5325414 0.4375395
```

```
## [9] 0.2847754
```

```
test2$eig
```

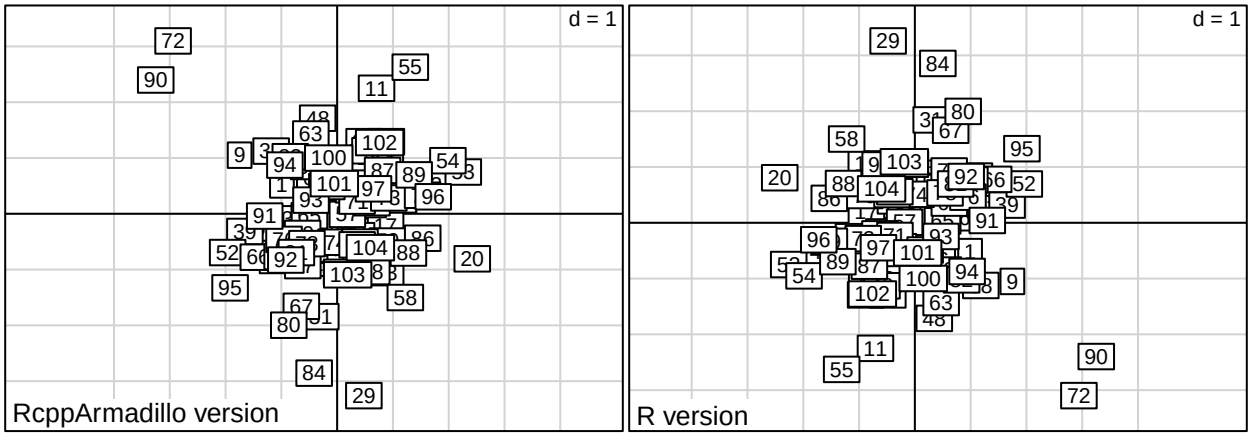
```
##            [,1]
##  [1,] 3.1013578
##  [2,] 1.3629834
##  [3,] 1.0323269
##  [4,] 0.9340533
##  [5,] 0.7397529
##  [6,] 0.5746693
##  [7,] 0.5325414
##  [8,] 0.4375395
##  [9,] 0.2847754
```

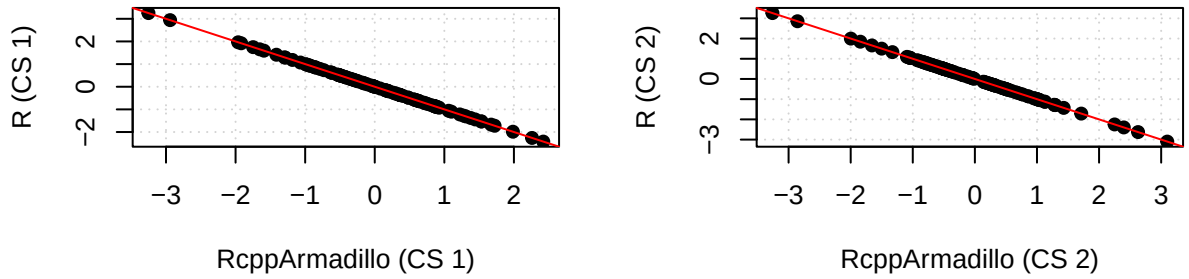
```
coordli <- test2$li
par(mfrow=c(2,2))
s.label(coordli,sub="RcppArmadillo version")
s.label(pca2$li,sub="R version")
plot(coordli[,1],pca2$li[,1],panel.first=c(grid()),xlab="RcppArmadillo (CS 1)",
     ylab="R (CS 1)",pch=19);abline(0,-1,col="red")
plot(coordli[,2],pca2$li[,2],panel.first=c(grid()),xlab="RcppArmadillo (CS 2)",
     ylab="R (CS 2)",pch=19);abline(0,-1,col="red")
```



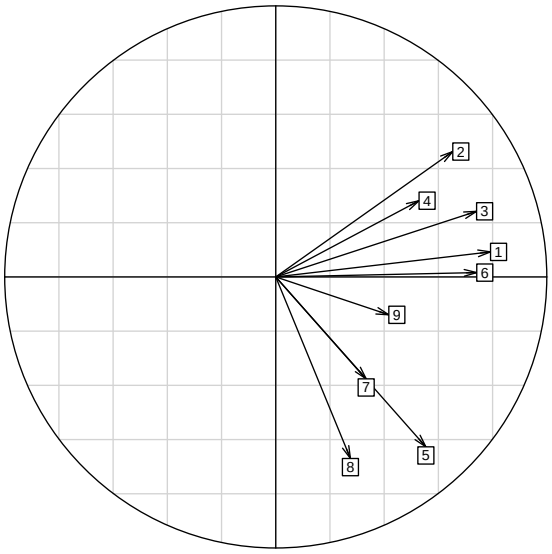

```
coordl1 <- test2$l1
par(mfrow=c(2,2))
s.label(coordl1,sub="RcppArmadillo version")
s.label(pca2$l1,sub="R version")
plot(coordl1[,1],pca2$l1[,1],panel.first=c(grid()),xlab="RcppArmadillo (CS 1)",
```

```
     ylab="R (CS 1)",pch=19);abline(0,-1,col="red")
plot(coordl1[,2],pca2$l1[,2],panel.first=c(grid()),xlab="RcppArmadillo (CS 2)",
     ylab="R (CS 2)",pch=19);abline(0,-1,col="red")
```

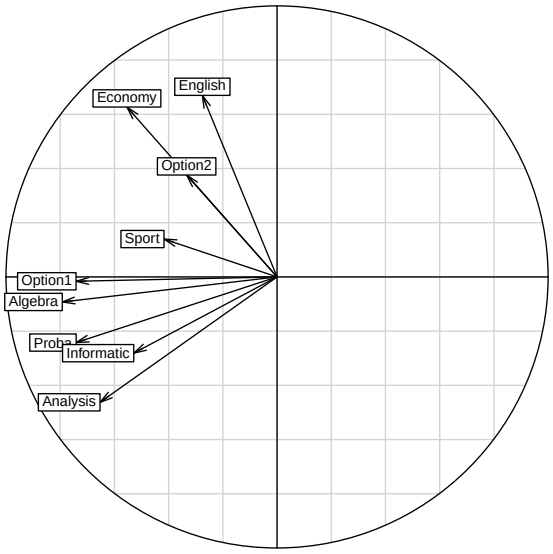




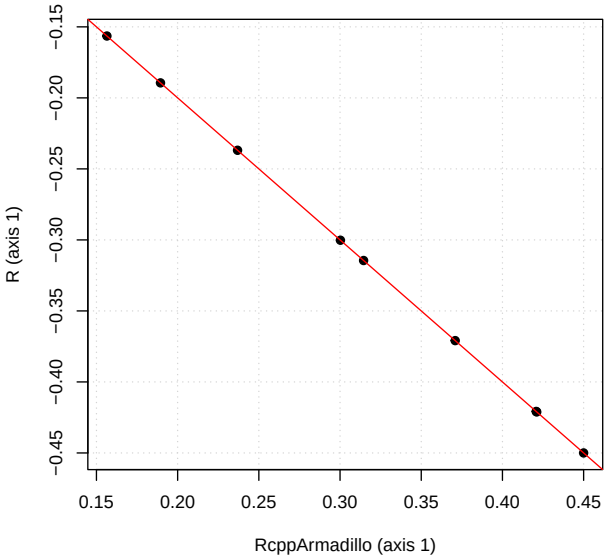

```
par(mfrow=c(2,2))
s.corcircle(test2$co,sub="RcppArmadillo version")
s.corcircle(pca2$co,sub="R version")
plot(test2$c1[,1],pca2$c1[,1],panel.first=c(grid()),xlab="RcppArmadillo (axis 1)",
     ylab="R (axis 1)",pch=19);abline(0,-1,col="red")
plot(test2$c1[,2],pca2$c1[,2],panel.first=c(grid()),xlab="RcppArmadillo (axis 2)",
     ylab="R (axis 2)",pch=19);abline(0,-1,col="red")
```

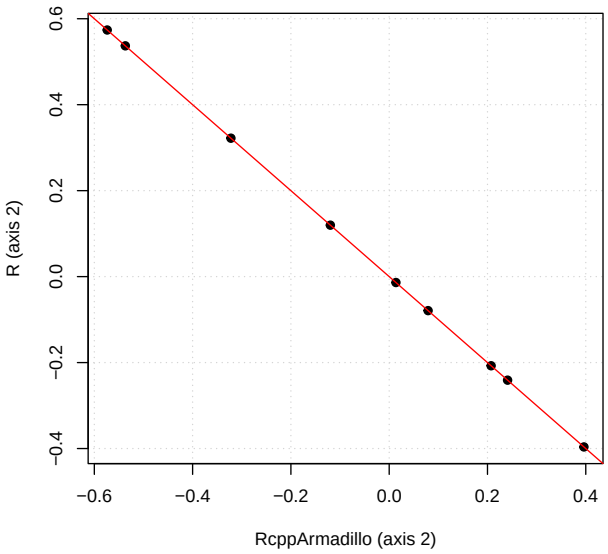


RcppArmadillo version



R version





7.4 Test COA

```
sourceCpp(file.path(getwd(),"src","utility.cpp"))
data(rpjdl)
chisq.test(rpjdl$fau)$statistic
```

```
## Warning in chisq.test(rpjdl$fau): L'approximation du Chi-2 est peut-être incorrecte
```

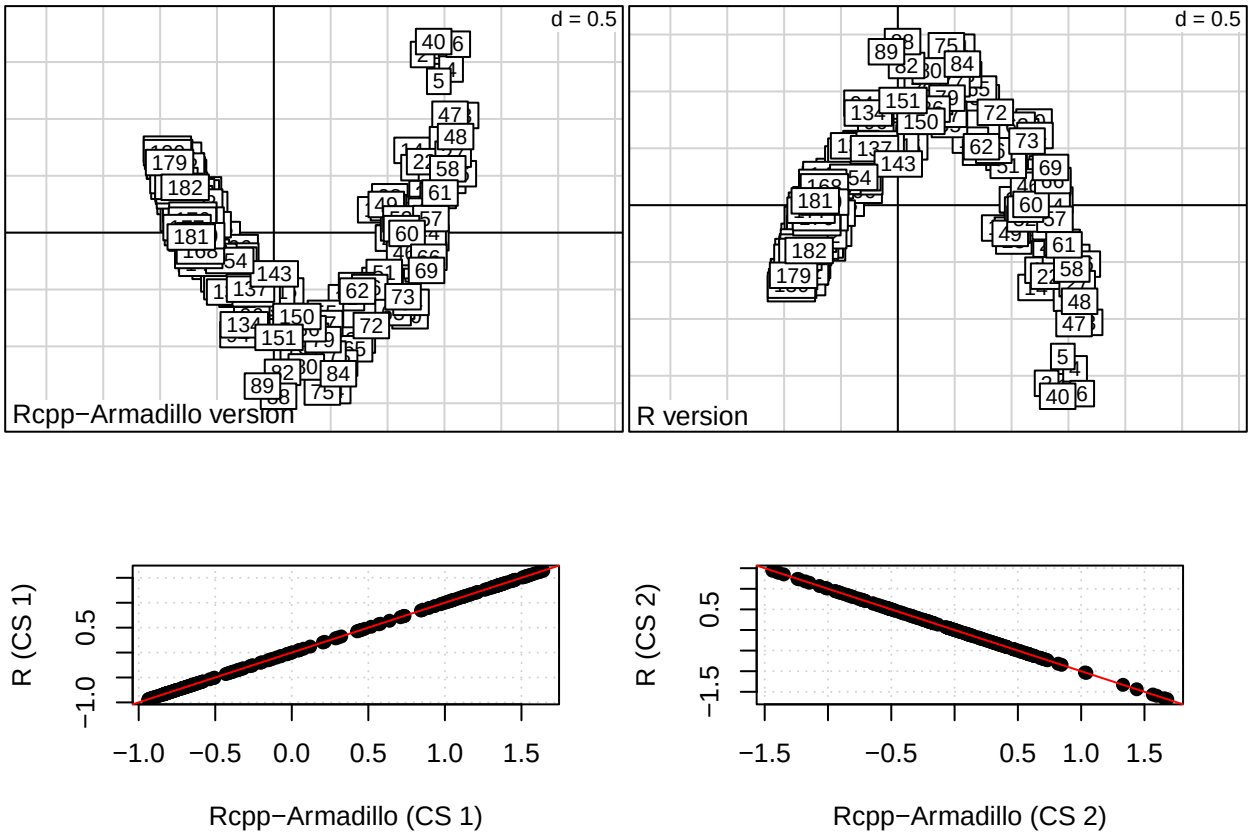
```
## X-squared
##  7323.597
```

```
rpjdl.coa <- coa1 <- dudi.coa(rpjdl$fau, scannf = FALSE, nf = 4)
sum(rpjdl.coa$eig)*rpjdl.coa$N # the same
```

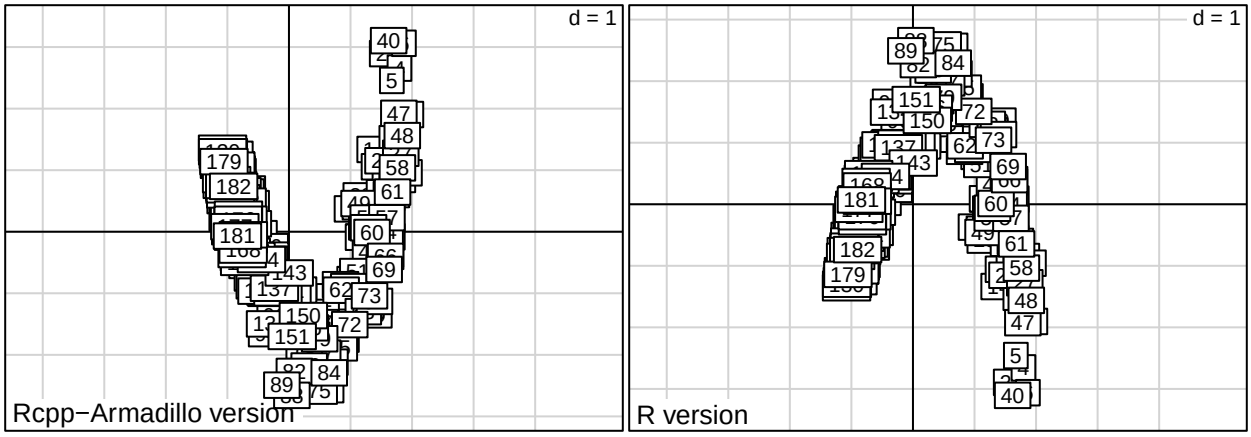
```
## [1] 7323.597
```

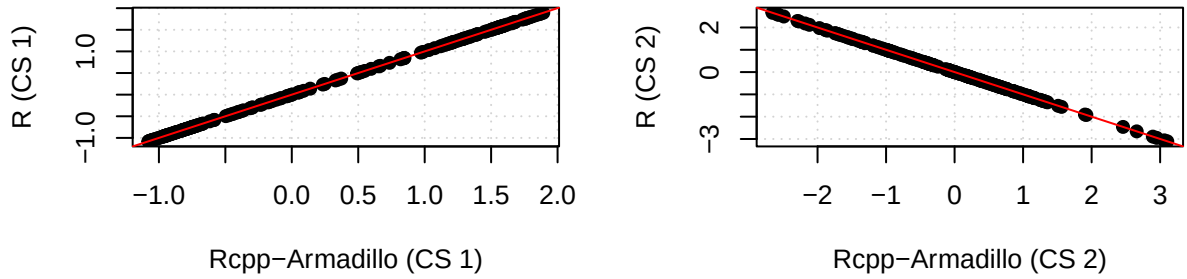
```
test <- arc_coa(as.matrix(rpjdl$fau),nf=2)
```

```
coordli <- test$li
par(mfrow=c(2,2))
s.label(coordli,sub="Rcpp-Armadillo version")
s.label(coa1$li,sub="R version")
plot(coordli[,1],coa1$li[,1],panel.first=c(grid()),xlab="Rcpp-Armadillo (CS 1)",
     ylab="R (CS 1)",pch=19);abline(0,1,col="red")
plot(coordli[,2],coa1$li[,2],panel.first=c(grid()),xlab="Rcpp-Armadillo (CS 2)",
     ylab="R (CS 2)",pch=19);abline(0,-1,col="red")
```

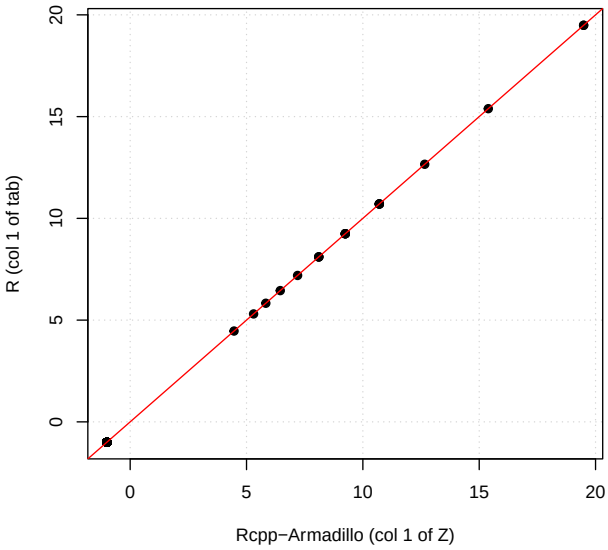


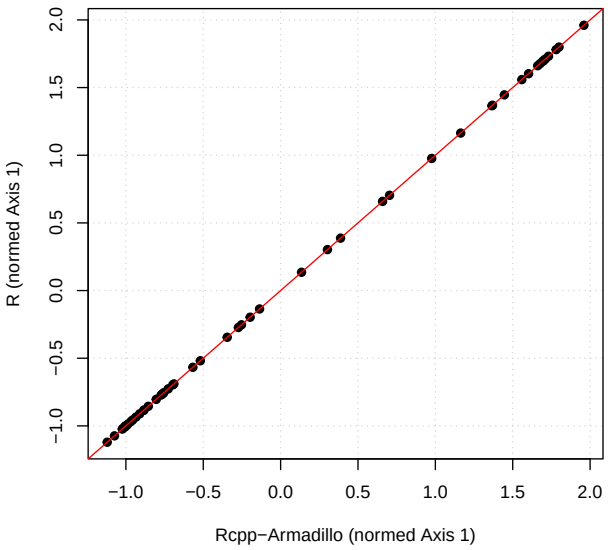

```
coordl1 <- test$l1
par(mfrow=c(2,2))
s.label(coordl1,sub="Rcpp-Armadillo version")
s.label(coa1$l1,sub="R version")
plot(coordl1[,1],coa1$l1[,1],panel.first=c(grid()),xlab="Rcpp-Armadillo (CS 1)",
     ylab="R (CS 1)",pch=19);abline(0,1,col="red")
plot(coordl1[,2],coa1$l1[,2],panel.first=c(grid()),xlab="Rcpp-Armadillo (CS 2)",
     ylab="R (CS 2)",pch=19);abline(0,-1,col="red")
```

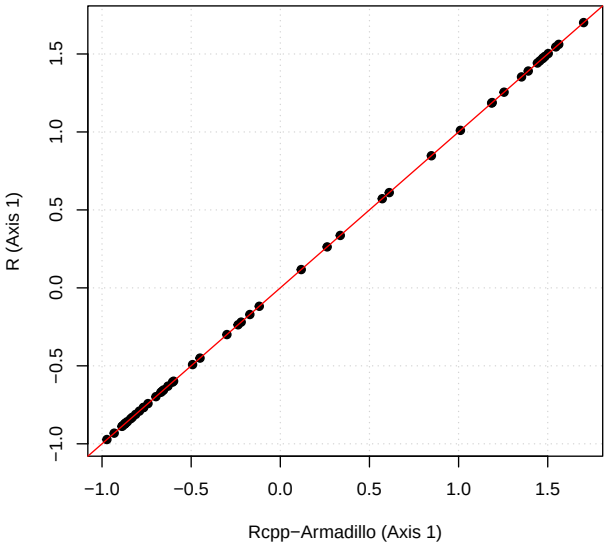





```
par(mfrow=c(2,2))
plot(test$X[,1],coa1$tab[,1],panel.first=c(grid()),xlab="Rcpp-Armadillo (col 1 of Z)",
     ylab="R (col 1 of tab)",pch=19);abline(0,1,col="red")
plot(test$c1[,1],coa1$c1[,1],panel.first=c(grid()),xlab="Rcpp-Armadillo (normed Axis 1)",
     ylab="R (normed Axis 1)",pch=19);abline(0,1,col="red")
plot(test$co[,1],coa1$co[,1],panel.first=c(grid()),xlab="Rcpp-Armadillo (Axis 1)",
     ylab="R (Axis 1)",pch=19);abline(0,1,col="red")
```







7.5 Test MCA

```
data(ours)
summary(ours)
```

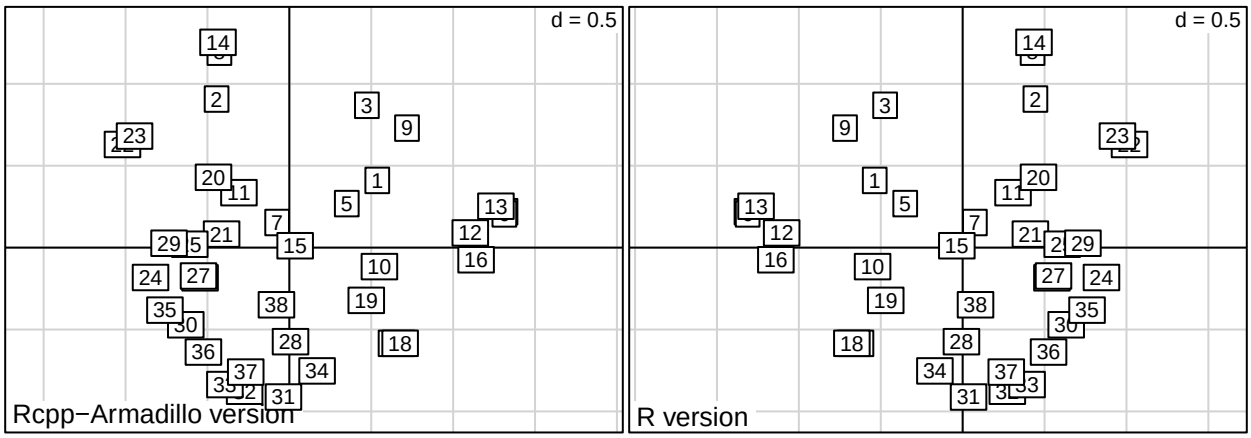
```
##  altit  deniv  cloiso domain boise  hetra  favor  inexp  citat  depart
##  1: 8   1:13   1:12   1: 9   1:10   1:19   1:15   1:20   1:22   AHP:5
##  2:17   2:14   2: 4   2:13   2:15   2: 5   2:12   2:10   2: 7   AM :4
##  3:13   3:11   3:22   3:16   3:13   3:14   3:11   3: 8   3: 4   D  :5
##                                                       4: 5   HP :8
##                                                              HS :4
##                                                              I  :5
##                                                              S  :7
```

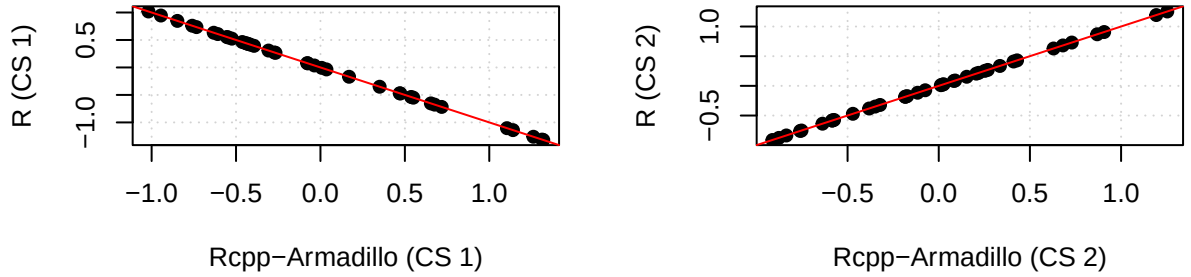
```
acm1 <- dudi.acm(ours, scan = FALSE)
```

for the first version of mca, we give the **disjonctif** table and the variable number.

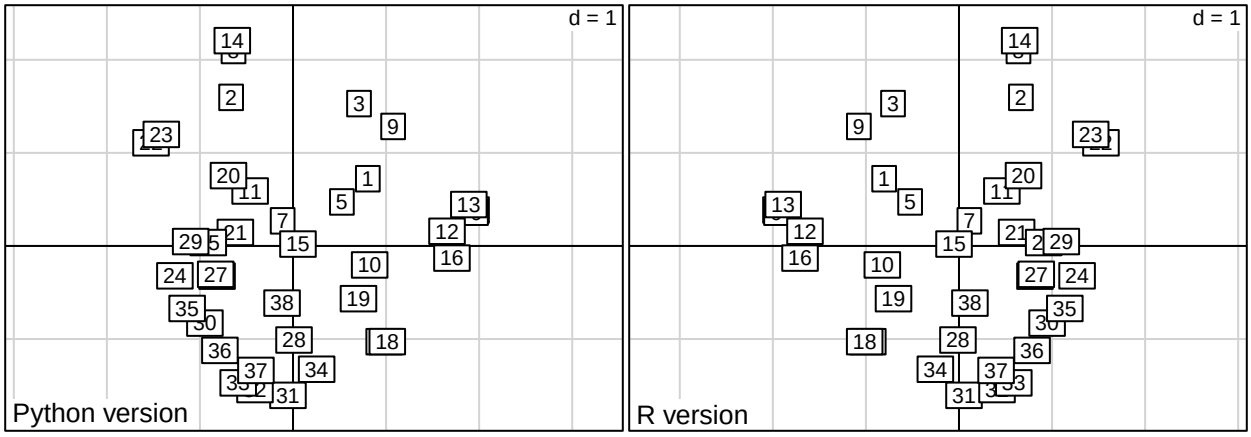
```
sourceCpp(file.path(getwd(),"src","utility.cpp"))
test <- arc_mca_proto(as.matrix(acm.disjonctif(ours)),lw=acm1$lw,v=ncol(ours),nf=2)
```

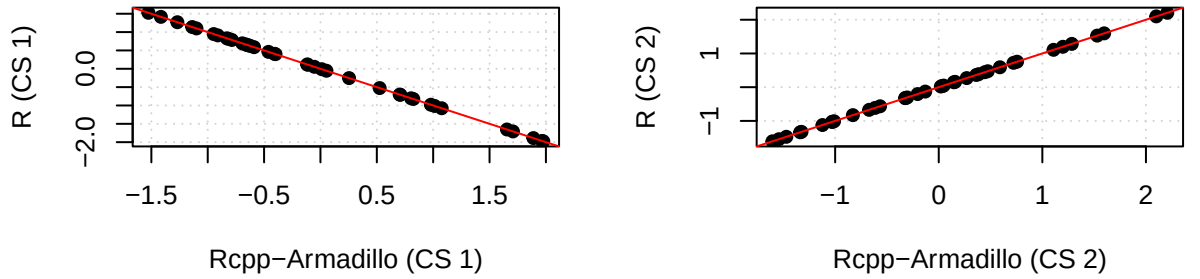
```
coordli <- test$li
par(mfrow=c(2,2))
s.label(coordli,sub="Rcpp-Armadillo version")
s.label(acm1$li,sub="R version")
plot(coordli[,1],acm1$li[,1],panel.first=c(grid()),xlab="Rcpp-Armadillo (CS 1)",
     ylab="R (CS 1)",pch=19);abline(0,-1,col="red")
plot(coordli[,2],acm1$li[,2],panel.first=c(grid()),xlab="Rcpp-Armadillo (CS 2)",
     ylab="R (CS 2)",pch=19);abline(0,1,col="red")
```



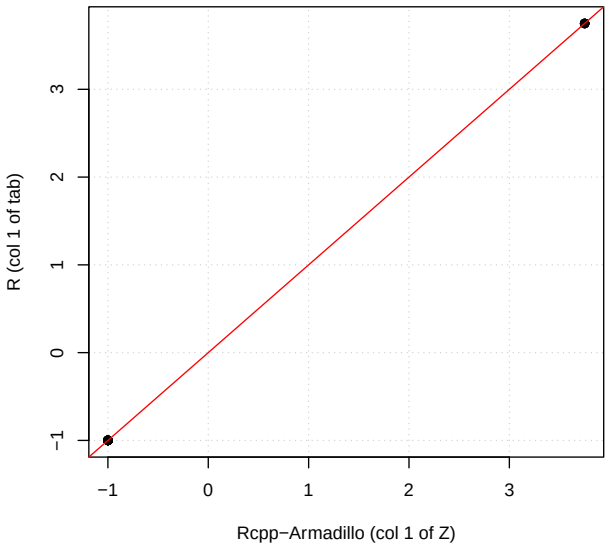


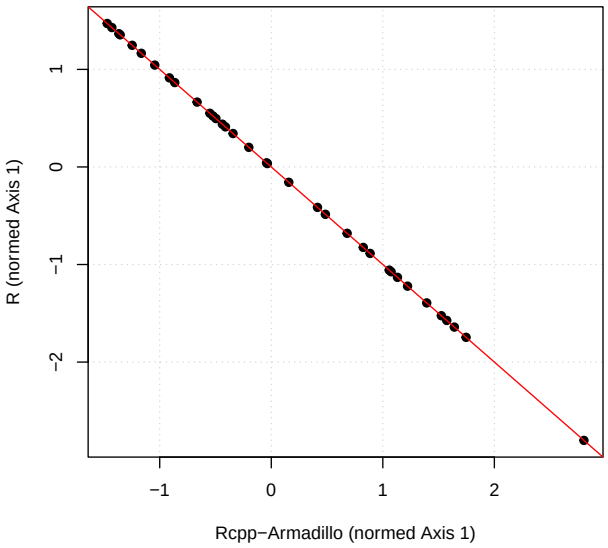

```
coordl1 <- test$l1
par(mfrow=c(2,2))
s.label(coordl1,sub="Python version")
s.label(acm1$l1,sub="R version")
plot(coordl1[,1],acm1$l1[,1],panel.first=c(grid()),xlab="Rcpp-Armadillo (CS 1)",
     ylab="R (CS 1)",pch=19);abline(0,-1,col="red")
plot(coordl1[,2],acm1$l1[,2],panel.first=c(grid()),xlab="Rcpp-Armadillo (CS 2)",
     ylab="R (CS 2)",pch=19);abline(0,1,col="red")
```

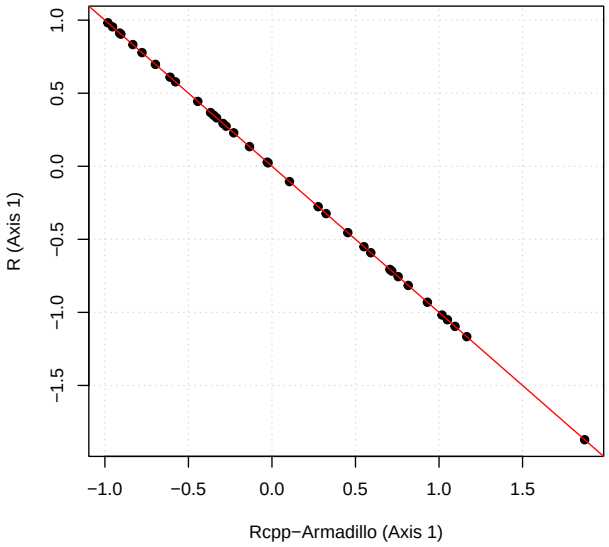





```
par(mfrow=c(2,2))
plot(test$X[,1],acm1$tab[,1],panel.first=c(grid()),xlab="Rcpp-Armadillo (col 1 of Z)",
     ylab="R (col 1 of tab)",pch=19);abline(0,1,col="red")
plot(test$c1[,1],acm1$c1[,1],panel.first=c(grid()),xlab="Rcpp-Armadillo (normed Axis 1)",
     ylab="R (normed Axis 1)",pch=19);abline(0,-1,col="red")
plot(test$co[,1],acm1$co[,1],panel.first=c(grid()),xlab="Rcpp-Armadillo (Axis 1)",
     ylab="R (Axis 1)",pch=19);abline(0,-1,col="red")
```







encoding **disjonctif** table

```
sourceCpp(file.path(getwd(),"src","utility.cpp"))
w <- factor(c("a","a","b","c"))
acm.util(w)
```

```
##      a b c
## [1,] 1 0 0
## [2,] 1 0 0
## [3,] 0 1 0
## [4,] 0 0 1
```

```
arc_acmutil(w)
```

```
##      [,1] [,2] [,3]
## [1,]    1    0    0
## [2,]    1    0    0
## [3,]    0    1    0
```

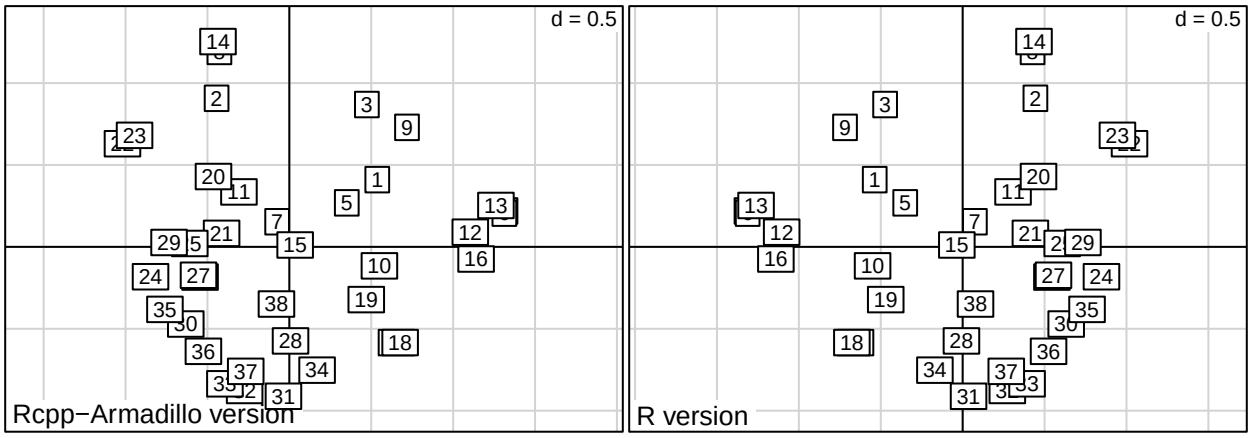
```
## [4,]    0    0    1
```

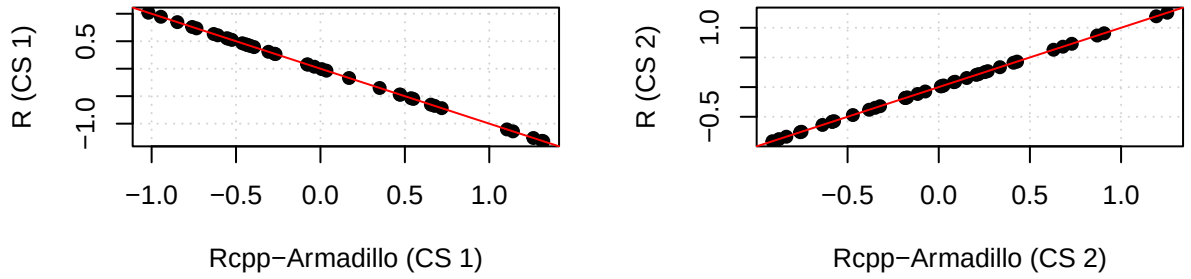
```
w1 <- factor(c("e","d","e","f"))
arc_disjonctif(cbind(w,w1))
```

```
##      [,1] [,2] [,3] [,4] [,5] [,6]
## [1,]    1    0    0    0    1    0
## [2,]    1    0    0    1    0    0
## [3,]    0    1    0    0    1    0
## [4,]    0    0    1    0    0    1
```

```
test <- arc_mca( as.matrix(apply(ours,2,function(x) as.numeric(factor(x)))),
                 lw=acm1$lw,nf=2)
```

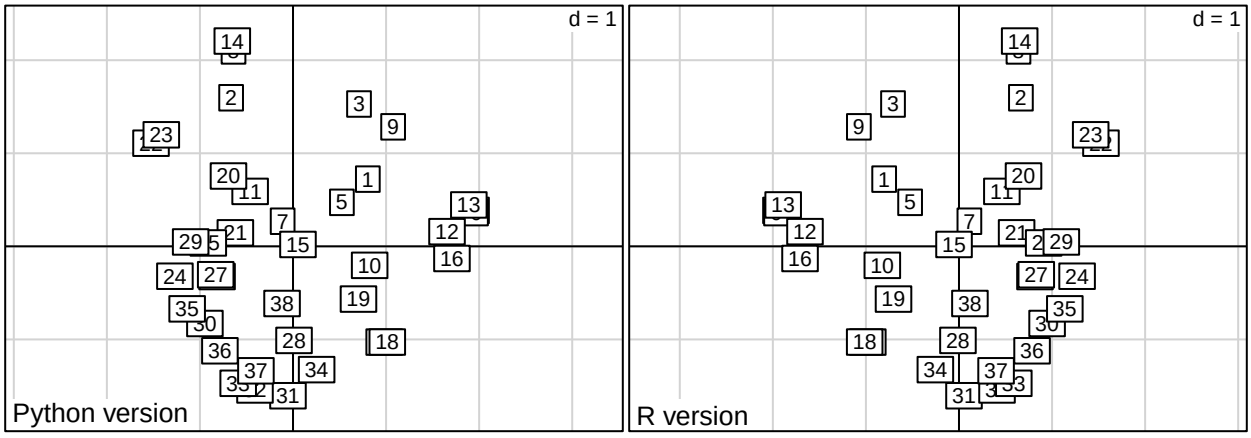
```
coordli <- test$li
par(mfrow=c(2,2))
s.label(coordli,sub="Rcpp-Armadillo version")
s.label(acm1$li,sub="R version")
plot(coordli[,1],acm1$li[,1],panel.first=c(grid()),xlab="Rcpp-Armadillo (CS 1)",
     ylab="R (CS 1)",pch=19);abline(0,-1,col="red")
plot(coordli[,2],acm1$li[,2],panel.first=c(grid()),xlab="Rcpp-Armadillo (CS 2)",
     ylab="R (CS 2)",pch=19);abline(0,1,col="red")
```

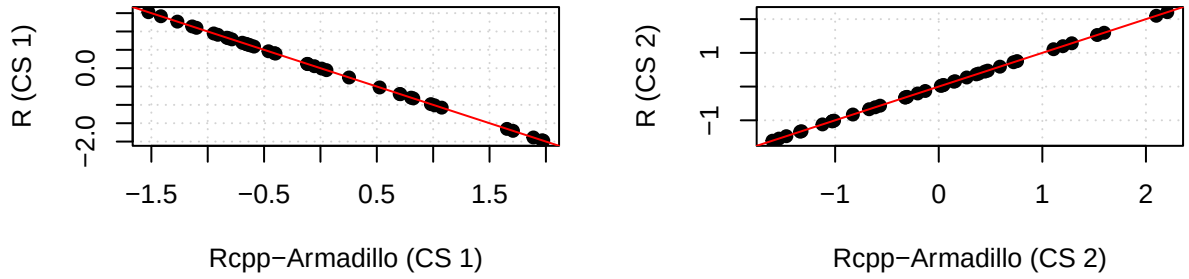




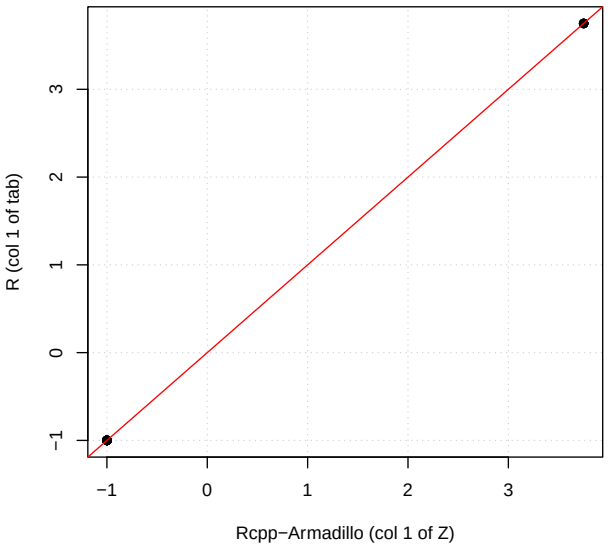

```
coordl1 <- test$l1
par(mfrow=c(2,2))
s.label(coordl1,sub="Python version")
s.label(acm1$l1,sub="R version")
plot(coordl1[,1],acm1$l1[,1],panel.first=c(grid()),xlab="Rcpp-Armadillo (CS 1)",
     ylab="R (CS 1)",pch=19);abline(0,-1,col="red")
```

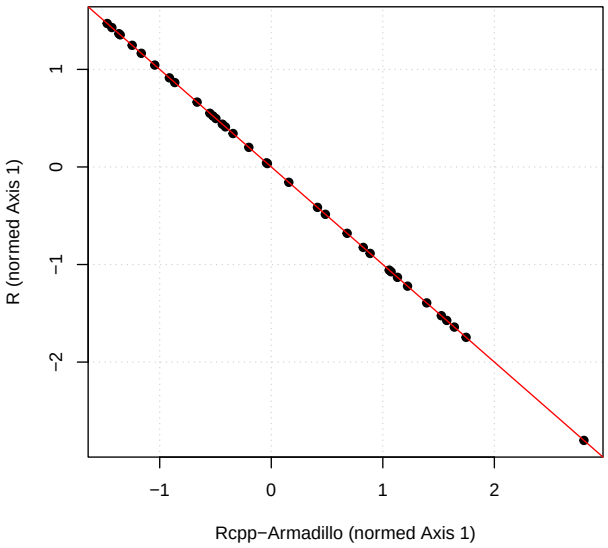
```
plot(coordl1[,2],acm1$l1[,2],panel.first=c(grid()),xlab="Rcpp-Armadillo (CS 2)",
     ylab="R (CS 2)",pch=19);abline(0,1,col="red")
```

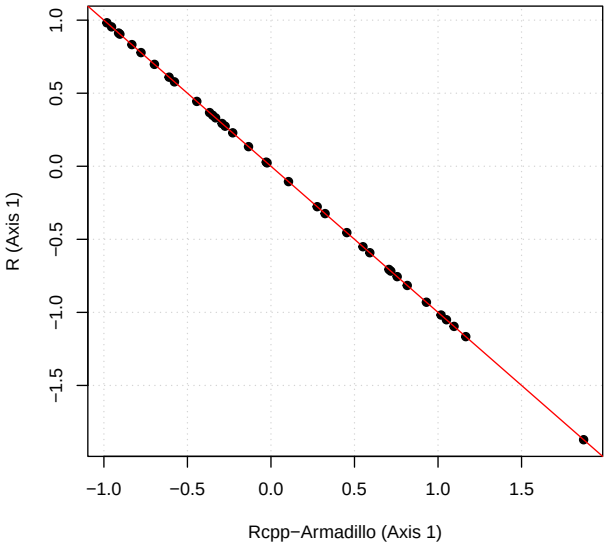





```
par(mfrow=c(2,2))
plot(test$X[,1],acm1$tab[,1],panel.first=c(grid()),xlab="Rcpp-Armadillo (col 1 of Z)",
     ylab="R (col 1 of tab)",pch=19);abline(0,1,col="red")
plot(test$c1[,1],acm1$c1[,1],panel.first=c(grid()),xlab="Rcpp-Armadillo (normed Axis 1)",
     ylab="R (normed Axis 1)",pch=19);abline(0,-1,col="red")
plot(test$co[,1],acm1$co[,1],panel.first=c(grid()),xlab="Rcpp-Armadillo (Axis 1)",
     ylab="R (Axis 1)",pch=19);abline(0,-1,col="red")
```







8 References

- R http://www.R-project.org/.
- R Core Team (2013). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL http://www.R-project.org/.
- Dray S, Dufour A (2007). "The ade4 Package: Implementing the Duality Diagram for Ecologists." *Journal of Statistical Software*, *22*(4), 1-20. doi:10.18637/jss.v022.i04 https://doi.org/10.18637/jss.v022.i04.
- Chessel D, Dufour A, Thioulouse J (2004). "The ade4 Package - I: One-Table Methods." *R News*, *4*(1), 5-10. https://cran.r-project.org/doc/Rnews/.
- Dray S, Dufour A, Chessel D (2007). "The ade4 Package - II: Two-Table and K-Table Methods." *R News*, *7*(2), 47-52. https://cran.r-project.org/doc/Rnews/.
- Thioulouse J, Dray S, Dufour A, Siberchicot A, Jombart T, Pavoine S (2018). *Multivariate Analysis of Ecological Data with ade4*. Springer. doi:10.1007/978-1-4939-8850-1 https://doi.org/10.1007/978-1-4939-8850-1.

- Escoufier, Y. (1987) The duality diagram : a means of better practical applications In Development in numerical ecology, Legendre, P. & Legendre, L. (Eds.) NATO advanced Institute, Serie G. Springer Verlag, Berlin, 139–156.
- Benzécri, J.P. and Coll. (1973) *L'analyse des données. II L'analyse des correspondances*, Bordas, Paris. 1-620.
- Greenacre, M. J. (1984) *Theory and applications of correspondence analysis*, Academic Press, London.
- Tenenhaus, M. & Young, F.W. (1985) An analysis and synthesis of multiple correspondence analysis, optimal scaling, dual scaling, homogeneity analysis ans other methods for quantifying categorical multivariate data. Psychometrika, 50, 1, 91-119.
- Lebart, L., A. Morineau, and M. Piron. 1995. Statistique exploratoire multidimensionnelle. Dunod, Paris.

9 Appendix

9.1 Additional R functions

```
source("/export/scratch/GITprojects/pbtools/trunk/Rcode/Rgraphics-0.1.R")
```

9.2 Session information

```
print(sessionInfo(),locale=FALSE)
```

```
## R version 4.3.2 (2023-10-31)
## Platform: x86_64-pc-linux-gnu (64-bit)
## Running under: Linux Mint 20.3
##
## Matrix products: default
## BLAS:   /usr/lib/x86_64-linux-gnu/blas/libblas.so.3.9.0
## LAPACK: /usr/lib/x86_64-linux-gnu/lapack/liblapack.so.3.9.0
##
## attached base packages:
##  [1] parallel  datasets  stats     graphics  utils     stats4    tools     grDevices
##  [9] methods   base
##
## other attached packages:
##  [1] RcppArmadillo_0.12.2.0.0    Rcpp_1.0.10
##  [3] reticulate_1.28             circlize_0.4.15
##  [5] minfi_1.46.0                bumphunter_1.42.0
##  [7] locfit_1.5-9.7              iterators_1.0.14
##  [9] foreach_1.5.2               Biostrings_2.68.0
## [11] XVector_0.40.0              SummarizedExperiment_1.30.1
## [13] Biobase_2.60.0              MatrixGenerics_1.12.0
## [15] matrixStats_0.63.0          xtable_1.8-4
## [17] tinytex_0.45                rmarkdown_2.21
## [19] knitr_1.42                  pixmap_0.4-12
## [21] ade4_1.7-22                 RColorBrewer_1.1-3
## [23] rtracklayer_1.60.0          GenomicRanges_1.52.0
## [25] GenomeInfoDb_1.36.0         IRanges_2.34.0
## [27] S4Vectors_0.38.1            BiocGenerics_0.46.0
##
## loaded via a namespace (and not attached):
##   [1] jsonlite_1.8.4           shape_1.4.6               rstudioapi_0.15.0
##   [4] magrittr_2.0.3           GenomicFeatures_1.52.0    GlobalOptions_0.1.2
```

```
##   [7] BiocIO_1.10.0             zlibbioc_1.46.0           vctrs_0.6.2
##  [10] multtest_2.56.0           memoise_2.0.1             Rsamtools_2.16.0
##  [13] DelayedMatrixStats_1.22.0 RCurl_1.98-1.12           askpass_1.1
##  [16] htmltools_0.5.5           S4Arrays_1.0.1            progress_1.2.2
##  [19] curl_5.0.0                rjags_4-14                Rhdf5lib_1.22.0
##  [22] rhdf5_2.44.0              nor1mix_1.3-0             plyr_1.8.8
##  [25] cachem_1.0.8              GenomicAlignments_1.36.0  lifecycle_1.0.3
##  [28] pkgconfig_2.0.3           Matrix_1.5-4              R6_2.5.1
##  [31] fastmap_1.1.1             GenomeInfoDbData_1.2.10   digest_0.6.31
##  [34] colorspace_2.1-0          siggenes_1.74.0           reshape_0.8.9
##  [37] AnnotationDbi_1.62.1      rprojroot_2.0.3           RSQLite_2.3.1
##  [40] base64_2.0.1              filelock_1.0.2            fansi_1.0.4
##  [43] httr_1.4.6                compiler_4.3.2            beanplot_1.3.1
##  [46] here_1.0.1                rngtools_1.5.2            bit64_4.0.5
##  [49] BiocParallel_1.34.1       DBI_1.1.3                 highr_0.10
##  [52] HDF5Array_1.28.1          biomaRt_2.56.0            MASS_7.3-60
##  [55] openssl_2.0.6             rappdirs_0.3.3            DelayedArray_0.26.2
##  [58] rjson_0.2.21              glue_1.6.2                quadprog_1.5-8
##  [61] restfulr_0.0.15           nlme_3.1-162              rhdf5filters_1.12.1
##  [64] grid_4.3.2                generics_0.1.3            gtable_0.3.3
##  [67] tzdb_0.3.0                preprocessCore_1.62.1     tidyr_1.3.0
##  [70] data.table_1.14.8         hms_1.1.3                 xml2_1.3.4
##  [73] utf8_1.2.3                pillar_1.9.0              stringr_1.5.0
##  [76] limma_3.56.1              genefilter_1.82.1         splines_4.3.2
##  [79] dplyr_1.1.2               BiocFileCache_2.8.0       lattice_0.21-8
##  [82] survival_3.5-5            bit_4.0.5                 GEOquery_2.68.0
##  [85] annotate_1.78.0           tidyselect_1.2.0          xfun_0.39
##  [88] scrime_1.3.5              stringi_1.7.12            yaml_2.3.7
##  [91] evaluate_0.21             codetools_0.2-19          tibble_3.2.1
##  [94] cli_3.6.1                 munsell_0.5.0             dbplyr_2.3.2
##  [97] coda_0.19-4               png_0.1-8                 XML_3.99-0.14
## [100] ggplot2_3.4.2             readr_2.1.4               blob_1.2.4
## [103] prettyunits_1.1.1         mclust_6.0.0              doRNG_1.8.6
## [106] sparseMatrixStats_1.12.0  bitops_1.0-7              scales_1.2.1
## [109] illuminaio_0.42.0         purrr_1.0.1               crayon_1.5.2
## [112] rlang_1.1.1               KEGGREST_1.40.0
```