

M2 AGES

# Introduction à la Morphométrie

Josef WILCZEK

## **Séance 1 (06/10/2016; 14-18h) – salle S212**

Introduction aux morphométrie géométrique  
I'ACP + I'ACP (TD)

## **Séance 2 (19/10/2016; 14-18h) – salle S212**

Applications de la morphométrie en archéologie  
Quelques études de cas  
DA + DA (TP)

## **Séance 3 (24/10/2016; 8-12h) – salle S06**

Analyses de Procrustes sur monnaies romaines (TP)

## **Séance 4 (25/10/2016; 14-17h) – salle S06**

Exams de la théorie  
Exams pratiques (deadline 18/12)

## **Introduction**

- Problématique archéologique
- Plusieurs types de variables
- Ce qui nous intéresse ici: la forme

## **Collecter les données**

- Distances, angles, etc...
- Contours ouverts
- Contours fermés
- Points homologues

## **Préparation et production des données**

- Bookstein registration
- Méthodes Procrustes
- Techniques de régression
- Méthodes de type Fourier

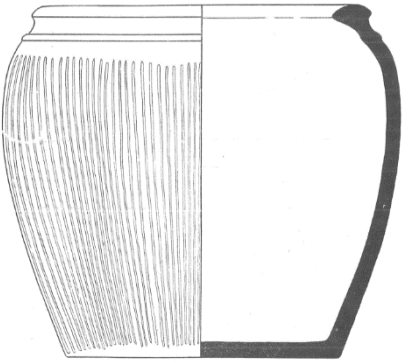
## **Traitement des données (deux techniques parmi d'autres)**

- Analyse en composantes principales, espace morphologique
- Analyse discriminante

Quelles variables?



ID : 80000169  
Type : MACEDONIAN KINGS  
Greek Region  
Issuer : Philip II  
Metal : Gold  
Denomination : Stater  
Struck / Cast : struck  
Date Struck : B.C. 323 - 315  
Weight : 8.58 g  
Diameter: 3.2 cm  
Obverse Description : Laureate head of Apollo  
right Reverse Legend : ΦΙΛΙΠΠΟΥ  
Reverse Description : Charioteer driving big a right, holding kentron in right hand, reins in left; trident below



ID : 9025661  
Type : SITULA  
Region : Europe Centrale  
Pate : Graphitique  
Datation : LT C1  
Weight : 3.4 kg  
Diameter: 30.2 cm  
Volume: 12 L  
Décoration: Vertical lines  
Chaine opératoire : 13



ID : 9025661  
Type : HT 3a  
Region : England  
Composition : 2.5Cu, 9.8Sn  
Datation : HB2  
Weight : 361 g  
Longueur: 15.1 cm  
Hauteur: 4.3 cm  
Décoration: -

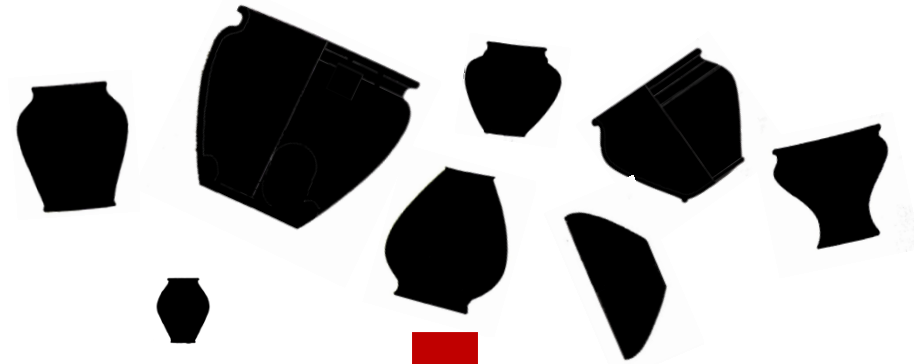


Taille

vs.



Forme



## 1) Collecter des données

## 2) Standardisation

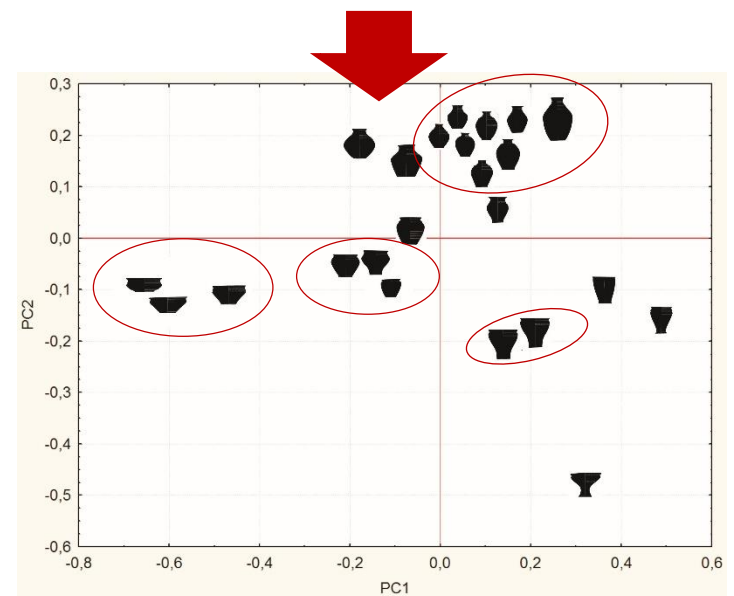
(position, taille et orientation)



## 3) Calcul des variables des formes

(« shape variables »)

## 4) Traitement des données et visualisation

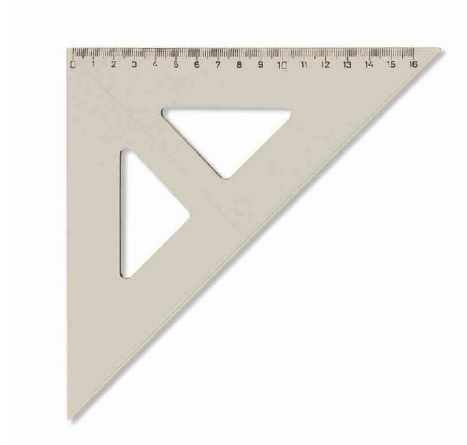


# 1) Collecter des données

## Dessins en 2D, photo



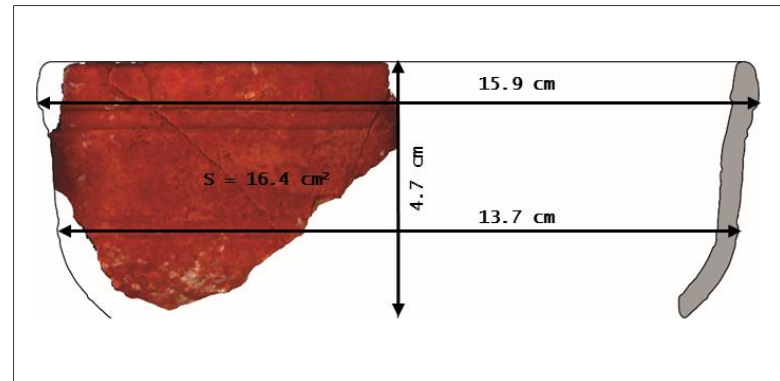
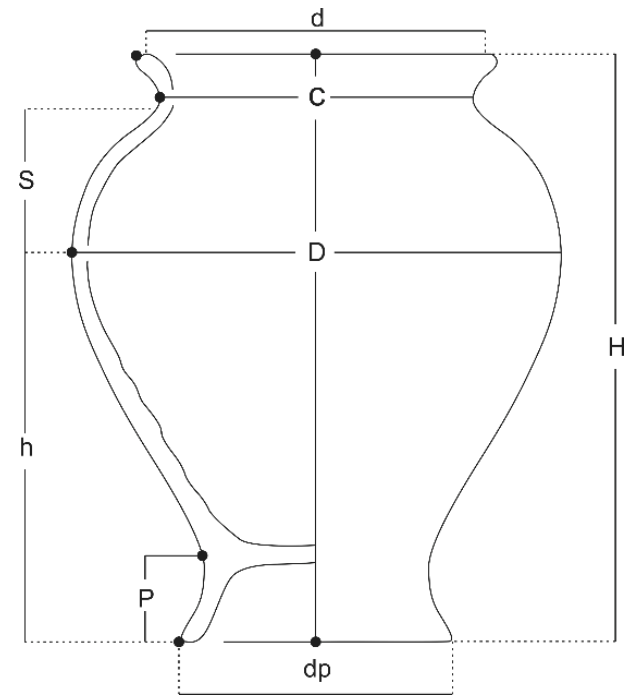
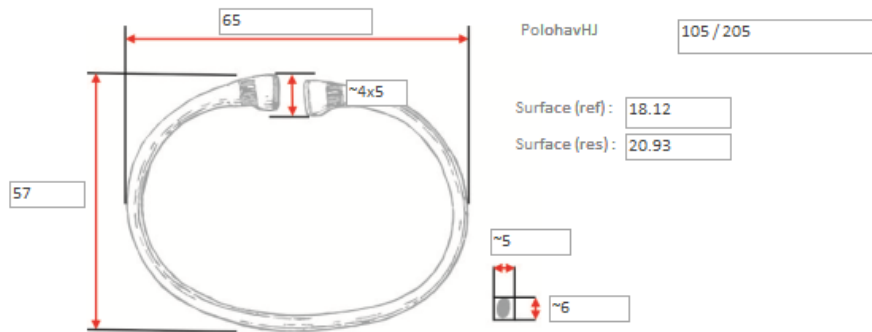
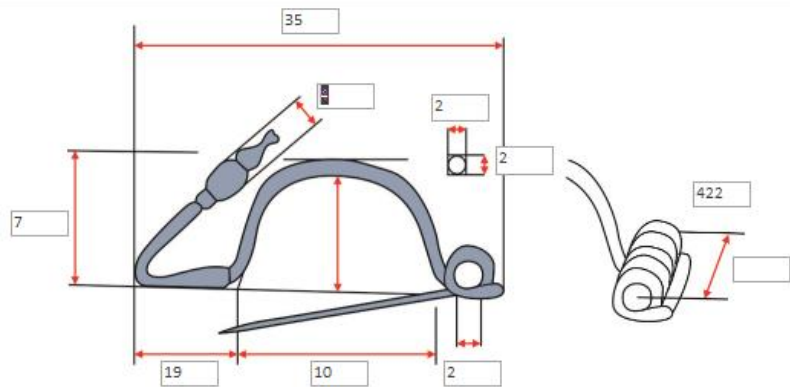
Conformateur



1) Collecter des données

Quoi?

Distances, angles, périmètres, surfaces, volumes, contours, ...





# 1) Collecter des données

On peut utiliser les images...

Pl. 26



P 1a



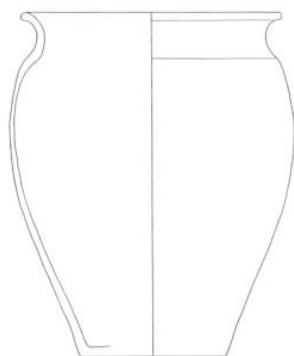
P 1b



P 2a



P 2b



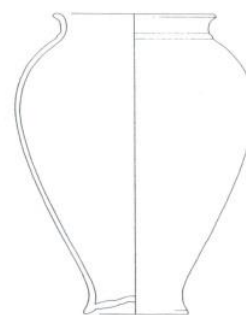
P 3a



P 3b

Ph. B. / T. L. 1995

Pl. 29



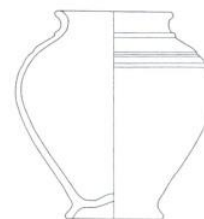
P 12a



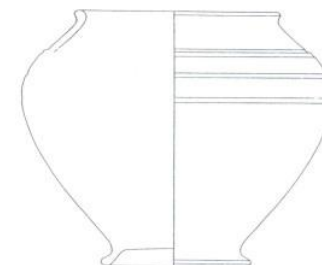
P 12b



P 13



P 14

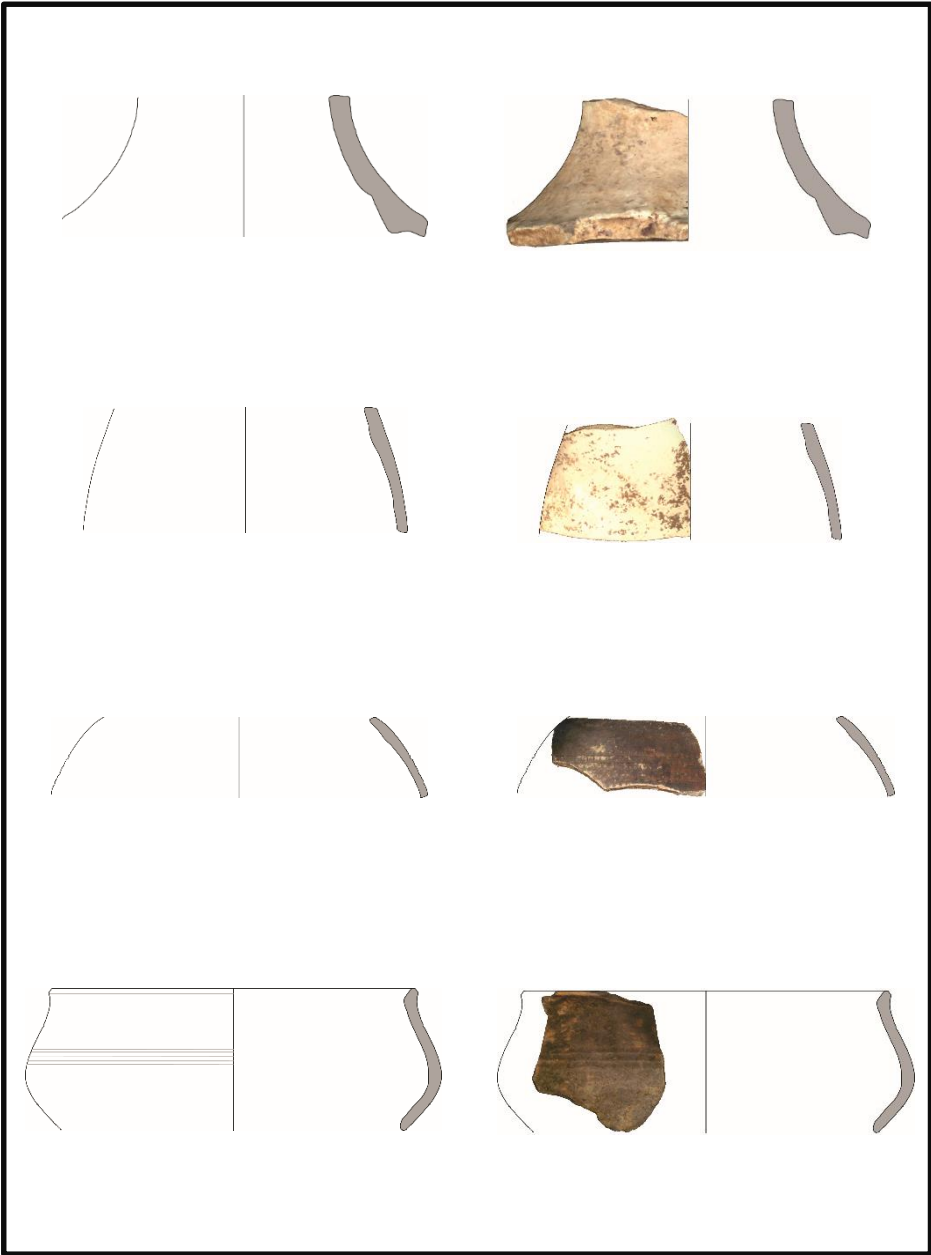
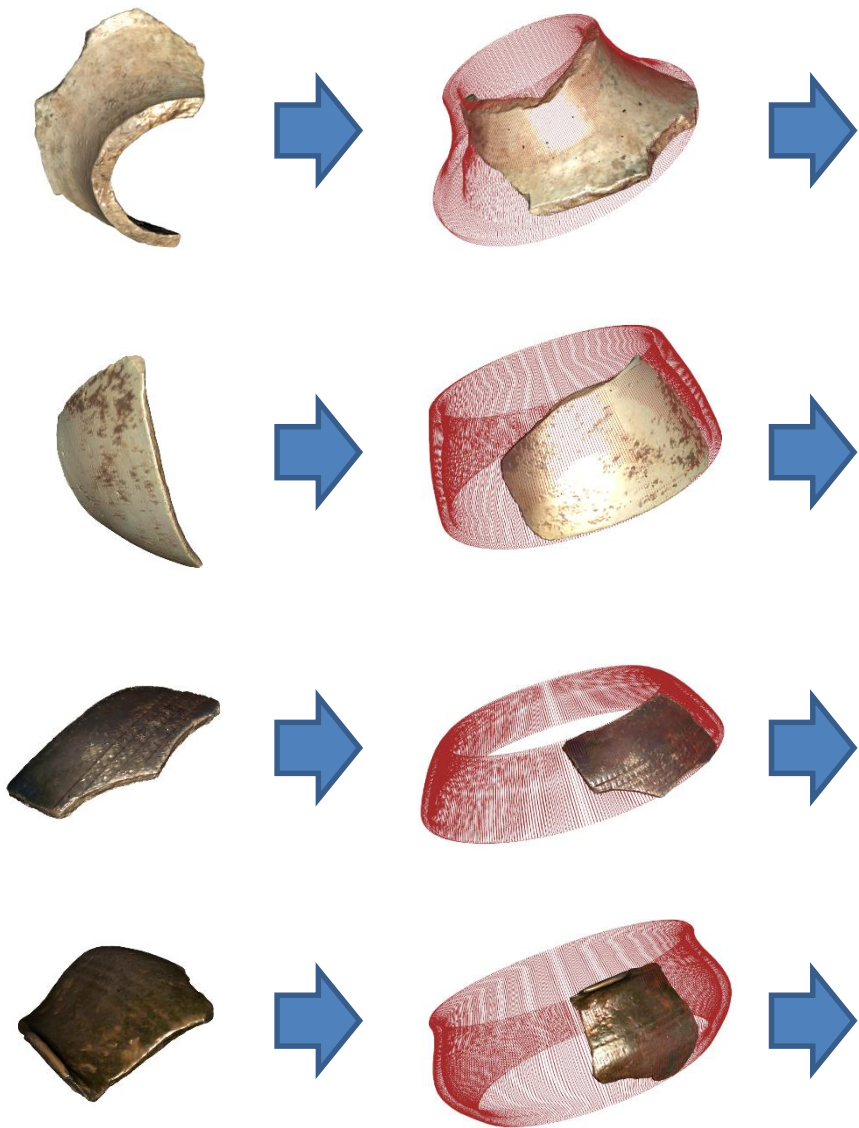


P 15

Ph. B. / T. L. 1995

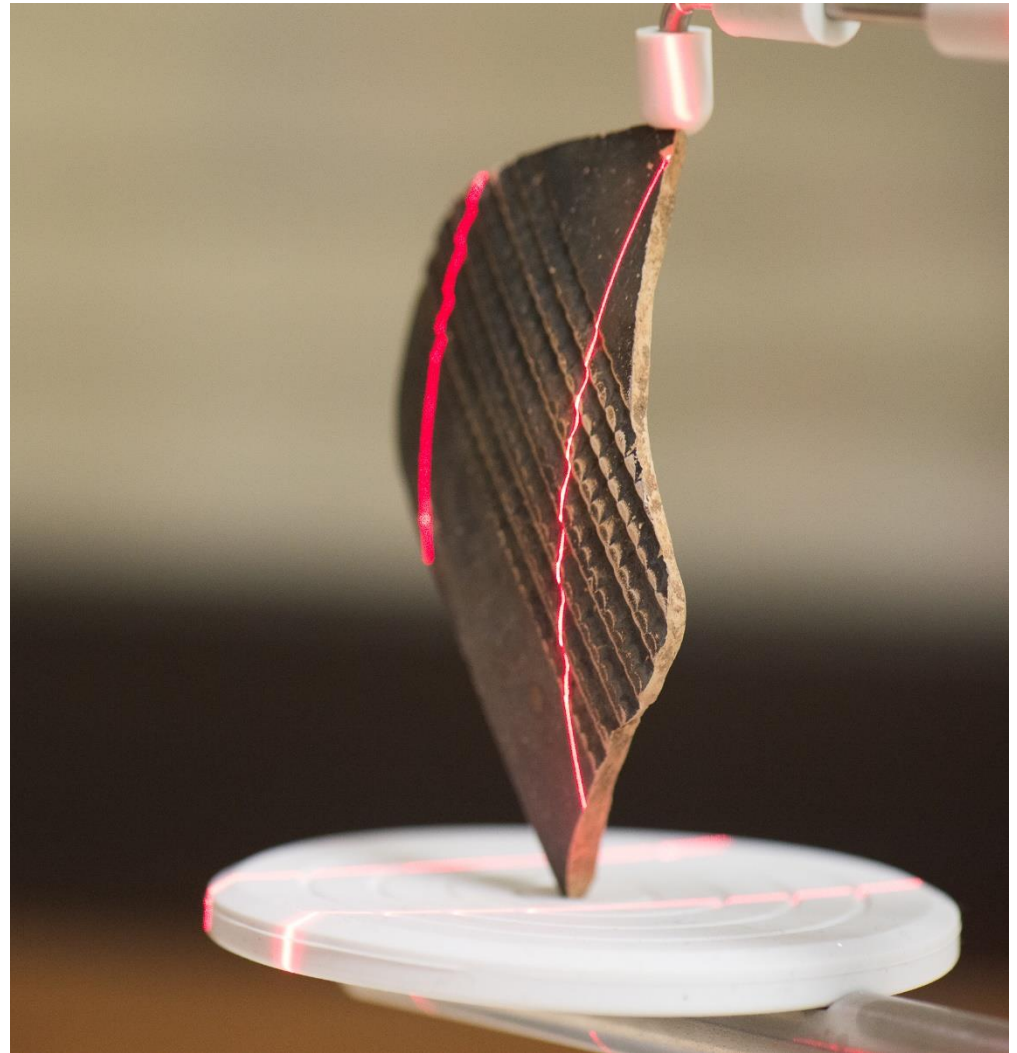
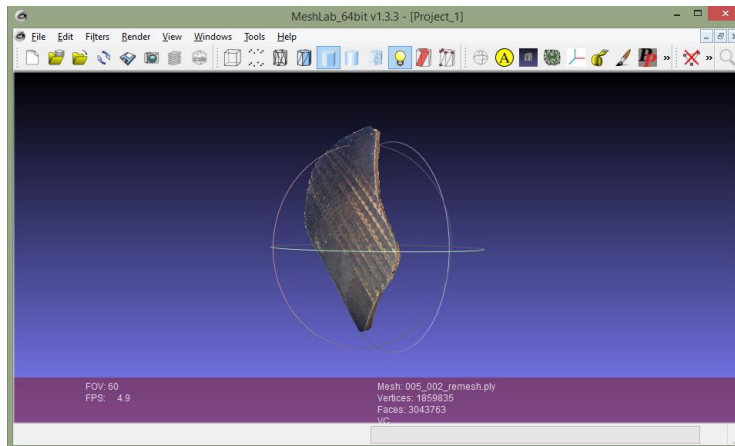
1) Collecter des données

...ou on peut traiter les models 3D



# 1) Collecter des données

## Scanner laser

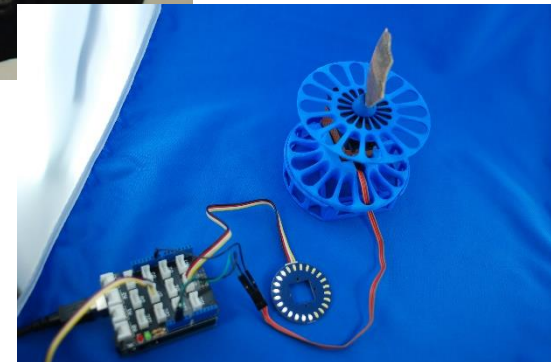


Scanner laser  
Nextengine, Matterform, Fuel3D

# 1) Collecter des données



Photogrammetrie  
PhotoScan



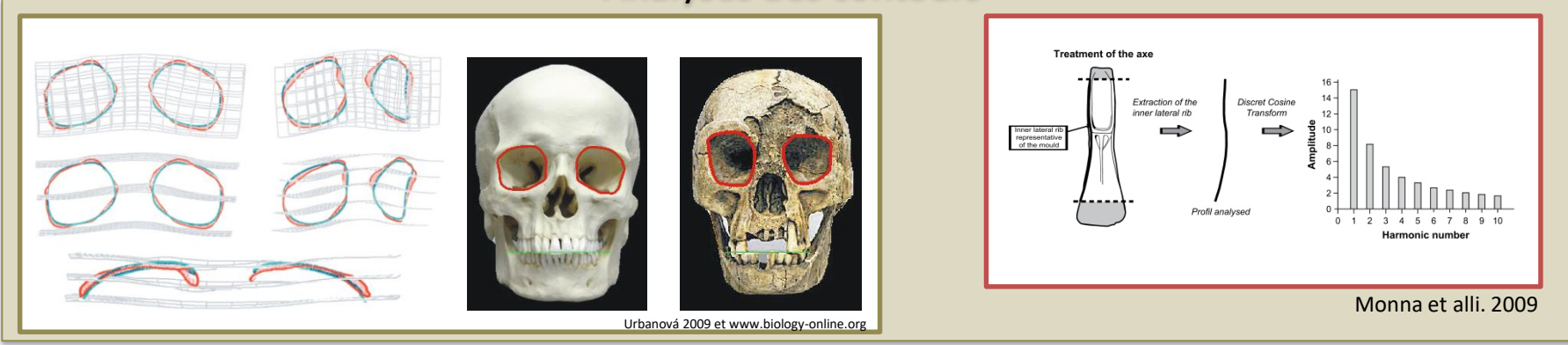


1) Collecter des données – Points homologues et les contours

Analyses des landmarks



Analyses des contours



# 1) Collecter des données – Points homologues

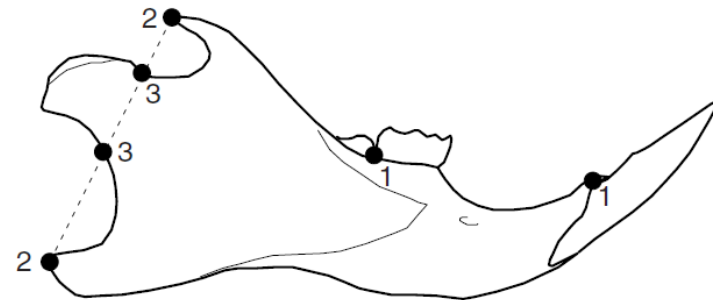
## Bookstein

Trois catégories de points homologues:

Type I = juxtaposition de parties définissant un point simple.

Type II: maximum de courbure

Type III: point extrême, obtenu par construction géométrique, qui peut correspondre à un centroïde, une intersection, avec des segments entre points homologues...



**Fig. 1.2.** The three types of landmarks illustrated on the jaw of a mouse

## Analyses des landmarks

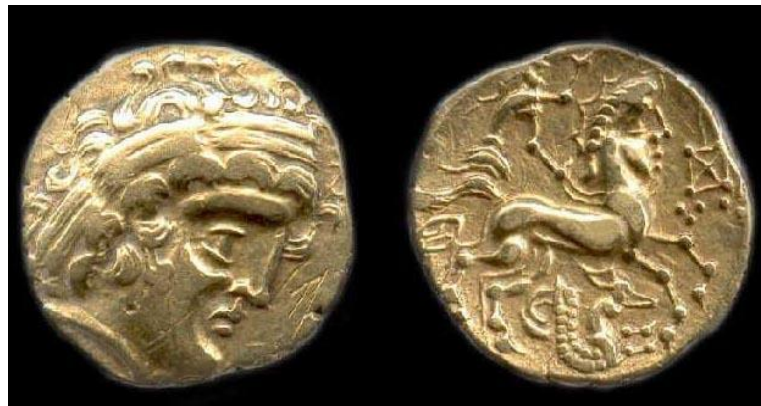


# 1) Collecter des données – Points homologues - Exemple

Un original



Des copies...



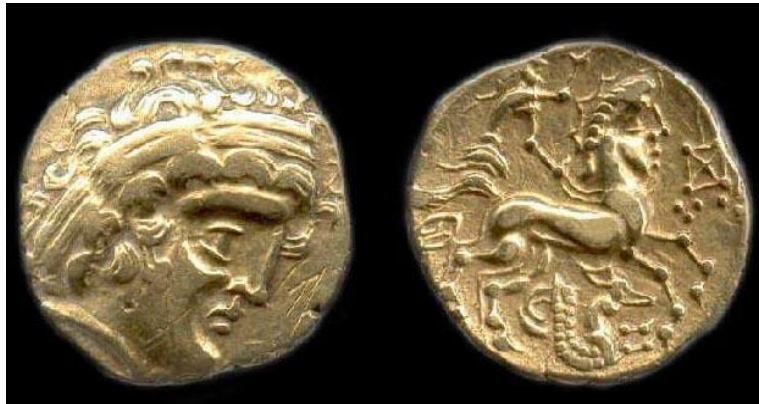


# 1) Collecter des données – Points homologues - Exemple

Un original



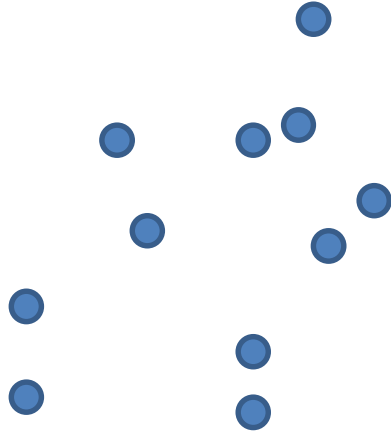
Des copies...



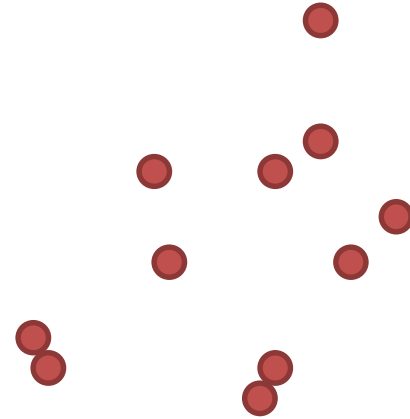


# 1) Collecter des données – Points homologues - Exemple

Un original



Des copies...



1) Collecter des données – Points homologues

Définition des vingt-neuf points repères (*landmarks*) sur le crâne en vue sagittale.

Conformation juvénile en pointillé,  
conformation adulte en trait continu.

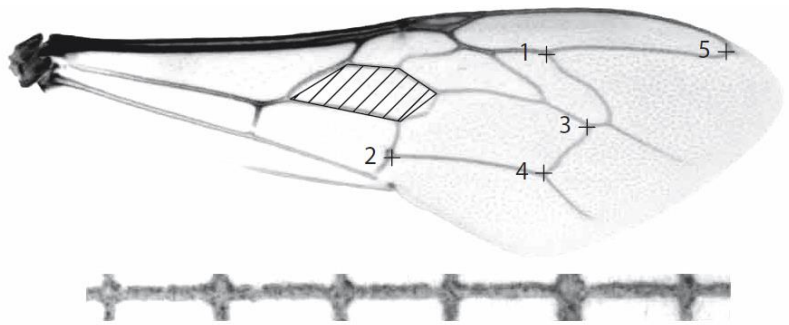
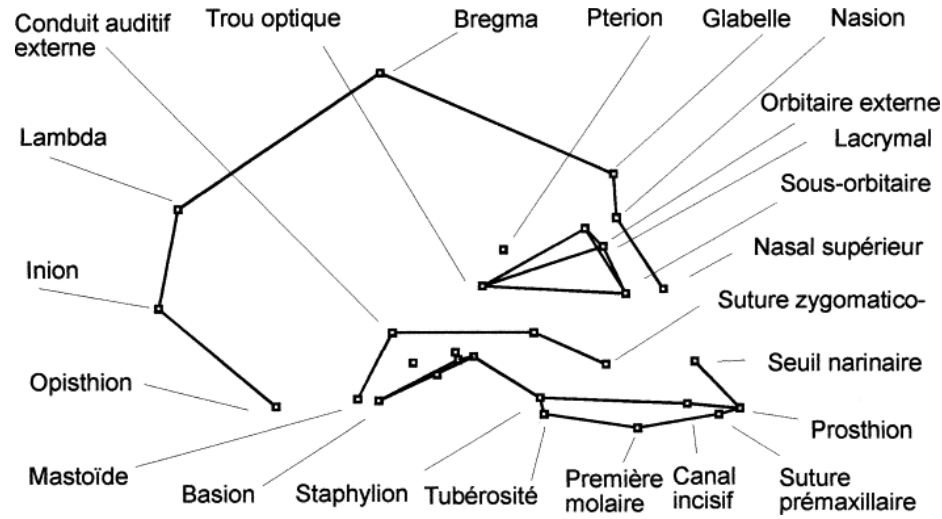
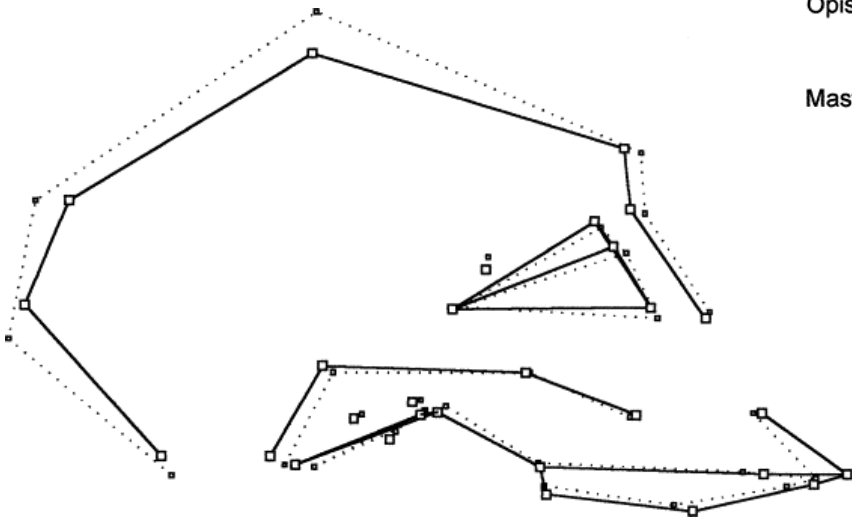
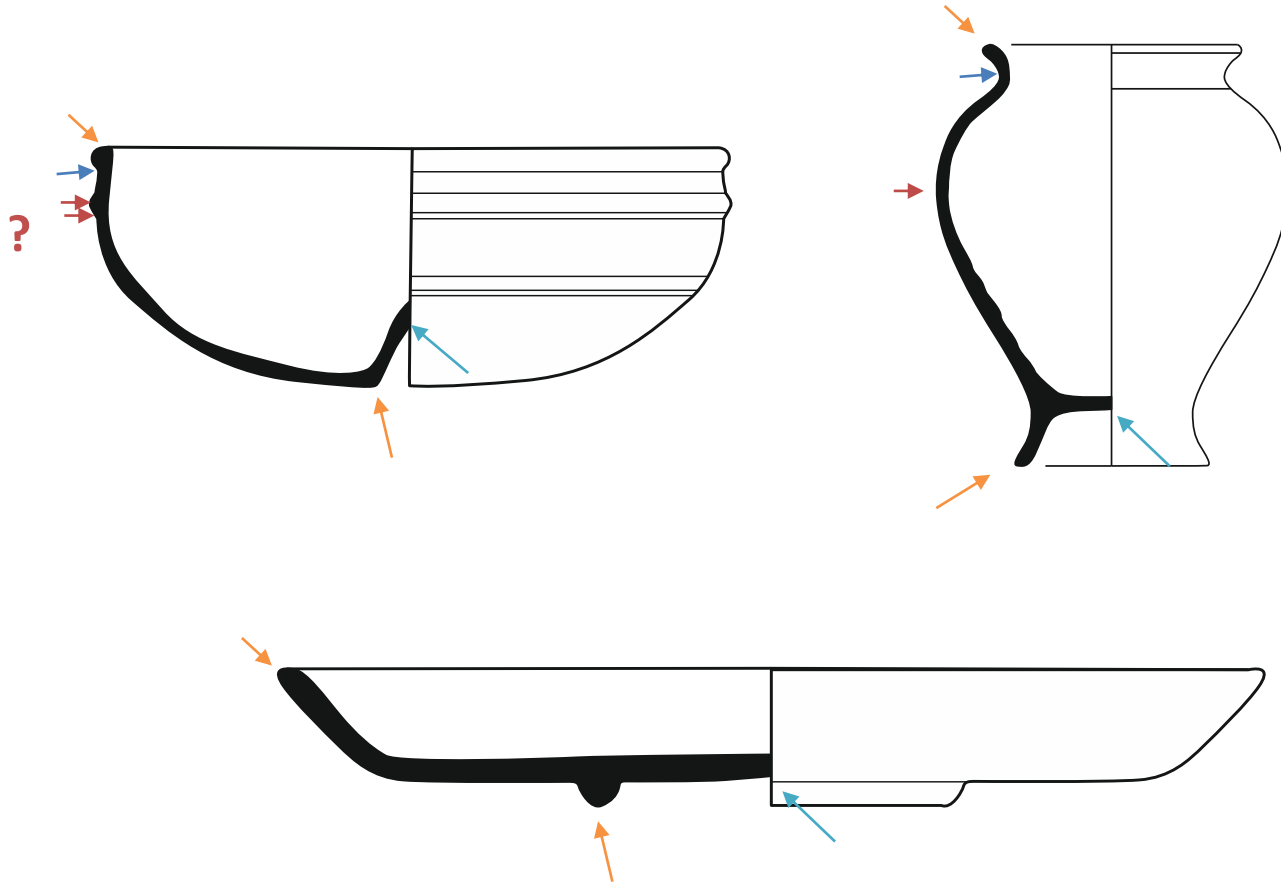


Fig. 2.7. Digitizing landmark locations with R

[x,y]

# 1) Collecter des données – Points homologues

En arqueo...



Limitations...

1) Collecter des données – Contours ouverts

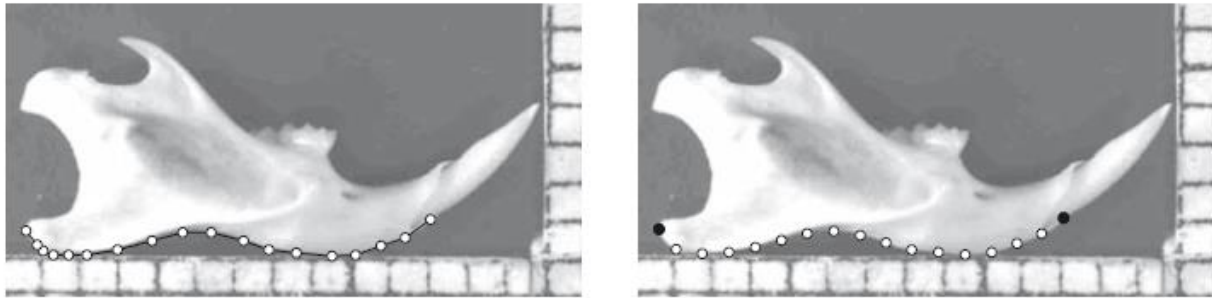
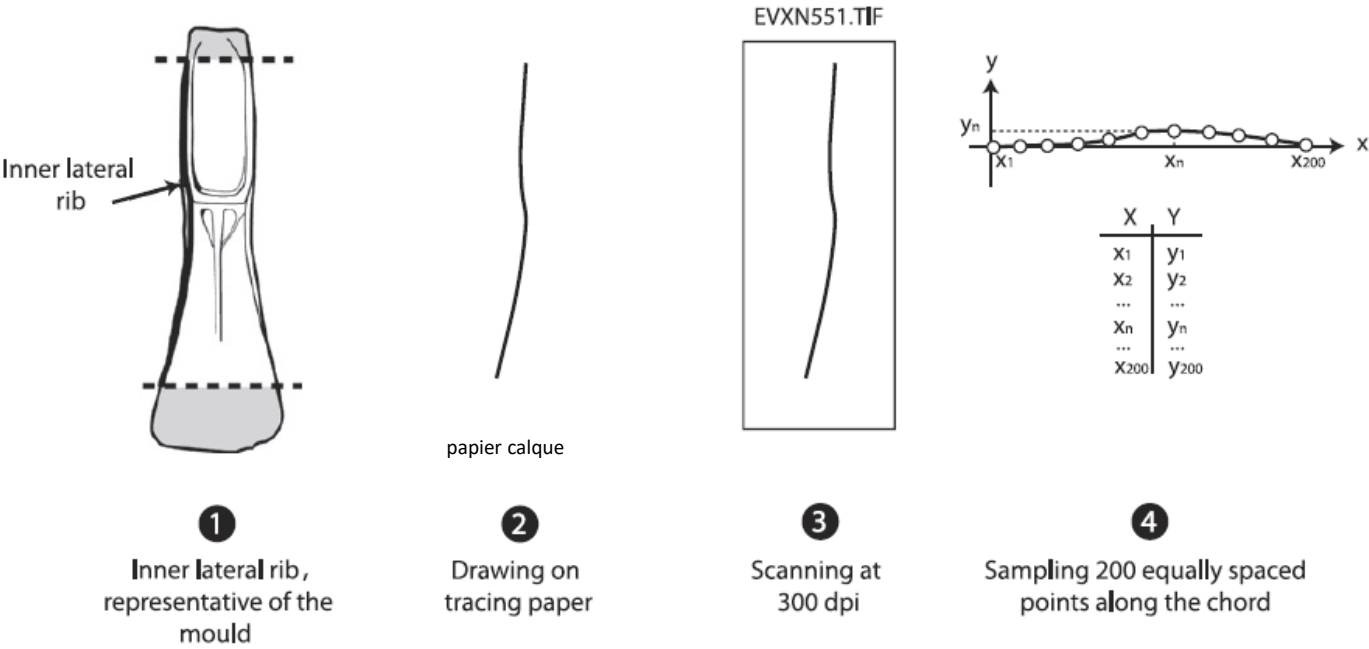
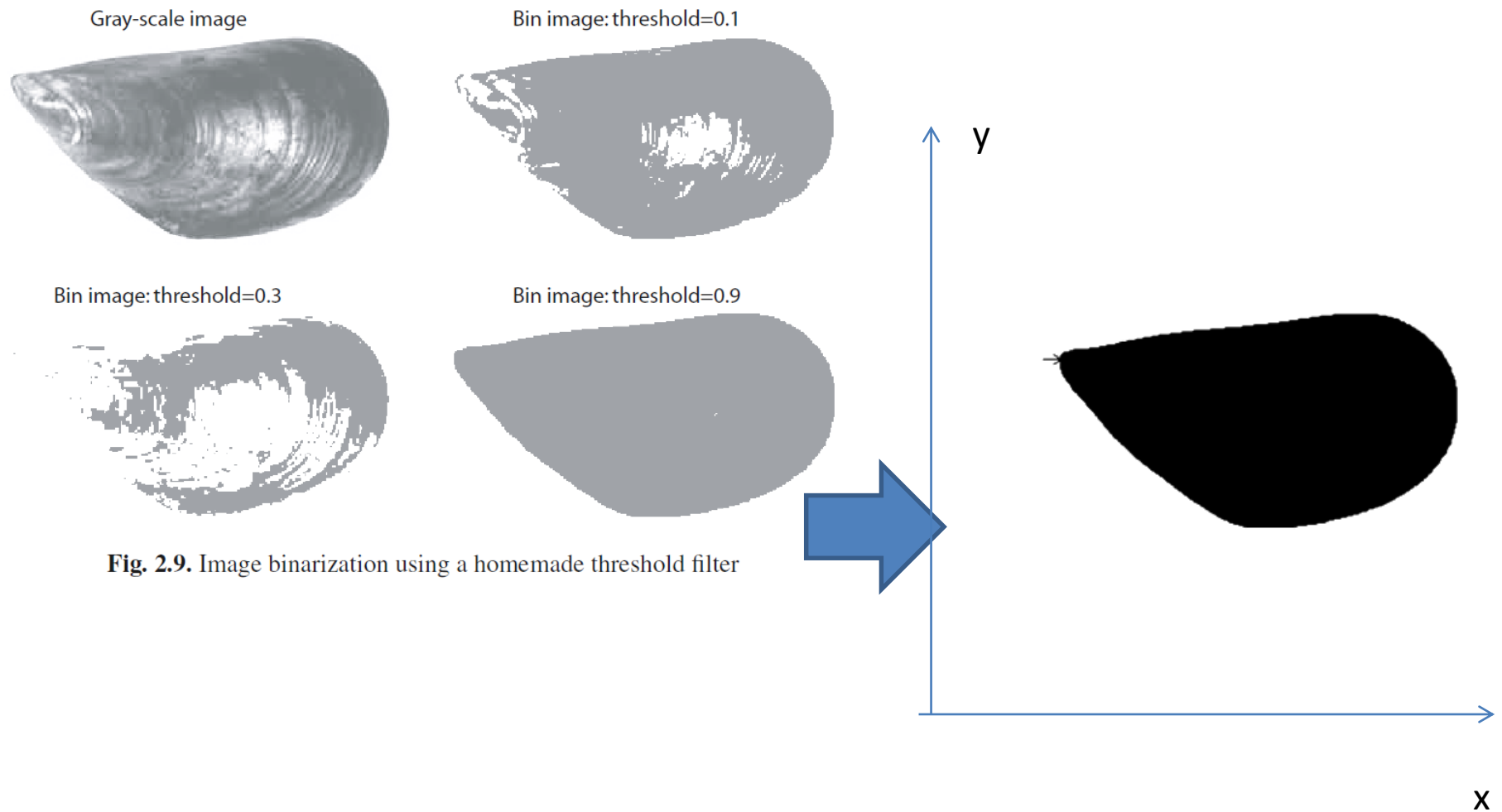


Fig. 2.11. Obtaining equally spaced pseudolandmarks using curve digitizing; pseudolandmarks are white circles, and ending landmarks are black rounds on the right side graph

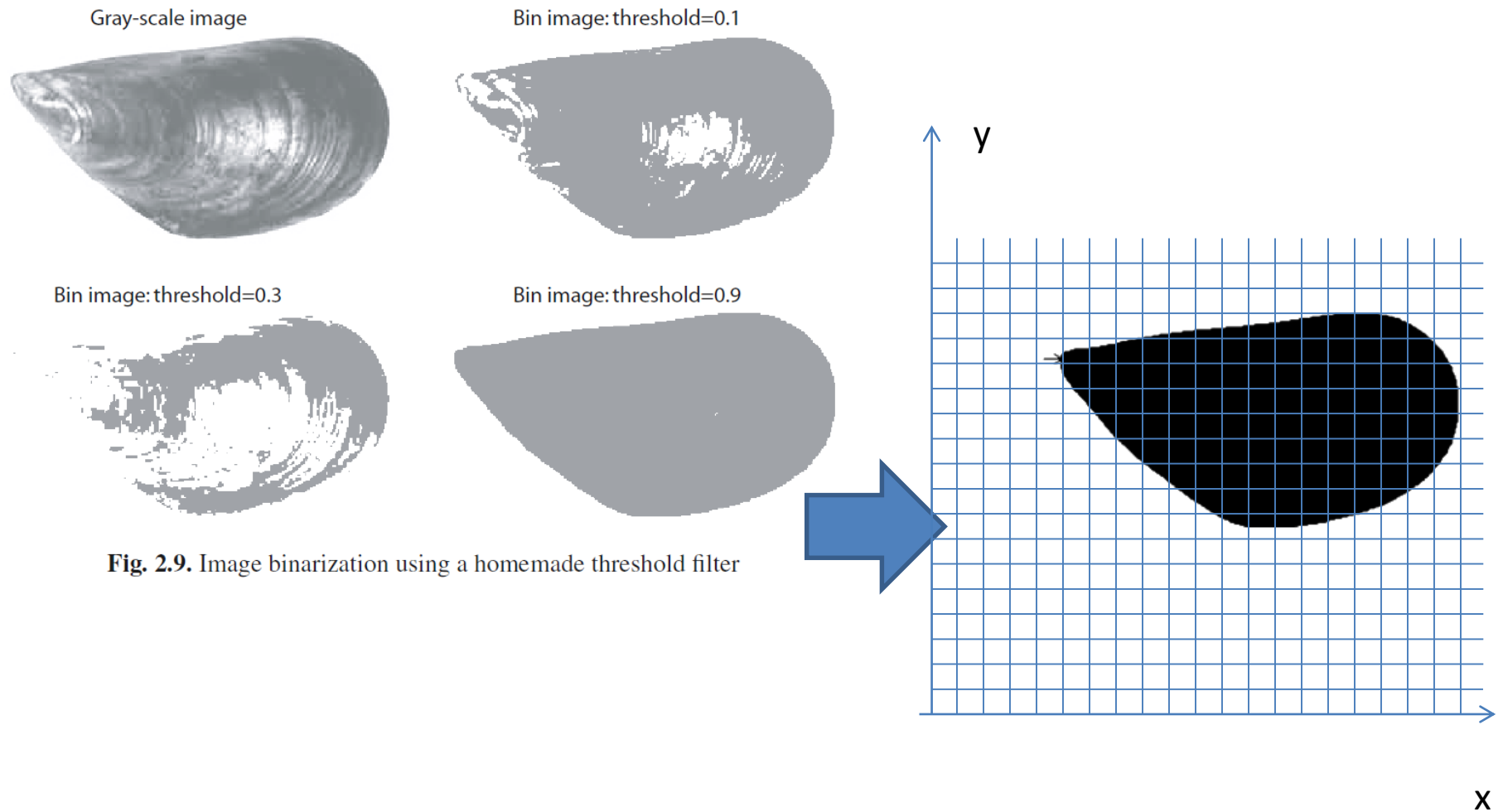
également espacées

Segmentation de ligne

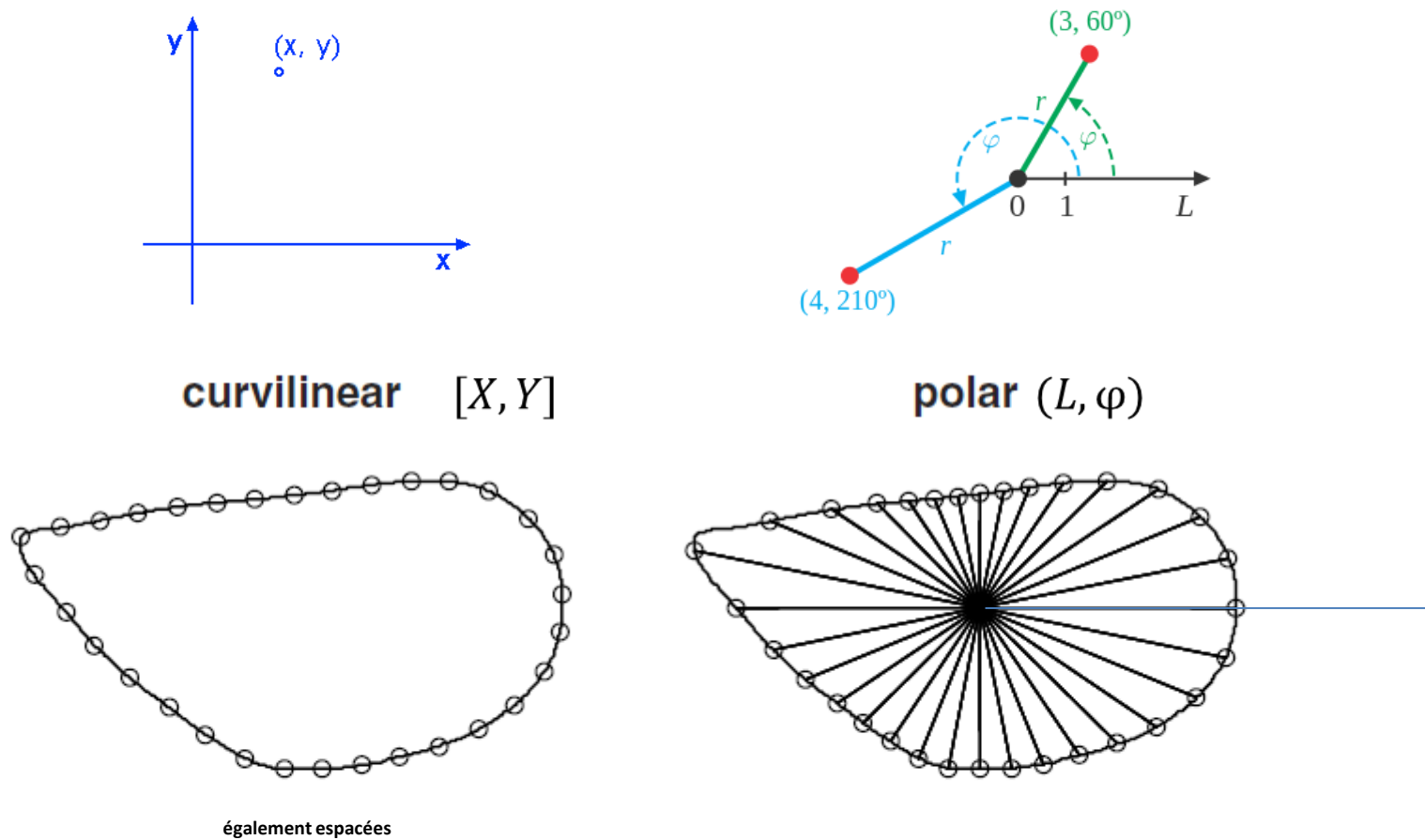
# 1) Collecter des données – Contours fermés



# 1) Collecter des données – Contours fermés



1) Collecter des données – Contours fermés



**Fig. 2.12.** Diverse types of pseudolandmarks automated digitizations with the outline of the *Mytilus* shell. On the left, pseudolandmarks are equally spaced following an equal curvilinear abscissa, while on the right, pseudolandmarks are spaced according to equally spaced angles between segments departing from the centroid to the outline



Chaincode

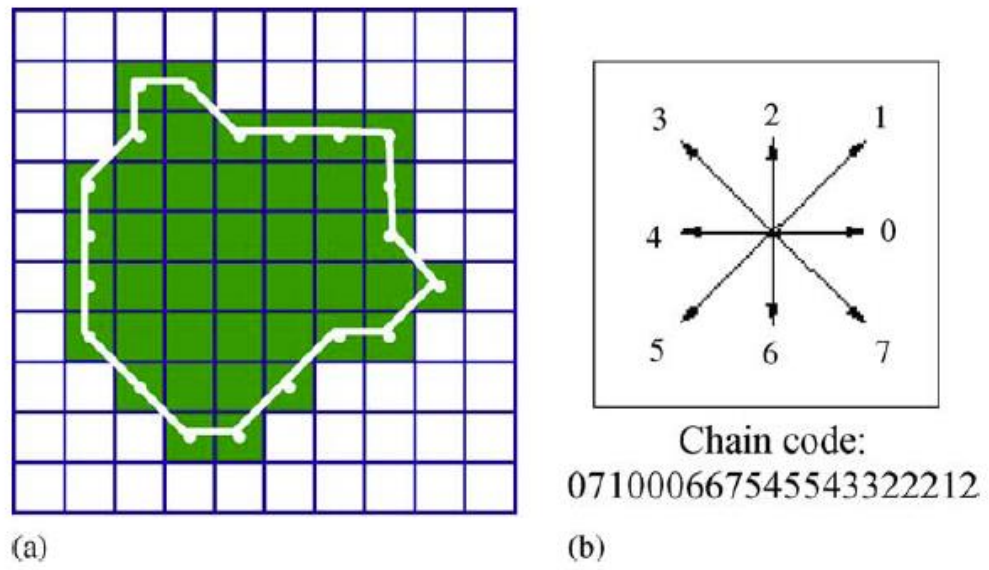
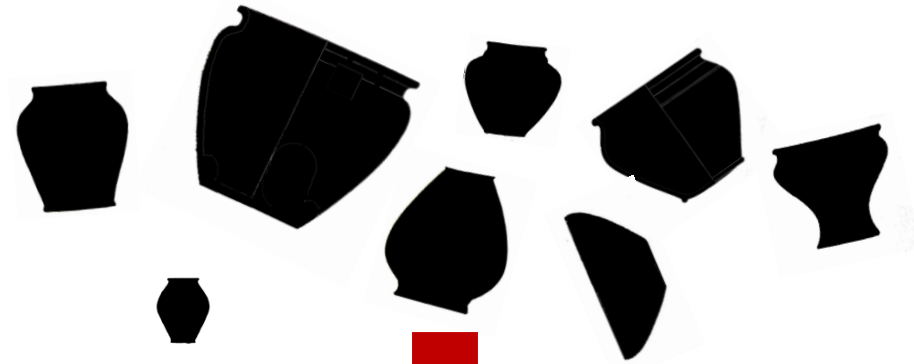


Fig. 2. (a) Binary pixel and shape pattern. (b) Differential chain code sequence starting in upper left hand corner (eight connected grids).



**1) Collecter des données**

**2) Standardisation**

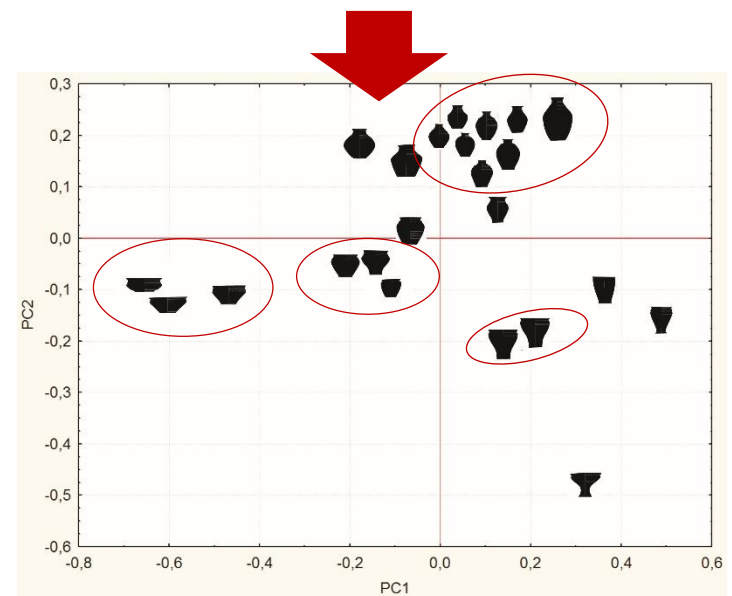
(position, taille et orientation)



**3) Calcul des variables des formes**

(« shape variables »)

**4) Traitement des données et visualisation**

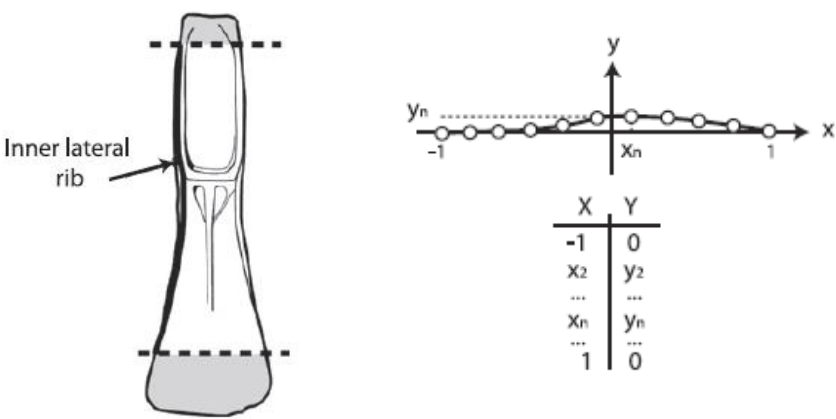


2) Standardisation

Baseline registration (Bookstein): élimination de l'effet taille

Principe:

- 1) Fixation des deux coordonnées (par ex. en [-1,0] et [1,0])



X	Y
-1	0
x <sub>2</sub>	y <sub>2</sub>
...	...
x <sub>n</sub>	y <sub>n</sub>
...	...
1	0

5

Baseline registration:  
First point at (-1,0),  
last point at (1,0)

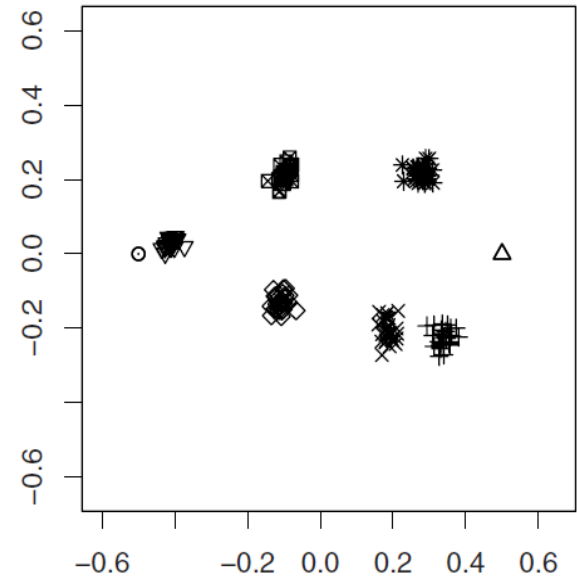
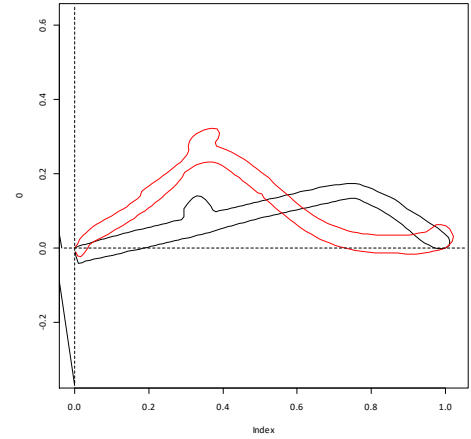
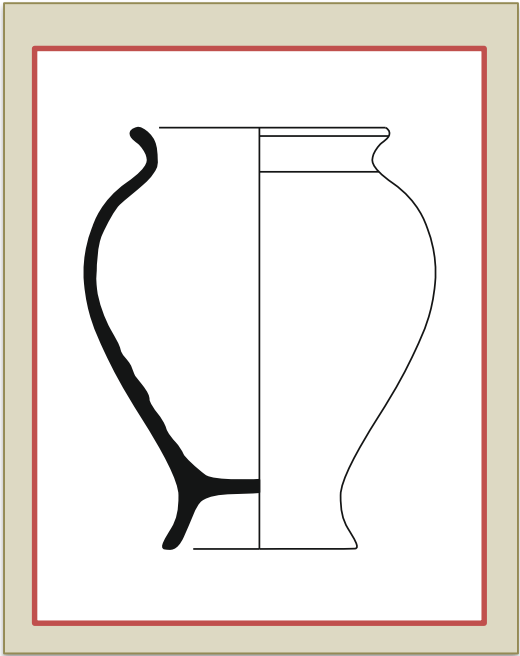
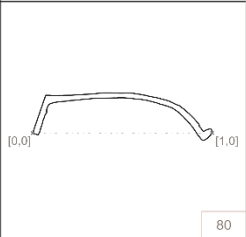
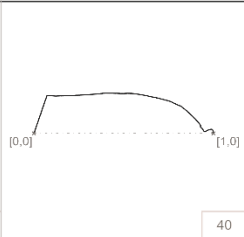
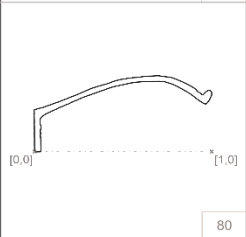
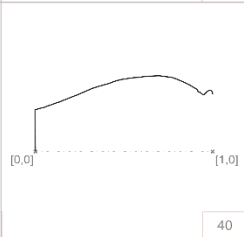
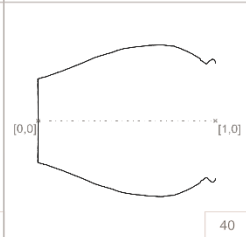
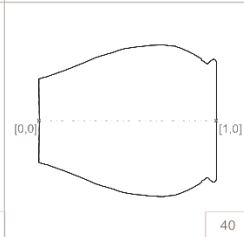
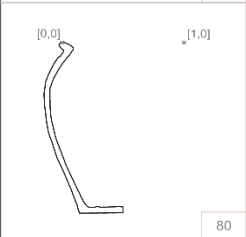
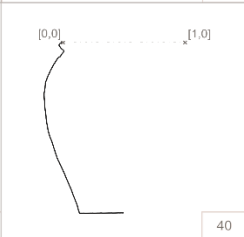
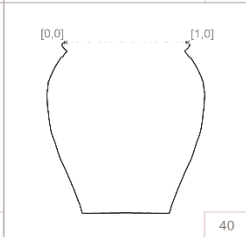
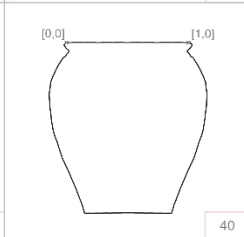
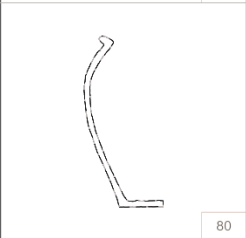
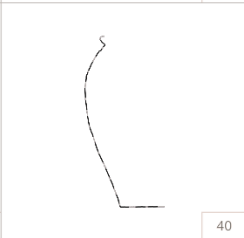
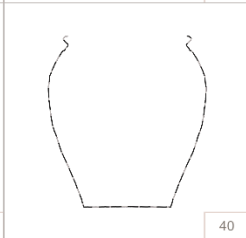
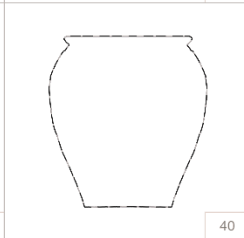

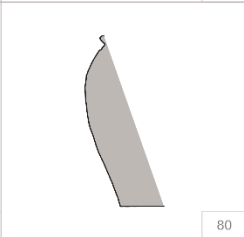
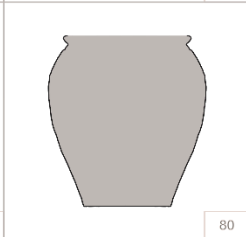
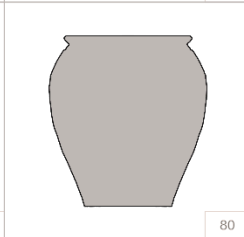


Fig. 4.3. A visualization of the Bookstein registration obtained with functions of the *shapes* package: configurations corresponding to eight landmarks digitized on the midline section for 30 female gorilla skulls are registered onto the first two landmarks

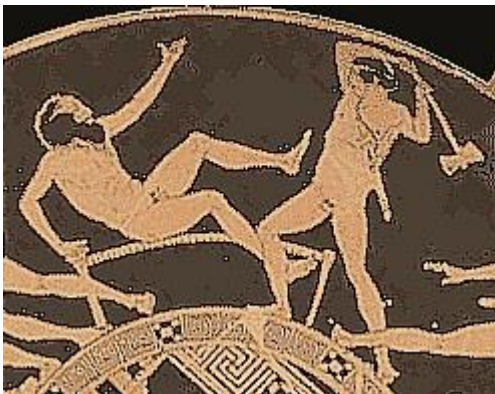
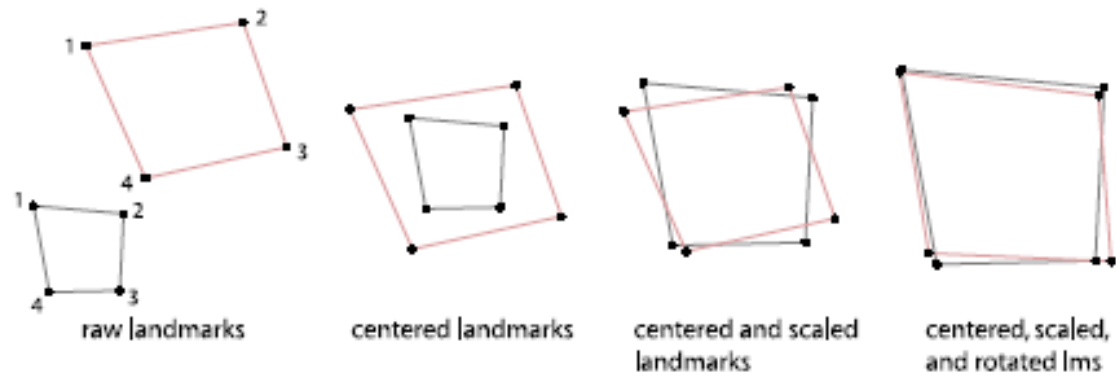
2) Standardisation



	EFA-HS	DCT-HS	DCT-SS	EFA-SS
a				
b				
c				
d				
e				

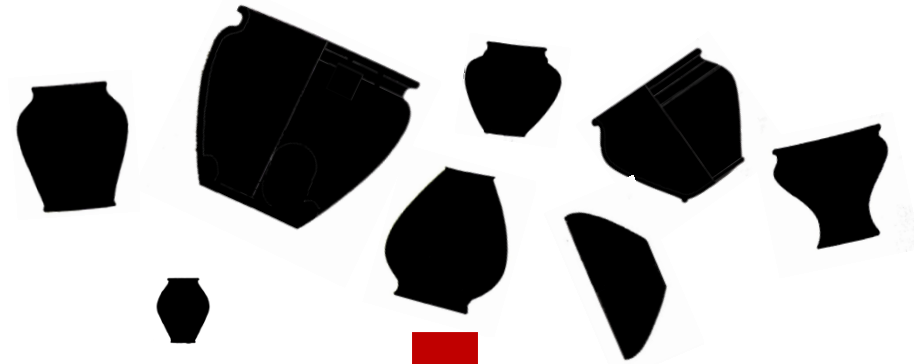
### Procrustes superimposition (on va voir plus tard...)

**Fig. 3** The three steps of Procrustes superimposition: translation to a common origin, scaling to unit centroid size, and rotation to minimize the sum of squared Euclidean distances among the homologous landmarks. The resulting landmark coordinates are called Procrustes shape coordinates



#### Principe - on a deux configurations:

- 1) Déplacement sur l'origine commun
- 2) Standardisation de la taille
- 3) Rotation pour minimiser la somme des distances carrées entre les landmarks



**1) Collecter des données**

**2) Standardisation**

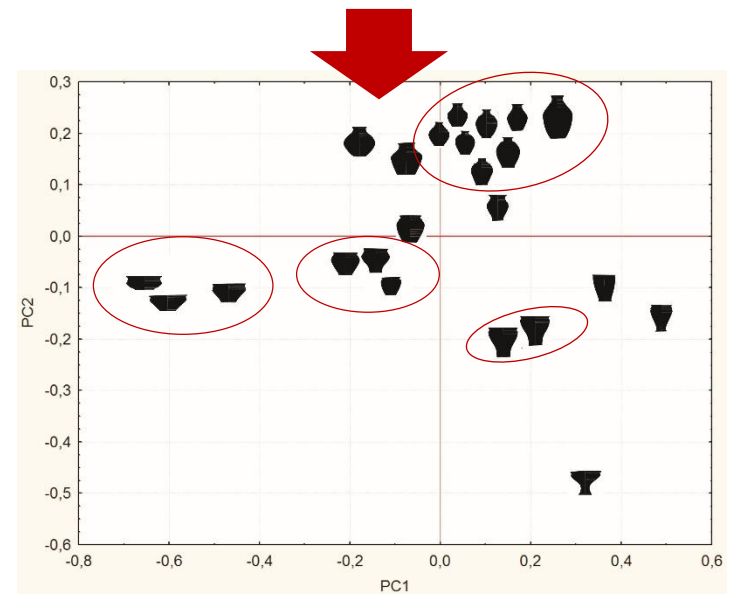
(position, taille et orientation)



**3) Calcul des variables des formes**

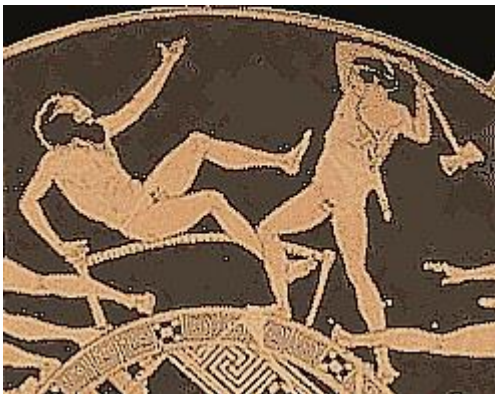
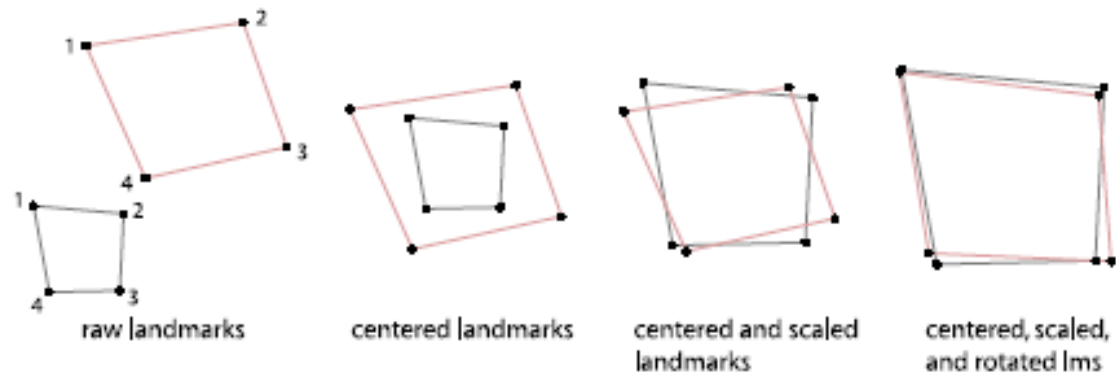
(« shape variables »)

**4) Traitement des données et visualisation**



#### Procrustes superimposition (on va voir plus tard...)

**Fig. 3** The three steps of Procrustes superimposition: translation to a common origin, scaling to unit centroid size, and rotation to minimize the sum of squared Euclidean distances among the homologous landmarks. The resulting landmark coordinates are called Procrustes shape coordinates



#### Principe - on a deux configurations:

- 1) Déplacement sur l'origine commun
- 2) Standardisation de la taille
- 3) Rotation pour minimiser la somme des distances carrées entre les landmarks

### 3) Calcul des variables des formes

#### Principe Thin-Plate Splines (TPS):

- Le changement d'une forme à l'autre peut être vu comme une déformation
- La quantité d'énergie (ou la force) qui est nécessaire à déformer un objet pour obtenir l'autre nous donne l'information sur leur similarités

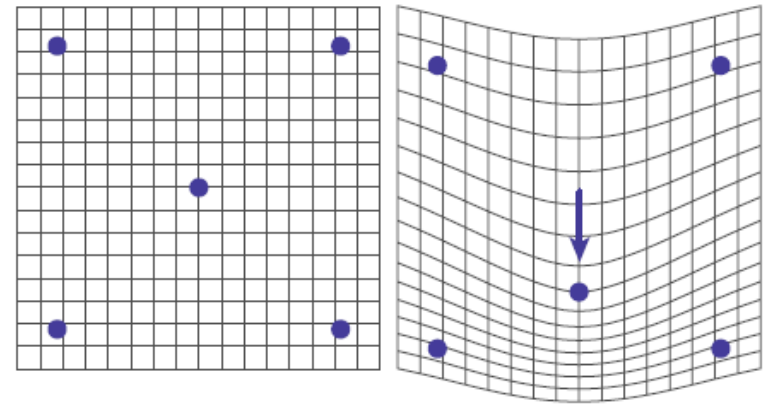


Fig. 4 A template configuration (*left*) and a target configuration (*right*) of five landmarks each. The deformation grid on the right illustrates the thin-plate spline function between these configurations as applied to the left regular grid—it is a visualization of the differences between the two shapes

Object 1



Object 2





### 3) Calcul des variables des formes

#### Principe Thin-Plate Splines (TPS):

- Le changement d'une forme à l'autre peut être vu comme une déformation
- La quantité d'énergie (ou la force) qui est nécessaire à déformer un objet pour obtenir l'autre nous donne l'information sur leur similarités

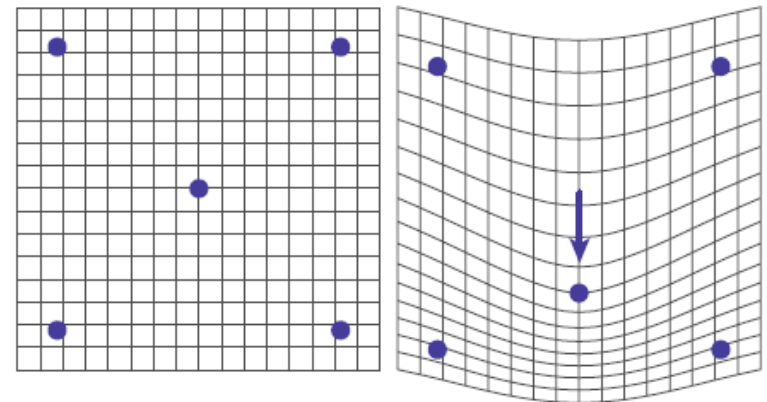


Fig. 4 A template configuration (*left*) and a target configuration (*right*) of five landmarks each. The deformation grid on the right illustrates the thin-plate spline function between these configurations as applied to the left regular grid—it is a visualization of the differences between the two shapes

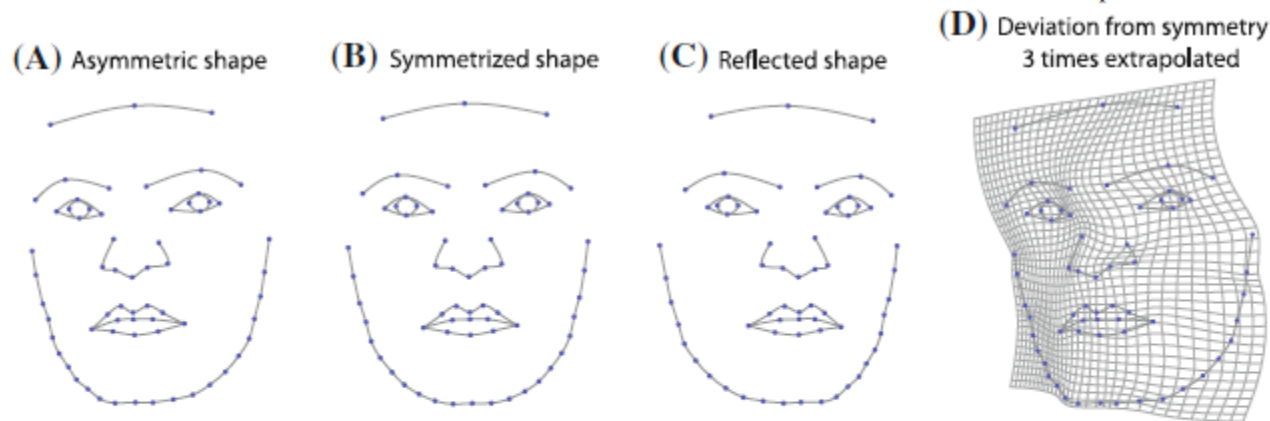
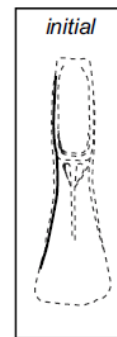
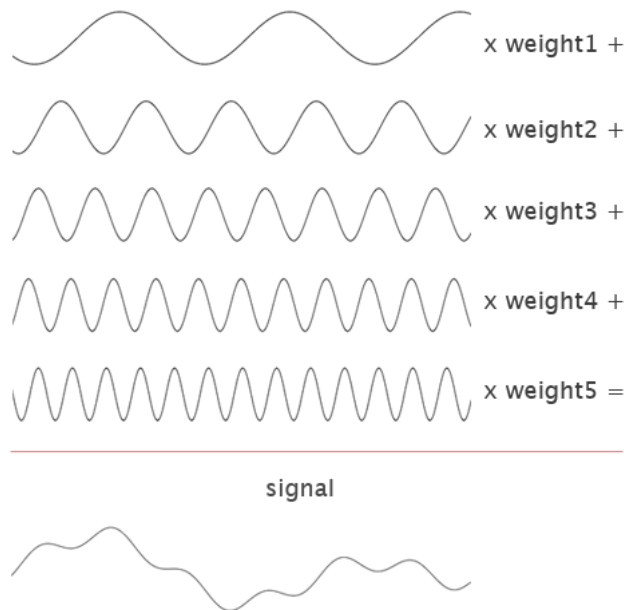


Fig. 7 Analysis of asymmetry in geometric morphometrics: a an asymmetric configuration of 64 landmarks digitized on a photograph of a human face. The configuration in b is the average of the asymmetric shape a and its reflection c after Procrustes fitting, resulting in a perfectly symmetric shape. The Procrustes distance

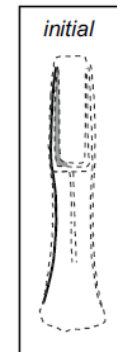
between a and c, which equals two times the distance between a and b, is a measure of total asymmetry. The deformation grid d from the symmetric consensus to the asymmetric shape in a visualizes the pattern of asymmetry. This deformation is extrapolated by a factor of three to ease its interpretation

### 3) Calcul des variables des formes

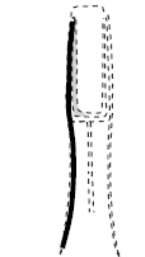
#### DCT: Fourier sur contours ouverts (Discrete Cosine Transform)



Norman axe



Breton axe

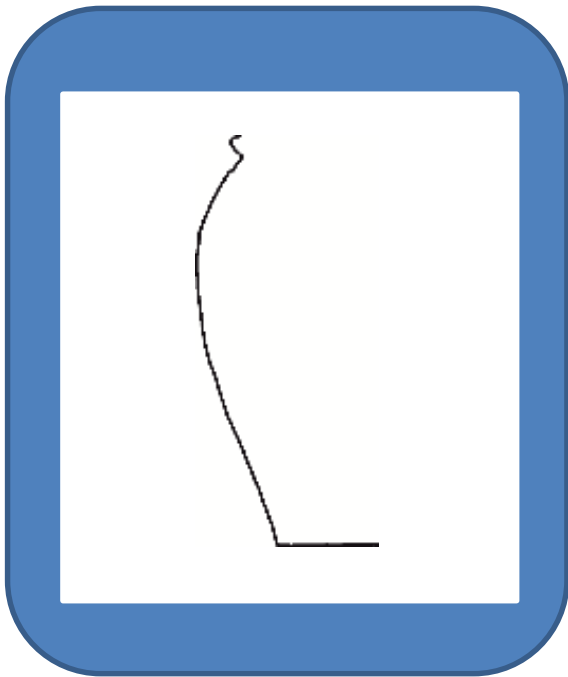


$$S[k] = \sum_{n=0}^{N-1} c[k]s[n] \cos\left(\frac{(2n+1)k\pi}{2N}\right)$$

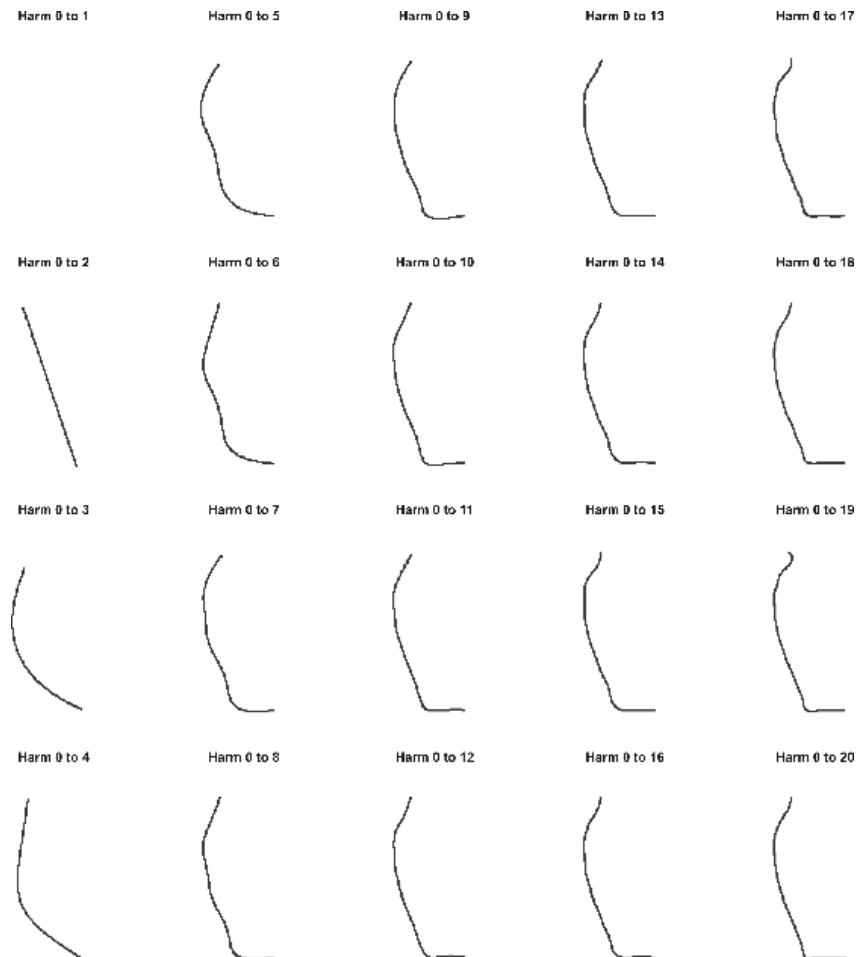
#### Principe:

- 1) Chaque contour peut être vu comme un signal et donc **peut être décomposé en infinité somme des ondes (harmonics)**
- 2) Chaque onde est pesé par 2 coefficients
- 3) Si on somme les ondes (coefficients), on peut reconstruire la contour originale

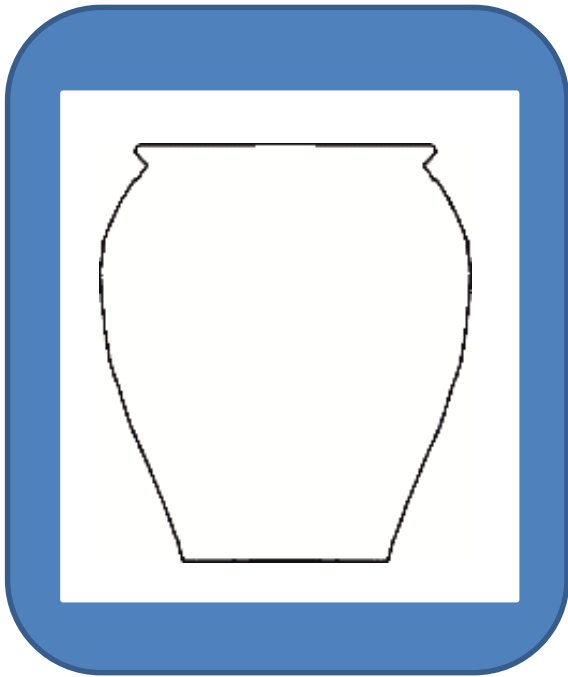
## DCT: Fourier sur contours ouverts (Discrete Cosine Transform)



La contour originale



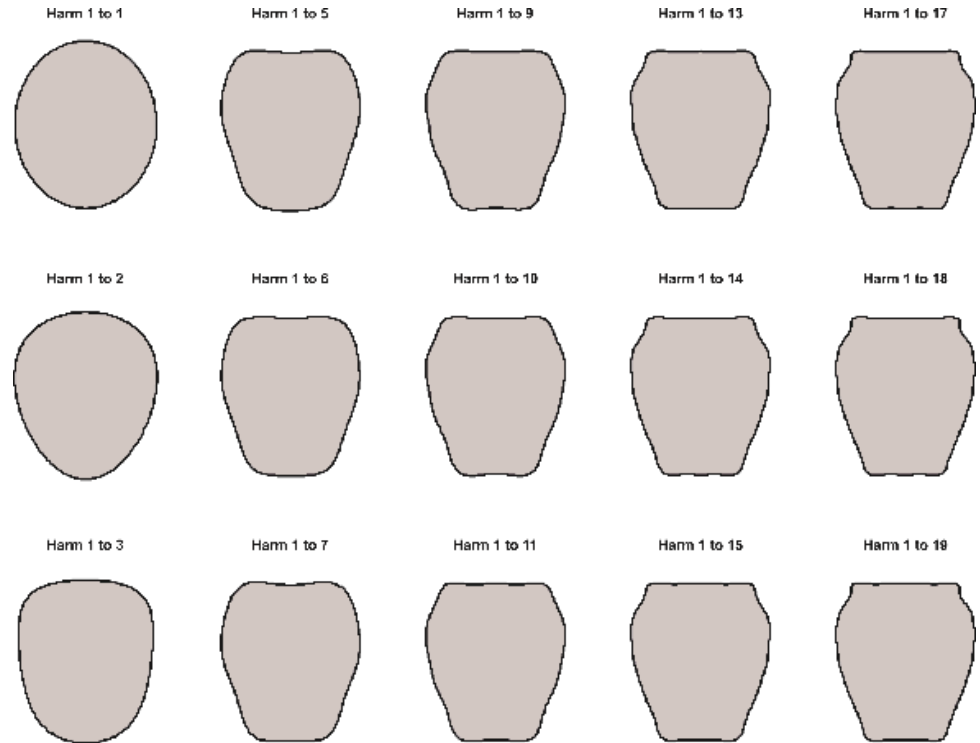
## EFA: Analyse de Fourier Elliptique



La contour originale

$$x(t) = a_0 + \sum_{j=1}^k a_j \cos\left(\frac{2j\pi t}{T}\right) + \sum_{j=1}^k b_j \sin\left(\frac{2j\pi t}{T}\right)$$

$$y(t) = c_0 + \sum_{j=1}^k a_j \cos\left(\frac{2j\pi t}{T}\right) + \sum_{j=1}^k d_j \sin\left(\frac{2j\pi t}{T}\right)$$



### Méthode (analogique a DCT):

- 1) Chaque contour peut être vu comme un signal et donc peut être **décomposé en infinité somme des ellipses** (harmonics)
- 2) Chaque ellipse est pesé par 4 coefficients
- 3) Si on somme les ellipses (coefficients), on peut reconstruire la contour originale

### 3) Calcul des variables des formes

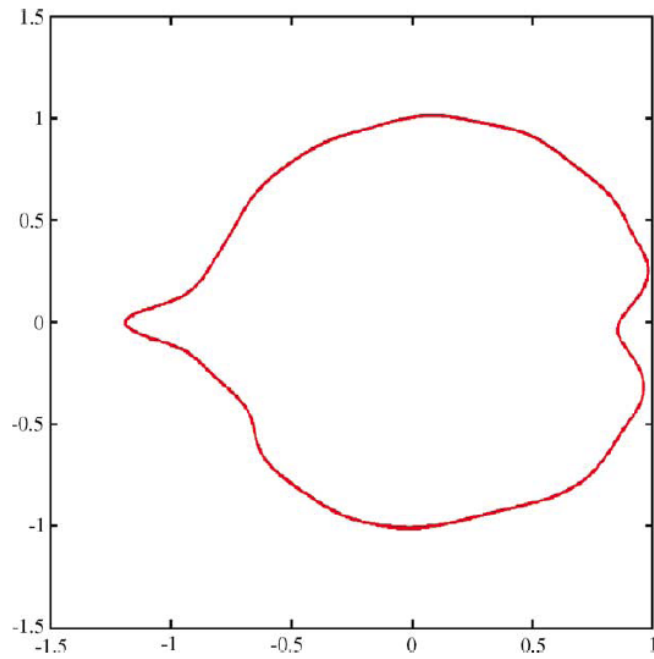


Fig. 4. Final normalized and rotated Elliptic Fourier leaflet boundary for velvetleaf.

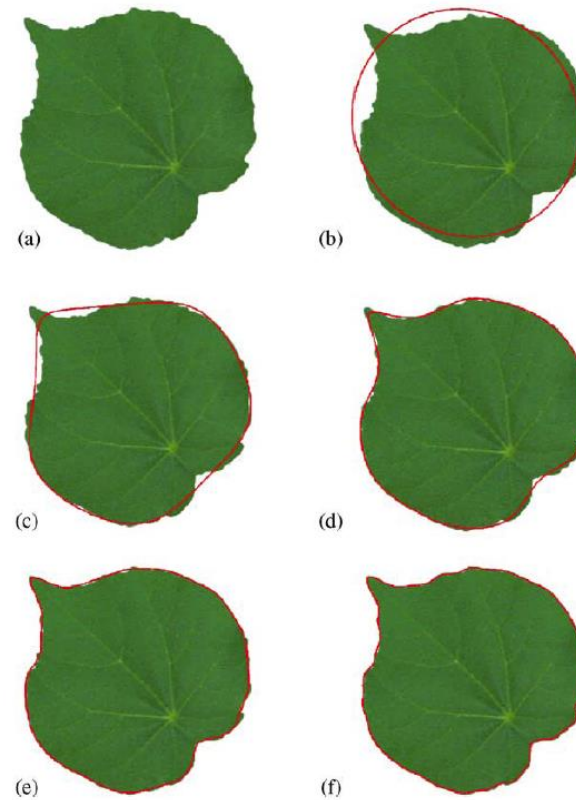
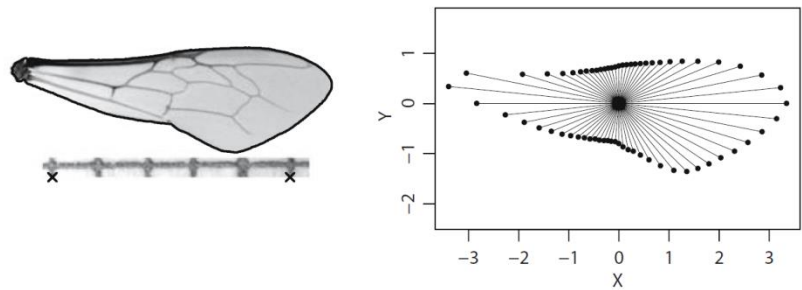


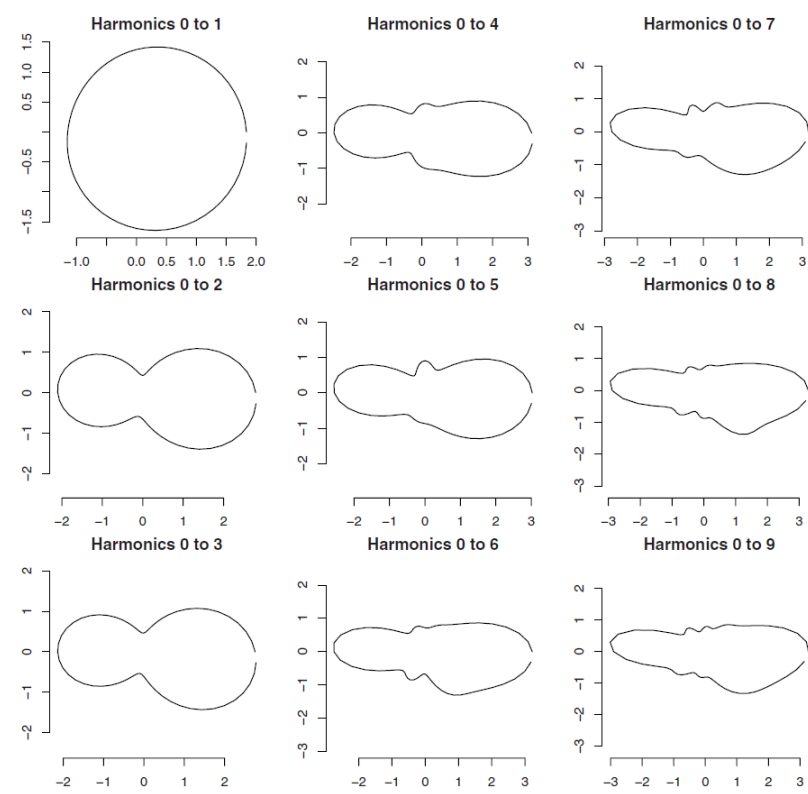
Fig. 3. Elliptic Fourier shape approximations for velvetleaf (*Abutilon theophrasti Medicus*) (a) original leaf image, (b) 1st EF harmonic, (c) 1st + 2nd + 3rd + 4th EF, (d) 1st + ... + 8th EF harmonics, (e) 1st + ... + 16th EF harmonics and (f) 1st + ... + 30th EF harmonics.

3) Calcul des variables des formes

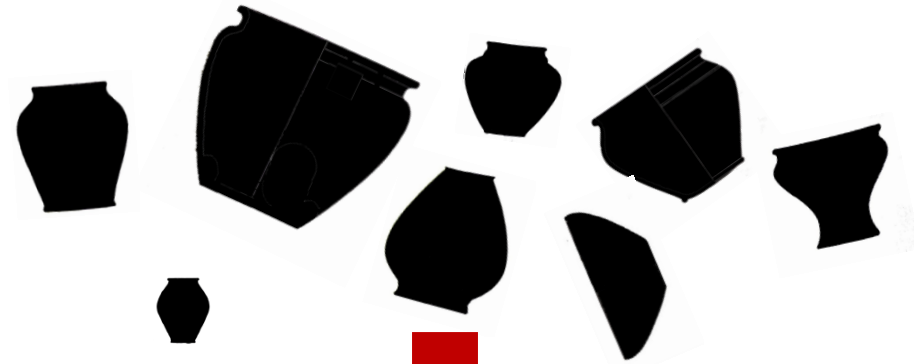


**Fig. 5.4.** Data acquisition for Fourier analysis of equally spaced radii. The two crosses on the left graph correspond to the landmarks digitized to obtain a scale on the millimeter paper to standardize coordinates by size

Analyse de Fourier appliquée aux variations de rayon



**Fig. 5.5.** Reconstruction of outlines by Fourier analysis applied to equally spaced radii



**1) Collecter des données**

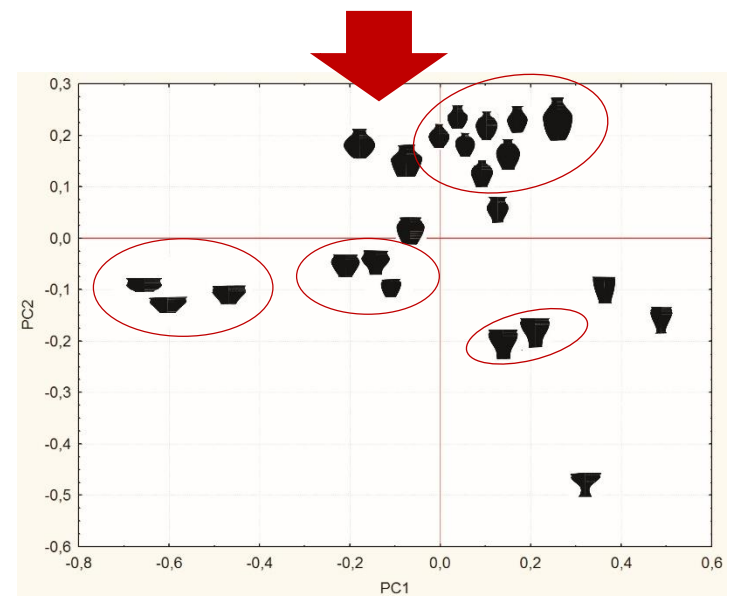
**2) Standardisation**

(position, taille et orientation)

**3) Calcul des variables des formes**

(« shape variables »)

**4) Traitement des données et visualisation**



Données quantitatives: ici longueurs - largeurs



FIG. 1 – Sir R.A. Fisher (1890-1962)



FIG. 2 – *I. setosa*, *I. versicolor*, *I. virginica*

Console ~/ ↻					
> data(iris)					
> iris					
	Sepal.Length	Sepal.width	Petal.Length	Petal.width	Species
1	5.1	3.5	1.4	0.2	setosa
2	4.9	3.0	1.4	0.2	setosa
3	4.7	3.2	1.3	0.2	setosa
4	4.6	3.1	1.5	0.2	setosa
5	5.0	3.6	1.4	0.2	setosa
6	5.4	3.9	1.7	0.4	setosa
7	4.6	3.4	1.4	0.3	setosa
8	5.0	3.4	1.5	0.2	setosa
9	4.4	2.9	1.4	0.2	setosa
10	4.9	3.1	1.5	0.1	setosa
11	5.4	3.7	1.5	0.2	setosa
12	4.8	3.4	1.6	0.2	setosa
13	4.8	3.0	1.4	0.1	setosa
14	4.3	3.0	1.1	0.1	setosa
15	5.8	4.0	1.2	0.2	setosa
16	5.7	4.4	1.5	0.4	setosa
17	5.4	3.9	1.3	0.4	setosa
18	5.1	3.5	1.4	0.3	setosa
19	5.7	3.8	1.7	0.3	setosa
20	5.1	3.8	1.5	0.3	setosa
21	5.4	3.4	1.7	0.2	setosa
22	5.1	3.7	1.5	0.4	setosa

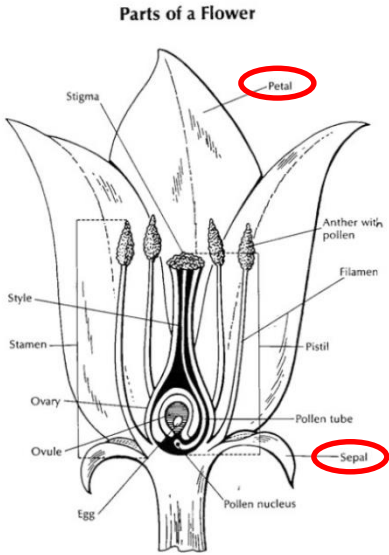
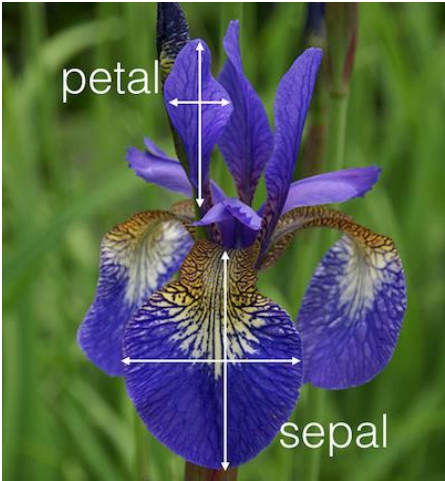


FIG. 3 – Description d'une fleur

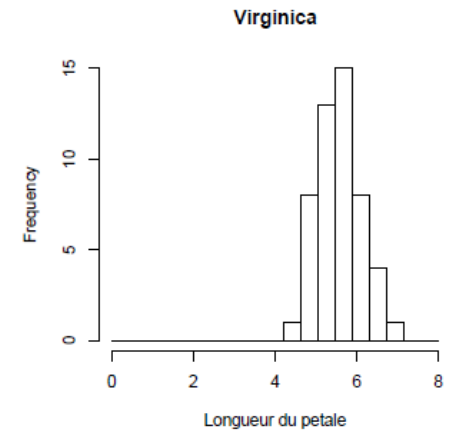
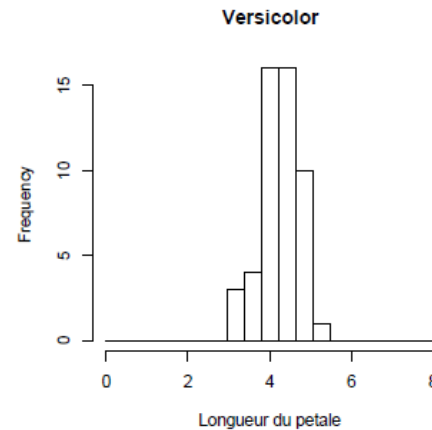
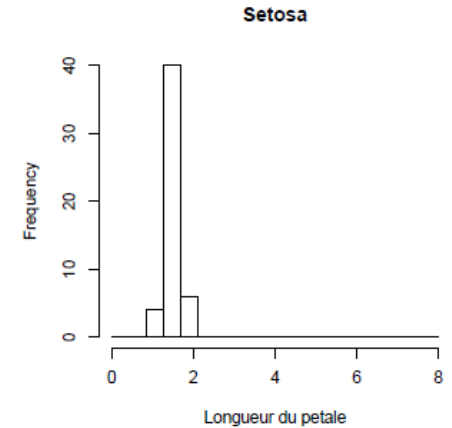
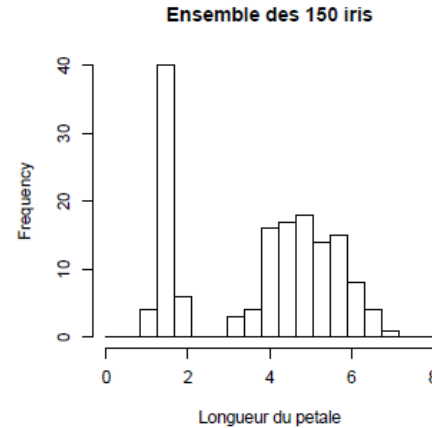
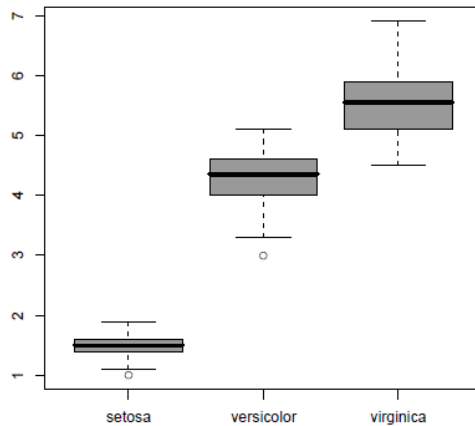




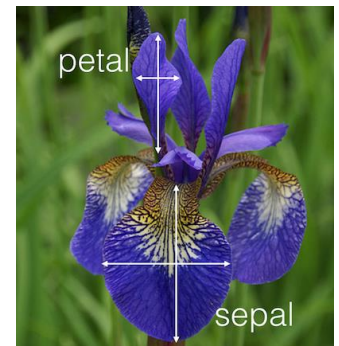
## Approche très basique

### Statistiques simples à une variable

```
boxplot(iris$Petal.Length ~ iris$Species, col = grey(0.6))
```

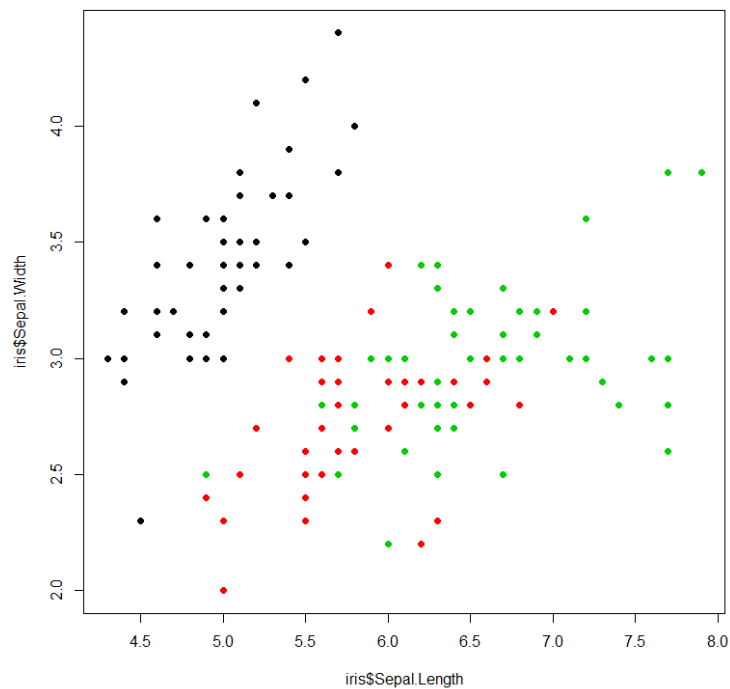


```
par(mfrow = c(2, 2))
brk = seq(from = 0, to = 8, length = 20)
hist(iris$Petal.Length, main = "Ensemble des 150 iris", xlab = "Longueur du petale",
     breaks = brk)
hist(iris$Petal.Length[iris$Species == "setosa"], main = "Setosa",
     xlab = "Longueur du petale", breaks = brk)
hist(iris$Petal.Length[iris$Species == "versicolor"], main = "Versicolor",
     xlab = "Longueur du petale", breaks = brk)
hist(iris$Petal.Length[iris$Species == "virginica"], main = "Virginica",
     xlab = "Longueur du petale", breaks = brk)
```

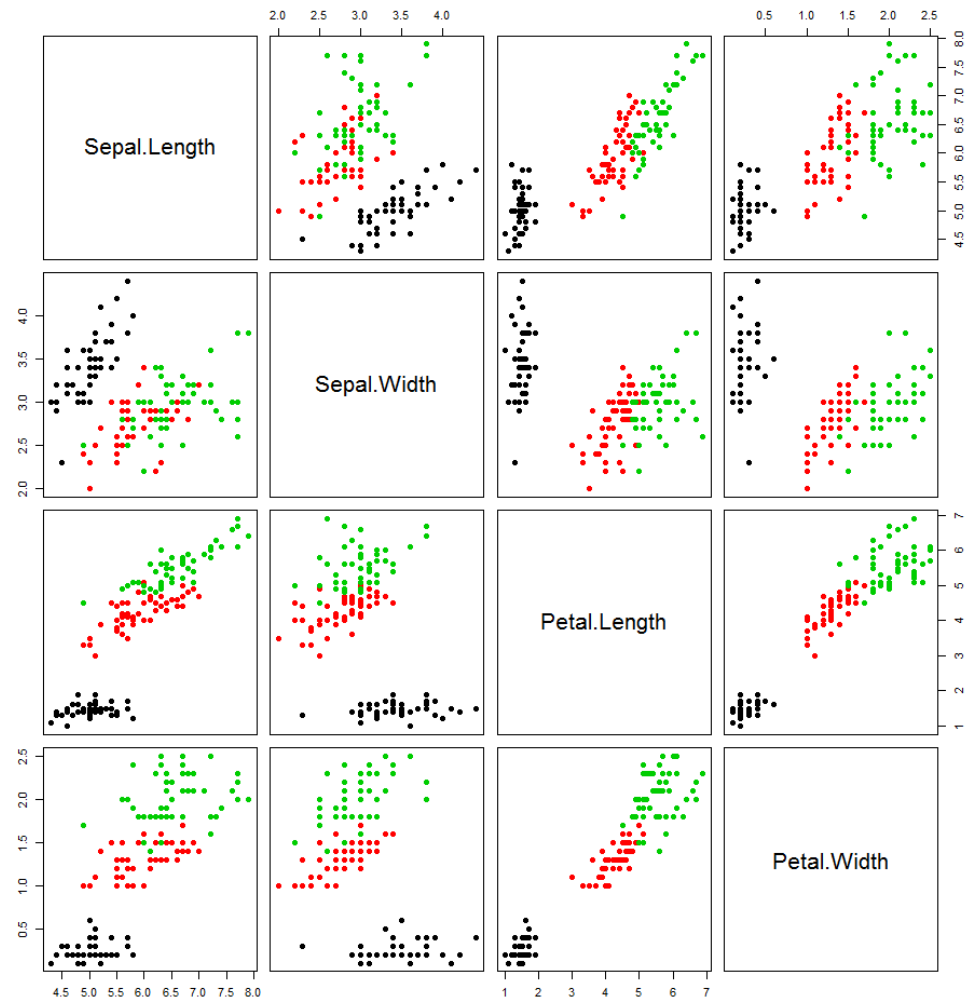


## Statistiques a plusieurs variables

```
plot(iris$Sepal.Length, iris$Sepal.Width, col=iris$Species, pch=19)
```



```
pairs(iris[,1:4], col=iris$Species, pch=19)
```

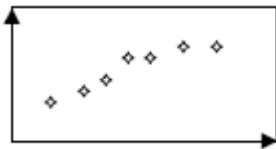


## Statistiques a plusieurs variable:

```
data(crimtab)
pairs(crimtab[,6:16])
```



Lorsqu'il n'y a que deux dimensions (largeur et longueur par exemple), il est facile de représenter les données sur un plan :



Avec trois dimensions (largeur, hauteur et profondeur par ex.), c'est déjà plus difficile :



Mais au delà de 3 dimensions, il est impossible de représenter les données sur un plan ou même de les visualiser mentalement.

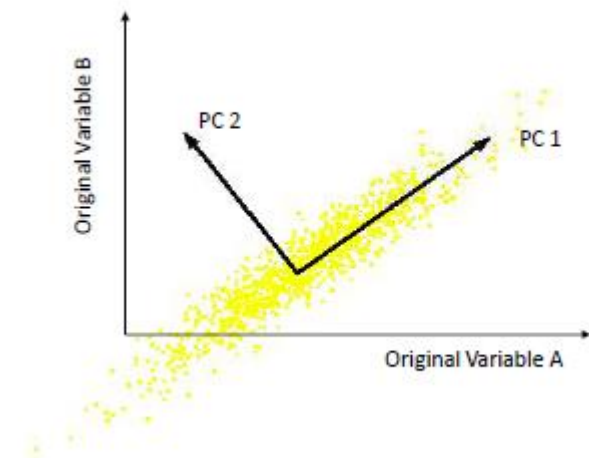
```
library(rgl); library(scatterplot3d)
irisCol=rep(1:3, each=50)
plot3d(x=iris$Petal.Length, y=iris$Petal.Width, z=iris$Sepal.Length, col=irisCol, size=10)
```

**But de l'ACP:** Réduire la dimension du problème (si possible à 2) tout en conservant le maximum d'information

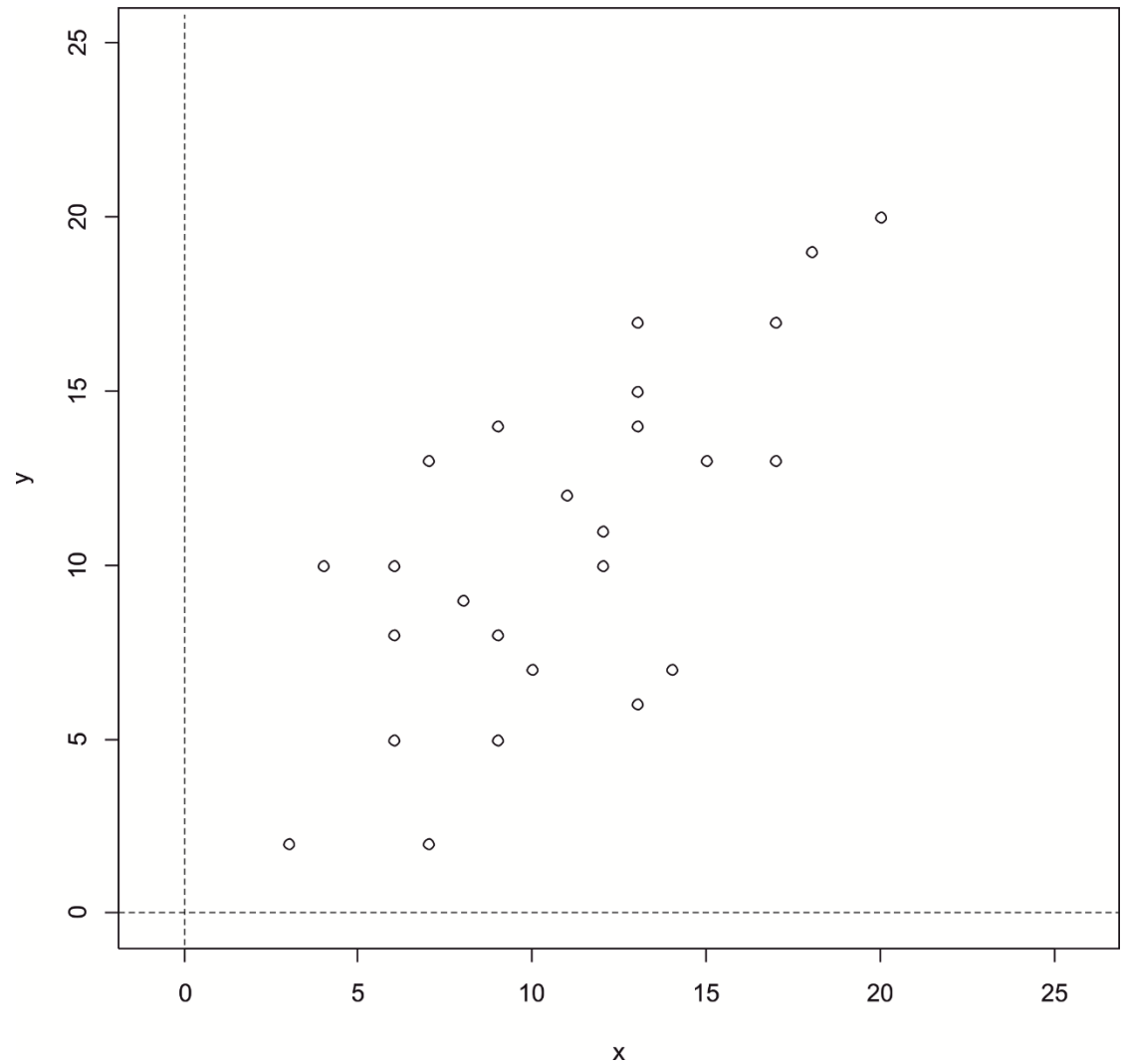


C'est possible? Oui! Comment?

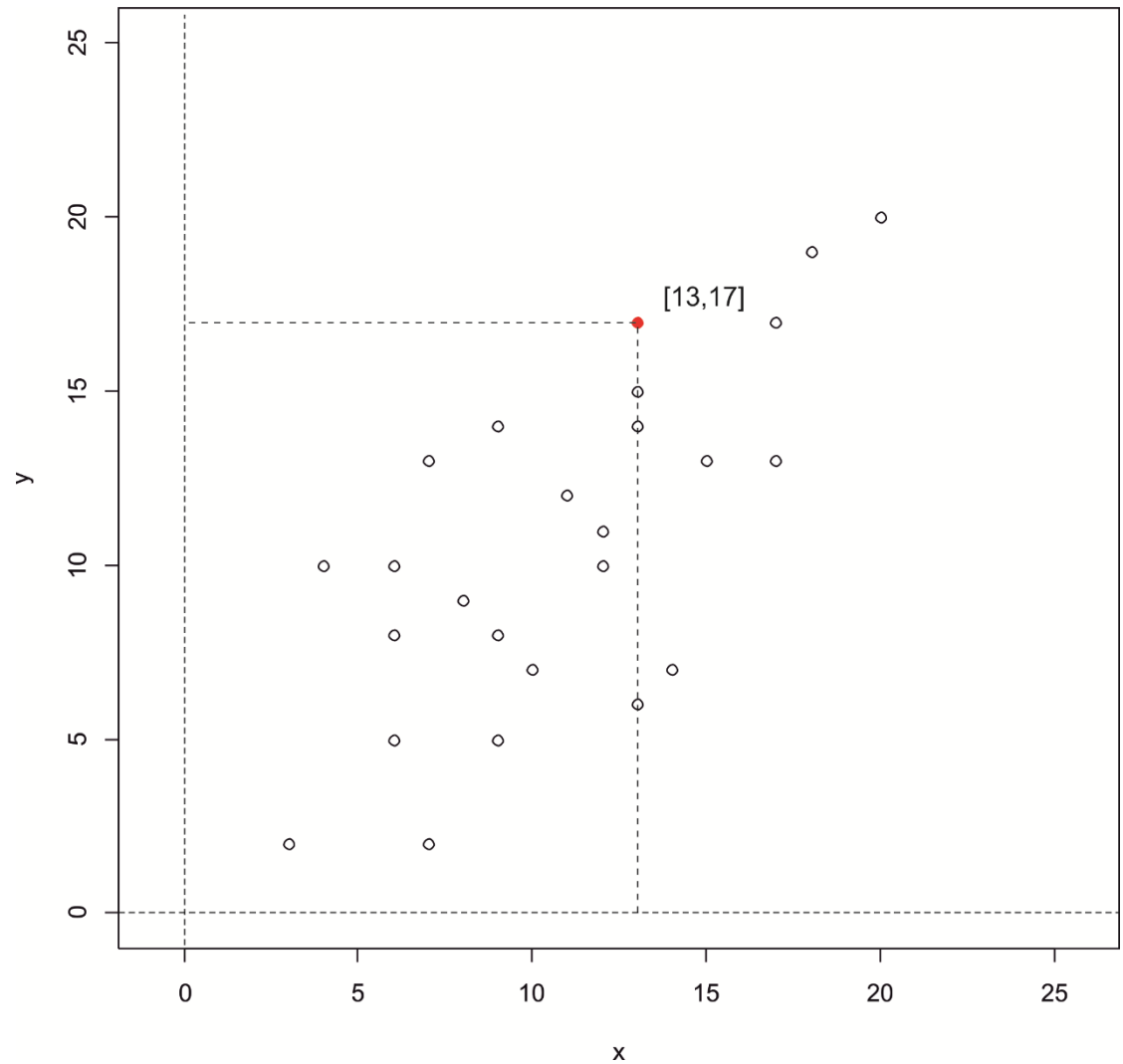
1. Les nouvelles variables sont des combinaisons linéaires des variables originales.
2. Elles sont orthogonales entre elles.
3. Elles capturent autant que possible la variance d'origine.
4. Ce sont les **composantes principales**



En plus, c'est assez simple...

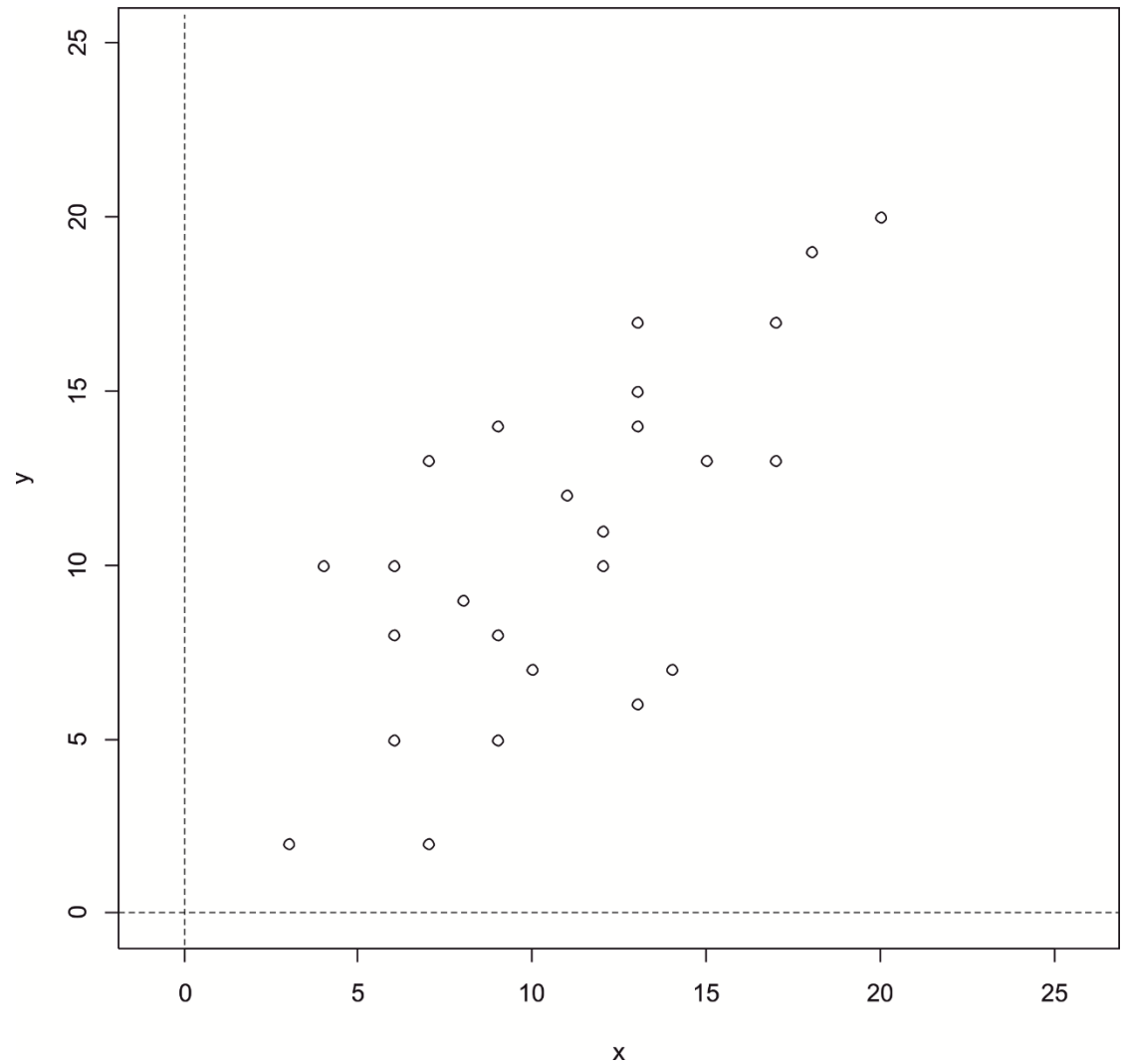


En plus, c'est assez simple...

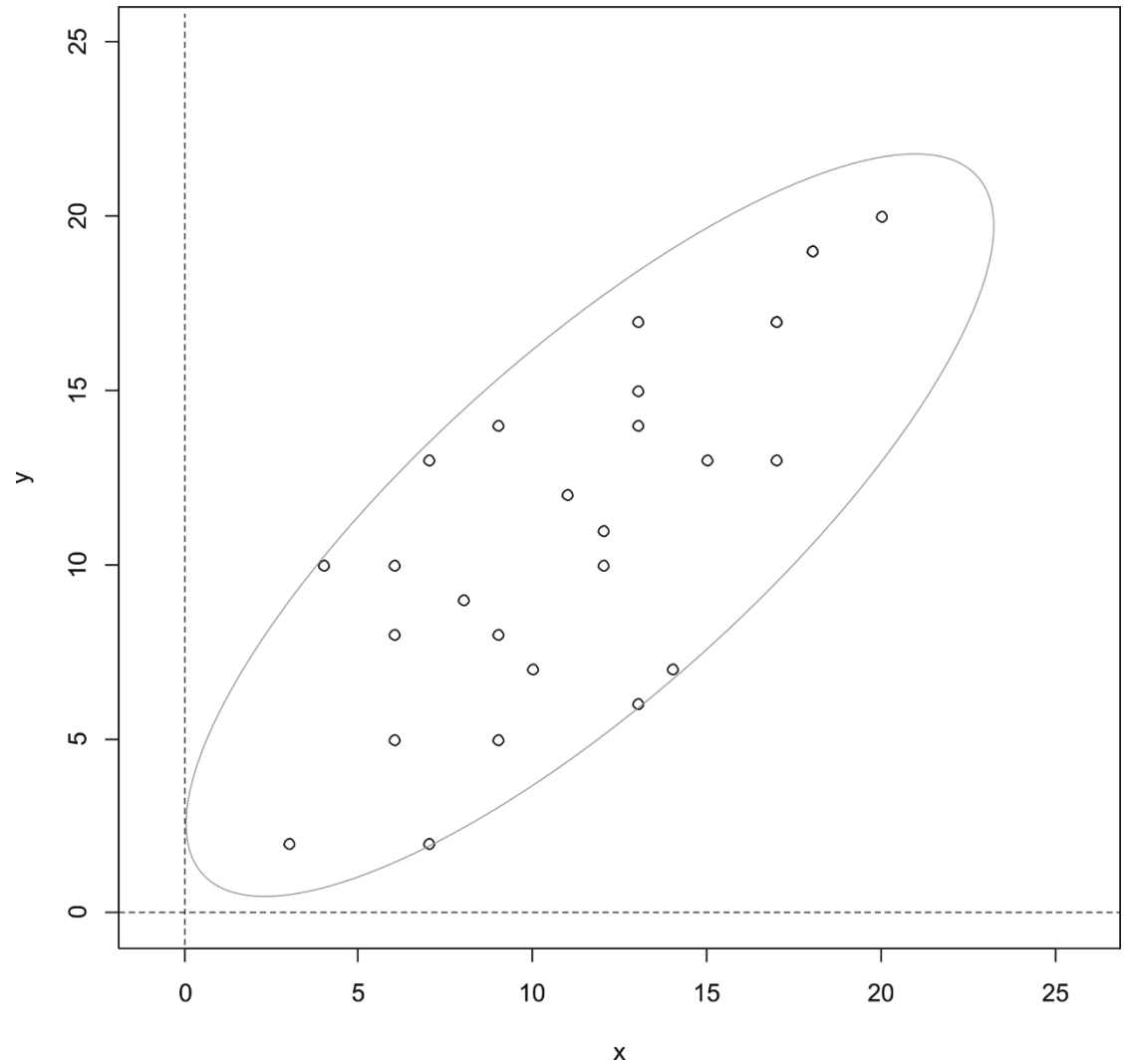




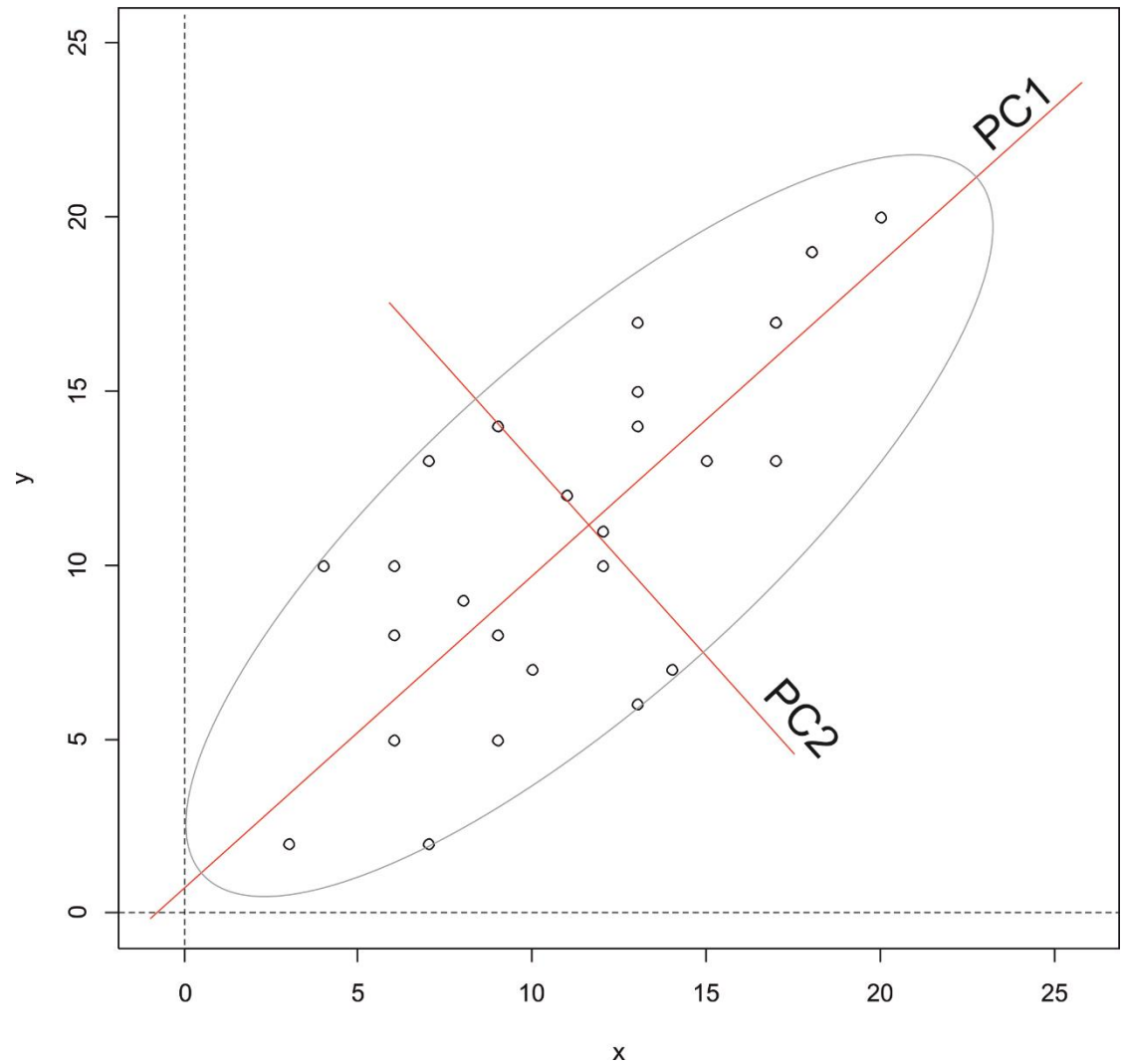
En plus, c'est assez simple...



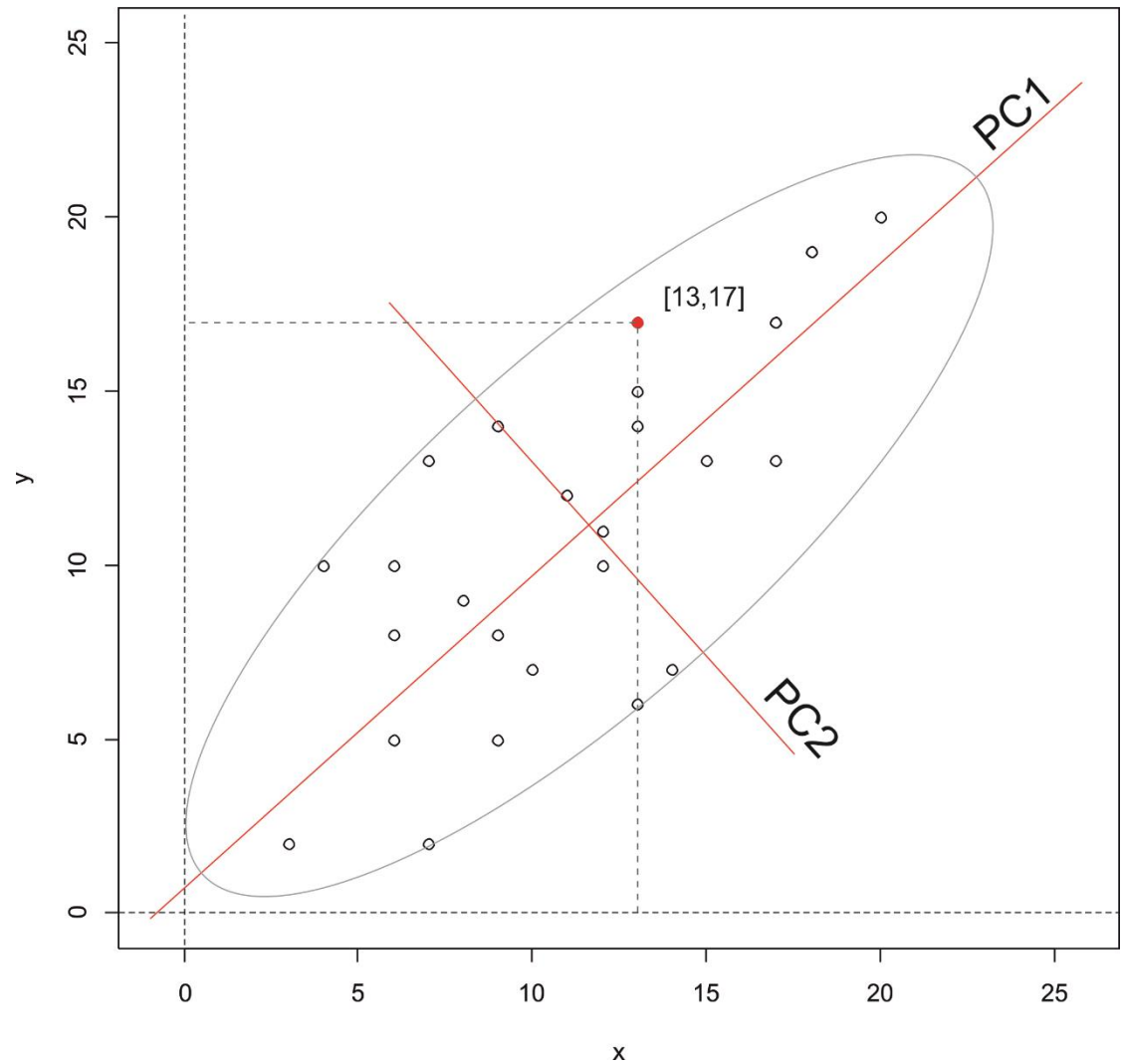
En plus, c'est assez simple...



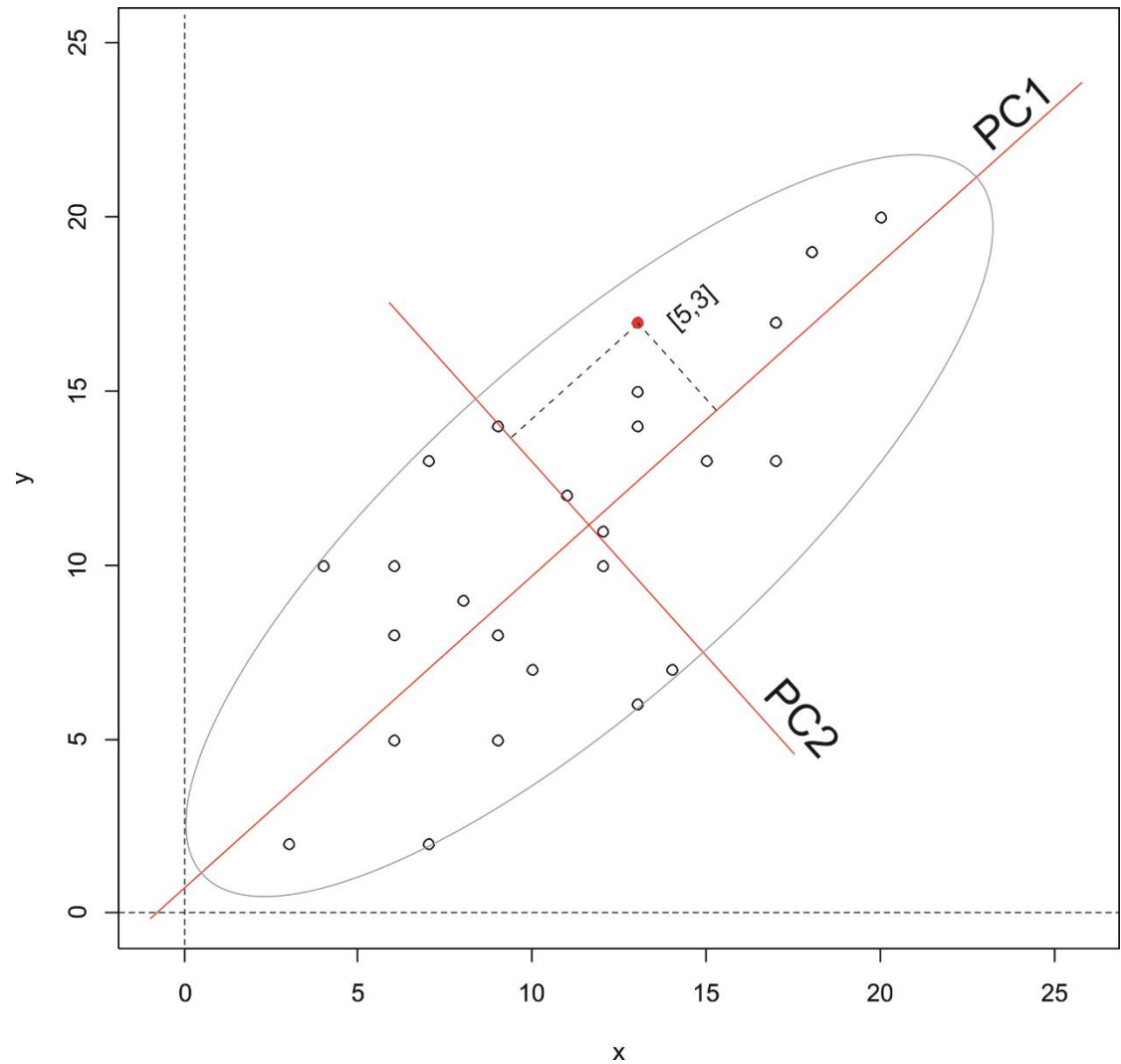
En plus, c'est assez simple...



En plus, c'est assez simple...



En plus, c'est assez simple...



## **Comment calculer l'ACP?**

- 1) Data
- 2) Calculer la matrice des variance-covariance / corrélation
- 3) Obtenir eigenvectors et eigenvalues
  - 1) Eigenvectors – vecteurs des axes principales
  - 2) Eigenvalues – valeurs associés à chaque des eigenvectors; donne l'information sur la variance
- 4) Projeter les données originales aux axes principales
  - Produit vectoriel des Data et Eigenvectors

$x = 1, 1, 2, 4, 5, 7, 8, 8, 8, 9$   
 $y = 1, 2, 3, 4, 5, 6, 7, 8, 9, 10$   
 $\bar{x} = 5.3, \bar{y} = 5.5$   
 $n = 10$

Covariance

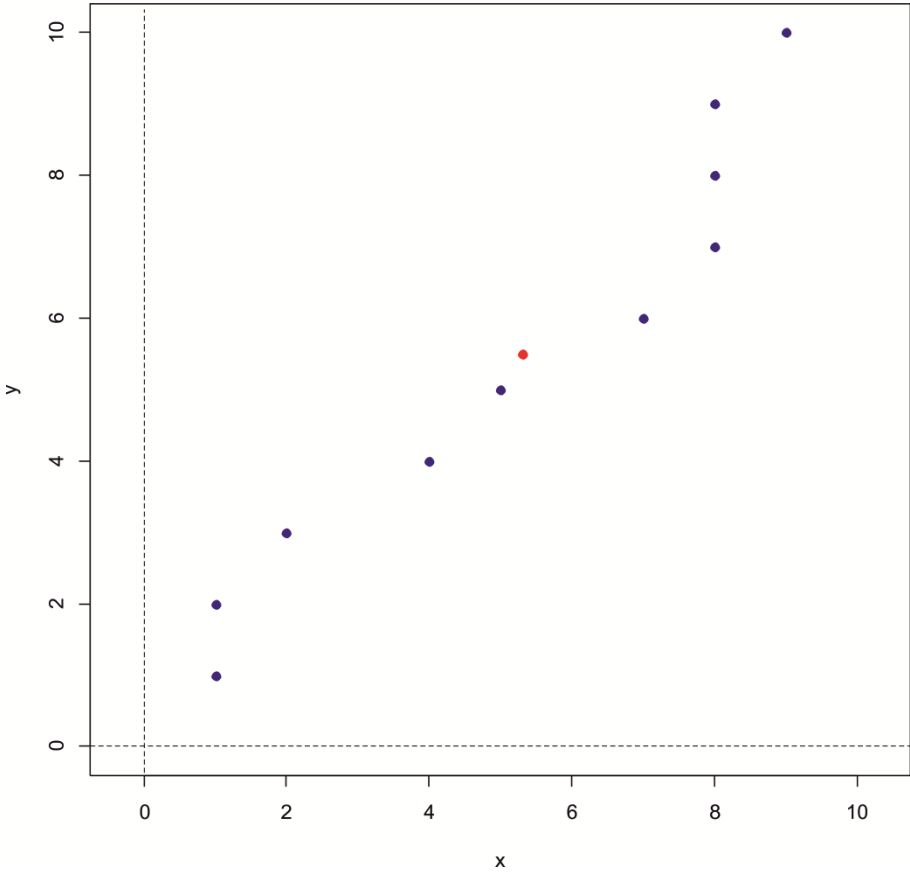
- Un nombre permettant d'évaluer
- le sens de variation de deux variables aléatoires
  - et qualifier leur dépendance

$$cov(X, Y) = \frac{\sum_{i=1}^n (X_i - \bar{X})(Y_i - \bar{Y})}{(n - 1)}$$
$$var(X) = \frac{\sum_{i=1}^n (X_i - \bar{X})(X_i - \bar{X})}{(n - 1)}$$

- Quelques propriétés
- $cov(X, Y) = cov(Y, X)$
  - $cov(X, X) = var(X, X)$

$cov(x, y) =$

$$\frac{(1 - 5.3) * (1 - 5.5) + (1 - 5.3) * (1 - 5.5) + (2 - 5.3) * (3 - 5.5) + \dots + (9 - 5.3) * (10 - 5.5)}{10 - 1} = \frac{82.5}{9} = 9.17$$





$x = 1, 1, 2, 4, 5, 7, 8, 8, 8, 9$   
 $y = 1, 2, 3, 4, 5, 6, 7, 8, 9, 10$   
 $\bar{x} = 5.3, \bar{y} = 5.5$   
 $n = 10$

$cov(x,y) =$

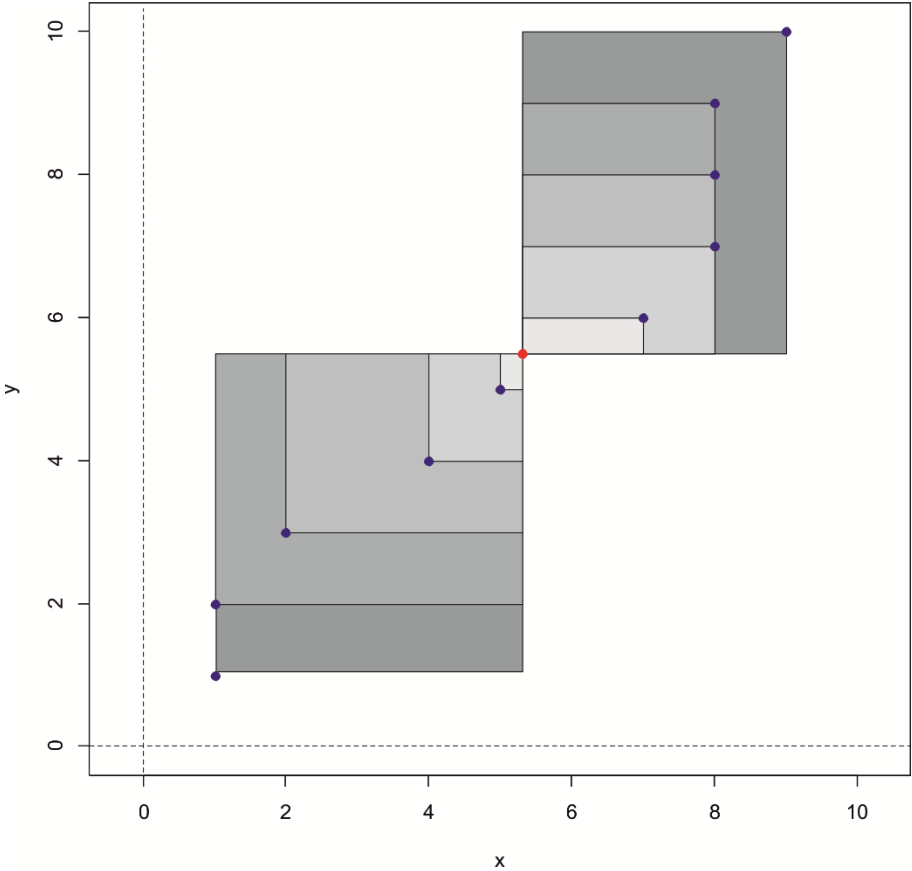
$$\frac{(1-5.3) * (1-5.5) + (1-5.3) * (1-5.5) + (2-5.3) * (3-5.5) + \dots + (9-5.3) * (10-5.5)}{10-1} = \frac{82.5}{9} = 9.17$$

Covariance

- Un nombre permettant d'évaluer
- le sens de variation de deux variables aléatoires
  - et qualifier leur dépendance

$$cov(X,Y) = \frac{\sum_{i=1}^n (X_i - \bar{X})(Y_i - \bar{Y})}{(n-1)}$$
  
$$var(X) = \frac{\sum_{i=1}^n (X_i - \bar{X})(X_i - \bar{X})}{(n-1)}$$

- Quelques propriétés
- $cov(X,Y)=cov (Y,X)$
  - $cov(X,X)=var (X,X)$



## Matrice de variance covariance

Covariance est possible seulement pour deux variables

Pour n-variables (dimensions) il faut calcules les covariances pour toutes les paires séparément

Pour 3 variables (x, y, z):

	x	y	z
x	var(x,x)	cov(x,y)	cov(x,z)
y	cov(y,x)	var(y,y)	cov(y,z)
z	cov(z,x)	cov(z,y)	var(z,z)

$$C = \begin{pmatrix} var(x,x) & cov(x,y) & cov(x,z) \\ cov(y,x) & var(y,y) & cov(y,z) \\ cov(z,x) & cov(z,y) & var(z,z) \end{pmatrix}$$

Quelques propriétés

-  $cov(x,x) = var(x,x)$

Voyons ce que ca donne sur R

```
> attach(bresil)
> x=cbind(Al,Ti,Zr,Pb,Na)
> x=x[1:15,]
> x
```

	Al	Ti	Zr	Pb	Na
[1,]	1238.000	169.0000	3.500000	14.50000	285.0000
[2,]	1273.000	101.0000	1.700000	22.10000	328.0000
[3,]	1885.000	150.0000	3.400000	35.50000	317.0000
[4,]	1830.000	147.0000	2.300000	12.00000	199.0000
[5,]	2101.000	172.0000	3.100000	18.70000	397.0000
[6,]	1593.000	148.0000	3.500000	21.70000	282.0000
[7,]	961.000	91.0000	1.900000	11.50000	197.0000
[8,]	2469.000	204.0000	3.600000	21.80000	363.0000
[9,]	2141.000	197.0000	5.900000	21.80000	374.0000
[10,]	4177.000	366.0000	8.200000	18.20000	414.0000
[11,]	1701.651	128.6372	3.429533	12.99523	226.2802
[12,]	2804.000	224.0000	5.100000	90.10000	447.0000
[13,]	1271.000	105.0000	2.300000	11.70000	321.0000
[14,]	2938.000	273.0000	3.600000	25.60000	345.0000
[15,]	3609.000	330.0000	6.800000	50.90000	582.0000

```
> |
```

```
> cov(x)
```

	Al	Ti	Zr	Pb	Na
Al	846927.774	72719.2740	1469.64265	8219.39892	68768.5569
Ti	72719.274	6673.2275	133.93321	624.29518	6058.2833
Zr	1469.643	133.9332	3.42210	14.63914	131.4817
Pb	8219.399	624.2952	14.63914	422.45710	1296.9092
Na	68768.557	6058.2833	131.48166	1296.90920	10078.4842

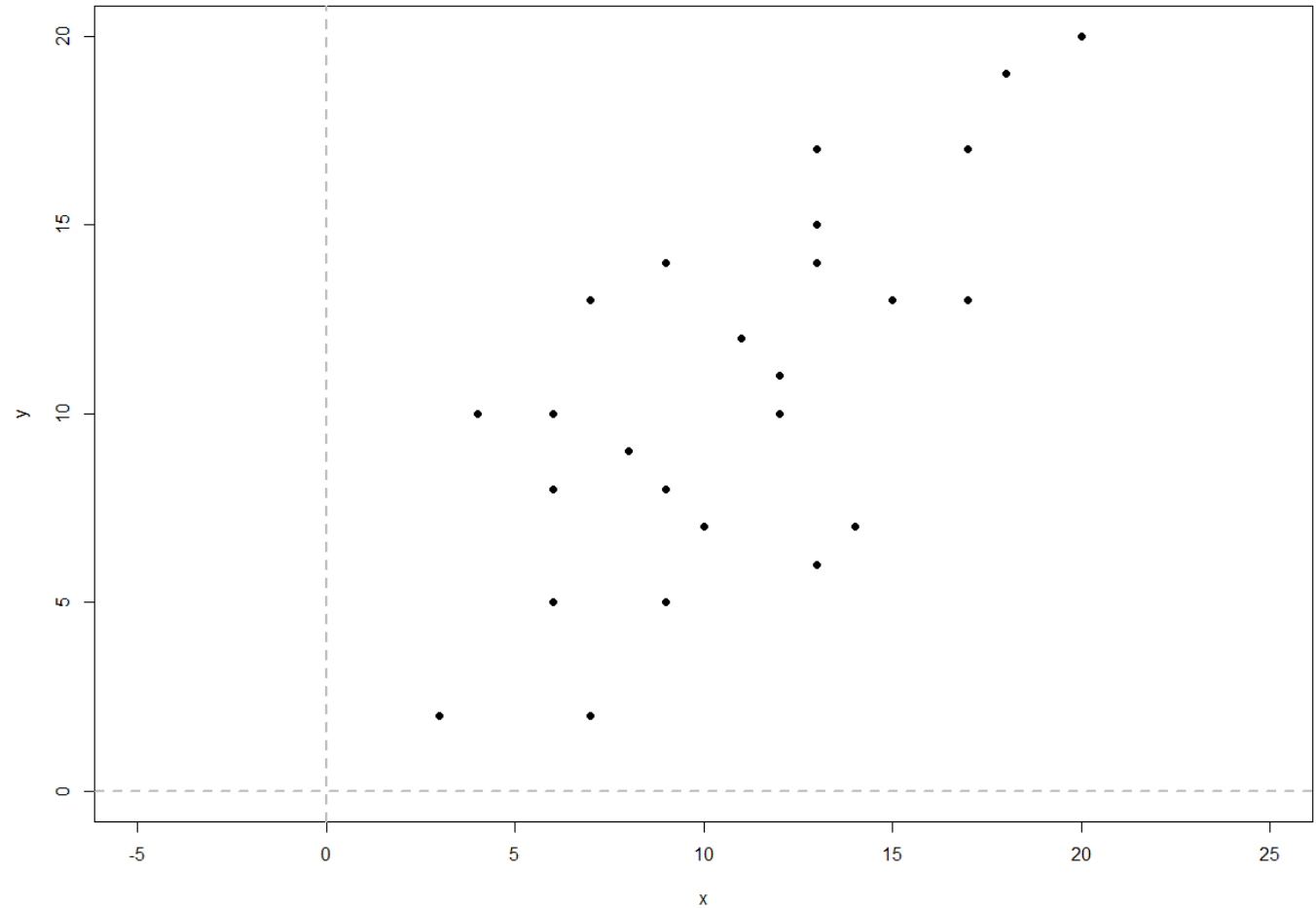
```
> cor(x)
```

	Al	Ti	Zr	Pb	Na
Al	1.0000000	0.9672937	0.8632612	0.4345358	0.7443362
Ti	0.9672937	1.0000000	0.8862862	0.3718183	0.7387271
Zr	0.8632612	0.8862862	1.0000000	0.3850154	0.7079803
Pb	0.4345358	0.3718183	0.3850154	1.0000000	0.6285223
Na	0.7443362	0.7387271	0.7079803	0.6285223	1.0000000

```
> |
```

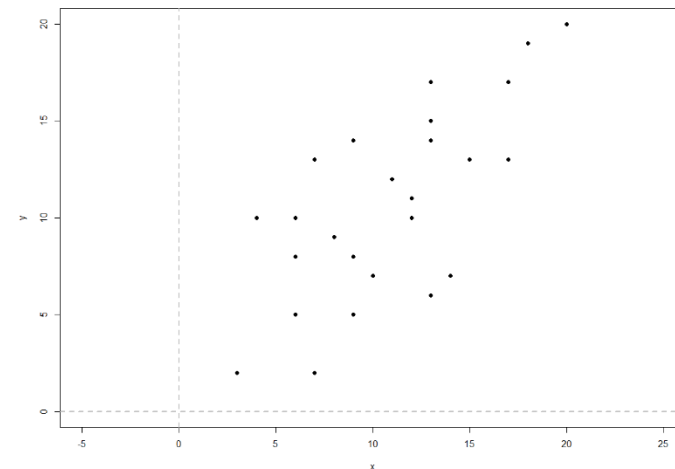
## Traitement des données – rappels: Analyse en composantes principales

```
> data
  x1 x2
1   3  2
2   4 10
3   6  5
4   6  8
5   6 10
6   7  2
7   7 13
8   8  9
9   9  5
10  9  8
11  9 14
12 10  7
13 11 12
14 12 10
15 12 11
16 13  6
17 13 14
18 13 15
19 13 17
20 14  7
21 15 13
22 17 13
23 17 17
24 18 19
25 20 20
> plot(data)
> |
```



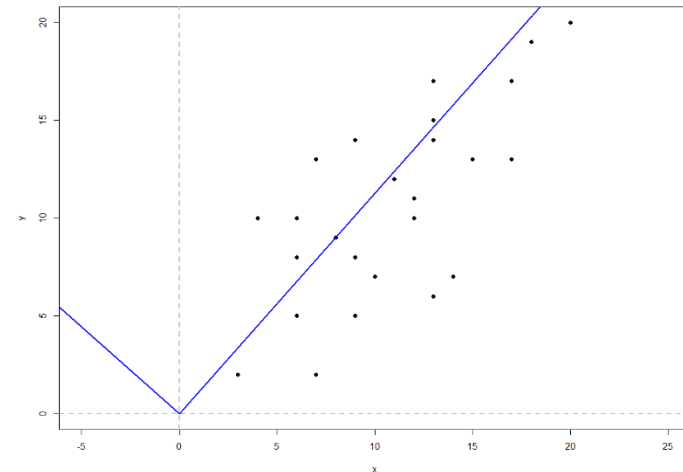
## Traitement des données – rappels: Analyse en composantes principales

```
> var(data$x1)    #variance de la variable x1
[1] 20.27667
> var(data$x2)    #variance de la variable x2
[1] 24.06
> var(data$x1)+var(data$x2) #somme des variances
[1] 44.33667
> cov(data$x1,data$x2) #covariance des variables x1,x2 identique a...
[1] 15.585
> cov(data$x2,data$x1)
[1] 15.585
> matcov=cov(data)  # matrice de variance covariance
> matcov
      x1      x2
x1 20.27667 15.585
x2 15.58500 24.060
> sum(diag(matcov)) # somme de la diagonale = somme des variances
[1] 44.33667
> |
```



# Traitement des données – rappels: Analyse en composantes principales

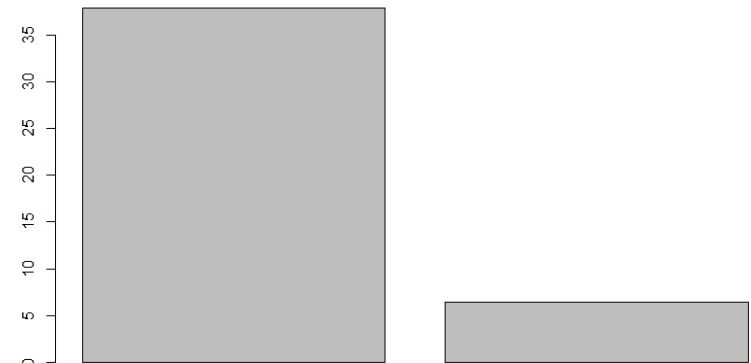
```
> var(data$x1) #variance de la variable x1
[1] 20.27667
> var(data$x2) #variance de la variable x2
[1] 24.06
> var(data$x1)+var(data$x2) #somme des variances
[1] 44.33667
> cov(data$x1,data$x2) #covariance des variables x1,x2 identique a...
[1] 15.585
> cov(data$x2,data$x1)
[1] 15.585
> matcov=cov(data) # matrice de variance covariance
> matcov
      x1      x2
x1 20.27667 15.585
x2 15.58500 24.060
> sum(diag(matcov)) # somme de la diagonale = somme des variances
[1] 44.33667
> |
```



```
> eig=eigen(cov(data)) #decomposition spectrale
> eig
$values
[1] 37.86772  6.46895

$vectors
      [,1]      [,2]
[1,] 0.6631391 -0.7484962
[2,] 0.7484962  0.6631391

> sum.eig.val=sum(eig$values) #somme totale de la variance
> barplot(eig$values)
> |
```



```
> eig$values[1]*100/sum(eig$values) #% de variance portee par l'axe 1
[1] 85.40948
> eig$values[2]*100/sum(eig$values) #% de variance portee par l'axe 2
[1] 14.59052
> |
```

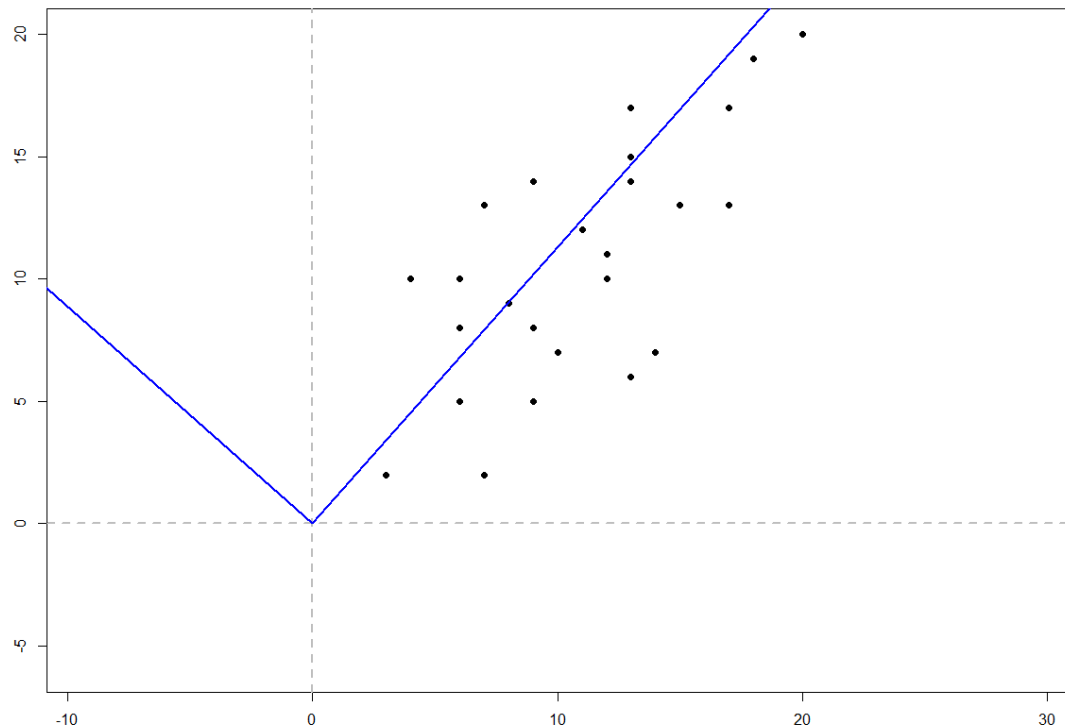
$$\% = \text{eigenvalue} * \frac{100}{\text{somme totale de la variance}}$$

## Traitement des données – rappels: Analyse en composantes principales

```
> t(eig$vectors[,1])%%eig$vectors[,2] # Les deux sont bien orthogonaux
      [,1]
[1,]      0
> t(eig$vectors[,1])%%eig$vectors[,1] # Ils ont bien une norme = 1
      [,1]
[1,]      1
> |
```

```
> SC=as.matrix(data)%%eig$vectors # calcul des PC scores
> SC # voici les nouvelles coordonnées
```

	[,1]	[,2]
[1,]	3.486410	-0.91921033
[2,]	10.137518	3.63740633
[3,]	7.721316	-1.17528155
[4,]	9.966804	0.81413576
[5,]	11.463796	2.14041397
[6,]	6.138966	-3.91319504
[7,]	14.372424	3.38133510
[8,]	12.041578	-0.01971749
[9,]	9.710733	-3.42077008
[10,]	11.956221	-1.43135277
[11,]	16.447198	2.54748185
[12,]	11.870864	-2.84298806
[13,]	16.276484	-0.27578871
[14,]	15.442631	-2.35056310
[15,]	16.191127	-1.68742400
[16,]	13.111785	-5.75161569
[17,]	19.099755	-0.44650286
[18,]	19.848251	0.21663624
[19,]	21.345243	1.54291445
[20,]	14.523421	-5.83697277
[21,]	19.677537	-2.60663432
[22,]	21.003815	-4.10362668
[23,]	23.997800	-1.45107026
[24,]	26.157931	-0.87328823
[25,]	28.232706	-1.70714149



```
# graphiquement, ca donne
```

```
> plot(SC)
> |
```

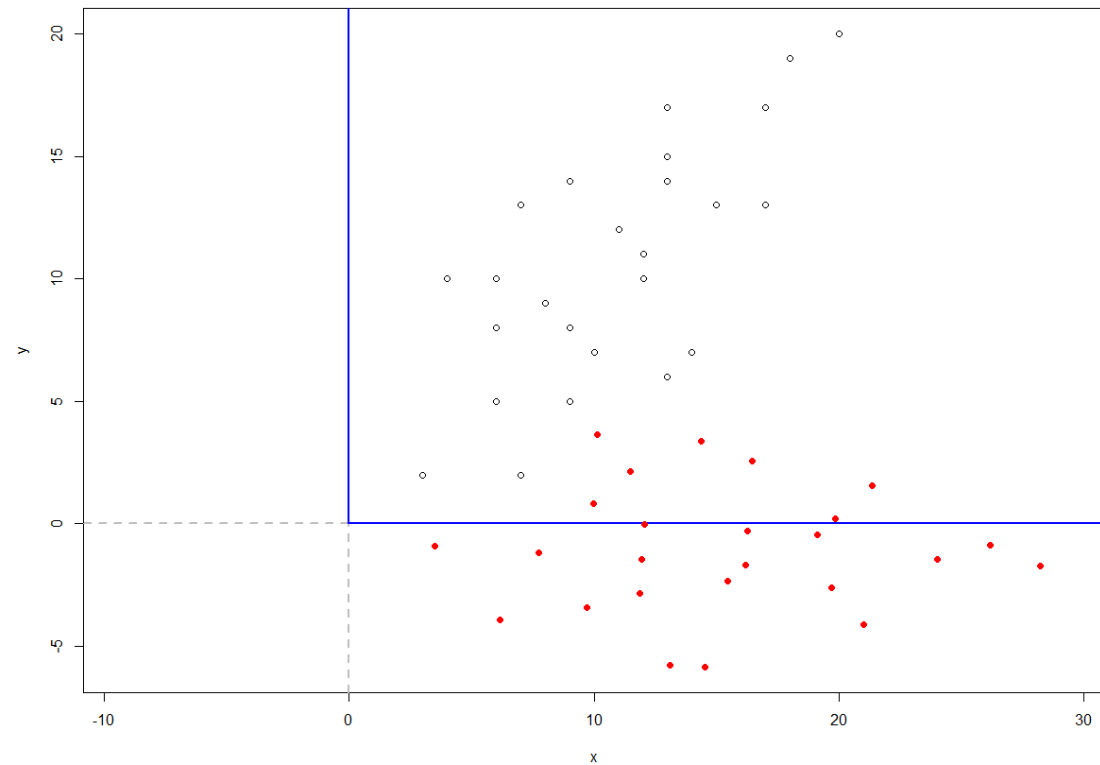


# Traitement des données – rappels: Analyse en composantes principales

```
> t(eig$vectors[,1])%%eig$vectors[,2] # Les deux sont bien orthogonaux
      [,1]
[1,]      0
> t(eig$vectors[,1])%%eig$vectors[,1] # Ils ont bien une norme = 1
      [,1]
[1,]      1
> |
```

```
> SC=as.matrix(data)%%eig$vectors # calcul des PC scores
> SC # voici les nouvelles coordonnées
```

	[,1]	[,2]
[1,]	3.486410	-0.91921033
[2,]	10.137518	3.63740633
[3,]	7.721316	-1.17528155
[4,]	9.966804	0.81413576
[5,]	11.463796	2.14041397
[6,]	6.138966	-3.91319504
[7,]	14.372424	3.38133510
[8,]	12.041578	-0.01971749
[9,]	9.710733	-3.42077008
[10,]	11.956221	-1.43135277
[11,]	16.447198	2.54748185
[12,]	11.870864	-2.84298806
[13,]	16.276484	-0.27578871
[14,]	15.442631	-2.35056310
[15,]	16.191127	-1.68742400
[16,]	13.111785	-5.75161569
[17,]	19.099755	-0.44650286
[18,]	19.848251	0.21663624
[19,]	21.345243	1.54291445
[20,]	14.523421	-5.83697277
[21,]	19.677537	-2.60663432
[22,]	21.003815	-4.10362668
[23,]	23.997800	-1.45107026
[24,]	26.157931	-0.87328823
[25,]	28.232706	-1.70714149



```
# graphiquement, ca donne
```

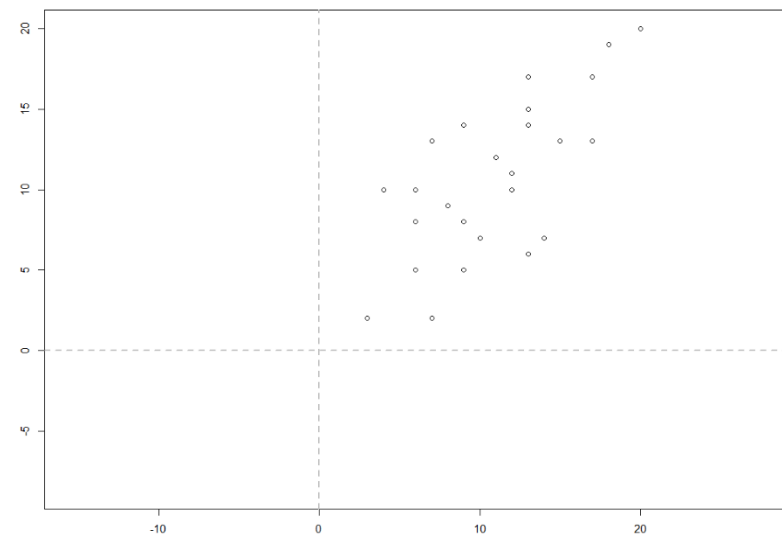
```
> plot(SC)
> |
```

# Traitement des données – rappels: Analyse en composantes principales

```
> # on aurait pu travailler sur ces variables centrees sur la moyenne
> data1=sweep(data, 2, colMeans(data)) # ote la moyenne par colonne
> summary(data1)
      x1              x2
Min.   :-7.880e+00  Min.   :-8.680e+00
1st Qu.: -3.880e+00  1st Qu.: -3.680e+00
Median :  1.200e-01  Median : -6.800e-01
Mean    :-7.824e-16  Mean    :  2.839e-16
3rd Qu.:  2.120e+00  3rd Qu.:  3.320e+00
Max.     :  9.120e+00  Max.     :  9.320e+00
> eigi=eigen(cov(data1))
> eigi                                # en fait rien ne change du cote des eigen
$values
[1] 37.86772  6.46895

$vectors
      [,1]      [,2]
[1,] 0.6631391 -0.7484962
[2,] 0.7484962  0.6631391

> SC1=as.matrix(data1)%*%eigi$vectors
> plot(SC1)                        # en revanche, les individus sur PC sont centres
```

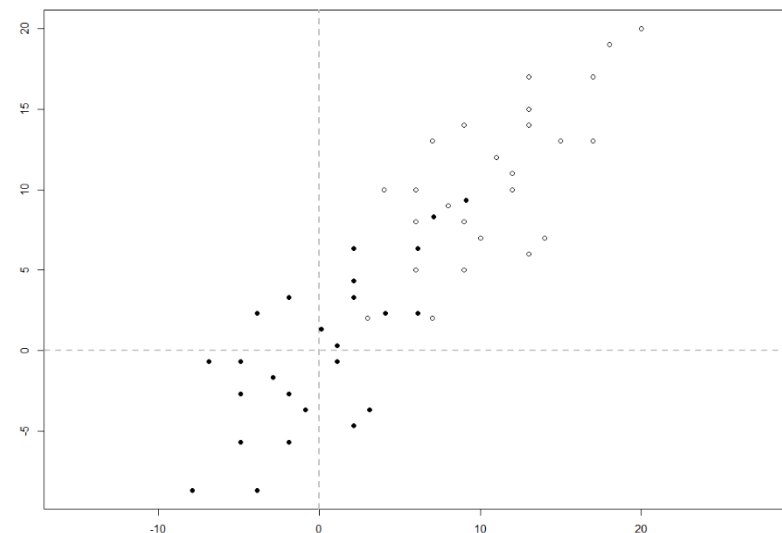


# Traitement des données – rappels: Analyse en composantes principales

```
> # on aurait pu travailler sur ces variables centrees sur la moyenne
> data1=sweep(data, 2, colMeans(data)) # ote la moyenne par colonne
> summary(data1)
      x1              x2
Min.   :-7.880e+00  Min.   :-8.680e+00
1st Qu.: -3.880e+00  1st Qu.: -3.680e+00
Median :  1.200e-01  Median : -6.800e-01
Mean    :-7.824e-16  Mean    :  2.839e-16
3rd Qu.:  2.120e+00  3rd Qu.:  3.320e+00
Max.    :  9.120e+00  Max.    :  9.320e+00
> eigi=eigen(cov(data1))
> eigi                                # en fait rien ne change du cote des eigen
$values
[1] 37.86772  6.46895

$vectors
      [,1]      [,2]
[1,] 0.6631391 -0.7484962
[2,] 0.7484962  0.6631391

> SC1=as.matrix(data1)%*%eigi$vectors
> plot(SC1)                                # en revanche, les individus sur PC sont centres
```

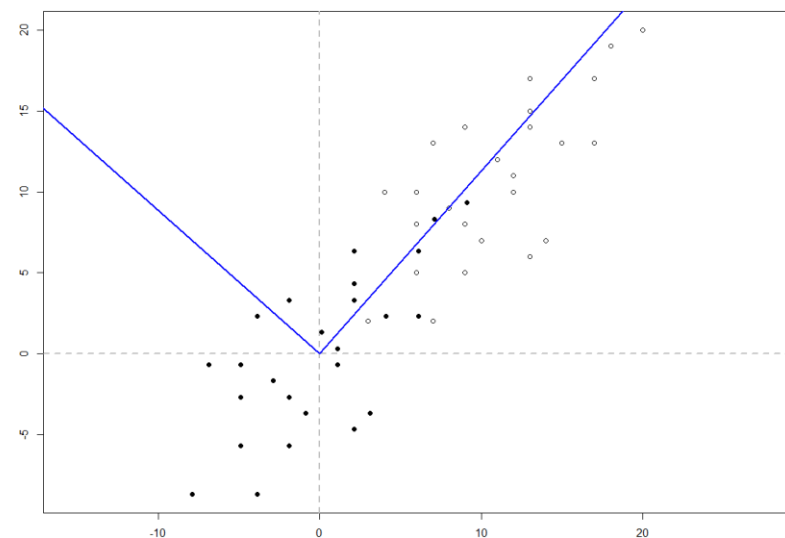


# Traitement des données – rappels: Analyse en composantes principales

```
> # on aurait pu travailler sur ces variables centrees sur la moyenne
> data1=sweep(data, 2, colMeans(data)) # ote la moyenne par colonne
> summary(data1)
      x1              x2
Min.   :-7.880e+00  Min.   :-8.680e+00
1st Qu.: -3.880e+00  1st Qu.: -3.680e+00
Median :  1.200e-01  Median : -6.800e-01
Mean    :-7.824e-16  Mean    :  2.839e-16
3rd Qu.:  2.120e+00  3rd Qu.:  3.320e+00
Max.     :  9.120e+00  Max.     :  9.320e+00
> eigi=eigen(cov(data1))
> eigi                                # en fait rien ne change du cote des eigen
$values
[1] 37.86772  6.46895

$vectors
      [,1]      [,2]
[1,] 0.6631391 -0.7484962
[2,] 0.7484962  0.6631391

> SC1=as.matrix(data1)%*%eigi$vectors
> plot(SC1)                        # en revanche, les individus sur PC sont centres
```

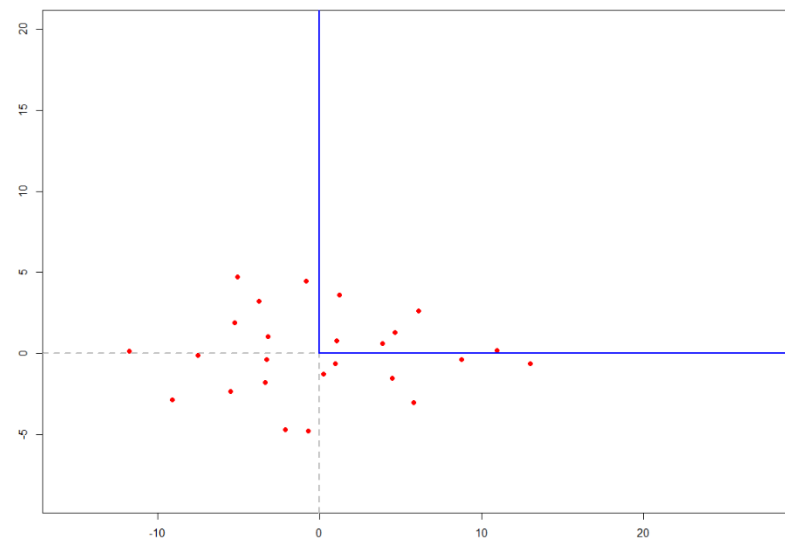


# Traitement des données – rappels: Analyse en composantes principales

```
> # on aurait pu travailler sur ces variables centrees sur la moyenne
> data1=sweep(data, 2, colMeans(data)) # ote la moyenne par colonne
> summary(data1)
      x1              x2
Min.   :-7.880e+00  Min.   :-8.680e+00
1st Qu.: -3.880e+00  1st Qu.: -3.680e+00
Median :  1.200e-01  Median : -6.800e-01
Mean    :-7.824e-16  Mean    :  2.839e-16
3rd Qu.:  2.120e+00  3rd Qu.:  3.320e+00
Max.     :  9.120e+00  Max.     :  9.320e+00
> eigi=eigen(cov(data1))
> eigi                                # en fait rien ne change du cote des eigen
$values
[1] 37.86772  6.46895

$vectors
      [,1]      [,2]
[1,] 0.6631391 -0.7484962
[2,] 0.7484962  0.6631391

> SC1=as.matrix(data1)%*%eigi$vectors
> plot(SC1)                                # en revanche, les individus sur PC sont centres
```

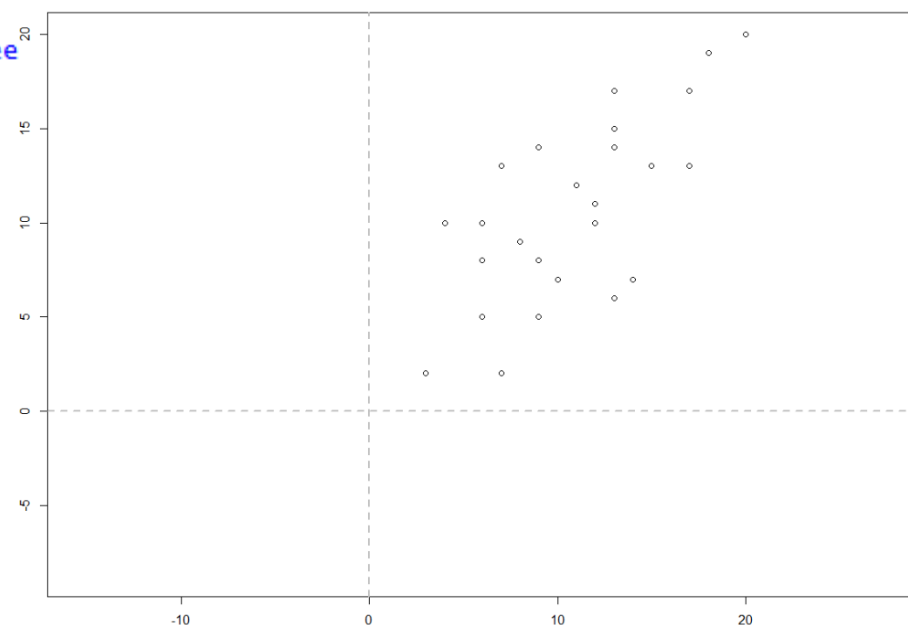


## Traitement des données – rappels: Analyse en composantes principales

```
> # on aurait pu travailler sur ces variables standardisees
> data2=scale(data) # centre - reduit
> sd(data2[,1])
[1] 1
> eig2=eigen(cor(data2)) #cov(data2)=cor(data2)!
> eig2
# ca change
$values
[1] 1.7056035 0.2943965

$vectors
      [,1]      [,2]
[1,] 0.7071068 -0.7071068
[2,] 0.7071068  0.7071068

> SC2=as.matrix(data2)%*%eig2$vectors
> plot(SC2)
```

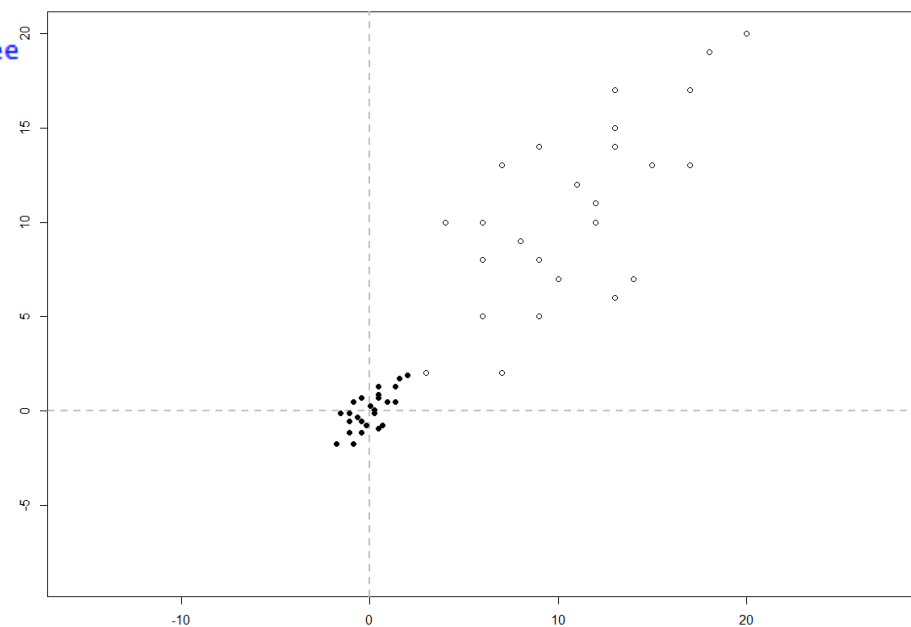


## Traitement des données – rappels: Analyse en composantes principales

```
> # on aurait pu travailler sur ces variables standardisee
> data2=scale(data) # centre - reduit
> sd(data2[,1])
[1] 1
> eig2=eigen(cor(data2))    #cov(data2)=cor(data2)!
> eig2                      # ca change
$values
[1] 1.7056035 0.2943965

$vectors
      [,1]      [,2]
[1,] 0.7071068 -0.7071068
[2,] 0.7071068  0.7071068

> SC2=as.matrix(data2)%*%eig2$vectors
> plot(SC2)
```

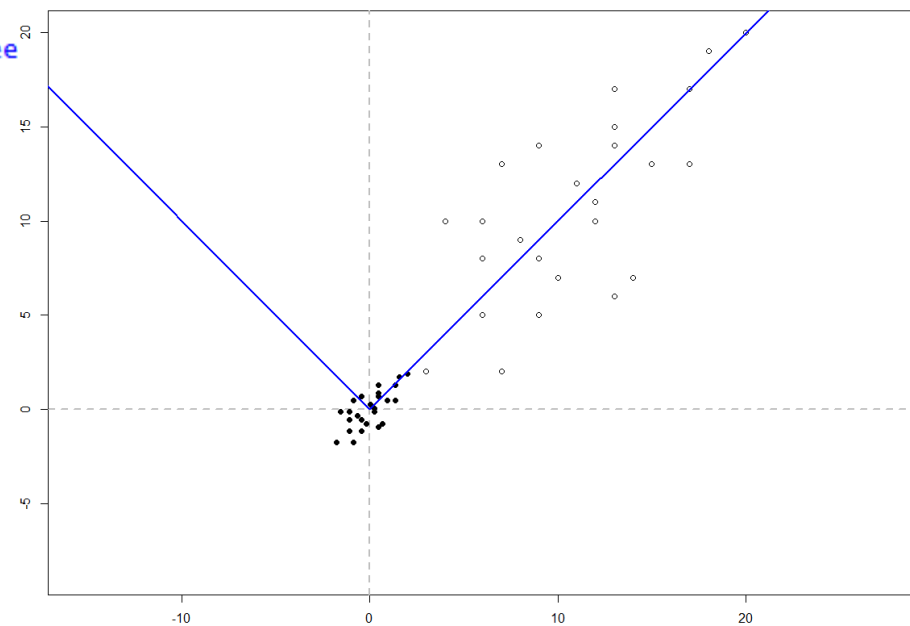


# Traitement des données – rappels: Analyse en composantes principales

```
> # on aurait pu travailler sur ces variables standardisee
> data2=scale(data) # centre - reduit
> sd(data2[,1])
[1] 1
> eig2=eigen(cor(data2)) #cov(data2)=cor(data2)!
> eig2
# ca change
$values
[1] 1.7056035 0.2943965

$vector
      [,1]      [,2]
[1,] 0.7071068 -0.7071068
[2,] 0.7071068  0.7071068

> SC2=as.matrix(data2)%*%eig2$vector
> plot(SC2)
```



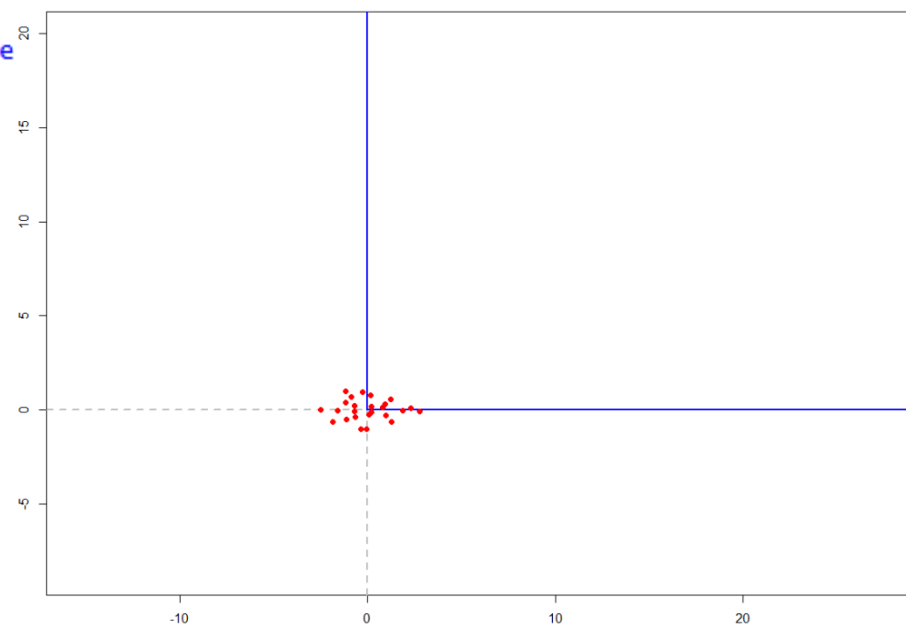


## Traitement des données – rappels: Analyse en composantes principales

```
> # on aurait pu travailler sur ces variables standardisee
> data2=scale(data) # centre - reduit
> sd(data2[,1])
[1] 1
> eig2=eigen(cor(data2)) #cov(data2)=cor(data2)!
> eig2
# ca change
$values
[1] 1.7056035 0.2943965

$vectors
      [,1]      [,2]
[1,] 0.7071068 -0.7071068
[2,] 0.7071068  0.7071068

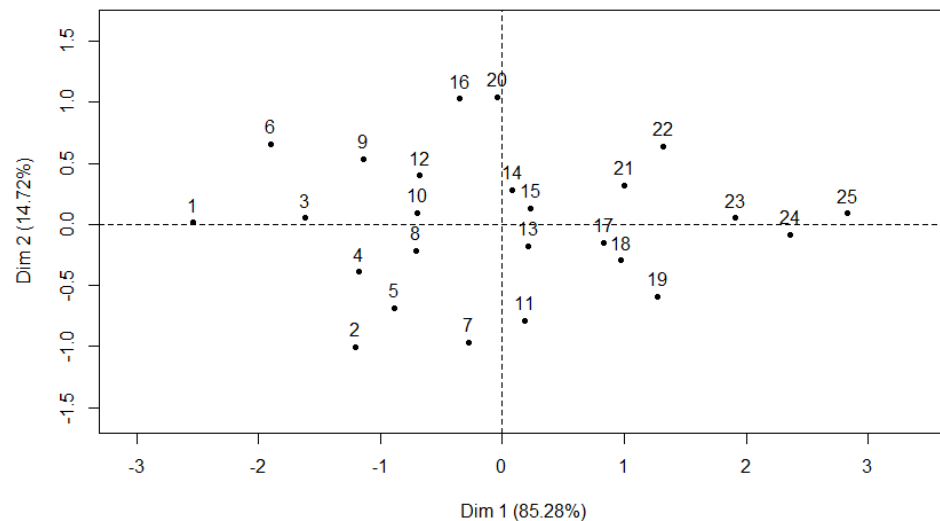
> SC2=as.matrix(data2)%*%eig2$vectors
> plot(SC2)
```



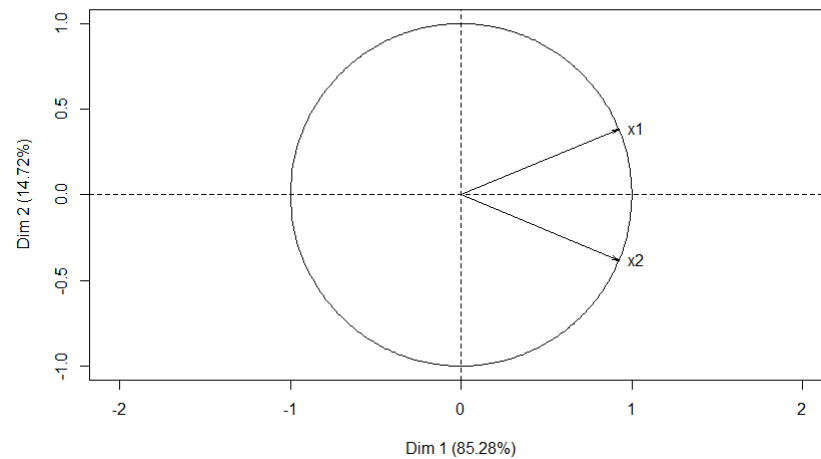
# Traitement des données – rappels: Analyse en composantes principales

```
> library(FactoMiner)
> pc=PCA(data,scale.unit=T)
Messages d'avis :
1: In function () : only one RStudio graphics device is permitted
2: In function () : only one RStudio graphics device is permitted
> |
```

Individuals factor map (PCA)



Variables factor map (PCA)



## Traitement des données – rappels: Analyse en composantes principales

```
> summary(pc)
      Length Class      Mode
eig    3      data.frame list
var    4      -none-     list
ind    4      -none-     list
svd    3      -none-     list
call   7      -none-     list
> pc$eig
      eigenvalue percentage of variance cumulative percentage of variance
comp 1  1.7056035           85.28017           85.28017
comp 2  0.2943965           14.71983           100.00000
> pc$var
$coord
      Dim.1      Dim.2
x1 0.9234727 0.3836643
x2 0.9234727 -0.3836643

$cor
      Dim.1      Dim.2
x1 0.9234727 0.3836643
x2 0.9234727 -0.3836643

$cos2
      Dim.1      Dim.2
x1 0.8528017 0.1471983
x2 0.8528017 0.1471983

$contrib
      Dim.1 Dim.2
x1    50    50
x2    50    50

> |
```

```
> pc$ind$coord
      Dim.1      Dim.2
1 -2.54001371 0.01416509
2 -1.20270320 -1.00260625
3 -1.61781461 0.05358349
4 -1.17642427 -0.38780685
5 -0.88216404 -0.68206708
6 -1.89893538 0.65524342
7 -0.28050411 -0.96318785
8 -0.70875499 -0.21439780
9 -1.13700587 0.53439224
10 -0.69561552 0.09300189
11 0.18716517 -0.78977880
12 -0.68247606 0.40040159
13 0.21344410 -0.17497940
14 0.07945345 0.27955041
15 0.22658357 0.13242030
16 -0.34879742 1.02834045
17 0.82824350 -0.14870047
18 0.97537361 -0.29583058
19 1.26963384 -0.59009081
20 -0.04139773 1.04147992
21 1.00165255 0.31896881
22 1.32219171 0.63950798
23 1.91071217 0.05098752
24 2.36524198 -0.08300313
25 2.83291126 0.09040592

> |
```