# Statistics with SciPy

# Python

Ricco Rakotomalala

http://eric.univ-lyon2.fr/~ricco/cours/cours_programmation_python.html

# Scipy ?

- SciPy is a library for scientific computing in Python.
- It incorporates, among others, modules for data analysis.
- The library is based on the data structures from NumPy (vectors and matrices)

## Reference

- Clustering package (`scipy.cluster`)
- Constants (`scipy.constants`)
- Discrete Fourier transforms (`scipy.fftpack`)
- Integration and ODEs (`scipy.integrate`)
- Interpolation (`scipy.interpolate`)
- Input and output (`scipy.io`)
- Linear algebra (`scipy.linalg`)
- Miscellaneous routines (`scipy.misc`)
- Multi-dimensional image processing (`scipy.ndimage`)
- Orthogonal distance regression (`scipy.odr`)
- Optimization and root finding (`scipy.optimize`)
- Signal processing (`scipy.signal`)
- Sparse matrices (`scipy.sparse`)
- Sparse linear algebra (`scipy.sparse.linalg`)
- Compressed Sparse Graph Routines (`scipy.sparse.csgraph`)
- Spatial algorithms and data structures (`scipy.spatial`)
- Special functions (`scipy.special`)
- Statistical functions (`scipy.stats`)
- Statistical functions for masked arrays (`scipy.stats.mstats`)
- C/C++ integration (`scipy.weave`)

We will focus in particular on statistical and data analysis modules.

It is not possible to describe all the functions in this slideshow. To go further, see the reference manual.

http://docs.scipy.org/doc/scipy/reference/

Treat one vector of values

Descriptive statistics, test for normality, hypothesis testing, etc.

# STATISTICS WITH ONE VECTOR

# Dataset: Beaded of Shoshoni (http://www.statsci.org/data/general/shoshoni.html)

**StatSci.org** / Home

OzDASL

## Width to Length Ratios in Shoshoni Handicraft

Keywords: t-test

---

### Description

The data are width-to-length ratios of beaded rectangles used by the Shoshoni Indians of America to decorate their leather goods. One might ask whether the golden rectangle (for which the width-to-length ratio is 0.618) can be considered an aesthetic standard for the Shoshonis just as it was for the Greeks and the Egyptians.

### Download

Data File (tab-delimited text)

### Source

Dubois, Cora (ed.) (1960). *Lowie's Selected Papers in Anthropology*. University of California Press, Berkeley.

Larsen, R.J., and Marx, M.L., *An Introduction to Mathematical Statistics and Its Applications* 2nd Edition. Prentice-Hall, 1986. Case Study 1.2.2.

Two outliers are removed (see Iglzwizc & Hoaglin Modified Z-score) (this kind of solution is questionable… but the purpose here is to describe the statistical functions of SciPy)

```
d = np.array([0.553,0.57,0.576,0.601,0.606,0.606,0.609,0.611,0.615,0.628,0.654,0.662,0.668,0.67,0.672,0.69,0.693,0.749])
```

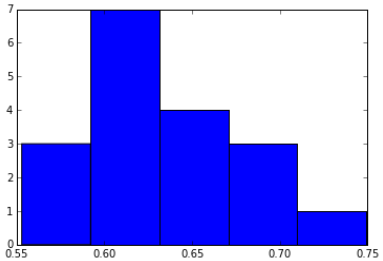
# Descriptive statistics

```
import numpy as np
import scipy.stats as stat #we use the alias stat to access to the functions in stats (SciPy)

#descriptive statistics
stat_des = stat.describe(d)
print(stat_des)
DescribeResult(nobs=18, minmax=(0.55300000000000005, 0.749), mean=0.63516666666666666,
variance=0.0025368529411764714, skewness=0.38763289979752136, kurtosis=-0.35873690487519205)

#stat_des is of the type "namedtuple“ - there are various ways to access to the properties

#index
print(stat_des[0]) # 18, the 1st property (indice 0) is nobs

#name
print(stat_des.nobs) # 18

# multiple assignment
n,mm,m,v,sk,kt = stat.describe(d)
print(n,m) # 18   0.635166, number of examples and the mean

#median using NumPy function
print(np.median(d)) # 0.6215

#percentile rank of a score
print(stat.percentileofscore(d,0.6215)) # 50.0, the half of the examples has a value lower than 0.6215
```

Goal: get the values of quantiles and cumulative distribution functions for various statistical distributions frequently used for statistical inference.

```
# standard normal distribution
print(stat.norm.ppf(0.95,loc=0,scale=1)) # percent point of 0.95 = 1.64485
print(stat.norm.cdf(1.96,loc=0,scale=1)) # 0.975

# Student distribution – degree of freedom = 30
print(stat.t.ppf(0.95,df=30)) # 1.6972
print(stat.t.cdf(1.96,df=30)) # 0.9703

# chi-squared distribution - df = 10
print(stat.chi.ppf(0.95,df=10)) # 4.2787
print(stat.chi.cdf(4.84,df=10)) # 0.9907

# Fisher-Snedecor distribution, df numerator = 1, df denominator = 30
print(stat.f.ppf(0.95,dfn=1,dfd=30)) # 4.1709
print(stat.f.cdf(3.48,dfn=1,dfd=30)) # 0.9281
```

Goal: testing the null hypothesis that a sample comes from a normal distribution.




Histogram (see the matplotlib module)

```
#D'Agostino and Pearson's test
ag = stat.normaltest(d) # warning message, n is too low for a reliable test
print(ag) # (0.714, 0.699), test  statistic and p-value (if p-value < α, reject normality hypothesis)

#Shapiro-Wilks' test
sp = stat.shapiro(d)
print(sp) # (0.961, 0.628), test statistic and p-value

#Anderson-Darling test, more general but can be used for normality test
ad = stat.anderson(d,dist="norm") # "norm" for normality test
print(ad) # (0.3403, array([ 0.503,  0.573,  0.687,  0.802,  0.954]), array([ 15. ,  10. ,   5. ,   2.5,   1. ])) ➔ test statistic, critical value for each alpha, here, we note that p-value is upper than 15%
```

For a description of theses approaches, see R. Rakotomalala, « Tests de normalité – Techniques empiriques et tests statistiques », Version 2.0, 2008 (in French).

# Random number generation - Sampling

Goal: a random number generator allows to perform simulations or using techniques based on resampling (bootstrap, etc.). Both SciPy, NumPy offer tools for this purpose.

```
#random sample (30 values) from a standard normal distribution
alea1 = stat.norm.rvs(loc=0,scale=1,size=30)
print(stat.normaltest(alea1)) # (2.16, 0.338), compatible with a normal distribution

#exponential distribution
alea2 = stat.expon.rvs(size=30)
print(stat.normaltest(alea2)) # (17.62, 0.00015), not compatible to normal dist. of course

#Numpy has also a generator
alea3 = np.random.normal(loc=0,scale=1,size=30)
print(stat.normaltest(alea3)) # (2.41, 0.299), compatible

#sampling size = 5 values among n = 18 [len(d)]
d1 = np.random.choice(d,size=5,replace=False) #without replacement
print(d1) # (0.69  0.628  0.606  0.662  0.668)

d2 = np.random.choice(d,size=5,replace=True) #with replacement
print(d2) # (0.654  0.67  0.628  0.654  0.609)
```

# Comparison to an assumed mean

Test : H0 : $\mu = 0.618$

H1 : $\mu \neq 0.618$

```
#test for a single population mean
print(stat.ttest_1samp(d,popmean=0.618))
# (1.446, 0.166), test statistic and p-value: p-value < α, reject of H0

#*** we detail the calculations ***
#sample mean
m = np.mean(d) # 0.6352
#sample standard deviation – ddof = 1 to perform the calculation : 1/(n-1)
sigma = np.std(d,ddof=1) # 0.0504
#test statistic
import math
t = (m - 0.618)/(sigma/math.sqrt(d.size))
print(t) # 1.446, we obtain the value above
#p-value – non directional test
#t Student distribution, cdf() : cumulative distribution function
p = 2.0 * (1.0 - stat.t.cdf(math.fabs(t),d.size-1))
print(p) # 0.166, p-value above
```

Treat two vectors of values

Comparing two populations, measures of association, etc.

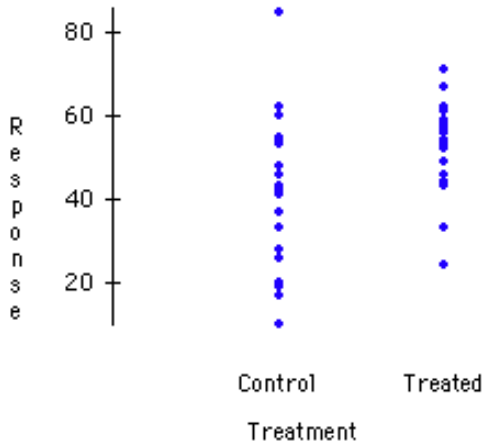
# STATISTICS WITH TWO VECTORS

# Dataset: DRP Scores (http://lib.stat.cmu.edu/DASL/Stories/ImprovingReadingAbility.html)

**Story Name:** Improving Reading Ability
**Story Topics:** Education
**Datafile Name:** DRP Scores
**Methods:** Two sample t-test , Summary statistics
**Abstract:** An educator conducted an experiment to test whether new directed reading activities in the classroom will help elementary school pupils improve some aspects of their reading ability. She arranged for a third grade class of 21 students to follow these activities for an 8-week period. A control classroom of 23 third graders followed the same curriculum without the activities. At the end of the 8 weeks, all students took a Degree of Reading Power (DRP) test, which measures the aspects of reading ability that the treatment is designed to improve.

Summary statistics on the two groups of children show that the average score of the treatment class was almost ten points higher than the average of the control class. A two-sample t-test is appropriate for testing whether this difference is statistically significant. The t-statistic is 2.31, which is significant at the .05 level.

**Image:** Boxplots of test score for treatment and control classes.



# Comparing two populations – Independent samples

```
import numpy as np
import scipy.stats as stat
#treated – values for the individuals who followed the treatment
dt = np.array([24,43,58,71,43,49,61,44,67,49,53,56,59,52,62,54,57,33,46,43,57])
#control – control sample
dc = np.array([42,43,55,26,62,37,33,41,19,54,20,85,46,10,17,60,53,42,37,42,55,28,48])

#t-test – comparing means – equal variances assumed
print(stat.ttest_ind(dt,dc)) # (t = 2.2665, p-value = 0.0286)
#Welch t-test – comparing means – equal variances not assumed
print(stat.ttest_ind(dt,dc,equal_var=False)) # (2.3109, 0.0264)

#Mann-Whitney test - non parametric – with correction of continuity
print(stat.mannwhitneyu(dt,dc)) # (stat. U = 135, p-value unilatérale = 0.00634)

#Bartlett test – comparing variances
print(stat.bartlett(dt,dc)) # (stat. = 3.8455, p-value = 0.0498)
#Ansari Bradley test
print(stat.ansari(dt,dc)) # (stat. = 266, p-value = 0.2477)
#Levene test
print(stat.levene(dt,dc)) # (stat. = 2.342, p-value = 0.1334)

#Kolomogorov-Smirnov test – distance between two empirical distribution functions
print(stat.ks_2samp(dt,dc)) # (stat. = 0.4699, p-value = 0.0099)
```

See R. Rakotomalala (in French):

Comparaison de populations – Tests paramétriques

Comparaison de populations – Test non paramétriques

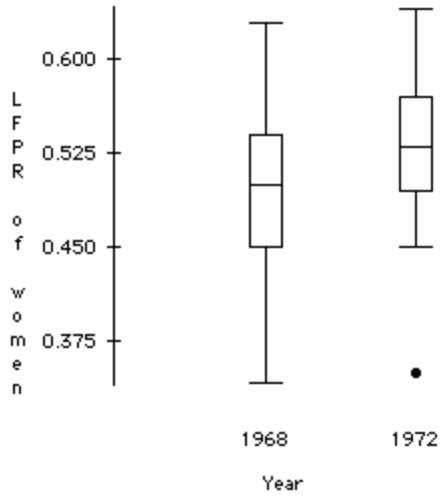
# Comparing populations – Paired samples (1/2)

**Story Name:** Women in the Labor Force
**Story Topics:** Sociology , Economics
**Datafile Name:** Labor Force
**Methods:** Boxplot , Paired t-test
**Abstract:** This dataset contains the labor force participation rate (LFPR) of women in 19 cities in the United States in each of two years (1968 and 1972). The data help to measure the growing presence of women in the labor force over this period.

It may seem reasonable to compare LFPR rates in the two years with a pooled t-test since the United States did not change much from 1968 to 1972. However, the data are natually paired because the measurements were made in the same cities for each of the two years. It is better to compare each city in 1972 to its own value in 1968.

The data offer a good example of how a paired t-test can be more effective when it is appropriate. Here, a pooled t-test is not significant, but a paired test is significant and the .05 level for testing the null hypothesis of no change in the LFPR between 1968 and 1972.

**Image:** Boxplots of the labor force participation rate of women in 19 U.S. cities in the years 1968 and 1972




http://lib.stat.cmu.edu/DASL/Stories/WomenintheLaborForce.html

# Comparing populations – Paired samples (2/2)

```
#paired samples test
d1968 = np.array([0.42,0.5,0.52,0.45,0.43,0.55,0.45,0.34,0.45,0.54,0.42,0.51,0.49,0.54,0.5,0.58,0.49,0.56,0.63])
d1972 = np.array([0.45,0.5,0.52,0.45,0.46,0.55,0.60,0.49,0.35,0.55,0.52,0.53,0.57,0.53,0.59,0.64,0.5,0.57,0.64])

#t-test related samples - parametric
print(stat.ttest_rel(d1968,d1972))# (test statistic = -2.45, p-value = 0.024)

#wilcoxon signed rank test – non parametric
print(stat.wilcoxon(d1968,d1972)) #  (test stat. = 16, p-value = 0.0122)
```

➔ The participation rate is significantly different at the 5% level in 1972.

See R. Rakotomalala (in French)

Comparaison de populations – Tests paramétriques

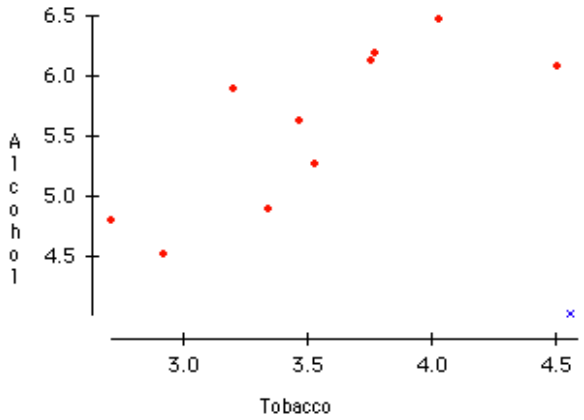
Comparaison de populations – Test non paramétriques

# Measures of association (1/2)

**Story Name:** Alcohol and Tobacco
**Story Topics:** Consumer , Health
**Datafile Name:** Alcohol and Tobacco
**Methods:** Correlation , Dummy variable , Outlier , Regression , Scatterplot
**Abstract:** Data from a British government survey of household spending may be used to examine the relationship between household spending on tobacco products and alcholic beverages. A scatterplot of spending on alcohol vs. spending on tobacco in the 11 regions of Great Britain shows an overall positive linear relationship with Northern Ireland as an outlier. Northern Ireland's influence is illustrated by the fact that the correlation between alcohol and tobacco spending jumps from .224 to .784 when Northern Ireland is eliminated from the dataset.

This dataset may be used to illustrate the effect of a single influential observation on regression results. In a simple regression of alcohol spending on tobacco spending, tobacco spending does not appear to be a significant predictor of tobacco spending. However, including a dummy variable that takes the value 1 for Northern Ireland and 0 for all other regions results in significant coefficients for both tobacco spending and the dummy variable, and a high R-squared.

**Image:** Scatterplot of Alcohol vs. Tobacco, with Northern Ireland marked with a blue X.




http://lib.stat.cmu.edu/DASL/Stories/AlcoholandTobacco.html

# Measures of association (2/2)

```
#two vectors for correlation and regression (without North Ireland)
dalc = np.array([6.47,6.13,6.19,4.89,5.63,4.52,5.89,4.79,5.27,6.08])
dtob = np.array([4.03,3.76,3.77,3.34,3.47,2.92,3.2,2.71,3.53,4.51])

#Pearson's correlation
print(stat.pearsonr(dalc,dtob)) # (r = 0.7843, p-value pour test t = 0.0072)

#Spearman's rank correlation
print(stat.spearmanr(dalc,dtob)) # (rho = 0.8303, p-value = 0.0029)

#Kendall's tau coefficient based on concordant and discordant pairs
print(stat.kendalltau(dalc,dtob)) # (tau = 0.6444, p-value = 0.0095)

#simple linear regression
print(stat.linregress(dalc,dtob)) # (pente = 0.6115, const = 0.1081, r =
0.7843, p-value test signif. pente = 0.0072, sigma err = 0.1710)
```

See R. Rakotomalala (in French)

Analyse de corrélation – Etude des dépendances

Econométrie – La régression simple et multiple

Comparing three or more populations (independent samples)

# STATISTICS WITH (K > 2) VECTORS

# Comparing K populations (1/2)

**Story Name:** Hot dogs
**Story Topics:** Food , Nutrition , Science
**Datafile Name:** Hot dogs
**Methods:** ANOVA
**Abstract:** People who are concerned about their health may prefer hot dogs that are low in salt and calories. The "Hot dogs" datafile contains data on the sodium and calories contained in each of 54 major hot dog brands. The hot dogs are classified by type: beef, poultry, and meat (mostly pork and beef, but up to 15% poultry meat).

A simple ANOVA model with type as the independent variable and calories as the dependent variable has an F-ratio of 16.074 , which is highly significant. Post-ANOVA analysis of group means shows that meat and beef hot dogs have approximately the same number of calories and poultry hot dogs generally have fewer calories than either beef or meat hot dogs.

The same model with sodium as the dependent variable has an overall F-ratio of 1.78, which is not significant. None of the individual differences are significant.

http://lib.stat.cmu.edu/DASL/Stories/Hotdogs.html

# Comparing K populations (2/2)

```
#salt – 3 independent samples
dbeef = np.array([495,477,425,322,482,587,370,322,479,375,330,300,386,401,645,440,317,319,298,253])
dmeat = np.array([458,506,473,545,496,360,387,386,507,393,405,372,144,511,405,428,339])
dpoultry = np.array([430,375,396,383,387,542,359,357,528,513,426,513,358,581,588,522,545])

#comparing variances
print(stat.levene(dbeef,dmeat,dpoultry)) # (stat. = 0.2494, p-value = 0.7802)

#comparing means -  one way anova - parametric
print(stat.f_oneway(dbeef,dmeat,dpoultry)) # (stat. F = 1.7778, p-value = 0.1793)

#Kruskal-Wallis test – one way anova on ranks – non-parametric
print(stat.kruskal(dbeef,dmeat,dpoultry)) # (stat. K = 4.1728, p-value = 0.0947)
```

➔ At 5% level, the contents in salt of the hot dogs are not different.

Hierarchical agglomerative clustering and K-means

# CLUSTER ANALYSIS

# Aim of the cluster analysis

Grouping a set of objects in such a way that objects in the same group (called a cluster) are more similar (in some sense or another) to each other than to those in other groups (clusters) (Wikipedia). The number of groups may be predetermined or obtained by calculations.

**Datafile Name:** Cities
**Datafile Subjects:** Economics
**Story Names:** Economics of Cities
**Reference:** *Prices and earnings around the globe* Economic Research Department, Union Bank of Switzerland, Zurich.
**Authorization:** Contact Authors
**Description:** The data were collected by the Economic Research Department of the Union Bank of Switzerland. They represent the economic conditions in 48 cities around in world in 1991.
**Number of cases:** 48
**Variable Names:**

1. City: City name
2. Work: Weighted average of the number of working hours in 12 occupations
3. Price: Index of the cost 112 goods and services excluding rent (Zurich = 100)
4. Salary: Index of hourly earnings in 12 occupations after deductions (Zurich = 100)

**The Data:**

| City | Work | Price | Salary | |
|---|---|---|---|---|
| Amsterdam | | 1714 | 65.6 | 49.0 |
| Athens | 1792 | 53.8 | 30.4 | |
| Bogota | 2152 | 37.9 | 11.5 | |
| Bombay | 2052 | 30.3 | 5.3 | |

*For the sake of simplicity…*

The instances with missing values are removed.

Only the variables **Price** and **Salary** are used.

# Load and transform the dataset

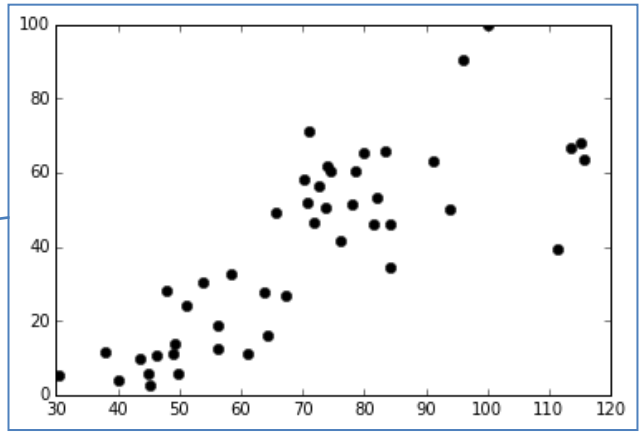
Layout of the data in the file

datacluster.txt

```
1  #Price  Salary
2  65.6    49
3  53.8    30.4
4  37.9    11.5
5  30.3    5.3
6  73.8    50.5
7  56.1    12.5
8  61  10.9
```

Dos\Windo UTF-8 INS

```
import numpy as np
import scipy.stats as stat
import scipy.cluster as cluster

#loading the data matrix
M = np.loadtxt("datacluster.txt",delimiter="\t",dtype=float)
#standard deviation of rows in each column
np.std(M,axis=0) # array([21.15, 24.487])

#scatter plot
import matplotlib.pyplot as plt
plt.plot(M[:,0],M[:,1],"ko")

#standardization of the data
#axis=0, calculation on rows for each column
#ddof = 0, using 1/(n-0) for the calculation of the variance
Z = stat.zscore(M,axis=0,ddof=0)
#standard deviation after the transformation
np.std(Z,axis=0) # array([1. , 1.])
```




The objective is to put the variables on the same scale.

# K-means

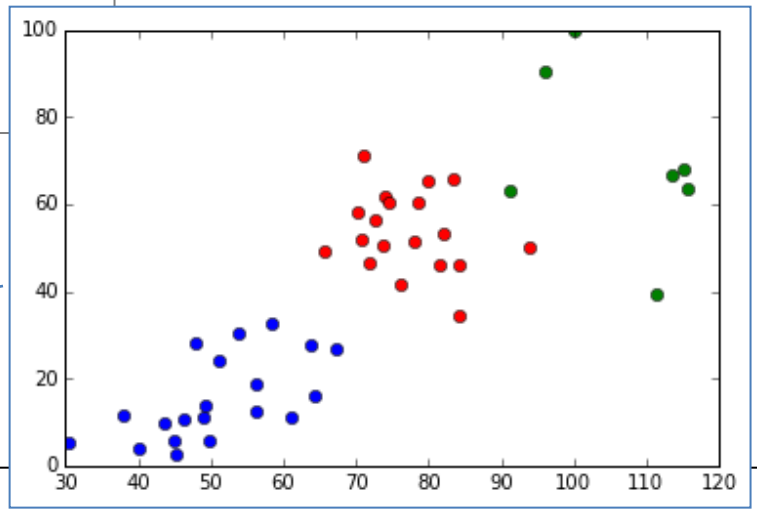
| City | Groupe affecté |
|---|---|
| Copenhagen | 2 |
| Geneva | 2 |
| Helsinki | 2 |
| Oslo | 2 |
| Stockholm | 2 |
| Tokyo | 2 |
| Zurich | 2 |
| Amsterdam | 1 |
| Brussels | 1 |
| Chicago | 1 |
| Dublin | 1 |
| Dusseldorf | 1 |
| Frankfurt | 1 |
| Houston | 1 |
| London | 1 |
| Los Angeles | 1 |
| Luxembourg | 1 |
| Madrid | 1 |
| Milan | 1 |
| Montreal | 1 |
| New York | 1 |
| Paris | 1 |
| Sydney | 1 |
| Taipei | 1 |
| Toronto | 1 |
| Vienna | 1 |
| Athens | 0 |
| Bogota | 0 |
| Bombay | 0 |
| Buenos Aires | 0 |
| Caracas | 0 |
| Hong Kong | 0 |
| Johannesburg | 0 |
| Kuala Lumpur | 0 |
| Lagos | 0 |
| Lisbon | 0 |
| Manila | 0 |
| Mexico City | 0 |
| Nairobi | 0 |
| Nicosia | 0 |
| Panama | 0 |
| Rio de Janeiro | 0 |
| Sao Paulo | 0 |
| Seoul | 0 |
| Singpore | 0 |
| Tel Aviv | 0 |

Using the standardized dataset.

```
#k-means into 3 groups
centroid,label = cluster.vq.kmeans2(Z,k=3)

#centroid of the groups
print(centroid)

#group membership for each instance
print(label) #[1  0  0  0  1  0  0  1  2  1  …]

#graphical representation of groups
plt.plot(M[label==0,0],M[label==0,1],"bo",
        M[label==1,0],M[label==1,1],"ro",
        M[label==2,0],M[label==2,1],"go")
```

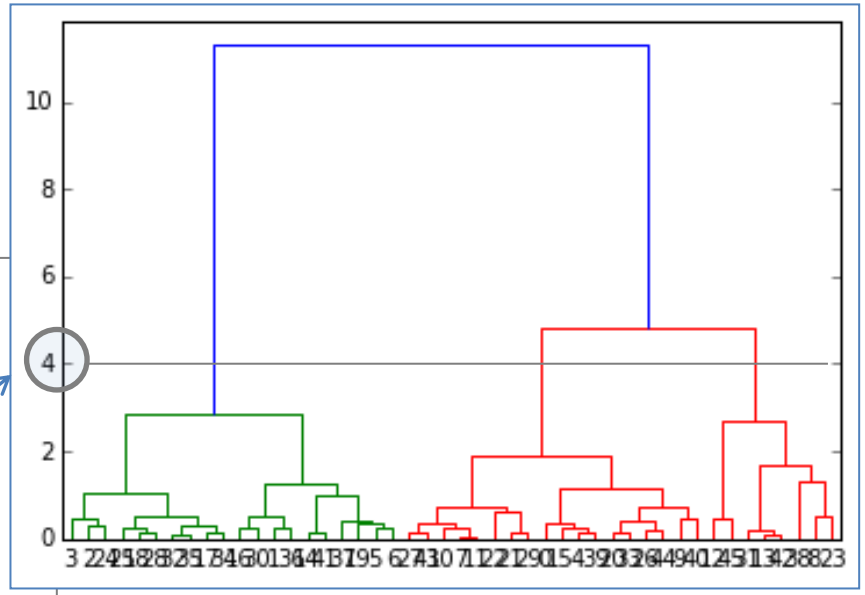
Les centres de classes sont calculées sur les données centrées et réduites initialement.

```
[[-0.91229626 -0.98442173]
 [ 0.33362129  0.57652783]
 [ 1.70101723  1.24777225]]
```

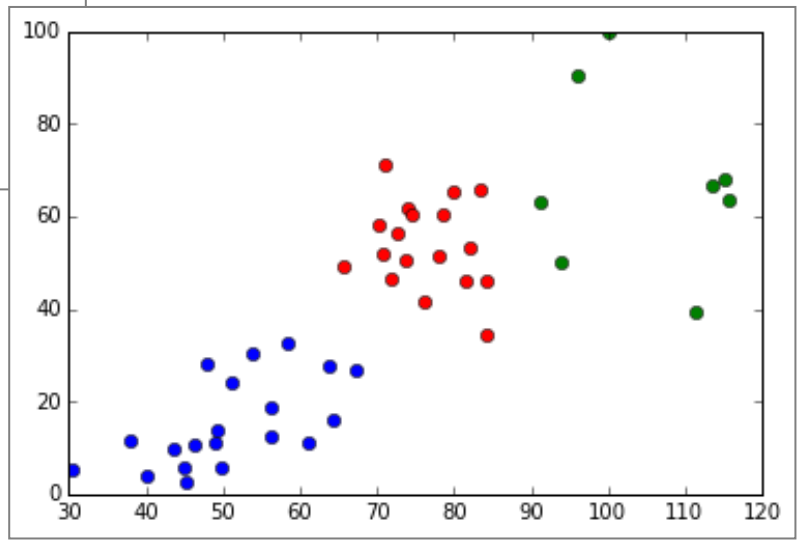


# HAC – Hierarchical Agglomerative Clustering

```
#Ward's method
W = cluster.hierarchy.ward(Z)

#displaying the dendrogram
cluster.hierarchy.dendrogram(W)

#cut the dendrogram to obtain groups
idx = cluster.hierarchy.fcluster(W,t=4,criterion="distance")

#graphical representation again
plt.plot(M[idx==1,0],M[idx==1,1],"bo",
        M[idx==2,0],M[idx==2,1],"ro",
        M[idx==3,0],M[idx==3,1],"go")
```




the cut leads to the creation of 3 groups…




…the same groups (except 1 city)
than the k-means algorithm.

# Références

Course materials (in French)
http://eric.univ-lyon2.fr/~ricco/cours/cours_programmation_python.html

Python website
Welcome to Python - https://www.python.org/
Python 3.4.3 documentation - https://docs.python.org/3/index.html

NumPy Manual
Numpy User Guide and Numpy Reference

SciPy manual
SciPy Reference Guide

POLLS (KDnuggets)
**Data Mining / Analytics Tools Used**
Python, 4th in 2015
**Primary programming language for Analytics, Data Mining, Data Science tasks**
Python, 2nd in 2015 (next R)