Correspondence Analysis & Related Methods

Michael Greenacre

SESSION 3:

MULTIDIMENSIONAL SCALING (MDS)
DIMENSION REDUCTION
CLASSICAL MDS
NONMETRIC MDS

Distances and dissimilarities…

- n objects
- d_{ij} = distance between object i and object j

Properties of a distance (metric)

1. $d_{ij} = d_{ji}$
2. $d_{ij} \geq 0$, $d_{ij} = 0 \Leftrightarrow i = j$
3. $d_{ij} \leq d_{ik} + d_{kj}$ *(the triangle inequality)*

(If 3. not satisfied we often talk of a ***dissimilarity***)

The chi-square distance is a true distance, whereas Bray-Curtis is a dissimilarity

Distances and maps…

CITIES	Amst.	Aths.	Barc.	Basel	Berlin	Bordx	
Amsterdam	0	2979	1533	768	676	1076	...
Athens	2979	0	3261	2594	2486	3250	...
Barcelona	1533	3261	0	1061	1945	600	...
Basel	768	2594	1061	0	884	898	...
Berlin	676	2486	1945	884	0	1631	...
Bordeaux	1076	3250	600	898	1631	0	...
:	:	:	:	:	:	:	

Multidimensional scaling (MDS)

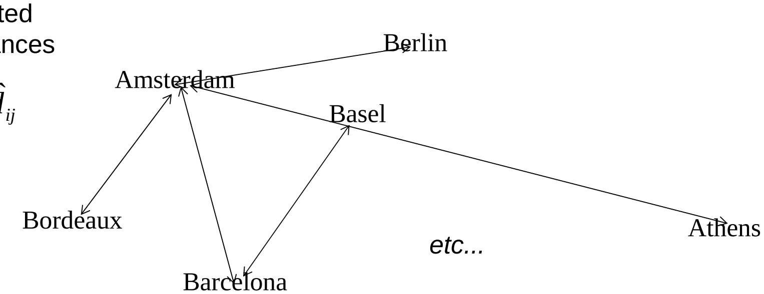
Observed distances

d_{ij}

CITIES	Amst.	Aths.	Barc.	Basel	Berlin	Bordx
Amsterdam	0	d_{12}	d_{13}	d_{14}	d_{15}	d_{16}
Athens	d_{21}	0	d_{23}	d_{24}	d_{25}	d_{26}
Barcelona	d_{31}	d_{32}	0	d_{34}	d_{35}	d_{36}
Basel	d_{41}	d_{42}	d_{43}	0	d_{45}	d_{46}
Berlin	d_{51}	d_{52}	d_{53}	d_{54}	0	d_{56}
Bordeaux	d_{61}	d_{62}	d_{63}	d_{64}	d_{65}	0

Fitted distances

$\hat{d}_{ij}$



Multidimensional scaling (MDS)

Objective is to minimize some measure of discrepancy, or error, between observed and fitted distances.

Observed distances

d_{ij}

Minimize $\sum_{ij}(d_{ij}-\hat{d}_{ij})^2$ also called “Sammon’s non-linear mapping”; R function **`sammon`**

or

Fitted distances

Minimize $\sum_{ij}(f(d_{ij})-\hat{d}_{ij})^2$ for any monotonically increasing function f

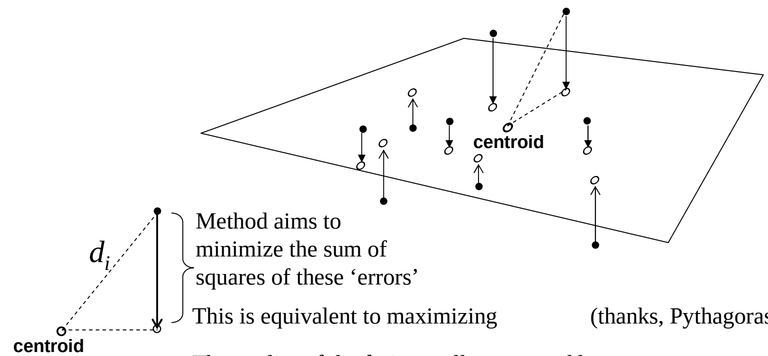
or

Maximize the agreement between the rank-ordered distances in the map and the rank-ordering of the original distances (nonmetric MDS), similar idea to that of Spearman’s rank correlation; R function **`isoMDS`**.

“Classical” MDS

Fits the distances indirectly.

Classical (“YoHoToGo”*) MDS situates the points in a space of as high dimensionality as possible to reproduce the observed distances and then projects the points onto low-dimensional suspaces, usually a plane:



This is equivalent to maximizing (thanks, Pythagoras!).

The quality of the fit is usually measured by expressed as a %.

*YoHoToGo = Young-Householder-Torgerson-Gower

R function **`cmdscale`**

Metric and nonmetric MDS

These methods fit the interpoint distances directly

Stress: measures the discrepancy between the observed distances (data) and the fitted distances (map)

Observed distances

d_{ij}

Fitted distances

$\hat{d}_{ij}$

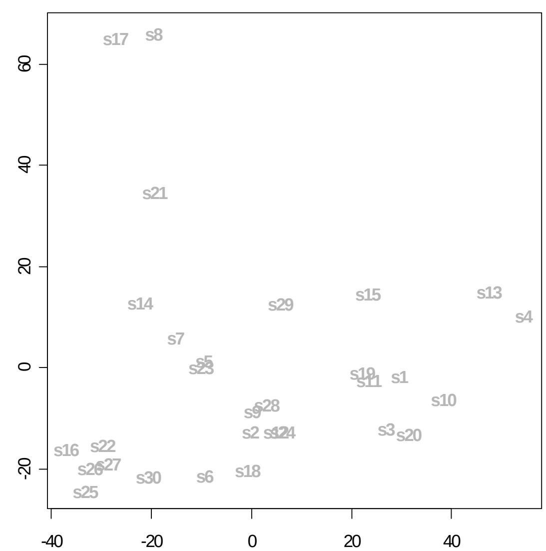
$$\text{Raw stress}: \sum_{ij}(d_{ij}-\hat{d}_{ij})^2$$

$$\text{Normalized stress}: \frac{\sum_{ij}(d_{ij}-\hat{d}_{ij})^2}{\sum_{ij} d_{ij}^2}$$

$$\text{Kruskal stress}: \sqrt{\frac{\sum_{ij}(d_{ij}-\hat{d}_{ij})^2}{\sum_{ij} \hat{d}_{ij}^2}}$$

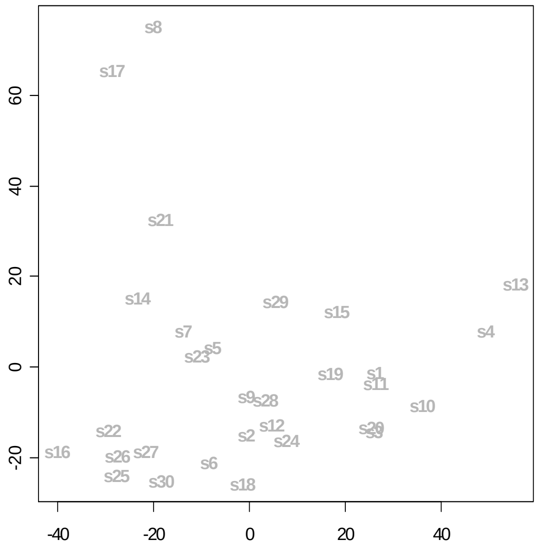
used in R function **`isoMDS`** for nonmetric MDS; can be thought of as a percentage error

MDS of Bray-Curtis dissimilarities - classical



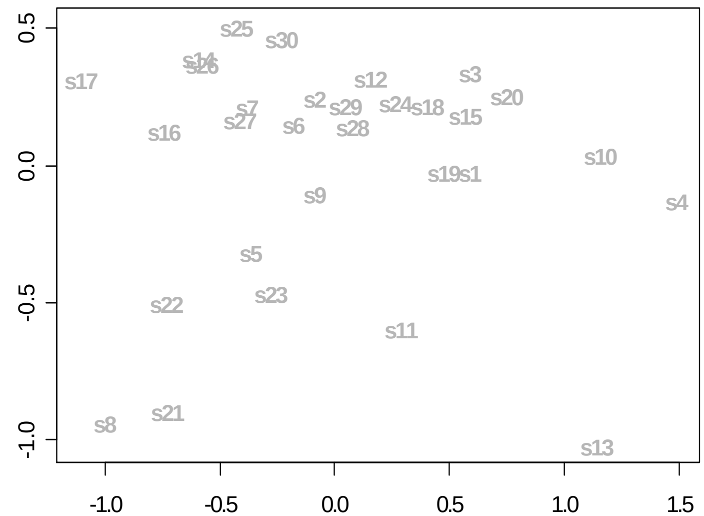
Goodness of fit: 53.1%

MDS of Bray-Curtis dissimilarities – nonmetric



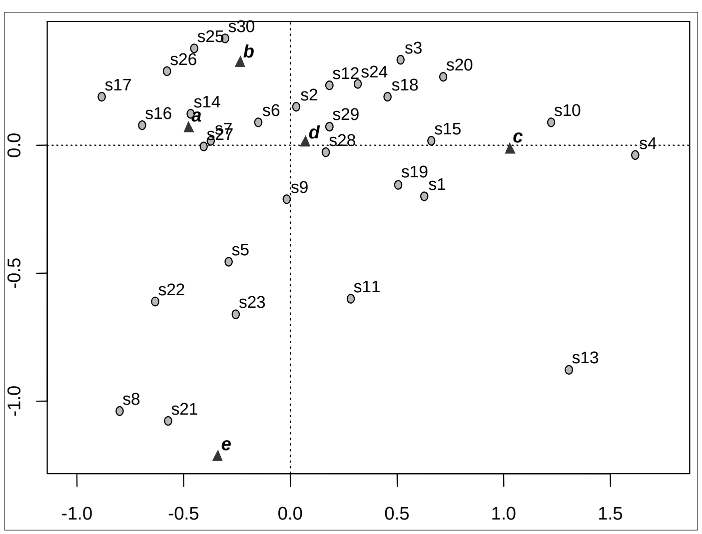
Stress:
13.5%

MDS of chi-square distances – classical



Goodness
of fit:
74.4%

Correspondence analysis



Notice that the rows and the columns are depicted in a joint map.

To be continued...

Goodness
of fit:
75.2%

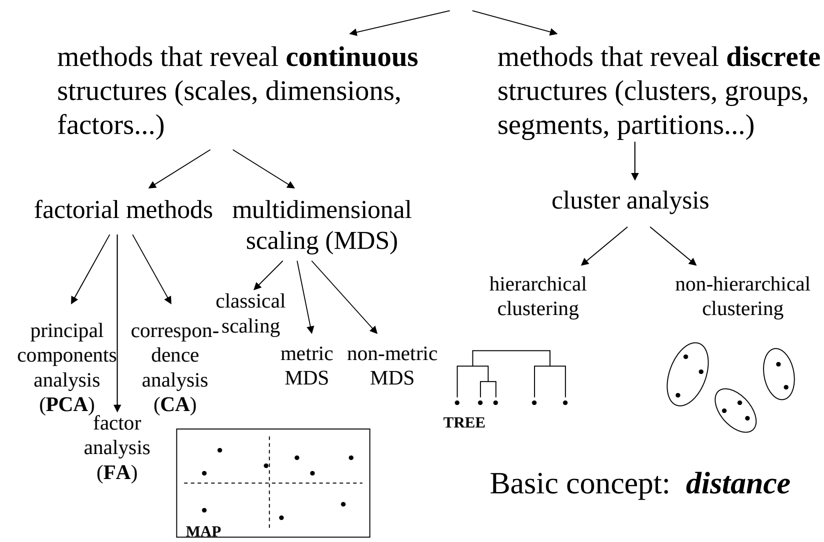
Correspondence Analysis & Related Methods

Michael Greenacre

SESSION 4:

CLASSICAL MDS – the computations

In this course we concentrate on the **STRUCTURAL** methods of multivariate analysis

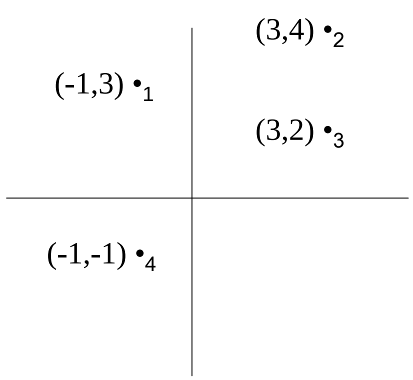


Basic concept: ***distance***

Classical scaling

- From a map to a distance matrix

points → distances



(squared) distance matrix

$$\begin{bmatrix} 0 & 17 & 17 & 16 \\ 17 & 0 & 4 & 41 \\ 16 & 4 & 0 & 25 \\ 16 & 41 & 25 & 0 \end{bmatrix}$$

Classical scaling

points → distances

- suppose you have n points $\mathbf{x}_i$ (i=1,...,n) in p -dimensional Euclidean space

$$\mathbf{X} = \begin{bmatrix} \mathbf{x}_1^{\mathsf{T}} \\ \mathbf{x}_2^{\mathsf{T}} \\ \vdots \\ \mathbf{x}_n^{\mathsf{T}} \end{bmatrix} = \overbrace{\begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1p} \\ x_{21} & x_{22} & \cdots & x_{2p} \\ \vdots & \vdots & \vdots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{np} \end{bmatrix}}^{p \text{ dimensions}} \quad n \text{ points}$$

- squared distance between the i-th and j-th points is

$$\delta_{ij} = \sum_{k=1}^{p} (x_{ik} - x_{jk})^2$$

(squared) distance matrix $$\boldsymbol{\Delta} = \begin{bmatrix} \delta_{11} & \delta_{12} & \cdots & \delta_{1n} \\ \delta_{21} & \delta_{22} & \cdots & \delta_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ \delta_{n1} & \delta_{n2} & \cdots & \delta_{nn} \end{bmatrix}$$

Classical scaling

- in matrix notation:

$$\boldsymbol{\Delta} = \begin{bmatrix} \delta_{11} & \delta_{12} & \cdots & \delta_{1n} \\ \delta_{21} & \delta_{22} & \cdots & \delta_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ \delta_{n1} & \delta_{n2} & \cdots & \delta_{nn} \end{bmatrix} = \mathbf{s}\mathbf{1}^\top + \mathbf{1}\mathbf{s}^\top - 2\mathbf{S}$$

where $\mathbf{S} = \mathbf{X}\mathbf{X}^\top$ and $\mathbf{s} = \mathrm{diag}(\mathbf{S})$ is matrix of scalar products

- the problem in classical scaling:

distances $\longrightarrow$ points

- given $\boldsymbol{\Delta}$ solve for $\mathbf{X}$

Classical scaling

- if we had $\mathbf{S}$ and had to recover $\mathbf{X}$ it would be simple:

$$\mathbf{S} = \mathbf{X}\mathbf{X}^\top$$

- recall the eigenvalue-eigenvector decomposition of a square symmetric matrix, for example of $\mathbf{S}$:

$$\mathbf{S} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^\top$$

where

$$\mathbf{U}\mathbf{U}^\top = \mathbf{I}\ ; \quad \boldsymbol{\Lambda} = \begin{bmatrix} \lambda_1 & 0 & \cdots & 0 \\ 0 & \lambda_2 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & \lambda_n \end{bmatrix} \qquad \lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n \geq 0$$

so a possible solution would be:

$$\mathbf{X} = \mathbf{U}\boldsymbol{\Lambda}^{1/2}$$

Classical scaling

- but we don't have the scalar products $\mathbf{S}$ but rather the squared distances $\boldsymbol{\Delta} = \mathbf{s}\mathbf{1}^\top + \mathbf{1}\mathbf{s}^\top - 2\mathbf{S}$
- we can recover the matrix of scalar products $\mathbf{S^*}$ with respect to the centroid of the n points by a transformation of $\boldsymbol{\Delta}$ called double-centring:

– subtract the row means from all the squared distances

– subtract column means from the resultant matrix

then multiply double-centred matrix by -1/2 to obtain $\mathbf{S^*}$

Then carry on as before:

$$\mathbf{S^*} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^\top$$
$$\mathbf{X^*} = \mathbf{U}\boldsymbol{\Lambda}^{1/2}$$

R code to double-centre and eigendecompose

```
# read in the squared distance matrix
d2   <- matrix(c(0,17,17,16,17,0,4,41,17,4,0,25,16,41,25,0),nrow=4)

# compute scalar products
n    <- nrow(d2)
ones <- rep(1,n)
I    <- diag(ones)
Sd   <- -0.5*(I-(1/n)*ones%*%t(ones)) %*% d2 %*% (I-(1/n)*ones%*%t(ones))

# compute eigenvalues and eigenvectors using R function eigen
Sd.eig <- eigen(Sd)

# compute coordinates and plot
X    <- Sd.eig$vectors[,1:2] %*% diag(sqrt(Sd.eig$values[1:2]))
plot(X, type="n")
text(X, labels=1:4)
```