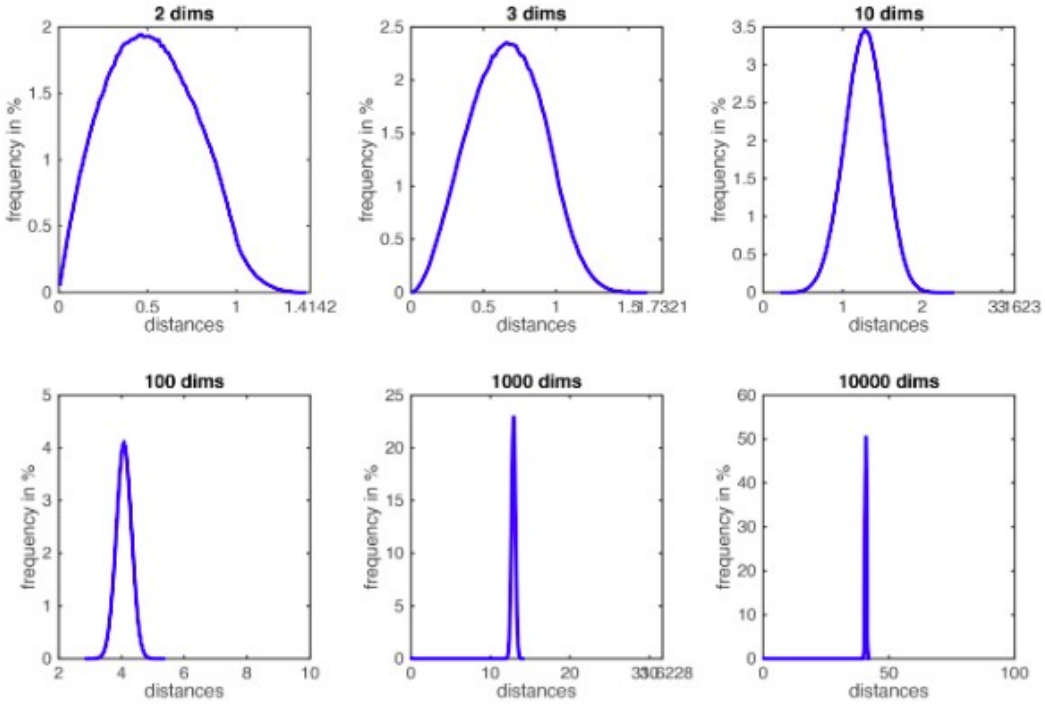
Multidimensional Scaling (MDS)

Tran Thi Hong Minh

Contents

Dimensionality Reduction

- Real data usually have huge number of dimensions. This will cause problems:
  - Data become **very sparse**, some algorithms become meaningless (e.g. density based clustering)
  - Several algorithms depend on the dimensionality and they become infeasible (e.g. k-nearest neighbor)



Dimensionality Reduction

- Usually the data can be described with fewer dimensions, **without losing much of the meaning** of the data.
- Why reduces dimensions?
  - Discover hidden correlations
  - Remove redundant and noisy features
  - Interpretation and visualization
  - Easier storage and processing of the data

What is MDS?

“Multidimensional scaling (MDS) is a technique for the analysis of **similarity** or **dissimilarity** data on a set of objects. MDS attempts to model such data **as distances among points** in a geometric space. The main reason for doing this is that one wants a graphical display of the structure of the data, one that is much easier to understand than an array of numbers and, moreover, one that **displays the essential information** in the data, **smoothing out noise**” (Borg and Groenen, 2005).

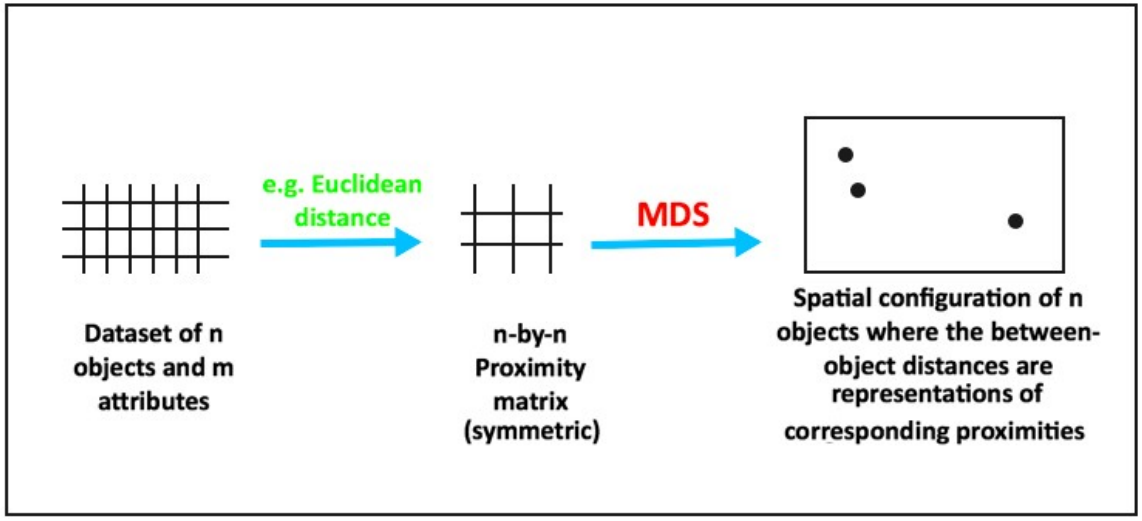


Figure2- From raw data to MDS configureation

A **proximity** is a number which indicates how similar or how different two objects are.

What is MDS?

- The starting point is a matrix of distances or **dissimilarities** between the data points, **D**.
- For example, if we have data on n different experimental units, then we would be given the distances d_{ij} between any pair of experimental units i and j . We compile these into a n × n **distance matrix D** = (d_{ij} : i , j = 1 , … , n) .
- The goal of MDS is to **find a set of points in a low-dimensional Euclidean space** R^r, usually R or R^2 or R^3, whose **inter-point distances are as close as possible to the d_{ij}**. That is, we want to find $\mathbf{y_1, \dots, y_n} \in \mathbf{R^r}$ whose distance matrix is approximately **D**, i.e., for which

$$\text{distance}(\mathbf{y}_i, \mathbf{y}_j) \approx d_{ij}.$$

Y: coordinate matrix

In other words

$$\begin{aligned} \text{Find} \quad & \mathbf{y}_1, \dots, \mathbf{y}_k \in \mathbb{R}^r \\ \text{to minimize} \quad & \sum_{i=1}^{n} \sum_{j=1}^{n} (d_{ij} - d(\mathbf{y}_i, \mathbf{y}_j))^2. \end{aligned}$$

Example. The distance matrix between pairs of cities D :

```
> real_dist
        Boston   DC   NY   LA   SC   SF
Boston       0  439  215 2979 2976 3095
DC         439    0  233 2631 2684 2799
NY         215  233    0 2786 2815 2934
LA        2979 2631 2786    0  384  382
SC        2976 2684 2815  384    0   87
SF        3095 2799 2934  382   87    0
```

MDS constructs a set of points $y_1, \ldots, y_n$, that have distances between them given by the distance matrix D. In other words, it creates a map with **a set of coordinates** for which the distances between points are **approximately the same as in the real data**.

Definition 6.1 The $n \times n$ matrix $\mathbf{D} = (d_{ij})_{i,j=1}^{n}$ is a **distance matrix** (sometimes called a **dissimilarity matrix**) if

1. $d_{ij} \geq 0$ for all $i, j = 1, \ldots, n$.
2. $d_{ii} = 0$ for $i = 1, \ldots, n$ and
3. $\mathbf{D} = \mathbf{D}^{\top}$, i.e., $\mathbf{D}$ is symmetric ($d_{ij} = d_{ji}$).

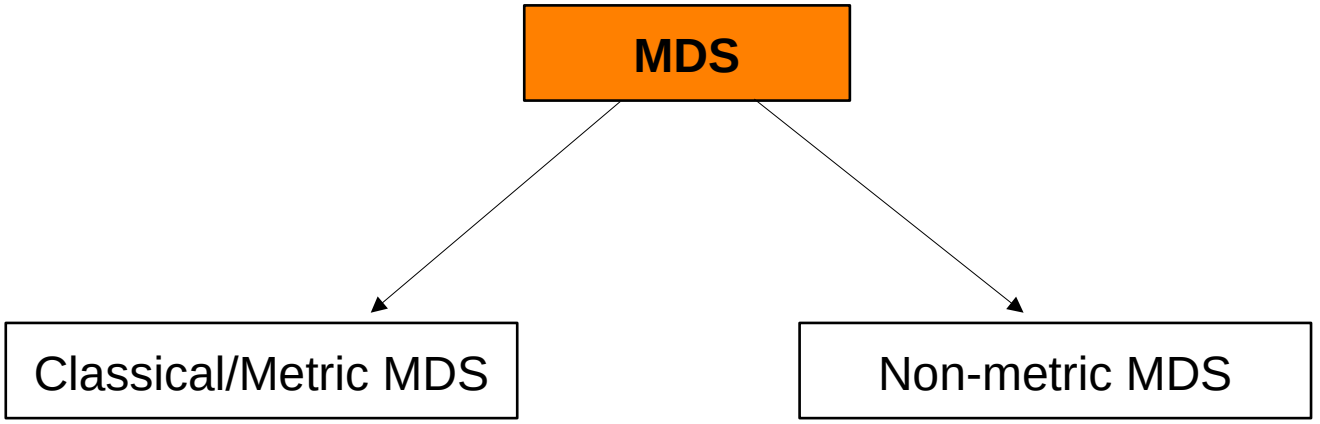
Note that we do not require distances to necessarily satisfy the triangle inequality

$$d_{ik} \leq d_{ij} + d_{jk}. \tag{6.2}$$

A distance function which always satisfies the triangle inequality is called a **metric distance** or just a **metric**. A distance function which does not always satisfy the triangle inequality is called a **non-metric** distance.

There are many ways to measure distance. Two common choices of metric distances are

- Euclidean distance $||\mathbf{x} - \mathbf{x}'||_2$, also know as the 'crow flies' distance.
- L_1 distance $||\mathbf{x} - \mathbf{x}'||_1$, sometimes called the Manhattan or taxicab metric.

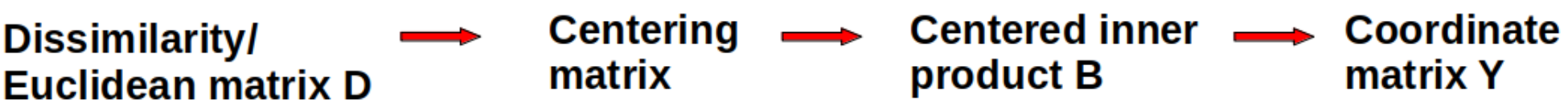

MDS
Classical/Metric MDS
Non-metric MDS

Classical MDS

- Euclidean distances are usually the first choice for a classical MDS. Since Euclidean distances satisfy the triangle inequality, it follows that Euclidean distance is a metric distance.
- The following steps summarize the algorithm of classical MDS:

1) Compute nxn **distance matrix D**
2) Calculate the **centered inner-product matrix B**
3) Output **coordinate matrix Y** by **eigenvalue decomposition** from B

} Package R: cmdscale



Classical MDS

1) Compute nxn distance matrix **D**

Definition 6.2 Suppose $\mathbf{D}$ is an $n \times n$ distance matrix. We say $\mathbf{D}$ is a **Euclidean distance matrix** if there is a set of points $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^p$ for some p, such that

$$d_{ij} = ||\mathbf{x}_i - \mathbf{x}_j||_2 = \sqrt{(\mathbf{x}_i - \mathbf{x}_j)^\top (\mathbf{x}_i - \mathbf{x}_j)}.$$

Proximity matrix

```
> real_dist
       Boston   DC   NY   LA   SC   SF
Boston      0  439  215 2979 2976 3095
DC        439    0  233 2631 2684 2799
NY        215  233    0 2786 2815 2934
LA       2979 2631 2786    0  384  382
SC       2976 2684 2815  384    0   87
SF       3095 2799 2934  382   87    0
```

Euclidean distance matrix/ Dissimilarity matrix

```
> D <- as.matrix(dist(real_dist, upper=T, diag=T))
> D
          Boston        DC        NY        LA        SC        SF
Boston    0.0000  824.4695  473.2621 6575.9121 6730.1758 6871.0509
DC      824.4695    0.0000  467.0814 6160.0016 6317.8128 6466.9119
NY      473.2621  467.0814    0.0000 6429.3201 6584.3743 6731.7656
LA     6575.9121 6160.0016 6429.3201    0.0000  620.9638  666.0638
SC     6730.1758 6317.8128 6584.3743  620.9638    0.0000  238.0945
SF     6871.0509 6466.9119 6731.7656  666.0638  238.0945    0.0000
```

Classical MDS

2) Calculate the centered inner-product matrix B

Definition 6.3 Given a distance matrix $\mathbf{D} = \{d_{ij}\}_{i,j=1}^{n}$, the **centred inner-product matrix** (also called the **centred-Gram matrix**) is

$$\mathbf{B} = \mathbf{HAH}, \tag{6.3}$$

where $\mathbf{A}$ is the matrix of negative square distances divided by two:

$$\mathbf{A} = \{a_{ij}\}_{i,j=1}^{n}, \quad \text{where} \quad a_{ij} = -\frac{1}{2}d_{ij}^2 \tag{6.4}$$

H is the n × n centering matrix: $H = I_n - n^{-1}J_n$

d_{ij}^2

$$a_{ij} = -\frac{1}{2} d_{ij}^2$$

```
> D^2
        Boston       DC       NY       LA       SC       SF
Boston       0   679750   223977 43242620 45295266 47211341
DC      679750        0   218165 37945620 39914758 41820949
NY      223977   218165        0 41336157 43353985 45316668
LA    43242620 37945620 41336157        0   385596   443641
SC    45295266 39914758 43353985   385596        0    56689
SF    47211341 41820949 45316668   443641    56689        0
```

```
> A <- -D^2/2
> A
           Boston          DC          NY          LA          SC          SF
Boston        0.0   -339875.0   -111988.5 -21621310.0 -22647633.0 -23605670.5
DC       -339875.0         0.0   -109082.5 -18972810.0 -19957379.0 -20910474.5
NY       -111988.5   -109082.5         0.0 -20668078.5 -21676992.5 -22658334.0
LA     -21621310.0 -18972810.0 -20668078.5         0.0   -192798.0   -221820.5
SC     -22647633.0 -19957379.0 -21676992.5   -192798.0         0.0    -28344.5
SF     -23605670.5 -20910474.5 -22658334.0   -221820.5    -28344.5         0.0
```

$H = I_n - n^{-1} J_n$

I_n

1	0	0	0	0	0
0	1	0	0	0	0
0	0	1	0	0	0
0	0	0	1	0	0
0	0	0	0	1	0
0	0	0	0	0	1

- 1/6 *

1	1	1	1	1	1
1	1	1	1	1	1
1	1	1	1	1	1
1	1	1	1	1	1
1	1	1	1	1	1
1	1	1	1	1	1

$$H = I_n - n^{-1}J_n$$

```
> H <- diag(6) - 1/6 * matrix(rep(1,6), nc=1)%*%matrix(rep(1,6), nr=1)
> H
          [,1]       [,2]       [,3]       [,4]       [,5]       [,6]
[1,]  0.8333333 -0.1666667 -0.1666667 -0.1666667 -0.1666667 -0.1666667
[2,] -0.1666667  0.8333333 -0.1666667 -0.1666667 -0.1666667 -0.1666667
[3,] -0.1666667 -0.1666667  0.8333333 -0.1666667 -0.1666667 -0.1666667
[4,] -0.1666667 -0.1666667 -0.1666667  0.8333333 -0.1666667 -0.1666667
[5,] -0.1666667 -0.1666667 -0.1666667 -0.1666667  0.8333333 -0.1666667
[6,] -0.1666667 -0.1666667 -0.1666667 -0.1666667 -0.1666667  0.8333333
```

```
> B <- H%*%A%*%H
> B
          [,1]      [,2]      [,3]      [,4]      [,5]      [,6]
[1,]  12013126  10333775  11384138 -10716460 -11271728 -11742850
[2,]  10333775   9334174  10047568  -9407436  -9920950 -10387130
[3,]  11384138  10047567  10979126 -10280229 -10818088 -11312514
[4,] -10716461  -9407436 -10280229   9796573  10074830  10532724
[5,] -11271728  -9920951 -10818088  10074830  10738683  11197254
[6,] -11742850 -10387130 -11312513  10532724  11197254  11712515
```

B the centered inner-product matrix. The inner product is **large if x_i is similar to x_j**, and **small if they are very different**

Classical MDS

3) Output coordinate matrix Y by **eigenvalue decomposition** from B

Suppose B has eigenvalues $\lambda 1 \geq \lambda 2 \cdots \geq \lambda k > 0$ and that it has **spectral decomposition**

$B = U \Lambda U^T$

$\Lambda = \text{diag}\{\lambda 1 \ldots \lambda k\}$ and U is the nxk matrix of k eigenvalues

(Keep only **few largest eigenvalues k** and corresponding eigenvectors)

Then **coordinate matrix Y**:

$$Y = [y1, \ldots, yn]^T = U\Lambda^{1/2}$$

```
> eigen(B)
eigen() decomposition
$values
[1]  6.404091e+07  2.786842e+05  2.111650e+05  3.172240e+04  1.171405e+04
[6] -6.510788e-10

$vectors
           [,1]        [,2]        [,3]        [,4]        [,5]       [,6]
[1,] -0.4306048 -0.61785770 -0.37012528  0.29981747  0.19819542 -0.4082483
[2,] -0.3791760  0.57808496  0.35381850  0.47247157  0.08342529 -0.4082483
[3,] -0.4136593  0.06000719  0.06253239 -0.75244055 -0.29755897 -0.4082483
[4,]  0.3879955  0.41994539 -0.71064324 -0.02756709  0.02578586 -0.4082483
[5,]  0.4085944 -0.15994217  0.39612371 -0.24101166  0.65253496 -0.4082483
[6,]  0.4268503 -0.28023766  0.26829393  0.24873026 -0.66238257 -0.4082483
```

u1 u2

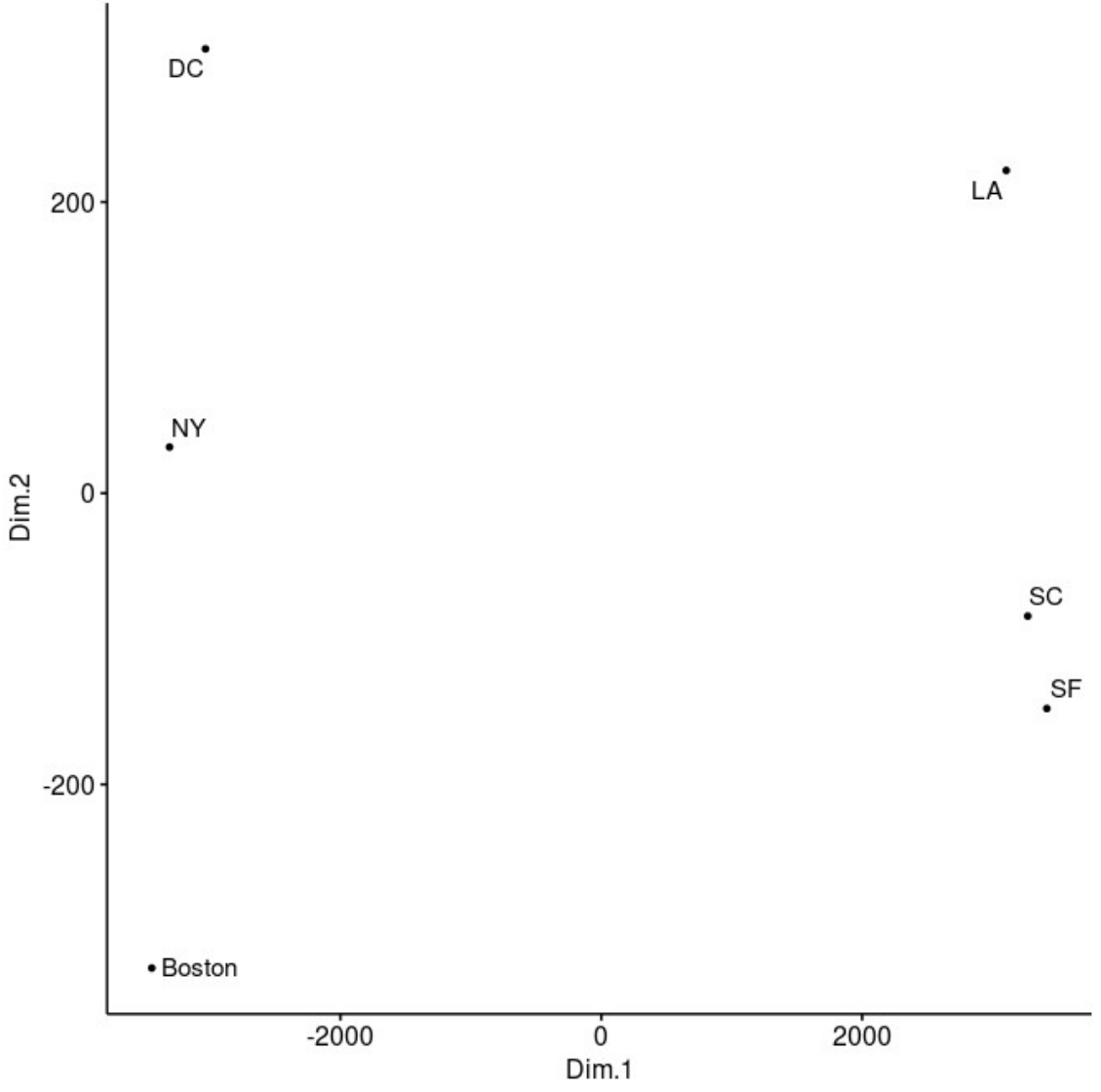
```
          [,1]     [,2]
[1,] 8002.557   0.0000
[2,]    0.000 527.9054
```

sqrt(λ1)	0
0	sqrt(λ2)

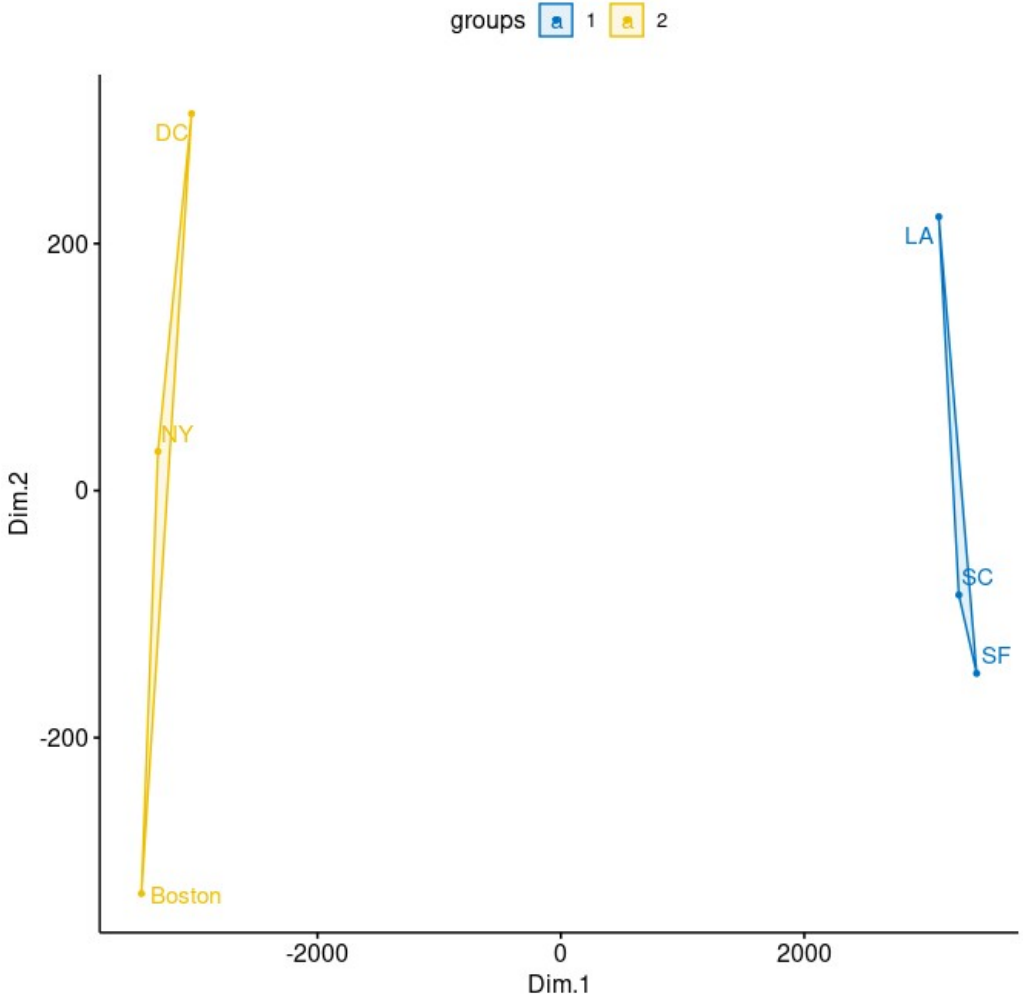
$$Y = U\Lambda^{1/2}$$

```
> Y
          [,1]       [,2]
[1,] -3445.940 -326.17044
[2,] -3034.377  305.17419
[3,] -3310.332   31.67812
[4,]  3104.956  221.69146
[5,]  3269.799  -84.43434
[6,]  3415.893 -147.93899
```

	[,1]	[,2]
Boston	-3445.940	-326.17044
DC	-3034.377	305.17419
NY	-3310.332	31.67812
LA	3104.956	221.69146
SC	3269.799	-84.43434
SF	3415.893	-147.93899



Cluster by k-means



Classical MDS: Goodness-of-Fit

```
> Y.mds
$points
           [,1]       [,2]
Boston -3445.940  326.17044
DC     -3034.377 -305.17419
NY     -3310.332  -31.67812
LA      3104.956 -221.69146
SC      3269.799   84.43434
SF      3415.893  147.93899

$eig
[1]  6.404091e+07  2.786842e+05  2.111650e+05  3.172240e+04  1.171405e+04
[6] -2.229196e-09

$x
NULL

$ac
[1] 0

$GOF
[1] 0.9960572 0.9960572
```

Classical MDS: Goodness-of-Fit

P_2 value

- P2 value is computed as the sum of the eigenvalues for the number of dimensions divided by the total sum of the eigenvalues.
- The eigenvalue is a measure of the generalized variance, so **P2 indicates the amount of generalized variance explained by the dimensions out of the total variance** in the matrix.
- The P2 values range from 0 to 1 with values closer to 1 indicating a better fit.
- If there are two dimensions, the P2 value is computed as follows:

$$P_2 = \frac{\Sigma(\lambda_1 + \lambda_2)}{\Sigma\lambda_N}.$$

Classical MDS: Goodness-of-Fit

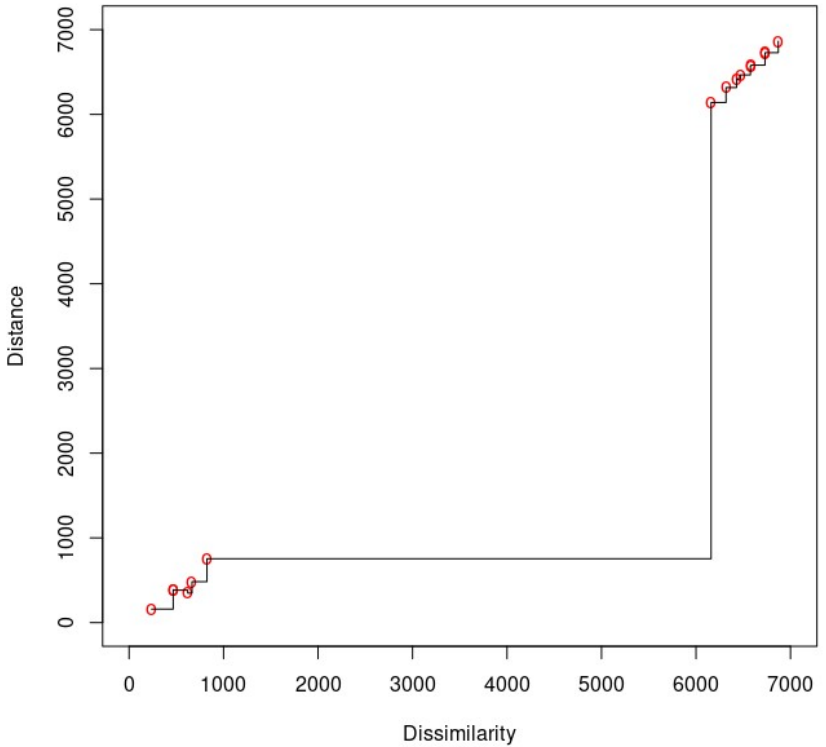
Mardia criterion

- Mardia criterion is a variation on P2.
- Mardia values range from 0 to 1 with values closer to 1 indicating a better model fit.
- The Mardia value for a two dimension solution would be computed as follows:

$$\text{Mardia} = \frac{\Sigma(\lambda_1^2 + \lambda_2^2)}{\Sigma\lambda_N^2}.$$

Classical MDS: Shepard diagram

Show the connection between distance from coordinate matrix Y and Euclidean distance (dissimilarity)



The coordinate values for each city do not fall far from the line → a good model fit

```
> D <- as.matrix(dist(real_dist, upper=T, diag=T))
> D
          Boston        DC        NY        LA        SC        SF
Boston    0.0000  824.4695  473.2621 6575.9121 6730.1758 6871.0509
DC      824.4695    0.0000  467.0814 6160.0016 6317.8128 6466.9119
NY      473.2621  467.0814    0.0000 6429.3201 6584.3743 6731.7656
LA     6575.9121 6160.0016 6429.3201    0.0000  620.9638  666.0638
SC     6730.1758 6317.8128 6584.3743  620.9638    0.0000  238.0945
SF     6871.0509 6466.9119 6731.7656  666.0638  238.0945    0.0000
```

```
> signif(dist(Y, upper=T, diag=T))
         1        2        3        4        5        6
1    0.000  753.645  382.681 6573.770 6720.090 6864.150
2  753.645    0.000  388.524 6139.900 6316.200 6466.170
3  382.681  388.524    0.000 6418.100 6581.160 6728.620
4 6573.770 6139.900 6418.100    0.000  347.687  483.020
5 6720.090 6316.200 6581.160  347.687    0.000  159.299
6 6864.150 6466.170 6728.620  483.020  159.299    0.000
```

$$\text{distance}(\mathbf{y}_i, \mathbf{y}_j) \approx d_{ij}.$$

Classical MDS: some explanations

Dissimilarity/ Euclidean matrix D → **Centering matrix** → **Centered inner product B** → **Coordinate matrix Y**

Similarity between vector a (SC) and b (SF)

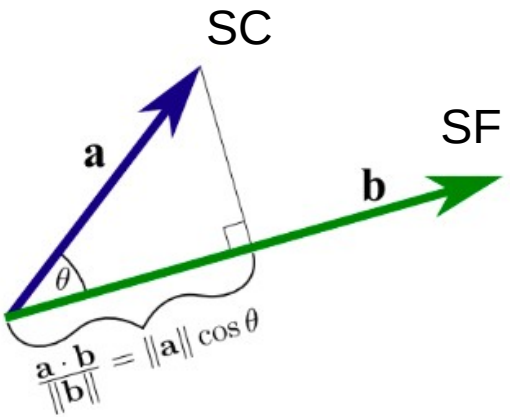
a = [a1 a2 … an], b = [b1 b2 … bn]

a.b = a1b1 + a2b2 +...+ anbn = Σaibi

This dot product a·b should depend on the magnitude of both vectors, ‖a‖ and ‖b‖.

It reflects both magnitude and direction

=> The inner product is **large if a is similar to b**, and **small if they are very different**



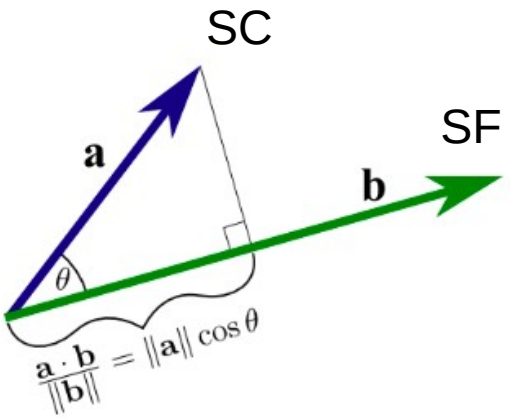
Classical MDS: some explanations

Dissimilarity/ Euclidean matrix D → **Centering matrix** → **Centered inner product B** → **Coordinate matrix Y**

Euclidean distance between a and b

$\Sigma(ai - bi)^2 = \Sigma(ai^2 + bi^2 - 2aibi) = \Sigma(ai)^2 + \Sigma(bi)^2 - 2\Sigma(aibi)$ (1)

Inner product



Replace coordinate matrix Y to (1) (Y preserves the real distance)

Y

SC/a	3269.799 -84.43434
SF/b	3415.893 -147.93899

Classical MDS: some explanations

Dissimilarity/ Euclidean matrix D → **Centering matrix** → **Centered inner product B** → **Coordinate matrix Y**

$D^2(Y) = \Sigma$

y^2_a	y^2_a
y^2_b	y^2_b

$+ \Sigma$

y^2_a	y^2_b
y^2_a	y^2_b

$- 2\Sigma$

$y_{a1}y_{a1} + y_{a2}y_{b1}$	$y_{a1}y_{a2} + y_{a2}y_{b2}$
$y_{b1}y_{a1} + y_{b2}y_{b1}$	$y_{b1}y_{a2} + y_{b1}y_{b2}$

$= c1' + 1c' - 2YY'$ $= c1' + 1c' - 2B$

Y

SC/a
SF/b

3269.799	-84.43434
3415.893	-147.93899

x

Y'

3259.799	3415.893
-84.43434	-147.93899

=

10738683	11197254
11197254	11712515

Classical MDS: some explanations

$H = I_n - n^{-1}J_n \quad = I_n - n^{-1}11'$

$B = HAH = -1/2^* H D^2 H \quad = -1/2^* H (c1' + 1c' - 2B) H$

$= HBH = B$

centering a vector of ones yields a vector of zeros

The centering around B can be removed because B is centered

Classical MDS: some explanations

$$Bu_1 = \lambda_1 u_1$$
$$Bu_2 = \lambda_2 u_2$$

$$B \begin{bmatrix} | & | \\ u1 & u2 \\ | & | \end{bmatrix} = \begin{bmatrix} | & | \\ u1 & u2 \\ | & | \end{bmatrix} \begin{bmatrix} \lambda 1 & 0 \\ 0 & \lambda 2 \end{bmatrix}$$

$$BU = U\Lambda$$

$$B = U\Lambda U^{-1}$$

$$U\Lambda U^{-1} = (U\Lambda^{1/2})(U\Lambda^{1/2})' = B = YY'$$

Non-Metric MDS

The assumption that proximities behave like distances might be too restrictive.

Metric vs. Non-metric

Proximity Matrix: Assign a numerical value to indicate the distance between pairs of objects

- Real distances between cities (metric)
- Rank of cities (non-metric)

Non-Metric MDS

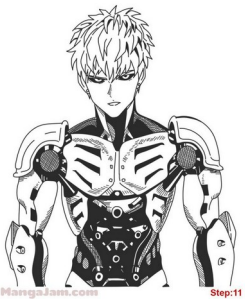
The non-metric MDS approach uses ordinal data

For example, how do you rank these pictures from simple to complex drawing?

1: very simple → 10: very complex

2

<



6

<

10

Non-metric MDS finds a low-dimensional representation, which respects the ranking of distances

Non-Metric MDS: Stress

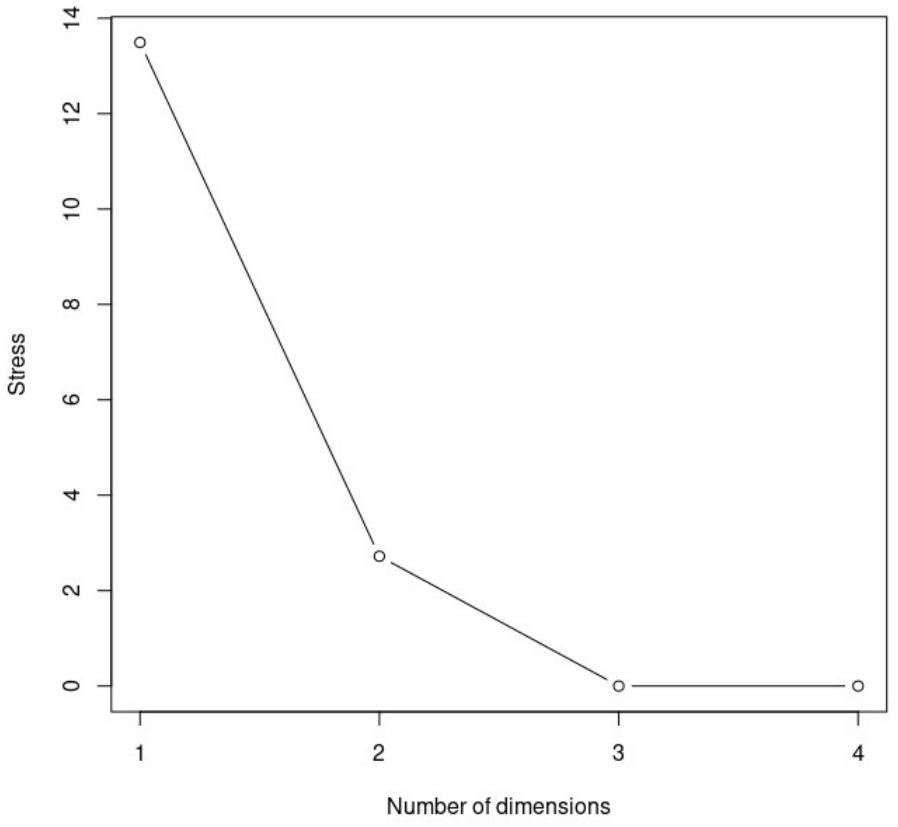
$$\text{STRESS} = \sqrt{\frac{\sum (f(p) - d)^2}{\sum d^2}}.$$

p is the proximities

d is the point distances

f(p) is scaled proximities

Goal: minimizes the squared differences between the optimally scaled proximities and the distances between the points



Non-Metric MDS: Stress

Judging the goodness of fit

The amount of stress may also be used for judging the goodness of fit of an MDS solution: a small stress value indicates a good fitting solution, whereas a high value indicates a bad fit. Kruskal (1964a) provided some guidelines for the interpretation of the stress value with respect to the goodness of fit of the solution (Table 2).

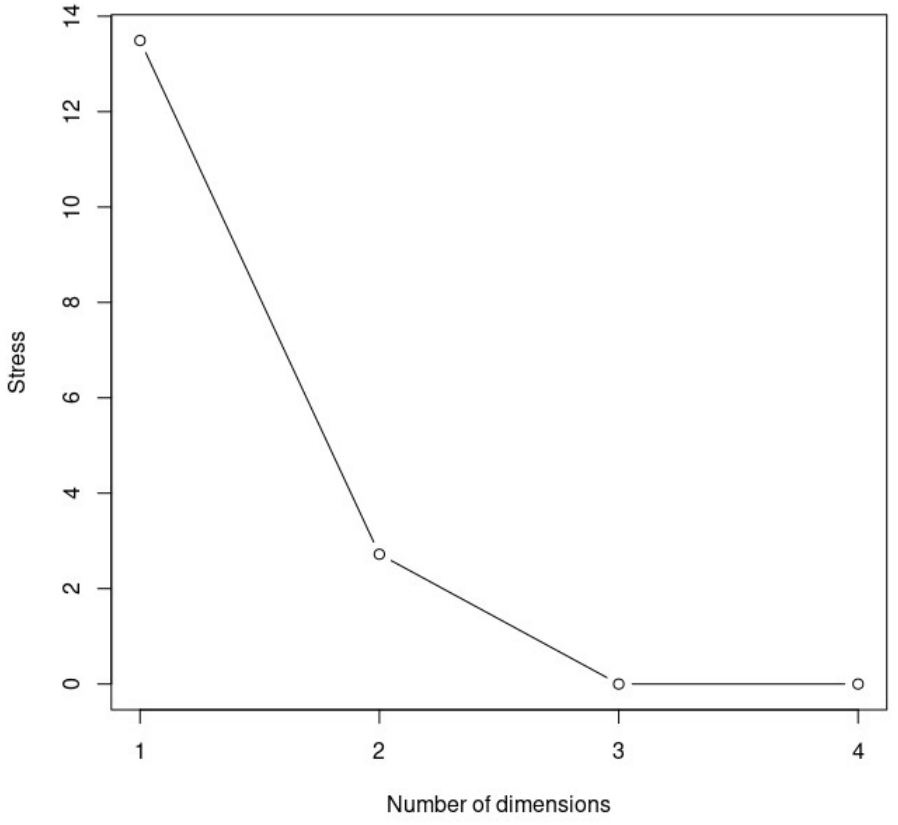
Stress	Goodness of fit
$> .20$	poor
.10	fair
.05	good
.025	excellent
.00	perfect

Table 2: Stress and goodness of fit.

Stress decreases as the number of dimensions increases. Thus, a two-dimensional solution always has more stress than a three-dimensional one

Non-Metric MDS: Scree plot

In a scree plot, the amount of stress is plotted against the number of dimensions. Since stress decreases with increasing dimensionality, one is looking for the lowest number of dimensions with acceptable stress.



Non-Metric MDS

The following steps summarize the algorithm of non-metric MDS:

1) Compute **correlation matrix**

2) Compute nxn **distance matrix D**

3) Calculate the **centered inner-product matrix B**

4) Output **coordinate matrix Y** by **eigenvalue decomposition** from B

} isoMDS

5) Compare the stress to some criterion. If the stress is small enough then exit the algorithm else choose another number of dimensions and run again

Example

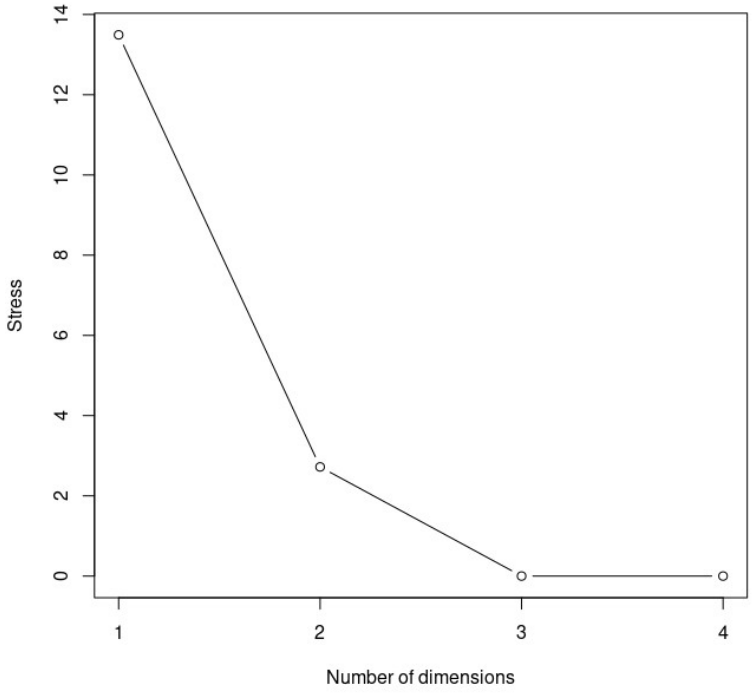
```
> dat
  Sample Species.1 Species.2 Species.3 Species.4 Species.5 Species.6
1      1        46        72         7        13         9       142
2      2        51       124         5        16         3        49
3      3        23        44         3         6         2        16
4      4        12        41         1        18         4        21
5      5         4        40         0        19         6        10
6      6        23        16         2        13         0         6
7      7        46        33        19        27        11        31
```

The similarity between two species means that they account for the same proportion in each sample group

```
## correlation matrix
dat_cor <- cor(dat)
> dat_cor
          Species.1     Species.2    Species.3  Species.4  Species.5  Species.6
Species.1 1.0000000  0.5971946598 0.6863907019  0.1461789 0.38248094  0.6076910
Species.2 0.5971947  1.0000000000 0.0002063497 -0.1006733 0.02996717  0.4787747
Species.3 0.6863907  0.0002063497 1.0000000000  0.6093863 0.73349359  0.2457174
Species.4 0.1461789 -0.1006732675 0.6093862668  1.0000000 0.65675867 -0.1033419
Species.5 0.3824809  0.0299671699 0.7334935872  0.6567587 1.00000000  0.5077296
Species.6 0.6076910  0.4787747226 0.2457173701 -0.1033419 0.50772964  1.0000000
```

```
dist_nmds <- cor2dist(dat_cor)
> dist_nmds
          Species.1 Species.2 Species.3 Species.4 Species.5 Species.6
Species.1 0.0000000 0.8975582 0.7919713 1.3067679 1.1113227 0.8857866
Species.2 0.8975582 0.0000000 1.4140676 1.4836935 1.3928624 1.0210047
Species.3 0.7919713 1.4140676 0.0000000 0.8838707 0.7300773 1.2282366
Species.4 1.3067679 1.4836935 0.8838707 0.0000000 0.8285425 1.4854911
Species.5 1.1113227 1.3928624 0.7300773 0.8285425 0.0000000 0.9922403
Species.6 0.8857866 1.0210047 1.2282366 1.4854911 0.9922403 0.0000000
```

```
> signif(dist(Y2, upper=T, diag=T))
          Species.1 Species.2 Species.3 Species.4 Species.5 Species.6
Species.1 0.0000000 0.9125900 0.9085760 0.9440290 0.9376470 1.0031300
Species.2 0.9125900 0.0000000 1.5997000 1.8533900 1.6222400 0.9762570
Species.3 0.9085760 1.5997000 0.0000000 0.9802880 0.0297536 0.9856680
Species.4 0.9440290 1.8533900 0.9802880 0.0000000 1.0006900 1.7328400
Species.5 0.9376470 1.6222400 0.0297536 1.0006900 0.0000000 0.9931410
Species.6 1.0031300 0.9762570 0.9856680 1.7328400 0.9931410 0.0000000
> signif(dist(Y3, upper=T, diag=T))
          Species.1 Species.2 Species.3 Species.4 Species.5 Species.6
Species.1  0.000000  0.906444  0.706195  1.291670  1.028890  0.875201
Species.2  0.906444  0.000000  1.400120  1.471960  1.390920  0.979028
Species.3  0.706195  1.400120  0.000000  0.858727  0.689664  1.193820
Species.4  1.291670  1.471960  0.858727  0.000000  0.828554  1.535370
Species.5  1.028890  1.390920  0.689664  0.828554  0.000000  0.916426
Species.6  0.875201  0.979028  1.193820  1.535370  0.916426  0.000000
> dist_nmds
          Species.1 Species.2 Species.3 Species.4 Species.5 Species.6
Species.1 0.0000000 0.8975582 0.7919713 1.3067679 1.1113227 0.8857866
Species.2 0.8975582 0.0000000 1.4140676 1.4836935 1.3928624 1.0210047
Species.3 0.7919713 1.4140676 0.0000000 0.8838707 0.7300773 1.2282366
Species.4 1.3067679 1.4836935 0.8838707 0.0000000 0.8285425 1.4854911
Species.5 1.1113227 1.3928624 0.7300773 0.8285425 0.0000000 0.9922403
Species.6 0.8857866 1.0210047 1.2282366 1.4854911 0.9922403 0.0000000
```



REFERENCES

1. *Book: Modern Multidimensional Scaling - Theory and Applications (Chapter 7 & 12).*

Ingwer Borg, Patrick J. F. Groenen. 2005

2. *An Introduction to MDS*. Florian Wickelmaier. 2003

3. https://rich-d-wilkinson.github.io/MATH3030/6-mds.html