HUDM5124: Introduction to Multidimensional Scaling, Clustering, and Related Methods

# **Session 5:** Some Practical Issues in Using MDS

# Two versions of stress

$$\text{Stress}(1) = \left[ \frac{\sum_{i<j} (\text{f}(\delta_{ij}) - d_{ij})^2}{\sum_{i<j} d_{ij}^{\,2}} \right]^{1/2} \qquad \text{Stress}(2) = \left[ \frac{\sum_{i<j} (\text{f}(\delta_{ij}) - d_{ij})^2}{\sum_{i<j} (d_{ij} - \bar{d})^2} \right]^{1/2}$$

The numerator term alone is sometimes referred to as “raw stress”.

Use of Stress(1) can result in degenerate configurations (i.e., points collapsing into a few clumps, a simplex, a ring).

Thus Stress(2) is generally recommended.

Degeneracy may also be affected by how the normalization of the configuration matrix X (step 2) is accomplished.

# Availability of software for nonmetric MDS:

The Kruskal (1964) algorithm (and variants) :

| program | Author | source | distributed as: |
|---|---|---|---|
| MDSCALE | Kruskal | NETLIB | FORTRAN source |
| KYST-2A | Young | NETLIB | FORTRAN source |
| SYSTAT | Wilkinson | SPSS | commercial package |
| *isoMDS | | R | public domain package |

Other algorithms/software for nonmetric MDS:

| program | Author | source | distributed as: |
|---|---|---|---|
| ALSCAL | Young | SPSS, SAS | commercial package |
| *PROXSCAL | Leiden group | SPSS | commercial package |
| *smacof | de Leeuw & Mair | R | public domain |

# Some practical issues:

**Choosing the dimensionality**. The problem of determining the dimensionality R of the solution space must be addressed. Often the dimensionality is selected *a priori* based on theory or *post-hoc* based on interpretability or on perception of an "elbow" in stress function.

**How much data?** Because we are using only the ordinal info in the proximities in nonmetric MDS, we must have a higher ratio of # of data pts (obs) to # of estimated parameters, the (n-1)R coordinates of X (rule of thumb: 7+ stimuli per dimension?).

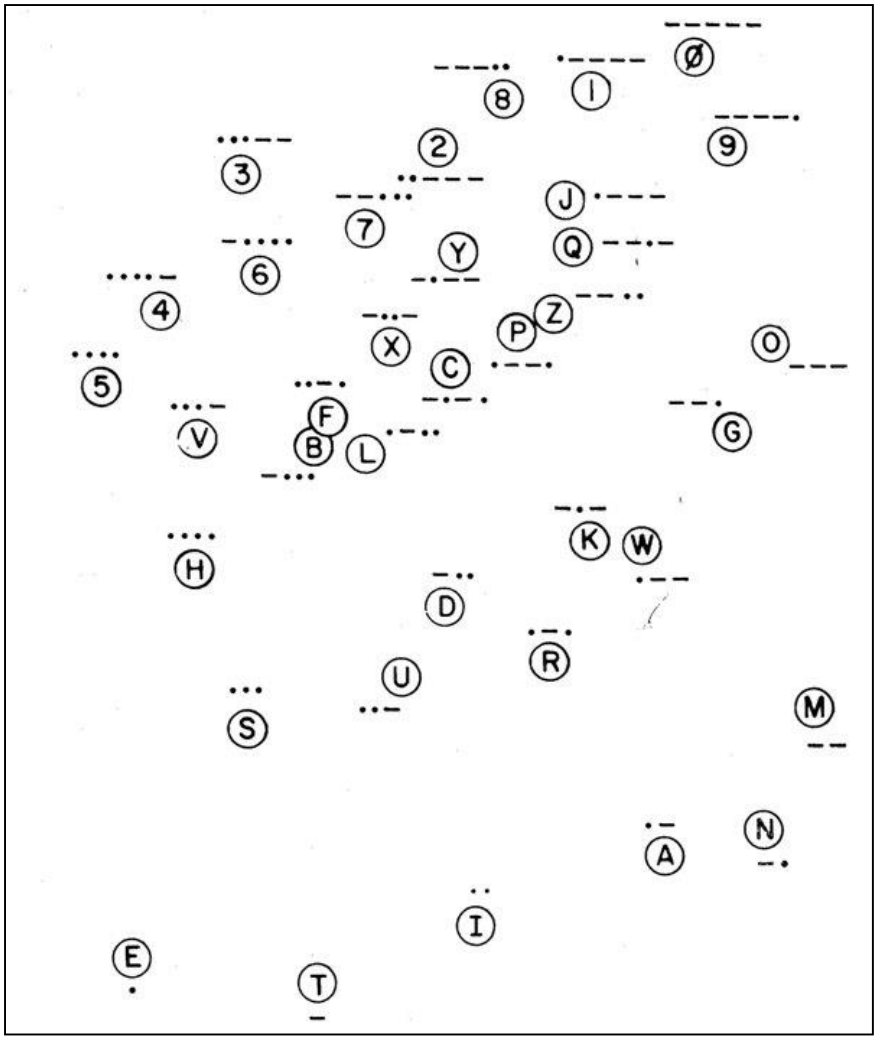
**Degenerate solutions.** Rings, clumps of multiple stimuli, etc. are often signs of a degenerate solution. **Fixes:** increase # of data pts, decrease dimensionality, use secondary approach to ties, try a metric solution.

**Interpreting solutions by eye.** Remember that the orientation of the solution w.r.t. the axes is arbitrary. High-dimensional graphical rotation software may be useful if R>2.

**Interpreting solutions "objectively".** Is there a relatively objective way to interpret dimensions? → regression of single "attributes" into the space: $A = b0 + b1X_1 + b2X_2 + ..$ Then plot regression coeff's.

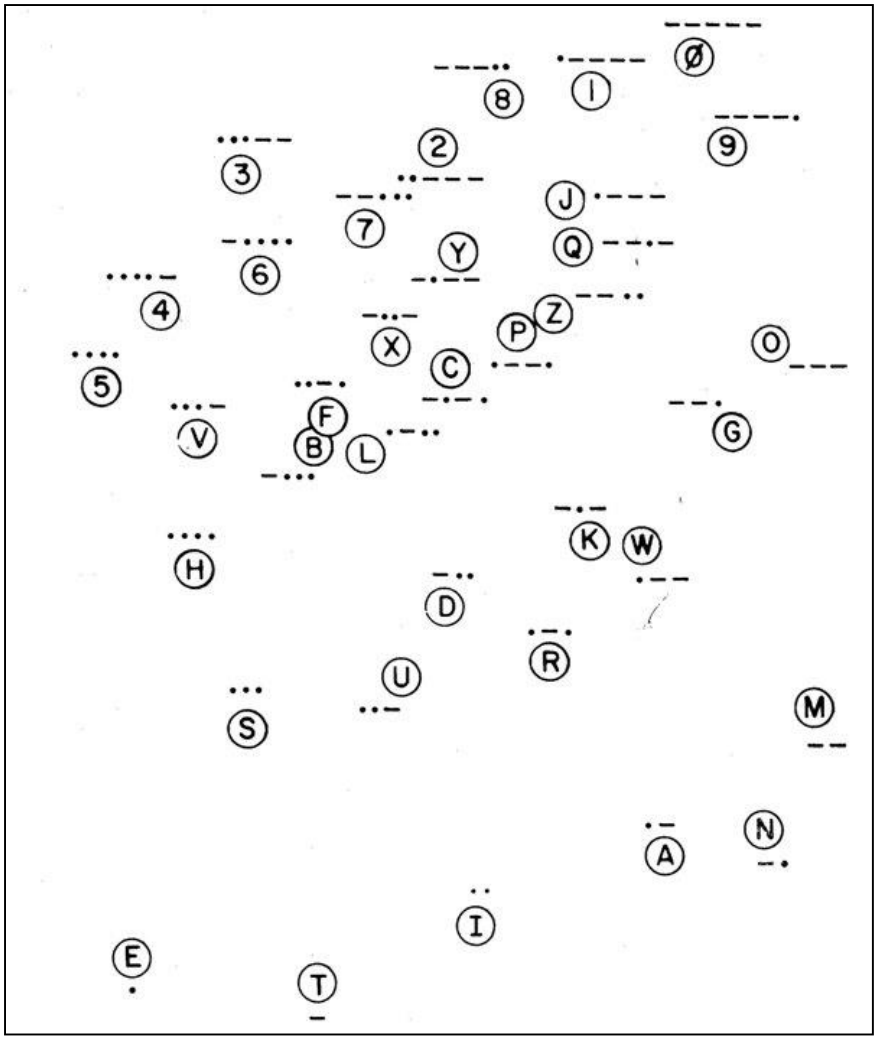
**Recovery of metric information:** even though this technique is "nonmetric", Young (1980) showed that good recovery of metric information is achieved

# 2D config, Morse code data - interpreting via attribute regression




?

# 2D config, Morse code data - interpreting via attribute regression




METHOD:

1) Define attribute vectors on stimuli:

A1 = # components of signal
A2 = proportion of dashes

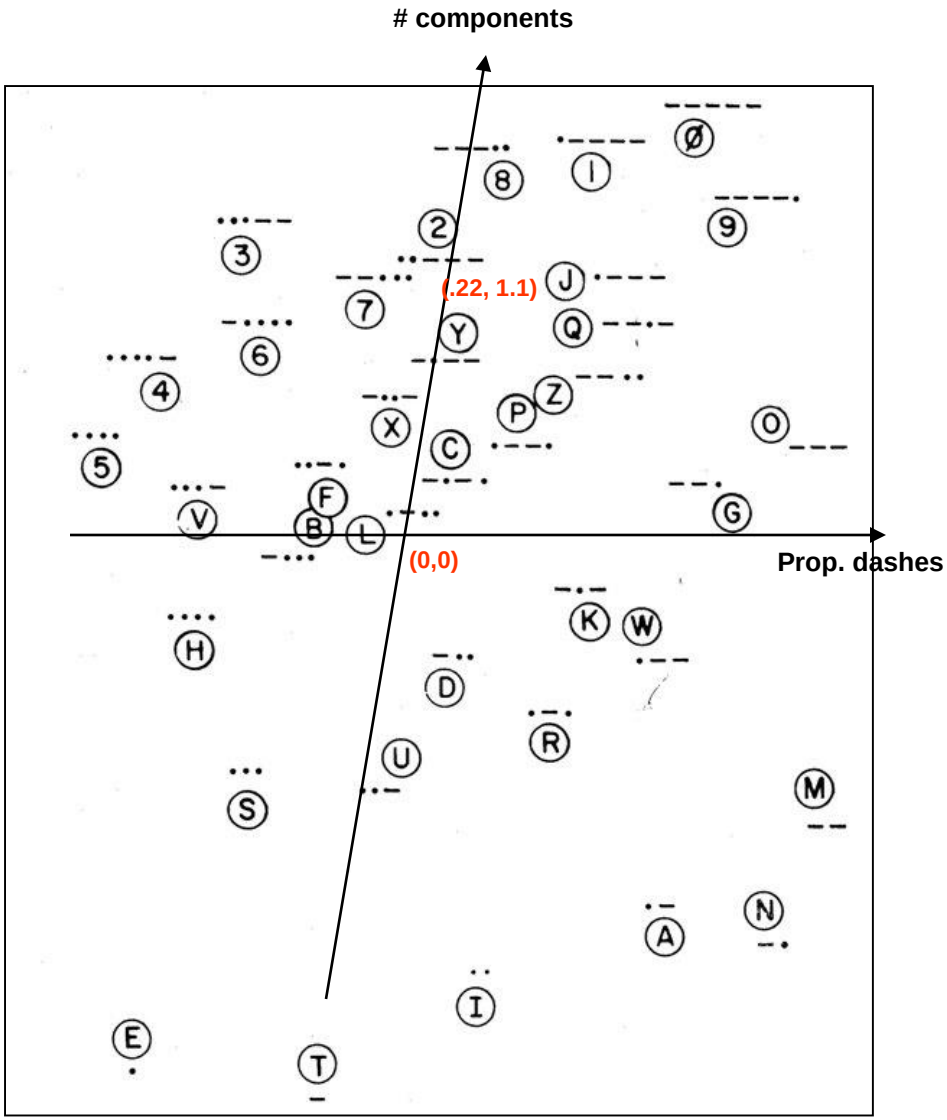
2) Regress each vector into config. space (X1 = horiz, X2 = vert)

A1 = 3.5 + 0.22 X1 + 1.10 X2

A2 = 0.5 + 0.92 X1 - 0.01 X2

3) For each attribute, plot the regression coefficients as a vector in the configuration space, through (0,0)

# 2D config, Morse code data - interpreting via attribute regression




## METHOD:

1) Define attribute vectors on stimuli:

A1 = # components of signal
A2 = proportion of dashes

2) Regress each vector into config. space (X1 = horiz, X2 = vert)

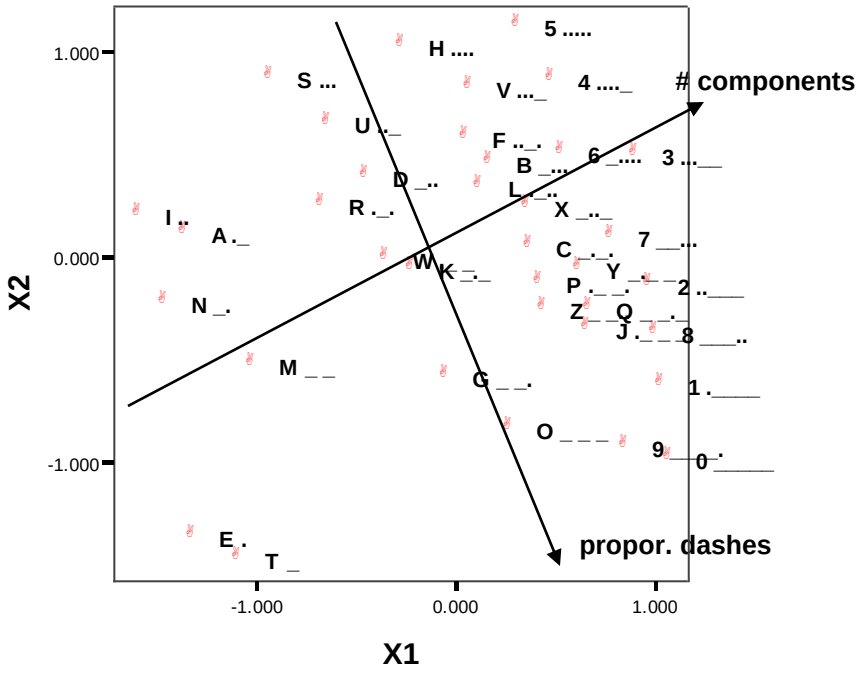
A1 = 3.5 + 0.22 X1 + 1.10 X2

A2 = 0.5 + 0.92 X1 - 0.01 X2

3) For each attribute, plot the regression coefficients as a vector in the configuration space, through (0,0)

# Example: plot of 2D Morse code solution (SPSS)

## interpretation via attribute regression




**Coefficients[a]**

| Model | | Unstandardized Coefficients | | Standardized Coefficients | | |
|---|---|---|---|---|---|---|
| | | B | Std. Error | Beta | t | Sig. |
| 1 | (Constant) | 3.667 | .051 | | 72.542 | .000 |
| | X1 | **1.356** | .065 | .910 | 20.784 | .000 |
| | X2 | **.603** | .080 | .330 | 7.544 | .000 |

a. Dependent Variable: Ncomp

**Coefficients[a]**

| Model | | Unstandardized Coefficients | | Standardized Coefficients | | |
|---|---|---|---|---|---|---|
| | | B | Std. Error | Beta | t | Sig. |
| 1 | (Constant) | .479 | .034 | | 14.302 | .000 |
| | X1 | **.111** | .043 | .287 | 2.573 | .015 |
| | X2 | **-.338** | .053 | -.712 | -6.379 | .000 |

a. Dependent Variable: PRdash

# Fitting an MDS configuration by majorization (smacof)

- Defining error in estimating an MDS configuration matrix X: (extracts from MMDS Ch. 8)

What does "for all available $\delta_{ij}$" mean? In practical research, we sometimes have *missing values*, so that some $\delta_{ij}$ are undefined. Missing values impose no restriction on any distances in $\mathbf{X}$. Therefore, we define fixed weights $w_{ij}$ with value 1 if $\delta_{ij}$ is known and $w_{ij} = 0$ if $\delta_{ij}$ is missing. Other values of $w_{ij}$ are also allowed, as long as $w_{ij} \geq 0$. This defines the final version of *raw Stress* (Kruskal, 1964b),

$$\sigma_r(\mathbf{X}) = \sum_{i<j} w_{ij}(d_{ij}(\mathbf{X}) - \delta_{ij})^2. \tag{8.4}$$

# On determining the global minimum of a function (e.g., stress-1)

or, more generally, for any point $P = (x, f(x))$,

$$\text{slope}(PQ) = \frac{f(x + \Delta x) - f(x)}{\Delta x}. \tag{8.8}$$

To find the tangent at point $P$ on the graph, it is necessary to move $Q$ very close to $P$. However, $Q$ should not become equal to $P$, because we need two points to uniquely identify the tangent line. This is expressed as follows:

$$\frac{dy}{dx} = \lim_{\Delta x \to 0} \frac{f(x + \Delta x) - f(x)}{\Delta x},$$

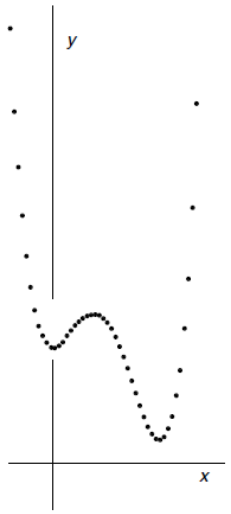



FIGURE 8.1. Some points for $y = 0.3x^4 - 2x^3 + 3x^2 + 5$.

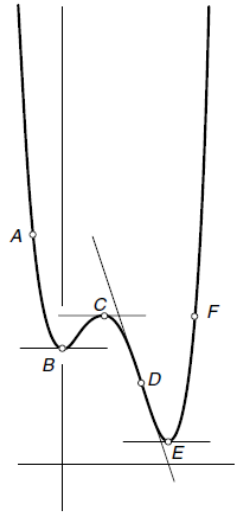



FIGURE 8.2. Graph of $y = 0.3x^4 - 2x^3 + 3x^2 + 5$, with tangent lines at points $B$, $C$, $D$, and $E$.

# Taking partial derivative of stress w.r.t a matrix X

TABLE 8.2. Example of differentiating the linear function tr $\mathbf{AX}$ with respect to an unknown matrix $\mathbf{X}$.

| | |
|---|---|
| (1) | $\mathbf{AX} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} x_{11} & x_{12} \\ x_{21} & x_{22} \end{bmatrix}$ |
| (2) | $f(\mathbf{X}) = \text{tr } (\mathbf{AX}) = a_{11}x_{11} + a_{12}x_{21} + a_{21}x_{12} + a_{22}x_{22}$ |
| (3) | $\partial f(\mathbf{X})/\partial \mathbf{X} = (\partial f(\mathbf{X})/\partial x_{ij})$ |
| (4) | $\begin{bmatrix} \partial f(\mathbf{X})/\partial x_{11} = a_{11} & \partial f(\mathbf{X})/\partial x_{12} = a_{21} \\ \partial f(\mathbf{X})/\partial x_{21} = a_{12} & \partial f(\mathbf{X})/\partial x_{22} = a_{22} \end{bmatrix} = \mathbf{A}'$ |
| (5) | rule: $\partial \text{tr } (\mathbf{AX})/\partial \mathbf{X} = \mathbf{A}'$ |

# Differentiating a matrix trace w/r/t matrix X

TABLE 8.3. Some rules for differentiating a matrix trace with respect to an unknown matrix $\mathbf{X}$; matrix $\mathbf{A}$ is a constant matrix; matrices $\mathbf{U}$, $\mathbf{V}$, $\mathbf{W}$ are functions of $\mathbf{X}$ (Schönemann, 1985).

---

(1) $\partial \text{tr}\,(\mathbf{A})/\partial \mathbf{X} = \mathbf{0}$

(2) $\partial \text{tr}\,(\mathbf{AX})/\partial \mathbf{X} = \mathbf{A}' = \partial \text{tr}\,[(\mathbf{AX})']/\partial \mathbf{X}$

(3) $\partial \text{tr}\,(\mathbf{X}'\mathbf{AX})/\partial \mathbf{X} = (\mathbf{A} + \mathbf{A}')\mathbf{X}$

(4) $\partial \text{tr}\,(\mathbf{X}'\mathbf{AX})/\partial \mathbf{X} = 2\mathbf{AX}$ if $\mathbf{A}$ is symmetric

(5) $\partial \text{tr}\,(\mathbf{U} + \mathbf{V})/\partial \mathbf{X} = \partial \text{tr}\,(\mathbf{U})/\partial \mathbf{X} + \partial \text{tr}\,(\mathbf{V})/\partial \mathbf{X}$

(6) $\partial \text{tr}\,(\mathbf{UVW})/\partial \mathbf{X} = \partial \text{tr}\,(\mathbf{WUV})/\partial \mathbf{X} = \partial \text{tr}\,(\mathbf{VWU})/\partial \mathbf{X}$
Invariance under "cyclic" permutations

(7) $\partial \text{tr}\,(\mathbf{UV})/\partial \mathbf{X} = \partial \text{tr}\,(\mathbf{U}_c\mathbf{V})/\partial \mathbf{X} + \partial \text{tr}\,(\mathbf{UV}_c)/\partial \mathbf{X}$
Product rule: $\mathbf{U}_c$ and $\mathbf{V}_c$ is taken as a constant matrix when differentiating

---

# Finding the minimum of a function by iterative majorization (de Leeuw, 1977)

The central idea of the majorization method is to replace iteratively the original complicated function $f(x)$ by an auxiliary function $g(x, z)$, where $z$ in $g(x, z)$ is some fixed value. The function $g$ has to meet the following requirements to call $g(x, z)$ a *majorizing function* of $f(x)$.

- The auxiliary function $g(x, z)$ should be simpler to minimize than $f(x)$. For example, if $g(x, z)$ is a quadratic function in $x$, then the minimum of $g(x, z)$ over $x$ can be computed in one step (see Section 8.2).
- The original function must always be smaller than or at most equal to the auxiliary function; that is, $f(x) \leq g(x, z)$.
- The auxiliary function should touch the surface at the so-called *supporting point* $z$; that is, $f(z) = g(z, z)$.

# The iterative majorization algorithm (cont.)

$x^*$. The last two requirements of the majorizing function imply the chain of inequalities

$$f(x^*) \leq g(x^*, z) \leq g(z, z) = f(z). \tag{8.11}$$

This chain of inequalities is named the *sandwich* inequality by De Leeuw (1993), because the minimum of the majorizing function $g(x^*, z)$ is squeezed between $f(x^*)$ and $f(z)$. A graphical representation of these inequalities is presented in Figure 8.4 for two subsequent iterations of iterative majorization of the function $f(x)$. The iterative majorization algorithm is given by

1. Set $z = z_0$, where $z_0$ is a starting value.
2. Find update $x^u$ for which $g(x^u, z) \leq g(z, z)$.
3. If $f(z) - f(x^u) < \varepsilon$, then stop. ($\varepsilon$ is a small positive constant.)
4. Set $z = x^u$ and go to 2.

# Illustration of the majorization algorithm

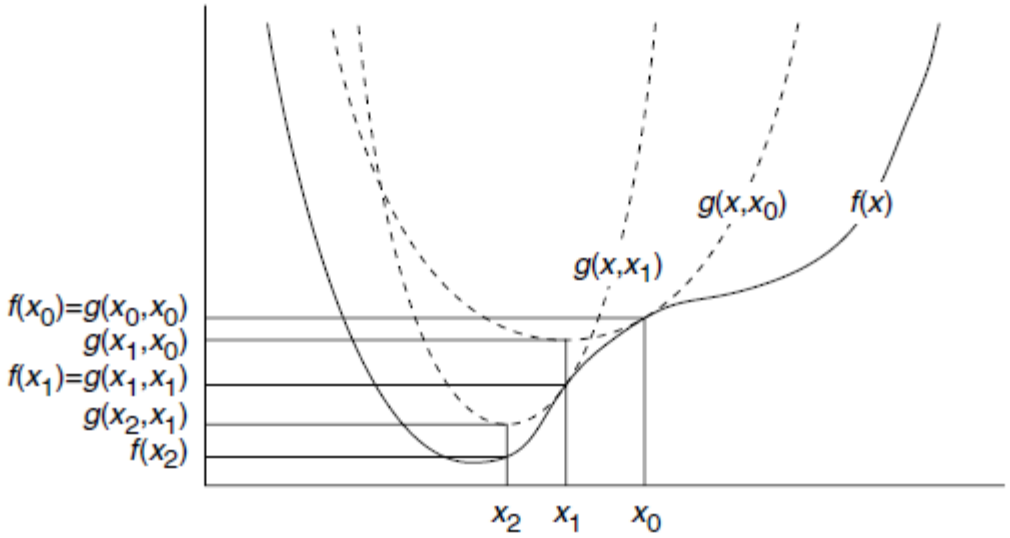



FIGURE 8.4. Illustration of two iterations of the iterative majorization method. The first iteration starts by finding the auxiliary function $g(x, x_0)$, which is located above the original function $f(x)$ and touches at the supporting point $x_0$. The minimum of the auxiliary function $g(x, x_0)$ is attained at $x_1$, where $f(x_1)$ can never be larger than $g(x_1, x_0)$. This completes one iteration. The second iteration is analogous to the first iteration.

# A linear majorizing function for a concave function..

given in Figure 8.5. But for such a function $f(x)$, it is always possible to have a straight line defined by $g(x, z) = ax + b$ (with $a$ and $b$ dependent on $z$) such that $g(x, z)$ touches the function $f(x)$ at $x = z$, and elsewhere the line defined by $g(x, z)$ is above the graph of $f(x)$. Clearly, $g(x, z) = ax + b$ is a linear function in $x$. Therefore, we call this type of majorization *linear majorization*. Any concave function $f(x)$ can be majorized by a linear function $g(x, z)$ at any point $z$. Thus, $g(x, z)$ satisfies all three requirements of a majorizing function. An example of a linear majorizing function $g(x, z)$ with supporting point $z$ of the concave function $f(x) = x^{1/2}$ is given in Figure 8.6.

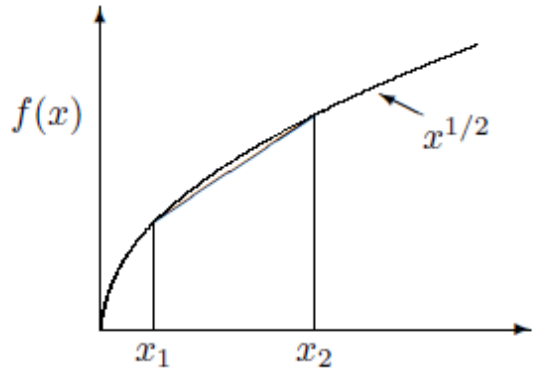



FIGURE 8.5. Graph of the concave function $x^{1/2}$.

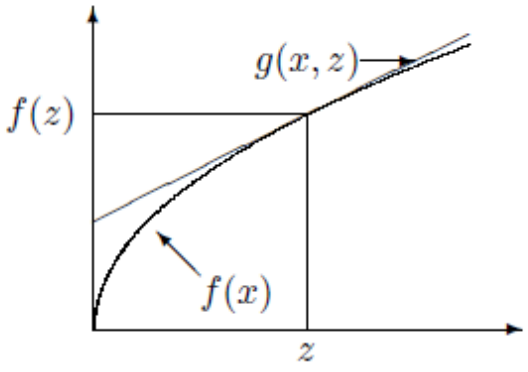



FIGURE 8.6. An example of linear majorization of the concave function $f(x) = x^{1/2}$ by the linear majorizing function $g(x, z)$.

# A quadratic majorizing function

The second class of functions that can be easily majorized is characterized by a bounded second derivative. For a function $f(x)$ with a bounded second derivative, there exists a quadratic function that has, compared to $f(x)$, a larger second derivative at any point $x$. This means that $f(x)$ does not have very steep parts, because there always exists a quadratic function that is steeper. This type of majorization can be applied if the function $f(x)$ can be majorized by $g(x,z) = a(z)x^2 - b(z)x + c(z)$, with $a(z) > 0$, and $a(z)$, $b(z)$, and $c(z)$ functions of $z$, but not of $x$. We call this type of majorization *quadratic majorization.*

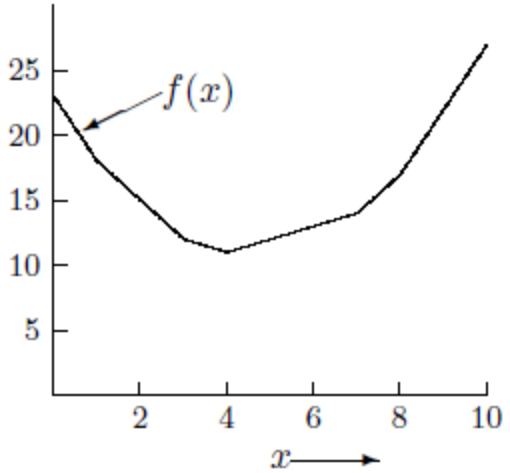



FIGURE 8.7. Graph of the function $f(x) = |x-1| + |x-3| + |x-4| + |x-7| + |x-8|$.

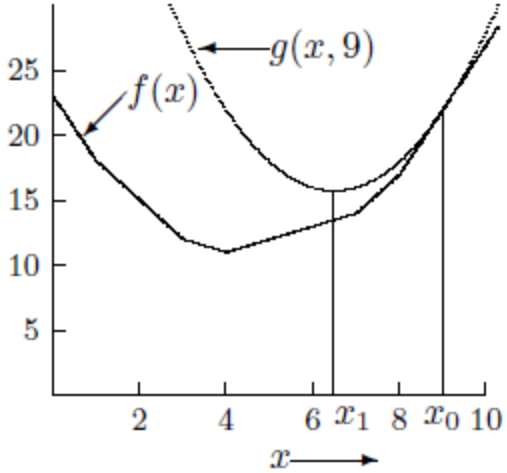



FIGURE 8.8. A quadratic majorizing function $g(x, x_0)$ of $f(x)$ with supporting point $x_0 = 9$.

# A simple example of majorizing to find an MDS solution

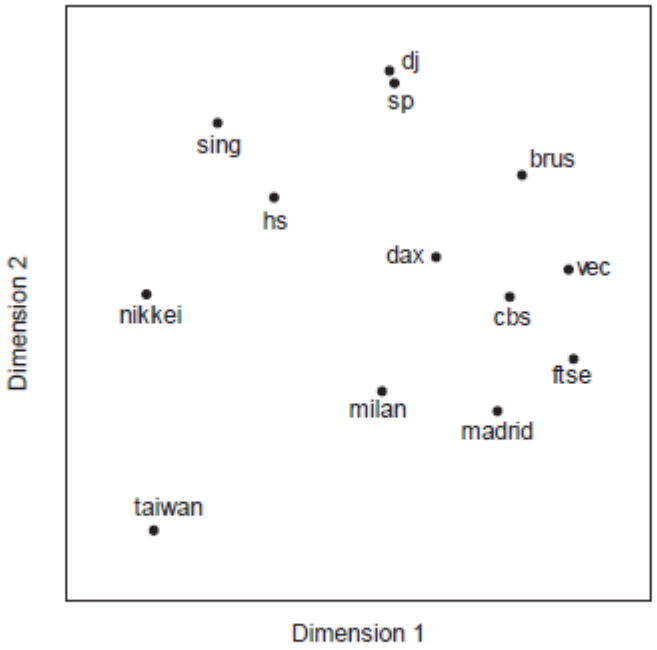
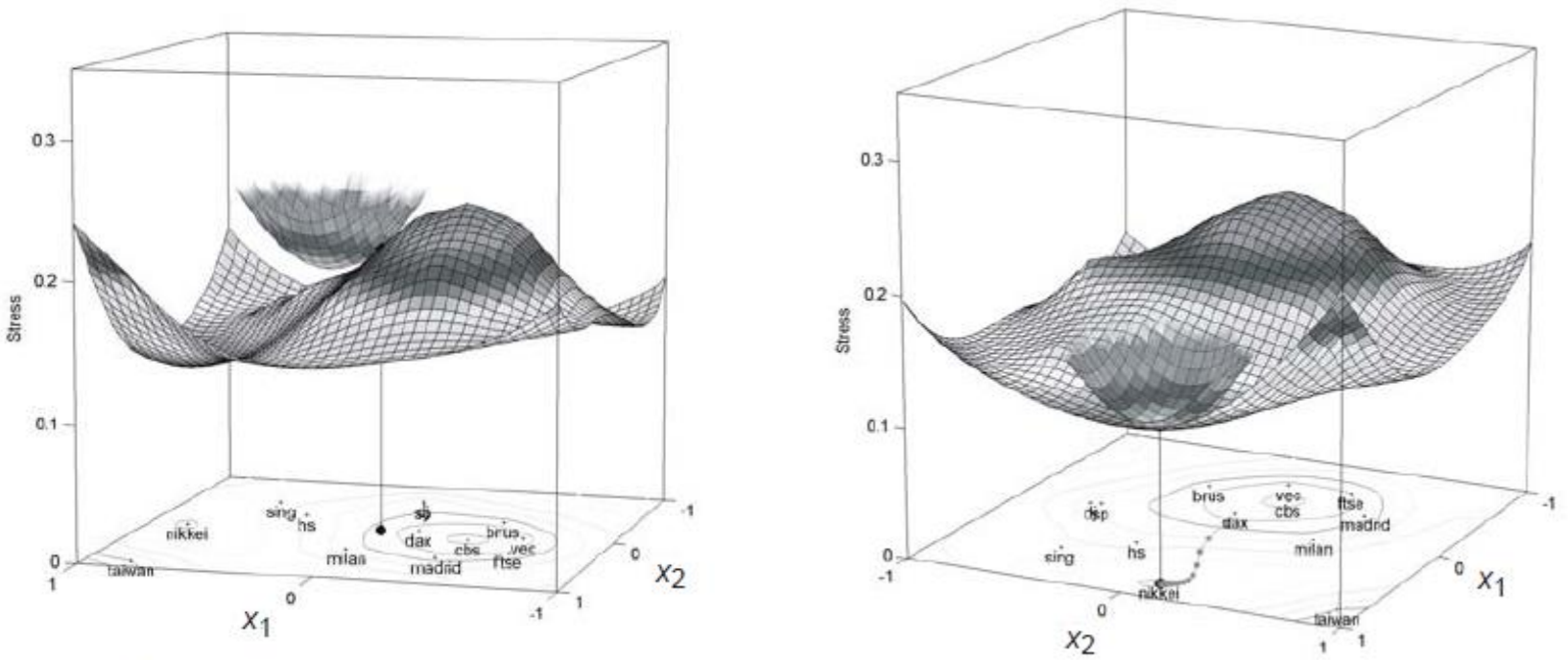



FIGURE 8.9. Ratio MDS solution of correlations between returns of 13 stock markets. The data are given in Exercise 3.3.

# A simple example of majorizing to find an MDS solution




a. First majorizing function

b. Final majorizing function

FIGURE 8.10. Visualization of the raw Stress function for the Stock market data where all coordinates are kept fixed except those of nikkei. For reference, the optimal position of nikkei is also shown. The upper panel shows the majorizing function with the origin as current estimate for the location of nikkei. The lower panel shows the final majorizing function and a trail of points in the $xy$-plane showing the positions of point nikkei in the different iterations.

# General method: majorizing to find an MDS solution

We now apply iterative majorization to the Stress function, which goes back to De Leeuw (1977), De Leeuw and Heiser (1977), and De Leeuw (1988). The acronym SMACOF initially stood for "Scaling by Maximizing a Convex Function," but since the mid-1980s it has stood for "Scaling by Majorizing a Complicated Function." Algorithms other than SMACOF

The Stress function (8.4) can be written as

$$\begin{aligned}\sigma_r(\mathbf{X}) &= \sum_{i<j} w_{ij}\left(\delta_{ij}-d_{ij}(\mathbf{X})\right)^2\\ &= \sum_{i<j} w_{ij}\delta_{ij}^2+\sum_{i<j} w_{ij}d_{ij}^2(\mathbf{X})-2\sum_{i<j} w_{ij}\delta_{ij}d_{ij}(\mathbf{X})\\ &= \eta_\delta^2+\eta^2(\mathbf{X})-2\rho(\mathbf{X}),\end{aligned}\tag{8.15}$$

and summing over all $i<j$ terms gives

$$\begin{aligned}\eta^2(\mathbf{X}) &= \sum_{i<j} w_{ij}d_{ij}^2(\mathbf{X})=\text{tr }\mathbf{X}'\left(\sum_{i<j} w_{ij}\mathbf{A}_{ij}\right)\mathbf{X}\\ &= \text{tr }\mathbf{X}'\mathbf{V}\mathbf{X}.\end{aligned}\tag{8.17}$$

We now switch to $-\rho(\mathbf{X})$, which is minus a weighted sum of the distances; that is,

$$-\rho(\mathbf{X}) = -\sum_{i<j}(w_{ij}\delta_{ij})d_{ij}(\mathbf{X}).$$

Combining (8.21) and (8.22), multiplying by $w_{ij}\delta_{ij}$, and summing over $i < j$ gives

$$\begin{aligned} -\rho(\mathbf{X}) &= -\sum_{i<j}(w_{ij}\delta_{ij})d_{ij}(\mathbf{X}) \\ &\leq -\text{tr } \mathbf{X}'\left(\sum_{i<j} b_{ij}\mathbf{A}_{ij}\right)\mathbf{Z} \\ &= -\text{tr } \mathbf{X}'\mathbf{B}(\mathbf{Z})\mathbf{Z}, \end{aligned} \tag{8.23}$$

Because equality occurs if $\mathbf{Z} = \mathbf{X}$, we have obtained the majorization inequality

$$-\rho(\mathbf{X}) = -\text{tr } \mathbf{X}'\mathbf{B}(\mathbf{X})\mathbf{X} \leq -\text{tr } \mathbf{X}'\mathbf{B}(\mathbf{Z})\mathbf{Z}.$$

Thus, $-\rho(\mathbf{X})$ can be majorized by the function $-\text{tr } \mathbf{X}'\mathbf{B}(\mathbf{Z})\mathbf{Z}$, which is a linear function in $\mathbf{X}$.

Thus $\tau(\mathbf{X}, \mathbf{Z})$ is a simple majorizing function of Stress that is quadratic in $\mathbf{X}$. Its minimum can be obtained analytically by setting the derivative of $\tau(\mathbf{X}, \mathbf{Z})$ equal to zero; that is,

$$\nabla\tau(\mathbf{X}, \mathbf{Z}) = 2\mathbf{V}\mathbf{X} - 2\mathbf{B}(\mathbf{Z})\mathbf{Z} = \mathbf{0},$$

so that $\mathbf{V}\mathbf{X} = \mathbf{B}(\mathbf{Z})\mathbf{Z}$. To solve this system of linear equations for $\mathbf{X}$, we would usually premultiply both sides by $\mathbf{V}^{-1}$. However, the inverse $\mathbf{V}^{-1}$ does not exist, because $\mathbf{V}$ is not of full rank. Therefore, we revert to the Moore–Penrose[3] inverse. The Moore–Penrose inverse of $\mathbf{V}$ is given by $\mathbf{V}^+ = (\mathbf{V} + \mathbf{1}\mathbf{1}')^{-1} - n^{-2}\mathbf{1}\mathbf{1}'$. The last term, $-n^{-2}\mathbf{1}\mathbf{1}'$, is irrelevant in SMACOF as $\mathbf{V}^+$ is subsequently multiplied by a matrix orthogonal to $\mathbf{1}$, because $\mathbf{B}(\mathbf{Z})$ also has eigenvector $\mathbf{1}$ with eigenvalue zero. This leads us to the update formula of the SMACOF algorithm,

$$\mathbf{X}^u = \mathbf{V}^+\mathbf{B}(\mathbf{Z})\mathbf{Z}. \tag{8.28}$$

If all $w_{ij} = 1$, then $\mathbf{V}^+ = n^{-1}\mathbf{J}$ with $\mathbf{J}$ the *centering matrix* $\mathbf{I} - n^{-1}\mathbf{1}\mathbf{1}'$, so that the update simplifies to

$$\mathbf{X}^u = n^{-1}\mathbf{B}(\mathbf{Z})\mathbf{Z}. \tag{8.29}$$

De Leeuw and Heiser (1980) call (8.28) the *Guttman transform*, in recognition of Guttman (1968).

# The smacof algorithm

The SMACOF algorithm for MDS can be summarized by

1. Set $\mathbf{Z} = \mathbf{X}^{[0]}$, where $\mathbf{X}^{[0]}$ is some (non)random start configuration. Set $k = 0$. Set $\varepsilon$ to a small positive constant.
2. Compute $\sigma_r^{[0]} = \sigma_r(\mathbf{X}^{[0]})$. Set $\sigma_r^{[-1]} = \sigma_r^{[0]}$.
3. While $k = 0$ or $(\sigma_r^{[k-1]} - \sigma_r^{[k]} > \varepsilon$ and $k \leq$ maximum iterations) do
4. Increase iteration counter $k$ by one.
5. Compute the Guttman transform $\mathbf{X}^{[k]}$ by (8.29) if all $w_{ij} = 1$, or by (8.28) otherwise.
6. Compute $\sigma_r^{[k]} = \sigma_r(\mathbf{X}^{[k]})$.
7. Set $\mathbf{Z} = \mathbf{X}^{[k]}$.
8. End while

# Using smacof
## (available in SPSS Proxscal or R smacof package)

R example:

```
# using smacof in R:
install.packages("smacof")
library(smacof)

help(smacofSym)        # smacof for symmetric proximity matrices
# note “mds” is a synonym for “smacofSym”

# example of using smacofSym with Torgerson start (delta=input diss matrix)
sol <- smacofSym(delta, ndim = 2, type = "ordinal", init = "torgerson", ties = “
primary“)

# example of using smacofSym with random start
sol <- smacofSym(delta, ndim = 2, type = “interval", init = “random“)
```