UL = MULTIDIMENSIONAL SCALING (MDS)

- "Multidimensional scaling (MDS) is a method that represents measurements of similarity (or dissimilarity) among pairs of objects as distances among points of a low-dimensional multidimensional space". (Borg & Groenen (2005))

Usually, 2D, for visualization purposes.

Instead of the usual observations $x_1, \dots, x_n \in \mathbb{R}^d$, the data consists of measurements of similarity / dissimilarity among pairs of objects: d_{ij} (e.g. $d_{ij} = d(x_i, x_j)$ where d = distance in a metric space).

(d_{ij}: object i, object j)

An object can be
- a person
- a country
- an attribute
- a stimulus ... /...

- Goal: to represent these objects in a 2D space

- Ex: Perception of Morse Signals (Rothkopf (1957))

There are 36 Morse signals (26 for the letters + 10 for the numbers). The task is to judge if two morse signals are the same or different. The conducted experiment consisted of 598 subjects (unfamiliar with the Morse code), who were shown 351 pairs. Each pair was presented in two orders. For example, for the pair (A, C) = (·–, –·–·), first ·– then –·–·, and also –·–· then ·–.

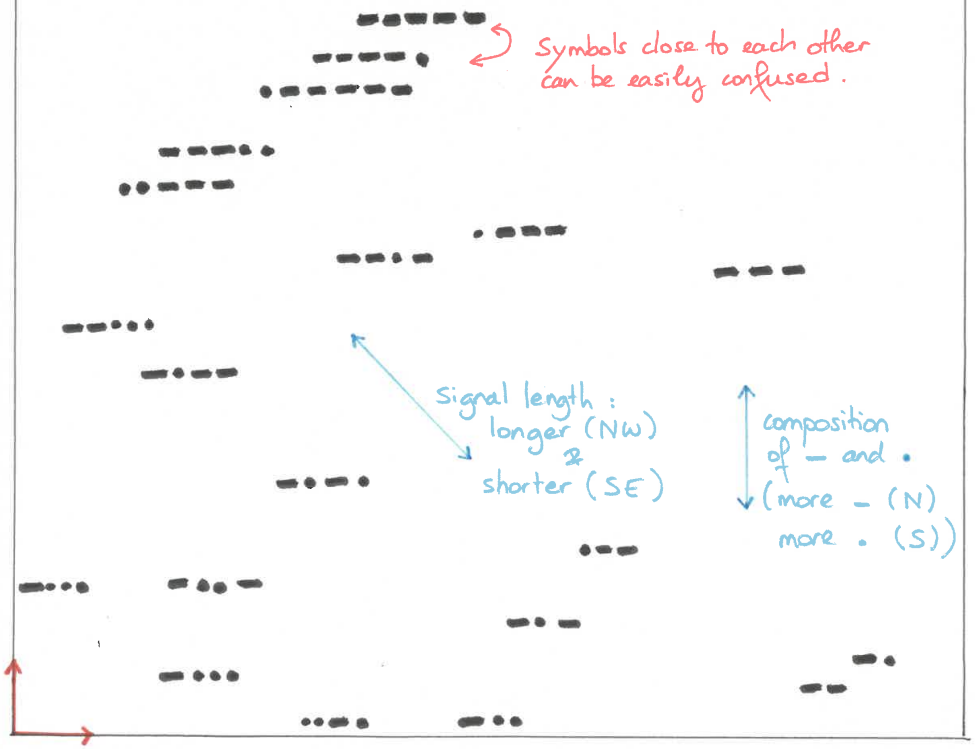
Which Morse codes are the most easy to be confused ?

At the end of the experiment, we obtain a confusion matrix

	A	B	...	9	0
A	92	4	...		3
B	5	84	...		4
⋮	⋮	⋮	⋱		⋮
9					
0	9	3	...		94

matrix of similarities

Since distances are always symmetric, the matrix is first symmetrized, and the diagonal terms are set to 0 ($d(x_i, x_i) = 0 \ \forall i$). MDS would provide a visual representation of the confusion matrix. It would look something like this :



I - CLASSICAL MDS.

- An $(n \times n)$ matrix $\mathcal{D}_X = (\delta_{ij})_{i,j}$ is called a DISTANCE or AFFINITY MATRIX if (i) it is symmetric $\delta_{ij} = \delta_{ji}$
  (ii) $\delta_{ii} = 0$
  (iii) $\delta_{ij} > 0 \quad \forall i \neq j$.

Typically, δ_{ij} represents the distance between two observations x_i and $x_j \in \mathbb{R}^d$ (d: a high dimensional space).

$\delta_{ij} = d(x_i, x_j) = \| x_i - x_j \|$

We assume here that d = Euclidean distance, but it could be something else.

Given $\mathcal{D}_X$, the task is to find n points $z_1, \dots, z_n \in \mathbb{R}^r$ $(r \ll d)$ such that $\mathcal{D}_Z := (d_{ij})_{i,j}$; $d_{ij} = d(z_i, z_j) = \| z_i - z_j \|$ is "similar" to $\mathcal{D}_X$:

$$\min_{z_1,\dots,z_n \in \mathbb{R}^r} \sum_{i,j=1}^{n} (\delta_{ij} - d_{ij})^2 \qquad \text{(MDS 1)}$$

δ_{ij} given; d_{ij} depend on z_i, z_j

$\delta_{ij} = \|x_i - x_j\|$

$d_{ij} = \| z_i - z_j \|$

MDS OPTIMIZATION PROBLEM.

- Fact: Consider $\mathcal{D}_X^{(2)} := (\delta_{ij}^2)_{i,j}$ = the matrix of square Euclidean distances. Then $-\frac{1}{2} J_n \mathcal{D}_X^{(2)} J_n = X X^t$, where

  $X = \begin{pmatrix} x_{11} & x_{1n} \\ x_{n1} & x_{nn} \end{pmatrix} = \begin{pmatrix} — x_1^t — \\ — x_n^t — \end{pmatrix}$ = matrix of obs. $(n \times d)$ (assume centered columns)

  $J_n = I_n - \frac{1}{n} 1 1^t$ = centering matrix $\quad 1 = \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix}$ $(1 \times n)$

→ First, note that for an $(n \times n)$ square matrix $A = (a_{ij})$,
AJ_n removes the mean of each row of A, while
$J_n A$ removes the mean of each column of A.

- $J_n A = A - \frac{1}{n} 1 1^t A = \left(\sum_{i=1}^n a_{i1} \quad \dots \quad \sum_{i=1}^n a_{in} \right)$

$$\| \quad \frac{1}{n} \begin{pmatrix} \sum a_{i1} & \dots & \sum a_{in} \\ \sum a_{i1} & \dots & \sum a_{in} \end{pmatrix}$$

$$\Rightarrow J_n A = \begin{pmatrix} a_{11} - a_{\cdot 1} & \dots & a_{1n} - a_{\cdot n} \\ \vdots & & \vdots \\ a_{n1} - a_{\cdot 1} & \dots & a_{nn} - a_{\cdot n} \end{pmatrix}$$

columns are centered

where $a_{\cdot j} = \frac{1}{n} \sum_{i=1}^n a_{ij}$ = column j mean

- likewise, $AJ_n = \begin{pmatrix} a_{11} - a_{1\cdot} & \dots & a_{1n} - a_{1\cdot} \\ \vdots & & \vdots \\ a_{n1} - a_{n\cdot} & \dots & a_{nn} - a_{n\cdot} \end{pmatrix}$

rows are centered

where $a_{i\cdot} = \frac{1}{n} \sum_{j=1}^n a_{ij}$ = row i mean

- In addition,

$$J_n A J_n = \begin{pmatrix} a_{11} - a_{1\cdot} - a_{\cdot 1} + a_{\cdot\cdot} & \dots & a_{1n} - a_{1\cdot} - a_{\cdot n} + a_{\cdot\cdot} \\ \vdots & & \\ a_{n1} - a_{n\cdot} - a_{\cdot 1} + a_{\cdot\cdot} & \dots & a_{nn} - a_{n\cdot} - a_{\cdot n} + a_{\cdot\cdot} \end{pmatrix}$$

where $a_{\cdot\cdot} = \frac{1}{n^2} \sum_{i,j=1}^n a_{ij}$

$\Rightarrow (J_n A J_n)_{ij} = a_{ij} - a_{i\cdot} - a_{\cdot j} + a_{\cdot\cdot}$ (*)

→ Put $B := -\frac{1}{2} J_n D_X^{(2)} J_n$.

We show next that $B = XX^t$ (assuming X has centered columns)

Note that the (i,j) element of the matrix $-2B$ is $\delta_{ij}^2 - \delta_{i\cdot}^2 - \delta_{\cdot j}^2 + \delta_{\cdot\cdot}^2$, where (follows from (*) page 4)

$$\begin{cases} \delta_{i\cdot}^2 = \frac{1}{n}\sum_{j=1}^n \|x_i - x_j\|^2 \\ \delta_{\cdot j}^2 = \frac{1}{n}\sum_{i=1}^n \|x_i - x_j\|^2 \\ \delta_{\cdot\cdot}^2 = \frac{1}{n^2}\sum_{i,j=1}^n \|x_i - x_j\|^2 \end{cases}$$

In addition, $\|x_i - x_j\|^2 = \|x_i\|^2 + \|x_j\|^2 - 2 x_i^t x_j$. $(i,j=1,\dots,n)$

Assuming that X has centered columns: $\sum_{i=1}^n x_{ij} = 0 \quad \forall j$,

we see that

$$\sum_{i=1}^n x_i^t x_j = \sum_{i=1}^n \sum_{k=1}^d x_{ik} x_{jk} = \sum_{k=1}^d x_{jk} \boxed{\sum_{i=1}^n x_{ik}} = 0$$

$= 0 \quad \forall\, k=1,\dots,d$

$$\Rightarrow \sum_{i=1}^n \|x_i - x_j\|^2 = \sum_{i=1}^n \|x_i\|^2 + n\|x_j\|^2 - 2\sum_{i=1}^n x_i^t x_j .$$

$$\boxed{\delta_{\cdot j}^2 = \frac{1}{n}\sum_{i=1}^n \|x_i\|^2 + \|x_j\|^2}$$

Likewise, $\sum_{j=1}^n \|x_i - x_j\|^2 = \sum_{j=1}^n \|x_j\|^2 + n\|x_i\|^2$, and

$$\boxed{\delta_{i\cdot}^2 = \frac{1}{n}\sum_{j=1}^n \|x_j\|^2 + \|x_i\|^2}$$

&

$$\boxed{\delta_{\cdot\cdot}^2 = \frac{2}{n}\sum_{i=1}^n \|x_i\|^2}$$

$$\Rightarrow \delta_{ij}^2 - \delta_{i\cdot}^2 - \delta_{\cdot j}^2 + \delta_{\cdot\cdot}^2 = \|x_i - x_j\|^2 - \|x_i\|^2 - \|x_j\|^2 = -2 x_i^t x_j .$$

$$\Rightarrow -\frac{1}{2}\left(\delta_{ij}^2 - \delta_{i\cdot}^2 - \delta_{\cdot j}^2 + \delta_{\cdot\cdot}^2\right) = x_i^t x_j . \quad (i,j=1,\dots,n)$$

$$= \left(XX^t\right)_{i,j} \quad \blacksquare$$

Provided X has centered columns, we only need to know $\delta_{ij} = \|x_i - x_j\|$ to calculate $\langle x_i, x_j \rangle$, we do not need to know x_i and x_j.

⇒ Instead of considering the original (MDS 1) (page 3) optimization problem, we may consider the matrix of square Euclidean distances $D_X^{(2)}$, apply the transform $B = -\frac{1}{2} J D_X^{(2)} J$, and search for low-dimensional representations $z_1,\dots,z_n$ such that

$$\min_{\substack{z_1,\dots,z_n \\ \in \mathbb{R}^r}} \sum_{i,j=1}^n \left(\langle x_i, x_j \rangle - \langle z_i, z_j \rangle \right)^2$$

(MDS 2) aka CLASSICAL MDS.

Remarks:

(i) $\langle x_i, x_j \rangle = (i,j)$ element of $B = -\frac{1}{2} J D_X^{(2)} J$

(ii) $z_i \in \mathbb{R}^r$, $x_i \in \mathbb{R}^d$, with $r \ll d$ (usually $r=2$)

(iii) Columns of $X = \begin{pmatrix} — x_1^t — \\ — x_n^t — \end{pmatrix}$ are assumed centered, so that $B = -\frac{1}{2} J D_X^{(2)} J$. Since the same linear transformation is applied to $D_Z^{(2)}$, the solution to (MDS 2) must also be centered.

→ Solution to (MDS 2) optimization problem.

Consider the SVD decomposition of $X = U \Lambda V^t$ ($n \times d$) ($n \times d$)($d \times d$)($d \times d$).

Then $XX^t = U \Lambda^2 U^t$, and the (i,j) entry of XX^t is $\langle x_i, x_j \rangle = (U \Lambda^2 U^t)_{ij}$

↑ term appearing in the optimization problem.

Then, the Eckart-Young theorem (p.⑧ in UL: PCA) ensures that the best rank r approximation to XX^t (measured with respect to Frobenius norm) is given by $U_r \Lambda_r^2 U_r^t$, where

- → Λ_r^2 ($r \times r$) = diagonal matrix, retaining the largest r values of Λ^2
- → U_r = ($n \times r$) matrix retaining the r columns of U associated with the r largest values of Λ^2.

⇒ Solution Z ($n \times r$) to (MDS 2) satisfies $ZZ^t = U_r \Lambda_r^2 U_r^t = (U_r \Lambda_r)(U_r \Lambda_r)^t$

⇒ $\boxed{Z = U_r \Lambda_r}$ ($n \times r$) ($n \times r$) ($r \times r$)

= PCA solution.

Classical MDS & PCA return the same solution.

• Algorithm. (Classical MDS)

(a) Start with a distance matrix D_X
(b) Compute $B = -\frac{1}{2} J_n D_X J_n$
(c) Consider the eigenvalue / eigenvector decomposition of B
(d) Retain the largest r components

II - THE SMACOF ALGORITHM

The SMACOF (Scaling by MAjorizing a COmplicated Function) approach searches to minimize the stress function

$$\sigma(Z) = \sum_{i<j} w_{ij} (\delta_{ij} - d_{ij})^2 \quad \text{(MDS 3)}$$

De Leeuw & Heiser (1977)

$Z = \begin{pmatrix} -z_1^t- \\ \vdots \\ -z_n^t- \end{pmatrix}$ ($n \times r$) or $Z = \begin{pmatrix} | & & | \\ z_1 & \cdots & z_r \\ | & & | \end{pmatrix}$

weights (≥ 0)

given

$d_{ij} = \| z_i - z_j \|$

Weight w_{ij} measures the importance of the residual $(\delta_{ij} - d_{ij})^2$. Setting $w_{ij} = 0$ can be used to handle missing data.

Note that the overall scaling does not matter: scaling δ_{ij} by a factor α and coordinates $z_1, \dots, z_n$ by the same factor, so that $d_{ij}(\alpha Z) = \alpha d_{ij}(Z)$, yields $\alpha^2 \sigma(Z)$. De Leeuw fixes $\sum_{i<j} w_{ij} \delta_{ij}^2$ to some fixed constant, in order to standardize the solution. He chooses $\sum_{i<j} w_{ij} \delta_{ij}^2 = \frac{n(n-1)}{2}$ so that dispairties are on average dispersed around 1.

$$\Rightarrow \sigma(Z) = \underbrace{\sum_{i<j} w_{ij} \delta_{ij}^2}_{= \frac{n(n-1)}{2}} + \underbrace{\sum_{i<j} w_{ij} d_{ij}^2}_{= \eta^2(Z)} - 2 \underbrace{\sum_{i<j} w_{ij} \delta_{ij} d_{ij}}_{= \rho(Z)}$$

• Expressions for $\eta^2(Z)$ and $\rho(Z)$: First, note that

$$d_{ij}^2 = \sum_{l=1}^{r} (z_{il} - z_{jl})^2 = \sum_{l=1}^{r} (z_l^t (e_i - e_j))^2$$

$$\cdots\ d_{ij}^2 = \sum_{l=1}^{r} z_l^t (e_i - e_j)(e_i - e_j)^t z_l$$
$$= \mathrm{Tr}\left\{ z^t (e_i - e_j)(e_i - e_j)^t z \right\}$$
$$= \mathrm{Tr}\left\{ z^t A_{ij} z \right\},$$

where we defined $A_{ij} := (e_i - e_j)(e_i - e_j)^t = \begin{pmatrix} & i & & j & \\ i & 1 & & -1 & \\ & & & & \\ j & -1 & & 1 & \end{pmatrix}$

It follows that

$$\eta^2(z) = \sum_{i<j} w_{ij} d_{ij}^2 = \sum_{i<j} w_{ij} \mathrm{Tr}\left\{ z^t A_{ij} z \right\}$$
$$= \mathrm{Tr}\left\{ z^t \left(\sum_{i<j} w_{ij} A_{ij} \right) z \right\}$$
$$= \mathrm{Tr}\left\{ z^t V z \right\},$$

where $V := \sum_{i<j} w_{ij} A_{ij}$.

↖ When all $w_{ij} = 1$, V simplifies to $V = n\underline{\underline{I}} - \underline{1}\,\underline{1}^t$.

Assuming all $d_{ij} > 0$,

$$\rho(z) = \sum_{i<j} w_{ij} \delta_{ij} d_{ij} = \sum_{i<j} w_{ij} \frac{\delta_{ij}}{d_{ij}} d_{ij}^2$$
$$= \sum_{i<j} w_{ij} \delta_{ij} d_{ij}^{-1} \mathrm{Tr}\left\{ z^t A_{ij} z \right\}$$
$$= \mathrm{Tr}\left\{ z^t \left(\sum_{i<j} w_{ij} \boxed{\delta_{ij} d_{ij}^{-1}} A_{ij} \right) z \right\}$$
$$\qquad\qquad \| \; s_{ij}(z)$$

If $d_{ij} = 0$, set $s_{ij}(z) = 0$

$$\Rightarrow \rho(z) = \mathrm{Tr}\left\{ z^t B(z) z \right\},$$

where
- $B(z) := \sum_{i<j} w_{ij} s_{ij}(z) A_{ij}$
- $s_{ij}(z) = \begin{cases} \delta_{ij} d_{ij}^{-1} & \text{if } d_{ij} > 0 \\ 0 & \text{if } d_{ij} = 0 \end{cases}$

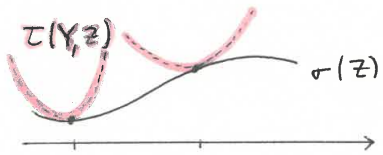
In summary,

$$\sigma(z) = \frac{n(n-1)}{2} + \mathrm{Tr}\left\{ z^t V z \right\} - 2\,\mathrm{Tr}\left\{ z^t B(z) z \right\}.$$

↖ To minimize this function, we use the SMACOF approach.

Idea: Instead of minimizing $\sigma(z)$ directly, minimize a majorizing function $\tau(Y, z)$ which satisfies:
- $\tau(Y, z)$ is easier to minimize (e.g. quad.)
- $\sigma(z) \le \tau(Y, z)$
- $\sigma(z) = \tau(z, z)$

↖ Y is called a supporting point



In our setting, consider

$$\tau(Y, z) := \frac{n(n-1)}{2} + \mathrm{Tr}\left\{ z^t V z \right\} - 2\,\mathrm{Tr}\left\{ z^t B(Y) Y \right\} \quad (*)$$

↳ Quadratic in z

↳ Clearly τ and σ coincide at $Y = z$

↳ $\sigma(z) \le \tau(Y, z)$; a consequence of the CS inequality.

Indeed, we want to show that $\mathrm{Tr}\left\{ z^t B(Y) Y \right\} \le \mathrm{Tr}\left\{ z^t B(z) z \right\}$

Assuming all $d_{ij}(Y) > 0$,

$$\mathrm{Tr}\left\{ z^t B(Y) Y \right\}$$
$$= \sum_{i<j} w_{ij} \frac{\delta_{ij}}{d_{ij}(Y)} \sum_{l=1}^{d} (z_{il} - z_{jl})(y_{il} - y_{jl})$$

$$\langle z_i - z_j, y_i - y_j \rangle = \sum_{l=1}^{d} (z_{il} - z_{jl})(y_{il} - y_{jl})$$

CS ↳ $\leq d_{ij}(z)\, d_{ij}(Y)$

$$\Rightarrow \mathrm{Tr}\{Z^t B(Y) Y\} \leq \sum_{i<j} w_{ij} \frac{\delta_{ij}}{d_{ij}(Y)} d_{ij}(z)\, d_{ij}(Y)$$

$$= \mathrm{Tr}\{Z^t B(Z) Z\}$$

Note that if $d_{ij}(Y) = 0$, then $y_i = y_j$ and $\mathrm{Tr}\{Z^t B(Y) Y\} = 0$ & the inequality is still true. ■

- A general iterative majorization algorithm:
  - (i) Start with $Y_0 = Z_0$
  - (ii) For $k = 0, 1, 2, \ldots$
    - Find Z_{k+1} s.t. $\tau(Y_k, Z_{k+1}) \leq \tau(Y_k, Y_k)$
    - Set $Y_{k+1} = Z_{k+1}$

And indeed, $\sigma(Z_{k+1}) \leq \tau(Z_k, Z_{k+1})$
$\leq \tau(Z_k, Z_k) = \sigma(Z_k)$

- It remains to compute the gradient of $\tau(Y, Z)$ with respect to Z:

$$\nabla\tau(Y, Z) = 2VZ - 2B(Y)Y = 0$$

↳ Need to solve $VZ = B(Y)Y$. (**)

$$Z = V^\dagger B(Y) Y,$$

where $V^\dagger$ is the Moore-Penrose inverse of V.

SMACOF Starting with Z_0, iterate $Z_{k+1} = V^\dagger B(Z_k) Z_k$

× Remark: SMACOF & Gradient Descent.

The stress function is $\sigma(Z) = \mathrm{Tr}\{Z^t V Z\} - 2\mathrm{Tr}\{Z^t B(Z) Z\}$ + constant.

Its gradient is

$$\nabla\sigma(Z) = 2VZ - 2B(Z)Z$$

↑ Not "that" obvious

Consider

$$\frac{d}{dz_{kl}} \left\{ \sum_{i<j} w_{ij} \delta_{ij} \left(\sum_{m=1}^{r} (z_{im} - z_{jm})^2 \right)^{1/2} \right\}$$

$1 \leq k \leq n$, $1 \leq l \leq r$

↑ either i or j must be equal to k, otherwise the derivative vanishes.

2 cases → $i = k < j$
↘ $i < j = k$.

- $i = k < j$ $= \sum_{k=i<j} w_{kj} \delta_{kj} \frac{1}{2} 2 \frac{(z_{kl} - z_{jl})}{d_{kj}(Z)}$
- $i < j = k$ $= \sum_{i<j=k} w_{ik} \delta_{ik} \frac{1}{2} 2 \left(-\frac{(z_{il} - z_{kl})}{d_{ik}(Z)}\right)$

↑ Compare these terms with the (k, l) entry of $B(Z)Z$:

$$B(Z)Z = \sum_{i<j} w_{ij} \delta_{ij} d_{ij}^{-1} A_{ij} Z$$

$A_{ij} Z$ = (l ↓)

$i = k < j$ → $\begin{pmatrix} \cdots & (z_{il} - z_{jl}) & \cdots \\ & \vdots & \\ \cdots & (z_{jl} - z_{il}) & \cdots \end{pmatrix}$ ← j

they are the same !

■

⇒ Gradient descent algorithm for stress minimization:

$$z_{k+1} = z_k - \eta_k \nabla \sigma(z_k)$$
$$= z_k - 2\eta_k \{ V z_k - B(z_k) z_k \}.$$

↑ Compare this iteration with the SMACOF algorithm

$$z_{k+1} = V^+ B(z_k) z_k$$
$$= (z_{k+1} - z_k) + V^+ B(z_k) z_k$$
$$= z_{k+1} - \frac{1}{2} V^+ \{ 2 V z_k - 2 B(z_k) z_k \}$$
$$= \text{Gradient descent algorithm with constant step size.}$$

× Remark: Constrained MDS

From the expression of $\tau(Y, z)$ [(*) p.10] and $\bar{z} := V^+ B(Y) Y$ [(**) p.11], we have

$$\tau(Y, z) = \mathrm{Tr}\{ z^t V z \} - 2 \underbrace{\mathrm{Tr}\{ z^t B(Y) Y \}}_{\mathrm{Tr}\{ z^t V \bar{z} \}} + \text{constant}$$

$$= \mathrm{Tr}\{ (z - \bar{z}) V (z - \bar{z}) \} - \mathrm{Tr}\{ \bar{z} V \bar{z} \} + \text{constant}$$

↑ The minimization of τ handles constraints on z.

E.g: Linear constraints: $z = HW$ → interpretable solution in terms of external variables.

observed covariates ↑ ↑ weights

⇒ $W_{update} = (H^t V H)^{-1} H^t V \bar{z}$.

References:

[1] P.J.F. Groenen & M. van de Velden.
Multidimensional Scaling by Majorization: A Review
Journal of Statistical Software, vol 73, issue 8 (2016).

[2] P.J.F. Groenen & I. Borg.
The Past, Present, and Future of Multidimensional Scaling.
Econometric Institute Report EI 2013-07.