Multi-dimensional Scaling (MDS)

Ke Chen

Department of Computer Science, The University of Manchester

Ke.Chen@manchester.ac.uk

Outline

Background

Motivation and application

Problem Formulation

Generic problem setting and properties

MDS Algorithm

Classical MDS (cMDS) and Stress-based MDS algorithms

Illustrative Example

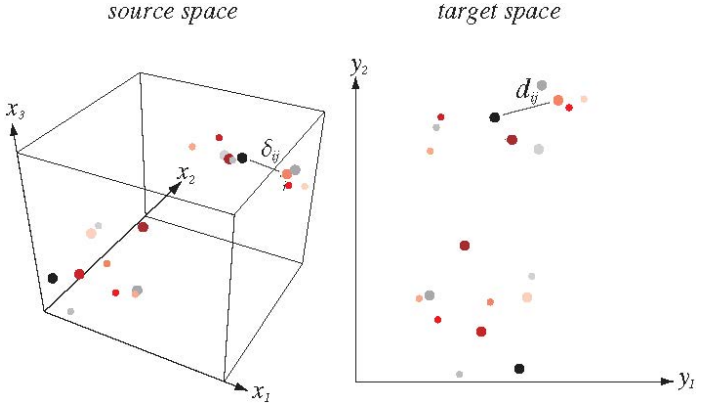
Synthetic data and Airline distance

Extension

Miscellaneous MDS algorithms

Background

- A collection of dimension reduction techniques that map the distance between observations in a high-dimensional source space into a low-dimensional target space
- Learn a configuration of points in target space of which inter-point distance preserves the distance between the corresponding points in source space
- To model the intrinsic manifold for low-dimensional representation and visualisation



Problem Formulation

- Given a distance matrix in d-dimensional source space, $\Delta = (\delta_{ij})_{N\times N}$, we have $\delta_{ii} = 0$, $\delta_{ij} > 0$ and $\delta_{ij} = \delta_{ji}$.
  (Note: MDS only needs to know $\Delta_{N\times N}$ comes from a matrix X of N data points without requiring any information on dimensionality and distance metric in source space.)
- Given a p-dimensional (Euclidean) target space ($p < d$), the distance between two data points, $\boldsymbol{z}_i$ and $\boldsymbol{z}_j$, is measured by $d_{ij} = ||\boldsymbol{z}_i - \boldsymbol{z}_j|| = \sqrt{(\boldsymbol{z}_i - \boldsymbol{z}_j)^T(\boldsymbol{z}_i - \boldsymbol{z}_j)}$.
- Problem: Find out N data points, $Z_{p\times N} = \{\boldsymbol{z}_1, \boldsymbol{z}_2, \cdots, \boldsymbol{z}_N\}$, in p-dimensional space such that $d_{ij} \approx f(\delta_{ij})$ for $i = 1, 2, \cdots, N$; $j = 1, \cdots, N$, where $f(\cdot)$ is a parametric monotonic increasing/decresing function, e.g., $f(\delta_{ij}) = \alpha + \beta\delta_{ij}$.
- To ensure that solution is unique, the constraint $\sum_{i=1}^{N} \boldsymbol{z}_i = \boldsymbol{0}$.
  (Note: without this constraint, if $\boldsymbol{z}_i^*$ and $\boldsymbol{z}_j^*$ are solutions, then so are $\boldsymbol{z}_i^* + \boldsymbol{c}$ and $\boldsymbol{z}_j^* + \boldsymbol{c}$ for any $\boldsymbol{c}$ since $d_{ij} = ||\boldsymbol{z}_i^* - \boldsymbol{z}_j^*|| = ||(\boldsymbol{z}_i^* + \boldsymbol{c}) - (\boldsymbol{z}_j^* + \boldsymbol{c})||$.)
- Most of MDS learning algorithms choose $f(\delta_{ij}) = \delta_{ij}$ such that $d_{ij} \approx \delta_{ij}$.

Classical MDS (cMDS)

- In d-diamensional (Euclidean) source space, a distance matrix, $\Delta = (\delta_{ij})_{N\times N}$, comes from unknown centralised data matrix of N points, $X_{d\times N}$, where $\delta_{ij}^2 = ||\boldsymbol{x}_i - \boldsymbol{x}_j||^2 = (\boldsymbol{x}_i - \boldsymbol{x}_j)^T(\boldsymbol{x}_i - \boldsymbol{x}_j)$.
- For a p-dimensional (Euclidean) target space ($p < d$), the distance between two data points, $\boldsymbol{z}_i$ and $\boldsymbol{z}_j$, is measured by $d_{ij}^2 = ||\boldsymbol{z}_i - \boldsymbol{z}_j||^2 = (\boldsymbol{z}_i - \boldsymbol{z}_j)^T(\boldsymbol{z}_i - \boldsymbol{z}_j)$.
- Problem: Find out N data points, $Z_{p\times N} = \{\boldsymbol{z}_1, \boldsymbol{z}_2, \cdots, \boldsymbol{z}_N\}$, in p-dimensional space to minimise the disparity loss:

$$\mathcal{L}(Z;\Delta) = \sum_{i=1}^{N}\sum_{j=1}^{N}\left(d_{ij}^2 - \delta_{ij}^2\right)^2 \quad \text{s.t.} \sum_{i=1}^{N}\boldsymbol{z}_i = \boldsymbol{0}.$$

- A solution is achieved by means of Euclidean vector space properties.

Classical MDS (cMDS)

- Euclidean vector space property: for a centralised data matrix, X, its inner-product (Gram) matrix, $G_{N\times N} = X^T X$, is expressible by its distance matrix, $\Delta^2 = \left(\delta_{ij}^2\right)_{N\times N}$:

$$G = X^T X = -\frac{1}{2} H \Delta^2 H; \quad H_{N\times N} = I_N - \frac{1}{N} \boldsymbol{e}\boldsymbol{e}^T,$$

where I_N is the identity matrix and $\boldsymbol{e} = [1\ 1 \cdots 1]^T$. In the element-wise notation,

$$g_{ij}^2 = \frac{1}{2}\left[\frac{1}{N}\sum_{k=1}^{N}\delta_{ik}^2 + \frac{1}{N}\sum_{k=1}^{N}\delta_{kj}^2 - \frac{1}{N^2}\sum_{k=1}^{N}\sum_{k=1}^{N}\delta_{kl}^2 - \delta_{ij}^2\right] \implies G = \left(g_{ij}^2\right)_{N\times N}.$$

- By means of the Euclidean vector space property, we also do the same in the target space: $Z^T Z = -\frac{1}{2} H D^2 H$ where $D^2 = \left(d_{ij}^2\right)_{N\times N}$.

Classical MDS (cMDS)

- Thus, the disparity loss can be rewritten as minimum of reconstruction errors:

$$\mathcal{L}(Z;\Delta) = \sum_{i=1}^{N}\sum_{j=1}^{N}\left(\boldsymbol{z}_i^T\boldsymbol{z}_j - \boldsymbol{x}_i^T\boldsymbol{x}_j\right)^2 = ||Z^TZ - X^TX||_F \;\text{ s.t. }\; \sum_{i=1}^{N}\boldsymbol{z}_i = \mathbf{0}.$$

- **Solution**: Conduct spectral decomposition on $G = X^TX$:
  - $G_{N\times N} = V_{N\times N}\Sigma_{N\times N}V_{N\times N}^T = \sum_{i=1}^{N}\lambda_i\boldsymbol{v}_i\boldsymbol{v}_i^T$, where $\boldsymbol{v}_i$ is the ith eigenvector of G corresponding to the eigenvalue, λ_i, and $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_N$.
  - Produce p-dimensional optimal coordinates ($p < d$) by setting $Z^* = V_p^T\Sigma_p^{\frac{1}{2}}$, where Σ_p is a diagonal matrix of top p eigenvalues and V_p is the matrix of the corresponding eigenvectors. In the vector-wise or element-wise notation: $\boldsymbol{z}_i^* = \sqrt{\lambda_i}\boldsymbol{v}_i$ or $z_{ij}^* = \sqrt{\lambda_i}v_{ij}$, $i = 1,\ldots,p$; $j = 1,\ldots,N$.
- Connection to PCA: When X is known, projecting N points with top p PCs achieved from its covariance matrix leads to the same embedding results (configuration) as MDS (up to an arbitrary rotation). Also, PoV is applicable to cMDS for proper p.

Stress-based MDS

- In MDS, many different stress (loss functions) proposed to carry out $d_{ij} \approx f(\delta_{ij})$.
- Commonly-used stress (loss functions): $\mathcal{L}_{ee}, \mathcal{L}_{ff}$ and $\mathcal{L}_{ef}$ w.r.t. $Z_{p\times N} = \{\boldsymbol{z}_1, \cdots, \boldsymbol{z}_N\}$
  - $\mathcal{L}_{ee}$: penalises large absolute errors

$$\mathcal{L}_{ee}(Z;\Delta) = \frac{\sum_{i<j}\left(d_{ij} - f(\delta_{ij})\right)^2}{\sum_{i<j}\delta_{ij}^2} = \frac{\sum_{i<j}\boxed{\left(d_{ij} - f(\delta_{ij})\right)}\boxed{\left(d_{ij} - f(\delta_{ij})\right)}}{\sum_{i<j}\delta_{ij}^2}.$$

  - $\mathcal{L}_{ff}$: penalises large relative errors

$$\mathcal{L}_{ff}(Z;\Delta) = \sum_{i<j}\left(\frac{d_{ij} - f(\delta_{ij})}{\delta_{ij}}\right)^2 = \sum_{i<j}\boxed{\left(\frac{d_{ij} - f(\delta_{ij})}{\delta_{ij}}\right)}\boxed{\left(\frac{d_{ij} - f(\delta_{ij})}{\delta_{ij}}\right)}.$$

  - $\mathcal{L}_{ef}$ (a.k.a. Sammon mapping when $f(\delta_{ij}) = \delta_{ij}$): trade-off between $\mathcal{L}_{ee}$ and $\mathcal{L}_{ff}$

$$\mathcal{L}_{ef}(Z;\Delta) = \frac{1}{\sum_{i<j}\delta_{ij}}\sum_{i<j}\frac{\left(d_{ij} - f(\delta_{ij})\right)^2}{\delta_{ij}} = \frac{1}{\sum_{i<j}\delta_{ij}}\sum_{i<j}\boxed{\left(d_{ij} - f(\delta_{ij})\right)}\boxed{\left(\frac{d_{ij} - f(\delta_{ij})}{\delta_{ij}}\right)}.$$

Stress-based MDS

- Solution: apply an optimisation method to minimise $\mathcal{L}_{ee}$, $\mathcal{L}_{ff}$, and $\mathcal{L}_{ef}$ w.r.t. Z.
- No analytic solution but an iterative method, e.g., stochastic gradient descent (SGD)
  1. Initialise p-dimensional embedded coordinates randomly: $\boldsymbol{z}_1^{(0)}, \boldsymbol{z}_2^{(0)}, \cdots, \boldsymbol{z}_N^{(0)}$
  2. Update $\boldsymbol{z}_k^{(t+1)} \leftarrow \boldsymbol{z}_k^{(t)} - \eta \nabla \mathcal{L}\left(\boldsymbol{z}_k\right)\big|_{\boldsymbol{z}_k = \boldsymbol{z}_k^{(t)}}$ $(0 < \eta < 1)$ for $k = 1, 2, \cdots, N$.
  3. Repeat step 2 until a stopping condition is satisfied.

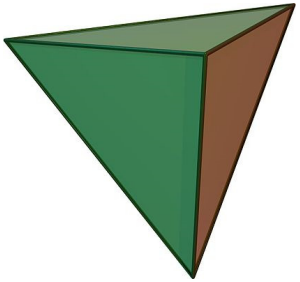
$$\nabla \mathcal{L}_{ee}(\boldsymbol{z}_k) = \frac{2}{\sum_{i<j} \delta_{ij}^2} \sum_{j \neq k} \left(d_{kj} - f\left(\delta_{kj}\right)\right) \frac{\boldsymbol{z}_k - \boldsymbol{z}_j}{d_{kj}},$$

$$\nabla \mathcal{L}_{ff}(\boldsymbol{z}_k) = 2 \sum_{j \neq k} \frac{d_{kj} - f\left(\delta_{kj}\right)}{\delta_{kj}^2} \frac{\boldsymbol{z}_k - \boldsymbol{z}_j}{d_{kj}},$$

$$\nabla \mathcal{L}_{ef}(\boldsymbol{z}_k) = \frac{2}{\sum_{i<j} \delta_{ij}} \sum_{j \neq k} \frac{d_{kj} - f\left(\delta_{kj}\right)}{\delta_{kj}} \frac{\boldsymbol{z}_k - \boldsymbol{z}_j}{d_{kj}},$$

where $d_{kj} = ||\boldsymbol{z}_k - \boldsymbol{z}_j||$. (above gradient calculated on mini-batch in step 2)

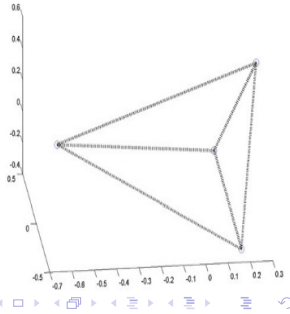
Synthetic dataset: tetrahedron



- Distance matrix (distance between vertices is 1)

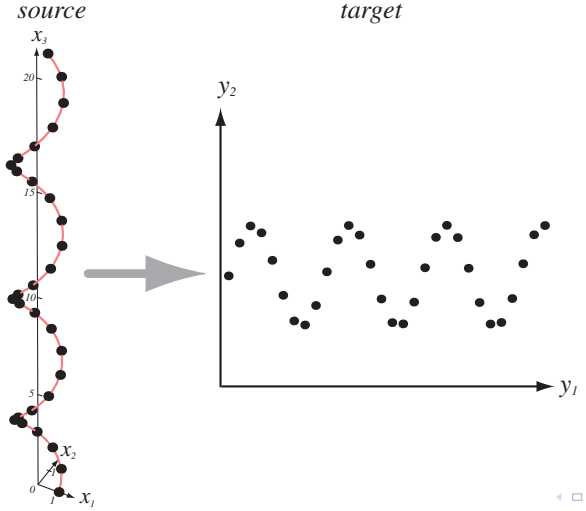
$$\Delta = \begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{bmatrix}$$

- Apply cMDS
  - convert Δ into the Gram matrix: $G_{4\times 4}$
  - Eigen analysis on $G_{4\times 4}$ to achieve eigenvalues $\lambda = 0.5, 0.5, 0.5, 0.0$ and corresponding eigenvectors
  - Using $p = 3$, the tetrahedron retrieved perfectly



Synthetic dataset: 3-D Spiral

- Data generation: $\boldsymbol{x}_k = \left[\cos(\frac{k}{\sqrt{2}})\ \sin(\frac{k}{\sqrt{2}})\ \frac{k}{\sqrt{2}}\right]^T,\ k = 0, 1, \cdots, 29.$
- Calculate data matrix, $\Delta_{30\times 30}$, apply Sammon mapping ($\mathcal{L}_{ef}$) and choose $p = 2$



Airline Distance

- Data: Airline distance between 18 cities, $\Delta_{18\times 18}$ (Source: Altas of the World)

	Beijing	Cape Town	Hong Kong	Honolulu	London	Melbourne
Cape Town	12947					
Hong Kong	1972	11867				
Honolulu	8171	18562	8945			
London	8160	9635	9646	11653		
Melbourne	9093	10338	7392	8862	16902	
Mexico	12478	13703	14155	6098	8947	13557
Montreal	10490	12744	12462	7915	5240	16730
Moscow	5809	10101	7158	11342	2506	14418
New Delhi	3788	9284	3770	11930	6724	10192
New York	11012	12551	12984	7996	5586	16671
Paris	8236	9307	9650	11988	341	16793
Rio de Janeiro	17325	6075	17710	13343	9254	13227
Rome	8144	8417	9300	12936	1434	15987
San Francisco	9524	16487	11121	3857	8640	12644
Singapore	4465	9671	2575	10824	10860	6050
Stockholm	6725	10334	8243	11059	1436	15593
Tokyo	2104	14737	2893	6208	9585	8159

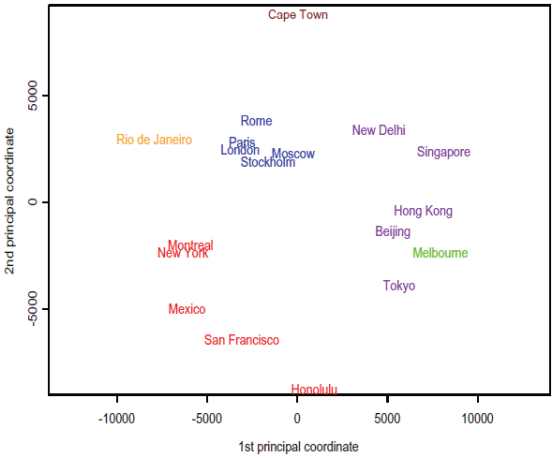
Airline Distance

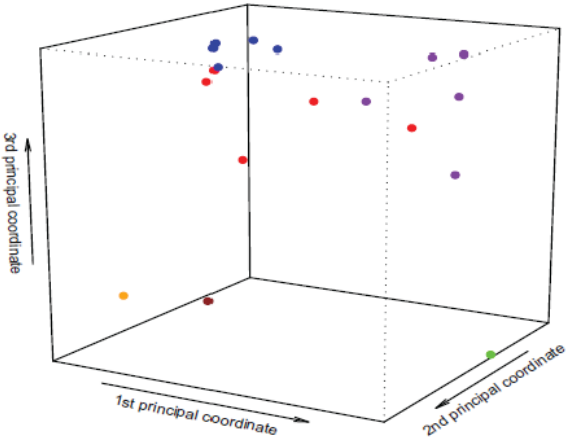
cMDS: 18 eigenvalues/top 3 eigenvectors

	Eigenvalues	Eigenvectors		
1	471582511	0.245	−0.072	0.183
2	316824787	0.003	0.502	-0.347
3	253943687	0.323	−0.017	0.103
4	−98466163	0.044	−0.487	-0.080
5	−74912121	−0.145	0.144	0.205
6	−47505097	0.366	−0.128	−0.569
7	31736348	−0.281	−0.275	−0.174
8	−7508328	−0.272	−0.115	0.094
9	4338497	−0.010	0.134	0.202
10	1747583	0.209	0.195	0.110
11	−1498641	−0.292	−0.117	0.061
12	145113	−0.141	0.163	0.196
13	−102966	−0.364	0.172	−0.473
14	60477	−0.104	0.220	0.163
15	−6334	−0.140	−0.356	−0.009
16	−1362	0.375	0.139	−0.054
17	100	−0.074	0.112	0.215
18	0	0.260	−0.214	0.173

Airline Distance

- Embedding results: $p = 2$ and $p = 3$





Extension

- **Non-metric MDS (nMDS)**: In many real applications of MDS, dissimilarities are known only by their rank order, e.g., alphabetic order of 26 letters in English. Metric MDS is extensible to nMDS.
- **Weighted MDS**: Stress-based metric MDS is extensible to weighted MDS in order to highlight certain aspects of dissimilarities in data.
- **Symbolic MDS**: MDS is extensible to dealing with symbolic data of interval of dissimilarity and distribution (histogram) of dissimilarity instead of distance between two points in metric MDS.
- **Interactive MDS**: An extension of MDS that allows for a completely new, direct, and intuitive interaction with an MDS user on the interplay of the specific dissimilarities in data.

Reference

If you want to deepen your understanding and learn something beyond this lecture, you can self-study the optional references below.

[Alpaydin, 2014] Alpaydin E. (2014): *Introduction to Machine Learning* (3rd Ed.), MIT Press. (Sections 6.6 & 6.7)

[Groenen & Borg, 2014] Groenen P. and Borg I. (2014): The past, present and future of multidimensional scaling. In *Visualization and Verbalization of Data*, Chapman & Hall. (Chapter 7)