FINM 331: DATA ANALYSIS FOR FINANCE AND STATISTICS
FALL 2015
MORE DATA ANALYTIC TOOLS

1. HYPERTEXT INDUCED TOPIC SELECTION

- we discuss the HITS algorithm, arguably the basis behind all modern search engines (e.g. Baidu, Bing, Google)
- we may view this as a special case of CA with several simplications, as a variant of PCA applied to network data, or as a standalone algorithm
- internet search engines perform essentially two tasks: (i) information retrieval, (ii) ranking
  - if you search for a word, say, 'shoe,' it would find all relevant webpages — those containing the word 'shoe' but even those that don't contain it — this is the problem of information retrieval and we will see how to do it with LSI in the next section
  - now the difficult part is how to present these results to a user — if there are three million pages about shoes, how should we order them (Google makes it a mission to return the webpage you are looking for within the top five search results, maximizing something it calls *happiness index*)
  - in the pre-Google days, search engines (e.g. Altavista, Excite, HotBot, Lycos, WebCrawler, etc — all defunct) order their search results essentially by analyzing the content of the webpages (e.g. counting the number of times 'shoe' appears on the page, so shoe merchants game the system by having the word appear $20,000$ times in invisible font on their webpages)
  - Google was the first to use the link structure of the world wide web to rank webpages but this idea appeared first in Jon Kleinberg's HITS, invented when he was an intern at IBM
- Kleinberg's idea is very simple — every webpage has a *hub score* and an *authority score*
- hubs and authorities are defined as follows
  (1) a good hub points to many good authorities
  (2) a good authority is pointed to by many good hubs
- now we quantify these mathematically
- the WWW is modelled as a *directed graph* (V, G)
  - $V = \{1, \dots, n\}$ is the set of vertices and represents all the webpages on the WWW
  - $E \subseteq V \times V$ is the set of directed edges, $(i, j) \in E$ if and only if webpage i points to webpage j
- $n \approx 50$ billion today
- the hub score and authority score of webpage i are denoted by x_i and y_i respectively
- a good hub points to many good authorities, so a plausible definition for the hub score of i would be

$$x_i \propto \sum_{j:(i,j)\in E} y_j$$

- a good authority is pointed to by many good hubs, so a plausible definition for authority score of i would be

$$y_i \propto \sum_{j:(j,i)\in E} x_j$$

Date: November 29, 2015, version 1.1. Comments, bug reports: lekheng@galton.uchicago.edu.

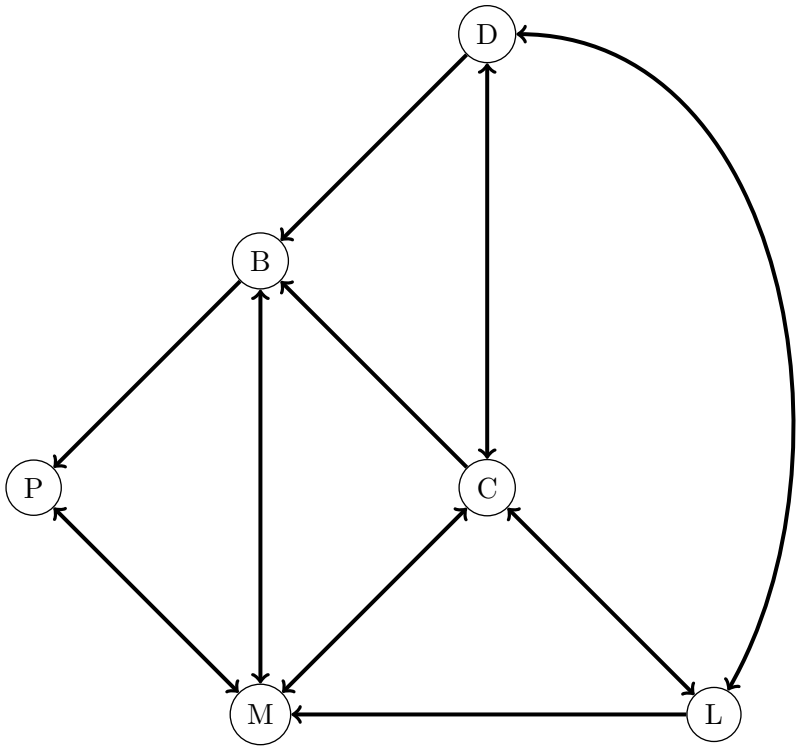

FIGURE 1. B = Baidu, C = Comcast, D = Dell, L = LinkedIn, M = Microsoft, P = Priceline.

- now we just want to use these scores to compare different webpages (higher score ⇒ better webpage) so it is the score relative to one another that matters, so we scale every webpage's score by a constant and require that

$$\|\mathbf{x}\|_2 = 1, \qquad \|\mathbf{y}\|_2 = 1$$

- so this gives us the following definition of hub and authority scores

$$x_i' := \sum_{j:(i,j)\in E} y_j, \qquad x_i := \frac{x_i'}{\|\mathbf{x}'\|_2}, \qquad i = 1, \dots, n,$$

$$y_i' := \sum_{j:(j,i)\in E} x_j, \qquad y_i := \frac{y_i'}{\|\mathbf{y}'\|_2}, \qquad i = 1, \dots, n$$

- our data matrix in this case is the *adjacency matrix* of the graph $A = [a_{ij}] \in \mathbb{R}^{n\times n}$ where

$$a_{ij} = \begin{cases} 1 & \text{if } (i,j) \in E, \\ 0 & \text{if } (i,j) \notin E, \end{cases}$$

  which is always a square matrix ($n = p$) but not symmetric (i points to j does not necessarily imply that j also points to i)
- note that a_{ij} is also a count/frequency, except that its value is either 0 or 1
- the definition of hub and authority scores may be expressed as

$$\mathbf{x}' = A\mathbf{y}, \qquad \mathbf{y}' = A^{\mathsf{T}}\mathbf{x}, \qquad \mathbf{x} = \frac{\mathbf{x}'}{\|\mathbf{x}'\|_2}, \qquad \mathbf{y} = \frac{\mathbf{y}'}{\|\mathbf{y}'\|_2} \tag{1.1}$$

- $\mathbf{x} \in \mathbb{R}^n$ is the vector of hub scores and $\mathbf{y} \in \mathbb{R}^n$ is the vector of authorities scores

- it follows from (1.1) that

$$AA^{\mathsf{T}}\mathbf{x} = \lambda\mathbf{x}, \qquad A^{\mathsf{T}}A\mathbf{y} = \lambda\mathbf{y}$$

- in HITS, we set $\mathbf{x}$ to be a principal eigenvector of AA^{T} and $\mathbf{y}$ to be a prinicipal eigenvector of $A^{\mathsf{T}}A$, i.e.,

$$\lambda = \lambda_{\max}(A^{\mathsf{T}}A) = \lambda_{\max}(AA^{\mathsf{T}}) = \sigma_{\max}(A)^2 = \|A\|_2^2$$

- so $\mathbf{x}$ is a principal left singular vector and $\mathbf{y}$ is a prinicipal right singular vector of A
- we may rank the webpages by the values of either their hub or authority scores, e.g. among all 3 million webpages related to shoes, the top link in your research result would be the webpage i whose y_i is largest, the second link would the the webpage j whose y_j is the second largest, etc
- you may of course also use hub scores x_i instead — the defunct search engine Teoma (acquired by Ask.com) ranked its regular search results using authority scores but included a link called 'resources' that point to related results with the highest hub scores
- Google simplified matters by finding a single score (called PageRank) for each webpage instead of two scores
- we used WWW but such techniques apply to any data of linked structures
- for example, for the graph in Figure 1,

$$A = \begin{array}{c} \\ B \\ C \\ D \\ L \\ M \\ P \end{array} \begin{array}{c} \begin{array}{cccccc} B & C & D & L & M & P \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \end{array}$$

and the hub and authority scores are given by

$$\mathbf{x} = \begin{bmatrix} 0.2501 \\ 0.5884 \\ 0.4459 \\ 0.4495 \\ 0.4005 \\ 0.1730 \end{bmatrix}, \qquad \mathbf{y} = \begin{bmatrix} 0.4938 \\ 0.4460 \\ 0.3572 \\ 0.3560 \\ 0.5028 \\ 0.2239 \end{bmatrix}$$

2. LATENT SEMANTIC INDEXING

- we discuss the LSI algorithm for information retrieval
- it is also known as latent semantic analysis or LSA in natural language processing
- we may view this as a special case of CA with several simplifications, as a variant of PCA applied to text data, or as a standalone algorithm
- we start from any text corpus, i.e., a collection of p documents (books, emails, blogs, tweets) and a collection of n terms (anchor text, keywords, statistically improbable phrases)
- this is processed into a data matrix $X \in \mathbb{R}^{n\times p}$ is called a *term-document matrix* where

$$x_{ij} = \text{number of occurrence of term } i \text{ in document } j$$

is called a *term frequency*

- usually x_{ij} is something more sophisticated — *term frequency–inverse document frequency* or tf–idf for short but we won't go into this minor complication

- we write

$$X = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1p} \\ x_{21} & x_{22} & \cdots & x_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{np} \end{bmatrix} = \begin{bmatrix} \mathbf{t}_1^\mathsf{T} \\ \mathbf{t}_2^\mathsf{T} \\ \vdots \\ \mathbf{t}_n^\mathsf{T} \end{bmatrix} = [\mathbf{d}_1, \mathbf{d}_2, \ldots, \mathbf{d}_p]$$

  where the jth document is abstracted into a vector of frequencies of terms appear in it

$$\mathbf{d}_j = \begin{bmatrix} x_{1j} \\ \vdots \\ x_{nj} \end{bmatrix} \in \mathbb{R}^n$$

  and the ith term is characterized by how often it appears in each of the documents

$$\mathbf{t}_i = \begin{bmatrix} x_{i1} \\ \vdots \\ x_{ip} \end{bmatrix} \in \mathbb{R}^p$$

- so the matrices $X^\mathsf{T}X \in \mathbb{R}^{p\times p}$ and $XX^\mathsf{T} \in \mathbb{R}^{n\times n}$ measure term-term correlation and document-document correlation respectively
- LSI is in essence a low-rank approximation of X
- recall that the Eckart–Young theorem tells us that if

$$X = U\Sigma V^\mathsf{T}$$

  is the SVD of X, then its best rank-r approximation is given by the matrix

$$X_r := U_r \Sigma_r V_r^\mathsf{T}$$

  where $U_r \in \mathbb{R}^{n\times r}$ comprises the first r columns of $U \in \mathbb{R}^{n\times n}$, $V_r \in \mathbb{R}^{p\times r}$ comprises the first r columns of $V \in \mathbb{R}^{p\times p}$, and $\Sigma_r = \operatorname{diag}(\sigma_1, \ldots, \sigma_r) \in \mathbb{R}^{r\times r}$ has diagonal entries given by the first r singular values of X
- in practice,

$$r \ll n, p$$

  and LSI is a *dimension reduction* technique
- LSI works by compressing the document vectors $\mathbf{d}_1, \ldots, \mathbf{d}_n \in \mathbb{R}^p$ and the term vectors $\mathbf{t}_1, \ldots, \mathbf{t}_p \in \mathbb{R}^p$ onto a much lower-dimensional space:

$$\begin{aligned} \widehat{\mathbf{d}}_j &= U_r^\mathsf{T}\mathbf{d}_j \in \mathbb{R}^r, \qquad j = 1, \ldots, p, \\ \widehat{\mathbf{t}}_i &= V_r^\mathsf{T}\mathbf{t}_i \in \mathbb{R}^r, \qquad i = 1, \ldots, n \end{aligned}$$

- we may view $\widehat{\mathbf{d}}_1, \ldots, \widehat{\mathbf{d}}_n \in \mathbb{R}^r$ and $\widehat{\mathbf{t}}_1, \ldots, \widehat{\mathbf{t}}_n \in \mathbb{R}^r$ as digests
- computationally it is much cheaper to work in low-dimensional space

- note that $\mathbf{d}_j = A\mathbf{e}_j$ where $\mathbf{e}_j \in \mathbb{R}^p$ is the jth standard basis vector

$$\begin{aligned}\widehat{\mathbf{d}}_j &= U_r^{\mathsf{T}}\mathbf{d}_j = U_r^{\mathsf{T}}A\mathbf{e}_j = U_r^{\mathsf{T}}U\Sigma V^{\mathsf{T}}\mathbf{e}_j\\ &= U_r^{\mathsf{T}}[U_r, U_{n-r}]\begin{bmatrix}\Sigma_r & \\ & \Sigma_{n-r}\end{bmatrix}V^{\mathsf{T}}\mathbf{e}_j\\ &= [U_r^{\mathsf{T}}U_r, U_r^{\mathsf{T}}U_{n-r}]\begin{bmatrix}\Sigma_r & \\ & \Sigma_{n-r}\end{bmatrix}V^{\mathsf{T}}\mathbf{e}_j\\ &= [I_r, 0]\begin{bmatrix}\Sigma_r & \\ & \Sigma_{n-r}\end{bmatrix}V^{\mathsf{T}}\mathbf{e}_j\\ &= \begin{bmatrix}\Sigma_r & \\ & 0\end{bmatrix}V^{\mathsf{T}}\mathbf{e}_j = \Sigma_r V_r^{\mathsf{T}}\mathbf{e}_j\end{aligned}$$

- in information retrieval, we are given a document query vector $\mathbf{q} \in \mathbb{R}^n$ and we compute

$$\widehat{\mathbf{q}} = U_r^{\mathsf{T}}\mathbf{q} \in \mathbb{R}^r$$

- we then compare $\widehat{\mathbf{q}}$ to $\widehat{\mathbf{d}}_1, \dots, \widehat{\mathbf{d}}_n$ in low-dimensional space $\mathbb{R}^r$ by computing the angles between them

$$\cos\theta_j = \frac{\widehat{\mathbf{d}}_j^{\mathsf{T}}\widehat{\mathbf{q}}}{\|\widehat{\mathbf{d}}_j\|_2\|\widehat{\mathbf{q}}\|_2} \tag{2.1}$$

  and choosing documents that make small angles with $\widehat{\mathbf{q}}$
- an alternative method compares $\mathbf{q} \in \mathbb{R}^n$ directly with the columns of $A_r \in \mathbb{R}^{n\times r}$

$$\cos\phi_j = \frac{(A_r\mathbf{e}_j)^{\mathsf{T}}\mathbf{q}}{\|A_r\mathbf{e}_j\|_2\|\mathbf{q}\|_2} = \frac{(U_r\Sigma_r V_r^{\mathsf{T}}\mathbf{e}_j)^{\mathsf{T}}\mathbf{q}}{\|U_r\Sigma_r V_r^{\mathsf{T}}\mathbf{e}_j\|_2\|\mathbf{q}\|_2} = \frac{(\Sigma_r V_r^{\mathsf{T}}\mathbf{e}_j)^{\mathsf{T}}(U_r^{\mathsf{T}}\mathbf{q})}{\|\Sigma_r V_r^{\mathsf{T}}\mathbf{e}_j\|_2\|\mathbf{q}\|_2} = \frac{\widehat{\mathbf{d}}_j^{\mathsf{T}}\widehat{\mathbf{q}}}{\|\widehat{\mathbf{d}}_j\|_2\|\mathbf{q}\|_2} \tag{2.2}$$

  and choosing documents that make small angles with $\mathbf{q}$
- this method of comparing objects via angular separation in (2.1) and (2.2) is called *cosine similarity*
- we may also query terms instead of documents in the same way
- there are many applications of LSA and LSI

  information retrieval: given a query, find matching documents
  document classification: comparing documents in low-dimensional space
  cross-language retrieval: find similar documents across langugages using a training set comprising a corpus of translated documents[1]
  synonymy and polysemy: find relations between terms[2]

3. MULTIDIMENSIONAL SCALING

- a Euclidean distance matrix (EDM) is defined to be a symmetric matrix of the form

$$D = \begin{bmatrix}d_{11} & d_{12} & \cdots & d_{1n}\\ d_{21} & d_{22} & \cdots & d_{2n}\\ \vdots & \vdots & \ddots & \vdots\\ d_{n1} & d_{n2} & \cdots & d_{nn}\end{bmatrix} \in \mathbb{R}^{n\times n}$$

[1]Standard corpus that have been translated into many languages include the Bible, the Koran, works of Karl Marx. These are usually used as the training set.

[2]Synonym: different words expressing the same idea (e.g. doctors and physicians). Polyseme: same word expressing related ideas (e.g. mole in the sense of a burrowing animal and mole in the sense of a spy). Homonym: same word expressing different ideas (e.g. bark in the sense of the sound a dog makes and bark in the sense of the outer layer of a tree trunk).

where

$$d_{ij}^2 = (\mathbf{x}_i - \mathbf{x}_j)^\mathsf{T}(\mathbf{x}_i - \mathbf{x}_j)$$

and $\mathbf{x}_1, \dots, \mathbf{x}_n \in \mathbb{R}^p$

- as usual we assemble $\mathbf{x}_1, \dots, \mathbf{x}_n \in \mathbb{R}^p$ into a data matrix

$$X = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1p} \\ x_{21} & x_{22} & \cdots & x_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{np} \end{bmatrix} = \begin{bmatrix} \mathbf{x}_1^\mathsf{T} \\ \mathbf{x}_2^\mathsf{T} \\ \vdots \\ \mathbf{x}_n^\mathsf{T} \end{bmatrix} \in \mathbb{R}^{n \times p}$$

- given any data matrix $X \in \mathbb{R}^{n \times p}$ we can easily construct its EDM $D \in \mathbb{R}^{n \times n}$
- the difficulty in MDS is about doing the reverse: given an EDM $D \in \mathbb{R}^{n \times n}$, construct the data matrix $X \in \mathbb{R}^{n \times p}$
- in its most general form, MDS can use any *metric* (i.e., distance) $d : \mathbb{R}^n \times \mathbb{R}^n \to [0, \infty)$ for

$$d_{ij} = d(\mathbf{x}_i, \mathbf{x}_j)$$

for example the p-norm

$$d_{ij} = \|\mathbf{x}_i - \mathbf{x}_j\|_p$$

- for simplicity we shall only discuss the case $p = 2$, i.e., when D is an EDM
- note the EMD of X and the EDM $X - \mathbf{1}_n\mathbf{c}^\mathsf{T}$ are always the same for any $\mathbf{c}^\mathsf{T} = [c_1, \dots, c_p]^\mathsf{T} \in \mathbb{R}^p$

$$X - \mathbf{1}_n\mathbf{c}^\mathsf{T} = \begin{bmatrix} x_{11} - c_1 & x_{12} - c_2 & \cdots & x_{1p} - c_p \\ x_{21} - c_1 & x_{22} - c_2 & \cdots & x_{2p} - c_p \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} - c_1 & x_{n2} - c_2 & \cdots & x_{np} - c_p \end{bmatrix} = \begin{bmatrix} (\mathbf{x}_1 - \mathbf{c})^\mathsf{T} \\ (\mathbf{x}_2 - \mathbf{c})^\mathsf{T} \\ \vdots \\ (\mathbf{x}_n - \mathbf{c})^\mathsf{T} \end{bmatrix} \in \mathbb{R}^{n \times p}$$

- so we can never X exactly from its EDM, at best we could only hope to recover it up to translation by an arbitrary vector
- in view of this, we require an additional condition: X has sample mean

$$\overline{\mathbf{x}} = \frac{1}{n}(\mathbf{x}_1 + \cdots + \mathbf{x}_n) = \frac{1}{n}X^\mathsf{T}\mathbf{1}_n = \mathbf{0},$$

or alternatively,

$$X = X_c = HX$$

where H is the centering matrix

$$H = I - \frac{1}{n}\mathbf{1}_n\mathbf{1}_n^\mathsf{T}$$

- we now define a related matrix $G \in \mathbb{R}^{n \times n}$ of X called the *Gram matrix* or *inner product matrix*

$$G = XX^\mathsf{T} = \begin{bmatrix} \mathbf{x}_1^\mathsf{T}\mathbf{x}_1 & \mathbf{x}_1^\mathsf{T}\mathbf{x}_2 & \cdots & \mathbf{x}_1^\mathsf{T}\mathbf{x}_n \\ \mathbf{x}_2^\mathsf{T}\mathbf{x}_1 & \mathbf{x}_2^\mathsf{T}\mathbf{x}_2 & \cdots & \mathbf{x}_2^\mathsf{T}\mathbf{x}_n \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{x}_n^\mathsf{T}\mathbf{x}_1 & \mathbf{x}_n^\mathsf{T}\mathbf{x}_2 & \cdots & \mathbf{x}_n^\mathsf{T}\mathbf{x}_n \end{bmatrix}$$

- more generally, if X is not already centered, then

$$G = HXX^\mathsf{T}H$$

- note that

$$d_{ij}^2 = (\mathbf{x}_i - \mathbf{x}_j)^\mathsf{T}(\mathbf{x}_i - \mathbf{x}_j) = \sum_{k=1}^{p}(x_{ik} - x_{jk})^2$$

$$g_{ij} = \mathbf{x}_i^\mathsf{T}\mathbf{x}_j = \sum_{k=1}^{p} x_{ik}x_{jk}$$

- from these we deduce the relation

$$d_{ij}^2 = \mathbf{x}_i^\mathsf{T}\mathbf{x}_i + \mathbf{x}_j^\mathsf{T}\mathbf{x}_j - 2\mathbf{x}_i^\mathsf{T}\mathbf{x}_j = g_{ii} + g_{jj} - 2g_{ij} \tag{3.1}$$

- since $X = X_c$,

$$\sum_{i=1}^{n} g_{ij} = 0, \qquad j = 1, \ldots, n,$$

and so if we sum (3.1) over i, over j, and over both i and j, we get

$$\frac{1}{n}\sum_{i=1}^{n} d_{ij}^2 = \left(\frac{1}{n}\sum_{i=1}^{n} g_{ii}\right) + g_{jj}, \tag{3.2}$$

$$\frac{1}{n}\sum_{j=1}^{n} d_{ij}^2 = g_{ii} + \left(\frac{1}{n}\sum_{j=1}^{n} g_{jj}\right), \tag{3.3}$$

$$\frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n} d_{ij}^2 = \frac{2}{n}\sum_{i=1}^{n} g_{ii} \tag{3.4}$$

- forming (3.2) + (3.3) − (3.4), we get

$$g_{ii} + g_{jj} = d_{i\bullet}^2 + d_{\bullet j}^2 - d_{\bullet\bullet}^2 \tag{3.5}$$

where

$$d_{\bullet j}^2 := \frac{1}{n}\sum_{i=1}^{n} d_{ij}^2, \qquad d_{i\bullet}^2 := \frac{1}{n}\sum_{j=1}^{n} d_{ij}^2, \qquad d_{\bullet\bullet}^2 := \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n} d_{ij}^2$$

- plugging (3.5) into (3.1), we getλ

$$g_{ij} = -\frac{1}{2}(d_{ij}^2 - d_{i\bullet}^2 - d_{\bullet j}^2 + d_{\bullet\bullet}^2) \tag{3.6}$$

- so we can obtain the Gram matrix G from the EDM D
- we will now rely on the following theorem

Theorem 1. *Given any symmetric matrix $D = [d_{ij}] \in \mathbb{R}^{n\times n}$, define $G = [g_{ij}] \in \mathbb{R}^{n\times n}$ by (3.6). Then D is an EDM if and ony if G is positive semidefinite. Furthermore, if D is the EDM of $X \in \mathbb{R}^{n\times p}$, then G is the Gram matrix of X.*

- the EVD of G takes the form

$$G = Q\Lambda Q^\mathsf{T}$$

where $Q = [\mathbf{q}_1, \ldots, \mathbf{q}_n] \in \mathbb{R}^{n\times n}$ and $\Lambda = \operatorname{diag}(\lambda_1, \ldots, \lambda_n) \in \mathbb{R}^{n\times n}$

- since $G = XX^\mathsf{T}$ is positive semidefinite, $\lambda_i > 0$ for $i = 1, \ldots, n$, and we may write

$$G = Q\Lambda^{1/2}\Lambda^{1/2}Q^\mathsf{T} = XX^\mathsf{T}$$

where we let

$$X := Q\Lambda^{1/2} = [\sqrt{\lambda_1}\mathbf{q}_1, \ldots, \sqrt{\lambda_n}\mathbf{q}_n]$$

- so we have recovered X from D
- the solution X is clearly not unique

- this method would have to be modified when D is not exactly an EDM
- MDS is used in situations where we have a pairwise comparison of dissimilarity of objects

$$\delta_{ij} = \text{dissimilarity between object } i \text{ and object } j,$$

  giving a matrix $\Delta = [\delta_{ij}] \in \mathbb{R}^{n\times n}$
- we then find an EDM $D \in \mathbb{R}^{n\times n}$ that approximates

$$D \approx \Delta,$$

  say,

$$\min_{D \text{ is an EDM}} \|D - \Delta\|_F^2$$

- this problem appears not just in data analysis but also in areas as diverse as protein folding and facility locations
- MDS can also be used like PCA where we start from a data matrix $X \in \mathbb{R}^{n\times p}$ and form its EDM D but we then try to fit D as the EDM of some data matrix $Y \in \mathbb{R}^{n\times q}$ where $q \ll p$, usually $q = 2$ or 3 like in PCA
- in other words, MDS allows us to squeeze p-dimensional data into 2- or 3-dimensional space for visualization purposes