CS 5785 – Applied Machine Learning – Lec. 14

Prof. Serge Belongie, Cornell Tech
Scribe: Todd Kawakita, Chuyang Wang, Wujing Yao

Oct. 19, 2017 (Under construction)

1 Procrustes Distance

1.1 Motivation

The *Procrustes distance* is a measure of dissimilarity defined on a pair of landmark datasets with known correspondences. For instance, you might have spatial coordinates such as handwriting or GPS traces that you'd like to compare after finding the best alignment between them. Fig. 1 shows an example of some handwriting data that we may wish to compare.

The method is named after a figure from Greek mythology, described as follows:

> In Greek mythology, Procrustes or "the stretcher [who hammers out the metal]," also known as Prokoptas or Damastes "subduer," was a rogue smith and bandit from Attica who physically attacked people by stretching them or cutting off their legs, so as to force them to fit the size of an iron bed. In general, when something is Procrustean, different lengths or sizes or properties are fitted to an arbitrary standard. [Wikipedia]

1.2 Procrustes Alignment

We have two sets of landmarks (or points) stored in the $N \times 2$ matrices $\mathbf{X}_1$ and $\mathbf{X}_2$, respectively. The example from HTF has $N = 96$. We assume the points are *in correspondence*, i.e., the landmark in the ith row of $\mathbf{X}_1$ corresponds to the landmark in the ith row of $\mathbf{X}_2$. In practice, if you were given two point sets like the ones in Fig. 1, you'd need to solve for the correspondences, which in general means finding the permutation between them while dealing with additions, deletions and noise. In this lecture, we put this issue aside and focus instaed on finding the best alignment between the point sets.

The Procrustes alignment is the *rigid transformation* that minimizes the following expression:

$$\min_{\mu,R} \|\mathbf{X}_2 - (\mathbf{X}_1\mathbf{R} + \vec{1}\mu^\top)\|_F$$

$\vec{1}$ is a vector of 1s and the subscript F stands for the Frobenius norm which is given by

$$\|\mathbf{A}\|_F^2 = tr(\mathbf{A}^\top\mathbf{A})$$

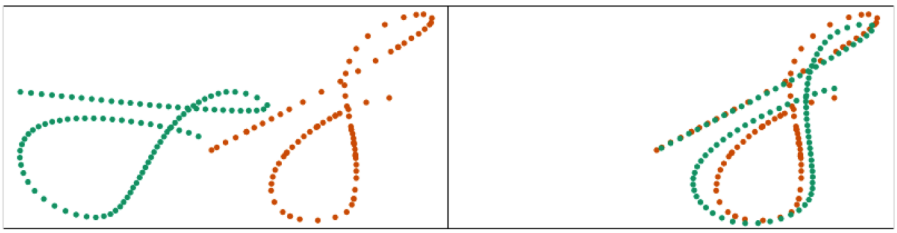

FIGURE
14.25

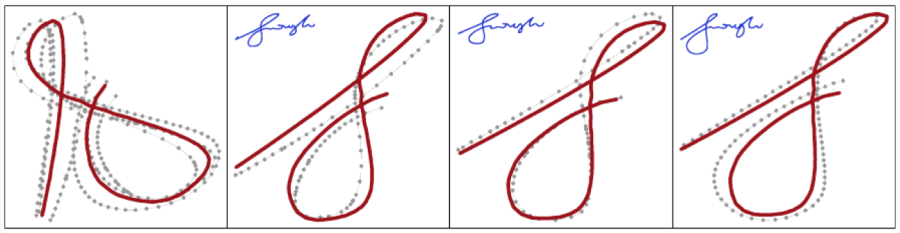

FIGURE
14.26

Figure 1: Procrustes alignment of handwritten symbols. Top: Two sets of landmarks are captured on a digitizing tablet, with set 1 shown in green and set 2 shown in orange. We assume they have known correspondences and we solve for the rigid transformation that best aligns the two point sets.

which is equivalent to the sum of the square of all the entries in $\mathbf{A}$. $\mathbf{R}$ is a 2×2 rotation matrix and μ is a 2×1 translation vector. Since this is a rigid transformation, we don't allow scaling, stretching or shearing. Only rotations and translations are allowed. Generalizations to more flexible transformations, including *similarity* or *affine* are straightforward, but we won't cover them here.

The 2×2 rotation matrix $\mathbf{R}$ is specified by a single parameter θ:

$$\mathbf{R}(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$$

The matrix product $\mathbf{X}_1\mathbf{R}$ applies a rotation of θ to each row of $\mathbf{X}_1$. The outer product $\vec{1}\mu^\top$, which has size $N \times 2$, copies the translation vector N times. Thus the quantity $\mathbf{X}_1\mathbf{R} + \vec{1}\mu^\top$ represents the rotated and translated version of $\mathbf{X}_1$ that we wish to be as close as possible to $\mathbf{X}_2$.

1.3 Solution

First compute the means of each point set and call them $\bar{x}_1$ and $\bar{x}_2$. Let $\tilde{\mathbf{X}}_1, \tilde{\mathbf{X}}_2$ denote centered (i.e., mean-subtracted) versions of $\mathbf{X}_1$ and $\mathbf{X}_2$. That leaves us to solve for $\mathbf{R}$, or equivalently, θ. Now compute the SVD of the inner product of the centered matrices:

$$\tilde{\mathbf{X}}_1^\top \tilde{\mathbf{X}}_2 = \mathbf{U}\mathbf{D}\mathbf{V}^\top$$

One can show the solution for $\mathbf{R}$ is given by:

$$\hat{\mathbf{R}} = \mathbf{U}\mathbf{V}^\top, \quad \hat{\mu} = \bar{x}_2 - \hat{\mathbf{R}}\bar{x}_1$$

In this expression for $\mathbf{R}$, we effectively set $\mathbf{D} = \mathbf{I}$ in the SVD which yields the rotation $\mathbf{R}$ closest to the best affine (linear) alignment. We skip the proof, which is based on differentiating the cost function w.r.t. θ.

The singular values appearing on the diagonal of $\mathbf{D}$ represent stretching along the x and y axes. We know that $\mathbf{U}$ and $\mathbf{V}$ are orthogonal, which means we can interpret them as rotation matrices. When operating on a 2D vector, the product of the three matrices $\mathbf{U}\mathbf{D}\mathbf{V}^\top$ can be regarded as the following sequence of operations: rotate by some angle, scale by some amount horizontally and vertically, rotate by a second angle. Forcing $\mathbf{D}$ to be the identity matrix eliminates the stretching part and just leaves the rotation part.

Procrustes alignment is often used as a component of shape-based recognition systems.

2 Classical Multidimensional Scaling (MDS)

Fig. 2 shows a table of dissimilarities between 11 countries obtained via a political science survey. As an example, the low value of 2.17 for FRA-BEL indicates that the respondents regarded France and Belgium as highly similar.

If we wanted to perform clustering on this dataset, we can't use K-means here because we don't have direct access to vector representations of the countries. As discussed earlier, we could instead use K-medoids to cluster the data. But what if we actually want to estimate a vectorial representation for the data? That is the goal of MDS.

TABLE 14.3. *Data from a political science survey: values are average pairwise dissimilarities of countries from a questionnaire given to political science students.*

	BEL	BRA	CHI	CUB	EGY	FRA	IND	ISR	USA	USS	YUG
BRA	5.58										
CHI	7.00	6.50									
CUB	7.08	7.00	3.83								
EGY	4.83	5.08	8.17	5.83							
FRA	2.17	5.75	6.67	6.92	4.92						
IND	6.42	5.00	5.58	6.00	4.67	6.42					
ISR	3.42	5.50	6.42	6.42	5.00	3.92	6.17				
USA	2.50	4.92	6.25	7.33	4.50	2.25	6.33	2.75			
USS	6.08	6.67	4.25	2.67	6.00	6.17	6.17	6.92	6.17		
YUG	5.25	6.83	4.50	3.75	5.75	5.42	6.08	5.83	6.67	3.67	
ZAI	4.75	3.00	6.08	6.67	5.00	5.58	4.83	6.17	5.67	6.50	6.92

Figure 2: Table of dissimilarities

Given an input of $N \times N$ dissimilarity matrix D, MDS provides us a set of N k-dimensional points that could have given rise to D. In principle, one could approach this problem by a succession of triangulation steps to solve for the relative location of all the points. The SVD provides us a more elegant solution.

As described in Ripley (1995), we start by converting the dissimilarity matrix into an inner product matrix. For any symmetric matrix $\mathbf{T}$, we can define $\mathbf{T}'$ as follows:

$$\mathbf{T}' = -\frac{1}{2}\left[\mathbf{T} - \frac{(\mathbf{T}\vec{1})\vec{1}^\top}{N} - \frac{\vec{1}(\mathbf{T}\vec{1})^\top}{N} + \frac{\vec{1}^\top\mathbf{T}\vec{1}}{N^2}\right]$$

where $\vec{1}$ is a length N column vector of $1s$. Pre- or post-multiplication by a vector of 1s is a linear algebraic trick for summing the rows or columns of a matrix, respectively. Sandwiching $\mathbf{T}$ in the form $\vec{1}^\top\mathbf{T}\vec{1}$ simply adds up all its entries.

The resulting matrix $\mathbf{T}'$ has the following properties:

1. If $\mathbf{T}$ was formed by computing pairwise Euclidean distances on the x_is, then $\mathbf{T}'$ contains the inner product between the x_is, i.e. $\mathbf{T}' = \mathbf{X}\mathbf{X}^\top$.

2. We can use the SVD to find the matrix square root of $\mathbf{T}'$, i.e., solve for $\mathbf{X}$ such that $\mathbf{T}' = \mathbf{X}\mathbf{X}^\top$, and the rows of $\mathbf{X}$ will contain the coordinates we seek.

The proof of this is based on the following observation. Recalling that $\|a\|^2 = a^\top a$, the squared distance between two points x_i and x_j is expressed as

$$\|x_i - x_j\|^2 = (x_i - x_j)^\top(x_i - x_j) = \|x_i\|^2 + \|x_j\|^2 - 2x_i^\top x_j$$

The transformation of $\mathbf{T}$ into $\mathbf{T}'$ effectively subtracts off the two terms corresponding to the norms of x_i and x_j and just leaves us with the inner product term, $x_i^\top x_j$.

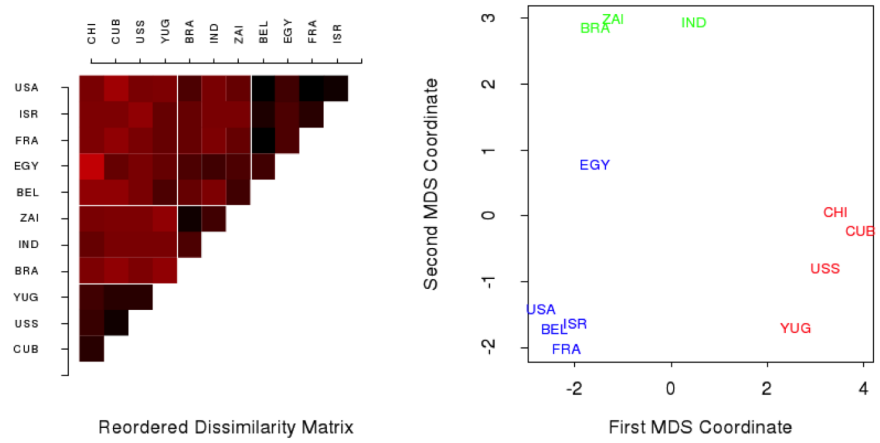

Figure 3: Classical MDS applied to country dissimilarity data.

Because $\mathbf{T}'$ can be expressed as $\mathbf{X}\mathbf{X}^\top$, it is positive semidefinite, which means all of its eigenvalues are nonnegative. This also means we can interpret it as a covariance matrix. As a result, MDS has strong conceptual links to PCA.

Fig. 3 shows the result of MDS applied to the country dissimilarity data using the first two recovered coordinates. It's up to us to interpret of the axes in this plot, e.g., one axis might capture capitalist vs. communist and the other rich vs. poor.

As a more concrete example, suppose you are given a distance matrix (in miles) between 9 cities, as shown in Fig. 4, and you want to construct a map (2D, in this case) of the city locations that are consisten with this distance matrix. We approach this with the caveat that the reconstruction (or embedding or coordinatization) we find will, in general, be related to the original city locations by an unknown rotation, translation and reflection, since those operations don't affect the pairwise distances.

		1	2	3	4	5	6	7	8	9
		BOST	NY	DC	MIAM	CHIC	SEAT	SF	LA	DENV
1	BOSTON	0	206	429	1504	963	2976	3095	2979	1949
2	NY	206	0	233	1308	802	2815	2934	2786	1771
3	DC	429	233	0	1075	671	2684	2799	2631	1616
4	MIAMI	1504	1308	1075	0	1329	3273	3053	2687	2037
5	CHICAGO	963	802	671	1329	0	2013	2142	2054	996
6	SEATTLE	2976	2815	2684	3273	2013	0	808	1131	1307
7	SF	3095	2934	2799	3053	2142	808	0	379	1235
8	LA	2979	2786	2631	2687	2054	1131	379	0	1059
9	DENVER	1949	1771	1616	2037	996	1307	1235	1059	0

For instance, given the matrix of distances among cities shown above, MDS produces this map:

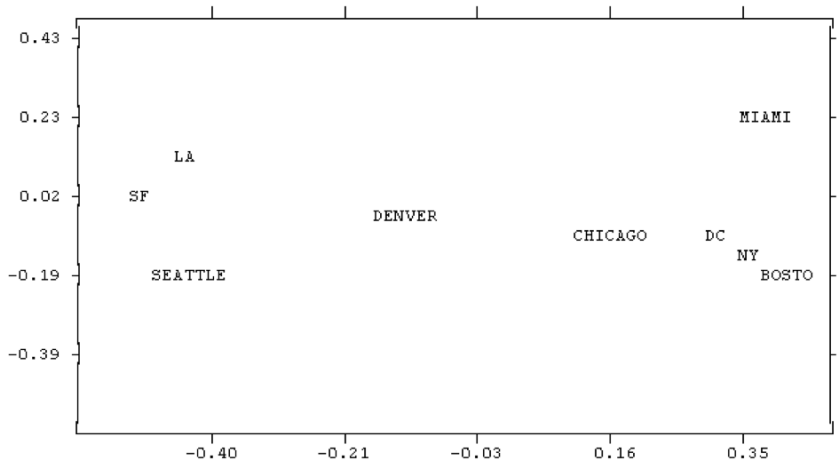

Figure 4: MDS applied to a mileage chart. The resulting 2D embedding is quite reasonable, apart from a north-south reflection. [http://www.analytictech.com/borgatti/mds.htm]