Classical Scaling via Torgerson and Guttman

Torgerson Classical Scaling is based on the following result:

Let $\mathbf{A}$ be a symmetric matrix of order $n \times n$. Suppose we want to find a matrix $\mathbf{B}$ of rank 1 (of order $n \times n$) in such a way that the sum of the squared discrepancies between the elements of $\mathbf{A}$ and the corresponding elements of $\mathbf{B}$ (i.e., $\sum_{j=1}^{n} \sum_{i=1}^{n} (a_{ij} - b_{ij})^2$) is at a minimum. It can be shown that the solution is $\mathbf{B} = \lambda \mathbf{k}\mathbf{k}'$ (so all columns in $\mathbf{B}$ are multiples of $\mathbf{k}$), where λ is the largest eigenvalue of $\mathbf{A}$ and $\mathbf{k}$ is the corresponding normalized eigenvector. This theorem can be generalized. Suppose we take the first r largest eigenvalues and the corresponding normalized eigenvectors. The eigenvectors are collected in an $n \times r$ matrix $\mathbf{K} = \{\mathbf{k}_1, \ldots, \mathbf{k}_r\}$ and the eigenvalues in a diagonal matrix $\mathbf{\Lambda}$. Then $\mathbf{K\Lambda K}'$ is an $n \times n$ matrix of rank r and is a least-squares solution for the approximation of $\mathbf{A}$ by a matrix of rank r. It is assumed, here, that the eigenvalues are all positive. If $\mathbf{A}$ is of rank r by itself and we take the r eigenvectors for which the eigenvalues are different from zero collected in a matrix $\mathbf{K}$ of order $n \times r$, then $\mathbf{A} = \mathbf{K\Lambda K}'$. Note that $\mathbf{A}$ could also be represented by $\mathbf{A} = \mathbf{LL}'$, where $\mathbf{L} = \mathbf{K\Lambda^{1/2}}$ (we factor the matrix), or as a sum of r $n \times n$ matrices — $\mathbf{A} = \lambda_1 \mathbf{k_1 k_1'} + \cdots + \lambda_r \mathbf{k_r k_r'}$.

Metric Multidimensional Scaling – Torgerson's Model (Gower's Principal Coordinate Analysis)

Suppose I have a set of n points that can be perfectly represented spatially in r dimensional space. The i^{th} point has coordinates $(x_{i1}, x_{i2}, \ldots, x_{ir})$. If $d_{ij} = \sqrt{\sum_{k=1}^{r} (x_{ik} - x_{jk})^2}$ represents the Euclidean distance between points i and j, then

$$d_{ij}^* = \sum_{k=1}^{r} x_{ik} x_{jk}, \text{where}$$

$$d_{ij}^* = -\frac{1}{2}(d_{ij}^2 - A_i - B_j + C); \qquad (1)$$

$$A_i = (1/n) \sum_{j=1}^{n} d_{ij}^2;$$

$$B_j = (1/n) \sum_{i=1}^{n} d_{ij}^2;$$

$$C = (1/n^2) \sum_{i=1}^{n} \sum_{j=1}^{n} d_{ij}^2.$$

Note that $\{d_{ij}^*\}_{n \times n} = \mathbf{X}\mathbf{X}'$, where $\mathbf{X}$ is of order $n \times r$ and the entry in the i^{th} row and k^{th} column is x_{ik}.

So, the Question: If I give you $\mathbf{D} = \{d_{ij}\}_{n \times n}$, find me *a* set of coordinates to do it. The Solution: Find $\mathbf{D}^* = \{d_{ij}^*\}$, and take its Spectral Decomposition. This is *exact* here.

To use this result to obtain a spatial representation for a set of n objects given *any* "distance-like" measure, p_{ij}, between objects i and j, we proceed as follows:

(a) Assume (i.e., pretend) the Euclidean model holds for p_{ij}.

(b) Define p_{ij}^* from p_{ij} using (1).

(c) Obtain a spatial representation for p_{ij}^* using a suitable value for r, the number of dimensions (at most, r can be no larger than the number of positive eigenvalues for $\{p_{ij}^*\}_{n \times n}$):

$$\{p_{ij}^*\} \approx \mathbf{X}\mathbf{X}'$$

(d) Plot the n points in r dimensional space.

A Well Known General Optimization Result Justifying the Guttman Scaling Method (The Courant-Fischer Theorem)

For an $n \times n$ symmetric matrix $\mathbf{A}$, let $\lambda_1 \geq \cdots \geq \lambda_n$ be its eigenvalues, and $\mathbf{P}_1, \ldots, \mathbf{P}_n$ the corresponding eigenvectors. Then,

$$\mathbf{A} = \lambda_1 \mathbf{P}_1 \mathbf{P}_1^{'} + \cdots + \lambda_n \mathbf{P}_n \mathbf{P}_n^{'}$$

and

$$\mathbf{I} = \mathbf{P}_1 \mathbf{P}_1^{'} + \cdots + \mathbf{P}_n \mathbf{P}_n^{'}$$

The maximum of

$$\frac{\mathbf{X}'\mathbf{A}\mathbf{X}}{\mathbf{X}'\mathbf{X}}$$

is λ_1, and obtained at $\mathbf{P}_1$; the minimum is λ_n, and obtained at $\mathbf{P}_n$; the maximum subject to being a solution orthogonal (i.e., inner products are zero) to $\mathbf{P}_1, \ldots, \mathbf{P}_k$ is λ_{k+1}, and is obtained for $\mathbf{P}_{k+1}$.

A Guttman Result

If $\mathbf{B}$ is a symmetric matrix of order n, having all its elements non-negative, the following quadratic form defined by the matrix $\mathbf{A}$ must be positive semi-definite:

$$\sum_{i<j} b_{ij}(x_i - x_j)^2 = \sum_{i,j} x_i a_{ij} x_j,$$

where

$$a_{ij} = \begin{cases} \sum_{k=1;k\neq i}^{n} b_{ik} & (i = j) \\ -b_{ij} & (i \neq j) \end{cases}$$

It can be shown that if all elements of $\mathbf{B}$ are positive, then $\mathbf{A}$ is of rank $n-1$, and has one smallest eigenvalue equal to zero with an associated eigenvector having all constant elements. Because all (other) eigenvectors must be orthogonal to the constant eigenvector, this gives a (normalizing) property that the entries in these other eigenvectors sum to zero.

The Guttman metric multidimensional scaling method uses the eigenvectors corresponding to the largest eigenvalues of $\mathbf{A}$ for the axes of the scaling. Each axis is normalized so the sum of the squared entries is equal to the corresponding eigenvalue. As a method of multidimensional scaling (mds), this Guttman result is one that seems to get reinvented periodically in the literature. Generally, this method has been used to provide rational starting points in iteratively-defined nonmetric mds. But more recently, the Guttman strategy (although not attributed to him as such) has been applied to graphs and the corresponding 0/1 adjacency matrix (treated as a similarity measure). In this case, we have what are called Laplacian eigenmaps, where the graphs are imbedded into a space by using the coordinates from the *smallest* nonzero eigenvectors.

What follows are M-files that provide both a Torgerson and a Guttman scaling of a proximity matrix (assumed keyed as a dissimilarity matrix):

Torgerson Scaling:

```
function [coordinates,vaf,vectors,roots] = torgerson(prox,numdim)

n = size(prox,1);

dstar = zeros(n,n);

a = (1/n)*(sum(prox.^2,2));
b = (1/n)*(sum(prox.^2,1));
c = (1/(n*n))*(sum(sum(prox.^2,2)));

for i = 1:n
    for j = 1:n

        dstar(i,j) = (-1/2)*((prox(i,j)^2) - a(i) - b(j) + c);
    end
end

[vectors,roots] = eig(dstar);

[vals,inds] = sort(diag(roots));

inds = flipud(inds);
vals = flipud(vals);

vectors = vectors(:,inds);
roots = diag(vals);
```

```
coordinates = vectors(:,1:numdim)*diag(sqrt(vals(1:numdim)));

distances = pdist(coordinates);

proximities = squareform(prox);

corrmatrix = corrcoef(distances',proximities');

vaf = corrmatrix(1,2)^2;
```

Guttman Scaling:

```
function [coordinates,vaf,vectors,roots] = guttman(prox,numdim)

n = size(prox,1);

amatrix = zeros(n,n);

proxsums = sum(prox);

for i = 1:n
    for j = 1:n

        if(i == j)
            amatrix(i,j) = proxsums(i);
        else
            amatrix(i,j) = -prox(i,j);
        end

    end
end
```

```
[vectors,roots] = eig(amatrix);

[vals,inds] = sort(diag(roots));

inds = flipud(inds);
vals = flipud(vals);

vectors = vectors(:,inds);
roots = diag(vals);

coordinates = vectors(:,1:numdim)*diag(sqrt(vals(1:numdim)));

distances = pdist(coordinates);

proximities = squareform(prox);

corrmatrix = corrcoef(distances’,proximities’);

vaf = corrmatrix(1,2)^2;
```

An example

Ekman produced a data set among 14 colors that varied primarily in hue, by presenting the colors to subjects two at a time and asking for a rating of “similarity” on a five-step scale. Ekman averaged the ratings over subjects and transformed these to range from 0 (for “no similarity at all”) to 1 (for “identity”). The Ekman data are given below, along with the wavelengths for each of the colors (and as `ekman.dat` at `http://cda.psych.uiuc.edu/594_datafiles`).

Wavelength	Label														
674(red)	a	1.0	.86	.42	.48	.18	.06	.07	.04	.02	.07	.09	.12	.13	.16
651	b	.86	1.0	.50	.44	.22	.09	.07	.07	.02	.04	.07	.11	.13	.14
628	c	.42	.50	1.0	.81	.47	.17	.10	.08	.02	.01	.02	.01	.05	.03
610	d	.48	.44	.81	1.0	.54	.25	.10	.09	.02	.01	.00	.01	.02	.04
600	e	.18	.22	.47	.54	1.0	.61	.31	.26	.07	.02	.02	.01	.02	.00
584(yellow)	f	.06	.09	.17	.25	.61	1.0	.62	.45	.14	.08	.02	.02	.02	.01
555	g	.07	.07	.10	.10	.31	.62	1.0	.73	.22	.14	.05	.02	.02	.00
537	h	.04	.07	.08	.09	.26	.45	.73	1.0	.33	.19	.04	.03	.02	.02
504(green)	i	.02	.02	.02	.02	.07	.14	.22	.33	1.0	.58	.37	.27	.20	.23
490	j	.07	.04	.01	.01	.02	.08	.14	.19	.58	1.0	.74	.50	.41	.28
472(blue)	k	.09	.07	.02	.00	.02	.02	.05	.04	.37	.74	1.0	.76	.62	.55
465	l	.12	.11	.01	.01	.01	.02	.02	.03	.27	.50	.76	1.0	.85	.68
445	m	.13	.13	.05	.02	.02	.02	.02	.02	.20	.41	.62	.85	1.0	.76
434(violet)	n	.16	.14	.03	.04	.00	.01	.00	.02	.23	.28	.55	.68	.76	1.0

We carry out a two-dimensional scaling of these 14 colors below [using the transformation to"distance-like measures" of -1.0(similarity value) +1.0]. The various commands and results using MATLAB are given on the pages to follow, including a set of commands to produce two-dimensional plots based on both the Guttman and Torgerson ideas discussed above, and a Procrustes comparison of the two representations.

```
load ekman.dat
```

```
prox = ((-1.0)*ekman) + 1.0;
```

```
n = size(prox,1);
```

```
[torg_coordinates,torg_vaf,torg_vectors,torg_roots] = torgerson(prox,2);

[gutt_coordinates,gutt_vaf,gutt_vectors,gutt_roots] = guttman(prox,2);

torg_vaf

gutt_vaf

for i = 1:n
    objectlabels{i,1} = int2str(i);
end

figure(1)

axis square

plot(torg_coordinates(:,1),torg_coordinates(:,2),'ko')

hold on

text(torg_coordinates(:,1),torg_coordinates(:,2),objectlabels,'fontsize',
    10,'verticalalignment','bottom')

figure(2)

axis square

plot(gutt_coordinates(:,1),gutt_coordinates(:,2),'ko')

hold on

text(gutt_coordinates(:,1),gutt_coordinates(:,2),objectlabels,'fontsize',
    10,'verticalalignment','bottom')
```

```
t_coord = [torg_coordinates(:,1),torg_coordinates(:,2)]

g_coord = [gutt_coordinates(:,1),gutt_coordinates(:,2)]

[d,z,transform] = procrustes(t_coord,g_coord);

figure(3)

axis square

plot(t_coord(:,1),t_coord(:,2),'rx',...
g_coord(:,1),g_coord(:,2),'b.',z(:,1),z(:,2),'go')

hold on

text(torg_coordinates(:,1),torg_coordinates(:,2),objectlabels,
'fontsize',8,'verticalalignment','bottom')

text(z(:,1),z(:,2),objectlabels,'fontsize',8,
'verticalalignment','bottom')

transform(1).b
transform(1).T
transform(1).c

corrcoef([t_coord,g_coord])
```

Exercises:

a) Redo the Ekman analyses to make sure the plots you can obtain are like the ones given. Also, you can see what you are getting with some color.

b) Replicate the analyses done in (a) with the Lincoln Mapquest data (I put the Mapquest data (`lincoln_eighth_circuit_time.dat`) at the same website referenced above for the Ekman data. When you do the objects labels, try:

```
object_labels = char('Urbana', 'Danville', 'Paris', 'Shelbyville',
'Sullivan', 'Decatur', 'Taylorville', 'Springfield', 'Pekin',
'Metamora', 'Bloomington', 'Mt. Pulaski', 'Clinton', 'Monticello');
```

Does it appear that the Guttman scaling is “as good” as the Torgerson scaling? What are your indications? And did you see any of this difference with the Ekman data.