GRAPH LAYOUT TECHNIQUES AND MULTIDIMENSIONAL DATA ANALYSIS

JAN DE LEEUW
UNIVERSITY OF CALIFORNIA, LOS ANGELES

GEORGE MICHAILIDIS
THE UNIVERSITY OF MICHIGAN

ABSTRACT. In this paper we explore the relationship between multivariate data analysis and techniques for graph drawing or graph layout. Although both classes of techniques were created for quite different purposes, we find many common principles and implementations. We start with a discussion of the data analysis techniques, in particular multiple correspondence analysis, multidimensional scaling, parallel coordinate plotting, and seriation. We then discuss parallels in the graph layout literature.

1. DATA AND GRAPHS

The amount of data and information collected and retained by organizations and businesses is constantly increasing, due to advances in data collection, computerization of transactions and breakthroughs in storage technology. Typically, the applications involve large-scale information banks, such as data warehouses ranging in size into terabytes, that contain interrelated data from a number of sources (e.g. customer and product databases). In order to extract useful information from such large datasets, it is necessary to be able to *identify* patterns, trends and relationships in the data and *visualize* their global structure to facilitate decision making. Graph theoretical concepts are capable of capturing complicated structures and relationships in both numerical and categorical data. In this paper we explore the relationship between multivariate data analysis and techniques for graph drawing or graph layout, and examine how coupling ideas from these two fields can lead to new and improved methodology and tools for mining large databases and presentation of large datasets.

1.1. Multivariables and Coding.

The data structure we are interested in consists of n observations on m categorical variables, where variable j has k_j categories (possible values). Using categorical variables causes no real loss of generality: so-called *continuous* variables are merely categorical variables with a large number of numerical categories. We use K for the total number of categories over all variables ($K = \sum_{j=1}^{m} k_j$).

In somewhat more abstract terminology, which generalizes to (theoretical) situations in which we have an infinite number of objects or variables, a *variable* is a function ϕ_j defined on the set of objects $\mathcal{I}$ and with range ∇_j ($\phi_j : \mathcal{I} \mapsto \nabla_j$). A continuous variable, for example, has domain $\mathcal{I} = \{1, 2, \ldots, n\}$ and range $\nabla_j = \mathbb{R}$. A *multivariable* $\Phi = \{\phi_j\}_{j \in \mathcal{J}}$ is a set of variables with the same *domain* $\mathcal{I}$ but with different *ranges* ∇_j.

In the finite case the variables are coded as m *indicator matrices* or *dummies* G_j, where G_j is a binary $n \times k_j$ matrix with exactly one non-zero element in each row i (indicating in which category of variable j object i falls). The $n \times K$ matrix $G = (G_1 | \ldots | G_m)$, which codes the multivariable, is called the *indicator supermatrix*. Obviously the representation of the data by the indicator supermatrix implies no loss of information, since all the original classifications are still present.

It must be emphasized that the coding is unique, given the definition of the multivariable. But in defining the multivariable, many choices must be made. Suppose, for instance, that the ranges ∇_j are the same for all j, with k elements. Then we can define a single variable ϕ on the new domain $\mathcal{I} \otimes \mathcal{J}$ with range ∇. This amounts to stacking the indicator matrices on top of each other. Or, even if the ranges are all different, we can define a single variable ϕ with domain $\mathcal{I}$ and range $\nabla_1 \otimes \ldots \otimes \nabla_m$, which amounts to coding the m variables *interactively*, so that each profile (cell) corresponds to a category of the interactive variable.

In the sequel we assume all these coding decisions have been made, and consequently we deal with a single indicator supermatrix G.

1.2. **Graphs.** One can represent all information in the data by a single bipartite graph[1] with $n + K$ vertices and nm edges. We call such an object the *multivariable graph*. Each edge connects an object and a category. Thus, all the n vertices associated with the objects have degree m, while the K vertices associated with the categories have varying degrees, equal to the number of objects in the category. In Figure 1 the multivariable graph of a toy example corresponding to a 4×3 contingency table with 7 objects is shown. The indicator supermatrix G is related

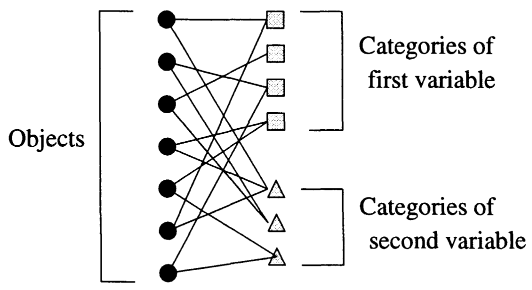

FIGURE 1. The multivariable graph of a toy example

[1] A bipartite graph is a 2-layered graph, where edges only go from one layer to the other layer.

in a very simple way to the *adjacency matrix* A of the graph. In fact,

$$A = \begin{bmatrix} 0 & G \\ G' & 0 \end{bmatrix} \tag{1.1}$$

1.3. **Example.** The same representation of a multivariate data structure is also used in *formal concept analysis* (FCA) [22]. This is a data analysis method which is quite popular in Germany. Multivariables are called a *many-valued contexts*. In this paper we shall use an example taken from the formal concept analysis literature [50]. Table 1 describes 21 sleeping bags in terms of five variables[2].

Sleeping Bag	**Fabricate**	**Temperature T**	**Weight W**	bf Price	**Material**
One Kilo Bag	Wolfskin	7^o C	940 g	149, –	Liteloft
Sund	Kodiak	3^o C	1880 g	139, –	Hollow Fiber
Kompakt Basic	Ajungilak	0^o C	1280 g	249, –	MTI Loft
Finmark Tour	Finmark	0^o C	1750 g	179, –	Hollow Fiber
Interlight Lyx	Caravan	0^o C	1900 g	239, –	Thermolite
Kompakt	Ajungilak	-3^o C	1490 g	299, –	MTI Loft
Touch the Cloud	Wolfskin	-3^o C	1550 g	299, –	Liteloft
Cat's Meow	The North Face	-7^o C	1450 g	339, –	Polarguard
Igloo Super	Ajungilak	-7^o C	2060 g	279, –	Terraloft
Donna	Ajungilak	-7^o C	1880 g	349, –	MTI Loft
Tyin	Ajungilak	-15^o C	2100 g	399, –	Ultraloft
Travellers Dream	Yeti	3^o C	970 g	379, –	Goose-downs
Yeti Light	Yeti	3^o C	800 g	349, –	Goose-downs
Climber	Finmark	-3^o C	1690 g	329, –	Duck-downs
Viking	Warmpeace	-3^o C	1200 g	369, –	Goose-downs
Eiger	Yeti	-3^o C	1500 g	419, –	Goose-downs
Climber light	Finmark	-7^o C	1380 g	349, –	Goose-downs
Cobra	Ajungilak	-7^o C	1460 g	449, –	Duck-downs
Cobra Comfort	Ajungilak	-10^o C	1820 g	549, –	Duck-downs
Foxfire	The North Face	-10^o C	1390 g	669, –	Goose-downs
Mont Blanc	Yeti	-15^o C	1800 g	549, –	Goose-downs

TABLE 1. Many-valued Context: *Sleeping Bags*

From the many-valued contexts FCA derives *formal contexts*, which are defined by binary variables derived from the many-valued contexts. From the many-valued sleeping bag context [50] derives a context by using a *terminology* for logical scaling. The terminology is given in Table 2[3]. This results in Table 3, which actually displays the indicator supermatrix for this example. There are indicator matrices G_j for Price, Fiber, and Quality. From here on, FCA goes its own (abstract algebraic) way. We are merely interested in the (critical) step to derive formal contexts

[2]Note that one could consider the bag's name as a sixth variable.

[3]Prediger used a terminology which defined "good" in such a way that it implied "acceptable". Because we prefer to use mutually exclusive categories we have redefined "acceptable" as "Prediger's acceptable but not Prediger's good".

cheap	$\mapsto$	$(Price \leq 250)$
not expensive	$\mapsto$	$(Price > 250 \wedge \leq 400)$
expensive	$\mapsto$	$(Price > 400)$
down fibers	$\mapsto$	$(Material = goose - downs \vee duck = downs)$
synthetic fibers	$\mapsto$	$(Material \neq goose - downs \vee duck = downs)$
good	$\mapsto$	$((0 < T \leq 7) \wedge (W \leq 1000)) \vee$
		$((-7 < T \leq 0) \wedge (W \leq 1400)) \vee$
		$((-15 < T \leq -7) \wedge (W \leq 1700) \vee$
		$(T \leq -15) \wedge (W \leq 2000))$
acceptable	$\mapsto$	$((0 < T \leq 7) \wedge (1000 < W \leq 1400)) \vee$
		$((-7 < T \leq 0) \wedge (1400 < W \leq 1700)) \vee$
		$((-15 < T \leq -7) \wedge (1700 < W \leq 2000) \vee$
		$(T \leq -15) \wedge (2000 < W)$
bad	$\mapsto$	$((0 < T \leq 7) \wedge (1400 < W)) \vee$
		$((-7 < T \leq 0) \wedge (1700 < W)) \vee$
		$((-15 < T \leq -7) \wedge (2000 < W)$

TABLE 2. Example of a Terminology

from many-valued contexts. In FCA this is called *scaling*, and various forms of scaling (conceptual scaling, logical scaling) are discussed in the literature.

	cheap	not expensive	expensive	down fibers	synthetic fibers	good	acceptable	bad
Sleeping Bag	**Price**			**Fiber**		**Quality**		
One Kilo Bag	1	0	0	0	1	1	0	0
Sund	1	0	0	0	1	0	0	1
Kompakt Basic	1	0	0	0	1	1	0	0
Finmark Tour	1	0	0	0	1	0	0	1
Interlight Lyx	1	0	0	0	1	0	0	1
Kompakt	0	1	0	0	1	0	1	0
Touch the Cloud	0	1	0	0	1	0	1	0
Cat's Meow	0	1	0	0	1	1	0	0
Igloo Super	0	1	0	0	1	0	0	1
Donna	0	1	0	0	1	0	1	0
Tyin	0	1	0	0	1	0	1	0
Travellers Dream	0	1	0	1	0	1	0	0
Yeti Light	0	1	0	1	0	1	0	0
Climber	0	1	0	1	0	0	1	0
Viking	0	1	0	1	0	1	0	0
Eiger	0	0	1	1	0	0	1	0
Climber light	0	1	0	1	0	1	0	0
Cobra	0	0	1	1	0	1	0	0
Cobra Comfort	0	1	0	1	0	0	1	0
Foxfire	0	0	1	1	0	1	0	0
Mont Blanc	0	0	1	1	0	1	0	0

TABLE 3. Derived Context: *Sleeping Bags*

1.4. Graph Layout.

1.4.1. *Pictures of Graphs.* We can make a drawing or layout of the multivariable graph, by placing the vertices at $n + K$ locations in the plane, or, more generally, in $\mathbb{R}^p$. We then connect each object with the categories it is in. Thus m lines are leaving from each object-point, and the number of lines arriving at a category-point is equal to the number of objects in the category.

The map of the $n + K$ points and the nm lines is called the *graph plot.* It is often useful to think of the graph plot as the overlay of m plots, one for each variable, which are called *star plots.* In the star plot for variable j there are $n + k_j$ points, and n lines, each line connecting an object with the category it is in for variable j. The star plot consists of k_j disjoint stars, one for each category, where the star consists of the lines connecting the category-point with the object-points of the objects in that category.

Again, the graph plot does not lead to any loss of information. All the relationships in the original data are still present in the plot. Nevertheless, it may be difficult to *reconstruct* the data from the drawing, because the graph plot has many overlapping lines and sometimes looks like one large black blob. It helps to look at the individual star plots, but even these can easily get too crowded and messy when n is large. Thus, we can now use the fact that our map of the objects and categories into the plane was completely arbitrary. Suppose we choose a map that makes the graph plot look as clear or clean or nice as possible. This is typically done with a graph layout algorithm, and of course there are many of these, because the words "clean" and "clear" and "nice" can be defined in many different ways.

1.4.2. *Overview of the Literature.* Since this paper is primarily written for statisticians, and not for computer scientists, we provide a brief overview of the literature in computer science and computational geometry about methods and criteria to draw graphs. A good overview of modern graph drawing is given in the chapters by [23] and [55]. There is also a very extensive annotated bibliography [16] and a comprehensive book has just appeared [3]. Since 1992 there also has been a yearly conference on graph drawing. There are many software packages available to draw graphs. See

```
http://www.cs.brown.edu/people/rt/gd.html
```

for references both to the symposia and the software packages. We single out one of these packages as an example, because it is so easily available. The Java Development Kit of Sun Microsystems comes with a GraphLayout applet demo. If you have a Java enabled browser, go to

```
http://www.javasoft.com/applets/jdk
```

and go to the demo section from there.

It must be emphasized from the start that in many cases the graph drawing algorithms discussed in the computational geometry literature are not intended as data analysis techniques. They have a different "aesthetic" purpose. If we limit ourselves to straight-line drawings[4] then the aesthetic criteria that are mentioned in [3, page 14-16] are

- display symmetry,
- avoid edge crossings,
- keep edge lengths uniform,
- distribute vertices uniformly,
- minimize aspect ratio,

and

[4]and consequently ignore polyline drawings, orthogonal drawings and bended edges

- minimize area,
- minimize total edge length,
- minimize maximum edge length.

The last group of three criteria is special, because they are "meaningful only if the drawing convention adopted prevents drawings from being arbitrarily scaled down" [3, page 14]. Not all of these criteria seem relevant for data analysis. Some of them were inspired by circuit board design, in which minimizing crossings is obviously relevant. But in data analysis we do not only want aesthetically pleasing drawings, we also want drawings that show us important and invariant aspects of the data.

1.4.3. *Classes of Algorithms.* Basically, there are two classes of graph layout or graph drawing algorithms. One class is based on logical or binary criteria in which properties of the graph, such as edge crossings, are counted and optimized. A problem with these algorithms is that many of the criteria lead to NP-hard problems, i.e. they are computationally infeasible even for fairly small problems. The other class is more interesting from the data analysis point of view. It thinks of the graph as a set of pegs connected by mechanical and/or electric forces. The vertices attract and repel each other. The total forces on the system can be summarized in a loss function, and that loss function can be minimized. This approach was introduced by [17], although there are earlier versions of similar algorithms in circuit board design. We shall discuss these *spring algorithms* in more detail below.

2. Multidimensional Data Analysis

2.1. **Multiple Correspondence Analysis.** *Multiple Correspondence Analysis*, or MCA, can be introduced in many different ways [4, 24, 25]. Usually, it is motivated in graphical language. Complicated multivariate data are made more accessible by displaying the main regularities of the data in scatterplots. Our discussion of MCA emphasizes the two types of plots we discussed earlier, the graph plot and the star plots. This emphasis was first used, briefly, in the review articles by de Leeuw et al. [14], Hoffman and de Leeuw [32], Michailidis and de Leeuw [47]. We think that it nicely captures the essential geometric characteristics of the technique.

2.1.1. *Loss Function.* The basic idea is that if we draw the nm edges, then the resulting graph plot will generally be more informative and more aesthetically pleasing if the edges are *short*. In other words, if objects are close to the categories they fall in, and categories are close to the objects falling in them. Thus we want to make a graph plot that "minimizes the amount of ink", i.e. the total length of all edges.

Actually, for computational reasons, we will minimize the total *squared* length of the edges. To formalize this "minimum squared ink" criterion in a convenient way, we use the indicator matrices G_j. If the $n \times p$ matrix X has the locations of the object vertices in $\mathbb{R}^p$, and Y_j has the location of the k_j category vertices of variable j, then the squared length of the n edges for variable j is

$$\sigma_j(X, Y_j) = \mathbf{SSQ}(X - G_jY_j), \tag{2.1}$$

where $\mathbf{SSQ}()$ is short for the sum of squares. The quantity (2.1) measures the amount of ink in the star plot of variable j.

The squared edge length over all variables is

$$\sigma(X, Y) = \sum_{j=1}^{m} \mathbf{SSQ}(X - G_jY_j), \tag{2.2}$$

and this is the function we want to minimize. The book by Gifi [24] is mainly about many different versions of this minimization problem, where the differences are a consequence of various *restrictions* imposed on the quantifications Y_j.

Minimizing (2.2) without any restrictions on the vertex locations is not possible. Or, more precisely, it is too easy. We just collapse all vertices into a single point, and we use no ink at all. Remember the quotation in Section 1.4.2 about graph layout techniques that only make sense if the drawing cannot be arbitrarily scaled down. It means that in order to get a nontrivial solution, we have to impose some form of *normalization*. In MCA we require that the columns of X add up to zero, and are *orthonormal*, i.e. satisfy $mX'X = I$[5].

2.1.2. *Equations.* One of the reasons why squared edge lengths are so appealing is that the MCA problem we are trying to solve is basically an eigenvalue problem. We discuss this in some detail, again following [24].

First we define some useful matrices. Define the $k_j \times k_\ell$ matrix $C_{j\ell} = G'_jG_\ell$. Matrix $C_{j\ell}$ is the *cross table* or *contingency table* of variables j and ℓ. Thus $D_j = C_{jj}$, where D_j is the diagonal matrix with the univariate marginals of variable j on the diagonal. The $K \times K$ supermatrix C is known in the correspondence analysis literature as the *Burt Matrix*, after [9]. Write $CY = mDY\Xi$ for the generalized eigenvalue problem associated with the Burt matrix.

We also define $P_j = G_jD_j^{-1}G'_j$; then, P_j is the *between-category* projector, which transforms each vector in $\mathbb{R}^n$ into a vector in $\mathbb{R}^n$ with category means. Moreover $Q_j = I - P_j$ transforms each vector into a *within-category* vector of deviations from category means. Write $P_\star$ for the average of the P_j's ($P_\star = \frac{1}{m}\sum_{j=1}^{m} P_j$), and write Λ for the diagonal matrix containing the eigenvalues of $P_\star$.

[5]Alternatively, we could normalize Y, i.e. require that $u'DY = 0$ and $Y'DY = I$. Here u is a vector with all elements equal to $+1$, and D is the $K \times K$ diagonal matrix with marginal frequencies of all m variables on the diagonal. It is shown in [24] that the two different normalizations lead to essentially the same solution.

Theorem 2.1. *Suppose $(\hat{X}, \hat{Y})$ solves the MCA problem. Then*

$$P_\star \hat{X} = \hat{X}\Lambda, \tag{2.3a}$$

$$C\hat{Y} = mD\hat{Y}\Lambda. \tag{2.3b}$$

Proof. Define $\sigma(X, \bullet)$ as the minimum of $\sigma(X, Y)$ over all Y. Clearly the minimum is attained for

$$\hat{Y}_j = D_j^{-1}G_j'X, \tag{2.4}$$

i.e. by locating a category quantification in the centroids of the objects in that category. We see that

$$\sigma(X, \bullet) = m\,\mathbf{tr}\,X'(I - P_\star)X. \tag{2.5}$$

Clearly we minimize $\sigma(X, \bullet)$ over $mX'X = I$ by choosing $\hat{X}$ equal to the eigenvectors corresponding with the p largest eigenvalues of $P_\star$. Thus $P_\star\hat{X} = \hat{X}\Lambda$ for MCA. Also, from (2.4), we see $G\hat{Y} = mP_\star\hat{X} = m\hat{X}\Lambda$ and thus $C\hat{Y} = mG'\hat{X}\Lambda = mD\hat{Y}\Lambda$. This proves (2.3b). □

There are several aspects of the proof which deserve some additional attention. Equation (2.4) is called the *centroid principle*. The centroid principle shows clearly how the star plots get their name in MCA. Category vertices are in the centroid of the vertices of the objects in the category, and if we have a clear separation of the k_j categories, we see k_j stars in $\mathbb{R}^p$. This also shows that in MCA the category vertices are in the convex hull of the object vertices, they form a more compact cloud.

There is one last important construct in MCA we like to mention. The matrix $\hat{Y}_j'D_j\hat{Y}_j = \hat{X}'P_j\hat{X}$ is known as the *discrimination matrix.* It is equal to the between-category dispersion matrix of variable j, i.e. to the size of the stars for that variable. The average discrimination matrix is equal to Λ, the diagonal matrix of eigenvalues. Since $P_\star$ is the average of m orthogonal projectors, we have $\Lambda \leq I$. This can also be seen from the fact that each element of Λ is the average, over all variables, of the ratio of the between-category variance and the total variance.

2.1.3. *Algorithm.* The basic algorithm for MCA is *alternating least squares,* also known in this context as *reciprocal averaging.* An iteration consists of two steps

$$Y_j^{(k)} = D_j^{-1}G_j'X^{(k)}, \tag{2.6a}$$

$$X^{(k+1)} = \frac{1}{m}\sum_{j=1}^{m} G_jY_j^{(k)}, \tag{2.6b}$$

i.e. we alternate between the *first* and *second centroid principle.* After some of these iterations we reorthonormalize X. This algorithm is identical to Bauer-Rutishauser simultaneous iteration [53], the natural generalization of the power method to compute some of the dominant eigenvalues of a symmetric matrix with the corresponding eigenvectors.

2.1.4. *Example.* If we apply (two-dimensional) MCA to the sleeping bag data, we find the solutions shown in the graph plot and the three star plots below. Notice that objects with similar profiles are mapped to identical points on the graph plot, a property following from the second centroid principle. Several things are immediately clear. There are good, expensive sleeping bags filled with down fibers and cheap, bad quality sleeping bags filled with synthetic fibers. There are also some intermediate sleeping bags in terms of quality and price filled either with down or synthetic fibers. Finally, there are some expensive ones of acceptable quality and some cheap ones of good quality. However, there are no bad expensive sleeping bags.

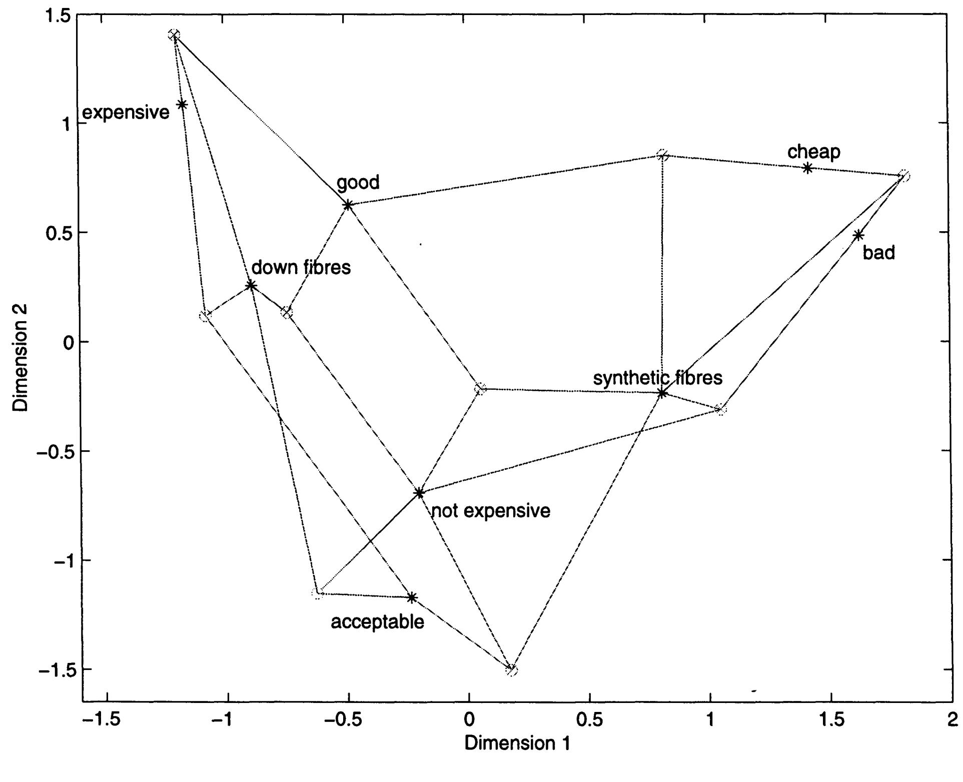

FIGURE 2. Graph plot of the MCA solution of the sleeping bag data

In this case, we could have seen this much faster by looking directly at the data, without using a computer at all. But the sleeping bag example is far from typical. In real-life MCA examples we often deal with thousands of objects and hundreds of variables (see for example [24, Chapter 13], [47], [45]).

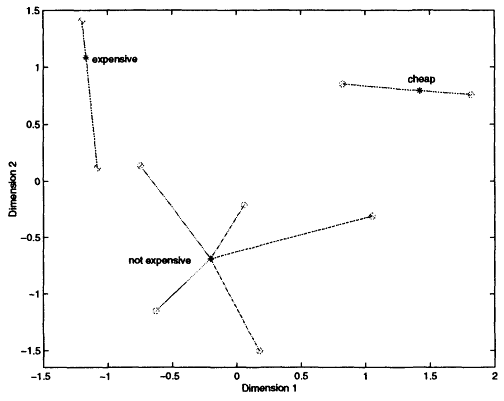

FIGURE 3. Star plot of variable Price

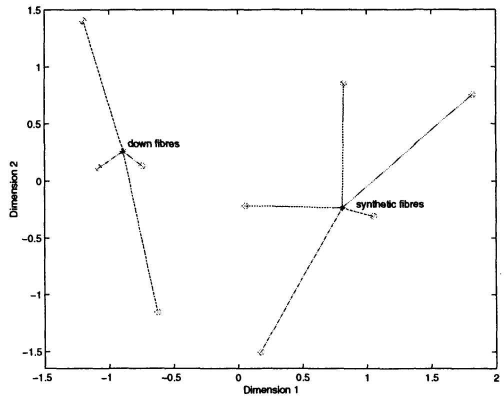

FIGURE 4. Star Plot of variable Fiber

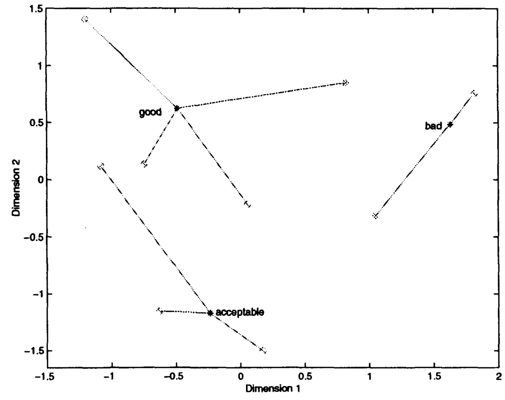

FIGURE 5. Star Plot of variable Quality

2.2. **Multidimensional Scaling.** Multidimensional scaling (MDS) is a class of techniques in which given distances or distance-like numbers are approximated by distances in low-dimensional Euclidean space. Thus, given distance-like numbers δ_{ij} (i.e. ($\delta_{ij} \geq 0$, $\delta_{ii} = 0$), often called *dissimilarities*, between n objects, we want to find n points x_i in $\mathbb{R}^p$ such that their Euclidean distance $d(x_i, x_j)$, which we also write as $d_{ij}(X)$, is approximately equal to δ_{ij}. MDS as a rigorous technique is due to [38, 39]. Theory and algorithms of MDS are most completely described in [7].

2.2.1. *Loss function.* We shall restrict our attention to using a least squares loss function, i.e. we want to create our picture in such a way that

$$\sigma(X) = \sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij}(\delta_{ij} - d_{ij}(X))^2 \tag{2.7}$$

is minimized over X. The w_{ij} are weights, which can be chosen to reflect variability, measurement error, or missing data.

In [41] the loss function has weights δ_{ij}^{-2}. The loss function is interpreted as the amount of *physical work* that must be done on elastic springs to stretch or compress them from an initial length δ_{ij} to a final length d_{ij}. Sammon [51] suggests $w_{ij} = \delta_{ij}^{-1}$ for a closely related problem. Connections with the spring algorithms for graph layout are already obvious, and will be examined in considerable detail below.

2.2.2. *Algorithm.* Using the unit vectors e_i of order n, we can define the matrices $A_{ij} = (e_i - e_j)(e_i - e_j)'$, i.e. A_{ij} has element $+1$ at positions (i,i) and (j,j), -1 at (i,j) and (j,i) and 0 everywhere else. Moreover, for a given real symmetric matrix C, we define the *Laplacian* of C as

$$\mathcal{L}(C) = \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}A_{ij}. \tag{2.8}$$

Alternatively $\mathcal{L}(C) = R - C$, where R is the diagonal matrix with row-sums of C. Now set

$$B(X) = \mathcal{L}(W\frac{\Delta}{D(X)}), \tag{2.9a}$$

$$V = \mathcal{L}(W), \tag{2.9b}$$

where in (2.9a) we take the Laplacian of the matrix with elements $w_{ij}\delta_{ij}/d_{ij}(X)$. Then the algorithm that updates X by

$$X^{(k+1)} = V^{+}B(X^{(k)})X^{(k)}, \tag{2.10}$$

with V^{+} the Moore-Penrose inverse of V, is shown to be globally convergent in [11] and linearly convergent in [12].

In order to compute the initial estimate of X Guttman [28] suggested to compute the dominant eigenvectors and eigenvalues of $\mathcal{L}(\Delta^2)$. This maximizes

$$\lambda(X) = \sum_{i=1}^{n}\sum_{j=1}^{n} \delta_{ij}^2 d_{ij}^2(X) \tag{2.11}$$

under the normalization condition $X'X = I$. This is easy to see, since $d_{ij}^2(X) = \mathbf{tr}X'A_{ij}X$, and thus $\lambda(X) = \mathbf{tr}X'\mathcal{L}(\Delta^2)X$.

2.2.3. *MDS and MCA.* There is a fairly straightforward connection between multidimensional scaling and MCA, outlined for example in [30]. Suppose we define the loss function (2.7) only for the off-diagonal submatrix G of the adjacency matrix A. Thus, the loss function $\sigma(X)$ becomes

$$\sigma(X) = \sum_{i=1}^{n}\sum_{k=1}^{K} w_{ik}(\delta_{ik} - d_{ik}(X))^2, \tag{2.12}$$

with

$$\delta_{ik} = \begin{cases} 1 & \text{if } i \text{ is in correspondence with } k, \\ 0 & \text{otherwise,} \end{cases} \tag{2.13a}$$

and

$$w_{ik} = \begin{cases} 1 & \text{if } i \text{ is in correspondence with } k, \\ 0 & \text{otherwise.} \end{cases} \tag{2.13b}$$

With these definitions, obviously $\sigma(X)$ is again the sum of squares of the distances between the objects and the categories they are in, i.e. our "minimum squared ink" criterion. We need some kind of normalization to find a nontrivial solution, and using $X'X = I$ produces MCA.

There is another way to introduce MCA using MDS ideas, which was first discussed by [13]. Since that paper the approach has been extended considerably by Meulman [42, 43] and it has been implemented in the computer program PIONEER [26].

The basic idea here is to scale the objects using some form of MDS. Suppose we define a distance-like measure δ_{ij} between each pair of objects. We now want to map the objects into points x_i in $\mathbb{R}^p$ such that the distance $d(x_i, x_k)$ approximates δ_{ij}.

We have not specified yet which distances we intend to use, and how we will measure approximation. In MCA we use *chi-square* or *Benzécri* distances. They are defined on the rows of G. Write g_i for the column-vector containing row i of G

and e_i for the i^{th} unit vector. Then

$$(2.14)\quad \delta_{ij}^2 = \frac{1}{m}(g_i - g_j)'D^{-1}(g_i - g_j) = \frac{1}{m}(e_i - e_j)'GD^{-1}G'(e_i - e_j) = (e_i - e_j)'P_\star(e_i - e_j) = (e_i - e_j)'X\Lambda X'(e_i - e_j) = (z_i - z_j)'(z_i - z_j),$$

where $P_\star$ is the average between-category projector of MCA, and where $Z = X\Lambda^{1/2}$. If we only use the first p dimensions, i.e. the first p columns of Z, then we underestimate the chi-square distance, i.e. we approximate δ_{ij}^2 from below. This defines MCA. We can also proceed the other way around and define Benzécri distances between columns of G, and again come up with the MCA solution for Y.

But instead of approximating squared distances from below, we can also use (2.7) on the Benzécri distances, or on any other distance function defined on the objects or categories. We will loose the duality between rows and columns we have in MCA, but we may find more interesting solutions. The solutions for the sleeping bag example using (2.7) and Benzécri distances on the objects and the categories of the variables, are given in Figures 6 and 7, respectively. If we compare them with the correspondence analysis solution, we see that the "horseshoe" or parabolic shape in Figure 2 is no longer there. Points are spread more uniformly in the plane. However, this also results in too many edge crossings for the solution based on distances between categories.

2.3. **Parallel Coordinate Plots.**

(PCP) There is another simple way to plot multivariate quantitative data. This is the *parallel coordinate plot* discussed by Inselburg and Dimsdale [33], Wegman [57]. In these plots, we draw m parallel straight lines, one for each variable. The objects are then plotted on each of the lines, and points corresponding with the same objects are connected by broken line segments (and perhaps colored with different colors).

2.3.1. *PCP and MCA.*

There are some interesting connections between PCP and MCA. Suppose we have the freedom to put the categories of the variables in arbitrary locations on the m parallel vertical lines, except that the categories of variable j must be on line j. Each object now defines a broken line through m category points. Suppose z_{ij} is the induced quantification of object i on variable j, which is the same number as the category quantification of the category of variable j that object i is in. We can partition the variance in the induced quantifications, as in Table 4. This measures in how far the object-lines deviate from the horizontal lines, by computing the variance around the best-fitting horizontal line. The best fitting horizontal line is, of course, the object score of MCA, i.e. $x_i = z_{i\bullet}$. Minimizing the ratio of the within-object variance $y'(D - \frac{1}{m}C)y$ to the total variance $y'Dy$ amounts to computing the first dimension of an MCA. Observe that this is the same as maximizing the between-object variance $\frac{1}{m}y'Cy$ for a given tot al variance, i.e.

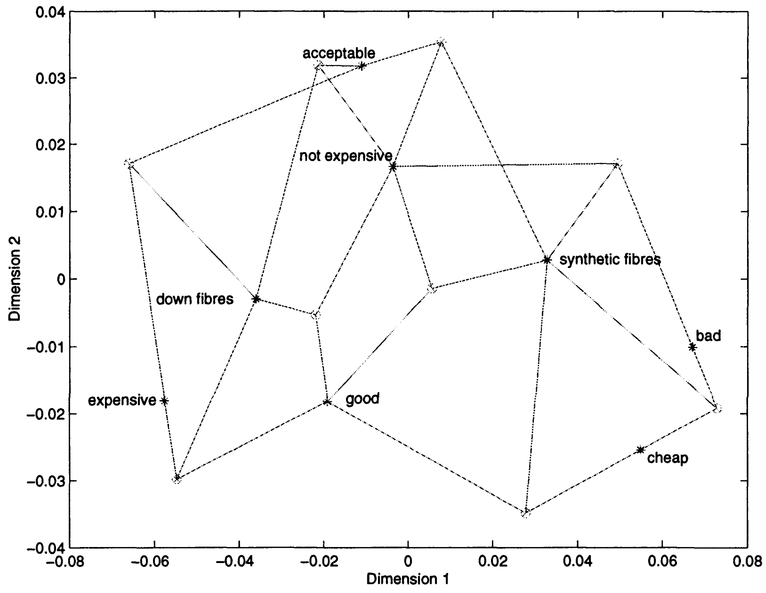

FIGURE 6. MDS solution using Benzécri distances between objects (the category points are computed as the centroids of the object points)

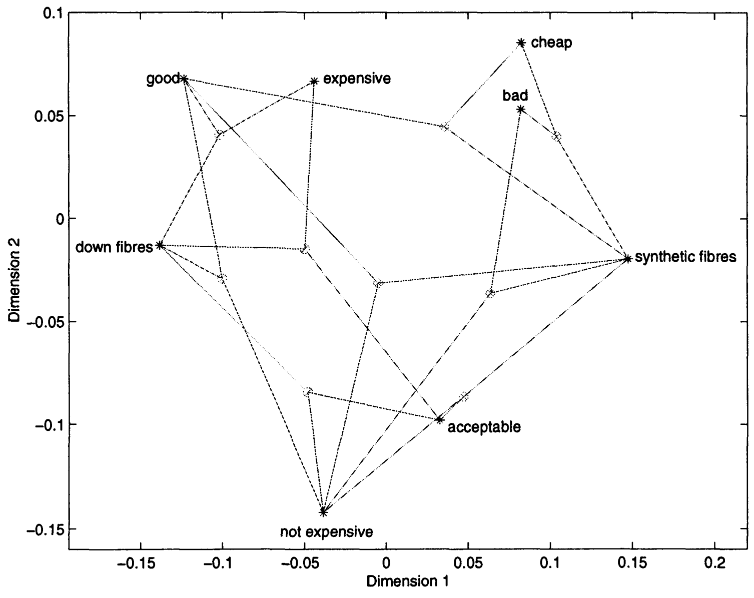

FIGURE 7. MDS solution using Benzécri distances between categories of the variables (the object points are computed as the centroids of the category points)

we also want the horizontal lines to be as far apart as possible. This is discussed in more detail in [24, Chapter 3].

Source	Sum of Squares	Matrix Expression
Within Objects, Between Variables	$\sum_{i=1}^{n}\sum_{j=1}^{m}(z_{ij} - z_{i\bullet})^2$	$y'(D - \frac{1}{m}C)y$
Between Objects	$m\sum_{i=1}^{n}(z_{i\bullet} - z_{\bullet\bullet})^2$	$\frac{1}{m}y'Cy$
Total Variance	$\sum_{i=1}^{n}\sum_{j=1}^{m}(z_{ij} - z_{\bullet\bullet})^2$	$y'Dy$

TABLE 4. Partitioning Quantification Variance

We illustrate the above with our sleeping bag example (see Figure 8). The basic classification of the bags is again obvious from this representation. Observe we have one crossing, basically because some sleeping bags filled with synthetic fibers are good, while some filled with down are only acceptable.

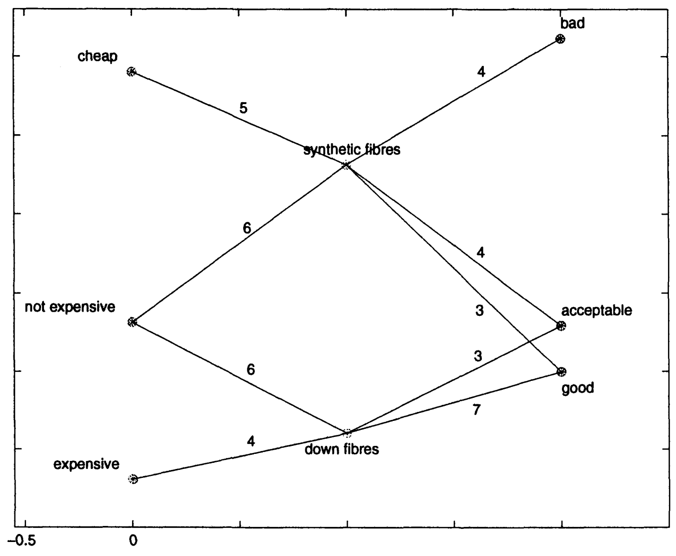

FIGURE 8. Parallel coordinate plot. The numbers on the edges indicate how many objects share that particular edge.

2.4. **Seriation.** *Seriation*, also known as *ordination*, is of importance in archaeology [1], DNA sequencing [5], hypertext ordering [6], ecology [19], and sparse matrix ordering [2].

The key concepts in this area are the Robinson and the Petrie matrices. The *Robinson matrix*, known in psychometrics as a *simplex*, is a symmetric matrix R such that $r_{ij} \leq r_{ik}$ if $j < k < i$ and $r_{ij} \geq r_{ik}$ if $i < j < k$. If rows and columns of a symmetric matrix can be permuted such that it becomes Robinson, we call it *pre-Robinson.* A *Petrie* matrix is a binary matrix for which in each row the ones form a consecutive sequence. If we can permute the rows of the binary matrix so that it becomes Petrie, we call it *pre-Petrie.* If both rows and columns can be made to have the "consecutive ones" property, we say the matrix is *two-way Petrie.*

A closely related matrix is the *Guttman matrix,* in which the pattern of one's is triangular. In both a Petrie matrix and a Guttman matrix, the pattern is basically one-dimensional. If artifacts are found in a particular time interval, or plants are found in a particular aridity interval, or if politicians vote for a proposition in a particular left-right interval, then we deal with a parallelogram structure. I such a case, we have *comparison data* in the sense of [10] or *non-cumulative items* in the sense of [49]. If subjects respond to test items, then they will give the correct response to all items that are easy enough. These are *dominance data,* or *cumulative items,* and they give rise to a triangular pattern.

In psychometrics the techniques to recover the triangular or parallelogram pattern are known as Guttman scaling or parallelogram analysis ([10, 20]).

2.4.1. *Seriation, MCA, and MDS.* In many seriation examples, MCA is used to find the ordering of the rows and columns of the matrix. It is shown by [31] that if a matrix can be permuted to become two-way Petrie, then correspondence analysis will find the order. It is shown by [27] that MCA produces the correct order for a Guttman matrix. Other situations in which MCA gives the "correct" order are discussed by [52].

Kendall [36] applied nonmetric multidimensional scaling to the product moment matrix GG', which is pre-Robinson if G is pre-Petrie. He developed the popular HORSHU method, that produced a two dimensional plot of the seriation, which often looked like a horseshoe (i.e. a quadratic curve in the plane). See [30] for additional discussion of the Robinson and Petrie forms of a matrix in the MCA and MDS contexts, and see [44] for some archaeological examples in which MCA is used.

2.4.2. *Spectral Methods of Seriation.* Recent reviews of seriation, from a modern computational point of view, can be found in [2] and [6]. Both papers rely on a spectral method of seriation, which is closely related to some of the techniques we have discussed above. They start with a binary data matrix, which they interpret as the adjacency matrix G of a bipartite graph. They then embed the matrix in a symmetric matrix A, using (1.1).

Define now the *Fiedler value* of a doubly-centered positive-semidefinite matrix as the smallest nonzero eigenvalue. The corresponding eigenvector is called the *Fiedler vector.* Seriation computes the Fiedler vector of the Laplacian $\mathcal{L}(A)$, and uses the rank order of the elements of the Fiedler vector to reorder rows and columns.

Computing the Fiedler vector means solving the eigenvalue problem

$$Gy = (m - \lambda)x, \tag{2.15a}$$

$$G'x = (D - \lambda I)y, \tag{2.15b}$$

which is of course very close to the equations

$$Gy = m\lambda x, \tag{2.16a}$$

$$G'x = \lambda Dy, \tag{2.16b}$$

that define MCA. As in MCA, it is shown next that if there is a permutation transforming the matrix to Robinson form, then the Fiedler ordering produces that permutation [2].

Theorem 2.2. *If a matrix A is Robinson, then it has a monotone Fiedler vector.*

Proof. We begin by defining two useful matrices, V_1 a $(n-1)\times n$ matrix given by

$$V_1 = \begin{bmatrix} -1 & 1 & 0 & \dots & 0 \\ 0 & -1 & 1 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & -1 & 1 \end{bmatrix} \tag{2.17}$$

and V_2 a $n \times (n-1)$ matrix given by

$$V_2 = \begin{bmatrix} 0 & 0 & 0 & \dots & 0 \\ 1 & 0 & 0 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ 1 & 1 & \dots & 1 & 0 \\ 1 & 1 & \dots & 1 & 1 \end{bmatrix} \tag{2.18}$$

Note that for any column vector x we get $V_1x = (x_2 - x_1, \dots, x_n - x_{n-1})'$, and that $V_1V_2 = I_{n-1}$ and $V_2V_1 = I_n - ue_1'$, where u is a vector comprised of ones and $e_1 = (1,\ 0, \dots, 0)'$. Consider now any eigenvector x of $\mathcal{L}(A)$, except the one corresponding to the $\lambda_1 = 0$ eigenvalue. We then have

$$\begin{aligned} \mathcal{L}(A)x = \lambda x &\Leftrightarrow V_1\mathcal{L}(A)x = \lambda V_1 x \Leftrightarrow V_1\mathcal{L}(A)(I_n - ue_1')x = \lambda V_1 x \\ &\Leftrightarrow V_1\mathcal{L}(A)(V_2V_1)x = \lambda V_1 x \Leftrightarrow VV_1y = \lambda x, \end{aligned} \tag{2.19}$$

where $V = V_1\mathcal{L}(A)V_2$ and $y = V_1x \neq 0$. Hence, λ is an eigenvalue for both $\mathcal{L}(A)$ and V, for eigenvectors of $\mathcal{L}(A)$ other than u (corresponding to $\lambda_1 = 0$). Some algebra shows that

$$V(i,j) = \sum_{k=j+1}^{n} (A(i,k) - A(i+1,k)), \text{ if } i < j, \tag{2.20a}$$

$$V(i,j) = \sum_{k=1}^{j} (-A(i,k) + A(i+1,k)), \text{ if } i > j, \tag{2.20b}$$

which implies that since A is assumed to be a Robinson matrix $V(i,j) \leq 0$ for all off-diagonal elements.

Define $\tilde{V} = \zeta I_{n-1} - V$ for some $\zeta \geq \max_i\{\lambda_i, V(i,i)\}$. Then, $\tilde{V}(i,j) \geq 0$ for all i, j, has eigenvalues given by $\tilde{\lambda}_i = \zeta - \lambda_i$ and shares the same set of eigenvectors with V. But by the Perron-Frobenius theorem, there exists a nonnegative eigenvector y corresponding to the largest eigenvalue for $\tilde{V}$ and to the smallest nonzero eigenvalue for V. But $y = V_1 x$ and therefore x is the Fiedler vector of $\mathcal{L}(A)$. Moreover, since $y \geq 0$ it implies that x is nondecreasing and the result follows. □

Theorem 2.3. *If a matrix A is pre-Robinson with a simple Fiedler value and a Fiedler vector without ties, then the permutation π induced by sorting the values in the Fiedler vector in increasing order makes A a Robinson matrix. The same holds true if the elements of the Fiedler vector are sorted in decreasing order.*

Proof. Due to the assumptions made, the Fiedler vector x is unique up to multiplication by a constant. Notice that permuting A merely changes the order of the entries in x. Suppose that a permuted version of A is Robinson. By Theorem 2.2 it has a monotone Fiedler vector x, which is unique since the Fiedler value is simple. Moreover, since x has no tied values, the permutation must correspond to either an increasing or a decreasing order of the values of x. □

In the presence of tied values in the Fiedler vector, one needs to examine all possible permutations induced (for more details see [2]).

2.4.3. *Example.* The ordering of the sleeping bags and categories computed by MCA is given in Table 5 below. We see in the parallelogram structure of the table the same ordering of sleeping bags and categories that we have already seen in other analyses. Observe that we don't quite have the consecutive ones property. Major deviations are the "One Kilo Bag" and the "Kompakt Basic", which are cheap and filled with synthetic fiber, but still classified as good. Similarly, although the "Eiger" is expensive and filled with down, it is not good, only acceptable. If we use the two eigenvectors corresponding to the two smallest nonzero eigenvalues of $\mathcal{L}(A)$ (i.e. the Fiedler value and the second smallest nonzero eigenvalue), we find the solution shown in Figure 9. The resulting graph plot has a lot of similarities to the one corresponding to the MCA solution (see Figure 2), since both techniques recover the parallelogram structure in the sleeping bag data. However, the absence of a centroid principle for the solution based on the Fiedler vectors results in placing most vertices (both objects and categories) on the periphery of the graph.

Sleeping Bag	expensive	down	good	acceptable	not expensive	synthetic	cheap	bad
Foxfire	1	1	1	0	0	0	0	0
Mont Blanc	1	1	1	0	0	0	0	0
Cobra	1	1	1	0	0	0	0	0
Eiger	1	1	0	1	0	0	0	0
Viking	0	1	1	0	1	0	0	0
Climber Light	0	1	1	0	1	0	0	0
Traveler's Dream	0	1	1	0	1	0	0	0
Yeti Light	0	1	1	0	1	0	0	0
Climber	0	1	0	1	1	0	0	0
Cobra Comfort	0	1	0	1	1	0	0	0
Cat's Meow	0	0	1	0	1	1	0	0
Tyin	0	0	0	1	1	1	0	0
Donna	0	0	0	1	1	1	0	0
Touch the Cloud	0	0	0	1	1	1	0	0
Kompakt	0	0	0	1	1	1	0	0
One Kilo Bag	0	0	1	0	0	1	1	0
Kompakt Basic	0	0	1	0	0	1	1	0
Igloo Super	0	0	0	0	1	1	0	1
Sund	0	0	0	0	0	1	1	1
Finmark Tour	0	0	0	0	0	1	1	1
Interlight Lyx	0	0	0	0	0	1	1	1

TABLE 5. MCA Seriation of Sleeping Bags

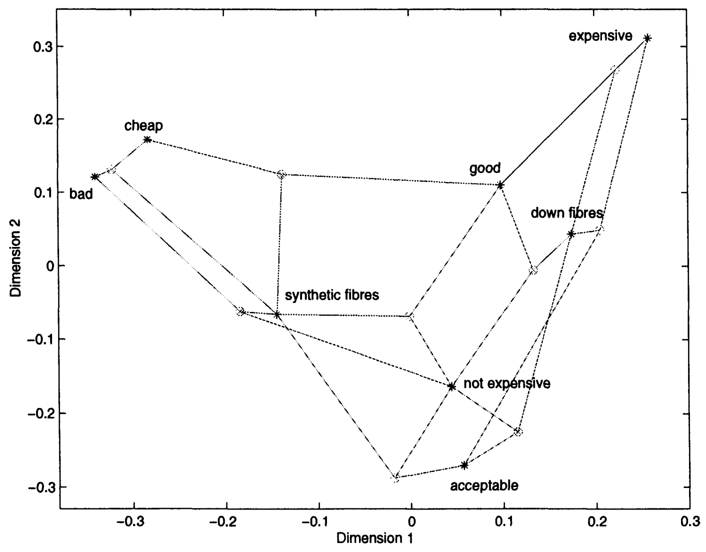

FIGURE 9. Graph plot of the sleeping bag data using Fiedler vectors.

3. Layout Methods

3.1. **Introduction.** In this section we discuss the two types of layout algorithms already mentioned in Section 1.4.3. The first class minimizes the number of edge crossings in a straight-line layout. It is especially interesting for graphs with layers, where the only edges are between layers. The second class are the spring or force-directed algorithms, which use a physical analogy to portray the graph as a mechanical and/or electric system in which forces pull and push the edges.

3.2. **Minimum Straightline Crossing Algorithms.** Suppose we have a bipartite graph. We agree to put the two layers of the graph at equal intervals on two parallel lines. We then find the permutations of the objects in each layer that minimizes the number of line crossings. There is also a one-sided version of the algorithm, in which the order in one layer is fixed. In [18] it is shown that even the one-sided problem is NP-hard, so heuristics are needed to solve even moderately sized problems. Among the heuristics discussed most frequently are the *barycenter* and *median* heuristics. In our context, we could fix the objects on one of the lines, and then compute the category positions as the means or medians of the objects in the category.

More interesting results on straightline crossing minimization are provided in [34]. The authors implement an exact algorithm for the one-sided case that turns out to work reasonably well even for problems with up to 60 vertices. For the two-sided problem the *iterated barycenter* method, which is basically what we call reciprocal averaging, turned out to be the best heuristic. In fact, it even outperforms the method which iteratively alternates the exact optimal one-sided solutions.

Let us first translate this into the graph-plot we deal with. The objects are located on one layer, while the categories of all variables on a second layer. But it is more interesting to look at the $m + 1$-layered graph, which has a layer for each variable, and an additional one for the objects. Take two variables, for instance, and locate the objects in the middle of the three parallel lines. Variable one is on the right of the object-line, variable two is on the left. If the graph-plot does not have any crossings, then the categories of a variable correspond with disjoint intervals of objects. Clearly m variables can be accommodated in a three-dimensional graph plot, in which the m lines are on a cylinder with the objects on the axes. Finding orderings without crossings is the same as *parallelogram analysis*.

3.2.1. *Ordering Variables in PCP.* One obvious problem in parallel coordinate plotting is how to order the variables, i.e. how to order the parallel vertical lines in the plane. This could be done by minimizing the line crossings in the m-layered graph. It could also be done by minimizing the amount of ink, as we do in MCA.

Of course the MCA solution is completely independent of the order of the variables. The amount of ink, i.e. the sum of squares of the lengths of the $n(m - 1)$

line segments, does depend on the order. Except for end effects, minimizing total squared length means maximizing

$$\lambda(y) = \sum_{j=1}^{m-1} y_j' C_{j,j+1} y_{j+1}. \tag{3.1}$$

over all y such that $y'Dy = 1$, and this is a function of the order of the variables. Finding the optimal order is, again, a form of seriation. It is related to the traveling salesman problem and the existence of Hamiltonian cycles in graphs. If homogeneity is large, i.e. if we can scale the variables such that the broken lines in the parallel coordinate plot are almost horizontal, then changing the order of the variables will make very little difference.

3.3. **Force-directed or Spring Algorithms.**

3.3.1. *General Idea.* In [3, Chapter 10] a general approach to force-directed graph drawing methods is outlined, that unifies many previous isolated and rather ill-defined methods. The force on vertex j is made up out of more elementary forces that are defined for each pair of vertices. There is a mechanical or spring force pulling at all pairs (i, j) that are connected, and there is an electrical force pushing at all pairs, also the ones that are not connected. Thus the force on vertex i is

$$F(i) = \sum_{j=1}^{n} a_{ij} f_{ij} - \sum_{j=1, j\neq i}^{n} g_{ij}, \tag{3.2a}$$

where $A = \{a_{ij}\}$ is the adjacency matrix of the graph. It is assumed, in addition, that the springs follow Hooke's law, and the electrical force follows an inverse square law. This means that

$$f_{ij} = w_{ij}(d_{ij}(X) - \delta_{ij})\frac{x_i - x_j}{d_{ij}(X)}, \tag{3.2b}$$

$$g_{ij} = u_{ij}\frac{1}{d_{ij}^2(X)}\frac{x_i - x_j}{d_{ij}(X)}, \tag{3.2c}$$

where δ_{ij} is the zero-energy length of the spring connecting i and j, w_{ij} is the stiffness of the spring, and u_{ij} is the strength of the electrical repulsion.

The choice of the forces is ad-hoc. In [17] and [54] logarithmic springs are used, i.e.

$$f_{ij} = w_{ij} \log\left(\frac{d_{ij}(X)}{\delta_{ij}}\right)\frac{x_i - x_j}{d_{ij}(X)}. \tag{3.3}$$

In [21], the attractive forces are proportional to the square of the distance, while the repulsive force is the inverse of the distance. Sugiyama and Misue [54] introduce a third force, by adding a magnetic field that works globally on all springs, and that can be parallel, radial, or concentric. This field will tend to influence the global form of the drawing. It is clear that spring algorithms are based on a simple and attractive idea, but implementing them requires a number of rather arbitrary choices. We concentrate on the simpler ones.

3.3.2. *Loss Function.* Implementing the spring algorithm of (3.2) means minimizing

$$\sigma(X)=\sum_{i=1}^{n}\sum_{j=1}^{n}a_{ij}w_{ij}(d_{ij}(X)-\delta_{ij})^2+\sum_{i=1}^{n}\sum_{j=1,j\neq i}^{n}u_{ij}\frac{1}{d_{ij}(X)}. \tag{3.4}$$

This is obviously close to the MDS problem of minimizing (2.7). The difference is that in the spring algorithm we add a penalty for points being too close together.

Along the same lines as before, we can show that the algorithm that updates X as follows

$$X^{(k+1)}=V^{+}\{B(X^{(k)})+H(X^{(k)})\}X^{(k)}, \tag{3.5a}$$

where

$$H(X)=\mathcal{L}(\frac{U}{D^3(X)}), \tag{3.5b}$$

and $U/D^3(X)$ is the matrix with elements $u_{ij}/d_{ij}^3(X)$, is globally convergent.

3.3.3. *The Barycentric Method.* One of the earliest graph drawing methods is the *barycentric method* of Tutte [56]. It is the special case of (3.2) in which $\delta_{ij}=0$, $w_{ij}=1$, and there are no electrical forces. Thus the loss function is simply given by

$$\sigma(X)=\sum_{i=1}^{n}\sum_{j=1}^{n}a_{ij}d_{ij}^2(X). \tag{3.6}$$

This is the same loss function as the one used in MCA, and it leads to a familiar problem. The minimizing solution is $X=0$. Unlike in MCA we do not normalize this problem away by requiring $X'X=I$, but we partition the vertex set into a set of (at least three) *fixed* vertices and *free* vertices. We then minimize over the free vertices. In a one-dimensional MCA context this approach was already discussed in [29].

Not surprisingly, the algorithm that solves this problem is to set the location of a free vertex equal to the centroid of its neighbors, and to cycle over free vertices. In the case of the graph with adjacency matrix given by (1.1) this is precisely the reciprocal averaging algorithm of MCA (without normalization, and without updating the fixed vertices). We refer to [3, Section 10.2] for a discussion on how well the barycentric method draws typical graphs. From the data analysis point of view, we merely have to compare normalizing by fixing a number of points with normalizing by requiring orthonormality. The obvious question in this context is "Which Points ?" to fix.

For our example, we give two different solutions. In the first one (see Figure 10) we fix the three categories of variable "Price" at the corners of an equilateral triangle, and we fit in the remaining points. In the second solution (see Figure 11) we fix the three categories of variable "Quality" in the same way. In both plots we insert the category quantifications by using the centroid principle, and we draw the graph plot. These graph plots are less satisfactory than the MCA graph plot 2. Fixed points are at the outskirts of the plot, the other points are clumped on the inside near the centroid of the plot. This becomes obvious if we rewrite the stationary equations for the barycentric method, with a number of category points fixed, as

$$X = \frac{1}{m}\{G_1Y_1 + G_2Y_2\}, \tag{3.7a}$$

$$Y_2 = D_2^{-1}G_2'X, \tag{3.7b}$$

where Y_1, G_1, D_1 correspond to the fixed vertices and Y_2, G_2, D_2 to the free vertices. If we solve these equations we find

$$X = \frac{1}{m}(I - \frac{1}{m}G_2D_2^{-1}G_2')^{-1}G_1Y_1, \tag{3.8a}$$

$$Y_2 = \frac{1}{m-1}D_2^{-1}G_2'G_1Y_1. \tag{3.8b}$$

This shows that objects and free categories will be inside the convex hull of the fixed categories, and clumped in the middle especially in case m is large.

As an experiment, we also implemented a version of the barycentric method using the penalty terms in (3.2), so that $\delta_{ij} = 0$ and $w_{ij} \equiv 1$ and $u_{ij} \equiv .01$. This does not look good at all, so it seems the penalties are much too harsh. More research, perhaps also with other penalty functions, is obviously needed here.

3.3.4. *More on Springs and MDS.* A two-dimensional graph layout algorithm that is basically MDS was proposed by Kamada and Kawai [35]. Essentially the same algorithm was proposed earlier in [37]. A (straightforward) three dimensional extension is discussed by Kumar and Fowler [40]. It is argued in [8] that three-dimensional pictures of graphs, such as the ones based on MDS, often are more "nice" than two-dimensional ones. The main idea in this class of graph layout algorithms is to approximate path length distances in a graph by Euclidean distances. We assume a connected graph with n vertices, in which there is a path between any two vertices. The loss function is (2.7), where δ_{ij} is the path length distance between nodes i and j, and w_{ij} is some known weight. Kamada and Kawai [35] suggest using w_{ij} proportional to δ_{ij}^{-2}. Kruskal and Seery [37] seem to use $w_{ij} \equiv 1$. They also assign a large number for the distance of a pair of vertices that are not connected.

Now of course our bipartite multivariable graph is not connected. In fact, the graph theoretical distances are either zero (for the self-distances), or one (for objects and the categories they are in), or two (for objects which share a category, or

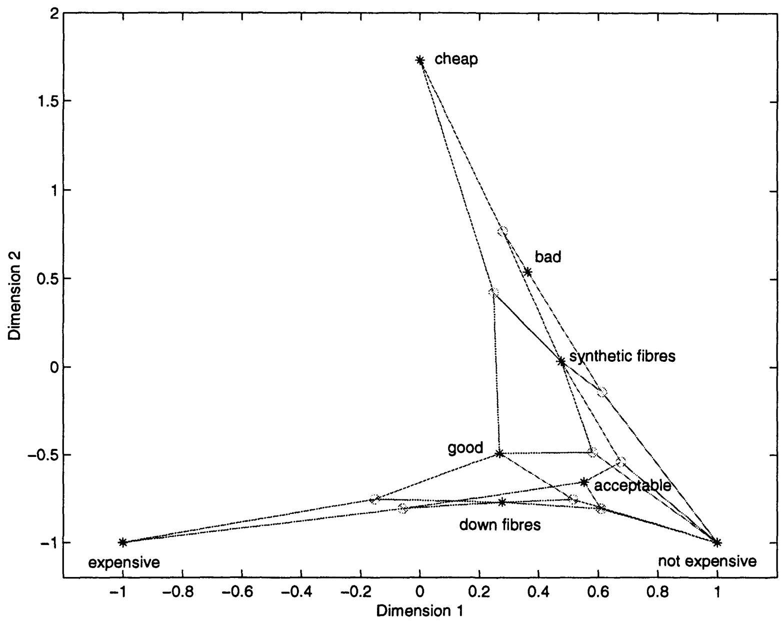

FIGURE 10. Barycentric solution with the points corresponding to variable Price fixed

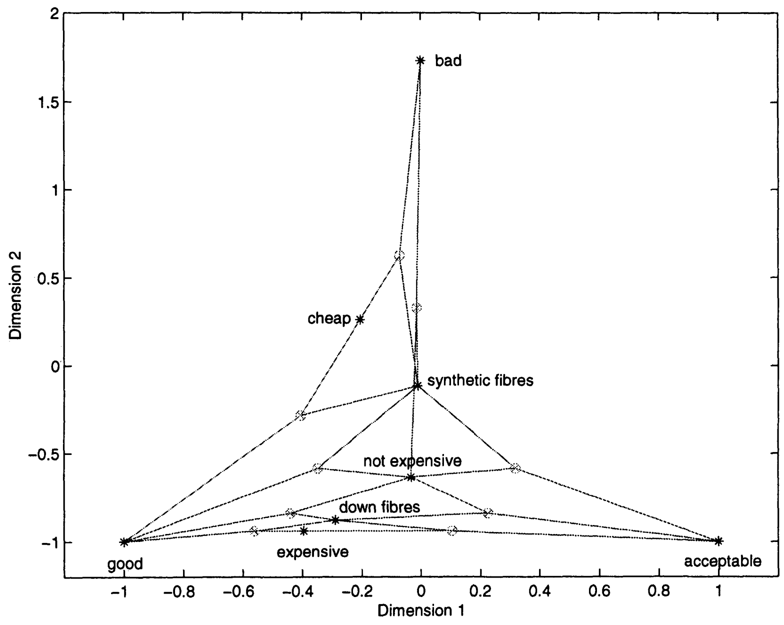

FIGURE 11. Barycentric solution with the points corresponding to variable Quality fixed

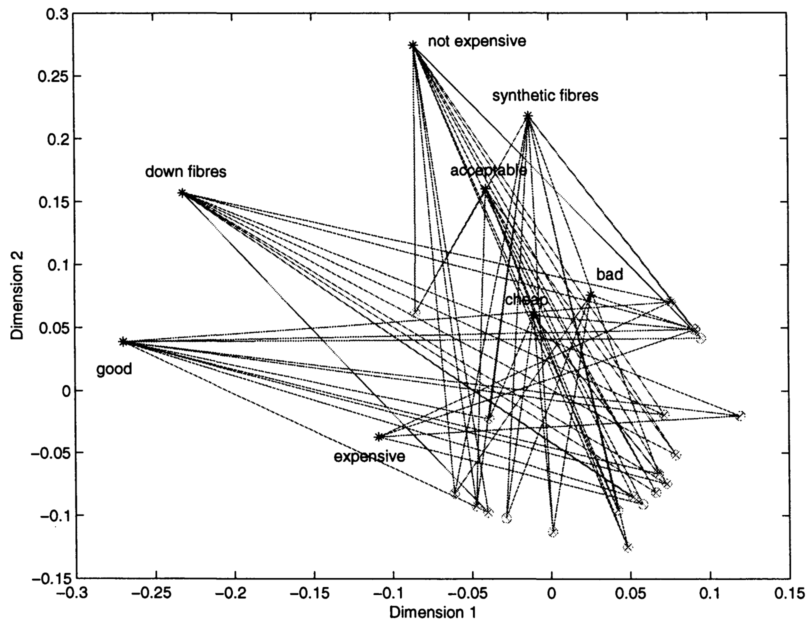

FIGURE 12. Barycentric Solution with Eades Penalties

categories which share an object) or infinity (for the rest). It may not be useful to apply MDS to these distances directly. In fact, Kruskal and Seery suggest using *non-metric multidimensional scaling*, in which we minimize

$$\sigma(X,\Delta) = \frac{\sum_{i=1}^{n}\sum_{j=1}^{n}(\delta_{ij} - d_{ij}(X))^2}{\sum_{i=1}^{n}\sum_{j=1}^{n} d_{ij}^2(X)}, \tag{3.9}$$

over all drawings X and over all $\Delta = \{\delta_{ij}\}$ that are monotonic with the graph theoretical distances.

On the other hand, we can use MDS on the off-diagonal distances only, as we have done in MCA, the barycentric method, and the spring algorithm with inverse-distance penalty terms.

4. CONCLUDING REMARKS

In this paper we have considered several popular multivariate data analysis techniques such as MCA, MDS, parallel coordinate plotting, seriation, and graph layout methods, such as force directed and minimum straightline crossing algorithms, and explored the relationship between the two classes. The representation of a multivariate (categorical) data set as a bipartite graph and the desire to make the patterns in the data more accessible by displaying them in a picture, provide the common links between these two sets of techniques. Moreover, it is shown that some of the popular graph drawing algorithms are closely related to MDS. Some of these techniques, such as MCA and spectral methods of seriation, are easy and inexpensive to apply to large data sets (both in terms of objects and variables), while the remaining ones are much more computationally demanding since they rely on iterative algorithms, thus rendering them inefficient for mining and analyzing large databanks. It is interesting to examine how these techniques perform when applied to more complicated data structures than the one examined here. A first step in that direction is taken in [46, 48], where MCA is extended to handle hierarchical data (e.g. students clustered within schools) that can be represented by direct sums of bipartite graphs. However, relational databases give rise to more complicated graph structures such as multipartite graphs and new tools are needed for their efficient visual representation. Finally, further research is required to shed light to the following third questions: first, what is the appropriate dimensionality that provides a "satisfactory" drawing of a graph, second, what are the most "useful" and "informative" distances to be approximated by MDS type methods and third how penalty methods can lead to improved drawings. A general data analytic framework is introduced in [15] that addresses the latter two questions.

REFERENCES

[1] B. Vincent Arlif. The archaeological seriation problem. Master's thesis, Institute of Computer Science, University of Copenhagen, Denmark, 1995.

[2] J.E. Atkins, E.G. Boman, and B. Hendrickson. A spectral algorithm for seriation and the consecutive ones problem. *SIAM Journal on Computing*, 28: 297–310, 1998.

[3] G. Di Battista, P. Eades, R. Tamassia, and I. G. Tollis. *Graph Drawing. Algorithms for the Visualization of Graphs*. Prentice Hall, 1998.

[4] J.P. Benzécri. *Correspondence analysis handbook*. Marcel Dekker, Inc., New York, New York, 1992.

[5] S. Benzer. The fine structure of the gene. *Scientific American*, 206:70–84, 1962.

[6] M.W. Berry, B. Hendrickson, and P. Raghavan. Sparse matrix reordering schemes for browsing hypertext. In J. Renegar, M. Shub, and S. Smale, editors, *Lectures in Applied Mathematics (LAM) Vol. 32: The Mathematics of Numerical Analysis*. American Mathematical Society, 1996.

[7] I. Borg and P. J. F. Groenen. *Modern Multidimensional Scaling*. Springer-Verlag, 1997.

[8] A. Buja, N. Dean, M. Littman, and D. Swayne. Higher dimensional representations of graphs. Technical report, DIMACS, Piscataway, NJ, 1995.

[9] C. Burt. The factorial analysis of qualitative data. *British Journal of Statistical Psychology*, 3:166–185, 1950.

[10] C. Coombs. *A Theory of Data*. Wiley, 1964.

[11] J. de Leeuw. Applications of convex analysis to multidimensional scaling. In J. Barra, H. Caussinus, and G. Romier, editors, *Recent Developments in Statistics*, 1977.

[12] J. de Leeuw. Convergence of the majorization algorithm for multidimensional scaling. *Journal of Classification*, 5:163–180, 1988.

[13] J. de Leeuw and J.J. Meulman. Principal component analysis and restricted multidimensional scaling. In W. Gaul and M. Schader, editors, *Classification as a Tool of Research*, Amsterdam, London, New York, Tokyo, 1986. North-Holland.

[14] J. de Leeuw, G. Michailidis, and D. Wang. Correspondence analysis techniques. In S. Ghosh, editor, *Multivariate analysis, design of experiments, and survey sampling*. Marcel Dekker, 1999.

[15] J. de Leeuw and G. Michailidis. Mulitvariate data analysis by constrained pulling. Technical report, UCLA Statistics Program

[16] G. di Battista, P. Eades, R. Tamassia, and I. G. Tollis. Algorithms for automatic graph drawing: an annotated bibliography. *Computational Geometry*, 4:235–282, 1994.

[17] P. Eades. A heuristic for graph drawing. *Congressus Numerantium*, 42:149–160, 1984.

[18] P. Eades and N. C. Wormald. Edge crossings in drawings of bipartite graphs. *Algorithmica*, 11:379–403, 1994.

[19] R.H. Whittaker (ed). *Ordination of Plant Communities*. W.H. Junk, 1978.

[20] H. Feger. *Structure Analysis of Co-occurrence Data*. Shaker, Aachen, Germany, 1994.

[21] T.M.J. Fruchterman and E.M. Reingold. Graph drawing by force-directed placement. *Software-Practice and Experience*, 21:1129–1164, 1991.

[22] B. Ganter and R. Wille. *Formale Begriffanalyse - Mathematische Grundlagen*. Springer Verlag, 1996.

[23] A. Garg and R. Tamassia. Advances in graph drawing. In *Algorithms and Complexity (Proc. CIAC' 94)*. Springer-Verlag, 1994.

[24] A. Gifi. *Nonlinear multivariate analysis*. Wiley, Chichester, England, 1990.

[25] M.J. Greenacre. *Theory and applications of correspondence analysis*. Academic Press, New York, New York, 1984.

[26] P. J. F. Groenen and J.J.F. Commandeur. PIONEER user's guide. Technical report, Department of Data Theory, University of Leiden, 1998.

[27] L. Guttman. The principal components of scale analysis. In *Measurement and prediction*. Princeton University Press, 1950.

[28] L. Guttman. A general nonmetric technique for fitting the smallest coordinate space for a configuration of points. *Psychometrika*, 33:469–506, 1968.

[29] M. J. R. Healy and H. Goldstein. An approach to the scaling of categorized attributes. *Biometrika*, 63:219–229, 1976.

[30] W.J. Heiser. *Unfolding Analysis of Proximity Data*. PhD thesis, University of Leiden, The Netherlands, 1981.

[31] M.O. Hill. Correspondence analysis: A neglected multivariate method. *Applied Statistics*, 23:340–354, 1974.

[32] D. L. Hoffman and J. de Leeuw. Interpreting multiple correspondence analysis as a multidimensional scaling method. *Marketing Letters*, 3:259–272, 1992.

[33] A. Inselburg and B. Dimsdale. Parallel coordinates: A tool for visualizing multi- dimensional geometry. In *Proceedings of the First IEEE Conference on Visualization*, 1990.

[34] M. Jünger and P. Mutzel. 2-layer straightline crossing minimization: Performance of exact and heuristic algorithms. *Journal of Graph Algorithms and Applications*, 1:1–25, 1997.

[35] T. Kamada and S. Kawai. An algorithm for drawing general undirected graphs. *Information processing Letters*, 31:7–15, 1989.

[36] D.G. Kendall. Seriation from abundance matrices. In F.R Hodson, D.G. Kendall, and P. Tatu, editors, *Mathematics in the Archaeological and Historical Sciences*. Edinburgh University Press, 1971.

[37] J. B. Kruskal and J. B. Seery. Designing network diagrams. In *Proceedings First General Conference on Social Graphics*. U.S. Department of the Census, 1980.

[38] J.B. Kruskal. Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis. *Psychometrika*, 29:1–28, 1964.

[39] J.B. Kruskal. Nonmetric multidimensional scaling: a numerical method. *Psychometrika*, 29:115–129, 1964.

[40] A. Kumar and R.H. Fowler. A spring modeling algorithm to position nodes of an undirected graph in three dimensions. Technical report, Department of Computer Science, University of Texas - Pan American, 1997.

[41] V. E. McGee. The multidimensional analysis of elastic distances. *British Journal of Mathematical and Statistical Psychology*, 19:181–196, 1966.

[42] J. J. Meulman. *A Distance Approach to Nonlinear Multivariate Analysis*. PhD thesis, University of Leiden, The Netherlands, 1986.

[43] J. J. Meulman. The integration of multidimensional scaling and multivariate analysis with optimal transformations. *Psychometrika*, 57:539–565, 1992.

[44] J.J. Meulman. *Homogeneity Analysis of Incomplete Data*. DSWO Press, 1982.

[45] G. Michailidis and J. de Leeuw. Nonlinear multivariate analysis of nels-88. Technical report, UCLA Statistics Program, Preprint 176, 1995.

[46] G. Michailidis and J. de Leeuw. Constrained homogeneity analysis, with applications to hierarchical data. Technical report, UCLA Statistics Program, Preprint 207, 1997.

[47] G. Michailidis and J. de Leeuw. The Gifi system for descriptive multivariate analysis. *Statistical Science*, 13:307–336, 1998.

[48] G. Michailidis and J. de Leeuw. Multilevel homogeneity analysis with differential weighting, to appear in. *Computational Statistics and Data Analysis*, 1999.

[49] F. Mosteller. A theory of scale analysis using noncumulative types of items. Technical report, Laboratory of Social Relations, Harvard University, 1949.

[50] S. Prediger. Symbolic objects in formal concept analysis. In G. Mineau and A. Fall, editors, *Proceedings of the Second International Symposium on Knowledge, Retrieval, Use and Storage for Efficiency*, 1997.

[51] J.W. Sammon. A nonlinear mapping for data structure analysis. *IEEE Transactions on Computers*, 18:401–409, 1969.

[52] B.F. Schriever. *Order Dependence.* Centre for Mathematics and Computer Science, Amsterdam, 1986.

[53] W.J. Stewart and A. Jennings. A simultaneous iteration method for real matrices. *ACM Transactions on Mathematical Software*, 7:184–198, 1981.

[54] K. Sugiyama and K. Misue. A simple and unified method for drawing graphs: magnetic spring algorithm. In R. Tamassia and G. Tollis, editors, *Proceedings Graph Drawing 94*. Springer Verlag, 1994.

[55] R. Tamassia. Graph drawing. In J. E. Goodman and J. O'Rourke, editors, *CRC Handbook of Discrete and Computational Geometry.* CRC Press, 1997.

[56] W.T. Tutte. How to draw a graph. *Proceedings London Mathematical Society*, 13:743–768, 1963.

[57] E.J. Wegman. Hyperdimensional data analysis using parallel coordinates. *Journal of the American Statistical Association*, 85:664–675, 1990.

Department of Statistics, UCLA, Los Angeles, CA 90095-1554
e-mail: deleeuw@stat.ucla.edu

Department of Statistics, The University of Michigan, Ann Arbor, MI 48109-1285
e-mail: gmichail@umich.edu