Data Visualization Through Their Graph Representations

II.4

George Michailidis

4.1 Introduction

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. Further, many attributes are also recorded, resulting in very high-dimensional data sets. Typically, the applications involve large-scale information banks, such as data warehouses that contain interrelated data from a number of sources. Examples of new technologies giving rise to large, high-dimensional data sets are high-throughput genomic and proteomic technologies, sensor-based monitoring systems, etc. Finally, new application areas such as biochemical pathways, web documents, etc. produce data with inherent *structure* that cannot be simply captured by numbers.

To extract useful information from such large and structured data sets, a first step is to be able to *visualize* their structure, identifying interesting patterns, trends, and complex relationships between the items. The main idea of visual data exploration is to produce a representation of the data in such a way that the human eye can gain insight into their structure and patterns. Visual data mining techniques have proven to be of particularly high value in exploratory data analysis, as indicated by the research in this area (Eick and Wills 1993a, b).

In this exposition, we focus on the visual exploration of data through their graph representations. Specifically, it is shown how various commonly encountered structures in data analysis can be represented by graphs. Special emphasis is paid to categorical data for which many commonly used plotting techniques (scatterplots, parallel coordinate plots, etc.) prove problematic. Further, a rigorous mathematical framework based on optimizing an objective function is introduced that results in a graph layout. Several examples are used to illustrate the techniques.

4.2 Data and Graphs

Graphs are useful entities since they can represent relationships between sets of objects. They are used to model complex systems (e.g., computer and transportation networks, VLSI and Web site layouts, molecules, etc.) and to visualize relationships (e.g., social networks, entity-relationship diagrams in database systems, etc.). In statistics and data analysis, we usually encounter them as dendrograms in cluster analysis, as trees in classification and regression, and as path diagrams in structural equation models and Bayesian belief diagrams. Graphs are also very interesting mathematical objects, and a lot of attention has been paid to their properties. In many instances, the right picture is the key to understanding. The various ways of visualizing a graph provide different insights, and often hidden relationships and interesting patterns are revealed. An increasing body of literature is considering the problem of how to draw a graph [see for instance the book by Di Battista et al. (1998) on graph drawing, the Proceedings of the Annual Conference on Graph Drawing, and the annotated bibliography by Di Battista et al. (1994)]. Also, several problems in distance geometry

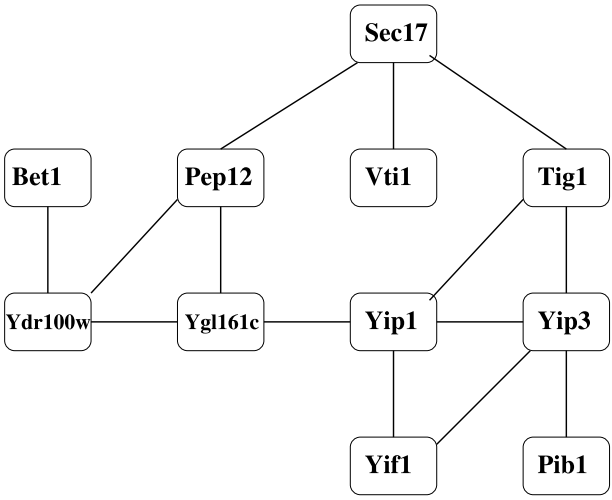

Figure 4.1. Graph representation of a small protein interaction network, with nodes corresponding to proteins and links to their physical interactions

and in graph theory have their origin in the problem of graph drawing in higher-dimensional spaces. Of particular interest in this study is the representation of data sets through graphs. This bridges the fields of multivariate statistics and graph drawing.

Figure 4.1 shows the graph representation of the protein interaction network implicated in the membrane fusion process of vesicular transport for yeast (Ito et al., 2000), with the nodes representing the proteins and the links the physical interactions between them.

However, graphs are also capable of capturing the structure of data commonly encountered in statistics, as the following three examples show. The first example deals with a contingency table (Table 4.1 and Fig. 4.2), where the nodes correspond to the categories and the weighted links represent the frequencies.

The second example deals with a small correlation matrix (Table 4.2 and Fig. 4.3), which can also be represented by a weighted graph, with the nodes representing the variables and the links the strength of the correlation.

Table 4.1. Contingency table of 5387 school children form Caithness, Scotland, classified according to two categorical variables, hair and eye color (Fisher, 1938)

	Hair color				
Eye color	Fair	Red	Medium	Dark	Black
Light	688	116	584	188	4
Blue	326	38	241	110	3
Medium	343	84	909	412	26
Dark	98	48	403	681	85

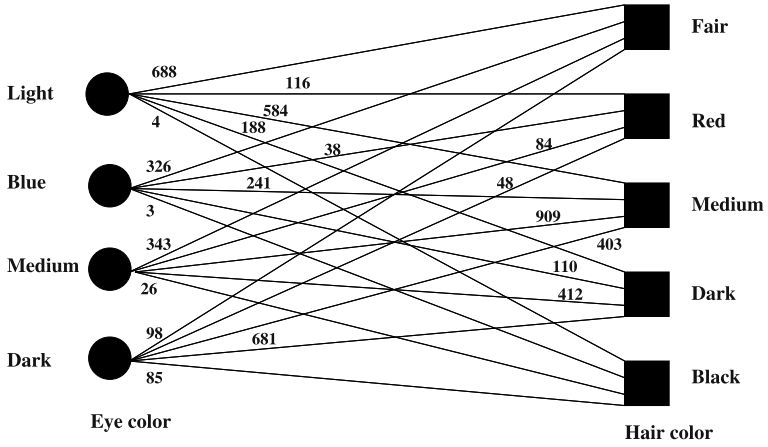

Figure 4.2. Weighted graph representation of a contingency table

Table 4.2. A small correlation matrix for four variables

	Var 1	Var 2	Var 3	Var 4
Var 1	1.00			
Var 2	0.72	1.00		
Var 3	0.43	0.84	1.00	
Var 4	0.96	0.57	0.05	1.00

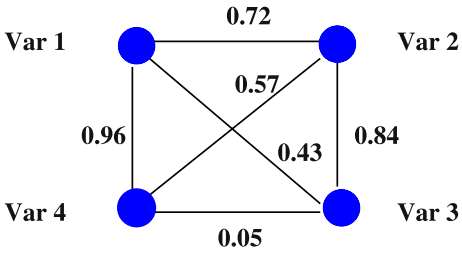

Figure 4.3. Representation of a small correlation matrix by a weighted graph

Another interesting data structure that can be represented successfully by a graph is that corresponding to a multivariate categorical data set, as the following example attests (Table 4.3). The data on 21 sleeping bags and their characteristics come from Prediger (1997) and have also been discussed in Michailidis and de Leeuw (2001).

4.3 Graph Layout Techniques

The problem of graph drawing/layout has received a lot of attention from various scientific communities. It is defined as follows: given a set of nodes connected by a set

Table 4.3. The superindicator matrix representation (Gifi, 1990) of a categorical data set

		Cheap	Not expensive	Expensive	Down fibers	Synthetic fibers	Good	Acceptable	Bad
	Sleeping Bag	Price			Fiber		Quality		
1	One Kilo Bag	1	0	0	0	1	1	0	0
2	Sund	1	0	0	0	1	0	0	1
3	Kompakt Basic	1	0	0	0	1	1	0	0
4	Finmark Tour	1	0	0	0	1	0	0	1
5	Interlight Lyx	1	0	0	0	1	0	0	1
6	Kompakt	0	1	0	0	1	0	1	0
7	Touch the Cloud	0	1	0	0	1	0	1	0
8	Cat's Meow	0	1	0	0	1	1	0	0
9	Igloo Super	0	1	0	0	1	0	0	1
10	Donna	0	1	0	0	1	0	1	0
11	Tyin	0	1	0	0	1	0	1	0
12	Travellers Dream	0	1	0	1	0	1	0	0
13	Yeti Light	0	1	0	1	0	1	0	0
14	Climber	0	1	0	1	0	0	1	0
15	Viking	0	1	0	1	0	1	0	0
16	Eiger	0	0	1	1	0	0	1	0
17	Climber light	0	1	0	1	0	1	0	0
18	Cobra	0	0	1	1	0	1	0	0
19	Cobra Comfort	0	1	0	1	0	0	1	0
20	Foxfire	0	0	1	1	0	1	0	0
21	Mont Blanc	0	0	1	1	0	1	0	0

of edges, identify the positions of the nodes in some space and calculate the curves that connect them. Hence, in order to draw a graph, one has to make the following two choices: (i) selection of the space and (ii) selection of the curves. For example, grid layouts position the nodes at points with integer coordinates, while hyperbolic layouts embed the points on a sphere. Most graph drawing techniques use straight lines between connected nodes, but some use curves of a certain degree (Di Battista et al., 1998).

Many layout algorithms are based on a set of *aesthetic* rules that the drawing needs to adhere to. Popular rules are that nodes and edges must be evenly distributed, edges should have similar lengths, edge crossings must be kept to a minimum, etc. Some of these rules are important in certain application areas. Further, many of these rules lead to a corresponding optimization problem, albeit intractable in certain cases. For example, the edge-crossing minimization is provably NP-hard and hence computationally intractable (Di Battista et al., 1998). In many cases, a basic layout is obtained by a computationally fast algorithm, and the resulting drawing is postprocessed to

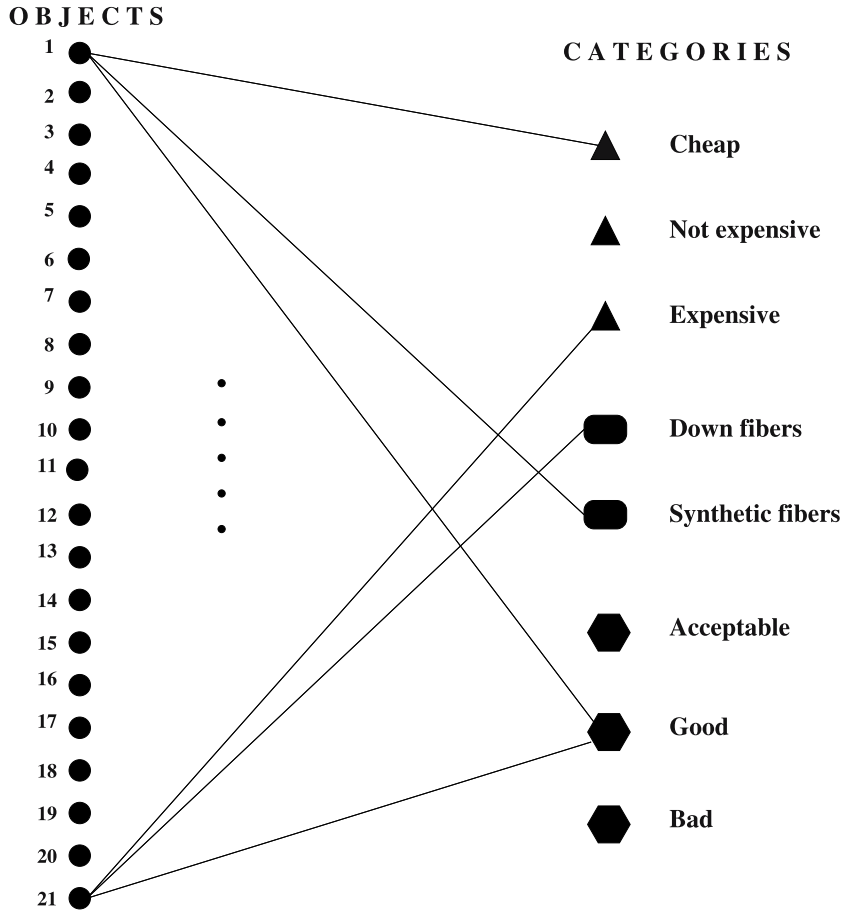

Figure 4.4. Graph representation of the sleeping bag data set presented in Table 4.2, with the *left set of nodes* corresponding to the objects (sleeping bags), the *right set of nodes* to the categories of the three attributes (price, fiber, quality), and selected *edges* capturing the relationship between objects and categories.

adhere to such aesthetic rules. The latter strategy proves particularly useful in the presence of large graphs and is adopted by several graph drawing systems, such as Nicheworks (Wills, 1997), GVF (Herman et al., 2000), and H3Viewer (Muentzer, 1998). Many systems also allow manual postprocessing of the resulting layout; see for example the Cytoscape visualization system (www.cytoscape.org).

The general problem of graph drawing discussed in this paper is to represent the edges of a graph as points in $\mathbb{R}^p$ and the vertices as lines connecting the points. Graph drawing is an active area in computer science, and it is very ably reviewed in the recent book by Di Battista et al. (1998). The choice of $\mathbb{R}^p$ is due to its attractive underlying geometry and the fact that it renders the necessary computations more manageable.

There are basically two different approaches to making such drawings. In the *metric* or *embedding* approach, one uses the path-length distance defined between the vertices of the graph and tries to approximate these distances by the Euclidean distance between the points. The area of embedding graph-theoretical distances is related to distance geometry, and it has been studied a great deal recently. In this paper, we adopt primarily the *adjacency model*, i.e., we do not emphasize graph-theoretical distances, but we pay special attention to which vertices are adjacent and which are not. Obviously, this is related to distance, but the emphasis is different. We use objective (loss) functions to measure the *quality* of the resulting embedding.

Force-directed Techniques

4.3.1

The class of graph-drawing techniques most useful for data visualization are *force-directed techniques*. This class of techniques borrows an analogy from classical physics, with the vertices being bodies with masses that attract and repel each other due to the presence of springs, or because the vertices have electric charges. This implies that there are 'physical' forces pulling and pushing the vertices apart, and the optimal graph layout will be one in which these forces are in equilibrium. An objective (loss) function that captures this analogy is given next:

$$Q(X|A,B) = \sum_{i=1}^{n}\sum_{j=1}^{n} a_{ij}\phi(d_{ij}(X)) - \sum_{i=1}^{n}\sum_{j=1}^{n} b_{ij}\psi(d_{ij}(X))\,, \tag{4.1}$$

where the $n \times p$ matrix X contains the coordinates of the n vertices in $\mathbb{R}^p$ and $d_{ij}(X)$ denotes the distances between points with coordinates x_i and x_j. The weights a_{ij} correspond to those in the adjacency matrix A of the graph G, while the pushing weights $B = \{b_{ij}\}$ could be derived either from the adjacency matrix or from an external constraint. Finally, the functions $\phi(\cdot)$ and $\psi(\cdot)$ are transformations whose role is to impose some aesthetic considerations on the layout. For example, a convex ϕ function will reinforce large distances by rendering them even larger and thus enable one to detect unique features in the data, while a concave transformation will dampen the effect of isolated vertices. Notice that this framework can accommodate both simple (i.e., $a_{ij} \in \{0,1\}$) and weighted (i.e., $a_{ij} \geq 0$) graphs. A popular force-directed technique that employs this pull-push framework is discussed in Di Battista et al. (1998), where the pulling is done by springs obeying Hooke's law (i.e., the force is proportional to the difference between the distance of the vertices and the zero-energy length of the spring), while the pushing is done by electrical forces following an inverse square law. Variations on this physical theme are used in several other algorithms (Fruchterman and Reingold 1991 and references therein).

Another way of incorporating a pushing component in the above objective function is through a normalization constraint. For example, one can require that $\eta(X) = 1$, and then the objective function takes the form by forming the Lagrangian:

$$Q(X|A) = \sum_{i=1}^{n}\sum_{j=1}^{n} a_{ij}\phi(d_{ij}(X)) - \lambda(\eta(X) - 1)\,. \tag{4.2}$$

It then becomes clear that the constraint term in the Lagrangian corresponds to the push component of $Q(\cdot)$. Examples of $\eta(X)$ include $\eta(X) = \text{trace}(X'X)$ or $\eta(X) = \det(X'X)$. Other possibilities include requiring the orthonormality of the points in the layout, such as $X'X = I_p$ or even fixing some of the Xs (Tutte, 1963).

Finally, this formulation allows one to incorporate into the force-directed framework the *metric* approach of graph drawing, where one works not with the adjacency matrix of the graph but with a distance matrix defined on the graph G. The goal then becomes to approximate graph-theoretic distances by Euclidean distances. Hence, the goal becomes to minimize

$$Q(X|W) = \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} \rho(d_{ij}(X)), \tag{4.3}$$

where

$$\rho(d_{ij}(X)) = (\eta(\delta_{ij}) - \eta(d_{ij}(X)))^2, \tag{4.4}$$

where $W = \{w_{ij}\}$ is a set of weights. The δ_{ij} correspond to path-length distances defined on the graph G, whereas the transformation η is usually the identity, the square, or the logarithm. Obviously $\rho(d(X))$ is not increasing and does not pass through zero; nevertheless, by expanding the square it becomes clear that it is equivalent to minimizing $Q(X|W)$ with $\phi(d) = \eta^2(d)$, $\psi(d) = \eta(d)$, and $b_{ij} = 2\eta(\delta_{ij})w_{ij}$. Thus, all points are pulling together, but points with large path-length distances are being pushed apart.

Next we examine in more detail the *metric* or *embedding* approach and the pulling under constraints model, which have proved particularly useful for drawing graphs obtained from data.

4.3.2 Multidimensional Scaling

The *metric* approach previously discussed corresponds to one version of multidimensional scaling (MDS). MDS is a class of techniques where a set of given distances is approximated by distances in low-dimensional Euclidean space. Formally, let $\{\delta_{ij},\ i, j = 1, \ldots, n\}$ be a set of distances. The goal is to identify the coordinates of n points x_i in $\mathbb{R}^p$ such that the Euclidean distance $d(x_i, x_j) \equiv d_{ij}(X)$ is approximately equal to δ_{ij}.

As mentioned before, for graph-drawing purposes, the δ_{ij} correspond to the shortest path distances defined on a graph G. A discussion of MDS as a graph-drawing technique is provided in Buja and Swayne (2002), where in addition other choices beyond Euclidean space are studied for the embedding space. The least-squares-loss (fit) function (known in the literature as Stress) introduced in Kruskal (1964) has the form

$$\sigma(X) = \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} (\delta_{ij} - d_{ij}(X))^2 \tag{4.5}$$

that is minimized over X. The w_{ij} are weights that can be chosen to reflect variability, measurement error, or missing data. This is precisely the objective function (4.3) derived from the general framework of force-directed techniques previously introduced and discussed.

A number of variations of (4.5) have appeared in the literature. In McGee (1966), 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}. On the other hand, the following choice of weights $w_{ij} = \delta_{ij}^{-1}$ is discussed in Sammon (1969).

Minimization of the loss function (4.5) can be accomplished either by an iterative majorization algorithm (Borg and Groenen 1997; De Leeuw and Michailidis 1998) or by a steepest descent method (Buja and Swayne 2002). The latter method is used in the implementation of MDS in the GGobi visualization system (Swayne et al., 1998). A 2-D MDS solution for the sleeping bag data is shown in Fig. 4.5. It can be seen that the solution spreads the objects in the data set fairly uniformly in the plane, and edge crossings are avoided.

We discuss next a fairly recent application of MDS. In many instances, the data exhibit nonlinearities, i.e., they lie on a low-dimensional manifold of some curvature. This has led to several approaches that still rely on the embedding (MDS) approach for visualization purposes but appropriately alter the input distances $\{\delta_{ij}\}$. A pop-

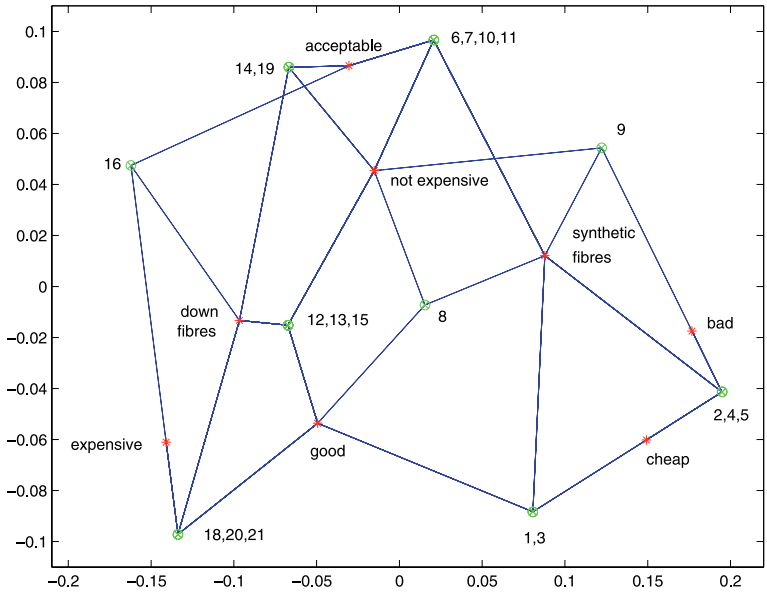

Figure 4.5. MDS representation of sleeping bag data set based on χ^2-distances. Due to the discrete nature of the data, multiple objects are mapped onto the same location, as shown in the *plot*. Further, for reference purposes, the categories to which the sleeping bags belong have been added to the plot at the *centroids* of the object points

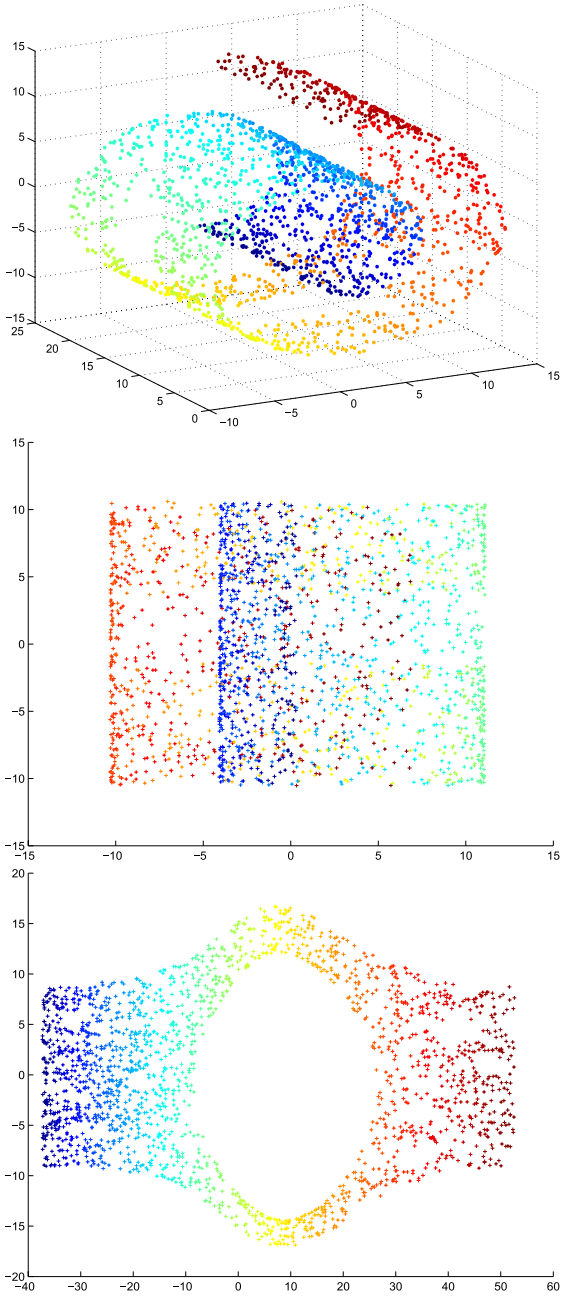

Figure 4.6. [This figure also appears in the color insert.] *Top panel*: original data (2000 data points) arranged along a nonlinear surface (Swiss Roll). *Middle panel*: 2-D MDS representation based on a complete weighted graph. *Bottom panel*: 2-D MDS representation based on 20 nearest-neighbor graph

ular and fairly successful nonlinear embedding technique is the Isomap algorithm (Tenenbaum et al., 2000). The main algorithm computes for each point in the data set a K-nearest neighbor graph and then stitches them together in the adjacency matrix. It then calculates distances using the resulting graph and then applies MDS. The main idea in the first step of the construction is to capture well the local geometry of the data. An illustration of the idea based on the Swiss Roll data set is shown next. Specifically, 2000 random points lying on a roll have been generated and their Euclidean pairwise distances computed. In addition, a graph that connects data points only with their closest 20 neighbors in the Euclidean metric was computed and shortest path distances calculated. Subsequently, MDS was applied to both distance matrices and a 2-D embedding obtained. The coloring scheme shows that straightforward MDS does not capture the underlying geometry of the roll (since the points do not follow their progression on the roll, blue, cyan, green, etc.), whereas the first dimension using the Isomap algorithm recovers the underlying structure. The hole in the middle is mostly due to the low density of orange points.

The Pulling Under Constraints Model

4.3.3

In this model, the similarity of the nodes is important. In the case of a simple graph, only connections between nodes are taken into consideration, whereas in a weighted graph, edges with large weights play a more prominent role. However, the normalization constraint, as discussed in Sect. 3, pushes points apart and avoids the trivial solution of all points collapsing to the origin. This model, under various distance functions, has been studied in a series of papers by Michailidis and de Leeuw (2001, 2004, 2005). We examine next the case of squared Euclidean distances, where $\phi(d(X)) \equiv d_{ij}^2(X)/2$, which turns out to be particularly interesting from a data visualization point of view. Some algebra shows that the objective function can be written in the following matrix algebra form:

$$Q(X|A) = \text{trace}(X'LX)\,, \tag{4.6}$$

where $L = D - A$ is the graph Laplacian (Chung, 1997), with D being a diagonal matrix containing the row sums of the adjacency matrix A. It can be seen that by minimizing (4.6), nodes sharing many connections would be pulled together, whereas nodes with few connections would end up on the periphery of the layout. For a weighted graph, the larger the weights, the stronger the bond between nodes and hence the more pronounced the clustering pattern.

A normalization constraint that leads to a computationally easy-to-solve problem is $X'DX = I_p$. Some routine calculations show that minimizing (4.6) subject to this constraint corresponds to solving a generalized eigenvalue problem. Further, notice that the solution is not orthogonal in Euclidean space, but in weighted (by D) Euclidean space. Figure 4.7 shows the graph layout for the small protein interactions network shown in Fig. 4.1. It can be seen that proteins PIB1 and BET1 that have very few interactions are located on the periphery of the layout. Moreover, the 'hub' pro-

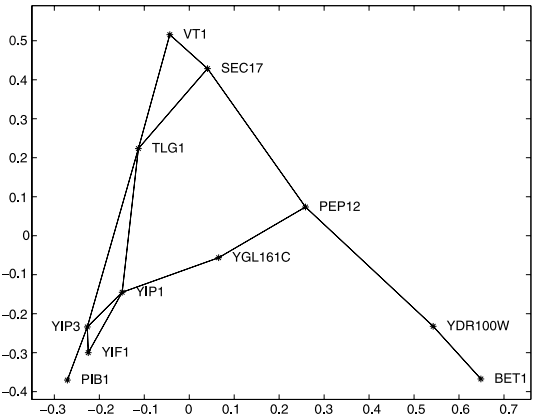

Figure 4.7. Two-dimensional layout of the small protein interaction network shown in Fig. 4.1

teins TLG1 and YIP1 are positioned close to the center of the plot, signifying their central role in this network.

The next example comes from the UCI machine learning repository. The data set consists of features of handwritten numerals (0–9) extracted from a collection of Dutch utility maps. There are 200 patterns per class, and 240 variables characterizing the pixel intensity of the underlying digital image of the digit have been collected. The pixel intensities are categorical and take values in the 0 to 7 range. This is an example where linear techniques such as principal component analysis fail to separate the classes (see top panel of Fig. 4.8).

The next set of plots in Fig. 4.9 shows the layouts of a few large graphs that have been used for testing graph partitioning algorithms (Walshaw, 2003). The first graph is comprised of 4720 vertices and 13 722 edges, the second of 10 240 vertices and 30 380 edges, and the third of 4253 vertices and 12 289 edges. They are derived from computational mechanics meshes and characterized by extreme variations in the mesh density and the presence of "holes." The layouts shown are based on weighted graphs that were built by considering for each vertex its ten nearest neighbors in the Euclidean metric and calculating exponentially decreasing weights. It can be seen that the layouts capture, to a large extent, the underlying structure of the graphs in terms of density and the presence of "holes."

4.3.4 Bipartite Graphs

As noted in Sect. 2, the graph representation of a contingency table and of a categorical data set has some special features, namely, the node set V can be partitioned into two subsets. For example, in the case of a contingency table, the categories of one variable form the first subset and those of the other variable the second one. Notice that there are only connections between members of these two subsets. An analogous situation arises in the case of categorical data, where the first subset of nodes corresponds to the objects (e.g., the sleeping bags) and the second subset to the categories

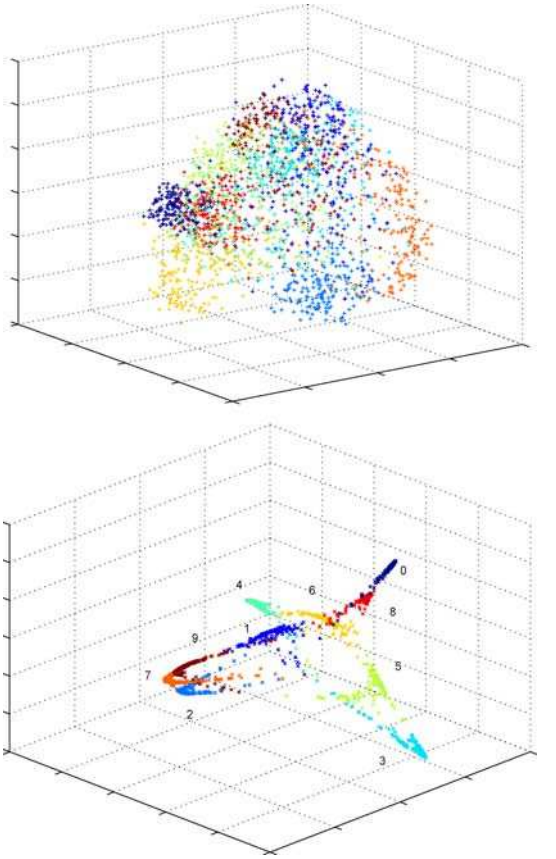

Figure 4.8. [This figure also appears in the color insert.] PCA layout of digits dataset (*top panel*) and the 3-D graph layout (*bottom panel*)

of all the variables. These are two instances where the resulting graph representation of the data gives rise to a *bipartite* graph. A slight modification of the $Q(\cdot)$ objective function leads to interesting graph layouts of such data sets. Let $X = [Z' \;\; Y']$, where Z contains the coordinates of the first subset of the vertices and Y those of the second subset. The objective function for squared Euclidean distances can then be written as (given the special block structure of the adjacency matrix A)

$$Q(Z, Y|A) = \text{trace}(Z'D_Z Z + Y'D_Y Y - 2Y'AZ)\,, \tag{4.7}$$

where D_Y is a diagonal matrix containing the column sums of A and D_Z another diagonal matrix containing the row sums of A. In the case of a contingency table, both D_Y and D_Z contain the marginal frequencies of the two variables, while for a multivariate categorical data set D_Y contains again the univariate marginals of all the categories of all the variables and $D_Z = JI$ is a constant multiple of the identity matrix, with J denoting the number of variables in the data set. A modification of

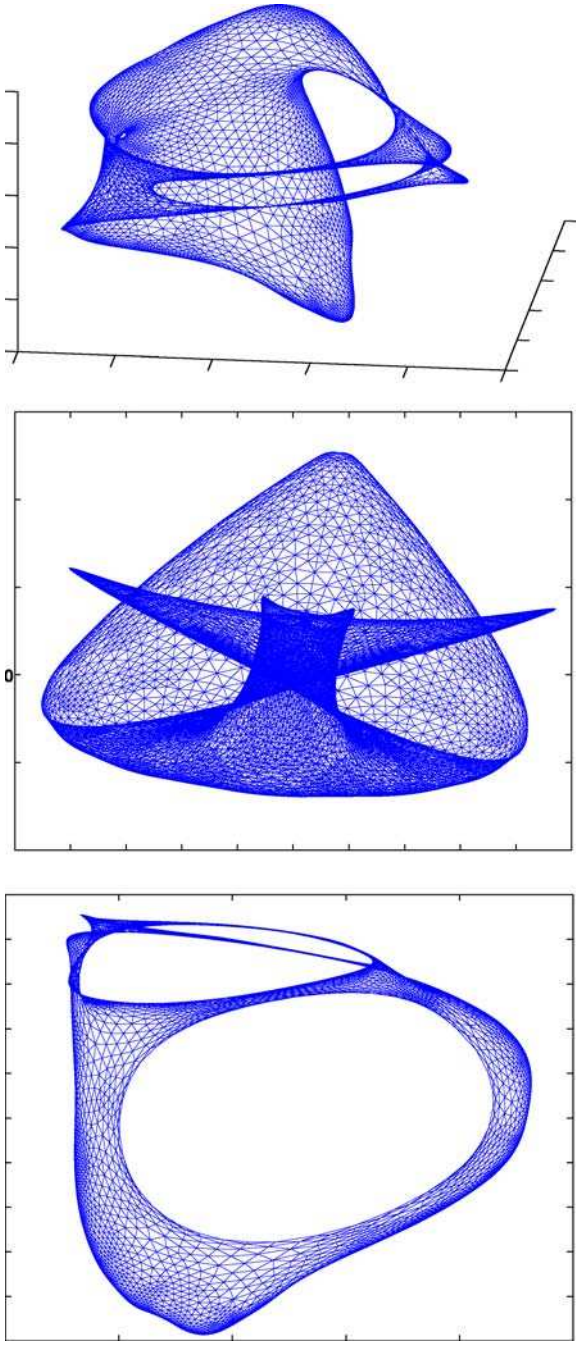

Figure 4.9. Layouts of large graphs derived from computational mechanics meshes and characterized by varying degrees of mesh density

the normalization constraint to this setting, namely, $Z'D_Z Z = I_p$, gives the following solution, which can be obtained through a block relaxation algorithm (Michailidis and de Leeuw, 1998):

$$Z = J^{-1}AY \text{ and } Y = D_Y^{-1}A'XZ.$$

Hence, the optimal solution satisfies the *centroid principle* (Gifi, 1990), which says that the category points in the optimal layout are at the center of gravity of the objects that belong to them. The above graph-drawing solution is known in multivariate analysis for contingency tables as correspondence analysis and for multivariate categorical data sets as multiple correspondence analysis (Michailidis and de Leeuw, 1998).

Figure 4.10 shows the graph layout of the sleeping bags data set. The solution captures the basic patterns in the data set, namely, that there are good-quality, expensive sleeping bags filled with down fibers and cheap, bad-quality sleeping bags filled with synthetic fibers. Further, there exist some sleeping bags of intermediate quality and price filled with either down or synthetic fibers. Notice that the centroid principle resulting from the partitioning of the vertex set proves useful in the interpretation of the layout. Further, the resulting layout is less 'uniform' than the one obtained through MDS and thus better captures features of the data.

It is interesting to note that the choice of the distance function coupled with a particular normalization has a significant effect on the aesthetic quality of the resulting

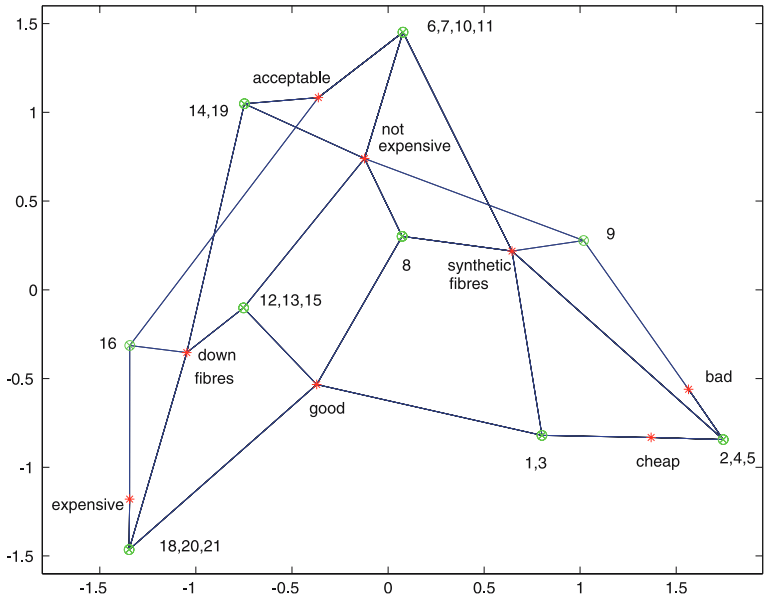

Figure 4.10. Graph layout of sleeping bags data based on objective function (4.7). Due to the discrete nature of the data, multiple objects are mapped on the same location, as shown in the plot. Further, for reference purposes, the categories to which the sleeping bags belong have been added to the plot at the *centroids* of the object points

graph layout. An extreme case occurs when $\phi(d(X)) \equiv d_{ij}(X)$ corresponds to Euclidean distances. Then, under the orthonormality constraint, the solution is rather uninteresting and consists of exactly $p + 1$ points, where p is the dimensionality of the embedding space. A mathematical explanation of this result is given for the 1-D case ($p = 1$) in de Leeuw and Michailidis (2004) and is also illustrated for the higher-dimensional case ($p \geq 2$) in Michailidis and de Leeuw (2005).

4.4 Discussion and Concluding Remarks

In this paper, the problem of data visualization through layouts of their graph representations is considered. A mathematical framework for graph drawing based on force-directed techniques is introduced, and several connections to well-known multivariate analysis techniques such as multidimensional scaling, correspondence, and multiple correspondence analysis are made.

Several extensions that may improve the quality of the graph layout are possible within this general framework. For example, logistic loss functions are explored in de Leeuw (2005), together with the arrangement of the nodes along Voronoi cells. The visualization of several related data sets through multilevel extensions of multiple correspondence analysis are explored in Michailidis and de Leeuw (2000). Finally, a version of multidimensional scaling for data that change over time is discussed in Costa et al. (2004).

Acknowledgement. This work was partially supported by NIH grant 5P41RR018627-03. The author would like to thank Jan de Leeuw for many fruitful discussions and suggestions on the topic over the course of the last 10 years and the editors of the Handbook for many comments that improved the presentation of the material.

References

Borg, I. and Groenen, P. (1997). *Modern Multidimensional Scaling: Theory and Applications*. Springer, Berlin Heidelberg New York.

Buja, A. and Swayne, D. (2002). Visualization methodology for multidimensional scaling. *J Classificat*, 19:7–43.

Chung, F.R.K. (1997). *Spectral Graph Theory*. American Mathematical Society, Providence, RI.

Costa, J., Patwari, N. and Hero, A.O. (2004). Distributed multidimensional scaling with adaptive weighting for node localization in sensor networks. *ACM J Sensor Network*, 2(1):39–64.

www.cytoscape.org.

De Leeuw, J. (2005). *Nonlinear Principal Component Analysis and Related Techniques*. Preprint # 427, Department of Statistics, University of California, Los Angeles.

De Leeuw, J. and Michailidis, G. (1998). Graph layout techniques and multidimensional data analysis. In: Bruss, F.T., Le Cam, L. (eds) *Game Theory, Optimal Stop-*

ping, Probability and Statistics. IMS Lecture Notes – Monograph Series, 35:219–248.

De Leeuw, J. and Michailidis, G. (2004). Weber correspondence analysis: the one-dimensional case. *J Comput Graph Stat*, 13:946–953.

Di Battista, G., Eades, P., Tamassia, R. and Tollis, I. (1994). Algorithms for drawing graphs: an annotated bibliography. *Comput Geom Theory Appl*, 4:235–282.

Di Battista, G., Eades, P., Tamassia, R. and Tollis, I. (1998). *Graph Drawing: Algorithms for Geometric Representation of Graphs*. Prentice-Hall, Upper Saddle River, NJ.

Eick, S.G. and Wills, G.J. (1993a) Navigating large networks with hierarchies. In: *Proceedings of Visualization Conference*, pp. 204–210.

Eick, S.G. and Wills, G.J. (1993b) High interaction graphics. *Eur J Operat Res*, 84:445–459.

Fisher, R.A. (1938). The precision of discriminant functions. *Ann Eugen*, 10:422–429.

Fruchterman, T.M.J. and Reingold, E.M. (1991). Graph drawing by force-directed placement. *Software: practice and engineering*, 21:1129–1164.

Gifi, A. (1990). *Nonlinear Multivariate Analysis*. Wiley, Chichester.

www.ggobi.org.

Herman, I., Melancon, G. and Marshall, M.S. (2000). Graph visualization and navigation in information visualization. *IEEE Trans Visualizat Comput Graph*, 6:24–43.

Ito, T., Tashiro, K., Muta, S., Ozawa, R., Chiba, T., Nishizawa, M., Yamamoto, K., Kuhara, S. and Sakaki, Y. (2000). Toward a protein-protein interaction map of the budding yeast: a comprehensive system to examine two-hybrid interactions in all possible combinations between the yeast interactions. *Proc Natl Acad Sci USA*, 97:1143–1147.

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

McGee, V.E. (1966). The multidimensional analysis of elastic distances. *Br J Math Stat Psychol*, 19:181–196.

Michailidis, G. and de Leeuw, J. (1998). The Gifi system of descriptive multivariate analysis. *Stat Sci*, 13:307–336.

Michailidis, G. and de Leeuw, J. (2000). Multilevel homogeneity analysis with differential weighting. *Comput Stat Data Anal*, 32:411–442.

Michailidis, G. and de Leeuw, J. (2001). Data visualization through graph drawing. *Comput Stat*, 16:435–450.

Michailidis, G. and de Leeuw, J. (2005). Homogeneity analysis using absolute deviations. *Comput Stat Data Anal*, 48:587–603.

Muentzer, T. (1998). Drawing large graphs with H3Viewer and Site Manager. In: Proceedings of Graph Drawing 98. *Lecture Notes in Computer Science*, vol 1547. Springer, Berlin, Heidelberg, New York, pp. 384–393.

Prediger, S. (1997). Symbolic objects in formal concept analysis. In: Mineau, G., Fall, A. (eds) *Proceedings of the 2nd international symposium on knowledge, retrieval, use and storage for efficiency*.

Sammon, J.W. (1969). A nonlinear mapping for data structure analysis. *IEEE Trans Comput*, 18:401–409.

Swayne, D.F., Cook, D. and Buja, A. (1998). XGobi: interactive dynamic data visualization in the X Window system. *J Comput Graph Stat*, 7:113–130.

Tenenbaum, J.B., da Silva, V. and Langford, J.C. (2000). A global geometric framework for nonlinear dimensionality reduction. *Science*, 22:2319–2323.

Tutte, W.T. (1963). How to draw a graph. *Proc Lond Math Soc*, 13:743–767.

Walshaw, C. (2003). A multilevel algorithm for force-directed graph drawing *J Graph Algor Appl*, 7:253–285.

Wills, G.J. (1997). NicheWorks – interactive visualization of very large graphs. In: *Proceedings of the conference on graph drawing*. Springer, Berlin Heidelberg New York.