RR 92-07

MISSING DATA IN THE DISTANCE APPROACH TO PRINCIPAL COMPONENTS ANALYSIS

Jacques J.F. Commandeur

Department of Data Theory
University of Leiden

MISSING DATA IN THE DISTANCE APPROACH TO PRINCIPAL COMPONENTS ANALYSIS

Abstract

In the distance approach to principal components analysis (PCA), a multidimensional scaling framework is used to approximate the high-dimensional distances between the objects of a multivariate data matrix with distances in a low-dimensional configuration. In this paper the distance approach to PCA is extended to the case where the multivariate data matrix is incomplete. An available case method is proposed where the root of the mean squared differences between the nonmissing elements of each pair of objects is used as a derived dissimilarity, to be approximated in low-dimensional space. The latter is combined with an approximation procedure which puts a larger emphasis on dissimilarities derived from ample between-object information than on dissimilarities derived from scarce between-object information. The method is illustrated with two examples. The solutions are compared with the complete case and with the results obtained with classical nonlinear PCA.

Key words: missing data, multidimensional scaling, principal components analysis, nonlinear transformations, distance approximation, majorization.

1 Introduction

The purpose of this paper is to generalize the multidimensional scaling framework proposed by De Leeuw and Meulman (1985) and Meulman (1986, 1992) for principal components analysis (PCA) to the case of missing data. These authors discussed the following problem. Given a data matrix $\mathbf{Z}$ containing information about m variables on n objects, how can we determine a configuration $\mathbf{X}$ of order (n x p), such that the Euclidean distances $d_{ij}(\mathbf{X})$ between the rows of $\mathbf{X}$ (conceived of as n points in p dimensions) approximate the Euclidean distances $d_{ij}(\mathbf{Z})$ between the rows of $\mathbf{Z}$ as well as possible for all $i, j = 1, \ldots, n$. The authors considered both the linear and the nonlinear case, that is, the case where the m columns of $\mathbf{Z}$ may contain either numerical, ordinal or nominal variables. If the approximation of $d_{ij}(\mathbf{Z})$ by $d_{ij}(\mathbf{X})$ is understood in a least squares sense, the formal problem is how to minimize the least squares loss function

$$f(\tilde{\mathbf{Z}}, \mathbf{X}) \equiv 1/2 \sum_{i=1}^{n} \sum_{j=1}^{n} [d_{ij}(\tilde{\mathbf{Z}}) - d_{ij}(\mathbf{X})]^2. \tag{1}$$

In (1), $\tilde{\mathbf{Z}}$ denotes the (n x m) matrix containing the *transformed* variables of $\mathbf{Z}$, $d^2_{ij}(\tilde{\mathbf{Z}}) \equiv (\tilde{\mathbf{z}}_i - \tilde{\mathbf{z}}_j)'(\tilde{\mathbf{z}}_i - \tilde{\mathbf{z}}_j)$, where $\tilde{\mathbf{z}}_i$ and $\tilde{\mathbf{z}}_j$ are column vectors of order (m x 1) containing rows i and j of $\tilde{\mathbf{Z}}$, and $d^2_{ij}(\mathbf{X}) \equiv (\mathbf{x}_i - \mathbf{x}_j)'(\mathbf{x}_i - \mathbf{x}_j)$, where $\mathbf{x}_i$ and $\mathbf{x}_j$ are column vectors of order (p x 1) containing rows i and j of matrix $\mathbf{X}$.

Loss function (1) belongs to a specific class of loss functions which are collectively called STRESS as introduced by Kruskal (1964). Meulman (1986, 1992) discussed a whole new class of techniques, generated by applying the distance approach associated with STRESS to (classical) multivariate analysis, and specifically proposed loss function (1) as an alternative for classical principal components analysis and its nonlinear variants.

Generally speaking, classical (nonlinear) PCA and loss function (1) both aim at representing points in high-dimensional space in low(er)-dimensional space, such that certain properties of high-dimensional space are retained as well as possible. Moreover, in allowing for transformations of the variables to improve the fit of the solution in the nonlinear case, both methods are concerned with replacing the given high-dimensional space $\mathbf{Z}$ by another high-dimensional space $\tilde{\mathbf{Z}}$.

This is where the resemblance stops, however, because classical (nonlinear) PCA and PCA via STRESS retain *different properties* of high-dimensional space. Restricting ourselves to the differences as they apply to the relations between the *objects* in the data, De Leeuw and Meulman have shown that classical (nonlinear) PCA approximates the squared Euclidean distances between the objects in the (transformed) data matrix with squared p-dimensional distances. Also, since the *projection* of high-dimensional space on low-dimensional space is a central feature of classical PCA, the distances between the n objects in low-dimensional space will perforce always be smaller than the original distances. To use the terminology of Meulman and De Leeuw, the approximation of high-dimensional distances in classical PCA and its nonlinear variants typically is *quadratic* and *from below*. With this method it is not uncommon to find large distances between objects in high-dimensional space being represented by small, or even zero, distances in low-dimensional space.

In PCA via STRESS, on the other hand, the approximation of the distances is *direct* and *from both sides*, to quote De Leeuw and Meulman again. The term 'direct' refers to the fact that the (unsquared) high-dimensional distances are approximated by (unsquared) low-dimensional distances. The approximation in (1) is 'from both sides' because the p-dimensional distances are allowed to be smaller as well as larger than the original distances. Moreover, it is less likely to find a large $d_{ij}(\tilde{\mathbf{Z}})$ being represented by a (very) small $d_{ij}(\mathbf{X})$ when using (1), because this would entail a large residual value for objects i and j, a situation that (1) is bound to avoid. Therefore, if one's objective is to obtain a direct optimal low-dimensional representation of the distances between the objects in m-dimensional space, the use of PCA via STRESS seems more appropriate than classical PCA and its nonlinear variants.

De Leeuw and Meulman (1985), and Meulman (1986, 1992), showed that (1) can be minimized by using a special majorization algorithm derived from the general majorization algorithm model proposed by De Leeuw and Heiser (1980). For fixed $\tilde{\mathbf{Z}}$, a convergent series of new configurations $\mathbf{X}$ is obtained by repeatedly computing the *Guttman transform*. In the linear case, that is all that needs to be done. In the non-linear case, an additional step is required where (1) is minimized with respect to $\tilde{\mathbf{Z}}$ for fixed $\mathbf{X}$. Meulman and De Leeuw discussed that this is achieved by computing yet another Guttman transform, followed by solving a *metric projection* problem which takes into account the (possibly different) measurement level(s) of the variables of $\mathbf{Z}$. In the non-linear case, the majorization algorithm alternates between these two steps until convergence is reached.

Their results are based on the assumption that the given matrix $\mathbf{Z}$ is *complete*, that is, that information about all m variables is available for all n objects. Unfortunately, in practice this is often not the case. On this subject Meulman (1986, p.177) remarked:

> "At this point there is enough indication that missing data problems can also be solved by STRESS algorithms, but the actual implementation, taking care of the painstaking details, has yet to be done."

This paper addresses the problem how to take care of these painstaking details for loss function (1), that is, it discusses how to determine a low-dimensional configuration **X** when the given data matrix **Z** contains missing entries in any arbitrary pattern.

In section 2 we first delineate a strategy for handling missing data, and show how to formalize this strategy into a generalized version of loss function (1). In section 3, 4, and 5 a convergent algorithm is developed for the estimation of the unknown parameters of this generalized loss function. Section 6 discusses the problem how to obtain initial estimates for the unknown parameters in the incomplete case. In section 7 we discuss some measures for the evaluation of the goodness-of-fit of the solutions, and also explain why the analysis of incomplete data generally yields larger residuals than the analysis of complete data. Finally, in section 8 two examples are presented of the STRESS analysis of an incomplete data matrix, and the results are compared with the analysis of the complete data matrix as well as with the results obtained in classical (nonlinear) PCA.

2 Generalizing PCA via STRESS to incomplete Z

In the literature on data analysis, many ways of handling missing data have been proposed. For a comprehensive overview see, for instance, Meulman (1982), and Little and Rubin (1987). We may usefully distinguish those approaches where values are inserted or estimated for the missing entries (either before or during the analysis) from those where no estimation or insertion takes place. Little and Rubin called the latter approaches *available case* methods. Little and Rubin's main criticism of available case methods is that they result in covariance and correlation matrices that are not necessarily positive definite. This calls for ad hoc modifications in analyses based on the covariance matrix, since these often require a positive definite matrix. However, PCA via STRESS is not based on the covariance or correlation matrix, and therefore no such ad hoc modifications are needed (although indefiniteness does have its repercussions in another area, as will be discussed in section 7). Inserting the mean values of the non-missing elements of the corresponding variables for the missing entries, and other fill-in strategies, have the obvious disadvantage that they at best provide an educated guess. More sophisticated estimation procedures for incomplete data, like the likelihood-based methods proposed by Little and Rubin and others, require assumptions about the probability distribution of the variables and about the randomness properties of the missing data. The available case method does not require any of these assumptions, and we will therefore present a solution where missing entries in the data matrix **Z** play no role whatsoever in the estimation of **X**.

As a first step in this direction, we redefine the squared Euclidean distance between rows i and j of incomplete $\tilde{\mathbf{Z}}$ as

$$\delta_{ij}^2 = \frac{1}{w_{ij}} d_{ij}^2(\tilde{\mathbf{Z}}\mathbf{M}_i\mathbf{M}_j). \qquad (2)$$

The root of (2) acts as a derived dissimilarity to be approximated in low-dimensional space. In (2),

$$w_{ij} \equiv (\mathrm{tr}\ \mathbf{M}_i\mathbf{M}_j) \qquad (3)$$

denotes the number of nonmissing dimensions for row pair i and j, $\mathbf{M}_i$ is the diagonal matrix of order (m x m) with ones on the diagonal where row i of $\mathbf{Z}$ contains information and zeroes elsewhere, and

$$d_{ij}^2(\tilde{\mathbf{Z}}\mathbf{M}_i\mathbf{M}_j) \equiv (\tilde{\mathbf{z}}_i - \tilde{\mathbf{z}}_j)'\mathbf{M}_i\mathbf{M}_j(\tilde{\mathbf{z}}_i - \tilde{\mathbf{z}}_j). \qquad (4)$$

In words, the effect of (4) is that the Euclidean distance is only calculated between those (possibly transformed) variables *for which information is available in both rows.* Thus, whenever information on a variable is missing in either one of the two rows this variable is not involved in the calculation of the distance. Stated differently, in (4) the distance is assumed to be *zero* for the 'missing' dimensions (i.e., for variables not containing information in both rows). This has the disadvantage that, in the long run, the average distance between pairs of rows containing lots of missing data will be smaller than the average distance between pairs of rows with hardly any or no missing data. To correct for this undesirable property we divide (4) by w_{ij} in (3) to obtain (2), which is the *mean* of the squared nonmissing differences between rows i and j. In the latter definition, the squared distance on each of the 'missing' dimensions is implicitly assumed to be equal to the mean squared distance on the nonmissing dimensions. This can be seen as follows. If we 'complete' the distance between rows i and j by replacing the distance on each missing dimension by (2), and then reapply definition (2) to the thus 'completed' distance, we obtain

$$\frac{1}{m} \{ (\tilde{\mathbf{z}}_i - \tilde{\mathbf{z}}_j)'\mathbf{M}_i\mathbf{M}_j(\tilde{\mathbf{z}}_i - \tilde{\mathbf{z}}_j) + \frac{m-w_{ij}}{w_{ij}} (\tilde{\mathbf{z}}_i - \tilde{\mathbf{z}}_j)'\mathbf{M}_i\mathbf{M}_j(\tilde{\mathbf{z}}_i - \tilde{\mathbf{z}}_j) \}$$

$$= \frac{1}{w_{ij}} (\tilde{\mathbf{z}}_i - \tilde{\mathbf{z}}_j)'\mathbf{M}_i\mathbf{M}_j(\tilde{\mathbf{z}}_i - \tilde{\mathbf{z}}_j) = \delta_{ij}^2,$$

which is exactly equal to the squared distance we already had for incomplete rows i and j.

Notice that (2) is not defined when $w_{ij} = 0$, that is, when two rows lack common information on all variables (which is not the same as 'have nothing in common'; that, we simply do not know). To handle this situation, and for other reasons to be explained shortly, we additionally use w_{ij} defined in (3) to obtain a *weighted* solution. Interpreted as weights, the terms defined in (3) reflect the *amount* of information shared by two objects, and are completely unrelated to the information itself. They satisfy the condition $0 \leq w_{ij} \leq m$. The larger the weight, the more information is shared by rows i and j. More specifically, a zero weight arises whenever two objects lack common information on all variables, while an m-weight implies that the corresponding two rows are both complete. A value in between 0 and m indicates that one or both rows contain at least one missing entry. At the same time, it indicates that the two objects share information on at least one variable.

Before further discussing the effect of weights (3), it is necessary to show how they are implemented. Therefore, we now present the following generalized loss function for a distance driven PCA of incomplete $\mathbf{Z}$:

$$\begin{aligned} f(\tilde{\mathbf{Z}}, \mathbf{X}) &\equiv 1/2 \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij}[\delta_{ij} - d_{ij}(\mathbf{X})]^2 \\ &= 1/2 \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij}[w_{ij}^{-1/2} d_{ij}(\tilde{\mathbf{Z}}\mathbf{M}_i\mathbf{M}_j) - d_{ij}(\mathbf{X})]^2. \qquad (5) \end{aligned}$$

In (5), δ_{ij} is defined as in (2), w_{ij} is defined as in (3), $d_{ij}(\tilde{\mathbf{Z}}\mathbf{M}_i\mathbf{M}_j)$ is defined as in (4), and $d_{ij}^2(\mathbf{X}) \equiv (\mathbf{x}_i - \mathbf{x}_j)'(\mathbf{x}_i - \mathbf{x}_j)$.

Definitions (2), (3), and (5) entail a number of convenient properties. If two objects do not share information on any one variable, their distance will be treated as missing in $\tilde{\mathbf{Z}}$ as well as in $\mathbf{X}$. Conversely, this means that the solution for each row of $\mathbf{X}$ (i.e., its location as a point in p-dimensional space) will always be based on its relation with respect to those other objects with which it shares information on at least one variable. Moreover, pairs of objects sharing information on lots of variables will be penalized more heavily for differences between their distance in high- and low-dimensional space than pairs of objects sharing information on only a few variables. Stated differently, the algorithm to be developed below will make larger efforts to fit distances based on ample between-object information than to fit distances based on scarce between-object information.

Since it follows from (4) that $d_{ij}(\tilde{\mathbf{Z}}\mathbf{M}_i\mathbf{M}_j) = 0$ if $w_{ij} = 0$, (5) may be written in the equivalent form

$$f(\tilde{\mathbf{Z}}, \mathbf{X}) = 1/2 \sum_{i=1}^{n} \sum_{j=1}^{n} [d_{ij}(\tilde{\mathbf{Z}}\mathbf{M}_i\mathbf{M}_j) - w_{ij}^{1/2} d_{ij}(\mathbf{X})]^2. \qquad (6)$$

Notice that if $\mathbf{Z}$ is complete, $\mathbf{M}_i = \mathbf{I}_m$ and therefore also $w_{ij} = m$ for $i, j = 1, \ldots, n$, meaning that loss function (6) then reduces to

$$f(\tilde{\mathbf{Z}}, \mathbf{X}) = 1/2 \sum_{i=1}^{n} \sum_{j=1}^{n} [d_{ij}(\tilde{\mathbf{Z}}) - m^{1/2} d_{ij}(\mathbf{X})]^2,$$

which only differs from (1) by the scaling constant $m^{1/2}$.

To avoid wordy descriptions, in the sequel we will refer to loss function (6) as PRINCESS, an abbreviation or shorthand for PRINcipal Components analysis via strESS. In section 3 we discuss the problem how to minimize (6) with respect to $\mathbf{X}$ for fixed $\tilde{\mathbf{Z}}$, and section 4 is devoted to the minimization of (6) with respect to $\tilde{\mathbf{Z}}$ for fixed $\mathbf{X}$. As the following sections will testify, it is the latter problem that needs the largest re-consideration.

3 Minimizing the generalized loss function with respect to X

In this section we assume $\tilde{\mathbf{Z}}$ to be fixed, and discuss the problem how to minimize

$$f(\mathbf{X}) = 1/2 \sum_{i=1}^{n} \sum_{j=1}^{n} [d_{ij}(\tilde{\mathbf{Z}}\mathbf{M}_i\mathbf{M}_j) - w_{ij}^{1/2} d_{ij}(\mathbf{X})]^2. \qquad (7)$$

Letting

$$\eta^2(\tilde{\mathbf{Z}}) = 1/2 \sum_{i=1}^{n} \sum_{j=1}^{n} d_{ij}^2(\tilde{\mathbf{Z}}\mathbf{M}_i\mathbf{M}_j), \qquad (8)$$

$$\eta^2(\mathbf{X}) = 1/2 \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} d_{ij}^2(\mathbf{X}), \qquad (9)$$

and

$$\rho(\mathbf{X}) = 1/2 \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij}^{1/2} d_{ij}(\tilde{\mathbf{Z}}\mathbf{M}_i\mathbf{M}_j) d_{ij}(\mathbf{X}), \qquad (10)$$

it is easily verified that (7) may be written as

$$f(\mathbf{X}) = \eta^2(\tilde{\mathbf{Z}}) + \eta^2(\mathbf{X}) - 2\,\rho(\mathbf{X}). \qquad (11)$$

By a straightforward application of the derivations discussed in De Leeuw and Heiser (1977, 1980) and Heiser and De Leeuw (1977), we find that (9) may be written as

$$\eta^2(\mathbf{X}) = 1/2 \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} d_{ij}^2(\mathbf{X}) = \text{tr } \mathbf{X}'\mathbf{V}\mathbf{X}, \tag{12}$$

where matrix $\mathbf{V} = \{v_{ij}\}$ is defined by

$$v_{ij} = \begin{cases} -w_{ij} & \text{for } i \neq j \\ \sum_{k \neq i}^{n} w_{ik} & \text{for } i = j \end{cases}, \tag{13}$$

and that (10) can be written as

$$\rho(\mathbf{X}) = 1/2 \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij}^{1/2} d_{ij}(\tilde{\mathbf{Z}}\mathbf{M}_i\mathbf{M}_j) d_{ij}(\mathbf{X}) = \text{tr } \mathbf{X}'\mathbf{B}(\mathbf{X})\mathbf{X}, \tag{14}$$

where matrix $\mathbf{B}(\mathbf{X}) = \{b_{ij}(\mathbf{X})\}$ is defined by

$$b_{ij}(\mathbf{X}) = \begin{cases} \left.\begin{array}{l} -w_{ij}^{1/2} d_{ij}(\tilde{\mathbf{Z}}\mathbf{M}_i\mathbf{M}_j)/d_{ij}(\mathbf{X}) \\ 0 \quad \text{if } d_{ij}(\mathbf{X}) = 0 \end{array}\right\} & \text{for } i \neq j \\ -\sum_{k \neq i}^{n} b_{ik}(\mathbf{X}) & \text{for } i = j \end{cases}. \tag{15}$$

Therefore, (7) can be written as

$$f(\mathbf{X}) = \eta^2(\tilde{\mathbf{Z}}) + \text{tr } \mathbf{X}'\mathbf{V}\mathbf{X} - 2 \text{ tr } \mathbf{X}'\mathbf{B}(\mathbf{X})\mathbf{X}, \tag{16}$$

where the term $\eta^2(\tilde{\mathbf{Z}})$ is independent of $\mathbf{X}$. A convergent algorithm for the minimization of (16) is obtained by repeatedly computing the *Guttman transform* defined as

$$\mathbf{X}^u = \mathbf{V}^+\mathbf{B}(\mathbf{X}^o)\mathbf{X}^o, \tag{17}$$

where $\mathbf{V}^+$ denotes the Moore-Penrose inverse of $\mathbf{V}$, and $\mathbf{X}^o$ is the current optimal $\mathbf{X}$. We skip all proofs, not only because these can be found elsewhere (cf., De Leeuw, 1977; De Leeuw and Heiser, 1980), but also because section 4 will provide these proofs in a generalized form. First, however, we will discuss ways to determine the Moore-Penrose inverse of **V** as required for the calculation of (17).

Although the Moore-Penrose of matrix **V** only has to be calculated once, this task easily becomes enormous if the number of objects is large. However, because we may assume without loss of generality that matrix **V** is irreducible, and since the rank of **V** is $(n-1)$ in that case, its generalized inverse can be calculated as

$$\mathbf{V}^{+} = (\mathbf{V} + \frac{1}{n}\mathbf{1}\mathbf{1}')^{-1} - \frac{1}{n}\mathbf{1}\mathbf{1}', \tag{18}$$

where **1** is an $(n \times 1)$ vector of ones (cf. De Leeuw, 1977). Thus, we only need to compute a real inverse of order $(n \times n)$. For large n, this still is an expensive job in terms of computation time.

Fortunately, in practice the number of variables in the analysis (m) will often be (much) smaller than the number of objects (n). We may take advantage of this situation by reducing (18) to a problem which only requires the calculation of the real inverse of an $(m \times m)$ matrix. To show how this is achieved, we first note that **V** may be written as

$$\mathbf{V} = (\text{diag } \mathbf{H}\mathbf{H}'\mathbf{1}\mathbf{1}') - \mathbf{H}\mathbf{H}', \tag{19}$$

where (diag $\mathbf{H}\mathbf{H}'\mathbf{1}\mathbf{1}'$) denotes the $(n \times n)$ diagonal matrix obtained by setting all off-diagonal elements of the matrix product $\mathbf{H}\mathbf{H}'\mathbf{1}\mathbf{1}'$ equal to zero, and **H** is defined as the $(n \times m)$ binary matrix containing ones where information in **Z** is available, and zeroes elsewhere. Letting $\mathbf{L} =$ (diag $\mathbf{H}\mathbf{H}'\mathbf{1}\mathbf{1}'$), and $\mathbf{E} = \mathbf{L} + (1/n)\mathbf{1}\mathbf{1}'$, both of order $(n \times n)$, we may write (18) as

$$\mathbf{V}^{+} = (\mathbf{E} - \mathbf{H}\mathbf{H}')^{-1} - \frac{1}{n}\mathbf{1}\mathbf{1}'. \tag{20}$$

According to the Woodbury formula (see, for instance, Press et al., 1986),

$$(\mathbf{A} + \mathbf{U}\mathbf{T}')^{-1} = \mathbf{A}^{-1} - \mathbf{A}^{-1}\mathbf{U}(\mathbf{I} + \mathbf{T}'\mathbf{A}^{-1}\mathbf{U})^{-1}\mathbf{T}'\mathbf{A}^{-1} \tag{21}$$

for nonsingular matrix **A** of order $(n \times n)$, and arbitrary matrices **U** and **T** of order $(n \times m)$. Applying (21) to the first term on the right hand side of (20), we obtain

$$(\mathbf{E} - \mathbf{H}\mathbf{H}')^{-1} = \mathbf{E}^{-1} + \mathbf{E}^{-1}\mathbf{H}(\mathbf{I} - \mathbf{H}'\mathbf{E}^{-1}\mathbf{H})^{-1}\mathbf{H}'\mathbf{E}^{-1}. \tag{22}$$

Again applying (21), but now to matrix **E**, results in

$$\mathbf{E}^{-1} = (\mathbf{L} + \frac{1}{n}\mathbf{1}\mathbf{1}')^{-1} = \mathbf{L}^{-1} - \mathbf{L}^{-1}(\frac{1}{n}\mathbf{1})(\mathbf{I} + \mathbf{1}'\mathbf{L}^{-1}(\frac{1}{n}\mathbf{1}))^{-1}\mathbf{1}'\mathbf{L}^{-1}$$

$$= \mathbf{L}^{-1} - \frac{\mathbf{L}^{-1}\mathbf{1}\mathbf{1}'\mathbf{L}^{-1}}{n + \mathbf{1}'\mathbf{L}^{-1}\mathbf{1}}. \qquad (23)$$

The (real) inverse of diagonal matrix **L** is simply calculated as $1/l_{ii}$ for $i = 1,\ldots, n$, where l_{ii} denotes diagonal element i of **L**.

All this means that, if $m < n$, a much more efficient procedure is available for the calculation of the Moore-Penrose inverse of **V** than the one suggested by (18). We start by calculating $\mathbf{L}^{-1} = (\text{diag } \mathbf{HH}'\mathbf{11}')^{-1} = \{1/l_{ii}\}$, and compute $\mathbf{E}^{-1}$ using (23). We then compute $(\mathbf{E} - \mathbf{HH}')^{-1}$ according to (22); this only requires the calculation of the real inverse of matrix $(\mathbf{I} - \mathbf{H}'\mathbf{E}^{-1}\mathbf{H})$ of order (m x m). The desired Moore-Penrose inverse is finally obtained using (20). Clearly, compared to (18) the efficiency of this procedure increases as the ratio m/n becomes smaller, while the case where $m \geq n$ still requires the calculation of (18).

4 Minimizing the generalized loss function with respect to $\tilde{\mathbf{Z}}$

For the complete case, De Leeuw and Meulman (1985), and Meulman (1986, 1992) showed that the minimization of (1) with respect to $\tilde{\mathbf{Z}}$ involves a two-step procedure. The first step consists of the computation of a new *unrestricted* m-dimensional configuration, say $\bar{\mathbf{Z}}$, based on the current values of $\mathbf{X}$. In the complete case, this means that the roles of $\mathbf{X}$ and $\mathbf{Z}$ are interchanged. Therefore, a completely analogous procedure to the one described in section 3 can be used to determine $\bar{\mathbf{Z}}$ as another Guttman transform (Meulman called this the *reversed* Guttman transform). In the second step the unrestricted matrix $\bar{\mathbf{Z}}$ is projected onto the constrained solution space, that is, the variables in $\bar{\mathbf{Z}}$ are made to satisfy the constraints imposed by their respective measurement levels, yielding a new optimal $\tilde{\mathbf{Z}}$.

In the following sections we will show how to generalize this two-step procedure to the incomplete case, and discuss the problem how to minimize

$$f(\tilde{\mathbf{Z}}) \equiv 1/2 \sum_{i=1}^{n} \sum_{j=1}^{n} [d_{ij}(\tilde{\mathbf{Z}}\mathbf{M}_i\mathbf{M}_j) - w_{ij}^{1/2} d_{ij}(\mathbf{X})]^2, \tag{24}$$

where $\mathbf{X}$ is assumed to be fixed. Section 4.1 deals with the first step, that is, it shows how to determine a new unrestricted $\mathbf{Z}$, while section 4.2 discusses the problem of computing a new $\tilde{\mathbf{Z}}$ satisfying the constraints.

4.1 Determining a generalized Guttman transform for incomplete $\mathbf{Z}$

Compared to the complete case, and to the situation discussed in section 2, the problem of minimizing

$$f(\mathbf{Z}) \equiv 1/2 \sum_{i=1}^{n} \sum_{j=1}^{n} [d_{ij}(\mathbf{Z}\mathbf{M}_i\mathbf{M}_j) - w_{ij}^{1/2} d_{ij}(\mathbf{X})]^2 \tag{25}$$

over all unrestricted matrices $\mathbf{Z}$ of order (n x m) is complicated by the combined presence of the weights w_{ij} in (25) and of the matrix products $\mathbf{M}_i\mathbf{M}_j$ in definition (4) of the distances $d_{ij}(\mathbf{Z}\mathbf{M}_i\mathbf{M}_j)$. These complicating factors call for a generalization of the procedure discussed in section 3.

In fact, in this paper we will provide a solution to the more general problem of minimizing the following loss function

$$h(\mathbf{Z}) \equiv 1/2 \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij}[d_{ij}(\mathbf{Z}\mathbf{M}_i\mathbf{M}_j) - d_{ij}(\mathbf{X})]^2, \tag{26}$$

and then derive the solution for (25) as a special case. In (26), w_{ij} is defined as in (3), and $d_{ij}(\mathbf{Z}\mathbf{M}_i\mathbf{M}_j)$ as in (4). To solve (26) over all unrestricted (n x m) matrices $\mathbf{Z}$, we start by defining

$$\eta^2(\mathbf{Z}) = 1/2 \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} d_{ij}^2(\mathbf{Z}\mathbf{M}_i\mathbf{M}_j), \tag{27}$$

and

$$\rho(\mathbf{Z}) = 1/2 \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} d_{ij}(\mathbf{X}) d_{ij}(\mathbf{Z}\mathbf{M}_i\mathbf{M}_j), \tag{28}$$

which allows us to write (26) as

$$h(\mathbf{Z}) = \eta^2(\mathbf{X}) + \eta^2(\mathbf{Z}) - 2\,\rho(\mathbf{Z}), \tag{29}$$

where $\eta^2(\mathbf{X})$ is defined by (9).

We now proceed to derive some convenient matrix expressions for (27) and (28). These matrix expressions are presented in the form of theorems. Each theorem is preceded by a number of definitions, the function of which will become clear in the corresponding proofs.

Define $\mathbf{W}$ as the (n x n) symmetric matrix with weights w_{ij}, and $\mathbf{w}_i$ as the (n x 1) vector containing column (or row) i of $\mathbf{W}$. Let $\mathbf{W}_i$ = diag ($\mathbf{w}_i\mathbf{1}'$) of order (n x n), and

$$\mathbf{A}^* = \left(\sum_{i=1}^{n} (\mathbf{M}_i \otimes \mathbf{W}_i) \right) - (\mathbf{I}_m \otimes \mathbf{W}) \tag{30}$$

of order (mn x mn), where the symbol $\otimes$ denotes the Kronecker product. Also, let $\mathbf{z}^*$ = (vec $\mathbf{Z}$) denote the column vector of order (mn x 1) obtained by stacking the columns of $\mathbf{Z}$ one underneath the other. Finally, let n_k denote the number of non-missing elements in column k of $\mathbf{Z}$ (k = 1, ..., m), $\mathbf{z}$ denote the column vector of order (Σn_k x 1) only containing the non-missing elements of vector $\mathbf{z}^*$, and $\mathbf{A}$ denote the matrix of order (Σn_k x Σn_k) obtained by deleting the rows and columns of $\mathbf{A}^*$ which correspond to the missing entries in $\mathbf{z}^*$.

Theorem 1 $\eta^2(\mathbf{Z}) = \mathbf{z}'\mathbf{Az}$.

Proof If we collect the squared distances $d^2_{ij}(\mathbf{ZM}_i\mathbf{M}_j)$ in the matrix $\Delta^2(\mathbf{Z})$ of order $(n \times n)$, then

$$\begin{aligned}\eta^2(\mathbf{Z}) &= 1/2 \sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij} d^2_{ij}(\mathbf{ZM}_i\mathbf{M}_j) \\ &= 1/2 \sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij}(\mathbf{z}_i - \mathbf{z}_j)'\mathbf{M}_i\mathbf{M}_j(\mathbf{z}_i - \mathbf{z}_j) \\ &= 1/2 \operatorname{tr} \Delta^2(\mathbf{Z})\mathbf{W},\end{aligned}$$

where the matrix $\Delta^2(\mathbf{Z})$ may be written as

$$\Delta^2(\mathbf{Z}) =$$

$$\begin{bmatrix} \mathbf{z}_1'\mathbf{M}_1\mathbf{z}_1 & \mathbf{z}_1'\mathbf{M}_1\mathbf{M}_2\mathbf{z}_1 & \cdots & \mathbf{z}_1'\mathbf{M}_1\mathbf{M}_n\mathbf{z}_1 \\ \mathbf{z}_2'\mathbf{M}_2\mathbf{M}_1\mathbf{z}_2 & \mathbf{z}_2'\mathbf{M}_2\mathbf{z}_2 & \cdots & \mathbf{z}_2'\mathbf{M}_2\mathbf{M}_n\mathbf{z}_2 \\ \vdots & \vdots & & \vdots \\ \mathbf{z}_n'\mathbf{M}_n\mathbf{M}_1\mathbf{z}_n & \mathbf{z}_n'\mathbf{M}_n\mathbf{M}_2\mathbf{z}_n & \cdots & \mathbf{z}_n'\mathbf{M}_n\mathbf{z}_n \end{bmatrix} + \begin{bmatrix} \mathbf{z}_1'\mathbf{M}_1\mathbf{z}_1 & \mathbf{z}_2'\mathbf{M}_1\mathbf{M}_2\mathbf{z}_2 & \cdots & \mathbf{z}_n'\mathbf{M}_1\mathbf{M}_n\mathbf{z}_n \\ \mathbf{z}_1'\mathbf{M}_2\mathbf{M}_1\mathbf{z}_1 & \mathbf{z}_2'\mathbf{M}_2\mathbf{z}_2 & \cdots & \mathbf{z}_n'\mathbf{M}_2\mathbf{M}_n\mathbf{z}_n \\ \vdots & \vdots & & \vdots \\ \mathbf{z}_1'\mathbf{M}_n\mathbf{M}_1\mathbf{z}_1 & \mathbf{z}_2'\mathbf{M}_n\mathbf{M}_2\mathbf{z}_2 & \cdots & \mathbf{z}_n'\mathbf{M}_n\mathbf{z}_n \end{bmatrix}$$

$$-2\begin{bmatrix} \mathbf{z}_1'\mathbf{M}_1\mathbf{z}_1 & \mathbf{z}_1'\mathbf{M}_1\mathbf{M}_2\mathbf{z}_2 & \cdots & \mathbf{z}_1'\mathbf{M}_1\mathbf{M}_n\mathbf{z}_n \\ \mathbf{z}_2'\mathbf{M}_2\mathbf{M}_1\mathbf{z}_1 & \mathbf{z}_2'\mathbf{M}_2\mathbf{z}_2 & \cdots & \mathbf{z}_2'\mathbf{M}_2\mathbf{M}_n\mathbf{z}_n \\ \vdots & \vdots & & \vdots \\ \mathbf{z}_n'\mathbf{M}_n\mathbf{M}_1\mathbf{z}_1 & \mathbf{z}_n'\mathbf{M}_n\mathbf{M}_2\mathbf{z}_2 & \cdots & \mathbf{z}_n'\mathbf{M}_n\mathbf{z}_n \end{bmatrix}.$$

Without any loss of generality, we assume at this point that matrix $\mathbf{Z}$ contains *structural zeroes* wherever an entry in $\mathbf{Z}$ is missing. Since in that case $\mathbf{M}_i\mathbf{z}_i = \mathbf{z}_i$ for $i = 1, \ldots, n$, and letting

$$\mathbf{C} = \begin{bmatrix} \mathbf{z}_1'\mathbf{M}_1\mathbf{z}_1 & \mathbf{z}_1'\mathbf{M}_2\mathbf{z}_1 & \cdots & \mathbf{z}_1'\mathbf{M}_n\mathbf{z}_1 \\ \mathbf{z}_2'\mathbf{M}_1\mathbf{z}_2 & \mathbf{z}_2'\mathbf{M}_2\mathbf{z}_2 & \cdots & \mathbf{z}_2'\mathbf{M}_n\mathbf{z}_2 \\ \vdots & \vdots & & \vdots \\ \mathbf{z}_n'\mathbf{M}_1\mathbf{z}_n & \mathbf{z}_n'\mathbf{M}_2\mathbf{z}_n & \cdots & \mathbf{z}_n'\mathbf{M}_n\mathbf{z}_n \end{bmatrix},$$

this allows us to write $\Delta^2(\mathbf{Z}) = \mathbf{C} + \mathbf{C}' - 2\,\mathbf{ZZ}'$. Conditional on the presence of structural zeroes in the missing entries of $\mathbf{Z}$, therefore, it is true that

$$\eta^2(\mathbf{Z}) = 1/2 \text{ tr } \Delta^2(\mathbf{Z})\mathbf{W}$$

$$= 1/2 \text{ tr } (\mathbf{C} + \mathbf{C}' - 2\,\mathbf{ZZ}')\mathbf{W}$$

$$= \text{tr } \mathbf{C'W} - \text{tr } \mathbf{ZZ'W}$$

$$= \sum_{i=1}^{n} \text{tr } \mathbf{W}_i\mathbf{ZM}_i\mathbf{Z}' - \text{tr } \mathbf{ZZ'W}. \qquad (31)$$

Applying several well known properties of the vec operator and the Kronecker product to (31) yields

$$\eta^2(\mathbf{Z}) = \sum_{i=1}^{n} \text{tr } \mathbf{W}_i\mathbf{ZM}_i\mathbf{Z}' - \text{tr } \mathbf{ZZ'W}$$

$$= \sum_{i=1}^{n} (\text{vec } \mathbf{W}_i\mathbf{Z})'(\text{vec } \mathbf{ZM}_i) - (\text{vec } \mathbf{Z})'(\text{vec } \mathbf{WZ})$$

$$= \sum_{i=1}^{n} [(\mathbf{I}_m \otimes \mathbf{W}_i) \text{ vec } \mathbf{Z}]'[(\mathbf{M}_i \otimes \mathbf{I}_n) \text{ vec } \mathbf{Z}] - (\text{vec } \mathbf{Z})'(\mathbf{I}_m \otimes \mathbf{W}) (\text{vec } \mathbf{Z})$$

$$= (\text{vec } \mathbf{Z})'[(\sum_{i=1}^{n} (\mathbf{M}_i \otimes \mathbf{W}_i)) - (\mathbf{I}_m \otimes \mathbf{W})](\text{vec } \mathbf{Z})$$

$$= \mathbf{z}^{*\prime}\mathbf{A}^*\mathbf{z}^*. \qquad (32)$$

However, we could only derive expression (32) on the condition that $\mathbf{z}^*$ contains structural zeroes. Since these structural zeroes are sure to wipe out the corresponding rows and columns of $\mathbf{A}^*$ in the product $\mathbf{z}^{*\prime}\mathbf{A}^*\mathbf{z}^*$, the latter rows and columns are completely irrelevant for the value of (32). Therefore, we may as well drop all elements of $\mathbf{z}^*$ containing structural zeroes, and delete the corresponding rows and columns of matrix $\mathbf{A}^*$. The theorem immediately follows. □

Note how the structural zeroes are only used as a temporary aid, to be thrown away in the final derivations: in the expression stated in the theorem, the missing entries have completely disappeared.

Matrices $\mathbf{A}^*$ and $\mathbf{A}$ have some important properties, that we will now discuss. First of all, they are both block-diagonal, where $\mathbf{A}^*$ contains m blocks of order (n x n) on the diagonal, and $\mathbf{A}$ m blocks of order (n_k x n_k). As a very simple example, suppose that $\mathbf{Z}$ contains information on only two variables for three objects, and that information about variable 1 is missing for object 2. In this special case matrix $\mathbf{A}^*$ has the following form:

$$\mathbf{A}^* = \begin{bmatrix} w_{13} & -w_{12} & -w_{13} & 0 & 0 & 0 \\ -w_{12} & w_{12}+w_{23}-w_{22} & -w_{23} & 0 & 0 & 0 \\ -w_{13} & -w_{23} & w_{13} & 0 & 0 & 0 \\ 0 & 0 & 0 & w_{12}+w_{13} & -w_{12} & -w_{13} \\ 0 & 0 & 0 & -w_{12} & w_{12}+w_{23} & -w_{23} \\ 0 & 0 & 0 & -w_{13} & -w_{23} & w_{13}+w_{23} \end{bmatrix}.$$

Since element z_{21} of matrix **Z** is missing, the second element of vector **z*** is a structural zero, and the value of $\eta^2(\mathbf{Z}) = \mathbf{z}^{*\prime}\mathbf{A}^*\mathbf{z}^*$ is unaffected if we use

$$\mathbf{A}^* = \begin{bmatrix} w_{13} & 0 & -w_{13} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ -w_{13} & 0 & w_{13} & 0 & 0 & 0 \\ 0 & 0 & 0 & w_{12}+w_{13} & -w_{12} & -w_{13} \\ 0 & 0 & 0 & -w_{12} & w_{12}+w_{23} & -w_{23} \\ 0 & 0 & 0 & -w_{13} & -w_{23} & w_{13}+w_{23} \end{bmatrix}.$$

Therefore, we may as well evaluate $\eta^2(\mathbf{Z})$ as $\mathbf{z}'\mathbf{A}\mathbf{z}$, where

$$\mathbf{A} = \begin{bmatrix} w_{13} & -w_{13} & 0 & 0 & 0 \\ -w_{13} & w_{13} & 0 & 0 & 0 \\ 0 & 0 & w_{12}+w_{13} & -w_{12} & -w_{13} \\ 0 & 0 & -w_{12} & w_{12}+w_{23} & -w_{23} \\ 0 & 0 & -w_{13} & -w_{23} & w_{13}+w_{23} \end{bmatrix}.$$

If the given data matrix **Z** is complete, the m blocks on the diagonal of **A*** and **A** are all equal to the (n x n) matrix **V** defined in (13) (cf. section 3). In that case it is also true that $\mathbf{V} = nm\mathbf{J}$, where $\mathbf{J} \equiv \mathbf{I} - \mathbf{1}\mathbf{1}'/n$, and the theorem therefore simplifies into $\eta^2(\mathbf{Z}) = nm \text{ tr } \mathbf{Z}'\mathbf{J}\mathbf{Z}$. Once **Z** is incomplete, however, the blocks of order (n x n) on the diagonal of **A*** are no longer equal. Neither are the blocks of order (n_k x n_k) in matrix **A**. More specifically, in that case there are as many different blocks as there are different patterns of missing entries in the columns of **Z**.

Although the m blocks of matrix **A** may differ, and may have varying orders, they share one important property. They are all symmetric matrices with negative off-diagonal and positive diagonal elements, whose rows and columns sum up to zero. Therefore, if we let $\mathbf{A}_k$ ($k = 1, \ldots, m$) denote the k-th block of order (n_k x n_k) on the diagonal of matrix **A**, each $\mathbf{A}_k$ is *positive semi-definite* with $\text{rank}(\mathbf{A}_k) = n_k - 1$. This means that the complete matrix **A** is also positive semi-definite, that is, $\mathbf{x}'\mathbf{A}\mathbf{x} \geq 0$ for any vector **x** of order ($\sum n_k$ x 1). The properties of **A** also imply that, if $\mathbf{A}^+$ is the Moore-Penrose inverse of **A**, then $\mathbf{A}^+\mathbf{A}\mathbf{z} = \mathbf{A}\mathbf{A}^+\mathbf{z}$ for all centered **z**. The latter properties do not hold for matrix **A***, which is *indefinite* (i.e., it has negative as well as positive eigenvalues) once **Z** is incomplete.

We now proceed to rewrite (28). First note that

$$\begin{aligned}\rho(\mathbf{Z}) &= 1/2 \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} d_{ij}(\mathbf{X}) d_{ij}(\mathbf{Z}\mathbf{M}_i \mathbf{M}_j) \\ &= 1/2 \sum_{i=1}^{n} \sum_{j=1}^{n} \frac{w_{ij} d_{ij}(\mathbf{X})}{d_{ij}(\mathbf{Z}\mathbf{M}_i \mathbf{M}_j)} d_{ij}^2(\mathbf{Z}\mathbf{M}_i \mathbf{M}_j) \\ &= 1/2 \sum_{i=1}^{n} \sum_{j=1}^{n} b_{ij}(\mathbf{Z}) d_{ij}^2(\mathbf{Z}\mathbf{M}_i \mathbf{M}_j), \end{aligned} \tag{33}$$

where

$$b_{ij}(\mathbf{Z}) \equiv \begin{cases} \dfrac{w_{ij} d_{ij}(\mathbf{X})}{d_{ij}(\mathbf{Z}\mathbf{M}_i \mathbf{M}_j)} & \text{if } i \neq j \\ 0 & \text{if } d_{ij}(\mathbf{Z}\mathbf{M}_i \mathbf{M}_j) = 0 \end{cases} . \tag{34}$$

Expression (33) for $\rho(\mathbf{Z})$ is very similar to expression (27) for $\eta^2(\mathbf{Z})$, the only difference being that the terms w_{ij} in (27) have been replaced by the terms $b_{ij}(\mathbf{Z})$ in (33). Therefore, let $\mathbf{B}^0(\mathbf{Z})$ denote the (n x n) symmetric matrix containing elements $b_{ij}(\mathbf{Z})$, and let $\mathbf{B}_i^0(\mathbf{Z}) = \text{diag}(\mathbf{b}_i^0(\mathbf{Z})\mathbf{1}')$, where $\mathbf{b}_i^0(\mathbf{Z})$ denotes the (n x 1) vector containing column (or row) i of matrix $\mathbf{B}^0(\mathbf{Z})$. Finally, define

$$\mathbf{B}^*(\mathbf{Z}) = [\sum_{i=1}^{n} (\mathbf{M}_i \otimes \mathbf{B}_i^0(\mathbf{Z}))] - [\mathbf{I}_m \otimes \mathbf{B}^0(\mathbf{Z})] \tag{35}$$

of order (mn x mn), and let $\mathbf{B}(\mathbf{Z})$ denote the matrix of order ($\sum n_k$ x $\sum n_k$) obtained by deleting the rows and columns of $\mathbf{B}^*(\mathbf{Z})$ which correspond to the missing entries in **z***.

Theorem 2 $\rho(\mathbf{Z}) = \mathbf{z}'\mathbf{B}(\mathbf{Z})\mathbf{z}$.

Proof If the missing entries in **Z** contain structural zeroes (24) may be written as

$$\begin{aligned}\rho(\mathbf{Z}) &= 1/2 \sum_{i=1}^{n} \sum_{j=1}^{n} b_{ij}(\mathbf{Z}) d_{ij}^2(\mathbf{Z}\mathbf{M}_i\mathbf{M}_j) \\ &= 1/2 \text{ tr } \Delta^2(\mathbf{Z})\mathbf{B}^0(\mathbf{Z}) \\ &= \sum_{i=1}^{n} \text{tr } \mathbf{B}_i^0(\mathbf{Z})\mathbf{Z}\mathbf{M}_i\mathbf{Z}' - \text{tr } \mathbf{Z}\mathbf{Z}'\mathbf{B}^0(\mathbf{Z}). \end{aligned} \tag{36}$$

Applying properties of the vec operator and the Kronecker product to (36) results in

$$\begin{aligned}\rho(\mathbf{Z}) &= \sum_{i=1}^{n} \text{tr } \mathbf{B}_i^0(\mathbf{Z})\mathbf{Z}\mathbf{M}_i\mathbf{Z}' - \text{tr } \mathbf{Z}\mathbf{Z}'\mathbf{B}^0(\mathbf{Z}) \\ &= (\text{vec } \mathbf{Z})'[\sum_{i=1}^{n} (\mathbf{M}_i \otimes \mathbf{B}_i^0(\mathbf{Z}))](\text{vec } \mathbf{Z}) - (\text{vec } \mathbf{Z})'(\mathbf{I}_m \otimes \mathbf{B}^0(\mathbf{Z}))(\text{vec } \mathbf{Z}) \\ &= (\text{vec } \mathbf{Z})'[(\sum_{i=1}^{n} (\mathbf{M}_i \otimes \mathbf{B}_i^0(\mathbf{Z}))) - (\mathbf{I}_m \otimes \mathbf{B}^0(\mathbf{Z}))](\text{vec } \mathbf{Z}) \\ &= \mathbf{z}^{*\prime}\mathbf{B}^*(\mathbf{Z})\mathbf{z}^*. \end{aligned} \tag{37}$$

Finally, because $\mathbf{z}^*$ contains structural zeroes nothing changes if we write (37) as in Theorem 2, where $\mathbf{B}(\mathbf{Z})$ is a block-diagonal matrix with m blocks of order $(n_k \times n_k)$ on its diagonal. □

It follows from (29), (12), and Theorems 1 and 2 that (26) may be written as

$$h(\mathbf{z}) = \text{tr } \mathbf{X}'\mathbf{V}\mathbf{X} + \mathbf{z}'\mathbf{A}\mathbf{z} - 2\, \mathbf{z}'\mathbf{B}(\mathbf{Z})\mathbf{z}. \tag{38}$$

With these matrix expressions, it becomes possible to develop a procedure for the minimization of (38) over the $(\Sigma n_k \times 1)$ vectors **z**. Before we can postulate the next theorem, we need the following definitions. Let **Y** be an arbitrary matrix of order $(n \times m)$ *containing the same missing entries as* **Z**, let $\mathbf{y}^* = (\text{vec } \mathbf{Y})$, and define

$$b_{ij}(\mathbf{Y}) = \begin{cases} \dfrac{w_{ij} d_{ij}(\mathbf{X})}{d_{ij}(\mathbf{Y}\mathbf{M}_i\mathbf{M}_j)} & \text{if } i \neq j \\ 0 & \text{if } d_{ij}(\mathbf{Y}\mathbf{M}_i\mathbf{M}_j) = 0 \end{cases}, \tag{39}$$

where $d_{ij}(\mathbf{YM}_i\mathbf{M}_j)$ is defined as in (4). Let the elements $b_{ij}(\mathbf{Y})$ be collected in the symmetric matrix $\mathbf{B}^0(\mathbf{Y})$ of order (n x n), and let $\mathbf{B}_i^0(\mathbf{Y}) = \text{diag}\,(\mathbf{b}_i^0(\mathbf{Y})\mathbf{1}')$, where $\mathbf{b}_i^0(\mathbf{Y})$ denotes the (n x 1) vector containing column (or row) i of matrix $\mathbf{B}^0(\mathbf{Y})$. Also, define

$$\mathbf{B}^*(\mathbf{Y}) = [\sum_{i=1}^{n} (\mathbf{M}_i \otimes \mathbf{B}_i^0(\mathbf{Y}))] - [\mathbf{I}_m \otimes \mathbf{B}^0(\mathbf{Y})] \tag{40}$$

of order (mn x mn), and let $\mathbf{B}(\mathbf{Y})$ denote the matrix of order ($\sum n_k$ x $\sum n_k$) obtained by deleting the rows and columns of $\mathbf{B}^*(\mathbf{Y})$ which correspond to the missing entries in $\mathbf{y}^*$. Finally, define

$$\mu(\mathbf{z}, \mathbf{y}) = \mathbf{z}'\mathbf{B}(\mathbf{Y})\mathbf{y}, \tag{41}$$

where $\mathbf{y}$ is the ($\sum n_k$ x 1) vector only containing the nonmissing entries of $\mathbf{y}^*$.

Theorem 3 $\rho(\mathbf{Z}) \geq \mu(\mathbf{z}, \mathbf{y})$ for all matrices $\mathbf{Z}$, $\mathbf{Y}$ containing the same missing entries.

Proof The Cauchy-Schwartz inequality states that for any two vectors $\mathbf{a}$ and $\mathbf{b}$ it is true that

$$(\mathbf{a}'\mathbf{a})^{1/2}(\mathbf{b}'\mathbf{b})^{1/2} \geq \mathbf{a}'\mathbf{b}.$$

Substituting $\mathbf{a} = \mathbf{M}_i\mathbf{M}_j(\mathbf{z}_i - \mathbf{z}_j)$ and $\mathbf{b} = \mathbf{M}_i\mathbf{M}_j(\mathbf{y}_i - \mathbf{y}_j)$ we obtain, for any pair of configurations $\mathbf{Z}$ and $\mathbf{Y}$,

$$\{(\mathbf{z}_i - \mathbf{z}_j)'\mathbf{M}_i\mathbf{M}_j(\mathbf{z}_i - \mathbf{z}_j)\}^{1/2}\{(\mathbf{y}_i - \mathbf{y}_j)'\mathbf{M}_i\mathbf{M}_j(\mathbf{y}_i - \mathbf{y}_j)\}^{1/2} \geq (\mathbf{z}_i - \mathbf{z}_j)'\mathbf{M}_i\mathbf{M}_j(\mathbf{y}_i - \mathbf{y}_j)$$

due to the idempotency of matrices $\mathbf{M}_i$ and $\mathbf{M}_j$. According to definition (4) we therefore have

$$d_{ij}(\mathbf{ZM}_i\mathbf{M}_j)d_{ij}(\mathbf{YM}_i\mathbf{M}_j) \geq (\mathbf{z}_i - \mathbf{z}_j)'\mathbf{M}_i\mathbf{M}_j(\mathbf{y}_i - \mathbf{y}_j). \tag{42}$$

Multiplying (42) with the nonnegative term $w_{ij}d_{ij}(\mathbf{X})/d_{ij}(\mathbf{YM}_i\mathbf{M}_j)$, and summing over i and j yields

$$\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij}d_{ij}(\mathbf{X})d_{ij}(\mathbf{ZM}_i\mathbf{M}_j) \geq \sum_{i=1}^{n}\sum_{j=1}^{n} \frac{w_{ij}d_{ij}(\mathbf{X})}{d_{ij}(\mathbf{YM}_i\mathbf{M}_j)}(\mathbf{z}_i - \mathbf{z}_j)'\mathbf{M}_i\mathbf{M}_j(\mathbf{y}_i - \mathbf{y}_j),$$

which may again be written as

$$\sum_{i=1}^{n}\sum_{j=1}^{n}\frac{w_{ij}d_{ij}(\mathbf{X})}{d_{ij}(\mathbf{ZM}_i\mathbf{M}_j)}d_{ij}^2(\mathbf{ZM}_i\mathbf{M}_j) \geq \sum_{i=1}^{n}\sum_{j=1}^{n}\frac{w_{ij}d_{ij}(\mathbf{X})}{d_{ij}(\mathbf{YM}_i\mathbf{M}_j)}(\mathbf{z}_i-\mathbf{z}_j)'\mathbf{M}_i\mathbf{M}_j(\mathbf{y}_i-\mathbf{y}_j),$$

and thus as

$$\sum_{i=1}^{n}\sum_{j=1}^{n} b_{ij}(\mathbf{Z})d_{ij}^2(\mathbf{ZM}_i\mathbf{M}_j) \geq \sum_{i=1}^{n}\sum_{j=1}^{n} b_{ij}(\mathbf{Y})(\mathbf{z}_i-\mathbf{z}_j)'\mathbf{M}_i\mathbf{M}_j(\mathbf{y}_i-\mathbf{y}_j), \quad (43)$$

where $b_{ij}(\mathbf{Z})$ is defined as in (34) and $b_{ij}(\mathbf{Y})$ as in (39).

Collecting the elements $(\mathbf{z}_i-\mathbf{z}_j)'\mathbf{M}_i\mathbf{M}_j(\mathbf{y}_i-\mathbf{y}_j)$ in the $(n \times n)$ matrix $\mathbf{F}$, the right hand side of (43) can be expressed as

$$\sum_{i=1}^{n}\sum_{j=1}^{n} b_{ij}(\mathbf{Y})(\mathbf{z}_i-\mathbf{z}_j)'\mathbf{M}_i\mathbf{M}_j(\mathbf{y}_i-\mathbf{y}_j) = \text{tr}\,\mathbf{FB}^0(\mathbf{Y}). \quad (44)$$

However, the value of (44) remains unchanged if we write matrix $\mathbf{F}$ as

$$\mathbf{F} = \begin{bmatrix} \mathbf{z}_1'\mathbf{M}_1\mathbf{y}_1 & \mathbf{z}_1'\mathbf{M}_1\mathbf{M}_2\mathbf{y}_1 & \cdots & \mathbf{z}_1'\mathbf{M}_1\mathbf{M}_n\mathbf{y}_1 \\ \mathbf{z}_2'\mathbf{M}_2\mathbf{M}_1\mathbf{y}_2 & \mathbf{z}_2'\mathbf{M}_2\mathbf{y}_2 & \cdots & \mathbf{z}_2'\mathbf{M}_2\mathbf{M}_n\mathbf{y}_2 \\ \vdots & \vdots & & \vdots \\ \mathbf{z}_n'\mathbf{M}_n\mathbf{M}_1\mathbf{y}_n & \mathbf{z}_n'\mathbf{M}_n\mathbf{M}_2\mathbf{y}_n & \cdots & \mathbf{z}_n'\mathbf{M}_n\mathbf{y}_n \end{bmatrix} + \begin{bmatrix} \mathbf{z}_1'\mathbf{M}_1\mathbf{y}_1 & \mathbf{z}_2'\mathbf{M}_1\mathbf{M}_2\mathbf{y}_2 & \cdots & \mathbf{z}_n'\mathbf{M}_1\mathbf{M}_n\mathbf{y}_n \\ \mathbf{z}_1'\mathbf{M}_2\mathbf{M}_1\mathbf{y}_1 & \mathbf{z}_2'\mathbf{M}_2\mathbf{y}_2 & \cdots & \mathbf{z}_n'\mathbf{M}_2\mathbf{M}_n\mathbf{y}_n \\ \vdots & \vdots & & \vdots \\ \mathbf{z}_1'\mathbf{M}_n\mathbf{M}_1\mathbf{y}_1 & \mathbf{z}_2'\mathbf{M}_n\mathbf{M}_2\mathbf{y}_2 & \cdots & \mathbf{z}_n'\mathbf{M}_n\mathbf{y}_n \end{bmatrix}$$

$$-2\begin{bmatrix} \mathbf{z}_1'\mathbf{M}_1\mathbf{y}_1 & \mathbf{z}_1'\mathbf{M}_1\mathbf{M}_2\mathbf{y}_2 & \cdots & \mathbf{z}_1'\mathbf{M}_1\mathbf{M}_n\mathbf{y}_n \\ \mathbf{z}_2'\mathbf{M}_2\mathbf{M}_1\mathbf{y}_1 & \mathbf{z}_2'\mathbf{M}_2\mathbf{y}_2 & \cdots & \mathbf{z}_2'\mathbf{M}_2\mathbf{M}_n\mathbf{y}_n \\ \vdots & \vdots & & \vdots \\ \mathbf{z}_n'\mathbf{M}_n\mathbf{M}_1\mathbf{y}_1 & \mathbf{z}_n'\mathbf{M}_n\mathbf{M}_2\mathbf{y}_2 & \cdots & \mathbf{z}_n'\mathbf{M}_n\mathbf{y}_n \end{bmatrix}.$$

Therefore, on the condition that the identical missing entries of $\mathbf{Z}$ and $\mathbf{Y}$ all contain structural zeroes, and letting

$$\mathbf{G} = \begin{bmatrix} \mathbf{z}_1'\mathbf{M}_1\mathbf{y}_1 & \mathbf{z}_1'\mathbf{M}_2\mathbf{y}_1 & \cdots & \mathbf{z}_1'\mathbf{M}_n\mathbf{y}_1 \\ \mathbf{z}_2'\mathbf{M}_1\mathbf{y}_2 & \mathbf{z}_2'\mathbf{M}_2\mathbf{y}_2 & \cdots & \mathbf{z}_2'\mathbf{M}_n\mathbf{y}_2 \\ \vdots & \vdots & & \vdots \\ \mathbf{z}_n'\mathbf{M}_1\mathbf{y}_n & \mathbf{z}_n'\mathbf{M}_2\mathbf{y}_n & \cdots & \mathbf{z}_n'\mathbf{M}_n\mathbf{y}_n \end{bmatrix},$$

we may write

$$\sum_{i=1}^{n}\sum_{j=1}^{n} b_{ij}(\mathbf{Y})(\mathbf{z}_i - \mathbf{z}_j)'\mathbf{M}_i\mathbf{M}_j(\mathbf{y}_i - \mathbf{y}_j) = \text{tr } \mathbf{F}\mathbf{B}^0(\mathbf{Y})$$
$$= \text{tr } (\mathbf{G} + \mathbf{G}' - 2\,\mathbf{Z}\mathbf{Y}')\mathbf{B}^0(\mathbf{Y})$$
$$= 2 \text{ tr } \mathbf{G}'\mathbf{B}^0(\mathbf{Y}) - 2 \text{ tr } \mathbf{Z}\mathbf{Y}'\mathbf{B}^0(\mathbf{Y})$$
$$= 2 \sum_{i=1}^{n} \text{tr } \mathbf{B}_i^0(\mathbf{Y})\mathbf{Z}\mathbf{M}_i\mathbf{Y}' - 2 \text{ tr } \mathbf{Z}\mathbf{Y}'\mathbf{B}^0(\mathbf{Y}). \tag{45}$$

Applying properties of the vec operator and the Kronecker product to (45) results in

$$\sum_{i=1}^{n}\sum_{j=1}^{n} b_{ij}(\mathbf{Y})(\mathbf{z}_i - \mathbf{z}_j)'\mathbf{M}_i\mathbf{M}_j(\mathbf{y}_i - \mathbf{y}_j)$$
$$= 2 \sum_{i=1}^{n} \text{tr } \mathbf{B}_i^0(\mathbf{Y})\mathbf{Z}\mathbf{M}_i\mathbf{Y}' - 2 \text{ tr } \mathbf{Z}\mathbf{Y}'\mathbf{B}^0(\mathbf{Y})$$
$$= 2\,(\text{vec } \mathbf{Z})'[\sum_{i=1}^{n}(\mathbf{M}_i \otimes \mathbf{B}_i^0(\mathbf{Y}))](\text{vec } \mathbf{Y}) - 2\,(\text{vec } \mathbf{Z})'(\mathbf{I}_m \otimes \mathbf{B}^0(\mathbf{Y}))(\text{vec } \mathbf{Y})$$
$$= 2\,(\text{vec } \mathbf{Z})'[(\sum_{i=1}^{n}(\mathbf{M}_i \otimes \mathbf{B}_i^0(\mathbf{Y}))) - (\mathbf{I}_m \otimes \mathbf{B}^0(\mathbf{Y}))](\text{vec } \mathbf{Y})$$
$$= 2\,\mathbf{z}^{*\prime}\mathbf{B}^*(\mathbf{Y})\mathbf{y}^*. \tag{46}$$

Finally, because $\mathbf{z}^*$ and $\mathbf{y}^*$ both contain structural zeroes at identical places, nothing changes if we write (46) as

$$\sum_{i=1}^{n}\sum_{j=1}^{n} b_{ij}(\mathbf{Y})(\mathbf{z}_i - \mathbf{z}_j)'\mathbf{M}_i\mathbf{M}_j(\mathbf{y}_i - \mathbf{y}_j) = 2\,\mathbf{z}'\mathbf{B}(\mathbf{Y})\mathbf{y}. \tag{47}$$

Combining (33), Theorem 2, (41), (43), and (47) we obtain Theorem 3. □

Since the latter theorem states that, for any pair of matrices $\mathbf{Z}$ and $\mathbf{Y}$ containing the same missing entries, $2\,\mathbf{z}'\mathbf{B}(\mathbf{Z})\mathbf{z} \geq 2\,\mathbf{z}'\mathbf{B}(\mathbf{Y})\mathbf{y}$, it is also true that

$$\text{tr } \mathbf{X}'\mathbf{V}\mathbf{X} + \mathbf{z}'\mathbf{A}\mathbf{z} - 2\,\mathbf{z}'\mathbf{B}(\mathbf{Z})\mathbf{z} \leq \text{tr } \mathbf{X}'\mathbf{V}\mathbf{X} + \mathbf{z}'\mathbf{A}\mathbf{z} - 2\,\mathbf{z}'\mathbf{B}(\mathbf{Y})\mathbf{y}. \tag{48}$$

If we define

$$g(\mathbf{z}, \mathbf{y}) = \text{tr } \mathbf{X}'\mathbf{V}\mathbf{X} + \mathbf{z}'\mathbf{A}\mathbf{z} - 2\,\mathbf{z}'\mathbf{B}(\mathbf{Y})\mathbf{y}, \tag{49}$$

it follows from (38), (48) and (49) that

$$h(\mathbf{z}) \leq g(\mathbf{z}, \mathbf{y}). \tag{50}$$

Since, moreover,

$$h(\mathbf{z}) = g(\mathbf{z}, \mathbf{y}) \quad \text{if } \mathbf{z} = \mathbf{y}, \tag{51}$$

it is said that the function $g(\mathbf{z}, \mathbf{y})$ *majorizes* $h(\mathbf{z})$. In words (50) and (51) express that, at an arbitrary point $\mathbf{z}$, the quadratic function $g(\mathbf{z}, \mathbf{y})$ is always above $h(\mathbf{z})$, except for the point $\mathbf{z} = \mathbf{y}$, where the two functions meet. The latter point is called the *supporting point.*

These two properties of $h(\mathbf{z})$ and $g(\mathbf{z}, \mathbf{y})$ are illustrated in Figure 1. The upper part of the quadratic function $g(\mathbf{z}, \mathbf{y})$ has been 'chopped off' in the figure in order to clarify the bowl-like shape of this function; in reality the bowl extends upwards into infinity. Note that Figure 1 only displays a three-dimensional cross section of the true situation. In reality, the function $g(\mathbf{z}, \mathbf{y})$ has the shape of a $(\sum n_k + 1)$-dimensional 'hyperbowl', while function $h(\mathbf{z})$ is a $(\sum n_k + 1)$-dimensional, irregularly shaped, mountainous 'hyperlandscape'. Even then, however, the hyperbowl defined by $g(\mathbf{z}, \mathbf{y})$ is floating everywhere above the hyperlandscape defined by $h(\mathbf{z})$, except for the supporting point $\mathbf{z} = \mathbf{y}$ where the two surfaces touch.

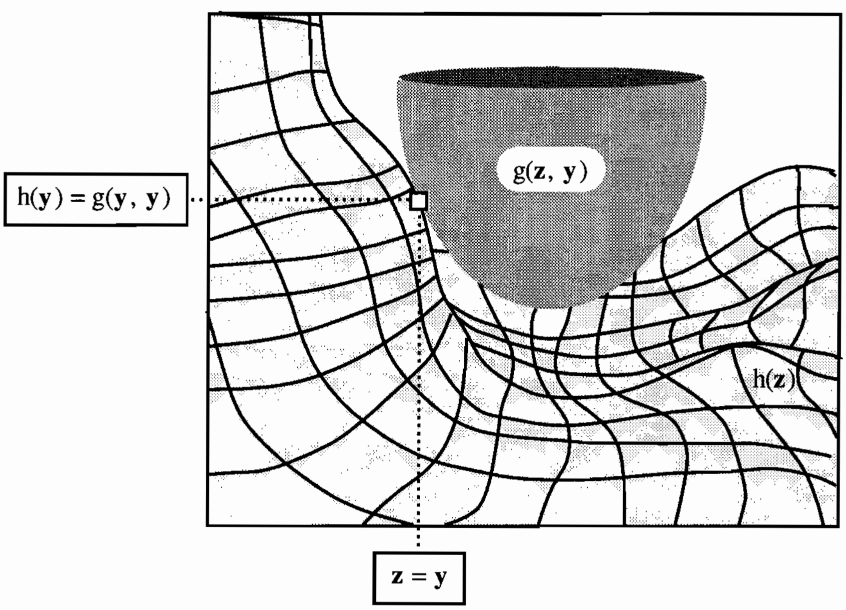

Figure 1 Illustration of $h(\mathbf{z})$ and the majorizing function $g(\mathbf{z}, \mathbf{y})$.

At this point we introduce the *generalized Guttman transform*

$$\bar{\mathbf{z}} = \mathbf{A}^{+}\mathbf{B}(\mathbf{Y})\mathbf{y}, \tag{52}$$

where $\mathbf{A}^{+}$ is the Moore-Penrose inverse of **A**.

Theorem 4 $h(\bar{\mathbf{z}}) \leq g(\bar{\mathbf{z}}, \mathbf{y}) \leq h(\mathbf{y})$.

Proof Substitution of (52) in (49) yields

$$\begin{aligned} g(\mathbf{z}, \mathbf{y}) &= \text{tr}\, \mathbf{X}'\mathbf{V}\mathbf{X} + \mathbf{z}'\mathbf{A}\mathbf{z} - 2\,\mathbf{z}'\mathbf{B}(\mathbf{Y})\mathbf{y} \\ &= \text{tr}\, \mathbf{X}'\mathbf{V}\mathbf{X} + \mathbf{z}'\mathbf{A}\mathbf{z} - 2\,\mathbf{z}'\mathbf{A}\mathbf{A}^{+}\mathbf{B}(\mathbf{Y})\mathbf{y}, \\ &= \text{tr}\, \mathbf{X}'\mathbf{V}\mathbf{X} - \bar{\mathbf{z}}'\mathbf{A}\bar{\mathbf{z}} + \mathbf{z}'\mathbf{A}\mathbf{z} + \bar{\mathbf{z}}'\mathbf{A}\bar{\mathbf{z}} - 2\,\mathbf{z}'\mathbf{A}\bar{\mathbf{z}} \\ &= \text{tr}\, \mathbf{X}'\mathbf{V}\mathbf{X} - \bar{\mathbf{z}}'\mathbf{A}\bar{\mathbf{z}} + (\mathbf{z} - \bar{\mathbf{z}})'\mathbf{A}(\mathbf{z} - \bar{\mathbf{z}}), \end{aligned}$$

meaning that, for fixed **y**, the global minimum of (49) with respect to **z** is attained for

$$\mathbf{z} = \bar{\mathbf{z}} = \mathbf{A}^{+}\mathbf{B}(\mathbf{Y})\mathbf{y}. \tag{53}$$

In words (53) expresses that, for fixed **y**, the 'bottom' of the hyperbowl is located straight above the point $\mathbf{z} = \bar{\mathbf{z}}$ (i.e., the generalized Guttman transform). A three-dimensional cross section of this situation is illustrated in Figure 2.

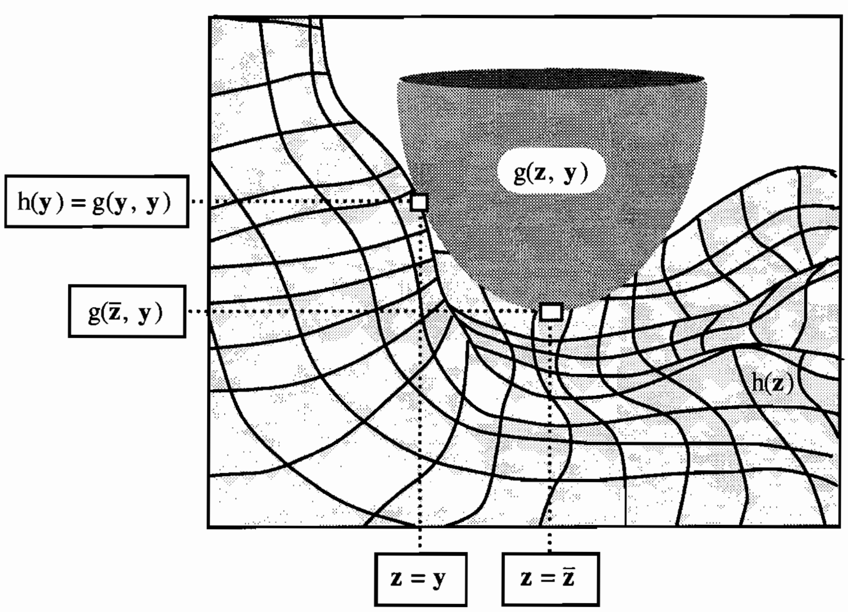

Figure 2 Illustration of global minimum of $g(\mathbf{z}, \mathbf{y})$ for $\mathbf{z} = \bar{\mathbf{z}}$.

Since h($\mathbf{z}$) ≤ g($\mathbf{z}$, $\mathbf{y}$) according to (50), this is also true for the point $\mathbf{z} = \bar{\mathbf{z}}$, or

$$h(\bar{\mathbf{z}}) \leq g(\bar{\mathbf{z}}, \mathbf{y}). \tag{54}$$

The latter inequality shows that the point on the hyperlandscape straight under the bottom of the hyperbowl will always be situated at the same height as, or lower than the bottom of the bowl.

It follows from (49) and (52) that

$$\begin{aligned} g(\bar{\mathbf{z}}, \mathbf{y}) &= \text{tr}\, \mathbf{X'VX} + \bar{\mathbf{z}}'\mathbf{A}\,\bar{\mathbf{z}} - 2\,\bar{\mathbf{z}}'\mathbf{B}(\mathbf{Y})\mathbf{y} \\ &= \text{tr}\, \mathbf{X'VX} + \bar{\mathbf{z}}'\mathbf{A}\,\bar{\mathbf{z}} - 2\,\bar{\mathbf{z}}'\mathbf{AA}^{+}\mathbf{B}(\mathbf{Y})\mathbf{y} \\ &= \text{tr}\, \mathbf{X'VX} + \bar{\mathbf{z}}'\mathbf{A}\,\bar{\mathbf{z}} - 2\,\bar{\mathbf{z}}'\mathbf{A}\,\bar{\mathbf{z}} \\ &= \text{tr}\, \mathbf{X'VX} - \bar{\mathbf{z}}'\mathbf{A}\,\bar{\mathbf{z}}. \end{aligned} \tag{55}$$

At the same time

$$\begin{aligned} h(\mathbf{y}) &= \text{tr}\, \mathbf{X'VX} + \mathbf{y}'\mathbf{A}\mathbf{y} - 2\,\mathbf{y}'\mathbf{B}(\mathbf{Y})\mathbf{y} \\ &= \text{tr}\, \mathbf{X'VX} + \mathbf{y}'\mathbf{A}\mathbf{y} - 2\,\mathbf{y}'\mathbf{AA}^{+}\mathbf{B}(\mathbf{Y})\mathbf{y} \\ &= \text{tr}\, \mathbf{X'VX} - \bar{\mathbf{z}}'\mathbf{A}\,\bar{\mathbf{z}} + \mathbf{y}'\mathbf{A}\mathbf{y} + \bar{\mathbf{z}}'\mathbf{A}\,\bar{\mathbf{z}} - 2\,\mathbf{y}'\mathbf{A}\bar{\mathbf{z}} \\ &= \text{tr}\, \mathbf{X'VX} - \bar{\mathbf{z}}'\mathbf{A}\,\bar{\mathbf{z}} + (\mathbf{y} - \bar{\mathbf{z}})'\mathbf{A}(\mathbf{y} - \bar{\mathbf{z}}). \end{aligned} \tag{56}$$

Since $(\mathbf{y} - \bar{\mathbf{z}})'\mathbf{A}(\mathbf{y} - \bar{\mathbf{z}}) \geq 0$, matrix $\mathbf{A}$ being positive semi-definite, it follows from (55) and (56) that

$$g(\bar{\mathbf{z}}, \mathbf{y}) \leq h(\mathbf{y}). \tag{57}$$

In words, (57) expresses that the bottom of the hyperbowl will also always be situated at the same height as, or lower than the supporting point where the two surfaces touch. Finally, combining (54) and (57) we get the chain in the theorem, which is also illustrated in Figure 3. □

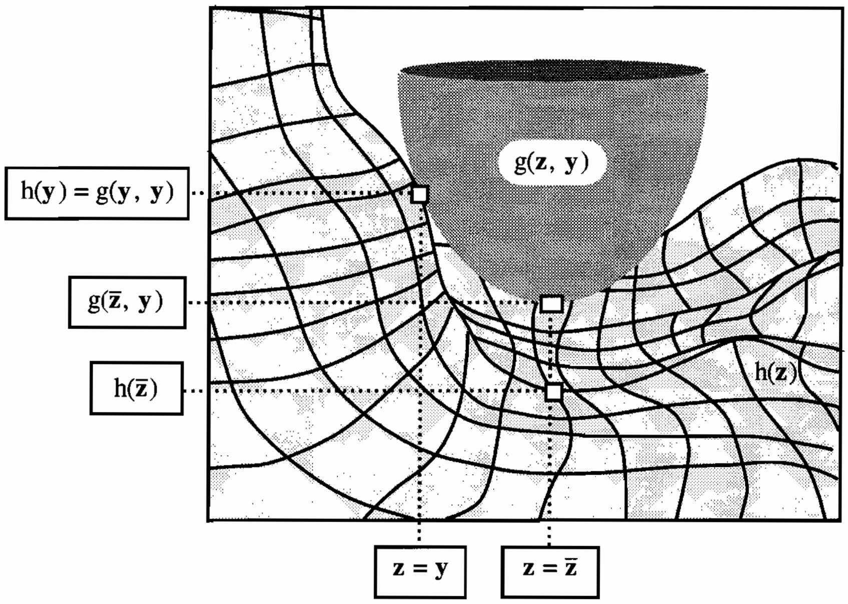

Figure 3 Illustration of the chain $h(\bar{\mathbf{z}}) \leq g(\bar{\mathbf{z}}, \mathbf{y}) \leq h(\mathbf{y})$.

Substituting $\mathbf{y} = \mathbf{z}^o$ in Theorem 4, we find that

$$h(\bar{\mathbf{z}}) \leq g(\bar{\mathbf{z}}, \mathbf{z}^o) \leq h(\mathbf{z}^o), \qquad (58)$$

meaning that an unrestricted update decreasing the stress in (26) is obtained by calculating

$$\bar{\mathbf{z}} = \mathbf{A}^+\mathbf{B}(\mathbf{Z}^o)\mathbf{z}^o, \qquad (59)$$

where $\mathbf{z}^o$ denotes the old optimal $\mathbf{z}$.

Because $\mathbf{A}$ and $\mathbf{B}(\mathbf{Z})$ are block-diagonal matrices with m blocks of order (n_k x n_k) on their diagonals, we never need to work with these full matrices, and we may update each column of $\mathbf{Z}$ separately. Thus, if we let $\mathbf{z}_k$ denote the (n_k x 1) vector containing the nonmissing elements of column k of $\mathbf{Z}$, and $\mathbf{A}_k^+$ and $\mathbf{B}_k(\mathbf{Z})$ denote the k-th block on the diagonal of $\mathbf{A}^+$ and $\mathbf{B}(\mathbf{Z})$, respectively, it is much more efficient to determine the generalized Guttman update as

$$\bar{\mathbf{z}}_k = \mathbf{A}_k^+\mathbf{B}_k(\mathbf{Z}^o)\mathbf{z}_k^o \quad \text{for } k = 1, \ldots, m, \qquad (60)$$

than using (59).

We are now in a position to derive the solution of (25), that is, of

$$f(\mathbf{Z}) = 1/2 \sum_{i=1}^{n} \sum_{j=1}^{n} [d_{ij}(\mathbf{ZM}_i\mathbf{M}_j) - w_{ij}{}^{1/2} d_{ij}(\mathbf{X})]^2$$

as a special case. In this situation, formulas (27) and (28) change into

$$\eta^2(\mathbf{Z}) = 1/2 \sum_{i=1}^{n} \sum_{j=1}^{n} d_{ij}^2(\mathbf{Z}\mathbf{M}_i\mathbf{M}_j), \tag{61}$$

and

$$\rho(\mathbf{Z}) = 1/2 \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij}^{1/2} d_{ij}(\mathbf{X}) d_{ij}(\mathbf{Z}\mathbf{M}_i\mathbf{M}_j), \tag{62}$$

respectively, and (30) becomes

$$\mathbf{A}^* = [(\sum_{i=1}^{n} \mathbf{M}_i) \otimes \mathbf{I}_n] - (\mathbf{I}_m \otimes \mathbf{1}\mathbf{1}'), \tag{63}$$

where the vector $\mathbf{1}$ is of order (n x 1). Matrix $\mathbf{A}$ of order ($\sum n_k$ x $\sum n_k$) simplifies into

$$\mathbf{A} = \mathbf{I}_m \otimes n_k \mathbf{J}_{n_k}, \tag{64}$$

where $\mathbf{J}_{n_k} = \mathbf{I}_{n_k} - \frac{1}{n_k}\mathbf{1}\mathbf{1}'$, that is, the centering matrix of order (n_k x n_k). Formula (34) changes into

$$b_{ij}(\mathbf{Z}) \equiv \begin{cases} \dfrac{w_{ij}^{1/2} d_{ij}(\mathbf{X})}{d_{ij}(\mathbf{Z}\mathbf{M}_i\mathbf{M}_j)} & \text{if } i \neq j \\ 0 & \text{if } d_{ij}(\mathbf{Z}\mathbf{M}_i\mathbf{M}_j) = 0 \end{cases}, \tag{65}$$

and (39) should be adapted accordingly. As concerns the remaining algebra, all derivations go through without any changes. However, due to (64) we have the convenient result that the generalized Guttman transform defined in (60) now simplifies into

$$\bar{\mathbf{z}}_k = \mathbf{A}_k^+ \mathbf{B}_k(\mathbf{Z}^o)\mathbf{z}_k^o = (n_k \mathbf{J}_{n_k})^+ \mathbf{B}_k(\mathbf{Z}^o)\mathbf{z}_k^o$$

$$= \frac{1}{n_k} \mathbf{J}_{n_k} \mathbf{B}_k(\mathbf{Z}^o)\mathbf{z}_k^o = \frac{1}{n_k} \mathbf{B}_k(\mathbf{Z}^o)\mathbf{z}_k^o \quad \text{for } k = 1, \ldots, m, \tag{66}$$

since $\mathbf{B}_k(\mathbf{Z}^o)\mathbf{z}_k^o$ is centered on the origin.

4.2 The metric projection problem for incomplete $\mathbf{Z}$

Once we have determined an *unrestricted* update $\bar{\mathbf{Z}}$ according to (66), the latter matrix must be restricted to satisfy the constraints imposed by the (possibly different) measurement levels of the variables. De Leeuw and Heiser (1980) showed that this can be achieved by solving the metric projection problem defined as

$$P_{\Omega_k}(\bar{\mathbf{z}}_k) = \{ \tilde{\mathbf{z}}_k \in \Omega_k \mid (\tilde{\mathbf{z}}_k - \bar{\mathbf{z}}_k)'\mathbf{J}_{n_k}(\tilde{\mathbf{z}}_k - \bar{\mathbf{z}}_k) = \min_{\mathbf{z}_k^* \in \Omega_k} (\mathbf{z}_k^* - \bar{\mathbf{z}}_k)'\mathbf{J}_{n_k}(\mathbf{z}_k^* - \bar{\mathbf{z}}_k) \}$$
$$\text{for } k = 1, \ldots, m. \quad (67)$$

where we use our notation for the matrices and vectors involved. In words, (67) expresses that, of all $(n_k \times 1)$ vectors $\mathbf{z}_k^*$ satisfying the constraints, we have to determine that one vector $\tilde{\mathbf{z}}_k$ that is as close as possible to the optimal unrestricted $\bar{\mathbf{z}}_k$ discussed in section 4.1. In addition to the constraints which correspond to the measurement level of variable k, we will also restrict the solution space Ω_k in (67) to satisfy $\mathbf{1}'\tilde{\mathbf{z}}_k = 0$ and $\tilde{\mathbf{z}}_k'\tilde{\mathbf{z}}_k = n_k/n$ for $k = 1, \ldots, m$. The latter normalization is important, because it prevents variables containing lots of missing entries to get disproportionally large values compared to variables that are (almost) complete. That the calculation of the generalized Guttman transform combined with the solutions of the metric projection problems (67) still result in a convergent algorithm can be proved as follows. Letting $\tilde{\mathbf{z}}_k^o$ denote the current solution satisfying the constraints for $k = 1, \ldots, m$, then

$$\begin{aligned} f_k(\tilde{\mathbf{z}}_k^o) &= c + n_k\, \tilde{\mathbf{z}}_k^{o\prime}\mathbf{J}_{n_k}\tilde{\mathbf{z}}_k^o - 2\, \tilde{\mathbf{z}}_k^{o\prime}\mathbf{B}_k(\tilde{\mathbf{Z}}^o)\tilde{\mathbf{z}}_k^o \\ &= c - n_k\, \bar{\mathbf{z}}_k'\mathbf{J}_{n_k}\bar{\mathbf{z}}_k + n_k\, \tilde{\mathbf{z}}_k^{o\prime}\mathbf{J}_{n_k}\tilde{\mathbf{z}}_k^o + n_k\, \bar{\mathbf{z}}_k'\mathbf{J}_{n_k}\bar{\mathbf{z}}_k - 2\, n_k\, \tilde{\mathbf{z}}_k^{o\prime}[\tfrac{1}{n_k}\mathbf{J}_{n_k}\mathbf{B}_k(\tilde{\mathbf{Z}}^o)\tilde{\mathbf{z}}_k^o] \\ &= c - n_k\, \bar{\mathbf{z}}_k'\mathbf{J}_{n_k}\bar{\mathbf{z}}_k + n_k\, (\tilde{\mathbf{z}}_k^o - \bar{\mathbf{z}}_k)'\mathbf{J}_{n_k}(\tilde{\mathbf{z}}_k^o - \bar{\mathbf{z}}_k) \quad \text{for } k = 1, \ldots, m, \end{aligned} \quad (68)$$

where $c \equiv \operatorname{tr} \mathbf{X}'\mathbf{V}\mathbf{X}$ is independent of $\tilde{\mathbf{z}}_k^o$. Moreover, due to (50) we may write

$$\begin{aligned} f_k(\tilde{\mathbf{z}}_k) &\leq g_k(\tilde{\mathbf{z}}_k, \tilde{\mathbf{z}}_k^o) \\ &\leq c + n_k\, \tilde{\mathbf{z}}_k'\mathbf{J}_{n_k}\tilde{\mathbf{z}}_k - 2\, \tilde{\mathbf{z}}_k'\mathbf{B}_k(\tilde{\mathbf{Z}}^o)\tilde{\mathbf{z}}_k^o \\ &\leq c - n_k\, \bar{\mathbf{z}}_k'\mathbf{J}_{n_k}\bar{\mathbf{z}}_k + n_k\, \tilde{\mathbf{z}}_k'\mathbf{J}_{n_k}\tilde{\mathbf{z}}_k + n_k\, \bar{\mathbf{z}}_k'\mathbf{J}_{n_k}\bar{\mathbf{z}}_k - 2\, n_k\, \tilde{\mathbf{z}}_k'[\tfrac{1}{n_k}\mathbf{J}_{n_k}\mathbf{B}_k(\tilde{\mathbf{Z}}^o)\tilde{\mathbf{z}}_k^o] \\ &\leq c - n_k\, \bar{\mathbf{z}}_k'\mathbf{J}_{n_k}\bar{\mathbf{z}}_k + n_k\, (\tilde{\mathbf{z}}_k - \bar{\mathbf{z}}_k)'\mathbf{J}_{n_k}(\tilde{\mathbf{z}}_k - \bar{\mathbf{z}}_k) \quad \text{for } k = 1, \ldots, m. \end{aligned} \quad (69)$$

But since the solution of (67) satisfies $(\tilde{\mathbf{z}}_k - \bar{\mathbf{z}}_k)'\mathbf{J}_{n_k}(\tilde{\mathbf{z}}_k - \bar{\mathbf{z}}_k) \le (\tilde{\mathbf{z}}_k^{\mathrm{o}} - \bar{\mathbf{z}}_k)'\mathbf{J}_{n_k}(\tilde{\mathbf{z}}_k^{\mathrm{o}} - \bar{\mathbf{z}}_k)$ for $k = 1, \ldots, m$, it follows from (68) and (69) that

$$\mathrm{f}_k(\tilde{\mathbf{z}}_k) \le \mathrm{g}_k(\tilde{\mathbf{z}}_k, \tilde{\mathbf{z}}_k^{\mathrm{o}}) \le \mathrm{f}_k(\tilde{\mathbf{z}}_k^{\mathrm{o}}) \quad \text{for } k = 1, \ldots, m.$$

This completes the proof.

As yet, we have not specified how to solve the metric projection problems in (67). This will now be discussed for numerical, ordinal, and nominal variables, respectively.

If $\mathbf{z}_k$ is a *numerical* variable, we have to determine that vector $\tilde{\mathbf{z}}_k$ that is as close as possible to the unrestricted update $\bar{\mathbf{z}}_k$, subject to the following constraints. The vector $\tilde{\mathbf{z}}_k$ must be a linear transformation of $\mathbf{z}_k$, where $\mathbf{z}_k$ denotes the original variable, that is, it must satisfy

$$\tilde{\mathbf{z}}_k = \mathrm{b}\ \mathbf{z}_k + \mathrm{a}, \tag{70}$$

subject to $\mathbf{1}'\tilde{\mathbf{z}}_k = 0$ and $\tilde{\mathbf{z}}_k'\tilde{\mathbf{z}}_k = n_k/n$. But the latter two constraints immediately fix the parameters a and b in (70), since it follows from $\mathbf{1}'\tilde{\mathbf{z}}_k = \mathbf{1}'(\mathrm{b}\ \mathbf{z}_k + \mathrm{a}) = 0$ that $\mathrm{a} = 0$ if $\mathbf{1}'\mathbf{z}_k = 0$, and from $\tilde{\mathbf{z}}_k'\tilde{\mathbf{z}}_k = (\mathrm{b}\ \mathbf{z}_k + \mathrm{a})'(\mathrm{b}\ \mathbf{z}_k + \mathrm{a}) = \mathrm{b}^2\ \mathbf{z}_k'\mathbf{z}_k = n_k/n$ that $\mathrm{b} = \sqrt{n_k/n}$ if $\mathbf{z}_k'\mathbf{z}_k = n_k/n$. Therefore, the metric projection problem (67) for numerical variables is simply solved by setting $\tilde{\mathbf{z}}_k = \mathbf{z}_k$, that is, by doing absolutely nothing. This also means that no computation of an unrestricted update according to (66) is required for numerical variables.

If $\mathbf{z}_k$ is an *ordinal* variable, the metric projection problem consists of finding that vector $\tilde{\mathbf{z}}_k$ as close as possible to the unrestricted update $\bar{\mathbf{z}}_k$, subject to the constraints that $\tilde{\mathbf{z}}_k$ is a monotone transformation of the original variable $\mathbf{z}_k$, and that $\mathbf{1}'\tilde{\mathbf{z}}_k = 0$ and $\tilde{\mathbf{z}}_k'\tilde{\mathbf{z}}_k = n_k/n$. Therefore, in this case the solution space Ω_k in (67) is the intersection of an n_k-dimensional cone (cf., Gifi, 1990) and the subspace containing the vectors satisfying $\mathbf{1}'\tilde{\mathbf{z}}_k = 0$ and $\tilde{\mathbf{z}}_k'\tilde{\mathbf{z}}_k = n_k/n$. This problem is solved by performing a monotone regression of $\bar{\mathbf{z}}_k$ on the original variable $\mathbf{z}_k$, and then normalizing the result. When defining Ω_k in (67), we can choose for either the primary or the secondary approach to ties.

Summarizing, the determination of an update for an ordinal variable in $\mathbf{Z}$ consists of the following steps. Letting $\tilde{\mathbf{z}}_k^{\mathrm{o}}$ denote the current solution for ordinal variable k, we first calculate the generalized Guttman transform (cf. section 4.1), that is,

$$\bar{\mathbf{z}}_k = \frac{1}{n_k}\mathbf{B}_k(\tilde{\mathbf{Z}}^{\mathrm{o}})\tilde{\mathbf{z}}_k^{\mathrm{o}},$$

to determine an unrestricted update for this variable. Then, we perform a monotone regression of the vector $\bar{\mathbf{z}}_k$ on the original variable $\mathbf{z}_k$, and normalize the result, yielding the optimal vector $\tilde{\mathbf{z}}_k$.

If $\mathbf{z}_k$ is a *nominal* variable, the metric projection problem is to find that vector $\tilde{\mathbf{z}}_k$ as close as possible to the unrestricted update $\bar{\mathbf{z}}_k$, subject to the following constraints. Objects of $\tilde{\mathbf{z}}_k$ belonging to the same category in the original variable $\mathbf{z}_k$ are required to have identical numerical values in $\tilde{\mathbf{z}}_k$. We also require that $\mathbf{1}'\tilde{\mathbf{z}}_k = 0$ and $\tilde{\mathbf{z}}_k'\tilde{\mathbf{z}}_k = n_k/n$.

Let c_k denote the number of categories of the original nominal variable $\mathbf{z}_k$, and $\mathbf{G}_k$ denote the indicator matrix (cf. Gifi, 1990) of order (n_k x c_k) corresponding to nominal variable $\mathbf{z}_k$. Also define $\mathbf{y}_k$ as the vector of order (c_k x 1) which only contains the c_k distinct category quantifications corresponding to variable $\mathbf{z}_k$. Then writing $\tilde{\mathbf{z}}_k$ in terms of $\mathbf{G}_k\mathbf{y}_k$ guarantees that objects belonging to the same category of variable $\mathbf{z}_k$ will obtain identical quantifications. Formally, therefore, the metric projection problem for nominal variables consists of finding the minimum of

$$g(\mathbf{y}_k) = (\mathbf{G}_k\mathbf{y}_k - \bar{\mathbf{z}}_k)'\mathbf{J}_{n_k}(\mathbf{G}_k\mathbf{y}_k - \bar{\mathbf{z}}_k), \tag{71}$$

subject to the constraints $\mathbf{1}'\mathbf{G}_k\mathbf{y}_k = 0$ and $\mathbf{y}_k'\mathbf{G}_k'\mathbf{G}_k\mathbf{y}_k = n_k/n$. This problem is solved by letting

$$\mathbf{y}_k = (\mathbf{G}_k'\mathbf{G}_k)^{-1}\mathbf{G}_k'\bar{\mathbf{z}}_k, \tag{72}$$

and then calculating

$$\tilde{\mathbf{z}}_k = \mathbf{J}_{n_k}\mathbf{G}_k\mathbf{y}_k \sqrt{\frac{n_k}{n\ \mathbf{y}_k'\mathbf{G}_k'\mathbf{J}_{n_k}\mathbf{G}_k\mathbf{y}_k}}\ . \tag{73}$$

where $\mathbf{J}_{n_k}$ is the centering matrix of order (n_k x n_k).

The update of a nominal variable of $\mathbf{Z}$ therefore requires the following calculations. Letting $\tilde{\mathbf{z}}_k^{\mathrm{o}}$ denote the current solution for nominal variable k, we first determine an unrestricted update with (66), that is,

$$\bar{\mathbf{z}}_k = \frac{1}{n_k}\mathbf{B}_k(\tilde{\mathbf{Z}}^{\mathrm{o}})\tilde{\mathbf{z}}_k^{\mathrm{o}},$$

(cf. section 4.1), and then we calculate (72) and (73) yielding the new restricted update for nominal variable k.

5 The algorithm

Applying the results obtained in sections 3 and 4, we may set up the following algorithm for the minimization of loss function (6).

1. Initialization

Depending upon whether $m < n$ or not, use the appropriate procedure described in section 3 to determine the Moore-Penrose inverse of matrix $\mathbf{V}$ defined in (13). Center and normalize the nonmissing elements of each column of $\mathbf{Z}$ such that $\mathbf{1}'\mathbf{z}_k = 0$, and $\mathbf{z}_k'\mathbf{z}_k = n_k/n$. Determine a first estimate for matrix $\mathbf{X}$ of order $(n \times p)$ (cf. section 6). Set a = 0.

*2. **X**–estimation step*

If a = 0 go to step 2.1, if a = 1 go to step 2.2.

2.1. a = 0

Repeatedly compute (17), that is, the Guttman transform $\mathbf{X}^{\mathrm{u}} = \mathbf{V}^{+}\mathbf{B}(\mathbf{X}^{\mathrm{o}})\mathbf{X}^{\mathrm{o}}$, until the loss in (6) is not further reduced beyond some preset threshold value, say ε_1. If all variables are numerical, rotate the configuration $\mathbf{X}$ to principal axes and stop. If $\mathbf{Z}$ also contains ordinal and/or nominal variables, set a = 1, and go to step 3 of the algorithm.

2.2. a = 1

Keeping the current quantifications in $\mathbf{Z}$ fixed, determine a new low-dimensional configuration $\mathbf{X}$ as the Guttman transform $\mathbf{X}^{\mathrm{u}} = \mathbf{V}^{+}\mathbf{B}(\mathbf{X}^{\mathrm{o}})\mathbf{X}^{\mathrm{o}}$.

3. $\tilde{\mathbf{Z}}$–estimation step

Keeping the current configuration $\mathbf{X}$ fixed, determine new quantifications $\tilde{\mathbf{Z}}$. For numerical variables, nothing needs to be done. For each ordinal and nominal variable with more than two categories:

a) first determine an unrestricted update $\bar{\mathbf{z}}_k = \frac{1}{n_k}\mathbf{B}_k(\tilde{\mathbf{Z}}^{\mathrm{o}})\tilde{\mathbf{z}}_k^{\mathrm{o}}$;

b) then solve $\tilde{\mathbf{z}}_k \in \mathrm{P}_{\Omega_k}(\bar{\mathbf{z}}_k)$, that is, project $\bar{\mathbf{z}}_k$ upon the appropriate restricted solution space Ω_k, and center and normalize the result (cf., section 4.2).

This completes an iteration. Evaluate loss function (6), and check whether the loss in this iteration and the loss in the previous iteration is smaller than some preset convergence criterion, say ε_2. If so, rotate the configuration $\mathbf{X}$ to principal axes and stop; otherwise go to step 2.

Notice that we may use different convergence criteria for the initial, numerical solution (ε_1) and for the final solution (ε_2). This algorithm, which we have implemented in the FORTRAN computer program PRINCESS, is guaranteed to converge, although not necessarily to the global minimum of (6). Compared to the complete case (cf., De Leeuw and Meulman, 1975; Meulman, 1986, 1992), the analysis of an incomplete matrix $\mathbf{Z}$ requires the additional determination and storage of matrix $\mathbf{V}^+$ of order (n x n). Step 2 of the algorithm consumes more CPU time than in the complete case, where we can do without matrix $\mathbf{V}^+$ in the calculation of an update for $\mathbf{X}$.

6 Determining a first estimate for X

As yet, we have not specified how we intend to determine a starting configuration $\mathbf{X}$ in step 1 of the algorithm. In the complete case, $\mathbf{X} = \mathbf{P}_p\mathbf{\Phi}_p$ is used as a first start, where $\mathbf{P}_p$ and $\mathbf{\Phi}_p$ are the respective p principal singular vectors and values in the singular value decomposition $\mathbf{Z} = \mathbf{P\Phi Q}'$. Alternatively, the same start is obtained by letting $\mathbf{X} = \mathbf{ZQ}_p$, where $\mathbf{Q}_p$ is the matrix containing the first p eigenvectors of matrix $\mathbf{Q}$ in the eigenvalue-eigenvector decomposition $\mathbf{Z}'\mathbf{Z} = \mathbf{Q\Phi}^2\mathbf{Q}'$. In the complete case, therefore, the algorithm is started with the p-dimensional solution of classical PCA. As several authors (see, for instance, Gower, 1966; Greenacre and Underhill, 1982; De Leeuw and Meulman, 1985; Meulman, 1986) have remarked, this start also happens to be the solution of classical (metric) multidimensional scaling if $\mathbf{Z}$ is column centered. In classical scaling (also called Torgerson-Gower or Young-Householder scaling) a p-dimensional configuration $\mathbf{X}$ is derived from a symmetric (n x n) matrix $\mathbf{\Delta}$ containing dissimilarities between n objects by applying a spectral decomposition to the double-centered matrix $-1/2\ \mathbf{J\Delta}^2\mathbf{J}$, as follows

$$-1/2\ \mathbf{J\Delta}^2\mathbf{J} = \mathbf{P\Phi}^2\mathbf{P}' = (\mathbf{P\Phi})(\mathbf{P\Phi})',$$

where $\mathbf{J}$ denotes the centering matrix. If we collect the squared distances $\{d^2_{ij}(\mathbf{Z})\}$ between the rows of $\mathbf{Z}$ in $\mathbf{\Delta}$, and let $\mathbf{d} = (\text{diag}\ \mathbf{ZZ}')\mathbf{1}$, we have that

$$-1/2\ \mathbf{J\Delta}^2\mathbf{J} = -1/2\ \mathbf{J}\{\mathbf{d1}' + \mathbf{1d}' - 2\ \mathbf{ZZ}'\}\mathbf{J} = -1/2\ \{-2\ \mathbf{JZZ}'\mathbf{J}\} = \mathbf{ZZ}',$$

because $\mathbf{J1} = \mathbf{0}$, and $\mathbf{JZ} = \mathbf{Z}$ if $\mathbf{Z}$ is column-centered. Therefore, it makes no difference in the complete case whether we use classical PCA or classical scaling to arrive at a first estimate of $\mathbf{X}$: the solutions will be exactly the same.

If $\mathbf{Z}$ is incomplete, this equivalence between classical PCA and classical scaling breaks down. The classical PCA solution can no longer be applied since the singular value decomposition of an incomplete matrix is not defined. Gifi (1990) provides a solution to this problem, which has been implemented in the computer program PRINCALS. Therefore, we could use the numerical PRINCALS solution as a first step. We could also construct matrix $\mathbf{\Delta}^2$ with elements $\{d_{ij}^2(\tilde{\mathbf{Z}}\mathbf{M}_i\mathbf{M}_j)\}$ defined in (4) for incomplete $\mathbf{Z}$ and then apply the classical scaling method, but this procedure would be very expensive since it yields an eigenvalue problem of order n. In practice, the initial p-dimensional configuration $\mathbf{X}$ in PRINCESS is simply determined as the classical PCA solution of $\mathbf{Z}$ after having supplemented its missing entries with zeroes.

7 About fit

At convergence of the algorithm proposed in section 5, the value of loss function (6) can be considered as an indication of 'badness-of-fit' of the solution: the smaller this value, the better the solution. However, this 'raw' stress or loss is not only a function of how well the high-dimensional distances between the objects are represented in low-dimensional space, but it is also a function of the numbers of objects involved in the analysis. In general, for fixed p the raw stress will increase as the number of objects becomes larger, irrespective of the 'goodness' of the solution. We can correct for this effect by taking the mean of the residuals, as De Leeuw and Meulman (1985) did for the complete case, that is, by dividing the raw stress by $1/2\; n(n-1)$. This corrects for the numbers of distances involved in the analysis, and thus implicitly for the numbers of objects. However, in the incomplete case the situation is further complicated by the fact that we are faced with all sorts of patterns of missing data in $\mathbf{Z}$. Since different patterns result in different weights w_{ij}, at convergence part of the value of (6) will depend upon which cells are missing, irrespective of the goodness of the solution.

Obviously, this calls for a more objective measure of fit, which allows for comparisons between different PRINCESS solutions. First note that the scalar α defined as

$$\alpha = \frac{\sum_{i<j} w_{ij}^{1/2} d_{ij}(\mathbf{X}) d_{ij}(\tilde{\mathbf{Z}}\mathbf{M}_i\mathbf{M}_j)}{\sum_{i<j} w_{ij} d_{ij}^2(\mathbf{X})}, \tag{74}$$

is the global minimizer of the function

$$f(\alpha) \equiv 1/2 \sum_{i=1}^{n} \sum_{j=1}^{n} [d_{ij}(\tilde{\mathbf{Z}}\mathbf{M}_i\mathbf{M}_j) - \alpha\, w_{ij}^{1/2} d_{ij}(\mathbf{X})]^2$$

$$= \sum_{i<j}^{n} [d_{ij}(\tilde{\mathbf{Z}}\mathbf{M}_i\mathbf{M}_j) - \alpha\, w_{ij}^{1/2} d_{ij}(\mathbf{X})]^2. \tag{75}$$

Substituting (74) in (75) we have

$$f(\tilde{\mathbf{Z}}, \mathbf{X}) = \Sigma_{i<j} d_{ij}^2(\tilde{\mathbf{Z}}\mathbf{M}_i\mathbf{M}_j) - \frac{[\Sigma_{i<j} w_{ij}^{1/2} d_{ij}(\mathbf{X}) d_{ij}(\tilde{\mathbf{Z}}\mathbf{M}_i\mathbf{M}_j)]^2}{\Sigma_{i<j} w_{ij} d_{ij}^2(\mathbf{X})}, \tag{76}$$

and dividing (76) by $\Sigma_{i<j} d_{ij}^2(\tilde{\mathbf{Z}}\mathbf{M}_i\mathbf{M}_j)$ we obtain

$$\frac{f(\tilde{\mathbf{Z}}, \mathbf{X})}{\Sigma_{i<j} d_{ij}^2(\tilde{\mathbf{Z}}\mathbf{M}_i\mathbf{M}_j)} + \frac{[\Sigma_{i<j} w_{ij}^{1/2} d_{ij}(\mathbf{X}) d_{ij}(\tilde{\mathbf{Z}}\mathbf{M}_i\mathbf{M}_j)]^2}{[\Sigma_{i<j} w_{ij} d_{ij}^2(\mathbf{X})]\, [\Sigma_{i<j} d_{ij}^2(\tilde{\mathbf{Z}}\mathbf{M}_i\mathbf{M}_j)]} = 1, \tag{77}$$

where the term

$$\frac{f(\tilde{\mathbf{Z}}, \mathbf{X})}{\Sigma_{i<j} d_{ij}^2(\tilde{\mathbf{Z}}\mathbf{M}_i\mathbf{M}_j)} \tag{78}$$

is *normalized* STRESS, and

$$\phi^2 \equiv \frac{[\Sigma_{i<j} w_{ij}^{1/2} d_{ij}(\mathbf{X}) d_{ij}(\tilde{\mathbf{Z}}\mathbf{M}_i\mathbf{M}_j)]^2}{[\Sigma_{i<j} w_{ij} d_{ij}^2(\mathbf{X})]\, [\Sigma_{i<j} d_{ij}^2(\tilde{\mathbf{Z}}\mathbf{M}_i\mathbf{M}_j)]} \tag{79}$$

is the square of Tucker's coefficient of congruence ϕ between the elements in $\{w_{ij}^{1/2} d_{ij}(\mathbf{X})\}$ and in $\{d_{ij}(\tilde{\mathbf{Z}}\mathbf{M}_i\mathbf{M}_j)\}$. Moreover, combining Theorem 1, formulas (61) and (64), and the fact that the non-missing elements of the variables satisfy $\mathbf{1}'\tilde{\mathbf{z}}_k = 0$ and $\tilde{\mathbf{z}}_k'\tilde{\mathbf{z}}_k = n_k/n$ for $k = 1, \ldots, m$, we find that

$$\begin{aligned} \Sigma_{i<j} d_{ij}^2(\tilde{\mathbf{Z}}\mathbf{M}_i\mathbf{M}_j) &= \tilde{\mathbf{z}}'\mathbf{A}\,\tilde{\mathbf{z}} = \tilde{\mathbf{z}}'(\mathbf{I}_m \otimes n_k \mathbf{J}_{n_k})\tilde{\mathbf{z}} \\ &= \Sigma_k n_k \tilde{\mathbf{z}}_k' \mathbf{J}_{n_k} \tilde{\mathbf{z}}_k = \frac{1}{n} \Sigma_k n_k^2. \end{aligned} \tag{80}$$

This proves that the sum of the squared distances $d_{ij}^2(\tilde{\mathbf{Z}}\mathbf{M}_i\mathbf{M}_j)$ remains constant during iterations, even though the quantifications are subject to change. It also means that we may write (77) in the convenient form

$$\frac{n f(\tilde{\mathbf{Z}}, \mathbf{X})}{\Sigma_k n_k^2} + \frac{n[\Sigma_{i<j} w_{ij}^{1/2} d_{ij}(\mathbf{X}) d_{ij}(\tilde{\mathbf{Z}}\mathbf{M}_i\mathbf{M}_j)]^2}{\Sigma_k n_k^2 [\Sigma_{i<j} w_{ij} d_{ij}^2(\mathbf{X})]} = 1. \tag{81}$$

The squared coefficient of congruence of Tucker (last term on the left hand side of (81)) and its counterpart, normalized STRESS (first term on the left hand side of (81)), are not sensitive to differences in scale of the configurations, to differences in weights arising from different patterns of missing data, or to differences in the numbers of objects. Therefore, in practice we use normalized STRESS to monitor convergence of the algorithm discussed in section 5 as well as to compare the fit of different PRINCESS solutions.

Another important aspect of the fit of PRINCESS solutions is the following. In contrast with the complete case, the high-dimensional distances between the rows of an incomplete matrix, whether they are defined as in (2) or as in (4), *do not necessarily all satisfy the triangle inequality*, that is, they do not necessarily satisfy

$$d_{AC} \leq d_{AB} + d_{BC} \qquad \text{for all rows A, B, C.}$$

In other words, if **Z** is incomplete, the $(n \times n)$ matrix of distances (2) or (4) is usually *indefinite*. Compared to the complete case, this introduces an additional source of STRESS. The implication is that, all other things being equal, the analysis of an incomplete data matrix usually results in a larger STRESS value than the analysis of a complete data matrix. This is also illustrated in the examples to be discussed in the following section.

8 Illustrations

In this section two examples are presented of the analysis of an incomplete data matrix using PRINCESS. The solutions are compared with the results obtained with the nonlinear PCA program PRINCALS (cf., Gifi, 1990). Both examples are analyses of data taken from Hartigan (1975). The same data are used in Gifi (1990) to illustrate the performance of HOMALS. The objects consist of a number of bolts, nails, screws, and tacks which have been classified on six criteria. Table 1 contains the complete data matrix, and explains the meaning of the categories of the six variables. In all analyses, we treated the six variables as (single) nominal, and computed a two-dimensional solution.

First, we analysed the complete Hartigan data with PRINCESS as well as with PRINCALS. Plots of the object scores obtained in the PRINCESS and PRINCALS analyses are given in Figure 4, and in Figure 5 we have plotted the distances in high-dimensional space against those in two-dimensional space for the PRINCESS solution. We present these results so as later to be able to compare them to the solutions obtained for incomplete data. Moreover, the solutions for the complete Hartigan data are interesting in their own right.

Table 1 Complete Hartigan hardware data.

Variable	1	2	3	4	5	6
1. Tack	2	1	1	1	1	2
2. Nail1	2	1	1	1	4	2
3. Nail2	2	1	1	1	2	2
4. Nail3	2	1	1	1	2	2
5. Nail4	2	1	1	1	2	2
6. Nail5	2	1	1	1	2	2
7. Nail6	2	2	1	1	5	2
8. Nail7	2	2	1	1	3	2
9. Nail8	2	2	1	1	3	2
10. Screw1	1	3	2	1	5	2
11. Screw2	1	4	3	1	4	2
12. Screw3	1	5	3	1	4	2
13. Screw4	1	4	3	1	2	2
14. Screw5	1	5	3	1	2	2
15. Bolt1	1	4	3	2	4	2
16. Bolt2	1	3	3	2	1	2
17. Bolt3	1	5	3	2	1	2
18. Bolt4	1	5	3	2	1	2
19. Bolt5	1	5	3	2	1	2
20. Bolt6	1	5	3	2	1	2
21. Tack1	2	1	1	1	1	1
22. Tack2	2	1	1	1	1	1
23. Nailb	2	1	1	1	1	1
24. Screwb	1	3	3	1	1	1

Variables	Categories		
1. Thread	1 = yes	2 = no	
2. Head	1 = flat	2 = cup	3 = cone
	4 = round	5 = cylinder	
3. Head indentation	1 = none	2 = star	3 = slit
4. Bottom	1 = sharp	2 = flat	
5. Length	in half inches		
6. Brass	1 = yes	2 = no	

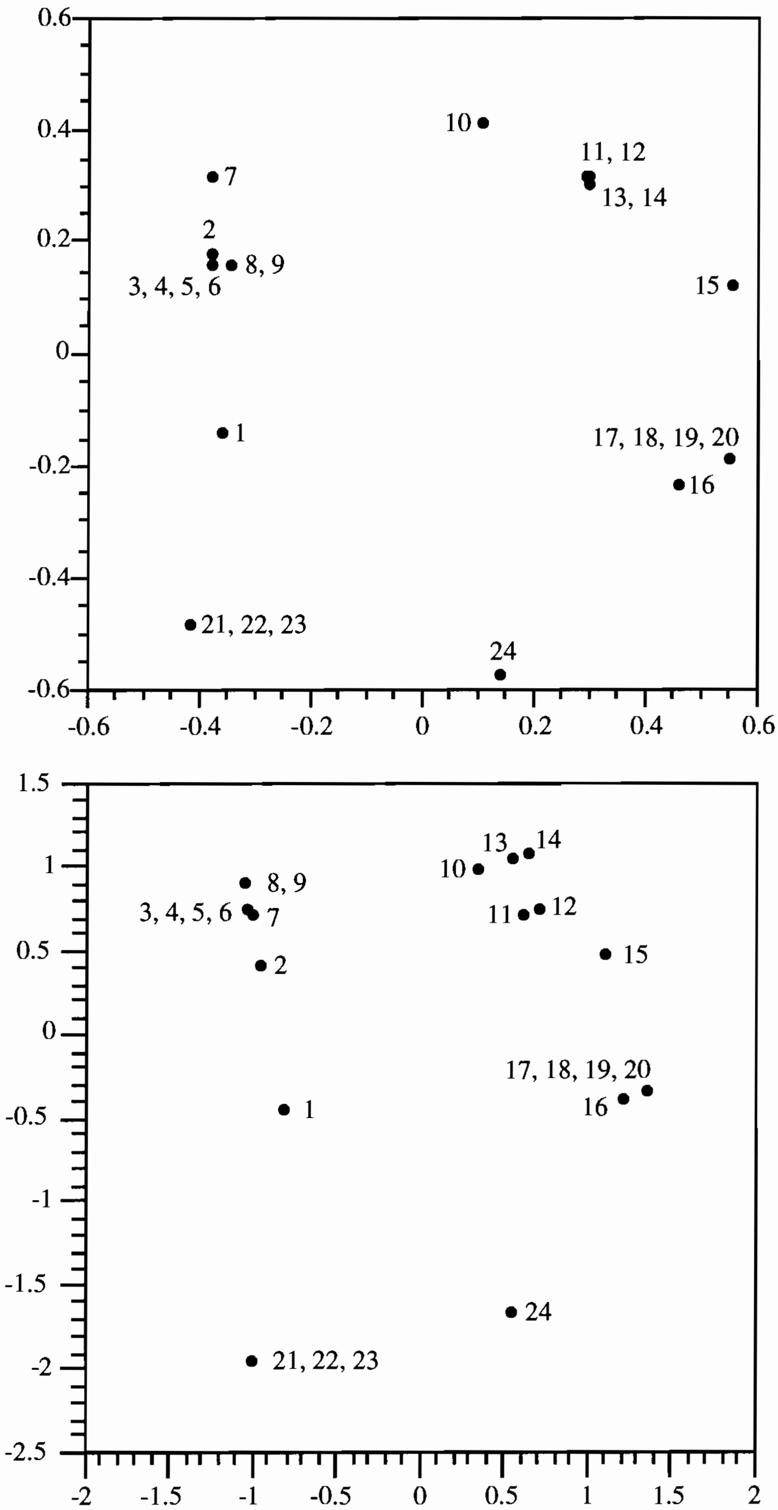

Figure 4 Plots of object scores obtained with PRINCESS (top) and PRINCALS (bottom) analysis of complete Hartigan data.

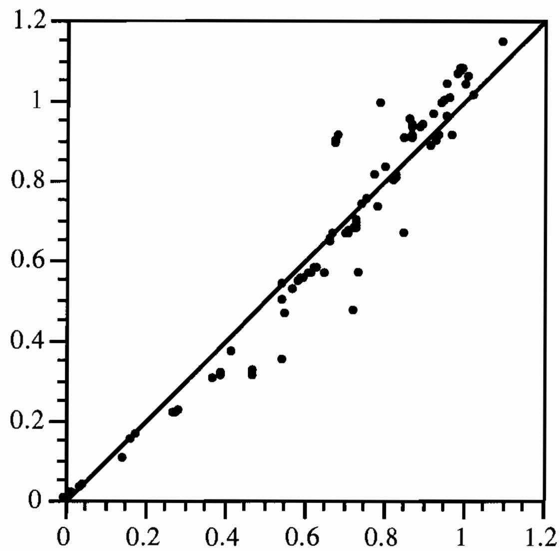

Figure 5 Distances in high-dimensional space (horizontal axis) versus distances in two-dimensional space (vertical axis) for PRINCESS analysis of complete Hartigan data.

As can be seen in Figure 4, in terms of object scores the solutions of the PRINCESS and the PRINCALS analysis of the complete Hartigan data are quite similar. The normalized STRESS (see section 7) of the PRINCESS solution is 0.0080. Therefore, Tucker's squared coefficient of congruence between the elements in $\{d_{ij}(\tilde{\mathbf{Z}})\}$ and in $\{d_{ij}(\mathbf{X})\}$ is $1 - 0.0080 = 0.9920$ for the PRINCESS solution. Figure 5 clearly illustrates how STRESS approximates the high-dimensional distances from both sides.

In Figure 6, the plot of the object scores obtained with PRINCESS is given labeled for each of the six variables separately. As these plots show, the first dimension discriminates the hardware with thread, having a cone, round or cylinder head with a star or slit indentation, from the hardware without thread, having a flat or cup head without indentation (first three plots in Figure 6). The first dimension also separates the hardware with a flat bottom from the hardware with a sharp bottom (fourth plot in Figure 6). The second dimension nicely separates the shortest hardware (one inch) from the hardware with greater lengths (fifth plot in Figure 6). The second dimension also discriminates between the hardware with brass from the hardware without brass (last plot in Figure 6).

If we interpret the plots in Figure 4 in terms of clusters, we detect four groups of very similar objects. The first cluster in the upper left quadrant contains all nails without brass (objects 2 through 9). The screws without brass form a second cluster in the upper right quadrant of the plot (objects 10 through 14). The bolts of one inch without brass are the third cluster, located in the lower right quadrant of the plot (objects 16 through 20). And the fourth cluster consists of two tacks and one nail with brass (objects 21, 22, and 23). Objects 1, 15,

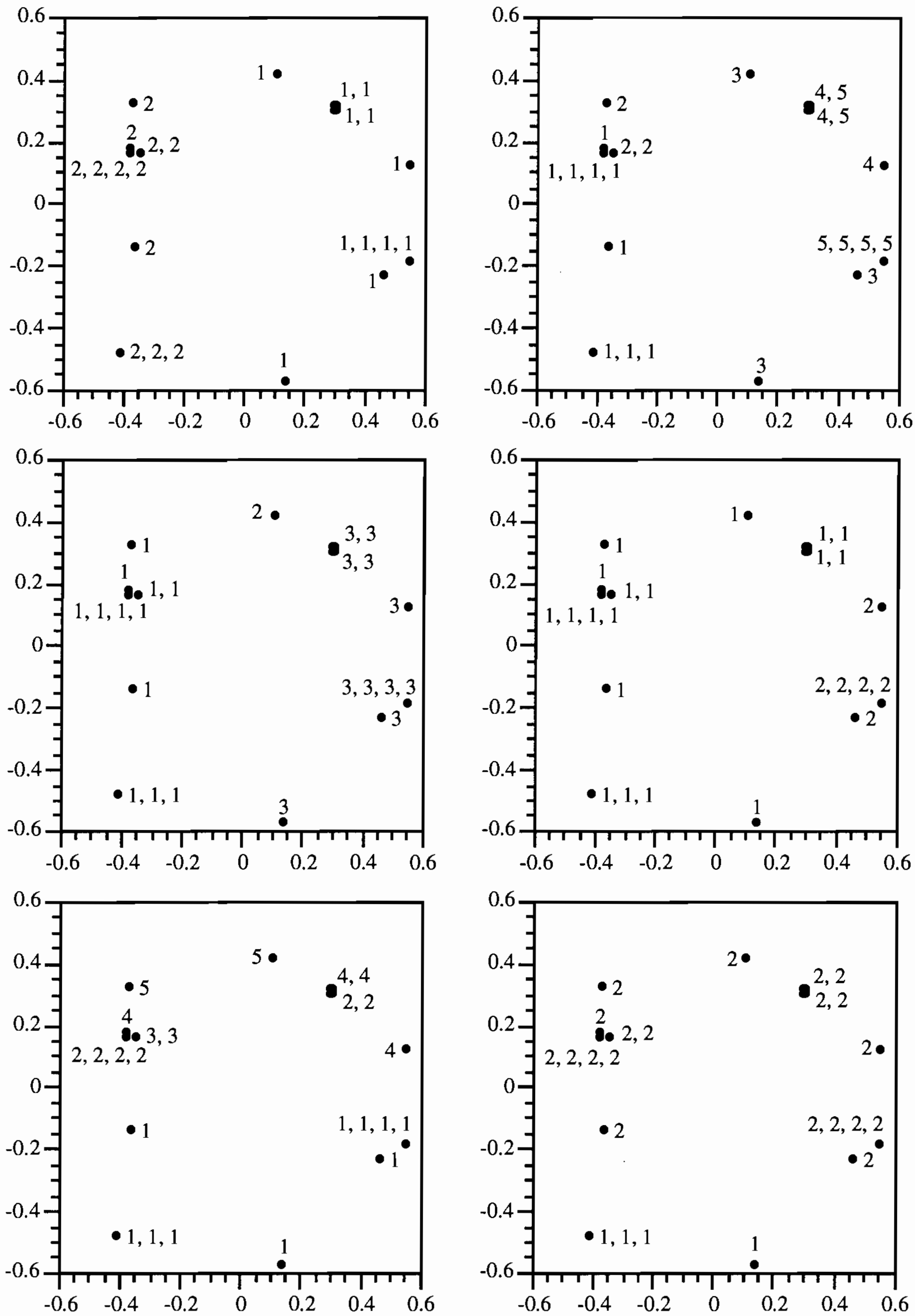

Figure 6 Hartigan object scores labeled by categories of variables Thread (top left), Head (top right), Head indentation (middle left), Bottom (middle right), Length (bottom left), and Brass (bottom right).

and 24, on the other hand, are located somewhere in between the four clusters. Object 1 matches the first cluster on all criteria but one: its length is one inch, and it is therefore located further down on the second dimension. In the same way, object 15 is located between the second and the third cluster, because it shares characteristics of both groups, and object 24 is located between the third and the fourth cluster, because it has something of both these clusters.

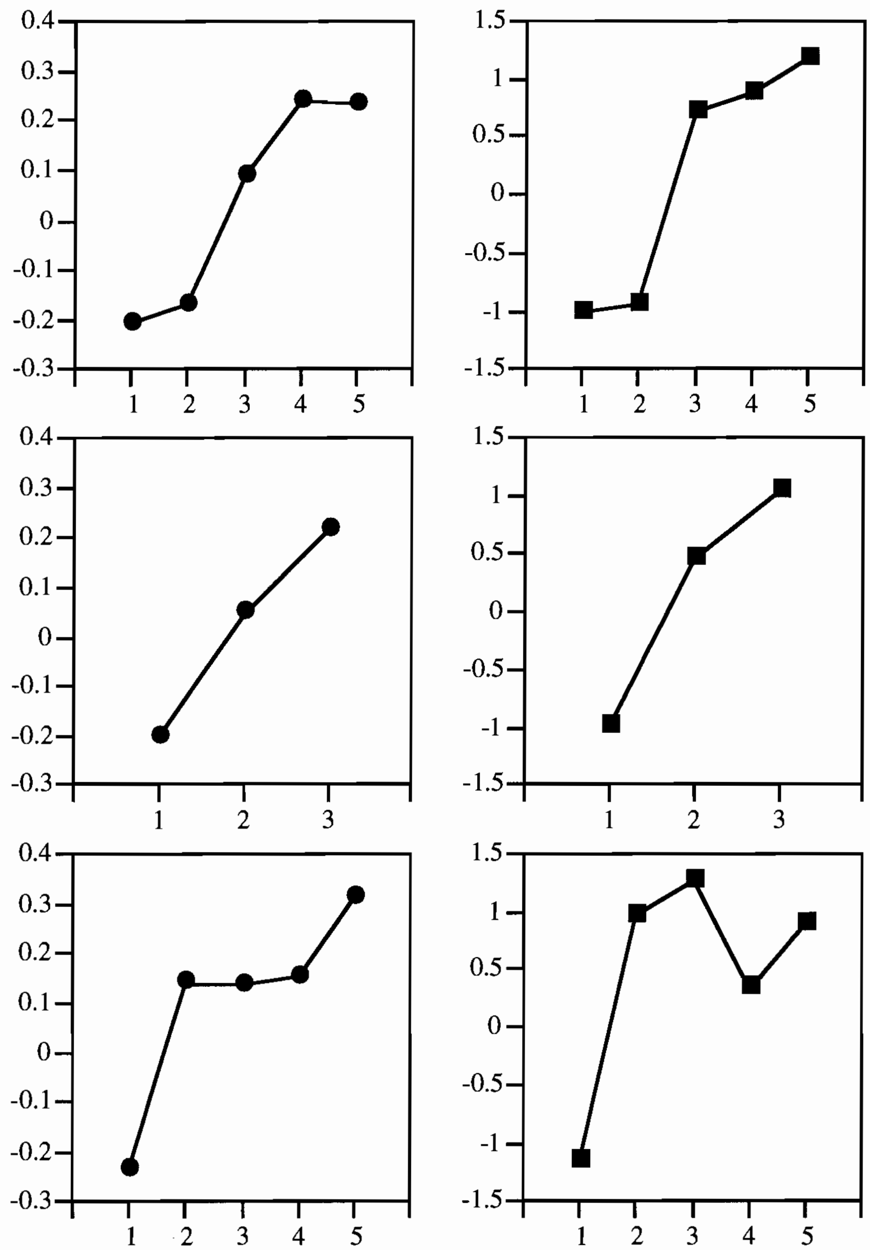

Figure 7 Transformations of the variables Head (top), Head indentation (middle), and Length (bottom) for the complete Hartigan data. PRINCESS transformations are on the left, those of PRINCALS on the right.

Comparing these plots with the solution obtained with HOMALS (see Gifi, 1990), where the variables are treated as multiple nominal data, it is interesting to see that the solution for single nominal data has a much nicer interpretation, especially with respect to the second dimension. Instead of distinguishing between the categories of the variables Length and Brass, the second dimension of the HOMALS solution primarily separates object 10 from the remaining hardware.

In section 1, we mentioned that nonlinear PCA via STRESS and classical nonlinear PCA retain different properties of the high-dimensional data matrix **Z**. As a result, the two methods yield different category quantifications of the variables in the non-linear case. To illustrate this point, in Figure 7 we have plotted the optimal category quantifications for PRINCESS and PRINCALS versus the original categories of the three non-dichotomous variables Head, Head indentation, and Length of the complete Hartigan data. The plots for the dichotomous variables Thread, Bottom, and Brass are not shown because these transformations are linear by definition. The difference in scale between the PRINCESS and PRINCALS plots in Figure 7 is the result of different variable normalization conventions used in the two methods.

For the variable Head both analyses yield monotone transformations of the original categories, except for the quantification of the fourth category (round head) in the STRESS solution which is slightly larger than that for the fifth category (cylinder head). But the type of monotone transformation is clearly different in the two solutions: in PRINCESS category 3 is better separated from categories 4 and 5 than in PRINCALS. For the variable Head indentation, we have almost identical (monotone) transformations in both analyses. By far the largest difference is to be found in the quantifications of the variable Length. While the distance approach yields an almost monotone transformation of the categories of this variable, PRINCALS results in a transformation which does not preserve their rank order at all. Also, while category 5 of variable Length obtains a distinct quantification in PRINCESS, this is more the case for category 4 in PRINCALS.

We will now discuss two examples where the data are no longer complete. We constructed the first example by randomly throwing away information about 44 of the 144 cells in the Hartigan data matrix, yielding the matrix given in Table 2. These data were analysed with the algorithm discussed in section 5, and with PRINCALS. Figure 8 contains plots of the object scores obtained in both analyses.

Inspection of the plots in Figure 8 shows that the PRINCESS and PRINCALS analysis of the data in Table 2 yield remarkably different results. In classical nonlinear PCA, object 18 is treated as an outlier which strongly dominates the solution. This may be caused by the fact that this is the only object for which information is available on only one variable: all we know about object 18 is that it has a flat bottom.

The normalized STRESS for the PRINCESS solution is 0.0314, and Tucker's squared coefficient of congruence is therefore 1 – 0.0314 = 0.9686. In Figure 9 we have plotted the elements in $\{d_{ij}(\tilde{\mathbf{Z}}\mathbf{M}_i\mathbf{M}_j)\}$ against those in $\{w_{ij}^{1/2}d_{ij}(\mathbf{X})\}$ for the PRINCESS solution. Because there are 9 missing distances in the data set in Table 2, the plot in Figure 9 contains

Table 2 Incomplete Hartigan hardware data (1).

Variable	1	2	3	4	5	6
1. Tack	2	1	–	1	1	2
2. Nail1	2	1	1	–	–	2
3. Nail2	–	1	1	1	–	2
4. Nail3	2	1	–	1	2	2
5. Nail4	–	1	1	1	2	2
6. Nail5	2	1	1	1	2	2
7. Nail6	2	2	1	–	5	2
8. Nail7	–	2	1	–	–	2
9. Nail8	–	–	1	1	3	2
10. Screw1	1	3	2	1	–	2
11. Screw2	1	4	–	–	4	–
12. Screw3	1	–	3	–	4	–
13. Screw4	1	4	–	–	2	–
14. Screw5	–	5	–	1	2	–
15. Bolt1	1	4	–	–	4	2
16. Bolt2	1	3	–	2	–	2
17. Bolt3	1	–	3	–	1	2
18. Bolt4	–	–	–	2	–	–
19. Bolt5	1	–	–	2	1	2
20. Bolt6	1	5	3	2	1	2
21. Tack1	2	1	1	1	1	–
22. Tack2	2	1	1	1	1	1
23. Nailb	2	1	1	–	–	–
24. Screwb	1	3	3	1	–	1

Variables	Categories		
1. Thread	1 = yes	2 = no	
2. Head	1 = flat	2 = cup	3 = cone
	4 = round	5 = cylinder	
3. Head indentation	1 = none	2 = star	3 = slit
4. Bottom	1 = sharp	2 = flat	
5. Length	in half inches		
6. Brass	1 = yes	2 = no	

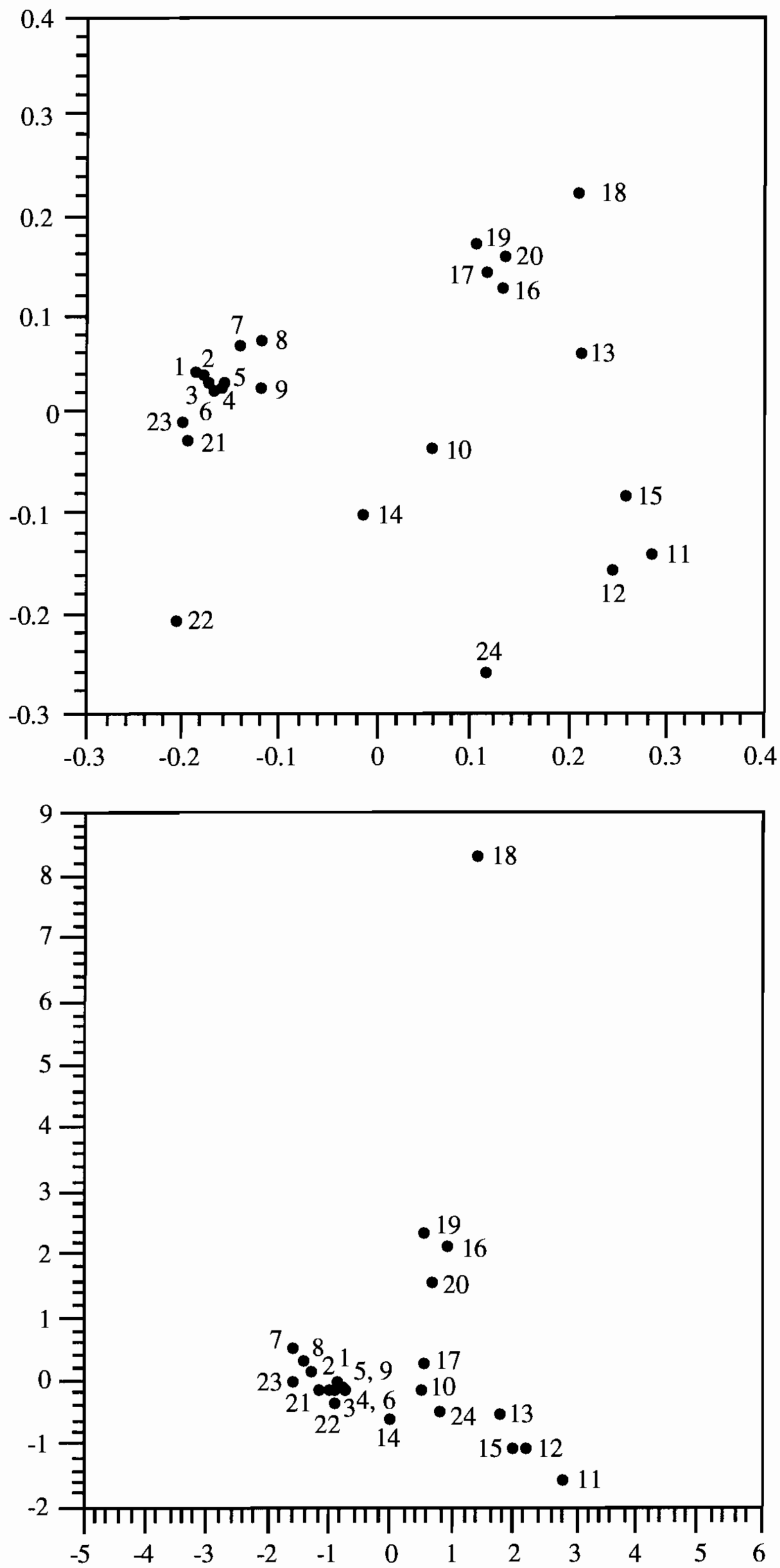

Figure 8 Plots of object scores obtained with PRINCESS (top) and PRINCALS (bottom) analysis of incomplete Hartigan data (1).

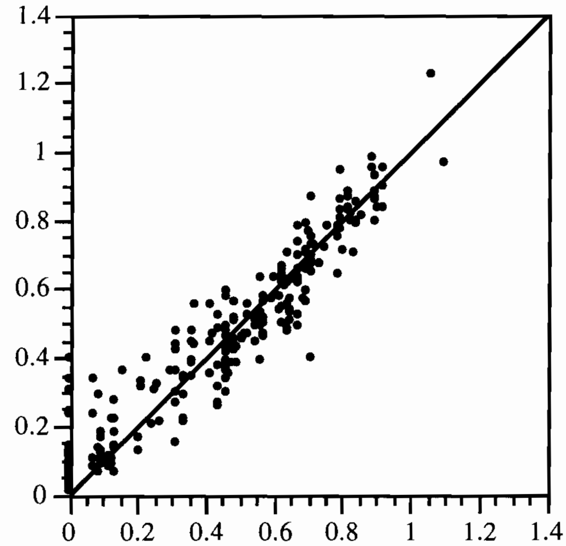

Figure 9 Distances in high-dimensional space (horizontal axis) versus distances in two-dimensional space (vertical axis) for PRINCESS analysis of incomplete Hartigan data (1).

276 – 9 = 267 points. Comparing the normalized STRESS of the present PRINCESS solution with that obtained in the analysis of the complete matrix of Table 1, we find that the former has a larger value. This is caused by violations of the triangle inequality arising in the case of missing data (see section 7). More specifically, we have calculated that out of the $\binom{24}{3}$ = 2024 distinct triples of high-dimensional distances δ_{ij} defined in (2) in the present data set, 162 triples can not be evaluated because one or more distances are missing, while 898 triples do not satisfy the triangle inequality. Obviously, this introduces additional STRESS compared to the complete case where the triangle inequality is never violated.

In Figure 10, the plot of the object scores obtained with PRINCESS is shown labeled for each of the six variables separately. Just as in the complete case, the first dimension discriminates the hardware with thread, having a cone, round or cylinder head with a star or slit indentation, from the hardware without thread, having a flat or cup head without indentation (first three plots in Figure 10). The separation of hardware with a flat bottom from that with a sharp bottom that we found on the first dimension of the complete solution is still present (fourth plot in Figure 10). Also just as in the complete case, the second dimension separates the hardware with brass from the hardware without brass (last plot in Figure 10). Unlike the solution for the complete data, however, the second dimension no longer discriminates the short hardware of one inch from the longer hardware (fifth plot in the figure).

In the PRINCESS solution, we see that objects 1 through 9 together with objects 21 and 23 form a rather tight group (see the upper plot in Figure 8). This group contains all the nails

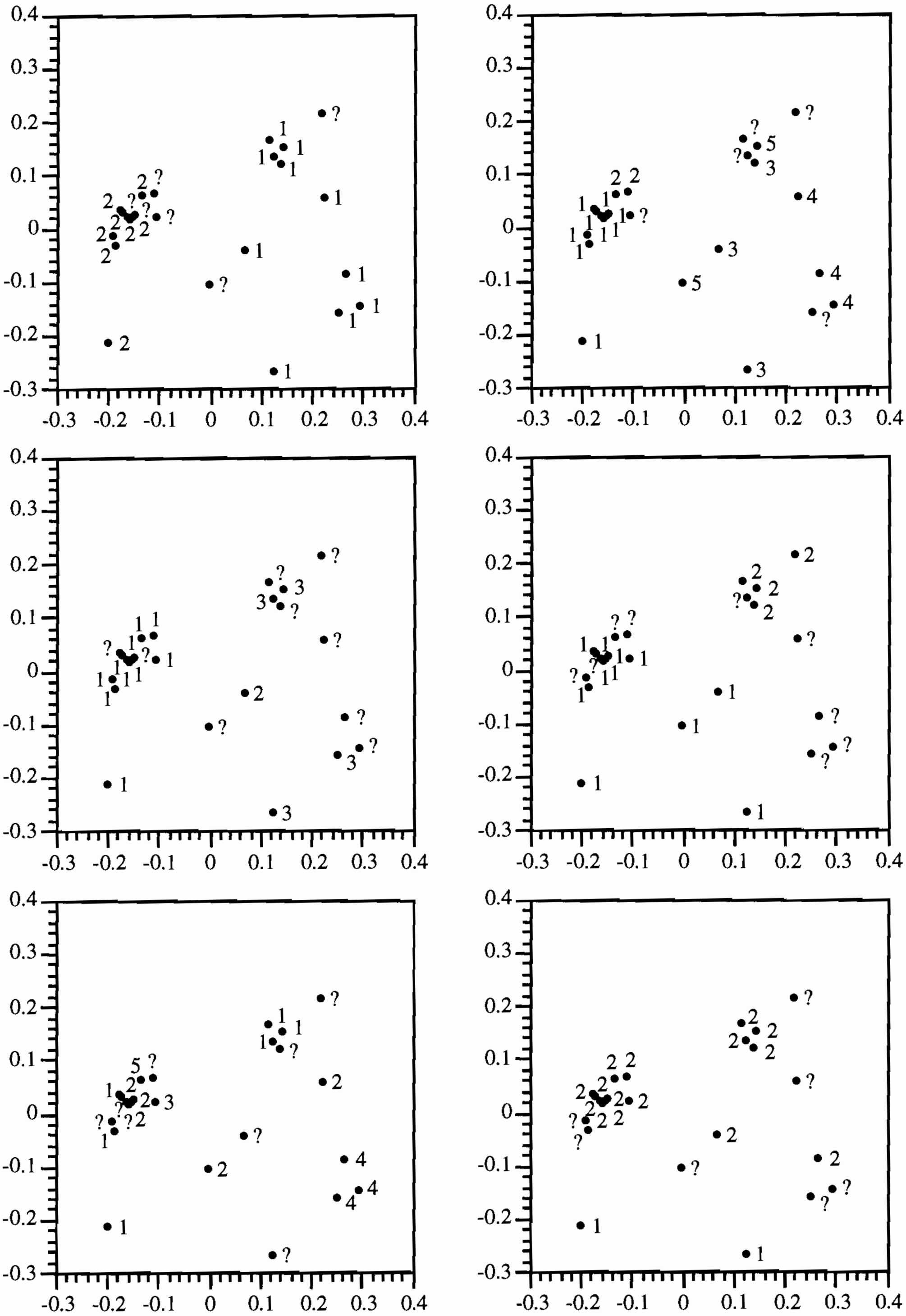

Figure 10 Object scores of PRINCESS analysis of incomplete Hartigan data (1) labeled by variable Thread (top left), Head (top right), Head indentation (middle left), Bottom (middle right), Length (bottom left), and Brass (bottom right). Question marks denote missing categories.

in the data set, and all the tacks, except for Tack2 (object 22) which is different from the other objects in this cluster in one important aspect: it is a brass tack. That objects 21 and 23 (Tack1 and Nailb) have joined this cluster, while they formed a separate cluster (together with Tack2) in the complete solution, has a very logical explanation. Information about variable Brass is now missing for these two objects, and, as far as information is available on the other criteria, they share the following characteristics with all of the remaining objects of the cluster: no thread, and a flat or cup head without indentation, together with a sharp bottom.

A second cluster contains objects 16, 17, 18, 19 and 20, that is, all the bolts in the data set with the exception of Bolt1. The combination of criteria that distinguishes these objects from the others is that they have a thread, a flat bottom, and that they are one inch long. This cluster is identical to the third cluster that we observed in the STRESS solution for the complete data set.

The last cluster that can be detected in the plot at the top of Figure 8 consists of objects 11, 12, and 15. These are the only three objects having a length of four inches. Also, they all three have a thread, and both objects 11 and 15 have a round head (information about this variable is missing for object 12).

Objects 10, 13, 14, 22, and 24 do not really belong to any of these three clusters, and are located somewhere in between the clusters (objects 10, 13, and 14), or at the edge of the plot (objects 22 and 24). Object 22 (Tack2) is special. Its contribution to the STRESS is by far the largest. This can also be seen in the scatter plot in Figure 9, where the two largest low-dimensional distances corresponding to a value of zero on the horizontal axis are the distances between objects 21 and 22, and objects 22 and 23. Table 2 shows that objects 21 and 22 are identical on all five criteria about which they share information, and these two objects should therefore obtain the same object scores in the two-dimensional solution. The same applies to objects 22 and 23, although these two objects only share information on three variables. The problem is probably caused by the variable Brass. The first category of this dichotomous variable only occurs twice in the present data set, and therefore gets a disproportionally large quantification, pushing objects 22 and 24 with brass far away from all the other objects without brass. At the same time, information about the variable Brass is missing for objects 21 and 23. As a result the latter two objects match the criteria of object 22 as well as those corresponding to the cluster containing objects 1 through 9. Apparently, the best compromise is to locate object 22 as in the upper plot of Figure 8.

Since the incomplete Hartigan data in Table 2 contain six variables, six different weights w_{ij} defined in (3) can arise in (5). These are, of course, 1, 2, 3, 4, 5, and 6. The mean raw residuals $[\delta_{ij} - d_{ij}(\mathbf{X})]^2$ associated with these weights in the present PRINCESS solution are

0.0152, 0.0045, 0.0021, 0.0012, 0.0024, and 0.0021, respectively. Ideally, the values of these mean residuals should rank from high to low, because in that case the algorithm would have consistently succeeded in better fitting distances based on ample between-object information than distances based on scarce between-object information. As the numbers in this example show, in practice this ideal is only partly fulfilled.

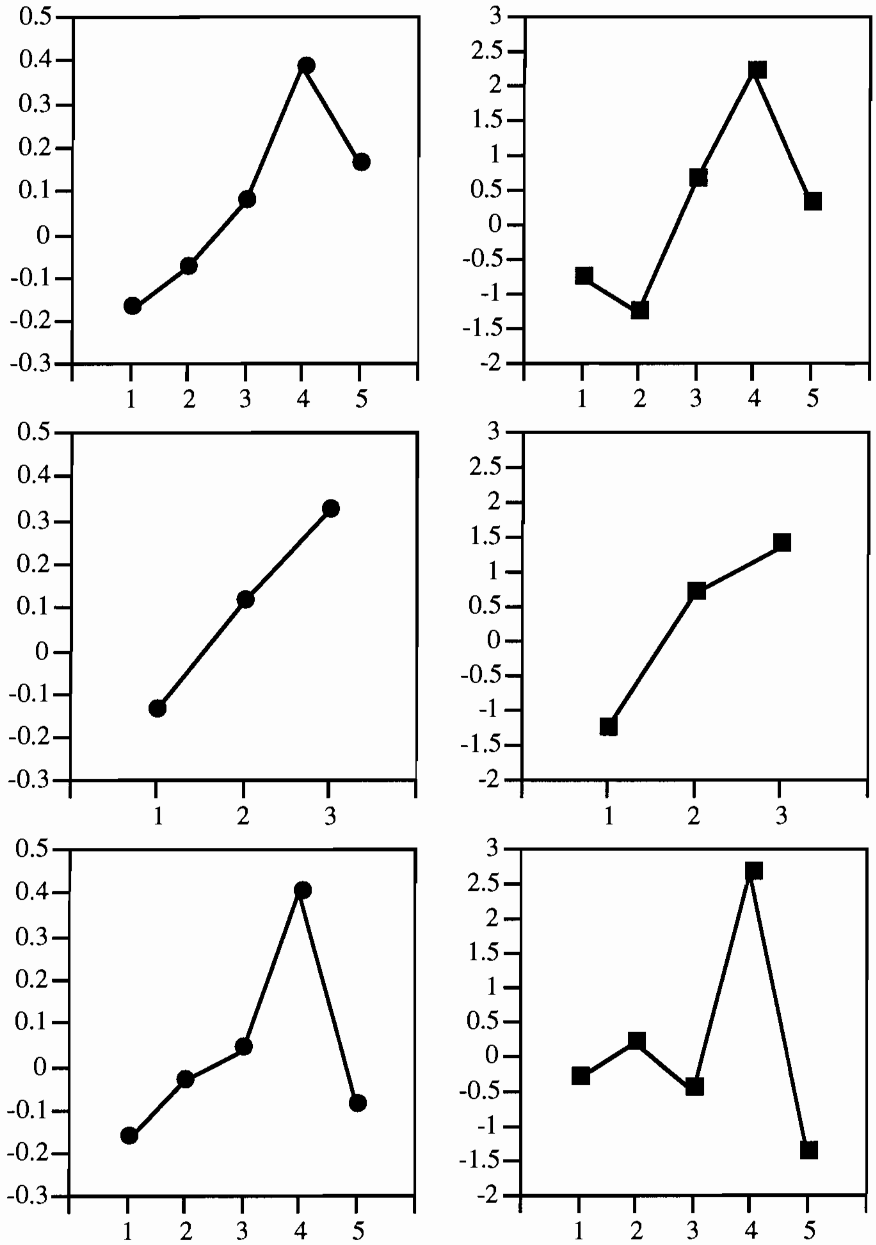

Figure 11 Transformations of the variables Head (top), Head indentation (middle), and Length (bottom) for the incomplete Hartigan data (1). PRINCESS transformations are on the left, PRINCALS transformations on the right.

Figure 11 contains the transformations plots of the categories of the three non-dichotomous variables for the PRINCESS and the PRINCALS solution. The figure shows that the transformations for the variable Head indentation are quite similar, while the two analyses yield different nonmonotone transformations for the variables Head and Length. This again illustrates that PRINCESS and PRINCALS retain different properties of high-dimensional space. Comparing the transformation plots in Figure 11 with those in Figure 7, we see that the analyses of the incomplete Hartigan data yields other transformations of the categories of the variables Head and Length than the analyses of the complete data matrix. This applies to PRINCESS as well as to PRINCALS.

We end this example by noting that we obtained the PRINCESS solution by using random starts, and keeping the best result. Therefore, this solution almost certainly represents the *global* minimum of (6). Applying the rational start discussed in section 6, the algorithm converges to a local minimum, with a normalized function value of 0.0417 (or, equivalently, a squared coefficient of congruence of 1 – 0.0417 = 0.9583). A plot of the object scores for this solution is given in Figure 12. Comparing Figures 12 and 8, we find that the main difference between the two solutions consists of a swapping in location of the cluster containing objects 11, 12, and 15 with the one containing objects 13, 16, 17, 18, 19, and 20.

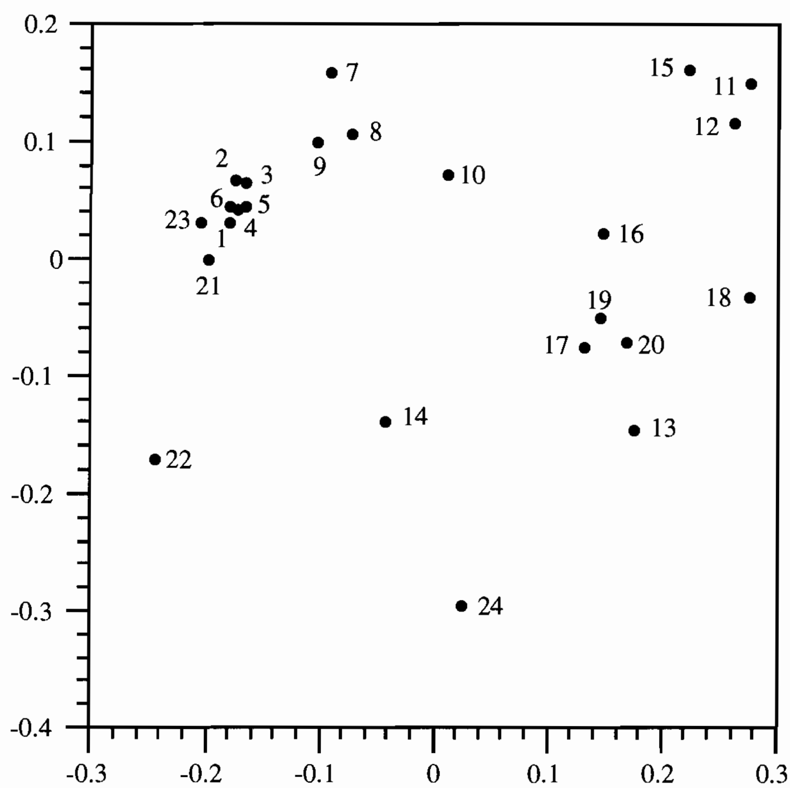

Figure 12 PRINCESS solution of incomplete Hartigan data (1) using the rational start for the algorithm.

For the second example of the STRESS analysis of an incomplete data matrix, we randomly threw away half of the information contained in the complete Hartigan data (i.e., 72 of the 144 cells in the matrix). Because this resulted in object 4 containing missing values on all variables, we arbitrarily reinserted its original value for the variable Length. This yielded the data matrix given in Table 3, which only contains 73 nonmissing cells. In Figure 13 the plots of the object scores from PRINCESS and PRINCALS are given. As these plots show, the present PRINCESS and PRINCALS solutions at least have somewhat more in common than in the previous incomplete example.

The normalized STRESS for the present PRINCESS solution is 0.0625, and Tucker's squared coefficient of congruence is 1 – 0.0625 = 0.9375. In Figure 14, we have plotted the elements in $\{d_{ij}(\tilde{\mathbf{Z}}\mathbf{M}_i\mathbf{M}_j)\}$ against those in $\{w_{ij}^{1/2} d_{ij}(\mathbf{X})\}$ for the PRINCESS solution. Since there are 47 missing distances in the present data set, the plot in Figure 14 contains 276 – 47 = 229 points.

Again, we find that the PRINCESS analysis of the incomplete Hartigan data given in Table 3 yields a larger STRESS value than the PRINCESS analysis of the complete data

Table 3 Incomplete Hartigan hardware data (2).

Variable	1	2	3	4	5	6
1. Tack	–	1	1	–	–	2
2. Nail1	2	1	1	1	–	2
3. Nail2	2	–	1	1	2	2
4. Nail3	–	–	–	–	2	–
5. Nail4	2	1	–	1	–	–
6. Nail5	–	1	1	1	–	–
7. Nail6	–	2	1	–	5	2
8. Nail7	2	2	1	–	3	2
9. Nail8	2	2	–	1	–	2
10. Screw1	1	–	–	1	–	–
11. Screw2	1	–	3	–	4	–
12. Screw3	–	–	–	1	–	–
13. Screw4	–	4	3	1	–	2
14. Screw5	1	–	–	–	–	2
15. Bolt1	–	4	–	2	4	–
16. Bolt2	–	–	3	–	1	2
17. Bolt3	1	–	–	–	1	–
18. Bolt4	–	5	3	2	1	–
19. Bolt5	–	5	3	2	–	2
20. Bolt6	1	–	–	–	–	2
21. Tack1	2	–	–	–	1	1
22. Tack2	2	–	–	–	–	1
23. Nailb	2	1	–	–	–	–
24. Screwb	–	3	3	–	–	1

Variables	Categories		
1. Thread	1 = yes	2 = no	
2. Head	1 = flat	2 = cup	3 = cone
	4 = round	5 = cylinder	
3. Head indentation	1 = none	2 = star	3 = slit
4. Bottom	1 = sharp	2 = flat	
5. Length	in half inches		
6. Brass	1 = yes	2 = no	

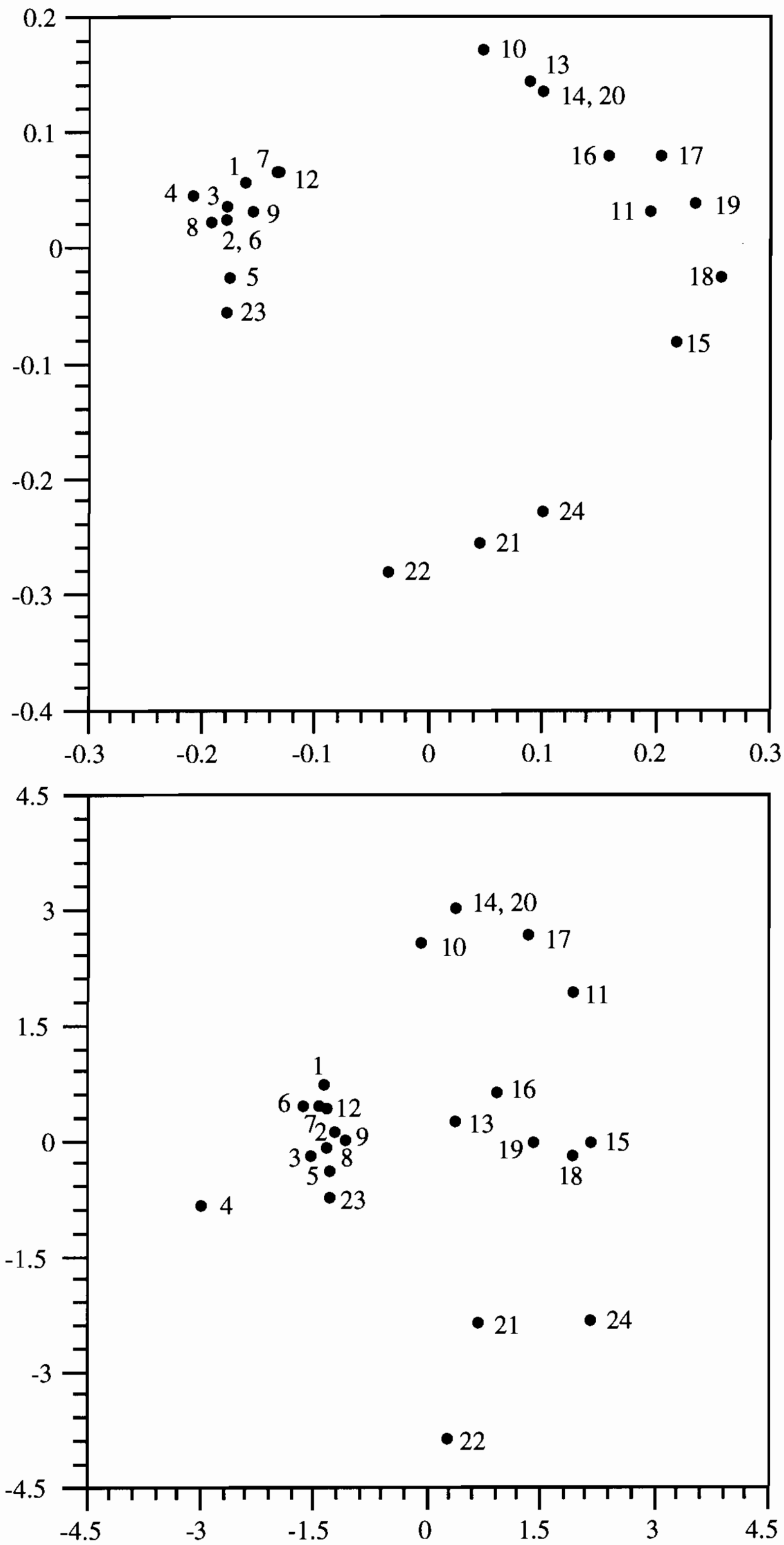

Figure 13 Plots of object scores obtained with PRINCESS (top) and PRINCALS (bottom) analysis of incomplete Hartigan data (2).

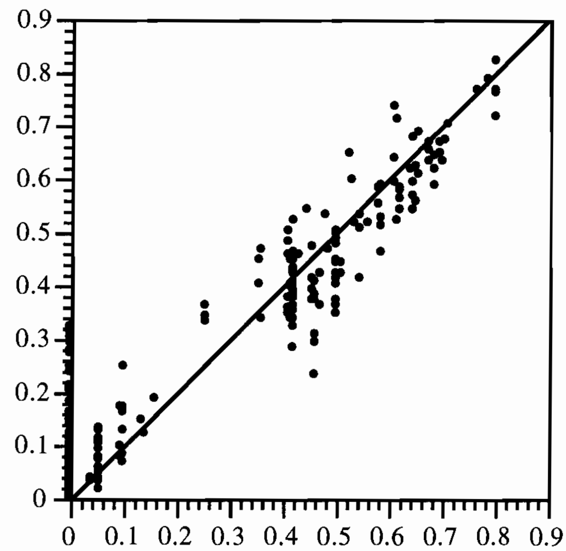

Figure 14 Distances in high-dimensional space (horizontal axis) versus distances in two-dimensional space (vertical axis) for PRINCESS analysis of incomplete Hartigan data (2).

matrix. Out of the 2024 triples of high-dimensional distances (2) in the incomplete Hartigan data in Table 3 that can be formed, 777 triples can not be evaluated because of missing distances, while 737 triples violate the triangle inequality. This introduces additional STRESS compared to the complete case.

Figure 15 contains plots of the PRINCESS object scores labeled for each variable separately. Inspection of these plots shows that, just as in the complete case, the first dimension of the solution separates the hardware with a flat or cup head without indentation from the hardware with a cone, round, or cylinder head with indentation (second and third plot in Figure 15). As in the complete case, the first dimension also discriminates between the hardware with a sharp bottom and that with a flat bottom (fourth plot in the figure). Unlike the solution for the complete data, however, the first dimension additionally separates the hardware of one and four inch from the hardware of two, three, and five inch (fifth plot in Figure 15). The distinction between the hardware with and without thread, that corresponds to the first dimension in the previous examples, is still present in the solution, but on an axis which now runs from the bottom left corner to the top right corner of the plot (first plot in the figure).

Just as in the previous two examples, the second dimension discriminates the hardware with brass from the hardware without brass (last plot in the figure). The separation between the hardware of one inch from the longer hardware that we found in the complete solution has disappeared in the present analysis. This was also the case in the analysis of the first incomplete data set.

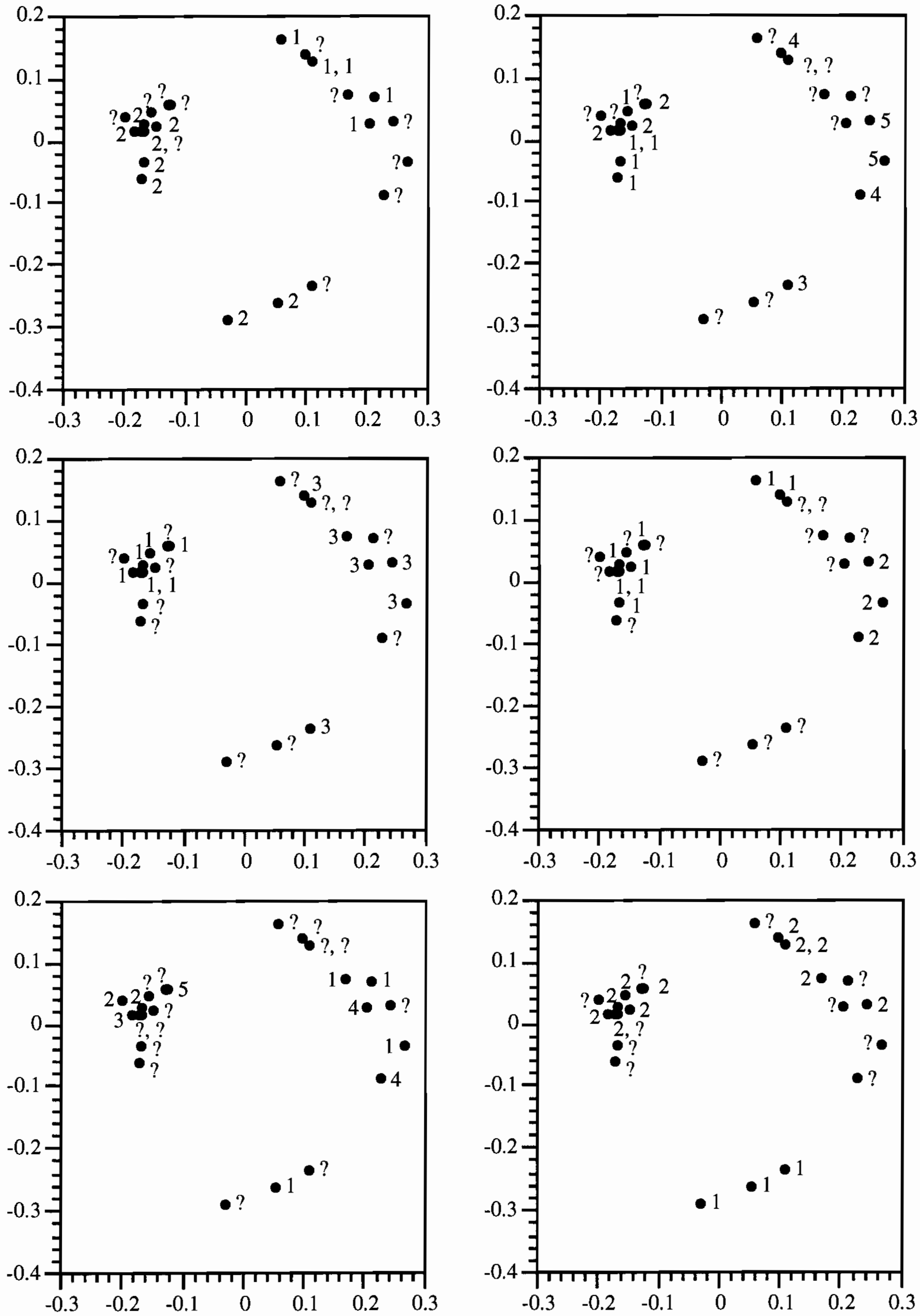

Figure 15 Object scores of PRINCESS analysis of incomplete Hartigan data (2) labeled by variable Thread (top left), Head (top right), Head indentation (middle left), Bottom (middle right), Length (bottom left), and Brass (bottom right). Question marks denote missing categories.

The plot roughly contains three clusters. The first cluster is identical to the first cluster in the previous example, except for the fact that object 12 has now joined the cluster, while object 22 has left it. The cluster contains all the nails in the data set, including Nailb (object 23). It also contains Tack1 (object 1) and Screw3 (object 12). As far as information is available, what uniquely binds these objects is that they all have a flat or cup head without indentation, and that their length is 2, 3 or 5 inches. Object 23 again belongs to this cluster because information on the variable Brass is missing for this object.

Object 22 has left the previous cluster because we now know that it has brass. The second cluster therefore consists of all the objects with brass: objects 21, 22, and 24.

The third rather loose cluster contains all the screws without brass (except for Screw3, i.e., object 12), and all the bolts in the data set. What makes these objects unique is that they have a thread, and a round or cylinder head. A subcluster within this cluster consists of the objects 15, 18, and 19, which have a flat bottom.

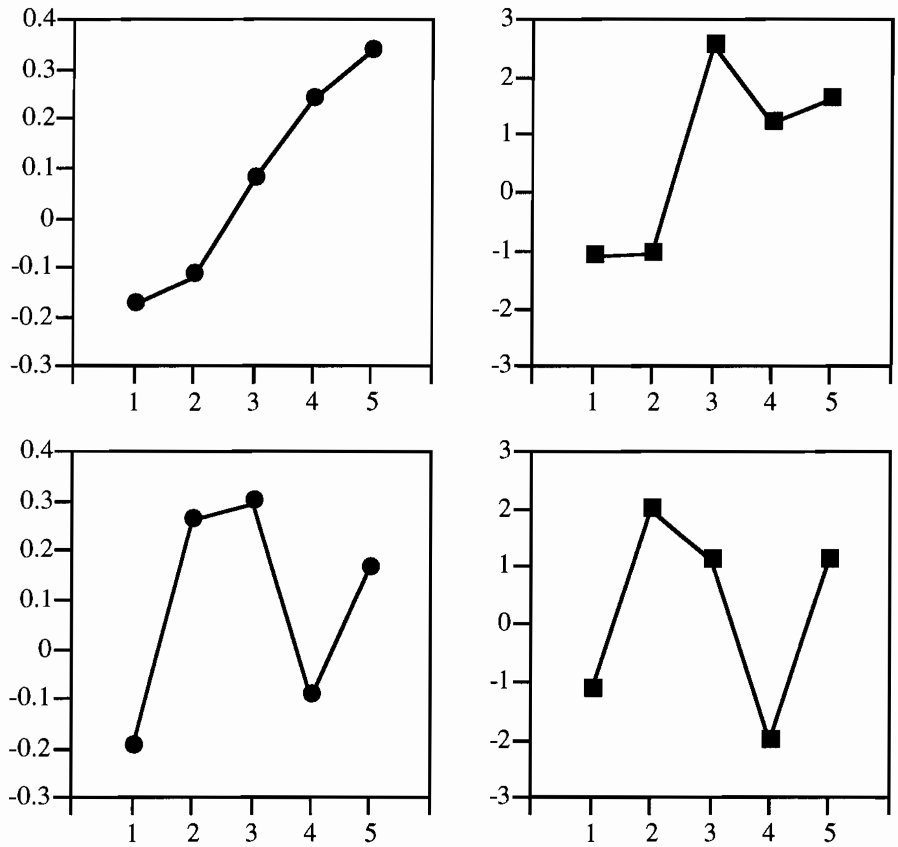

Figure 16 Transformations of the variables Head (top) and Length (bottom) for the incomplete Hartigan data (2). PRINCESS transformations are on the left, PRINCALS transformations on the right.

The only weights (3) occurring in this data set are 1, 2, 3, and 4. The mean residuals associated with these weights are 0.0179, 0.0020, 0.0013, and 0.0004, respectively. Therefore, the algorithm was consistently successful in better fitting distances with larger weights than distances with smaller weights. Trials with different random starts in the PRINCESS algorithm suggest that the solution for the incomplete Hartigan data (2) that we just discussed can not be improved upon.

Figure 16 contains the transformation plots of PRINCESS and PRINCALS for the only two non-dichotomous variables left in the data matrix in Table 3: Head and Length. These plots again illustrate how classical nonlinear PCA and the distance approach to nonlinear PCA optimize different aspects of a numerically identical high-dimensional space **Z**. We finally note that, for PRINCESS as well as for PRINCALS, the transformations in Figure 16 are different from those obtained in the analysis of the complete Hartigan data (see Figure 7), except for the PRINCESS transformations of the categories of variable Head which are almost identical.

9 Discussion

In this paper a solution has been proposed for the problem of missing data in the distance approach to principal components analysis. The solution may be classified in terms of what Little and Rubin called the 'available case method' or, equivalently, of what is called the 'missing data deleted option' in the Gifi-system. The root of the mean squared differences between the nonmissing elements in each row pair of the data matrix is used as a derived dissimilarity, to be approximated in low-dimensional space. The latter is combined with an implementation of weights in the STRESS loss function, to the effect that residuals corresponding to distances based on ample between-object information are penalized more heavily than residuals corresponding to distances based on scarce between-object information.

This method for performing a principal components analysis of incomplete data via STRESS seems to give satisfactory results. In the incomplete case, the examples discussed in section 8 show that the low-dimensional configuration of objects obtained with the classical nonlinear PCA program PRINCALS can be remarkably different from the one obtained with PRINCESS. Since PRINCESS explicitly aims at an optimal low-dimensional representation of the distances between the objects of a given data matrix, for the incomplete case this suggests that one should be careful in giving a distance interpretation to the object scores obtained with PRINCALS, as is often done in practice.

Compared to the complete case, the available case method for the analysis of incomplete data proposed in this paper inevitably introduces an additional source of stress caused by violations of the triangle inequality. All other things being equal, this has the effect that the sum of discrepancies between high- and low-dimensional distances will usually be larger for incomplete than for complete data.

As concerns further research, we yet have to investigate whether other rational starts of the algorithm than the one proposed in section 6 may help to avoid convergence to local minima, a situation that we encountered in one of the examples. We also intend to investigate ways to speed up the convergence rate of the algorithm discussed in section 5 by using a fixed stepsize parameter, as has been discussed in, e.g., De Leeuw and Heiser (1980).

Acknowledgements

I thank Patrick Groenen, Willem Heiser, Ivo van der Lans, and Jacqueline Meulman for helpful suggestions and comments concerning the contents of this paper.

References

De Leeuw, J. (1977). Application of convex analysis to multidimensional scaling. In J.R. Barra et al. (Eds), *Recent developments in statistics.* Amsterdam: North Holland Publishing company.

De Leeuw, J. (1984). *Convergence of the majorization algorithm for multidimensional scaling.* Research Report RR-84-07. Leiden: Department of Data Theory.

De Leeuw, J. & Heiser, W.J. (1977). Convergence of correction matrix algorithms for multidimensional scaling. In J. Lingoes et al. (Eds.), *Geometric representations of relational data.* Ann Harbor: Mathesis Press.

De Leeuw, J. & Heiser, W.J. (1980). Multidimensional scaling with restrictions on the configuration. In: P.R. Krishnaiah (Ed.), *Multivariate analysis, Vol V.* Amsterdam: North Holland Publishing company.

De Leeuw, J. & Meulman, J.J. (1985). *Alternative approximations in principal components analysis.* Research Report RR-85-13. Leiden: Department of Data Theory.

Gifi, A. (1990). *Nonlinear multivariate analysis.* Chichester: Wiley.

Gower, J.C. (1966). Some distance properties of latent roots and vector methods used in multivariate analysis. *Biometrika, 53*, 325-338.

Greenacre, M.J. & Underhill, L.G. (1982). Scaling a data matrix in a low dimensional Euclidean space. In D.M. Hawkins (Ed.), *Topics in applied multivariate analysis.* Cambridge: Cambridge University Press.

Hartigan, J.A. (1975). Necessary and sufficient conditions for asymptotic joint normality of a statistic and its subsample values. *AS, 3*, 573-580.

Heiser, W.J. & De Leeuw, J. (1977). *SMACOF-I.* (3d ed. 1986). Research Report UG-86-02. Leiden: Department of Data Theory.

Little, R.J. & Rubin, D.B. (1987). *Statistical analysis with missing data.* New York: Wiley.

Meulman, J.J. (1982). *Homogeneity analysis of incomplete data.* Leiden: DSWO Press.

Meulman, J.J. (1986). *A distance approach to nonlinear multivariate analysis.* Leiden: DSWO Press.

Meulman, J.J. (1992). The integration of multidimensional scaling and multivariate analysis with optimal transformations. *Psychometrika, 57*, 539-565.