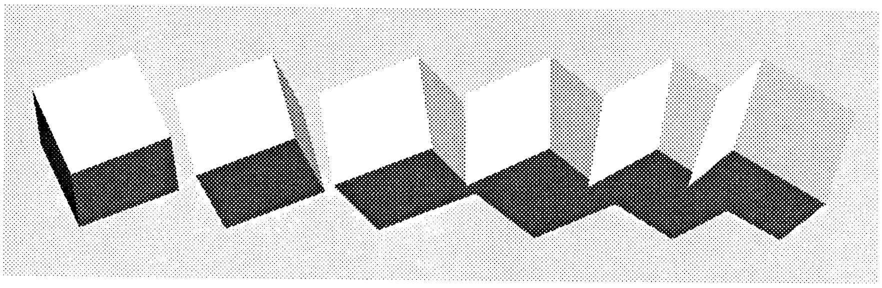

Chapter 4

The stimulus weighting models

4.1 Introduction

In this chapter we return to the situation at the end of Chapter 2, and from there we start off into a completely different direction. Instead of matching n configurations by means of differential weighting of dimensions, we now investigate the effect of differentially weighting the p *stimulus points* of each configuration. An illustration is given in Figure 4.1 where stimulus weighting is applied to a two-dimensional configuration consisting of the vertices of a pentagon. The stimulus points in the figure are numbered from one to five before transformation and from one prime to five prime after transformation. For each stimulus point of the configuration a line is generated by connecting the stimulus point with the origin of m-dimensional space, and this line is then allowed to be shortened or lengthened. In other words, the stimulus points are conceived of as vectors emanating from the origin O, and weighting these vectors has the effect of moving the stimulus points either closer to or further away from the origin.

If we would translate the pentagon in Figure 4.1 to a different location and then connect the stimulus points with the origin, this would yield a different set of vectors

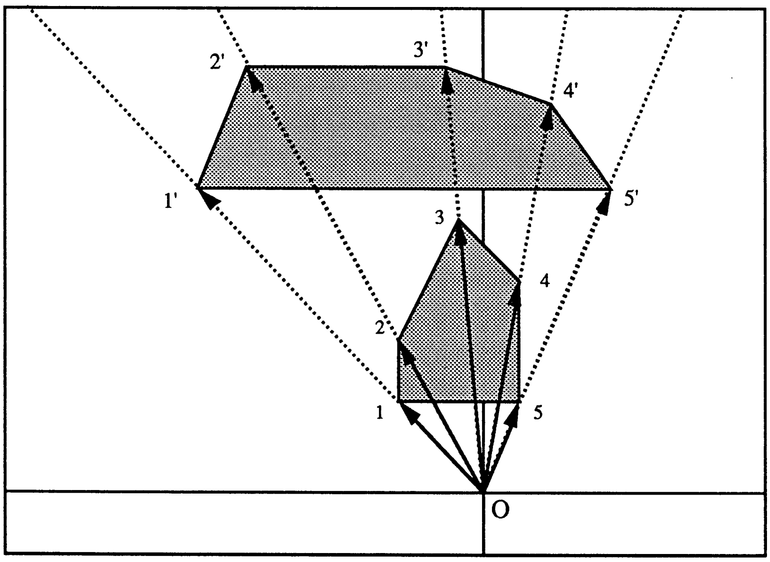

Figure 4.1 Illustration of stimulus weighting.

than the one shown in the figure. Since the weighting of the latter vectors would also result in a different set of configurations, the location of a configuration is clearly of crucial importance in stimulus weighting.

This chapter deals with the matching of n configurations by means of translations, orthonormal transformations, and stimulus weighting. Lingoes and Borg (1978) showed that these three transformations result in two different stimulus weighting models. In the first model only one optimal translation is determined for all configurations, while the second model allows for a different optimal translation for each separate configuration. In the following sections we generalize the stimulus weighting models to m dimensions, as well as to the case of missing data. We also propose a new approach to the fitting of these models, which, just like in the dimension weighting models, depends on determining a new centroid configuration instead of using the centroid configuration $\mathbf{Z}$ obtained in GPA. Before presenting our alternative approach, however, we first discuss the manner in which Lingoes and Borg constructed their models.

4.2 Geometry and algebra

Given n configurations $\mathbf{X}_j$ (j = 1, ..., n) of order (p x m), let $\mathbf{Z}$ of order (p x m) be the corresponding matrix of centroids obtained in GPA. Also, let $\mathbf{g}_j$ and $\mathbf{h}_j$ be unknown translation vectors of order (m x 1), $\mathbf{T}_j$ be an unknown orthonormal matrix of order (m x m), $\mathbf{V}_j$ be an unknown (p x p) diagonal matrix containing stimulus weights, and $\mathbf{E}_j$ be a (p x m) matrix of residuals. With these definitions, Lingoes and Borg proposed the following model:

$$(\mathbf{X}_j - \mathbf{1g}_j')\mathbf{T}_j = \mathbf{V}_j(\mathbf{Z} - \mathbf{1h}_j') + \mathbf{E}_j, \qquad \text{for } j = 1, ..., n, \qquad (4.1)$$

where the unknown $\mathbf{g}_j$, $\mathbf{h}_j$, $\mathbf{T}_j$, and $\mathbf{V}_j$ are to be estimated by minimizing the sum of squared residuals, that is, by minimizing $\sum \text{tr } \mathbf{E}_j'\mathbf{E}_j$. The diagonal matrix $\mathbf{V}_j$ in (4.1) has the effect of changing the lengths of the vectors generated by connecting the stimulus points in $(\mathbf{Z} - \mathbf{1h}_j')$ with the origin. Characteristic of (4.1) is that the weighting of stimulus points is applied to $(\mathbf{Z} - \mathbf{1h}_j')$, and not to the stimulus points of the individual configurations. In principle, one could also use a model like

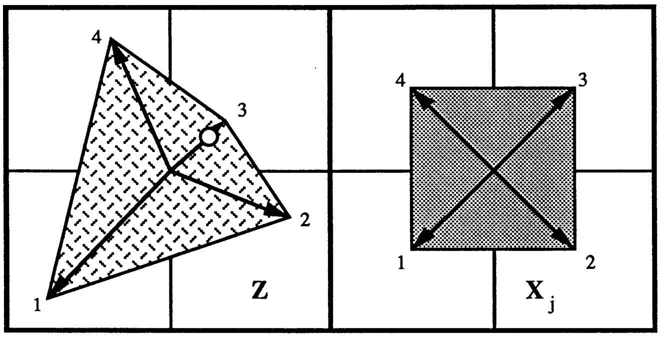

Figure 4.2 Illustration of necessity of translation of **Z** in the case of stimulus weighting.

$$\mathbf{V}_j(\mathbf{X}_j - \mathbf{1g}_j')\mathbf{T}_j = (\mathbf{Z} - \mathbf{1h}_j') + \mathbf{E}_j, \qquad \text{for } j = 1, ..., n, \tag{4.2}$$

where the stimulus points of the individual configurations are weighted. But we will not pursue this course, and stick to the stimulus weighting of the centroid configuration, as Lingoes and Borg did.

The translations $\mathbf{h}_j$ play an essential role in model (4.1). The necessity of an optimal translation of **Z** in the case that its stimulus points are weighted is illustrated with a simple example in Figures 4.2 and 4.3. Supposing that the coordinates of the stimulus points in **Z** and in $\mathbf{X}_j$ are represented by the end points of the vectors shown in Figure 4.2, it is impossible to obtain $\mathbf{X}_j$ by just lengthening or shortening the vectors representing the stimulus points contained in **Z**. Therefore, if only stimulus weights are allowed the match between **Z** and $\mathbf{X}_j$ will never be perfect. A perfect match is immediately obtained, however, by translating **Z** to the point (0.5, 0.4) and then weighting the four vectors representing the four stimuli with weights 0.5, 1.0, 5.0 and (1/1.2), respectively. The point (0.5, 0.4) is indicated by a white circle in Figure 4.2. In matrix notation we have

$$\mathbf{V}_j(\mathbf{Z} - \mathbf{1h}_j') = \begin{bmatrix} 0.5 & 0 & 0 & 0 \\ 0 & 1.0 & 0 & 0 \\ 0 & 0 & 5.0 & 0 \\ 0 & 0 & 0 & 1/1.2 \end{bmatrix} \left(\begin{bmatrix} -1.5 & -1.6 \\ 1.5 & -0.6 \\ 0.7 & 0.6 \\ -0.7 & 1.6 \end{bmatrix} - \begin{bmatrix} 0.5 & 0.4 \\ 0.5 & 0.4 \\ 0.5 & 0.4 \\ 0.5 & 0.4 \end{bmatrix} \right)$$

$$= \begin{bmatrix} -1.0 & -1.0 \\ 1.0 & -1.0 \\ 1.0 & 1.0 \\ -1.0 & 1.0 \end{bmatrix} = \mathbf{X}_j,$$

where $\mathbf{h}_j$ is the optimal translation vector, and $\mathbf{V}_j$ contains the optimal stimulus weights to achieve a perfect match between $\mathbf{Z}$ and $\mathbf{X}_j$. These transformations are also illustrated geometrically in Figure 4.3, where the translation of $\mathbf{Z}$ to the new location (0.5, 0.4) creates the possibility of applying stimulus weights that yield $\mathbf{X}_j$.

In stimulus weighting two translation options are available. The first option is to translate $\mathbf{Z}$ only once such that the match is simultaneously optimized over all n individual configurations. The matrix notation for this model is:

$$(\mathbf{X}_j - \mathbf{1g}_j')\mathbf{T}_j = \mathbf{V}_j(\mathbf{Z} - \mathbf{1h}') + \mathbf{E}_j \quad \text{for } j = 1, ..., n, \tag{4.3}$$

where $\mathbf{h}$ is an unknown translation vector of order (m x 1). Lingoes and Borg (1987) discussed that this model has a psychological interpretation only if the stimulus weights are non-negative. People (represented by the individual configurations) then share a common point of view (represented by the origin of the optimally translated centroid configuration $(\mathbf{Z} - \mathbf{1h}')$) from where they perceive the surrounding world (represented by the centroid configuration itself), but differ in the *centrality* they

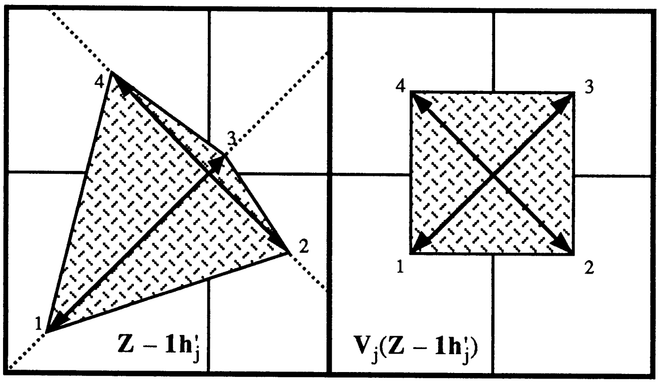

Figure 4.3 Geometry of optimal translation and stimulus weighting of **Z** in Figure 4.2.

assign to the stimuli under investigation. That is, some people experience a stimulus as being more peripheral, while it is a more central feature to other people.

If negative weights occur, however, such an interpretation breaks down. In that case a stimulus point is not only moved closer to or further away from the origin, but its associated vector is also made to point in the opposite direction. But even in this case the weights provide valuable information. Since the stimuli in **Z** are weighted from one and the same optimal origin **h** for all n configurations, the stimulus weights in (4.3) are directly comparable over configurations. Inspection of these weights can therefore give information about which specific stimuli are responsible for the possible increase in fit compared to the more parsimonious GPA model. Weights that considerably differ from +1 (that is, weights that are close to zero or even negative, and weights much larger than +1) indicate that the corresponding stimulus points in **Z** do not represent the stimuli in the individual configurations very well.

The alternative for model (4.3) is to translate **Z** optimally for each individual configuration separately. The algebra corresponding to this model has already been given in (4.1). Psychologically, this model implies that people not only differ in the centrality they assign to the stimuli under investigation, but also perceive the group configuration from different points of view, in contrast with the common point of view they share in model (4.3).

Characteristic of the stimulus weighting models of Lingoes and Borg is that they are fitted using the optimal centroid configuration **Z** obtained in GPA. But **Z** is not necessarily optimal in models (4.1) and (4.3). By sticking to **Z** a source of error is introduced which can obscure the real fit of these models. We therefore propose to determine *new* optimal centroid configurations instead of borrowing them from a previous model. This should eliminate the source of error that we just mentioned.

Let $\tilde{\mathbf{X}}_j$ denote the optimally transformed configuration $\mathbf{X}_j$ after GPA (i.e., $\tilde{\mathbf{X}}_j = s_j(\mathbf{X}_j - \mathbf{1u}_j')\mathbf{R}_j\mathbf{K}$, see Chapter 2). Our alternative for model (4.3) is in matrix notation

$$(\tilde{\mathbf{X}}_j - \mathbf{1g}_j')\mathbf{T}_j = \mathbf{V}_j\mathbf{Y} + \mathbf{E}_j \quad \text{for } j = 1, ..., n, \tag{4.4}$$

where the (p x m) centroid configuration **Y** is assumed to be unknown. It may be noted that the translation vector **h** is lacking in (4.4). This latter vector no longer needs to be estimated since the new matrix of centroids **Y** will automatically be in an optimal location with respect to stimulus weighting. We use the optimally transformed configurations $\tilde{\mathbf{X}}_j$ instead of the raw $\mathbf{X}_j$'s in (4.4) because we expect that this results in more efficient algorithms, the $\tilde{\mathbf{X}}_j$'s already being optimal with respect to distance preserving transformations. Taking into account that information about some stimuli in some configurations may be missing, model (4.4) becomes

$$\mathbf{M}_j(\tilde{\mathbf{X}}_j - \mathbf{1g}_j')\mathbf{T}_j = \mathbf{M}_j\mathbf{V}_j\mathbf{Y} + \mathbf{M}_j\mathbf{E}_j \quad \text{for } j = 1, ..., n, \tag{4.5}$$

where $\mathbf{M}_j$ is a given diagonal matrix of order (p x p) with ones on the diagonal if the corresponding rows in $\mathbf{X}_j$ are not missing, and zeroes elsewhere. In full we call (4.5) the *stimulus weighting model with free centroid configuration*, and will refer to it as the STIMFREE model. The STIMFREE model is developed in section 4.3 of this chapter.

Our alternative for Lingoes and Borg's model (4.1) is in matrix notation

$$(\tilde{\mathbf{X}}_j - \mathbf{1g}_j')\mathbf{T}_j = \mathbf{V}_j(\mathbf{Y} - \mathbf{1h}_j') + \mathbf{E}_j \quad \text{for } j = 1, ..., n, \tag{4.6}$$

where $\mathbf{Y}$ is, again, an unknown centroid configuration. Generalizing (4.6) to the case of incomplete configurations yields the following model

$$\mathbf{M}_j(\tilde{\mathbf{X}}_j - \mathbf{1g}_j')\mathbf{T}_j = \mathbf{M}_j\mathbf{V}_j(\mathbf{Y} - \mathbf{1h}_j') + \mathbf{M}_j\mathbf{E}_j \quad \text{for } j = 1, ..., n. \tag{4.7}$$

In full we call this model the *stimulus weighting model with idiosyncratic translations of a free centroid configuration*, and will refer to it as the STIMIDIO model. The STIMIDIO model is the subject of section 4.4.

As we noted earlier in this section non-negative stimulus weights are amenable to a psychological interpretation while negative weights are difficult to interpret. It is for this reason that we will discuss not only how to determine optimal stimulus weights in models (4.5) and (4.7) when the weights may take any value, but also how to determine weights subject to the condition that they must be non-negative.

The remainder of Chapter 4 is organized as follows. In sections 4.3 and 4.4 the stimulus weighting models (4.5) and (4.7) are developed and algorithms for the estimation of the unknown transformation parameters are presented. Section 4.5 deals with measures of fit and an analysis of variation for both models. In section 4.6 results of the analysis of some constructed data sets according to the STIMFREE and STIMIDIO model are shown.

4.3 The STIMFREE model

4.3.1 Introduction

In the following sections the unknown parameters minimizing the least squares loss function

$$f(\mathbf{G},\mathbf{T},\mathbf{V},\mathbf{Y}) = \sum_{j=1}^{n} \text{tr}\, [(\tilde{\mathbf{X}}_j - \mathbf{1g}_j')\mathbf{T}_j - \mathbf{V}_j\mathbf{Y}]'\mathbf{M}_j[(\tilde{\mathbf{X}}_j - \mathbf{1g}_j')\mathbf{T}_j - \mathbf{V}_j\mathbf{Y}], \tag{4.8}$$

corresponding to the STIMFREE model (4.5) are determined. In (4.8) it is assumed that the unknown translation vectors $\mathbf{g}_j$ are collected in the (m x n) matrix $\mathbf{G}$, that the unknown orthonormal matrices $\mathbf{T}_j$ are collected in the (nm x m) supermatrix $\mathbf{T}$, and that the unknown diagonal matrices $\mathbf{V}_j$ are collected in the (np x p) supermatrix $\mathbf{V}$. In section 4.3.2 we first determine the translation vectors $\mathbf{g}_j$ minimizing (4.8), and show how to eliminate them from the STIMFREE loss function. In sections 4.3.3, 4.3.4, and 4.3.5 the simplified loss function is optimized with respect to the orthonormal matrices $\mathbf{T}_j$, the stimulus weights $\mathbf{V}_j$, and the centroid configuration $\mathbf{Y}$, respectively. In section 4.3.6 a direct approach is set up for the STIMFREE model, yielding a more efficient procedure for the determination of the orthonormal matrices $\mathbf{T}_j$. The uniqueness properties of the STIMFREE model are discussed in section 4.3.7, and section 4.3.8 contains an algorithm for the fitting of this model on n configurations.

4.3.2 Translations

To minimize (4.8) with respect to the unrestricted translation vectors $\mathbf{G}$ for fixed $\mathbf{T}$, $\mathbf{V}$, and $\mathbf{Y}$, let $\mathbf{A}_j = \tilde{\mathbf{X}}_j\mathbf{T}_j - \mathbf{V}_j\mathbf{Y}$, consider only one particular translation vector $\mathbf{g}_j$, and rewrite the loss function as

$$f(\mathbf{g}_j) = c_j^2\mathbf{g}_j'\mathbf{g}_j - 2\, c_j\mathbf{g}_j'\mathbf{r}_j + d_j, \tag{4.9}$$

where d_j is independent of $\mathbf{g}_j$, $\mathbf{r}_j \equiv \mathbf{T}_j\mathbf{A}_j'\mathbf{M}_j\mathbf{1}/\sqrt{\mathbf{1}'\mathbf{M}_j\mathbf{1}}$, and $c_j \equiv \sqrt{\mathbf{1}'\mathbf{M}_j\mathbf{1}}$. The solution for (4.9) has been discussed previously (see, e.g., section 2.3.1), and is given by

$$\mathbf{g}_j = c_j^{-1}\mathbf{r}_j. \tag{4.10}$$

Re-expressing (4.10) in the original matrices and vectors yields

$$\mathbf{g}_j = \frac{(\tilde{\mathbf{X}}_j - \mathbf{V}_j\mathbf{Y}\mathbf{T}_j')'\mathbf{M}_j\mathbf{1}}{\mathbf{1}'\mathbf{M}_j\mathbf{1}}, \tag{4.11}$$

and substitution of (4.11) in (4.8) gives

$$f(\mathbf{T},\mathbf{V},\mathbf{Y}) = \sum_{j=1}^{n} \operatorname{tr} (\tilde{\mathbf{X}}_j\mathbf{T}_j - \mathbf{V}_j\mathbf{Y})'\mathbf{C}_j(\tilde{\mathbf{X}}_j\mathbf{T}_j - \mathbf{V}_j\mathbf{Y}), \tag{4.12}$$

where $\mathbf{C}_j \equiv \mathbf{M}_j(\mathbf{I} - \mathbf{1}\mathbf{1}'\mathbf{M}_j/\mathbf{1}'\mathbf{M}_j\mathbf{1})$. After elimination of the $\mathbf{g}_j$ from the loss function, the STIMFREE model (4.5) can be simplified to

$$\mathbf{C}_j\tilde{\mathbf{X}}_j\mathbf{T}_j = \mathbf{C}_j\mathbf{V}_j\mathbf{Y} + \mathbf{C}_j\mathbf{E}_j, \qquad \text{for } j = 1, 2, ..., n. \tag{4.13}$$

Since $\mathbf{C}_j\mathbf{1} = \mathbf{0}$ it is also true that

$$\mathbf{C}_j\tilde{\mathbf{X}}_j = s_j\mathbf{C}_j(\mathbf{X}_j - \mathbf{1}\mathbf{u}_j')\mathbf{R}_j\mathbf{K} = s_j\mathbf{C}_j\mathbf{X}_j\mathbf{R}_j\mathbf{K} - s_j\mathbf{C}_j\mathbf{1}\mathbf{u}_j'\mathbf{R}_j\mathbf{K} = s_j\mathbf{C}_j\mathbf{X}_j\mathbf{R}_j\mathbf{K},$$

and we therefore define $\tilde{\mathbf{X}}_j = s_j\mathbf{X}_j\mathbf{R}_j\mathbf{K}$ throughout the rest of section 4.3.

4.3.3 Orthonormal transformations

To determine the matrix $\mathbf{T}$ minimizing (4.12) for fixed $\mathbf{V}$ and $\mathbf{Y}$, define $\mathbf{A}_j = \mathbf{V}_j\mathbf{Y}$, and rewrite (4.12) as

$$f(\mathbf{T}) = \sum_{j=1}^{n} \operatorname{tr} (\tilde{\mathbf{X}}_j\mathbf{T}_j - \mathbf{A}_j)'\mathbf{C}_j(\tilde{\mathbf{X}}_j\mathbf{T}_j - \mathbf{A}_j). \tag{4.14}$$

We refer to section 3.3.3.of Chapter 3 for a detailed discussion of the solution of (4.14) with respect to the unknown orthonormal matrices $\mathbf{T}_j$. This solution is analytic and guarantees to yield the global minimum of (4.14).

4.3.4 Stimulus weights

For fixed orthonormal matrices **T** and centroid configuration **Y**, the problem discussed in this section is how to minimize

$$f(\mathbf{V}) = \sum_{j=1}^{n} \text{tr}\,(\mathbf{A}_j - \mathbf{C}_j\mathbf{V}_j\mathbf{Y})'(\mathbf{A}_j - \mathbf{C}_j\mathbf{V}_j\mathbf{Y}) \tag{4.15}$$

where $\mathbf{A}_j \equiv \mathbf{C}_j\tilde{\mathbf{X}}_j\mathbf{T}_j$. Only considering one $\mathbf{V}_j$, and expanding (4.15) gives

$$f(\mathbf{V}_j) = d_j + \text{tr}\,\mathbf{Y}'\mathbf{V}_j\mathbf{C}_j\mathbf{V}_j\mathbf{Y} - 2\,\text{tr}\,\mathbf{A}_j'\mathbf{C}_j\mathbf{V}_j\mathbf{Y}, \tag{4.16}$$

where d_j is a constant with respect to $\mathbf{V}_j$. Let $\mathbf{b}_j = (\text{diag}\,\mathbf{Y}\mathbf{A}_j')\mathbf{1}$, that is, let $\mathbf{b}_j$ be the (p x 1) vector containing the diagonal elements of $\mathbf{Y}\mathbf{A}_j'$. Let $\mathbf{v}_j = \mathbf{V}_j\mathbf{1}$, that is, let $\mathbf{v}_j$ be the (p x 1) vector consisting of the p unknown stimulus weights. This makes it possible to write (4.16) as

$$f(\mathbf{v}_j) = d_j + \mathbf{v}_j'(\mathbf{C}_j \odot \mathbf{Y}\mathbf{Y}')\mathbf{v}_j - 2\,\mathbf{b}_j'\mathbf{v}_j, \tag{4.17}$$

where $\odot$ denotes the Hadamard product. But, if configuration $\mathbf{X}_j$ has missing rows, then the corresponding rows *and* columns of $(\mathbf{C}_j \odot \mathbf{Y}\mathbf{Y}')$ contain zero elements only, and the corresponding elements of $\mathbf{b}_j$ are also zero. If we let p_j be the number of non-missing rows of $\mathbf{X}_j$, this means that we only need to determine p_j stimulus weights, because the values of the remaining $(p - p_j)$ elements of $\mathbf{v}_j$ in (4.17) are irrelevant for the value of the loss function. Therefore, letting $\mathbf{D}_j$ be the $(p_j \text{ x } p_j)$ matrix only containing the non-zero elements of $(\mathbf{C}_j \odot \mathbf{Y}\mathbf{Y}')$, $\mathbf{b}_j^*$ be the $(p_j \text{ x } 1)$ vector of non-zero elements of $\mathbf{b}_j$, and $\mathbf{v}_j^*$ be the $(p_j \text{ x } 1)$ vector of corresponding elements of $\mathbf{v}_j$, loss function (4.17) may be rewritten as

$$f(\mathbf{v}_j^*) = d_j + \mathbf{v}_j^{*\prime}\mathbf{D}_j\mathbf{v}_j^* - 2\,\mathbf{b}_j^{*\prime}\mathbf{v}_j^* = d_j + e_j + \|\, \mathbf{D}_j^{1/2}\mathbf{v}_j^* - \mathbf{D}_j^{-1/2}\mathbf{b}_j^* \,\|^2, \tag{4.18}$$

where e_j is, again, a term independent of $\mathbf{v}_j^*$. It follows that the global minimum of (4.18) is attained for

$$\mathbf{v}_j^* = \mathbf{D}_j^{-1}\mathbf{b}_j^*. \tag{4.19}$$

By replacing the appropriate elements of $\mathbf{v}_j$ of order (p x 1) by the elements of $\mathbf{v}_j^*$ of order $(p_j \text{ x } 1)$ given by (4.19), and by setting the remaining $(p - p_j)$ elements of $\mathbf{v}_j$

equal to some arbitrary number (zero, for instance), the global minimum of (4.17) with respect to $\mathbf{v}_j$ is attained. Finally, given the latter optimal $\mathbf{v}_j$, the global minimum of (4.16) is found where

$$\mathbf{V}_j = \mathbf{I}_p \odot (\mathbf{v}_j \mathbf{1}'). \tag{4.20}$$

The computation of (4.20) for j = 1, ..., n is guaranteed to yield the global minimum of (4.15) with respect to $\mathbf{V}$.

A disadvantage of the calculation of the stimulus weights using (4.19) is that it may result in negative weights. As we discussed in section 4.2, negative stimulus weights do not have a meaningful interpretation. Therefore, we use the nonnegative least squares (NNLS) algorithm of Lawson and Hanson (1974) to determine the global minimum of (4.18) subject to the linear constraint $\mathbf{v}_j^* \geq 0$. For a detailed description of the NNLS algorithm we refer to Lawson and Hanson (1974, p.158 - 165).

4.3.5 The centroid configuration Y

For fixed $\mathbf{T}$ and $\mathbf{V}$, we want to find the minimum of

$$f(\mathbf{Y}) = \sum_{j=1}^{n} \operatorname{tr} (\mathbf{A}_j - \mathbf{B}_j\mathbf{Y})'\mathbf{C}_j(\mathbf{A}_j - \mathbf{B}_j\mathbf{Y}) \tag{4.21}$$

where $\mathbf{A}_j \equiv \mathbf{C}_j\tilde{\mathbf{X}}_j\mathbf{T}_j$ and $\mathbf{B}_j \equiv \mathbf{C}_j\mathbf{V}_j$. Collecting the n matrices $\mathbf{A}_j$ in the supermatrix $\mathbf{A}$ of order (np x m), and collecting the n matrices $\mathbf{B}_j$ in the supermatrix $\mathbf{B}$ of order (np x p), (4.21) may be written as

$$f(\mathbf{Y}) = \operatorname{tr} (\mathbf{A} - \mathbf{B}\mathbf{Y})'(\mathbf{A} - \mathbf{B}\mathbf{Y}). \tag{4.22}$$

This is the classical multivariate multiple regression problem with the well-known solution $\mathbf{Y} = (\mathbf{B}'\mathbf{B})^{-1}\mathbf{B}'\mathbf{A}$, which can be re-expressed in terms of the original matrices as

$$\mathbf{Y} = (\sum_{j=1}^{n} \mathbf{V}_j\mathbf{C}_j\mathbf{V}_j)^{-1}(\sum_{j=1}^{n} \mathbf{V}_j\mathbf{C}_j\mathbf{A}_j). \tag{4.23}$$

The sum of the matrix products $\mathbf{V}_j\mathbf{C}_j\mathbf{V}_j$ yields a matrix of full rank for which a proper inverse exists.

In general the optimal $\mathbf{Y}$ in (4.23) is *not* a column centered matrix, in contrast with the centroid configurations found in the GPA and in the dimension weighting models. This is not surprising since, as we already noted in section 4.2, the location of $\mathbf{Y}$ is essential once stimulus weights are taken into account.

4.3.6 The direct approach in STIMFREE

In this section we show that the centroid configuration $\mathbf{Y}$ can be eliminated from (4.12). This yields a loss function which makes it possible to set up a more efficient procedure for the determination of the $\mathbf{T}_j$'s than the one given in section 4.3.3.

Since the minimization of (4.22) is a multiple regression problem, given optimal regression weights $\mathbf{Y}$, the space spanned by the predicted $\mathbf{BY}$ is orthogonal to the space spanned by the residual matrix $(\mathbf{A} - \mathbf{BY})$. At the global minimum of (4.22) it is therefore true that

$$f(\mathbf{Y}) = \text{tr}\,(\mathbf{A} - \mathbf{BY})'(\mathbf{A} - \mathbf{BY}) = \text{tr}\,\mathbf{A}'(\mathbf{A} - \mathbf{BY}), \tag{4.24}$$

where the matrices $\mathbf{A}$ and $\mathbf{B}$ are defined as in section 4.3.5. Substitution of $\mathbf{Y} = (\mathbf{B}'\mathbf{B})^{-1}\mathbf{B}'\mathbf{A}$ in (4.24) yields

$$f((\mathbf{B}'\mathbf{B})^{-1}\mathbf{B}'\mathbf{A}) = \text{tr}\,\mathbf{A}'\mathbf{A} - \text{tr}\,\mathbf{A}'\mathbf{B}(\mathbf{B}'\mathbf{B})^{-1}\mathbf{B}'\mathbf{A},$$

which can be re-expressed in terms of the original matrices as

$$f(\mathbf{T},\mathbf{V}) = \sum_{j=1}^{n} \text{tr}\,\tilde{\mathbf{X}}'_j\mathbf{C}_j\tilde{\mathbf{X}}_j - \text{tr}\,(\sum_{j=1}^{n}\mathbf{V}_j\mathbf{C}_j\tilde{\mathbf{X}}_j\mathbf{T}_j)'(\sum_{j=1}^{n}\mathbf{V}_j\mathbf{C}_j\mathbf{V}_j)^{-1}(\sum_{j=1}^{n}\mathbf{V}_j\mathbf{C}_j\tilde{\mathbf{X}}_j\mathbf{T}_j). \tag{4.25}$$

The minimization of (4.25) is equivalent to the maximization of

$$g(\mathbf{T},\mathbf{V}) = \text{tr}\,(\sum_{j=1}^{n}\mathbf{V}_j\mathbf{C}_j\tilde{\mathbf{X}}_j\mathbf{T}_j)'(\sum_{j=1}^{n}\mathbf{V}_j\mathbf{C}_j\mathbf{V}_j)^{-1}(\sum_{j=1}^{n}\mathbf{V}_j\mathbf{C}_j\tilde{\mathbf{X}}_j\mathbf{T}_j) \tag{4.26}$$

under the constraint that $\mathbf{T}_j'\mathbf{T}_j = \mathbf{T}_j\mathbf{T}_j' = \mathbf{I}_m$ for each j.

It does not seem possible to determine the $\mathbf{V}_j$'s maximizing (4.26) for fixed $\mathbf{T}$. On the other hand, to maximize (4.26) with respect to $\mathbf{T}$ for fixed $\mathbf{V}$, define $\mathbf{D}_j = \mathbf{V}_j\mathbf{C}_j\tilde{\mathbf{X}}_j$, and $\mathbf{F} = (\sum_{j=1}^{n}\mathbf{V}_j\mathbf{C}_j\mathbf{V}_j)^{-1}$. This allows us to rewrite (4.26) as

$$g(\mathbf{T}) = \text{tr}\,(\sum_{j=1}^{n} \mathbf{D}_j\mathbf{T}_j)'\mathbf{F}(\sum_{j=1}^{n} \mathbf{D}_j\mathbf{T}_j), \tag{4.27}$$

and, if we only consider one $\mathbf{T}_j$, as

$$g(\mathbf{T}_j) = \text{tr}\,\mathbf{T}_j'\mathbf{D}_j'\mathbf{F}\mathbf{D}_j\mathbf{T}_j + 2\,\text{tr}\,\mathbf{T}_j'\mathbf{D}_j'\mathbf{F}(\sum_{i\neq j} \mathbf{D}_i\mathbf{T}_i) + d_j, \tag{4.28}$$

where d_j is a term independent from $\mathbf{T}_j$. Therefore, the maximization of (4.28) is equivalent to the maximization of

$$h(\mathbf{T}_j) = \text{tr}\,\mathbf{T}_j'\mathbf{D}_j'\mathbf{F}(\sum_{i\neq j} \mathbf{D}_i\mathbf{T}_i) \tag{4.29}$$

subject to $\mathbf{T}_j'\mathbf{T}_j = \mathbf{T}_j\mathbf{T}_j' = \mathbf{I}_m$. This is the orthonormal Procrustes problem. We refer to section 2.3.4 of Chapter 2 for a detailed discussion of the solution of (4.29).

Since there is no analytical solution for **T** in (4.27), an alternative least squares algorithm must be used where (4.27) is consecutively maximized for j = 1, 2, ..., n and **T** is updated after each step. This process must be repeated until n steps jointly fail to raise (4.27) above some threshold value. We have found that this procedure for determining orthonormal matrices $\mathbf{T}_j$ results into a more efficient algorithm for the estimation of the unknown parameters in the STIMFREE model than the procedure discussed in section 4.3.3. That is, while both procedures require about an identical amount of CPU-time within one iteration, convergence of the algorithm is considerably faster when using the updating procedure for the $\mathbf{T}_j$'s described in this section.

4.3.7 Uniqueness of the STIMFREE solution

The solution for the STIMFREE model belongs to the following family of equivalent solutions:

$$f(\mathbf{T},\mathbf{V},\mathbf{Y}) = \sum_{j=1}^{n} \text{tr}\,(\tilde{\mathbf{X}}_j\mathbf{T}_j\mathbf{R} - \mathbf{V}_j\mathbf{K}\mathbf{K}^{-1}\mathbf{Y}\mathbf{R})'\mathbf{C}_j(\tilde{\mathbf{X}}_j\mathbf{T}_j\mathbf{R} - \mathbf{V}_j\mathbf{K}\mathbf{K}^{-1}\mathbf{Y}\mathbf{R}), \tag{4.30}$$

where **R** is an arbitrary (m x m) orthonormal matrix, and **K** is an arbitrary diagonal matrix of order (m x m). In words, (4.30) expresses that the solution in STIMFREE is unique up to a simultaneous rotation by **R** of the whole solution, and, more importantly, up to a weighting **K** of the optimal stimulus weights $\mathbf{V}_j$ together with an inverse weighting of the optimal centroid **Y**.

The rotational indeterminacy of the STIMFREE model can be used to rotate the solution to its principal components. This point is further discussed in section 4.5. The freedom we have in choosing a particular weighting matrix **K** allows us to choose one yielding a uniquely weighted solution. This is achieved by setting **K** equal to

$$\mathbf{G} = (\text{diag } \mathbf{YY}')^{1/2}, \tag{4.31}$$

which yields a weighted centroid configuration

$$\mathbf{Y}^* = \mathbf{G}^{-1}\mathbf{Y} = (\text{diag } \mathbf{YY}')^{-1/2}\mathbf{Y}, \tag{4.32}$$

with unique stimulus weights

$$\mathbf{V}_j^* = \mathbf{V}_j\mathbf{G} = \mathbf{V}_j(\text{diag } \mathbf{YY}')^{1/2}, \qquad \text{for } j = 1, 2, \ldots, n. \tag{4.33}$$

Since premultiplying **Y** with $(\text{diag } \mathbf{YY}')^{-1/2}$ has the effect of unit normalizing the rows of **Y**, the matrix product $\mathbf{Y}^*\mathbf{Y}^{*\prime}$ contains the cosines of the angles between the stimulus points of **Y** conceived of as vectors connected with the origin.

Except that the unit normalization of the rows of **Y** results in a uniquely weighted solution, it also has another advantage. In the original model (4.3) of Lingoes and Borg (see section 4.2), the effect of a stimulus weight is conditional upon the length of the vector in ($\mathbf{Z} - \mathbf{h}$) that is being weighted. If a stimulus vector in ($\mathbf{Z} - \mathbf{h}$) has a length of 0.08, for example, weighting this vector with a factor 10 has the effect of moving the end point of the vector only (10)(0.08) – 0.08 = 0.72 units further away from the origin. But a much smaller weight of 1.144 will do the same job for a vector of length 5.00, since (1.144)(5.00) – 5.00 = 0.72 units also. In general, therefore, information about the value of a weight is not, in itself, enough to assess the effect or impact of the weight. This interdependence between weight effect and vector length completely vanishes for the unique stimulus weights $\mathbf{V}_j^*$ given in (4.33). Since the lengths of the stimulus vectors in $\mathbf{Y}^*$ are all equal to one, the impact of the unique weights can be assessed unambiguously in the STIMFREE model. Below, we will propose a special 'impact index' for the STIMFREE stimulus weights. First, however, another problem concerning the stimulus weights must be discussed.

Although the stimulus weights calculated in (4.33) are unique, they are, in general, not comparable over configurations. This is due to the fact that the n configurations $\tilde{\mathbf{X}}_j$ will usually have different sizes (i.e., the values of tr $\tilde{\mathbf{X}}'_j\mathbf{C}_j\tilde{\mathbf{X}}_j$ will be different for j = 1, ..., n). As a consequence, the values of the stimulus weights corresponding to a 'small' configuration will be relatively larger than those corresponding to a 'large' configuration. This problem is easily remedied, however, by calculating

$$\mathbf{V}^{**}_j = \mathbf{V}^*_j/(\mathrm{tr}\,\tilde{\mathbf{X}}'_j\mathbf{C}_j\tilde{\mathbf{X}}_j)^{1/2} \tag{4.34}$$

for each j, since in that case idiosyncracies in stimulus weighting due to differences in size of the configurations disappear.

As promised, we end this section by presenting an 'impact index' for the stimulus weights in (4.34). This index makes it particularly easy to assess the amount of displacement caused by a stimulus weight. The impact index d_{ij} is defined as follows:

$$d_{ij} = |\,1 - v_{ij}\,|. \tag{4.35}$$

In (4.35), v_{ij} denotes the stimulus weight calculated in (4.33) for stimulus i corresponding to configuration j, and d_{ij} denotes the impact index for v_{ij}. The definition covers positive as well as negative stimulus weights. The impact index is always non-negative. In words, the index d_{ij} is equal to the distance between the two end points of stimulus vector i of $\mathbf{Y}^*$ before and after stimulus weighting with v_{ij}. Therefore, an index of zero indicates that nothing changed after stimulus weighting. The larger the index, the further the end point of the vector has been moved away from its original place by the stimulus weight.

The geometry of the impact index is illustrated in Figure 4.4, where the black arrow represents a unit normalized vector of $\mathbf{Y}^*$. Of the three examples given in the figure, the smallest impact or displacement is achieved by a stimulus weight of 1.5 (middle of the figure), since the displacement of the end point of the vector is only 0.5. A larger displacement of the end point of the stimulus vector of $\mathbf{Y}^*$ is obtained for a stimulus weight of 0.3: the value of the corresponding impact index is 0.7 (left of the figure). The largest impact shown in the figure corresponds to a negative stimulus weight of -0.4.

The impact index can be used to test for differences between configurations or stimuli with standard linear statistical techniques. In this way homogeneous groups of

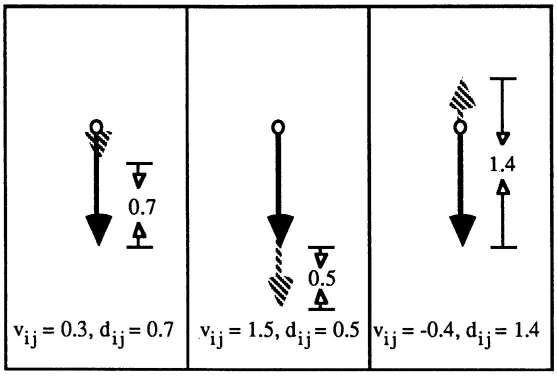

Figure 4.4 Three illustrations of relation between unique stimulus weight and impact index.

stimuli may be formed, for instance, yielding a better fit when analyzed separately with the more parsimonious GPA and dimension weighting models.

4.3.8 The algorithm

In the MATCHALS program the algorithm for the estimation of the unknown parameters in the STIMFREE model consists of the following steps.

a) Before starting to iterate, initialize each $\mathbf{T}_j$ on $\mathbf{T}_j = \mathbf{I}_m$, and $\mathbf{Y}$ on $\mathbf{Y} = \mathbf{ZK}$ (i.e., the optimal centroid configuration found in GPA, rotated to the principal components of the GPA solution).

b) For every j, compute new stimulus weights using the procedure given in section 4.3.4. By default the MATCHALS program computes nonnegative weights.

c) Determine the proper inverse of (Σ $\mathbf{V}_j\mathbf{C}_j\mathbf{V}_j$), and calculate a new $\mathbf{T}_j$ for each j using the procedure described in section 4.3.6, where $\mathbf{T}$ must be updated each time a new $\mathbf{T}_j$ has been estimated.

d) Evaluate loss function (4.26). If the difference between the function value in this iteration and in the previous iteration is smaller than some convergence criterion, go to step e). Otherwise, calculate a new centroid configuration **Y** with (4.23) using the proper inverse of (Σ $\mathbf{V}_j\mathbf{C}_j\mathbf{V}_j$) computed in step c), and go to step b).

e) Print the history of iterations and **Y**. Compute the normalized and uniquely weighted group configuration $\mathbf{Y}^*$ and the normalized and uniquely weighted stimulus weights $\mathbf{V}_j^*$ according to (4.32) and (4.33), respectively, and print $\mathbf{Y}^*$ and $\mathbf{V}_j^*/(\text{tr } \tilde{\mathbf{X}}'_j\mathbf{C}_j\tilde{\mathbf{X}}_j)^{1/2}$ for each j. Finally, compute the impact index for each stimulus weight using (4.35), and print this also.

The solution calculated in step e) of the algorithm for STIMFREE is unique up to a simultaneous rotation of the complete solution. In section 4.5.2 we show how this last indeterminacy can be used to rotate the solution to its principal components.

Unfortunately, this algorithm often converges to a local minimum. We observed this annoying property when analyzing data constructed in such a way that a perfect solution was known to exist: the algorithm often converged to a solution that, although leaving only a small residual sum of squares, was not the global minimum of zero loss. In section 4.6 examples of this situation are given.

As yet we can only speculate about the possible causes of this problem. Different starting values for **Y** in step a) of the algorithm could be tried to investigate the stability of the solution. Instead of using **ZK** from GPA as initial estimate for **Y**, for instance, one could start the STIMFREE algorithm with the centroid configuration obtained in DIMFREE or DIMIDIO. Also, the examples discussed in section 4.6 consist of only four configurations containing the coordinates of eight stimuli. We have not investigated whether better results are obtained when analyzing larger numbers of configurations and stimuli.

4.4 The STIMIDIO model

4.4.1 Introduction

The least squares loss function corresponding to the STIMIDIO model (4.7) is

$$f(\mathbf{G},\mathbf{H},\mathbf{T},\mathbf{V},\mathbf{Y}) = \sum_{j=1}^{n} \| \mathbf{M}_j[(\tilde{\mathbf{X}}_j - \mathbf{1g}_j')\mathbf{T}_j - \mathbf{V}_j(\mathbf{Y} - \mathbf{1h}_j')] \|^2. \tag{4.36}$$

In (4.36) it is assumed that the unknown translation vectors $\mathbf{g}_j$ and $\mathbf{h}_j$ are collected in the (m x n) matrices **G** and **H**, respectively. The unknown orthonormal matrices $\mathbf{T}_j$ are assumed to be collected in the (nm x m) supermatrix **T**, and the unknown diagonal matrices $\mathbf{V}_j$ in the (np x p) supermatrix **V**.

In section 4.4.2 we show how to eliminate the translation vectors **G** from loss function (4.36). In sections 4.4.3, 4.4.4, 4.4.5, and 4.4.6 the simplified loss function is optimized with respect to the orthonormal matrices $\mathbf{T}_j$, the stimulus weights $\mathbf{V}_j$, the translation vectors $\mathbf{h}_j$, and the centroid configuration **Y**, respectively. In section 4.4.7 the uniqueness properties of the STIMIDIO model are investigated, and in section 4.4.8 an algorithm is presented for the estimation of the model parameters.

4.4.2 Translations of the $\mathbf{X}_j$

To minimize (4.36) with respect to **G** for fixed **H**, **V**, **T**, and **Y**, let $\mathbf{A}_j = \tilde{\mathbf{X}}_j\mathbf{T}_j - \mathbf{V}_j(\mathbf{Y} - \mathbf{1h}_j')$, and consider only one particular translation vector $\mathbf{g}_j$ to rewrite the loss function as

$$f(\mathbf{g}_j) = d_j + c_j^2\mathbf{g}_j'\mathbf{g}_j - 2\, c_j\mathbf{g}_j'\mathbf{r}_j, \tag{4.37}$$

where d_j is independent of $\mathbf{g}_j$, $\mathbf{r}_j \equiv \mathbf{T}_j\mathbf{A}_j'\mathbf{M}_j\mathbf{1}/\sqrt{\mathbf{1}'\mathbf{M}_j\mathbf{1}}$, and $c_j \equiv \sqrt{\mathbf{1}'\mathbf{M}_j\mathbf{1}}$. The solution for (4.37) has already been discussed several times (see, e.g., section 2.3.1), and is given by

$$\mathbf{g}_j = c_j^{-1}\mathbf{r}_j = \frac{[\tilde{\mathbf{X}}_j - \mathbf{V}_j(\mathbf{Y} - \mathbf{1h}_j')\mathbf{T}_j']'\mathbf{M}_j\mathbf{1}}{\mathbf{1}'\mathbf{M}_j\mathbf{1}}. \tag{4.38}$$

Substitution of (4.38) in (4.36) gives

$$f(\mathbf{H},\mathbf{T},\mathbf{V},\mathbf{Y}) = \sum_{j=1}^{n} \text{tr}\, [\tilde{\mathbf{X}}_j\mathbf{T}_j - \mathbf{V}_j(\mathbf{Y} - \mathbf{1h}_j')]'\mathbf{C}_j[\tilde{\mathbf{X}}_j\mathbf{T}_j - \mathbf{V}_j(\mathbf{Y} - \mathbf{1h}_j')] \tag{4.39}$$

with $\mathbf{C}_j \equiv \mathbf{M}_j(\mathbf{I} - \mathbf{11}'\mathbf{M}_j/\mathbf{1}'\mathbf{M}_j\mathbf{1})$. Note that, unlike in the dimension weighting models, substitution of (4.38) in (4.36) does not result in the simultaneous elimination of the translation vectors $\mathbf{H}$. After the elimination of $\mathbf{G}$ from (4.36) the STIMIDIO model (4.7) can be simplified to

$$\mathbf{C}_j\tilde{\mathbf{X}}_j\mathbf{T}_j = \mathbf{C}_j\mathbf{V}_j(\mathbf{Y} - \mathbf{1h}_j') + \mathbf{C}_j\mathbf{E}_j, \qquad \text{for } j = 1, 2, ..., n. \tag{4.40}$$

Since $\mathbf{C}_j\mathbf{1} = \mathbf{0}$, and therefore

$$\mathbf{C}_j\tilde{\mathbf{X}}_j = s_j\mathbf{C}_j(\mathbf{X}_j - \mathbf{1u}_j')\mathbf{R}_j\mathbf{K} = s_j\mathbf{C}_j\mathbf{X}_j\mathbf{R}_j\mathbf{K},$$

we define $\tilde{\mathbf{X}}_j = s_j\mathbf{X}_j\mathbf{R}_j\mathbf{K}$ throughout the remaining of Chapter 4.

4.4.3 Orthonormal transformations

To determine the unknown orthonormal matrices $\mathbf{T}_j$ minimizing (4.39) for fixed $\mathbf{H}$, $\mathbf{V}$, and $\mathbf{Y}$, we define $\mathbf{A}_j = \mathbf{V}_j(\mathbf{Y} - \mathbf{1h}_j')$ and rewrite (4.39) as

$$f(\mathbf{T}) = \sum_{j=1}^{n} \text{tr}\, (\tilde{\mathbf{X}}_j\mathbf{T}_j - \mathbf{A}_j)'\mathbf{C}_j(\tilde{\mathbf{X}}_j\mathbf{T}_j - \mathbf{A}_j). \tag{4.41}$$

This problem has already been discussed in section 3.3.3 of Chapter 3. Therefore, we refer to the latter section for a detailed description of the solution of this problem. This solution is guaranteed to yield the global minimum of (4.41).

4.4.4 Stimulus weights

For fixed **H**, **T**, and **Y**, the problem is how to minimize

$$f(\mathbf{V}) = \sum_{j=1}^{n} \text{tr}\,(\mathbf{A}_j - \mathbf{C}_j\mathbf{V}_j\mathbf{B}_j)'(\mathbf{A}_j - \mathbf{C}_j\mathbf{V}_j\mathbf{B}_j), \tag{4.42}$$

where $\mathbf{A}_j \equiv \mathbf{C}_j\tilde{\mathbf{X}}_j\mathbf{T}_j$ and $\mathbf{B}_j \equiv \mathbf{Y} - \mathbf{1h}_j'$. This problem is again identical to the problem discussed in section 4.3.4. Let p_j be the number of non-missing rows of $\mathbf{X}_j$. Define $\mathbf{D}_j$ as the (p_j x p_j) matrix containing the rows *and* columns of $(\mathbf{C}_j \odot \mathbf{B}_j\mathbf{B}_j')$ corresponding to the non-missing rows of $\mathbf{X}_j$. Also, define $\mathbf{b}_j^*$ as the (p_j x 1) vector consisting of the elements of $\mathbf{b}_j \equiv (\text{diag}\,\mathbf{B}_j\mathbf{A}_j')\mathbf{1}$ corresponding to the non-missing rows of $\mathbf{X}_j$, and $\mathbf{v}_j^*$ as the (p_j x 1) vector containing the corresponding stimulus weights of $\mathbf{v}_j \equiv \mathbf{V}_j\mathbf{1}$. Then, if we consider only one $\mathbf{V}_j$, function (4.42) may be written as

$$f(\mathbf{v}_j^*) = d_j + \mathbf{v}_j^{*\prime}\mathbf{D}_j\mathbf{v}_j^* - 2\,\mathbf{b}_j^{*\prime}\mathbf{v}_j^* = d_j + e_j + \|\, \mathbf{D}_j^{1/2}\mathbf{v}_j^* - \mathbf{D}_j^{-1/2}\mathbf{b}_j^* \,\|^2, \tag{4.43}$$

where d_j and e_j are terms independent of $\mathbf{v}_j^*$. If the stimulus weights are not restricted, the global minimum of (4.43) is attained for

$$\mathbf{v}_j^* = \mathbf{D}_j^{-1}\mathbf{b}_j^*. \tag{4.44}$$

Inserting these p_j optimal weights in the appropriate places in the vector $\mathbf{v}_j$, setting the remaining ($p - p_j$) stimulus weights equal to some arbitrary value, and then calculating $\mathbf{V}_j$ as

$$\mathbf{V}_j = \mathbf{I}_p \odot (\mathbf{v}_j\mathbf{1}'), \tag{4.45}$$

yields the global minimum of (4.42) with respect to one particular $\mathbf{V}_j$. Calculating (4.44) and (4.45) for j = 1, ..., n is guaranteed to give the global minimum of (4.42) with respect to **V**.

The nonnegative least squares algorithm of Lawson and Hanson (1974) can be used to determine the global minimum of (4.43) subject to $\mathbf{v}_j^* \geq 0$.

4.4.5 Idiosyncratic translations of Y

Defining $\mathbf{A}_j = \tilde{\mathbf{X}}_j\mathbf{T}_j - \mathbf{V}_j\mathbf{Y}$, and considering only one $\mathbf{h}_j$, the problem for fixed $\mathbf{T}$, $\mathbf{V}$, and $\mathbf{Y}$ is how to minimize

$$f(\mathbf{h}_j) = \sum_{j=1}^{n} \text{tr}\,(\mathbf{A}_j + \mathbf{V}_j\mathbf{1}\mathbf{h}_j')'\mathbf{C}_j(\mathbf{A}_j + \mathbf{V}_j\mathbf{1}\mathbf{h}_j')$$

$$= c_j^2\mathbf{h}_j'\mathbf{h}_j + 2\,c_j\mathbf{h}_j'\mathbf{r}_j + d_j, \qquad (4.46)$$

where d_j is a term independent of $\mathbf{h}_j$, $\mathbf{r}_j \equiv (\mathbf{A}_j'\mathbf{C}_j\mathbf{V}_j\mathbf{1})/\sqrt{(\mathbf{1}'\mathbf{V}_j\mathbf{C}_j\mathbf{V}_j\mathbf{1})}$, and $c_j \equiv \sqrt{(\mathbf{1}'\mathbf{V}_j\mathbf{C}_j\mathbf{V}_j\mathbf{1})}$. The global minimum of (4.46) is attained for

$$\mathbf{h}_j = -c_j^{-1}\mathbf{r}_j = \frac{(\mathbf{V}_j\mathbf{Y} - \tilde{\mathbf{X}}_j\mathbf{T}_j)'\mathbf{C}_j\mathbf{V}_j\mathbf{1}}{\mathbf{1}'\mathbf{V}_j\mathbf{C}_j\mathbf{V}_j\mathbf{1}}. \qquad (4.47)$$

Computing (4.47) for j = 1, ..., n is guaranteed to give the global minimum of (4.39) with respect to $\mathbf{H}$ for fixed $\mathbf{T}$, $\mathbf{V}$, and $\mathbf{Y}$.

4.4.6 The centroid configuration Y

To minimize (4.39) with respect to $\mathbf{Y}$ for fixed $\mathbf{H}$, $\mathbf{T}$, and $\mathbf{V}$, define $\mathbf{A}_j = \tilde{\mathbf{X}}_j\mathbf{T}_j + \mathbf{V}_j\mathbf{1}\mathbf{h}_j'$, and rewrite (4.39) as

$$f(\mathbf{Y}) = \sum_{j=1}^{n} \text{tr}\,(\mathbf{A}_j - \mathbf{V}_j\mathbf{Y})'\mathbf{C}_j(\mathbf{A}_j - \mathbf{V}_j\mathbf{Y}). \qquad (4.48)$$

As has already been discussed in section 4.3.5, the global minimum of (4.48) is attained for

$$\mathbf{Y} = (\sum_{j=1}^{n} \mathbf{V}_j\mathbf{C}_j\mathbf{V}_j)^{-1}(\sum_{j=1}^{n} \mathbf{V}_j\mathbf{C}_j\mathbf{A}_j). \qquad (4.49)$$

Just as in STIMFREE, this centroid configuration will, in general, not be a column centered matrix.

4.4.7 Non-uniqueness of the STIMIDIO solution

Unfortunately, the STIMIDIO model is so full of indeterminacies that it is not possible to reduce these to a unique solution. We start by noting that the value of the STIMIDIO loss function is unchanged by the following two transformations:

$$f(\mathbf{H},\mathbf{T},\mathbf{V},\mathbf{Y}) = \sum_{j=1}^{n} \| \mathbf{C}_j\tilde{\mathbf{X}}_j\mathbf{T}_j\mathbf{P} - \mathbf{C}_j\mathbf{V}_j[(\mathbf{Y} - \mathbf{1d}') - (\mathbf{1h}_j' - \mathbf{1d}')]\mathbf{P} \|^2, \quad (4.50)$$

where $\mathbf{P}$ is an arbitrary orthonormal matrix of order (m x m), and $\mathbf{d}$ an arbitrary translation vector of order (m x 1). These transformations of the solution express that the whole solution may be simultaneously rotated, and that the centroid configuration $\mathbf{Y}$ may be translated arbitrarily if a reverse translation is applied to the translation vectors $\mathbf{h}_j$.

We could use the latter indeterminacy to translate $\mathbf{Y}$ to a unique location. An obvious choice would be to translate $\mathbf{Y}$ such that it becomes column centered. This is accomplished by taking

$$\mathbf{d} = \mathbf{Y}'\mathbf{1}/\mathbf{1}'\mathbf{1}, \quad (4.51)$$

since

$$(\mathbf{Y} - \mathbf{1d}')'\mathbf{1} = \mathbf{Y}'\mathbf{1} - \mathbf{d1}'\mathbf{1} = \mathbf{Y}'\mathbf{1} - \mathbf{Y}'\mathbf{11}'\mathbf{1}/\mathbf{1}'\mathbf{1} = \mathbf{0}, \quad (4.52)$$

yielding a relocated centroid configuration

$$\mathbf{Y}^* = \mathbf{Y} - \mathbf{1d}' = \mathbf{Y} - \mathbf{11}'\mathbf{Y}/\mathbf{1}'\mathbf{1}, \quad (4.53)$$

and relocated translation vectors

$$\mathbf{h}_j^* = \mathbf{h}_j - \mathbf{d} = \mathbf{h}_j - \mathbf{Y}'\mathbf{1}/\mathbf{1}'\mathbf{1}. \quad (4.54)$$

The rotational indeterminacy of the STIMIDIO solution could be used to rotate it to principal components.

However, this would not make the solution unique. Suppose that we analyze the same data set twice with STIMIDIO using two different initial estimates for the centroid $\mathbf{Y}$. Assume for the sake of simplicity that we obtain perfect solutions in both analyses, and that all configurations are complete. Let $\mathbf{V}_j(\mathbf{Y} - \mathbf{1h}_j')$ be the first STIMIDIO solution, and $\tilde{\mathbf{V}}_j(\tilde{\mathbf{Y}} - \mathbf{1}\tilde{\mathbf{h}}_j')$ the second solution for j = 1, ..., n. Letting $\mathbf{J}$ =

$(\mathbf{I}_p - \mathbf{11'/1'1})$, and since $\mathbf{C}_j = \mathbf{J}$ when a configuration is complete, it follows from (4.50) that the relation between the two solutions will be

$$\mathbf{J\tilde{X}}_j\mathbf{T}_j = \mathbf{JV}_j(\mathbf{Y} - \mathbf{1h}_j') = \mathbf{J\tilde{V}}_j(\mathbf{\tilde{Y}} - \mathbf{1\tilde{h}}_j')\mathbf{P}, \quad \text{for } j = 1, \ldots, n, \tag{4.55}$$

where $\mathbf{P}$ is an orthonormal matrix. Thus, the two solutions will be rotated versions of one another. Moreover, since matrix $\mathbf{J}$ has the effect of centering the matrix products in (4.55) on the origin, the matrix products $\mathbf{V}_j(\mathbf{Y} - \mathbf{1h}_j')$ and $\mathbf{\tilde{V}}_j(\mathbf{\tilde{Y}} - \mathbf{1\tilde{h}}_j')\mathbf{P}$ are free to be located anywhere in m-dimensional space. This means that another indeterminacy crops up in the STIMIDIO solution: dropping the centring matrices $\mathbf{J}$ in (4.55) yields

$$\mathbf{V}_j(\mathbf{Y} - \mathbf{1h}_j') = \mathbf{\tilde{V}}_j(\mathbf{\tilde{Y}} - \mathbf{1\tilde{h}}_j')\mathbf{P} - \mathbf{1k}', \quad \text{for } j = 1, \ldots, n, \tag{4.56}$$

where $\mathbf{k}$ is a translation vector of order (m x 1). Due to the indeterminacies in (4.56), it is impossible to obtain a unique centroid configuration $\mathbf{Y}$ and unique stimulus weights $\mathbf{V}_j$ in the STIMIDIO model. Therefore, the STIMIDIO solution is not unique.

In contrast with the STIMFREE model, the stimulus weights in the STIMIDIO model are not comparable over configurations. The reason is that the centroid configuration $\mathbf{Y}$ is translated differently for each configuration j. The optimal translation vectors $\mathbf{h}_j$, however, can be compared over configurations. Since these n translations can be plotted as points in m-dimensional space, the inspection of such a plot could be used to detect clusters of translation points (or 'points of view'), which may be related to some background information if the configurations represent subjects, for instance.

4.4.8 The algorithm

The algorithm corresponding to the STIMIDIO model consists of the following steps.

a) Before starting to iterate, initialize the matrices $\mathbf{T}_j$ on the optimal $\mathbf{T}_j$ found in STIMFREE, and $\mathbf{Y}$ on the optimal and uniquely weighted $\mathbf{Y}^*$ found in STIMFREE. Initialize the vectors $\mathbf{h}_j$ on $\mathbf{h}_j = \mathbf{0}$ for $j = 1, \ldots, n$.

b) For every j, compute new stimulus weights using the procedure given in section 4.4.4. By default the MATCHALS program uses the NNLS algorithm of Lawson and Hanson (1974) to restrict the weights to be nonnegative.
c) Compute new optimal translation vectors $\mathbf{h}_j$ with (4.47) for each j.
d) Determine new orthonormal matrices $\mathbf{T}_j$ in (4.41) using the procedure described in section 3.3.3 of Chapter 3.
e) Again compute new optimal translation vectors $\mathbf{h}_j$ with (4.47) for each j.
f) Compute a new centroid configuration $\mathbf{Y}$ using (4.49).
g) Evaluate loss function (4.39). If the difference between the function value in this iteration and in the previous iteration is smaller than some convergence criterion, go to step h). Otherwise go to step b).
h) Print the history of iterations. Calculate the translated $\mathbf{Y}^*$ and $\mathbf{h}_j^*$ according to (4.53) and (4.54), respectively, and print them. Also print the stimulus weights $\mathbf{V}_j$ for each j.

The reason that the translation vectors are calculated twice within one iteration (i.e., in steps c) and e) of the algorithm) is that, in comparison with the $\mathbf{V}_j$, $\mathbf{T}_j$, and $\mathbf{Y}$, the estimation of the $\mathbf{h}_j$ is relatively cheap in terms of computation time. The STIMIDIO algorithm has the same problem as the STIMFREE algorithm: it easily converges to a local minimum. Moreover, in practice we have found this algorithm sometimes to be very slow in convergence. Examples of both problems are discussed in section 4.6.2. We must warn the reader that the calculation of $\mathbf{Y}^*$ and $\mathbf{h}_j^*$ according to (4.53) and (4.54) in step h) of the algorithm does *not* result in a unique solution.

Although we will provide an analysis of variation for the STIMIDIO model in the following section, and discuss two examples of the analysis of configurations with this model in section 4.6.2, this will mainly be done for the sake of completeness. Considering all the problems involved in STIMIDIO we advise against using this model. Possibly better results would be obtained by imposing extra restrictions on the parameters of the STIMIDIO model. One obvious restriction would be to require that (diag $\mathbf{YY}'$) = $\mathbf{I}_p$. However, we have not investigated this option.

4.5 Analysis of variation in the stimulus weighting models

4.5.1 Introduction

The following sections deal with the partitioning of the total sum of squares of the n configurations for the stimulus weighting models. In section 4.5.2 we discuss the analysis of variation for the STIMFREE model (4.5); the analysis of variation for the STIMIDIO model (4.7) is presented in section 4.5.3.

4.5.2 Partitioning of the sums of squares in STIMFREE

The linear model (4.13) underlying the STIMFREE model (4.5) is (see section 4.3.2)

$$\mathbf{C}_j\tilde{\mathbf{X}}_j\mathbf{T}_j = \mathbf{C}_j\mathbf{V}_j\mathbf{Y} + \mathbf{C}_j\mathbf{E}_j \quad \text{for } j = 1, 2, ..., n,$$

from which it follows that

$$\text{tr } \mathbf{T}_j'\tilde{\mathbf{X}}_j'\mathbf{C}_j\tilde{\mathbf{X}}_j\mathbf{T}_j = \text{tr } (\mathbf{V}_j\mathbf{Y} + \mathbf{E}_j)'\mathbf{C}_j(\mathbf{V}_j\mathbf{Y} + \mathbf{E}_j),$$

and therefore that

$$\text{tr } \tilde{\mathbf{X}}_j'\mathbf{C}_j\tilde{\mathbf{X}}_j = \text{tr } \mathbf{Y}'\mathbf{V}_j\mathbf{C}_j\mathbf{V}_j\mathbf{Y} + \text{tr } \mathbf{E}_j'\mathbf{C}_j\mathbf{E}_j + 2 \text{ tr } \mathbf{Y}'\mathbf{V}_j\mathbf{C}_j\mathbf{E}_j. \tag{4.57}$$

Whether we use unrestricted or nonnegative weights (see section 4.3.4) it is true for optimal $\mathbf{V}_j$ that (tr $\mathbf{Y}'\mathbf{V}_j\mathbf{C}_j\mathbf{E}_j$) = 0. Defining $\mathbf{b}_j$, $\mathbf{b}_j^*$, $\mathbf{v}_j$, $\mathbf{v}_j^*$, and $\mathbf{D}_j$ as we did in section 4.3.4, the latter trace may be written as

$$\begin{aligned}\text{tr } \mathbf{Y}'\mathbf{V}_j\mathbf{C}_j\mathbf{E}_j &= \text{tr } \mathbf{Y}'\mathbf{V}_j\mathbf{C}_j\tilde{\mathbf{X}}_j\mathbf{T}_j - \text{tr } \mathbf{Y}'\mathbf{V}_j\mathbf{C}_j\mathbf{V}_j\mathbf{Y} \\ &= \mathbf{b}_j'\mathbf{v}_j - \mathbf{v}_j'(\mathbf{C}_j\odot\mathbf{Y}\mathbf{Y}')\mathbf{v}_j = \mathbf{b}_j^{*\prime}\mathbf{v}_j^* - \mathbf{v}_j^{*\prime}\mathbf{D}_j\mathbf{v}_j^*.\end{aligned} \tag{4.58}$$

In the case of unrestricted weights the optimal vector $\mathbf{v}_j^*$ is calculated using (4.19), and substitution of (4.19) in (4.58) yields

$$\mathrm{tr}\ \mathbf{Y'V}_j\mathbf{C}_j\mathbf{E}_j = \mathbf{b}_j^{*\prime}\mathbf{D}_j^{-1}\mathbf{b}_j^* - \mathbf{b}_j^{*\prime}\mathbf{D}_j^{-1}\mathbf{D}_j\mathbf{D}_j^{-1}\mathbf{b}_j^* = 0. \tag{4.59}$$

The same holds for nonnegative stimulus weights. We skip the proof. Thus, (4.57) may be written as

$$\mathrm{tr}\ \tilde{\mathbf{X}}'_j\mathbf{C}_j\tilde{\mathbf{X}}_j = \mathrm{tr}\ \mathbf{Y'V}_j\mathbf{C}_j\mathbf{V}_j\mathbf{Y} + \mathrm{tr}\ \mathbf{E}_j'\mathbf{C}_j\mathbf{E}_j. \tag{4.60}$$

It immediately follows from (4.60) that

$$0 \le (\mathrm{tr}\ \mathbf{Y'V}_j\mathbf{C}_j\mathbf{V}_j\mathbf{Y})/(\mathrm{tr}\ \tilde{\mathbf{X}}'_j\mathbf{C}_j\tilde{\mathbf{X}}_j) \le 1, \qquad \text{for } j = 1, 2, \ldots, n, \tag{4.61}$$

showing that (4.61) is perfectly suited as a measure of fit for each separate configuration j in the STIMFREE solution. It further follows from (4.60) that

$$\sum_{j=1}^{n} \mathrm{tr}\ \tilde{\mathbf{X}}'_j\mathbf{C}_j\tilde{\mathbf{X}}_j = \sum_{j=1}^{n} \mathrm{tr}\ \mathbf{Y'V}_j\mathbf{C}_j\mathbf{V}_j\mathbf{Y} + \sum_{j=1}^{n} \mathrm{tr}\ \mathbf{E}_j'\mathbf{C}_j\mathbf{E}_j,$$

and therefore that

$$n = \mathrm{tr}\ \mathbf{Y}'(\sum_{j=1}^{n} \mathbf{V}_j\mathbf{C}_j\mathbf{V}_j)\mathbf{Y} + f(\mathbf{T},\mathbf{V},\mathbf{Y}), \tag{4.62}$$

with $f(\mathbf{T},\mathbf{V},\mathbf{Y})$ as in (4.12), and thus that

$$0 \le (1/n)\ \mathrm{tr}\ \mathbf{Y}'(\sum_{j=1}^{n} \mathbf{V}_j\mathbf{C}_j\mathbf{V}_j)\mathbf{Y} \le 1. \tag{4.63}$$

Hence, (4.63) is a measure of total fit in the STIMFREE model.

The measure of configuration fit (4.61) is equal to the squared correlation between the elements of $\mathbf{C}_j\tilde{\mathbf{X}}_j\mathbf{T}_j$ and the elements of $\mathbf{C}_j\mathbf{V}_j\mathbf{Y}$. This is true because

$$r^2(\mathbf{C}_j\tilde{\mathbf{X}}_j\mathbf{T}_j,\mathbf{C}_j\mathbf{V}_j\mathbf{Y}) = \frac{(\mathrm{tr}\ \mathbf{Y'V}_j\mathbf{C}_j\tilde{\mathbf{X}}_j\mathbf{T}_j)^2}{(\mathrm{tr}\ \tilde{\mathbf{X}}'_j\mathbf{C}_j\tilde{\mathbf{X}}_j)(\mathrm{tr}\ \mathbf{Y'V}_j\mathbf{C}_j\mathbf{V}_j\mathbf{Y})}$$

$$= (\mathrm{tr}\ \mathbf{Y'V}_j\mathbf{C}_j\mathbf{V}_j\mathbf{Y})/(\mathrm{tr}\ \tilde{\mathbf{X}}'_j\mathbf{C}_j\tilde{\mathbf{X}}_j), \tag{4.64}$$

due to the fact that $(\mathrm{tr}\ \mathbf{Y'V}_j\mathbf{C}_j\mathbf{E}_j) = 0$.

Table 4.1 Contribution of individual configurations in the STIMFREE model.

configuration j	SS_{fit}	$SS_{residual}$	SS_{total}
1	$\text{tr}\, \mathbf{Y'V_1C_1V_1Y}$	$\text{tr}\, \mathbf{E_1'C_1E_1}$	$\text{tr}\, \mathbf{\tilde{X}_1'C_1\tilde{X}_1}$
2	$\text{tr}\, \mathbf{Y'V_2C_2V_2Y}$	$\text{tr}\, \mathbf{E_2'C_2E_2}$	$\text{tr}\, \mathbf{\tilde{X}_2'C_2\tilde{X}_2}$
⋮	⋮	⋮	⋮
n	$\text{tr}\, \mathbf{Y'V_nC_nV_nY}$	$\text{tr}\, \mathbf{E_n'C_nE_n}$	$\text{tr}\, \mathbf{\tilde{X}_n'C_n\tilde{X}_n}$
$\sum_{j=1}^{n}$	$\text{tr}\, \mathbf{Y'}(\sum_{j=1}^{n} \mathbf{V_jC_jV_j})\mathbf{Y}$	$\sum_{j=1}^{n} \text{tr}\, \mathbf{E_j'C_jE_j}$	n

The results obtained thus far can be used to further decompose the sums of squares in (4.62) allowing one to assess the relative contribution of individual configurations, stimuli and dimensions to the STIMFREE solution. Table 4.1 shows the decomposition of (4.62) with respect to individual configurations, and is, in fact, nothing more than (4.60) in tabulated form.

Table 4.2 Contribution of individual stimuli in the STIMFREE model.

stimulus i	SS_{fit}	$SS_{residual}$	SS_{total}
1	a_{11}*	b_{11}**	d_{11}***
2	a_{22}*	b_{22}**	d_{22}***
⋮	⋮	⋮	⋮
p	a_{pp}*	b_{pp}**	d_{pp}***
$\sum_{i=1}^{p}$	$\text{tr}\, \mathbf{Y'}(\sum_{j=1}^{n} \mathbf{V_jC_jV_j})\mathbf{Y}$	$\sum_{j=1}^{n} \text{tr}\, \mathbf{E_j'C_jE_j}$	n

* a_{ii} is diagonal element ii of $\sum_{j=1}^{n} \{2\, \mathbf{C_jV_jY(C_j\tilde{X}_jT_j)'} - \mathbf{C_jV_jY(C_jV_jY)'}\}$

** b_{ii} is diagonal element ii of $\sum_{j=1}^{n} \mathbf{C_jE_j(C_jE_j)'}$

*** d_{ii} is diagonal element ii of $\sum_{j=1}^{n} \mathbf{C_j\tilde{X}_jT_j(C_j\tilde{X}_jT_j)'}$

Table 4.3 Contribution of individual dimensions in the STIMFREE model.

dimension k	SS_{fit}	$SS_{residual}$	SS_{total}
1	e_{11}*	f_{11}**	g_{11}***
2	e_{22}*	f_{22}**	g_{22}***
⋮	⋮	⋮	⋮
m	e_{mm}*	f_{mm}**	g_{mm}***
$\sum_{k=1}^{m}$	$\text{tr}\,\mathbf{Y}'(\sum_{j=1}^{n} \mathbf{V}_j\mathbf{C}_j\mathbf{V}_j)\mathbf{Y}$	$\sum_{j=1}^{n} \text{tr}\,\mathbf{E}_j'\mathbf{C}_j\mathbf{E}_j$	n

* e_{kk} is diagonal element kk of $\mathbf{\Lambda}$

** f_{kk} is diagonal element kk of $\sum_{j=1}^{n} (\tilde{\mathbf{X}}_j\mathbf{T}_j\mathbf{P} - \mathbf{V}_j\mathbf{Y}\mathbf{P})'\mathbf{C}_j(\tilde{\mathbf{X}}_j\mathbf{T}_j\mathbf{P} - \mathbf{V}_j\mathbf{Y}\mathbf{P})$

*** g_{kk} is diagonal element kk of $\sum_{j=1}^{n} (\mathbf{C}_j\tilde{\mathbf{X}}_j\mathbf{T}_j\mathbf{P})'\mathbf{C}_j\tilde{\mathbf{X}}_j\mathbf{T}_j\mathbf{P}$

In Table 4.2 the relative contribution of each individual stimulus to the STIMFREE solution is assessed. Unfortunately, these sums of squares are *not* independent, both with and without missing data. It is for this reason that we propose to calculate the amount of variation for each stimulus according to the expression given in the column titled SS_{fit} in Table 4.2, since this at least guarantees that SS_{fit} and $SS_{residual}$ add up to SS_{total} for each stimulus i, and that $\sum SS_{fit}$ is equal to the total sum of squares in (4.62). But we repeat that these calculations do not result in independent components.

The last decomposition of the sums of squares in (4.62) is given in Table 4.3. This partitioning allows one to assess the relative contribution of the dimensions to the STIMFREE solution. Since this solution may be simultaneously rotated, just like in GPA, we propose to rotate the whole solution to the principal components of $\mathbf{Y}'(\Sigma\mathbf{V}_j\mathbf{C}_j\mathbf{V}_j)\mathbf{Y}$ by determining the eigenvector-eigenvalue decomposition $\mathbf{Y}'(\Sigma\mathbf{V}_j\mathbf{C}_j\mathbf{V}_j)\mathbf{Y} = \mathbf{P\Lambda P}'$, and rotating the whole solution with $\mathbf{P}$. This has the effect that the proportion of sum of squares accounted for by the first dimension will always be the largest in STIMFREE, the proportion accounted for by the second dimension will always be second largest, etc. The sums of squares in Table 4.3 can be determined independently both with and without missing data.

4.5.3 Partitioning of the sums of squares in STIMIDIO

The linear model (4.40) corresponding to the STIMIDIO model (4.7) is (see section 4.4.2):

$$\mathbf{C}_j\tilde{\mathbf{X}}_j\mathbf{T}_j = \mathbf{C}_j\mathbf{V}_j(\mathbf{Y} - \mathbf{1h}_j') + \mathbf{C}_j\mathbf{E}_j \qquad \text{for } j = 1, 2, ..., n.$$

Defining $\mathbf{B}_j = (\mathbf{Y} - \mathbf{1h}_j')$, and applying the same procedures to (4.40) as those discussed in the previous section, it is easily verified that the following relations hold for the STIMIDIO model as well:

$$\text{tr } \tilde{\mathbf{X}}'_j\mathbf{C}_j\tilde{\mathbf{X}}_j = \text{tr } \mathbf{B}_j'\mathbf{V}_j\mathbf{C}_j\mathbf{V}_j\mathbf{B}_j + \text{tr } \mathbf{E}_j'\mathbf{C}_j\mathbf{E}_j, \quad \text{for } j = 1, 2, ..., n. \tag{4.65}$$

Therefore

$$0 \leq (\text{tr } \mathbf{B}_j'\mathbf{V}_j\mathbf{C}_j\mathbf{V}_j\mathbf{B}_j)/(\text{tr } \tilde{\mathbf{X}}'_j\mathbf{C}_j\tilde{\mathbf{X}}_j) \leq 1, \tag{4.66}$$

and thus

$$(\text{tr } \mathbf{B}_j'\mathbf{V}_j\mathbf{C}_j\mathbf{V}_j\mathbf{B}_j)/(\text{tr } \tilde{\mathbf{X}}'_j\mathbf{C}_j\tilde{\mathbf{X}}_j) = r^2(\mathbf{C}_j\tilde{\mathbf{X}}_j\mathbf{T}_j, \mathbf{C}_j\mathbf{V}_j\mathbf{B}_j) \tag{4.67}$$

is a measure of fit for each separate configuration j in STIMIDIO. Again, it follows from (4.65) that

$$n = \sum_{j=1}^{n} \text{tr } \mathbf{B}_j'\mathbf{V}_j\mathbf{C}_j\mathbf{V}_j\mathbf{B}_j + f(\mathbf{H},\mathbf{T},\mathbf{V},\mathbf{Y}), \tag{4.68}$$

with $f(\mathbf{H},\mathbf{T},\mathbf{V},\mathbf{Y})$ as defined in (4.39); therefore

$$0 \leq (1/n)\sum_{j=1}^{n} \text{tr } \mathbf{B}_j'\mathbf{V}_j\mathbf{C}_j\mathbf{V}_j\mathbf{B}_j \leq 1,$$

is a measure of total fit in the STIMIDIO model.

For a further decomposition of the sums of squares in (4.68) with respect to individual configurations, stimuli and dimensions we refer to Tables 4.1, 4.2, and 4.3, respectively, where in all tables one should now read $\mathbf{B}_j = \mathbf{Y} - \mathbf{1h}_j'$ instead of $\mathbf{Y}$. The MATCHALS program computes and prints all decompositions discussed in this and the previous section.

4.6 Illustrations

In the next two sections constructed data sets are matched with the stimulus weighting models. The algorithms presented in sections 4.3.8 and 4.4.8 were programmed in APL, and we used a convergence criterion of 1E-7 throughout. In section 4.6.1 the results of two analyses of a constructed data set with the STIMFREE model are given, and in section 4.6.2 we present the results of the STIMIDIO analysis of a constructed data set.

4.6.1 STIMFREE analysis of a constructed data set

To investigate the performance of the STIMFREE algorithm, we constructed a data set such that we knew in advance that a perfect solution exists when the admissible transformations are those corresponding to STIMFREE model (4.5). The 'mould' of this example consists of the coordinates of a regular octagon, a two-dimensional figure, which we located in three-dimensional space. The coordinates of the octagon are given in matrix **A** in Table 4.4. Since the first column of **A** contains zero elements only, the octagon is situated in the plane spanned by the y- and z-axes, and, therefore, orthogonal to the x-axis. This also means that we used a singular matrix as mould (i.e., **A** has rank two instead of three).

By translating this octagon located in three-dimensional space with the vector $\mathbf{t}' = (2.5, 0, 0)$, and by differently weighting the vectors obtained by connecting the vertices of the translated octagon with the origin, a set of four configurations was constructed. The last step in the construction consisted of a different rotation of each configuration. In this way four matrices of full column rank were obtained from a rank-deficient mould. Table 4.4 contains all transformation parameters that we used to construct this data set. By calculating $\mathbf{X}_j = \mathbf{V}_j(\mathbf{A} - \mathbf{jt}')\mathbf{R}_j$ for $j = 1, \ldots, 4$ we obtained the data given in Table 4.5. The geometry of the octagon in two dimensions, and its location in three-dimensional space after translation is shown in Figure 4.5.

We wanted to find out whether the STIMFREE algorithm would recover the common structure in the data of Table 4.5. A generalized Procrustes analysis of these data yielded a loss of 0.7129, which means that the relative distances preserving model accounts for $(4 - 0.7129)/4 = 82\%$ of the total variation in the data.

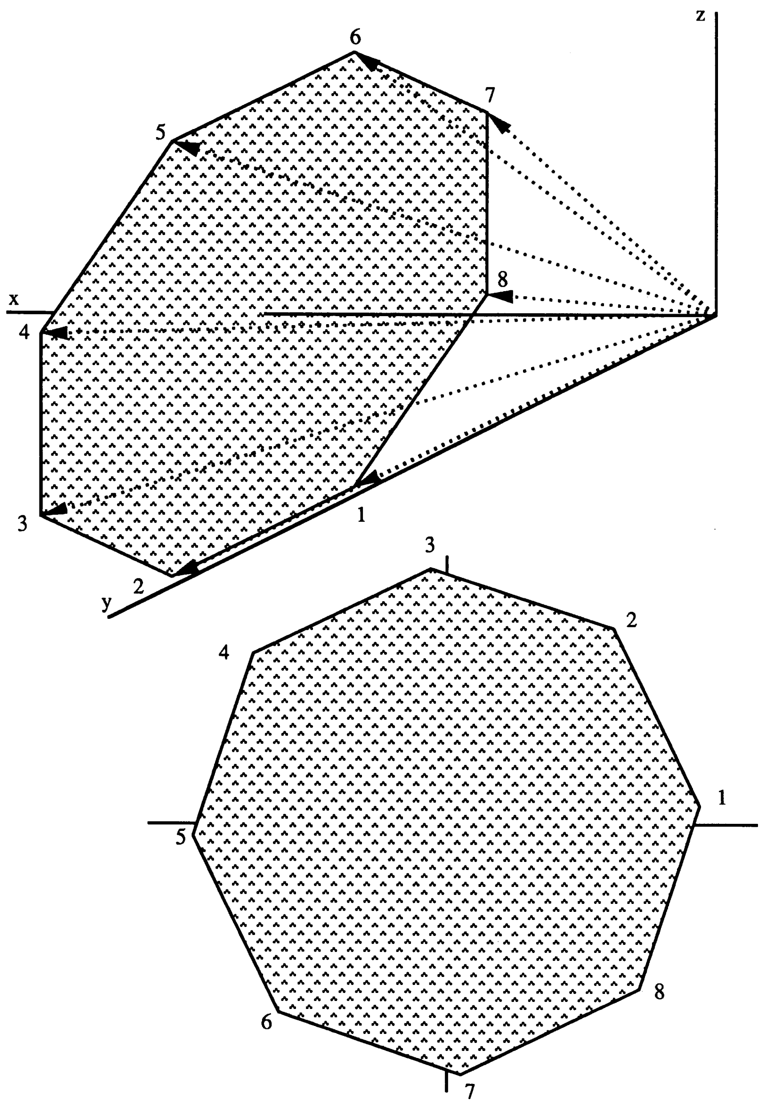

Figure 4.5 Geometry of mould (**A** – **1t'**) in three dimensions (top), and of **A** in two dimensions (bottom).

Table 4.4 Matrices and vector used in the construction of data in Table 4.5.

$$\mathbf{A} = \begin{bmatrix} 0 & -0.5000 & -1.2071 \\ 0 & 0.5000 & -1.2071 \\ 0 & 1.2071 & -0.5000 \\ 0 & 1.2071 & 0.5000 \\ 0 & 0.5000 & 1.2071 \\ 0 & -0.5000 & 1.2071 \\ 0 & -1.2071 & 0.5000 \\ 0 & -1.2071 & -0.5000 \end{bmatrix} \qquad \mathbf{t} = \begin{bmatrix} 2.5 \\ 0 \\ 0 \end{bmatrix}$$

$$\mathbf{R}_1 = \begin{bmatrix} -0.6077 & 0.5803 & -0.5422 \\ 0.7527 & 0.2030 & -0.6263 \\ -0.2534 & -0.7887 & -0.5601 \end{bmatrix} \qquad \mathbf{R}_2 = \begin{bmatrix} 0.9761 & -0.0977 & 0.1940 \\ 0.0487 & 0.9688 & 0.2430 \\ 0.2117 & 0.2277 & -0.9504 \end{bmatrix}$$

$$\mathbf{R}_3 = \begin{bmatrix} 0.2592 & -0.3102 & -0.9146 \\ -0.2659 & 0.8875 & -0.3764 \\ -0.9285 & -0.3408 & -0.1476 \end{bmatrix} \qquad \mathbf{R}_4 = \begin{bmatrix} -0.2315 & -0.1672 & -0.9584 \\ 0.3997 & 0.8818 & -0.2504 \\ 0.8870 & -0.4410 & -0.1373 \end{bmatrix}$$

stimulus weights $\mathbf{V}_j$

configuration	stimulus 1	2	3	4	5	6	7	8
1	0.900	0.700	0.600	0.800	1.000	0.300	0.400	0.500
2	1.000	0.750	0.250	0.375	1.375	0.875	1.500	0.500
3	1.000	1.500	0.900	1.300	0.100	0.300	1.200	1.400
4	0.100	0.600	0.500	1.100	0.300	0.700	1.000	0.900

Table 4.5 Constructed data for analysis with STIMFREE model.

$$X_1 = \begin{bmatrix} 1.3039 & -0.5401 & 2.1103 \\ 1.5410 & -0.2780 & 1.2029 \\ 1.5327 & -0.4867 & 0.5277 \\ 1.8409 & -1.2800 & 0.2556 \\ 1.5898 & -2.3012 & 0.3663 \\ 0.2511 & -0.7513 & 0.2978 \\ 0.1936 & -0.8360 & 0.7326 \\ 0.3687 & -0.6507 & 1.1958 \end{bmatrix} \quad X_2 = \begin{bmatrix} -2.7202 & -0.5151 & 0.5407 \\ -2.0036 & 0.3403 & 0.5878 \\ -0.6219 & 0.3250 & 0.0709 \\ -0.8534 & 0.5728 & -0.2501 \\ -2.9706 & 1.3799 & -2.0774 \\ -1.9330 & 0.0304 & -1.5346 \\ -3.5898 & -1.2170 & -1.8803 \\ -1.3025 & -0.5195 & -0.1516 \end{bmatrix}$$

$$X_3 = \begin{bmatrix} 0.6056 & 0.7431 & 2.6529 \\ 0.5096 & 2.4459 & 3.4149 \\ -0.4544 & 1.8155 & 1.7155 \\ -1.8633 & 2.1793 & 2.2860 \\ -0.1902 & 0.0808 & 0.1920 \\ -0.4908 & -0.0239 & 0.6890 \\ -0.9497 & -0.5594 & 3.2005 \\ 0.1920 & -0.1756 & 3.9406 \end{bmatrix} \quad X_4 = \begin{bmatrix} -0.0692 & 0.0509 & 0.2687 \\ -0.1753 & 0.8347 & 1.4619 \\ 0.3088 & 0.8515 & 1.0812 \\ 1.6550 & 1.3881 & 2.2276 \\ 0.5547 & 0.0980 & 0.6315 \\ 1.0146 & -0.3886 & 1.6488 \\ 0.5397 & -0.8669 & 2.6295 \\ -0.3125 & -0.3833 & 2.4901 \end{bmatrix}$$

Submitting the optimally transformed configurations from GPA to a STIMFREE analysis, it appeared that the algorithm of section 4.3.8 converged in 39 iterations. The complete history of iterations is shown in Table 4.6. The algorithm converged to a local minimum, since the loss at convergence is unequal to zero. At the local minimum, the STIMFREE model accounts for (4 – 0.0354)/4 = 99% of the total variation in the four configurations. A striking feature in the history of iterations for this data set is the sudden drop in loss at iteration number 10.

The optimal transformation parameters corresponding to the STIMFREE analysis of this data set are given in Table 4.7. In Figure 4.6 the location of the uniquely weighted centroid configuration $\mathbf{Y}^*$ rotated to principal components is shown in three-dimensional space. The coordinates of $\mathbf{Y}^*$ rotated to principal components are given at the top of Table 4.7, where $\mathbf{Q}$ is the matrix of eigenvectors in the eigenvalue-eigenvector decomposition $\mathbf{Y}^{*\prime}\mathbf{Y}^* = \mathbf{Q}\Lambda\mathbf{Q}'$. Figure 4.6 also contains a plot of the 'flattened' centroid configuration $\mathbf{Y}^*$. As coordinates of the flattened configuration, the first two eigenvectors of the eigenvalue-eigenvector decomposition $\mathbf{J}\mathbf{Y}^*(\mathbf{J}\mathbf{Y}^*)' = \mathbf{P}\Phi\mathbf{P}'$ were used, where $\mathbf{J}$ is the centring matrix for the columns of $\mathbf{Y}^*$.

As Figure 4.6 shows, the basic structure of the original mould is recovered by the STIMFREE algorithm. But the recovery is not perfect since the octagon at the bottom

Table 4.6 History of iterations of STIMFREE analysis of data in Table 4.5.

Iteration number	TOTAL LOSS stimulus weighting step	rotation step
1	0.43568246	0.41794632
2	0.38971249	0.38360589
3	0.35985999	0.35345816
4	0.31872274	0.30527786
5	0.25953362	0.23876176
6	0.20702699	0.19223683
7	0.17755962	0.16975370
8	0.16280736	0.15892717
9	0.15542006	0.14343834
10	0.04549479	0.03975107
11	0.03752761	0.03677707
12	0.03620780	0.03598696
13	0.03568135	0.03559487
14	0.03545197	0.03543438
15	0.03540051	0.03539813
16	0.03538608	0.03538524
17	0.03537998	0.03537958
18	0.03537716	0.03537696
19	0.03537583	0.03537573
20	0.03537520	0.03537515
21	0.03537490	0.03537488
22	0.03537476	0.03537475
23	0.03537469	0.03537469

of the figure is not regular. Moreover, the stimulus points of the optimal $\mathbf{Y}^*$ shown at the top of the figure are not exactly located in one plane, as the figure might suggest. This is indicated by the eigenvalues of $\mathbf{JY}^*(\mathbf{JY}^*)'$: the eigenvalues in $\mathbf{\Phi}$ are 0.7922, 0.7415, and 0.0028. Since the third eigenvalue is unequal to zero, there is some scatter in the stimulus points of $\mathbf{Y}^*$ on the third principal component, while the scatter of the stimulus points in the original mould **A** on the third principal component is zero, of course.

Table 4.7 also contains the optimal raw stimulus weights at the local optimum of the STIMFREE loss function, together with the unique weights and the impact indices corresponding to the weights. The latter index shows how far a weight displaces the end point of the (unit normalized) stimulus vector of $\mathbf{Y}^*$. Thus, in this example the largest displacement is caused by stimulus weight 5 for configuration 1, while

Table 4.7 Results of STIMFREE algorithm for data in Table 4.5.

$$\mathbf{Y} = \begin{bmatrix} -0.1066 & 0.4310 & 0.2908 \\ 0.0441 & 0.4053 & 0.4333 \\ 0.0819 & 0.1870 & 0.4005 \\ 0.1087 & 0.0721 & 0.5863 \\ -0.0524 & -0.0062 & 0.4058 \\ -0.2536 & 0.0272 & 0.4687 \\ -0.4772 & 0.2158 & 0.5813 \\ -0.3232 & 0.3415 & 0.4097 \end{bmatrix} \quad \mathbf{Y}^*\mathbf{Q} = \begin{bmatrix} 0.8553 & 0.4946 & 0.1543 \\ 0.9070 & 0.2135 & 0.3631 \\ 0.9238 & -0.1155 & 0.3650 \\ 0.8819 & -0.3987 & 0.2515 \\ 0.9060 & -0.4148 & -0.0847 \\ 0.9035 & -0.1933 & -0.3826 \\ 0.9003 & 0.0987 & -0.4239 \\ 0.9130 & 0.3321 & -0.2368 \end{bmatrix}$$

$$\mathbf{T}_1 = \begin{bmatrix} 0.9925 & 0.1035 & -0.0656 \\ -0.0656 & 0.8339 & 0.5495 \\ 0.1115 & -0.5421 & 0.8329 \end{bmatrix} \quad \mathbf{T}_2 = \begin{bmatrix} 0.9126 & -0.1847 & -0.3647 \\ -0.0820 & 0.7912 & -0.6060 \\ -0.4005 & -0.5829 & -0.7070 \end{bmatrix}$$

$$\mathbf{T}_3 = \begin{bmatrix} 0.9508 & 0.1570 & 0.2670 \\ -0.2561 & 0.8834 & 0.3925 \\ -0.1742 & -0.4416 & 0.8801 \end{bmatrix}. \quad \mathbf{T}_4 = \begin{bmatrix} 0.9979 & 0.0588 & 0.0256 \\ -0.0462 & 0.9360 & -0.3490 \\ -0.0444 & 0.3471 & 0.9368 \end{bmatrix}$$

raw stimulus weights $\mathbf{V}_j$

configuration	stimulus 1	2	3	4	5	6	7	8
1	1.5321	1.1892	1.2986	0.5265	0.0000	1.2753	0.8749	1.1713
2	1.1686	0.7844	0.3674	0.4129	2.1209	1.0498	1.2016	0.5109
3	1.0967	1.4514	1.1611	1.2423	0.1773	0.3506	0.8892	1.2983
4	0.2349	0.8691	0.9562	1.4980	0.6515	1.1010	1.0650	1.2177

unique stimulus weights $\mathbf{V}_j^*/(\sqrt{\text{tr}\ \mathbf{X}_j'\mathbf{C}_j\mathbf{X}_j})$

configuration	stimulus 1	2	3	4	5	6	7	8
1	0.8130	0.7074	0.5836	0.3161	0.0000	0.6803	0.6844	0.7304
2	0.6307	0.4746	0.1680	0.2522	0.8826	0.5696	0.9560	0.3240
3	0.5669	0.8410	0.5083	0.7267	0.0707	0.1822	0.6776	0.7886
4	0.1260	0.5228	0.4345	0.9096	0.2695	0.5939	0.8424	0.7678

impact index of $\mathbf{V}_j^*/(\sqrt{\text{tr}\ \mathbf{X}_j'\mathbf{C}_j\mathbf{X}_j})$

configuration	stimulus 1	2	3	4	5	6	7	8
1	0.1870	0.2926	0.4164	0.6839	1.0000	0.3197	0.3156	0.2696
2	0.3693	0.5254	0.8320	0.7478	0.1174	0.4304	0.0440	0.6760
3	0.4331	0.1590	0.4917	0.2733	0.9293	0.8178	0.3224	0.2114
4	0.8740	0.4772	0.5655	0.0904	0.7305	0.4061	0.1576	0.2322

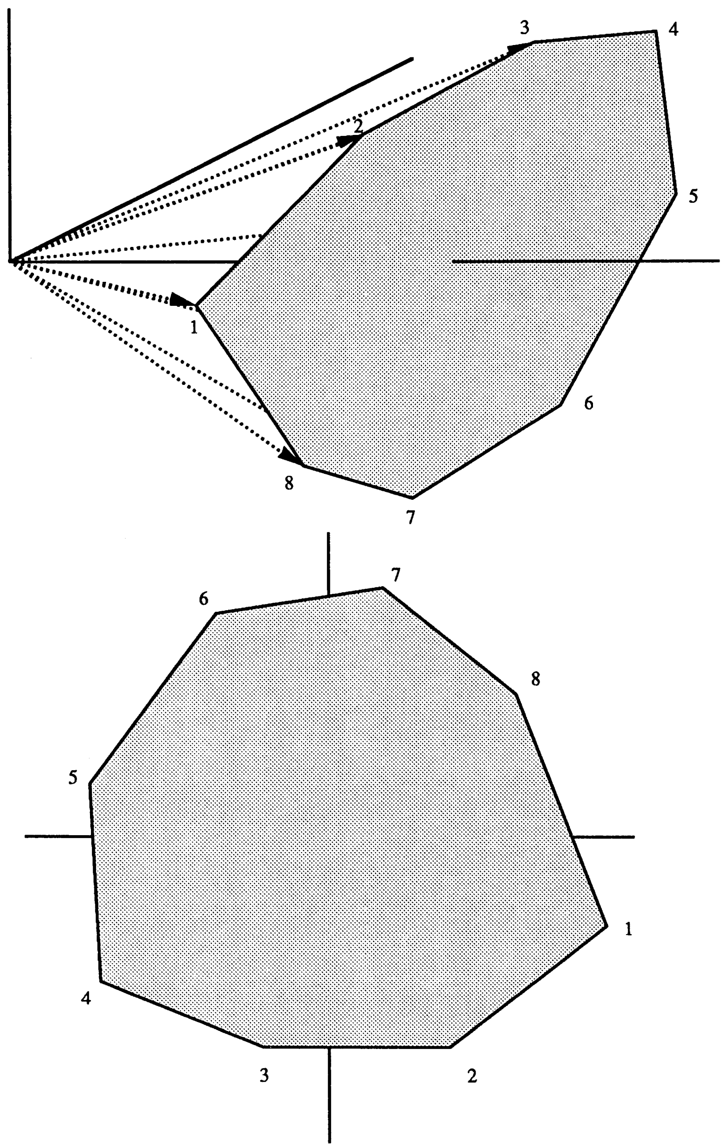

Figure 4.6 Plot of $\mathbf{Y}^*$ rotated to principal components after STIMFREE analysis of data in Table 4.5 (top), and plot of 'flattened' $\mathbf{Y}^*$ (bottom).

stimulus weight 7 for configuration 2 hardly displaces stimulus point 7 of $\mathbf{Y}^*$ at all. Averaged over the four configurations it is stimulus 5 which needs the largest displacement (the mean impact index for this stimulus is 0.6943), while the mean impact for stimulus 7 is the smallest (i.e., 0.2099).

That the weight for stimulus 5 in configuration 1 is equal to zero indicates that this weight would have been negative if we had not used the nonnegative least squares algorithm of Lawson and Hanson (1974) to estimate the weights.

We also analysed the four configurations in Table 4.5 treating the following rows as missing: rows 1, 3, 6, and 8 of configuration 2, rows 1, 3, 5, and 7 of configuration 3, and rows 3 and 7 of configuration 4. A generalized Procrustes analysis of these incomplete configurations yielded a loss of 0.4427, and, thus, a fit of $(4 - 0.4427)/4 = 0.89$.

The algorithm for the STIMFREE model converged in 195 iterations to a local minimum, since the loss at convergence was 0.0469 and not zero. The transformation parameters at the local minimum are given in Table 4.8, and Figure 4.7 contains a plot of the uniquely weighted $\mathbf{Y}^*$ in three dimensions as well as a 'flattened' version of $\mathbf{Y}^*$.

At the local minimum the STIMFREE model accounts for $(4 - 0.0389)/4 = 99\%$ of the total variation in the four incomplete configurations. At the top of Figure 4.7 a plot of $\mathbf{Y}^*$ is shown rotated to its principal components. The corresponding coordinates are given at the top of Table 4.8, where $\mathbf{Q}$ is the matrix of eigenvectors in the eigenvalue-eigenvector decomposition $\mathbf{Y}^{*\prime}\mathbf{Y}^* = \mathbf{Q\Lambda Q}'$. The coordinates of the 'flattened' version of $\mathbf{Y}^*$ at the bottom of Figure 4.7 are the first two eigenvectors of matrix $\mathbf{P}$ in the eigenvalue-eigenvector decomposition $\mathbf{JY}^*(\mathbf{JY}^*)' = \mathbf{P\Phi P}'$, where $\mathbf{J}$ is the centring matrix for the columns of $\mathbf{Y}^*$. The flattened version of $\mathbf{Y}^*$ illustrates particularly well how much (or, perhaps more appropriately: how little) of the original regular octagon has been preserved in the STIMFREE solution. Stimuli 1 through 5 still more or less form the contour of the original octagon, but the location of stimuli 6, 7, and 8 is quite different from what we would expect. Moreover, information is lost in the flattened version of $\mathbf{Y}^*$. The eigenvalues in $\mathbf{\Phi}$ are 1.9181, 1.4338, and 0.2839, showing that there is some scatter in the stimulus points of $\mathbf{Y}^*$ on the third principal component.

The impact indices corresponding to the unique stimulus weights in Table 4.8 show that stimulus 5 for configuration 2 needs the largest displacement, while

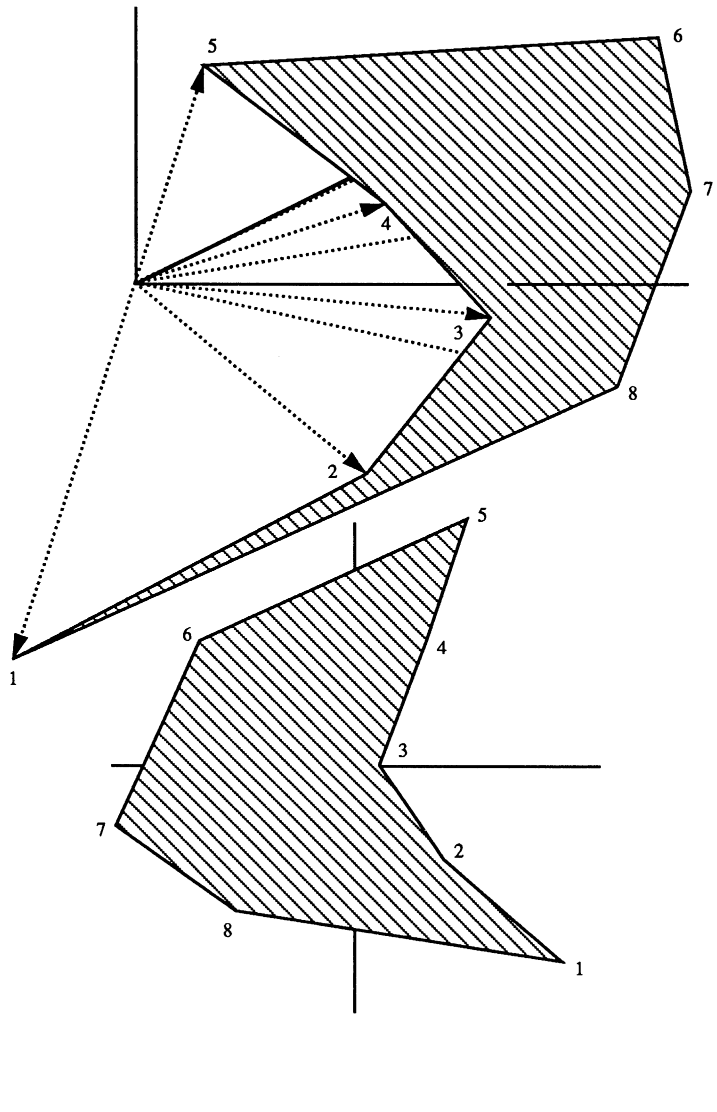

Figure 4.7 Plot of $\mathbf{Y}^*$ rotated to principal components after STIMFREE analysis of incomplete configurations of Table 4.5 (top), and plot of 'flattened' $\mathbf{Y}^*$ (bottom).

Table 4.8 Results of STIMFREE analysis of configurations in Table 4.5 containing missing data.

$$\mathbf{Y} = \begin{bmatrix} 0.1684 & -0.3071 & 0.1562 \\ 0.1947 & -0.2083 & 0.5434 \\ 0.1923 & 0.0042 & 0.6994 \\ 0.4848 & 0.2177 & 0.6786 \\ 0.3677 & 0.2415 & 0.1830 \\ -0.0930 & 0.4020 & 0.4974 \\ -0.3685 & 0.1475 & 0.3632 \\ -0.2334 & -0.0880 & 0.5540 \end{bmatrix} \quad \mathbf{Y^*Q} = \begin{bmatrix} 0.4099 & 0.7076 & -0.5756 \\ 0.8957 & 0.3265 & -0.3017 \\ 0.9934 & 0.1100 & -0.0338 \\ 0.8968 & 0.2773 & 0.3448 \\ 0.5630 & 0.3919 & 0.7276 \\ 0.7811 & -0.5115 & 0.3581 \\ 0.5581 & -0.8114 & -0.1739 \\ 0.8088 & -0.3818 & -0.4472 \end{bmatrix}$$

$$\mathbf{T}_1 = \begin{bmatrix} 0.6781 & 0.1661 & 0.7160 \\ 0.0037 & 0.9734 & -0.2292 \\ -0.7350 & 0.1581 & 0.6594 \end{bmatrix} \quad \mathbf{T}_2 = \begin{bmatrix} 0.9841 & 0.0473 & -0.1711 \\ -0.1326 & 0.8366 & -0.5316 \\ 0.1180 & 0.5458 & 0.8296 \end{bmatrix}$$

$$\mathbf{T}_3 = \begin{bmatrix} 0.9609 & 0.0465 & -0.2731 \\ 0.0965 & 0.8679 & 0.4872 \\ 0.2597 & -0.4945 & 0.8295 \end{bmatrix} \quad \mathbf{T}_4 = \begin{bmatrix} 0.9969 & 0.0419 & 0.0659 \\ -0.0442 & 0.9985 & 0.0336 \\ -0.0644 & -0.0364 & 0.9973 \end{bmatrix}$$

raw stimulus weights $\mathbf{V}_j$

configuration	stimulus 1	2	3	4	5	6	7	8
1	1.5537	1.0312	0.9199	0.9159	1.7497	0.6305	0.4089	0.3055
2	—	1.1950	—	0.8276	0.0000	—	1.2666	—
3	—	0.3647	—	0.8267	—	1.3470	—	1.1738
4	0.1752	0.8420	—	1.1650	0.4158	0.8718	—	1.2540

unique stimulus weights $\mathbf{V}_j^*/(\sqrt{\text{tr } \mathbf{X}_j'\mathbf{C}_j\mathbf{X}_j})$

configuration	stimulus 1	2	3	4	5	6	7	8
1	0.5069	0.5384	0.5677	0.6716	0.7092	0.3467	0.1872	0.1579
2	—	0.8125	—	0.7903	0.0000	—	0.7550	—
3	—	0.2736	—	0.8710	—	1.0642	—	0.8718
4	0.0631	0.4851	—	0.9427	0.1860	0.5290	—	0.7153

impact index for $\mathbf{V}_j^*/(\sqrt{\text{tr } \mathbf{X}_j'\mathbf{C}_j\mathbf{X}_j})$

configuration	stimulus 1	2	3	4	5	6	7	8
1	0.4981	0.4616	0.4323	0.3284	0.2908	0.6533	0.8128	0.8421
2	—	0.1875	—	0.2097	1.0000	—	0.2450	—
3	—	0.7264	—	0.1290	—	0.0642	—	0.1282
4	0.8984	0.5149	—	0.0573	0.8140	0.4710	—	0.2847

stimulus 6 for configuration 3 and stimulus 4 for configuration 4 are almost not displaced at all. Averaged over configurations, stimulus 5 again needs the largest displacement (the mean index for this stimulus being 0.7016), but the mean impact for stimulus 4 is now the smallest (i.e., 0.1811). The zero weight for stimulus 5 corresponding to configuration 2 indicates that a negative weight would have been obtained if we had not restricted the weights to be nonnegative.

Summarizing, both examples illustrate that the STIMFREE algorithm is not guaranteed to converge to a global minimum, even if a perfect solution exists.

4.6.2 STIMIDIO analysis of a constructed data set

Again, we constructed a data set such that we knew in advance that a perfect solution exists when the admissible transformations are those corresponding to the STIMIDIO model. In the construction of this data set the same transformation parameters were used as in the previous section, except that the regular octagon was translated *differently* for each configuration. The following translation vectors were applied: $\mathbf{t}'_1 = (2.5, 0, 0)$, $\mathbf{t}'_2 = (0.8, -1.5, 3.0)$, $\mathbf{t}'_3 = (5.2, 2.2, -4.0)$, and $\mathbf{t}'_4 = (-0.4, 0, -3.0)$, and four configurations were constructed by calculating $\mathbf{X}_j = \mathbf{V}_j(\mathbf{A} - \mathbf{1t}'_j)\mathbf{R}_j$ for j = 1, ..., 4, with $\mathbf{A}$, $\mathbf{V}_j$, and $\mathbf{R}_j$ as given in Table 4.4. This resulted in the data shown in Table 4.9.

A generalized Procrustes analysis of these four complete configurations yielded a loss of 1.3432 in 6 iterations, which means that the GPA model accounts for 66% of the total variation in these configurations. When the results from GPA were submitted to the STIMFREE algorithm of section 4.3.8, this algorithm converged in 126 iterations yielding a loss of 0.1779. Thus, 96% of the total variation is accounted for by the STIMFREE model.

Table 4.9 Constructed data for analysis with STIMIDIO model.

$$\mathbf{X}_1 = \begin{bmatrix} 1.3039 & -0.5401 & 2.1103 \\ 1.5410 & -0.2780 & 1.2029 \\ 1.5327 & -0.4867 & 0.5277 \\ 1.8409 & -1.2800 & 0.2556 \\ 1.5898 & -2.3012 & 0.3663 \\ 0.2511 & -0.7513 & 0.2978 \\ 0.1936 & -0.8360 & 0.7326 \\ 0.3687 & -0.6507 & 1.1958 \end{bmatrix} \quad \mathbf{X}_2 = \begin{bmatrix} -1.6229 & 0.0889 & 4.0863 \\ -1.1807 & 0.7933 & 3.2470 \\ -0.3475 & 0.4760 & 0.9573 \\ -0.4419 & 0.7993 & 1.0795 \\ -1.4618 & 2.2103 & 2.7978 \\ -0.9728 & 0.5588 & 1.5678 \\ -1.9438 & -0.3111 & 3.4381 \\ -0.7538 & -0.2176 & 1.6212 \end{bmatrix}$$

$$\mathbf{X}_3 = \begin{bmatrix} -3.2233 & -1.7349 & 5.3601 \\ -5.2338 & -1.2712 & 7.4756 \\ -3.9004 & -0.4148 & 4.1519 \\ -6.8409 & -1.0421 & 5.8053 \\ -0.5731 & -0.1670 & 0.4627 \\ -1.6395 & -0.7673 & 1.5011 \\ -5.5443 & -3.5331 & 6.4491 \\ -5.1685 & -3.6448 & 7.7306 \end{bmatrix} \quad \mathbf{X}_4 = \begin{bmatrix} 0.1298 & -0.1298 & -0.0504 \\ 1.0185 & -0.2500 & -0.4528 \\ 1.3036 & -0.0524 & -0.5144 \\ 3.8436 & -0.6004 & -1.2827 \\ 1.1516 & -0.4444 & -0.3258 \\ 2.4074 & -1.6541 & -0.5850 \\ 2.5293 & -2.6747 & -0.5616 \\ 1.4781 & -2.0104 & -0.3819 \end{bmatrix}$$

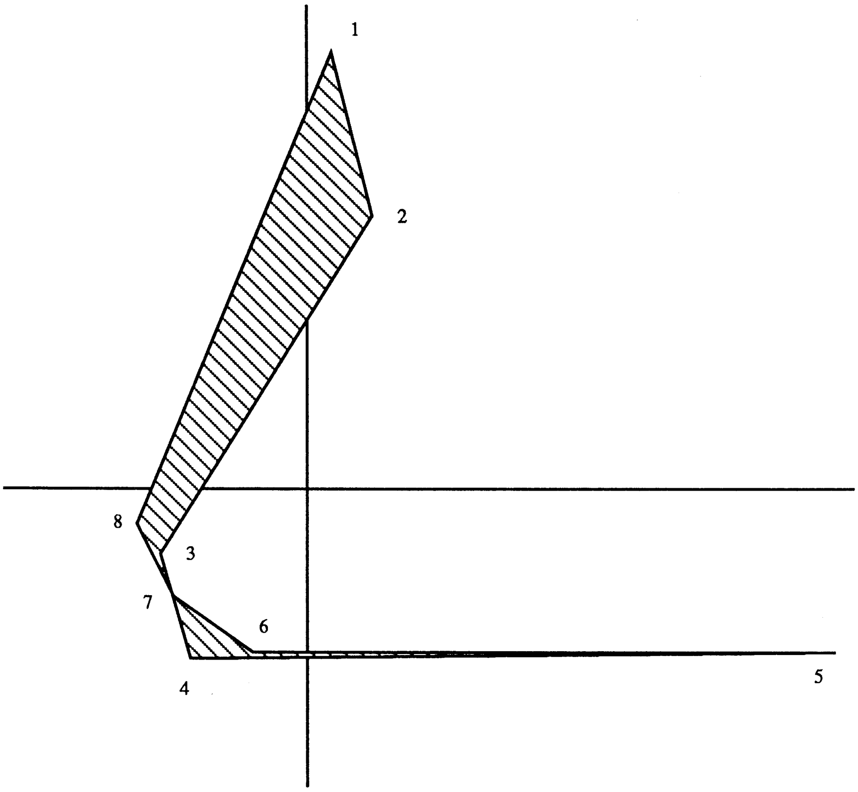

Figure 4.8 Plot of 'flattened' non-unique centroid configuration $\mathbf{Y}^*$ after STIMIDIO analysis of data in Table 4.9.

Then the optimally transformed $\mathbf{X}_j$'s from GPA and the optimal $\mathbf{Y}^*$ obtained in STIMFREE were used as input to the STIMIDIO algorithm of section 4.4.8. Convergence was very slow: it took the algorithm 1660 iterations to converge. The value of the STIMIDIO loss function at convergence was 0.0076, meaning that the solution accounts for almost all of the variation in the configurations. Since the solution was not perfect, however, the STIMIDIO algorithm converged to a local minimum.

The results of the STIMIDIO analysis are given in Table 4.10, where $\mathbf{Y}^*$ denotes the column centered centroid configuration and $\mathbf{h}_j^*$ is a relocated translation vector (cf., section 4.4.7). We emphasize that the parameters in the table are *not* unique. Figure 4.8 contains a plot of the two principal components of $\mathbf{Y}^*$. Clearly, the original mould

Table 4.10 Results of STIMIDIO algorithm for complete configurations in Table 4.9.

$$\mathbf{Y} = \begin{bmatrix} 0.1194 & -1.7662 & -1.0168 \\ -0.0694 & -1.8490 & -1.0795 \\ -0.0200 & -1.3124 & -0.5568 \\ -0.1076 & -1.4060 & -0.5824 \\ -0.5885 & -3.2111 & -1.6892 \\ -0.0233 & -1.6756 & -0.6123 \\ 0.0808 & -1.4554 & -0.4829 \\ 0.1265 & -1.3361 & -0.4587 \end{bmatrix} \qquad \mathbf{Y}^* = \begin{bmatrix} 0.1797 & -0.0148 & -0.2069 \\ -0.0091 & -0.0976 & -0.2697 \\ 0.0403 & 0.4391 & 0.2530 \\ -0.0474 & 0.3455 & 0.2274 \\ -0.5282 & -1.4596 & -0.8794 \\ 0.0370 & 0.0759 & 0.1975 \\ 0.1410 & 0.2960 & 0.3269 \\ 0.1867 & 0.4154 & 0.3512 \end{bmatrix}$$

$$\mathbf{h}_1' = [\ 0.3197 \quad -0.0551 \quad 0.4582\ ] \qquad \mathbf{h}_1^{*\prime} = [\ -0.3800 \quad -1.6964 \quad -1.2681\ ]$$

$$\mathbf{h}_2' = [\ -0.9083 \quad 1.0728 \quad 0.2676\ ] \qquad \mathbf{h}_2^{*\prime} = [\ 0.8480 \quad -2.8242 \quad -1.0774\ ]$$

$$\mathbf{h}_3' = [\ 0.1373 \quad -0.9825 \quad -0.2175\ ] \qquad \mathbf{h}_3^{*\prime} = [\ -0.1976 \quad -0.7689 \quad -0.5923\ ]$$

$$\mathbf{h}_4' = [\ 0.1071 \quad -1.1123 \quad -0.3810\ ] \qquad \mathbf{h}_4^{*\prime} = [\ -0.1674 \quad -0.6391 \quad -0.4288\ ]$$

$$\mathbf{T}_1 = \begin{bmatrix} 0.7598 & -0.4744 & -0.4446 \\ 0.5507 & 0.8331 & 0.0520 \\ 0.3457 & -0.2843 & 0.8942 \end{bmatrix} \qquad \mathbf{T}_2 = \begin{bmatrix} 0.3805 & 0.7914 & 0.4784 \\ -0.6630 & 0.5941 & -0.4555 \\ -0.6447 & -0.1439 & 0.7508 \end{bmatrix}$$

$$\mathbf{T}_3 = \begin{bmatrix} 0.5675 & -0.7199 & -0.3995 \\ 0.6865 & 0.6817 & -0.2532 \\ 0.4546 & -0.1306 & 0.8811 \end{bmatrix} \qquad \mathbf{T}_4 = \begin{bmatrix} 0.9719 & -0.2351 & 0.0087 \\ 0.2057 & 0.8672 & 0.4535 \\ -0.1142 & -0.4390 & 0.8912 \end{bmatrix}$$

stimulus weights $\mathbf{V}_j$

	stimulus							
configuration	1	2	3	4	5	6	7	8
1	2.0101	1.9543	2.8346	2.5131	1.0514	2.4451	2.7935	2.9887
2	0.4358	0.5135	0.8487	0.8213	0.4026	0.6852	0.5339	0.7530
3	0.4782	0.6354	0.9249	1.2568	0.0000	0.1802	1.4389	2.0072
4	0.2462	0.4099	1.5878	2.6265	0.1754	1.3604	2.8486	3.1353

A of Table 4.4 is distorted beyond recognition in the STIMIDIO solution. In contrast with the STIMFREE model, where the stimuli of the unique centroid configuration all lie on the unit (hyper)sphere, and where the angles between the vectors representing these stimuli are uniquely determined, in STIMIDIO the vectors representing the stimuli in $\mathbf{Y}^*$ all have different lengths, and the angles between these vectors are not unique.

Table 4.11 Results of STIMIDIO algorithm for incomplete configurations in Table 4.9.

$$\mathbf{Y} = \begin{bmatrix} -0.6402 & -0.6612 & 0.7795 \\ 0.3585 & -0.0273 & 0.5257 \\ 0.9972 & -0.2586 & 0.4716 \\ 1.0498 & 0.1619 & 0.0010 \\ 1.0198 & 0.2799 & -0.6301 \\ 0.4134 & -0.6230 & -0.0998 \\ 0.3939 & -0.8694 & -0.2797 \\ 0.2602 & -1.1327 & -0.2416 \end{bmatrix} \quad \mathbf{Y}^* = \begin{bmatrix} -1.1217 & -0.2699 & 0.7136 \\ -0.1231 & 0.3640 & 0.4599 \\ 0.5156 & 0.1327 & 0.4058 \\ 0.5683 & 0.5532 & -0.0648 \\ 0.5382 & 0.6712 & -0.6959 \\ -0.0682 & -0.2317 & -0.1656 \\ -0.0877 & -0.4781 & -0.3455 \\ -0.2214 & -0.7414 & -0.3074 \end{bmatrix}$$

$$\mathbf{h}'_1 = [\ -0.2774 \quad 0.3284 \quad 0.0511\ ] \qquad \mathbf{h}^{*\prime}_1 = [\ 0.7590 \quad -0.7198 \quad 0.0147\ ]$$

$$\mathbf{h}'_2 = [\ 0.6731 \quad 0.0549 \quad -0.3765\ ] \qquad \mathbf{h}^{*\prime}_2 = [\ -0.1915 \quad -0.4462 \quad 0.4424\ ]$$

$$\mathbf{h}'_3 = [\ -0.1296 \quad -0.1081 \quad -0.7653\ ] \qquad \mathbf{h}^{*\prime}_3 = [\ 0.6112 \quad -0.2832 \quad 0.8311\ ]$$

$$\mathbf{h}'_4 = [\ -0.3646 \quad -0.6176 \quad 0.6585\ ] \qquad \mathbf{h}^{*\prime}_4 = [\ 0.8462 \quad 0.2263 \quad -0.5927\ ]$$

$$\mathbf{T}_1 = \begin{bmatrix} 0.9677 & -0.2518 & 0.0154 \\ 0.2504 & 0.9661 & 0.0628 \\ -0.0306 & -0.0569 & 0.9979 \end{bmatrix} \quad \mathbf{T}_2 = \begin{bmatrix} 0.7114 & -0.1077 & 0.6945 \\ 0.2952 & 0.9426 & -0.1563 \\ -0.6377 & 0.3162 & 0.7024 \end{bmatrix}$$

$$\mathbf{T}_3 = \begin{bmatrix} 0.9017 & 0.1682 & 0.3983 \\ 0.0468 & 0.8779 & -0.4766 \\ -0.4298 & 0.4484 & 0.7837 \end{bmatrix} \quad \mathbf{T}_4 = \begin{bmatrix} 0.7713 & 0.5057 & -0.3865 \\ -0.2187 & 0.7809 & 0.5852 \\ 0.5977 & -0.3668 & 0.7129 \end{bmatrix}$$

stimulus weights $\mathbf{V}_j$

configuration	stimulus 1	2	3	4	5	6	7	8
1	0.0631	0.5037	0.4699	0.6363	0.6891	0.7728	0.5552	0.3727
2	—	0.0000	—	1.3249	0.9820	—	0.4398	—
3	—	0.2010	—	0.2779	—	0.9160	—	0.1146
4	0.6082	0.1348	—	0.5875	0.0763	0.6232	—	0.4711

Introducing missing data in the four configurations in Table 4.9, that is, treating rows 1, 3, and 6 of $\mathbf{X}_2$ as missing, rows 1, 3, 5, and 7 of $\mathbf{X}_3$ as missing, and rows 3 and 7 of $\mathbf{X}_4$ as missing, we obtained the following results. A generalized Procrustes analysis of these incomplete configurations yielded a loss of 0.7392 in 5 iterations. The STIMFREE algorithm converged in 19 iterations to a loss of 0.1070. The STIMIDIO algorithm of section 4.4.7 converged in 440 iterations, yielding a perfect

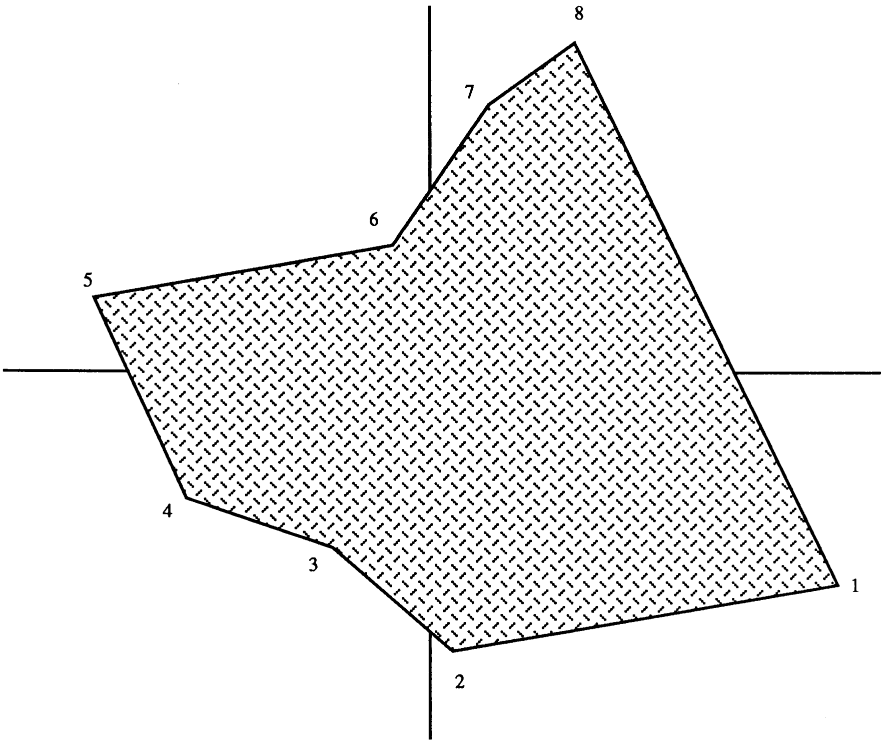

Figure 4.9 Plot of 'flattened' non-unique centroid configuration after STIMIDIO analysis of incomplete configurations.

solution of zero loss. The (non-unique) results of this latter analysis are given in Table 4.11, and Figure 4.9 contains a 'flattened' plot of the centroid configuration $\mathbf{Y}^*$.

Summarizing, these examples show that convergence of the STIMIDIO algorithm can be extremely slow, and that the algorithm easily converges to a local minimum. Because the parameters are also not uniquely determined, the STIMIDIO model is not a very attractive model, and we do not recommend its use.