PSYCHOMETRIKA—VOL. 55, NO. 1, 123–149
MARCH 1990

LEAST SQUARES METRIC, UNIDIMENSIONAL SCALING OF MULTIVARIATE LINEAR MODELS

KEITH T. POOLE

The squared error loss function for the unidimensional metric scaling problem has a special geometry. It is possible to efficiently find the global minimum for every coordinate conditioned on every other coordinate being held fixed. This approach is generalized to the case in which the coordinates are polynomial functions of exogenous variables. The algorithms shown in the paper are linear in the number of parameters. They always descend and, at convergence, every coefficient of every polynomial is at its global minimum conditioned on every other parameter being held fixed. Convergence is very rapid and Monte Carlo tests show the basic procedure almost always converges to the overall global minimum.

Key words: city block scaling, metric unfolding, constrained coordinates.

1. Introduction

The purpose of this paper is to show a general solution for the metric unidimensional similarities and unfolding problems when a squared error loss function is used. The coordinates can be simple, as in ordinary scaling, or they can be polynomial functions of exogenous variables, as in time series scaling. The basic approach is to transform the loss function from a continuous form to a discrete form which can be done by utilizing the ordering information of the coordinates. The key to the family of algorithms developed below is that the combinatorial problem that results from transforming the loss function can be conditionally solved by checking a small number of possibilities. Consequently, the algorithms are linear in the number of parameters. At every step the global minimum with respect to the parameter being estimated is found conditioned on the remaining parameters being held fixed. At convergence, every parameter is at its global minimum conditioned on all other parameters being held fixed. This strong form of local minimum is very rare and, as the number of parameters increases, Monte Carlo tests suggest that the likelihood of converging to the overall global minimum increases. In large problems, the algorithms almost certainly reach the overall global minimum.

Section 2 shows the simple conditional global minimum algorithm and how it is applied to similarities and unfolding problems. Evidence is also presented regarding the likelihood of the algorithm converging to the overall global minimum. Section 3 extends the algorithm to cover city block scaling and scaling problems in which the coordinates are polynomial functions of exogenous variables. Finally, section 4 presents a brief empirical example of an exploratory time series unfolding analysis using interest group ratings of members of the U.S. Congress. A proof that the algorithm always converges to a conditional global minimum is presented in the Appendix.

The author thanks Ivo Molenaar, three anonymous referees, and Howard Rosenthal for their many helpful comments. Requests for reprints should be sent to Keith T. Poole, Graduate School of Industrial Administration, Carnegie-Mellon University, Pittsburgh, PA 15213.

0033-3123/90/0300-9038 $00.75/0
© 1990 The Psychometric Society

2. Simple Metric Unidimensional Scaling

2.1 *The Similarities Problem*

In the metric unidimensional similarities and unfolding problems, the data ar assumed to be Euclidean distances plus some unknown observational error. If a standard squared error loss function is used, it is well known that estimating the coordinate consists of solving a combinatorial problem (Defays, 1978; Heiser, 1981; Hubert & Arabie, 1986). To illustrate, let z_j be the j-th ($j = 1, \ldots, q$) stimulus coordinate, let be the q length vector of coordinates, and let d^*_{jm} be the observed distance betwee stimulus j and stimulus m. Given these definitions, the standard squared error lo function for the metric unidimensional similarities problem is:

$$\mu = \sum_{j=1}^{q} \sum_{m=1}^{q} e^2_{jm} = \sum_{j=1}^{q} \sum_{m=1}^{q} (d^*_{jm} - d_{jm})^2,$$

where $d_{jm} = |z_j - z_m|$. Let $\delta^*_{jm} = +1$ if $z_j > z_m$ and $\delta^*_{jm} = -1$ if $z_j < z_m$. The case $z_j = z_m$ does not present a problem because de Leeuw (1984) has proven that at a loc minimum $d_{jm} = 0$ only if $d^*_{jm} = 0$ (see Appendix). This allows d_{jm} to be written as $\delta^*_{jm}($ $- z_m)$. Substituting into (1):

$$\mu = \sum_{j=1}^{q} \sum_{m=1}^{q} (\delta^*_{jm} d^*_{jm} + z_m - z_j)^2.$$

The loss given by (2) is exactly equal to that computed in (1) provided that th proper δ^*'s are used. An important feature of (2) is that, given **z**, the loss is minimiz by δ^*'s which are consistent with the ordering of the points. In other words, the lo given by (2) is always greater than or equal to that given by (1) provided the d*'s a *nonnegative*. The case of negative d^*'s will be taken up in section 3.

Since (2) is quadratic in z_j, the apparent solution is simply the centroid of the $+ \delta^*_{jm} d^*_{jm}$; that is, the estimated coordinate, $\hat{z}_j$ is:

$$\hat{z}_j = \frac{\sum_{m \neq j}^{q} [z_m + \delta^*_{jm} d^*_{jm}]}{q - 1}.$$

However, because the δ^*'s are defined in terms of the ordering of the z_j's, a minim must have the property that the δ^*'s produce $\hat{z}_j$'s which reproduce the same δ (Defays, 1978). This is a purely combinatorial problem. To see this, let Δ^* be the q q skew symmetric matrix of δ^*'s defined by the ordering relations of the $\hat{z}_j$'s, and le be the q by q skew symmetric matrix of δ's used to compute the $\hat{z}_j$'s. Without loss generality, the centroid of the z's can be set equal to zero which allows the loss functi to be written solely as a function of the ordering relations:

$$\mu = \sum_{j=1}^{q} \sum_{m=1}^{q} d^{*2}_{jm} + 2 \sum_{j=1}^{q} \frac{\left[\sum_{m=1}^{q} \delta_{jm} d^*_{jm} \right]^2}{q} - 4 \sum_{j=1}^{q} \sum_{m=1}^{q} \delta^*_{jm} d^*_{jm} \frac{\left[\sum_{m=1}^{q} \delta_{jm} d^*_{jm} \right]}{q}.$$

Any arbitrarily chosen Δ induces a Δ* with the corresponding squared loss given by (4). A minimum is defined as Δ = Δ*. In particular, note that when Δ = Δ*, the second and third terms of (4) become

$$-2\sum_{j=1}^{q}\frac{\left[\sum_{m=1}^{q}\delta_{jm}d^{*}_{jm}\right]^{2}}{q} \qquad (5)$$

so that, subject to the constraint that Δ = Δ*, minimizing (4) is equivalent to maximizing (5) (Defays, 1978; Heiser, 1981, p. 26).

Combinatorial optimization algorithms to find the *global* maximum of (5) are given by Defays (1978; using a branch and bound approach) and by Hubert and Arabie (1986; using a dynamic programming approach). Neither of these algorithms is linear in the number of parameters and can be practically applied only in small problems ($q \leq 20$). For example, with reference to (5), the Hubert and Arabie algorithm finds the maximum of the sum of squares over the index m for each value of j. For each j this involves checking every possible $q - 1$ length vector $\boldsymbol{\delta}_j$. When the optimal j is found, the problem is not only reduced by one parameter, but it is also known that one set of points is less than j and another set of points is greater than j (either set could be empty). Now, because of his partitioning, for each of the remaining $q - 1$ j's, every possible $q - 2$ length vector of δ's does not necessarily have to be checked. Consequently, although the number of computations required to solve for the optimal permutation is not linear in q, the algorithm does become computationally less burdensome as the order of successive points is identified.

I now show a *partial* combinatorial algorithm which is linear in the number of parameters. It converges to a solution in which every z_j is at a *global minimum* conditioned on every other point being held fixed. In addition, Monte Carlo testing suggests that the likelihood of the algorithm converging to the overall global minimum Δ matrix *increases as q increases* so that for large problems ($q > 20$), the algorithm is almost certain to have converged to the global minimum Δ matrix.

The basic structure of the algorithm is simple: $q - 1$ points are held fixed and a solution for the q-th point is found. I work with the loss function given by (2) and solve for the q-th point by using (3). By fixing $q - 1$ points, there are only q possible patterns of δ*'s corresponding to the possible spatial arrangements of the q points. (I will denote a δ* pattern as $\boldsymbol{\delta}^*$ where $\boldsymbol{\delta}^*$ is $q - 1$ length vector). It is a simple matter to try all q possible $\boldsymbol{\delta}$ and select the one with the minimum loss. This process is illustrated in Figure 1.

Figure 1 displays a five point example where z_1 through z_4 are fixed and z_5 is to be estimated. Letters A through E denote the 5 possible regions that z_5 could be located in and the corresponding 5 patterns of δ's are shown in the middle of the figure. Note that as z_5 is "moved" from one region to an adjacent one, only one δ changes. Computationally, this makes the search very efficient as only one term in the coordinate formula and in the loss function changes from one pattern to the next.

Note that, when all q possible $\boldsymbol{\delta}$ are tried, δ's are used in the loss formula, (2), instead of δ*'s. In terms of the definitions introduced above, this assumes that $\boldsymbol{\delta} = \boldsymbol{\delta}^*$ for each pattern. Obviously, this will not always be true; there will exist $\boldsymbol{\delta}$, which when used in (3), produces a coordinate such that $\boldsymbol{\delta} \neq \boldsymbol{\delta}^*$. However, when the d^* are nonnegative, the $\boldsymbol{\delta}$ pattern with the minimum loss will reproduce itself; that is, $\boldsymbol{\delta} = \boldsymbol{\delta}^*$. To see why this must be true, assume the opposite—that is, for $\boldsymbol{\delta}$ which produces the

Delta Patterns Produced By Locating z_5 In Regions A-E

z_1 z_2 z_3 z_4

	A*	B	C	D	E
z_1	-1	+1	+1	+1	+1
z_2	-1	-1	+1	+1	+1
z_3	-1	-1	-1	+1	+1
z_4	-1	-1	-1	-1	+1

***The column below each letter is the delta pattern produced b locating z_5 in the corresponding region.**

FIGURE 1

minimum loss, $\boldsymbol{\delta} \neq \boldsymbol{\delta}^*$. If $\boldsymbol{\delta} \neq \boldsymbol{\delta}^*$, using $\boldsymbol{\delta}^*$ in (2) *must reduce the loss* because this h the effect of changing the squared terms in (1) corresponding to the inconsistent δ from $(d^*_{jm} + d_{jm})^2$ to $(d^*_{jm} - d_{jm})^2$. Now, suppose $\boldsymbol{\delta}^*$ is used to compute a ne coordinate in (3). Consider the loss equation, (2). Note that $\hat{z}_j$ is the centroid of th $\delta^*_{jm} d^*_{jm} + z_m$. Consequently, the loss must again decrease when this new $\hat{z}_j$ is used (2). Hence, $\boldsymbol{\delta}^*$ must produce a lower loss than $\boldsymbol{\delta}$. This is a contradiction, hence $\boldsymbol{\delta} = \boldsymbol{\delta}$

Note also that, geometrically, the $z_m + \delta^*_{jm} d^*_{jm}$ terms in (2) and (3) are simp straight line equations. That is, the d^*_{jm} are vectors with the corresponding z_m as origi with directions given by the δ^*_{jm}. By this interpretation, (2) is the sum of squar distances from the points given by the line equations to their centroid.

Computationally, the algorithm proceeds as follows. First, the $q - 1$ points a rank-ordered. Second, the δ's are all set to +1—this corresponds to the assumption th the q-th point is to the right of the largest of the $q - 1$ points—and the loss, (2), an coordinate estimate, (3), are calculated and saved. Third, using the pattern in Figure as an example, the q-th point is assumed to lie in region D which corresponds to change of one δ from +1 to −1. This process is continued until the q-th point is to th left of the smallest point—region A in Figure 1. The loss and coordinate estimate for th current pattern of δ's can be gotten by one calculation using the values stored from th previous pattern. Because every possible pattern is checked in order, the result of th process is that the *global minimum* for the q-th z_j is found *conditioned* on the remaining points being held fixed.

Each point is estimated in a like manner. At convergence, the configuration can be improved still further by checking the effect of interchanging every *adjacent* pair of points (i.e., $q - 1$ calculations). The algorithm is linear, it always descends, and at convergence, *every point is at a global minimum conditioned on every other point being held fixed.*

In contrast, simple gradient-type algorithms can be guaranteed to descend only to minimums at which every point is at a *local* minimum conditioned on the remaining points being held fixed. In this regard, these two types of local minima can be characterized as "strong" and "weak" respectively. These terms are appropriate because, if an ordinary ("weak") local minimum is used as a starting configuration, the algorithm will *descend* to a conditional global ("strong" local) minimum. Strong minima are a subset of the weak minima and the *un*conditional (overall) global minimum is a subset (of size one—although it is possible, but not probable, that two solutions could have an identical loss) of the strong minima.

Because every point in a strong or conditional global minimum is at a global minimum conditioned on the other points being held fixed, the "jointness" of the constraints of the points on each other will increase geometrically with the number of points. Consequently, a reasonable conjecture is that the number of conditional global minimums should be a very small subset of the total number of minimums. In addition, I conjecture that as q increases the number of conditional global minimums as a *percentage* of the number of local minimums should *decrease*. For large values of q, conditional global minimums should be very rare.

Extensive Monte Carlo testing of the algorithm (a portion of these tests are shown in Table 1) suggests that there are two distinct classes of conditional global minimums. One class can be described as *interior*—that is, if multiple points in the "true" configuration are very close together, then, at high noise levels, several conditional global minimums may result. The other class can be described as *folded*—that is, points from the opposite ends of the configuration get placed next to one another. These two classes are very distinct and the folded minima typically produce a squared error loss at least twice as great as the interior minima.

In practice, it is easy to get from a folded minimum to an interior minimum. When the algorithm has converged to a conditional global minimum, a simple way to check if it is a folded minimum is to rerun the algorithm on a subset of the points (e.g., the first two-thirds, in rank order, from the estimated configuration). Then restart the algorithm with the new estimates of the subset substituted in the original configuration. If the original solution was folded, this process has the effect of "unfolding" one of the ends of the configuration. In practice I checked configurations by using the first two-thirds and second two-thirds. This procedure is linear in the number of points and almost always converges to the overall (*un*conditional) global minimum from any starting configuration.

Although any starting configuration can be used for the algorithm, excellent starts are obtained by an Eckart-Young (1936; E-Y) decomposition of the double centered distance matrix. For the 300 exhaustive-search experiments shown in Table 1, more than one conditional global minimum was found in 170 of the experiments. Using E-Y starts and the folding check procedure, the algorithms converged to the overall global minimum in 168 of the 170 (98.8%) cases of multiple conditional global minimums and, overall, 298 of the 300 experiments.

Table 1 shows the results of three different classes of experiments which utilized three different types of configurations and two different types of error. The top portion

Table 1
<u>Monte Carlo Results: CGM Algorithm</u>

q q !/2			q q !/2			q q !/2			q q !/2			
6 360			7 2520			8 20160			9 181440			
No. of Minima[a]	No. of C.G.M.[b]	No. of Int. C.G.M[c]	No. of Minima[a]	No. of C.G.M.[b]	No. of Int. C.G.M[c]	No. of Minima[a]	No. of C.G.M.[b]	No. of Int. C.G.M.[c]	No. of Minima[a]	No. of C.G.M.[b]	No. of Int. C.G.M[c]	Global Min Found[d]
Uniform Random Stimuli [-.5,+.5] and Uniform Error [-.5,+.5]												
76.2	1.64 (.0215)[e]	1.04	388.0	2.56 (.0066)	1.12	1467.6	3.16 (.0022)	1.28	8110.3	3.64 (.0004)	1.16	97.0% (65/67)
Uniform Random Stimuli [-1.0,+1.0] and Normal Variable Variance Error												
43.8	1.40 (.0320)	1.04	127.8	2.00 (.0157)	1.00	410.4	2.36 (.0057)	1.12	1974.2	2.20 (.0011)	1.12	100.0% (50/50)
Evenly Spaced Stimuli [-1.0,+1.0] and Normal Variable Variance Error												
56.3	1.52 (.0270)	1.12	210.4	1.72 (.0082)	1.24	632.8	2.00 (.0032)	1.12	2306.5	2.40 (.0010)	1.12	100.0% (53/53)

[a] Average number of minimums found for 25 experiments
[b] Average number of conditional global minimums found for 25 experiments
[c] Average number of interior conditional global minimums found for 25 experiments
[d] When there were multiple C.G.M.s this is the percentage of the multiple cases that the algorithm descended to the global minimum

of Table 1 shows the results of experiments in which the true configurations of points were drawn randomly from a uniform distribution over [−.5, +.5]. The random error, which was added to the corresponding true distances to produce the d^*'s, was also drawn from a uniform [−.5, +.5] distribution (in all cases, if a negative d^* was produced, it was replaced with its absolute value). In the middle portion of Table 1 the true configurations were drawn randomly from a uniform [−1, +1] distribution and the random error was drawn from a normal distribution with mean d_{jm} and variable variance $\sigma^2 d_{jm}^2$. Finally, in the bottom portion of Table 1 *every* true configuration is *evenly spaced* over [−1, +1] and the error was drawn from a normal distribution with mean d_{jm} and variable variance $\sigma^2 d_{jm}^2$. The same level of random error was used throughout all the experiments—the ratio of the standard deviation of the error to the standard deviation of the true distances was held at approximately 1.0.

There are $q!/2$ possible unique Δ matrices for the simple unidimensional metric similarities problem. This number is true assuming that the q by q matrix of d^*'s does not have any duplicate rows/columns so that the q points are distinct. For each value of q from 6 to 9, 25 configurations were generated for each of the three types of experiments described above and every $q!/2$ possible Δ matrix for each of the 25 configurations was checked in order to find every minimum. Table 1 shows, for each value of q and the three general types of experiments, the average number of total minimums, conditional global minimums, and interior conditional global minimums found using the folding check procedure. For example, for $q = 9$, an average of 8110.3 minimums were found for the model in which the points were drawn from a uniform [−.5, +.5] distribution with uniform [−.5, +.5] error. An average of 3.6 conditional global minimums and 1.2 interior conditional global minimums were found for the 25 experiments.

The proportion of minima that are conditional global is indeed very small (the numbers in parentheses) and the proportion declines as q increases. The number of conditional global minimums found tends to increase with q—however, the number actually *declined* from $q = 8$ to $q = 9$ for the random points normal variable variance experiments. More important however, is the pattern revealed in the evenly spaced stimuli experiments where the results are more comparable across values of q. Although the number of conditional global minimums increases slowly with q, the number of these that are *interior* conditional global minimums found with the folding check procedure is stable as q increases (1.1). Finally, as noted above, if the E-Y starting configuration is used, the global minimum is reached almost every time (the final column of the table).

To check the performance of the algorithm at larger values of q, 1000 randomly generated starting configurations were used and the results compared with the solution arrived at using the E-Y starting configuration with the folding check procedure. Ten experiments were performed for each value of q equal to 12, 14, 16, 18, 20, 25, 30, and 35 using evenly spaced points and normal variable-variance error at the same level as above. Using E-Y starts with the folding check procedure, the algorithm reached the minimum configuration found in the 1000 random starts in 79 of the 80 experiments. The one failure occurred for $q = 35$. However, in this one failure, the loss produced by the E-Y solution differed from the minimum loss by only .001. This set of experiments was repeated using random points and normal variable variance error and the global minimum was reached in all 80 experiments.

In terms of iterations, with E-Y starts, the algorithm typically takes only 4 to 5 iterations (where an iteration consists of estimating each of the q points as described above) to reach a conditional global minimum and, if the folding check procedure is used, another 4 to 5 iterations. Using random starts adds about 2 iterations. The algorithm can easily be run on any small desk top computer for problems up to size $q = 5$ (depending upon the type of micro-computer, 2 to 10 minutes is typical). I have an lyzed problems up to $q = 9750$ (however, about 70% of the matrix was missing data) o a Control Data CYBER 205 supercomputer (about 20,000 seconds were required).

Missing data poses no problem for the algorithm. For example, suppose d^*_{12} missing. When z_1 is estimated as described above, z_2 is not used, and vice vers Equations (1) through (4) could be reformulated in the manner of Heiser (1981)—that i let $w_{jm} = 0$ if d^*_{jm} is missing, $w_{jm} = 1$ otherwise, and multiply the squared error tem by the corresponding w_{jm}'s—but I will avoid doing so to simplify the presentation. T only limitation that missing data imposes on the algorithm is a practical one; namel there should be at least two d^*_{jm}'s for every z_j (although I recommend using at least fo to five).

2.2 The Unfolding Problem

Let x_i be the i-th ($i = 1, \ldots, p$) individual coordinate, $p > q$, let **x** be the p leng vector of individual coordinates, and let d^*_{ij} be the observed distance between indivi ual i and stimulus j. Following the approach used above, define $\delta^*_{ij} = +1$ if $x_i > z_j$ a $\delta^*_{ij} = -1$ if $x_i < z_j$. The case of $x_i = z_j$ does not present a problem because of Leeuw's theorem (see Appendix). Given these definitions, the loss function is:

$$\mu = \sum_{i=1}^{p} \sum_{j=1}^{q} (\delta^*_{ij} d^*_{ij} - x_i + z_j)^2.$$

Because (5) is quadratic in x_i and z_j the apparent solutions are

$$\hat{z}_j = \frac{\sum_{i=1}^{p} [x_i - \delta^*_{ij} d^*_{ij}]}{p} = \bar{x} - \frac{\sum_{i=1}^{p} \delta^*_{ij} d^*_{ij}}{p}$$

$$\hat{x}_i = \frac{\sum_{j=1}^{q} [z_j + \delta^*_{ij} d^*_{ij}]}{q} = \bar{x} - \frac{\sum_{i=1}^{p} \sum_{j=1}^{q} \delta^*_{ij} d^*_{ij}}{pq} + \frac{\sum_{j=1}^{q} \delta^*_{ij} d^*_{ij}}{q},$$

where $\bar{x}$ is the centroid of the x_i.

In the context of unfolding, Δ^* is the p by q matrix defined by the ordering relatio of the $\hat{x}_i$ and $\hat{z}_j$, and Δ is the p by q matrix used to compute the $\hat{x}_i$ and $\hat{z}_j$. This allo the loss function, (6), to be rewritten as a function of the ordering relations:

$$\mu = \sum_{i=1}^{p} \sum_{j=1}^{q} \left[\delta^*_{ij} d^*_{ij} + \frac{\sum_{i=1}^{p} \sum_{j=1}^{q} \delta_{ij} d^*_{ij}}{pq} - \frac{\sum_{j=1}^{q} \delta_{ij} d^*_{ij}}{q} - \frac{\sum_{i=1}^{p} \delta_{ij} d^*_{ij}}{p} \right]^2.$$

As with the similarities problem, any arbitrarily chosen Δ induces a Δ^* with the co responding loss given by (9). Again, a minimum occurs when $\Delta = \Delta^*$.

The algorithm for solving for the x_i and z_j is basically the same as that describ for the similarities problem. However, the unfolding version is somewhat simplier. F example, each x_i is estimated with respect to the fixed set of z_j. Consequently, there a only $q + 1$ possible $\boldsymbol{\delta}$ corresponding to the possible spatial arrangements of the qz_j points and x_i. Because the $q + 1$ $(p + 1)$ possible $\boldsymbol{\delta}$ are the same for each $x_i(z_j)$, the unfolding version of the algorithm is computationally simplier. In contrast, in the similarities version, the q possible $\boldsymbol{\delta}$ usually change as each z_j is estimated.

Given that $p > q$, there are $(p + q)!/2$ possible orderings of the x_i and z_j but only $(q!/2)(q + 1)^p$ possible Δ matrices. This results from the lack of information about within set orderings. For example, suppose p is large relative to q. As illustrated by Figure 1, there are $q + 1$ regions that each x_i could be in. Accordingly, there will be multiple x_i in at least one region. Any permutation of the x_i within that region produces the same Δ matrix.

Because the number of Δ matrices is so large even for small p and q, an exhaustive search for minima could not be performed. However, in Poole (1984) I developed a vector interpretation of the algorithm outlined above and performed a large Monte Carlo study of it. I found the algorithm to be very robust. It converged quickly to a solution regardless of the level of error and was highly accurate in reproducing the "true" coordinates even when purely random starting coordinates were used.

3. Complex Metric Unidimensional Scaling

This section shows how the conditional global minimum (CGM) algorithm outlined above can be extended to cover two large classes of problems—city block similarities/unfolding analyses; and scaling problems in which the coordinates are constrained to be polynomial functions of exogenous variables.

3.1 City Block Scaling

To simplify the exposition, I will analyze the city block unfolding problem and extend the results to the similarities problem where appropriate.

The loss function for the city block unfolding problem is

$$\mu = \sum_{i=1}^{p} \sum_{j=1}^{q} \left(d^*_{ij} - \sum_{k=1}^{s} |x_{ik} - z_{jk}| \right)^2, \tag{10}$$

where s is the number of dimensions. This would be a simple problem if the d^*'s were also indexed by dimension (i.e., d^*_{ijk}). Each dimension could be estimated separately using the CGM algorithm and the results simply added up. The layer of difficulty added by (10) is to somehow "divide" the d^*'s into s parts and estimate best fitting coordinates using the CGM algorithm from the "divided" d^*'s.

More formally, let d_{ijk} be the "observed distance" on the k-th dimension such that

$$d^*_{ij} = \sum_{k=1}^{s} d_{ijk} = \sum_{k=1}^{s} \delta^*_{ijk}(x_{ik} - z_{jk}) + \varepsilon_{ij} \tag{11}$$

where $\delta^*_{ijk} = +1$ if $x_{ik} > z_{jk}$, $\delta^*_{ijk} = -1$ if $x_{ik} < z_{jk}$, and ε_{ij} is the unknown observation error which is to be minimized. Let $l = 1, \ldots, s$, be an alternative index over the number of dimensions. Using these definitions, (10) can be rewritten into a form similar to (6)

$$\mu_l = \sum_{i=1}^{p} \sum_{j=1}^{q} \left(\delta^*_{ijl} \left[d^*_{ij} - \sum_{k \neq l}^{s} \delta^*_{ijk}(x_{ik} - z_{jk}) \right] - x_{il} + z_{jl} \right)^2$$

$$= \sum_{i=1}^{p} \sum_{j=1}^{q} (\delta^*_{ijl}[d_{ijl} + \varepsilon_{ij}] - x_{il} + z_{jl})^2. \tag{12}$$

Note that, because the partition of d^* into the s $d^{\sim}$ terms is arbitrary, the $s - 1$ $d^{\sim}$'s can be set equal to the corresponding $\delta^*(x - z)$ for $k \neq l$. This has the effect of loading the unobserved error onto the l-th dimension. The crucial difference between (12) and (6) is that the quantity $d_{\tilde{i}jl} + \varepsilon_{ij}$ must itself be estimated and it can be *negative* (Hubert & Arabie, 1988, p. 469). The presence of negative distances does not affect the basic structure of the CGM algorithm provided the mean of the distances is positive. If there are too many negative distances, then the best possible solution is a degenerate one—all the coordinates are equal to zero.

The presence of negative distances makes the algorithm computationally less efficient but still linear in the number of points. As explained above, the $z + \delta^*d^*$ or $x + \delta^*d^*$ terms in the loss expressions are straight line equations which define points. The coordinate being estimated is simply the centroid of these points and the loss associated with the $\boldsymbol{\delta}$ used to compute the points is the sum of the squared distances from the points to their centroid. If $\boldsymbol{\delta} \neq \boldsymbol{\delta}^*$, the loss computed in this fashion is larger than the actual loss computed directly as in (1). However, the minimum loss computed from (2) or (6) has the property that $\boldsymbol{\delta} = \boldsymbol{\delta}^*$. This technique breaks down if the d^*'s can be negative. In (2) and (6), the loss is *reduced* by using an inconsistent δ^* with a negative d^*. In short, with negative d^*, the geometry breaks down. The centroid of the points defined by the straight line equations is still the correct coordinate estimate, but now the loss associated with the coordinate cannot be computed from (2) or (6).

Obtaining the loss directly as in (1), is computationally burdensome for large problems. For example, suppose $\mathbf{x}$ is held fixed and the z's are to be estimated with the presence of some negative distances as in (12). First $\mathbf{x}$ must be rank ordered so that the $p + 1$ possible $\boldsymbol{\delta}$ patterns can be produced. Second, $\boldsymbol{\delta}$ is initialized to $+1$. Third, each of the $p + 1$ possible $\hat{z}_j$ are calculated using (7) and the loss (p calculations) for each is computed. This approach is not linear in p.

This process can be made more efficient by working with a loss function which is, so to speak, midway between (1) and (2). In particular, given $\hat{z}_j$ from (7), the loss can be written as

$$\mu_j = \sum_{i=1}^{p} (d^*_{ij} - |x_i - \hat{z}_j|)^2 = \sum_{i=1}^{p} (d^*_{ij} - \delta^*_{ij}(x_i - \hat{z}_j))^2$$

$$= \sum_{i=1}^{p} d^{*2}_{ij} + \sum_{i=1}^{p} x_i^2 + p\hat{z}_j^2 - 2 \sum_{i=1}^{p} \delta^*_{ij} d^*_{ij} x_i + 2\hat{z}_j \sum_{i=1}^{p} \delta^*_{ij} d^*_{ij} - 2\hat{z}_j \sum_{i=1}^{p} x_i, \tag{13}$$

where the δ^* are as defined above and d^* is used rather than $d^{\sim} + \varepsilon$ (as in (12)) for notational simplicity. Note that if $d^{\sim} + \varepsilon$ were used, the loss would be written as μ_{jl}—that is, the loss with respect to $\hat{z}_j$ on dimension l. The computational efficiency of (13) stems from the fact that the various sums involving the δ^*'s can be calculated separately from the $\hat{z}_j$'s and the two can be "put together" by a knowledge of what regions vis a vis the x_i's (as in Figure 1) the $\hat{z}_j$'s are in.

The calculations proceed as follows. First, $\mathbf{x}$ is rank ordered. Second, $\boldsymbol{\delta}$ is initialized to $+1$ and the sum of squared d^*'s, the sum of x_i, the sum of d^*x_i, and sum of squared x_i are computed. Third, the $p + 1$ possible $\hat{z}_j$ and $p + 1$ possible sum of cross product terms involving δ^* are calculated. Because $\boldsymbol{\delta}$ is initialized to $+1$'s, the remaining p possible $\Sigma\delta^*d^*$ and $\Sigma\delta^*d^*x_i$ sums can be gotten one at a time by subtracting in the order of the changes in sign of the δ_{ij}'s $- 2d^*$ and $-2d^*x_i$ respectively from Σd^* and Σd^*x_i. Fourth, the $\hat{z}_j$ are rank ordered. Fifth, the region vis a vis the x_i's for each $\hat{z}_j$ is determined—which determines the appropriate $\boldsymbol{\delta}^*$—and (13) can be calculated from the stored sums. Because both the x_i's and $\hat{z}_j$'s are rank ordered, it is a simple matter to determine the region $\hat{z}_j$ is in very quickly.

Because of the efficiency of widely available algorithms to sort real numbers, this algorithm, in terms of CPU time, is linear in p. However, it takes twice as long as the simple CGM algorithm which can be used when there are no negative distances. For ease of exposition, I will refer to the negative distance conditional global minimum algorithm as the NDCGM algorithm. As with the simple CGM algorithm, the NDCGM algorithm always descends, and at convergence, every point is at a global minimum conditioned on every other point being held fixed.

Given a partition of d^*_{ij} into the s d_{ijl}, then the NDCGM algorithm can be used to estimate the x_{ik} and z_{jk}. That is, given a partition, the algorithm will always produce a solution in which every coordinate is at a global minimum conditional on every other coordinate being held fixed. The problem is that this will be true of every partition of the d^*_{ij}. How then to partition the d^*_{ij}?

The method of partitioning that I will now describe is somewhat arbitrary but it works well in practice. To begin, all coordinates are initialized to zero and the d^*'s are divided by s to obtain target distances for the estimate of the first dimension coordinates; that is $d_{ij1} = d^*_{ij}/s$. The NDCGM algorithm is then applied to these $d^{\sim}$ and first estimates of the z_{j1} and x_{i1} are obtained. If $s = 2$, then I set $d_{ij2} = d^*_{ij} - |\hat{x}_{i1} - \hat{z}_{j1}|$, and if $s \geq 3$ then I set $d_{ij2} = d^*_{ij}/(s - 1) - |\hat{x}_{i1} - \hat{z}_{j1}|$. The NDCGM algorithm is then applied to these $d^{\sim}$ and first estimates of the z_{j2} and x_{i2} are obtained. In general, *on the first iteration*, if $1 < l \leq s$, then

$$d_{ijl} = \frac{d^*_{ij}}{(s + 1 - l)} - \sum_{k=1}^{l-1} |\hat{x}_{ik} - \hat{z}_{jk}|.$$

On subsequent iterations, if $1 \leq l \leq s$, then

$$d_{ijl} = d^*_{ij} - \sum_{k \neq l}^{s} |\hat{x}_{ik} - \hat{z}_{jk}|.$$

This procedure always converges to a solution for which the sum of squared error is the same for each dimension. That is, with respect to (12)

$$\mu_1 = \mu_2 = \cdots = \mu_s$$

Note that the fact that the sum of squared error is the same *with respect to the partitioning* of the d^*'s does *not* mean that the *variance of the estimated coordinates* is the same on each dimension.

Where the procedure departs from more traditional scaling methods is that *the starting coordinates for each dimension during every iteration are set to zero.* I impose this constraint because the target distances for any dimension which are analyzed by the NDCGM algorithm change at each iteration. In traditional scaling methods either the target distances remain constant throughout the iterative process or, as in Kruskal's (1964a, 1964b) algorithm, the target distances are altered in such a way that the sum of squared error decreases. In either case, it is appropriate to use the coordinates estimated during the previous iteration to begin the current iteration. Here, neither condition holds and the rationale is absent. I tried using previously estimated coordinates as starts for the current iteration and I found that they did not perform as well as always starting the NDCGM algorithm from the coordinates all at zero.

Table 2 shows a representative portion of a larger Monte Carlo study of the city block scaling method just outlined. The table is divided into two parts—the upper half reports experiments using eight different configurations of points for similarities data, and the lower half reports experiments using eight different configuration of points for unfolding data. The coordinates for the experiments were drawn randomly from a uniform [$-.5$, $+.5$] distribution and the d^*'s were generated by adding uniform [$-.5$, $+.5$] random error to the d's. If this produced a negative value for d^*, the absolute value was used.

As in the experiments shown in Table 1, the level of error was measured by the ratio of the standard deviation of the introduced error to the standard deviation of the true distances. Table 2 displays three different levels of error—zero, and standard deviation ratios of .45 and .85. Kruskal's Stress Formula 1 was used to measure the recovery of the d^*'s and the recovery of the true coordinates was measured by the Pearson r-square. A separate r-square is shown for each dimension. Each entry in the table is the average for 10 simulations under the indicated conditions. Standard deviations are in parentheses.

The primary message of Table 2 is that the data requirements for city block scaling are clearly much greater than standard Euclidean scaling. Reliable results at high levels of error for similarities data in two dimensions are obtained when the number of stimuli is greater than or equal to 20 and for three dimensions, it appears that at least 30 stimuli are required. At high levels of error for unfolding problems, at least 15 stimuli are required for reliable estimates in two dimensions and at least 25 are required for three dimensions.

3.2 *Scaling With Constrained Coordinates*

This section shows how the CGM algorithm can be extended to scaling problems in which the coordinates are constrained to be polynomial functions of exogenous variables. For example, suppose Ψ is a p by n matrix where the n columns of Ψ are measures of specific characteristics of p individuals—personal income, social class, race, sex, political party affiliation, and so on. Suppose further that we have the individuals' preferences for a set of stimuli and that we believe that these preferences are determined by the specific characteristics. This model is championed by economists of the "Chicago school". They believe that if the economic interests can be properly defined and measured then these variables will account for individual preferences (e.g., voting in legislatures—see Peltzman, 1984).

Given this theory of behavior, an appropriate model of the individual coordinates is

$$x_i = \chi_1 \psi_{i1} + \chi_2 \psi_{i2} + \cdots + \chi_n \psi_{in}, \tag{14}$$

where the χ's are the coefficients to be estimated. The squared error loss function for this problem is

$$\mu = \sum_{i=1}^{p} \sum_{j=1}^{q} (d_{ij}^* - d_{ij})^2 = \sum_{i=1}^{p} \sum_{j=1}^{q} \left(d_{ij}^* - \left| \sum_{m=1}^{n} \psi_{im} \chi_m - z_j \right| \right)^2. \tag{15}$$

In this form, there are $q + n$ parameters to be estimated—the q z's and n χ's. Given an estimate of the coefficient vector, χ, by holding χ fixed each z_j can be estimated

TABLE 2

MONTE CARLO RESULTS
SIMILARITIES DATA

				NO ERROR Recovery of				SDRATIO=.45 Recovery of				SDRATIO=.85 Recovery of		
p	q	s	d*	Coordinates			d*	Coordinates			d*	Coordinates		
	10	3	.052*	.839*	.805	.709	.118	.641	.588	.447	.189	.664	.577	.492
			(.017)	(.178)	(.171)	(.219)	(.020)	(.232)	(.253)	(.299)	(.028)	(.204)	(.202)	(.278)
	20	3	.054	.855	.794	.783	.158	.850	.800	.797	.253	.783	.663	.583
			(.050)	(.197)	(.250)	(.301)	(.023)	(.175)	(.195)	(.193)	(.025)	(.119)	(.223)	(.276)
	40	3	.014	.995	.975	.959	.161	.963	.935	.921	.276	.814	.808	.787
			(.035)	(.011)	(.073)	(.120)	(.016)	(.043)	(.131)	(.171)	(.023)	(.192)	(.181)	(.261)
	60	3	.000	1.000	1.000	1.000	.167	.974	.950	.949	.286	.926	.906	.894
			(.000)	(.000)	(.000)	(.000)	(.019)	(.013)	(.084)	(.099)	(.013)	(.029)	(.121)	(.109)
	10	2	.055	.901	.889		.168	.856	.782		.275	.701	.629	
			(.053)	(.132)	(.137)		(.043)	(.131)	(.212)		(.046)	(.230)	(.340)	
	20	2	.009	.994	.994		.183	.925	.885		.322	.838	.749	
			(.004)	(.006)	(.006)		(.032)	(.087)	(.169)		(.031)	(.083)	(.249)	
	40	2	.000	1.000	1.000		.174	.991	.990		.333	.952	.945	
			(.000)	(.000)	(.000)		(.007)	(.002)	(.002)		(.014)	(.016)	(.016)	
	60	2	.000	1.000	1.000		.181	.993	.993		.348	.967	.966	
			(.000)	(.000)	(.000)		(.005	(.002)	(.002)		(.009)	(.011)	(.012)	
							UNFOLDING DATA							
75	10	3	.049	.933	.891	.850	.111	.641	.632	.629	.257	.546	.464	.438
			(.032)	(.112)	(.138)	(.135)	(.011)	(.259)	(.165)	(.315)	(.010)	(.225)	(.230)	(.295)
75	15	3	.024	.964	.927	.899	.147	.717	.681	.548	.283	.644	.543	.542
			(.019)	(.064)	(.177)	(.180)	(.024)	(.189)	(.201)	(.263)	(.023)	(.211)	(.247)	(.180)
75	20	3	.017	.986	.984	.984	.148	.831	.817	.745	.252	.701	.600	.596
			(.022)	(.026)	(.023)	(.023)	(.015)	(.123)	(.150	(.213)	(.012)	(.170)	(.231)	(.261)
75	25	3	.012	.998	.991	.989	.152	.875	.792	.776	.226	.771	.725	.620
			(.016)	(.003)	(.015)	(.022)	(.019)	(.106)	(.191)	(.314)	(.018)	(.140)	(.201)	(.236)
75	10	2	.019	.982	.978		.192	.888	.848		.327	.746	.640	
			(.016)	(.018)	(.021)		(.035)	(.097)	(.225)		(.021)	(.132)	(.220)	
75	15	2	.013	.992	.990		.184	.967	.967		.343	.810	.799	
			(.016)	(.013)	(.023)		(.011)	(.014)	(.016)		(.024)	(.137)	(.181)	
75	20	2	.007	.998	.997		.170	.961	.950		.318	.792	.734	
			(.004)	(.004)	(.007)		(.015)	(.040)	(.054)		(.023)	(.196)	(.191)	
75	25	2	.007	.999	.999		.176	.980	.975		.338	.918	.801	
			(.008)	(.002)	(.003)		(.014)	(.011)	(.007)		(.025)	(.037)	(.236)	

*Entries are mean STRESS values. Standard deviations are in parentheses.

*Entries are mean Pearson r-square values. Standard deviations are in parentheses.

using the simple CGM algorithm. The z_j's are independent of each other in terms of the estimation—that is, when a particular z_j is estimated, the remaining $q - 1$ z's play no role in the estimation. In contrast, each χ is estimated by holding $\mathbf{z}$ and the remaining $n - 1$ χ's fixed. This is done as follows. Assume that $n - 1$ χ are known, then the corresponding $n - 1$ $\psi_{im}\chi_m$ terms can be combined with z_j and treated as a point. Specifically, let t index the χ that is to be estimated and define

$$y_{ijt} = z_j - \sum_{m \neq t}^{n} \psi_{im}\chi_m$$

so that

$$d_{ij} = \left| \sum_{m=1}^{n} \psi_{im}\chi_m - z_j \right| = \left| \psi_{it}\chi_t - y_{ijt} \right|.$$

Now, let $\delta^*_{ijt} = +1$ if $\psi_{it}\chi_t > y_{ijt}$ and $\delta^*_{ijt} = -1$ if $\psi_{it}\chi_t < y_{ijt}$. These definitions allow the loss function, (15), to be rewritten as

$$\mu = \sum_{i=1}^{p} \sum_{j=1}^{q} (\delta^*_{ijt} d^*_{ij} + y_{ijt} - \psi_{it}\chi_t)^2, \tag{16}$$

and the solution for χ_t is

$$\hat{\chi}_t = \frac{\sum_{i=1}^{p} \left[\psi_{it} \sum_{j=1}^{q} (\delta^*_{ijt} d^*_{ij} + y_{ijt}) \right]}{q \sum_{i=1}^{p} \psi_{it}^2}. \tag{17}$$

With Figure 1 I illustrated the simple geometry that underlies the CGM algorithm. In the simple unfolding problem, when the q z_j are held fixed, there are only $q + 1$ regions x_i could be in. Corresponding to these $q + 1$ possibilities are $q + 1$ $\boldsymbol{\delta}$ patterns and one of these $\boldsymbol{\delta}$ generates the conditional global minimum. The "target" x_i can be thought of as first being in region E of Figure 1—producing a $\boldsymbol{\delta}$ of all plus ones—and then being "moved" to region D—producing a pattern of $-1, +1, +1, \ldots, +1$—and so on until it is "moved" to region A and produces a $\boldsymbol{\delta}$ of minus ones. The key to the simplicity of the algorithm is the fact that the order of the changes in sign of the δ's is the same as the order of the regions. Thus when the $\boldsymbol{\delta}$ vectors are displayed in matrix form as in Figure 1 they form a perfect Guttman scale pattern—that is, the lower left triangle consists of minus ones and the upper right triangle consists of plus ones.

This simplicity of structure does not occur with (16) and (17). Although there is only one parameter, χ_t, being estimated, there are in fact p "targets" because χ_t is multiplied by the p ψ_{it}'s and it is the product, $\psi_{it}\chi_t$ which, of necessity, defines the ordering relations vis a vis the y_{ijt}. In this context, although there are pq y_{ijt} points, each target is being "moved" vis a vis only q of the points and generates q δ's—specifically, $\psi_{it}\chi_t$ generates $\delta_{i1t}, \delta_{i2t}, \ldots, \delta_{iqt}$. Consequently each $\boldsymbol{\delta}$ pattern has length pq—$q\delta$'s for each of the p targets—and there are $pq + 1$ possible $\boldsymbol{\delta}$ patterns corresponding to the $pq + 1$ regions defined by the pq y_{ijt}.

Because the ψ_{it} are constants that can be negative or positive, the initial $\boldsymbol{\delta}$ pattern may not be all plus ones. The initial pattern can be obtained by assuming that χ_t is a very large positive number such that for every $\psi_{it} > 0$, $\psi_{it}\chi_t$ is greater than the largest y_{ijt} and for every $\psi_{it} < 0$, $\psi_{it}\chi_t$ is smaller than the smallest y_{ijt}. In short, the p "targets" can be divided into two groups corresponding to the signs of the ψ_{it} and the initial $\boldsymbol{\delta}$ consists of +1's and −1's in direct correspondence to the signs of the ψ_{it}. Now, if $\psi_{it} = 0$, then the target, so to speak, never "moves". In this case a direct inspection of (16) shows that the loss will be minimized if $\delta_{ijt} = -1$ when $y_{ijt} > 0$, and if $\delta_{ijt} = +1$ when $y_{ijt} < 0$. The δ_{ijt}'s corresponding to ψ_{it}'s equal to zero remain fixed throughout.

Given the initial $\boldsymbol{\delta}$ pattern, the order of the changes in sign of the δ's must be determined. The initial pattern was determined by letting χ_t be a very large positive number. Now consider what happens when χ_t is slowly reduced in magnitude. For some value of χ_t it will be the case that for a particular ψ_{it} and y_{ijt}, $\psi_{it}\chi_t = y_{ijt}$, and the corresponding δ_{ijt} must be the first one to change in sign. In other words, the δ_{ijt} corresponding to the largest y_{ijt}/ψ_{it} ratio is the first one to change in sign. The order of the changes in sign of the δ's is obtained by rank ordering the pq y_{ijt}/ψ_{it} ratios. As explained above, if $\psi_{it} = 0$, then the corresponding δ_{ijt} never changes sign and the ratio need not be formed.

Given the initial $\boldsymbol{\delta}$ and the ordering information for the sign changes, the calculations are now very similar to those performed in the simple CGM algorithm. In particular, only one calculation is needed to obtain the new estimate of χ_t from (17). The loss function can be expanded out as in (13) and only two cross product terms—δd^*y and $\delta d^*\psi$—have to be changed. If the d^* are nonnegative, then, for the $\boldsymbol{\delta}$ with the minimum loss, $\boldsymbol{\delta} = \boldsymbol{\delta}^*$. If some of the d^*'s are negative, the NDCGM algorithm can be used as a subroutine. In either case, because all possible $\boldsymbol{\delta}$ patterns are checked, the global minimum for χ_t is found conditioned on all the remaining χ's being held fixed. The algorithm is linear in the number of parameters. In addition, the algorithm always descends, and at convergence, every χ_t is at its global minimum conditioned on the remainder being held fixed. I will refer to this algorithm as the polynomial coefficient conditional global minimum (PCCGM) algorithm.

In summary, the calculations of the PCCGM algorithm proceed as follows. First, the y_{ijt}, the sums of squared d^*'s, y_{ijt}'s, ψ_{it}'s and their appropriate cross products which appear in (16) and (17), are computed; and $\boldsymbol{\delta}$ is initialized according to the signs of the ψ_{it}'s. Second, the pq (minus the number of ψ_{it} equal to zero) y_{ijt}/ψ_{it} ratios are rank ordered to determine the order of the changes in sign of the initialized $\boldsymbol{\delta}$. Third, in order, change one δ_{ijt} at a time, update and store $\hat{\chi}_t$ from (16), and use the new $\hat{\chi}_t$ to calculate the loss in (15).

The PCCGM algorithm can also be used with similarities problems. Suppose d^*_{ijl} is the judged similarity between stimulus j and stimulus l by the i-th individual and Π is a q by n matrix where the n columns of Π are measures of specific characteristics of the stimuli. For example, in marketing a researcher may gather the judged similarities between a set of q products by p individuals and wish to test whether or not the similarity judgements are based upon objectively measured characteristic of the stimuli—e.g., sweetness, texture, color, etc. An individual differences approach to this problem is to define

$$z_{ji} = \chi_{i1}\pi_{j1} + \chi_{i2}\pi_{j2} + \cdots + \chi_{in}\pi_{jn},$$

which, for the i-th individual, produces the loss function

$$\mu = \sum_{j=1}^{q}\sum_{l=1}^{q}\left(d^*_{ijl} - \left|\sum_{m=1}^{n}(\pi_{jm} - \pi_{lm})\chi_{im}\right|\right)^2 .$$

The PCCGM algorithm can then be used to estimate the χ vector for each individual.

3.3 Time Series Scaling

This section shows how the PCCGM algorithm can be extended to scaling problems in which preferences/similarities are gathered over time—that is, three way type data problems (e.g., Carroll, Pruzansky, & Kruskal, 1980; DeSarbo & Carroll, 1984). For example, suppose p individuals report their preferences for q stimuli at r different times so that r distinct p by q matrices of preferential choice data are produced. Stated in its simplest form, the time series problem is: Can the coordinates of a stimulus/individual be assumed to be systematically related over time. If the answer is no, then, in the unidimensional unfolding context, $r(p + q)$ parameters must be estimated—the x_i's and z_j's for each of the r matrices. In order to answer yes, the researcher must state an explicit theory of behavior which "links" the coordinates together over time.

An example of such a theory of behavior is the "Chicago school" model used above. Suppose Ψ_t is the p by n matrix of economic characteristics at time t. Many of these economic characteristics will change over time—for example, personal income—and this can be the only source of change of individual behavior. Given this theory of behavior, an appropriate model of an individual's coordinates at time t is

$$x_{it} = \chi_1 \psi_{i1t} + \chi_2 \psi_{i2t} + \cdots + \chi_n \psi_{int}. \tag{18}$$

If the stimuli are such that they can be considered to be fixed through time, then the loss function for this problem is

$$\mu = \sum_{i=1}^{p} \sum_{j=1}^{q} \sum_{t=1}^{r} \left(d^*_{ijt} - \left| \sum_{m=1}^{n} \psi_{im}\chi_m - z_j \right| \right)^2. \tag{19}$$

The number of parameters to be estimated is the same as that for (15)—q z_j and the n χ's. The stimuli do not have to be constant through time. The stimuli can be functions of exogenous variables similar to (18). However, such a model must be founded on a model of behavior of the stimuli. In any case, the PCCGM algorithm can estimate the coefficients.

The "Chicago school" model is very restrictive and not appropriate in many circumstances. A less restrictive—albeit atheoretical—model of time series unfolding/similarities data is to assume that individual/stimuli coordinates are polynomial functions of time. This is a very useful exploratory technique. It is both flexible and restrictive. It is flexible in that the order of the polynomial function can be adjusted to accomodate a large class of systematic movements over time. It is restrictive in that the number of estimated parameters is typically much less than the number generated by performing separate yearly scalings. In particular, individual i's coordinate at time t is

$$x_{it} = \chi_{i0} + \chi_{i1}\psi_{t1} + \chi_{i2}\psi_{t2} + \cdots + \chi_{in_i}\psi_{tn_i}, \tag{20}$$

where $m = 0, 1, \ldots, n_i$ is the degree of the polynomial. I index n by individual because, empirically, the data will limit the number of terms that can be estimated—some individuals will be in a time series longer than others. This formulation allows a variety of representations of time; ordinary time, the log of time, Legendre polynomials, and so on. For example, using integers to denote the time periods $t = 1, \ldots, r$, $\psi_{t1} = t$, $\psi_{t2} = t^2$, $\psi_{t3} = t^3$, and so on. The first three terms of a Legendre polynomial representation of time are

$$\psi_{t1} = -1 + (t-1)\frac{2}{r-1} \qquad \text{for } t = 1, \ldots r$$

$$\psi_{t2} = \frac{3\psi_{t1}^2 - 1}{2}$$

$$\psi_{t3} = \frac{5\psi_{t1}^3 - 3\psi_{t1}}{2}.$$

I prefer Legendre polynomials for the time terms because they are *orthogonal* on the interval $[-1, +1]$. This property allows as much variance as possible to be picked up by the constant term (χ_{i0}) before the linear term (χ_{i1}) is estimated, and so on (Hinich & Roll, 1981). This orthogonality property makes them ideal for exploratory data analysis. In particular, the coefficients are comparable across individuals in empirical applications.

The counterpart to (20) for a stimulus coordinate at time t is

$$z_{jt} = \zeta_{j0} + \zeta_{j1}\psi_{t1} + \zeta_{j2}\psi_{t2} + \cdots + \zeta_{jn_j}\psi_{tn_j}. \tag{21}$$

Let $l = 0, 1, \ldots, n$ to be an alternative index of the polynomial degree and let it denote the χ or ζ coefficient being estimated. To perform a time series unfolding using (20) and (21), define

$$y_{ijtl} = \sum_{m=0}^{n_j} \psi_{tm}\zeta_{jm} - \sum_{m \neq l}^{n_i} \psi_{tm}\chi_{im},$$

so that

$$d_{ijt} = |x_{it} - z_{jt}| = \left| \sum_{m=0}^{n_i} \psi_{tm}\chi_{im} - \sum_{m=0}^{n_j} \psi_{tm}\zeta_{jm} \right| = |\psi_{tl}\chi_{il} - y_{ijtl}|,$$

where, for completeness, let $\psi_{t0} = 1$. I left ψ_{t0} out of (20) and (21) to simplify the presentation. Now, let $\delta_{ijtl}^* = +1$ if $\psi_{tl}\chi_{il} > y_{ijtl}$ and $\delta_{ijtl}^* = -1$ if $\psi_{tl}\chi_{il} < y_{ijtl}$. This allows the loss function to be written as

$$\mu = \sum_{i=1}^{p} \sum_{j=1}^{q} \sum_{t=1}^{r} (\delta_{ijtl}^* d_{ijt}^* + y_{ijtl} - \psi_{tl}\chi_{il}), \tag{22}$$

and the solution for χ_{il} is

$$\hat{\chi}_{il} = \frac{\sum_{t=1}^{r} \psi_{tl} \left[\sum_{j=1}^{q} (\delta_{ijtl}^* d_{ijt}^* + y_{ijtl}) \right]}{q \sum_{t=1}^{r} \psi_{tl}^2}. \tag{23}$$

The steps of the PCCGM algorithm are the same as those described above only here the ordering information is obtained from the pr ratios y_{ijtl}/ψ_{tl} and the **δ** patterns are of length pr. When the ζ's are being estimated, the corresponding **δ** patterns are of length qr. Note that when the constant terms, χ_{i0} and ζ_{j0}, are being estimated, the rati are simply the corresponding y_{ijtl}'s (because $\psi_{t0} = 1$) so that the estimation steps a equivalent to the simple CGM algorithm. The simple CGM algorithm is a special ca of the PCCGM algorithm which occurs when a column of Ψ is all ones.

Technically, the PCCGM algorithm can be implemented without starting values f the χ's and ζ's—that is, they can all be set equal to zero as in the NDCGM algorith As a practical matter, however, it is better to have starting estimates of either the x_{i} or z_{jt}'s—one of the nice features of the PCCGM algorithm is that only starting estimat for one set of points are needed to begin the iterative process. In practice, a straig forward way to obtain estimates of the z_{jt}'s is to perform an Eckart-Young (19 decomposition of the double centered average distance matrix; that is, the p by q matr of the average distances over the r time periods. This approach produces starting valu which are the same for every time period; that is, $z_{jt} = \zeta_{j0}$ for $t = 1, \ldots, r$. These a held fixed and the PCCGM algorithm is used to estimate each coefficient of each x_{it} ascending order—χ_{i0} then χ_{i1} then χ_{i2} and so on. The estimated value for χ_{i0} is use as shown above, to form the y_{ijtl} used in the estimation of χ_{i1}, then χ_{i0} and χ_{i1} are us to estimate χ_{i2}, and so on. The resultant x_{it}'s are held fixed and the coefficients for t z_{jt}'s are estimated in the same fashion. This completes an iteration. The sum of squar error always declines at every phase of this process. This implementation of t PCCGM algorithm was used in the empirical application shown below.

To illustrate the exploratory Legendre time series technique using the PCCG algorithm, an analysis of 21 yearly matrices of interest group ratings of members of t U.S. Congress is shown in section 4 and a Monte Carlo analysis of artificial data whi exactly duplicates the structure of the interest group data is shown in section 5. Sp cifically, not every interest group and member of Congress was present in every ye The artificial individuals/stimuli will be in the time series in exactly the same pattern their real counterparts. The aim of the Monte Carlo study is to check the reliability the estimates of the coordinates under a variety of possible error conditions. Becau an extensive set of Monte Carlo tests of the simple CGM algorithm (i.e., when $n =$ is shown in Poole (1984), the Monte Carlo tests of the PCCGM algorithm will confined to the linear ($n = 1$) and cubic ($n = 3$) models respectively.

4. An Application of the PCCGM Algorithm to Interest Group Ratings of Congre 1959–1981

Every year a wide variety of interest groups issue ratings of the members Congress. A rating is an agreement score between the member and the stated positio of the interest group. To rate a member of Congress, an interest group normally sele 10 to 50 roll call votes and computes the percentage of "correct" votes by the memb

The ratings can be regarded as preference data—the higher the rating, the more t group prefers the member. In terms of a geometric model, the higher the rating t smaller the distance between the member and the interest group on the underlyi evaluative dimensions. Accordingly, the ratings were converted to distances by app ing the linear transformation

$$d^*_{ijt} = \frac{100 - R_{ijt}}{50} = d_{ijt} + e_{ijt}, \qquad (2$$

where R_{ijt} is the rating of the i-th member by the j-th interest group at time t. Th division by 50 is simply a convenient scaling factor—it produces d^*'s which range fro 0 to 2 so that the recovered space ranges from approximately -1 to $+1$. A fu developed spatial model of the ratings and issues relating to (24) are discussed in detail in Poole and Daniels (1985).

There is substantial evidence that members of Congress vote consistently over long periods of time. Voting histories are very important and members pay close attention to how they voted in the past when deciding to vote (Asher & Weisberg, 1978; Clausen, 1973; Fenno, 1978; Fiorina, 1974). Because the ratings are based on the roll calls, it should be the case that the members' positions over time will be approximated by low order polynomials. Interest groups on the other hand, should be more stable than the members because of the relative narrowness of their policy concerns.

A total of 203,387 ratings were issued by 59 interest groups over the 1959–1981 period. However, 14 of the 59 are not really "groups", rather they represent various ratings compiled by *Congressional Quarterly*—these are primarily the well known Conservative Coalition and Presidential Support scores.

Very few of the interest groups are in the dataset for the entire 23 year period. The number of interest groups issuing rating ranged from a low of 8 in 1959 and 1960 to a high of 37 in 1979 (a complete listing of the groups is available from the author on request). If all 59 groups had issued ratings for all 23 years then the total number of ratings would be 725,995 (59 × 23 × 535). Given that the actual number of ratings is 203,387, this means that about 72% of the data is "missing".

A total of 261 senators and 1258 representatives are represented in the dataset. Consequently, $p = 1519$, $q = 59$, $r = 23$, and the model is estimated for $n = 0, 1, 2, 3$. Because of the size of the dataset and the need to do extensive Monte-Carlo work to verify the results of the estimation procedure, the program was implemented on a Control Data Cyber 205 supercomputer. A typical estimation takes 300 to 450 seconds depending upon n. The results of the estimations are displayed in Table 3.

Table 3

Unidimensional Time-Series Unfolding Results

	Degree of Polynomial	Pearson r-square	Total Parameters	Parameters Added
Constant	0	.740	1578	----
Linear	1	.765	2836	1258
Quadratic	2	.776	3957	1121
Cubic	3	.782	4954	997

Although it is the squared error that is being minimized, I use the Pearson r-square computed between the transformed ratings (the d^*_{ijt}'s) and the distances produced from the estimated coordinates as a measure of fit because it is easily interpretable. The r-square for the "constant" model ($n = 0$) in which legislators and interest groups are constrained to have the same coordinate in every year is .740. In contrast, the average

Table 4

Magnitude of Annual Change in Spatial Coordinate Given by Estimated Linear Trend Parameters

Group	In Sample 3 to 9 Years			In Sample 10 to 23 Years		
	Average#	Standard Deviation	Number	Average	Standard Deviation	Number
Interest Groups	.0301	.0368	27* (129)*	.0150	.0293	15 (231)
Senators	.0290	.0357	103 (545)	.0171	.0174	110 (1690)
Representatives	.0313	.0385	591 (3287)	.0194	.0181	414 (6267)
Legislators Entering Congress After 1959						
Senators	.0266	.0340	81 (436)	.0205	.0200	42 (600)
Representatives	.0282	.0329	423 (2369)	.0184	.0177	215 (3031)

*The numbers in parentheses are the total observations; that is, the number in the indicated group times the number of years.

#Average annual change in spatial coordinate.

of the r-squares of the separate yearly unfoldings is .807. As indicated in the table, the number of parameters estimated in the constant model is 1578 (i.e., $p + q$) while the number for the separate yearly unfoldings is 12,650. The increase of .067 in r-square is purchased at the price of 11,072 parameters.

In contrast, to move from the constant model to the linear ($n = 1$), quadratic ($n = 2$), and cubic ($n = 3$) models requires far fewer parameters. A linear term was estimated for every legislator with 3 or more years of interest group data. The figures for the quadratic and cubic terms were 4 and 5 years respectively. In terms of years, the requirements were set as low as possible to give the benefit of doubt to the higher order models. Note, however, that the ratio of parameters (χ's or ζ's) to ratings (d^*_{ijt}'s) *is always large*. That is, if a linear polynomial is fitted to a legislator who is only in the time series for three years the number of ratings is equal to the sum of the three years worth of interest groups. The smallest this number can be is 27—the sum of the q's associated with 1959, 60, and 61. The corresponding minimum numbers for the quadratic and cubic models are 37 and 48 respectively. Thus, although there is enough data to estimate the parameters reliably, fitting an n degree polynomial to $n + 1$ years does not satisfactorily address the *behavioral* question (is the individual/stimulus really moving through time in this way?).

On its face, the increase of .025 in the r-square from the constant to the linear model appears significant even though the number of parameters is increased by 1261.

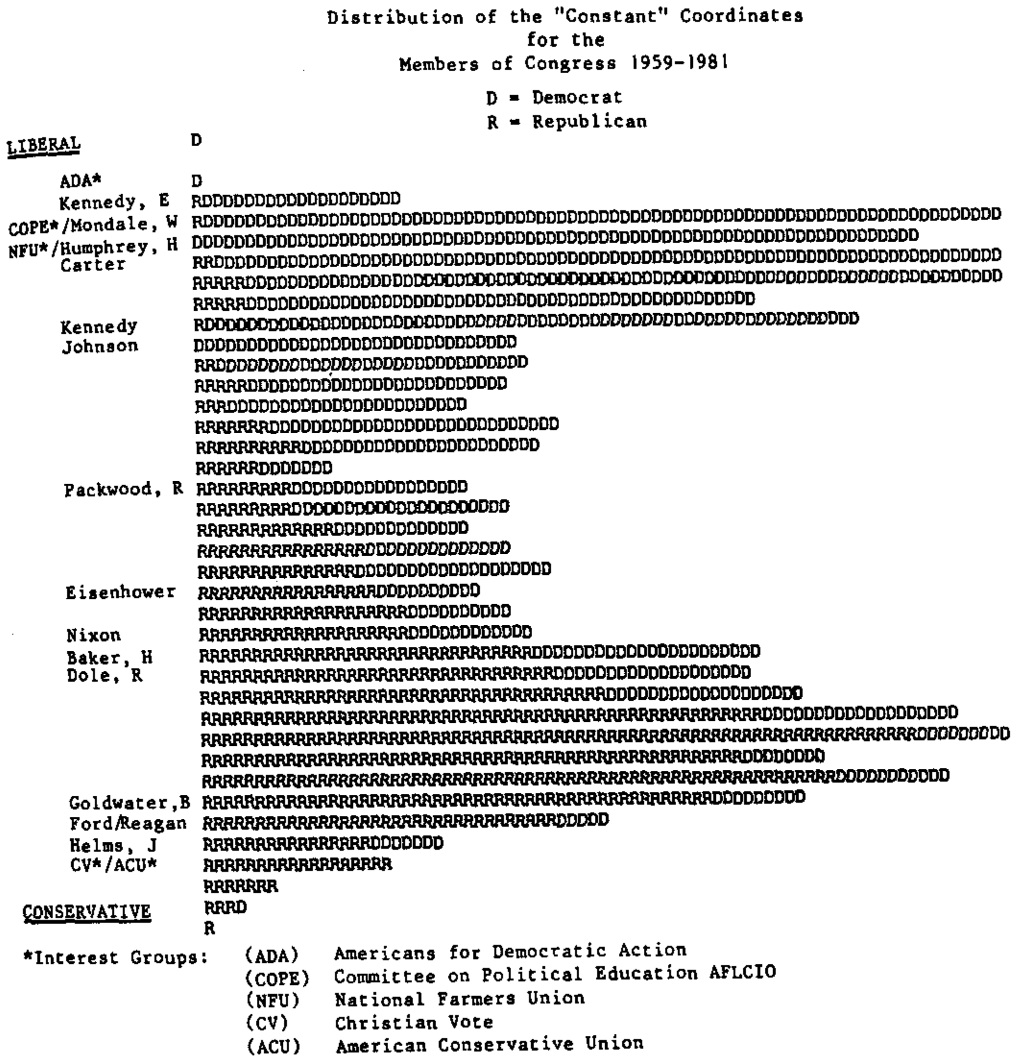

FIGURE 2

However, when viewed at the level of the individual legislator rather than the aggregate, the linear results appear less significant. Table 4 displays the mean and standard deviation of the absolute value of the yearly change in position implied by the linear coefficients. The table shows the results for those legislators and interest groups in the dataset for less than 10 years and those in the dataset 10 or more years. The results are striking. The legislators and groups in the dataset for shorter periods of time had substantially greater average yearly movements and the standard deviations of these average yearly movements are larger as well. Over short periods of time, the linear trend appears to be fitting noise rather than genuine trend. Over longer periods of time the interest groups, as expected, are more stable than the legislators—but not by much.

The results presented in the top portion on Table 4 are contaminated by the fact that many members of Congress were approaching the end of their careers in 1959 so that they are really not "short period" members. Consequently, in the bottom half of

Table 4, the calculations are redone using only those members who began their servi after 1959. The results are robust to this modification.

The dimension recovered from the interest group data is the liberal/conservati continuum familiar to students of politics (Kritzer, 1978; Poole, 1981; Poole Rosenthal, 1986). Figure 2 shows the distribution of the constant coordinates for t legislators over the dimension. For reference, the locations of a number of well-kno political figures and interest groups are indicated. On the whole, the interest grou tend to be more extreme than the legislators. The presidents, however, tend to located more towards the medians of their respective political parties.

5. Monte Carlo Tests

To perform the Monte Carlo tests I exactly duplicated the missing data pattern a coordinate structure of the 261 senators and the interest groups that rated them (t total number of ratings for the senators was 38,094). (The tests were limited to t senatorial dataset to make the Monte Carlo work more practical in terms of comput resources.) Thus, if a senator was in the data from 1973–78, then his /her artific counterpart was in the data for the same period.

The polynomial coefficients for the linear and cubic models were randomly gen ated by drawing them from a normal distribution with mean zero and variance of o As with the actual data, if a senator or interest group was in the dataset for 3 or mo years, a linear polynomial was estimated; if the period was 5 or more years, the cu polynomial was estimated. The d_{ijt}'s corresponding to the R_{ijt}'s in the real data we computed from the artificially generated x_{it}'s and z_{jt}'s, random error added, and t resulting d^*_{ijt}'s were converted into ratings using the inverse of the transformat shown in (24). If the addition of the error produced a negative d^* it was truncated zero. Similarly, if the inverse transformation produced a negative rating, it was tru cated to zero. The R_{ijt}'s were then fed into the program which produced the empiri results discussed in section 4.

Three types of random error were used in the Monte Carlo tests: normal with me equal to the true distance d_{ijt} with constant variance σ^2; normal with mean d_{ijt} wi variable variance of $\sigma^2 d^2_{ijt}$; and log normal with mean log (d_{ijt}) with constant varia σ^2. The log normal model is the most realistic because of its positive skewness (Ra say, 1977). Ten runs were made at three error levels for each model. The results shown in Table 5.

The entries in Table 5 are the mean Pearson r-squares for the ten runs in ea category. The standard deviations are shown in parentheses. The level of erro measured as the ratio of the standard deviation of the e_{ijt} to the standard deviatio the d_{ijt}. Under the Distances heading of the table the r-squares are computed betw the d^*'s and the d's computed from the estimated coordinates. Under the Coordina heading the r-squares are computed between the true x_{it}'s/z_{jt}'s and their estima counterparts. The number of R_{ijt}'s was 38,094 and the total number of coordinates 2684 ((261 + 59) × 23 − missing).

Each level of error is slightly more than double the error level below it. T Medium level is just above that encountered in the interest group data. At this level error, the recovery of the coordinates is excellent for both the linear and cubic mo els—the lowest r-square is .937 for the normal variable-variance model. At the hi level of error, which is about three times the error level in the interest group data, recovery of the coordinates is still fairly good—the r-squares for all three mod exceed .7. Consequently, a fair conclusion is that the coordinates recovered from interest group ratings are reliably estimated.

Table 5

Monte Carlo Results for Linear and Cubic Time-Series Models

	Linear					
	Distances			Coordinates		
Mean Error* Introduced	Normal	N.Var.Var	Log N.	Normal	N.Var.Var	Log N.
Low .35	.906# (.001)	.900 (.001)	.899 (.001)	.996 (.004)	.998 (.001)	.998 (.000)
Medium .80	.618 (.002)	.605 (.002)	.619 (.003)	.990 (.002)	.990 (.002)	.990 (.003)
High 1.75	.285 (.003)	.368 (.003)	.371 (.005)	.953 (.004)	.970 (.004)	.968 (.007)
	Cubic					
Low 35	.872 (.001)	.910 (.001)	.908 (.001)	.996 (.000)	.996 (.004)	.996 (.004)
Medium .80	.596 (.002)	.632 (.003)	.646 (.003)	.972 (.005)	.937 (.017)	.956 (.017)
High 1.75	.210 (.015)	.372 (.006)	.399 (.005)	.721 (.020)	.841 (.028)	.872 (.032)

* Ratio of the standard deviation of the error to the standard deviation of the true distances.

# Each entry is the mean Pearson r-square for 10 runs at indicated level of error. Standard deviations are in parentheses.

6. Conclusion

The purpose of this paper was to show a general approach to least squares, unidimensional scaling. The key to the approach is to use the ordering information contained in the parameters to transform the standard squared error loss function into a *discrete* as opposed to a continuous form. The resultant combinatorial problem can then be conditionally solved by evaluating a small number of possibilities given by the basic geometry of the parameters.

The family of unidimensional algorithms discussed in this paper all have the property that they always descend, and at convergence, every parameter is at its global minimum conditioned on every other parameter being held fixed. In addition, the algorithms are linear in the number of parameters. This strong form of local minimum reached by the CGM family of algorithms is rare and the likelihood of the overall global minimum being reached is very high regardless of the size of the problem. Finally, as the Monte-Carlo work shows, it reliably recovers randomly generated parameters even at high levels of both error and missing data.

Appendix

The purpose of this appendix is to show a proof that the family of conditional global minimum algorithms converge to a solution of the least squares metric unidimensional scaling problem in which every parameter is at its global minimum when every other parameter is held fixed. This boils down to proving that the global minimum is found for any arbitrarily chosen parameter when the remainder are held fixed. I will show this for the time series unfolding problem because the proofs for all the other models follow immediately by simply redefining the appropriate sets of parameters. In addition, I will assume that some of the observed distances can be negative—as with the NDCGM algorithm.

The squared error loss function for the time series model is

$$\mu = \sum_{i=1}^{p} \sum_{j=1}^{q} \sum_{t=1}^{r} w_{ijt}(d_{ijt}^* - d_{ijt})^2, \tag{A1}$$

where w_{ijt} is a missing data dummy variable; that is, $w_{ijt} = 0$ if d_{ijt}^* is missing, and $w_{ijt} = 1$ if d_{ijt}^* is not missing. The coordinate equations are

$$x_{it} = \chi_{i0}\psi_{t0} + \chi_{i1}\psi_{t1} + \cdots + \chi_{in_i}\psi_{tn_i} \tag{A2}$$

$$z_{jt} = \zeta_{j0}\pi_{t0} + \zeta_{j1}\pi_{t1} + \cdots + \zeta_{jn_j}\pi_{tn_j}, \tag{A3}$$

where $\psi_{i0} = \pi_{j0} = 1$, $n_i < r$, and $n_j < r$ for all i and j. Ordinarily, $\Psi = \Pi$ but I distinguish between the two for expository purposes. The case of only one time period, that is, $r = 1$, corresponds to simple unfolding or similarities analysis. If the i (j) index is dropped from the χ's (ζ's) and the t index on the ψ's (π's) is changed to i (j), then the individual (stimuli) coordinates are polynomial functions of exogenous variables as in the "Chicago school" economic variables model. Let $l = 0, 1, \ldots, n$ be an alternative index of the polynomial degree and let it denote the parameter being estimated. Define

$$y_{ijtl} = \sum_{m=0}^{n_i} \psi_{tm}\chi_{im} - \sum_{m \neq l}^{n_j} \pi_{tm}\zeta_{jm}, \tag{A4}$$

which allows the loss function to be rewritten as

$$\mu = \sum_{i=1}^{p} \sum_{j=1}^{q} \sum_{t=1}^{r} w_{ijt}(\delta_{ijt}^* d_{ijt}^* + y_{ijtl} - \pi_{tl}\zeta_{jl})^2, \tag{A5}$$

where $\delta_{ijtl}^* = +1$ if $\pi_{tl}\zeta_{jl} > y_{ijtl}$ and $\delta_{ijtl}^* = -1$ if $\pi_{tl}\zeta_{jl} < y_{ijtl}$. The case of $\pi_{tl}\zeta_{jl} = y_{ijtl}$ does not present a problem because de Leeuw (1984) has proven that at a local minimum $d_{ijt} = 0$ only if $d_{ijt}^* = 0$ (see below). In any case, suppose that $\pi_{tl}\zeta_{jl} = y_{ijtl}$ so that $d_{ijt} = 0$. For this particular i, j and t the squared error is $\delta_{ijtl}^{*2} d_{ijt}^{*2} = d_{ijt}^{*2}$ so that the total sum of squared error in (A5) corresponding to ζ_{jl} is the same for δ_{ijtl}^* equal to -1 or $+1$.

When $\boldsymbol{\delta}^*$ is fixed, the value of ζ_{jl} that minimizes the sum of the squared error is simply the weighted sum of the r time period centroids of the $y + \delta^* d^*$; namely

$$\hat{\zeta}_{jl} = \frac{\sum_{t=1}^{r} \left[\pi_{tl} \sum_{i=1}^{p} w_{ijt}(\delta_{ijtl}^* d_{ijt}^* + y_{ijtl}) \right]}{\sum_{i=1}^{p} \sum_{t=1}^{r} w_{ijt}\pi_{tl}^2}. \tag{A6}$$

For a fixed $\boldsymbol{\delta}^*$, (A5) is a convex function with respect to ζ_{jl} and has a unique minimum, $\hat{\zeta}_{jl}$, given by (A6). However, $\hat{\zeta}_{jl}$ may not produce the same $\boldsymbol{\delta}^*$ used to compute it. To keep this distinction clear, let $\boldsymbol{\delta}$ be the *rp* length vector used to compute $\hat{\zeta}_{jl}$ in (A6) and let $\boldsymbol{\delta}^*$ be, as above, the actual ordering induced by $\hat{\zeta}_{jl}$.

The key to the algorithm is expressing the loss function in terms of the δ's which has the effect of turning a continuous problem into a discrete problem. By construction, the loss expressed by (A5) is exactly equal to the loss expressed by (A1) for all values of ζ_{jl}. Every possible value of ζ_{jl} produces a $\boldsymbol{\delta}^*$. However, there can only be $rp + 1$ unique $\boldsymbol{\delta}^*$ patterns because there are only *rp* y_{ijtl}'s. This has the effect of dividing the range of ζ_{jl}—which is $-\infty < \zeta_{jl} < +\infty$—into $rp + 1$ regions and all values of ζ_{jl} within a specific region produces the same $\boldsymbol{\delta}^*$. These regions can be identified by their endpoints which are the *rp* ratios y_{ijtl}/π_{tl}. If $\pi_{tl} = 0$, then $\delta^*_{ijtl} = +1$ if $y_{ijtl} < 0$ and $\delta^*_{ijtl} = -1$ if $y_{ijtl} > 0$. If $y_{ijtl} = 0$ and $\pi_{tl} = 0$, then δ^*_{ijtl} can equal either $+1$ or -1—the loss in (A5) is the same either way. In any case, the δ^*_{ijtl} corresponding to π_{tl}'s equal to zero remain fixed in each of the $rp + 1$ $\boldsymbol{\delta}^*$ patterns. Consequently, there will not be $rp + 1$ *unique* $\boldsymbol{\delta}^*$ if some of the π_{tl}'s are equal to zero.

Now, in keeping with the distinction made above, when each one of these $rp + 1$ possible $\boldsymbol{\delta}^*$ is used as $\boldsymbol{\delta}$ in (A6), $rp + 1$ *possible* minimum values, the $\hat{\zeta}_{jl}$, for the loss function, (A5), are produced. I stress the word possible because each of the $rp + 1\hat{\zeta}_{jl}$ has an *implied* pattern of δ's, $\boldsymbol{\delta}^*$, which may not be consistent with the original pattern. Clearly, a minimum must have the property that $\boldsymbol{\delta} = \boldsymbol{\delta}^*$. In fact, as I will now demonstrate with Lemmas 1 through 3, $\hat{\zeta}_{jl}$ is a minimum of (A5) and (A1) if and only if $\boldsymbol{\delta} = \boldsymbol{\delta}^*$.

Lemma 1. If $\hat{\zeta}_{jl}$ from (A6) is a local minimum of (A5) and (A1), then $\boldsymbol{\delta} = \boldsymbol{\delta}^*$.

Proof. Assume that $\hat{\zeta}_{jl}$ is produced by $\boldsymbol{\delta}$ and is a local minimum of (A1). By construction, $\hat{\zeta}_{jl}$ and $\boldsymbol{\delta}^*$ produce the same loss in (A5) as $\hat{\zeta}_{jl}$ does in (A1). With fixed $\boldsymbol{\delta}^*$, (A5) is convex with respect to ζ_{jl} and has a unique minimum. Consequently, the $\hat{\zeta}_{jl}$ produced by $\boldsymbol{\delta}^*$ in (A6) must be the same as that produced by $\boldsymbol{\delta}$. This can only occur if $\boldsymbol{\delta} = \boldsymbol{\delta}^*$. □

Lemma 2 (de Leeuw). If $\hat{\zeta}_{jl}$ from (A6) is a local minimum of (A5) and (A1) and for some *i*, *j*, and *t*, $d_{ijt} = 0$, then $d^*_{ijt} = 0$.

Proof. By definition, $d_{ijt} = 0$ so that $\pi_{tl}\hat{\zeta}_{jl} = y_{ijtl}$. But if $d_{ijt} = 0$ then either $\delta^*_{ijtl} = +1$ or $\delta^*_{ijtl} = -1$ will produce the correct loss in (A5) regardless of the value of d^*_{ijt}. However, if $d^*_{ijt} \neq 0$ then $\hat{\zeta}_{jl}$ is not unique. This is a contradiction. Consequently, $d^*_{ijt} = 0$ and, by Lemma 1, $\boldsymbol{\delta} = \boldsymbol{\delta}^*$ for either $\delta^*_{ijtl} = +1$ or -1. □

Lemma 3. If $\boldsymbol{\delta} = \boldsymbol{\delta}^*$, then $\hat{\zeta}_{jl}$ from (A6) is a local minimum of (A5) and (A1).

Proof. For fixed $\boldsymbol{\delta}$, (A5) is a convex function with respect to ζ_{jl} and has a unique minimum. Therefore, since $\boldsymbol{\delta} = \boldsymbol{\delta}^*$, $\hat{\zeta}_{jl}$ is a minimum of (A5). Let ζ^- and ζ^+ denote the minimum and maximum values of ζ_{jl} such that $\zeta^- \leq \hat{\zeta}_{jl} \leq \zeta^+$ and $\boldsymbol{\delta}^*$ is produced by all values in the interval. The endpoints of a region which produces a $\boldsymbol{\delta}^*$, ζ^- and ζ^+, are equal to the appropriate y_{ijtl}/π_{tl} ratios. If $\zeta^- < \hat{\zeta}_{jl} < \zeta^+$, then $\hat{\zeta}_{jl}$ is a minimum of (A1) because, with fixed $\boldsymbol{\delta}^*$, (A5) and (A1) are exactly equal over the interval $[\zeta^-, \zeta^+]$ and $\hat{\zeta}_{jl}$ is a minimum of (A5). If $\hat{\zeta}_{jl} = \zeta^-$ or $\hat{\zeta}_{jl} = \zeta^+$, then for some *i*, *j*, and *t*, $d_{ijt} = 0$ and the corresponding δ^*_{ijtl} can equal to $+1$ or -1. By assumption, $\boldsymbol{\delta} = \boldsymbol{\delta}^*$ and this must be true for δ^*_{ijtl} equal to $+1$ or -1. This can only happen if $d^*_{ijt} = 0$. If $\hat{\zeta}_{jl} = \zeta^+$ let ζ^{++} denote the value of $\hat{\zeta}_{jl}$ such that $\hat{\zeta}_{jl} = \zeta^+ < \zeta^{++}$ and $\boldsymbol{\delta}^*$, with the exception of δ^*_{ijtl}, is produced by all values of $\hat{\zeta}_{jl}$ in the interval. However, since δ^*_{ijtl} is irrelevant to the calculation of the loss in (A5), (A5) and (A1) are exactly equal over the interval [ζ ζ^{++}]. Hence, if $\zeta^- < \hat{\zeta}_{jl} = \zeta^+ < \zeta^{++}$, then $\hat{\zeta}_{jl}$ is a minimum of (A5) and (A1). A simil argument can be made for the case of $\hat{\zeta}_{jl} = \zeta^{-1}$. If $\hat{\zeta}_{jl} = \zeta^- = \zeta^+$, then two d_{ijt}'s a equal to zero and the corresponding two δ^*_{ijtl}'s are irrelevant. In this instance, ζ^{--} $\zeta^- = \hat{\zeta}_{jl} = \zeta^+ < \zeta^{++}$ and the same result follows. Finally, if $\hat{\zeta}_{jl} = \zeta^+ = \zeta^{++}$ or $\hat{\zeta}_{jl}$ $\zeta^- = \zeta^{--}$, then, again, two d_{ijt}'s are equal to zero and the corresponding δ^*_{ijtl}'s a irrelevant. This analysis can be extended indefinitely until a region is identified su that (A5) and (A1) are exactly equal over the interval and $\hat{\zeta}_{jl}$ is not equal to eith endpoint of the interval.

The next two Lemmas concern the existence of minima. For the first, I need define some terms. Let $d^*_{\max}$ be the largest observed distance and let $y_{\max}$ and y_m denote the largest y_{ijtl} and smallest y_{ijtl} respectively. In addition, let $\mu(\cdot)$ denote th loss from using the parameters inside the parentheses in (A1).

Lemma 4. There exists no minima of (A1) outside the interval

$$[y_{\min} - d^*_{\max}, y_{\max} + d^*_{\max}].$$

Proof. Consider any $\pi_{tl}\zeta_{jl} > y_{\max} + d^*_{\max}$. By construction, every d_{ijt} comput using $\pi_{tl}\zeta_{jl}$ is larger than the corresponding d_{ijt} computed at the point $y_{\max} + d^*_m$ Consequently,

$$\mu(\pi_{tl}\zeta_{jl} > y_{\max} + d^*_{\max}) > \mu(\pi_{tl}\zeta_{jl} = y_{\max} + d^*_{\max}).$$

Furthermore, let ε be any arbitrarily small positive constant. By construction

$$\mu(\pi_{tl}\zeta_{jl} + \varepsilon) > \mu(\pi_{tl}\zeta_{jl} \geq y_{\max} + d^*_{\max}),$$

so that (A1) is strictly increasing beyond $y_{\max} + d^*_{\max}$. A similar argument holds for a $\pi_{tl}\zeta_{jl} < y_{\min} - d^*_{\max}$.

The next Lemma was proven by Defays (1978) for the constant coordinate met similarities problem.

Lemma 5. There exists at least one $\boldsymbol{\delta}$ such that $\boldsymbol{\delta} = \boldsymbol{\delta}^*$.

Proof. Since (A1) is continuous, by the Bolzano-Weierstrass theorem, it will ha at least one minimum over the interval $[y_{\min} - d^*_{\max}, y_{\max} + d^*_{\max}]$. By constructi a $\boldsymbol{\delta}$ always exists such that the loss in (A5) and in (A1) with respect to ζ_{jl} is identic By Lemmas 1, 2 and 3, $\boldsymbol{\delta} = \boldsymbol{\delta}^*$.

These results enable me to state the algorithm in the form of a theorem:

Theorem. The polynomial coefficient conditional global minimum (PCCGM) al rithm always finds the global minimum of any parameter of (A1) conditioned on remaining parameters being held fixed.

Proof. The algorithm computes the loss associated with each of the $rp + 1$ possi $\boldsymbol{\delta}$'s so that by Lemmas 1 through 5, all possible minimums are found.

References

Asher, Herbert B., & Herbert F. Weisberg. (1978). Voting change in congress: Some dynamic perspect on an evolutionary process. *American Journal of Political Science*, *22*, 391–425.

Carrol, J. Douglas, Pruzansky, Sandra, & Kruskal, Joseph B. (1980). CANDELINC: A general approach to multidimensional analysis of many-way arrays with linear constraints on parameters. *Psychometrika, 45*, 3–24.

Clausen, Aage. (1973). *How congressmen decide: A policy focus.* New York: St. Martin's Press.

Defays, D. (1978). A short note on a method of seriation. *British Journal of Mathematical and Statistical Psychology, 31*, 49–53.

de Leeuw, Jan. (1984). Differentiability of Kruskal's stress at a local minimum. *Psychometrika, 49*, 111–114.

DeSarbo, Wayne, & Carroll, J. Douglas. (1984). Three-way metric unfolding via alternating weighted least squares. *Psychometrika, 50*, 275–300.

Eckart, Carl, & Young, Gale. (1936). The approximation of one matrix by another of lower rank. *Psychometrika, 1*, 211–218.

Fenno, Richard F. (1978). *Home style: House members in their districts.* Boston: Little, Brown and Co.

Fiorina, Morris. (1974). *Representatives, roll calls, and constituencies.* Lexington, MA: Heath.

Heiser, Willem J. (1981). *Unfolding analysis of proximity data.* Leiden: University of Leiden, Department of Psychology.

Hinich, Melvin J., & Roll, Richard. (1981). Measuring nonstationarity in the parameters of the market model. *Research in Finance, 3*, 1–51.

Hubert, Lawrence, & Arabie, Phipps. (1986). "Unidimensional Scaling and Combinatorial Optimization." In de Leeuw et al. (Eds.), *Multidimensional data analysis.* Leiden: DSWO Press.

Hubert, Lawrence, & Arabie, Phipps. (1988). Relying on necessary conditions for optimization: Unidimensional scaling and some extensions. In *Classification and related methods of data analysis.* Amsterdam: North-Holland.

Kritzer, Herbert M. (1978). Ideology and American political elites. *Public Opinion Quarterly, 42*, 484–502.

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

Kruskal, Joseph B. (1964b). Nonmetric multidimensional scaling: A numerical method. *Psychometrika, 29*, 115–130.

Peltzman, Sam. (1984). Constituent interest and congressional voting. *Journal of Law and Economics, 27*, 181–210.

Poole, Keith T. (1981). Dimensions of interest group evaluation of the U.S. Senate, 1969–1978. *American Journal of Political Science, 25*, 49–67.

Poole, Keith T. (1984). Least squares metric, unidimensional unfolding. *Psychometrika, 49*, 311–323.

Poole, Keith T., & Daniels, R. Steven. (1985). Ideology, party, and voting in the U.S. Congress, 1959–80. *American Political Science Review, 79*, 373–399.

Poole, Keith T., & Rosenthal, Howard. (1986). The dynamics of interest group evaluations of Congress (GSIA Working Paper 1-86-87). Pittsburgh, PA: Carnegie Mellon University.

Ramsay, James O. (1977). Maximum likelihood estimation in multidimensional scaling. *Psychometrika, 42*, 241–266.

Manuscript received 10/20/86
Final version received 11/28/88