Journal of Classification 17:197-223 (2000)
DOI: 10.1007/s003570000019

Using Quadratic Assignment Methods to Generate Initial Permutations for Least-Squares Unidimensional Scaling of Symmetric Proximity Matrices

Michael J. Brusco

The Florida State University

Stephanie Stahl

Tallahassee, Florida USA

Abstract: Combinatorial solution procedures for least-squares unidimensional scaling of symmetric proximity matrices frequently consist of two integrated processes: (a) the identification of a permutation of objects, and (b) the estimation of coordinate values on the continuum. These procedures typically require an initial permutation of objects. It is generally known that their final unidimensional scaling solutions are often very sensitive to these starting permutations, particularly when the number of objects is large (> 20). This paper demonstrates that, relative to random starting permutations, substantial improvements in final seriation quality and computational efficiency can be realized by using starting permutations obtained via solution to a quadratic assignment problem (QAP). Three methods—locally-optimal pairwise interchange (LOPI), simulated annealing (SA) and a hybrid (LOPI-SA)—were evaluated regarding their effectiveness and efficiency for solving the QAP. The results revealed that SA and LOPI-SA efficiently provided very good QAP solutions that subsequently led to good least-squares solutions.

Keywords: Unidimensional scaling; Quadratic assignment; Simulated annealing; Seriation

---

Author's Address: Please send correspondence to Michael J. Brusco, IMS Department, College of Business, Florida State University, Tallahassee, FL 32306-1110, USA. Telephone: (850) 644-6512, FAX: (850) 644-8225; e-mail: mbrusco@cob.fsu.edu

1. Introduction

The metric unidimensional scaling of a non-negative symmetric proximity matrix, $\mathbf{P} = \{p_{ij}\}$, has traditionally focused on the placement of n objects along a continuum so as to minimize a measure of error between the proximity measures and the pairwise distances associated with the coordinates (Brusco 1999; de Leeuw and Heiser 1977, 1980; Defays 1978; Eisler 1973; Groenen 1993; Groenen and Heiser 1996; Groenen, Heiser, and Meulman 1999; Hubert 1974; Hubert and Arabie 1986, 1988; Hubert, Arabie, and Meulman 1997; Pliner 1996; Poole 1990; Simantiraki 1996). Perhaps the most common approach has focused on the establishment of coordinates $(x_1, x_2,\ldots,x_n)$ along a linear continuum with the objective of minimizing the following least-squares measure:

$$Min: Z_1 = \sum_{i<j} (p_{ij} - |x_i - x_j|)^2 \tag{1}$$

Using notation similar to De Soete, Hubert, and Arabie (1988), the Defays (1978) reformulation of this problem can be stated as:

$$\underset{\psi \in \Psi}{Max}: Z_2 = \sum_{k=1}^{n} (t_k^{(\psi)})^2 \tag{2}$$

where

$$t_k^{(\psi)} = (u_k^{(\psi)} + v_k^{(\psi)})/n \qquad \forall k = 1,\ldots,n;$$

$$u_k^{(\psi)} = \sum_{l=1}^{k-1} p_{\psi(k),\psi(l)} \qquad \forall k = 2,\ldots,n;$$

$$v_k^{(\psi)} = \sum_{l=k+1}^{n} p_{\psi(k),\psi(l)} \qquad \forall k = 1,\ldots,n-1;$$

$$u_1^{(\psi)} = 0, \quad v_n^{(\psi)} = 0;$$

and

Ψ = the set of all permutations, $\{\psi_1, \psi_2,\ldots,\psi_{n!}\}$, of objects $1,\ldots,n$;
$\psi(k)$ = the object in location k of the sequence ψ, $\forall k = 1,\ldots,n$.

Hubert, Arabie, and Meulman (1997) recently proposed a generalized version of (1) via the inclusion of a constant, c, as follows:

$$Min: Z_3 = \sum_{i<j} (p_{ij} + c - |x_i - x_j|)^2 \tag{3}$$

An important advantage of this generalized model is that it enables the removal of the non-negativity restrictions on the proximity measures. Hubert, Arabie, and Meulman (1997) also presented a model comparable to (3) that corresponds to a closed circular continuum. Citing the work of Plutchik and Conte (1997), Hubert, Arabie, and Meulman (1997) observed the fact that circular representations are prevalent in a wide spectrum of areas in psychology. Defining x_0 as the total length of the continuum, the circular unidimensional-scaling model they presented is:

$$Min: Z_4 = \sum_{i<j} (p_{ij} + c - \min\{|x_i - x_j|, x_0 - |x_i - x_j|\})^2 \qquad (4)$$

The combinatorial optimization problems associated with (1) – (4) are generally quite difficult to solve optimally for $n > 20$, and there has been considerable research interest in the development of efficient and effective heuristic procedures. Regardless of which of the four objective criteria described above is used, the essence of most heuristic solution approaches is an integrated, two-stage process. The first stage of the process is to identify a permutation or sequence of objects along the continuum, whereas the second stage focuses on the establishment of specific values on the coordinate axis. Local-search procedures may generate feasible permutations via pairwise interchange of objects (De Soete, Hubert, and Arabie 1988; Groenen 1993; Heiser 1989; Hubert, Arabie, and Meulman 1997), object-block rotations (Hubert and Arabie 1994; Hubert, Arabie, and Meulman 1997), and/or object insertions (Hubert and Arabie 1994; Hubert, Arabie, and Meulman 1997). For each permutation generated, the coordinate values must be established and the appropriate objective criterion value evaluated. For Z_2, the specification of coordinates can be conducted through the use of algebraic formulae (Defays 1978), whereas Hubert, Arabie, and Meulman (1997) described an iterative projection procedure for estimating the coordinates for Z_3 and Z_4. These two-stage procedures are known to be sensitive to the initial permutation that is manipulated so as to reduce the least-squares error in the objective function. In most instances, solution procedures have typically been implemented using 20 to 100 randomly generated starting permutations (Brusco 1999; De Soete, Hubert, and Arabie 1988; Groenen 1993; Hubert, Arabie, and Meulman 1997, Hubert and Schultz 1976; Pliner 1996). For large problems, the final permutations and their corresponding objective values, as well as the computer time required to obtain the final permutations, should be considerably improved through the use of better initial sequences. That is, a good starting permutation should improve the quality of the final

permutation and reduce the computational time required to converge to a near-optimal solution. Hence, a method for generating a starting permutation that improves the computational effectiveness and efficiency of solution methods for (1) – (4) is desirable.

This paper demonstrates that the solutions to a simpler seriation problem associated with the quadratic assignment problem (QAP) can provide very good starting permutations for least-squares unidimensional scaling algorithms. We present the results of a computational study that investigated the utility of QAP solution methods for efficiently improving initial random sequences. These improved sequences were subsequently passed to a pairwise interchange / block reversal / object insertion algorithm that generates solutions to the least-squares unidimensional scaling problem posed by (2). The results indicated that good starting sequences lead to much better final solutions for (2), as well as much faster convergence to solutions and thus require less computation time.

Linear and circular seriation problems corresponding to the QAP are briefly reviewed in Section 2. Section 2 also describes the heuristics we used to generate solutions to the QAP. An overview of the computational study and a brief description of the algorithm for solving (2) are presented in Section 3. The computational results for a set of large problems are also given in Section 3. The potential applicability of QAP methods for providing initial sequences for other least-squares objective criteria, such as (3) and (4), is addressed in Section 4 via demonstrations using two well-studied sets of proximity measures (Levelt, van de Geer, and Plomp 1966; Rothkopf 1957). Section 5 presents conclusions and avenues for further research.

2. Methodology

2.1 Quadratic Assignment Problem

The QAP was originally developed by Koopmans and Beckmann (1957) within the context of economic assignment of facilities. Since that time, it has it has received considerable attention in fields such as computer science (Taillard 1991), engineering (Steinberg 1961), and business (Armour and Buffa 1963; Nugent, Vollman, and Ruml 1968). The role of QAP in combinatorial data analysis was thoroughly developed by Hubert and Schultz (1976) and Hubert (1987, Chapter 4), who noted potential applications for hierarchical clustering, subset identification, linear and circular seriation, and the discrete placement of objects in multidimensional

space. The QAP has subsequently been deployed in a variety of psychological applications. For example, Holloway (1982), Margolin and Wampold (1981), and Wampold and Margolin (1982) have deployed the QAP as a method for independence testing in counseling and clinical psychology studies. Harris and Packard (1985) used QAP to compare intensity judgments of emotion words to a hypothesized structure. Gliner (1981) and Medina-Díaz (1993) have reported studies that used QAP models in the assessment of cognitive structure.

In this paper, solutions to seriation problems posed as QAPs are used as initial permutations for solution procedures for (2). Defining the pairwise distance between objects in locations k and l along a continuum as $d_{kl} = |\, k - l \,|$, the QAP can be described as:

$$\underset{\psi \in \Psi}{Max} : Z_5 = \sum_{k<l} d_{kl}\, p_{\psi(k),\psi(l)} \tag{5}$$

Hubert and Schultz (1976) originally presented (5) as a linear seriation problem and noted that the distance measure, $d_{kl} = |\, k - l \,|$, can be attributed to Szczotka (1972). This measure is also a specific case of the rectangular distance associated with the facility layout problem (Nugent, Vollman, and Ruml 1968). Hubert and Schultz (1976) also investigated a circular seriation problem using the following measure, which is also attributed to Szczotka (1972):

$$\text{If } n \text{ is even, then } d_{kl} = \left\{ \begin{array}{ll} |\, k-l \,| & \text{if } |\, k-l \,| \leq \dfrac{n}{2} \\ n - |\, k-l \,| & \text{if } |\, k-l \,| > \dfrac{n}{2} \end{array} \right\};$$

$$\text{if } n \text{ is odd, then } d_{kl} = \left\{ \begin{array}{ll} |\, k-l \,| & \text{if } |\, k-l \,| \leq \dfrac{n-1}{2} \\ n - |\, k-l \,| & \text{if } |\, k-l \,| > \dfrac{n-1}{2} \end{array} \right\}.$$

Solution procedures for the QAP posed by (5) should tend to separate dissimilar objects just as procedures designed for (1) – (4). That is, the maximization of (5) will tend to give greater distance in the sequence to pairs of objects with higher dissimilarity. It is important to note that we are not suggesting that the QAP objective criterion (5) is a superior measure to

the least-squares criteria in (1) – (4). Rather, we propose that the solution of the QAP (5) can provide an important supplement to the more computationally demanding least-squares methods by supplying a good initial permutation. The evaluation of alternative sequences for (5) can be conducted in an extremely efficient manner because there is no need to re-estimate coordinate locations when the sequence is modified. To illustrate, consider the case of a pairwise interchange of the objects in locations k and l. For the least-squares methods, this interchange would necessitate the use of formulae (in the case of (2)) or iterative projective mapping (in the cases of (3) and (4)) to specify new coordinate values and the resulting effect on the objective function. However, for the QAP objective posed by (5), the effect of the interchange of the objects in locations k and l on the objective function can be rapidly evaluated using the following equation:

$$\Delta Z_5^{(k,l)} = \sum_{m \neq k, m \neq l} (d_{mk} - d_{ml})(p_{\psi(m),\psi(l)} - p_{\psi(m),\psi(k)}) \tag{6}$$

2.2 Overview of Existing Solution Procedures

Optimal solution procedures for the QAP are primarily based on branch-and-bound algorithms, and Hubert and Schultz (1976) provided a review of some of the early methods in this area. During the past 25 years, there has been some advancement in the development of branch-and-bound methods for the QAP (Hahn, Grant, and Hall 1998), particularly with respect to the establishment of tighter lower bounds (Li, Pardalos, Ramakrishnan, and Resende 1994; Resende, Ramakrishnan, and Drezner 1995). Nevertheless, as Resende, Ramakrishnan, and Drezner (1995) have recently observed, obtaining optimal solutions to QAPs with n as small as 16 can still be extraordinarily difficult.

For large QAPs ($n > 20$), a variety of heuristic procedures have been offered. Many early solution procedures are based on the pairwise interchange strategy described in the previous subsection (Armour and Buffa 1963; Nugent, Vollman, and Ruml 1968). More recently, a wide variety of more sophisticated local-search methods have been developed and tested. These methods include simulated annealing (Burkard and Rendl 1984; Connolly 1990; Heragu and Alfa 1992; Laursen 1993; Wilhelm and Ward 1987), tabu search (Skorin-Kapov 1990; Taillard 1991), genetic algorithms (Ahuja, Orlin, and Tiwari 2000; Tate and Smith 1995), as well as hybrids of these procedures (Chiang and Chiang 1998; Gambardella, Taillard, and Dorigo, 1999).

In this paper, we evaluate the efficacy of three methods for solving the QAP posed by (5). The first method consists of replications of a locally-optimal pairwise interchange (LOPI) heuristic (Armour and Buffa 1963). The second method is a simulated annealing (SA) implementation comparable to those used by Wilhelm and Ward (1987) and Heragu and Alfa (1992). The third method is a hybrid approach that uses both LOPI and SA to develop QAP solutions. Each of the three methods was supplied with sets of initial permutations that were generated based on a uniform distribution. In this paper all uniform random variates were generated using a procedure suggested by Knuth (1997, Chapter 3). The three solution methods are described in the following subsections.

2.3 Locally-Optimal Pairwise Interchange (LOPI) Method

There are a variety of possible implementations for LOPI algorithms (see, for example, Groenen 1993, Chapter 4). The particular implementation used for the QAP in this paper is based on replications of the early facility layout procedure developed by Armour and Buffa (1963). We use 80 replications in our implementation. For each replication, the randomly generated initial permutation serves as the incumbent sequence, ψ^{I}, to initiate the LOPI algorithm. The algorithm consists of the following two steps.

Step 1: For the incumbent sequence, ψ^{I}, compute $\Delta Z_5^{(k,l)} \ \forall\ k = 1,\ldots,n-1;\ l = k+1,\ldots,n$. Let $\Delta Z_5^{\max} = \max_{k,l}(\Delta Z_5^{(k,l)})$ and let k' and l' indicate the locations that correspond to $\Delta Z_5^{\max}$.

Step 2: If $\Delta Z_5^{\max} \leq 0$, then store the incumbent permutation, ψ^{I}, and STOP. otherwise, interchange the objects in locations k' and l' to create a new incumbent permutation, ψ^{I}, and return to Step 1.

Step 1 consists of the evaluation of all possible pairwise interchanges of objects in the incumbent sequence. If none of these interchanges results in an improvement of the objective function, then the LOPI algorithm terminates in Step 2 by storing the incumbent permutation, ψ^{I}. Otherwise, the interchange that results in the greatest improvement in the objective function is made in the incumbent sequence and the procedure is repeated by returning to Step 1. The above algorithm places no bound on the number of iterations within a replication. Upon completion of 80 replications of the LOPI algorithm, a total of 80 permutations and their corresponding objective

function (5) values are stored. The permutations associated with the 20 largest (best) objective function values are subsequently passed to an algorithm designed to minimize (2).

2.4 Simulated Annealing (SA) Method

The SA algorithm has been successfully applied to solve QAPs for a number of years (Burkard and Rendl 1984; Connolly 1990; Heragu and Alfa 1992; Laursen 1993; Wilhelm and Ward 1987), and has also been tested by De Soete, Hubert, and Arabie (1988) within the context of least-squares unidimensional scaling. Simulated annealing provides a local-search process that allows for the probabilistic acceptance of inferior solutions. The probability is a function of the magnitude of the worsening of the objective function value, as well as the progress into the heuristic algorithm. As the progress into the solution algorithm increases, the probability of accepting an inferior solution decreases. The purpose of accepting an inferior solution is to dislodge a local optimum so as to investigate other local optima that might be preferable. Toward the completion of the SA procedure, the search process tends to converge to a local optimum rather than jump to other solution neighborhoods.

Within the context of combinatorial optimization, SA terminology tends to reflect its origin in statistical mechanics (Metropolis, Rosenbluth, Rosenbluth, Teller, and Teller 1953). The *temperature*, T_h, controls the probability of accepting an inferior solution, and the *cooling factor*, r ($0 < r < 1$), controls the rate at which the probability decreases. The *temperature length*, *TL*, represents the maximum number of solutions that will be evaluated at any given temperature. During the execution of the annealing algorithm, a generation mechanism is applied to obtain a neighboring permutation, ψ^J, of the current incumbent permutation, ψ^I. Consistent with previous research (Heragu and Alfa 1992; Wilhelm and Ward 1987), the generation mechanism deployed herein consists of a pairwise interchange of objects in two randomly selected locations, k and l. If this interchange improves the objective function value, then ψ^J replaces ψ^I as the incumbent permutation (i.e., object locations are interchanged). Otherwise, the probability of ψ^J replacing ψ^I as the incumbent permutation is computed as, $e^{\Delta Z_5^{(k,l)}/T_h}$. If the number of solutions evaluated at the current temperature is equal to *TL*, or if the number of accepted inferior solutions at the current temperature equals a maximum limit, *nlimit* (see Heragu and Alfa 1992), then the temperature is reduced by the cooling factor. The total number of

temperature reductions, *TR*, controls the number of times that the annealing schedule will be implemented.

We used 20 replications of the SA algorithm in our implementation. For each replication, the initial randomly generated permutation served as the incumbent sequence, ψ^I, to initiate the SA algorithm. The details of the algorithm are as follows:

Step 0: Initialize $\psi^B = \psi^I$, *TR*, *TL*, *nlimit*, T_1; r, $h = 1$.

Step 1: If $h > TR$, then return ψ^B and STOP. Otherwise, set $nsol = 0$ and $nacc = 0$, and go to Step 2.

Step 2: If $nsol = TL$ or $nacc = nlimit$, then set $h = h + 1$, set $T_h = rT_{h-1}$, and go to Step 1. Otherwise, go to Step 3.

Step 3. Select a location k based on a uniform distribution and a second location l, $l \neq k$, based on a uniform distribution. Compute $\Delta Z_5^{(k,l)}$ using (6), set $nsol = nsol + 1$, and evaluate the solution using the following logic;

Step 3a: If $\Delta Z_5^{(k,l)} \geq 0$, then interchange objects in locations k and l to obtain new incumbent sequence ψ^I with objective function $Z_5(\psi^I)$; go to Step 4.

Step 3b. Generate a uniform random variate, μ. If $\mu \leq e^{\Delta Z_5^{(k,l)}/T_h}$, then interchange objects k and l in the incumbent sequence ψ^I, update $Z_5(\psi^I)$, and set $nacc = nacc + 1$. Go to Step 2.

Step 4: If $Z_5(\psi^I) > Z_5(\psi^B)$, then set $\psi^B = \psi^I$. Go to Step 2.

Step 0 of the SA algorithm initializes parameter values and counters. The incumbent permutation is stored as the best solution, ψ^B. In Step 1, a test is made to ensure that the maximum number of iterations of the annealing schedule (*TR*) will not be exceeded. Heragu and Alfa (1992) have observed that smaller values of *TR* (say 10 to 50) are somewhat ineffective for large QAPs and this motivated our selection of a larger value. For our study, the number of temperature reductions was specified at $TR = 240$. In Step 2, two possible criteria are tested for termination of solution evaluations at the current temperature. Specifically, tests are performed to evaluate whether the number of solutions evaluated at the current temperature equals the temperature length (i.e., $nsol = TL$) and whether the maximum number of accepted inferior solutions at any temperature has been attained (i.e., *nacc* =

nlimit). If either termination criterion is met, then the outer loop counter, h, is incremented, the temperature is reduced, and control returns to Step 1. Consistent with Heragu and Alfa (1992), we used parameter settings of $TL = 100n$ and $nlimit = 10n$.

In Step 3 of the SA algorithm, two locations are randomly selected and the effect on the objective function value of interchanging the objects in those locations is computed using (6). If this interchange is beneficial to the incumbent permutation (ψ^I), then the locations of the objects are interchanged in the incumbent permutation (Step 3a) and control is subsequently passed to Step 4. If the interchange investigated in Step 3 would result in a permutation with a lower (worse) objective value than the incumbent permutation, ψ^I, then a decision regarding the acceptance of this inferior solution is based on a probability distribution, as shown in Step 3b. This probability is function of both the magnitude of the negative effect on the objective criterion, as well as the current temperature, T_h. During the early stages of the algorithm, a rather high probability of accepting an inferior solution is desirable (Heragu and Alfa 1992; Wilhelm and Ward 1987). Heragu and Alfa (1992) used an arbitrary initial temperature of 999.0. However, for test problems with large objective values, this value does not allow for a very high probability of accepting many of the inferior solutions. We concluded that it was best to determine the initial temperature using problem-specific information. Therefore, for the initial random permutation, we computed $\Delta Z_5^{(k,l)} \; \forall \; k = 1,\ldots,n\text{-}1; \; l = k\text{+}1,\ldots,n$. We subsequently set $T_1 = \left|\min_{k,l}(\Delta Z_5^{(k,l)})\right|$. Thus the value of T_1 was set equal to the value corresponding to the most detrimental pairwise interchange of objects in the initial random permutation. For $h = 1$, the probability of accepting an interchange associated with this observed worst case would be $e^{-1} \approx .37$. As noted above, the temperature is gradually reduced by the cooling factor, r, prior to initiating the next iteration of the outer loop. The relatively large value of TR enabled us to use a slower cooling process and hence our value of $r = .95$ is slightly larger than the value of .9 which has been used in previous studies (Heragu and Alfa 1992; Wilhelm and Ward 1987).

Step 4 ensures the proper updating of the permutation associated with the largest corresponding objective function (5) value identified by the SA algorithm. If the improved incumbent permutation (ψ^I) yields a better objective function value than the best permutation (ψ^B), then the incumbent permutation is stored as the best permutation identified thus far in the SA

algorithm. The final permutations associated with the 20 replications of the SA algorithm are subsequently passed to an algorithm designed to minimize (2).

2.5 A Hybrid Method Integrating LOPI and SA (LOPI-SA)

Hybrid solution procedures for the QAP have become increasingly popular in recent years (Chiang and Chiang 1998; Gambardella, Taillard, and Dorigo 1999; Heragu and Alfa, 1992). We developed a hybrid method (LOPI-SA) that uses both LOPI and SA to generate solutions to QAP. We used 20 replications of this LOPI-SA algorithm in our implementation. Like the LOPI and SA algorithms, the initial random permutation for each replication serves as the incumbent sequence, ψ^I, to initiate the LOPI-SA algorithm. The initial permutation is passed to the LOPI algorithm (as described in Section 2.3). Upon termination of the LOPI algorithm, the incumbent sequence is passed to the SA algorithm (as described in Section 2.4) to see if it can be further improved. The SA algorithm is applied as described in Section 2.4, except that three parameter settings are modified. First, we determined that running the LOPI method prior to SA might provide some additional information that could be used to set the initial temperature. Therefore, in the LOPI-SA algorithm, we set T_1 equal to the absolute value of the worst (smallest) value of (6) observed across the entire execution of LOPI. The second change was that the number of temperature reductions was set at $TR = 100$ to provide a more equitable CPU time comparison between SA and LOPI-SA. Accordingly, the third change involved the reduction of r from .95 to .9. The parameter settings of $TR = 100$ and $r = .9$ are the same as those used by Heragu and Alfa (1992) in their hybrid SA algorithm.

Upon completion of the SA algorithm, a final combinatorial polishing step is implemented. As suggested by De Soete, Hubert, and Arabie (1988), the best permutation, ψ^B, is passed to a LOPI routine, which ensures that the final permutation is locally-optimal with respect to all pairwise interchanges. The final permutations associated with the 20 replications of the LOPI-SA algorithm are subsequently passed to an algorithm designed to minimize (2).

2.6 Benchmarking the QAP Solution Methods

In Sections 3 and 4 of this paper, we investigate the efficacy of using QAP solution methods to provide good starting permutations for least-squares unidimensional scaling algorithms. However, prior to this

investigation, we wanted to ensure that our proposed methods were actually capable of providing good solutions to the QAP. The LOPI, SA, and LOPI-SA algorithms were written in Fortran Powerstation 1.0 and implemented on a 400 MHz Pentium II PC in a Windows 98 environment. We tested the methods on two of the most popular sets of test problems from the QAP literature. Each of these test problems was originally posed as a minimization problem. Because our methods were designed to maximize (5), we simply pre-multiplied the proximity matrices by –1 to convert them to maximization problems. The first set of problems were the $n = 15$, $n = 20$, and $n = 30$ problems from Nugent, Vollman, and Ruml (1968), whereas the second set of problems were originally developed by Skorin-Kapov (1990) and range in size from $n = 42$ to $n = 90$. These test problems, and a variety of others, can be obtained from QAPLIB – A Quadratic Assignment Problem Library (Burkard, Karisch, and Rendl, 1997) – which is maintained on websites at the Technical University Graz in Austria and the Technical University of Denmark. We obtained the problems from the Graz website at <www.opt.math.tu-graz.ac.at>. In addition to the problem data sets, the website also contains the best-known objective-function values that we used as a benchmark.

The results for our QAP solution methods are reported in Table 1, along with the best-known objective-function values and the best solution obtained by Heragu and Alfa's (1992) SA implementation. Direct comparisons of CPU times for the best-known solutions (or for Heragu and Alfa's (1992) methods) are not possible because of differences in hardware and software platforms. However, many of the best-known solutions were obtained using computationally intensive parallel implementations of tabu search (Taillard 1991), which require considerable storage and CPU time. The results for the LOPI (400 replications) method were, not surprisingly, rather unimpressive. Although LOPI is very efficient for small problems, its computation time grew rapidly for the larger Skorin-Kapov problems, and its solution quality was consistently inferior to SA and LOPI-SA. These latter two methods (both of which used 20 replications) performed very well. Both SA and LOPI-SA identified the optimal (or best-known) solution for each of the Nugent, Vollman, and Ruml (1968) test problems. LOPI-SA obtained better solutions than Heragu and Alfa (1992) for 6 of the 7 Skorin-Kapov test problems. SA yielded better solutions than those reported by Heragu and Alfa (1992) for each of the seven test problems, and provided the best known solution for the test problems with $n = 42$ and $n = 56$. SA obtained better solutions than LOPI-SA for 5 of the 7 Skorin-Kapov test problems. The departure from the best-known solutions was quite small for

Table 1. Performance comparison of LOPI, SA, and LOPI-SA solutions with the best-known solutions and Heragu and Alfa solutions for the Nugent, Vollman, and Ruml (1968) and Skorin-Kapov (1990) QAPs. These QAPs are minimization problems and thus smaller values indicate better solutions.

	Objective Function Values					Total CPU Seconds***		
	Best Known*	Heragu & Alfa**	LOPI	SA	LOPI-SA	LOPI	SA	LOPI-SA
Nugent *et al.* (1968) $n = 15$	575	575	575	575	575	1.59	36.97	27.33
Nugent *et al.* (1968) $n = 20$	1285	1285	1297	1285	1285	2.69	53.61	37.57
Nugent *et al.* (1968) $n = 30$	3062	3062	3088	3062	3062	11.87	94.15	61.18
Skorin-Kapov (1990) $n = 42$	7906	7927	7994	7906	7909	49.76	161.15	100.35
Skorin-Kapov (1990) $n = 49$	11693	11739	11771	11709	11707	128.36	228.71	143.30
Skorin-Kapov (1990) $n = 56$	17229	17236	17343	17229	17262	262.26	314.61	196.53
Skorin-Kapov (1990) $n = 64$	24249	24300	24496	24251	24277	518.50	438.25	283.47
Skorin-Kapov (1990) $n = 72$	33128	33254	35507	33177	33208	948.01	614.34	387.72
Skorin-Kapov (1990) $n = 81$	45449	45603	45893	45571	45550	1648.70	786.53	550.03
Skorin-Kapov (1990) $n = 90$	57767	57946	58432	57876	57904	3093.96	1047.87	786.10

*This column contains the best known objective value for these QAPs as reported by QAPLIB (Burkard *et al.*, 1997).
**This value contains the best objective values obtained by Heragu & Alfa's (1992) SA implementation.
***The reported CPU times are based on computational implementation on a 400MHz, Pentium II PC in a Windows 98 operating environment.

both SA and LOPI-SA, even for the largest test problems. For LOPI-SA, the largest deviation from a best-known solution (approximately .24%) occurred for the $n = 72$ test problem. The total CPU times for both methods were quite reasonable, with LOPI-SA having a sizable advantage for most problems. It is widely known that there is an inherent tradeoff between solution quality and computational effort associated with local-search methods for QAP and related combinatorial-optimization problems (Brusco 1999; De Soete, Hubert, and Arabie 1988; Groenen 1993; Johnson, Aragon, McGeoch, and Schevon 1989, 1991; Laursen 1993). Although we observed improvement for SA and LOPI-SA when increasing the number of replications from 20 to 100, we believed that the results for 20 replications were of sufficient quality to ensure that these methods would supply good initial permutations for least-squares unidimensional scaling algorithms in a reasonable amount of time.

3. Computational Study

The primary objective of the computational study was to investigate the utility of the three QAP solution methods for providing initial permutations for least-squares unidimensional scaling algorithms. Of particular interest were the benefits of better initial permutations for large symmetric proximity matrices. Using only 20, 50, or even 100 randomly generated initial permutations for problems with $n = 50$ or $n = 100$ might not enable convergence to good solutions for (1) – (4). Although the number of random starting sequences could be increased, the large number of possible pairwise interchanges, block rotations, and insertions (in conjunction with the necessary coordinate re-estimation) makes such a strategy computationally infeasible. In this section, we present the details of a computational study designed to evaluate the ability of the LOPI, SA, and LOPI-SA algorithms to provide good starting solutions for an algorithm designed for large problems associated with objective function (2). A secondary objective of this study, which builds on the findings in Section 2.6, pertains to a comparative evaluation of these methods regarding their relative solutions to (5). This information might be useful to quantitative psychologists who make use of the QAP for some of its other applications in combinatorial data analysis, such as those described by Hubert (1987, Chapter 4).

3.1 Heuristic Procedure for Least-Squares (2) Problem

Heuristic procedures for solving (2) are often comparable to those for solving (5), except that coordinate re-estimation is required for the former. De Soete, Hubert, and Arabie (1988), Heiser (1989), and Groenen (1993, Chapter 4) have proposed and tested LOPI algorithms for searching permutations for (2). Recently, Hubert and Arabie (1994) proposed that LOPI can be augmented by order reversals of object blocks in the sequence, as well as the insertion of object blocks between any two other objects in the sequence. Hubert, Arabie, and Meulman (1997) subsequently implemented a pairwise interchange / object-block reversal / object insertion method within the context of (3) and (4). Brusco (1999) subsequently found that an integrated pairwise interchange, object-block reversal, and single-object insertion strategy (hereafter referred to as PIRI for pairwise interchange / reversal / insertion) was also quite effective for (2). Briefly, our implementation of the PIRI algorithm investigates neighborhood changes in an incumbent permutation by examining all pairwise interchanges, followed by block reversals for all blocks of size b, $4 \leq b \leq n\text{-}1$, and finally all single-object insertions. Each time a permutation is modified via one of these operations, the formulae in (2) are called upon for any necessary coordinate re-estimation. The PIRI procedure terminates when none of the operations results in any improvement in (2).

3.2 Test Problems and Computational Procedures

Twenty test problems from Brusco's (1999) recent study were used to complete the experimental analysis. These large unidimensional scaling problems consist of 100 × 100 symmetric proximity matrices. The proximities were integer values on the interval [0, 100] and were generated based on a uniform distribution. These problems, which contain considerable error, are recognized as particularly difficult because heuristic procedures tend to display a "uniqueness" property with respect to replications of their implementation for such problems. In other words, if 50 replications of a heuristic are applied to such problems, the best solution identified is typically associated with only one (unique) replicate.

Like the LOPI, SA, and LOPI-SA methods, the PIRI algorithm was written in Fortran Powerstation 1.0 and implemented on a 400 MHz Pentium II PC in a Windows 98 environment. Solutions to the QAP (5) for each of the 20 test problems were obtained using the LOPI, SA and LOPI-SA methods. The objective function (5) values, total CPU time, and 20

permutations generated by each of the three methods for each of these test problems were subsequently stored. Next, for each of the test problems, the 20 permutations corresponding to each of the three QAP solution methods were submitted as initial permutations to the PIRI algorithm. The objective function values for (2) and total CPU time for PIRI were subsequently collected.

3.3 Computational Results for QAP (5) Solutions

The best objective function values (5) and total CPU times for permutations generated by each of the three methods for the 20 test problems are compared in Table 2. The strengths and weaknesses of the three methods—LOPI, SA, and LOPI-SA—are clear from these results. SA and LOPI-SA consistently found better objective function values (5) than LOPI. The objective function values associated with SA were slightly better on average than those produced by LOPI-SA. On average, SA produced an objective function value that was $\approx$ .0006% larger than that of LOPI-SA. Furthermore, SA was better for 25% of the problems; LOPI-SA was better for 20% of the problems; SA and LOPI-SA had the same objective function value for 55% of the problems.

LOPI performed somewhat better than SA according to CPU time, requiring less for all of the test problems. On average, SA required approximately 18% more CPU time than LOPI. LOPI-SA consistently required less CPU time than both LOPI and SA. The average CPU time for LOPI-SA was 384.68s (25%) less than the corresponding average for LOPI. LOPI-SA provided an even larger computational savings relative to SA, requiring an average of 654.27s (36%) less CPU time.

For these large (n = 100) QAPs, it is easy to examine the trade-off between computing time and solution quality for the three solution methods. Compared to the other two methods, LOPI sacrificed solution quality for (5). It also required significantly more computational effort than LOPI-SA. When compared to SA, LOPI-SA required significantly less computational effort without sacrificing a significant decrease in the final objective function value (5).

3.4 Computational Results for Least-Squares (2) Solutions

Table 3 provides the objective function values (2) obtained via the application of PIRI when using initial permutations generated by LOPI, SA, and LOPI-SA. Also included in Table 3 are two sets of objective function

Table 2. Performance comparison of LOPI, SA, and LOPI-SA with respect to their best objective values for (5) and total CPU time.

	Best Objective Function (5) Value*			Total CPU Seconds**		
	LOPI	SA	LOPI-SA	LOPI	SA	LOPI-SA
Problem 1	8924067	8934182	8934182	1480.29	1724.22	1101.48
Problem 2	8924433	8936327	8936327	1491.61	1864.33	1091.98
Problem 3	9018404	9022971	9022687	1544.23	1867.52	1107.79
Problem 4	8972048	8986618	8986618	1438.56	1733.78	1303.16
Problem 5	8971893	8978456	8978645	1573.95	1740.42	1098.79
Problem 6	8996100	9014224	9014224	1475.30	1753.88	1101.69
Problem 7	9020872	9035250	9035250	1498.92	1801.77	1162.11
Problem 8	8852019	8856892	8856897	1516.22	1887.74	1105.48
Problem 9	8966753	8974226	8974308	1570.87	1802.71	1231.87
Problem 10	9003692	9013357	9013357	1536.00	1681.82	1168.81
Problem 11	8950765	8958114	8958436	1541.86	1952.16	1096.53
Problem 12	9025513	9034017	9034017	1468.87	2030.60	1154.04
Problem 13	8997919	9008308	9008308	1465.63	1673.03	1144.21
Problem 14	8896101	8910442	8910373	1631.24	1673.36	1095.87
Problem 15	8894847	8911652	8911652	1467.12	1850.22	1160.03
Problem 16	9098562	9106250	9106250	1475.30	1733.72	1144.87
Problem 17	8998297	9001146	9000153	1481.78	1946.83	1153.05
Problem 18	8938877	8947099	8946974	1601.81	1855.88	1227.81
Problem 19	8916382	8923737	8923737	1636.50	1666.77	1200.39
Problem 20	8934406	8945833	8945584	1754.98	1801.94	1107.51
Mean	8965098	8974955	8974899	1532.55	1802.14	1147.87
# of times best	0	16	15			

*The best (largest) objective function value (5) obtained across 80 replications for LOPI and 20 replications for SA and LOPI-SA.

**The total CPU time (across all replications) on a 400MHz Pentium II PC.

values (2) from Brusco's (1999) recent study. The first set of results, Random, were obtained by applying 20 replications of PIRI to randomly-generated initial permutations. The second set of results, MBLSH, were obtained via 20 replications of the morph-based local-search heuristic developed by Brusco (1999).

The results in Table 3 illuminate the importance of supplying promising permutations to least-squares unidimensional scaling algorithms. It is clear that supplying a better initial permutation generally results in a better least-squares objective-function value (2). Supplying PIRI with initial

permutations from either SA or LOPI-SA always enabled a better objective function value (2) to be found relative to instances where the initial permutations were provided by Random or LOPI. It is also encouraging to observe that both SA and LOPI-SA provided a better solution than MBLSH for each of the 20 test problems. Thus, we have established new benchmarks for the complete set of problems from Brusco's (1999) study. The average objective function value (2) associated with the SA initial permutations was .0018% larger than the corresponding average for the LOPI-SA permutations. Furthermore, comparing the results for SA and LOPI-SA, the initial permutations from SA yielded better least-squares objective function values (2) for 30% of the problems; LOPI-SA initial permutations were better for 15% of the problems; and their permutations resulted in the same least-squares objective function value for 55% of the problems.

Table 3 reports the total CPU time for Random, MBLSH, LOPI, SA, and LOPI-SA. The total CPU times for the Random and MBLSH procedures, which were obtained from Brusco's (1999) study, are directly comparable to the other methods because they were obtained on the same hardware and software platform. The reported solution times for LOPI, SA, and LOPI-SA are the sum of two components: (a) the time to solve the QAP's (from Table 2), and (b) the time resulting from the application of PIRI to the initial permutations.

Our results indicate that the CPU time used to obtain improved initial permutations using SA or LOPI-SA is more than offset by the savings in CPU time for the PIRI algorithm, which has less improvement to make on the initial sequences. This conclusion is supported by the fact that the total CPU time consumed when using either SA or LOPI-SA was always less than the total CPU times when the initial permutations were supplied by either Random or LOPI. The MBLSH and SA results are comparable in terms of CPU time, whereas LOPI-SA always required less CPU time than either MBLSH or SA.

The experimental results in Table 3 are important because they suggest that initial permutations with better objective values for (5) tend to yield better objective function values for (2) upon completion of the PIRI algorithm. Recalling that LOPI generated consistently inferior objective function values for (5), we notice that the least-squares objective function values for (2) obtained after LOPI passed its 20 permutations to PIRI were also consistently inferior to the least-squares objective function values based on initial permutations from SA and LOPI-SA. SA, which provided slightly better objective function values for (5) relative to LOPI-SA, also provided

Table 3. Performance results pertaining to the best least-squares objective function (2) values and total CPU time. The "Random" and "MBLSH" results are from Brusco (1999). The remaining results are based on the application of PIRI using initial permutations from the LOPI, SA, and LOPI-SA solutions to the QAP (5). The CPU times for LOPI, SA, and LOPI-SA include the time to generate the initial permutations (reported in Table 2) plus the time to run the PIRI algorithm.

	Best Objective Function (2) Value*					Total CPU Seconds**				
	Random	MBLSH	LOPI	SA	LOPI-SA	Random	MBLSH	LOPI	SA	LOPI-SA
Problem 1	95819.62	95837.48	95722.45	**95844.13**	**95844.13**	2758.19	1995.44	3008.60	2247.82	1637.71
Problem 2	95682.16	95687.94	95872.43	**95892.81**	**95892.81**	2625.11	2034.49	3033.69	2326.74	1649.85
Problem 3	97561.28	97718.80	97723.11	**97757.85**	97750.76	3341.83	2063.04	3414.71	2377.66	1686.26
Problem 4	96599.19	96831.68	96712.95	**96974.23**	**96974.23**	3117.90	2571.17	3060.06	2309.18	2267.00
Problem 5	96619.16	96824.27	96761.25	96852.68	**96852.77**	2694.65	2517.13	3117.96	2364.15	1691.60
Problem 6	97478.50	97554.68	97521.44	**97558.39**	**97558.39**	3125.32	2337.98	3233.30	2261.55	1800.34
Problem 7	97644.55	97929.53	97957.05	**98041.38**	**98041.38**	3023.81	2615.93	2929.23	2588.03	1896.02
Problem 8	93965.69	93736.44	94150.74	**94181.35**	**94181.35**	3062.10	2331.36	3472.94	2586.83	1835.34
Problem 9	96481.99	96733.91	96655.88	96734.05	**96734.08**	3137.19	2562.61	3430.04	2510.70	2085.58
Problem 10	97428.42	97531.99	97409.15	97532.40	**97533.98**	2851.89	2349.12	2998.87	2378.49	1879.22
Problem 11	96103.02	96365.23	96259.63	**96398.93**	96396.23	2826.41	2315.66	3147.01	2570.56	1684.07
Problem 12	97777.51	97999.70	97873.51	**98003.40**	**98003.40**	3182.72	2663.16	3163.88	2608.36	1656.51
Problem 13	97453.45	97444.35	97445.88	**97483.87**	**97483.87**	3329.26	2408.58	3015.47	2483.07	1807.43
Problem 14	95187.09	94963.36	95226.88	**95328.36**	95327.34	2935.56	2333.79	3180.79	2205.15	1674.62
Problem 15	95277.49	95133.80	95189.19	**95359.62**	**95359.62**	2906.37	2350.15	3173.76	2475.00	1769.98
Problem 16	99328.69	99607.80	99454.35	**99610.94**	**99610.94**	2721.33	2542.87	3286.58	2336.80	1612.45
Problem 17	97187.50	97254.39	97257.78	**97297.47**	97276.36	2961.36	2381.63	3097.24	2561.84	1752.89
Problem 18	96035.78	96117.06	96104.64	**96121.31**	96120.96	2864.48	2168.84	3064.40	2403.88	1940.41
Problem 19	95618.52	95528.77	95630.10	**95645.77**	**95645.77**	2957.53	1996.00	3020.36	2324.29	1974.89
Problem 20	95992.39	96023.74	96070.68	**96106.84**	96103.28	2889.75	1915.75	3330.64	2452.64	1815.17
Mean	96562.10	96641.25	96649.95	96736.29	96734.58	2965.64	2322.74	3158.98	2418.64	1805.87
# of times best	0	0	0	17	14					

*The best (largest) objective function value (2) for each solution method. Values in bold indicate best value of (2) across all five methods.

**The total CPU time (across all 20 replications) used by each heuristic on a 400MHz Pentium II PC. The hardware and software platform for the Random and MBLSH results from Brusco (1999) were the same as the platform used for LOPI, SA, and LOPI-SA.

initial permutations that yielded slightly better least-squares objective function values for (2). These findings confirm our supposition that PIRI solution quality improves when the initial sequences are based on good solutions to (5), relative to circumstances where the initial sequences are based on inferior solutions to (5). Because initial permutations associated with better solutions to (5) also paid off from a computational standpoint due to faster convergence of the PIRI algorithm, we believe it is prudent to spend a sufficient amount of computational effort to obtain good QAP solutions as initial permutations.

4. Improved Starting Sequences for other Least-Squares Measures

We have demonstrated that improved starting sequences can provide considerable solution quality and CPU time benefits for least-squares unidimensional scaling problems posed by (2). For other least-squares criteria, such as (3) and (4), improved starting sequences might be even more important because the coordinate re-estimation for such criteria requires more sophisticated solution procedures. In this section, we provide some insight as to the potential efficacy of QAP-based initial permutations for (3) and (4) by comparing QAP-generated permutations to solutions obtained by Hubert, Arabie, and Meulman (1997) for two previously published data sets.

Tables 4 and 5 present proximity measures from two well-studied problems in quantitative psychology. The lower triangle of Table 4 provides the dissimilarity measures among digits from Rothkopf's (1957) Morse code study. These measures are based on proportions associated with subject judgment of pairs of symbols in different orders. Hubert and Schultz (1976) and Hubert, Arabie, and Meulman (1997) have previously fit unidimensional scaling models to these proximity data. The upper triangle of Table 4 contains pairwise distance between object locations for both the linear and circular distance measures described in Section 2.1. The lower triangle of Table 5 presents proximity (dissimilarity) data pertaining to a study of judgments of complex tonal intervals (Levelt, van de Geer, and Plomp 1966). These data have been discussed and analyzed by Borg and Lingoes (1979, 769-774) and Hubert, Arabie, and Meulman (1997). The dissimilarity measures in Table 5 are identical to those reported by Hubert, Arabie, and Meulman (1997) and were obtained by subtracting the similarity measures originally reported by Levelt, van de Geer, and Plomp (1966) from 32. The upper triangle of Table 5 contains the corresponding linear and circular pairwise distance measures between object locations.

Table 4. The Morse code dissimilarity measures from Hubert, Arabie, and Meulman (1997, p. 259), which are based on data from Rothkopf (1957), are contained in the lower diagonal of the matrix, whereas the upper triangle contains d_{kl} = "p/q" where p (q) is Szczotka's (1972) linear (circular) distance between object locations k and l. When reading the lower triangle of the matrix, the row and column headings correspond to the Morse code symbols. When reading the upper triangle, the row and column headings correspond to locations on the continuum.

Symbol (Location)	0 (1)	1 (2)	2 (3)	3 (4)	4 (5)	5 (6)	6 (7)	7 (8)	8 (9)	9 (10)
0 ----- (1)	-	1/1	2/2	3/3	4/4	5/5	6/4	7/3	8/2	9/1
1 •---- (2)	.75	-	1/1	2/2	3/3	4/4	5/5	6/4	7/3	8/2
2 ••--- (3)	1.69	.82	-	1/1	2/2	3/3	4/4	5/5	6/4	7/3
3 •••-- (4)	1.87	1.54	1.25	-	1/1	2/2	3/3	4/4	5/5	6/4
4 ••••- (5)	1.76	1.85	1.47	.89	-	1/1	2/2	3/3	4/4	5/5
5 ••••• (6)	1.77	1.72	1.33	1.32	1.41	-	1/1	2/2	3/3	4/4
6 -•••• (7)	1.59	1.51	1.66	1.53	1.64	.70	-	1/1	2/2	3/3
7 --••• (8)	1.26	1.50	1.57	1.74	1.81	1.56	.70	-	1/1	2/2
8 ---•• (9)	.86	1.45	1.83	1.85	1.90	1.84	1.38	.83	-	1/1
9 ----• (10)	.95	1.63	1.81	1.86	1.90	1.64	1.70	1.22	.41	-

4.1 Linear Unidimensional Scaling

The LOPI-SA algorithm was applied to both the Morse code and tonal interval data under the assumption of Szczotka's (1972) linear distance (this is the same distance measure Hubert and Schultz (1976) used for this data set). For the Morse code data from Rothkopf (1957), all twenty replications of the algorithm yielded the same permutation: {5, 4, 3, 6, 7, 2, 8, 1, 9, 0}. This same sequence was also obtained 48 out of 50 times by Hubert and Schultz (1976) and for each of the 100 replications performed by Hubert, Arabie, and Meulman (1997) using a solution algorithm for (3). For the Levelt, van de Geer, and Plomp (1966) data, the LOPI-SA algorithm yielded the permutation, {1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 15, 13, 14}, for each of the 20 replications. This is very close to the permutation, {1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15}, which was found by Hubert, Arabie, and Meulman (1997) using the algorithm designed for (3). These results suggest that an initial permutation based on a QAP solution (5) can be very close to (if not the same as) the final permutation associated with the linear least-squares unidimensional scaling solution for (3).

4.2 Circular Unidimensional Scaling Demonstration

The LOPI-SA algorithm was also applied to both the Morse code and tonal interval data under the assumption of Szczotka's (1972) circular dis-

Table 5. The musical complex-tonal dissimilarity measures from Hubert, Arabie, and Meulman (1997, p. 260), which are based on data obtained by Levelt, van de Geer, and Plomp (1966), are contained in the lower diagonal of the matrix, whereas the upper triangle contains d_{kl} = "p/q" where p (q) is Szczotka's (1972) linear (circular) distance between object locations k and l. When reading the lower (upper) triangle of the matrix, the row and column headings correspond to the frequency ratios (locations on the continuum).

Frequency / Location	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1 15:16	-	1/1	2/2	3/3	4/4	5/5	6/6	7/7	8/7	9/6	10/5	11/4	12/3	13/2	14/1
2 11:12	0	-	1/1	2/2	3/3	4/4	5/5	6/6	7/7	8/7	9/6	10/5	11/4	12/3	13/2
3 8:9	3	0	-	1/1	2/2	3/3	4/4	5/5	6/6	7/7	8/7	9/6	10/5	11/4	12/3
4 5:6	13	10	4	-	1/1	2/2	3/3	4/4	5/5	6/6	7/7	8/7	9/6	10/5	11/4
5 4:5	18	15	9	4	-	1/1	2/2	3/3	4/4	5/5	6/6	7/7	8/7	9/6	10/5
6 3:4	17	22	19	10	7	-	1/1	2/2	3/3	4/4	5/5	6/6	7/7	8/7	9/6
7 5:7	24	24	18	7	8	5	-	1/1	2/2	3/3	4/4	5/5	6/6	7/7	8/7
8 2:3	23	22	18	19	14	11	10	-	1/1	2/2	3/3	4/4	5/5	6/6	7/7
9 5:8	26	25	19	12	11	15	10	5	-	1/1	2/2	3/3	4/4	5/5	6/6
10 3:5	20	21	20	7	12	8	19	7	8	-	1/1	2/2	3/3	4/4	5/5
11 4:7	25	21	17	18	14	18	16	19	5	2	-	1/1	2/2	3/3	4/4
12 8:15	25	22	25	22	15	22	13	14	14	14	6	-	1/1	2/2	3/3
13 1:2	24	29	23	23	18	17	23	18	20	19	10	6	-	1/1	2/2
14 4:9	23	18	26	24	24	20	23	22	22	15	12	19	3	-	1/1
15 2:5	23	18	26	22	25	20	22	2	14	14	18	19	7	2	-

tance. For the Morse code data from Rothkopf (1957), each of the 20 replications of the algorithm yielded the same permutation: {1, 2, 3, 4, 5, 6, 7, 8, 9, 0}. This is the same permutation found by Hubert and Schultz (1976) in 46 out of 50 replications, as well as by Hubert, Arabie, and Meulman (1997) using an algorithm for (4). For the Levelt, van de Geer, and Plomp (1966) data, each of the 20 replications of the algorithm yielded the sequence: {2, 1, 3, 4, 5, 6, 7, 9, 8, 10, 11, 12, 15, 13, 14}. This is very close to the permutation {1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15} identified by Hubert, Arabie, and Meulman (1997).

The fact that the LOPI-SA algorithm designed for (5) often found the same, or nearly the same, sequence as algorithms designed for (3) and (4) is very encouraging. If LOPI-SA is used to provide these sequences to algorithms for (3) and (4), then convergence of those algorithms to high-quality solutions within a reasonable amount of CPU time should be achieved. The total CPU time (on a 400MHz Pentium II microcomputer) required to run 20 replications of the LOPI-SA algorithm was less than 3 seconds for the Rothkopf (1957) data and less than 5 seconds for the Levelt, van de Geer, and Plomp (1966) data.

5. Conclusions and Extensions

This paper has demonstrated that solutions to the quadratic assignment problem can provide effective and efficient initial permutations for large least-squares unidimensional scaling problems. Good solutions to a linear-seriation QAP were provided by simulated annealing, as well as a hybrid pairwise interchange / simulated annealing method, within a reasonable amount of CPU time. However, the ease of computation for the QAP, and the wealth of previous research into the methods for solving the QAP, suggest that the development of algorithms incorporating other methods—such as object-block rotations or object insertions—to solve the problem would be interesting. Other methods such as tabu search and genetic algorithms, as well as a variety of possible metaheuristic hybrids, might provide even better QAP solutions.

The provision of initial permutations from SA and LOPI-SA to a least-squares unidimensional-scaling algorithm for (2) resulted in rapid convergence to high-quality solutions. Given the tradeoff that exists between computational effort and solution quality for large problems, it seems prudent to consider using simpler QAP-based seriation methods (using criteria such as (5)) to get good starting points for least-squares algorithms, which have a greater computational burden. This conclusion is further strengthened by our findings for the two small data sets from the literature. The fact that the QAP permutations were very comparable (if not

identical) to those found by Hubert, Arabie, and Meulman (1997) for generalized least-squares measures (3) and (4) encourages a reasonable hypothesis that QAP solutions can be of service in providing good initial permutations to methods for these measures. In light of the intuitive appeal of these measures (Hubert, Arabie, and Meulman 1997), the utility of QAP-based starting points for larger problem instances of (3) and (4) would be an interesting topic for future investigation.

There are at least two other important extensions of the research presented herein. The first of these is associated with the implications of better starting solutions for direct solution approaches to (1), such as distance smoothing (Groenen, Heiser, and Meulman 1999; Pliner 1996). These methods also require starting solutions as input, and QAP solutions could be used in this regard. The key would be to parameterize the distance smoothing algorithm so that, in its early stages, it would not discard too much of the information from the starting solution. The second extension concerns the use of QAP methods to provide starting solutions for combinatorial approaches to multidimensional city-block scaling (Arabie 1991; Carroll and Arabie 1980, 1998; Heiser 1989; Hubert, Arabie, and Hesson-McInnis 1992). For example, in the two-dimensional case with n = 100, one might consider a 10 × 10 square with rectangular-unit distance as the measure of distance between pairs of blocks defining the square. Such a problem is representative of the traditional facility layout problem in business and engineering applications, and the QAP solution to such a problem might provide at least some idea as to orderings of objects on each dimension.

References

AHUJA, R. K., ORLIN, J. B., and TIWARI, A. (2000), "A Greedy Genetic Algorithm for the Quadratic Assignment Problem," *Computers and Operations Research*, *27*, 917-934.

ARABIE, P. (1991), "Was Euclid an Unnecessarily Sophisticated Psychologist?" *Psychometrika*, *56*, 567-587.

ARMOUR, G. C., and BUFFA, E. S. (1963), "A Heuristic Algorithm and Simulation Approach to the Relative Location of Facilities," *Management Science*, *9*, 294-309.

BORG, I., and LINGOES, J. C. (1979), "Multidimensional Scaling with Side Constraints on the Distances," in *Geometric Representations of Relational Data*, Eds., J. C. Lingoes, E. E. Roksam, and I. Borg, Ann Arbor, MI: Mathesis Press, pp. 753-790.

BRUSCO, M. J. (1999), "Morph-Based Local-Search Heuristics for Large-Scale Combinatorial Data Analysis," *Journal of Classification*, *16*, 163-180.

BURKARD, R. E., KARISCH, S. E., and RENDL, F. (1997), "QAPLIB – A Quadratic Assignment Problem Library," *Journal of Global Optimization*, *10*, 391-403.

BURKARD, R. E., and RENDL, F. (1984), "A Thermodynamically Motivated Simulation Procedure for Combinatorial Optimization Problems," *European Journal of Operational Research, 17*, 169-174.

CARROLL, J. D., and ARABIE, P. (1980), "Multidimensional Scaling," *Annual Review of Psychology, 31*, 607-649.

CARROLL, J. D., and ARABIE, P. (1998), "Multidimensional Scaling," in *Management, Judgment, and Decision Making*, Ed., M. H. Birnbaum, San Diego: Academic Press, pp. 179-250.

CHIANG, W., and CHIANG, C. (1998), "Intelligent Local Search Strategies for Solving Facility Layout Problems with the Quadratic Assignment Problem Formulation," *European Journal of Operational Research, 106*, 457-488.

CONNOLLY, D. T. (1990), "An Improved Annealing Scheme for the QAP," *European Journal of Operational Research, 46*, 93-100.

DEFAYS, D. (1978), "A Short Note on a Method of Seriation," *British Journal of Mathematical and Statistical Psychology, 31*, 49-53.

DE LEEUW, J., and HEISER, W. J. (1977), "Convergence of Correction-Matrix Algorithms for Multidimensional Scaling," in *Geometric Representations of Relational Data: Readings in Multidimensional Scaling*, Eds., J. C. Lingoes, E. E. Roksam, and I. Borg, Ann Arbor, MI: Mathesis Press, pp. 735-752.

DE LEEUW, J., and HEISER, W. J. (1980), "Multidimensional Scaling with Restrictions on the Configuration of Points," in *Multivariate Analysis* (Vol. 5), Ed., P. R. Krishnaiah Amsterdam: North-Holland, pp. 501-522.

DE SOETE, G., HUBERT, L., and ARABIE, P. (1988), "The Comparative Performance of Simulated Annealing on Two Problems of Combinatorial Data Analysis," in *Data Analysis and Informatics* (*Vol.* 5), Ed., E. Diday (Ed.), Amsterdam: North-Holland, pp. 489-496.

EISLER, H. (1973), "The Algebraic and Statistical Tractability of the City Block Metric," *British Journal of Mathematical and Statistical Psychology, 26*, 212-218.

GAMBARDELLA, L. M., TAILLARD, E. D., and DORIGO, M. (1999), "Ant Colonies for the Quadratic Assignment Problem," *Journal of the Operational Research Society, 50*, 167-176.

GLINER, G. (1981), "A Note on a Statistical Paradigm for the Evaluation of Cognitive Structure in Physics Instruction," *Applied Psychological Measurement, 5*, 493-502.

GROENEN, P. J. F. (1993), *The Majorization Approach to Multidimensional Scaling: Some Problems and Extensions*, Leiden, The Netherlands: DSWO Press.

GROENEN, P. J. F., and HEISER, W. J. (1996), "The Tunneling Method for Global Optimization in Multidimensional Scaling," *Psychometrika, 61*, 529-550.

GROENEN, P. J. F., HEISER, W. J., and MEULMAN, J. J. (1999), "Global Optimization in Least-Squares Multidimensional Scaling by Distance Smoothing," *Journal of Classification, 16*, 225-254.

HAHN, P., GRANT, T., and HALL, N. (1998), "A Branch-and-Bound Algorithm for the Quadratic Assignment Problem Based on the Hungarian Method," *European Journal of Operational Research, 108*, 629-640.

HARRIS, F. N., and PACKARD, T. (1985), "Intensity Judgments of Emotion Words: Implications for Counselor Training," *Journal of Counseling Psychology, 32*, 288-291.

HEISER, W. J. (1989), "The City-Block Model for Three-Way Multidimensional Scaling," in *Multiway Data Analysis*, Eds., R. Coppi and S. Belasco, Amsterdam: North-Holland, pp. 395-404

HERAGU, S. S., and ALFA, A. S. (1992), "Experimental Analysis of Simulated Annealing Based Algorithms for the Layout Problem," *European Journal of Operational Research*, *57*, 190-202.

jHOLLOWAY, E. L. (1982), "Interactional Structure of the Supervision Interview," *Journal of Counseling Psychology*, *29*, 309-317.

HUBERT, L. J. (1974), "Some Applications of Graph Theory and Related Nonmetric Techniques to Problems of Approximate Seriation: The Case of Symmetric Proximity Measures," *British Journal of Mathematical and Statistical Psychology*, *27*, 133-153.

HUBERT, L. J. (1987), *Assignment Methods in Combinatorial Data Analysis*. New York: Marcel Dekker.

HUBERT, L. J., and ARABIE, P. (1986), "Unidimensional Scaling and Combinatorial Optimization," in *Multidimensional Data Analysis*, Eds., J. de Leeuw, W. Heiser, J. Meulman, and F. Critchley, Leiden, The Netherlands: DSWO Press, pp. 181-196.

HUBERT, L. J., and ARABIE, P. (1988), "Relying on Necessary Conditions for Optimization: Unidimensional Scaling and Some Extensions," in *Classification and Related Methods for Data Analysis*, Ed., H. H. Bock, Amsterdam: North Holland, pp. 463-472

HUBERT, L., and ARABIE, P. (1994), "The Analysis of Proximity Matrices Through Sums of Matrices Having (Anti-) Robinson Forms," *British Journal of Mathematical and Statistical Psychology*, *47*, 1-40.

HUBERT, L. J., ARABIE, P., and HESSON-MCINNIS, M. (1992), "Multidimensional Scaling in the City-Block Metric: A Combinatorial Approach," *Journal of Classification*, *9*, 211-236.

HUBERT, L., ARABIE, P., and MEULMAN, J. (1997), "Linear and Circular Unidimensional Scaling for Symmetric Proximity Matrices," *British Journal of Mathematical and Statistical Psychology*, *50*, 253-284.

HUBERT, L. J., and SCHULTZ, J. V. (1976), "Quadratic Assignment as a General Data Analysis Strategy," *British Journal of Mathematical and Statistical Psychology*, *29*, 190-241.

JOHNSON, D. S., ARAGON, C. R., MCGEOCH, L. A., and SCHEVON, C. (1989), "Optimization by Simulated Annealing: An Experimental Evaluation: Part I, Graph Partitioning," *Operations Research*, *37*, 865-892.

JOHNSON, D. S., ARAGON, C. R., MCGEOCH, L. A., and SCHEVON, C. (1991), "Optimization by Simulated Annealing: An Experimental Evaluation: Part II, Graph Coloring and Number Partitioning," *Operations Research*, *39*, 378-406.

KNUTH, D. E. (1997), *The Art of Computing: Vol. 2. Seminumerical Algorithms*, Reading, MA: Addison-Wesley.

KOOPMANS, T. C., and BECKMANN, M. (1957), "Assignment Problems and the Location of Economic Activities," *Econometrica*, *25*, 53-76.

LAU, K., LEUNG, P. L., and TSE, K. (1998), "A Nonlinear Programming Approach to Metric Unidimensional Scaling," *Journal of Classification*, *15*, 3-14.

LAURSEN, P. S. (1993). Simulated annealing for the QAP – Optimal tradeoff between simulation time and solution quality. *European Journal of Operational Research*, *69*, 238-243.

LEVELT, W. J. M., VAN DE GEER, J. P., and PLOMP, R. (1966), "Triadic Comparisons of Musical Intervals," *British Journal of Mathematical and Statistical Psychology, 19*, 163-179.

LI, Y., PARDALOS, P. RAMAKRISHNAN, K., and RESENDE, M. (1994), "Lower Bounds for the Quadratic Assignment Problem," *Annals of Operations Research, 50*, 387-410.

MARGOLIN, G., and WAMPOLD, B. E. (1981), "A Sequential Analysis of Conflict and Accord in Distressed and Nondistressed Marital Partners," *Journal of Consulting and Clinical Psychology, 49*, 554-567.

MEDINA-DÍAZ, M. (1993), "Analysis of Cognitive Structure Using the Linear Logistic Test Model and Quadratic Assignment," *Applied Psychological Measurement, 17*, 117-130.

METROPOLIS, N., ROSENBLUTH, A., ROSENBLUTH, M., TELLER, A., and TELLER, E. (1953), "Equation of State Calculations by Fast Computing Machines," *Journal of Chemical Physics, 21*, 1087-1092.

NUGENT, C. E., VOLLMAN, T. E., and RUML, J. (1968), "An Experimental Comparison of Techniques for the Assignment of Facilities to Locations," *Operations Research, 16*, 150-173.

POOLE, K. T. (1990). Least squares metric, unidimensional scaling of multivariate linear models. *Psychometrika, 55*, 123-149.

PLINER, V. (1996). Metric unidimensional scaling and global optimization. *Journal of Classification, 13*, 3-18.

PLUTCHIK, R., and CONTE, H. R. (Eds) (1997). *Circumplex Models of Personality and Emotions*. Washington, D.C.: American Psychological Association.

RESENDE, M. G. C., RAMAKRISHNAN, K. G., and DREZNER, Z. (1995), "Computing Lower Bounds for the Quadratic Assignment Problem with an Interior Point Algorithm for Linear Programming," *Operations Research, 43*, 781-791.

ROTHKOPF, E. Z. (1957), "A Measure of Stimulus Activity and Errors in Some Paired-Associate Learning Tasks," *Journal of Experimental Psychology, 53*, 94-101.

SIMANTIRAKI, E. (1996), "Unidimensional Scaling: A Linear Programming Approach Minimizing Absolute Deviations," *Journal of Classification, 13*, 19-25.

SKORIN-KAPOV, J. (1990), "Tabu Search Applied to the Quadratic Assignment Problem," *ORSA Journal on Computing, 2*, 33-45.

STEINBERG, L. (1961), "The Backboard Wiring Problem: A Placement Algorithm," *SIAM Review, 3*, 37-50.

SZCZOTKA, F. (1972), "On a Method of Ordering and Clustering of Objects," *Zastosowania Mathemetyki, 13*, 23-33.

TAILLARD, E. (1991), "Robust Taboo Search for the Quadratic Assignment Problem, *Parallel Computing, 17*, 443-455.

TATE, D. M., and SMITH, A. E. (1995), "A Genetic Approach to the Quadratic Assignment Problem," *Computers and Operations Research, 22*, 73-83.

WAMPOLD, B. E., and MARGOLIN, G. (1982), "Nonparametric Strategies to Test the Independence of Behavioral States in Sequential Data,". *Psychological Bulletin, 92*, 755-765.

WILHELM, M. R., and WARD, T. L. (1987), "Solving Quadratic Assignment Problems bySimulated Annealing," *IIE Transactions, 19*, 107-119.