A COMPARATIVE STUDY OF ALGORITHMS FOR MATRIX BALANCING

MICHAEL H. SCHNEIDER
Johns Hopkins University, Baltimore, Maryland

STAVROS A. ZENIOS
University of Pennsylvania, Philadelphia, Pennsylvania

(Received February 1987; revisions received March 1988, February 1989; accepted March 1989)

The problem of adjusting the entries of a large matrix to satisfy prior consistency requirements occurs in economics, urban planning, statistics, demography, and stochastic modeling; these problems are called *Matrix Balancing Problems.* We describe five applications of matrix balancing and compare the algorithmic and computational performance of balancing procedures that represent the two primary approaches for matrix balancing—matrix scaling and nonlinear optimization. The algorithms we study are the **RAS** algorithm, a diagonal similarity scaling algorithm, and a truncated Newton algorithm for network optimization. We present results from computational experiments with large-scale problems based on producing consistent estimates of Social Accounting Matrices for developing countries.

A problem that occurs frequently in economics, urban planning, statistics, demography, and stochastic modeling is to adjust the entries of a large matrix to satisfy consistency requirements. It is typically posed as follows:

> Given a rectangular matrix A, determine a matrix X that is *close* to A and satisfies a given set of linear restrictions on its entries.

A well studied instance of this problem occurring in transportation planning and input–output analysis requires that A be adjusted so that the row and column totals equal fixed positive values. A related problem occurring in developmental economics requires that the row and column totals (of a square matrix) be equal to each other, but not necessarily to prespecified values.

These problems are known as *balancing* the matrix A. Because we want to discuss several balancing problems, each with different consistency requirements, the definition of a balanced matrix is problem dependent. That is, a matrix is defined to be *balanced* if it satisfies the given set of linear restrictions for the problem. In general, there are infinitely many matrices satisfying the consistency restrictions. The problem is to find a matrix that satisfies the restrictions and is related to the original matrix A in a suitably defined manner. Applications of matrix balancing can be found in the literature in journals of mathematical programming, statistics, economics, urban planning, numerical analysis, and mathematics. Matrix balancing is an important problem that has attracted attention in many different fields.

Algorithms for matrix balancing can be separated into two broad classes—scaling algorithms and optimization algorithms. Scaling algorithms multiply the rows and columns of A by positive constants until the matrix is balanced. Optimization algorithms minimize a penalty function that measures the deviation of a candidate balanced matrix from the original matrix A; the balance conditions are constraints in the optimization model so that the optimal solution is the balanced matrix *closest* to A. Specific methods used for balancing matrices include matrix scaling, nonlinear network methods, conjugate gradient algorithms, Lagrangian relaxation, and successive overrelaxation.

We have two major objectives in this paper. First, we want to unify the diverse applications of matrix balancing problems under a common framework. After studying the literature in several areas, we feel that researchers in different fields have been following parallel paths. A unifying thread may be found among these diverse applications. Second, we want to compare three representative algorithms for matrix balancing and show that each approach can be used to solve large-scale balancing problems. We have been involved with developing two of these methods.

Subject classifications: Economics, input-output analysis: estimating social accounting matrices. Networks/graphs, applications: network models in economics, statistics and urban planning. Programming, nonlinear algorithms/applications: scaling and nonlinear optimization for matrix balancing.

0030-364X/90/3803-0439 $01.25

The matrix balancing applications that we will describe can generally be formulated as one of two problems.

Problem 1. *Given an $m \times n$ nonnegative matrix A and positive vectors u and v of dimensions m and n, respectively, determine a nearby $m \times n$ nonnegative matrix X such that*

$$\sum_{j=1}^{n} x_{ij} = u_i \quad \textit{for } i = 1, 2, \ldots, m$$

$$\sum_{i=1}^{m} x_{ij} = v_j \quad \textit{for } j = 1, 2, \ldots, n$$

and $x_{ij} > 0$ only if $a_{ij} > 0$.

Problem 2. *Given an $n \times n$ nonnegative matrix A, determine a nearby $n \times n$ nonnegative matrix X such that*

$$\sum_{j=1}^{n} x_{ij} = \sum_{j=1}^{n} x_{ji}, \quad i = 1, 2, \ldots, n$$

and $x_{ij} > 0$ only if $a_{ij} > 0$.

Of course, for either problem to be well posed, some restrictions must be placed on the adjustments that can be made to A so that the requirement of a *nearby matrix* is well defined. Different algorithmic approaches follow naturally from different types of restrictions.

In Section 1, we describe five applications of matrix balancing. In Section 2, we describe two scaling algorithms, and in Section 3 we describe a nonlinear network model for balancing matrices. In Section 4, we compare these algorithms with respect to their efficiency, applicability, quality of solutions, ease of use, and data requirements. Furthermore, we investigate the relative efficiency of the algorithms by testing them on real problems that arise in economic equilibrium modeling. In Section 5, we draw conclusions and suggest directions for future research.

1. APPLICATIONS OF MATRIX BALANCING

1.1. Economics: Social Accounting Matrices (SAMs)

A Social Accounting Matrix, or SAM, is a square matrix A whose entries represent the flow-of-funds between the national income accounts of a country's economy at a fixed point in time. Each index of A represents an account, or agent, in the economy. Entry a_{ij} is positive if agent j receives funds from agent i. A SAM is a *snapshot* of the critical variables in a general equilibrium model describing the circular flow of financial transactions in an economy. For balancing problems arising from estimating SAMs, the balance conditions are the a priori accounting identities that each agent's total expenditures and total receipts must be equal. That is, for each index $i = 1, 2, \ldots, n$ of A, the sum of the entries in row i must equal the sum of the entries in column i. Pyatt and Round (1985) contains a collection of papers that give an introduction to Social Accounting Matrices.

The agents of an economy include institutions, factors of production, households, and the rest-of-the-world (to account for transactions with the economies of other countries). Figure 1 shows the major relationships between accounts in a simplified SAM. Briefly, the production activities generate value-added which flows to the factors of production—land, labor, and capital. Factor income is the primary source of income for institutions—households, government and firms—who purchase goods and services supplied by productive activities, thereby completing the cycle. This simple model produces a SAM with three agents and three nonzero entries. Of course, to be useful for equilibrium modeling, this highly aggregated model must be disaggregated into subaccounts for each sector of the economy. Estimation of disaggregated SAMs with up to 260 agents has been reported in the literature (Barker, van der Ploeg and Weale 1984), although their solution involved a major computational study. As the methods we are studying become widely known, it is likely that much larger SAMs will be estimated and their solution will be a routine operation.

SAMs are used as the core data base for complex economy-wide general equilibrium models. The

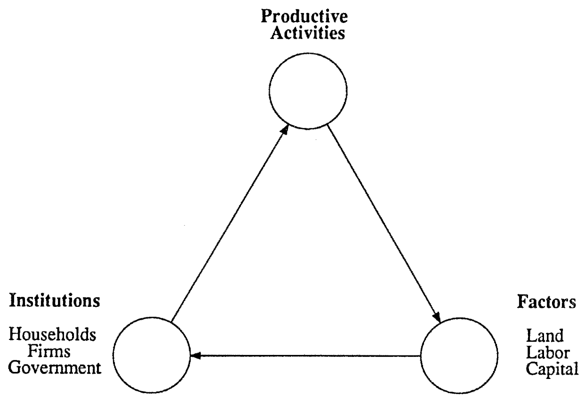


Figure 1. A simplified SAM.

entries of the SAM provide a convenient data base to estimate parameters in an equilibrium model (see, for example, Adelman and Robinson 1978 or Dervis, De Melo and Robinson 1982). For example, researchers at the World Bank have developed specialized modeling systems based on SAM data bases (Drud and Kendrick 1986). Similar models have been developed by the United Nations Statistical Office (van Tongeren 1986, 1987), the Cambridge Growth Research Project (Bacharach 1970, Stone 1985), and the Statistical Bureaus of developing and industrialized countries (Pyatt and Round).

Inconsistent data are an inherent problem when statistical methods are used to estimate underlying economic models (see, for example, Morgenstern 1963). In particular, the direct estimate of a SAM is never balanced. The following quote of Sir Richard Stone (van der Ploeg 1982, page 186) summarizes the sources of inconsistency in SAM modeling.

... it is impossible to establish by direct estimation a system of national accounts free of statistical discrepancies, residual errors, unidentified items, balancing entries and the like since the information available is in some degree incomplete, inconsistent and unreliable. Accordingly, the task of measurement is not finished when the initial estimates have been made and remains incomplete until final estimates have been obtained which satisfy the constraints that hold between their true values.

Therefore, the raw estimates of a SAM must be adjusted so that the consistency requirements are satisfied. The balancing problem of adjusting the initial matrix so that the row and column sums are equal is an example of Problem 2. This problem motivates much of our research on matrix balancing.

A matrix balancing problem also arises when *partial survey* methods are used to estimate a SAM. Frequently, estimates are available of the total expenditures and receipts for each agent in an economy but current data are not available for the individual transactions between the agents. If a complete (balanced) SAM is available from an earlier period, then the SAM must be updated to reflect the recent index totals. The problem is then to adjust the entries of the old matrix A so that the row and column totals equal the given fixed amounts; this is an example of Problem 1 (see Miller and Blair 1985). A similar balancing problem occurs when the entries of an input–output matrix must be updated to be consistent with exogenous estimates of the total levels of primary inputs and final demands (see, for example, Morrison and Thumann 1980, Harrigan and Buchanan 1984 or Jensen and McGaurr 1977).

1.2. Transportation: Estimating Origin–Destination Matrices

Networks are used in urban transportation analysis to model traffic flow over physical transportation systems involving, for example, urban highways, subways, bus routes, or intercity airways. The transportation system is represented as a network with vertices corresponding to intersections and arcs corresponding to travel links. Each arc has an associated *travel cost function* representing the disutility of travel (which we will refer to as travel time) as a function of the total flow along that arc. A user entering the transportation network must choose a path from an origin vertex to a destination vertex. The matrix representing the number of travelers between each pair of vertices is called the *origin–destination matrix*.

The *traffic assignment* problem is to determine the flow along each arc and the travel time between each origin–destination pair given the network representing the system, the travel cost functions, and the origin–destination matrix. For example, under the assumption that users choose the path with the lowest travel time, the system is in equilibrium if no motorist can lower travel time by altering his route. The resulting steady-state distribution of traffic flows is called a *user equilibrium* (see, for example, Sheffi 1985 or Abdfulaal and LeBlanc 1979). We assume that the model represents the network at a peak travel time and that the number of *trips* is fixed. That is, the total travel demand is represented by a fixed $n \times m$ *origin–destination matrix* A, where a_{ij} is the rate of flow from origin zone i to destination zone j.

The origin–destination matrix, A, is a required input for a wide variety of transportation planning models (LeBlanc and Farhangian 1982, Nguyen 1984). Matrix balancing problems arise when origin–destination matrices are estimated from observed traffic flows on selected sets of arcs (Carey, Hendrickson and Siddharthan 1981, Jefferson and Scott 1979, McNeil 1983), Van Zuylen and Willumsen 1980). For example, data might be collected that measure the total flow out of each origin vertex and into each destination vertex. The estimated origin–destination matrix should have the property that when it is used as an input into the underlying traffic assignment model, the traffic flows produced as outputs should equal the observed traffic flows. Since there may be many matrices with this property, the estimated matrix is chosen by minimizing the distance between A and a fixed reference origin–destination matrix A_0, which could be an unreliable estimate based on a previous study (see, for example, Sheffi or Nguyen).

In practice, the total flow out of each origin and into each destination is frequently used for observed traffic flows. If there is a single route connecting each origin and destination, then the resulting problem of estimating an origin–destination matrix can be formulated as Problem 1.

A more general framework for estimating origin-destination matrices is produced by assuming that a_{ij}, the aggregate flow from origin i to destination j, is determined by a direct demand function

$$a_{ij} = g(\alpha_{ij}, \beta) + \epsilon_{ij} \quad \text{for each } (i, j)$$

where α_{ij} is a fixed vector describing the socioeconomic characteristics and travel time for pair (i, j), and ϵ_{ij} is an error term. The vector β is then estimated together with the matrix A (Nguyen, Ben-Akiva 1987). For fixed flows, β is estimated using least-squares, whereas for fixed β, A is estimated using matrix balancing. Therefore, this problem can be decomposed by alternately fixing A and estimating β, and then fixing β and estimating A. This produces an iterative method converging to optimal joint estimates for A and β. The subproblem of estimating A given a fixed vector β is precisely Problem 1 (Carey, Hendrickson and Siddharthan).

1.3. Statistics: Contingency Tables

Contingency tables are used to classify items by several criteria, each of which is partitioned into a finite number of categories. For example, suppose that N individuals in a population are classified according to the criteria marital status and age, which are divided into m and n categories, respectively. The mn distinct classifications are called *cells*, and the resulting $m \times n$ contingency table contains the number of individuals in each cell.

For many applications, it is important to estimate the underlying cell probabilities. Determining the cell probabilities directly by sampling the population is generally not possible because such a procedure is often prohibitively expensive. Thus, cell probabilities must be derived from partial observations. A matrix balancing problem arises when prior values for the cell probabilities are revised so that they are consistent with known information. For example, suppose that the distribution of each criterion within the population is known and that prior cell values are given. The prior values could be obtained from an out-of-date general census, or from another population with similar ethnic and socioeconomic characteristics, or from a contingency table derived from sampling. The prior cell values must be adjusted so that they are consistent with the known marginal probabilities. This problem can be formulated as determining a matrix with given row and column sums that is *close* to the given matrix of prior cell values. This is an instance of Problem 1.

Deming and Stephan (1940) and Stephan (1942) describe applications of matrix balancing procedures for deriving probability estimates from the 1940 census data for the United States. Friedlander (1961) discusses the use of these techniques by the British government for estimating cell probabilities given known marginal probabilities. Ireland and Kullback (1968) discuss algorithms for estimating the underlying cell probabilities of two-way contingency tables based on the principle of minimum discrimination information—entropy minimization—and describe the extension to four-way tables with different marginal totals (one-, two- and three-way marginal totals are given). See also, Darroch and Ratcliff (1982) and Nishisato (1980) for a discussion of further applications of scaling techniques in statistics.

1.4. Demography: Interregional Migration

Estimating interregional migration flows based on partial and outdated information is a recurrent problem in demography. Such problems arise when figures are available for net in- and out-migration from every region and an estimate is needed of the interregional migration patterns. In the United States, for example, flow matrices with detailed migration characteristics become available once every ten years from the general census of the population. In the interim, however, net migration estimates for every region are available as by-products from annual population estimates. (Net migration is the difference between total population change and changes from births and deaths.) An updated migration matrix is then needed that reconciles the out-of-date migration patterns with the more recent net figures.

Demographers have postulated several models for estimating interregional migration including gravity models, Markov or fixed transition probability models, and doubly constrained minimum information (MI) models. For a discussion of the alternative formulations to this problem see, for example, Plane (1982) or Eriksson (1980). Plane develops the doubly constrained MI model and shows that it contains as special cases both the gravity and the fixed rate models.

The doubly constrained MI model gives rise to a matrix balancing problem as follows: Let M denote total population in all regions, and for each $i = 1, 2, \ldots, n$, let O_i and I_i be, respectively, the net out-migration and in-migration for region i. Let x_{ij} for $i, j = 1, 2, \ldots, n$, $i \neq j$ be the model estimated probabilities that any individual in the system in a region i to region j migrant, and let a_{ij} be some prior estimate of the probabilities. The MI model can be written as

$$\min_{x} \sum_{i=1}^{n} \sum_{\substack{j=1 \\ j\neq i}}^{n} x_{ij}\left[\ln\left(\frac{x_{ij}}{a_{ij}}\right) - 1\right]$$

$$\text{subject to} \quad \sum_{\substack{j=1 \\ j\neq i}}^{n} x_{ij} = \frac{O_i}{M} \quad \text{for } i = 1, 2, \ldots, n$$

$$\sum_{\substack{i=1 \\ i\neq j}}^{n} x_{ij} = \frac{I_j}{M} \quad \text{for } j = 1, 2, \ldots, n.$$

The constraints are precisely those of Problem 1. An additional constraint is often imposed on the total distance (D) traveled by the migrants. If d_{ij} is the distance between regions i and j, the additional constraint is

$$\sum_{i} \sum_{j\neq i} x_{ij} d_{ij} = D.$$

1.5. Stochastic Modeling: Estimating Transition Probabilities

The following problem arises when the underlying transition probability matrix of a discrete time Markov chain with finite state space is estimated from macro data:

> Given the observed frequency distribution of a Markov chain over n states during each of T time periods, estimate the (one-step) transition probabilities.

Problems of this sort typically appear in marketing research. For example, for a particular product a market survey is conducted to estimate the proportion of customers using brand i where $i = 1, 2, \ldots, n$ during each of the time periods $t = 0, 1, \ldots, T$. Assuming that the transition probabilities are constant during the time interval of the survey, we want to estimate the underlying transition probability matrix. Let x_{it} be the probability of the ith state during period t, and let x_t be the (column) vector $(x_{it}; i = 1, 2, \ldots, n)^T$. Let a_{ij} be the probability of transition from state i to state j, and let A be the transition probability matrix (a_{ij}), $i, j = 1, 2, \ldots, n$.

The probability vector at time $t + 1$ can be obtained from that at time t by applying the transition probability matrix A as

$$x_{t+1} = A^T x_t \quad \text{for } t = 0, 1, \ldots, T-1 \tag{1}$$

where A^T is A transpose. In practice, the vectors x_t are estimated by the observed frequency distributions, and matrix A is computed so that the equations in (1) are satisfied approximately. There are $n(n - 1)$ probabilities a_{ij} to be estimated and, in general, $T > n$ and there is no matrix A such that (1) is satisfied exactly.

For any given matrix A we call the vectors $x_{t+1} - A^T x_t$ the *discrepancy vectors.* Given a symmetric positive semidefinite matrix V we can define a quadratic function by

$$\sum_{t} (x_{t+1} - A^T x_t)^T V (x_{t+1} - A^T x_t). \tag{2}$$

The matrix V could be, for example, an estimate of the inverse of the covariance matrix of the discrepancy vectors. Since matrix A must be a Markov matrix, we require that for each row the sum of the entries be equal to one and that all entries be nonnegative. Thus, the matrix estimation problem can be modeled as minimizing (2) subject to the constraints

$$\sum_{j=1}^{n} a_{ij} = 1 \quad \text{for } i = 1, 2, \ldots, n$$

and

$$a_{ij} \geqslant 0 \qquad \text{for every } i, j.$$

The resulting problem is a reduced version of Problem 1 without constraints on the column sums. The estimated matrix A is not related to any prior estimate of the transition probabilities. Instead the model is minimizing a quadratic function of the discrepancy vectors which, in turn, depend on the computed matrix A.

The following is a typical example. Consider the market shares of new and used cars among car buyers in the United States over a period of five years. We want to determine the probability of a buyer of a used car switching over to a new car and vice versa. One is therefore estimating the entries of a Markovian matrix, subject to the constraints that all entries are between 0 and 1 and that the sum of the probabilities over all states is equal to 1. The estimation of transition probabilities from a set of data on the market shares of three cigarette brands is discussed in Theil and Rey (1966).

2. SCALING ALGORITHMS FOR MATRIX BALANCING

In this section, we describe two scaling algorithms for matrix balancing—the **RAS** algorithm for Problem 1, and the **DSS** algorithm for Problem 2. Scaling algorithms are simple iterative algorithms tailored, somewhat, to specific problems. They are easy to implement and require minimal programming effort to produce effective, numerically stable algorithms.

A *scaling of a matrix A* is defined to be pre- and post-multiplication by positive-definite diagonal matrices (of the appropriate dimensions). An algorithm is a *scaling algorithm* if each iteration is a scaling. In this section, we describe some known methods and results for scaling algorithms and describe some of our research on scaling methods for Problem 2. These algorithms can be interpreted as coordinate ascent methods applied to the dual of an entropy minimization problem (see Cottle, Duval and Zikan 1986, Lamond and Stewart 1981, Schneider 1989a, b, Schneider and Schneider 1988, and Tseng and Bertsekas 1987).

2.1. The RAS Algorithm

The **RAS** algorithm is a scaling method for solving Problems 1 in which an $m \times n$ nonnegative matrix A must be adjusted so that its row and column totals equal given positive vectors u and v. The **RAS** algorithm alternates between scaling rows and columns of A. Thus, each row is multiplied by a positive constant so that the row has the desired sum. Then the same operation is performed on the columns forcing the desired column sums. Of course, scaling the columns changes the row sums (and, similarly, scaling the rows changes the column sums). The algorithm iterates between row and column scalings until the matrix is balanced.

The **RAS** algorithm is an old method that has been independently discovered and analyzed by researchers working in different areas and in different countries. Kruithof (1937) proposed the algorithm, which he called the *method of twin factors*, as a procedure for predicting traffic flows between exchanges in a telephone service network. Deming and Stephan used **RAS**, which they called the *method of iterative proportions*, to find an approximate solution to the least-squares problem for estimating the cell frequencies of a contingency table with known marginal totals (see also, Stephan). Other early results are also contained in Bacharach (1970), Bregman (1967), and Sinkhorn (1964). Bregman attributes the method to the Russian architect Sheleikhovskii in the 1930s.

The algorithm is formally defined as follows.

The RAS Algorithm

Input. An $m \times n$ nonnegative matrix A and positive vectors u and v of dimensions m and n, respectively.

Step 0. (Initialization) Set $k = 0$ and $A^0 = A$.

Step 1. (Row Scaling) For $i = 1, 2, \ldots, m$ define

$$\rho_i^k = \frac{u_i}{\sum_j a_{ij}^k}$$

and update A^k by

$$a_{ij}^k \leftarrow \rho_i^k a_{ij}^k, \quad i = 1, 2, \ldots, m \quad \text{and} \quad j = 1, 2, \ldots, n.$$

Step 2. (Column Scaling) For $j = 1, 2, \ldots, n$ define

$$\sigma_j^k = \frac{v_j}{\sum_i a_{ij}^k}$$

and define A^{k+1} by

$$a_{ij}^{k+1} = a_{ij}^k \sigma_j^k, \quad i = 1, 2, \ldots, m \quad \text{and} \quad j = 1, 2, \ldots, n.$$

Step 3. Replace $k \leftarrow k + 1$, and return to Step 1.

The graph structure underlying matrix balancing problems is valuable for understanding both the mathematical structure and the algorithms. The connection to graph theory is established by associating with a matrix A a directed graph $G = (V, E)$ representing the sparsity pattern of A. For Problem 1 the natural graph to associate with A is a bipartite whose vertices correspond to the rows and columns of A. We need to introduce some notation so that we can give consistent definitions of graphs. For a positive integer n we define the set $\{1', 2', \ldots, n'\}$ to be a copy of the integers $\{1, 2, \ldots, n\}$ where i' is the image of i. In the following definition of the transportation graph (and the transportation graph with backarcs in Section 3) the (i, j)th entry of the matrix A is associated with the arc (i, j') from vertex i to vertex j'.

Definition 1. *For an $m \times n$ nonnegative matrix A, define the transportation graph of A to be the bipartite graph $G = (V, E)$ where*

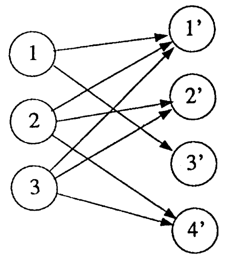
$$V = \{1, 2, \ldots, m\} \cup \{1', 2', \ldots, n'\}$$

and

$$E = \{(i, j') \mid 1 \leq i \leq m,\ 1 \leq j \leq n \text{ and } a_{ij} > 0\}.$$

(See Figure 2.) When there is no possibility of confusion, we will denote an arc of E by (i, j) rather than (i, j').

There is an obvious correspondence between nonnegative matrices and positive functions defined on the arcs of a bipartite graph. That is, the positive elements of A are viewed as weights assigned to the



$$A = \begin{bmatrix} 1 & 0 & 4 & 0 \\ 3 & 2 & 0 & 5 \\ 4 & 1 & 0 & 8 \end{bmatrix}$$

Figure 2. The transportation graph of A.

associated arcs of G. Thus, the element $a_{ij} > 0$ is the weight of arc (i, j) in the (transportation) graph of A. We will identify a nonnegative matrix X with the positive function x_{ij} defined on the arcs $(i, j) \in E$.

It is easy to see that the balance conditions for Problem 1 are the constraints

$$\begin{aligned} &\sum_{\{j \mid (i,j) \in E\}} x_{ij} = u_i && \text{for } i = 1, 2, \ldots, m \\ &\sum_{\{i \mid (i,j) \in E\}} x_{ij} = v_j && \text{for } j = 1, 2, \ldots, n \qquad (3) \\ &x_{ij} \geqslant 0 && \text{for } (i, j) \in E. \end{aligned}$$

These are precisely the constraints of a (sparse) transportation problem. The original matrix A is used as the initial approximate solution for (3).

Within the graph model, the **RAS** algorithm cycles through the vertices of G and multiplies the weights on all arcs incident to a vertex by the positive constant necessary to force the sum of the weights equal to the desired total. Bregman showed that if (3) is feasible, then the sequence $\{A^k\}$ generated by the **RAS** algorithm converges to the (unique) balanced matrix that minimizes

$$\sum_{(i,j) \in E} x_{ij} \ln(x_{ij}/a_{ij})$$

subject to (3). Since scaling cannot add any strictly positive entries to A, the feasibility of (3) is the weakest condition under which $\{A^k\}$ can converge to a balanced matrix.

A stronger convergence result for the **RAS** algorithm is also possible. Let X be the limit of the sequence A^k generated by the **RAS** algorithm. Then each iterate A^k is a scaling of the original matrix A. Stronger assumptions are needed, however, to ensure that X is also a scaling of A, that is, to ensure that there exist positive–definite diagonal matrices R and S of dimensions n and m, respectively, such that $X =$ **RAS**. Note that estimating doubly-stochastic matrices is a special case of Problem 1 in which $m = n$, and u and v are vectors of all 1's. For this case, Sinkhorn showed that if A is strictly positive, then such matrices R and S exist. This was generalized to necessary and sufficient conditions by Sinkhorn and Knopp (1967) and Brualdi, Parter and Schneider (1966) and to the general case (i.e., $m \neq n$ and arbitrary vectors u and v) by Brualdi (1968) and Menon and Schneider (1969). These conditions are based on the underlying sign pattern of matrix A (see also, Marshall and Olkin 1968).

If the limit matrix $X =$ **RAS** for positive-definite diagonal matrices R and S, then it can be shown that

$$r_{ii} = \prod_{k=1}^{\infty} \rho_i^k \quad \text{for } i = 1, 2, \ldots, m$$

and

$$s_{jj} = \prod_{k=1}^{\infty} \sigma_j^k \quad \text{for } j = 1, 2, \ldots, n$$

where ρ_i^k and σ_j^k are defined by Steps 1 and 2 of the **RAS** algorithm (Bachem and Korte 1979). This structural form of the output from the **RAS** algorithm is, presumably, the origin of the name of the algorithm.

2.2. The Diagonal Similarity Scaling (DSS) Algorithm

In this section, we describe the Diagonal Similarity Scaling (**DSS**) Algorithm for solving Problem 2 in which a given square matrix A must be adjusted so that for each index i, the sum of the elements in the ith row and ith column are equal. The method is essentially analogous to the **RAS** algorithm for Problem 1. The method scans the indices of A and finds a row–column pair that maximally violates the *row sum equals column sum* condition. The resulting row and column are scaled by multiplicative reciprocals so that their sums are equal. This operation is repeated producing an infinite sequence of matrices which, under mild conditions, converges to a balanced matrix.

We first state the algorithm and then summarize some of the relevant convergence analysis. Let A be a $n \times n$ nonnegative matrix. We want to adjust A so that

$$\sum_{j=1}^{n} a_{ij} = \sum_{j=1}^{n} a_{ji} \quad \text{for } i = 1, 2, \ldots, n.$$

The **DSS** algorithm for balancing A is formally described as follows.

The Diagonal Similarity Scaling (DSS) Algorithm

Input. An $n \times n$ nonnegative matrix A.
Step 0. (Initialization) Set $k = 0$ and $A^0 = A$.

Step 1. (Index Selection) For $i = 1, 2, 3, \ldots, n$ define

$$u_i = \sum_{j=1}^{n} a_{ij}^k \quad \text{and} \quad v_i = \sum_{j=1}^{n} a_{ji}^k.$$

Define p as the minimum index satisfying

$$|u_p - v_p| = \max_{1 \le i \le n} |u_i - v_i|.$$

Step 2. (Compute Scaling) Define α_k so that

$$\alpha_k u_p = \frac{1}{\alpha_k} v_p.$$

That is

$$\alpha_k = \sqrt{\frac{v_p}{u_p}}.$$

Step 3. (Update) Define A^{k+1} by

$$a_{ij}^{k+1} = \begin{cases} \alpha_k a_{ij}^k & \text{if } i = p, j \ne p \\ \dfrac{1}{\alpha_k} a_{ij}^k & \text{if } j = p, i \ne p \\ a_{ij}^k & \text{otherwise} \end{cases}$$

Step 4. Set $k \leftarrow k + 1$, and return to Step 1.

It is easy to see that the **DSS** algorithm is a matrix scaling algorithm. Let D^k be the diagonal matrix whose ith diagonal entry for $i = 1, 2, \ldots, n$ is defined by

$$d_{ii}^k = \begin{cases} \alpha_k & \text{if } i = p \text{ and} \\ 1 & \text{otherwise.} \end{cases} \tag{4}$$

Then A^{k+1} defined in Step 3 of the **DSS** algorithm is the matrix scaling

$$A^{k+1} = D^k A^k (D^k)^{-1}.$$

Thus, in Step 3 the pth row of A^k is multiplied by α_k and the pth column is multiplied by α_k^{-1}, thereby balancing the pth index.

The **DSS** algorithm can also be described in graphic terms as iteratively multiplying weights in a directed graph. The natural graph for Problem 2 is a *transshipment graph* with n vertices and an arc for every nonzero entry of A.

Definition 2. *For an $n \times n$ nonnegative matrix A, define the transshipment graph of A to be the directed graph $G = (V, E)$ where*

$$V = \{1, 2, 3, \ldots, n\}$$

and

$$E = \{(i, j) \mid 1 \le i, j \le n, a_{ij} > 0\}.$$

(See Figure 3.)

As in Section 2.2, there is an obvious correspondence between $n \times n$ nonnegative matrices and positive functions defined on the arcs of the transshipment graph. Again, we will identify the matrix X with the positive function x_{ij} defined on the arcs $(i, j) \in E$. The balance conditions require that the weight function x defined on E satisfies flow conservation in G. That is, the balance conditions are equivalent to requiring that the final estimated matrix X satisfies

$$\sum_{\{j \mid (i,j) \in E\}} x_{ij} - \sum_{\{j \mid (j,i) \in E\}} x_{ji} = 0 \quad \text{for } i = 1, 2, \ldots, n$$

and (5)

$$x_{ij} \ge 0$$

The **DSS** algorithm scans the vertices of G and selects the vertex p that maximally violates the flow conservation conditions. Then the weights on arcs directed into and out of p are scaled so that flow conservation is satisfied at vertex p.

For an arbitrary nonnegative matrix A, the sequence A^k generated by the **DSS** algorithm will converge to a unique balanced matrix X. (A constant can be added to the diagonal elements to ensure that division by zero never occurs.) If A is irreducible (or, equivalently, if the graph G is the disjoint union of strongly connected graphs), then the limit matrix X is given by $X = DAD^{-1}$ where

$$D = \prod_{k=1}^{\infty} D^k$$

and D^k is defined by (4). Furthermore, the limit matrix X is the unique minimizer of the function

$$\sum_{(i,j) \in E} x_{ij} \left[\ln\left(\frac{x_{ij}}{a_{ij}}\right) - 1 \right] \tag{6}$$

subject to the constraints (5). The algorithm is similar to the preconditioning scheme of Osborne (1960) and Grad (1971). Further existence and uniqueness results

$$A = \begin{bmatrix} 0 & 4 & 3 & 0 \\ 2 & 0 & 1 & 3 \\ 0 & 0 & 0 & 6 \\ 5 & 1 & 0 & 0 \end{bmatrix}$$

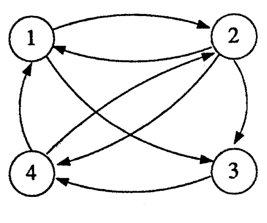


Figure 3. The transshipment graph of A.

for diagonal similarity scaling can be found in Eaves et al. (1985) and Schneider (1989a, b).

The **DSS** algorithm can be extended to handle lower and upper bounds on the original weights by imposing bounds on the equivalent entropy minimization problem. That is, given lower and upper bounds l_{ij} and u_{ij} for each $(i, j) \in E$, we can minimize (6) subject to (5) and the constraints $l_{ij} \leq x_{ij} \leq u_{ij}$. In this more general setting, the algorithm is a diagonal similarity scaling together with a truncation step to ensure that the entries of the matrix remain within their bounds. The resulting matrix balancing problem, which we call *truncated matrix scaling*, is described in Schneider (1989a, b).

3. NETWORK OPTIMIZATION ALGORITHMS FOR MATRIX BALANCING

Large-scale linear and nonlinear network models can be solved efficiently using special purpose algorithms. This is due primarily to the development of algorithms that exploit the structure of the network basis. Reports in the literature discuss the solution of linear network problems with tens of thousands of variables and nonlinear problems with thousands of variables. The status of large-scale network optimization is surveyed in Dembo, Mulvey and Zenios (1989).

Network models underlying Problems 1 and 2 were introduced in Sections 2.1 and 2.2. One of the earliest observations on the equivalence between matrix balancing problems and network optimization is found in Bacharach. The same observation was made in Byron (1978). None of the above studies, however, exploits the network structure in the developed algorithms.

Bachem and Korte (1978) designed a nonlinear programming algorithm for the transportation networks motivated by Problem 1. They later characterized minimum norm solutions for the same problem class (Bachem and Korte 1980). Cottle, Duval and Zikan used Lagrangian relaxation for the same problem, exploiting the structure of the bipartite transportation graph. Klincewicz (1989) developed an exact dual Newton algorithm for the same problem.

Zenios, Drud and Mulvey (1989) discuss the use of nonlinear network models for the estimation of social accounting matrices in an operational setting. They suggest a modification of the transportation model that offers additional modeling flexibility. They report on the development of the modeling system GAMS/SAMBAL that integrates a network optimizer, GENOS of Mulvey and Zenios (1988), with a high level modeling language, GAMS of Bisschop and Meeraus (1982).

The nonlinear network model we consider in the sequel is a transportation graph with one node for every row and one for every column in the matrix and one arc for every nonzero entry. In addition, *backarcs* are introduced to convert the transportation into a circulation network and to enforce the condition that total flow from a column is equal to the total inflow in the corresponding row, as in Problem 2.

Definition 3. *For an $n \times n$ nonnegative matrix A, define the transportation graph of A with backarcs to be the directed graph $G = (V, E)$ where*

$$V = \{1, 2, \ldots, n\} \cup \{1', 2', \ldots, n'\}$$

and

$$E = E_1 \cup E_2$$

where

$$E_1 = \{(i, j') \mid 1 \leq i, j \leq n, \text{ and } a_{ij} > 0\}$$

and

$$E_2 = \{(i', i) \mid i = 1, 2, 3, \ldots, n\}.$$

(See Figure 4.)

For notational convenience, we denote an arc of E_1 by (i, j) (rather than (i, j')) and denote the flow on

	LAB	H1	H2	P1	P2	TOT
LAB		•	•	•	•	•
H1	•					•
H2	•					•
P1		•	•		•	•
P2		•	•	•		•
TOT	•	•	•	•	•	

(Blank entries indicate impossible transactions)

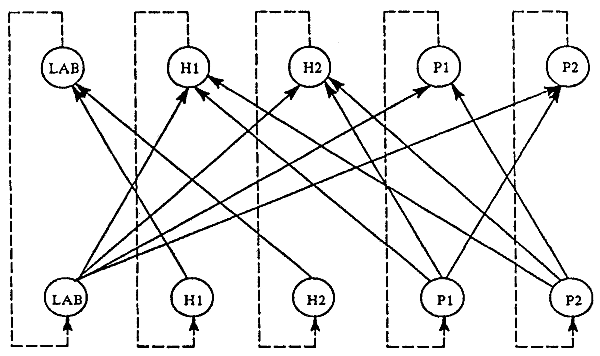


Figure 4. A simple SAM and the corresponding network model.

arc (i, j) by x_{ij}. Similarly, we denote an arc of E_2 by (i, i) and denote the flow by y_{ii}. The context will make it clear, for example, whether (2, 2) is the arc (2, 2′) of E_1 or the arc (2′, 2) of E_2. The arcs of E_2 are called *backarcs*.

As in Section 2.2, we identify an $n \times n$ matrix X with the positive function x_{ij} defined on the arcs $(i, j) \in E_1$. We use $Y = (y_{ii} \mid i = 1, 2, \ldots, n)$ to denote the vector of flows on the backarcs E_2. Using the transportation graph with backarcs, the balance constraints that the adjusted matrix must satisfy can be expressed as

$$\sum_{\{j \mid (i,j) \in E_1\}} x_{ij} = y_{ii} \quad \text{for } i = 1, 2, \ldots, n$$

$$\text{and} \qquad (7)$$

$$\sum_{\{i \mid (i,j) \in E_1\}} x_{ij} = y_{jj} \quad \text{for } j = 1, 2, \ldots, n.$$

That is, if (X, Y) is interpreted as a flow in the underlying graph, then flow conservation must hold at every vertex, and (X, Y) must be a circulation. An optimization problem is formulated by requiring that X be as close as possible to the original data A, and Y be as close as possible to estimates on the marginal totals, which we denote by $\eta = \{\eta_i\}$. Deviations are measured by a penalty function

$$F(X, Y; A, \eta) = \sum_{(i,j) \in E_1} f_{ij}(x_{ij}; a_{ij}) + \sum_{(i,i) \in E_2} g_i(y_{ii}; \eta_i).$$

(We will write just $f_{ij}(x_{ij})$, $g_i(y_{ii})$, omitting the dependence on a_{ij} and η_i.) In addition, it is often the case in applications that the modeler knows the range within which the variables should be forced to lie. These constraints are incorporated into the model by specifying lower and upper bounds, l_{ij} and u_{ij}, on the variables x_{ij}. Then the network optimization model of the balancing problem can be formulated as

$$\min_{x} \left\{ \sum_{(i,j) \in E_1} f_{ij}(x_{ij}) + \sum_{(i,i) \in E_2} g_i(y_{ii}) \right\}$$

$$\text{subject to constraints (7)} \qquad (8)$$

$$l_{ij} \leq x_{ij} \leq u_{ij}.$$

In practice, either quadratic or entropy functions are used for balancing matrices. These are defined as (see Figure 5)

- Quadratic Penalty

$$f_{ij}(x_{ij}) = w_{ij}(x_{ij} - a_{ij})^2$$

or

$$g_i(y_{ii}) = w_i(y_{ii} - \eta_i)^2$$

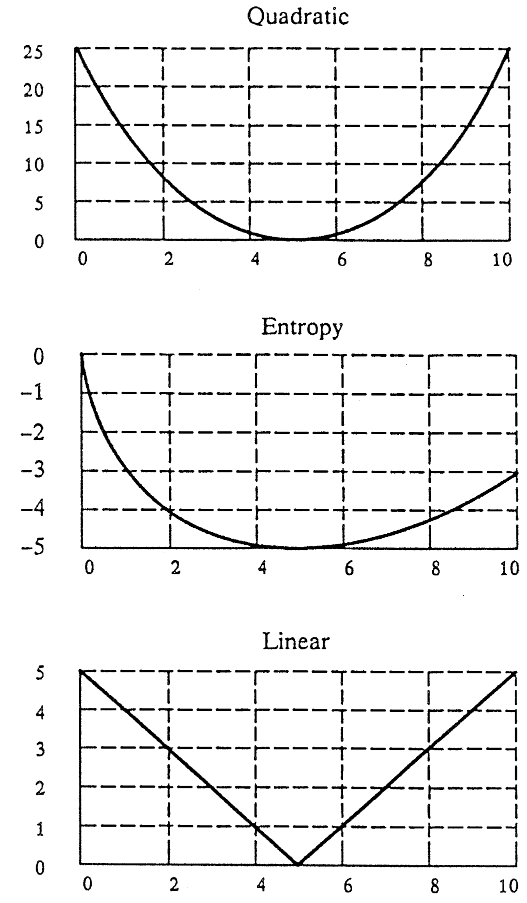


Figure 5. Penalty functions for network optimization models.

- Entropy Penalty

$$f_{ij}(x_{ij}) = w_{ij} x_{ij} \left[\ln\left(\frac{x_{ij}}{a_{ij}}\right) - 1 \right]$$

or

$$g_i(y_{ii}) = w_i y_{ii} \left[\ln\left(\frac{y_{ii}}{\eta_i}\right) - 1 \right].$$

The quadratic penalty function can be extended to a nonseparable function of the form

$$\sum_{(i,j),(k,l)} (x_{ij} - a_{ij}) w_{(ij,kl)} (x_{kl} - a_{kl}).$$

Some comments are in order. The formulation of the matrix balancing problems as a circulation network with backarcs allows the estimation of the marginal totals together with the entries of the matrix. Marginal totals may, of course, be kept fixed with the use of upper and lower bounds on the backarcs equal to the desired value.

The two penalty functions given above are the only ones used in practice, with the nonseparable extension appearing less frequently in applications. Although other penalties are possible (such as piecewise linear of the form $|x_{ij} - a_{ij}|$), the two given here generate, in general, solutions that are acceptable to the modelers. The entropy function has a theoretical justification based on principles of information theory. The quadratic function has a statistical interpretation as a generalized least-squares estimator. Using a quadratic function with terms weighted by the inverse of the original estimates ($w_{ij} = 1/a_{ij}$) results in a chi-square estimate. A piecewise linear penalty function produces problems that are computationally easy to solve because the underlying network optimization model can be converted to a linear program. There is no clear theoretical justification, however, for the use of a piecewise linear penalty. Furthermore, the solutions obtained with the nonlinear penalties have a large number of small deviations: Each entry of the matrix is adjusted in proportion to its magnitude in order to satisfy the balancing conditions. A piecewise linear penalty function, on the other hand, results in extreme point solutions in which a small number of entries are changed significantly from their original values.

The lower and upper bounds on variables are introduced to eliminate values in the balanced matrix that seem unreasonable to the modeler. They offer some control over the mechanized estimation of the matrix, but no systematic way for assigning their values is available. The objective function weights, w_{ij}, reflect the relative reliability of the original estimates. They are assigned by the user to ensure that reliable data in the original matrix are not perturbed by the optimization model. A systematic way of assigning weights is possible whenever the entries are obtained from sampling procedures and a prior estimate of the variance matrix V is available. The inverse standard error may be interpreted as an index of the reliability of the SAM entries. Usually V is diagonal, its inverse is easily computed, and a consistent weighting mechanism assigns weights as the inverse of the variance matrix. This approach has been adopted by Byron and is employed in the test problem STONE in a later section. In the nonseparable extension of the quadratic penalty a variance–covariance matrix is used as the weighing matrix with the same effect. Note, however, that its inverse may not easily be computable in this case.

Finally, several variations of the matrix balancing problems presented here can be solved using nonlinear network models. Zenios (1988) discusses a variety of formulations. For example, a simplified version of Problem 1 when $m = n$ can be solved with the network model presented here if the flows on the backarcs are fixed by setting the upper bound equal to the lower bound (and equal to the exogenous total value of the row or column). If we set, in addition, the remaining upper bounds equal to infinity and the lower bounds equal to zero, the solution with entropy penalties is, in this case, identical to the **RAS** solution.

There are important implications in recognizing that matrix balancing problems have the structure of network models. First, we can exploit the structure of the network basis in designing efficient algorithms. Second, we can use several suitable penalty functions, incorporate weights reflecting the reliability of the various estimates and bound the values of the balanced matrix.

4. CRITIQUE OF THE ALGORITHMS

In this section, we compare the different algorithms for matrix balancing problems. The goal is not so much to identify an overall *best* methodology. Rather we identify the qualities of a balancing procedure that are of importance to a potential user and indicate situations where a particular procedure might be preferred.

The three procedures discussed in Sections 2 and 3 (**RAS**, **DSS**, and network optimization) are compared with respect to the following criteria: 1) applicability, 2) ability to include available information, 3) ease of implementation and use, and 4) accuracy and computational efficiency (see Table I). A general discussion on the first three criteria is followed by a computational study of the algorithms on a set

Table I
Properties of the Matrix Balancing Algorithms

Characteristics	**RAS**	**DSS** Algorithm	Network Model
Entropy model	Yes	Yes	Yes
Least squares approximation	No	No	Yes
Chi-square approximation	No	No	Yes
Incorporate data reliability	No	No	Yes
Preserve nonnegativity	Yes	Yes	Yes
Incorporate user imposed bounds	No	Yes	Yes
Estimate unknown row-column totals	No	Yes	Yes
Incorporate given row-column totals	Yes	Yes	Yes

Table II
SAM Test Problems

Problem	Account	Transactions	Description
SAM1	5	8	Demonstration test problem
SAM6	18	46	SAM models from
SAM10	25	63	Chapters 6 and 10 of Drud and Kendrick (1986)
STONE	6	12	Stone's demonstration SAM
SAMTU	8	19	SAM for Turkey 1973
SAMKE	25	177	SAM for Kenya 1976
SAMBO	63	598	SAM for Botswana 1974–75
SAMMO80	232	1664	SAM for Morocco 1980
SAMMO85	232	1664	Update SAMMO80 using totals from 1985

of balancing problems for Social Accounting Matrices (SAMs). The test problems were drawn from recent planning models by the economists of the World Bank. Their characteristics are summarized in Table II. A detailed description of the test problems (which are available for other researchers) can be found in Zenios (1986).

Applicability. The **RAS** algorithm has a restricted set of applications. It can only be used when the row and column totals of the balanced matrix are given. It does not apply to situations where the matrix totals have to be estimated along with the cell values. The network optimization and the diagonal similarity scaling procedure are applicable to both Problems 1 and 2. In some cases, users impose additional requirements for a balanced matrix, for example, the entries of a row should have a certain proportional relation to each other. Optimization models can be designed to solve such instances, although the network structure is destroyed and computations become more expensive. The **RAS** and the **DSS** algorithm cannot solve such problems.

Ability to Include Available Information. Input data for the **RAS** algorithm include the unbalanced matrix estimate and row/column totals of the balanced matrix. The network optimization and the **DSS** algorithm have the minimal data requirement of the unbalanced matrix. When the following additional data are available, both models can use them in a consistent manner: estimates for row and column totals and allowable ranges for certain elements of the balanced matrix. Both procedures may keep a subset of the elements of the balanced matrix fixed to their original values. In addition, the network model can incorporate weights for the reliability of the data of the unbalanced matrix and for the totals.

Ease of Implementation and Use. **RAS** and **DSS** are extremely easy to implement and use. Both can be implemented with very little programming effort and are reliable and stable.

The network optimization model requires the use of nonlinear programming algorithms with sophisticated data structure. Software for these problems usually requires a few years of careful programming to reach a stable and reliable state. Potential users of this model should not attempt to develop their own software, but look instead for the publicly available codes. Such codes are, in general, quite robust and efficient, but their use may require some expertise in network optimization. Recent trends in integrating network optimization capabilities into high-level modeling languages make these tools more widely accessible (Zenios 1990a).

Accuracy and Computational Efficiency. Evaluating the accuracy of the competing methodologies is a difficult task because it is not clear what constitutes an *accurate* solution. The three algorithms estimate a matrix based on different underlying assumptions that may be more appropriate for one application versus another. It suffices to say that all three procedures do not appear to suffer from any form of numerical instabilities. In addition, for instances of problems where an exact equivalence existed between the different algorithms, we observed that the same balanced matrix was obtained. For example, we used both the **RAS** algorithm and the network optimization model with entropy penalties to update the SAM for Morocco from 1980 to 1985. **RAS** produced a balanced matrix with row totals equal to prespecified values within an error of $\pm 10^{-8}$. The network algorithm produced a balanced matrix with errors $\pm 10^{-16}$. The objective value of the entropy penalty was -28760.7494, in agreement to four decimal points for both approaches. Differences between the entries of the two balanced matrices were insignificant with respect to the precision of the computer used (10^{-8}).

Our objective in evaluating the computational performance of these algorithms is to show that each method is capable of solving large-scale balancing problems effectively. Each algorithm was implemented in a different computational environment and programming language; therefore, the solution times are not directly comparable. The network optimization algorithm is written in FORTRAN 77. All testing was performed on an IBM 3081-D running CMS and the program was compiled with the optimization OPT(2) option of the FORTRAN compiler. The scaling algorithm was written in C; all testing was performed on a VAX 11/750 (with a floating point accelerator) running the UNIX operating system. The **RAS** algorithm was implemented inside GAMS using the GAMS command language; all testing was performed on the IBM 3081-D.

We solved seven of the test problems from Table II with scaling and all nine test problems with network optimization. The results of these experiments are summarized in Tables III and IV. The **RAS** algorithm cannot solve the first eight test problems since the row and column totals are unknown. The last problem, SAMMO85, required updating the 1980 SAM for Morocco so that it was consistent with new 1985 estimates of the total expenditures in each account. Therefore, it is a problem of Type 1 and was solved using network optimization and the **RAS** algorithm. The results are summarized in Table V. The **RAS** algorithm is a much simpler method and solves the problem in less time than the network algorithm. However, **RAS** is basically a dual algorithm and the solution obtained does not satisfy exactly the balancing constraints. The maximum (percentage) difference between the sum of any row and the prespecified total is $10^{-4}\%$.

Table III
Balancing Time Using Network Optimization

Problem	Penalty	Objective	CPU Seconds
SAM1	Quadratic	0.5483	0.04
	entropy	−222.7254	0.06
SAM6	Quadratic	11.7582	0.19
	entropy	−2197.0223	0.20
SAM10	Quadratic	8.4057	0.41
	entropy	−3212.7782	0.37
STONE	Quadratic	14.63	0.31
	entropy	−1025.99	0.44
SAMTU	Quadratic	6.1906	0.09
	entropy	−1643.3762	0.09
SAMKE	Quadratic	6.0907	1.75
	entropy	−7768.3461	1.41
SAMBO	Quadratic	10.9080	4.47
	entropy	−1641.8505	5.02
SAMMO80	Quadratic	0.00016	15.23
	entropy	−525441.5295	12.28

Table IV
Balancing Time Using Scaling

Problem	Tolerance	Iterations	CPU Seconds	Deviation
SAM1	10^{-4}	12	0.02	0.0080
	10^{-6}	20	0.03	0.0002
	10^{-8}	26	0.04	0.0000
SAM6	10^{-4}	83	0.37	0.2506
	10^{-6}	154	0.67	0.0029
	10^{-8}	250	1.09	0.0000
SAM10	10^{-4}	100	0.65	0.6819
	10^{-6}	274	1.62	0.0056
	10^{-8}	451	2.65	0.0001
STONE	10^{-4}	12	0.02	0.0242
	10^{-6}	16	0.02	0.0013
	10^{-8}	28	0.05	0.0000
SAMTU	10^{-4}	2	0.05	0.0962
	10^{-6}	37	0.08	0.0013
	10^{-8}	59	0.12	0.0001
SAMKE	10^{-4}	55	0.33	0.7292
	10^{-6}	125	0.75	0.0097
	10^{-8}	198	1.13	0.0001
SAMBO	10^{-4}	299	4.39	0.6051
	10^{-6}	658	9.79	0.0087
	10^{-8}	1083	16.17	0.0001

In Table IV, the results from the computational experiment are shown for three levels of the termination criteria (listed as the *Tolerance*). The scaling algorithm terminated when

$$\max_{1 \le i \le n} \frac{|r_i - c_i|}{c_i} \le \epsilon$$

where r_i and c_i are the row and column totals of the adjusted matrix. Three values of ϵ were used—10^{-4}, 10^{-6} and 10^{-8}. The table shows the number of iterations and the CPU seconds (on a VAX-11/750) needed to solve the problem. The last column, *Deviation*, measures the accuracy of the final solution defined as

$$\sum_{i=1}^{n} |r_i - c_i|.$$

Finally an attempt was made to evaluate the quality of the solutions obtained with the different algorithms. Noting that the network optimization model for Problem 1 with entropy penalties is equivalent to **RAS** we used the GAMS/SAMBAL system to emulate both algorithms on a sample test problem. The results are

Table V
Solution Times for Problem SAMMO85

Characteristic	RAS	Network Optimization
Iterations	75	20
CPU seconds	23.7	68.9
Entropy value	−28760.7494	−28760.7494

summarized in Table VI, together with Byron's solution to the same problem. Byron formulated the problem as a weighted least-squares model and used conjugated gradients for its soluiton. We observe the agreement between Byron's solution and the solution of the network model with a weighted quadratic objective. Solutions with alternative penalty functions are significantly different. This points to the difficulty of choosing a suitable model when solving practical applications, and shows the advantage of having a methodology that can readily accommodate any choice.

A large example, SAMKE, was solved using the network model with both a quadratic and an entropy penalty. No significant difference was observed between the two solutions. The average percentage difference between the entries of the two balanced matrices was 0.04% with no two values differing by more than 0.3%. Figure 6 shows the distribution of errors for both the quadratic and the entropy penalties. We observe a desirable distribution of errors around zero, with the entropy distribution slightly skewed toward positive values.

5. CONCLUSIONS

This paper has primarily shown that algorithms for the matrix balancing problem have reached the stage where large-scale problems can be solved on a routine basis. Although practitioners—mainly economists—have estimated accounting matrices with a few hundred accounts, their work involved a major computational undertaking. This need not be the case any more. We have shown that three numerical procedures are currently available for problems of this nature.

The **RAS** algorithm is certainly the easiest to use, and preliminary results indicate that it is also very efficient. It fails, however, to solve one of the two major types of problems that fall in the matrix balancing framework. Network optimization provides a powerful tool that can solve all instances of problems of this form and provides several useful extensions. For example, users may impose bounds on the values of the balanced matrix or use different models for balancing the matrix (entropy, chi-square and least-square estimates are some options). Algorithms for this problem are difficult to implement but public codes are available and are, in general, robust and efficient. The diagonal scaling algorithm provides a compromise between the simplicity of **RAS** and the complexity of the network models. The procedure is simple to implement and can be extended to provide

Table VI
Solutions to the STONE Problem Using Network Optimization

Entries of A	Original Estimate	Byron's Estimate	Quadratic Penalty	Entropy Penalty	Weighted Quadratic	Weighted Entropy
LAB.H1	15	14.54	14.53	13.93	14.54	14.94
LAB.H2	3	2.98	2.91	2.89	2.98	3.00
LAB.P1	130	125.57	126.80	127.08	125.57	124.28
LAB.P2	80	83.05	84.24	84.42	83.06	88.39
H1.LAB	Unknown	54.43	52.57	52.09	54.43	54.86
H2.LAB	Unknown	171.71	175.91	176.23	171.71	175.75
P1.H1	15	15.09	15.87	15.89	15.09	15.01
P1.H2	130	136.98	137.51	137.71	136.99	138.51
P1.P2	20	25.33	22.98	23.20	25.33	22.11
P2.H1	25	24.79	22.18	22.27	24.79	24.91
P2.H2	40	31.74	35.49	25.64	31.74	34.25
P2.P1	55	51.84	49.56	49.71	51.84	51.34
Tot-LAB	220	226.14	228.48	228.32	226.14	230.62
Tot-H1	Unknown	54.43	52.75	52.09	54.43	54.86
Tot-H2	Unknown	171.71	175.91	176.23	171.71	175.75
Tot-P1	190	177.40	176.37	176.80	177.41	175.63
Tot-P2	105	108.38	107.23	107.62	107.62	110.50

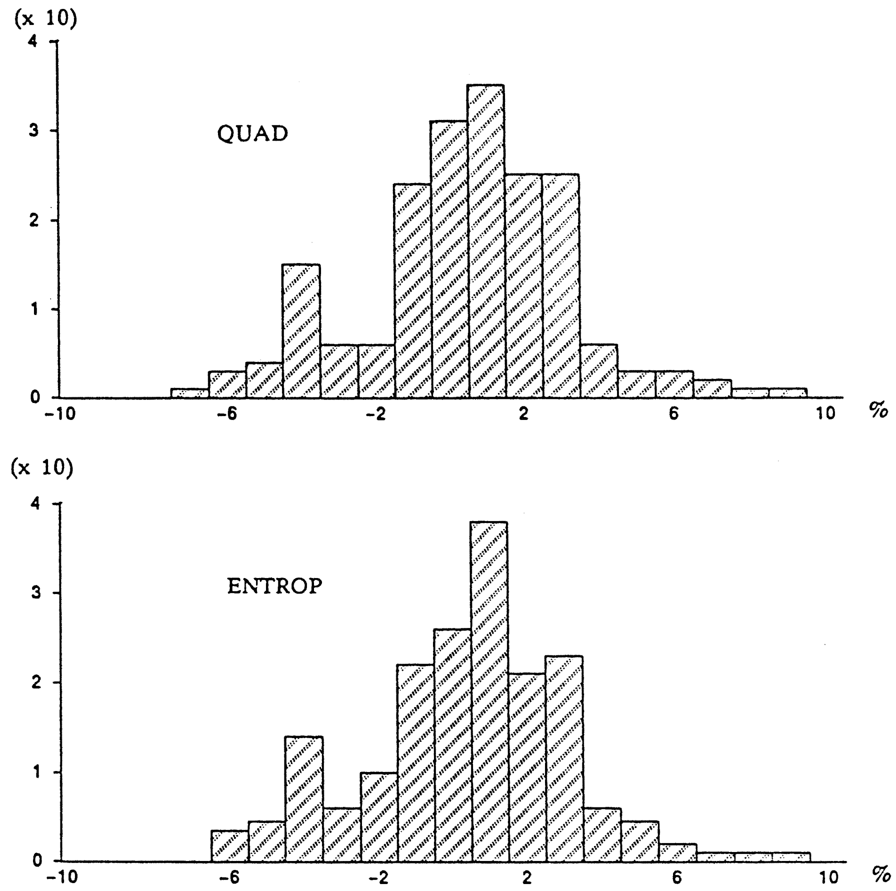


Figure 6. Distribution of errors for SAMKE.

several of the features otherwise available only by optimization models. Preliminary evidence indicates that the scaling procedure is capable of solving matrix balance problems as accurately as optimization methods.

Finally, we point out that both **RAS** and the **DSS** admit rather straightforward extensions for parallel computations. Zenios (1990b) and Zenios and Iu (1990) have shown that parallel versions of **RAS** can achieve impressive performance on vector supercomputers like the CRAY X-MP, shared memory multiprocessors, and massively parallel computers with thousands of processing elements like the Connection Machine. As a result of these developments we are able to solve matrix balancing problems with thousands of rows and columns and hundreds of thousands of nonzero entries.

ACKNOWLEDGMENT

We thank the referee for suggesting many ways in which to improve the exposition of the paper. The work of the first author was partially supported by National Science Foundation (NSF) grants ECS85-04195 and ECS87-18971. The work of the second author was partially supported by NSF grants ECS87-18971 and CCR88-11135, AFOSR grant 89-0145 and a grant from the University of Pennsylvania Research Foundation.

REFERENCES

ABDFULAAL, M., AND L. J. LEBLANC. 1979. Methods for Combining Modal Split and Equilibrium Assignment Models. *Trans. Sci.* **13,** 292–314.

ADELMAN, I., AND S. ROBINSON. 1978. *Income Distribution Policy in Developing Countries: A Case Study of Korea.* Oxford University Press.

BACHARACH, M. 1970. *Biproportional Matrices and Input-Output Change.* Cambridge University Press, London.

BACHEM, A., AND B. KORTE. 1978. An Algorithm for Quadratic Optimization Over Transportation Polytopes. *Zeitschrift fur Angewante Mathematik und Mechanik* **58,** 549–461.

BACHEM, A., AND B. KORTE. 1979. On the *RAS* Algorithm. *Computing* **23,** 189–198.

BACHEM, A., AND B. KORTE. 1980. Minimum Norm Problems Over Transportation Polytopes. *Lin. Alg. Its Applic.* **31,** 103–118.

BARKER, T., F. VAN DER PLOEG AND M. WEALE. 1984. A Balanced System of National Accounts for the United Kingdom. *The Review of Income and Wealth* **30**(4), 461–485.

BEN-AKIVA, M. 1987. Methods to Combine Different Data Sources and Estimate Origin-Destination Matrices. Technical Report, Department of Civil Engineering, MIT, Cambridge, Mass.

BISSCHOP, J., AND A. MEERAUS. 1982. On the Development of a General Algebraic Modeling System in a Strategic Planning Environment. *Math. Prog. Studies* **20,** 1–29.

BREGMAN, L. M. 1967. Proof of the Convergence of Sheleikhovskii's Method for a Problem With Transportation Constraints. *USSR Computational Math. and Mathem. Phys.* **1**(1), 191–204.

BRUALDI, R. A. 1968. Convex Sets of Non-Negative Matrices. *Can. J. Math.* **20,** 144–157.

BRUALDI, R. A., S. PARTER AND H. SCHNEIDER. 1966. The Diagonal Equivalence of a Nonnegative Matrix to a Stochastic Matrix. *J. Math. Anal. Appl.* **16**(1), 31–50.

BYRON, R. P. 1978. The Estimation of Large Social Account Matrices. *J. Roy. Statist. Soc. Ser. A* **141,** 359–367.

CAREY, M., C. HENDRICKSON AND K. SIDDHARTHAN. 1981. A Method for Direct Estimation of Origin/Destination Trip Matrices. *Trans. Sci.* **15,** 32–49.

COTTLE, R. W., S. G. DUVALL AND K. ZIKAN. 1986. A Lagrangian Relaxation Algorithm for the Constrained Matrix Problem. *Naval Res. Logist. Quart.* **33,** 55–76.

DARROCH, J. H., AND D. RATCLIFF. 1982. Generalized Iterative Scaling for Log-Linear Models. *Ann. Math. Stat.* **19,** 190–212.

DEMBO, R. S., J. M. MULVEY AND S. A. ZENIOS. 1989. Large-Scale Nonlinear Network Models and Their Applications. *Opns. Res.* **37,** 353–372.

DEMING, W. E., AND F. F. STEPHAN. 1940. On a Least Squares Adjustment of Sampled Frequency Table When the Expected Marginal Totals are Known. *Ann. Math. Stat.* **11,** 427–444.

DERVIS, K., J. DE MELO AND S. ROBINSON. 1982. *General Equilibrium Models for Development Policy.* Cambridge University Press, Cambridge.

DRUD, A., AND D. KENDRICK. 1986. HERCULES—A System for Economywide Models. Technical Report, Development Research Department, The World Bank, Washington, D.C.

EAVES, B. C., A. J. HOFFMAN, U. G. ROTHBLUM AND H. SCHNEIDER. 1985. Line-Sum-Symmetric Scalings of Square Nonnegative Matrices. *Math. Prog. Studies* **25,** 124–141.

ERIKSSON, J. 1980. A Note on Solution of Large Sparse Maximum Entropy Problems With Linear Equality Constraints. *Math. Prog.* **18,** 146–154.

FRIEDLANDER, D. 1961. A Technique for Estimating a Contingency Table Given the Marginal Totals and Some Supplementary Data. *J. Roy. Stat. Soc. Ser. A* **124,** 412–420.

GRAD, J. 1971. Matrix Balancing. *Computer J.* **14,** 280–284.

HARRIGAN, F., AND I. BUCHANAN. 1984. A Quadratic Programming Approach to Input-Output Estimation and Simulation. *J. Region. Sci.* **24,** 339–358.

IRELAND, C. T., AND S. KULLBACK. 1968. Contingency Tables With Given Marginals. *Biometrika* **55,** 179–188.

JEFFERSON, T. R., AND C. H. SCOTT. 1979. The Analysis of Entropy Models With Equality and Inequality Constraints. *Trans. Res.* **13B,** 123–132.

JENSEN, R. C., AND D. MCGAURR. 1977. Reconciliation Techniques in Input-Output Analysis: Some Comparisons and Implications. *Urban Studies* **14,** 327–337.

KLINCEWICZ, J. G. 1989. Implementing an Exact Newton Method for Separable Convex Transportation Problems. *Networks* **19,** 95–105.

KRUITHOF, J. 1937. Telefoonverkeersrekening. *De Ingenieur* **3,** 15–25.

KRUITHOF, J. Undated. Telephone Traffic Calculus. Technical Report, Bell Telephone Manufacturing Company, Antwerp, Belgium. (Translation of Kruithof 1937).

LAMOND, B., AND N. F. STEWART. 1981. Bregman's Balancing Method. *Trans. Res.* **15B,** 239–248.

LEBLANC, L. J., AND K. FARHANGIAN. 1982. Selection of a Trip Table Which Reproduces Observed Link Flows. *Trans. Res.* **16B,** 83–88.

MARSHALL, A. W., AND I. OLKIN. 1968. Scaling of Matrices to Achieve Specified Row and Column Sums. *Numerishe Mathematik* **12,** 83–90.

MCNEIL, S. 1983. Quadratic Matrix Entry Estimation Methods. Ph.D. Thesis, Department of Civil Engineering, Carnegie-Mellon University, Pittsburgh.

MENON, M. V., AND H. SCHNEIDER. 1969. The Spectrum of a Nonlinear Operator Associated With a Matrix. *Lin. Alg. Its Appl.* **2,** 321–334.

MILLER, R. E., AND P. D. BLAIR. 1985. *Input-Output Analysis: Foundations and Extensions.* Prentice Hall, Englewood Cliffs, N.J.

MORGENSTERN, O. 1963. *On the Accuracy of Economic Observations.* Princeton University Press, Princeton, N.J.

MORRISON, W. I., AND R. G. THURMANN. 1980. A Lagrangian Multiplier Approach to the Solution of a Special Constrained Matrix Problem. *J. Region. Sci.* **20,** 279–292.

MULVEY, J. M., AND S. A. ZENIOS. 1988. GENOS 1.0; A Generalized Network Optimization System. Decision Sciences Working Paper 87-12-3, The Wharton School, University of Pennsylvania, Philadelphia.

NGUYEN, S. 1984. Estimation Origin-Destination Matrices From Observed Flows. In *Transportation Planning Models*, Michael Florian (ed.). Elsevier Science Publishers, Amsterdam.

NISHISATO, S. 1980. Analysis of Categorical Data: Dual Scaling and Its Applications. Volume 24 of *Mathematical Expositions*. University of Toronto Press.

OSBORNE, E. E. 1960. On Pre-Conditioning of Matrices. *J. Assoc. Comput. Mach.* **7,** 338–345.

PLANE, D. A. 1982. An Information Theoretic Approach to the Estimation of Migration Flows. *J. Region. Sci.* **22,** 441–456.

PYATT, G., AND J. I. ROUND. 1985. Social Accounting Matrices for Development Planning. In *Social Accounting Matrices: A Basis for Planning*, G. Pyatt and J. I. Round (eds.). The World Bank, Washington, D.C.

SCHNEIDER, H., AND M. H. SCHNEIDER. 1988. A Simple Iterative Algorithm for Balancing Matrices. Technical Report, Department of Mathematical Sciences, The Johns Hopkins University, Baltimore.

SCHNEIDER, M. H. 1989a. Matrix Scaling, Entropy Minimization, and Conjugate Duality (I): Positivity Conditions. *Lin. Alg. Its Appl.* **114/115,** 785–813.

SCHNEIDER, M. H. 1990. Matrix Scaling, Entropy Minimization, and Conjugate Duality (II): The Dual Problem. *Math. Prog.* **48** (1).

SHEFFI, Y. 1985. *Urban Transportation Networks.* Prentice-Hall, Englewood Cliffs, N.J.

SINKHORN, R. 1964. A Relationship Between Arbitrary Positive Matrices and Doubly Stochastic Matrices. *Ann. Math. Statis.* **35,** 876–879.

SINKHORN, R., AND P. KNOPP. 1967. Concerning Nonnegative Matrices and Doubly Stochastic Matrices. *Pac. J. Math.* **212,** 343–348.

STEPHAN, F. F. 1942. An Iterative Method of Adjusting Sample Frequency Tables When Expected Marginal Totals are Known. *Ann. Math. Statis.* **13,** 166–178.

STONE, R. 1985. The Disaggregation of the Household Sector in the National Accounts. In *Social Accounting Matrices: A Basis for Planning*, G. Pyatt and J. I. Round (eds.). The World Bank, Washington, D.C.

THEIL, H., AND G. REY. 1966. A Quadratic Programming Approach to the Estimation of Transition Probabilities. *Mgmt. Sci.* **12,** 714–721.

TSENG, P., AND D. P. BERTSEKAS. 1987. Relaxation Methods for Problems With Strictly Convex Separable Costs and Linear Constraints. *Math. Prog.* **38,** 303–321.

VAN DER PLOEG, F. 1982. Reliability and the Adjustment of Sequences of Large Economic Accounting Matrices. *J. Roy. Statis. Soc. Ser. A* **145,** 169–194.

VAN TONGEREN, J. W. 1986. Development of an Algorithm for the Compilation of National Accounts and Related Systems of Statistics. *Rev. of Income and Wealth* **Series 32,** 25–68.

VAN TONGEREN, J. W. 1987. The SNA Structure—Synthesis of Earlier Proposals. Technical Report, Statistical Office of the United Nations, The United Nations, New York.

VAN ZUYLEN, H. J., AND L. G. WILLUMSEN. 1980. The Most Likely Trip Matrix Estimated From Traffic Counts. *Trans. Res.* **14B,** 281–293.

ZENIOS, S. A. 1986. A Library of SAM Balancing Models. Technical Report, Department of Civil Engineering, Princeton University, Princeton, N.J.

ZENIOS, S. A. 1988. Network Based Modeling Systems for Matrix Balancing. Decision Sciences Working Paper 88-09-01, The Wharton School, University of Pennsylvania, Philadelphia.

ZENIOS, S. A. 1990a. Incorporating Network Optimization Capabilities into a High-Level Programming Language. *ACU Trans. on Mathematical Software* (to appear).

ZENIOS, S. A. 1990b. Matrix Balancing on a Massively Parallel Connection Machine. *ORSA J. on Computing* **2** (2).

ZENIOS, S. A., AND S-L. IU. 1990. Vector and Parallel Computing for Matrix Balancing. *Annals of O.R.* **22,** 161–180.

ZENIOS, S. A., A. DRUD AND J. M. MULVEY. 1989. Balancing Large Social Accounting Matrices With Nonlinear Network Programming. *Networks* **19,** 569–585.