BALANCING A SOCIAL ACCOUNTING MATRIX:

Theory and application (revised edition)

André Lemelin[1]

Ismaël Fofana[2]

John Cockburn[3]

December 2013

1 Centre INRS-UCS, Université du Québec, Montréal; corresponding author: andre_lemelin@ucs.inrs.ca
2 IFPRI, Dakar
3 PEP, Université Laval, Québec

BALANCING A SOCIAL ACCOUNTING MATRIX: THEORY AND APPLICATION[1]

Summary

Computable general equilibrium modeling requires a consistent and coherent benchmark data set, most of which is generally organized in the form of a Social Accounting Matrix (SAM). These data generally come from quite diverse sources and correspond to different periods of time. As a result, they often present inconsistencies. We present a simple computer program that makes it possible to reconcile this information in order to balance a SAM. The program minimizes the changes to the base data using various optimizing techniques, including cross-entropy and least squares. The program is an attractive and easy alternative to the arbitrary and time-consuming manual and other methods generally used to balance SAMs.

Introduction

Computable General Equilibrium (CGE) modeling and Social Accounting Matrix (SAM) based research requires the use of the most recent economic data available in a coherent framework. However, these data generally come from quite diverse sources – e.g. input-output tables, national accounting data, household surveys, firm surveys, labor market surveys, government accounts, international trade accounts, etc. – and correspond to different periods of time. For example, input-output data are generally prepared every five years or more, whereas national data on income, production, trade, etc. are generated annually. In some cases, it is possible to start with an existing SAM, which is then updated using new data. In other cases, the SAM must be constructed in its entirety. "The problem in estimating a disaggregated SAM for a recent year is to find an efficient (and cost-effective) way to incorporate and reconcile information from a variety of sources, including data from the prior years" (Robinson and al., 2001).

Mechanical balancing techniques, however, are not a substitute for care and judgment in setting up and organizing the best data base possible, with adequate verifications and, as the case may be, corrections. We return to this issue in our concluding remarks.

[1] This is a revised version of a working paper of the Centre Interuniversitaire sur le Risque les Politiques Economiques et L'Emploi (CIRPEE), Université Laval, August 2005. The authors are grateful to Jeff Round whose comments helped in improving the paper.

In the first part of this paper, we present a literature review on some commonly used approaches for balancing a SAM. The second part presents a program that implements one of these approaches, which we find well adapted to SAM balancing problems commonly faced by PEP researchers.

Part 1 – Matrix balancing methods

1. The RAS Method

RAS is a widely used procedure to balance input-output tables, which may also be applied to SAMs. It is used when new information on the matrix row and column sums becomes available and we want to update an existing matrix.

Define T as an $n \times n$ matrix of SAM transactions, where t_{ij} is a cell value and let

$$t_{.j} = \sum_i t_{ij} \qquad [001]$$

$$t_{i.} = \sum_j t_{ij} \qquad [002]$$

The problem is to generate a new $n \times n$ matrix T^1 from an existing matrix T^0 of the same dimension while respecting new given row and column totals. Let $\hat{t}_{i.}$ represent the new row totals, and $\hat{t}_{.j}$ the new column totals. Then the new matrix must satisfy

$$\sum_i t_{ij}^1 = \hat{t}_{.j} \qquad [003]$$

$$\sum_j t_{ij}^1 = \hat{t}_{i.} \qquad [004]$$

Note that given the identity $\sum_j \hat{t}_{.j} \equiv \sum_i \hat{t}_{i.}$, there are only $(2n - 1)$ independent constraints altogether in [003] and [004]. The RAS method is an iterative algorithm to find a biproportional adjustment that satisfies [003] and [004]. A biproportional adjustment is of the form

$$t_{ij}^1 = r_i t_{ij}^0 s_j \qquad [005]$$

In matrix notation:

$$\mathbf{T}^1 = \widetilde{\mathbf{R}}^T \mathbf{T}^0 \widetilde{\mathbf{S}}^T \qquad [006]$$

where the tilde (~) indicates diagonal matrices of elements r_i and s_j respectively. It is easily verified that the r_i and s_j are defined only up to a factor of proportionality κ, since

$$t_{ij}^{1} = \left(\kappa\, r_i\right) t_{ij}^{0} \left(\frac{s_j}{\kappa}\right) \qquad [007]$$

is equivalent to [005]. So the value of one of the multipliers is arbitrary, which leaves $(2n - 1)$ unknown multipliers, determined from the $(2n - 1)$ independent row and column restrictions in [003] and [004]. The RAS iterative adjustment procedure finds a solution to this problem, if a solution exists.

We now describe the RAS algorithm formally. In what follows, superscripts between parentheses (0),(1),(2)… refer to iteration steps and the caret (^) is used for new column or row values, as in [003] and [004].

- Step 1:

$$\lambda_i^{(1)} = \frac{\hat{t}_{i.}}{\sum_j t_{ij}^{0}} \Rightarrow t_{ij}^{(1)} = \lambda_i^{(1)} t_{ij}^{0} \Rightarrow \mu_j^{(1)} = \frac{\hat{t}_{.j}}{\sum_i t_{ij}^{(1)}} \Rightarrow t_{ij}^{(2)} = \mu_j^{(1)} t_{ij}^{(1)} \qquad [008]$$

- Step 2:

$$\lambda_i^{(2)} = \frac{\hat{t}_{i.}}{\sum_j t_{ij}^{(2)}} \Rightarrow t_{ij}^{(3)} = \lambda_i^{(2)} t_{ij}^{(2)} \Rightarrow \mu_j^{(2)} = \frac{\hat{t}_{.j}}{\sum_i t_{ij}^{(3)}} \Rightarrow t_{ij}^{(4)} = \mu_j^{(2)} t_{ij}^{(3)} \qquad [009]$$

…

- Step n:

$$\lambda_i^{(n)} = \frac{\hat{t}_{i.}}{\sum_j t_{ij}^{(2n-2)}} \Rightarrow t_{ij}^{(2n-1)} = \lambda_i^{(n)} t_{ij}^{(2n-2)} \Rightarrow \mu_j^{(n)} = \frac{\hat{t}_{.j}}{\sum_i t_{ij}^{(2n-1)}} \Rightarrow t_{ij}^{(2n)} = \mu_j^{(n)} t_{ij}^{(2n-1)} \qquad [010]$$

This process is continued until the iterations converge. We can summarize these steps by:

$$t_{ij}^{(2n-1)} = \left(\prod_{h=1}^{n-1} \mu_j^{(h)}\right)\left(\prod_{k=1}^{n} \lambda_i^{(k)}\right) t_{ij}^{0}\text{; for odd-rank values, } t_{ij}^{(1)}, t_{ij}^{(3)}, t_{ij}^{(5)} \ldots \qquad [011]$$

$$t_{ij}^{(2n)} = \left(\prod_{h=1}^{n} \mu_j^{(h)}\right)\left(\prod_{k=1}^{n} \lambda_i^{(k)}\right) t_{ij}^{0}\text{; for even-rank values, } t_{ij}^{(2)}, t_{ij}^{(4)}, t_{ij}^{(6)} \ldots \qquad [012]$$

With $L_i^{(n)} = \left(\prod_{k=1}^{n} \lambda_i^{(k)}\right)$ and $M_j^{(n)} = \left(\prod_{h=1}^{n} \mu_j^{(h)}\right)$, we rewrite [011] and [012]

$$t_{ij}^{(2n-1)} = L_i^{(n)}\, M_j^{(n-1)} t_{ij}^{0}\text{; for odd-rank values, } t_{ij}^{(1)}, t_{ij}^{(3)}, t_{ij}^{(5)} \ldots \qquad [013]$$

$$t_{ij}^{(2n)} = L_i^{(n)}\, M_j^{(n)} t_{ij}^{0}\text{; for even rank-values, } t_{ij}^{(2)}, t_{ij}^{(4)}, t_{ij}^{(6)} \ldots \qquad [014]$$

The iterative procedure converges if $\lim_{n\to\infty} \lambda_i^{(n)} = 1$ and $\lim_{n\to\infty} \mu_j^{(n)} = 1$. Then the result is given by

$$t_{ij}^{1} = L_i M_j t_{ij}^{0} \quad [015]$$

where

$$L_i = \lim_{n\to\infty} \left(\prod_{k=1}^{n} \lambda_i^{(k)} \right) \quad [016]$$

$$M_j = \lim_{n\to\infty} \left(\prod_{h=1}^{n} \mu_j^{(h)} \right) \quad [017]$$

Given [010] and [014], for even rank values, $t_{ij}^{(2)}, t_{ij}^{(4)}, t_{ij}^{(6)} \ldots$, we have

$$\frac{L_i^{(n)}}{L_i^{(n-1)}} = \frac{\left(\prod_{k=1}^{n} \lambda_i^{(k)} \right)}{\left(\prod_{k=1}^{n-1} \lambda_i^{(k)} \right)} = \lambda_i^{(n)} = \frac{\hat{t}_{i.}}{\sum_j t_{ij}^{(2n-2)}} = \frac{\hat{t}_{i.}}{\sum_j L_i^{(n-1)} M_j^{(n-1)} t_{ij}^{0}} \quad [018]$$

It follows that

$$\lim_{n\to\infty} \lambda_i^{(n)} = \frac{\hat{t}_{i.}}{\lim_{n\to\infty} \sum_j t_{ij}^{(2n-2)}} = \frac{\hat{t}_{i.}}{\lim_{n\to\infty} \sum_j L_i^{(n-1)} M_j^{(n-1)} t_{ij}^{0}} \quad [019]$$

$$\lim_{n\to\infty} \lambda_i^{(n)} = \frac{\hat{t}_{i.}}{\sum_j \left(\lim_{n\to\infty} L_i^{(n-1)} \right) \left(\lim_{n\to\infty} M_j^{(n-1)} \right) t_{ij}^{0}} \quad [020]$$

$$\lim_{n\to\infty} \lambda_i^{(n)} = \frac{\hat{t}_{i.}}{\sum_j L_i M_j t_{ij}^{0}} = \frac{\hat{t}_{i.}}{\sum_j t_{ij}^{1}} \quad [021]$$

Therefore, $\lim_{n\to\infty} \lambda_i^{(n)} = 1$ implies that constraints [004] are fulfilled. Similarly, given [010] and [013], for odd rank values, $t_{ij}^{(1)}, t_{ij}^{(3)}, t_{ij}^{(5)} \ldots$, we have

$$\frac{M_j^{(n)}}{M_j^{(n-1)}} = \frac{\left(\prod_{h=1}^{n} \mu_j^{(h)} \right)}{\left(\prod_{h=1}^{n-1} \mu_j^{(h)} \right)} = \mu_j^{(n)} = \frac{\hat{t}_{.j}}{\sum_i t_{ij}^{(2n-1)}} = \frac{\hat{t}_{.j}}{\sum_i L_i^{(n)} M_j^{(n-1)} t_{ij}^{0}} \quad [022]$$

It follows that

$$\lim_{n\to\infty} \mu_j^{(n)} = \frac{\hat{t}_{.j}}{\lim_{n\to\infty} \sum_i t_{ij}^{(2n-1)}} = \frac{\hat{t}_{.j}}{\lim_{n\to\infty} \sum_i L_i^{(n)}\, M_j^{(n-1)} t_{ij}^0} \qquad [023]$$

$$\lim_{n\to\infty} \mu_j^{(n)} = \frac{\hat{t}_{.j}}{\sum_i \left(\lim_{n\to\infty} L_i^{(n)} \right) \left(\lim_{n\to\infty} M_j^{(n-1)} \right) t_{ij}^0} \qquad [024]$$

$$\lim_{n\to\infty} \mu_j^{(n)} = \frac{\hat{t}_{.j}}{\sum_i L_i M_j t_{ij}^0} = \frac{\hat{t}_{.j}}{\sum_i t_{ij}^1} \qquad [025]$$

Therefore, $\lim_{n\to\infty} \mu_j^{(n)} = 1$ implies that constraints [003] are fulfilled.

Bacharach (1970) has shown that, if a solution exists, the RAS method yields a set of positive multipliers $r_i = L_i$ and $s_j = M_j$ that are unique up to a factor of proportionality, and that satisfy [003] and [004]. As Robinson *et al*. (2001) point out, for the method to work, the matrix must be "connected", a generalization of the notion of "indecomposable" (Bacharach, 1970, p. 47); for example, RAS fails when there is a column or row of zeros, because no proportional adjustment can possibly make their sum non-zero. Although the RAS procedure is presented here in its application to a (square) SAM, it is not a requirement for the matrix to be square. Also note that the RAS method can be applied to adjusting coefficients a_{ij} rather than transaction flows t_{ij} as in [008], [009] and [010]. Dietzenbacher and Miller (2009) show that the two procedures are equivalent. The RAS method has been extended to accommodate uncertainty in the row and column totals, as well as negative cell values (Günlük-Şenesen and Bates, 1988; Junius and Oosterhaven, 2003). When there is a solution, the RAS method has the advantage of being simple to apply. But, this simplicity has some disadvantages: (1) it lacks theoretical foundations, (2) it cannot – or at least cannot easily – accommodate some types of information other than row and column totals (for example we cannot impose constraints on subtotals other than row or column totals[2]). Because of these disadvantages, many researchers prefer use the more flexible cross-entropy method.

2. The Cross-Entropy Method

Entropy is a concept from mechanical physics that was re-interpreted in the information theory developed by Shannon (1948), and applied to the problem of estimation and statistical inference by Jaynes (1957). Theil (1967) brought the approach to economics, and Kapur and Kesavian (1992) and Golan, Judge and

---

2 RAS can easily accommodate fixing *individual* cell values simply by removing any fixed cell elements and adjusting the corresponding row and column totals accordingly before applying the procedure.

Miller (1996) developed econometric applications. Cross-entropy as defined by Kullback and Leibler (1951) is a measure of the one-way divergence[3] of a posterior probability distribution (the adjusted matrix), from a prior distribution (the unadjusted matrix). The minimum cross-entropy adjustment process consists in adjusting a matrix to its marginal totals (and, as the case may be, to other constraints) in such a way as to minimize that divergence. This technique is readily interpreted in information theory as minimizing the quantity of extraneous information imposed upon the *a priori* matrix to fit the marginal totals; for that reason, the minimum cross-entropy criterion is also known as the minimum information-gain principle[4].

It is known at least since Bacharach (1970) that the RAS method applied to a non-negative matrix minimizes cross-entropy. It was natural to apply cross-entropy to SAM balancing. A cross-entropy SAM-balancing method was developed by Sherman Robinson and colleagues at IFPRI (Robinson, Cattaneo and El-Said, 1998; Robilliard and Robinson, 1999; Robinson, Cattaneo and El-Said, 2001; Robinson and El-Said, 2000).[5] Several related approaches have been proposed in the literature. Lemelin (2009) shows that applying the minimum cross-entropy criterion can lead to different results, depending on the particular form of the objective function and the set of constraints.

2.1 RAS-EQUIVALENT PROBLEM

First consider the cross-entropy equivalent of the RAS bi-proportional adjustment procedure. As we shall see in section 3 below, and as shown by Bacharach (1970), Macgill (1977) and Günlük-Şenesen and Bates (1988), the RAS-adjusted matrix is the solution to the following problem:

$$\min_{\{t_{ij}^1\}} H' = \sum_i \sum_j t_{ij}^1 \ln \frac{t_{ij}^1}{t_{ij}^0} \qquad [026]$$

subject to

$$\sum_i t_{ij}^1 = \hat{t}_{.j} \qquad [003]$$

---

3 It is not a distance measure, because it is not symmetrical.

4 More elaborate presentations of the minimum information-gain principle and the cross-entropy adjustment method are found in Lemelin (2009 and 2011), where the reference list contains material for further reading.

5 "The cross-entropy method uses all available information, including prior parameter estimates, and supports estimation even in a 'data sparse' environment. The use of the cross-entropy measure in the estimation criterion has been justified on the basis of axiomatic arguments concerning its desirability both as a measure of 'information' and as a criterion for inference. There are close links between the minimum cross-entropy criterion and the maximum likelihood estimators, but the cross-entropy criterion requires fewer statistical assumptions in that its application does not require specification of an explicit likelihood function. In our case, this sparseness in assumptions is desirable since we have no knowledge about the form of any underlying probability distributions" (Robilliard and Robinson 2001).

$$\sum_j t_{ij}^1 = \hat{t}_{i.} \qquad [004]$$

where $\hat{t}_{.j}$ and $\hat{t}_{i.}$ are known column and row sums which satisfy $\sum_j \hat{t}_{.j} = \sum_i \hat{t}_{i.}$. Now let

$$p_{ij}^k = t_{ij}^k \Big/ t_{..}^k \text{, with } t_{..}^k = \sum_i \sum_j t_{ij}^k \qquad [027]$$

$$p_{i.}^k = t_{i.}^k \Big/ t_{..}^k \text{, with } t_{i.}^k = \sum_j t_{ij}^k \qquad [028]$$

$$p_{.j}^k = t_{.j}^k \Big/ t_{..}^k \text{, with } t_{.j}^k = \sum_i t_{ij}^k \qquad [029]$$

where $k = 0$ for the unadjusted prior SAM values, and $k = 1$ for the adjusted SAM values. Then the problem defined by [026], [003] and [004] is equivalent to

$$\min_{\{p_{ij}^1\}} H = \sum_i \sum_j p_{ij}^1 \ln \frac{p_{ij}^1}{p_{ij}^0} \qquad [030]$$

subject to

$$\sum_i p_{ij}^1 = \hat{p}_{.j} \qquad [031]$$

$$\sum_j p_{ij}^1 = \hat{p}_{i.} \qquad [032]$$

where $\hat{p}_{i.} = \hat{t}_{i.} \Big/ \sum_h \hat{t}_{h.}$ and $\hat{p}_{.j} = \hat{t}_{.j} \Big/ \sum_h \hat{t}_{.h}$. Indeed, substitute [027] into [030], and find

$$\min_{\{t_{ij}^1\}} H = \sum_i \sum_j \left(t_{ij}^1 \Big/ t_{..}^1\right) \ln \frac{\left(t_{ij}^1 \Big/ t_{..}^1\right)}{\left(t_{ij}^0 \Big/ t_{..}^0\right)} \qquad [033]$$

$$\min_{\{t_{ij}^1\}} H = \left(\frac{1}{t_{..}^1}\right) \sum_i \sum_j t_{ij}^1 \left[\ln\left(t_{ij}^1 \Big/ t_{ij}^0\right) - \ln\left(t_{..}^1 \Big/ t_{..}^0\right)\right] \qquad [034]$$

$$\min_{\{t_{ij}^1\}} H = \left(\frac{1}{t_{..}^1}\right) \sum_i \sum_j t_{ij}^1 \ln\left(t_{ij}^1 \Big/ t_{ij}^0\right) - \left(\frac{1}{t_{..}^1}\right) \sum_i \sum_j t_{ij}^1 \ln\left(t_{..}^1 \Big/ t_{..}^0\right) \qquad [035]$$

$$\min_{\{t_{ij}^1\}} H = \left(\frac{1}{t_{..}^1}\right) \sum_i \sum_j t_{ij}^1 \ln\left(t_{ij}^1 \Big/ t_{ij}^0\right) - \ln\left(t_{..}^1 \Big/ t_{..}^0\right) \qquad [036]$$

Since the second term in [036] is a known constant, minimizing *H'* as defined in [026] is equivalent to minimizing *H* as defined in [030], given that constraints [003] and [004] are easily verified to be identical to [031] and [032].

The minimand *H* in [030] is the proper Kullback-Leibler (1951) cross-entropy measure of the one-way[6] divergence between the new estimated two-dimensional probability distribution $\{p_{ij}^1\}$ and the prior probability distribution $\{p_{ij}^0\}$. It can be interpreted in information theory as the amount of "extraneous" information brought into $\{p_{ij}^1\}$ compared to $\{p_{ij}^0\}$. Applied to the problem of balancing a SAM, the problem is to find a new SAM T^1, close to an existing SAM T^0, by minimizing the cross-entropy distance between them, while respecting all constraints, as stated in [026], [003] and [004].

2.2 IFPRI APPROACH

Returning to the IFPRI approach, Robinson *et al*. (2001) extend the cross-entropy approach to include a stochastic treatment of errors in the control totals[7]. Since the SAM-balancing program presented here is based on the assumption that row and column totals are known with certainty, it has no error components. So we shall examine the basic, deterministic form of the IFPRI adjustment procedure, It is clearly specified in Robinson *et al*. (2001, equation 10, p. 51; also, p. 56) that the problem they solve in the deterministic case is

$$\min_{\{a_{ij}^1\}} H'' = \sum_i \sum_j a_{ij}^1 \ln \frac{a_{ij}^1}{a_{ij}^0} \qquad [037]$$

subject to

$$\sum_i a_{ij}^1 = 1 \qquad [038]$$

$$\sum_j a_{ij}^1 \hat{t}_{.j} = \hat{t}_{i.} \qquad [039]$$

where

$$a_{ij}^k = t_{ij}^k \Big/ t_{.j}^k \,,\ k = 1, 2 \qquad [040]$$

---

6 Kullback-Leibler cross-entropy is not symmetrical. In other words, the prior and the posterior distributions are not interchangeable.

7 See also Robilliard and Robinson (1999), Robinson and al. (1998 and 2000).

are the SAM equivalents of input-output coefficients. Equation [038] is equivalent to [003] and [039] to [004]. Note that, in view of [038], each column of the coefficient matrix may be formally viewed as a probability distribution, so that the minimand in [037] is a sum of several Kullback-Leibler cross-entropy measures, one per column. Now, given known row and column sums, adjusting the SAM coefficients as in the IFPRI method, rather than the transactions matrix, is tantamount to assigning weights to the terms in the right-hand side of [026] above; these weights are inversely proportional to the new column sums[8]. This assertion is easily demonstrated (McDougall, 1999). Given [040] and [027], we have

$$a_{ij}^{k} = p_{ij}^{k}\left(t_{..}^{k} \Big/ t_{.j}^{k}\right) \qquad [041]$$

Substitute into [037] and find

$$H'' = \sum_i \sum_j p_{ij}^{1}\left(t_{..}^{1} \Big/ t_{.j}^{1}\right) \ln \frac{p_{ij}^{1}\left(t_{..}^{1} \Big/ t_{.j}^{1}\right)}{p_{ij}^{0}\left(t_{..}^{0} \Big/ t_{.j}^{0}\right)} \qquad [042]$$

$$H'' = \sum_i \sum_j p_{ij}^{1}\left(t_{..}^{1} \Big/ t_{.j}^{1}\right) \ln \frac{p_{ij}^{1}}{p_{ij}^{0}} + \sum_i \sum_j p_{ij}^{1}\left(t_{..}^{1} \Big/ t_{.j}^{1}\right) \ln \frac{\left(t_{..}^{1} \Big/ t_{.j}^{1}\right)}{\left(t_{..}^{0} \Big/ t_{.j}^{0}\right)} \qquad [043]$$

$$H'' = \sum_i \sum_j p_{ij}^{1}\left(t_{..}^{1} \Big/ t_{.j}^{1}\right) \ln \frac{p_{ij}^{1}}{p_{ij}^{0}} + \sum_j \left(\sum_i p_{ij}^{1} \right)\left(t_{..}^{1} \Big/ t_{.j}^{1}\right) \ln \frac{\left(t_{..}^{1} \Big/ t_{.j}^{1}\right)}{\left(t_{..}^{0} \Big/ t_{.j}^{0}\right)} \qquad [044]$$

$$H'' = \sum_i \sum_j p_{ij}^{1}\left(t_{..}^{1} \Big/ t_{.j}^{1}\right) \ln \frac{p_{ij}^{1}}{p_{ij}^{0}} + \sum_j p_{.j}^{1}\left(t_{..}^{1} \Big/ t_{.j}^{1}\right) \ln \frac{\left(t_{..}^{1} \Big/ t_{.j}^{1}\right)}{\left(t_{..}^{0} \Big/ t_{.j}^{0}\right)} \qquad [045]$$

$$H'' = \sum_i \sum_j p_{ij}^{1}\left(t_{..}^{1} \Big/ t_{.j}^{1}\right) \ln \frac{p_{ij}^{1}}{p_{ij}^{0}} + \sum_j \left(t_{.j}^{1} \Big/ t^{1}\right)\left(t_{..}^{1} \Big/ t_{.j}^{1}\right) \ln \frac{\left(t_{..}^{1} \Big/ t_{.j}^{1}\right)}{\left(t_{..}^{0} \Big/ t_{.j}^{0}\right)} \qquad [046]$$

$$H'' = \sum_i \sum_j p_{ij}^{1}\left(t_{..}^{1} \Big/ t_{.j}^{1}\right) \ln \frac{p_{ij}^{1}}{p_{ij}^{0}} + \sum_j \ln \frac{\left(t_{..}^{1} \Big/ t_{.j}^{1}\right)}{\left(t_{..}^{0} \Big/ t_{.j}^{0}\right)} \qquad [047]$$

Under the constraint that $t_{.j}^{1} = \hat{t}_{.j}$, the second term is a constant, and minimizing H'' is equivalent to minimizing

$$H''' = \sum_i \sum_j p_{ij}^{1}\left(\hat{t}_{..} \Big/ \hat{t}_{.j}\right) \ln \frac{p_{ij}^{1}}{p_{ij}^{0}} \qquad [048]$$

---

8 McDougall (1999, p. 5-7) makes the same point.

This demonstrates that the Robinson-IFPRI minimand is a weighted sum of column-wise entropies, where the weights are inversely proportional to column sums. Consequently, the basic IFPRI adjustment procedure will lead to a different solution from the RAS-equivalent procedure. This may seem intriguing, in view of the demonstration by Dietzenbacher and Miller (2009, hereafter DM) that applying the RAS procedure to the transactions matrix or to the coefficient matrix leads to the same solution. Note, however, that only two of the three components of the problem defined by [037], [038] and [039] are strictly equivalent to the corresponding components of the RAS-equivalent problem: while [038] is equivalent to [003] and [039] to [004], we have just shown that [037] is *not* equivalent to [026]. In contrast, DM compare the transactions matrix adjustment problem and the coefficients matrix adjustment problem, each one formally stated as a group of three sets of constraints: the first is the requirement that the adjustment be biproportional; the two other sets of constraints are the marginal total constraints ([003] and [004] for the transactions matrix; and [038] and [039] for the coefficients matrix). What DM demonstrate is that each set of constraints in either problem is strictly equivalent to the corresponding set of constraints of the other problem. The two problems, therefore, are the same, and in view of Bacharach's demonstration that the solution is unique, then applying the RAS iterative algorithm to the transactions matrix or to the coefficients matrix must yield the same solution.

2.3 THE PEP SAMBAL SAM BALANCING PROGRAM

In contrast to the RAS-equivalent approach and the IFPRI method, the SAM balancing program proposed here does not assume that row and column sums are known. Specifically, the minimand is defined as in [030], but constraints [031] and [032] are absent. The problem which the SAMBAL program solves is

$$\min_{\{p_{ij}^1\}} H = \sum_i \sum_j p_{ij}^1 \ln \frac{p_{ij}^1}{p_{ij}^0} \qquad [030]$$

subject to

$$\sum_i \sum_j p_{ij}^1 = 1 \qquad [049]$$

$$p_{i.}^1 = p_{.j}^1, \text{ for } i = j \qquad [050]$$

Constraint [049] is an implication of definition [027]: the sum of probabilities in a distribution must equal 1. Constraint [050] is specific to the double-entry accounting structure of a SAM: the total receipts of every account (SAM row) must be equal to its total payments (corresponding SAM column). However, solving the problem defined by [030], [049] and [050] will only characterize the matrix structure (as

represented by the p_{ij}^1), not its magnitude. The magnitude of the SAM flows is fixed thanks to an additional constraint on the grand total of SAM flows, applied ex post:

$$t_{ij}^1 = p_{ij}^1 t_{..}^0 \qquad [051]$$

The SAMBAL approach is particularly appropriate, as its name suggests, to balance a SAM in the final stage of its construction. It does not require knowledge of the marginal totals. Constraints on the marginal totals, if one or more of them are known, are added optionally. If all marginal totals are known and added as constraints, then the SAMBAL program yields the RAS-equivalent solution.

The standard entropy method does not allow negative values. To solve this problem, SAMBAL simply transposes these values to their counterpart cells before balancing the SAM. Indeed, as the SAM represents flows from one account to another, a negative flow from account A to account B is equivalent, in terms of accounting, to a positive flow of equal magnitude from account B to account A. The reverse transposition is performed after the SAM has been balanced, to restore negative values to their original positions. It should be noted however, that the reverse transposition eliminates one of any pair of cross-flows of opposite signs: if the original matrix has both a negative flow from A to B and a positive flow from B to A, one of the two will be set to zero in the balanced matrix.

3. Comparing Cross-Entropy and RAS Methods

We have already stated the equivalence between the RAS method and the cross-entropy approach defined in [026], [003] and [004], which uses an entropy-based minimand and a constraint set appropriate to a "type 1" balancing problem[9]. Here we demonstrate that equivalence. We have shown in section 2.1 that the problem defined by [026], [003] and [004] is equivalent to the one defined by [030], [031] and [032]. The solution to the latter problem can be found from the first-order conditions derived from the Lagrangian

$$\Lambda = \sum_i \sum_j p_{ij}^1 \ln \frac{p_{ij}^1}{p_{ij}^0} + \sum_i \alpha_i \left(\sum_j p_{ij}^1 - \hat{p}_{i.} \right) + \sum_j \beta_j \left(\sum_i p_{ij}^1 - \hat{p}_{.j} \right) \qquad [052]$$

---

9 Following Schneider and Zenios (1990), Jeffery I Round, (2003, p. 175) distinguishes:
Problem 1: If $\mathbf{X} = \left[x_{ij} \right]$ is an $m \times n$ non-negative matrix and **u** and **v** are positive vectors of orders m and n respectively then determine an $m \times n$ matrix **X*** 'close to' **X** such that $\sum_j x_{ij}{}^* = u_i$ and $\sum_i x_{ij}{}^* = v_j$; and $x_{ij}{}^* > 0$ if and only if $x_{ij} > 0$ $(\forall\, i, j)$.
Problem 2: If $\mathbf{X} = \left[x_{ij} \right]$ is an $(n \times n)$ non-negative matrix and **u** and **v** are positive vectors [of order n] then determine an $(n \times n)$ matrix **X*** 'close to' **X** such that $\sum_j x_{ij}{}^* = \sum_j x_{ji}{}^*$ $(\forall\, i)$ and $x_{ij}{}^* > 0$ if and only if $x_{ij} > 0$ $(\forall\, i, j)$.

$$\Lambda = \sum_i \sum_j p_{ij}^1 \ln p_{ij}^1 - \sum_i \sum_j p_{ij}^1 \ln p_{ij}^0 + \sum_i \alpha_i \left(\sum_j p_{ij}^1 - \hat{p}_{i.} \right) + \sum_j \beta_j \left(\sum_i p_{ij}^1 - \hat{p}_{.j} \right) \quad [053]$$

The first-order conditions are:

$$\frac{\partial \Lambda}{\partial \beta_j} = \sum_i p_{ij}^1 - \hat{p}_{.j} = 0 \quad [054]$$

$$\frac{\partial \Lambda}{\partial \alpha_i} = \sum_j p_{ij}^1 - \hat{p}_{i.} = 0 \quad [055]$$

$$\frac{\partial \Lambda}{\partial p_{ij}} = 1 + \ln p_{ij}^1 - \ln p_{ij}^0 + \alpha_i + \beta_j = 0 \quad [056]$$

Condition [056] can be written as

$$p_{ij}^1 = p_{ij}^0 \, e^{-\left(1+\alpha_i+\beta_j\right)} \quad [057]$$

where α_i and β_j are Lagrangian multipliers associated with the information on row and column sums. The outcome is determined so as to satisfy constraints [054] and [055], which are exactly the same as [031] and [032]. The strict convexity of the objective function ensures the uniqueness of the solution when it exists, so that the second-order conditions do not have to be checked.

Using [027], we have

$$\left(t_{ij}^1 / t_{..}^1\right) = \left(t_{ij}^0 / t_{..}^0\right) e^{-\left(1+\alpha_i+\beta_j\right)} \quad [058]$$

$$t_{ij}^1 = t_{ij}^0 \left(t_{..}^1 / t_{..}^0\right) e^{-\left(1+\alpha_i+\beta_j\right)} \quad [059]$$

$$t_{ij}^1 = t_{ij}^0 \, e^{\ln\left(t_{..}^1 / t_{..}^0\right)} e^{-\left(1+\alpha_i+\beta_j\right)} \quad [060]$$

$$t_{ij}^1 = t_{ij}^0 \, e^{-\left[1-\ln\left(t_{..}^1 / t_{..}^0\right)+\alpha_i+\beta_j\right]} \quad [061]$$

Now, recalling definitions [016] and [017], let

$$\alpha_i' = -\left[1 - \ln\left(t_{..}^1 / t_{..}^0\right) + \ln L_i\right] \quad [062]$$

$$\beta_j' = -\ln M_j \quad [063]$$

so that

$$L_i = e^{-\left[1-\ln\left(t_{..}^1 / t_{..}^0\right)+\alpha_i'\right]} \quad [064]$$

$$M_j = e^{-\beta'_j} \qquad [065]$$

Substitute [064] and [065] into [015] and there follows

$$t^1_{ij} = t^0_{ij}\ e^{-\left[1-\ln\left(t^1_{..}/t^0_{..}\right)+\alpha'_i+\beta'_j\right]} \qquad [066]$$

So we see that the outcome from the RAS method respects the first-order conditions of the cross-entropy problem: equation [022] is equivalent to [061] and to [057], with $\alpha_i = \alpha'_i$ and $\beta_i = \beta'_i$; and from [021] and [025], we know that constraints [053[054] and [055] (equivalent to [031] and [032]) are verified. Since the solution to [030], [031] and [032] is unique, then if the RAS procedure converges, it converges with a unique outcome that is also the solution of the cross-entropy minimization problem.

Macgill (1977) defined the necessary and sufficient conditions for the RAS procedure to converge. They are:

$$\sum_i \hat{t}_{i.} = \sum_j \hat{t}_{.j} \qquad [067]$$

For all $t^0_{gh} = 0$ in the *a priori* matrix, $\hat{t}_{g.} \le \sum_{j\neq h} \hat{t}_{.j}$ [068]

For all $t^0_{gh} = 0$ in the *a priori* matrix, $\hat{t}_{.h} \le \sum_{i\neq g} \hat{t}_{i.}\ \overline{x_{\bullet h}} \le \sum_{i\neq g} \overline{x_{i\bullet}}$ [069]

If the cross-entropy method is equivalent to the RAS technique, then what is its advantage over the latter? First, the cross-entropy method is founded in information theory, while RAS lacks theortical foundations. Second, the cross-entropy method offers greater flexibility by allowing the user to make use of any information additional to row and column totals by imposing supplementary linear constraints (in particular, the user may fix the sum of groups of cell values. Finally, while the computational facility of the iterative RAS procedure used to be an advantage when computational power was limited, nowadays, posing the problem as an optimization problem opens the door to considering other objective functions.

4. Ordinary Least Squares and other Methods

Günlük-Şenesen and Bates (1988) and Round (2003), among others, discuss other SAM balancing techniques, such as the Stone-Byron method. The methods differ mainly according to the choice of a minimand, the manner in which uncertainty is handled, and the solution technique (iterative procedure or mathematical optimization using a solver program). In this simple introduction, we do not explore the more complex approaches involving error terms or estimated variances (sometimes subjectively estimated). Among alternative minimands, we consider only one, Ordinary Least Squares (OLS), which is

offered in the Sambal SAM balancing program. The basic principle is the same as in the cross-entropy approach: to minimize the discrepancy between an *a priori* matrix and an *a posteriori* matrix which obeys all known constraints. OLS, or more generally a quadratic form, may be considered as an alternative minimand to cross-entropy. In SAMBAL, the OLS minimand is defined as

$$\min_{\{p^1\}} \sum_i \sum_j \left(\frac{p_{ij}^1 - p_{ij}^0}{p_{ij}^0} \right)^2 \qquad [070]$$

which, given [027] and [051] is equivalent to

$$\min_{\{p^1\}} \sum_i \sum_j \left(\frac{t_{ij}^1 - t_{ij}^0}{t_{ij}^0} \right)^2 \qquad [071]$$

It is the sum of squared deviations in relative terms. It is combined with the same constraints as the cross-entropy minimand:

$$\sum_i \sum_j p_{ij}^1 = 1 \qquad [049]$$

$$p_{i.}^1 = p_{.j}^1 \qquad [050]$$

Although the quadratic loss function is widespread, its theoretical foundations in the SAM balancing context are weak, compared to the cross-entropy loss function, which is grounded in information theory.

5. Is There a Preferred Method for Balancing a SAM?

On this issue, we quote Round (2003, p. 179):

> "Robinson, et al (2001) carry out a range of Monte Carlo experiments which suggest the superiority of the CE method over RAS in those circumstances (under problem 1 conditions) where comparisons are valid. Günlük-Şenesen and Bates (1988) also conduct experiments with several balancing methods under similar problem 1 conditions and observe more mixed outcomes. One problem in carrying out experiments is that the criteria for assessing success (the measures of closeness of an adjusted matrix to a 'true' matrix) are intimately related to the choice of minimand. Therefore there is an inherent bias built into any experimentation, which makes objectivity difficult.
>
> The relatively close analytical relationships between the most frequently used alternative methods for balancing SAMs suggest that if the required adjustments are relatively small then the differences between the methods are likely also to be small."

But over and above the issue of choosing a balancing method, Round (2003), in his closing remarks, insists: "It would surely be preferable to devote most energy to a careful assembly of the initial estimtes and to rely on mechanical methods only as a last resort. A method of smoothing weak initial estimates is unlikely to generate reliable final estimates, however efficient that method is." We wholeheartedly concur

with Round (2003), that all practitioners should maintain a strong element of the judgment approach and not see any SAM-balancing program as a magic solution. In taking stock of the initial data inconsistencies, care should be taken to ensure that all possible sources of error in the initial data have been explored and eliminated. One needs to think long and hard about the possible causes of these inconsistencies and the appropriate adjustments. For example, as Round (2003) points out, ousehold surveys tend to understate income. Before applying a program that will modify all the cell values of a SAM to ensure consistency, one may consider adjusting the household income estimate obtained from the household survey.

To summarize, a balancing program should be used ideally as a polishing tool for a SAM that is already close to be balanced. That is one reason why the SAMBAL program presented below allows the user to incorporate judgment concerning the greater reliability of specific cell values – albeit somewhat rigidly – by fixing values or sums of groups of values in the program.

Part 2 – A GAMS Program for Balancing a SAM

We now present a GAMS[10] code to balance an unbalanced SAM in eleven steps. Only steps 1, 2 and 10 require modifications by the user, although modifications are also possible at step 8. To illustrate, we introduce new data into a previously balanced social account matrix. We also show how to add new accounts into a balanced matrix.

STEP 1: DEFINE MATRIX ACCOUNTS

The user must first declare the SAM accounts. The GAMS command "SET" is used. The set of accounts set is named I and contains all accounts including the account "TOT" for total. Instead of actual account names, it is possible to assign numbers to the accounts when the dimensions are large, e.g. A1, A2, A3… A4000. If the set of accounts forms a sequence as above, it may be defined by the first and last element, separated by an asterisk, as follows

```
SETS I MATRIX ACCOUNTS   / A1 * A4000 /;
```

In that case, the "TOT" account will be the last: e.g. A4000. We then declare set INT as a subset of I including all SAM accounts except the total. In this section, you can add as many other sets or subsets as you need (e.g. for later definition of constraints).

```
$TITLE              SAMBAL
$STITLE        SAM BALANCING PROGRAM
```

[10] The General Algebraic Modeling System (GAMS) is a high-level modeling system for mathematical programming and optimization. For more information: www.gams.com.

```
*1. Input names of matrix accounts

I MATRIX ACCOUNTS    /LDO     Labor
                      KDO     Capital
                      LANDO   Land
                      RP      Rural poor households
                      UP      Urban poor households
                      RR      Rural rich households
                      UR      Urban rich households
                      FIRM    Firms
                      GOV     Government
                      ROW     Rest of the world
                      AGR     Agriculture
                      IND     Industry
                      SER     Tradable service sector
                      NTSER   Non tradable services sector
                      DAGR    Domestic demand for agricultural products
                      DIND    Domestic demand for industrial products
                      DSER    Domestic demand for tradable services
                      DNTSER  Domestic demand for non-tradable services
                      EAGR    Exports of agricultural products
                      EIND    Exports of industrial products
                      ESER    Exports of services
                      ACC     Accumulation account
                      TOT     Total /

*** Alternative code
*** I MATRIX ACCOUNTS  /     A1*A16        /

INT(I) All accounts except total
ALIAS(I,J);
ALIAS(INT,JNT);
INT(I) = YES;
INT("TOT") = NO;
*** Alternative code
*** INT("A16") = NO;
```

STEP 2: INPUT DATA

In this section, the user must input the data from the unbalanced matrix "SAM0". This can be done in many ways. First, the data can be entered manually as a GAMS "TABLE". Alternatively, the TABLE statement can be saved as a space-delimited ASCII file ("Save as" *.prn format in Excel) and then directly imported into the GAMS code with the "$include" command. Finally, data can be imported directly from an Excel spreadsheet using the GAMS GDX facility[11] to first transform the Excel matrix into a GDX file, and then to read the data from the GDX file. In this case, matrix position, name and range should be given and must ***exclude*** the row and column totals.

[11] http://www.gams.com/dd/docs/tools/gdxutils.pdf

***2. Input initial matrix**

```
PARAMETER SAM0(I,J);
*-------------------------------------------------------------------------
* Data input method #1: Table
*-------------------------------------------------------------------------
$ontext
TABLE SAM0(I,J)

                 LDO       KDO      LANDO        RP        UP        RR
RP             10575       324       1099
UP             11351       998
RR              2000       503       3496
UR              2100      6003
FIRM                      4496        471
GOV                        180                   75       496       153
ROW                         39        880
DAGR                                             6003      4246      2347
DIND                                             3419      4421      1803
DSER                                             2500      3335      1501
ACC                                                                     205
TOT            26026     12543       5946     11997     12498      6009


+                UR      FIRM       GOV       ROW       AGR       IND
LDO                                                    10175      2300
KDO                                                     1868      6976
LANDO                                                   5959
UP                                   146
UR                        1898
GOV               502     1292                         1075      3476
ROW                        374
DAGR             1653                                  2548      1302
DIND             4851                                  2000      3604
DSER             2175                                  1992      2858
DNTSER                               8703
EAGR                                             6891
EIND                                             1161
ESER                                             2203
ACC               821     1411       1714      5422
TOT             10002     4975      10563     15677     25617     20516


+                SER     NTSER       DAGR       DIND       DSER    DNTSER
LDO             9597      3953
KDO             3702
GOV              706                  123       2576
ROW                                  2381      10305      1705
AGR                                 18634
IND                                             19350
SER                                                        18508
NTSER                                                                8700
DAGR            1203        96
DIND            1950      2351
DSER            3551      2295
TOT            20709      8695      21138      32231      20213      8700
```

```
+               EAGR      EIND      ESER       ACC       TOT
LDO                                                    26025
KDO                                                    12546
LANDO                                                   5959
RP                                                     11998
UP                                                     12495
RR                                                      5999
UR                                                     10001
FIRM                                                    4967
GOV              -89                                   10565
ROW                                                    15684
AGR             6984                                   25618
IND                       1158                         20508
SER                                 2198               20706
NTSER                                                   8700
DAGR                                          1731     21129
DIND                                          7831     32230
DSER                                                   20207
DNTSER                                                  8703
EAGR                                                    6891
EIND                                                    1161
ESER                                                    2203
ACC                                                     9573
TOT             6895      1158      2198      9562
;
$offtext

*---------------------------------------------------------------------------
* Data input method #2: *.prn file
*---------------------------------------------------------------------------
* The unbalanced SAM in SAM_unbal1.xls contains NO cross-flows of opposite
* signs.
*$include SAM_unbal1.prn

*---------------------------------------------------------------------------
* Data input method #3: GDX facility
*---------------------------------------------------------------------------
* Note that the specified range, A1:W23 does NOT contain the TOTAL row and
* column.
* The unbalanced SAM in SAM_unbal1.xls contains NO cross-flows of opposite
* signs.
$CALL GDXXRW SAM_unbal1.xls Par=SAM0 rng=A1:W23
$GDXIN SAM_unbal1.gdx
$LOAD SAM0
```

STEP 3: TREATMENT OF NEGATIVE VALUES

The standard entropy method does not allow negative values. To solve this problem, we simply transpose these values to their counterpart cell before balancing the SAM. Indeed, as the SAM represents flows from one account to another, a negative flow from account A to account B is equivalent to an equal positive flow from account B to account A. For example, if SAM0(6,5) is negative, an equal amount is

added to cell SAM0(5,6) and the cell SAM0(6,5) is set equal to zero. To recall where these negative values belong, the program first defines a new matrix, NEG. The positions of all negative values are indicated in the new matrix by a "1". This step does not require any modifications by the user.

```
*3. Position of negative values in unbalanced matrix are noted
PARAMETER
NEG(I,J) Matrix of negative values;
NEG(INT,JNT)$[(SAM0(INT,JNT) LT 0)
               AND (SAM0(JNT,INT)-SAM0(INT,JNT) GT 0)
               AND (SAM0(JNT,INT)+SAM0(INT,JNT) LT 0)]=1;

*Negative matrix values are transposed
SAM0(INT,JNT)$[(SAM0(JNT,INT) LT 0)
         AND (SAM0(INT,JNT)-SAM0(JNT,INT) GT 0)]=SAM0(INT,JNT)-SAM0(JNT,INT);
SAM0(JNT,INT)$[(SAM0(JNT,INT) LT 0)
         AND (SAM0(INT,JNT)-SAM0(JNT,INT) GT 0)]=0;
```

STEP 4: NORMALIZE INITIAL MATRIX CELL VALUES

The program defines a total of cell values for the unbalanced matrix, "TOTO", and calculates its value. All matrix transaction flows are then normalized by dividing by "TOTO". This step does not require any modifications by the user.

```
*4. Initial matrix values transformed in proportion
PARAMETER
TOTO  total of initial matrix ;
TOTO  = SUM((INT,JNT), SAM0(INT,JNT));
SAM0(INT,JNT) = SAM0(INT,JNT)/TOTO;
```

STEP 5: TREATMENT OF ZERO VALUES

To avoid having to take the log of zero in the entropy method, the program adds a small amount (delta) to each cell value. This parameter must be as small as possible to avoid influencing the results. By default, it is equal to .0000000000001. This step does not require any modifications by the user.

```
*5. Non-zero log parameter
SCALARS
delta      Non-zero log parameter;
delta      =.0000000000001;
```

STEP 6: DEFINITION OF VARIABLES USED IN OPTIMIZATION PROCESS

A new matrix "NSAM" and variable "OPT" are declared. OPT is the objective function variable which has to be minimized. The number of variables is I × I matrix cells, plus the optimum variable. This step does not require any modifications by the user.

```
*6. Definition of variables
VARIABLES
SAM(I,J)      New SAM with transposed negative values
OPT           Distance variable;
```

STEP 7: DECLARATION AND STATEMENT OF MODEL EQUATIONS

Three sets of equations are used to balance the matrix: an optimization equation "OPTIMIZE", defined differently for the OLS and CE approaches, and two sets of constraint equations, the first equalizing each row total to its corresponding column total, the second setting the sum of all proportions to one.

```
*7. Program equations
EQUATIONS
OPTIMIZE        Optimization criterion
CONSTRAINT(I)   Equality between matrix and row sums
CONSTRAINT1     Sum of proportions equals one
;


*Least squares (in percentage form) optimization criterion
* OPTIMIZE..  OPT =E= SUM((INT,JNT)$SAM0(INT,JNT),
*                                    (NSAM(INT,JNT)/SAM0(INT,JNT)-1)**2);

*Entropy optimization criterion
OPTIMIZE..     OPT =E= SUM((INT,JNT)$(SAM0(INT,JNT) NE 0),(SAM(INT,JNT))
                 *(LOG(SAM(INT,JNT)+delta)-LOG(SAM0(INT,JNT)+delta)));

*Equality between row and column sums
CONSTRAINT(INT).. SUM(JNT,SAM(INT,JNT))=E=SUM(JNT,SAM(JNT,INT));

*Proportions sum equal one
CONSTRAINT1..     SUM((INT,JNT),SAM(INT,JNT))=E=1 ;
```

STEP 8: INITIALIZATION OF VARIABLES

All variables are initialized in this section. Also, matrix cell values are limited between 0 and infinity and empty cells remain empty. No user modifications are necessary, but the user can fix any cell values (or combination of cell values), as desired. Be careful not to fix too many values, as the program may not find a solution (infeasible problem).

```
*8. Cell values between 0 and infinity and empty cells remain empty
SAM.L(I,J)                          = SAM0(I,J);
SAM.LO(INT,JNT)                     = 0 ;
SAM.UP(INT,JNT)                     = +INF ;
SAM.FX(INT,JNT)$(NOT SAM0(INT,JNT))  = 0 ;
OPT.L                               = 0;

* Fix any variables as desired. However, be careful not to fix too many
* variables, otherwise the program will not find an optimal solution.
*SAM.FX("RP",INT)=SAM0("RP",INT);
*SAM.FX("UP",INT)=SAM0("UP",INT);
```

STEP 9: MODEL SOLVING

All equations are used in the model solving statement. The solver is chosen in the OPTION statement. The SOLVE command minimizes the variable OPT using a non-linear solver algorithm. This step does not require any modifications by the user.

```
*9. Model
MODEL SAMBAL / ALL /;

* Model attributes (if desired)
* the "workspace" attribute tells the solver how much workspace in Megabytes
* to allocate for problem solution.
*SAMBAL.workspace = 10;
* The "optfile" attribute tells the solver to use a solver options file.
*SAMBAL.optfile = 1;

* OPTION statements

* Choose solver
* OPTION NLP             = MINOS5;
* OPTION NLP             = CONOPT;
* OPTION NLP             = CONOPT2;
 OPTION NLP            = CONOPT3;

* The "iterlim" option sets a limit on the number of solver iterations
OPTION iterlim = 99999;

SOLVE SAMBAL USING NLP MINIMIZING OPT;
```

STEP 10: RESULTS COPIED TO A NEW MATRIX

Results are copied into a new matrix NSAM defined previously.

```
PARAMETER  NSAM(I,J)      New (balanced) matrix;
NSAM(INT,JNT)=SAM.L(INT,JNT);
```

Negative values are re-transposed to their original position in the new matrix.

```
NSAM(INT,JNT)$(NEG(INT,JNT)=1)=-NSAM(JNT,INT);
NSAM(JNT,INT)$(NEG(INT,JNT)=1)=0;
```

The new SAM is transformed into transaction flows.

```
* Transformation of proportions into SAM transaction flow
NSAM(INT,JNT) = NSAM(INT,JNT)*TOTO;
```

The new SAM is then stored in a GDX file, which is subsequently transformed into an Excel file. The "GDX2XLS" routine[12] creates an Excel spreadsheet in the form of a database, with one cell value per line. It can be returned to matrix form using Excel's Pivot Table facility, but the accounts may not be in

[12] Erwin Kalvelagen, http://www.gams.com/dd/docs/tools/gdx2xls.pdf

the desired order. A better option is to use the GDXXRW.EXE program[13], which enables to create an Excel table. Here, the NSAM parameter of the GDX file Results.gdx is written to output file NewSAM.xls; the data range in the Excel sheet, including row and column headings, is A1:W23, with one row dimension (rdim) and one column dimension (cdim).

```
* Export results first to a GDX file, then to an Excel file
execute_unload "Results", NSAM;
*execute '=gdx2xls Results.gdx';
Execute 'GDXXRW.EXE Results.gdx O=NewSAM.xls par=NSAM rng=A1:W23 rdim=1 cdim=1';
```

STEP 11: PROBLEM DETECTION

A parameter matrix named PROBS is created. It takes the value 1 if the corresponding value in the new balanced matrix is zero, while the corresponding value in the *a priori* matrix is non-zero. All values of PROBS should be zero.

```
*11. Problem detection
PARAMETER
PROBS(I,J);
PROBS(I,J)=0;
PROBS(I,J)$(NSAM(I,J) EQ 0 AND SAM0(I,J) NE 0)=SAM0(I,J);
DISPLAY PROBS;
```

REFERENCES

Bacharach, M. (1970) *Biproportional matrices and input-output change*, Cambridge University Press, London.

Byron, R P (1978) "The Estimation of Large Social Account Matrices", *Journal of the Royal Statistical Society*, Series A, 141 (3); 359-367.

Channing, A., **Robinson** S. and **Tarp** F. (1999) *Parameter Estimation for a Computable General Equilibrium Model: A Maximum Entropy Approach*, Purdue University, IFPRI and University of Copenhagen. Discussion paper No 40.

Dietzenbacher, E. and R.E. **Miller** (2009) "RAS-ing the transactions or the coefficients: It makes no difference", *Journal of Regional Science* 49(3), 555-566.

GAMS Development Corporation (2013) *GAMS GDX facilities and tools*, GAMS Development Corporation.
http://www.gams.com/dd/docs/tools/gdxutils.pdf

Golan, A., G. **Judge** and D. **Miller** (1996). Maximum entropy econometrics: robust estimation with limited data, John Wiley & Sons.

Günlük-Şenesen, G. and J. M. **Bates** (1988) "Some Experiments with Methods of Adjusting Unbalanced Data Matrices", *Journal of the Royal Statistical Society*, 141; 473-490.

Hildreth, C. and **Houck**, J.P. (1968) "Some estimators for a linear model with random coefficients", *Journal of American Statistical Association*, 63, 583-594.

[13] See "5.3.9 Read / Write spreadsheet: Example 13" in GAMS Development Corporation (2013).

Jaynes, E. T. (1957) "Information theory and statistical mechanics", *Physical Review*, vol. 106, 620-630 et vol. 108, 171-190.

Junius, T. and J. **Oosterhaven** (2003) "The solution of updating or regionalizing a matrix with both positive and negative entries", *Economic Systems Research*, 15, 87-96.

Kapur, J. N. and H. K. **Kesavan** (1992) *Entropy Optimization Principles with Applications*, New York, Academic Press.

Kullback, S. and R. A. **Leibler** (1951), "On Information and Sufficiency" *Ann. Math. Stat*. 4, 99-111.

Lemelin, A. (2011) *Balancing international financial data: Fun and games with GAMS and cross-entropy*, Poverty & Economic Policy Research Network (PEP), MPIA Working Paper 2011-07. http://www.pep-net.org/publications/working-papers/

Lemelin, A. (2009) "A GRAS variant solving for minimum information loss", *Economic Systems Research*, 21(4), 399-408.

Lemelin, A. (2002) *Aide-mémoire sur l'entropie*, INRS-Urbanisation, Coll. Inédits, janvier, 105 p.

McDougall, R. A. (1999) *Entropy Theory and RAS are Friends*, GTAP Working Paper No. 03, GTAP, Purdue, Indiana. https://www.gtap.agecon.purdue.edu/resources/download/10.pdf.

Macgill, S. M. (1977) "Theoretical properties of biproportional matrix adjustments", *Environment and Planning A*, 9: 687-701.

Macgill, S. M. and A. G. **Wilson** (1979) "Equivalences and similarities between some alternative urban and regional models", *Sistemi Urbani*, 1: 9-40.

Miller, Ronald E., and **Blair** Peter (2009) *Input-output analysis : foundations and extensions*, Cambridge University Press.

Peeters, L. and **Surry** Y. (1999) *Farm cost allocation based on the maximum entropy methodology. The case of Saskatchewan crop farms*, Agriculture and Agri-Food Canada, Ottawa. http://ageconsearch.umn.edu/bitstream/54461/2/allocation_e.pdf

Peeters, L. and **Surry**, Y. (2002) *Maximum entropy estimation of random input-output coefficients. An application to farm accounting data*, Paper presented at the 10th Congress of the European Association of Agricultural Economists (EAAE), Saragossa, Spain (2002-08-28).

Robilliard, A. S., and **Robinson**, S., 1999 *Reconciling Household Surveys and National Accounts Data Using a Cross Entropy Estimation Method*, IFPRI. Discussion Paper No 50.

Robinson, S., and M. **El-Said** (2000) *GAMS Code for Estimating a Social Accounting Matrix (SAM) Using Cross Entropy (CE) Methods*, IFPRI. Discussion Paper No 64.

Robinson, S., A. **Cattaneo**, & M. **El-Said** (2001) "Updating and estimating a Social Accounting Matrix using Cross-Entropy methods", *Economic Systems Research*, 13, 47-64.

Robinson, S., A. **Cattaneo** and M. **El-Said** (1998) *Estimating a Social Accounting Matrix Using Cross Entropy Methods*, IFPRI. Discussion Paper No 33.

Round, J. I. (2003). "Constructing SAMs for development policy analysis. Lessons learned and challenges ahead", *Economic Systems Research*, 15(2), 161-183.

Schneider, M. H. and **Zenios**, S. A. (1990) "A comparative study of algorithms for matrix balancing", *Operations Research*, 38, 439–455.

Shannon, C. E. (1948) "A Mathematical theory of communication", *Bell System Technical Journal* 27, 379-423.

Stone, R (1977) “Foreword” to G. Pyatt and A. Roe and Associates, *Social Accounting for Development Planning*, Cambridge University Press, Cambridge; xvi-xxxi.

Stone, J. R. N., D. **Champernowne** and J. E. **Meade** (1942) “The Precision of National Income Estimates”, *Review of Economic Studies*, Vol IX, No 2; 111-125.

Theil, H. (1967), *Economics and information theory*, Rand McNally & Company, Chicago, Studies in mathematical and managerial economics, 7, 488 pages.