Reconstructing Matrices with Linear Programming

Luis Frank [*]

Abstract
The article explains how to adjust, balance and temporarily disaggregate matrices (to "reconstruct" matrices, in a broad sense) with linear programming. The main advantage of this method compared to RAS (standard method recommended in National Accounts manuals) is that it allows reconstructing matrices even with missing data in the benchmark matrix, or with marginal vectors that are inconsistent with each other. Reconstructing matrices with linear programming also allows enriching the solution by incorporating alternative or complementary information to the adjustment process. The article also includes numerical examples with small matrices to show the performance of the proposed method.

Key Words: matrix adjustment, matrix balancing, matrix reconciliation, linear programming, official statistics, National Accounts.

1. Introduction

Suppose we are provided with a benchmark matrix $\mathbf{X}_0$ of dimensions $n \times k$ and two vectors $\mathbf{u}$ and $\mathbf{v}'$ of dimensions $n \times 1$ and $1 \times k$, respectively, and we are asked to find another matrix $\mathbf{X}$ (of the same size as $\mathbf{X}_0$) such that $\mathbf{u} = \mathbf{X}\mathbf{1}_k$ and $\mathbf{v}' = \mathbf{1}_n'\mathbf{X}$. Or suppose we are provided with two matrices, $\mathbf{X}_0$ and $\mathbf{Y}_0$, and the corresponding summation vectors $\mathbf{u}$ and $\mathbf{v}'$, and we are asked to find two matrices $\mathbf{X}$ and $\mathbf{Y}$ whose rows and columns add up to these vectors and also the summ of the columns of $\mathbf{X}$ equal that of $\mathbf{Y}$. These kinds of requirements appear frequently in National Accounts offices, for example, to balance supply and use tables or to update input-output matrices, and have motivated a broad amount of methods (see [7, § 18.1] for a comprehensive review on this topic) known as methods for updating, reconciling, balancing or projecting matrices.[1] Among these methods, the most widely disseminated and also recommended by the National Accounts manuals ([6, § IX] y [7, § 2]), is the RAS method of bi-proportional adjustment.

The RAS method is an algorithm that iteratively multiplies each row of $\mathbf{X}_0$ by the ratio $u_i/(\mathbf{x}_{0i}'\mathbf{1}_k)$, and each column by $v_j/(\mathbf{1}_n'\mathbf{x}_{0j})$, until convergence. Although it is an effective method for finding $\mathbf{X}$ it has the disadvantage of reducing the adjustment only to the information provided by the benchmark matrix $\mathbf{X}_0$ and the totals $\mathbf{u}$ and $\mathbf{v}$. That is, the choice of this method implies a decision to exclude from benchmarking any other potentially useful information to reconstruct $\mathbf{X}$ other than that provided by $\mathbf{X}_0$, $\mathbf{u}$ and $\mathbf{v}$. This decision has serious implications, since (i) the selection of the method constitutes a way of (involuntary) manipulation of the results by exclusion of alternatives or complementary but relevant information sources; and, (ii) the choice of the RAS method in particular leads to sub-optimal results because it can not handle rows or columns with missing or defective data in the benchmark matrix $\mathbf{X}_0$. The seriousness of these arguments justifies the search for an alternative method for matrix-reconstruction with the following features: (i) be efficient in the use of information, that is, to use all available information sources; (ii) allow the

[*]Universidad of Buenos Aires. Facultad de Agronomía. Av. San Martín 4453, C1417DSE. Buenos Aires, Argentina.

[1]Throughout the paper we will talk in general of methods for matrix-recostruction, since they are intended for remaking, repairing or reproducing matrices based on new information about their structure.

reconstruction of $\mathbf{X}$ with incomplete or defective benchmark matrices; (iii) be solvable with known optimization criteria and standard algorithms; (iv) be extensible to more complex problems, such as balancing of complementary matrices or even temporal disaggregation of matrices.

2. Objectives

The objective of the following paper is twofold: first, to present a more flexible adjustment method than the RAS method; and second, to extend the method to solve related problems, such as balancing of complementary matrices or temporarily disaggregating of matrices. To that end we will use Linear Programming to minimize the sum of the absolute discrepancies between the benchmark matrix and the new given totals.[2] Along the paper we will not focus on optimization algorithms but on the statement of the set of linear constraints that solve each type of problem. In the end, we expect to compile a sort of methodological guide for matrix reconstruction useful for official statistics offices. This guide complements two other papers of the author [1, 2] on interpolation and reconciliation of series with linear programming.

Before continuing, it is convenient to make a brief digression to clarify the notation that we will follow throughout the paper. All matrices and vectors are written in bold; matrices are written upper-case and vectors lower-case. Scalars are written in italics. Vectors are always column vectors. The subindices at the foot of each matrix indicate the dimensions of the matrix as the product of rows by column. The dimensions of square matrices are written with a single subindex at the foot. Bold numbers refer to matrices or vectors whose elements are the same as the name of the matrix. For example, $\mathbf{1}_n$ represents a column vector of 1 of dimension $n \times 1$. $\mathbf{I}$ is the identity matrix, that is, a square matrix with 1 on the diagonal and 0 elsewhere.

3. El Basic Program

The general method is to solve a linear program that minimizes a certain distance between the benchmark matrix and the adjusted or reconstructed matrix (this is the so-called objective function) subject to a system of linear constraints. Formally, the program is expressed

$$\min_{\mathbf{x}} \left\{\mathbf{z}'\mathbf{x}\right\} \quad \text{subject to} \quad \mathbf{A}\mathbf{x} = \mathbf{b},\ \mathbf{x} \geq \mathbf{0}, \tag{1}$$

where $\mathbf{z}$ is a weighting vector, and $\mathbf{A}$ and $\mathbf{b}$ are known matrices that define system of linear constraints that relates the adjusted matrix with the benchmark matrix. The solution $\mathbf{x}$ is a function of the adjusted matrix. The main complexity of the program is to define the proper set of the linear constraints. That is why we will first focus on this point and leave for later the description of the objective function. So next we will describe step by step how to build the system of constraints that will allow us to find $\mathbf{X}$.

[2]Lahr and de Mesnard [5] were the first authors that suggested the minimization of absolute errors to adjust matrices, but discouraged it's use because of the possibility of getting negative elements in the solution (see [5, pp. 123-124]), which shows that they did not visualize the adjustment as an optimization problem by Linear Programming. Huang et al. cite Huang2008 picked up the point and proposed various objective functions to deal with the problem of negative elements, but always under the premise that the problem should be solved by non-linear programming.

First, we establish that the solution $\mathbf{x} = \text{vec}(\mathbf{X})$ must be compatible with the vectors $\mathbf{u}$ and $\mathbf{v}$. That is, $\mathbf{1}'_n\mathbf{X} = \mathbf{v}'$ and $\mathbf{X}\mathbf{1}_k = \mathbf{u}$, as it would be expect from any adjustment method. These restrictions can be expressed together as follows:

$$\begin{bmatrix} \mathbf{1}'_k \otimes \mathbf{I}_n \\ \mathbf{I}_k \otimes \mathbf{1}'_n \end{bmatrix} \text{vec}(\mathbf{X}) = \begin{bmatrix} \mathbf{u} \\ \mathbf{v} \end{bmatrix}, \tag{2}$$

where the operator vec(.) is a function that vectorizes the matrix in the argument, placing the column $j+1$ below the column j; and the symbol $\otimes$ represents the Kronecker product. Note that, strictly speaking, the system (2) does not establish that the sum of elements of $\mathbf{u}$ must be equal to that of $\mathbf{v}$, that is, that $\mathbf{1}'_n\mathbf{u} = \mathbf{1}'_k\mathbf{v}$. The relaxation of this condition is useful because it often happens that these sums do not match by rounding of figures, which breaks the RAS algorithm. Let's call $\mathbf{A}_{11}$ to the matrix that premultiplies the left side of equality, $\mathbf{x}_1$ to vec($\mathbf{X}$) and $\mathbf{b}_1$ to the vector on the right hand side, so that the expression (2) can be rewritten more compactly as $\mathbf{A}_{11}\mathbf{x}_1 = \mathbf{b}_1$. The dimension of $\mathbf{A}_{11}$ is $(n+k) \times nk$, that of $\mathbf{x}_1$ is $nk \times 1$, and $\mathbf{b}_1$ has dimension $(n+k) \times 1$.

Second, we are interested in introducing a set of constraints to (i) exploit all the information provided by $\mathbf{X}_0$; and, (ii) incorporate independent information from $\mathbf{X}_0$, and from $\mathbf{u}$ and $\mathbf{v}$ of course. To comply with (i) we interpret each element of $\mathbf{X}_0$ as the sum of the corresponding element of $\mathbf{X}$ plus a "discrepancy" that might be positive and negative in a way similar to the objective programming model quoted by H. P. Williams [8, pp. 32-34]. Then, for all the elements of $\mathbf{X}_0$ y $\mathbf{X}$ this relationship may be expressed as

$$\begin{bmatrix} \mathbf{I}_{nk}, & \mathbf{I}_{nk} \otimes \mathbf{d}' \end{bmatrix} \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{bmatrix} = \text{vec}(\mathbf{X}_0), \tag{3}$$

where $\mathbf{d}' = [1, -1]$, and $\mathbf{x}_2$ is a vector of positive discrepancies, that is $\mathbf{x}_2 \geq \mathbf{0}_{2nk}$. Logically, it is possible to add to the system as many rows as additional information sources become available, although for explanatory clarity we will discuss this possibility in the next section. Then, if we call $\mathbf{A}_{22} = \mathbf{I}_{nk} \otimes \mathbf{d}'$ and $\mathbf{b}_2 = \text{vec}(\mathbf{X}_0)$, expressions (2) and (3) can be composed in a single block system like the following.

$$\begin{bmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{A}_{21} & \mathbf{A}_{22} \end{bmatrix} \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{bmatrix} = \begin{bmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \end{bmatrix}, \quad \mathbf{A}_{12} = \mathbf{0}_{(n+k)\times 2nk}, \quad \text{and} \quad \mathbf{A}_{21} = \mathbf{I}_{nk}. \tag{4}$$

$\mathbf{A}_{22}$ has dimension $nk \times 2nk$, and the vectors $\mathbf{x}_2$ y $\mathbf{b}_2$ have dimensions $2nk \times 1$ y $nk \times 1$, respectively. This system, together with the objective function that follows, defines the elementary linear program of a whole family of adjustment or reconciliation problems, so we will return to it several times throughout the text.

At this point we are able to define the objective function $\mathbf{z}'\mathbf{x}$. The chosen optimization criterium is to minimize the sum of absolute discrepancies between $\mathbf{X}_0$ and $\mathbf{X}$, for which we decompose $\mathbf{z}'\mathbf{x}$ in the sum of two terms, $\mathbf{z}'_1\mathbf{x}_1 + \mathbf{z}'_2\mathbf{x}_2$. In the first term $\mathbf{z}_1 = \mathbf{0}_{nk}$, while in the second $\mathbf{z}_2 = \mathbf{1}_{2nk}$, so that the objective function is simply the sum of the absolute discrepancies between the elements of $\mathbf{X}$ and $\mathbf{X}_0$. In the appendix we show a numerical example with fictitious data. We will not delve here into the statistical properties of the discrepancies and just consider the vector $\mathbf{x}_1$ as a mere *solution* to the problem posed, which has the advantage of being unique under the chosen optimization criterium.

4. Missing Data and Inconsistent Marginal Vectors

Program (1) together with the set of linear constraints (4) can be expanded or reformulated to incorporate independent information that compensates for missing data in the reference matrix $\mathbf{X}_0$, or simply to enrich the information provided by it, or even to temporarily disaggregate one or more reference matrices. Next, we typify these problems and describe the appropriate system of constraints to solve each case.

4.1 Alternative or Complementary Information

Occasionally, additional information sources other than matrix $\mathbf{X}_0$ become available to reconstruct the structure of $\mathbf{X}$. This information can be incorporated into the system of constraints (4) adding rows and columns to it. There are two possible (although not exclusive) ways to incorporate complementary information. One, represented by the system $\mathbf{A}_{31}\mathbf{x}_1 = \mathbf{b}_3$ in the expression (5) in which we introduce strict equality constraints on r elements of $\mathbf{X}$, as if those elements were perfectly known, for example, through out a census. Then, the structure of $\mathbf{A}_{31}$ will be that of an array of the r rows of $\mathbf{I}_{nk}$ that correspond to the elements of $\mathbf{x}$ known perfectly. The other way, represented by the last line of blocks in (5), is a set of approximate constraints on s elements of $\mathbf{X}$ in the same fashion as in $[\mathbf{A}_{21}\ \mathbf{A}_{22}]\mathbf{x} = \mathbf{b}_2$. The structure of $\mathbf{A}_{41}$ is similar to that of $\mathbf{A}_{31}$ although the s elements of $\mathbf{A}_{41}$ no will match the r elements of $\mathbf{A}_{31}$ since the exact constraint would override the approximate constraint.

$$\left[\begin{array}{cc:c} \mathbf{A}_{11} & \mathbf{A}_{12} & \mathbf{0}_{(n+k)\times 2s} \\ \mathbf{A}_{21} & \mathbf{A}_{22} & \mathbf{0}_{nk\times 2s} \\ \hdashline \mathbf{A}_{31} & \mathbf{0}_{r\times 2nk} & \mathbf{0}_{r\times 2s} \\ \mathbf{A}_{41} & \mathbf{0}_{s\times 2nk} & \mathbf{I}_s \otimes \mathbf{d}' \end{array}\right] \left[\begin{array}{c} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \hdashline \mathbf{x}_{2'} \end{array}\right] = \left[\begin{array}{c} \mathbf{b}_1 \\ \mathbf{b}_2 \\ \hdashline \mathbf{b}_3 \\ \mathbf{b}_4 \end{array}\right] \tag{5}$$

It is also possible to bound certain elements of $\mathbf{X}$. To that end, we just add another line of blocks to the matrix on the left hand side of (5) (although the only relevant block on that line is $\mathbf{A}_{51}$) and replace the sign of equality by another of inequality as shown in (6). Each element of $\mathbf{X}$ to be bounded will add a couple of rows to $\mathbf{A}_{51}$ and $\mathbf{b}_5$ in a similar fashion to those of the exact constraints. However, the row corresponding to the upper bound, both in $\mathbf{A}_{51}$ and in $\mathbf{b}_5$, should be multiplied by -1 to let the inequality sign of the upper bound be the same as that of the lower bound. In those software where the inequality sign can be defined by the user this caution is unnecessary. Example A.1 of the appendix will help to better understand this structure.

$$\left[\begin{array}{ccc} \mathbf{A}_{51} & \mathbf{0}_{t\times 2nk} & \mathbf{0}_{t\times 2s} \end{array}\right] \left[\begin{array}{c} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \mathbf{x}_{2'} \end{array}\right] \geq \mathbf{b}_5 \tag{6}$$

It should be noted that if a certain element of $\mathbf{X}$ appears bounded the other constraints that could have been introduced on it are not canceled at all except for those of strict equality. Besides, if we expand the system of linear constraints the weighting vector of the objective function must be expanded too. If we put together expressions (5) and (6) in a single system, then $\mathbf{z}'$ will be

$$\mathbf{z}' = \left[\begin{array}{ccc} \mathbf{0}'_{nk} & \mathbf{1}'_{2nk} & \mathbf{1}'_{2s} \end{array}\right]. \tag{7}$$

4.2 Missing or Unlikely Data in the Benchmark Matrix

It is often the case that some elements of the benchmark matrix are faulty, either because there are missing data or because the data provided is really incredible. In general, data may be missing (i) because $\mathbf{X}_0$ comes from a small or badly drawn sample; and, (ii) because the database that lead to $\mathbf{X}_0$ was originaly classified in categories that did not match strictly to those of the current classification. In contrast, the presence of unlikely data is typically due to poor sampling designs or samples poorly expanded to the whole population. Whatever the cause of these defects, the obvious strategy to follow would be to delete from $[\mathbf{A}_{21}\ \mathbf{A}_{22}]\,\mathbf{x} = \mathbf{b}_2$ the suspicious rows. The practitioner should not worry about the fact that removing rows from block $[\mathbf{A}_{21}\ \mathbf{A}_{22}]$ leaves twice the amount of null columns in $\mathbf{A}_4$ since these do not represent an obstacle to compute of the solution. In principle, there would be no limit of the number of rows to be removed, but through numerical simulations we observed that the solution $\mathbf{X}$ may vary dramatically if the quantity of faulty elements of $\mathbf{X}_0$ is high. Then, if several rows of $[\mathbf{A}_{21}, \mathbf{A}_{22}]$ and $\mathbf{b}_2$ ought to be removed it is advisable to add additional information as explained in the previous section. Or, if additional information were not available, to bound the missing data or to benchmark it with reliable (although outdated) data from $\mathbf{X}_0$. This last strategy may be carried out simply adding additional constraints emulating the system $\mathbf{A}_{51}\mathbf{x}_1 \geq \mathbf{b}_5$ but replacing the exogenous bounds in the $\mathbf{b}_5$ with appropriate elements of $\mathbf{X}_0$.

4.3 Faulty Marginal Vectors

Virtually all adjustment methods assume that the marginal sums $\mathbf{u}$ and $\mathbf{v}$ are known precisely. However, it is possible that these vectors are faulty, either because some elements of $\mathbf{u}$ or $\mathbf{v}$ come from unreliable sources, or because they are directly unlikely, or simply because they are unknown. In such a case, the expression (2) can be reformulated to introduce discrepancy terms in the same way as in (3). In the extreme case that all the elements of $\mathbf{u}$ and $\mathbf{v}$ were faulty, the expression (2) might be rewritten as follows

$$\left[\begin{array}{c:cc} \mathbf{1}'_k \otimes \mathbf{I}_n & \mathbf{I}_n \otimes \mathbf{d}' & \mathbf{0}_{n\times 2k} \\ \mathbf{I}_k \otimes \mathbf{1}'_n & \mathbf{0}_{k\times 2n} & \mathbf{I}_k \otimes \mathbf{d}' \end{array}\right] \left[\begin{array}{c} \mathbf{x}_1 \\ \hdashline \mathbf{x}_2 \\ \mathbf{x}_{2'} \end{array}\right] = \left[\begin{array}{c} \mathbf{u} \\ \mathbf{v} \end{array}\right]. \tag{8}$$

For reasons that will become apparent later, we will call $\mathbf{A}_{13}$ to the matrix of dimension $(n+k) \times 2(n+k)$ on the right hand side of the dotted line, and $\mathbf{A}_{23}$ to the null array $\mathbf{0}_{nk\times 2(n+k)}$.

Now, because $\mathbf{u}$ and $\mathbf{v}$ are faulty, the sum of elements of one vector will not agree with that of the other. That is, the condition $\mathbf{1}'_n\mathbf{u} = \mathbf{v}'\mathbf{1}_k$ is no longer satisfied. Although this discrepancy is not an obstacle for finding a solution, one might desire also to get consistent marginal sums. To achieve this goal, it could be established that the sum of positive discrepancies of $\mathbf{u}$ be equal to that of $\mathbf{v}$, and that the sum of negative discrepancies of $\mathbf{u}$ be equal to that of $\mathbf{v}$. This is equivalent to introduce the following blocks into the system of constraints:

$$\left[\begin{array}{c:cc} \mathbf{1}'_k \otimes \mathbf{I}_n & \mathbf{I}_n \otimes \mathbf{d}' & \mathbf{0}_{n\times 2k} \\ \mathbf{I}_k \otimes \mathbf{1}'_n & \mathbf{0}_{k\times 2n} & \mathbf{I}_k \otimes \mathbf{d}' \\ \hdashline \mathbf{0}_{2\times nk} & \mathbf{1}'_n \otimes \mathbf{I}_2 & -\mathbf{1}'_k \otimes \mathbf{I}_2 \end{array}\right] \left[\begin{array}{c} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \mathbf{x}_{2'} \end{array}\right] = \left[\begin{array}{c} \mathbf{u} \\ \mathbf{v} \\ \hdashline \mathbf{0}_{2\times 1} \end{array}\right]. \tag{9}$$

The previous system is also useful for resolving small discrepancies between $\mathbf{u}$ and $\mathbf{v}$ due to rounding of figures. The weighting vector of the objective function for this specification of the system of constraints is $\mathbf{z}' = [\mathbf{0}'_{nk}\ \mathbf{1}'_{2n+2k}]$.

4.4 Forcing Signs in the Reconstructed Matrix

So far we have assumed that all elements of $\mathbf{X}$ were non-negative, and we restricted the solution accordingly. However, some elements may have a negative sign, not only because it is indicated so in the benchmark matrix, but also because it might be indecated by exogenous sources, or by theoretical considerations. To specify the sign of negative elements we proceed in the same way as to introduce bounds, but in this case calling $\mathbf{A}_{51} = -\mathbf{A}^*_{31}$ y $\mathbf{b}_5 = \mathbf{0}$, and replacing the sign of equality by one of inequality, that is, $-\mathbf{A}^*_{31}\mathbf{x}_1 \geq \mathbf{0}$. The asterisk shows that we do not refer to the same $\mathbf{A}_{31}$ matrix of (5) but to another, which for simplicity we will describe as an identity matrix of $nk \times nk$ without the rows of positive elements. The original RAS method does not allow to operate with negative values, so Junius and Oosterhaven [4] developed the generalized RAS method or GRAS.

5. Balancing Complementary Matrices

It is often necessary to balance two matrices so that the sum of the columns of one equals that of the other. That is, we are provided with two benchmark matrices, $\mathbf{X}_0$ and $\mathbf{Y}_0$, and two pairs of marginal totals, and we are asked to find two other matrices ($\mathbf{X}$ and $\mathbf{Y}$) that meet the equilibrium condition $\mathbf{X1}_k = \mathbf{Y1}_p = \mathbf{u}$, where $\mathbf{u}$ is the unknown marginal total that balances both matrices but is not necessarily equal to any of the given marginal totals $\mathbf{u}_\mathbf{X}$ and $\mathbf{u}_\mathbf{Y}$ although keeps a minimum distance with them. This problem typically appears in material balances in which the supply of each product must match its demand. The linear program associated with this problem is may be written

$$\min_{\mathbf{x},\mathbf{y}} \left\{\mathbf{z}'_\mathbf{x}\mathbf{x} + \mathbf{z}'_\mathbf{y}\mathbf{y}\right\} \text{ s. a } \mathbf{Ax} = \mathbf{b},\ \mathbf{By} = \mathbf{c},\ \left(\mathbf{1}'_k \otimes \mathbf{I}_n\right)\mathbf{x}_1 = \left(\mathbf{1}'_p \otimes \mathbf{I}_n\right)\mathbf{y}_1,\ \mathbf{x} \geq \mathbf{0},\ \mathbf{y} \geq \mathbf{0}.$$

The set of constraints can be expressed in a more compact fashion grouping them in a general system like the one below. We will call $h = n + p$ and $g = n + k$ to facilitate the notation but we warn the reader that we will permanently redefine these letters for convenience.

$$\left[\begin{array}{ccc:ccc} \mathbf{A}_{11} & \mathbf{A}_{12} & \mathbf{A}_{13} & \mathbf{0}_{g\times np} & \mathbf{0}_{g\times 2np} & \mathbf{0}_{g\times 2h} \\ \mathbf{A}_{21} & \mathbf{A}_{22} & \mathbf{A}_{23} & \mathbf{0}_{nk\times np} & \mathbf{0}_{nk\times 2np} & \mathbf{0}_{nk\times 2h} \\ \hdashline \mathbf{0}_{h\times nk} & \mathbf{0}_{h\times 2nk} & \mathbf{0}_{h\times 2g} & \mathbf{B}_{11} & \mathbf{B}_{12} & \mathbf{B}_{13} \\ \mathbf{0}_{np\times nk} & \mathbf{0}_{np\times 2nk} & \mathbf{0}_{np\times 2g} & \mathbf{B}_{21} & \mathbf{B}_{22} & \mathbf{B}_{23} \\ \hdashline \mathbf{1}'_k \otimes \mathbf{I}_n & \mathbf{0}_{n\times 2nk} & \mathbf{0}_{n\times 2g} & -\mathbf{1}'_p \otimes \mathbf{I}_n & \mathbf{0}_{n\times 2np} & \mathbf{0}_{n\times 2h} \end{array}\right] \left[\begin{array}{c} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \hdashline \mathbf{y}_1 \\ \mathbf{y}_2 \end{array}\right] = \left[\begin{array}{c} \mathbf{b}_1 \\ \mathbf{b}_2 \\ \hdashline \mathbf{c}_1 \\ \mathbf{c}_2 \\ \hdashline \mathbf{0}_n \end{array}\right], \tag{10}$$

where $\mathbf{B}$ is an an array analogous to $\mathbf{A}$ and $\mathbf{c}$ is a vector analogous to $\mathbf{b}$. The upper left area of the matrix that pre-multiplies the solution is an augmented version of $\mathbf{A}$ in equation (4) but with the blocks $\mathbf{A}_{13}$ and $\mathbf{A}_{23}$ (already defined in the section dealing with the case of faulty marginal totals). The addition of these blocks is essential to find a non-trivial solution. Omitting them would mean that the marginal sums $\mathbf{u}_\mathbf{X}$ and $\mathbf{u}_\mathbf{Y}$ do not change but at the same time are identical. On the diagonal after $\mathbf{A}$, the reader may find matrix $\mathbf{B}$ which is completely which is completely analogous to $\mathbf{A}$. In fact, if $\mathbf{Y}$ had the same dimensions as $\mathbf{X}$, $\mathbf{B}$ would be identical to $\mathbf{A}$. The balance constraint between $\mathbf{X}$ and $\mathbf{Y}$ is located in the last line. Logically, if the balance is not intended to be perfect but only approximate, the system (10) should be expanded in the same way as in (3) adding a discrepancy block.

$$\left[\begin{array}{cc:c} \mathbf{A} & \mathbf{0}_{m\times r} & \mathbf{0}_{m\times 2n} \\ \mathbf{0}_{l\times s} & \mathbf{B} & \mathbf{0}_{l\times 2n} \\ \hdashline \mathbf{D}_{11} & \mathbf{D}_{12} & \mathbf{I}_n \otimes \mathbf{d}' \end{array}\right] \left[\begin{array}{c} \mathbf{x} \\ \mathbf{y} \\ \hdashline \mathbf{e} \end{array}\right] = \left[\begin{array}{c} \mathbf{b} \\ \mathbf{c} \\ \hdashline \mathbf{0}_n \end{array}\right] \tag{11}$$

where

$$\mathbf{D}_{11} = \left[\begin{array}{cc} \mathbf{1}'_k \otimes \mathbf{I}_n & \mathbf{0}_{n\times 2(nk+g)} \end{array}\right] \quad \text{and} \quad \mathbf{D}_{12} = \left[\begin{array}{cc} -\mathbf{1}'_p \otimes \mathbf{I}_n & \mathbf{0}_{n\times 2(np+h)} \end{array}\right],$$

and $m = g + nk$; $l = h + np$; $r = 3np + 2h$; y $s = 3nk + 2g$. Again we warn the reader not to confuse these variables with others of the same name in other sections of the paper. The objective function of the program (10) is the sum of the discrepancy terms in $\mathbf{x}_2$ and $\mathbf{y}_2$ obtained through the product $\mathbf{z}' \, [\mathbf{x}', \mathbf{y}']'$, where

$$\mathbf{z}' = \begin{bmatrix} \mathbf{0}'_{nk} & \mathbf{1}'_{2(nk+g)} & \mathbf{0}'_{np} & \mathbf{1}'_{2(np+h)} & \mathbf{1}'_{2n} \end{bmatrix}.$$

In case of imperfect equilibrium, the objective function is similar, except that in the weighting vector the last block $\mathbf{1}'_{2(n+h)}$ is replaced by $\mathbf{1}'_{2(2n+h)}$. In the appendix we present a numerical example.

6. Temporal Disaggregation of Matrices

Sometimes we need to disaggregate a matrix temporarily using a set of time series related to one of the dimensions of the matrix. For example, we might be interested in decomposing an annual supply matrix using a set of time series of quarterly gross production value (GPV) by industry. To do so we are provided with the following information:

(i) The annual GPV by product ($\mathbf{u}$) and industry ($\mathbf{v}$). The first is known only for the year in question. The sum of the elements $\mathbf{u}$ matches that of $\mathbf{v}$; the dimensions of $\mathbf{u}$ and $\mathbf{v}$ are $n \times 1$ and $k \times 1$, respectively.

(ii) A succession of quarterly industry GPV vectors of which we are only interested in the four ($\mathbf{v}^{(1)}, \ldots, \mathbf{v}^{(4)}$) that add up to the annual total $\mathbf{v}$.

(iii) An indefinite number of quarterly series (complementary to those of the GVP) of certain products by industry. These series may contain precise or approximate values. In both cases, these are continuous series, although later we will relax this assumption.

The reader will notice that so far we do not require an annual benchmark matrix, or quarterly benchmark matrices $\mathbf{X}_0^{(t)}$ for the task. So, based on the given information, we intend to find four matrices $\mathbf{X}^{(t)}$ that satisfy the following conditions:

(i) The sum of quarterly totals $\mathbf{1}'_n\mathbf{X}^{(1)} + \cdots + \mathbf{1}'_n\mathbf{X}^{(4)}$ and $\mathbf{X}^{(1)}\mathbf{1}_k + \cdots + \mathbf{X}^{(4)}\mathbf{1}_k$ must equal the given annual totals $\mathbf{v}'$ and $\mathbf{u}$.

(ii) The rows of each $\mathbf{X}^{(t)}$ must add up to the corresponding quarterly vector $\mathbf{v}^{(t)}$, that is $\mathbf{1}'_n\mathbf{X}^{(t)} = \mathbf{v}^{(t)\prime}$.

(iii) The solution must also match those elements of $\mathbf{X}^{(t)}$ that are known exactly from alternative sources. We will assume that we know r elements belonging to $r/4$ alternative series.

(iv) The discrepancy between the solution and certain s elements of $\mathbf{X}^{(t)}$ that are known only approximately must be minimal. We will assume that these s elements belong to $s/4$ alternative time series.

The four conditions can be expressed in a single system of restrictions like the one shown below. For ease of reading we have divided **A** and **b** with dotted lines.

$$\left[\begin{array}{ccc:ccc}
\mathbf{A}_{11} & \cdots & \mathbf{A}_{11} & \mathbf{0}_{(n+k)\times 2nk} & \cdots & \mathbf{0}_{(n+k)\times 2nk} \\ \hdashline
\mathbf{I}_k \otimes \mathbf{1}'_n & \cdots & \mathbf{0}_{k\times nk} & \mathbf{0}_{k\times 2nk} & \cdots & \mathbf{0}_{k\times 2nk} \\
\vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\
\mathbf{0}_{k\times nk} & \cdots & \mathbf{I}_k \otimes \mathbf{1}'_n & \mathbf{0}_{k\times 2nk} & \cdots & \mathbf{0}_{k\times 2nk} \\ \hdashline
\mathbf{A}_{31} & \cdots & \mathbf{0}_r & \mathbf{0}_{r\times 2nk} & \cdots & \mathbf{0}_{r\times 2nk} \\
\vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\
\mathbf{0}_r & \cdots & \mathbf{A}_{31} & \mathbf{0}_{r\times 2nk} & \cdots & \mathbf{0}_{r\times 2nk} \\ \hdashline
\mathbf{A}_{41} & \cdots & \mathbf{0}_s & \mathbf{A}_{22} & \cdots & \mathbf{0}_{s\times 2nk} \\
\vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\
\mathbf{0}_s & \cdots & \mathbf{A}_{41} & \mathbf{0}_{s\times 2nk} & \cdots & \mathbf{A}_{22}
\end{array}\right]
\left[\begin{array}{c}
\mathbf{x}_1^{(1)} \\ \vdots \\ \vdots \\ \mathbf{x}_1^{(4)} \\ \mathbf{x}_2^{(1)} \\ \vdots \\ \vdots \\ \vdots \\ \mathbf{x}_2^{(4)}
\end{array}\right]
=
\left[\begin{array}{c}
\mathbf{b}_1 \\ \hdashline \mathbf{v}^{(1)} \\ \vdots \\ \mathbf{v}^{(4)} \\ \hdashline \mathbf{b}_3^{(1)} \\ \vdots \\ \mathbf{b}_3^{(4)} \\ \hdashline \mathbf{b}_4^{(1)} \\ \vdots \\ \mathbf{b}_4^{(4)}
\end{array}\right] \tag{12}$$

The first condition is represented by the upper left area. This condition ensures that the sum of quarterly totals, by row and column, matches **u** and $\mathbf{v}'$. The immediately lower area represents a block diagonal matrix in which each block $\mathbf{I}_k \otimes \mathbf{1}'_n$ reproduces the set of constraints in (2) that guarantees that The sum of the rows matches the given total. In this case the condition is met for each quarter, as established in (ii). The next two areas below introduce strict and approximate equality restrictions, respectively, based on alternative time series. By representing conditions (iii) and (iv) in this fashion we have assumed that all alternative series are continuous, and that is why all diagonal blocks are the same. If the time series were discontinuous, the blocks within each area would be different because of the elimination of those rows with missing data.

On the basis of the system (12) we can propose some variants adapted to the available information. For example,

(a) If we are provided with an annual benchmark matrix, we could introduce a system of restrictions of the type

$$\left[\begin{array}{ccc} \mathbf{1}'_4 \otimes \mathbf{I}_{nk} & \mathbf{0}_{nk\times h} & \mathbf{I}_{nk} \otimes \mathbf{d}' \end{array}\right] \mathbf{x} = \mathbf{b}_2, \tag{13}$$

where h is the number of columns needed to make the system conformable with other constraints, and $\mathbf{b}_2$ is the vectorized annual benchmark matrix, that is $\mathbf{b}_2 = \mathrm{vec}(\mathbf{X}_0)$.

(b) If quarterly benchmark matrices were available, they could be introduced in the system of constraints in the same way as the proxy series.. That is, $\mathbf{b}_4^{(t)} = \mathrm{vec}(\mathbf{X}_0^{(t)})$ and logically $s = nk$.

7. Conclusion

In the paper we show how to adjust, balance and disaggregate (*reconstruct* in general) matrices with linear programming, motivated mainly by the inability of standard methods (particularly RAS) to incorporate complementary information and deal with missing or faulty data. This inability is currently a matter of debate in the field of public statistics, as points out in the 2018 edition of the United Nations manual on supply, use and input-output tables [7, §18.14 p. 477],

> “It is important to note that although the projection problem has given rise to a number of attractive mathematical features, they are often not combined with survey data, other data sources or expert opinions on certain key elements like rows, columns or individual cells.”

In addition to this remark about the waste of useful information we add our own concern about the possibility of involuntarily manipulating results by selecting methods that exclude relevant information. Adjusting with linear programming solves this and other related problems such as that of missing or faulty data (a problem seldom treated in the bibliography) but has the disadvantage of producing results that are incomparable. This is so because when applying a standard method (RAS, for example) the differences between two reconstructed matrices can only be due to differences in the given data, that is, $\mathbf{X}_0$, $\mathbf{u}$ and $\mathbf{v}$. However, with the procedure described above, the differences between two solutions, besides being explained by the input data, can also be explained by the available sources of information that complement the benchmark matrix, the quantity of missing or (discarded) faulty data, the quantity and precision of related time series produced by the statistical office, etc. In other words, the adjustment method we propose relies heavily on the quality of the Statistical System as a whole because it depends completely on the abundance of information of good quality.

A. Numerical Examples

Example A.1. (Adjustment of a matrix to given marginal totals) $\mathbf{X}_0$ is the matrix we want to adjust, and $\mathbf{u}$ and $\mathbf{v}'$ are the marginal totals of a hypothetical matrix $\mathbf{X}$ consistent with them. The reader can verify that neither the sum of the columns of $\mathbf{X}_0$ match $\mathbf{u}$, nor does the sum of the rows of the same matrix match $\mathbf{v}\,'$. However, the sum of the $\mathbf{u}$ elements is equal to that of the elements of $\mathbf{v}'$.

$$\mathbf{X}_0 = \begin{bmatrix} 5 & 3 \\ 1 & 2 \\ 9 & 1 \end{bmatrix}, \quad \mathbf{u} = \begin{bmatrix} 7 \\ 4 \\ 7 \end{bmatrix} \quad \text{y} \quad \mathbf{v}' = \begin{bmatrix} 11 & 7 \end{bmatrix}.$$

The expansion of the expression (4) is the following. For clarity, the elements of the solution keep the subscript they would have in $\mathbf{X}$, and the elements we called e are errors or discrepancies. The subscripts of the discrepancies are those of the associated element of $\mathbf{X}$. The supra-indexes $+$ or $-$ indicate the sign of the difference between each element of the outdated matrix and the corresponding element in the solution matrix. The weighting vector of the objective function for this problem is $\mathbf{z}' = [\mathbf{0}_6' \ \mathbf{1}_{12}']$.

$$\left[\begin{array}{cccccc:cccccccccccc}
1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\hdashline
1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\hdashline
1 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1
\end{array}\right]
\begin{bmatrix} x_{11} \\ x_{21} \\ x_{31} \\ x_{12} \\ x_{22} \\ x_{32} \\ e_{11}^- \\ e_{11}^+ \\ \vdots \\ e_{32}^- \\ e_{32}^+ \end{bmatrix}
=
\left[\begin{array}{c} 7 \\ 4 \\ 7 \\ \hdashline 11 \\ 7 \\ \hdashline 5 \\ 1 \\ 9 \\ 3 \\ 2 \\ 1 \end{array}\right].$$

Next we transcribe the solution found with linear programming (LP) and for comparison purposes the solution found with RAS. The latter was obtained in the ninth iteration and with a maximum tolerance of 1×10^{-5}. Both calculations were made with own software written for Euler Math Toolbox.[3] In the case of the LP method we used the simplex algorithm included in Euler. The fact that the elements of the LP solution are all integers is not casual but the consequence of using integers in the benchmark matrix and marginal totals. The reader can verify that both solutions satisfy $\mathbf{v}' = \mathbf{1}_3'\mathbf{X}$ and $\mathbf{u} = \mathbf{X}\mathbf{1}_2$.

$$\mathbf{X}_{\text{LP}} = \begin{bmatrix} 4 & 3 \\ 1 & 3 \\ 6 & 1 \end{bmatrix} \quad \mathbf{X}_{\text{RAS}} = \begin{bmatrix} 3,85 & 3,15 \\ 1,07 & 2,93 \\ 6,08 & 0,92 \end{bmatrix}.$$

$\square$

Example A.2. (Approximate Matrix Balancing) Let us consider again matrices $\mathbf{X}_0$, $\mathbf{u}_\mathbf{X}$ and $\mathbf{v}_\mathbf{X}$ from example A .1, plus the following three analogous matrices:

$$\mathbf{Y}_0 = \begin{bmatrix} 3 & 5 \\ 2 & 2 \\ 3 & 4 \end{bmatrix}, \quad \mathbf{u}_\mathbf{Y} = \begin{bmatrix} 9 \\ 5 \\ 5 \end{bmatrix} \quad \text{y} \quad \mathbf{v}_\mathbf{Y}' = \begin{bmatrix} 6 & 13 \end{bmatrix}.$$

[3]Software freely downloadable from `http://euler-math-toolbox.de/download.html`

We want to find a vector $\mathbf{u}^*$ that has a minimum distance with $\mathbf{u_X}$ and $\mathbf{u_Y}$, two matrices $\mathbf{X}$ and $\mathbf{Y}$ compatible with $\mathbf{u}^*$ and the corresponding vectors $\mathbf{v}^*_\mathbf{X}$ and $\mathbf{v}^*_\mathbf{X}$. The constraint system associated with this problem is relatively simple because the matrices to be balanced are of the same dimensions, so $\mathbf{A} = \mathbf{B}$. Then, the system of constraints will be

$$\begin{bmatrix} \mathbf{A} & \mathbf{0}_{11\times28} \\ \mathbf{0}_{11\times28} & \mathbf{A} \\ \mathbf{D}_{11} & \mathbf{D}_{12} \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ \mathbf{y} \\ \mathbf{e} \end{bmatrix} = \begin{bmatrix} \mathbf{b} \\ \mathbf{c} \\ \mathbf{0}_3 \end{bmatrix}$$

where $\mathbf{A}$, $\mathbf{D}_{11}$ and $\mathbf{D}_{12}$, $\mathbf{b}$ y $\mathbf{c}$ are the matrices defined next.

$$\mathbf{A} = \left[\begin{array}{c:ccc} \mathbf{1}_2' \otimes \mathbf{I}_3 & \mathbf{I}_3 \otimes \mathbf{d}' & \mathbf{0}_{3\times4} & \mathbf{0}_{3\times12} \\ \mathbf{I}_2 \otimes \mathbf{1}_3' & \mathbf{0}_{2\times6} & \mathbf{I}_2 \otimes \mathbf{d}' & \mathbf{0}_{2\times12} \\ \hdashline \mathbf{I}_6 & \mathbf{0}_{6\times6} & \mathbf{0}_{6\times4} & \mathbf{I}_6 \otimes \mathbf{d}' \end{array}\right], \quad \mathbf{b} = \left[\begin{array}{c} \mathbf{u_X} \\ \mathbf{v_X} \\ \hdashline \mathrm{vec}(\mathbf{X}_0) \end{array}\right], \quad \mathbf{c} = \left[\begin{array}{c} \mathbf{u_Y} \\ \mathbf{v_Y} \\ \hdashline \mathrm{vec}(\mathbf{Y}_0) \end{array}\right],$$

$$\mathbf{D}_{11} = \begin{bmatrix} \mathbf{1}_2' \otimes \mathbf{I}_3 & \mathbf{0}_{3\times22} \end{bmatrix} \quad \text{y} \quad \mathbf{D}_{12} = \begin{bmatrix} -\mathbf{1}_2' \otimes \mathbf{I}_3 & \mathbf{0}_{3\times22} \end{bmatrix}.$$

For brevity it is not possible to fully expand these matrices to show each of its elements, but the reader can verify that the blocks to the left of the dotted line in $\mathbf{A}$ are identical to those on the left hand side of the dotted line of the homologous matrix in example 1, and that the last column of blocks is equal to the areas on the right side of the dotted line. The weighting vector of the target function is $\mathbf{z}' = [\mathbf{0}_6' \ \mathbf{1}_{22}' \ \mathbf{0}_6' \ \mathbf{1}_{22}']$. The solution to the program is as follows.

$$\mathbf{X} = \begin{bmatrix} 5 & 3 \\ 1 & 3 \\ 6 & 1 \end{bmatrix}, \quad \mathbf{Y} = \begin{bmatrix} 1 & 7 \\ 2 & 2 \\ 3 & 4 \end{bmatrix}, \quad \mathbf{u}^*_\mathbf{X} = \mathbf{u}^*_\mathbf{Y} = \begin{bmatrix} 8 \\ 4 \\ 7 \end{bmatrix} \quad \text{y} \quad \begin{bmatrix} \mathbf{v}'_\mathbf{X} \\ \mathbf{v}'_\mathbf{Y} \end{bmatrix} = \begin{bmatrix} 12 & 7 \\ 6 & 13 \end{bmatrix}.$$

□

Example A.3. (Temporal disaggregation) Let us assume that matrix $\mathbf{X}_0$ and the vectors $\mathbf{u}$ and $\mathbf{v}$ from the first example are annual totals, and we are asked to find four quarterly matrices whose margins add up exactly to $\mathbf{u}$ and $\mathbf{v}$. For that purpose we are also given two quarterly series $v_j^{(1)}, \ldots, v_j^{(4)}$ and an additional proxy series whose elements are $x_{31}^{(1)}, \ldots, x_{31}^{(4)}$. With this information we will make two computations, one omitting $\mathbf{X}_0$ and another one exploiting the information provided by this matrix.

$$\begin{bmatrix} \mathbf{v}^{(1)} & \ldots & \mathbf{v}^{(4)} \end{bmatrix} = \begin{bmatrix} 1 & 3 & 5 & 2 \\ 4 & 1 & 2 & 0 \end{bmatrix} \quad \text{y} \quad \begin{bmatrix} x_{31}^{(1)} & \ldots & x_{31}^{(4)} \end{bmatrix} = \begin{bmatrix} 2 & 3 & 3 & 1 \end{bmatrix}.$$

(a) Disaggregation omitting $\mathbf{X}_0$. The constraint system is a simplified version of the expression (12). The blocks $\mathbf{A}_{11}$ have been reduced to $\mathbf{1}_2' \otimes \mathbf{I}_3$ because the sum of the quarterly vectors $\mathbf{v}^{(t)}$ is exactly equal to the annual total $\mathbf{v}$. We have also omitted the part of the system (12) that involves the blocks $\mathbf{A}_{31}$ since we do not have exact information about any of the elements of $\mathbf{X}^{(t)}$, and we represent the blocks $\mathbf{A}_{41}$ as vectors, since in this case $\mathbf{A}_{41} = \mathbf{a}_{41}' = [\mathbf{0}_2', 1, \mathbf{0}_3']$.

$$\left[\begin{array}{ccc:ccc} \mathbf{1}_2' \otimes \mathbf{I}_3 & \ldots & \mathbf{1}_2' \otimes \mathbf{I}_3 & \mathbf{0}_{3\times2} & \ldots & \mathbf{0}_{3\times2} \\ \hdashline \mathbf{I}_2 \otimes \mathbf{1}_3' & \ldots & \mathbf{0}_{2\times6} & \mathbf{0}_{2\times2} & \ldots & \mathbf{0}_{2\times2} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ \mathbf{0}_{2\times6} & \ldots & \mathbf{I}_2 \otimes \mathbf{1}_3' & \mathbf{0}_{2\times2} & \ldots & \mathbf{0}_{2\times2} \\ \hdashline \mathbf{a}_{41}' & \ldots & \mathbf{0}_6' & \mathbf{d}' & \ldots & \mathbf{0}_2' \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ \mathbf{0}_6' & \ldots & \mathbf{a}_{41}' & \mathbf{0}_2' & \ldots & \mathbf{d}' \end{array}\right] \begin{bmatrix} \mathbf{x}_1^{(1)} \\ \vdots \\ \mathbf{x}_1^{(4)} \\ \mathbf{x}_2^{(1)} \\ \vdots \\ \mathbf{x}_2^{(4)} \end{bmatrix} = \begin{bmatrix} \mathbf{u} \\ \mathbf{v}^{(1)} \\ \vdots \\ \mathbf{v}^{(4)} \\ x_{31}^{(1)} \\ \vdots \\ x_{31}^{(4)} \end{bmatrix}$$

The following are the four matrices found. The reader can verify that the columns of the four matrices add up to $\mathbf{u}$ while the rows add up to $\mathbf{u}$.

$$\mathbf{X}^{(1)} = \begin{bmatrix} 0 & 4 \\ 1 & 0 \\ 0 & 0 \end{bmatrix}, \quad \mathbf{X}^{(2)} = \begin{bmatrix} 0 & 0 \\ 0 & 1 \\ 3 & 0 \end{bmatrix}, \quad \mathbf{X}^{(3)} = \begin{bmatrix} 0 & 2 \\ 2 & 0 \\ 3 & 0 \end{bmatrix}, \quad \mathbf{X}^{(4)} = \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 1 & 0 \end{bmatrix}.$$

(b) Disaggregation with $\mathbf{X}_0$. In this case, we add a line of blocks of $\mathbf{I}_6$ and additional discrepancy terms to the previous system, as well as the vector $\mathbf{b}_2 = \text{vec}(\mathbf{X}_0)$ on the right hand side of equality.

$$\left[\begin{array}{ccc:ccc:c} \mathbf{1}_2' \otimes \mathbf{I}_3 & \cdots & \mathbf{1}_2' \otimes \mathbf{I}_3 & \mathbf{0}_{3\times 2} & \cdots & \mathbf{0}_{3\times 2} & \mathbf{0}_{3\times 12} \\ \hdashline \mathbf{I}_2 \otimes \mathbf{1}_3' & \cdots & \mathbf{0}_{2\times 6} & \mathbf{0}_{2\times 2} & \cdots & \mathbf{0}_{2\times 2} & \mathbf{0}_{2\times 12} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathbf{0}_{2\times 6} & \cdots & \mathbf{I}_2 \otimes \mathbf{1}_3' & \mathbf{0}_{2\times 2} & \cdots & \mathbf{0}_{2\times 2} & \mathbf{0}_{2\times 12} \\ \hdashline \mathbf{a}_{41}' & \cdots & \mathbf{0}_6' & \mathbf{d}' & \cdots & \mathbf{0}_2' & \mathbf{0}_{1\times 12} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \\ \mathbf{0}_6' & \cdots & \mathbf{a}_{41}' & \mathbf{0}_2' & \cdots & \mathbf{d}' & \mathbf{0}_{1\times 12} \\ \hdashline \mathbf{I}_6 & \cdots & \mathbf{I}_6 & \mathbf{0}_{6\times 2} & \cdots & \mathbf{0}_{6\times 2} & \mathbf{I}_6 \otimes \mathbf{d}' \end{array}\right] \left[\begin{array}{c} \mathbf{x}_1^{(1)} \\ \vdots \\ \mathbf{x}_1^{(4)} \\ \hdashline \mathbf{x}_{2(1)}^{(1)} \\ \vdots \\ \mathbf{x}_{2(1)}^{(4)} \\ \hdashline \mathbf{x}_{2(2)} \end{array}\right] = \left[\begin{array}{c} \mathbf{u} \\ \hdashline \mathbf{v}^{(1)} \\ \vdots \\ \mathbf{v}^{(4)} \\ \hdashline x_{31}^{(1)} \\ \vdots \\ x_{31}^{(4)} \\ \hdashline \mathbf{b}_2 \end{array}\right]$$

Solving the program under this specification and $\mathbf{z}' = [\mathbf{0}_{24}'\, \mathbf{1}_{20}']$ we get the matrices

$$\mathbf{X}^{(1)} = \begin{bmatrix} 1 & 1 \\ 0 & 2 \\ 0 & 1 \end{bmatrix}, \quad \mathbf{X}^{(2)} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 2 & 0 \end{bmatrix}, \quad \mathbf{X}^{(3)} = \begin{bmatrix} 1 & 2 \\ 1 & 0 \\ 3 & 0 \end{bmatrix}, \quad \mathbf{X}^{(4)} = \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 1 & 0 \end{bmatrix}.$$

□

References

[1] Frank L. (2019). Desagregación temporal de series económicas con programación lineal. Revista Ensayos de Política Econmica. Aceptado, 6 de febrero de 2019.

[2] Frank L. (2019). Interpolating Data with Linear Programming. Statistical Journal of the IAOS. Enviado, 1ero. de octubre de 2018.

[3] Huang W., Kobayashi S. y H. Tanji (2008). Updating an InputOutput Matrix with Sign-Preservation: Some Improved Objective Functions and their Solutions. Economic Systems Research 20: 111-123.

[4] Junius T. y J. Oosterhaven (2003). The solution of updating or regionalizing a matrix with both positive and negative entries. Economic Systems Research 15: 87-96.

[5] Lahr M. L. y L. de Mesnard (2004). Biproportional Techniques in InputOutput Analysis: Table Updating and Structural Analysis. Economic Systems Research 16: 115-134.

[6] United Nations, Statistics Division, Department of Economic and Social Affairs (1999). Handbook of Input-Output Table Compilation and Analysis. Studies in Methods. Series F No. 74. Handbook of National Accounting. New York.

[7] United Nations, Statistics Division, Department of Economic and Social Affairs (2018). Handbook on Supply, Use and InputOutput Tables with Extensions and Applications. Studies in Methods. Series F No. 74, Rev. 1. Handbook of National Accounting. New York.

[8] Williams, H.P. (2013). Model Building in Mathematical Programming. 5th ed. John Wiley & Sons Ltd. West Sussex. 432 p.