From Marginals to Array Structure with the Shuttle Algorithm

Buzzigoli, Lucia and Giusti, Antonio

Dipartimento di Statistica "G. Parenti", Università di Firenze

June 2006

From Marginals to Array Structure with the Shuttle Algorithm

Lucia Buzzigoli and Antonio Giusti

Dipartimento di Statistica "Giuseppe Parenti" - Università di Firenze
Viale Morgagni, 59 - I 50134 Firenze (Italy)
e-mail: lucia@ds.unifi.it and giusti@ds.unifi.it

Abstract. In many statistical problems there is the need to analyze the structure of an unknown *n*-dimensional array given its marginal distributions. The usual method utilized to solve the problem is linear programming, which involves a large amount of computational time when the original array is large. Alternative solutions have been proposed in the literature, especially to find less time consuming algorithms. One of these is the shuttle algorithm introduced by Buzzigoli and Giusti [1] to calculate lower and upper bounds of the elements of an *n*-way array, starting from the complete set of its (*n*-1)-way marginals. The proposed algorithm, very easy to implement with a matrix language, shows interesting properties and possibilities of application. The paper presents the algorithm, analyses its properties and describes its disadvantages. It also suggests possible applications in some statistical fields and, in particular, in Symbolic Data Analysis and, finally, shows the results of some simulations on randomly generated arrays.

Key-words: shuttle algorithm, linear programming, statistical disclosure control, linked tables, zero restrictions.

1 Introduction

Many different kinds of statistical data analyses aim to investigate the structure of an unknown multidimensional array, starting from a limited amount of information about it. In Symbolic Data Analysis (SDA in the following) this issue is connected with the relationship between first order and second order objects (Billard and Diday [2]).
When the dimensions of the multidimensional array are considerable, methods of complex data analysis can be of help in investigating the original data structure.
Traditionally, a first step for solving this problem could be that of calculating the range of the array elements, given the known data. In this context, a useful instrument is linear programming (LP in the following). In fact, LP and integer LP are widely used to solve minimization/maximization problems that satisfy linear constraints. Nevertheless, in statistical applications which involve large datasets the use of LP is often difficult because the computational effort becomes considerable. Secondly,

integer LP doesn't always produce integer results, even if the original data are integer and the problem is surely feasible.
When it is not possible to decompose the initial optimisation problem in smaller pieces, the only possible alternative is the search of computational more efficient algorithms. A number of methods based on networks, graph and combinatorial theory have been proposed in the literature in order to find computational efficient solutions, but a general theory is not available, yet.
The shuttle algorithm (SA in the following), which deals with datasets of non-negative integers, received great attention in this field of studies since the initial proposal by Buzzigoli and Giusti [1]. Although the algorithm doesn't always produce exact LP solutions, its computational advantages are so significant that more deepen analyses are needed, especially from an empirical point of view.
The structure of the paper is the following. In section 2 we introduce the problem, the notation used to represent it and we mention LP method as the 'classic' method to calculate lower and upper bounds of an array. Section 3 presents the SA, its properties and its disadvantages. Section 4 discusses some possible applications of the algorithm to statistical problems. Section 5 shows the results of some simulations in order to investigate its performance in particular situations. Some concluding remarks will end the paper.

2 The starting problem

Let's consider an n-way array of non negative integer values with generic element

$$x_{i_1,i_2,...,i_n}$$

Its marginals represent the projections of the original points on the sub-space defined by the n-m variables (with $m \leq n$) which are not included in the marginal.
Let's indicate
- the (n-1)-dimensional marginals as:

$$x_{+,i_2,...,i_n}, x_{i_1,+,...,i_n}, \ldots, x_{i_1,i_2,...,+}$$

where, for instance

$$x_{+,i_2,i_3,...,i_n} = \sum_{i_1} x_{i_1,i_2,i_3,...,i_n}$$

- the (n-2)-dimensional marginals as:

$$x_{+,+,i_3,...,i_n}, x_{+,i_2,+...,i_n}, \ldots, x_{i_1,+,+,...,i_n}, \ldots, x_{i_1,+,i_3,...,+}, \ldots, x_{i_1,i_2,i_3,...,+,+}$$

where, for instance

$$x_{+,+,i_3,\ldots,i_n} = \sum_{i_1,i_2} x_{i_1,i_2,i_3,\ldots,i_n}$$

- …
- the uni-dimensional marginals as:

$$x_{+,+,+,\ldots,i_n}, x_{i_1,+,+\ldots,+}, \ldots, x_{+,i_2,+,\ldots,+}$$

- and the scalar representing the general total as:

$$x_{+,+,+,\ldots,+}$$

The problem we want to solve is to determine the lower and the upper bound of every element of the original array knowing all its n (n-1)-way marginal arrays.
The solution of the system for a generic $l \times m \times n$ array is usually found with integer linear programming. Given a vector y containing the elements of the n-way array arranged in lexicographical order (clearly the lexicographical order is an extension of the *vec* operator for n-way arrays (n>2)), LP operates on each element of the array separately, minimising and maximising y_k (k=1,2,...,q), subject to a finite number of linear constraints of the form

$$Ay=b$$

and

$$Cy \geq 0$$

where:

- A is a (0-1) $r \times q$ matrix
- r is the total number of elements in the given marginal tables ($l \times m + m \times n + l \times n$);
- q is equal to the product of the array dimensions ($l \times m \times n$);
- y is a q-dimensional non-negative column vector;
- C is the $q \times q$ identity matrix;
- b is an r-dimensional column vector containing the marginal values.

This kind of problem is solved by the simplex algorithm [3], as it is common practice for LP problems.
Nevertheless, using LP and the simplex algorithm in our context is very time consuming, especially when the dimensions of the array increase, since the algorithm must be applied $2 \times q$ times. Actually, the array structure produces some obvious and some less obvious links among the marginals and among the upper/lower bounds and the marginals, that could be exploited to solve the problem. On the contrary, in the LP setting these links are considered only to build the constraints. As a consequence, in this particular field of application, LP and simplex algorithm seem to be particularly burdensome from a computational point of view, because they do not take into account that the $2 \times q$ minimisations/maximisations are linked to each other.
Alternative solutions are therefore necessary, in order to find more computational efficient algorithms. Various proposals have been made in the literature. An interesting method, which can be useful to partly overcome the computational problems of LP, is that relating to the graph theory, which has been extensively used

in statistical disclosure control literature (see §4), especially for applications involving cell suppression in two-dimensional tables ([4]; [5] and [6]). Chowdhury et al. [7] and Roehrig et al. [8] propose a method based on network models that can be applied only to 3-way tables, taking advantages "of the structure that results from the interlocked nature of the three-dimensional disclosure problem" ([8]; sect. 5). Therefore, even if the procedure seems promising, for the time being it can be applied in a limited number of real situations.
Moreover, in three- and higher-dimensional tables the property of integrality of LP doesn't hold anymore. Therefore, starting from integer marginals, LP can result in some non integer upper or lower bounds [9].
Our proposal is based on a completely different and very simple principle, generating an iterative procedure which surely produces integer results.

3 The Shuttle Algorithm

Given the n-way array X and the complete set of the n (n-1)-way marginal distributions:

$$x_{+,i_2,\ldots,i_n}, x_{i_1,+,\ldots,i_n}, \ldots, x_{i_1,i_2,\ldots,+} .$$

the SA is an iterative procedure which finds the lower (or upper) bounds of each cell of X using the marginal distributions and the array formed by the upper (or lower) bounds computed in the previous step.
The logic underlying our procedure is the following:

- the upper bound of the generic element cannot be greater than the lowest difference between each of its marginals and the sum of the lower bounds of the other elements along the same dimension (row, column, etc.); to start the procedure the lower bounds are all set to zero;
- the lower bound of the generic element cannot be less than the highest positive difference between each of its marginals and the sum of the upper bounds of the other elements along the same dimension (row, column, etc.); if no difference is positive the lower bound remains zero;
- if some of the lower bounds are greater than zero the previously computed upper bounds could obviously change; revised upper bounds imply the revision of the previously computed lower bounds and so on.

In order to formalize this logic, we consider two n-way arrays X^{Ui} and X^{Li}, which are the arrays containing, respectively, the upper and the lower bounds of the elements of X computed at the i-th iteration. Their generic elements are:

$$x^{U_i}_{i_1,i_2,\ldots i_n} \quad \text{and} \quad x^{L_i}_{i_1,i_2,\ldots i_n} .$$

The algorithm is based on two steps which are repeated alternatively: each step uses the matrix X and either of the matrices X^{Li} or X^{Ui}, to calculate, respectively, the upper or the lower bound of each element of the original array. The lower bounds can be

used to revise upper bounds (repetition of first step) and the revised upper bounds to recalculate new lower bounds (repetition of the second step), and so on.
We named it "shuttle" because it alternatively jumps from one array to the other calculating X^{Ui} at every odd step and X^{Li} at every even step.
In summary, the algorithm can be described as follows.
1st step:

$$x^{U_s}_{i_1,i_2,\ldots i_n} = \min(x^{U_{s-1}}_{i,i_2,\ldots,i_n}, x_{+,i_2,\ldots,i_n} - \sum_{i\neq i_1} x^{L_{s-1}}_{i,i_2,\ldots i_n}, x_{i_1,+,\ldots,i_n} - \sum_{i\neq i_2} x^{L_{s-1}}_{i_1,i,\ldots i_n}, \ldots$$
$$x_{i_1,i_2,\ldots,+} - \sum_{i\neq i_n} x^{L_{s-1}}_{i_1,i_2,\ldots i})$$

2nd step:

$$x^{L_s}_{i_1,i_2,\ldots i_n} = \max(x^{L_{s-1}}_{i_1,i_2,\ldots i_n}, x_{+,i_2,\ldots,i_n} - \sum_{i\neq i_1} x^{U_s}_{i,i_2,\ldots i_n}, x_{i_1,+,\ldots,i_n} - \sum_{i\neq i_2} x^{U_s}_{i_1,i,\ldots i_n}, \ldots$$
$$\ldots, x_{i_1,i_2,\ldots,+} - \sum_{i\neq i_n} x^{U_s}_{i_1,i_2,\ldots i})\ .$$

where $x^{L_0}_{i_1,i_2,\ldots,i_n} = 0$ and $x^{U_0}_{i_1,i_2,\ldots,i_n} = x_{++\ldots+} = N$.

It obviously stops when $X^{U_s} = X^{U_{s-1}}$ or $X^{L_s} = X^{L_{s-1}}$.
Differently from LP, it is obvious that the SA always produces integer bounds. It can be shown that it always converges in a finite number of steps and that, for specific cases of relevance, the dimensions of the original array bound the number of steps needed to reach a stop (Buzzigoli and Giusti [10] and [11]): for an $m\times n$ array the SA needs at most two steps to get to the solution; for 3-way arrays the algorithm surely reaches the stop after 3 steps in the $2\times2\times2$ case only; moreover, in this case the SA finds the LP solution. For a general $l\times m\times n$ array this is not the case.
Therefore, we proved that shuttle solutions surely coincide with LP solutions only for particular cases, but Roehrig [12] showed that the SA is related to the dual of the LP approach. Therefore, it seems that the methods may have some general connections although they solve the problem of finding bounds with very different foundations.
Dobra and Fienberg [13] and Dobra [14] have proposed an extension of the SA (called generalized shuttle algorithm), but the bounds are sharp only for specific kinds of arrays and to generalize the procedure the authors still need to apply a variant of LP. A final solution is therefore still not available. The same conclusion is given in a recent paper dealing with the approximation of exact conditional inference on contingency tables (Chen et al. [15]), where the authors apply both LP and the SA on some real multiway tables to compare their performances.
Another shortcoming of SA has been pointed out by Cox [16], who shows with some examples that the SA is insensitive to whether a feasible table exists, once a set of

marginals is given: actually this is not a real problem, because, as we will se in the next paragraph, the main field of application of the algorithm is in situations where the original *n*-way array is available and the marginals are derived from it; so they are surely compatible with a table.
The main advantages of the SA are the computational ones. First of all, it is very easy to implement with a matrix language that allows the definition of *n*-way arrays, even if the use of a traditional compiled programming language is more effective. Secondly, it is very fast (see §5). Thirdly, it has limited storage and memory requirements: the storage occupation for data, results and working data, is mostly represented by the marginal distributions (given data) and by *n*+3 arrays of the same dimensions of the unknown array (where *n* is the number of the dimensions of the original array, two arrays represent the results, the other *n*+1 arrays are working data). Finally, from a statistical point of view, as we will see in the next paragraph, the algorithm has interesting links with Fréchet bounds, Bonferroni bounds and log-linear models as well (for an interesting review on these topics, see [17]).

4 Some links to statistical aspects

The SA is a computational tool that can be useful in several fields of statistical theory and applications. In this section we mention four of them: statistical disclosure control, database management, probability theory, symbolic data analysis.
Its main advantages are in practical situations, where the LP approach is computationally too burdensome, but it has also interesting theoretic properties, as we will see later.
Originally, the algorithm was proposed by Buzzigoli and Giusti [1] as an alternative solution to LP in statistical disclosure control (SDC).
It is well known that National Statistical Institutes are required by law to protect the confidentiality of the individual information they collect by means of surveys or censuses: this means that the data which are released to the public must be compiled in such a way that the identification of respondents is no longer possible. Data are protected by means of particular statistical and mathematical procedures, which are traditionally labelled as SDC methods. Statistical agencies usually release cross-classified tables and microdata files. Although the difference between microdata and tabular data is not so great as it seems (individual files can be seen as a huge table deriving from *n* crossings, where *n* is the number of variables surveyed on each individual), traditionally the treatment of SDC methods for these two data products is different. During the past ten years the research on these topics has been extensive: a relevant literature was produced by official statisticians and academic researchers. Interesting reviews on this subject can be found in [18], [19] and [20]. More recent contributions are contained in [21].
In our work we focussed on a particular aspect of SDC for frequency count tabular data: tables containing cells with low frequencies are generally considered at risk, because individuals with rare characteristics are usually more recognisable than others. Therefore, most SDC methods for tables of counts have the purpose to avoid the release of tables with low frequency cells ('sensitive' cells).

All the tables released by a statistical agency can be seen as a set of sub-tables deriving from an original complete table. These tables are obviously linked - and therefore called 'linked tables' [22] - among each other, because they represent different aggregations of the same data, and they often share some variables. If the original complete table contains sensitive cells, the agencies have to find out how much information on the original table can be derived from the partial tables before publishing them.

Thanks to LP techniques the lower and upper bound of the original frequency of each cell can be found. The original value of a cell can be recalculated exactly when the lower and the upper bound for that cell are the same.

The problem of disclosure auditing in a set of tables linked over some variables is not yet completely solved. The various proposals in the literature are not satisfactory, as they usually refer to limited dimension cases, which are not of great interest for official producers of statistics, or to specific tabular structures. On one hand LP may be computationally burdensome when the dimensions of the released tables become large. On the other, the National Statistical Institutes work with large tables and arrays: efficient heuristics are therefore needed, which can determine the lower and upper bounds of the unknown original array starting from the released tables within a reasonable time and with a reasonable computational effort. The SA can be one of these heuristics. For instance, it has been adopted in the Neighborhood Statistics project of the Office of National Statistics (UK) to audit random rounding procedures (Armitage et al. [23]).

Another field of application, which is connected to the previous one, is the auditing of web-based query systems for statistical databases. If an original n-dimensional table is made available on web for the users, who can make queries to get only its margins, each time the generic user makes a new query the system must examine the query combining the new requested table with the previously released tables. The new table can be released only if the risk of disclosure is reasonably low. To audit dynamically the sequence of queries computationally efficient algorithms are particularly necessary, also because statistical databases are often very large and in this case the response time becomes critical. The generalized version of SA is implemented in a table server used by the National Institute of Statistical Science to check tabular summaries of confidential microdata (Dobra et al. [24], Karr et al. [25]).

In other fields of statistical application we face the problem of finding the probability distribution corresponding to some given marginal distributions (Frechet [26]). In the last century the Italian school gave interesting contributions on this subject (Bonferroni [27], Rizzi [28], Dall'Aglio [29] and [30]) and, more recently, the literature on probability theory produced new results regarding the case of multidimensional marginals (Dall'Aglio [31], [32] and [33]). Nevertheless, we would like to stress that many of these contributions face the problem of the existence of the multi-dimensional probability distribution, given a certain number of marginal distributions, while in our context this is not a problem: the various marginals are surely compatible with an n-dimensional array, because they are obtained from it collapsing the various dimensions. In this sense our problem is simpler and focuses mainly on computational issues.

Fienberg [17] proposes an original statistical reading of linked tables disclosure problems, presenting m-dimensional Fréchet and Bonferroni bounds for k-way tables

(recently analyzed by Rüschendorf [34] and Galambos and Simonelli [35]). The author reinterprets the Fréchet and Bonferroni equations in terms of counts instead of probabilities, and shows some interesting results that are connected to our analysis.
In particular, he uses a 'combined' solution mixing Fréchet and Bonferroni equations. In the case of a 3-way table, given all the 2-way marginals, this combination leads to the following solution:

$$x_{ijk}^{U} = \min(x_{ij+}, x_{i+k}, x_{+jk}, x_{ijk} + \bar{x}_{ijk})$$

$$x_{ijk}^{L} = \max(x_{+++} - S_{1[ijk]} + S_{2[ijk]} - \min(\bar{x}_{ij+}, \bar{x}_{i+k}, \bar{x}_{+jk}), 0)$$

where:

$$S_{1[ijk]} = x_{i++} + x_{+j+} + x_{++k} \ ,$$

$$S_{2[ijk]} = x_{ij+} + x_{i+k} + x_{+jk} \ ,$$

$\bar{x}_{ijk}$ is the diagonally complementary count opposite x_{ijk} in the 2^3 table formed by collapsing all the remaining categories for each of the three variables in the table into a single complementary category, $\bar{x}_{ij+}, \bar{x}_{i+k}, \bar{x}_{+jk}$ are the 2-way marginals of the diagonal complementary element.
It is very easy to find the same result with the SA (Buzzigoli and Giusti [10]; sect. 4.4) and it can be shown that applying it to collapsed tables of higher dimensions it is possible to reproduce the alternating sequence of signs implied by the Bonferroni inequalities (Fienberg [17]; sect.5). Therefore the SA gets a simple iterative procedure to find the bounds, and can also be seen as a useful tool to implement the automated calculus of the combined Fréchet-Bonferroni bounds in a rapid and efficient way.
Another interesting field of analysis is the link between contingency tables with fixed marginals and the theory of log-linear models [36]. Dobra [37] using the graphical version of log-linear models exploits this connection to take advantage of cell relationships inherent the tabular structure and to generalize the SA.
Finally, the algorithm has promising links with symbolic data analysis. It is well known that the input of SDA is a symbolic data table: according to Diday's definition (Diday [38]) the columns of this table are "symbolic variables" while the rows are "symbolic descriptions" of second order objects.
In this context the unknown *n*-way array of counts *X* can be easily interpreted as the original data matrix containing a row of information for each statistical unit (first order objects) and a column for each surveyed variable, while the marginals of the array *X* can be seen as different symbolic tables (second order objects): in each table the columns are the variables included in the corresponding marginal and the rows are descriptions of classes of individuals made of conditional distributions.
The SA often gets a partial information on the unknown content of the array cells. In such a situation for some cells we get an interval like

$$[\, x^{U_i}_{i_1,i_2,\ldots i_n} , x^{L_i}_{i_1,i_2,\ldots i_n} \,]$$

which includes the real count for the cell.
The SA output is a number of first order objects which represent an approximation of the original data. This number is the amount of the statistical units (extent) which satisfy the characteristic properties (intent) specified by particular combination of modalities corresponding to the cell.
This aspect deserves further research, because it opens new possible developments in data preparation activities for SDA.

5 Some simulation results

Some simulations were carried out to test the performance of the SA from a computational point of view (Buzzigoli and Giusti [39] and [40]). Here we update the oldest experiments and summarize the results.
An interesting problem is the application of the SA in monitoring cell suppressions for SDC purposes [19]. Cell suppression is one of the most popular methods to avoid disclosure of confidential data: the cells of the table which are at risk of identification are obscured together with all the cells that can be used to re-calculate their values.
In this case the original *n*-way array is partly known and the algorithm can be used to check whether it is possible to identify the suppressed cells using the known cells and the marginals of the table.
This situation is known in SDC literature as "zero-restrictions" [41], because when some entries in the table are given, we can subtract their values from the corresponding marginals, fixing the original values to zero. For instance, if we suppose that the element x_{i_1,i_2,i_3} of a three-dimensional table is known, the shuttle can be applied to three sets of marginals (layer totals, row totals and column totals) where x_{+,i_2,i_3} is replaced by $x_{+,i_2,i_3} - x_{i_1,i_2,i_3}$, $x_{i_1,+,i_3}$ by $x_{i_1,+,i_3} - x_{i_1,i_2,i_3}$ and $x_{i_1,i_2,+}$ by $x_{i_1,i_2,+} - x_{i_1,i_2,i_3}$; therefore x_{i_1,i_2,i_3} is set to zero (i.e. $x^{U}_{i_1,i_2,i_3} = x^{L}_{i_1,i_2,i_3}$ =0).
The same condition arises when one of the marginals is zero, because in count tables this means that all the cells contributing to that marginal are identically zero.
In presence of zero restrictions, the SA is not always able to find LP solutions. This is obvious, because the sequence of iterations cannot detect the particular links among the various dimensions of the array originated by the null entries. Some discussion on this aspect can be found in [37].
Nevertheless, it could be interesting to make some simulations to find out the percentage of success of the SA in comparison to the simplex method in presence and in absence of zero-restrictions and to compare the computational time between the algorithms.
For all the simulations we used an APL2 interpreter running on a personal computer powered by a 3.21Ghz Pentium IV with 2Gb of central memory. We used the APL2

language, despite the lower performances of an interpreter with respect to a compiler, because the translation of the algorithm in this kind of "matrix" language is easier and the resulting functions are more general and flexible. As is well known, APL2 permits the utilisation of nested generalised arrays; this fact contributes to reduce the use of iterative structures inside the code and allows for the reduction of the execution time.
Table 1 presents the results of the application of LP and SA on four groups of three-dimensional 2×4×4 simulated arrays. Each group is composed by 21,000 arrays with different value ranges: 0-1 and 0-2 (zero-restrictions), 1-2 and 1-3.
The column labelled "# of errors" shows how many times SA and LP results differ. In other words, it quantifies the cases where SA doesn't find the sharpest bounds.

Table 1. Simulation results on groups of 21,000 2×4×4 arrays

Arrays	# of errors	% of errors	LP/SA comp. time
0-1 arrays	208	0.99	173,577
0-2 arrays	550	2.62	162,478
1-2 arrays	0	0.00	263,450
1-3 arrays	0	0.00	288,873

In 0-1 arrays the percentage of 'errors' is less than 1% and the LP computational time is almost 174,000 times the SA computational time; in 0-2 arrays the percentage of 'errors' is more than doubled (around 2.6%). In 1-2 and 1-3 arrays there are no 'errors': it means that there are no differences between LP and SA results. The ratio between LP and SA computational time raises to over 263,000.
Although these results are surely influenced by the fact that the APL2 functions we used for LP applications are not optimised, the difference is so great that it cannot be only due to this.
Starting from this last observation, another issue worth exploring relates to the comparison of the SA performance in solving arrays of different structure. To this end we ran a second set of simulations on randomly generated arrays in order to assess the computing time and the stability of the algorithm across the experiments.
The simulation was conducted again on three dimensional arrays with two layers: the various experiments differ by the number of rows and columns (3 sets: 2×4×4, 2×5×5, 2×6×6) and by the range of values of cell counts (4 sets: 0-1, 0-10, 0-100, 0-1000).
The results are in Table 2.
In the first part of the table we have the results for 2×4×4 arrays: each row identifies the range of the values contained in the arrays. For each different kind of array structure 100 groups of 10000 arrays were simulated and the SA computing time was calculated; finally, the average (Mean) and the coefficient of variation (CV) of the computing time were estimated over each sample of 100 experiments and reported in the table.
Analogously for 2×5×5 and 2×6×6 arrays. Therefore we simulated 12,000,000 arrays in all.

Table 2. Simulation results on 2×4×4, 2×5×5 and 2×6×6 arrays: mean and coefficient of variation of computing time (in milliseconds) for each group of 100 experiments with 10,000 arrays.

2×4×4 arrays			
Value limits	Mean	CV	% change in comp.time
0-1	2895.11	0.0078	…
0-10	2407.17	0.0073	- 16.9
0-100	2328.94	0.0060	- 3.2
0-1000	2322.80	0.0041	- 0.3

2×5×5 arrays			
Value limits	Mean	CV	% change in comp.time
0-1	2898.71	0.0105	…
0-10	2322.51	0.0109	- 19.9
0-100	2225.00	0.0104	- 4.2
0-1000	2210.80	0.0092	- 0.6

2×6×6 arrays			
Value limits	Mean	CV	% change in comp.time
0-1	2876.57	0.0037	…
0-10	2264.53	0.0101	- 21.3
0-100	2184.69	0.0109	-3.5
0-1000	2173.44	0.0104	- 0.5

Comparing the results for the arrays of the same size we see that the mean computing time decreases as the values range increases. Therefore, in this experiment the mean is inversely related to the width of the values range; the CV stresses how little the variation in mean computing time is, signalizing the stability of the algorithm. Comparing the results for the same values range we see that the mean computing time decreases as the size of the arrays increases.

In conclusion, the table shows that, for these dimensions, the SA converges faster and faster as the structure of the arrays becomes more 'complex', in terms of dimensions or of values range. This derives from the two steps which constitute the algorithm: as the number of layers/rows/columns increases, the probability that the first term in the min/max expressions will be enhanced decreases. This is clearly a remarkable characteristic of the algorithm, especially when we compare it with the simplex method in LP: when we apply the simplex method the computational effort increases with the number of layers/rows/columns of the table because the constraints that must be satisfied in each minimization/maximization step increase as well.

Another interesting result is reported in the last column of Tab.2, which contains the percentage change in computing time when we increase the values range, given the array structure. For instance, in the 2×4×4 section: ((2407.17/2895.11)-1)x100=-16.9, ((2328.94/2407.17)-1)x100=-3.2, etc.. The decrease in computing time when we pass from 0-1 arrays to 0-10 arrays is evident, whatever the structure of the array is (from 16.9% to 21.3%), while the decrease is much smaller passing from 0-10 to 0-100

(from a minimum of 3.2 to a maximum of 4.2) and from 0-100 to 0-1000 (from a minimum of 0.3 to a maximum of 0.6). Moreover, the percentage decrease is quite stable for all the array structures considered in the experiment..
These results confirm that the 0-1 case is the most problematic for the SA, due to the high probability to have zero-restrictions.
Finally, we would like to stress again that storage and memory requirements of the SA remain very limited.

6 Concluding remarks

The SA seems to have interesting properties that could be exploited in various fields of statistical application that involve the computation of lower and upper bounds of an array of integer values given its marginals. Various problems, currently solved with different methodologies, could be brought back to this algebraic definition.
The algorithm is very easy to implement on a personal computer, has a low requirement of memory space and gives always integer results. It is very efficient and in our experiment its speed increases with the increasing of array dimensions and of values range.
The main advantage of the algorithm is in practical situations, especially in what the disclosure control of linked tables is concerned. Such a problem is felt at National Statistical Institutes in a particular way since the institutional duties related to disclosure may find it computationally burdensome to use a LP approach when released tables are large. In this respect the SA with its limited computing time and convergence properties appears to be a promising solution.
Moreover, the SA shows also interesting theoretical developments, because it seems to have relevant links with some probabilistic and statistical issues, such as statistical disclosure control, log-linear models, graph theory, database query auditing, etc..
Finally, the behavior of the algorithm in simulated conditions shows that its links with LP still need to be explored, in order to exploit its computational properties at the most.

References

1. Buzzigoli, L. and Giusti, A.: An Algorithm to Calculate the Lower and Upper Bounds of the Elements of an Array Given its Marginals. Working Paper 70. Dipartimento di Statistica "Giuseppe Parenti", Firenze (1996)
2. Billard, L. and Diday, E.: From the Statstics of Data to the Statistics of nowledhe: Symbolic Data Analysis. Journal of the American Statistical Association, 98, 462, 470-487 (2003)
3. Brickman, L.: Mathematical Introduction to Linear Programming and Game Theory. Springer-Verlag, New York (1989)
4. de Carvalho, F.D., Deallert, N.P., and de Sanches Osorio, M.: Statistical Disclosure in Two-Dimensional Tables: General Tables. Journal of the American Statistical Association, 89, 1547-1557 (1994)
5. Cox, L.H.: Suppression Methodology and Statistical Disclosure Control. Journal of the American Statistical Association, 75, 377-385 (1980)

6. Cox, L.H.: Network Models for Complementary Cell Suppression. Journal of the American Statistical Association, 90, 1453-1462 (1995)
7. Chowdhury, S.D., Duncan, G.T., Krishnan, R., Roehrig, S.F., and Mukherjee, S.: Disclosure Detection in Multivariate Categorical Databases: Auditing Confidentiality Protection Through Two New Matrix Operators. Management Science, 45, 1710-23 (1999)
8. Roehrig., S.F., Padman, R., Duncan, G.T. and Krishnan, R.: Disclosure Detection in Multiple Linked Categorical Datafiles: A Unified Network Approach. In: Statistical Data Protection - Proceedings of the conference, 131-147. Eurostat, Luxembourg (1998)
9. Duncan, G.T., Fienberg, S.E., Krishnan, R., Padman, R. and Roehrig, S.F.: Disclosure Limitation Methods and Information Loss for Tabular Data. In: Doyle et al. (eds.). Confidentiality, Disclosure, and Data Access: Theory and Practical Applications for Statistical Agencies. Elsevier Science, Amsterdam (2001)
10. Buzzigoli, L. and Giusti, A.: Statistical disclosure control problems for linked tables. Italian Journal of Applied Statistics, 10, 443-458 (1998)
11. Buzzigoli, L. and Giusti, A.: An algorithm to calculate the lower and upper bounds of the elements of an array given its marginals. In: Statistical Data Protection - Proceedings of the Conference, 131-147. Eurostat, Luxembourg (1999)
12. Roehrig, S. F.: Auditing Disclosure in Multi-Way Tables With Cell Suppression: Simplex and Shuttle Solutions. Paper presented at: Joint Statistical Meeting 1999, August 5-12, Baltimore (1999)
13. Dobra, A., and Fienberg, S.E.: Bounds for cell entries in Contingency Tables Given Marginal Totals and Decomposable Graphs. Proceedings of the National Academy of Sciences, 97, 11185-92 (2000)
14. Dobra, A.: Computing Sharp Integer Bounds for Entries in Contingency Tables Given a Set of Fixed Marginals. Technical Report, Department of Statistics. Carnegie Mellon University, Pittsburgh (2001)
15. Chen, Y., Dinwoodie, I. H., and Sullivant, S.: Sequential Importance Sampling for Multiway Tables. The Annals of Statistics, **34**, 1, *in press* (2006)
16. Cox, L.H.: Bounds on Entries in 3-Dimensional Contingency Tables Subject to Given Marginal Totals. In: Domingo-Ferrer, J. (Ed.). Inference Control in Statistical Databases : From Theory to Practice. Springer-Verlag, Heidelberg (2002)
17. Fienberg, S.E.: Fréchet and Bonferroni Bounds for Multi-way Tables of Counts With Applications to Disclosure Limitation. In: Statistical Data Protection - Proceedings of the conference, 115-129. Luxembourg: Eurostat (1999)
18. Federal Committee On Statistical Methodology: Report on Statistical Disclosure Limitation Methodology. Statistical Policy Working Paper 22, Statistical Policy Office, Office of Information and Regulatory Affairs. Office of Management and Budget, Washington D.C. (1994)
19. Willenborg, L.C.R.J. and de Waal, A.G.: Statistical Disclosure Control in Practice. Lecture Notes in Statistics. Springer-Verlag, New York (1996)
20. Domingo-Ferrer, J. (ed.): Proceedings of the SDP'98. IOS Press, Luxembourg (1998)
21. Doyle, P., Lane, J.I., Theeuwes, J.J.M. and Zayatz, L.V. (eds.): Confidentiality, Disclosure, and Data Access: Theory and Practical Applications for Statistical Agencies. Elsevier Science, Amsterdam (2001)
22. de Vries, R.E.: Disclosure Control of Tabular Data Using Subtables. Netherlands Central Bureau of Statistics, Voorburg (1993)
23. Armitage, P., Merret, K., Lyons, A. and Tame, E.: Neighbourhood Statistics in England and Wales: Disclosure Control Problems and Solutions. In: Work session on Statistical data confidentiality. Part 2. Monographs of Official Statistics. Eurostat, Luxembourg (2004)
24. Dobra, A., Karr A.F. and Sanil A.P.: Preserving Confidentiality of High-dimensional Tabulated Data: Statistical and Computational Issues. Technical Report no.130, National Institute of Statistical Science (2002)

25. Karr, A.F., Dobra A. and Sanil A.P.: Table Server Protect Confidentiality in Tabular Data Releases. Communications of the ACM, 46, 1 (2003)
26. Frechet, M.: Sur les tableau de corrélation dont le marge sont données. Ann. Univ. Lyons Sect. A, Ser. 3, 14, 53-77 (1951)
27. Bonferroni, C.E.: Teoria statistica delle classi e calcolo delle probabilità. Pubblicazioni del R. Istituto Superiore di Scienze Economiche e Commerciali di Firenze, 8, 1-62 (1936)
28. Rizzi, A.: Osservazioni sulle classi di Fréchet delle funzioni di ripartizione a più variabili. Bollettino Unione Matematica Italiana, 12, 269-277 (1957)
29. Dall'Aglio, G.: Sulle distribuzioni doppie con margini assegnati soggette a delle limitazioni. Giornale dell'Istituto Italiano degli Attuari, XXIII-XXIV, 94-105 (1960)
30. Dall'Aglio, G.: Les fonctions extremes de la classe de Fréchet a 3 dimensions. Pubblic. de l'Institute de Statistique de l'Université de Paris, 9, 175-188 (1961)
31. Dall'Aglio, G., Kotz, S. and Salinetti, G.: Advances in Probability Distributions with Given Marginals. Kluwer Academic Publishers, Dordrecht (1990)
32. Rüschendorf, L.: Bounds for Distributions With Multivariate Marginals, Stochastic Orders and Decision under Risk, IMS Lecture Notes - Monograph Series, 19, 285-310 (1991)
33. Genest, C., Quesada Molina, J.J. and Rodriguez Lallena, J.A.: De l'impossibilité de construire des lois à marges multidimensionnelles données à partir de copules. C.R. Acad. Sci. Paris, t. 320, Série I, 723-726 (1995)
34. Rüschendorf, L.: Developments on Fréchet Bounds. In L. Rüschendorf, B. Schweizer and M.D. Taylor (Eds.): Distributions with Fixed Marginals and Related Topics. IMS Lecture Notes - Monograph Series, 28, 273-296 (1996)
35. Galambos, J. and Simonelli, I.: Bonferroni-type Inequalities with Applications. Springer-Verlag, New York (1996)
36. Fienberg, S.E. and Makov, U.E.: Confidentiality Uniqueness and Disclosure Limitation for Categorical Data. Journal of Official Statistics, 14, 385-397 (1998)
37. Dobra, A.: Statistical Tools for Disclosure Limitation in Multi-Way Contingency Tables. PhD Thesis. Carnegie Mellon University, Pittsburgh (2002)
38. Diday, E.: An Introduction to Symbolic Data Analysis and the Sodas Software. The Electronic Journal of Symbolic Data Analysis, 0, 0 (2002)
39. Buzzigoli, L. and Giusti, A.: Disclosure Control On Multi-Way Tables By Means Of The Shuttle Algorithm: Extensions And Experiments. In: Compstat 2000 – Proceedings in Computational Statistics, 229-234. Phisica Verlag, Heidelberg (2000)
40. Buzzigoli, L. and Giusti, A.: Shuttle algorithm and Simplex Method: Some experiments. Poster presented at COMPSTAT 2004, August 23-27, Prague (2004)
41. Cox, L.H.: Some remarks on research directions in statistical data protection. In: Statistical data protection - Proceedings of the Conference, 163-176. Eurostat, Luxembourg (1999)