Proceedings of the Annual Meeting of the American Statistical Association, August 5-9, 2001

USING LINEAR PROGRAMMING FOR CELL SUPPRESSION IN STATISTICAL TABLES: THEORY AND PRACTICE

Paul B. Massell, U.S. Census Bureau
SRD, Room 3209-4, U.S. Census Bureau, Washington, D.C. 20233

Keywords: disclosure limitation; confidentiality, cell suppression; linear programming

Abstract[1]. A common disclosure limitation method for tabular data is cell suppression. The cell suppression method for statistical tables with marginals involves two steps : (1) determining which cells are sensitive and must therefore be suppressed (i.e., not published) (2) determining which additional cells should be suppressed so that a data intruder, despite knowing the additive relationships of the tables, will not be able to estimate the sensitive cells too precisely. It has been known for more than twenty years that the second step of the cell suppression method, usually called the secondary or complementary cell suppression problem, can be formulated as a linear programming (LP) problem. However, until recently it did not appear to be a practical alternative to programs based on network flow or other algorithms. It now appears that due to improvements in LP solvers and in computer speed, suppression programs based on the LP formulation can be used for sets of large 3D linked tables. We review the LP formulation and then present some timing results.

1. Introduction

A commonly used disclosure limitation method for tabular data is cell suppression (ref: WIL). The cell suppression method for statistical tables with marginals involves two steps: (1) determining which cells are sensitive and must therefore be suppressed (i.e., not published) (2) determining which additional cells should be suppressed so that a data intruder, despite knowing the additive relationships of the tables, will not be able to estimate the sensitive cells too precisely. It has been known for more than twenty years that the second step of the cell suppression method, usually called the secondary or complementary cell suppression problem, can be formulated as a linear programming (LP) problem (ref: ROB, SAN). LP models used for cell suppression have desirable properties not possessed by network flow models. Tables of all dimensions and structures can be viewed as LP models. Thus, for all tables, the LP approach will identify a suppression pattern that provides at least the desired amount of protection for each primary cell (ref: COX). Network flow methods are designed to work on networks and a 2D table with a hierarchy in at most one dimension may be modeled as a network. However, when network flow methods are applied to non-network LP problems (not surprisingly) they may not work well. Since tables of dimension three and higher cannot, in general, be modeled as networks, applying network flow methods to them, will often lead to a poor suppression pattern (either undersuppression or oversuppression). It is worth noting here that network models may be viewed as a special type of LP model (see discussion below).

The main reason for continuing to use network flow models in the computational module of suppression programs has been the short run times of suppression programs based on these models. Until recently, for most realistic sets of large tables (linked or not) from a given survey, the network flow models were fast enough to allow disclosure analysis to proceed in a reasonable time whereas the LP based programs were not. (When applied to a pure network problem, the CPLEX network optimizer may run 100 times faster than the CPLEX simplex optimizer (ref: CPX, p. 154). Results from a simplified version of the suppression program that used CPLEX as an LP solver (written by Jim Fagan of the Census Bureau) indicated a significant reduction in running times from earlier LP based programs using a slower LP solver and run on slower machines. We modified an earlier LP based subroutine (written by Fagan and Laura Zayatz) that used this slower LP solver. At the U.S. Census Bureau, we are now finding that due to use of faster computers and a faster, more flexible LP package, the LP based programs are likely to be used for a much wider class of suppression tasks. In addition to the above improvements, which might be called "system improvements", slight reductions in run times have been achieved from restructuring the code so that some part of the solution from one primary can be used for the next primary being protected.

[1] This paper reports the results of research and analysis undertaken by Census Bureau staff. It has undergone a Census Bureau review more limited in scope than that given to official Census Bureau publications. This report is released to inform interested parties of ongoing research and to encourage discussion of work in progress.

2. The LP formulation of the cell suppression problem.

Basic terminology for statistical tables.

Let us define some terms that are useful when discussing statistical tables with totals included. Assume that it makes sense to add the cell values. Consider the set of cells that exist before marginals are constructed. We call these cells the interior cells. The marginal row is the row of totals formed by summing all the rows of interior cells. Similarly for the marginal column and higher dimensional marginals. The marginal cells, i.e., the cells that appear in any of the marginals, are called, in contrast, the exterior cells.

The LP formulation of the cell suppression problem requires that we express the marginal relationships in terms of linear equations. This is best described with an example. Suppose we have a 2 row by 3 column table of interior cells which becomes a 3 row by 4 column table when marginals are included. Let x(i,j) denote the value of the flow in cell (i,j) for some protection pattern being evaluated during the optimization procedure. The flow values are subject to the same constraints as the original cell values. Therefore we have:

x(1,1) + x(1,2) + x(1,3) = x(1,4) for the first row; similarly for the 2nd and 3rd rows.

We also will have :

x(1,1) + x(2,1) = x(3,1) for the first column; similarly for columns 2 through 4.

Note that the constraint for the (marginal) 3rd row and the (marginal) 4th column, given the other constraints are redundant. They both express (indirectly) that the grand total cell, x(3,4), is the sum of all the interior cells. Fortunately, most modern LP packages can quickly remove such redundancies from the full list of linear constraints that are supplied to it.

Let us introduce the notion of a "shaft" and relate it to the marginals of the 2D table above and higher dimensional tables. Each linear equation that is used to describe the marginals of a table is called a shaft. In the above 2D table, we have (2 + 1) + (3+1) = 7 shafts. Let us define a dimension 'i' shaft (denoted dim-i) as a marginal relation one gets by fixing a value for each dimension in the table <u>other</u> <u>than</u> i; this makes a shaft a one-dimensional object. For example, consider a table that, with marginals, has size 4 x 5 x 3. Then, if we fix the column as '3' and the level as '2', the corresponding dim-1 shaft, denoted (sum, 3, 2), expresses the relation: (1,3,2) + (2,3,2) + (3,3,2) = (4,3,2) . Thus there are 5 x 3 = 15 dim-1 shafts, 4 x 3 = 12 dim-2 shafts, and 4 x 5 = 20 dim-3 shafts. This yields 47 total shafts for this table; this includes some redundancies.

The set of all these shafts constitutes the set of linear equation constraints for the LP problem we are formulating. There is another set of constraints that are expressed as inequalities; viz., the variables in cell suppression problems typically are required to be non-negative; we'll call this the positivity assumption (for variables). In addition, there are upper bounds for each cell variable.

3. Informational description of suppression problem

It is useful to interpret the constraints using an informal notion of information. The constraints contain enough information to allow reconstruction of a limited amount of suppressed information of either interior or exterior cells. For example, if exactly one cell is suppressed in any shaft then clearly the suppressed cell value can be solved for (i.e., reconstructed or determined) exactly. This does not require the positivity assumption. On the other hand, if two or more cells in each shaft are suppressed, often the suppressed cells (values) cannot be determined exactly. However an estimate can, in general, be constructed using the positivity assumption. Clearly the marginals associated with a given cell provide a set of upper bounds for the cell value because of the positivity assumption. The cell suppression problem may be viewed as purposeful and controlled reduction of tabular information with the goal of achieving a very specific degree of fuzziness (i.e., ambiguity) about the values of each of the suppressed cells.

When we collect data for later publication as a table, it is generally the case that the value of one cell places no restriction on the value of another cell. However, once we have all the data for the interior cells of the table and begin to calculate the marginals (i.e., exterior) cells based on the interior cells, the cells values are linked (a physicist might say "coupled"). Suppression may be viewed as a weakening of that linkage (or coupling).

4. The Network Flow Model viewed as a special case of the LP model

A network flow model may be expressed as an LP model with a special structure. A network flow model, when expressed as an LP model, must have a constraint matrix in which all the coefficients have the value 0,1, or -1. Recall that the rows of a constraint matrix express the constraints, the columns represent the variables. In addition each variable must appear in at most two rows (i.e., constraints) with at most one coefficient of +1 and one coefficient of -1. It is easy to see that the constraint matrix for a simple 2D table can be written in this form. This is because in a simple 2D table the two variables for a given cell (viz. the positive flow variable and the negative flow variable appear only once in a row shaft (constraint) and only once in a column shaft. These two constraints can easily be written so that each of these variables appears with a +1 coefficient in one of these constraints and with a -1 in the other.

Using the language of combinatorial optimization theory, we can say that an LP model has the form of a network flow model if the constraint matrix is "totally unimodular." (ref: NEM, p. 540-542). In general, LP models for tables of dimension 3 or higher cannot be put into this form; however, CPLEX is able to search for and extract submatrices of the constraint matrix that have the network form. Since network flow models can be solved faster than the typical LP problem, this extraction of network submatrices may lead to shorter running times.

5. Integer Programming Formulation of the Secondary Cell Suppression Problem

Two of the cost functions for the secondary cell suppression problem are : (1) the total number of cells to be suppressed or (2) the total value of the cells to be suppressed . In order to express these functions exactly, one needs to associate a binary variable to each cell ; it will take on the value '1' if the cell is selected for suppression or the value '0' otherwise. Such binary variables are available in integer programming models but not in linear programming models. Linear programming models use the same continuous variables in the cost function that are used in the constraints. These continuous variables, which one may call "flow variables", represent the change in value of cell required to provide the desired protection to the primary. In contrast, integer programs use these continuous variables in the constraints, but use binary variables in the cost function. In integer programs, these two sets of variables are related by inequalities of the form x # U · z where x is continuous and z is binary and 'U' is the upper bound of the variable x (ref: FIS).

Specifically, in a linear programming model in which x(i) represents the flow associated with cell 'i', the cost function Cost is expressed as:

$$\text{Cost} = \sum_{cells\ i} c(i) \cdot x(i)$$

where for case (1), total number, c(i)=1 for each cell i and for case (2), total value, c(i)=value(i). The value of the flow variables x(i) is determined by the optimizer which determines the minimum value of Cost.

6. The need for auditing programs

Some suppression programs implement a method that is known, from theory, to achieve the desired amount of protection for each primary. This is the case for those based on the integer programming method. It appears to hold also for those based on the LP formulation (ref: ROE2).
(The concern here is that the suppressed values are often known to be integral but that fact is not built into the LP based programs.) For any method for which it does hold, it is not necessary to evaluate the results of the suppression program with an 'auditing' program to see if the desired protection has been achieved for each primary. We say such programs are 'self-auditing.' However, when using programs based on network flow for tables of dimension greater than two, we have no such theoretical guarantees. That is, these programs are not 'self-auditing' and we must run an audit program based on a theoretically correct method that is designed to handle tables with dimension those of the input tables. Currently we use 3D or 4D audit programs based on the LP method to evaluate the results of network based programs when they are used to suppress 3D and 4D tables.

7. Warm Starts.

Our suppression programs protect the set of primaries sequentially; i.e., they find a suppression pattern that protects a given primary suppression, update the table database by flagging the newly selected complementaries, and then update the various quantities that will be used in determining a suppression pattern for the next primary. Some of the LP quantities change from primary to primary but there are also some that are static. The idea of a warm start for the suppression problem is based on reusing the static quantities (e.g., equality constraints) from primary to primary along with the basis generated by solving the optimization problem for the previous primary (ref: ROE). In the table below, we describe the 'dynamic status' of each of the key quantities used in the LP formulation of the cell suppression program. By 'dynamic' we mean that the quantity changes from primary to primary within a given table; by 'static' we mean it does not change. The term 'capacity' (used below) measures how much protection a given cell can provide to a given primary.

Dynamic Status of Key Quantities used in LP Formulation of the Suppression Program

----

LP quantities : dynamic (D) or static (S)

1. cost coefficients
   (D) since they depend on the flag status of each cell, which may change
2. bounds
   (D) since they depend on capacities which depend on the primary
3. equality constraints
   (S) since they express the additive structure of the given table

In the simplest cases of the secondary cell suppression problem, the capacities of all the cells are independent of the primary being protected. For example, sometimes the capacity of each cell is equal to its value. In these cases, the bounds are a static quantity and only the cost coefficients are dynamic.

There is an interesting and often positive consequence of using the warm start mode. The warm start uses as a starting basis the final basis from the previous primary. In general, the starting basis does not affect the final solution, i.e. selection of a suppression pattern. However, when there are multiple suppression patterns with the same value of the cost function, the pattern that is returned does depend on the initial basis. The selection of a suppression pattern is likely to affect the pattern and costs for succeeding primaries.

8. Choice of LP optimizer

There are really two decisions to make.
(1) Choice of which LP package to use. The best commercial packages often require a license with a substantial fee.
(2) Choice of which LP optimizer to select from the set of those available in the chosen LP package.
There are many LP packages available (ref: SOF). The Census Bureau disclosure limitation research group has a license for CPLEX. For this package, we found that there are great improvements in speed when one uses the optimizer based on the dual simplex method as opposed to that based on the primal simplex method. The CPLEX manual (ref: CPX) indicates that many LP problems are solved faster by the dual simplex optimizer than by the primal simplex optimizer. "For example, a primal-degenerate problem with little variability in the right-hand side coefficients but significant variability in the cost coefficients will usually lend itself to the dual simplex optimizer." (ref: CPX, p.70).

9. Test Results

Data: from 1997 Census of Wholesale Trade; Set of Linked Tables
Row structure and meaning:
The rows represent NAICS codes; 42 is the code for wholesale trade establishments.
The full NAICS has several levels of hierarchy representing production functions for the establishments. These are increasingly specific as one "descends" the hierarchy. For our testing we used only the highest level part of that hierarchy. 42 = 421 + 422; where 421 and 422 are each further partitioned into nine production types. Viewing 42 as level 1, we have:
Number of Rows: 21
Level 1 = 2 Level 2's ; each Level 2 = 9 Level 3's

Column structure and meaning:
Each table is defined by a column relation that expresses a partition of some geographical unit into smaller units (e.g., a state into counties, a Metropolitan Statistical Area (MSA) into counties)
We used the first five of the 90 relations for the state of California.
Number of Columns: 30
Structure of Linked Column Relations:
Relation 1: Level 1 (#1) = 26 Level 3's
Relation 2: Level 2 (#1) = 4 Level 3's (from the 26 in relation 1)
Relation 3: Level 2 (#2) = 2 Level 3's (" ")
Relation 4: Level 2 (#3) = 6 Level 3's (" ")
Relation 5: Level 1 (#1) =The three Level 2's in Relations 2,3,4 (=12 Level 3's) + 14 Level 3's
Note: Relation 5 is implied by the first four relations.

Variable for the 3rd table dimension: structure and meaning:
There 3 operational categories for wholesale trade plus a sum level.
Total number of cells
There are 21 x 30 x 4 = 2520 cells of which 18 x 26 x 3 = 1404 are interior cells.
There are 411 primary suppressions with a total value of 47M (M=Million dollars).

TABLE: RUN TIMES AND SUPPRESSION PATTERNS FOR VARIOUS MODELS

	Run Time	Suppression Pattern (#C's, total value)	
Low Protection Level			
LP (warm)	12 min	706 C	val=259 M
LP (cold)	18 min	717 C	val=306 M
network	28 sec	837 C	val=291 M

Moderate Protection Level

LP (warm)	13 min	735 C	val=276M
LP (cold)	20 min	758 C	val=325 M
network	32 sec	841 C	val=312 M

Full Protection Level

LP (warm)	18 min	815 C	val=372 M
LP (cold)	25 min	848 C	val=418 M
network	33 sec	871 C	val=324 M

To explain these protection levels, we first have to define 'backtracking' (ref: Wil,p.204). A complementary suppression that provides protection to a primary will need protection for itself (so that it cannot be determined too precisely). If this complementary appeared in a table that was previously suppressed, the program has to ensure that the complementary has the needed amount of protection in the previous table. This may require that the program revisit the previous table and protect the complementary there as if it were a primary. Backtracking can account for a significant fraction of the total run time; to reduce this time one can decrease the amount of protection sought for these complementaries during the backtracking stage. That decrease is relative to the full protection level; in the table above we describe it as low or moderate protection levels.

On the DEC-ALPHA the LP programs ran slightly faster (about 25%). The speed superiority of the network approach may be due partly to non-modeling aspects of the programs (e.g. I/O) as well as the modeling differences (network vs. LP). However, after some detailed timing analysis, it appears that about 80 to 90 % of the total run time required for protecting a primary is consumed by the routine that implements the LP model and calls the LP solver.

Description of computers used in testing programs.
When comparing program running times, ideally the programs should be run on the same computer and that computer should be the one that will be used for production runs.
However, due to practical constraints and convenience we have not achieved this goal so far.
The production computers that are used for cell suppression runs are as follows:
(1) Network flow based programs are run on a DEC machine with the VMS operating system.
(2) LP based programs are run on a DEC Alpha machine with the UNIX operating system.
However most of our test runs were run on a SUN machine with SUN-OS/UNIX.

10. Conclusions

1. LP based programs vs. network flow based programs

It appears that the fastest LP based programs are sufficiently fast for some production work. They are currently slower than the network based programs by a factor greater than 20 (see table); however this is partly due to some I/O features that have not yet been implemented in the LP based programs. Since both theory and experience indicate that the LP based program produce a superior suppression pattern, it appears that LP based programs should be used unless the projected reduction in time by running the network based programs is crucial for meeting production deadlines. Note that in the test case, the LP based programs produced patterns with 6% to 16% fewer complementaries (depending on the case).

2. Choice of LP package

It appears that CPLEX is a very fast LP solver, has a choice of three optimizers, and allows great flexibility in modifying specific aspects of an LP program and in extracting the desired information from the solution. It is well documented and we have not noticed any bugs.

3. Choice of LP solver and running mode

It appears that there is significant reduction (factor of 3) in running times when choosing the solver based on the dual simplex method versus that based on the primal simplex method.This is due to the specific structure of the suppression problem. It appears there is a 1/3 reduction in running times when running the dual simplex solver in the warm start mode versus the cold start mode. The warm start mode generated fewer complementary suppressions in the test cases (see explanation in the last lines of section 7).

11. Goals for future work

It is possible that the warm start option could be improved by examining the basis that CPLEX forms in the course of solving the LP problem for a given primary. Perhaps a better understanding of how capacities depend on the primary being protected would allow one to change the bounds for a fewer number of cells as one traverses the list of primaries. Some effort may be given to extending the current set of suppression programs so that they can handle several linked tables (i.e., column relations) at once; this would allow such sets of linked tables to undergo suppression without the need for backtracking. The LP based programs are well suited for this since any set of (additive) columns relations can be expressed in an LP model. The LP solver can easily eliminate redundancies that often exist in the column relations.

Acknowledgments
Thanks to Jim Fagan for teaching me both the theory and implementation of LP suppression models. Thanks to Carol Blatt for many helpful discussions on a wide range of programming topics and for running the programs that produced many of the entries in the comparison table above. Thanks to Steve Roehrig for an explanation about the usefulness of warm starts for the suppression problem and suggestions for its implementation. Thanks to Phil Steel for some programming design suggestions.

References

COX: Cox, L. H., (1995), "Network Models For Complementary Cell Suppression", JASA, June 1995, v.90, p.1453-1462.

CPX: ILOG CPLEX 6.5 User's Manual, March 1999, publ. ILOG, Inc.

FIS: Fischetti,M., Salazar Gonzalez, J.J., (2000), Models and Algorithms for Optimizing Cell Suppression in Tabular Data with Linear Constraints, JASA, 9/2000, v. 95, no. 451, pp. 916-228

MAS: Massell, P.B., (2001), Cell Suppression and Audit Programs used for Economic Magnitude Data, Statistical Research Division Report RR2001-01, U.S. Census Bureau, http://www.census.gov/srd/papers/pdf/rr2001-01.pdf

NEM: Nemhauser, G.L., Wolsey, L.A., Integer and Combinatorial Optimization, Wiley, 1988

ROB: Robertson, D., (1994), Automated Disclosure Control at Statistics Canada, Proceedings of the Second International Seminar on Statistical Confidentiality, Luxemburg

ROE: Roehrig, S. , (2001), personal communication on the value of warm starts in the LP formulation of the suppression problem

ROE2: Roehrig, S., (2001), Integer Bounds for Suppressed Cells in Multi-Way Tables; handout from a presentation (based on work in progress.)

SAN: Sande, G., (1984), Automated Cell Suppression to Preserve Confidentiality of Business Statistics, Statistical Journal of the United Nations ECE, 2, p. 33-41.

SOF: A directory of optimization software: http://www-fp.mcs.anl.gov/otc/Guide/SoftwareGuide/index.html

WIL: Willenborg, L., de Waal, T., Elements of Statistical Disclosure Control, Lecture Notes in Statistics, v. 155, Springer, 2001.