OPTIMIZATION MODELS AND PROGRAMS FOR CELL SUPPRESSION IN STATISTICAL TABLES

Paul B. Massell
Statistical Research Division, U.S. Census Bureau, Washington, D.C. 20233-9100
paul.b.massell@census.gov

Keywords: cell suppression, network optimization, linear programming, integer programming, confidentiality, statistical disclosure control

Abstract[1]. There has been much international research in recent years on the cell suppression approach to protecting a statistical table. In this paper, the author compares some of the new and old methods from both a theoretical and practical point of view. The basic theoretical difference among the approaches is the type of mathematical programming method that is used to solve the optimization problem involved. However, the usefulness of a cell suppression program often depends on features in addition to the optimization method used; e.g., (1) ability to handle linked tables, (2) allowance for capacities, (3) guarantee of adequate suppression, (4) set of expressible cost functions and (5) adequate speed for major production runs. The author will compare some of the optimization methods and some of the programs that have been reported in the literature in recent years; e.g., network based programs (Jewett), extended network methods (J. Castro), linear programming based programs (Massell), integer programming based programs (Fischetti, Salazar), hypercube based programs (Giessing), and programs (e.g., Argus) that use several methods.

1.0 Introduction

The existence of sensitive cells in statistical tables that are released by statistical agencies has long been recognized. In frequency count tables, also known as contingency tables, cells with a value, especially a count of '1', may reveal confidential data. For example, a table of demographic data in which a cell that is defined to be the number of males of a certain race with income in a certain range in a certain locality, would be sensitive if it had a count of '1' because neighbors of this man might well be able to associate him with this cell and thus uncover his income. With economic data, sensitive cells generally arise in magnitude tables, i.e., tables in which the data displayed is that for some variable that measures some (continuous) economic quantity such as sales. For such data, the largest contributors of the magnitude variable are often known to table users. Often the number of contributors for each cell is published in an associated frequency count table. If there were one contributor, its value (e.g., total sales) would be revealed directly to a table user if the cell were published. This would violate the pledge of confidentiality to this contributor. Thus the cell cannot be released as is. It must be suppressed or otherwise modified. Even if there were two contributors to a cell, we would suppress the cell because the second contributor could, by subtraction of its own value, determine the value of its competitor. In general, in determining sensitivity, we make assumptions about what is known by companies about their competitors, and how good an a priori estimate they have of their competitors' cell contributions. Ways of measuring sensitivity of cells, determining how much protection each sensitive cell requires, and ways of achieving this protection have been developed over the past thirty years. Work continues on these topics but the major advances of recent years have centered on the computational aspects of the problem. The protection of sensitive cells is most commonly achieved by suppressing additional cells, called complementary cells. The reason for these complementary (also called 'secondary') suppressions is to make it impossible for any table user to estimate closely the value of the sensitive cells or that of any respondent contribution to such cells directly from the table. The determination of an optimal set of complementary cells is called the (complementary) Cell Suppression Problem (CSP).

The CSP can be stated in a pure mathematical form in which all the details of preprocessing and implementation details are ignored. In that pure form, the CSP has been shown to be an NP-hard optimization problem (one of the major algorithm classifications). Since the CSP is an easy-to-state yet challenging optimization problem that arises in a familiar setting (viz., additive statistical tables) the CSP has attracted the attention of researchers in the academic community who are experts in mathematical programming

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

algorithms commonly studied in the field of operations research. Network flow and linear programming have been applied to the CSP for over twenty years (ref: K-S-1). More recently, integer programming (ref: F-S-1), extensions of network flow (ref: Castro), and an algorithm called "hypercube" (ref: G-1) have been implemented. Sometimes a direct implementation of an algorithm leads to a program with long runtimes; in those cases, additional algorithmic features are needed to make the program fast enough for production work. For example, use of dynamic constraint generation and branch and cut methods have greatly speeded up integer programming programs (ref: F-S-1).

The description below of several algorithms and associated implementations can help a given agency (or organization or person) decide which features are desirable for protecting their tables. They may decide to acquire an existing program or to write their own. Agencies that already have a suppression program may decide to modify it to incorporate features of other programs.

2.0 Description of the input data: (1) table data (2) auxiliary data

Let us begin our description of suppression programs by describing the data and aspects of protection directly related to the data. We assume throughout this paper that we are given a table of dimension d of size (n1, n2,..., nd) that includes marginal totals; i.e., we assume the tables to be additive. We usually assume the cell entries are non-negative although this is mainly for simplicity of description. The data may be un-rounded or rounded. Rounding, by itself, provides some protection beyond that provided by cell suppression. Often it is known that the suppressed values must be integral. This fact may also affect the amount of protection provided by a given suppression pattern. Let us mention here an implementation feature that often modifies some data values. This feature is the recalculation of marginals from the interior data to ensure additivity.

There is a practical aspect of the CSP that is sometimes ignored in research papers but can be quite important in practice for certain types of tables. It fits in this section because it requires auxiliary input data. It is sometimes called the "common respondent problem" and can make the calculation of protection quite complicated. It arises because a table user may be a respondent who contributed data which is a component of some cell entries. Such a respondent will know one summand of one or more suppressed cells. Of course, one important goal of the suppression selecting process is to ensure that one respondent cannot learn sensitive information about other respondents. If the protection process achieves this goal, one says that protection is at the respondent level rather than at the cell level. Building in such protection requires access to data at the respondent level; i.e. microdata. Specifically, one must know the respondents (e.g., establishments or companies) and their values that contribute to each cell value.

3.0 Commonly used protection methods for magnitude and frequency data

Magnitude data tables are commonly produced for economic data. In such tables, cell values are enormously varied, sometimes differing by several orders of magnitude. Cell suppression has been used successfully for this type of data. For frequency data, it is possible to use cell suppression but other methods are often used instead because of ease of use. Traditional methods are collapsing of categories (table redesign), and rounding (of various types). One newer method is perturbation of the underlying microdata.

4.0 Table structure

When we discuss the various mathematical programming algorithms that can be used for the CSP, we will need to take into account the structure of the table. The simplest structure is that of a simple table, in which there are no subtotals in any of the dimensions; i.e., each dimension has exactly one total level which is the sum of all the other levels. A 'simple' table has the minimum number of additive relations for a table of its size. Additive relations form the constraints in the various mathematical programming models we consider below. The next simplest case is a table in which there is hierarchy of depth one for exactly one of the dimensions. It turns out that network flow models can model tables of (at most) dimension two and a hierarchy (of any depth) in at most one dimension but cannot model more highly structured tables (ref: F-1). Here we are referring to standard network flow models, not the extensions to such models that have been described recently by Castro. These standard models can be shown to be equivalent to a special class of LP models. However, this special class can be run much faster using the network flow algorithm than using one of the LP algorithms (e.g., simplex) (ref: I-1, chap. 6).

5.0 Consistency of data protection across tables and over time

5.1 Processing a set of tables with overlapping data (i.e., linked tables) on one run

One aspect of cell suppression, which is often given only light treatment in theoretical papers, is how to treat linked tables in a consistent way. By 'consistent way' we mean that if a given cell appears in two or more tables that are undergoing suppression during the same computer run, that cell should be given the same suppression status in all the tables. ("Suppression status" usually includes some of the following 4 cases: display (true) value, suppress, display an interval that includes the exact value, display a value chosen from that interval). "Backtracking" is used in the implementations we are most familiar with (ref: J-1, M-1) to ensure consistency. Currently, these implementations deal only with 2 cases of suppression status, exact value or suppression. In these programs, once a given cell is chosen as a complementary suppression, all tables previously suppressed (in the given computer run) that contain the new 'C' cell, are revisited with the goal of ensuring that in each such tables, the new C cell has adequate protection. It is easy to see that this revisiting can greatly increase the runtimes when many C cells are selected in the overlapping regions of the linked tables.

5.2 Processing a set of tables with overlapping data (i.e., linked tables) on more than one run

In the programs referred to in section 5.1, there is mechanism that allows cells belonging to linked tables to be treated in a consistent way, even if the tables in the set are processed on different computer runs. This mechanism involves the notion of "freezing cells"; i.e., the suppression status of the each cell is written in the master data file at the end of the first computer run. The program then tries to preserve that suppression status on future runs. That is, if a cell was suppressed on the first run, then all future runs begin with that cell suppressed. Conversely, if a cell is deemed important to display (i.e., not to suppress), then an attempt is made to display that cell on all runs. This can be usually be accomplished by associating a very high cost to a cell that we want to display (ref: J-1; M-1; G-2)

6.0 Output Data

When an input table has undergone traditional cell suppression the output is a table in which either the final cell entry is the same as the initial cell entry (a numerical value) or is a symbol (e.g., D) indicating that the cell has been suppressed (i.e., in traditional (complete) suppression, no numerical information is given for such cells). There are alternatives to this traditional output format. For example, one can, instead of the writing a symbol for a suppressed cell, display the audit interval for that cell; i.e., the [min,max] feasibility interval that can be calculated by an audit program whose input is the table with suppressions. Another way of displaying the output table comes from a new protection method, called "interval publication" (ref: S-1). In this method, for sensitive cells, the (desired) protection interval replaces the true value, and some other (secondary) cells also have their values replaced by intervals so that the sensitive cells are protected. (See section 8.0 on "interval publication").

7.0 Cost functions and information loss

In the network flow and LP models, if one wants to select a suppression pattern by minimizing the total value of the cells selected for suppression, one can do this approximately with the continuous variables that are defined for the cells (the ith cost coefficient is the value for cell i, the ith variable represents the 'flow' through cell i). However, in order to do this precisely or to minimize the total number of suppressed cells, ones need to use binary variables that assign a '1' to each primary suppression and each potential complementary suppression and a '0' to the displayed cells. Binary variables are not available in network and LP models but are available in integer programming models. Thus the latter models allow for a wider range of cost functions, including some very natural ones. Updating of the cost coefficients is frequently done as a new sensitive cell is about to be protected. Specifically, all suppressed cells determined prior to the current stage of the program, can be assigned a zero cost. In the hypercube method, the loss function is linear in the logarithm of the cell value. Negative costs are associated to any sensitive cell or complementary cell that had been suppressed at an earlier stage
(ref: G-2).

There is a simple way to reduce the amount of oversuppression after an adequate suppression pattern has been found for a given sensitive cell. One creates a new and much smaller problem that treats all the (preliminary) complementary cells selected for that sensitive cell as the set of possible complementary cells. One then uses a new loss function, perhaps one that is quite different from the original one, and tries to see if a strict subset of the set of preliminary complementary cells suffice to protect the sensitive cell. In fact, if the second loss function is chosen well, it may be possible to produce a final suppression pattern similar to that which one gets from use of a single loss function expressed using binary variables. There are additional ways to test for oversuppression; e.g., by varying the suppression pattern in reasonable ways and checking each variation using an auditing program to see if it provides sufficient protection. This will tend to produce a locally minimal but, in general, not globally optimal, suppression pattern.

8.0 Interval Publication and Information Loss

There is another way of constructing an information loss function. The method we describe below is given in a paper on interval publication (ref: S-1) but could also be used in standard (complete) cell suppression. The method is based on the assumption that a typical user has some knowledge of some of the cell values prior to release of the table (undergoing suppression). We then assume that this a priori knowledge, which may be based on information from various sources, can be expressed in the form of a priori bounds for each cell value. We then use these bounds to express the information loss for each cell. For example, say, a user knows that cell A, prior to release of the given table, has a value in the interval [a1,b1]; we could call this the uncertainty interval. The narrower this interval, the more the user knows a priori about the cell value. A reasonable measure for this loss of information (due to suppression of the cell) is 'b1-a1'. This means the information loss when a cell is suppressed is proportional to the width of the uncertainty interval that a typical user has for that cell value. In the extreme case in which users know the (exact) cell value prior to agency release of the table, there would be no information loss in suppressing that cell. The author believes that these ideas can be extended to the case of a priori knowledge about each respondent's value; i.e., knowledge about the summands for each cell. This idea is perhaps most important in the case in which a user is a respondent. When the a priori bounds vary among the group of table users, one has to decide which bounds to use in the information loss function.
Interval publication may be compared with controlled rounding or even source data (e.g., microdata) perturbation in that the information loss is not concentrated in a small number of suppressed cells but is spread over a wider number of cells. (One could say the information loss is smoothed over a broader part of the table than in complete cell suppression). It is claimed that clever use of integer programming methods (e.g., branch and cut) can make interval publication much faster than the standard implementation of LP methods for complete cell suppression (ref: S-1).

9.0 Mathematical Programming Options for solving the complete CSP

9.1. Ordinary Network Flow

References: (theory: Co-1; practice: J-1)
Range of application:
Provides desired protection for 2d -tables with a hierarchy in at most 1 dimension.
Can be extended for n-dim tables with n > 2 or for 2-d tables beyond those above but desired protection is not guaranteed for these cases
Limitations: Oversuppression is common for all cases but usually tolerable; undersuppression for the cases mentioned above; limited types of cost functions can be expressed
Strong Points: extremely fast; often 30 times faster than LP (ref: I-1, chap. 6; M-1)

9.2 Linear Programming

References: (theory M-2; K-S-1; practice: M-1; K-S-1)
Range of application:
Provides desired protection for tables of all dimensions and degrees of hierarchy.
Limitations: Oversuppression is common but usually modest; limited types of cost functions can be expressed;slow for large tables (e.g., 100 by 100) since runtime may increase with cube of problem size;
Strong Points: easy to implement the optimization problem; can call a standard LP package after the LP problem related structures (constraint matrix, b vector, etc.) have been defined

9.3 Integer Programming

References: (theory: J-S-2; practice: J-S-2, K-S-1)
Range of application:
Provides desired protection for tables of all dimensions and degrees of hierarchy.
Can provide optimal solution if all sensitive cells are protected simultaneously.
Limitations: Very slow even on tables of moderate size if implemented in a straight forward way; (can be speeded up greatly with clever use of branch and cut methods, but these require specialized knowledge). The first implementations of this method (see references) may have had protection only at the cell level, not at the respondent level, since there was no mention of cell capacities. However, capacity code could easily be added.

Strong Points: Can express the full range of cost functions since binary and integer variables can be used in the model description

9.4 The hypercube method

References: (theory: G-R-1; G-2; practice: G-R-1; G-1, G-H-1)
Range of application.
Provides protection from exact disclosure for tables of all dimensions and degrees of hierarchy.
Divides n-dim tables with hierarchical structure into a set of n-dim tables without substructure.
Limitations: Often oversuppresses (estimate of 30% in ref: G-1) and it may not find the best suppression pattern even for a single sensitive cell. This occurs because it is considering only the simplest types of suppression patterns.
Strong Points. Very fast for the reason that it considers a single type of suppression pattern and thus can handle enormous tables. Will likely be the algorithm applied to large tables in Argus (ref: A-1).

9.5 Extensions of network flows methods (e.g., multicommodity networks; networks with side constraints)

References: (theory: Ca-1; practice: Ca-1)
Range of applications: Can be used to protect 3-dim tables including ones with hierarchies and ones that are linked.
Limitations: Current versions cannot handle tables of dim $n > 3$. First few computational experiments show longer runtimes than non-network model application of dual simplex method. Does not preserve the integrality property of minimum cost network flow models.
Strong Points: Can be implemented using a general linear programming solver that exploits the network with side constraints structure of the constraint matrix for the CSP. Can be used in conjunction with the integer programming methods after problem has been decomposed into lower dimensional subtables. Future work may lower runtimes.

10.0 Summary:

There are a range of techniques currently being used or in development that are increasing the amount of information released from statistical tables while satisfying sensible confidentiality requirements. With the choice of algorithm, choice of information loss functions, and use of audit functions to revise the initial suppression pattern, the user now has more options for doing cell suppression, both traditional and its newer variants and more control over the final suppression pattern. He also has a fairly good idea of the tradeoffs involving runtimes and information loss that each method entails. The pure cell suppression problem for individual independent tables is a computationally difficult problem. When one adds the complications due to linked tables, common respondents, and the need to limit the cells that can be suppressed, it becomes even more difficult. It appears that the best approach to program design is to have two or more main computational algorithms, which can be selected either by the user or by a "smart" program, that possess a reasonable balance of information loss, risk, and runtime. In some cases, a suppression run followed by an audit and a second suppression provides such a reasonable tradeoff. The problem of consistency of suppressions among tables from different runs needs to be considered; construction of a database containing records for each cell previously released is one approach currently being implemented.

The author views this paper as a first effort to give a comparative analysis of cell suppression methods; he plans to provide more detailed analyses in future publications.

Acknowledgments: I have benefitted greatly from discussions with Laura Zayatz, Jim Fagan, Bob Jewett, Carol Blatt of the Census Bureau and Steve Roehrig of Carnegie Mellon University.

References:

A-1:Argus: online documentation http://www.cbs.nl/sdc/task-tt.htm; http://www.unece.org/stats/documents/2001/03/confidentiality/2.e.pdf

Ca-1: Castro, Jordi,(2002) , "Network Flow Heuristics for Complementary Cell Suppression: an Empirical Evaluation and Extensions", in Inference Control in Statistical Databases: From Theory to Practice, Lecture Notes in Computer Science, vol. LN2316, Springer

Co-1: Cox, Lawrence H., (1992), "Solving Confidentiality Problems in Tabulations Using Network Optimization: A Network Model for Cell Suppression in the U.S. Economic Censuses", Proceedings of the International Seminar on Statistical Confidentiality, Dublin.

Co-2: Cox, Lawrence H., (2000), "A Comprehensive Methodology for Complementary Cell Suppression in Tabular Data", unpublished.

F-1: Fagan, James, (2001), personal communication (a proof of the how network flow methods can be used to model 2-d with a hierarchy in one dimension)

F-S-1: Fischetti, Matteo, and Salazar-Gonzalez, Juan-Jose, (2000), "Models and Algorithms for Optimizing Cell Suppression in Tabular Data with Linear Constraints", Journal of the American Statistical Association.

G-1: Giessing, Sarah, (1999), "A Survey on Packages for Automated Secondary Cell Suppression", Proceedings of the Eurostat/UN-ECE Work Session on Statistical Data Confidentiality, March, Thessaloniki

G-2: Giessing, Sarah, (2001) "New Tools for Cell Suppression in Tau-Argus: One Piece of the CASC Project Work Draft", Joint ECE/Eurostat Work Session on Statistical Data Confidentiality, Macedonia

G-3: Giessing, Sarah, (2001) "Nonpertubative Disclosure Control Methods for Tabular Data", Chapter 9 (p.185-213) of book "Confidentiality, Disclosure, and Data Access", Elsevier, 2001

G-H-1: Giessing, Sarah, and Hundepool, Anco, (2001) "The CASC Project: Integrating Best Practice Methods for Statistical Confidentiality", in Pre-Proceedings of the NTTS & ETK 2001 Conference, Crete, (http://webfarm.jrc.cec.eu.int/etk-ntts/Papers/final_papers/79.pdf)

G-R-1: Giessing, Sarah, and Repsilber, Dietz, (2002), Tools and Strategies to Protect Multiple Tables with the GHQUAR Cell Suppression Engine, in Inference Control in Statistical Databases: From Theory to Practice, Lecture Notes in Computer Science, vol. LN2316, Springer

I-1: Ilog Cplex 7.1 User's Manual (2001)

J-1: Jewett, R. (1993), " Disclosure Analysis for the 1992 Census", unpublished manuscript, U.S. Census Bureau, Economic Statistical Methods and Programming Division

K-S-1: Kirkendall, Nancy, and Sande, Gordon, (1998), "Comparisons of Systems Implementing Automated Cell Suppression for Economic Statistics", J. Official Statistics, (v.14, #4) p.513-536

M-1: Massell, Paul 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)

M-2: Massell, Paul B., (2001), "Using Linear Programming for Cell Suppression in Statistical Tables: Theory and Practice", in 2001 Proceedings of the ASA, Alexandria, VA. [CR-ROM].

S-1: Salazar-Gonzalez, Juan-Jose, (2002), "A Unified Mathematical Programming Framework for different Statistical Disclosure Limitation Methods for Tabular Data", unpublished manuscript.

W-d-W-1: Willenborg, Leon, and de Waal, Ton, (2001), Elements of Statistical Disclosure Control, Lecture Notes in Statistics, v. 155, Springer