

STATISTICAL DISCLOSURE CONTROL OF TABULAR FORMAT MAGNITUDE DATA -- WHY IT IS NOT A GOOD IDEA TO USE HOME GROWN (MANUAL OR AUTOMATED) CELL SUPPRESSION PROCEDURES

Ramesh A Dandekar

U. S. Department of Energy, Washington DC 20585
Ramesh.dandekar@eia.doe.gov

Abstract

Multiple variations of complementary cell suppression procedures are used by statistical agencies to protect sensitive tabular format *magnitude data* from a statistical disclosure. In this paper, we address potential weaknesses associated with some of the most commonly used home-grown cell suppression procedures to protect tabular format *magnitude data*. We also provide some guidance to the survey operations staff on the standard procedures that need to be followed to ensure adequate protection from statistical disclosure of sensitive tabular cells in statistical publications.

Introduction

Statistical agencies routinely use complementary cell suppression techniques to protect sensitive tabular cells from statistical disclosure in published tables. Irrespective of the technical advances in the linear programming (LP) based cell suppression methods, there is a tendency among statistical agencies to rely heavily on the variations of “home grown” manual cell suppression procedures. Over time, in an attempt to increase the efficiency and to maintain the consistency of these manual cell suppression procedures, attempts are made to automate the logic used in these same “home grown” manual cell suppression procedures. In this paper, we use the basic principles from graph theory to demonstrate the potential weaknesses associated with flawed “automated/manual” home grown cell suppression procedures. In the recent past, these flawed procedures have been advocated to the statistical community in the conference and workshop setting. Our objective here is to make the statistical community aware of the potential pitfalls associated with these procedures and to emphasize the importance of using appropriate solution procedures based on the linear programming models.

Typical Manual Cell Suppression Procedure

Three basic pieces of information are required to properly identify sensitive tabular cells. They are: a) cell value, b) largest contribution to the cell value, and c) second largest contribution to the cell value. To get around the “relatively complex computational logic/task” required to determine the largest and the second largest contribution to each and every published cell in all tables in statistical publication, often a total number of contributors to the cell value is used as a surrogate variable. A total cell count of either two contributors or three contributors is used solely to declare the table cell as sensitive. For a discussion in this paper, we assume that the proper procedure based on the cell value, largest and second largest contribution to the cell value in combination with appropriate sensitivity rule is used to identify sensitive table cells.

Almost always, after a table cell is identified as a sensitive cell, the amount of protection required by the sensitive cell is completely ignored during the search for potential candidate complementary suppression cells. Historically it is known that (1) “minimizing the information loss” should be the primary objective in the manual search for potential candidate table cells as complementary suppression cells, and that (2) suppressing only one cell in a given table row or table column (or in any given table dimension) causes exact disclosure of the suppressed cell, when marginal total cell values are readily available. Based on these two criteria, the individuals tasked to perform manual cell suppression procedures are often instructed to ensure that (1) at least two table cells are suppressed in every table dimension and (2) cells selected for the

complementary cell suppression task are the smallest available in the pool of candidate cells available in a given row or column. Over time, these two basic principles used in typical “home grown” manual complementary cell suppression procedure evolve into computer programming steps as a part of the automation of complementary cell suppression task. Figure 1 summarizes steps in one such manual complementary cell suppression procedure. Figure 2, in combination with the criterion identified in Figure 1, shows steps in widely advocated automated complementary cell suppression procedures.

FIGURE 1

TYPICAL MANUAL CELL SUPPRESSION PROCEDURES

- Identify Sensitive Tabular Cell Typically Table Cells Containing Three or Less Respondents are Identified as Sensitive.
- Ensure That There Are At least Two Suppressed Cells in a given row/column/level or a Dimension.
- In an Attempt to minimize information Loss, Smallest possible cell values are selected for complementary cell suppression task in a given row/column/level.
- Magnitude of Protection Required by the Sensitive Cell is Typically completely ignored

FIGURE 2

WIDELY USED (AUTOMATED) MANUAL CELL SUPPRESSION PROCEDURE

- (1) IF THE PRESENT ROW OR COLUMN REQUIRES COMPLEMENTARY SUPPRESSION, USE AS A COMPLEMENT THE CELL IN THE CORRESPONDING ROW OR COLUMN THAT HAS PREVIOUSLY BEEN USED AS A COMPLEMENT THE LARGEST NUMBER OF TIMES.
- (2) IF TWO OR MORE CELLS TIE FOR BEING USED AS A COMPLEMENT THE LARGEST NUMBER OF TIMES, USE AS COMPLEMENT THE CELL AMONG THE TIES WITH THE SMALLEST VALUE.
- (3) IF THERE ARE TWO OR MORE TIES FOR THE SMALLEST VALUE, PICK THE COMPLEMENT RANDOMLY.
- (4) ANY ROW OR COLUMN WITH A CELL SUBJECT TO PRIMARY SUPPRESSION BECOMES A ROW OR COLUMN REQUIRING COMPLEMENTARY SUPPRESSION.
- (5) ONCE PRIMARY SUPPRESSION TRIGGERS COMPLEMENTARY SUPPRESSION IN ANY ROW OR COLUMN, COMPLEMENTARY SUPPRESSION CONTINUES UNTIL THAT ROW OR COLUMN HAS AT LEAST TWO CELLS THAT HAVE BEEN SUPPRESSED.

Prepared By Ramesh A. Dandekar 18

Illustrative Example

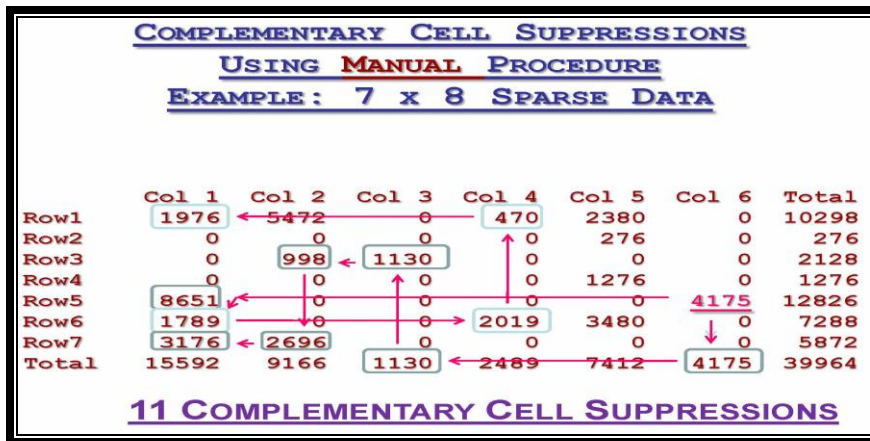
To illustrate the basic logic behind the widely advocated “home grown” complementary cell suppression procedure, we use a simple two dimensional 7 columns by 8 rows table as in Figure 3. The table contains *one sensitive* cell in row 5, column 6 position. We use the notation **Cnm** to identify a table cell in row n and in column m. Using this notation, the sensitive cell in Figure 3 is identified as C56 with a value of C56 = 4175. To protect this sensitive cell and by using the procedure from Figure 1 and Figure 2, the complementary suppression cells are selected in the following sequence:

a) To protect sensitive cell in 5th row, C56 => C51 (to protect C56 in row 5) => C61 (to protect C51 in column 1) => C64 (to protect C61 in row 6) => C14 (to protect C64 in column 4) => C11 (to protect C14 in row 1). We now have more than two suppressions in column 1 (C11, C51, C61).

b) We then proceed to protect sensitive cell in 6th column C56 => C86 (to protect C56 in column 6) => C83 (to protect C86 in row 3) => C33 (to protect C83 in column 3) => C32 (to protect C33 in row 3) => C72 (to protect C32 in column 2) => C71 (to protect C72 in row 7). We now have more than two suppressions in column 1 (C11, C51, C61, C71).

A total of eleven complementary cell suppressions are required to protect sensitive table cell C56.

FIGURE 3



Separability of Cell Suppression Pattern

Based on the basic principles from a graph theory, the suppression pattern consisting of 12 suppressions from Figure 3 is made up of two separate components. The first separable component consists of eight suppressions C56 => C51 => C71 => C72 => C32 => C33 => C83 => C86 (Figure 4) and it includes sensitive cell C56. The second separable component consists of 4 suppressions C11 => C61 => C64 => C14 (Figure 5). The second component is not connected to the sensitive cell C56 at all. The lack of connection of Figure 5 suppressions with the sensitive cell C56, makes these four suppression unnecessary and therefore should be considered over-suppressions. *Over-suppression is one major drawback of the “home grown” complementary cell suppression procedure.*

FIGURE 4

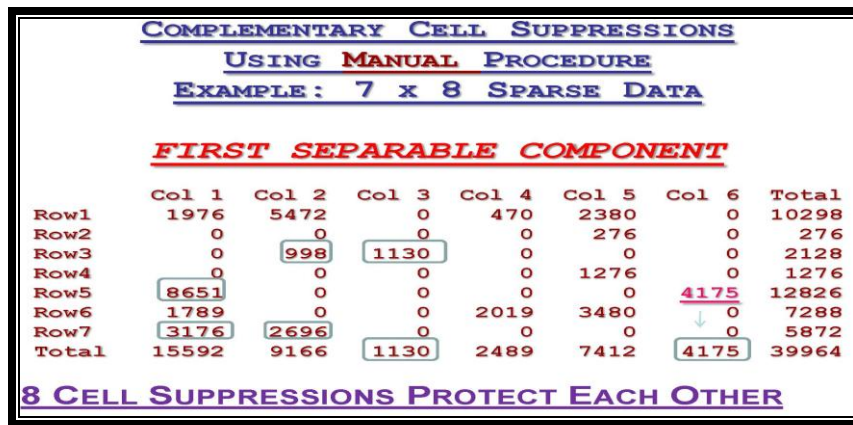


FIGURE 5

**COMPLEMENTARY CELL SUPPRESSIONS
USING MANUAL PROCEDURE
EXAMPLE: 7 x 8 SPARSE DATA**

SECOND SEPARABLE COMPONENT

	Col 1	Col 2	Col 3	Col 4	Col 5	Col 6	Total
Row1	1976	5472	0	470	2380	0	10298
Row2	0	0	0	0	276	0	276
Row3	0	998	1130	0	0	0	2128
Row4	0	0	0	0	1276	0	1276
Row5	8651	0	0	0	0	4175	12826
Row6	1789	0	0	2019	3480	0	7288
Row7	3176	2696	0	0	0	0	5872
Total	15592	9166	1130	2489	7412	4175	39964

**4 CELL SUPPRESSIONS PROTECT EACH OTHER
UNFORTUNATELY THERE IS NOTHING TO PROTECT**

Figure 6 shows the lower and the upper bounds of the suppressed cells from Figure 3. These bounds are determined by using a linear programming audit of the cell suppression pattern. The Disclosure Audit Software (DAS) developed by the Federal Committee on Statistical Methodology (FCSM) was used to perform this task. Based on the DAS outcome in Figure 6, the distance between the true cell value from the lower and upper bounds for the eight cells belonging to the first component are determined by cells C32 (value = 998) and C33 (value = 1130). The distance between the true cell values from the lower and the upper bounds for the four cells belonging to the second component are determined by cells C11 (value = 1976) and C14 (value = 470). The outcome in Figure 6 further confirms the existence of two distinct, separable, and independent cell suppression patterns in the table shown in Figure 3. Notice that cells C32 and C33 were selected as complementary suppression cells because they were the smallest among the candidate complementary suppression cells. Relatively small values for C32 and C33, such as say 5 units, would have brought lower and upper bounds for the remaining six cells too close to the true call values. This would have caused statistical disclosure for the remaining six cells belonging to the first component.

FIGURE 6

Low	TRUE	Max	True-Low	Max-True	Component
0	1976	2446	1976	470	1
7521	8651	9649	1130	998	2
1319	1789	3765	470	1976	1
2178	3176	4306	998	1130	2
0	998	2128	998	1130	2
1566	2696	3694	1130	998	2
0	1130	2128	1130	998	2
0	1130	2128	1130	998	2
0	470	2446	470	1976	1
43	2019	2489	1976	470	1
3177	4175	5305	998	1130	2
3177	4175	5305	998	1130	2

Naked Cell Suppression Pattern

Another undesirable property of the “home grown” complementary cell suppression procedure is that it creates naked suppressions (suppressed cell, for which values could be determined exactly). This property is relatively easy to demonstrate by using any generic table. In Figure 7 and in Figure 8, we use two different generic two-dimensional tables containing 10 rows and 10 columns. The last column in these two tables is the sum of the first nine columns. Similarly, the last row in these two tables is the sum of the first nine rows. Both the tables have only one sensitive cell located in row 5 and column 5. The sensitive cell is denoted by a symbol “P”.

FIGURE 7

ONE SENSITIVE CELL "P" AND EIGHTEEN COMPLEMENTARY SUPPRESSIONS									
						X12		X13	
X2		X3							
					X10	X11			
			X8		X9				
X1				P					
	X6		X7						
				X15			X16		
	X5	X4						X14	
				X18			X17		

FIGURE 8

ONE SENSITIVE CELL "P" AND EIGHTEEN COMPLEMENTARY SUPPRESSIONS									
X8	X9								
X7	X6						CRITICAL X		
	X5	X4							
		X3	X2						
			X1	P					
				X10	X11				
					X12	X13			
						X14	X15	X18	
							X16	X17	

In both of these tables, a total of eighteen complementary cell suppressions are required to protect sensitive cell "P" by using the typical homegrown complementary cell suppression procedure. The complementary suppression cells were selected in an increasing sequence from X1 to X18. Cell Xn (n= 1 to 18) was selected to protect cell Xn-1¹, entirely based on the criteria that the cell Xn was determined to be the smallest value cell in a given column or a given row during the search process to locate the small value cell. Based on the separability principles, both Figure 7 and Figure 8 cell suppressions have three "independent" components, identified by using a separate color scheme. In both tables, a component associated with the sensitive cell "P" consists of naked suppressions. The values for these cells (including the value of the sensitive cell "P") can be determined exactly. The other two components are not at all "connected" to the sensitive cell and therefore should be considered over-suppressions. It is relatively easy to quantitatively verify these findings by performing suppression pattern audits of tables in Figure 7 and in Figure 8. This could be done by first populating these tables with any numeric non-zero (and additive) table cell values and thereafter by performing a cell suppression audit of resultant table to determine the lower and the upper bounds on the suppressed cells.

¹ An exception to this rule is when Xn is used to directly protect sensitive cell. In Figure 7, X15 and in Figure 8, X10 is used to directly protect sensitive cell.

FIGURE 9

SUPER IMPOSE SUPPRESSION PATTERN FROM PREVIOUS SLIDE									
PERFORM SUPPRESSION PATTERN AUDIT USING DAS TO VERIFY DISCLOSURES									
11	12	13	14	15	16	17	18	19	135
21	22	23	24	25	26	27	28	29	225
31	32	33	34	35	36	37	38	39	315
41	42	43	44	45	46	47	48	49	405
51	52	53	54	55	56	57	58	59	495
61	62	63	64	65	66	67	68	69	585
71	72	73	74	75	76	77	78	79	675
81	82	83	84	85	86	87	88	89	765
91	92	93	94	95	96	97	98	99	855
459	468	477	486	495	504	513	522	531	4455

We have used cell values from Figure 9 and have super-imposed cell suppression patterns from Figure 7 and Figure 8 to quantitatively verify the findings based on graph theory. The cell suppression audit outcome is presented in Figure 10 and Figure 11. Figure 10 and Figure 11 cell suppression audit findings further confirms the existence of “**naked suppressions**” and “**over-suppressions**” associated with the typical “home grown” complementary cell suppression procedures.

FIGURE 10

Low	TRUE	Max	Status	Tru-Low	Max-Tru	Component
0	17	36		17	19	1
0	19	36		19	17	1
21	21	21	**	0	0	2
23	23	23	**	0	0	2
19	36	55		17	19	1
18	37	54		19	17	1
27	44	63		17	19	1
27	46	63		19	17	1
51	51	51	**	0	0	2
55	55	55	**	0	0	2
45	62	81		17	19	1
45	64	81		19	17	1
0	75	153		75	78	3
0	78	153		78	75	3
63	82	99		19	17	1
83	83	83	**	0	0	2
72	89	108		17	19	1
17	95	170		78	75	3
23	98	176		75	78	3

FIGURE 11

Low	TRUE	Max	Status	Tru-Low	Max-Tru	Component
0	11	23		11	12	1
0	12	23		12	11	1
9	21	32		12	11	1
11	22	34		11	12	1
32	32	32	**	0	0	2
33	33	33	**	0	0	2
43	43	43	**	0	0	2
44	44	44	**	0	0	2
54	54	54	**	0	0	2
55	55	55	**	0	0	2
65	65	65	**	0	0	2
66	66	66	**	0	0	2
76	76	76	**	0	0	2
77	77	77	**	0	0	2
87	87	87	**	0	0	2
0	88	177		88	89	3
0	89	177		89	88	3
9	98	186		89	88	3
11	99	188		88	89	3

Solution By Using Network Flow Model

In Figure 12 we show the complementary cell suppression outcome from the network flow model made available to other Federal statistical agencies by the U. S. Census Bureau, when used on a table in Figure 3. A total of five complementary cell suppressions are required to protect the sensitive cell at 10% protection level. The overall quantity suppressed is minimal in an attempt to minimize the information loss.

FIGURE 12

EXAMPLE : 7 x 8 TABLE

	COL 1	COL 2	COL 3	COL 4	COL 5	COL 6	TOTAL
Row1	1976	5472	0	470	2380	0	10298
Row2	0	0	0	0	276	0	276
Row3	0	998	1130	0	0	0	2128
Row4	0	0	0	0	1276	0	1276
Row5	8651	0	0	0	0	4175	12826
Row6	1789	0	0	2019	3480	0	7288
Row7	3176	2696	0	0	0	0	5872
TOTAL	15592	9166	1130	2489	7412	4175	39964

ONE SENSITIVE CELL

FIVE COMPLEMENTARY CELL SUPPRESSIONS

Summary Findings and Conclusions

“Home grown” complementary cell suppression procedures which have been traditionally used by the statistical community for the last ten to fifteen years are flawed. These procedures have a tendency to over-suppress, under-protect and often create naked suppressions. Often these procedures fail to protect sensitive tabular cells. Complementary cell suppression procedures based on network flow models and on generic linear programming models have the better track record in protecting sensitive tabular cells. There is no substitute for linear programming-based cell suppression procedures.

References

- Dandekar R. A. (2001) "[Synthetic Tabular Data: A Better Alternative To Complementary Data Suppression - Original Manuscript Dated December 2001](#)". Energy Information Administration, U. S. Department of Energy. Also available from CENEX-SDC Project International Conference, PSD2006, Rome, Italy, December 13-15, 2006, Companion CD Proceedings ISBN: 84-690-2100-1.
- Dandekar R. A. and Cox L. H. (2002), [Synthetic Tabular Data: An Alternative to Complementary Cell Suppression, 2002](#). Manuscript, Energy Information Administration, U. S. Department of Energy.
- Dandekar, R.A (2003), [Cost Effective Implementation of Synthetic Tabulation \(a.k.a. Controlled Tabular Adjustments\) in Legacy and New Statistical Data Publication Systems](#), working paper 40, UNECE Work session on statistical data confidentiality (Luxembourg, 7-9 April 2003)
- Dandekar Ramesh A. (2004), [Maximum Utility-Minimum Information Loss Table Server Design for Statistical Disclosure Control of Tabular Data](#), pp 121-135, Lecture Notes in Computer Science, Publisher: Springer-Verlag Heidelberg, ISSN: 0302-9743, Volume 3050 / 2004, Title: Privacy in Statistical Databases: CASC Project International Workshop, PSD 2004, Barcelona, Spain, June 9-11, 2004.
- Dandekar Ramesh A. (2005), "[Complementary Cell Suppression Software Tools for Statistical Disclosure Control - Reality Check](#)", Joint UNECE/Eurostat Work Session on Statistical Data Confidentiality (Geneva, Switzerland, 9-11 November, 2005)
- Dandekar Ramesh A. (2007), "[Comparative Evaluation of Four Different Sensitive Tabular Data Protection Methods Using a Real Life Table Structure of Complex Hierarchies and Links](#)", working paper 17, UNECE Work session on statistical data confidentiality (Manchester, United Kingdom, Dec 17-19, 2007)
- Evans T., Zayatz L., Slanta J. (1998), "Using Noise for Disclosure Limitation of Establishment Data", Journal of Official Statistics, Vol.14, No.4, 1998. pp. 537-551
- Maybee, Dandekar, Bishop (1988), "A Graph-Theoretic Method for Determining Separability of Multi-Dimensional Arrays", Manuscript, Energy Information Administration, U. S. Department of Energy.

**A Graph-Theoretic Method for Determining
Separability of Multi-Dimensional Arrays**

John S. Maybee¹
Program In Applied Mathematics
University of Colorado
Boulder, CO 80309-0526

Ramesh A. Dandekar
Yvonne M. M. Bishop
Energy Information Administration
United States Department of Energy
Washington, DC 20585

Abstract

A graph-theoretic approach is used to separate sparse multi-dimensional arrays into multiple blocks of non-zero arrays. The separability technique demonstrated in this paper has a significant application in sparse multi-dimensional data analyses. Typical application areas include: contingency table analyses, multi-variate sampling of sparse data matrices and **determining suitable complementary suppression patterns** to avoid disclosure of individual respondent's data in tabulated data.

Key Words and Phrases: Sparse data matrix, multidimensional arrays, separability, multi-variate sampling, contingency tables, **disclosure**.

¹The research of this author was supported by the Office of Naval Research under contract N00014-88-K-0087 and by the United States Department of Energy while on the sabbatical leave.

Introduction

In many situations, instead of working directly on a large sparse multi-dimensional array of data, it is advantageous to first separate the big array of data into blocks of smaller individual non-zero arrays. The separation of the sparse multi-dimensional array into its smaller counterparts not only reduces the complexity of the analytical task, but in many instances saves on the computational resources. This is easy to visualize by considering, as an example, the following three dimensional 5 x 5 x 3 Boolean array. In this array the value one indicates a non-zero count in the cell.

1	0	0	0	1	0	1	1	1	0	1	0	0	0	1
0	1	1	1	0	1	0	1	1	1	0	0	1	1	0
1	0	0	0	1	0	1	1	1	0	0	0	0	0	0
0	1	1	1	0	1	1	1	1	1	0	1	1	1	0
1	0	0	0	1	0	1	1	1	0	1	0	0	0	1

In this array each 5 x 5 block is labeled from left to right with the third subscript. The first two subscripts are row and column subscripts, respectively, within each block. By using the graph-theoretic method from this paper, the Boolean array could be rearranged in to two or more blocks of non-zero cells to give the following appearance:

1	1	0	0	0	1	1	0	0	0	0	0	1	1	1
1	1	0	0	0	1	1	0	0	0	0	0	1	1	1
1	1	0	0	0	0	0	0	0	0	0	0	1	1	1
0	0	1	1	1	0	0	1	1	1	1	1	1	1	1
0	0	1	1	1	0	0	1	1	0	1	1	1	1	0

The rearranged array contains blocks of nonzero cells. Depending on the circumstances, increased efficiency could result by processing these individual blocks of arrays separately instead of processing the original sparse data matrix in its entirety. In some other instances, such as in the analysis of contingency tables, unless the separability is detected incorrect degrees of freedom will be associated with models fitted to the data.

Mathematical Background

A multidimensional array is a function of several integer variables. Therefore, it is appropriate to write such an array in the functional form $A = A(i_1, i_2, \dots, i_p)$. Here, each i_r is an integer from the set $I_k = \{1, 2, \dots, N_k\}$. Therefore the domain of the function is the set $I = I_1 \times I_2 \times \dots \times I_p$ which can obviously also be regarded as a lattice of points in the p -dimensional Euclidian space R^p . Since the separability problem is concerned with the presence or absence of zero values at points of the domain of A , it is sufficient to consider a Boolean array. Thus, one may look upon A as a mapping $A: I \rightarrow \{0, 1\}$.

It is essential to regard i_k as the k -th variable in the definition of A . The ordering of the variables can be emphasized by writing the array A in the form $A_{1,2,\dots,p}(i_1, i_2, \dots, i_p)$. A convenient notation is then to write $P = (1, 2, \dots, p)$ and hence also to write $A_P(i_1, i_2, \dots, i_p)$.

We need to define connectedness and separability in general terms. Suppose Q is a proper subset of P (subsets of P will always be regarded as ordered). Define $Q' = P \setminus Q$.

Then a useful notation is to write

$$A_P(i_1, i_2, \dots, i_p) = A_{QQ'}(i_1, i_2, \dots, i_p) .$$

For example, if $p = 5$, $P = (1, 2, 3, 4, 5)$, and $Q = (2, 3, 5)$, then $Q' = (1, 4)$ and

$$A_{QQ'}(i_1, i_2, i_3, i_4, i_5) = A_{(2,3,5)(1,4)}(i_1, i_2, i_3, i_4, i_5) .$$

With the above notation two elements of A are called Q -adjacent if they have the same Q subscripts and both have the value one. Thus the elements

$$A_{(2,3,5)(1,4)}(i_1, i_2, i_3, i_4, i_5) \text{ and}$$

$$A_{(2,3,5)(1,4)}(j_1, j_2, j_3, j_4, j_5)$$

are $(2, 3, 5)$ -adjacent if $i_2 = j_2, i_3 = j_3, i_5 = j_5$ and both of these elements have the value one.

The algorithm to be presented below is concerned with the following rather general situation. Suppose Q_1, Q_2, \dots, Q_r are each proper subsets of P . (Usually one will have $Q_1 \sqcup Q_2 \sqcup \dots \sqcup Q_r = P$). Two elements of A are called (Q_1, Q_2, \dots, Q_r) -adjacent if they each have the value one and they are Q_k -adjacent for at least one k satisfying $1 \leq k \leq r$.

It is convenient to denote by B , the subset of points of I at which the array A takes on the value one. The array is then called (Q_1, Q_2, \dots, Q_r) -connected if any two points in B can be regarded as the endpoints of a finite sequence of points in B with the property that each pair of successive points in the sequence is (Q_1, Q_2, \dots, Q_r) -adjacent. If the array A is not (Q_1, Q_2, \dots, Q_r) -connected, it is called (Q_1, Q_2, \dots, Q_r) -separable.

Now it is well known (see, for example, [V62], [BHM80], and [GLM81]) that separability problems can generally be solved by reducing them to a graph-theoretic form. This is because very efficient algorithms exist for finding the components of a graph. Thus a graph $G = (V, E)$ is constructed for the above separability problem as follows.

The vertex set $V = B$. Two elements in V are adjacent (i.e., define an element of E) if, and only if, they are Q_i -adjacent for at least one value of i , $1 \leq i \leq r$. In this way, the graph G has the same adjacencies as the array A itself and can be labeled $G(Q_1, Q_2, \dots, Q_r)$. We will show that A is (Q_1, Q_2, \dots, Q_r) -separable if, and only if, $G(Q_1, Q_2, \dots, Q_r)$ is not connected and, moreover, the components of A , when A is separable, are completely determined by the components of $G(Q_1, Q_2, \dots, Q_r)$.

Mathematical Proof

Theorem: The array A is (Q_1, Q_2, \dots, Q_r) -connected if and only if $G(Q_1, Q_2, \dots, Q_r)$ is a connected graph.

(Recall that a graph G is called connected if, given any two vertices x and y , there exists a path in G joining x and y . This means a sequence $[x_1, \dots, x_p]$ of vertices exists in G with $x_1 = x$, $x_p = y$ with the property that each of the edges $[x_i, x_{i+1}]$, $i = 1, \dots, p-1$, belongs to E .)

Proof: Suppose that A is (Q_1, Q_2, \dots, Q_r) -connected. This means that given any two points, i and j in B , there is a finite sequence $[i_1, i_2, \dots, i_p]$ of points in B , with $i_1 = i$, $i_p = j$ and i_k, i_{k+1} Q_x -adjacent for some x , for $k = 1, 2, \dots, p-1$. By the definition of $G = (V, E)$, $V = B$, hence the sequence of i_1, \dots, i_p corresponds to a sequence of vertices x_1, \dots, x_p in G with $x_1 = x$ which corresponds to i , and $x_p = y$ which corresponds to j . Since i_k and i_{k+1} are Q_x -adjacent in A , they correspond to an edge $[x_k, x_{k+1}]$ in E . This being true for each value $k = 1, 2, \dots, p-1$, it follows that x and y are connected in G if i and j are connected. Thus, the connectedness of $G(Q_1, Q_2, \dots, Q_r)$ follows from that of A since the elements of V are in one-to-one correspondence to those of B .

For the converse, suppose $G(Q_1, Q_2, \dots, Q_r)$ is a connected graph. Let x and y be distinct vertices of G and $[x_1, \dots, x_p]$ a path in G joining them. Let i_1, \dots, i_p be the corresponding points of the set B , $i_1 = i$ corresponding to x and $j = i_p$ corresponding to y .

Since $[x_k, x_{k+1}] \in E$, $k = 1, 2, \dots, p-1$, the points i_k, i_{k+1} must be Q_i -adjacent for some i for $k = 1, 2, \dots, p-1$. Thus i and j are connected in A . Again it follows that A is (Q_1, Q_2, \dots, Q_r) -connected because of the one-to-one correspondence between V and B . \square

Next, observe that a connected (Q_1, Q_2, \dots, Q_r) component of the array A is defined as a maximal (Q_1, Q_2, \dots, Q_r) -connected subset if A is separable. Likewise, a connected component of $G(Q_1, Q_2, \dots, Q_r)$ is a maximal connected subset. The components of G and A must therefore correspond both in number and size as a result of the above proof that the two concepts of connectedness are equivalent.

Since the graph we are using to investigate separability may be used to investigate separability for a rectangular matrix, we must explain why it is consistent with known methods for solving this problem. Let $A = [a_{ij}]$ be an m by n matrix. It is well known that one can associate with A a bipartite graph $G(A) = (R, C, E)$ as follows. The vertex set consists of disjoint sets $R = \{r_1, r_2, \dots, r_m\}$ and $C = \{c_1, c_2, \dots, c_n\}$ with an edge $[r_i, c_j] \in E$ if and only if a_{ij} is not zero. Then A is separable if and only if $G(A)$ is not connected and when this happens, the nonzero blocks of A are in one to one correspondence both in number and size to the components of $G(A)$. By this we mean that, if $G_0(A)$ is a component of $G(A)$ containing m_0 elements of R and n_0 elements of C , then there will correspond a block of A of size m_0 by n_0 .

The graph $G(A)$ is not the graph used in our method. We are in the matrix case $G((1), (2)) = (V, E)$ where V contains μ elements which are in one to one correspondence with the nonzero entries of A , and there is an edge in E joining the vertices (i_0, j_0) and (i_1, j_1) and only if either $i_0 = i_1$ or $j_0 = j_1$. It is obvious that $G((1), (2))$ is the line graph of the bipartite graph $G(A)$. Our method works because $G((1), (2))$ is connected if and only if $G(A)$ is connected and when $G(A)$ is not connected, both graphs have the same number of components. If $G_0((1), (2))$ is a component of $G((1), (2))$, then the corresponding block of A has the same number of nonzero entries as the number of vertices in the component $G_0((1), (2))$.

In general, for a very large (and not so sparse) matrix A , the graph $G((1), (2))$ has many more vertices than the graph $G(A)$ which always has $m+n$ vertices. Therefore, it is usually more efficient to use $G(A)$ in the matrix case. Of course $G(A)$ can be recovered

from $G((1), (2))$ and vice versa. The point is that our algorithm is not the most efficient one for the matrix case.

For an array of several dimensions we do not have any general method of making our algorithm more efficient. How does one relate $G((1,3,4), (2,5))$ to graph having a fixed number of vertices? Fortunately in most applications in physics and statistics the multidimensional arrays that occur in practice are not too large. Hence neither are the graphs we use. Nevertheless it would be a big improvement if our graphs could be related in a systematic way with graphs having a fixed number of vertices.

Potential Applications

(A) Multi-dimensional Contingency Tables

The separability of a multi-dimensional contingency table is important in two situations. In the first, it affects the computation of degrees of freedom used to assess the goodness-of-fit of a hierarchical model. In the second, it imposes constraints on the allocation of sample sizes to strata in multi-attribute sampling.

Contingency tables are frequently analyzed by fitting hierarchical log-linear models. Maximum likelihood cell estimates are obtained by the iterative proportional fitting algorithm. A measure of goodness-of-fit between the estimates and the observed cell counts is computed. This measure, together with the appropriate degrees of freedom, is used to provide the probability that the observed data could have arisen from the specified model. (For details see for example BFH75)

The presence of occasional cells with structural zeroes does not require any alteration of the usual iterative algorithm in order to obtain estimates for the non-zero cells. The degrees of freedom are easily modified.

Degrees of freedom can be defined as "total number of cells estimated" minus "number of parameters fitted." The total number of cells estimated is then counted as the total number of elementary cells in the array minus the number of cells with structural zeroes. The number of parameters fitted is equal to the number of non-redundant cells in the minimum set of marginal elements comprising the sufficient statistics. This is unaltered from the number for a complete table when these sufficient statistics do not contain elements with zero entries. The number of parameters is reduced by the number of empty cells in the sufficient statistics if they occur.

The difficulty arises when the table is so sparse that (1) the elementary cells can be separated into two or more subsets, and (2) the sufficient statistics computed for each subset of elementary cells constitute a subset of the sufficient statistics for the complete array.

Example for two-dimensional arrays

The rows and columns of a separable two-dimensional array can always be arranged so that the non-zero cells form two or more diagonal blocks as in Table 1.

Table 1

					Totals
x_{11}	x_{12}	0	0	0	$x_{1.}$
x_{21}	x_{22}	0	0	0	$x_{2.}$
0	0	x_{33}	x_{34}	x_{35}	$x_{3.}$
0	0	x_{43}	x_{44}	x_{45}	$x_{4.}$
$x_{.1}$	$x_{.2}$	$x_{.3}$	$x_{.4}$	$x_{.5}$	$x_{..}$

Here, the x_{ij} are positive cell entries, and the marginal totals $\{x_{i.}\}$ and $\{x_{.j}\}$ are all positive. The marginal totals are the elements of the sufficient statistics for the model of independence. There is one redundancy because $\sum x_{i.} = \sum x_{.j}$. So the number of parameters fitted by the usual strategy that ignores separability is $(4 + 5 - 1) = 8$. The number of non-zero cells is 10, so the usual approach yields (erroneously) $10 - 8 = 2$ degrees of freedom.

The existence of separability means that the 4 positive cells in the top left-hand corner comprise a component separate from the remaining positive cells. Estimates under the independence model can be derived from the 4-cell component using the same marginal values $x_{1.}, x_{2.}$ for variable 1 and $x_{.1}, x_{.2}$ for variable 2, as would be used when fitting the model to the whole array. The model for this component alone has 1 degree of freedom for assessing goodness-of-fit. Repeating the procedure with the 6 positive cells in the bottom right-hand corner, we have 2 degrees of freedom. Thus by separating the components of the table, it is apparent that the correct degrees of freedom are 3, not the 2 computed when separability was ignored. The reason for this difference in the number of degrees of freedom is that the separability of the sufficient statistics imposes an additional constraint, namely $x_{1.} + x_{2.} = x_{.1} + x_{.2}$. Thus the number of independent parameters fitted is reduced from 8 to 7, and subtracted from the 10 fitted cells yields the correct 3 degrees of freedom.

(B) Allocating strata sample sizes.

When a survey is conducted to obtain estimates for several variables, a variety of sampling strategies have been proposed. If a master frame list with attributes correlated with the variables of interest is available, the following procedure is a contender to minimize the number of respondents required to attain specified coefficients of variation. The members of the frame are segregated into strata according to magnitude on each attribute; thus, they are categorized into a multidimensional array. Then the total sample size is allocated to the strata for each attribute, using classical univariate methods. These strata-specific sample sizes are then allocated to elementary cells using an iterative proportional fitting algorithm. The result is fractional sample sizes in each elementary cell that mimic the correlation structure and distribution of the members of the frame. These fractional requirements are rounded to whole numbers so that samples can be randomly selected in each elementary cell.

The presence of separability in the frame array will impose further constraints on the strata-specific sample sizes. If separability is not detected, the allocation to elementary cells of the sample sizes for each attribute is not possible unless these constraints are met. In the array used as an example above, let $\{n_{i.}\}$ and $\{n_{.j}\}$ be the desired sample sizes such that $\sum n_{i.} = \sum n_{.j} = N$, and $n_{i.} \leq x_{i.}$ for all i , $n_{.j} \leq x_{.j}$ for all j . The separability imposes the additional constraint that $n_{1.} + n_{2.} = n_{.1} + n_{.2}$. Thus, it is desirable to test for separability before attempting to allocate the sample sizes to the elementary cells if the number of zeroes in the frame array is large, namely, it exceeds $r + c - 1$ in an $r \times c$ array.

(C) Complementary Suppression Patterns to Avoid Disclosure.

Detection of separability is also important when cell suppression is used to avoid disclosure of individual responses in tabulated data.

In a two-way table, for example, if one sensitive cell x_{ij} is suppressed, then other complementary cells must be suppressed to ensure that this cell is not reproducible from the marginal totals. In a complete table this is achieved by any choice of nonzero x_{im} , x_{nj} and x_{nm} . If the table is separable, it is necessary for the complementary cells to be selected from the same separate component as the suppressed cell.

Overview of the Software Algorithm

The user sees elements of the graph $G(Q_1, \dots, Q_r)$ as p-tuples of integers as explained above. These p-tuples, representing the points at which the array A takes on the value one, constitute a portion of the input data to the computer program.

In addition to the indices of the nonzero cells of the array, the user supplies:

- (a) P , the number of variables in the array.
- (b) I_1, I_2, \dots, I_p , the numbers of categories in each of the P variables.
- (c) R , the number of sets defining the type of separability to be investigated.
- (d) The number of variables in each of the sets Q_1, Q_2, \dots, Q_r .
- (e) The identification numbers of the variables constituting each of the sets.

The last three items, (c) - (e), are recognized as the descriptors of the sufficient statistics defining a specific model.

If separability is detected, and the array can be separated into components, print outs for each component includes:

- (a) The indices of the cells in the component.
- (b) The number of cells in the component.
- (c) The number of distinct categories in each of the P variables.
- (d) The number of distinct combination of categories of variables present in the component.

From this information, the user can compute the number of degrees of freedom associated with each component. An example of this is illustrated below.

The graph-theoretic algorithmic work form a set of arrays that are either adjacency lists for G whose elements are integers or are pointer lists which keep track of the location of distinct adjacency lists. These latter arrays are also simply lists of integers. Therefore, it is necessary to compress p-tuples of

integers into single integers as required by the graph-theoretic algorithm. Towards the end of the algorithm, the integers are then decomposed back into p-tuples of integers required for the output. The algorithm for compression is reverse lexicographic ordering (first index changes most rapidly, second next, etc.). The decomposition is implemented, therefore, as the inverse of reverse lexicographic ordering.

The algorithm used for finding components of the graph G is Nonrecursive Depth First Search which, as the name implies, is a fast depth first search algorithm designed to create a spanning tree. This is the fastest known algorithm for finding components of a graph.

Sample solution for three-dimensional array

Consider as an example 5x5x3 Boolean array mentioned earlier. In the array the value one indicates a non-zero count in the cell.

1 0 0 0 1	0 1 1 1 0	1 0 0 0 1
0 1 1 1 0	1 0 1 1 1	0 0 1 1 0
1 0 0 0 1	0 1 1 1 0	0 0 0 0 0
0 1 1 1 0	1 1 1 1 1	0 1 1 1 0
1 0 0 0 1	0 1 1 1 0	1 0 0 0 1

This array is read as follows. Each 5x5 block is labeled from left to right with the third subscript. The first two subscripts are row and column subscripts, respectively, within each block. The number of variables P is 3 and the number of categories are $I_1 = 5, I_2 = 5, I_3 = 3$. The non-zero cells in the array are represented by the sequence of 3-tuples or indices as follows, in any order.

```

1 1 1
3 1 1
.
.
.
.
5 5 3

```

To test this array for $\{(1,2), (1,3), (2,3)\}$ -separability, one sets $r = 3, Q_1 = (1,2), Q_2 = (1,3), Q_3 = (2,3)$. The sample input is in Diagram 2. The sample printout is in Diagram 3, and we find that the array is separable into two components.

The number of parameters associated with this model are (number of elements in two-dimensional sets) - (number of elements in over-lapping one-dimensional sets) + (common grand total). For component one, this gives

$$(6 + 4 + 5) - (3 + 2 + 2) + 1 = 9.$$

The number of positive cells in the component is 10. Therefore, the number of degrees of freedom for component one is $10 - 9 = 1$. For component two we have similarly for the number of parameters fitted

$$(19 + 11 + 9) - (5 + 5 + 3) + 1 = 27.$$

Together with the 29 positive cells, this gives $29 - 27 = 2$ degrees of freedom.

Fitting the model for no three-factor effect to this array requires fitting the three sufficient statistics $Q_1, Q_2,$ and Q_3 . By looking at the separable components, we find that this model has 3 degrees of freedom. If we had not determined separability, we would have observed that Q_2 had one zero cell so would have computed number of parameters fitted as

$$(25 + 14 + 15) - (5 + 5 + 3) + 1 = 42$$

and the number of fitted cells as 39. This would have yielded negative degrees of freedom.

The following interchanges of category ordering enable us to visualize the structure more clearly:

Variable 1 (rows) categories 2 and 5
 Variable 2 (columns) categories 2 and 5
 Variable 3 (block) categories 2 and 3.

The appearance of the array is then of diagonal form in each block as follows:

1	1	0	0	0	1	1	0	0	0	0	0	1	1	1
1	1	0	0	0	1	1	0	0	0	0	0	1	1	1
1	1	0	0	0	0	0	0	0	0	0	0	1	1	1
0	0	1	1	1	0	0	1	1	1	1	1	1	1	1
0	0	1	1	1	0	0	1	1	0	1	1	1	1	0

The first separable component consists of the upper left hand positive cells in the first and second blocks.

Conclusion

The graph-theoretic approach demonstrated in this paper has many applications in sparse multi-variate data analysis.

References

- [BFH75] Y. Bishop, S. Fienberg, P. Holland, Discrete Multivariate Analysis: Theory and Practice, MIT Press, Cambridge, MA.
- [BHM80] R. Brualdi, F. Harary, Z. Miller, Bigraphs versus digraphs via matrices, J. Graph Theory.
- [GLM81] H. Greenberg, R. Lundgren, J. Maybee, Graph theoretic methods for the qualitative analysis of rectangular matrices, SIAM J. on Algebraic and Discrete Methods.
- [V62] R. Varga, Matrix Iterative Analysis, Prentice Hall, Englewood Cliffs, NJ.

Synthetic Tabular Data - A Better Alternative to Complementary Data Suppression - Original Manuscript Dated December 2001

Ramesh A. Dandekar

Energy Information Administration, Department of Energy, Washington DC, USA
Ramesh.Dandekar@eia.doe.gov

Abstract. Complementary cell suppression is often used for statistical disclosure limitation in tabular data, and in particular for magnitude data such as aggregate economic statistics. Cell suppression results in missing data. This can complicate and sometimes thwart thorough analysis. Suppressed entries could be replaced by interval estimates of their hidden values, but this too presents analytical challenges. Expected values, often close to original values, threaten disclosure. These approaches are also limited in the ability to preserve additive structure. We demonstrate the use of synthetic tables to prevent disclosure of sensitive information. Synthetic tables are relatively easy to generate and provide significantly more information, compared to conventional tables protected by complementary data suppression techniques. Relatively low computational burden associated with synthetic tables make them much more attractive relative to other methods of tabular data protection. The accuracies of synthetic tabular cells are easy to control, making them a useful tool for dissemination of statistical information. The primary criterion for a valid synthetic table is that the value presented for a disclosure cell lie outside the cell's disclosure interval. The secondary objective is to hold a maximum number of synthetic cells to their true values. This is accomplished via iterative refinement of synthetic tabular cells using a variation of classical gradient search in a manner analogous to partial cell suppression.

1. Introduction

Procedures used to protect sensitive cells in tabular data have slowly evolved over the last four decades. From the very beginning federal statistical agencies realized that just withholding the value for sensitive cells was not a good enough strategy to protect sensitive information in tables containing marginal entries. The concept of complementary data suppression was, therefore, introduced and practiced ever since to protect sensitive cells from disclosure. At first it was thought that suppressing at least two cells in any given row or column offered adequate protection from disclosure for sensitive cells. Over the years statistical offices realized that the minimum of two suppressed cells in a row or column strategy did not work as well for sparse and multi-dimensional tables. Procedures based on network flow as well as linear programming techniques were introduced to increase the reliability and the efficiency of complementary data suppression procedures. In recent years, it has become increasingly

obvious that by using commonly practiced missing data analysis techniques, probabilistic estimates for the suppressed tabular cells can be derived with great accuracy, limiting the applicability of complementary cell suppression techniques for statistical disclosure prevention.

In this paper, we demonstrate an entirely different approach to the age-old objective of protecting sensitive data cells from disclosure in complex multi-dimensional link tables containing hierarchical structures. In our approach, we completely discard the notion of complementary data suppression and in its place advocate the use of synthetic tables to disseminate statistical information. Our proposed method completely eliminates the information loss associated with complementary data suppression procedures. Our proposed approach requires a fraction of the computational resources required by the complementary cell suppression methods, and offers multiple alternatives to produce synthetic data by appropriate selection of an objective function that satisfies a wide variety of requirements for different statistical offices.

2 Basic Concept

The basic concept of generating synthetic tabular data that closely mimics the overall characteristics of real tabular data is quite straightforward. In synthetic tables, values for all the sensitive cells are kept at a safe distance away from their true cell values. The remaining non-sensitive cell values in the table are then adjusted from their true values as little as possible by using some predetermined criteria to make the table contents additive in all the dimensions as in the original table. The optimal mathematical structure of a synthetic table is relatively easy to specify by using mixed integer linear programming formulation¹.

Minimize $SUM [c_i (y_i^+ + y_i^-)]$

Subject to

$$M (y_i^+ - y_i^-) = 0$$

$$0 \leq y_i^+ \leq UB_i$$

$$0 \leq y_i^- \leq LB_i$$

$$y_{ik}^+ \geq BOUND_{ik} * I_{ik}$$

$$y_{ik}^- \geq BOUND_{ik} * (1 - I_{ik})$$

where

I_{ik} is a binary zero/one variable

$BOUND_{ik}$ is confidentiality bound for sensitive cell ik

ik ($k = 1, \dots, p$) p sensitive cells

$i = 1, \dots, n$ n non-zero table cells

y_i^+ = positive adjustment to cell value

y_i^- = negative adjustment to cell value

UB_i and LB_i Upper and lower cell bounds

c_i = cost function.

¹ Fischetti, Salazar naming conventions are used

Five different cost functions are commonly used. They are: 1) constant, 2)log(value), 3) value, 4) 1/value, and 5) log(value)/value, where 'value' denotes the cell value.

It is generally known that the mixed integer linear programming formulations are suitable only to solve small problems. We, therefore, propose using a simplified linear programming formulation, which could be used in practice to generate large complex synthetic tables containing linked hierarchical structures. In our simplified linear programming formulation, the binary integer variable is replaced by a binary constant. We assign the binary constant the value of zero or one prior to finding a linear programming solution by using the following simple heuristic:

- Arrange all the sensitive cells in the table, in an increasing order of magnitude of the cell values.
- Using an alternate sequence, assign the value of zero and one to the binary constant associated with each sensitive cell, except when the values of the sensitive cells are identical.
- When the values of the sensitive cells are identical, assign the same binary constant value to them all.
- To ensure that our select direction of adjustment is followed, we assign relatively high value to the cost coefficient associated with sensitive cell adjustment component in the opposite direction.

3. Illustrative Example

We use a hypothetical 3 dimensional table, containing 10 columns, 6 rows and 4 levels to demonstrate generation of a synthetic table to disseminate statistical information. Our table contains 191 non-zero cells, of which 24 cells are sensitive cells. The location of the sensitive cells, their cell values and required cell protection values are as follows:

SENSITIVE CELLS AND PROTECTION REQUIREMENT

Col	Row	Lev	Val	Prot	Col	Row	Lev	Val	Prot	Col	Row	Lev	Val	Prot
2	1	1	714	39	2	1	2	539	59	2	4	3	644	35
4	1	2	70	7	4	1	3	614	34	4	2	2	786	87
4	2	3	928	51	4	4	2	382	42	4	6	2	1238	17
5	1	1	140	7	6	2	2	1074	59	6	3	2	544	30
7	1	3	549	61	7	3	2	631	70	7	5	2	726	40
7	5	3	134	7	8	1	3	92	10	8	4	2	1050	58
8	5	1	664	36	8	5	4	664	36	9	2	1	1042	57
9	3	3	820	91	9	5	2	1598	88	9	5	4	1598	88

By using the traditional complementary cell suppression technique, our test example requires 39 complementary cell suppressions to adequately protect 24 sensitive cells. The entire table contents are displayed below. In the table, the complementary suppression cells are marked by a symbol c next to the cell value. The sensitive cells are identified by a symbol w next to the cell value. In addition, to the symbols w and c, we use gray shades to identify suppressed cells. The gray shades are to emphasize that the numeric values for these cells are prevented from a display to data users. The

complementary cell suppression technique in this example results in a significant amount of information loss, rendering the table useless for many practical applications.

DATA SUPPRESSION -- (10x6x4) TABLE

6764	714w	3356	4067c	140w	--	3932	1478c	--	20451
1994c	--	5593	--	3022	3504c	--	3220	1042w	18375
3744c	--	3708	--	3678c	2502c	--	--	--	13632
2810c	10632c	--	2445c	--	--	2313	2978	7548c	28726
3682	--	--	--	4667	1988c	1748	664w	--	12749

18994	11346	12657	6512	11507	7994	7993	8340	8590	93933
--	539w	--	70w	--	7472	715c	3832	--	12628
2253c	--	4948	786w	472	1074w	1830	5030	--	16393
640c	--	986	--	--	544w	631w	48c	750c	3599
1334c	--	1016	382w	3175	3302c	3803	1050w	--	14062
1648	2814	--	--	--	2102c	726w	--	1598w	8888

5875	3353c	6950	1238w	3647	14494	7705	9960	2348	55570
--	3552c	3476	614w	1916c	1131	549w	92w	1772	13102
--	--	3222	928w	--	--	308c	429	87c	4974
4145	--	--	3692	2115c	4196	414c	3804c	820w	19186
5995	644w	--	--	2410	1677c	--	1912c	4134c	16772
2016	--	--	2212	2826	1627c	134w	--	--	8815

12156	4196c	6698	7446c	9267	8631	1405	6237	6813	62849

6764	4805	6832	4751	2056	8603	5196	5402	1772	46181
4247	--	13763	1714	3494	4578	2138c	8679	1129c	39742
8529	--	4694	3692	5793	7242	1045c	3852c	1570c	36417
10139	11276	1016	2827	5585	4979	6116	5940	11682	59560
7346	2814	--	2212	7493	5717	2608	664w	1598w	30452

37025	18895	26305	15196	24421	31119	17103	24537	17751	212352

To generate a synthetic table that retains most of the statistical characteristics of the original unsuppressed table above, we use the linear programming heuristic formulation specified earlier. We make use of a cost function, which is proportional to the cell value, to identify the required controlled adjustments to select few non-sensitive cell values. The cell value adjustments are such that resulting table is additive in all the dimensions and at the same time the published estimates for the sensitive cells are kept outside of their disclosure range. The table below summarizes the cell locations and magnitude of required controlled adjustments to the true cell values. We have highlighted sensitive cells, in addition to marking them with symbol w, so that readers can easily verify that all adjustments to sensitive cells are beyond their respective confidentiality bounds.

CONTROLLED ADJUSTMENTS (10x6x4) TABLE

--	39w	--	-41	-8w	--	--	10	--	--
--	--	--	--	9	22	--	26	-57w	--
--	--	--	--	8	-8	--	--	--	--
5	-35	--	-38	--	--	11	--	57	--
-5	--	--	--	--	-14	55	-36w	--	--
=====									
--	4	--	-79	9	--	66	--	--	--
--	-62w	--	7w	--	--	55	--	--	--
--	--	--	78w	-9	-59w	--	-10	--	--
--	--	--	--	--	30w	-70w	-48	88	--
-5	--	--	38w	--	-80	-11	58w	--	--
5	14	--	--	--	109	-40w	--	-88w	--
=====									
--	-48	--	123w	-9	--	-66	--	--	--
--	9	--	34w	8	--	-55w	-10w	14	--
--	--	--	51w	--	--	-161	-16	126	--
--	--	--	-41	-8	-22	70	84	-83w	--
--	35w	--	--	--	80	--	-58	-57	--
--	--	--	-88	--	-58	146w	--	--	--
=====									
--	44	--	-44	--	--	--	--	--	--
--	-14	--	--	--	--	--	--	14	--
--	--	--	129	--	-37	-161	--	69	--
--	--	--	-41	--	--	--	36	5	--
--	--	--	--	--	--	--	--	--	--
--	14	--	-88	--	37	161	-36w	-88w	--
=====									
--	--	--	--	--	--	--	--	--	--

After applying the linear programming-based control adjustments to the original table, our synthetic table will appear as shown below. Once again, in the following table we highlight all the sensitive cells for the ease of understanding of our readers. In a real application synthetic tables will be published in their entirety. Depending on the accuracy of the data a synthetic table represents, statistical offices as an option might decide to attach some kind of quality indicators to select a few cells where the published cell values are beyond some pre-determined tolerance level

SYNTHETIC(10x6x4) TABLE

6764	753	3356	4026	132	--	3932	1488	--	20451
1994	--	5593	--	3031	3526	--	3246	985	18375
3744	--	3708	--	3686	2494	--	--	--	13632
2815	10597	--	2407	--	--	2324	2978	7605	28726
3677	--	--	--	4667	1974	1803	628	--	12749

18994	11350	12657	6433	11516	7994	8059	8340	8590	93933

--	477	--	77	--	7472	770	3832	--	12628
2253	--	4948	864	463	1015	1830	5020	--	16393
640	--	986	--	--	574	561	0	838	3599
1329	--	1016	420	3175	3222	3792	1108	--	14062
1653	2828	--	--	--	2211	686	--	1510	8888

5875	3305	6950	1361	3638	14494	7639	9960	2348	55570

--	3561	3476	648	1924	1131	494	82	1786	13102
--	--	3222	979	--	--	147	413	213	4974
4145	--	--	3651	2107	4174	484	3888	737	19186
5995	679	--	--	2410	1757	--	1854	4077	16772
2016	--	--	2124	2826	1569	280	--	--	8815

12156	4240	6698	7402	9267	8631	1405	6237	6813	62849

6764	4791	6832	4751	2056	8603	5196	5402	1786	46181
4247	--	13763	1843	3494	4541	1977	8679	1198	39742
8529	--	4694	3651	5793	7242	1045	3888	1575	36417
10139	11276	1016	2827	5585	4979	6116	5940	11682	59560
7346	2828	--	2124	7493	5754	2769	628	1510	30452

37025	18895	26305	15196	24421	31119	17103	24537	17751	212352

In the synthetic table, true values are published for 103 cells. For the remaining 88 cells (including sensitive cells), the published cell values are altered slightly from their true values to protect the sensitive cell values from being disclosed within an allowable protection interval. Most of the cell values of the marginal cells are unaffected in the synthetic table. In addition, the table values are additive in all the dimensions.

4. Multi Dimensional Linked Tables

The linear programming heuristic identified above, to generate synthetic tabular data, is applicable to protect sensitive information in all the generic, multi-dimensional, linked tables containing hierarchical structure. We next provide the overall performance statistics for synthetic tables generated by using two different test examples of multi-dimensional linked tables. These two test cases were created by this author to enable testing of algorithm developed by Fischetti and Salazar, to generate optimum complementary cell suppression pattern.

The first test example consists of 2, five-dimensional linked sections of a six dimensional table (6x4x16x4x4x4). The table contains 1254 non-zero cells. Of this total, 1089 cells are non-sensitive and 165 cells are sensitive. Fischetti and Salazar determined that the

optimum complementary cell suppression results in 419 suppressed cells, which is 34% of total non-zero cells.

The second example consists of 4, five-dimensional linked sections of a nine dimensional table (4*29*3*4*5*6*5*4*5). The table contains 1141 non-zero cells, of which 831 cells are non-sensitive and 310 cells are sensitive. Fischetti & Salazar determined that the optimum complementary cell suppression results in 491 suppressed cells, which is 43% of total non-zero cells.

The synthetic tables generated by using these two test examples provide additive tables containing cell values for all the non-zero cells in the original test examples. In the following two tables we summarize the overall performance statistics of change from the true value of nonzero synthetic cells by ten different percent change from true value categories. We use five different cost functions that are commonly used in tabular cell protection to demonstrate five different possible formulations for synthetic tables.

NUMBER OF CELLS BY PERCENT CHANGE CATEGORY²

2 Sections Of Six Dimensional Linked Table

Percent change from true value	cost function used for optimization				
	const	log(value)	value	1/value	log(value)/value
.00- .10	691{ 55.3%}	716{ 57.5%}	749{ 60.4%}	720{ 57.5%}	687{ 54.8%}
.10- .50	189{ 70.4%}	154{ 69.8%}	120{ 70.1%}	231{ 75.9%}	254{ 75.1%}
.50- 1.00	91{ 77.7%}	72{ 75.6%}	37{ 73.1%}	47{ 79.6%}	56{ 79.6%}
1.00- 1.50	38{ 80.7%}	27{ 77.8%}	41{ 76.4%}	22{ 81.4%}	28{ 81.8%}
1.50- 2.00	22{ 82.5%}	33{ 80.4%}	22{ 78.1%}	14{ 82.5%}	14{ 82.9%}
2.00- 5.00	52{ 86.6%}	52{ 84.6%}	63{ 83.2%}	47{ 86.3%}	42{ 86.3%}
5.00- 10.00	73{ 92.5%}	88{ 91.7%}	98{ 91.1%}	119{ 95.8%}	100{ 94.3%}
10.00- 15.00	58{ 97.1%}	56{ 96.1%}	51{ 95.2%}	51{ 99.8%}	69{ 99.8%}
15.00- 30.00	19{ 98.6%}	24{ 98.1%}	30{ 97.7%}	2{100.0%}	3{100.0%}
30.00-100.00	17{100.0%}	24{100.0%}	29{100.0%}	0{100.0%}	0{100.0%}
unchanged cells	390{ 31.2%}	422{ 33.9%}	651{ 52.5%}	319{ 25.5%}	257{ 20.5%}

4 Sections Of Nine Dimensional Linked Table

Percent change from true value	cost function used for optimization				
	const	log(value)	value	1/value	log(value)/value
.00- .10	431{ 38.1%}	397{ 35.1%}	494{ 44.0%}	320{ 29.3%}	333{ 29.9%}
.10- .50	96{ 46.6%}	134{ 46.9%}	33{ 46.9%}	46{ 33.5%}	69{ 36.1%}
.50- 1.00	59{ 51.8%}	48{ 51.2%}	27{ 49.3%}	23{ 35.6%}	46{ 40.3%}
1.00- 1.50	35{ 54.9%}	23{ 53.2%}	29{ 51.9%}	23{ 37.7%}	27{ 42.7%}
1.50- 2.00	33{ 57.8%}	29{ 55.8%}	13{ 53.0%}	25{ 40.0%}	15{ 44.0%}
2.00- 5.00	85{ 65.3%}	90{ 63.7%}	86{ 60.7%}	83{ 47.6%}	90{ 52.1%}
5.00- 10.00	256{ 87.9%}	259{ 86.6%}	212{ 79.5%}	242{ 69.7%}	266{ 76.0%}
10.00- 15.00	55{ 92.8%}	64{ 92.3%}	57{ 84.6%}	60{ 75.2%}	62{ 81.6%}
15.00- 30.00	32{ 95.6%}	45{ 96.3%}	58{ 89.8%}	81{ 82.6%}	59{ 86.9%}
30.00-100.00	50{100.0%}	42{100.0%}	115{100.0%}	190{100.0%}	146{100.0%}
unchanged cells	353{ 31.2%}	329{ 29.1%}	453{ 40.3%}	287{ 26.3%}	302{ 27.1%}

² The numbers in the parentheses are cumulative percentages associated with the cell count.

From the summary statistics above it is clear that, by proper selection of the appropriate cost function, the controlled adjustments could be targeted to specific non-sensitive cell categories. Irrespective of the choice of the cost function, approximately 75% of the nonzero cell values in the first test case and 50% of the nonzero cell values in the second test case are altered within less than 1% of their true cell value. The synthetic cells undergoing changes in excess of 5% of true cell value are typically sensitive cells, which are otherwise blocked from publication using the complementary cell suppression method.

The quality of cell level information from the synthetic table could be conveyed to data users by using different strategies. As an option, a quality indicator, such as g (good), f (fair), and p (poor) could be assigned to each synthetic cell to inform the data user of the level of accuracy of information contained in each synthetic cell. Other options include: (1) providing overall percent accuracy of the published information, or (2) dividing the cells in multiple size categories and providing overall percent accuracy for each size category separately.

We have used only five basic cost functions to demonstrate the synthetic data generation technique in the linear programming environment. We, however, believe that a much wider spectrum of cost functions is readily available to the potential practitioner of synthetic tables to get a wide variety of desired results.

5. Iterative Refinement of LP Solution

The primary objective of the synthetic table is to provide estimates for sensitive cells, which are outside the disclosure limits. The secondary objective is to hold a maximum number of synthetic cells to their true cell values. This could be accomplished via an iterative refinement of the linear programming solution as follows:

- Exclude all the synthetic cells that are at the true cell value from future LP formulation.
- For the remaining non-sensitive synthetic cells, replace the original cost function with the new cost function, which is an inverse proportion of change in the cell value in the previous LP solution.
- Find the revised linear programming solution to generate a new subset of synthetic cell values.

In theory, multiple iterations could be performed to successively increase the number of synthetic cells that are at their true cell values. However in practice, by increasing the number of iterations estimates for more and more sensitive cells fall within their respective disclosure limits.

The table below summarizes the overall statistics for the second test example, after single iterative refinement of the LP heuristic.

Second Test Example After Single Iterative Refinement

Percent change from true value	c o s t f u n c t i o n u s e d f o r o p t i m i z a t i o n				
	const	log(value)	value	1/value	log(value)/value
.00- .10	669{ 58.6%}	649{ 57.0%}	625{ 55.1%}	603{ 53.4%}	567{ 50.0%}
.10- .50	89{ 66.4%}	94{ 65.3%}	73{ 61.6%}	100{ 62.3%}	83{ 57.4%}
.50- 1.00	38{ 69.8%}	44{ 69.2%}	45{ 65.5%}	32{ 65.1%}	46{ 61.4%}
1.00- 1.50	31{ 72.5%}	22{ 71.1%}	26{ 67.8%}	18{ 66.7%}	28{ 63.9%}
1.50- 2.00	29{ 75.0%}	15{ 72.4%}	10{ 68.7%}	15{ 68.0%}	24{ 66.0%}
2.00- 5.00	60{ 80.3%}	84{ 79.8%}	81{ 75.8%}	69{ 74.1%}	95{ 74.4%}
5.00- 10.00	103{ 89.3%}	100{ 88.6%}	156{ 89.6%}	139{ 86.4%}	166{ 89.1%}
10.00- 15.00	33{ 92.2%}	57{ 93.6%}	40{ 93.1%}	43{ 90.3%}	29{ 91.6%}
15.00- 30.00	61{ 97.5%}	27{ 96.0%}	38{ 96.5%}	38{ 93.6%}	35{ 94.7%}
30.00-100.00	28{100.0%}	46{100.0%}	40{100.0%}	72{100.0%}	60{100.0%}
unchanged cells	543{ 47.6%}	556{ 48.9%}	565{ 49.8%}	498{ 44.1%}	513{ 45.3%}

From the table above, it is clear that the number of synthetic cells at their true cell value increases dramatically

6. Conclusion

Synthetic tabular data offers a significantly more attractive option for dissemination of statistical data containing sensitive information than conventional complementary cell suppression. Conventional complementary cell suppression methods result in too significant an amount of information loss, irrespective of how close one attempts to get to the optimum cell suppression choice. The overall information generated by using the complementary cell suppression method fails to compare favorably in regard to the practical utility of information provided by synthetic tables. The computational resources required to generate a synthetic table is a very small fraction of the resources required to generate a table protected by the complementary cell suppression method.

In this paper we have demonstrated a simple heuristic to generate synthetic tabular data. The heuristic is based on the linear programming formulation. However, we believe that computational techniques, such as iterative proportional fitting and the EM algorithm, could also be used effectively to generate synthetic tabular data.

References

M. Fischetti, J. J. Salazar, "Models and Algorithms for Optimizing Cell Suppression Problem in Tabular Data with Linear Constraints", working paper, 1998..