FCSM/CDAC Disclosure Limiting Auditing Software: DAS

Mark A. Schipper
Ruey-Pyng Lu
Energy Information Administration

BTS Confidentiality Seminar Series
June 11, 2003
To protect confidentiality, agencies suppress table cells that might reveal individual data. Software exists to select cells for suppression, provides no evaluation (http://www.eia.doe.gov/oss/disclosure.html).

Auditing finds the lower and upper bounds on the values of a withheld (suppressed) cell.

EIA lead an inter-agency project to prepare table auditing software, produced FCSM DAS.
Common Problem Seeking A Common Solution

- Seven Agencies Funded Software ($250k)
  - Bureau of Labor Statistics
  - Bureau of Economic Analysis
  - Bureau of the Census
  - National Center for Education Statistics
  - Internal Revenue Service
  - National Science Foundation
  - Energy Information Administration
Planned Uses of DAS

• Bureau of Labor Statistics (BLS)
  – DAS was tested and approved for use on Windows NT
  – Future BLS Statistical Order will require the use of DAS with the following:
    • ES-202 – Covered Employment and Wages
    • OSHS - Occupational Safety and Health Statistics
    • CES - Current Employment Statistics
    • OES - Occupational Employment Statistics
Energy Information Administration
- Joint project with US Bureau of the Census working on developing auditing tools for processing of the 2002 Manufacturing Energy Consumption Survey

National Science Foundation
- Initial contact with NSF’s contractor on executing DAS software
SWP Paper 22: Report on Statistical Disclosure Limitation Methodology

• Auditing Software (mid 1970’s)
  – U.S. Census Bureau (Cox, 1980)
  – Statistics Canada (Sande, 1984)

• Audit systems produce upper and lower estimates for the suppressed cell based on linear combinations of published cells

• If software is already available, why DAS?
Software Requirements

• must be written in SAS® code, using macros language;
• must use the PROC LP (SAS/OR Software) as the linear optimizer;
• must be able to specify (as a LP model) and efficiently audit tables of up to 5 dimensions;
Requirements Continued...

- must display model results (e.g., minimum, maximum, protection range, and appropriate quality warnings) for all suppressed values;
- must use ASCII format for model statement input files; and,
- must pre-verify internal consistency of audit tables.
Modules of Software

- Front-End User Interface
- Pre-Verification of Audit Table(s)
  - Ensure Feasible Linear Model
    - Published Cell Values Sum to Published Totals
  - Rounding of Continuous Cell Values
  - Negative Cell Values
- Linear Program Modeling
- Results Display
Pre-Verification

- **Verify Aggregates**
  - Dimension Totals and Marginal Totals

- **Assume Maximum from Rounding Process**
  - \( e = \text{Max} \{e_i\} \ \forall \ i \)
  - \( e \) is dictated by the rounding process; if rounded to integer \( e = 0.5 \)
  - \( e \) is a variable defined by the user

- **Pre-Verification Satisfies Inequality**
  - \( X_i - ne \leq X \pm e \leq X_i + ne \)
### 2-D Example: Unrounded Table

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.6</td>
<td>0.6</td>
<td>2.2</td>
<td>3.4</td>
</tr>
<tr>
<td>1.0</td>
<td>1.0</td>
<td>0.6</td>
<td>2.6</td>
</tr>
<tr>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>3.0</td>
</tr>
<tr>
<td>Total</td>
<td>2.6</td>
<td>2.6</td>
<td>3.8</td>
</tr>
</tbody>
</table>
### 2-D Example: Unrounded and Suppressed Table

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.6</td>
<td>0.6</td>
<td>2.2</td>
<td></td>
<td>3.4</td>
</tr>
<tr>
<td>1.0</td>
<td>V1</td>
<td>V2</td>
<td></td>
<td>2.6</td>
</tr>
<tr>
<td>1.0</td>
<td>V3</td>
<td>V4</td>
<td></td>
<td>3.0</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>2.6</strong></td>
<td><strong>2.6</strong></td>
<td><strong>3.8</strong></td>
<td><strong>9.0</strong></td>
</tr>
</tbody>
</table>
Operations Research

• Linear Programming (LP) Model
  – Objective Min or Max $v$; Subject to:
    • $1.0 + v_1 + v_2 = 2.6$ (1)
    • $1.0 + v_3 + v_4 = 3.0$ (2)
    • $0.6 + v_1 + v_3 = 2.6$ (3)
    • $2.2 + v_2 + v_4 = 3.8$ (4)
    • $0.6 + 0.6 + 2.2 + 1.0 + 1.0 + v_1 + v_2 + v_3 + v_4 = 9.0$ (5)
    • $v \geq 0$
  – Feasible LP Model
### LP Model Solutions

<table>
<thead>
<tr>
<th></th>
<th>Maximum</th>
<th>Minimum</th>
</tr>
</thead>
<tbody>
<tr>
<td>V1</td>
<td>1.6</td>
<td>0.0</td>
</tr>
<tr>
<td>V2</td>
<td>1.6</td>
<td>0.0</td>
</tr>
<tr>
<td>V3</td>
<td>2.0</td>
<td>0.4</td>
</tr>
<tr>
<td>V4</td>
<td>1.6</td>
<td>0.0</td>
</tr>
</tbody>
</table>
### 2-D Example: Suppressed and Rounded

<table>
<thead>
<tr>
<th></th>
<th>V1</th>
<th>V2</th>
<th>V3</th>
<th>V4</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>1</td>
<td>V1</td>
<td>V2</td>
<td></td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>1</td>
<td>V3</td>
<td>V4</td>
<td></td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>Total</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td></td>
<td>9</td>
</tr>
</tbody>
</table>
Operations Research

- Linear Programming (LP) Model 1
  - Objective Min or Max v; Subject to:
    - $1 + v_1 + v_2 = 3$ (1)
    - $1 + v_3 + v_4 = 3$ (2)
    - $1 + v_1 + v_3 = 3$ (3)
    - $2 + v_2 + v_4 = 4$ (4)
    - $1 + 1 + 2 + 1 + v_1 + v_2 + v_3 + v_4 = 9$ (5)
    - $v \geq 0$
  - Infeasible LP Model 1 due to Independent Rounding!
Infeasibility via Rounding

- Adding LP Constraints (1) and (2)
  - $v_1 + v_2 + v_3 + v_4 = 4$
- Adding LP Constraints (3) and (4)
  - $v_1 + v_2 + v_3 + v_4 = 4$
- However, reducing Constraint (5) yields
  - $v_1 + v_2 + v_3 + v_4 = 3$
- Hence, the LP model is not feasible.
- What to do?
How To Ensure Feasibility?

- Accounting for Independent Rounding
  - Add Surplus and Slack Variables to LP Equality Constraints - Not Used
  - Directly Adjust Table(s) - Not Used
  - Represent Rounding Found in Each Published Cell – Option in Current Use
  - “Best Fit” table approach (Stephen F. Roehrig, Carnegie Mellon University) – Future?
From Tables to Constraints

- For each non-zero, unsuppressed cell value \( u \), create a new variable \( x \) and add the following constraint for each non-zero, unsuppressed cell.

\[
\begin{align*}
    u - e & \leq x \leq u + e \\
    \end{align*}
\]

- For withheld cells, associate a variable \( x \), constrained only by non-negativity.
New LP Model Format

<table>
<thead>
<tr>
<th></th>
<th>X1</th>
<th>X2</th>
<th>X3</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>X4</td>
<td>X5</td>
<td>X6</td>
<td></td>
<td>X11</td>
</tr>
<tr>
<td>X7</td>
<td>X8</td>
<td>X9</td>
<td></td>
<td>X12</td>
</tr>
<tr>
<td>Total</td>
<td>X13</td>
<td>X14</td>
<td>X15</td>
<td>X16</td>
</tr>
</tbody>
</table>
Revised LP Model

- Linear Programming (LP) Model 2
  - Objective Min or Max \( x \); Subject to:
    - \( x_1 + x_2 + x_3 = x_{10} \) (row 1)
    - \( x_4 + x_5 + x_6 = x_{11} \) (row 2)
    - \( x_7 + x_8 + x_9 = x_{12} \) (row 3)
    - ...and so forth
    - \( u - e \leq X \leq u + e \) or \( X \) is non-negative
  - where \( u \) denotes non-zero, unsuppressed cell values and \( e \) is the max (+) rounding value
### Revised Model Solutions

<table>
<thead>
<tr>
<th></th>
<th>Maximum</th>
<th>Minimum</th>
</tr>
</thead>
<tbody>
<tr>
<td>V1</td>
<td>2.985648844</td>
<td>2.58562E-05</td>
</tr>
<tr>
<td>V2</td>
<td>2.985648844</td>
<td>2.58562E-05</td>
</tr>
<tr>
<td>V3</td>
<td>2.985648844</td>
<td>2.58562E-05</td>
</tr>
<tr>
<td>V4</td>
<td>2.985648844</td>
<td>2.58562E-05</td>
</tr>
</tbody>
</table>
Is there a another way?

- Assuming all $e_i$'s take the maximum value has some ill effects
  - With large tables (i.e., large $n$) likely to obtain wide inequality bounds in verification and optimal solution sets (Kirkendall, Lu, Schipper, Roehrig 2001)

- Is there a better ways to assign values to $e_i$?
  - Heuristically assign a value to $e$
    - Best-Fit Approach
Directly adjust table cells in the LP model

- Goal: Produce an additive table that generates the published table, given independent rounding

“Best-Fit” table exists where objective function is the sum of absolute deviations

- Minimize \[ Z = \sum |a_{ij} - x_{ij}| \] where \( i, j \) range over table rows and columns, \( a_{ij} \) are the published values, and \( x_{ij} \) are the LP variables
Software Status

- Distributed Beta Version in August 2000 to agencies on CDAC Sub-Committee
- Demonstration at EIA – March 2, 2001
  - Test files (csv format) provided by BEA and EIA
- Potential Additions
  - Add a user-friendly display manager system
  - Add a make-tables-add function (e.g., “Best Fit”)
  - Add a non-SAS optimizer for optimization speed – CPLEX (www.cplex.com)
- Completed inter-agency agreements in August 2001 and distributed copies to those agencies.
System Requirements

• Operating Systems
  – Windows 95, 98, NT, and 2000
  – UNIX

• Operating Platforms
  – Stand-Alone PC
  – Windows “box”
  – UNIX “box”