Using BCD-CTA for difficult tables: a practical experiment with a real Eurostat table

Jordi Castro   José A. González   Anco Hundepool

Universitat Politècnica de Catalunya
Barcelona, Catalonia

Leidschendam
The Netherlands

jordi.castro@upc.edu   jose.a.gonzalez@upc.edu   hundepool@ziggo.nl

This work has been supported by grants MTM2012-31440 of the Spanish research program, and project DwB INFRA-2010-262608 of the EU FP7.

Contents

Contents

Protection of tables published at the EU-level

- The goal:
  1. At national level: NSIs collect and process the information of each member state (primary and secondary suppressions).
  2. The NSIs send their tables to Eurostat.
  3. Eurostat compiles the complete European table, with the obligation to guarantee the confidentiality of the national data.
- The solution approach of S. Giessing, A.H., J.C. (2009) was to compute rounded figures for the EU-cells at risk.
- CTA was used in one of the steps of this solution approach.

The instance that motivated this work

- ▷ A.H. working on real Eurostat dataset (EU structural business statistics).
- ▷ Three-dimensional table.
  - First: EU-member state (27 plus the EU-total).
  - Second: NACE hierarchical classification (120 codes).
  - Third: size-class (5 codes plus a total)
- ▷ Number of cells: $28 \cdot 120 \cdot 6 = 20160$.
- ▷ Number of constraints: 8280.
- ▷ A.H. found unacceptable long running CTA times for this instance.
- ▷ A.H. came to J.C. and J.A.G.: what to do?
- ▷ Solution: either use BCD-CTA or try different weights.

Contents

MILP-CTA: Example

ORIGINAL TABLE. Protection levels: $x_{23} \geq 45$ or $x_{23} \leq 35$

	Z_1	Z_2	Z_3	TOTAL
E_1	20	24	28	72
E_2	38	38	40	116
E_3	40	39	42	121
TOTAL	98	101	110	309

MILP-CTA: Example

ORIGINAL TABLE. Protection levels: $x_{23} \geq 45$ or $x_{23} \leq 35$

	Z_1	Z_2	Z_3	TOTAL
E_1	20	24	28	72
E_2	38	38	40	116
E_3	40	39	42	121
TOTAL	98	101	110	309

PROTECTED TABLE: either ... or ...

	Z_1	Z_2	Z_3	
E_1	25	24	23	72
E_2	33	38	45	116
E_3	40	39	42	121
TOTAL	98	101	110	309

Z_1	Z_2	Z_3	
15	24	33	72
43	38	35	116
40	39	42	121
98	101	110	309

Parameters of MILP-CTA

- Set of cells $a_i, i = 1, \ldots, n$.
- Set $\mathcal{S} = \{i_1, i_2, \ldots, i_s\} \subseteq \{1, \ldots, n\}$ of indices of sensitive cells.
- Linear relations $Aa = b$.
- Lower and upper protection level for each sensitive cell $i \in \mathcal{S}$: lpl_i and upl_i.
- Lower and upper bound for each cell: l_{x_i} and u_{x_i}.
- Cell weights w_i for cost of adjustment of each cell.

Aim of CTA

Find released values x such that:

- Remain near a for some distance ℓ.
- Satisfy the linear relations $Ax = b$
- Satisfy the bounds: $l_x \leq x \leq u_x$
- Satisfy the protection levels: either $x_i \geq a_i + upl_i \quad \text{or} \quad x_i \leq a_i - lpl_i$.

The optimization problem is:

$$\begin{array}{rl} \min\limits_{x} & ||x - a||_\ell \\ \text{subject to} & Ax = b \\ & l_x \leq x \leq u_x \\ & x_i \leq a_i - lpl_i \quad \text{or} \quad x_i \geq a_i + upl_i \quad i \in \mathcal{S}. \end{array}$$

The MILP-CTA problem for ℓ_1

- Defining *cell deviations* as: $z = x - a$,
- and introducing binary variables for sensitive cells: $y_i, i \in \mathcal{S}$
  $y_i = 1 \Leftrightarrow$ *upside* protected: $x_i \geq a_i + upl_i$;
  $y_i = 0 \Leftrightarrow$ *downside* protected: $x_i \leq a_i - lpl_i$.

MILP-CTA model:

$$\begin{array}{ll}
\min\limits_{z^+, z^-, y} & \sum\limits_{i=1}^{n} w_i(z_i^+ + z_i^-) \\
\text{subject to} & A(z^+ - z^-) = 0 \\
& 0 \leq z^+ \leq u_z, \quad 0 \leq z^- \leq -l_z \\
& y \in \{0,1\}^s \\
& \left.\begin{array}{lll} upl_i\, y_i & \leq z_i^+ \leq u_{z_i} y_i \\ lpl_i(1-y_i) & \leq z_i^- \leq -lz_i(1-y_i) \end{array}\right\} i \in \mathcal{S}
\end{array}$$

Contents

CD and BCD algorithms

- Coordinate descent: family of optimization algorithms that successively optimize along coordinate directions.
- Very popular in 80s and 90s.
- Recently gained reputation for approximate solutions in big-data.
- Block coordinate descent (BCD): optimize on a block of variables.
- BCD solves a sequence of subproblems on a subset of variables, the other variables kept fixed.

BCD-CTA algorithm

Step 0 Initialization. Set outer iteration counter: $t = 0$. Set initial values, hopefully feasible, to y.

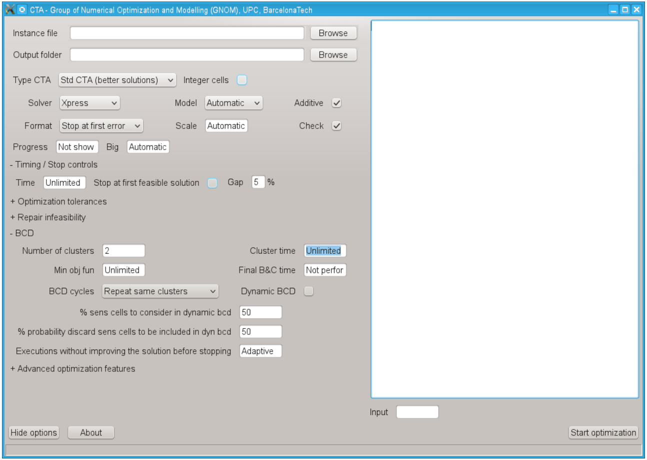
Step 1 $t = t + 1$. Set inner iteration counter $i = 0$.
Divide y into k blocks: $y = \{y^{1,i}, \ldots, y^{k,i}\}$, not necessarily of the same size.

- **Step 1.1** $i := i + 1$. Solve CTA with respect to block $y^{i,i}$, taking into account that $y^{j,i}$ is fixed for $j \neq i$.
Let $y^{i,i+1} = (y^{i,i})^*$ (the point at the optimum).
Let $y^{j,i+1} = y^{j,i}$ for $j \neq i$.
- **Step 1.2** If $i < k$ go to Step 1.1.

Step 2 Check for end conditions: if apply, stop, and return the current best solution. Otherwise, go to Step 1

BCD-CTA in τ-Argus

- BCD-CTA available in τ-Argus.
- Implemented within DwB EU project.



Contents

Preliminary results

- ▷ Three-dimensional Eurostat SBS table.
- ▷ Used both MILP-CTA and BCD-CTA as implemented in τ-Argus.
- ▷ CPLEX used as LP/MILP solver.
- ▷ Initially used $w_i = 1$ and MILP-CTA: stopped after 7692 seconds.
- ▷ Then we tried BCD-CTA: decent solution in 19 seconds.
- ▷ We tried other weights: unexpectedly much faster.

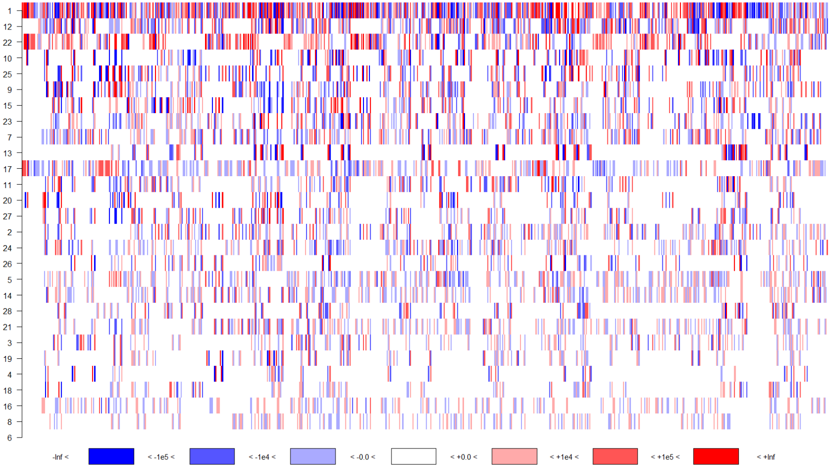
Summary of results

- Results for MILP-CTA and BCD-CTA with three different weights.
- $\bar{x}$ CTA: average deviation of CTA solution.
- $\bar{x}$ published: average deviation of published tables with full approach.
- BCD-CTA did a good job for CPU time and “$\bar{x}$ CTA”.
- Best combination for “$\bar{x}$ published” is likely $w_i = 1/\sqrt{a_i}$ with MILP-CTA.

	MILP-CTA			BCD-CTA		
w_i	CPU	$\bar{x}$ CTA	$\bar{x}$ published	CPU	$\bar{x}$ CTA	$\bar{x}$ published
1	7692	38933	64490	19	46443	91977
$1/a_i$	85	46904	89451	16	45900	93617
$1/\sqrt{a_i}$	179	40717	65297	16	42785	85098

Graphical representation of solution

Solution with one minute of BCD-CTA followed by two hours of MILP-CTA with weights $w_i = 1$.



Contents

Conclusions and future work

- ▷ Cell weights have a large implication in solutions and CPU time.
- ▷ Cell weights based on cell hierarchy (J.C., S. Giessing 2006 paper).
- ▷ BCD-CTA has enormous gain in CPU time, with a price in solution.
- ▷ BCD-CTA available in τ-Argus: it allows solution of very large and intractable tables by other approaches.
- ▷ From an optimization point of view: why the behaviour of the MILP solver changes so drastically with different weights?

Thanks for your attention!