Project N°: 262608

ACRONYM: Data without Boundaries

DELIVERABLE D11.1 – Part B

(Exploratory Report on the Future of SDC-Software Tools in General and the ECTA Method in More Details)

WORK PACKAGE 11

(Improved Methodologies for Managing Risks of Access to Detailed OS Data)

REPORTING PERIOD:	From: Month 1	To: Month 18
PROJECT START DATE:	1st May 2011	DURATION: 48 Months
DATE OF ISSUE OF DELIVERABLE:	March 2013	VERSION: 1.0
DOCUMENT PREPARED BY:	Partner 15	Destatis

Combination of CP & CSA project funded by the European Community
Under the programme "FP7 - SP4 Capacities"
Priority 1.1.3: European Social Science Data Archives and remote access to Official Statistics

Report WP11, Task 1

Part B: Software tools for assessing disclosure risk and producing lower risk tabular data

February 2013

Sarah Giessing
Statistisches Bundesamt, 65180 Wiesbaden, Germany
Email: Sarah.Giessing@destatis.de

1. Introduction

This is a report in the scope of the Data without Boundaries (DwB) project[1]. The general aim of this project is to widen and enhance access to statistical databases for researchers working through an archive or otherwise outside the safe setting of a remote access or research data centre environment.
Part B of task 1 in the project workpackage 11 is concerned with aspects of disclosure risk control of a typical scenario. According to that scenario, researchers are allowed to work with complete, original records of the data subjects to perform a specific class of analysis, i.e. to form groups of the data subjects, to count group frequencies or sum up a magnitude variable for the individual data subjects within a group. Usually such a computation is done not only for a single group, but for a system of groups, referred to as "table". The output of this kind of analysis is referred to as "frequency" or "magnitude" "tabular data". The data subject groups are referred to as "table-cells". Principally, this type of output has high disclosure risk potential (compared to other types of analysis results, like for example regression coefficients). See Hundepool et al. (2012), chapters 4 and 5 for introduction into the typical disclosure risks of tabular data. Because of this high risk potential, disclosure risks have to be assessed and to be reduced, if necessary. before this type of output can be released. Simply redesigning the analysis (*viz.*: by increasing the size of the groups of data subjects) often yields a poor balance in the reduction of disclosure risk vs. information loss.

Data producing agencies like national statistical institutes have developed disclosure limitation methods and techniques for tabular data. The most popular technique is cell suppression. Cell suppression techniques select certain sets of cells that are to be "suppressed". Suppression means that the respective output (the frequency or the sum of a magnitude variable) will not be released. The selection process must be designed in such a way that it yields an acceptable amount of disclosure risk reduction.

Some NSIs and other organisations use software tools that implement this selection process. One such tool is τ-ARGUS (c.f. Hundepool et al. (2011)) developed by Statistics Netherlands with support from several European projects. This task should assess the range of tools available to NSIs, including τ-ARGUS, and select those tools capable of cost efficient development and use by data service providers. If no suitable software tool is found, an architecture document should be supplied as a basis for future development.

This report is structured as follows: Section 2 explains methodological background and specific methodological issues relevant for the eligibility of a tool. Those tools that satisfy the respective methodological criteria for eligibility are compared from a methodology perspective in section 3.

[1] See http://www.dwbproject.org/

Results of some empirical studies comparing the performance of those tools are reported in section 4. A few practical aspects are discussed briefly in section 5. Section 6 summarizes the report and draws some conclusions.

2. Methodological Background

The methodological background and specific methodological issues that are relevant for the eligibility of a tool are thoroughly explained in chapter 4 of Hundepool et al. (2012)[2].

A cell suppression procedure involves three steps: ***(1) Primary risk assessment, (2) secondary risk assessment*** **and** ***(3) secondary cell suppression.***

The ***primary risk assessment*** step identifies the so called primary sensitive cells. If those cells would be released, it might impose a disclosure risk for the data of at least one of the data subjects in the group that this cell relates to. In understanding such disclosure risks and identifying sensitive cells, readers need to consider a range of intruder scenarios that lead to disclosure. Hundepool et al. (2012) provide many examples and illustrations of intruder scenarios and explain established systematic techniques that may be used to identify the sensitive cells. This step is comparatively simple one. Nevertheless, in the research situation considered in this report, a researcher might be tempted to use software tools for the analysis that do not provide – at least not in straightforward processing – the special meta-information needed for this step.

However, simply suppressing the values of the sensitive cells will not completely protect them when marginal totals of these cells are published, because an intruder may recalculate the dropped values by way of simple subtraction. Therefore, to completely protect sensitive cells, often one or more non-sensitive cells must be suppressed as well. The most common way is that for each primary suppressed cell there should be at least one other suppressed cell in the same row and one other suppressed cell in the same column, or, to use a more general term, in the same table relation. Note that this is a necessary criterion, but not always sufficient, when more sophisticated “attacks” (beyond a simple subtraction) are taken into account. Literature formalizes those more sophisticated attacks as linear programming problems, the so called “attacker problem”.

Obviously, cell suppression requires the definition of table relations. This is the main part of the step we refer to here as ***secondary risk assessment***. The step also defines how much protection must be provided to the primary cells. This part is especially important, if in the case of skewed magnitude data primary risk assessment accounts not only for cases where data of a data subject can re-computed exactly, but also where releasing the cell value makes it possible to closely infer a data subjects data. In those cases, secondary risk assessment must reflect risks not only of exact, but also of inferential disclosure.

Secondary risk assessment should also account for special risky situations that occur because of cells that must be protected because they relate to only one data subject (so called “*singleton*”). The problem is that their suppression can be easily undone by a special user of the published table, e.g. the respective data subject herself. Secondary risk assessment must reflect that such a cell cannot always be used to protect another primary sensitive cell: When for example two of those cells are the only suppressed cells in a table relation, each of the two data subjects (singletons) can recalculate the suppressed value of the other one by simple subtraction.

Standard statistical software packages fail to support the secondary risk assessment step[3]. Specialized tools for cell suppression provide the information. This step is the basis to create input,

[2] The respective parts of this textbook have been contributed by one author of this report. The present section recalls ideas and concepts as explained by this textbook as far as they are relevant for this report.

[3] An exception is the commercial package SuperCross (see http://www.spacetimeresearch.com/supercross.html for more information) which is linked to the τ-ARGUS package to some degree. Regarding the commercial statistical package SAS, Statistics Sweden and Destatis both have developed tools to link SAS with τ-ARGUS.

especially meta-information, to step 3. All modern cell suppression packages support the definition of table relations in a situation when tables are defined by crossing one or more categorical variables. They also create appropriate relation schemes in those cases where categorical variables have hierarchical structure, defining sub-margins for the table. Some packages also support the definition of linked tables to some degree. Linked tables are defined as two or more tables presenting data on the same response variable and sharing cell(s) from the same category(ies) of at least one variable that defines the table.

Secondary risk assessment is a very crucial step for cell suppression, especially when multiple linked tables are released from the same survey. Where this step fails to represent existing links between some of the tables or sub-tables, it may cause a risk that users comparing the protected tables may find that they can undo some of the protection. Secondary risk assessment usually requires some expertise, not only in knowing how to use a suitable software package. Some packages also assess special “singleton” risks to some degree.

The most challenging of those three steps is the last one, e.g. ***secondary cell suppression***. For each primary cell that needs protection, there are usually many possible choices of secondary cells. The goal is to find a valid set of secondary suppressions that provide the required protection to the primary cells with a minimum loss of information connected to it. For a mathematical statement of the secondary cell suppression problem (CSP) see e.g. Fischetti and Salazar (2000), who state the CSP as a computationally extremely hard mixed integer linear programming problem. For a survey of the respective literature see for example Duncan, Elliot, Salazar-González (2011), 4.2.1. In section 3 we recall the brief explanation and comparison of the most prominent algorithms for solving the CSP, or a heuristic relaxation as presented in 4.4.3 of Hundepool et al. (2012). In connection with those algorithms, heuristic methods avoiding at least to some extent singleton and multicell risks have been proposed in the literature and are mentioned in the respective paragraphs. Also, all of those algorithms offer features to protect not only against exact, but also inferential disclosure risks.

It is important to note that the solution to a CSP depends very much on the set of table equations stated. In general, the more table equations are involved, the higher becomes the complexity of a CSP. If the secondary risk assessment step defines a large table model (e.g., a large number of table equations), the CSP will become computationally too difficult to be solved in practice. The three algorithms explained in section 3 below vary considerably with respect to the “size” of a problem they can handle. The ILP algorithm is particularly sensitive to the size of a problem. The network flow heuristic can only handle a special type of structure in the table equations, but on those it is very efficient.

A typical approach to avoid “intractability” problems can be briefly outlined as follows: The original algorithm is applied separately (in a modular way, so to say) to sub-problems relating to subsets of the full set of table equations. Any resulting secondary suppressions that also appear in a table equation belonging to one of the other subsets in recorded. When the CSP algorithm for this other equation set is processed, they will be suppressed and protected there as well. This procedure is looped across the subsets of table equations until no further suppressions occur.

This technique usually increases the speed of a protection process enormously. The disadvantage is that there are constellations where a suppression pattern computed this way might be unpicked. I.e., it might be possible to re-compute a protected cell when taking into account table equations from different subsets for setting up the respective attacker problem. See Cox (2001) for discussion of residual disclosure risks related to “non-global” methods for secondary cell suppression.

It is reasonable to assume that the risk that such an attack actually takes place decreases, when the set of equations taken into account for the sub-problems increases because of the complexity and effort of setting up and solving large attacker problems. Section 3.2 illustrates the technique at the example of two methods implemented in the software package τ−ARGUS.

In practice, protection strategies implicitly relate to either of the risk models below (or sometimes a combination thereof). For further explanations see 4.3.4 of Hundepool et al. (2012). The simpler the risk model, the higher will be the risk that cells protected by a respective strategy might be re-computable. Note that if a procedure implements a method to avoid singleton and multicell risks, the risk model of this special facility is not necessarily the same as for the strategy in general.

Risk model I: Relation

The “Relation” model is based on the assumption that it is most important (or even enough) to prevent a certain kind of spontaneous disclosure, requiring only one simple, straightforward computation.

Risk model II: Subtable

This model assumes the following scenario: An interested data user might compute the feasibility intervals of a protected sensitive cell by taking into account the set of table relations of just one subtable[4] of an eventually much larger hierarchical table. For a skilled user with some linear programming knowledge this would be a task that does not require much effort. However, we assume a skilled user here, so the probability of the attack is certainly lower than the one of the ‘Relation’-model above.

Risk model III: Table

The assumption of this model is that a data user might take into account the full set of table relations of a protected hierarchical table, or even of a set of protected linked tables, when computing feasiblity intervals for protected cells. This is certainly possible for a skilled user. However, in large hierarchical tables creating the required input information for a suitable analysis program is a major effort indeed which makes this scenario less likely than that of the ‘subtable’-model above.

3. Methodological Comparison of Algorithms for Secondary Cell Suppression

In this section we briefly outline and compare the concept of some prominent algorithms and procedures for secondary cell suppression. We also mention tools implementing those algorithms and note if a technique is implemented to prevent singleton disclosure. Table 1 provides an overview, observing which risk model underlies the concept in general, and regarding the singleton disclosure avoidance technique in particular.

Because of relatively high disclosure risks, we do not consider algorithms that generally take into account only risk model I (relation).

Integer Linear Programming (ILP) Approach

Fischetti and Salazar (2000) present an integer linear programming approach for an optimal solution of the secondary cell suppression problem stated above. It solves the problem of finding a set of secondary suppressions with a minimum loss of information associated to their suppression, subject to constraints expressing that all primary sensitive cells are sufficiently protected. The algorithm is very efficient, for details see Salazar (2008). However, computing times grow rapidly

[4] Subtables are certain parts of tables where the categories of the spanning variables define a hierarchical tree structure. A subtable is a part of a table, where for any of the hierarchical spanning variables only one parent node category and the respective child node categories appear. See 4.1.2 of Hunepool et al. (2012)

when the table size increases. The method is implemented in the τ-ARGUS package and referred to there as method "optimal". A re-implementation by Meindl (2010), is available in the package sdcTable. A heuristic for how to solve singleton problems is outlined at the end of the next paragraph relating to the linear programming approach. It is implemented in the τ-ARGUS version of the algorithm, but not in sdcTable.

Linear Programming (LP) Approach

A computationally much cheaper Linear Programming approach has been described in Sande (1984), Robertson (1993, 1994) and Frolova, Fillion, Tambay (2009). Instead of the computation intensive integer linear programming problem, only a linear programming relaxation is solved. The algorithm will determine for each sensitive cell a congruent table, where the value of this cell, and also the values of some other cells have been altered. "Congruent table" means that all table relations of the original table still hold for the altered table. The algorithm will minimize the sum of the deviations between true and altered cell values, weighted by some suitable cell costs.

Sande/Robertson propose to make the algorithm obtain a suitable cell suppression pattern in the following way: the algorithm is carried out two times. In the first run, cell costs should be assigned that increase slightly with the cell value, like for instance the logarithm of the cell values.

In the second run, only cells can be altered that have been altered also in the first run. This time a decreasing cost function is used which will make the algorithm change fewer cells. The secondary suppressions are then those altered in the second run.

Versions of this algorithm are underlying the CONFID and CONFID2 packages of Statistics Canada and also the commercial package ACS (Sande, 1999). The US Census Bureau is currently developing a new package based on a similar approach.

Robertson (2000) proposes a heuristic which avoids singleton and multicell disclosure risks[5] efficiently, e.g. without tendency for over-suppression. However, sufficient protection is guaranteed only according to the (weakest) risk model "relation". E.g. the heuristic ensures that two respondents contributing to different sensitive cells within the same table relation will not be able to disclose each others contribution by subtracting their own contribution from the difference between an unsuppressed relation-total and the sum of the unsuppressed cells in that relation. The heuristic is implemented in CONFID/CONFID2 as well as in connection with Fischetti/Salazar's integer linear programming approach (in particular also in connection with the heuristic for hierarchical and linked tables introduced below) in the τ-ARGUS software package.

Network flow heuristics

Network flow heuristics for secondary cell suppression build on the fact, that a suppressed cell in a two-dimensional table is safe from exact disclosure if, and only if, the cell is contained in a 'cycle', or 'alternating path' of suppressed cells. The network flow cell suppression heuristic is based on the solution of a sequence of shortest-path subproblems that guarantee a feasible pattern of suppressions (i.e., one that satisfies the protection levels of sensitive cells). Hopefully, this feasible pattern will be close to the optimal one. See Castro, (2003a) and Castro (2003b) for details about an implementation included in the τ-ARGUS software package. An earlier implementation of a network flow heuristic is included in a cell suppression package of the US Census Bureau, Jewett (1993). A major disadvantage of network flow heuristics in the context of cell suppression is that

[5] C.f. 4.3.3 of of Hunepool et al. (2012)

they can only be used when the tables that have to be protected can be modelled as at most two-dimensional tables with at most one hierarchically substructured spanning variable.

Regarding singleton and multicell disclosure, the implementation of Jewett (1993) offers a heuristic that dynamically assigns for each target primary suppression so called capacity constraints to potential secondary suppressions which solves the problem and does not lead to a tendency for oversuppression. In connection with Castro’s algorithm, no solution to avoid this kind of risk is available so far.

Hypercube Method

A hypercube method for cell suppression builds on the fact that a suppressed cell in a simple n-dimensional table without substructure cannot be disclosed exactly if that cell is contained in a pattern of suppressed, nonzero cells, forming the corner points of a hypercube. An implementation of such a method has been described in depth in Repsilber (1994), or Repsilber (2002). For a briefer descriptions see Giessing and Repsilber (2002)

A hypercube method will construct successively for any primary suppression all possible hypercubes with this cell as one of the corner points.

If the disseminator requires protection against inferential disclosure, for each hypercube a lower bound can be calculated for the width of the feasibility interval for the primary suppression that would result from the suppression of all corner points of the particular hypercube. To compute that bound, it is not necessary to implement the time consuming solution to the Linear Programming problem and it is possible to consider bounds on cell values that are assumed to be known to an intruder. If it turns out that the bound for the feasibility interval width is sufficiently large, that hypercube becomes a feasible solution. The algorithm then selects from the set of feasible hypercubes the one with the lowest sum of cell costs associated to its corner points. These corner points will then be added to the suppression pattern. Repsilber (1994) proposes to assign cell costs dynamically as logarithmic transformation of the cell value but also depending on the level of a cell to avoid suppression of marginal cells.

Unlike the ‘cycle’ criterion of the network flow heuristics, the ‘hypercube criterion’ is a sufficient but not a necessary criterion for a ‘safe’ suppression pattern. Thus, it may happen that the ‘best’ suppression pattern may not be a set of hypercubes – in which case, of course, the hypercube method will miss the best solution and lead to some overprotection.

Regarding the problem of singleton disclosure, the implementation of Repsilber (1994) makes sure that a single respondent cell will never appear to be corner point of one hypercube only, but of two hypercubes at least. This implies protection not only under the weak (Relation) risk model, but also considering the more rigorous model (subtable) of 4.3.4.

Special algorithms for Hierarchical and Linked Tables

This paragraph explains how algorithms for secondary cell suppression, like an ILP or a Hypercube algorithm, can be applied to hierarchical and linked tables in a practical way. Although both algorithms can and have indeed been implemented in a way to deal with hierarchical and generally linked tables directly, in practice application to large hierarchical and linked tables becomes too computer extensive. Note that network flow heuristics on the other hand are usually fast enough. If it would make sense to build an LP approach into the kind of heuristic described below is an open research issue. For large subtables with more than two or even three dimensions for which computation of the optimal integer linear programming suppression pattern tends to be time consuming, it might possibly be an interesting alternative.

As explained in section 2, the original algorithm (like ILP or Hypercube) is applied separately to sub-problems relating to subsets of the full set of table equations. The technique implemented here corresponds to the risk model "Sub-table", i.e. the equation subsets are the equation sets corresponding to a sub-table of the original hierarchical table. Those sub-tables are protected separately. Any complementary suppressions belonging also to one of the other sub-tables are noted; they are suppressed in this other sub-table as well, and the cell suppression procedure will be repeated.

Note that the approach will provide a sub-optimal solution that minimises the information loss per sub-table, but not necessarily the global information loss of the complete set of hierarchically linked tables.

For this kind of looping procedures the order sequence of the subtables has certainly an impact on the result. Typically, such a looping procedure deals with hierarchical tables using a top-down approach. The basic idea behind the top-down approach is to start with the highest levels of the variables and calculate the secondary suppressions for the resulting table. The suppressions in the interior of the protected table are then transported to the corresponding marginal cells of the tables that appear when crossing lower levels of the two variables. In the τ-ARGUS package the respective procedures corresponding to the ILP and hypercube algorithm are referred to as "modular" and "hypercube" method. SdcTable refers to the respective re-implementation of the modular method as "hitas" method.

For generally linked tables, to determine a suitable order sequence to handle the (sub-) tables is less obvious. De Wolf, Giessing (2008) discusses several options. In the following we briefly outline two of them, referred to as "adapted modular" and "traditional".

The adapted modular approach will first protect all subtables of a certain level in the hierarchies– no matter to which table those subtables belong – and then it will descend to the next level. The τ-ARGUS package offers an implementation of this technique in connection with the ILP algorithm, see De Wolf and Hundepool (2010).

The traditional way is to protect all subtables of one table – and then proceed to the next table. For a set of two linked tables this means we carry over the secondary suppressions of the first table to the 2nd, protect the 2nd table, carry over suppressions to the first table, and protect the first table again. For sets of more than two linked tables a suitable looping scheme is not that obvious, see f.i. the discussion in Giessing, (2009). Repsilber's implementation of the hypercube method builds on the scheme denoted as 'simple linked tables sequence' in Giessing (2009). It is offered in the τ-ARGUS package and in sdcTable.

It is important to observe that the methodology outlined here for linked tables can only work well when protecting a set of linked tables jointly. The problem of assigning secondary cell suppressions in a set of tables that must be consistent with the cell suppressions of another set of linked tables released earlier and linked to the 'current' set may well turn out to be infeasible. Note that this is a major concern with cell suppression, especially relevant for the third party research situations described in the introduction. Therefore, in DwB WP11 there is a strong focus on the study and development of perturbative protection methods as alternative to cell suppression.

Table 1 below provides an overview of the algorithms and procedures explained in this section and in which tools they are implemented. It compares which of the risk models introduced at the end of section 2 underlies the concept in general, and regarding the singleton disclosure risk avoidance technique in particular.

Table 1: Risk models of cell suppression algorithms

Tool	**τ-ARGUS**		**sdcTable**		**CONFID2**		**USBC**	
	General	***Singleton***	***General***	***Singleton***	***General***	***Singleton***	***General***	***Singleton***
ILP	table	relation	table	No	-	-	-	-
LP	-	-	-	-	table	relation[6]	table	table[7]
Network	table	No	-				table	table
Special procedures for hierarchical tables								
ILP	subtable	relation	subtable	No				
Hypercube[8]	subtable	table	Subtable	Subtable(?)				

4. Empirical Comparison of Algorithms for Secondary Cell Suppression

Hundepool et al. (2012), section 4.7, summarizes results of two evaluation studies comparing the τ-ARGUS algorithms Modular ILP, Hypercube and a relaxed variant of the latter, Hyper0, which does not provide protection against inferential disclosure. Those studies did not involve any of the American packages. Neither did they involve the implementation of the network flow heuristic of τ-ARGUS[9]. Both studies are based on detailed tabulations of a strongly skewed magnitude variable. One of the studies used 2- and 3-dimenional hierarchical tables, with a total number of cells varying between 460 and 150 000. For the other study, results have been derived for tables with 7 level hierarchical structure (given by the NACE economy classification) of the first dimension, and 4 level structure of the second dimension given by the variable Region. The main conclusion of this comparison was the following: While both methods, the Modular ILP method and the Hypercube method, basically satisfy the same standard regarding disclosure risk management, the Modular ILP gives much better results regarding information loss. Even compared to a variant of Hypercube (Hyper0) with relaxed protection standard, it performs clearly better. This holds true especially for the stronger aggregated parts of large and detailed hierarchical tables. Although longer computation times for this method (compared to the fast hypercube method) can be a nuisance for the modular ILP method, the results clearly justify this additional effort.

Although not originally foreseen in the project plan, it makes some sense to compare empirically not only τ-ARGUS package algorithms, but also to compare with the two American packages and the implementation in sdcTable. However, a rigorous empirical testing of the packages would be a major effort, clearly out of the scope of this project task. On the other hand, there are interesting open research questions, like for example if it may makes sense for future versions of ARGUS to add a variant of the special procedures for hierarchical/linked tables that builds on LP instead of ILP based secondary cell suppression algorithm. It is also interesting how processing times of the LP algorithms behave for very large applications, and if they remain tractable without a special procedure for hierarchical/linked tables. In order to get at least some preliminary empirical information, a little study has been added to task 1 of WP11. Using a synthetic dataset with a strongly skewed magnitude variable, a very large, 3-dimensional table with 7-level hierarchical structures in one variable has been set up. This table has been shipped to the software suppliers, along with the results of primary and secondary risk assessment (in suitable format). The output of

[6] Information based on the assumption that the method suggested by Robertson (2000) to be implemented for CONFID is implemented in CONFID2 as well.

[7] In current implementation: through post-processing

[8] The details presented here refer to the program GHMITER which implements Repsilber's hypercube method outlined in sec. 3.and is offered by the τ-ARGUS package. It is not clear to us, if the hypercube method offered by sdcTable uses exactly the same program, or is a re-implementation of it and if the properties of the re-implementation are exactly the same as of the original.

[9] The network flow heuristic of τ-ARGUS does not yet offer a singleton disclosure risk avoidance technique and was therefore not included in those studies.

a successful application of the respective tool should be returned to DwB partner Destatis for comparison. The respective tests for the τ-ARGUS modular ILP and hypercube method have been carried out by DwB partner Destatis.

In the following we discuss some results. It is important to note that these results will not fully reflect the capabilities of the tools. With such large applications, for optimization based algorithms it is always important to fine-tune a tool (like through parameters etc.) to optimize the results. The fine tuning can have a strong effect on the outcome. As it was not a real application, merely for the sake of testing testers could not afford to spend much resources (regarding both, men power, as well as computer power) on the application[10].

Another reason for possible lack in comparability is that because the τ-ARGUS ILP algorithm was used from within a modular procedure for hierarchical/linked tables, the risk concept is weaker than that of the USBC tool with LP based algorithm, as explained above. This may result in less suppression. On the other hand, some residual disclosure risk regarding the strict risk model "table" has to be expected for the outcome of a modular ILP algorithm. Rigorous assessment of those risks which can be very time consuming for such a large application was out of scope of this work.

Another issue affecting comparibility is the LP solver used to run the application. US Census Bureau and Destatis have been using the commercial package CPLEX as LP solver. With sdcTable the less powerful but free and open-source solver GLPK was in place[11]. In order to be able to solve large applications like our test application in reasonable time on a standard PC with such a less powerful solver a greedy algorithm (top-down sub-table approach and solving each sub-table using the greedy algorithm with possible backtracking) has been implemented in sdcTable[12].

Processing times are not directly comparable either, because the applications have been run on different machines and also because of the LP solver issue. We report them here anyway as far as they have been recorded and made available to us:

The US Census Bureau has reported a processing time of about 10 hours. With τ-ARGUS, several variants of the modular ILP algorithm have been tried. Computing times on a standard PC vary between 20 minutes and about 15 hours. They depend slightly on the particular choice of parameters, but in the first place on the choice of the secondary risk concept: if the data are processed taking into account the information needed to avoid not only exact, but also inferential disclosure risks, run times are between 13 and 15 hours. If risks of inferential disclosure are ignored, computing time reduces to between 45 and 20 minutes (!). Obviously, the option to avoid inferential disclosure makes CSP problems much more difficult (time consuming) to solve and probably also increases the number of loops in the modular processing. In the current implementation of the sdcTable greedy algorithm this option is not (yet) available, e.g. it prevents exact disclosure only[13]. The output of a successful application of sdcTable has been delivered, without information on the computing time[14]. Computing times for the hypercube method (in τ-ARGUS) were negligible (below 2 minutes).

For sake of simplicity and because such an option is not implemented in sdcTable except for the hypercube method, all tests have been carried out without activation of singleton disclosure risk avoidance techniques. Table 3 presents the number of secondary suppressions and the sum of their values. The first three columns relate to results obtained when taking into account requirements to

[10] In fact, Statistics Canada was interested initially to contribute to the experiment but as yet could not spend enough resources on the project.

[11] For τ-ARGUS, a version based on a free solver will become available only later in the course of the DwB project. With SdcTable, it is in principle possible to run the algorithms with commercial solvers like CPLEX or XPRESS as well. Because Statistics Austria does not have a license for those packages this was no option for our little testing experiment.

[12] A methodological description of the greedy algorithm is not available to us; there is no reference therefore to this algorithm in table 1.

[13] Other cell suppression algorithms of sdcTable allow for inferential disclosure risk avoidance, but not the greedy heuristic.

[14] Although of course recorded the processing time has unfortunately not been reported to us.

avoid inferential disclosure. The remaining columns relate to results obtained when these kinds of risks are ignored. As processing times are short in that constellation, the modular ILP of τ-ARGUS has been tried with different settings of a parameter (‘lam’) affecting what is considered ‘optimal’ by the ILP algorithm.

Table 3: Number and sum of the cell values of secondary suppressed cells obtained by different tools

Information Loss measure	Solutions avoiding inferential disclosure risks			Solutions ignoring inferential disclosure risks				
	Tau-mod (lam=0.4)	USBC	Hyper-cube	Tau-mod			sdcTable greedy alg.	Hyper-cube
				(lam=1)	(lam=0.4)	(lam=0)		
sum_supp (in mill.)	414	596	2070	219	192	217	361	666
num_supp	19287	31733	24961	10876	10860	10436	13591	19106

In both settings the τ-ARGUS modular ILP method performs best, both in terms of number of suppressions and also regarding the sum of the values of the suppressed cells. Even though in the first setting (avoiding inferential disclosure risk) the hypercube method suppressed a smaller number of cells (ca. 25000 cells) compared to the LP software of the US Census Bureau (ca. 31000 cells) the sum of their values is more than 3 times higher which is probably because the hypercube method tends to suppress more cells on higher hierarchical levels (which tend to have larger cell values) than LP or ILP based methods.

In the solutions obtained when ignoring risks of inferential disclosure, the number of secondary suppressions is generally much lower. The modular ILP version of sdcTable manages a much better solution than the hypercube method (of τ-ARGUS), but results obtained by the modular ILP version of τ-ARGUS are clearly better than those of the sdcTable greedy algorithm. As mentioned above, the greedy algorithm was the only option to protect this dataset in our experiment with sdcTables because of problems with GLPK when using the sdcTable modular ILP method with such complex datasets and lack of a commercial solver licence at Statistics Austria[15].

To what extent the fairly large difference in the solution provided by the modular ILP based ARGUS algorithm and the LP based algorithm of the Census Bureau is due to differences in the risk model (and hence some underprotection always to be expected for results produced by modular processing when assuming the stricter risk model) is not clear. To answer this question it would take a rigorous audit of the solution and/or some post-processing to fix underprotection problems. This is again out of scope for this study, especially because no suitable tool exists for such a post-processing step.

5. Practical issues

Like the present report, also Ichim and Franconi (2009) and Sukasih et al (2011) compare cell suppression packages. They focus on more practical issues. Sukasih et al (2011) explains that the development of software for disclosure control has been based on individual needs of the statistical institutes developing them. This puts some constraints on the ease of sharing tools, even if agencies were willing to share them. According to Sukasih et al (2011) only the packages τ-Argus and sdcTable are really publicly available. Indeed, for sdcTable the source code is distributed over the web as free and open-source under the General Public Licence 2.

[15] A comparision of commercial and non-commercial solvers using sdcTable has been carried out with a temporary license for the commercial solvers (see Meindl and Templ (2012)) which has expired in the meantime

In the present report, we have also mentioned tools of the US Census Bureau and of Statistics Canada. In principle the SAS based Canadian tool Confid2 might be attractive to institutes with SAS based production environment. Statistics Canada offers a license for Confid2 at about 30,000 Canadian Dollar. The tool of the US Census Bureau discussed in the previous sections is still under development. There are no plans yet of eventually sharing the future tool.

Quoting from Sukasih et al (2011) " sdcTable is an SDL package for the statistical software R (http://www.r-project.org/). R and its packages, including sdcTable, are freeware". A big advantage of sdcTable over τ-ARGUS is that it is free and open source software: Because of this, users can study what is going on inside each process. They may eventually modify the codes to their specific needs. This will of course only be an advantage for expert users. The second major advantage is that its ILP, modular ILP and greedy algorithm can be used with a free solver. This does not only save users from license costs, but also from the complexity of installing such a license, which can be quite considerable in practice, sometimes. The other advantage is that sdcTable is fully platform independent. It is, e.g., available for Windows, Linux, and Mac operating system. To be able to use it, however, the user needs knowledge of R programming. The package does not yet offer a graphical user interface. Compared to τ-Argus, sdcTable is relatively new; the development has been underway for only the past 3 or 4 years. As yet, its development has not been influenced much by contributions of users outside Statistics Austria.

τ-Argus runs on Windows platform only. It is relatively easy to use, with a menu-driven user interface. Since the start of its development by Statistics Netherlands (CBS) in the 199ties, its evolution has been influenced by suggestions from users outside of CBS, in particular from partners in a number of European cooperation projects. One of the major disadvantages, e.g. the dependence of the ILP and modular ILP algorithm implemented in the package from commercial solvers is soon to be fixed in the course of the DwB project.

6. Summary and Conclusions

The idea of this report has been to assess the range of tools for cell suppression available to NSIs, and select those tools capable of cost efficient development and use by data service providers. If no suitable software tool is found, an architecture document should be supplied as a basis for future development. The concept of the report was to first introduce into the methodological background and to introduce into the three main steps of cell suppression: (1) primary and (2) secondary risk assessment and (3) secondary cell suppression. It has been explained that – with the exception of one commercial tool (Super Cross, c.f. section 2) – statistical packages like f.i. SPSS or SAS fail to support even the comparatively simple risk assessment steps (as for SAS, see footnote 3).

Regarding the two risk assessment steps, the open source package sdcTable might indeed qualify as a tool "capable of cost efficient development and use by data service providers" – not because it already offers the full range of features that are desirable for risk assessment, but as free and open source package, data service providers might eventually be able to create or have created such features on their own behalf.

However, this report has emphasized that the crucial step for a tool is step (3), secondary cell suppression. The focus of this report has been to explain differences and similarities of the respective algorithms especially regarding the underlying risk concepts. We have also explained that different algorithms may be based on different risk models (section 2). Comparing their performance, while ignoring those differences is to some degree like comparing apples to oranges.

Assuming a strict perspective, only secondary cell suppression algorithms that build on the most rigorous risk concept will reliably provide suppression patterns that provide full protection. For the protection of the large, hierarchical 3-dimensional table used for the testing reported in section 4, only the two American packages (by USBC and by Statistics Canada) would be eligible from this point of view[16]. In practice however, most data producers (statistical agencies as well as data service providers) still use manual secondary cell suppression techniques or home bread tools which will usualy lead to results that provide protection only considering the weakest risk concept mentioned in the report. Any of the algorithms considered for this survey supports a stricter risk concept. Even adopting those with a comparatively weak risk concept means progress. However, it must be noticed that, like the τ-ARGUS network flow algorithm, sdcTable still has a major flaw which rules it out for practical use for many agencies: it does not provide an automated singleton disclosure risk avoidance technique (except eventually with the hypercube method).

For those agencies facing in particular the task of having to protect large 3-dimensional hierarchical magnitude tables, if they can accept the less stringent risk model tailored to hierarchical table models, the modular ILP algorithm in connection with a commercial LP solver as implemented in τ-ARGUS will be the best choice, if performance of the software is the main issue. This is evident from the results of a recent empirical comparison presented in section 4.

As explained in the task description of WP11 of the DwB project, the starting point of this work is the problem that the future of the τ-ARGUS tool has been uncertain for some time, due to dependency on expertise in CBS (Netherlands NSI), in particular because one of the main developers is soon to retire. Removing this uncertainty will be costly. Considering also practical issues raised in section 5, two obvious alternatives are to either invest into sdcTable, or to migrate τ-ARGUS into a platform-independent open-source package which basically means to re-implement major parts of the package. As there is still a major gap between the maturity of the two tools (like the missing menu driven user interface in sdcTables, some missing techniques, etc.) Eurostat has taken a decision to finance the migration of τ-ARGUS by CBS. To prepare for this decision, an architecture document has been drawn up as deliverable of a co-operation project in 2011 (see Hundepool (2011)). Thanks to the Eurostat funding, by end of 2014 the then emerging open source version of τ-ARGUS will hopefully qualify as “tool for cost-efficient development and use by data service providers”, like formulated in our task description.

References

Castro, J. (2003 a), *‘Minimum-Distance Controlled Perturbation Methods for Large-Scale Tabular Data Protection’*, accepted subject to revision to European Journal of Operational Research, 2003

Castro, J. (2003 b) *User’s and programmer’s manual of the network flows heuristics package for cell suppression in 2D tables* Technical Report DR 2003-07, Dept. of Statistics and Operations Research, Universitat Politècnica de Catalunya, Barcelona, Spain,2003; See http://neon.vb.cbs.nl/casc/deliv/ 41D6_NF1H2D-Tau-Argus.pdf

Cox, L. (2001), ‘*Disclosure Risk for Tabular Economic Data’*, In: ‘Confidentiality, Disclosure, and Data Access: Theory and Practical Applications for Statistical Agencies’ Doyle, Lane, Theeuwes, Zayatz (Eds), North-Holland

De Wolf, P.P., Giessing, S. (2008) *How to make the τ-ARGUS Modular Method Applicable to Linked Tables*. In: Domingo-Ferrer, Josep; Saygin, Yücel (Eds.): Privacy in Statistical Databases 2008, Lecture Notes in Computer Science , Vol. 5262, Springer, Heidelberg (2008), p.227-238.

De Wolf, P.P., Hundepool, A. (2010) ‘*Three Ways to Deal with a Set of Linked SBS Tables Using τ-ARGUS’*. In: Domingo-Ferrer, Josep; Magkos, Emmanouil (Eds.): Privacy in Statistical Databases 2010, Lecture Notes in Computer Science , Vol. 6344, Springer, Heidelberg (2010), p.66-73.

Duncan, G.T., Elliot, M., Salazar-González, J.J. 2011, ‘Statistical Confidentiality Principles and Practice’, Statistics for Social and Behavioral Sciences, Springer New York Dondrecht Heidelberg London

Frolova, O., Fillion, J.M., Tambay, J.L. (2009) ‘*CONFID2: Statistics Canada’s new Tabular Data Confidentiality Software*’, SSC Annual Meeting, Proceedings of the Survey Methods Section.

[16] Βοτη, τ-ARGUS and sdcTables also provide an algorithm that supports the strict risk concept. This algorithm is however too slow in practice for larger tables.

Fischetti, M, Salazar Gonzales, J.J. (2000), '*Models and Algorithms for Optimizing Cell Suppression Problem in Tabular Data with Linear Constraints'*, in Journal of the American Statistical Association, Vol. 95, pp 916

Giessing, S. and Repsilber, D. (2002), '*Tools and Strategies to Protect Multiple Tables with the GHQUAR Cell Suppression Engine'*, in 'Inference Control in Statistical Databases' Domingo-Ferrer (Editor), Springer Lecture Notes in Computer Science Vol. 2316

Giessing, S. (2009), '*Techniques for Using τ-Argus Modular on Sets of Linked Tables*', paper presented at the Joint UNECE/Eurostat Work Session on Statistical Data Confidentiality (Bilbao, 2-4 December 2009) available at http://www.unece.org/stats/documents/2009.12.confidentiality.htm

Hundepool, A, Wetering, A van de, Ramaswamy, R, Wolf, PP de, Giessing, S, Fischetti, M, Salazar, JJ, Castro, J, Lowthian, P, (2011), *τ-ARGUS 3.5 user manual*, Statistics Netherlands, The Hague, The Netherlands. http://neon.vb.cbs.nl/casc

Hundepool, A., Domingo-Ferrer, J., Franconi, L., Giessing, S., Schulte Nordholt, E., Spicer, K., De Wolf, P.P. (2012) *Statistical Disclosure Control*. Wiley

Hundepool, A., (2011) *Vision on the new architecture of the Open Source Argus Software*, report, available at http://neon.vb.cbs.nl/casc/ESSNet2/TheNewArgus.pdf

Ichim, D. and Franconi, L. (2009) *On the sustainability of the SDC software tools*, report, available from http://neon.vb.cbs.nl/casc/..%5Ccasc%5CESSnet%5CSustainability%20of%20SDC%20software%20tools.pdf

Jewett, R. (1993), '*Disclosure Analysis for the 1992 Economic Census'*. Unpublished Manuscript. Economic Statistical Methods and Programming Division, Bureau of the Census, Washington, DC

Kraftling, A. (2011) *SAS2Argus user manual*, available at http://neon.vb.cbs.nl/casc/ESSNet2/SAS2Argus%20User%20manual%20(Ver%201.0).pdf

Meindl, B. (2010), *sdcTable: statistical disclosure control for tabular data*, R package version 0.0.16. [Online]. Available: http://CRAN.R-project.org/package=sdcTable

Meindl, B. and Templ, M. (2012) *Analysis of commercial and free and open source solvers for linear optimization problems.* Deliverable of the ESSnet on common tools and harmonised methodology for SDC in the ESS.

Salazar, J.J. (2003) "*Partial Cell Suppression: a New Methodology for Statistical Disclosure Control*", Statistics and Computing, 13, 13-21

Repsilber, R. D. (1994), '*Preservation of Confidentiality in Aggregated data'*, paper presented at the Second International Seminar on Statistical Confidentiality, Luxembourg, 1994

Repsilber, D. (2002), '*Sicherung persönlicher Angaben in Tabellendaten' -* in Statistische Analysen und Studien Nordrhein-Westfalen, Landesamt für Datenverarbeitung und Statistik NRW, Ausgabe 1/2002 (in German)

Robertson, D. (1993), '*Cell Suppression at Statistics Canada*', Proceedings Annual Research Conference, U.S. Bureau of the Census.

Robertson, D. (1994), '*Automated Disclosure Control at Statistics Canada*', paper presented at the Second International Seminar on Statistical confidentiality, Luxemburg, 1994

Robertson, D. (2000), '*Improving Statistics Canada's cell suppression software (CONFID*)',in J.G.Bethlehem, P.G.M. van der Hejden ed., Proceedings in Computational Statistics 2000, Physica-Verlag

Sande, G. (1984), ‚*Automated cell suppression to preserve confidentiality of business statistics'*. Stat. J. Unitd Nat. ECE2. 33-41

Sande, G. (1999) ‚*Structure of the ACS automated cell suppression system*' In Statistical Data Confidentiality. Proceedings of the Joint Eurostat/UnECE Work Session on Statistical Confidentiality, Skopje, pp. 105-121

Schmidt, K., Giessing, S. (2011) *A SAS-Tool for Managing Secondary Cell Suppression on Sets of Linked Tables by τ-Argus Modular*, report, available at http://neon.vb.cbs.nl/casc/ESSNet2/Appendix3APaper_Schmidt.pdf

Sukasih, A., Jang, D., Edson, D. (2011) *Using Tau-Argus and sdcTable to Conduct Secondary Cell Suppression for Linked Tables*, paper presented at the 2011 Joint Statistical Meetings of the American Statistical Association, Section on Survey Methodology

Acknowledgement

The author is grateful to the support of this work offered by Phil Steel (US Census Bureau), Mike Sirois (Statistics Canada) and Bernhard Meindl (Statistics Austria) who ran their cell suppression tools on a huge test data set for the survey reported in this paper. Katrin Schmidt (Destatis) prepared the data, ran the τ-ARGUS jobs, and checked and compared output data sets. Their help with this difficult and time consuming task is very much appreciated.