

τ

ARGUS

Version 2.2



User's Manual

Document: 4.2-D1
Project: CASC-project
Date: April 2003
BPA no: 769-02-TMO

Statistics Netherlands
P.O. Box 4000
2270 JM Voorburg
The Netherlands
email: ahnl@rnd.vb.cbs.nl

Contributors: Anco Hundepool, Aad van de Wetering and Ramya Ramaswamy.
Peter-Paul de Wolf (Hitas), Sarah Giessing (GHMiter)
Matteo Fischetti, Juan-José Salazar and Alberto Caprara (Optimisation)
Jordi Castro (Network solutions)

Contents

Preface.....	3
About the name ARGUS.....	3
Contact	3
Acknowledgments.....	4
1.Introduction.....	6
2.Producing safe tables.....	6
2.1 Sensitive cells in magnitude tables.....	6
2.2 Sensitive cells in frequency count tables.....	7
2.3 Table redesign.....	8
2.4 Secondary cell suppression.....	8
2.5 Information loss in terms of cell weights.....	8
2.6 Series of tables.....	9
2.7 The Hypercube/GHMITER method.....	9
2.7.1 The method.....	9
2.8 The ARGUS implementation of GHMITER.....	10
2.8.1 References on GHMiter.....	11
2.9 Hitas.....	12
2.10 Network Solution for Large Unstructured 2 dimensional tables.....	13
2.11 Functional design of τ -ARGUS.....	15
3.A tour of τ -ARGUS.....	16
3.1 Preparation.....	16
3.1.1 Open a microdata file.....	16
3.1.2 The Metafile.....	18
3.1.3 Specification.....	19
3.1.4 Open a Table.....	21
3.2 The process of disclosure control.....	27
3.2.1 Table protection.....	28
3.3 Saving the safe table.....	32
4.Description of the Menu Items.....	34
4.1 Main Window.....	34
4.2 The File menu.....	35
4.2.1 File Open Microdata.....	35
4.2.2 File Open Table.....	35
4.2.3 File Exit.....	36
4.3 The Specify menu.....	36
4.3.1 Specify Metafile [For Microdata].....	36
4.3.2 Specify Specify Tables.....	39
4.3.3 Specify TableMetadata.....	41
4.4 The Modify menu.....	41
4.4.1 Modify Select Table.....	42
4.4.2 Modify View Table.....	42
4.4.3 Linked Tables.....	47
4.5 The Output menu.....	48
4.5.1 Output Save Table.....	48
4.5.2 Output View Report.....	49
4.6 The Help menu.....	49
4.6.1 Help Contents.....	49
4.6.2 Help Options.....	49
4.6.3 Help About.....	50

Preface

This is the user's manual of τ -ARGUS version 2.2. τ -ARGUS is a software tool designed to assist a data protector in producing safe tables. This version is the second release of τ -ARGUS in the CASC-project. With respect to the τ -ARGUS version from the previous project we have made a major step forward and τ -ARGUS has now facilities to protect hierarchical and linked tables.

The purpose of τ -ARGUS is to protect tables against the risk of disclosure, i.e. the accidental or deliberate disclosure of information related to individuals from a statistical table. This is achieved by modifying the table so that it contains less, or less detailed, information. τ -ARGUS allows for several modifications of a table: a table can be redesigned, meaning that rows and columns can be combined; sensitive cells can be suppressed and additional cells to protect these can be found in some optimum way (secondary cell suppression).

τ -ARGUS is one of a twin set of disclosure control packages. Within the CASC-project a tool for microdata - called μ -ARGUS - is also being developed, which is the twin brother of τ -ARGUS.¹ This is manifest not only when one looks at the (user inter)faces of both packages, but also when one would look at the source code: the bodies of the twins are so much combined that they in fact are like Siamese twins!

About the name ARGUS

Somewhat jokingly the name ARGUS can be interpreted as the acronym of 'Anti-Re-identification General Utility System'². As a matter of fact, the name ARGUS was inspired by a myth of the ancient Greeks. In this myth Zeus has a girl friend named Io. Hera, Zeus' wife, did not approve of this relationship and turned Io into a cow. She let the monster Argus guard Io. Argus seemed to be particularly well qualified for this job, because it had a hundred eyes that could watch over Io. If it would fall asleep only two of its eyes were closed. That would leave plenty of eyes to watch Io. Zeus was eager to find a way to get Io back. He hired Hermes who could make Argus fall asleep by the enchanting music on his flute. When Hermes played his flute to Argus this indeed happened: all its eyes closed, one by one. When Hermes had succeeded in making Argus fall asleep, Argus was decapitated. Argus' eyes were planted onto a bird's tail - a type of bird that we now know under the name of peacock. That explains why a peacock has these eye-shaped marks on its tail. This also explains the picture on the cover of this manual. It is a copperplate engraving of Gerard de Laresse (1641-1711) depicting the process where the eyes of Argus are being removed and placed on the peacock's tail.³

Like the mythological Argus, the software is supposed to guard something, in this case data. This is where the similarity between the myth and the package is supposed to end, as we believe that the package is a winner and not a loser as the mythological Argus is.

Contact

¹ See Anco Hundepool et al., 2003, μ -ARGUS version 3.2 user's manual, Statistics Netherlands, Voorburg, The Netherlands.

² This interpretation is due to Peter Kooiman, and dates back to around 1992 when the first prototype of ARGUS was being built by Wil de Jong.

³ The original copy of this engraving is in the collection of 'Het Leidsch Prentenkabinet' in Leiden, The Netherlands.

Feedback from users will help improve future versions of τ -ARGUS and is therefore greatly appreciated. The authors of this manual can be contacted directly for suggestions that may lead to improved versions of τ -ARGUS in writing or otherwise; e-mail messages can also be sent to argus@cbs.nl.

Acknowledgments

τ -ARGUS has been developed as part of the CASC project that was partly sponsored by the EU under contract number IST-2000-25069. This support is highly appreciated. The CASC (Computational Aspects of Statistical Confidentiality) project is part of the Fifth Framework of the European Union. The main part of τ -ARGUS has been developed at Statistics Netherlands by Aad van de Wetering and Ramya Ramaswamy (who wrote the kernel) and Anco Hundepool (who wrote the interface). However this software would not have been possible without the contributions of several others, both partners in the CASC-project and outsiders.

The German partners Statistisches Bundesamt (Sarah Giessing and Dietz Repsilber) have contributed the GHMITER software, which offers a solution for secondary cell suppression based on hypercubes. Peter-Paul de Wolf has build a search algorithm based on the non-hierarchical optimal solutions. This algorithm will break down a large hierarchical table into small non-hierarchical subtables, which will then be protected. The optimisation routines have been developed by JJ Salazar cs. of the University La Laguna Tenerife, Spain. Additionally Jordi Castro has developed a solution based on networks.

For solving these optimisation problems τ -ARGUS uses commercial LP-solvers. Traditionally we use Xpress as an LP-solver. This package is kindly made available for users of τ -ARGUS at a special agreement between the τ -ARGUS-team and DASH-optimisation, the developers of Xpress. Alternatively τ -ARGUS can also use the Cplex-package. Also with CPLEX there is an arrangement for the use of τ -ARGUS. It is the choice of the users. However users having a licence for one of these packages can use their current licence for τ -ARGUS as well.

The CASC-project

The CASC project on the one hand can be seen as a follow up of the SDC-project of the 4th Framework. It will build further on the achievements of that successful project. On the other hand it will have new objectives. It will concentrate more on practical tools and the research needed to develop them. For this purpose a new consortium has been brought together. It will take over the results and products emerging from the SDC-project. One of the main tasks of this new consortium will be to further develop the ARGUS-software, which has been put in the public domain by the SDC-project consortium and is therefore available for this consortium. The main software developments in CASC are μ -ARGUS, the software package for the disclosure control of microdata while τ -ARGUS handles tabular data.

The CASC-project will involve both research and software development. As far as research is concerned the project will concentrate on those areas that can be expected to result in practical solutions, which can then be built into (future version of) the software. Therefore the CASC-project has been designed round this software twin ARGUS. This will make the outcome of the research readily available for application in the daily practice of the statistical institutes.

CASC-partners

At first sight the CASC-project team had become rather large. However there is a clear structure in the project, defining which partners are working together for which tasks. Sometimes groups working closely together have been split into independent partners only for administrative reasons.

Institute	Short	Country
1. Statistics Netherlands	CBS	NL

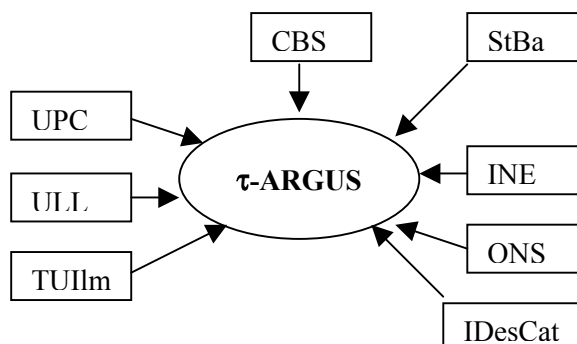
2. Istituto Nazionale di Statistica	ISTAT	I
3. University of Plymouth	UoP	UK
4. Office for National Statistics	ONS	UK
5. University of Southampton	SOTON	UK
6. The Victoria University of Manchester	UNIMAN	UK
7. Statistisches Bundesamt	StBA	D
8. University La Laguna	ULL	ES
9. Institut d'Estadística de Catalunya	IDESCAT	ES
10. Institut Nacional de Estadística	INE	ES
11. TU Ilmenau	TUilm	D
12. Institut d'Investigació en Intel·ligència Artificial-CSIC	CIS	ES
13. Universitat Rovira i Virgili	URV	ES
14. Universitat Politècnica de Catalunya	UPC	ES

Although Statistics Netherlands is the main contractor, the management of this project is a joint responsibility of the steering committee. This steering committee constitutes of 5 partners, representing the 5 countries involved and also bearing a responsibility for a specific part of the CASC-project:

CASC Steering Committee

Institute	Country	Responsibility
Statistics Netherlands	Netherlands	Overall manager Software development
Istituto Nazionale di Statistica	Italy	Testing
Office for National Statistics	UK	
Statistisches Bundesamt	Germany	Tabular data
Universitat Rovira i Virgili	Spain	Microdata

The CASC tabular data team



1. Introduction

The growing demands from researchers, policy makers and others for more and more detailed statistical information leads to a conflict. The statistical offices collect large amounts of data for statistical purposes. The respondents are only willing to provide the statistical offices with the required information if they can be certain that these statistical offices will treat their data with the utmost care. This implies that their confidentiality must be guaranteed. This imposes limitations on the amount of detail in the publications. Practice and research have generated insights into how to protect tables, but the problem is certainly not definitively settled.

Before we go into more details, the basic ideas on which τ -ARGUS is based, we give a sketch of the general ideas. At first sight one might find it difficult to understand that information presented in tabular form presents a disclosure risk. After all one might say that the information is presented only in aggregate form.

2. Producing safe tables

Safe tables are produced from unsafe ones by applying certain SDC measures to the tables. In the current section these SDC measures - as far as they are implemented in τ -ARGUS - are discussed in the present section. Some key concepts such as sensitive cells, information loss and the like are discussed as well.

2.1 Sensitive cells in magnitude tables

The well-known dominance rule is often used to find the sensitive cells in tables, i.e. the cells that can not be published as they might reveal information on individual records. More particularly, this rule states that a cell of a table is unsafe for publication if a few (n) major contributors to a cell are responsible for a certain percentage (k) of the total of that cell. The idea behind this rule is that in that case at least the major contributors themselves can determine with great precision the contributions of the other contributors to that cell. The choice $n=3$ and $k=70\%$ is not uncommon, but τ -ARGUS will allow the users to specify their own choice.

As an alternative the prior-posterior rule has been proposed. The basic idea is that a contributor to a cell has better chances to estimate the competitors in a cell than an outsider and also that these kind of intrusions can occur rather often. The precision with which a competitor can estimate is a measure of the sensitivity of a cell. The worst case is that the second largest contributor will be able to estimate the largest contributor. If this precision is more than $p\%$ the cell is considered unsafe. An extension is that also the global knowledge about each cell is taken into account. In that case we assume that each intruder has a basic knowledge of the value of each contributor of $q\%$. Note, that it is actually the ratio p/q which determines which cells are considered safe, or unsafe. In this version of ARGUS, the q -parameter is fixed to 100. Literature refers to this rule as (minimum protection of) $p\%$ -rule. If the intention is to state a prior-posterior rule with parameters p_0 and q_0 , where $q_0 < 100$, choose the parameter p of the $p\%$ -rule as $p = p_0/q_0 * 100$.

With these rules as a starting point it is easy to identify the sensitive cells, provided that the tabulation package has the facility not only to calculate the cell totals, but also to calculate the number of contributors and the n individual contributions of the major contributors. Tabulation packages like ABACUS (from Statistics Netherlands) and the package 'SuperCross' developed in Australia by Space-Time Research have that capacity. In fact τ -ARGUS not only stores the sum of the n major contributions for each cell, but the individual contributions themselves. The reason for this is that this is very handy in case rows and columns etc. in a table are combined. By merging and sorting the sets of individuals contributions of the cells to be combined, one can quickly determine the major contributions of the new cell, without going back to the original file. This implies that one can quickly

apply the dominance rule to the combined cells. Combining rows and columns (table redesign) is one of the major instruments to reduce the number of unsafe cells.

This too is the reason why τ -ARGUS reads microdata files. However due to continuous demands from users we have now also build the option to read ready made tables, however with the restriction that the options for table redesign will not be available..

A problem, however, arises when also the marginals of the table are published. It is no longer enough to just suppress the sensitive cells, as they can be easily recalculated using the marginals. Even if it is not possible to exactly recalculate the suppressed cell, it is possible to calculate an interval that contains the suppressed cell. This is possible if some constraints are known to hold for the cell values in a table. A common found constraint is that the cell values are all nonnegative.

If the size of such an interval is rather small, then the suppressed cell can be estimated rather precisely. This is not acceptable either. Therefore it is necessary to suppress additional information to achieve that the intervals are sufficiently large.

Several solutions are available to protect the information of the sensitive cells:

- Combining categories of the spanning variables (table redesign). Larger cells tend to protect the information about the individual contributors better.
- Suppression of additional (secondary) cells to prevent the recalculation of the sensitive (primary) cells.

The calculation of the optimal set (with respect to the loss of information) of secondary cells is a complex OR-problem. τ -ARGUS will be build around this solution and takes care of the whole process. A typical τ -ARGUS session will be one in which the users will first be presented with the table containing only the primary unsafe cells. The user can then choose how to protect these cells. This can involve the combining of categories, equivalent to the global recoding of μ -ARGUS. The result will be an update of the table with fewer unsafe cells (certainly not more) if the recoding has worked. At a certain stage the user requests the system to solve the remaining unsafe cells by finding secondary cells to protect the primary cells.

At this stage the user can choose between several options to protect the primary sensitive cells. Either he/she chooses the hypercube method or the optimal solution. In this case he/she also has to select the solver to be used, Xpress or Cplex. After this the table can be stored and can be published.

2.2 Sensitive cells in frequency count tables

In the simplest way of using Tau-Argus sensitive cells in frequency count tables are defined as those cells that contain a frequency that is below a certain threshold value. This threshold value is to be provided by the data protector. This way of identifying unsafe cells in a table is the one that is implemented in the current version of τ -ARGUS It should be remarked, however, that this is not always an adequate way to protect a frequency count table.⁴ Yet it is applied a lot,. Rather than mechanically applying a dominance rule or a p% rule one should think about possible disclosure risks that a frequency count table poses and possible disclosure scenarios in order to simulate the behaviour of an intruder. Such an analysis would probably come up with different insights than using a simple thresholding rule, *e.g.* like the one sketched in the reference just mentioned.

⁴ See for instance Leon Willenborg and Ton de Waal, 1996, Statistical disclosure control in practice, Springer-Verlag, New York, Section 6.3.

2.3 Table redesign

If a large number of sensitive cells are present in a table, it might be an indication that the spanning variables are too detailed. In that case one could consider combining certain rows and columns in the table. (This might not always be possible because of publication policy.) Otherwise the number of secondary cell suppressions might just be too enormous. The situation is comparable to the case of microdata containing many unsafe combinations. Rather than eliminating them with local suppressions one can remove them by using global recodings. For tabular data we use the phrase “table redesign” to denote an operation analogous to global recoding in microdata sets. The idea of table redesign is to combine rows, columns etc., by adding the cell contents of corresponding cells from the different rows, columns etc. It is a property of the dominance rule that a joint cell is safer than any of the individual cells. So as a result of this operation the number of unsafe cells is reduced. One can try to eliminate all unsafe combinations in this way, but that might lead to an unacceptably high information loss. Instead, one could stop at some point, and eliminate the remaining unsafe combinations by using other techniques such as cell suppression.

2.4 Secondary cell suppression

Once the sensitive cells in a table - either of magnitude or a frequency count type - have been identified, possibly following table redesign it might be a good idea to suppress these values. In case no constraints on the possible values in the cells of a table exist this is easy: one simply removes the cell values concerned and the problem is solved. In practice, however, this situation hardly ever occurs. Instead one has constraints on the values in the cells due to the presence of marginals and lower bounds for the cell values (typically 0). The problem then is to find additional cells that should be suppressed in order to protect the sensitive cells. The additional cells should be chosen in such a way that the interval of possible values for each sensitive cell value is sufficiently large. What is “sufficiently large” is to be specified by the data protector by specifying the protection intervals.

In general the secondary cell suppression problem turns out to be a hard problem, provided the aim is to retain as much information in the table as possible, which, of course, is a quite natural requirement. The optimisation problems that will then result are quite difficult to solve and require expert knowledge in the area of combinatorial optimisation.

2.5 Information loss in terms of cell weights

In case of secondary cell suppression it is possible that a data protector might want to differentiate between the candidate cells for secondary suppression. It is possible that he/she would like to preserve the content of certain cells as much as possible and is willing to sacrifice the values of other cells instead. A mechanism that can be used to make such a distinction between cells in a table is that of cell weights. In τ -ARGUS it is possible to associate different weights with the cells in a table. The higher the weight the more important the corresponding cell value is considered and the less likely it will be suppressed. We shall interpret this by saying that the cells with the higher associated weights have a higher information content. The aim of secondary cell suppression can be summarised by saying that a safe table should be produced from an unsafe one, by minimising the information loss, expressed as the sum of the weights associated with the cells that have secondarily been suppressed.

τ -ARGUS offers several ways to compute these weights. The first option is to compute these weights as the sum of the contributions to a cell. Secondly this weight can be the frequency of the contributors to a cell, and finally each cell can be weighted as one, minimising the number of suppressed cells.

2.6 Series of tables

In τ -ARGUS it is possible to specify a series of tables that will be protected one by one, and independently of each other. It is more efficient to choose this option since τ -ARGUS requires only a single run through the microdata in order to produce the tables. But also for the user it is often more attractive to specify a series of tables and let τ -ARGUS protect them in a single session, rather than have several independent sessions.

2.7 The Hypercube/GHMITER method⁵

In order to ensure tractability also of big applications, τ -ARGUS interfaces with the GHMITER hypercube method of R. D. Repsilber of the Landesamt für Datenverarbeitung und Statistik in Nordrhein-Westfalen/Germany, offering a quick heuristic solution. The method has been described in depth in [1], [2] and [3], for a briefer description see [4].

2.7.1 The method

The approach 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.

The algorithm subdivides n-dimensional tables with hierarchical structure into a set of n-dimensional sub-tables without substructure. These sub-tables are then protected successively in an iterative procedure that starts from the highest level. Successively, for each primary suppression in the current sub-table, all possible hypercubes with this cell as one of the corner points are constructed.

For each hypercube, a lower bound is calculated for the width of the suppression 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. If it turns out that the bound is sufficiently large, the hypercube becomes a feasible solution. For any of the feasible hypercubes, the loss of information associated with the suppression of its corner points is calculated. The particular hypercube that leads to minimum information loss is selected, and all its corner points are suppressed.

After all sub-tables have been protected once, the procedure is repeated in an iterative fashion. Within this procedure, when cells belonging to more than one sub-table are chosen as secondary suppressions in one of these sub-tables, in further processing they will be treated like sensitive cells in the other sub-tables they belong to. The same iterative approach is used for sets of linked tables.

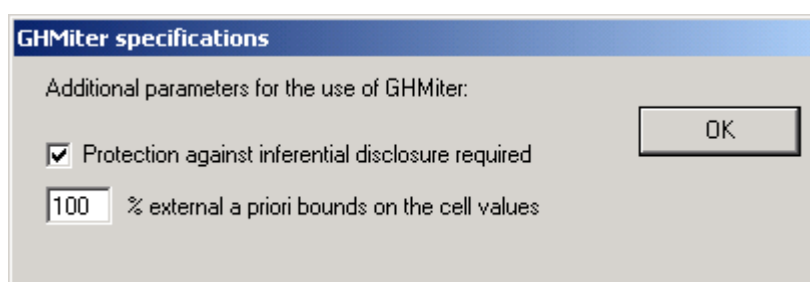
It should be mentioned here that the ‘hypercube criterion’ is a sufficient but not a necessary criterion for a ‘safe’ suppression pattern. Thus, for particular subtables the ‘best’ suppression pattern may not be a set of hypercubes – in which case, of course, the hypercube method will miss the best solution and lead to some overprotection. Other simplifications of the heuristic approach that add to this tendency for over-suppression are the following: when assessing the feasibility of a hypercube to protect a specific target suppressions against interval disclosure, the method

- is not able to consider protection maybe already provided by other cell suppressions (suppressed cells that are not corner points of this hypercube) within the same sub-table,
- does not consider the sensitivity of multi-contributor primary suppressions properly, that is, it does not consider the protection already provided in advance of cell suppression through aggregation of these contributions,
- attempts to provide the same *relative* ambiguity to (eventually large) secondary suppressions that have been selected to protect cells in a linked sub-table, as if they were single-respondent primary suppressions, while actually it would be enough to provide the same *absolute* ambiguity as required by the corresponding primary suppressions.

⁵ The section on GHMiter has been contributed by Sarah GIESSING, *Federal Statistical Office of Germany 65180 Wiesbaden E-mail: sarah.giessing@destatis.de*

2.8 The ARGUS implementation of GHMITER

- In the implementation offered by ARGUS, GHMITER makes sure that a single respondent cell will never appear to be corner point of one hypercube only, but of two hypercubes at least. Otherwise it could happen that a single respondent, who often can be reasonably assumed to know that he is the only respondent, could use his knowledge on the amount of his own contribution to recalculate the value of any other suppressed corner point of this hypercube.
- For tables presenting magnitude data, τ -ARGUS will ensure that GHMITER selects secondary suppressions that protect the sensitive cells properly, at least to the extent possible. It is assumed that users of the table can estimate any cell value to within some percentage of its actual value in advance of the publication, the so called *a priori* bound. By default τ -ARGUS assumes this percentage to be 100, but the user is offered to change it in the screen below:



Considering the given *a priori* bounds, τ -ARGUS will compute a suitable *sliding protection ratio* (for explanation see [5], τ -ARGUS will display the value of this ratio in the report file) to be used by GHMITER, to make it select secondary suppressions that are sufficiently large. This approach ensures that a user of the resulting protected table when using, apart from the assumed *a priori* information, only information provided by the data of the protected table would normally not be able to derive any bounds for the contribution of any respondent to a particular sensitive cell close enough to disclose this contribution according to the primary sensitivity rule in use.

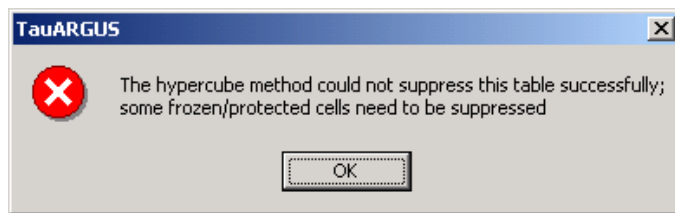
- Note, if in the screen above the option “*Protection against inferential disclosure required*” is inactivated, GHMITER will not check whether secondary suppressions are sufficiently large.
- As mentioned above, GHMITER is unable to 'add' the protection given by multiple hypercubes. In certain situations, considering the given *a priori* bounds, it is not possible to provide sufficient protection to a particular sensitive cell (or secondary suppression) by suppression of one single hypercube. In such a case, GHMITER is unable to confirm that this cell has been protected properly, according to the specified *sliding protection ratio*. It will then reduce the *sliding protection ratio* automatically, and individually, step by step for those cells, the protection of which the program cannot confirm otherwise. In steps 1 to 9 we divide the original ratio by k , values of k from 2 to 10, and if this still does not help, in step 10 we divide by an extremely large value, and finally, if even that does not solve the problem, step 11 will set the ratio to zero). The τ -ARGUS report file will display the number of cases where the sliding protection range was reduced, by finally confirmed sliding protection ranges.

Note, that that the number of cases with range reduction reported by this statistic in the report file is very likely to exceed the actual number of cells concerned, because cells belonging to multiple (sub-) tables are counted multiple times. In our experience this concerns particularly the cases, where the protection level was reduced to an ‘infinitely’ small (positive) value (in step 10, see above). Step 10 is usually required to confirm protection of large, high level secondary suppressions, which are likely to appear in multiple tables, especially in processing of linked tables. By the way, terms “reduction of the *sliding protection ratio*” and “reduction of the *protection level*” are used synonymously in the report file.

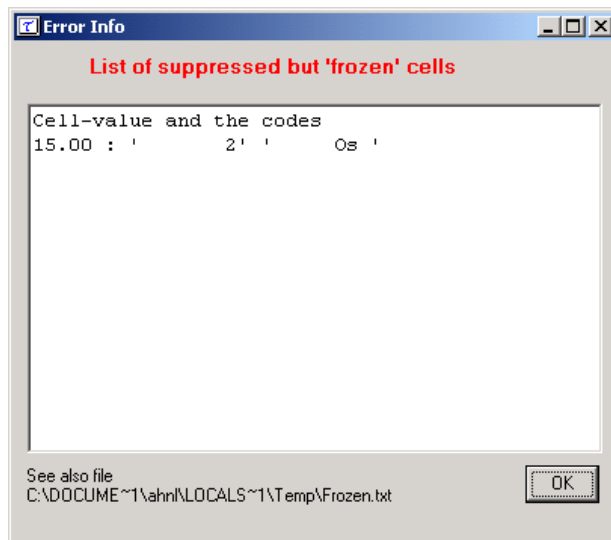
- Note that step 11 will make cells eligible for secondary suppression that τ -ARGUS considers as ‘protected’ (so called ‘frozen’ cells, for discussion of this option see for instance [5]).

⁶ The file will not be produced for tables with more than four dimensions or more than 50 000 cells.

As this is inconsistent with the current view on protected cells in τ -ARGUS this will lead to the following error message:



Codes and cell values of those suppressed *frozen* cells are then displayed by ARGUS:



When the status of these cells is changed into 'unprotected' before re-running the hypercube method, the solution will be a feasible solution for τ -ARGUS.

2.8.1 References on GHMiter

- [1] Repsilber, R. D. (1994), 'Preservation of Confidentiality in Aggregated data', paper presented at the Second International Seminar on Statistical Confidentiality, Luxembourg, 1994
- [2] Repsilber, D. (1999), 'Das Quaderverfahren' - in *Forum der Bundesstatistik, Band 31/1999: Methoden zur Sicherung der Statistischen Geheimhaltung*, (in German)
- [3] 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)
- [4] 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.
- [5] Giessing, S. (2003), 'Co-ordination of Cell Suppressions: strategies for use of GHMITER', Proceedings of the Joint ECE/Eurostat work session on statistical data confidentiality (Luxembourg, 7-9 April 2003)

2.9 *Hitas*

HiTaS is a heuristic approach to cell suppression in hierarchical tables. Hierarchical tables are specially linked tables: at least one of the spanning variables exhibits a hierarchical structure, *i.e.* contains (many) sub-totals.

In Fischetti and Salazar (1998) a theoretical framework is presented that should be able to deal with hierarchical and generally linked tables. In the sequel this will be called the mixed integer approach. In that framework, additional constraints to a linear programming problem are generated. The number of added constraints however, grows rapidly when dealing with hierarchical tables, since many dependencies exist between all possible (sub-)tables containing many (sub-)totals. The implemented heuristic approach (HiTaS) deals with a large set of (sub-)tables in a particular order. A non hierarchical table can be considered to be a hierarchical table with just one level. In that case, the approach reduces to the original mixed integer approach and hence provides the optimal solution. In case of a hierarchical table, 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.

In the following section, a short description of the approach is given. For a more detailed description of the method, including some examples, see *e.g.*, De Wolf (2002).

HiTaS deals with cell suppression in hierarchical tables using a top-down approach. The first step is to determine the primary unsafe cells in the base-table consisting of all the cells that appear when crossing the hierarchical spanning variables. This way all cells, representing a (sub-)total or not, are checked for primary suppression. Knowing all primary unsafe cells, the secondary cell suppressions have to be found in such a way, that each (sub-)table of the base-table is protected and that the different tables cannot be combined to undo the protection of any of the other (sub-)tables. 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 is then transported to the corresponding marginal cells of the tables that appear when crossing lower levels of the two variables. All marginal cells, both suppressed and not suppressed, are then ‘fixed’ in the calculation of the secondary suppressions of that lower level table, *i.e.*, they are not allowed to be (secondarily) suppressed. This procedure is then repeated until the tables that are constructed by crossing the lowest levels of the spanning variables are dealt with.

A suppression pattern at a higher level only introduces restrictions on the marginal cells of lower level tables. Calculating secondary suppressions in the interior while keeping the marginal cells fixed, is then independent between the tables on that lower level, *i.e.*, all these (sub-)tables can be dealt with independently of each other. Moreover, added primary suppressions in the interior of a lower level table are dealt with at that same level: secondary suppressions can only occur in the same interior, since the marginal cells are kept fixed.

However, when several empty cells are apparent in a low level table, it might be the case that no solution can be found if one is restricted to suppress interior cells only. Unfortunately, backtracking is then needed.

Obviously, all possible (sub)tables should be dealt with in a particular order, such that the marginal cells of the table under consideration have been protected as the interior of a previously considered table. To that end, certain groups of tables are formed in a specific way (see De Wolf (2002)). All tables within such a group are dealt separately, using the mixed integer approach.

The number of tables within a group is determined by the number of parent-categories the variables have one level up in the hierarchy. A parent-category is defined as a category that has one or more sub-categories. Note that the total number of (sub-)tables that have to be considered thus grows rapidly.

Singletons

Singleton cells should be treated with extra care. The single respondent in this cell could easily undo the protection if no extra measures were taken. The most dangerous situation is that there are only two

singletons in a row, or one singleton and one other primary unsafe cell. These singletons could easily disclose the other cell.

In the current implementation we have made sure that at least two singletons in one row or column cannot disclose each other information. For this we will increase the protection margins of these singletons such that the margin of the largest is greater than the cell-value of the smallest.

References on HITAS

Fischetti, M. and J.J. Salazar-González (1998). *Models and Algorithms for Optimizing Cell Suppression in Tabular Data with Linear Constraints*. Technical Paper, University of La Laguna, Tenerife.

P.P. de Wolf (2002). *HiTaS: a heuristic approach to cell suppression in hierarchical tables*. Proceedings of the AMRADS meeting in Luxembourg (2002).

Additional reading on the optimisation models can be found at the CASC-website (<http://neon.vb.cbs.nl/casc/RelatedPapers.html/99wol-heu-r.pdf>)

2.10 Network Solution for Large Unstructured 2 dimensional tables

Here only the introduction and references to a large document by Jordi Castro are shown.

The network flows package for cell suppression (NF CSP) implements two heuristics for the protection of statistical data in two-dimensional tables. The heuristics are improved versions (i.e., faster) than those originally presented in [5] and [7] for the secondary cell suppression problem. General details about the algorithms implemented in NF CSP can be found in [3]; a thorough description will be provided in a future paper.

In the first heuristic, derived from [5], only flows 0 or 1 are sent through the network. We will refer to it as the "0-1-flows" heuristic. The second will be denoted as the "n-flows" heuristics, since the network can transport any positive flow.

The current package is linked with three network flows solvers: CPLEX 7.5 [6], PPRN [4], and an efficient implementation of the bidirectional Dijkstra's algorithm for shortest-paths (that will be denoted as "Dijkstra") [1]. Later releases of CPLEX will also work if the interface routines are the same than for version 7.5. The 0-1-flows heuristic can use any of the three solvers. The network flows problems formulated by the n-flows heuristic can only be solved with PPRN and CPLEX. PPRN and Dijkstra were implemented at the Dept. of Statistics and Operations Research of the Universitat Politècnica de Catalunya, and are included in NF CSP. PPRN was originally developed during 1992–1995, but it had to be significantly improved within the CASC project to work with NF CSP. Dijkstra was completely developed in the scope of CASC.

The third solver, CPLEX 7.5, is a commercial tool, and requires purchasing a license. However, PPRN is a fairly good replacement—although not so robust—for the network flows routines of CPLEX7.5. Therefore, in principle, there is no need for an external commercial solver. More-over, solver Dijkstra is by far the most efficient option, although it can only deal with problems formulated by the 0-1 flows heuristics. It should be used whenever possible for efficiency reasons.

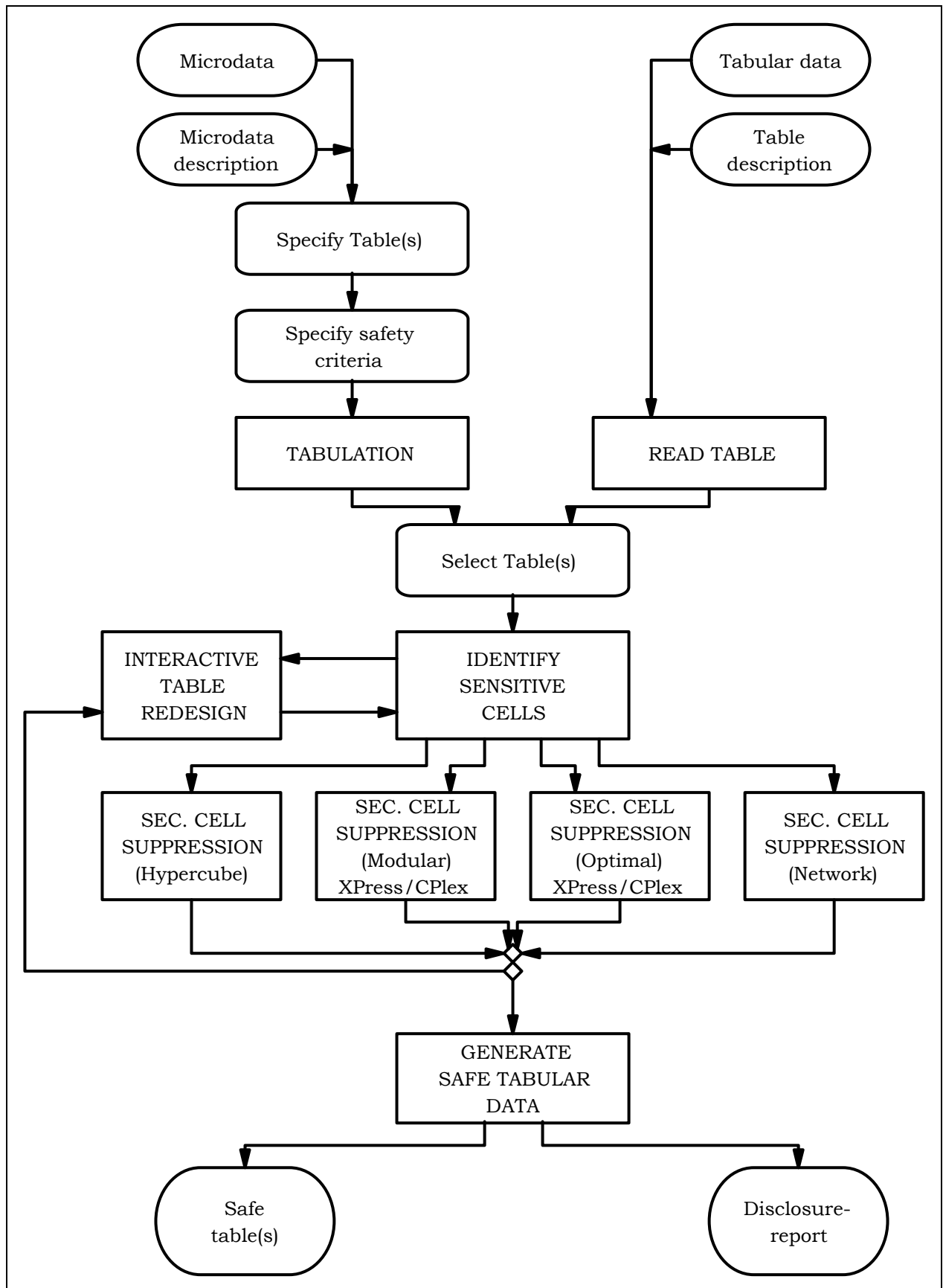
Even though two of the three solvers are included in the distribution of NF CSP, this document only describes the features of the heuristics, and from the user's point of view. A detailed description of PPRN and Dijkstra's solvers can be found in [2, 4] and [1], respectively.

A full description of the network flows solution can be found in a separate document [8]. The structure of the document is as follows. Section 2 introduces a simple program that shows how to use NF CSP from the user's application. Section 3 describes the main options and features of the package. In Section 4 we present the set of routines to interface with NF CSP, grouped by functional categories. A final Appendix lists all the files and routines of NF CSP.

[1] Ahuja, R.K., Magnanti, T.L., Orlin, J.B., Network Flows, Prentice Hall (1993).

- [2] Castro, J., PPRN 1.0, User's Guide, Technical report DR 94/06 Dept. of Statistics and Operations Research, Universitat Politècnica de Catalunya, Barcelona, Spain, 1994.
- [3] Castro, J., Network flows heuristics for complementary cell suppression: an empirical evaluation and extensions, in LNCS 2316, Inference Control in Statistical Databases, J. Domingo-Ferrer (Ed), (2002) 59–73.
- [4] Castro, J., Nabona, N. An implementation of linear and nonlinear multicommodity network flows. European Journal of Operational Research 92, (1996) 37–53.
- [5] Cox, L.H., Network models for complementary cell suppression. J. Am. Stat. Assoc. 90, (1995) 1453–1462.
- [6] ILOG CPLEX, ILOG CPLEX 7.5 Reference Manual Library, ILOG, (2000).
- [7] Kelly, J.P., Golden, B.L, Assad, A.A., Cell Suppression: disclosure protection for sensitive tabular data, Networks 22, (1992) 28–55.
- [8] Castro, J. 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/41D5-NF-Tau-Argus.pdf>

2.10 2.11 Functional design of τ -ARGUS



3. A tour of τ -ARGUS

This section will give the reader an introduction to the use of τ -ARGUS. Some Windows experience is assumed. In section 4 a more systematic description of the different parts of τ -ARGUS will be given.

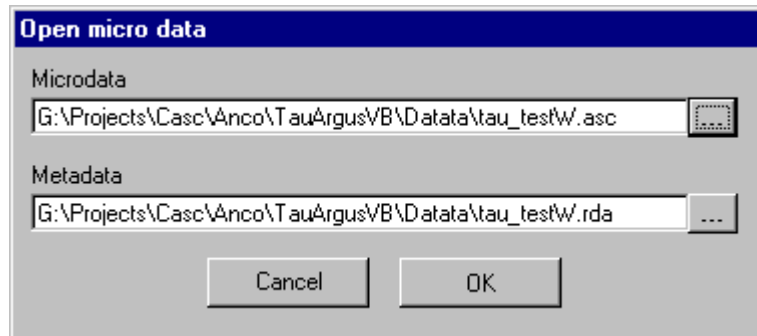
3.1 Preparation

To start the disclosure control with τ -ARGUS there are two possible options.

- Open a microdata file (3.1.1 - 3.1.3).
- Open a table (3.1.4).

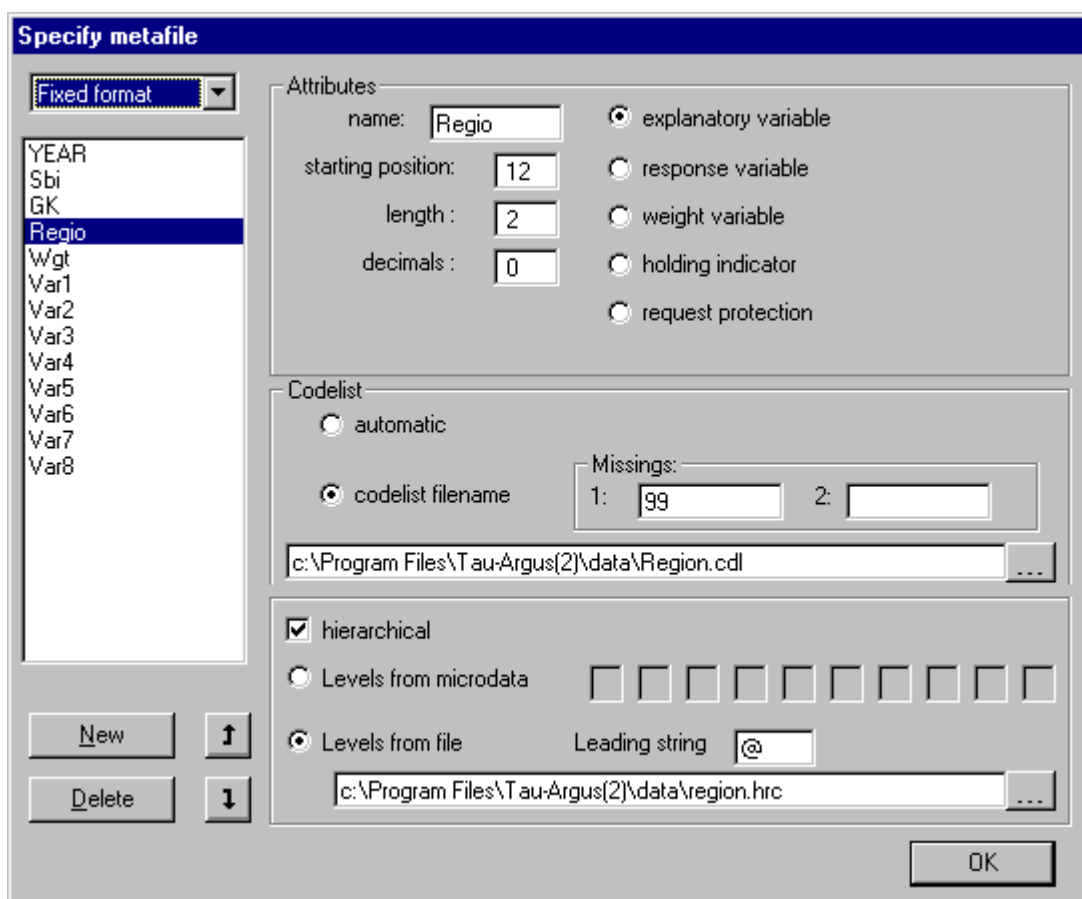
3.1.1 Open a microdata file

Both a microdata file and the metadata describing this microdata file are required. The microdata file must be either a fixed format ASCII file or a free format file with a specified separator. If you click (File|Open Microdata) you can specify the name of the microdata file and the name of the file containing the metadata.



The program assumes the extension .ASC for the datafile and .RDA for the metadata, but you can use your own extensions. The metadata is stored in a separate file. If the name of the metadata file is the same as the datafile, except for the extension, τ -ARGUS will fill in this file automatically. If no metadata file is specified, the program has the facility to let the user specify the metadata interactively via the menu option (Specify|Metafile). This is also the place to make changes to the metadata. In subsection 3.1.2 we will give a description of the metadata file for τ -ARGUS.

When you enter or change the metadata interactively using τ -ARGUS the option (Specify|Metafile) will bring you to this screen:



The left pane shows the names of the variables. Besides information on the position of the variables you can indicate whether a variable can be used as explanatory variable in a table or as a response variable. It is also possible to indicate that one variable can be used as a sample weight variable. These sample weights can be used in applying the safety rules (see 3.1.2)

As is shown on the top left of the screen, the input datafile can be entered in fixed format or in free format with a specified separator.

In the bottom half of the window all kind of details on the codelist can be stored.

- The codelists: τ -ARGUS will explore the datafile and build the codelist for the explanatory variables. This is the automatic option, However additionally the user can specify a codelist file. This codelist will be used to provide more meaningful labels attached to the codes in some of the screens of τ -ARGUS.

Missing values. τ -ARGUS needs to know which missing values are attached to a codelist. Additionally for each code the missing values (at least one) should be specified. In many surveys two missing values are used; e.g. one for *don't know* and one for *refusal*.

- Hierarchical codes. As τ -ARGUS has now the facilities to protect hierarchical tables, you can instruct τ -ARGUS here on the nature of these hierarchical codelists. There are basically two options.
 1. The hierarchy can be derived from the digits of the individual codes. Each digit denotes a level and if required some digits can be grouped together.
 2. A file containing the hierarchical structure is specified. In this file the level of the nesting is indicated by a special character/string. In the example above the @ had been selected.

3.1.2 The Metafile

The metafile describes the variables in the microdata file, both the record layout and some additional information necessary to perform the SDC-process. Each variable is specified on one main line followed by one or more option lines.

1. The first line gives the name of the variable followed by the starting position for each record, the width of the field and either one or two missing value indicators for the record.
2. The following lines specify specific characteristics of the variable:

• <RECODEABLE>	This variable can be recoded and used as an explanatory variable in a table
• <CODELIST>	This explanatory variable has a codelist. The name of the codelist file follows this keyword. Default extension .CDL
• <NUMERIC>	This variable can be used as cell-item.
• <DECIMALS>	The number of decimal position for this variable
• <WEIGHT>	This variable contains the weighting scheme
• <HIERARCHICAL>	This variable is hierarchical
• <HIERLEVELS>	The hierarchy is derived from the digits of the codes itself. The specification is followed by a list of integers denoting the width of each level. The sum of these integers should be the width of the total code
• <HIERCODELIST>	The name of the file describing the hierarchical structure. Default extension .HRC
• <HIERLEADSTRING>	The string/character that is used to indicate the depth of a code in the hierarchy

An example of a metadata file is shown below.

Here the variable 'Year' for each record begins on position 1, is 2 characters long and missing values are represented by 99. It is also recodeable.

The following lines give the information for variable 'Sbi'. This variable begins on position 4 and is 5 characters long. Missing values are represented by 99999 and as well as being recodeable this variable is hierarchical and the hierarchy levels are shown. The first 3 characters are in the top hierarchy level, the 4th character in the second level and the 5th character in the lowest level.

```

YEAR 1  2 99
  <RECODEABLE>
Sbi 4  5 99999
  <RECODEABLE>
  <HIERARCHICAL>
  <HIERLEVELS> 3 1 1 0 0
GK 9  2 99
  <RECODEABLE>
Regio 12  2 99
  <RECODEABLE>
  <CODELIST> Region.cdl
  <HIERCODELIST> Region2.hrc
  <HIERLEADSTRING> @
  <HIERARCHICAL>
Wgt 14  4 9999
  <NUMERIC>
  <DECIMALS> 1
  <WEIGHT>
Var1 19  9 999999999
  <NUMERIC>
Var2 28 10 999999999
  <NUMERIC>

```

```
<DECIMALS> 2
.....
.....
```

An example of a codelist file (Region.CDL)

Here 1 represents Groningen etc.

```
1, Groningen
2, Friesland
3, Drenthe
4, Overijssel
5, Flevoland
6, Gelderland
7, Utrecht
8, Noord-Holland
9, Zuid-Holland
10, Zeeland
11, Noord-Brabant
12, Limburg
Nr, North
Os, East
Ws, West
Zd, South
```

Example of a file with the hierarchical structure (Regio.HRC)

Here Regions 1,2 and 3 (Groningen, Friesland and Drenthe) are part of the North etc.

```
Nr
@ 1
@ 2
@ 3
Os
@ 4
@ 5
@ 6
@ 7
Ws
@ 8
@ 9
@10
Zd
@11
@12
```

3.1.3 Specification.

When the metadata is ready, the user can specify the tables he/she wants to protect. Via Specify|Tables you come into a window to specify the tables. In the current version of τ -ARGUS you can specify more than one table, but each table is protected separately, unless you have a set of linked tables. This version of ARGUS has a first implementation of linked tables.. In the upper part of the window, the first pane shows the explanatory variables and the third pane shows the response variables.

The explanatory variables

On the left is the listbox with the explanatory variables

When you click on ‘>’ or ‘<’ you transport the selected variable to the next box. From the left box with explanatory variables you can select the variables that will be used in the row or the column of the table. Up to four explanatory variables can be selected to create a table.

The response variable

The response variable is the variable that will be used to calculate the cell totals. From the list of response variables you can select a variable as response variable (the cell-item). This is the variable for which the table to be protected is calculated i.e. used to calculate the cell totals.

The shadow variable

The shadow variable is the variable that is used to apply the safety rule. By default this is the response variable, but it is possible to select another variable. The safety rules are built on the principle of the characteristics of the largest contributors to a cell. If a variable other than the response variable is a better indicator for the size of a company, this variable can be used here ; e.g. the turnover (as a proxy for the size of the enterprises) can be a suitable variable to apply the dominance rule, although the table is constructed with an other variable.

The cost variable

This variable describes the cost of each cell. These costs are minimised when the secondary suppressed cells are calculated (See section 2.5). By default this is the response variable but another choice is possible. It is also possible to use the frequency of the cells as a cost-function, This will minimise the number of records contributing to the cells to be suppressed. A third option is that the number of cells to be suppressed is minimised, irrespective of the size of their contributions (unity option).

Weight

If the data file has a sample weight, specified in the meta data, the table can be computed taking this weights into account by clicking the ‘Apply Weights’ option’. The safety rules (see below) have been extended to allow the weights to be applied here by clicking on the ‘Apply Weights in Safety Rule’ option.

The safety rule

The concept of safety rules is explained in section 2.1. On the left side of the window the type of rule can be selected and the value of the parameters. Additionally the minimum number of contributors can be chosen.

For the dominance rule and the $p\%$ -rule safety ranges can be derived automatically. The theory gives formulas for the upper limit only, but for the lower limit we have chosen a symmetric range. However the theory does not provide any safety range for the frequency rule and of course not for the cells that have been made unsafe by manual intervention of the user. So in these two cases the user must provide a safety-range percentage.

The parameters for the dominance rule (see section 2.1) can be specified. Not only the parameters $(n, k\%)$ for the dominance rule can be specified, but also the minimum number of records contributing to a cell can be specified. Alternatively the minimum protection of the $(p\%)-$ rule can be used to identify the primary unsafe cells.

It is also possible to specify no safety rule apart from a minimum frequency value. A further option is to apply the Request Rule. This is a special option applicable in certain countries relating to foreign trade statistics. Here cells are protected when the largest contributor represents over (for example) 70% of the total and that contributor asked for protection. Therefore you need a variable indicating the request. This rule cannot be applied along with any other rule. The request code asked for will replace the value for that contributor.

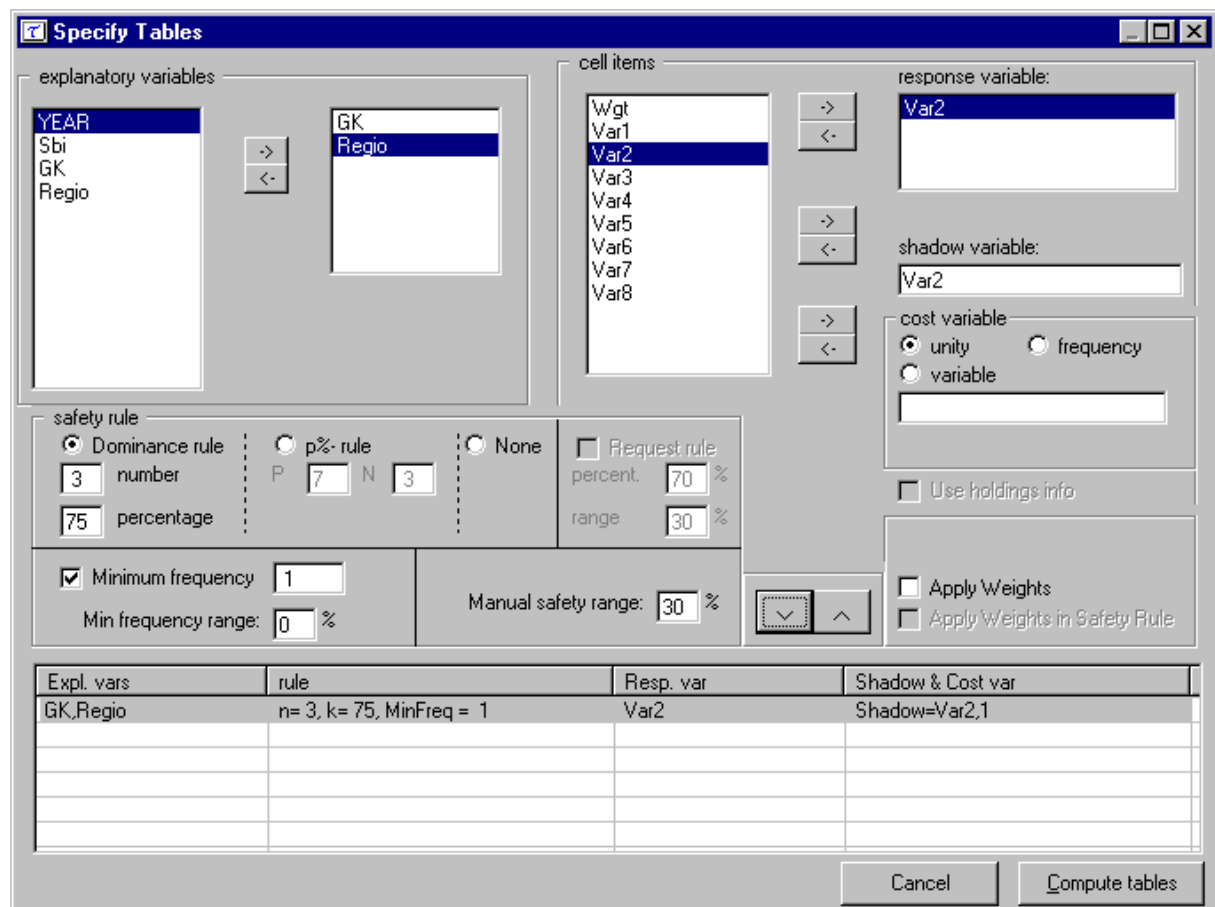
When a cell is set manually unsafe (an option to discussed later), τ -ARGUS cannot calculate safety-ranges itself. So the user must supply a safety-percentage manually. The safety range

corresponding to the minimum frequency rule is approximately zero. So, as a rule, a small positive value should be entered in the box Minimum frequency range.

When you have filled in everything you click ‘√’ to transport all the specified parameters to the ‘listwindow’ on the bottom. You can specify as many tables as you want, but as the size of the memory of a computer is still restricted you should not overdo it. If you want to modify an already made table you press the ‘^’ button.

More than one Response Variable can be specified by the user. This will produce tables for each of each of the Response Variables using the Spanning variables specified.

Pressing the ‘Compute tables’ button will invoke τ -ARGUS to actually compute the tables requested and you are ready to start the process of disclosure control. τ -ARGUS will come back with the main window showing you the number of unsafe cells per variable per dimension.

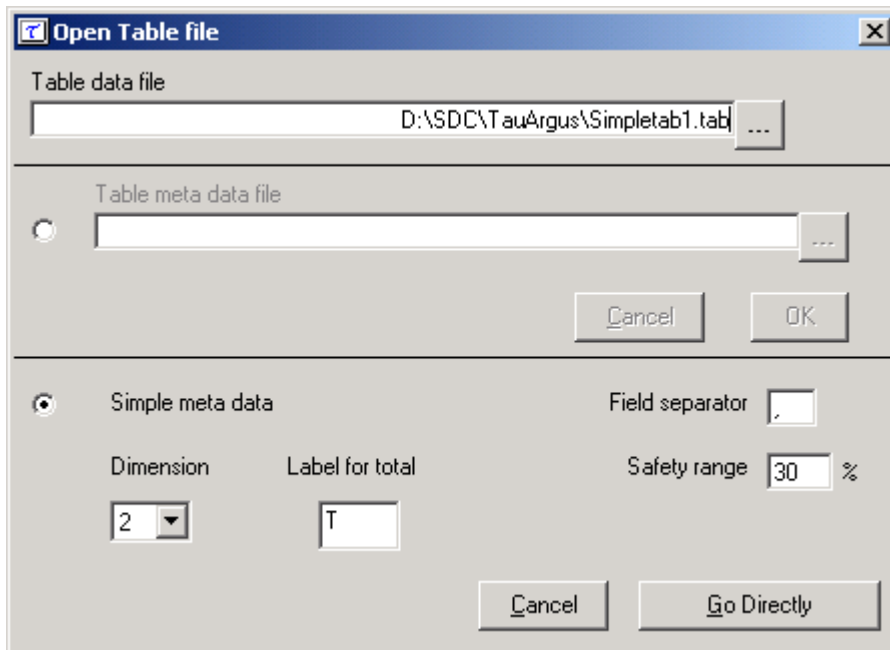


3.1.4 Open a Table

This is the second of these options outlined at the beginning of section 3.1 and is reached by selecting ‘Open a Table’ on the main window of Tau-Argus.

The datafile containing the table to be opened (in the format given below) needs to be specified in the top line. A metadata file can be entered explicitly by the user in this window or created within the program. A simple alternative is to apply the simple metadata option where the dimension of the table is specified along with a label for the marginal totals.

If the ‘Simple metadata’ route is followed, click on ‘Go Directly’ to create the table. Here basic metadata can be generated. The ‘Manual Safety Range’ is equivalent to ‘Manual Safety Range’ in the ‘Specify tables’ window. The ‘Field Separator’ distinguishes between the cells in the datafile. The next allowed operation is to view the table.



If the ‘Table metadata file’ option is chosen (even if no file is selected) the ‘OK’ button can be clicked and this allows the metafile to be specified under ‘Specify|Metafile’. Here the variables can be specified as required. The options are:

- ‘Explanatory Variable’ – The spanning variables used to produce the table.
- ‘Response Variable’- The variable used to calculate the cell totals.
- ‘Weight Variable’ – If each cell has an associated weight, the variable can be declared here.

‘Frequency’ – This indicates the number of observations making up the cell total. If there is no frequency variable each cell is assumed to consist of a single observation.

‘topN variable’ – This shows the values of the top n contributors to each cell. The pre-defined value for TopN is 1. The first variable declared as ‘topN’ will contain the largest values in each cell, the second variable so declared will contain the second largest values *etc.*

‘Status Indicator’ allows the Status option to be highlighted in the left-hand pane and the status codes for each cell can be changed. The most frequently occurring codes are to declare a cell either Safe or Unsafe.

The default name for the explanatory variable occurring first in each line of the file containing the table is 'ExpVar1'.

Below are displayed the 'Specify metafile' windows as they look for both an explanatory variable and a status variable when there is no frequency variable.

Specify metafile

Free format:

Separator:

ExpVar1
ExpVar2
RespVar
Status

New
Delete

Attributes:

name:

length:

decimals:

explanatory variable frequency
 response variable topN variable
 weight variable status indicator
 holding indicator
 request protection

Codelist:

automatic codelist filename

Code for Total:

Missings: 1: 2:

hierarchical

Levels from microdata

Levels from file Leading string:

Specify metafile

Free format

Separator

ExpVar1
ExpVar2
RespVar
Status

Attributes

name: explanatory variable frequency
 response variable topN variable
length: weight variable status indicator
decimals: holding indicator
 request protection

safe
unsafe
protect

Codelist

automatic Code for Total
 codelist filename Missings: 1: 2:

hierarchical

Levels from microdata

Levels from file Leading string

New
Delete

OK

Below the 'Specify metafile' window is displayed where frequency data is present.

Specify metafile

Free format

Separator

Expvar1
Expvar2
Respvar
Freq
Top1
Top2
Top3
Status

Attributes

name: explanatory variable frequency
 response variable topN variable
length: weight variable status indicator
decimals: holding indicator
 request protection

Codelist

automatic Code for Total
 codelist filename Missings: 1: 2:

hierarchical

Levels from microdata

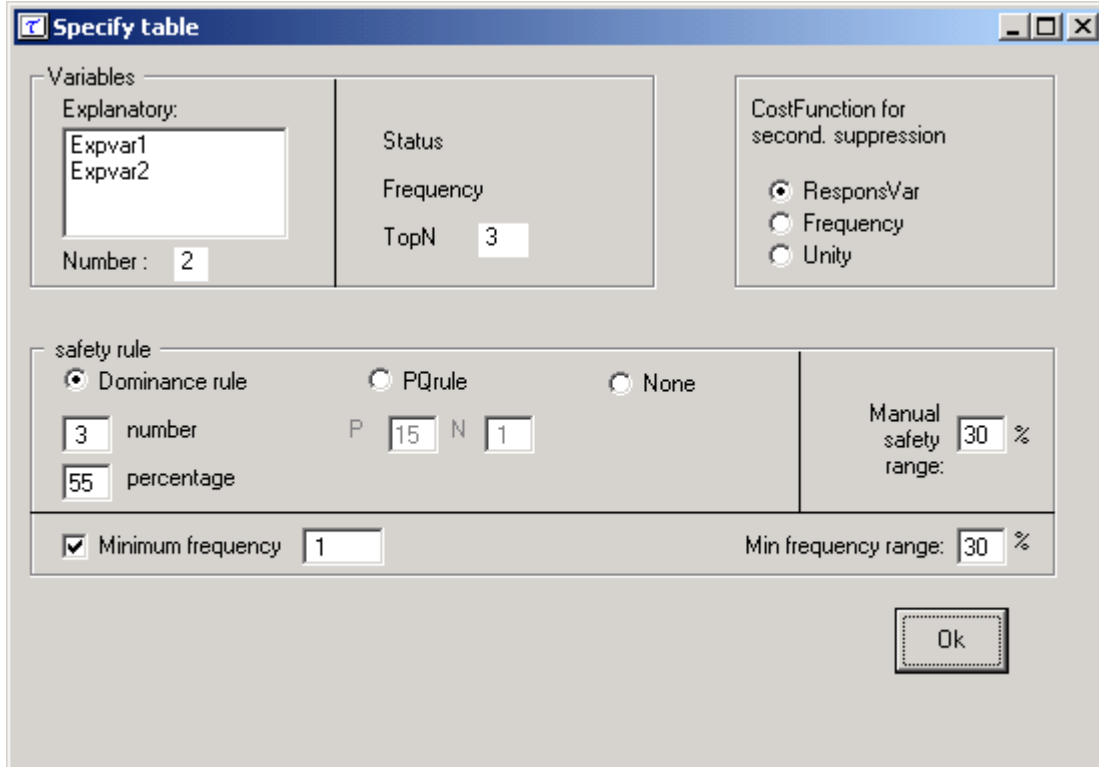
Levels from file Leading string

New
Delete

OK

When the 'Specify|Metafile' option is followed the 'Specify|Table metadata' option is also available and the window is displayed here. This will allow the application of safety rules such as the Dominance Rule and the p% rule, which are described in sections 2.1 and 3.1.3.

As also outlined in section 3.1.3 there are options for Cost function for secondary suppression. If a minimum frequency for cell safety if required, this can be entered in this window as well.



Two examples of datafile for 2 way tables are shown here. Note the cells have already been specified manually to be unsafe or safe.

Datafile 1 shows 2 explanatory variables, 1 response variable and an indication of whether a cell is safe or unsafe.

```

T,  T, 1200 , u
T,  A,  300 , s
T,  B,  300 , s
T,  C,  300 , s
T,  D,  300 , s
1,  T,  400 , s
1,  A,  100 , s
1,  B,  100 , s
1,  C,  100 , s
1,  D,  100 , s
2,  T,  400 , s
2,  A,  100 , s
2,  B,  100 , u
2,  C,  100 , u
2,  D,  100 , s
3,  T,  400 , s
3,  A,  100 , s
3,  B,  100 , s
3,  C,  100 , s
3,  D,  100 , s

```

Datafile 2 shows 2 explanatory variables, 1 response variable, cell frequency, the top 3 values in each cell and an indication of whether a cell is safe or unsafe.

```

T,  T, 1200 ,12, 40,30,20 ,u
T,  A, 300 ,3, 40,30,20 ,s
T,  B, 300 ,3, 40,30,20 ,s
T,  C, 300 ,3, 40,30,20 ,s
T,  D, 300 ,3, 40,30,20 ,s
1,  T, 400 ,4, 40,30,20 ,s
1,  A, 100 ,3, 40,30,20 ,s
1,  B, 100 ,3, 40,30,20 ,s
1,  C, 100 ,3, 40,30,20 ,s
1,  D, 100 ,3, 40,30,20 ,s
2,  T, 400 ,4, 40,30,20 ,s
2,  A, 100 ,3, 40,30,20 ,s
2,  B, 100 ,3, 40,30,20 ,s
2,  C, 100 ,3, 40,30,20 ,u
2,  D, 100 ,3, 40,30,20 ,u
3,  T, 400 ,4, 40,30,20 ,s
3,  A, 100 ,3, 40,30,20 ,s
3,  B, 100 ,3, 40,30,20 ,s
3,  C, 100 ,3, 40,30,20 ,s
3,  D, 100 ,3, 40,30,20 ,s

```

The next stage is to view the table and the operations that follow are as for entering metadata. Below is an example of the table for datafile 1, with the unsafe cells declared on the data file highlighted. This table will be explained in depth in section 3.2.1.

The screenshot displays the 'Table: ExpVar1 x ExpVar2' window. On the left is a table with columns 'tot', 'A', 'B', 'C', 'D', and 'X'. The 'tot' column has values 1,200, 400, 400, 400, and a red 'X'. The 'A' column has values 300, 100, 100, 100, and a red 'X'. The 'B' column has values 300, 100, 100, 100, and a red 'X'. The 'C' column has values 300, 100, 100, 100, and a red 'X'. The 'D' column has values 300, 100, 100, 100, and a red 'X'. The 'X' column has values -, -, -, -, and a red 'X'. On the right is the 'Cell Information' panel with fields for Value (1,200), Status (Unsafe (manual)), Cost (1,200), Shadow (1,200), # contributions (1), Top n of shadow, and Up/Low levels (360, 360). Below this are buttons for 'Set to Safe', 'Set to Unsafe', 'Set to Protected', 'Read Hist.', 'Recode', and 'Suppress' options (HyperCube, Modul/XPress, Modul/Cplex, Opt/Netw., Opt/XPress, Opt/Cplex). At the bottom are buttons for 'Write table' and 'Close'. At the bottom left are checkboxes for '3 dig. separator' and 'Output View', and buttons for 'Select Table', 'Change View', and 'Table Summary'.

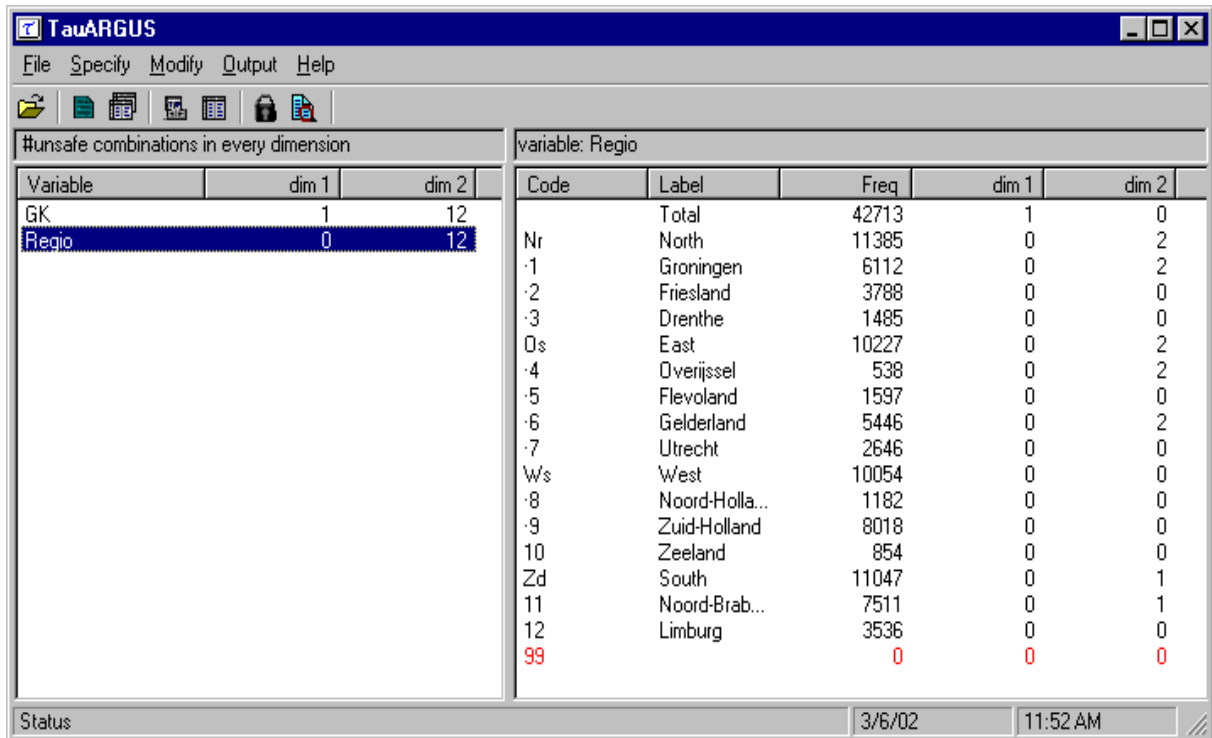
	tot	A	B	C	D	X
tot	1,200	300	300	300	300	-
1	400	100	100	100	100	-
2	400	100	100	100	100	-
3	400	100	100	100	100	-
X	-	-	-	-	-	-

3.2 The process of disclosure control

When the table(s) have been calculated, the main-window of τ -ARGUS will show again with an overview of all the unsafe cells per variable (over all the tables). If you had specified more than one table you have to choose a table with 'Modify|Select table'. In this example you can go directly to 'Modify|View table'.

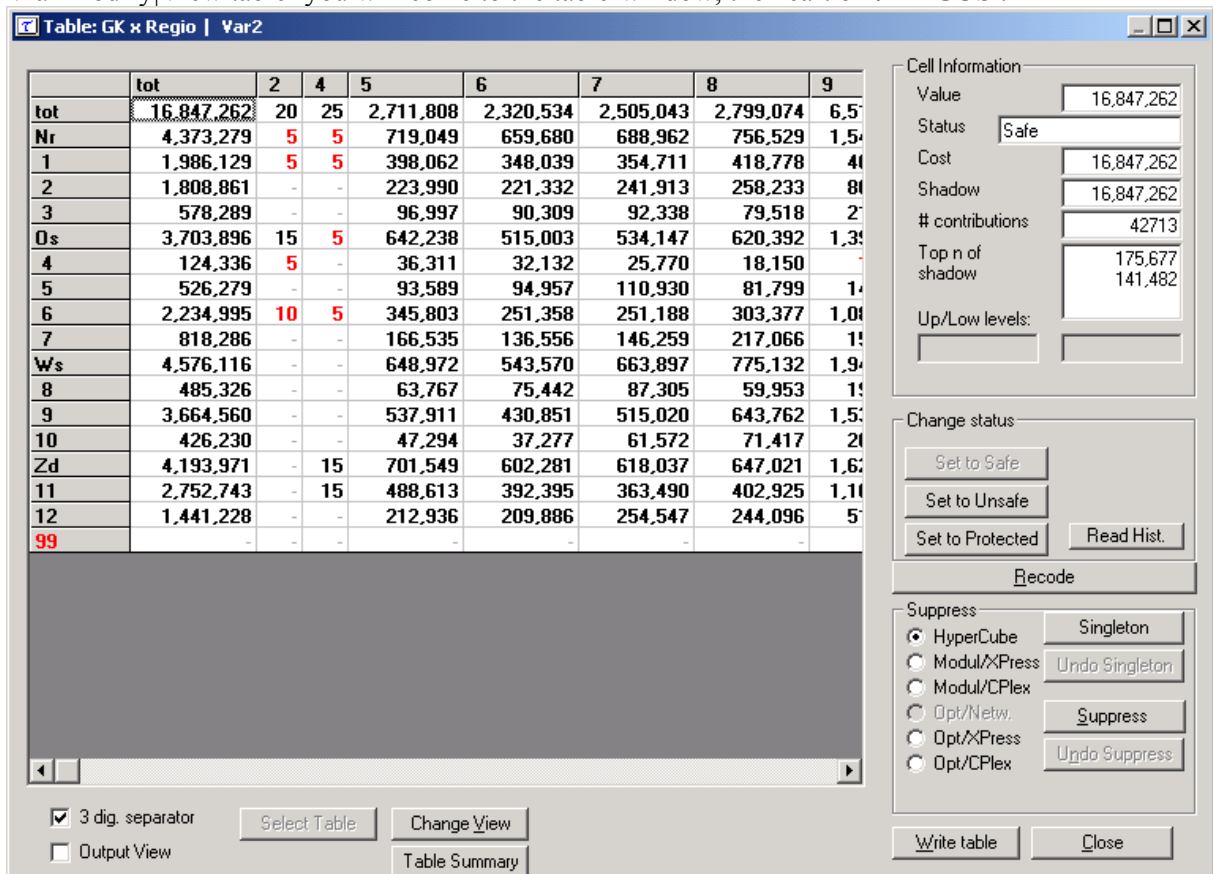
The window below is the main menu for τ -ARGUS, showing the number of unsafe combinations per variable. For example there are 12 unsafe cells in the 2 dimensional aspects of the table and a single unsafe cell for a one way marginal total for the variable 'GK'.

The right hand window gives the equivalent information for each level of the variable indicated on the left.



3.2.1 Table protection

Via 'Modify|View table' you will come to the table-window, the heart of τ -ARGUS :



This window shows the table you have selected with Modify|View Table. On the left side the table itself is shown in a spreadsheet view. Safe cells are black, unsafe cells are red, secondary suppressed cells are blue and empty cells have a hyphen (-). The two check-boxes on the left-bottom give you some control over the layout.

- Clicking on the 3-digit separator will show the cell-values with this separator. The separator is chosen according to the general Windows settings
- The Output view shows the table, with all the suppressed cells replaced by an 'X'; this is how the safe table will be published.

The options at the bottom of the table:

Via 'Change view' you can transpose the table. You simply indicate which variable will be the row-variable and which will be the column variable and the table will be transposed. If more than two explanatory variables had been selected the other variables will be in the layer and shown as combo-boxes on the top.

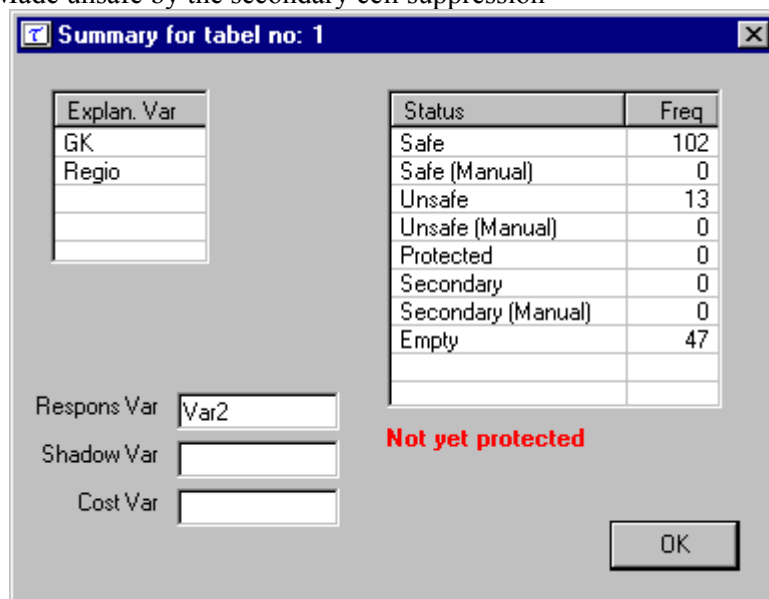
Clicking on 'Table-summary' will give you an overview of variables involved and the number of safe and unsafe cells at the particular stage (i.e. before or after recoding or secondary suppressions have been carried out).

When you click on a cell, information about this cell is visible in Cell Information pane
You can see the following information:

1. The cell-value
2. The cell status
3. The number of contributors to a cell
4. The largest contributors.

Status is the status of the cell, this can be:

- Safe: Does not violate the safety rule
- Safe (from manual): manually made safe during this session
- Unsafe: According to the safety rule
- Unsafe (from manual): manually made unsafe during this session
- Suppressed: Made unsafe by the secondary cell suppression



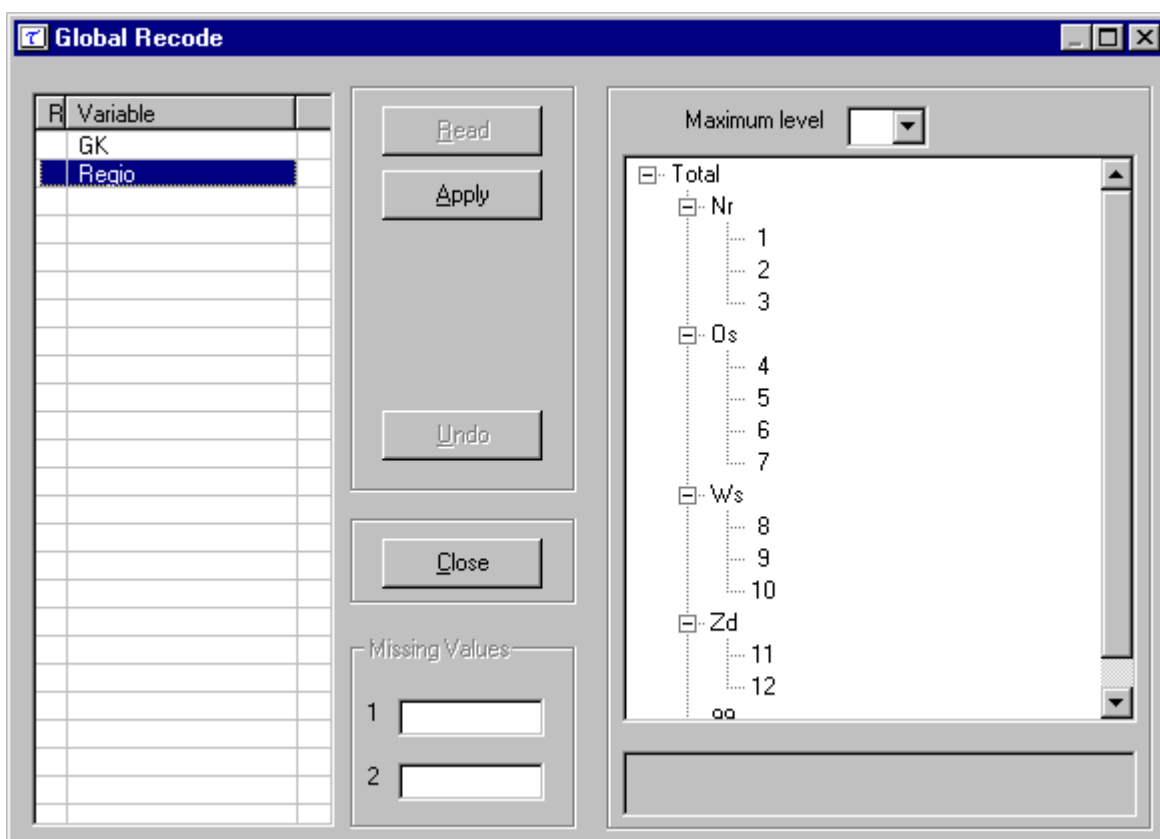
- Protected: Cannot be selected as a candidate for secondary cell suppression
- Zero: Value is zero and cannot be suppressed.
- Empty: No records contributed to this cell and the cell cannot be suppressed.

These status options for the cells can only be changed when it is logically correct to do so. For example after computing a table an Unsafe cell (i.e. selected for primary suppression) can be set to 'Safe' but not 'Protected' as a cell selected for primary suppression cannot be selected for secondary suppression. Of course if it is declared 'Safe' it is still a candidate for secondary suppression.

The second pane ('Change Status') on the right will allow you to change the cell-status. Only a few (logical) transitions are allowed. Changing the cell status can be useful if an unsafe cell may be published because of external reasons (permissions to publish). Changing to protect will prevent the system selecting this cell as a secondary suppression.

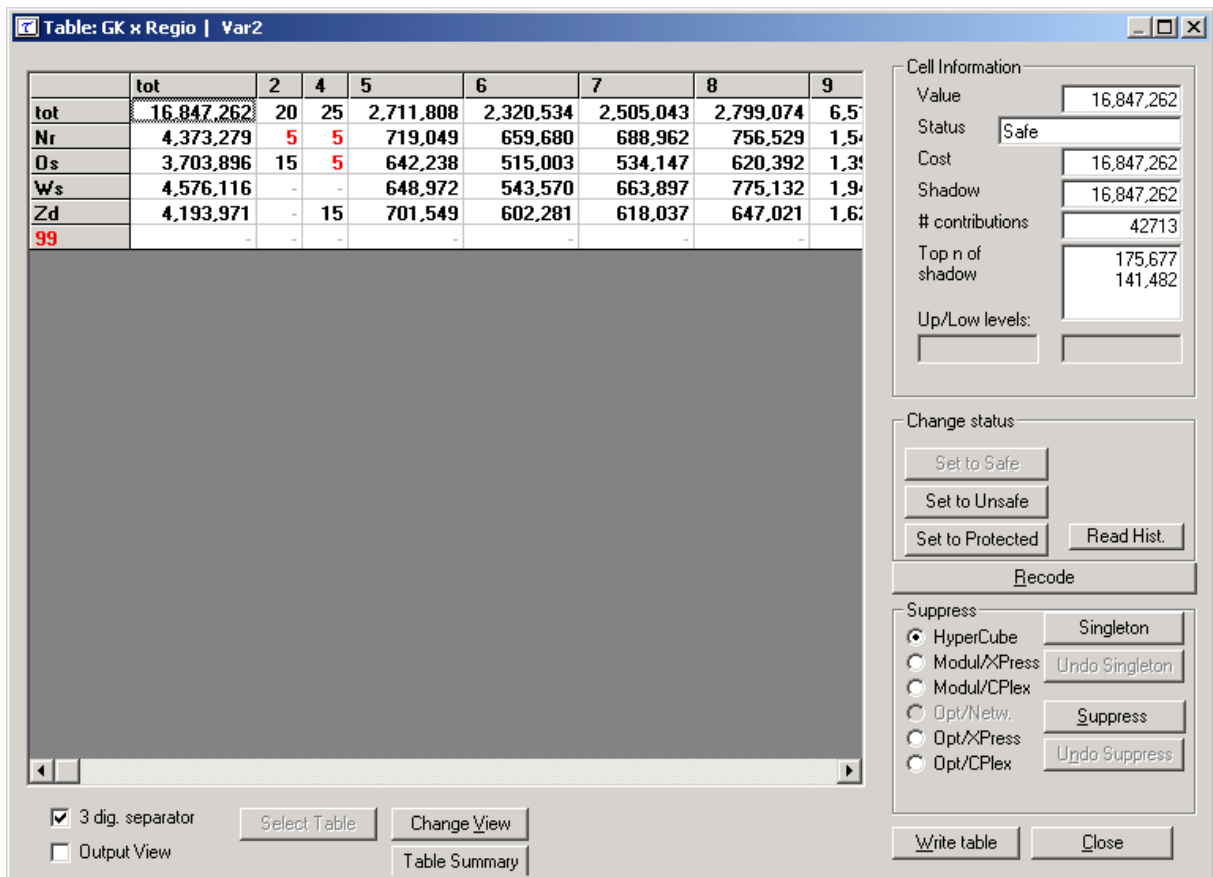
The recode button will bring you to the recoding system. Recoding is a very powerful method of protecting a table. Collapsed cells usually have more contributors and therefore tend to be much safer. Only microdata is available for recoding.

Specifying and selecting recodings

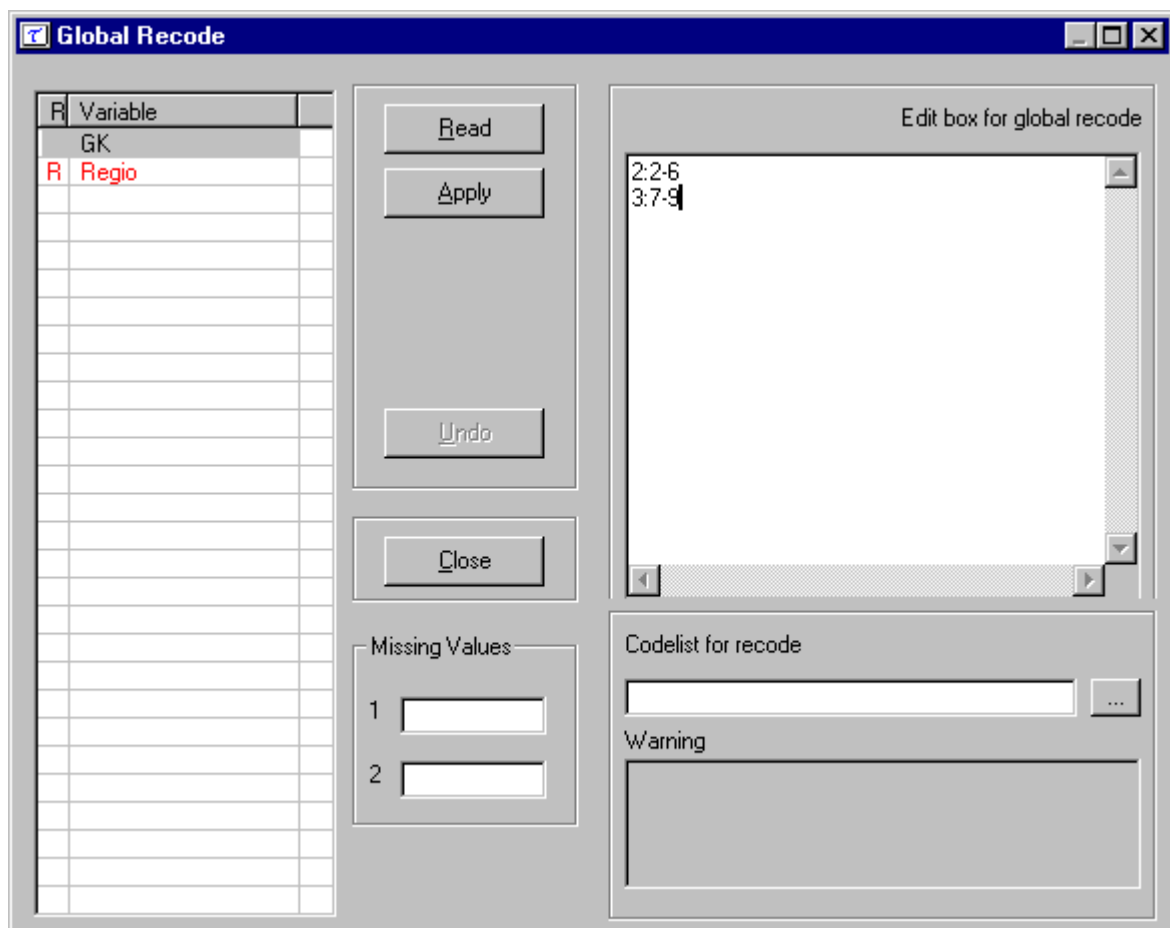


In the above example we have selected the 'Region' variable to recode. This window will behave differently whether the variable selected is hierarchical or not. In the hierarchical case the codes are shown in a hierarchical tree. The standard windows facilities to manipulate trees can be applied here as well. Folding and unfolding of branches is carried out by clicking on the '+' or '-' boxes which results in showing or omitting codes from the table. In this version of τ -ARGUS this is the only allowable recoding of a hierarchical codelist.

Pressing the 'Apply' button followed by 'Close' will actually apply the selecting recoding. The undo-button is possible to go back to the original recoding scheme. Below 'Region' has been recoded into 4 groups and the table now created is displayed.



In the case of a non-hierarchical codelist the right hand pane will be an editbox. In this editbox a recoding scheme can be specified as can be seen below with the variable 'GK'.



The syntax is as follows:

A recode groups together categories as they appear in the original microdata and makes each group a new category with a new code. The syntax of the recode file is as follows: each line in the file corresponds with one new category. The code of the new category is placed before a colon (:). The old categories to be grouped into the new category are placed behind the colon. Single categories are separated by commas (,) and if a hyphen (-) is placed between two categories it refers to all subsequent categories between and including these two categories. If a hyphen is only placed before or after a category, this refers to all categories before or after and including this category respectively.

Example: The line “7:-4,6,8-10,13-“ means that the categories 0, 1, 2, 3, 4, 6, 8, 9, 10, 13, 14, ... (if present) are recoded as 7.

Via the read-button you can read an existing recode scheme.

In the non-hierarchical case you could change the codes for missing values here, if you wish and indicate a new codelist.

The suppress-pane.

The Suppress button is the most important button. It will activate the modules for computing the necessary secondary suppressions. There are a number of options here.

- Hypercube
- Modul/Xpress
- Modul/Cplex
- Network
- Opt/Xpress
- Opt/Cplex

The ‘Singleton’ option is a special call to GHMiter to protect singleton cells only. See section 2.9. It is only sensible for the Opt/Xpress and Opt/Cplex options (i.e. the full optimisation routine). It uses a pre-processing GHMITER to protect singleton cells only.

The ‘Read Hist’ option is an *a priori* option to be mainly used for microdata which allows you to feed Tau-Argus a list of cells where the status of the standard rules can be overruled i.e. the status of the cells is already specified. It is free format. The format will be:

Code of first spanning variable, Code of second spanning variable, Status of cell

```
Nr, 4, u  
Zd, 6, p
```

The hypercube method (see section 2.7) will calculate a suppression pattern without using an LP-optimisation module. So it can be used without any additional licence.

In more detail, Opt/Xpress and Opt/Cplex are the full optimisation solutions while Modul/Xpress and Modul/Cplex are modular (partial) optimisation routines. The theory behind these modular routines is expanded in section 2.9. The Xpress and Cplex methods are two different implementation of HITAS, the search-algorithms described in section 2.9.

The Network approach is Jordi Castro’s network solution for non hierarchical two-dimensional tables. See section 2.10.

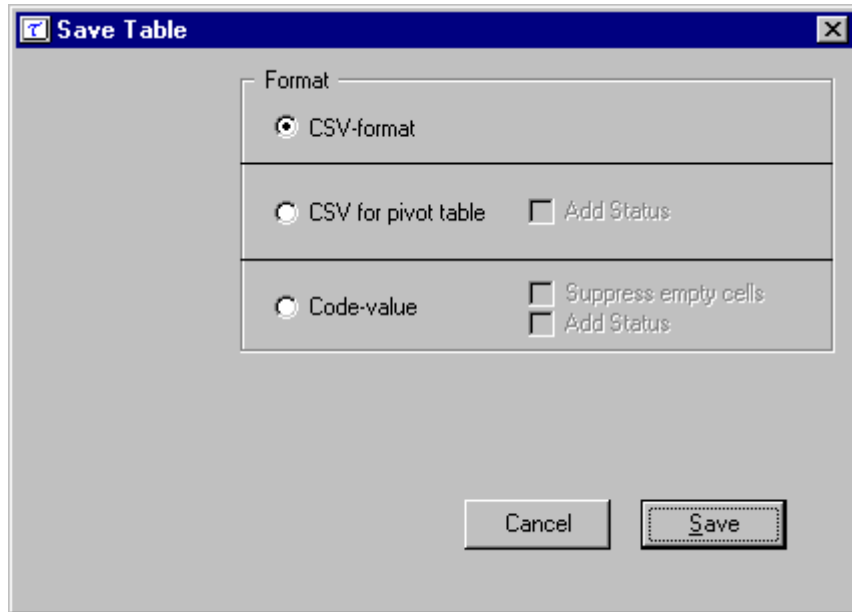
In every case you will in principle end with a safe table, indicated by the additional blue (secondary) cells.

You are now ready to store the table (with the ‘write table’) button. This is the same option as ‘Output|Save table’

3.3 Saving the safe table

When the table is safe it can be written to the hard disk of the computer. You have three options:

1. CSV-format; a format easily read by Excel.
2. CVS-format for pivot table. Nice for manipulating the table in Excel.
3. Code-Value. This displays the table as a text file with each line showing a row and column value as well as the total for the cell.

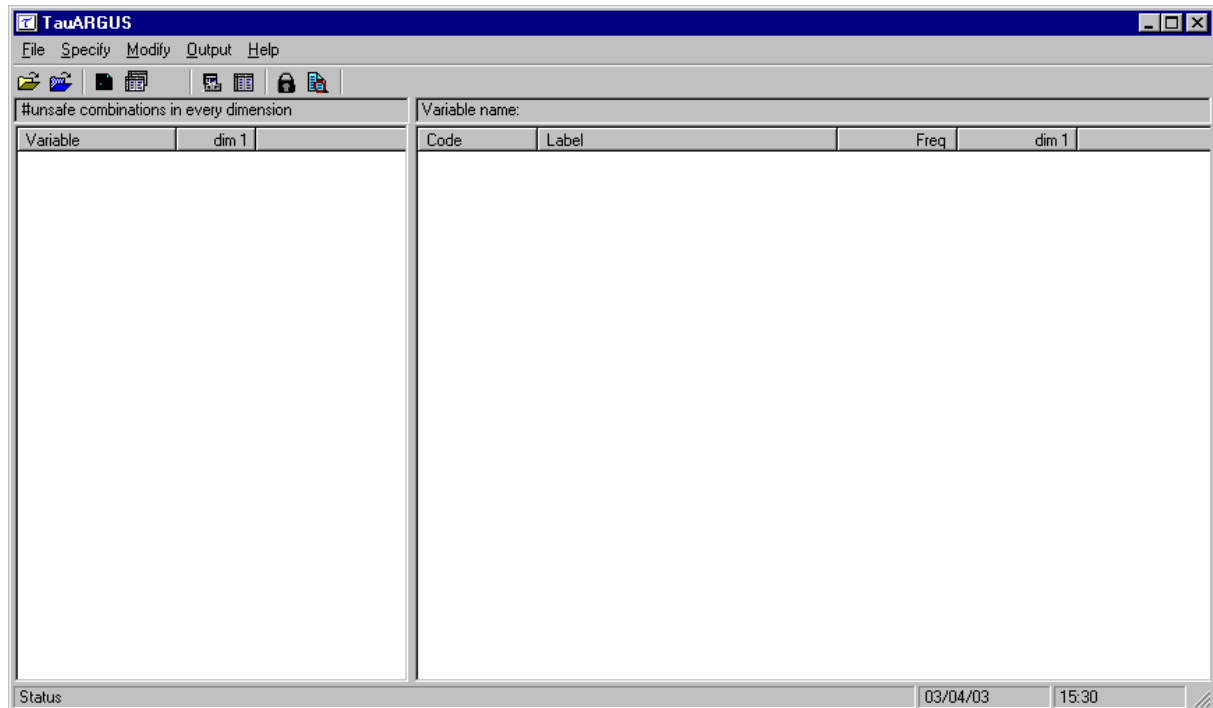


And of course an HTML report file is written to a user specified directory.

4. Description of the Menu Items

In this section we will give a description of the program by menu-item. The information in this section is the same as the information shown when the help-facility of τ -ARGUS is invoked.

4.1 Main Window



There are five menu headings. Under 'File' either a microdata file or tabular data file can be opened. 'Specify' allows the metadata to be entered or edited as well as letting the user specify the tables of particular interest along with primary suppressions. Under 'Modify' this table can be viewed and any secondary suppressions carried out. 'Output' allows the suppressed table to be saved and finally there is a 'Help' option.

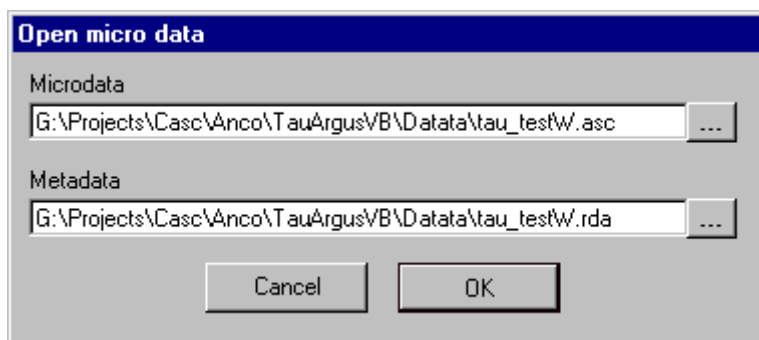
Overview of the menu-items

<u>File</u>	<u>Specify</u>	<u>Modify</u>	<u>Output</u>	<u>Help</u>
<u>Open Microdata</u>	<u>Metafile</u>	<u>Select Table</u>	<u>Save table</u>	<u>Contents</u>
<u>Open Table</u>	<u>Specify Tables</u>	<u>ViewTable</u>	<u>View Report</u>	<u>Options</u>
<u>Exit</u>	<u>Table Metadata</u>	<u>Linked Tables</u>		<u>About</u>


4.2 The File menu

4.2.1 File|Open Microdata

The File|Open microdata menu allows you to specify the Microdata file and the meta data file



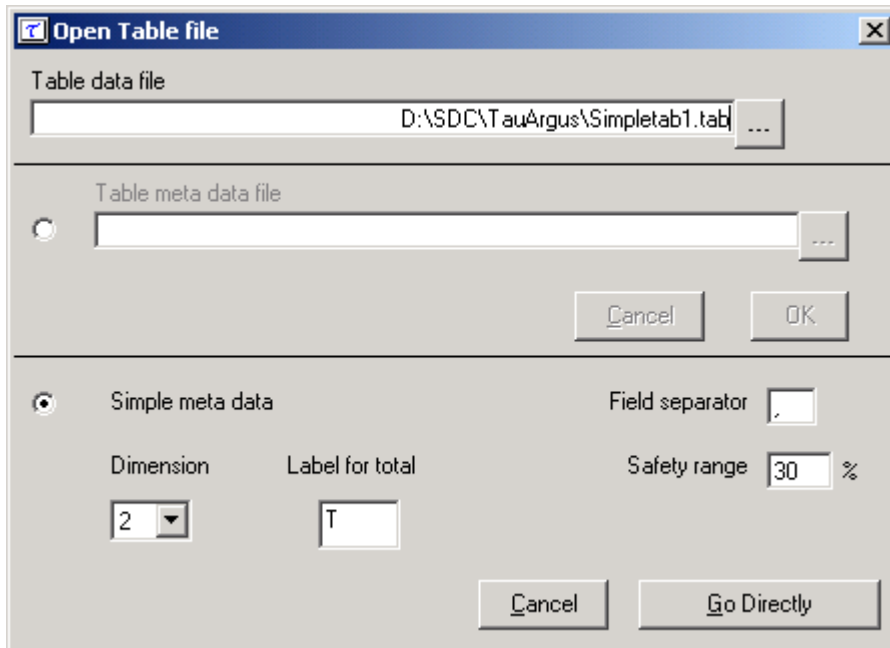
In this dialog box you can select the microdata-file and the corresponding metafile.

- By default the microdata-file has extension .asc and the metafile .rda.
- When you click on  you get an open file dialog box. In this box you can search for the files you want to use. You can choose other file-types when you click on the file-types listbox. When you have selected the microdata-file a suggestion for the metafile (with the same name but with the extension .rda) is given but only when this file exists.
- Before you click **OK** you must have filled in the name of the microdata-file.

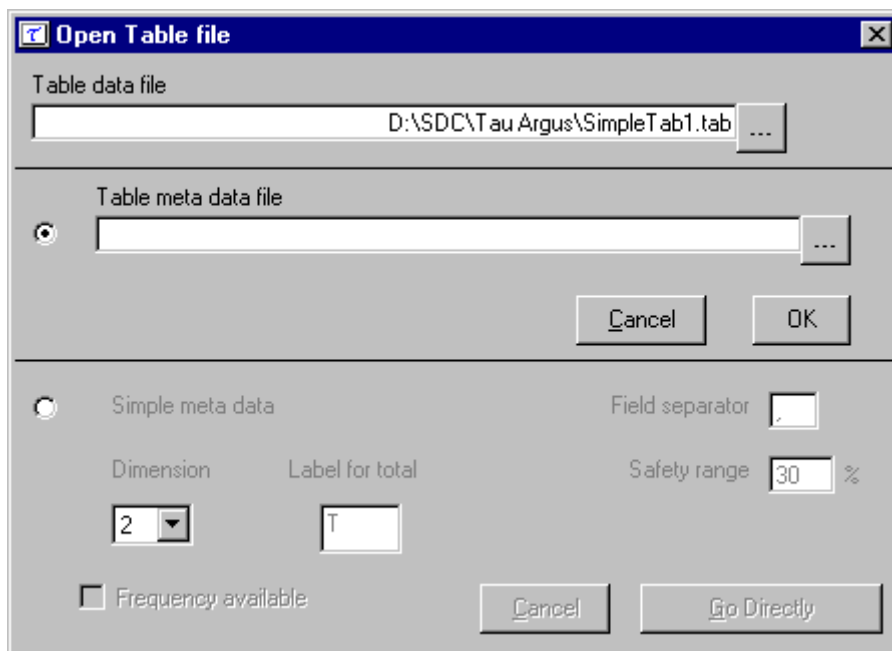
4.2.2 File|Open Table

The File|Open table allows a previously written table to a file to be opened (see section 3.1.4 for an example of the file format).

The simplest approach is to generate the metafile when entering the table by applying simple meta data. Here the table dimension and the labels for the marginal totals can be entered along with the 'Field Separator' and 'Safety range'. See section 3.1.4 for more details.



The default extension for a table is .tab and if a metadata file is to be imported the default extension is .rda. Other extensions can be used if required. If no table metadata file is specified however more detailed metadata can be created. (See sections 4.3.1 and 4.3.3).



4.2.3 File|Exit

Terminates the τ -ARGUS-session.

4.3 The Specify menu

4.3.1 Specify|Metafile [For Microdata]

Clicking on 'Specify|metafile' gives the user the opportunity to either edit the metafile already read

in or to enter the metafile information directly at the terminal. This option is only used when the data have been entered in the form of microdata. If an already calculated table has been entered modification to the metafile are made through 'Specify|table' metafile'.

In this dialog box all attributes of the variables can be specified. In section 3.1.2 we already have explained the layout of this file.

If under File|Open Microdata a .rda file has been specified, this dialog box shows the contents of this file. If no .rda file has been specified the information can be specified in this dialog box after pushing the New button. As default "newvar" is substituted. Apart from defining a new variable, an existing one can be modified or deleted.

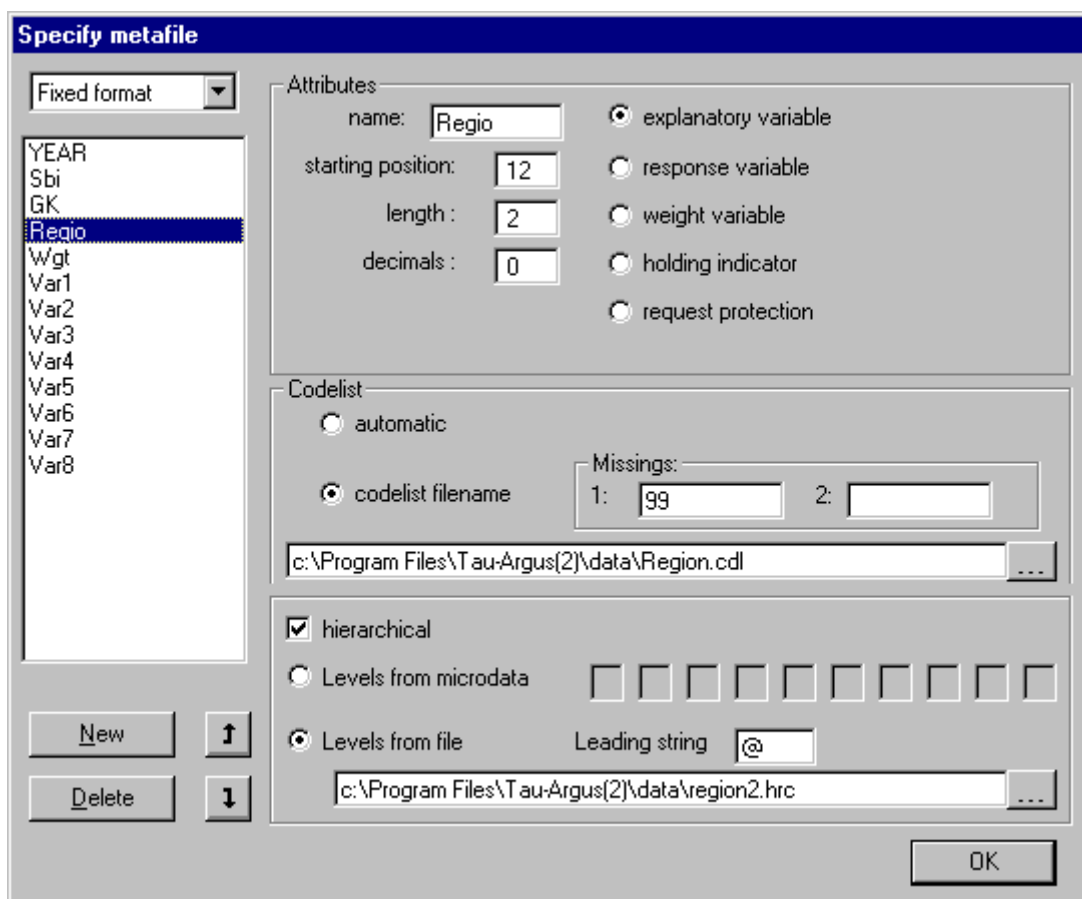
The following attributes can be specified:

- name of the variable
- its first position in the data file
- its length and the number of decimals.

Furthermore the kind of variable can be specified: explanatory variable, response variable or weight variable. An explanatory variable can be used as a spanning variable in the row or the column of the table, a response (numerical) variable can be used as cell-item. A weight variable specifies the weight of the record and is based on the (post)sampling design used.

The Holding indicator allows several contributions to one cell coming from one holding to be counted together as one contribution in both the safety rules and in the marginals. As an example a company in the totals and marginals will be considered as one enterprise/record although you might have records for each individual branch of the company. A restriction is that the data set is sorted, at least all records for one holding should be together.

The Request protection option is used if the Request Rule under 'Specify tables' is to be applied to this particular variable. See section 3.1.3.



Codelist

As this version of τ -ARGUS is the first version that can handle hierarchical codelists, this information should be made known to τ -ARGUS.

If 'automatic' option is selected in the non hierarchical case τ -ARGUS will always explore the datafile itself and make the codelist. Alternatively the user can specify a codelist, but that is only a list containing the labels attached to the codes. These labels are only used to enhance the information by τ -ARGUS on the screen, but τ -ARGUS will work only with the codes, that it has found when it explored the datafile

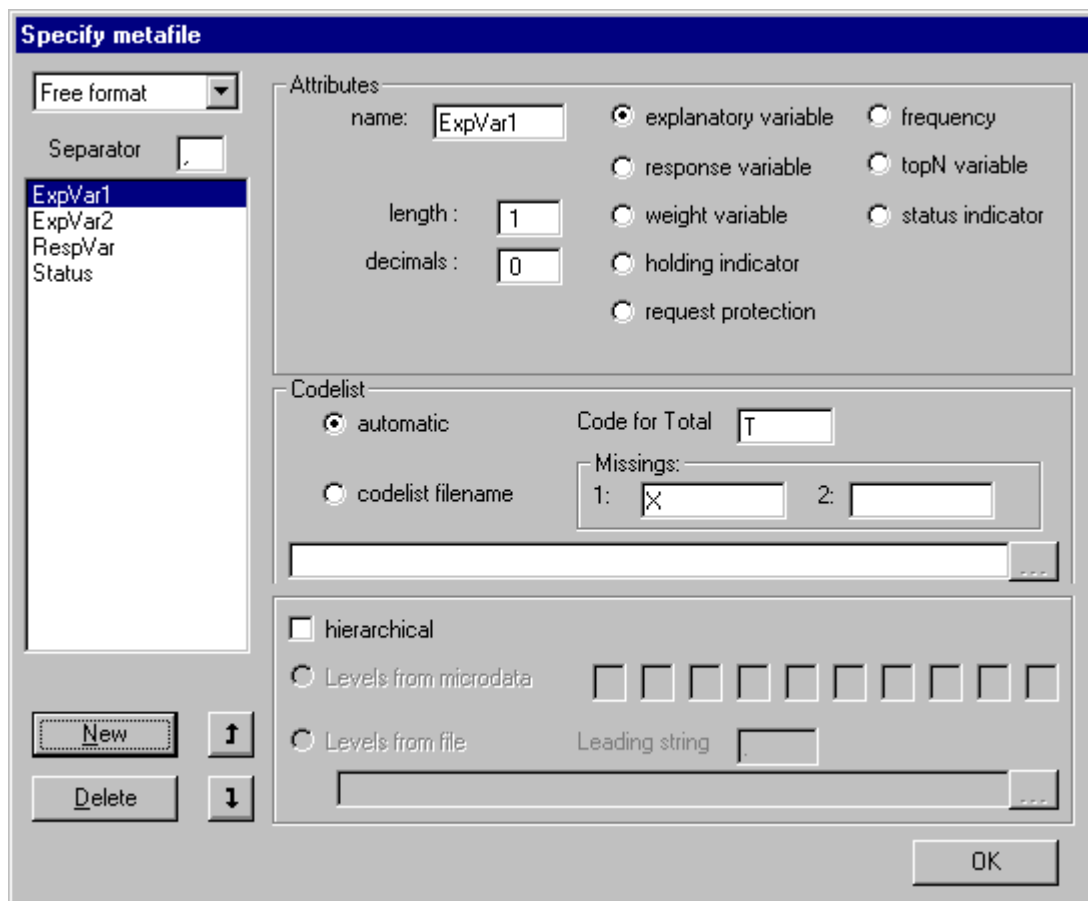
In the hierarchical case there are two options.

1. The hierarchy is derived from the digits in the codes. Each digit of the code denotes a level in the hierarchy. This is the traditional system for common code lists like industry code (NACE) etc. Additionally in τ -ARGUS you can specify that sometimes two or more digits together denote one level.
2. If the coding system used does not contain itself the information about the hierarchy, this information should be supplied to τ -ARGUS in an extra file (default extension .hrc). The layout of this file has been explained already in section 3.1.2. In this file the tree structure of the hierarchy is stored. A special character is used to indicate the depth in the hierarchy of a code. In this example an "@" has been chosen.

Additionally for each code the missing values (at least one) should be specified. In many surveys two missing values are used; e.g. one for *don't know* and one for *refusal*.

Specify|Metafile [For Tabular data]

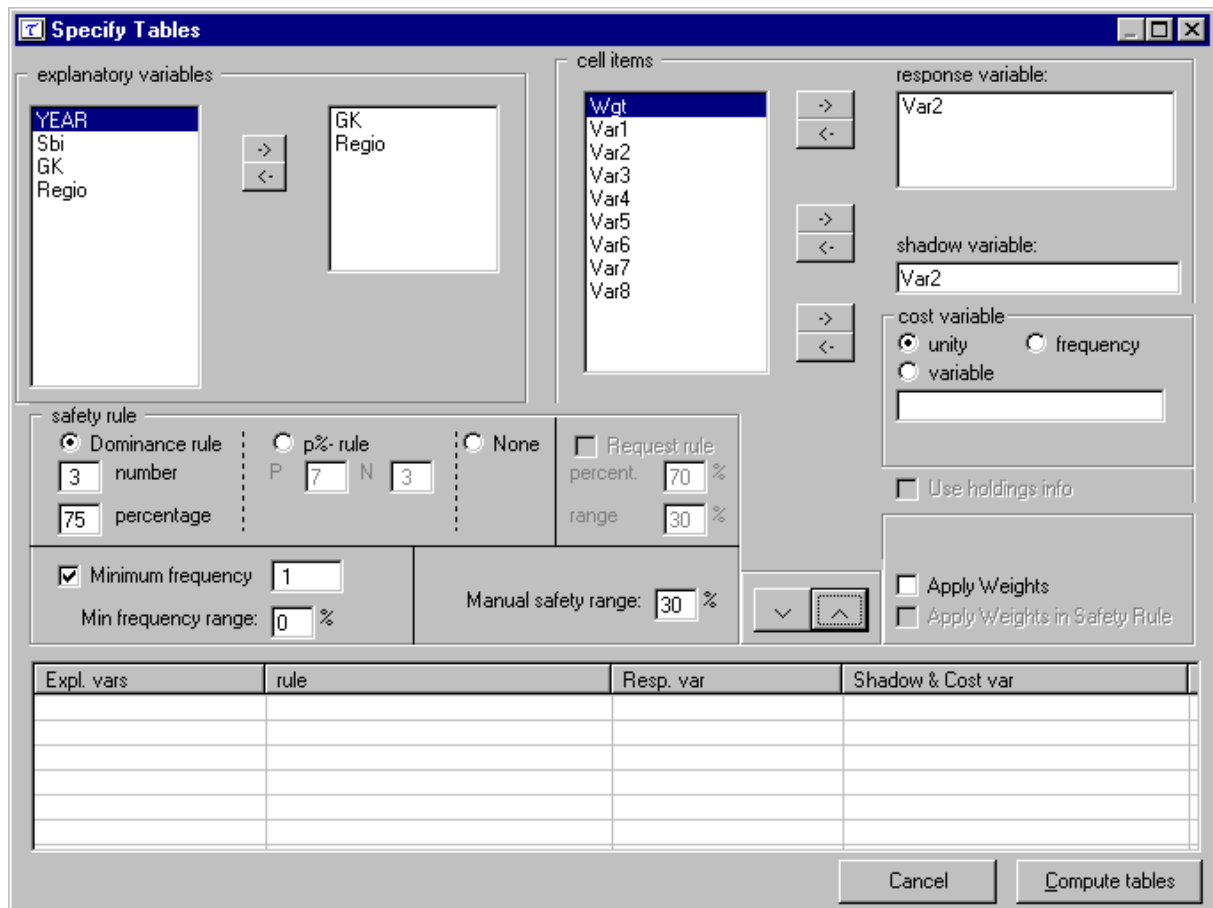
The window here is similar to that for microdata with a few changes. Noticeably the Status Indicator can be altered along with the code for the total. This window is not operational if the simple metadata option is chosen.



4.3.2 Specify|Specify Tables

In this dialog box you can specify the tables you want to protect. In one run of τ -ARGUS more than one table can be specified, but the tables will be protected separately.

Also you have to specify the parameters for the dominance rule and the minimum of records in a cell. At this moment τ -ARGUS only allows for up to 4-dimensional tables, but due to the capacities of the LP-solver used (Xpress or CPLEX) and the complexity of the optimisations involved 4-dim tables can only be protected by the hypercube method, see section 2.7



The explanatory variables

On the left is the listbox with the explanatory variables

When you click on '>' or '<' you transport the selected variable to the next box. From the left box with explanatory variables you can select the variables that will be used in the row or the column of the table.

The response variable

From the list of response variables you can select a variable as response variable (the cell-item). This is the variable for which the table to be protected is calculated.

The shadow variable

The shadow variable is the variable that is used to apply the safety rule. By default this is the response variable, but it is possible to select another variable. The safety rules are built on the principle of the characteristics of the largest contributors to a cell. If a variable other than the response variable is a better indicator for the size of a company, this variable can be used here.

The cost variable

This variable describes the cost of each cell. These costs are minimised when the secondary suppressed cells are calculated. By default this is the response variable but another choice is possible. It is also possible to use the frequency of the cells as a cost-function, this will minimise the number of records contributing to the cells to be suppressed. A third option is that the number of cells to be suppressed in minimised, irrespective of the size of their contributions.

Weight

If the data file has a sample weight, specified in the meta data, the table can be computed taking this weights into account. For this purpose the dominance rule has been extended.

The safety rule

The concept of safety rules is explained in section 2.1. On the left side of the window the type of rule can be selected and the value of the parameters. Additionally the minimum number of contributors can be chosen.

For the dominance-rule and the p %-rule, we compute symmetric safety ranges automatically.

As a rule the minimum frequency range should be a small positive value. A manual safety range is also required for cells that have been made unsafe by intervention of the users.

So in these two cases the user must provide a safety-range percentage.

When you have filled in everything you click ‘v’ to transport all the specified parameters to the ‘listwindow’ on the bottom. You can specify as many tables as you want, but as the size of the memory of a computer is still restricted you should not overdo it. If you want to modify a already made table you press the ‘^’ button.

Pressing the ‘Compute tables’ button will invoke τ -ARGUS to actually compute the tables requested and you are ready to start the process of disclosure control. τ -ARGUS will come back with the main window showing you the number of unsafe cells per variable per dimension.

4.3.3 Specify|TableMetadata

This option is specifically to apply safety rules to the table prior to primary suppression. As secondary suppression is an option The Manual Safety Range has to be set. This is equivalent to ‘Manual Safety Range’ in the ‘Specify tables’ window.

4.4 The Modify menu

4.4.1 Modify|Select Table

In this dialog box you can select the table you want to see. If you have specified only one table, this table will be selected automatically and this option cannot be accessed.

4.4.2 Modify|View Table

This window shows the table you have selected with Modify|View Table. On the left side the table itself is shown in a spreadsheet view. Safe cells are black, unsafe cells are red, secondary suppressed cells are blue and empty cells have a hyphen (-). The two check-boxes on the left-bottom give you some control over the layout. In the example shown here the complete table cannot be seen on the screen. The cursor at the bottom of the table can be used to display the remaining columns.

- The 3-digit separator will show the cell-values with this separator (See ‘Options at the bottom of the table’ for more details’. The separator is chosen according to the general Windows settings.
- The Output view shows the table, with all the suppressed cells replaced by an ‘X’; this is how the safe table will be published.

When you click on a cell in the main body of the table, information about this cell is visible in **Cell Information pane**.

- You can see the following information:
1. The cell-value
 2. The cell status
 3. The number of contributors to a cell
 4. The largest contributors.

Status is the status of the cell, this can be:

- Safe: Does not violate the safety rule

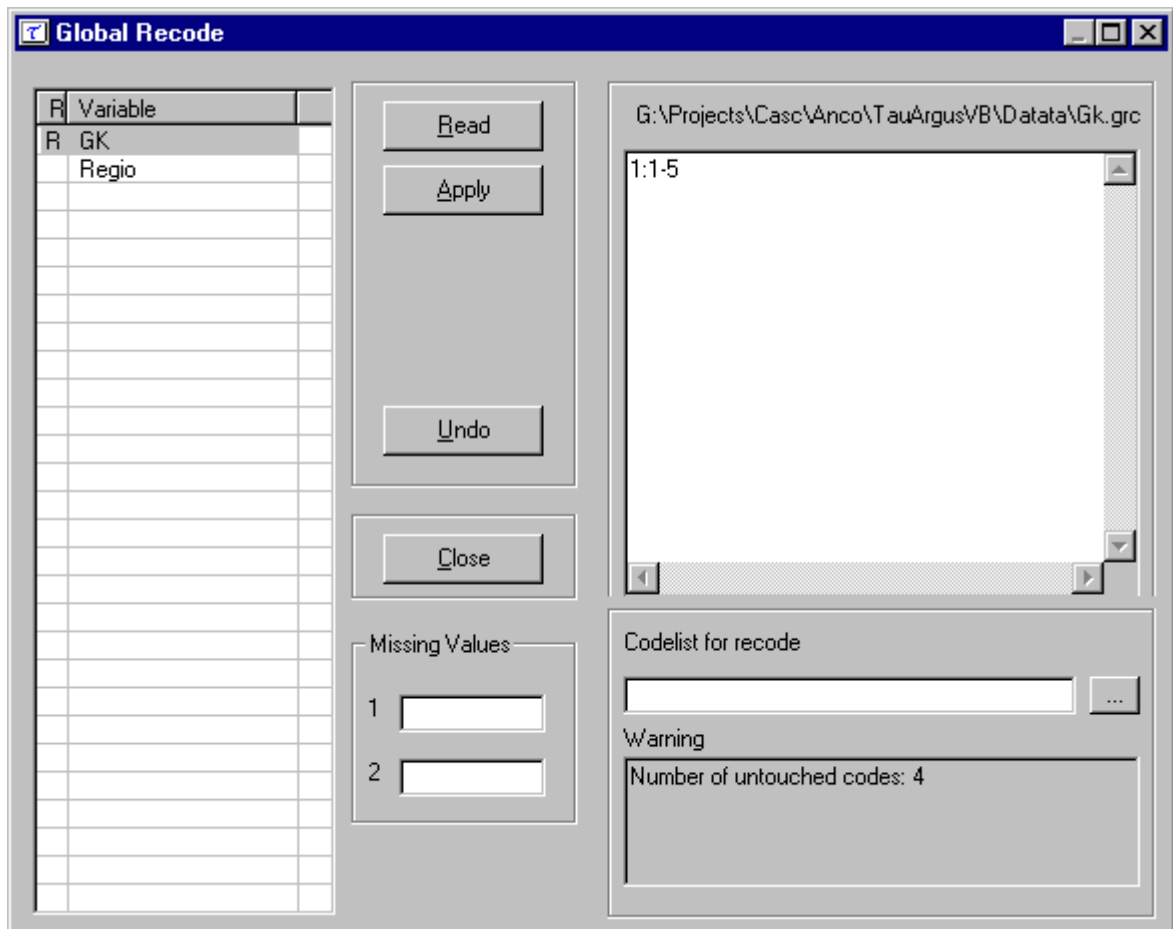
- Safe (from manual): manually made safe during this session
- Unsafe: According to the safety rule
- Unsafe (from manual): manually made unsafe during this session
- Suppressed: Made unsafe by the secondary cell suppression
- Protected: Cannot be selected as a candidate for secondary cell suppression
- Zero: Value is zero and cannot be suppressed.
- Empty: No records contributed to this cell and the cell cannot be suppressed.

The second pane ('Change Status') on the right will allow you to change the cell-status. Only a few (logical) transitions are allowed. Changing the cell status can be useful if an unsafe cell may be published because of external reasons (permissions to publish). Changing to protect will prevent the system to select this cell as a secondary suppression.

The recode button will bring you to the recoding system. Recoding is a very powerful method of protecting a table. Collapsed cells tend to have more contributors and therefore tend to be much safer.

4.4.2.1 Recoding a non-hierarchical variable

There is a big difference in recoding a hierarchical variable compared to a non-hierarchical variable.



In the non-hierarchical case you can specify a global recoding manually. Either you enter the recoding described below manually or you read it from a file. The default extension for this file is .GRC.

There are some rules how you have to specify a recode scheme. All codelists are treated as alphanumeric codes. This means that codelists are not restricted to numerical codes only. However this also implies that the codes '01' and '1' are considered different codes and also 'aaa' and 'AAA' are different. In a recoding scheme you can specify individual codes separated by a comma (,) or ranges of

codes separated by a hyphen (-). The range is determined by treating the codes as strings and using the standard string comparison. E.g. `0111` < `11` as the `0` precedes the `1` and `ZZ` < `a` as the uppercase `Z` precedes the lowercase `a`. Special attention should be paid when a range is given without a left or right value. This means every code less or greater than the given code. In the first example the new category 1 will contain all the codes less than or equal to 49 and code 4 will contain everything larger than or equal to 150.

Example:

for a variable with the categories 1,...,182 a possible recode is then:

- 1: - 49
- 2: 50 - 99
- 3: 100 - 149
- 4: 150 –

for a variable with the categories 01 till 10 a possible recode is:

- 1: 01 , 02
- 2: 03 , 04
- 3: 05 - 07
- 4: 08 , 09 , 10

Don't forget the colon (:) if you forget it the recode will not work.

The recode 3: 05-07 is the same as 3: 05,06,07 you can choose what you like best.

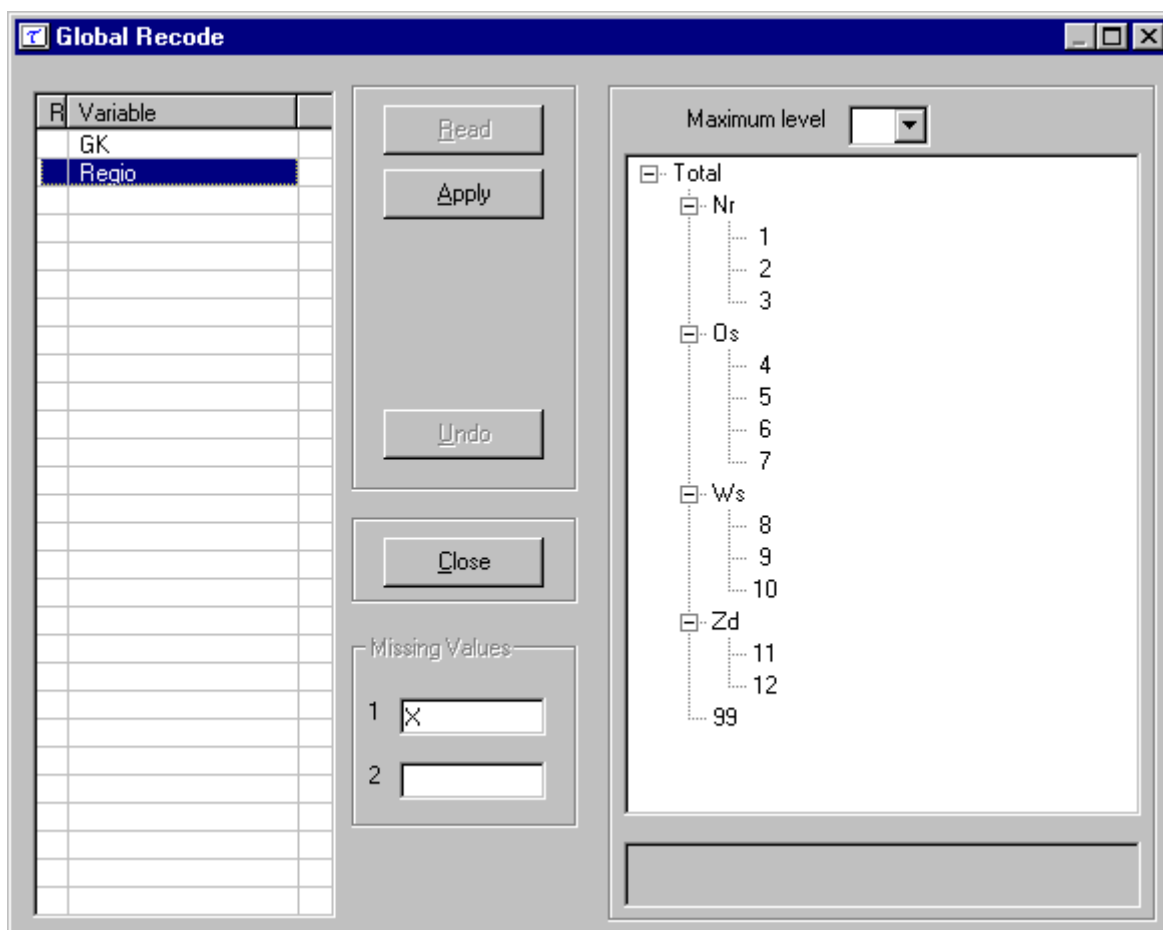
Additionally you can change the coding for the missing values by entering these codes in the relevant textboxes. And you can specify the name of a new codelist with the labels for the new coding scheme.

Pressing the 'Apply' button will actually restructure your table. And if required you can always undo a recoding.

If you apply a recoding τ -ARGUS will present you with the results. This can be that certain codes could not be found or that you did not follow the above described syntax. In that case an error message will be shown. Alternatively a warning could be issued; e.g. if you did not recode all original codes, τ -ARGUS will inform you. But it can be your purpose and there is no objection to it. In the example above τ -ARGUS informs you that 4 codes have not been changed.

4.4.2.2 Recoding a hierarchical variable

In the hierarchical case the code scheme is typically a tree. To global recode a hierarchical variable means that you manipulate a tree. The standard Windows tree view is used to present a hierarchical code.



You can fold and unfold certain parts of a tree with the standard Windows actions (clicking on '+' and '-').

The combo box at the top of the screen offers the opportunity to fold and unfold the tree to a certain level.

Additionally you can change the coding for the missing values by entering these codes in the relevant textboxes.

Pressing the 'Apply' button will actually restructure your table. if required you can always undo a recoding.

The Options at the Bottom of the table

Change View

When you click on Change View in the Table window after clicking on Modify|View Table at an earlier stage this dialog box pops up. You can specify which variable you want in the row and the column. In the two-dimensional case you can only transpose the table. In the higher dimensional case the remaining variables will be in the layer. For these layer variables a combo-box will appear at the top of the table, where you can select a code. This will show the corresponding slice of the table.

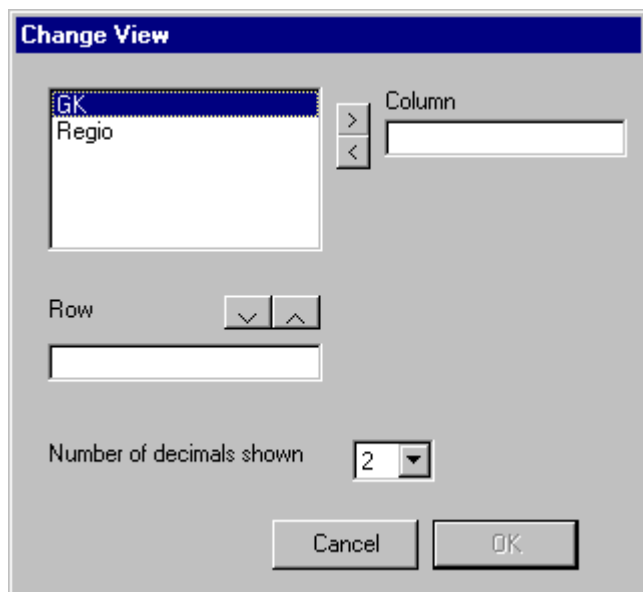
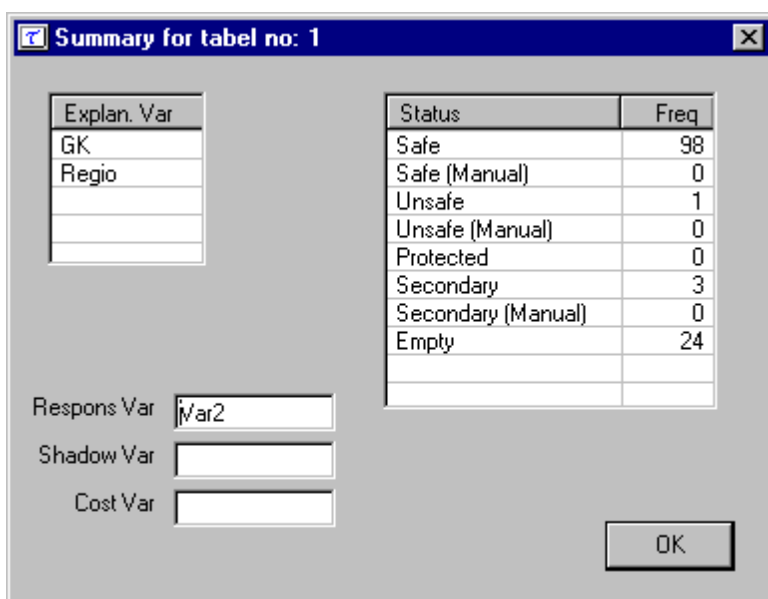


Table summary

The table summary will give an overview of the number of cells according to their status.



3 dig separator

This removes or inserts the comma separating the thousands for the values in the table.

Output View

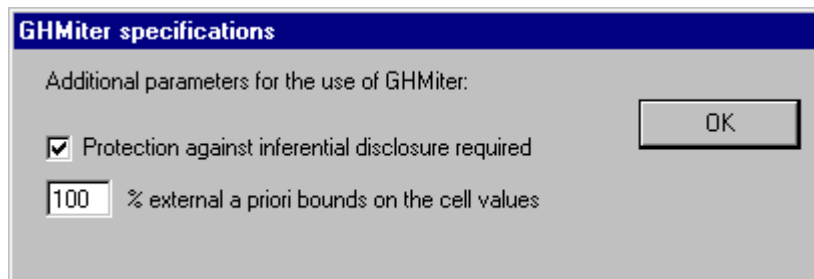
This options allows the table to be shown as it will be output, with suppressed cells (primary and secondary) replaced by a X.

Secondary suppressions

We now discuss the actions in the suppress pane in the table window (after selecting 'modify|table'.

With suppress you can protect your table by adding additional cells to be suppressed. This is necessary to make a safe table. In this version of τ -ARGUS you can choose between the hypercube method and the optimal solutions. In this version we have implemented both the full optimal and modular (partial) search-algorithms for hierarchical tables, (which use the HITAS-approach (See section 2.9)). This partial method will break the hierarchical table down to several non-hierarchical tables, protect them and compose a protected table from the smaller pieces. As this method uses the optimisation routines, an LP-solver is required. Either Xpress or CPLEX is required. It is the responsibility of the users of τ -ARGUS to apply for a licence of one of these commercial packages themselves. Information on obtaining one of these licences will be found in a 'read.me' file that will be supplied with the software.

Just select one of the options and press the 'Suppress'-button. τ -ARGUS will start working for you and finally it will show you a protected table. The secondary suppressed cells will be shown in blue. If you had selected the hypercube method τ -ARGUS will ask you to complete the following information.



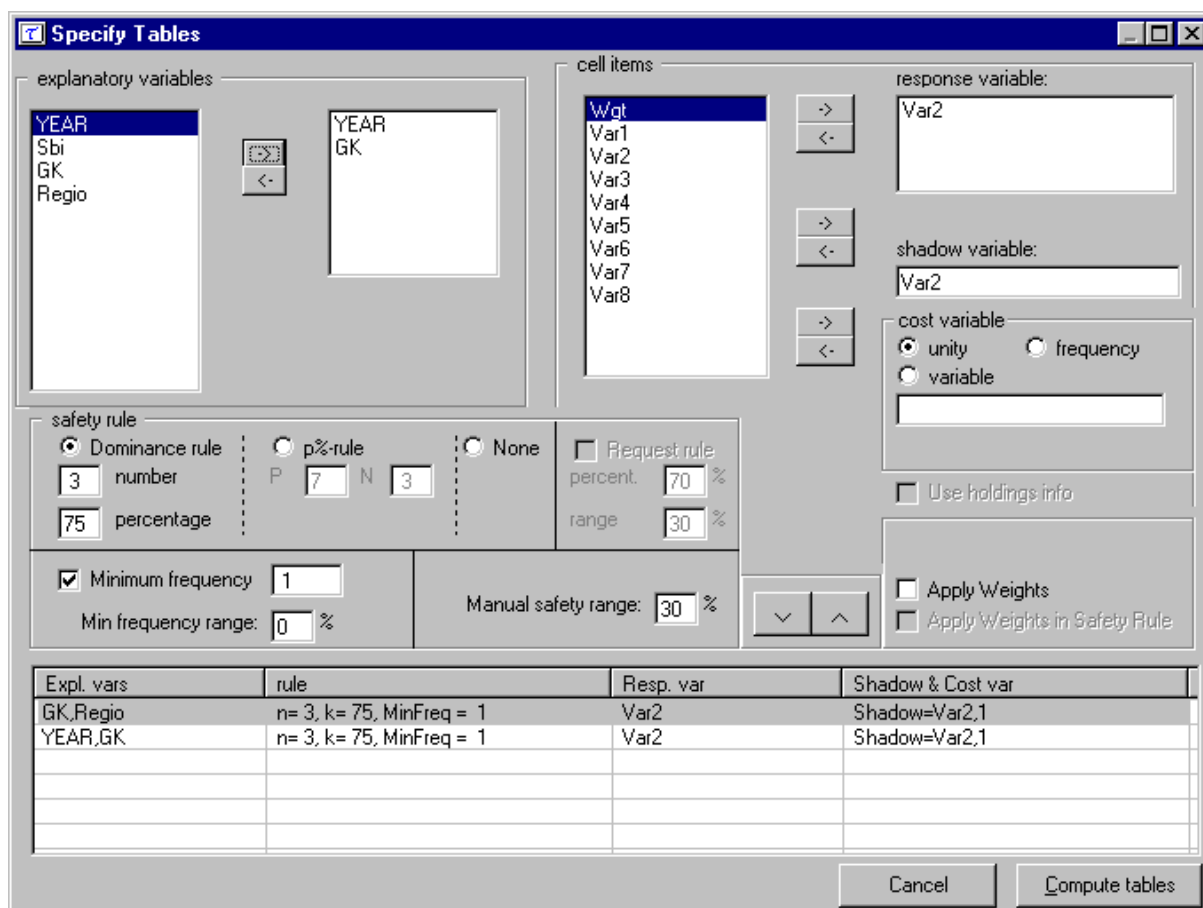
.. The ratio parameter is chosen by the software. More information on the hypercube method can be found in section 2.7

When you are satisfied with the table you can store the table. Press the write-table button. This is virtually the same button as via the menu Output|Save table (section 4.5.1)

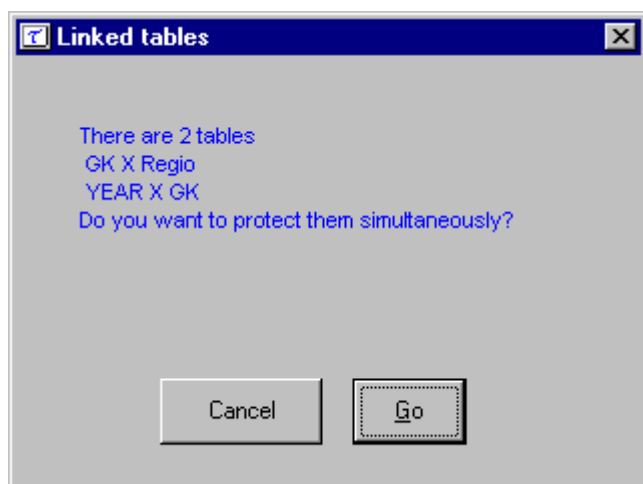
4.4.3 Linked Tables

This option is available when the tables specified have at least one spanning variable in common and the same response variable.

An example is shown.



After the tables have been computed, under the 'Modify Tables' option, 'Linked Tables' is available. The following table appears. Clicking on 'Go' will protect the tables simultaneously.



This procedure is only available when microdata is read into the program. Currently when tabular data are entered only one table can be entered thus making the possibility of linked tables impossible.

4.5 The Output menu

4.5.1 Output|Save Table

Basically you have three options of storing the tables

1. As a CSV file. This Comma separated file can easily be read into Excel. Often it is better to click on this file in the windows explorer, than reading it into Excel by using the File|open

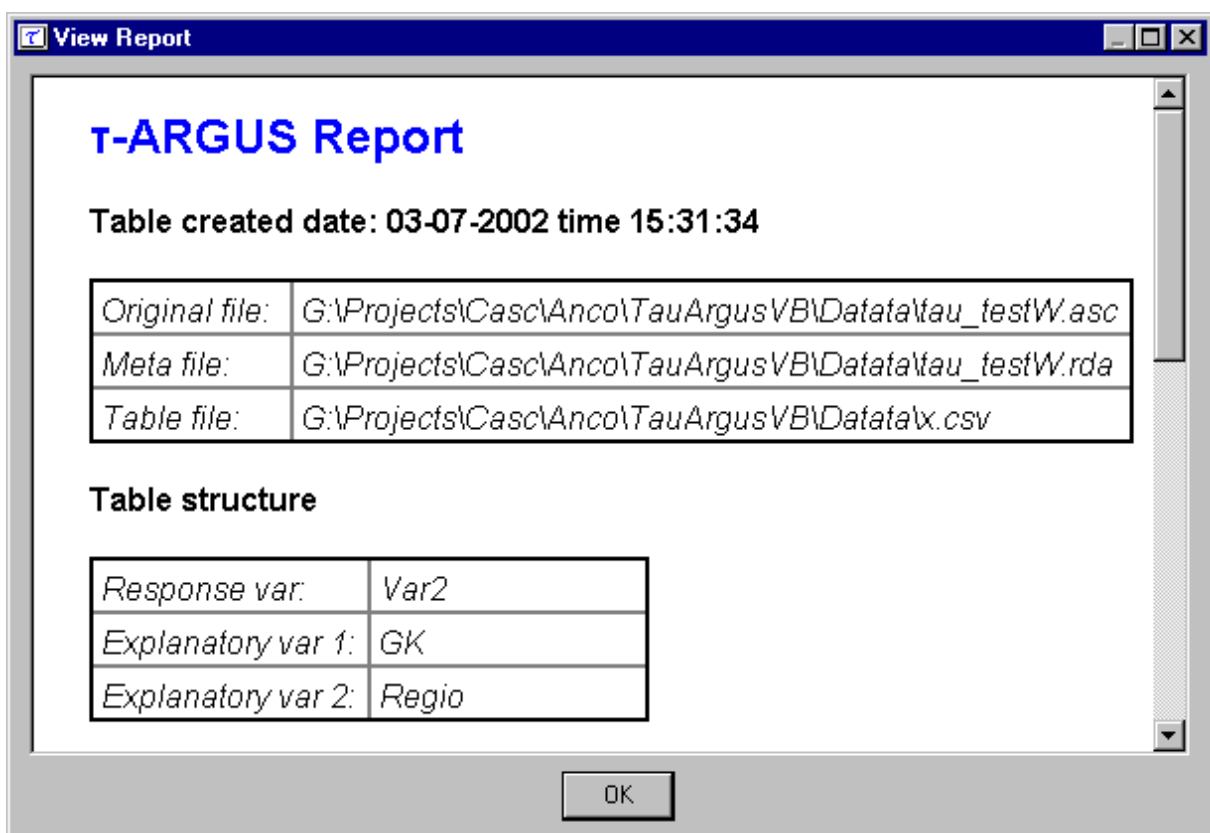
option of Excel.

2. A CSV-file for a pivot table. This offers you the opportunity to make use of the facilities of pivot table in Excel. The status of each cell can be added here as an option.
3. A file in the format code value separated by commas. Here the cell status is again an option. Also empty cells can be suppressed from the output file if required.

Finally a report will be generated to a directory specified by the user. This report will be shown, when the table has been written. As this is an HTML-file it can be viewed easily later.

4.5.2 Output|View Report

Views the report file which has been generated with Output|Save Table

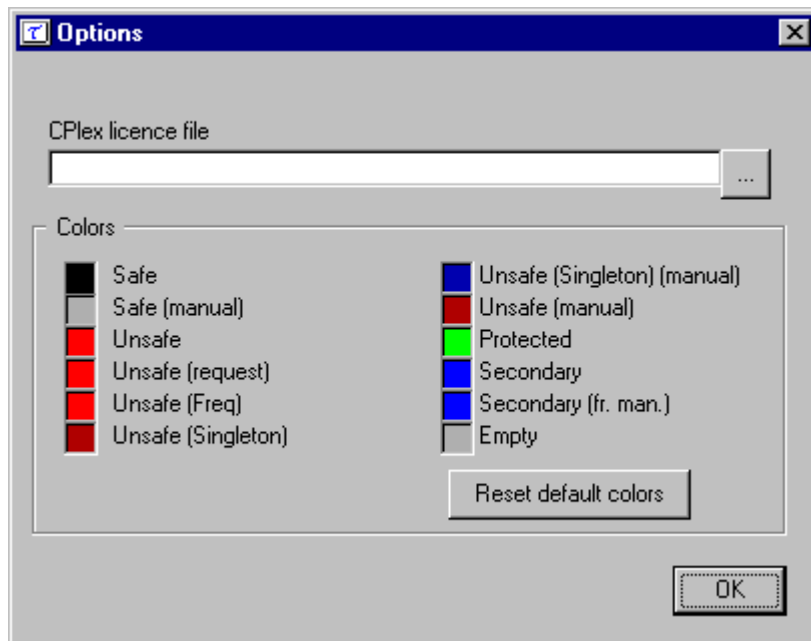


4.6 The Help menu

4.6.1 Help|Contents

Shows the contents page of the help file.
This program has context-sensitive help.

4.6.2 Help|Options



There are a number of options which can be changed here. Firstly if the CPLEX optimisation routine is being used, the location of the licence file can be specified here. Also the default colours for the differently specified cells can be altered.

4.6.3 Help|About

Shows the about box.