μ ARGUS

version 3.2

User's Manual

Document: 2-D4
Project: CASC-project
Date: February 2003
BPA no: 768-02-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
Luisa Franconi and Alessandra Capobianchi (Individual risk model)
Peter-Paul de Wolf (PRAM)
Josep Domingo and Vicenc Torra (Numerical micro aggregation and rank swapping)
Ruth Brand and Sarah Giessing (Sullivan Masking)

Contents

Preface

This is the users manual of version 3.2 of μ–ARGUS, a software package to assist in producing safe microdata. The package has been developed under Windows NT and runs under Windows versions from Windows 95. This is the first version of μ-ARGUS that has been produced in the fifth Framework CASC (Computational Aspects of Statistical Confidentiality) project. The current version builds further on the μ-ARGUS version that has been produced in the fourth framework SDC project. The purpose of the present manual is to give a potential user enough information so that he can understand the general principles on which μ–ARGUS is based, and also allow him to apply the package. So it contains both general background information and detailed program information.

μ–ARGUS is one part of a twin, the other one named τ–ARGUS. τ–ARGUS is a tool to help produce safe tables. Although the twins look quite different, from the inside they have a lot in common. In a sense they are Siamese twins, since their bodies have various organs in common. Nevertheless, a description of τ–ARGUS cannot be found in the present manual, but in a separate one.[1]

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 Lairesse (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 looser as the mythological Argus is.

Contact

Feedback from users will help improve future versions of μ–ARGUS and is therefore greatly appreciated. Suggestions for improvements can be sent to argus@cbs.nl.

Acknowledgements

μ–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 the μ–ARGUS software 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. New developments included in this version are the risk-approach, coordinated by Luisa Franconi cs from Istat and the Post Randomisation method

[1] Anco Hundepool et al 2003, *τ*–ARGUS user manual Version 2.2, Department of Statistical Methods, 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 built by Wil de Jong.

[3] The original copy of this engraving is in the collection of ‘Het Leidsch Prentenkabinet’ in Leiden, The Netherlands.

(PRAM), based on work by Peter Kooiman, Peter-Paul de Wolf and Ardo van den Hout at CBS. Further developments include work on Numerical Micro Aggregation and Rank Swapping by Josep Domingo and Vicenc Torra, and on Sullivan Masking by Ruth Brand and Sarah Giessing.

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 Nationale 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'Estadistica de Catalunya	IDESCAT	ES
10. Institut National de Estadística	INE	ES
11. TU Ilmenau	TUIlm	D
12. Institut d'Investigació en Intelligència Artificial-CSIC	CIS	ES
13. Universitat Rovira i Virgili	URV	ES
14. Universitat Politècnica de Catalunya	UPC	ES

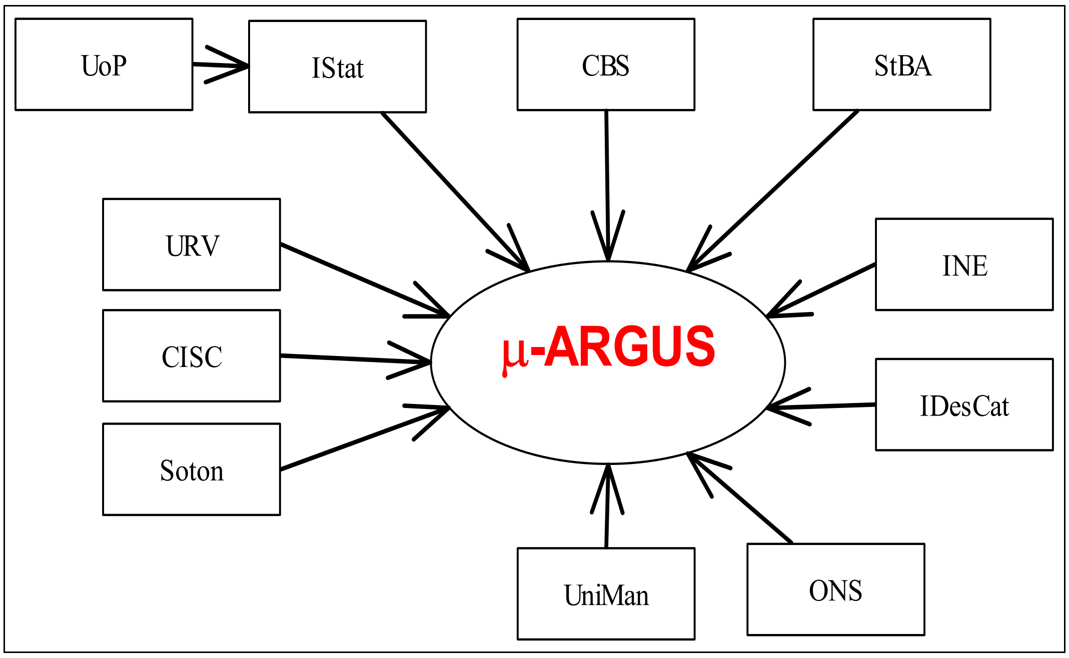
Although Statistics Netherlands is the main contractor, the management of this project is a joint responsibility of the steering committee. This steering committee constitutes of 5 partners, representing the 5 countries involved and also bearing a responsibility for a specific part of the CASC-project:

CASC Steering Committee

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

The CASC-microdata team

Several partners of the CASC-team work together and contribute in various stages to the development of μ-ARGUS. These contributions are either the actual software development, the research underlying this software development or the testing of the software.



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 a statistical office with the required information if they can be certain that their data will be used with the greatest care, and in particular will not jeopardise their privacy. So statistical use of the data by outside users should not lead to a compromise of confidentiality. However, making sure that microdata cannot be misused for disclosure purposes, requires, generally speaking, that they should be less detailed, or modified in another way that hampers the disclosure risk. This is in direct conflict with the wish of researchers to have as detailed data as possible. Much detail allows not only more detailed statistical questions to be answered, but also more flexibility: the user can lump together categories in a way that suits his purposes best. The field of statistical disclosure control in fact feeds on this trade-off: How should a microdata set be modified in such a way that another one is obtained with acceptable disclosure risk, and with minimum information loss? And how exactly can one define disclosure risk? And how should one quantify information loss? And once these problems have been solved - no matter how provisionary - the question is how all this wisdom can actually be applied in case of real microdata. If a certain degree of sophistication is reached the conclusion is inescapable: specialised software is needed to cope with this problem and μ–ARGUS is such software.

In order to be able to use μ–ARGUS one should understand the ideas about disclosure risk, information loss, etc. on which it is based. Therefore the section 2 is devoted to giving the necessary background information. In later sections more detailed information about the working and use of μ–ARGUS can be found.

1.1 System Requirements

In general terms, the user requires a reasonably powerful Windows PC (Windows 95 or later). The RAM available will influence the performance, as with all applications. After installation, 32Mb is a minimum requirement for running the program. Any libraries required (except those for Sullivan Masking) will be installed automatically on installation, provided, of course, that the installer has appropriate installation rights, e.g. administrator rights on a Win2000 PC.

2. Producing safe microdata

The purpose of the present chapter is to give the necessary background information for the use of μ–ARGUS. Producing safe micro data is not a trivial issue. It should first be explained when microdata are considered safe or unsafe. It should also be explained how unsafe data can be modified to become safe.

2.1 Safe and unsafe microdata

μ–ARGUS is based on a view of safety/unsafety of microdata that is used at Statistics Netherlands. In fact the incentive to build a package like μ–ARGUS was to allow data protectors at Statistics Netherlands to apply the general rules for various types of microdata easily, and to relieve them from the chore and tedium that producing a safe file in practice can involve. Not only should it be easy to produce safe microdata, it should also be possible to generate a logfile that documents the modifications of a microdata file.

When implementing μ–ARGUS it was the objective to produce a package that, on the one hand, is able to handle the specific rules that Statistics Netherlands applies to produce safe data, and on the other hand is more general. Nevertheless the generality has a bound. Despite unavoidable limitations, we believe that μ–ARGUS can justly be called a general-purpose package to help produce safe microdata. It is "general" within its philosophy, in the same sense as a Volkswagen Beetle is general in its capability of transporting a limited number of persons, but not 10 tons of bricks.

So we can say that the development of μ–ARGUS was heavily inspired by the type of rules that Statistics Netherlands uses to protect its microdata, rather than the precise form of the rules. So in order to be able to understand the general ideas that μ–ARGUS uses, it is necessary that the reader has a good understanding of the type of rules that it is supposed to support. A brief discussion of these is given below. A more extensive treatment can be found in Willenborg and De Waal (1996).[4]

The aim of statistical disclosure control is to limit the risk that sensitive information of individual respondents can be disclosed from data that are released to third party users. In case of a microdata set, i.e. a set of records containing information on individual respondents, such a disclosure of sensitive information of an individual respondent can occur after this respondent has been re-identified. That is, after it has been deduced which record corresponds to this particular individual. So, the aim of disclosure control should help to hamper re-identification of individual respondents represented in data to be published.

An important concept in the theory of re-identification is a key. A key is a combination of (potentially) identifying variables. An identifying variable, or an identifier, is one that may help an intruder re-identify an individual. Typically an identifying variable is one that describes a characteristic of a person that is observable, that is registered (identification numbers, etc.), or generally, that can be known to other persons. This, of course, is not very precise, and relies on one's personal judgement. But once a variable has been declared identifying, it is usually a fairly mechanical procedure to deal with it in μ–ARGUS.

In order to develop a theory of re-identification one should have a disclosure scenario.[5] Such a scenario is essentially a re-identification model, in which it is stated what sort of information an intruder is supposed to have, how he uses this information to re-identify individuals represented in a given microdata set, what sort of disclosures such a person is likely to make, and what motives he has for making such disclosures (he might want to know something about a particular individual, but his motive could also be to embarrass the data collector/releaser). More formally, such a disclosure scenario may lead to a disclosure risk model, i.e. a statistical model that gives a probability that disclosures are being made by an intruder, with an assumed level of knowledge of the population and for a given data file. Such disclosure risk models can range from a fairly simple, intuitive model to a

[4] Leon Willenborg and Ton de Waal (1996), Statistical Disclosure Control in Practice, Lecture Notes in Statistics Vol. 111, Springer-Verlag, New York. See in particular Section 4.2.9.

[5] This concept is developed and used in Gerhard Paass and Udo Wauschkuhn, 1985, Datenzugang, Datenschutz und Anonymisierung, R. Oldenbourg Verlag, Munich.

rather sophisticated one. Of the latter type are models that either yield estimations of the number of unsafe combinations, or give for each record in the file a probability that an intruder will disclose it.

In a disclosure scenario, keys (as described above) are supposed to be used by an intruder to re-identify a respondent. Re-identification of a respondent can occur when this respondent is rare in the population with respect to a certain key value, i.e. a combination of values of identifying variables. Hence, rarity of respondents in the population with respect to certain key values should be avoided. When a respondent appears to be rare in the population with respect to a key value, then disclosure control measures should be taken to protect this respondent against re-identification. In practice, however, it is not a good disclosure control strategy to attempt to prevent only the occurrence of respondents in the data file who are rare in the population (with respect to a certain key). For this several reasons can be given. Firstly, there is a practical reason: rarity in the population, in contrast to rarity in the data file, is hard to establish. There is generally no way to determine with certainty whether a person who is rare in the data file (with respect to a certain key) is also rare in the population. Secondly, an intruder may use another key than the key(s) considered by the data protector. For instance, the data protector may consider only keys consisting of at most three variables while the intruder may use a key consisting of four variables.

Therefore, it is better to avoid the occurrence of combinations of scores that are rare in the population instead of avoiding only population-uniques. To define what is meant by rare the data protector has to choose a threshold value $D_{k,}$ for each key value k, where the index k indicates that the threshold value may depend on the key k under consideration. A combination of scores, i.e. a key value, that occurs not more than D_k times in the population is considered unsafe, a key value that occurs more than D_k times in the population is considered safe. The unsafe combinations must be protected, while the safe ones may be published.

Re-identification of an individual can take place when several values of so-called identifying variables, such as ‘Place of residence’, ‘Sex’ and ‘Occupation’, are taken into consideration. The values of these identifying variables can be assumed to be known to relatives, friends, acquaintances and colleagues of a respondent. When several values of these identifying variables are combined a respondent may be re-identified. Consider for example the following record obtained from an unknown respondent:

‘Place of residence = Urk’, ‘Sex = Female’ and ‘Occupation = Statistician’.

Urk is a small fishing-village in the Netherlands, in which it is unlikely that many statisticians live, let alone female statisticians. So, when we find a statistician in Urk, a female statistician moreover, in the microdata set, then she is probably the only one. When this is indeed the case, anybody who happens to know this rare female statistician in Urk is able to disclose sensitive information from her record if such information is contained in this record.

There is a practical problem when applying the above rule that the occurrence (in the data file) of combinations of scores that are rare in the population should be avoided. Namely, it is usually not known how often a particular combination of scores occurs in the population. In many cases one only has the data file itself available to estimate the frequency of a combination of scores in the population. In practice one therefore uses the estimated frequency of a key value k to determine whether or not this key value is safe or not in the population. When the estimated frequency of a key value, i.e. a combination of scores, is at least equal to the threshold value D_k, then this combination is considered safe. When the estimated frequency of a key value is less than the threshold value D_k, then this combination is considered unsafe. An example of such a key is ‘Place of residence’, ’Sex’, ’Occupation’.

2.2 Statistical disclosure control measures

When a file is considered unsafe it has to be modified in order to produce a safe file. This modification can generally be done using so-called SDC techniques. Examples of such techniques are global recoding (grouping of categories), local suppression, PostRAndomisation Method (PRAM[6]), adding

[6] See P.-P. de Wolf, J.M. Gouweleeuw, P. Kooiman and L.C.R.J. Willenborg, Reflections on PRAM, Proceedings SDP98, Lisbon.

noise. The effect of applying such techniques to a microdata set is that its information content is decreased. Variables are specified in fewer, and less detailed categories; values in certain records are suppressed; or, values in certain records are replaced by other values.

2.2.1 Global recoding

In case of global recoding several categories of a variable are collapsed into a single one. In the above example we can recode the variable ‘Occupation’, by combining the categories ‘Statistician’ and ‘Mathematician’ into a single category ‘Statistician or Mathematician’. When the number of female statisticians in Urk plus the number of female mathematicians in Urk is sufficiently high, then the combination ‘Place of residence = Urk’, ‘Sex = Female’ and ‘Occupation = Statistician or Mathematician’ is considered safe for release. Note that instead of recoding ‘Occupation’ one could also recode ‘Place of residence’ for instance.

It is important to realise that global recoding is applied to the whole data set, not only to the unsafe part of the set. This is done to obtain a uniform categorisation of each variable. Suppose, for instance, that we recode the ‘Occupation’ in the above way. Suppose furthermore that both the combinations ‘Place of residence = Amsterdam’, ‘Sex = Female’ and ‘Occupation = Statistician’, and ‘Place of residence = Amsterdam’, ‘Sex = Female’ and ‘Occupation = Mathematician’ are considered safe. To obtain a uniform categorisation of ‘Occupation’ we would, however, not publish these combinations, but only the combination ‘Place of residence = Amsterdam’, ‘Sex = Female’ and ‘Occupation = Statistician or Mathematician’.

2.2.2 Local suppression

When local suppression is applied one or more values in an unsafe combination are suppressed, i.e. replaced by a missing value. For instance, in the above example we can protect the unsafe combination ‘Place of residence = Urk’, ‘Sex = Female’ and ‘Occupation = Statistician’ by suppressing the value of ‘Occupation’, assuming that the number of females in Urk is sufficiently high. The resulting combination is then given by ‘Place of residence = Urk’, ‘Sex = Female’ and ‘Occupation = missing’. Note that instead of suppressing the value of ‘Occupation’ one could also suppress the value of another variable of the unsafe combination. For instance, when the number of female statisticians in the Netherlands is sufficiently high then one could suppress the value of ‘Place of residence’ instead of the value of ‘Occupation’ in the above example to protect the unsafe combination. A local suppression is only applied to a particular value. When, for instance, the value of ‘Occupation’ is suppressed in a particular record, then this does not imply that the value of ‘Occupation’ has to be suppressed in another record. The freedom that one has in selecting the values that are to be suppressed allows one to minimise the number of local suppressions.

2.2.3 Top and bottom coding

Global recoding is a technique that can be applied to general categorical variables, i.e. without any requirement of the type. In case of ordinal categorical variables one can apply a particular global recoding technique namely top coding (for the larger values) or bottom coding (for the smaller values). When, for instance, top coding is applied to an ordinal variable, the top categories are lumped together to form a new category. Bottom coding is similar, except that it applies to the smallest values instead of the largest. In both cases, the problem is to calculate “tight” threshold values. In case of top coding this means that the threshold value should be chosen in such a way that the top category is as small as possible under the condition that this category is large enough to guarantee safety. In other words the threshold value should be chosen as large as possible, under the safety constraint. In case of bottom coding this threshold value should be as small as possible, of course. The safety is checked in terms of the frequencies of the value combinations that have to be checked.

Top and bottom coding can also be applied to continuous variables. What is important is that the values of such a variable can be linearly ordered. It is possible to calculate threshold values and lump all values larger than this value together (in case of top coding) or all smaller values (in case of bottom coding). Checking whether the top (or bottom) category is large enough is also feasible.[7]

2.2.4 The Post RAndomisation Method (PRAM)

General introduction

PRAM is a disclosure control technique that can be applied to categorical data. Basically, it is a form of deliberate misclassification, using a known probability mechanism. Applying PRAM means that for each record in a microdatafile, the score on one or more categorical variables is changed. This is done, independently of the other records, using a predetermined probability mechanism. Hence the original file is perturbed, so it will be difficult for an intruder to identify records (with certainty) as corresponding to certain individuals in the population. Since the probability mechanism that is used when applying PRAM is known, characteristics of the (latent) true data can still be estimated from the perturbed data file.

Another technique that is closely related to PRAM is the technique called Randomised Response (RR), see e.g. Warner (1965). Whereas RR is applied *before* the data are obtained, PRAM is applied *after* the data have been obtained. Both methods use known probability mechanisms to change scores on categorical variables. RR is used when interviewers have to deal with highly sensitive questions, on which the respondent is not likely to report true values in a face-to-face interview setting. By embedding the question in a pure chance experiment, the true value of the respondent is never revealed to the interviewer.

PRAM, the method

In this section a short theoretical description of PRAM is given. For a detailed description of the method, see e.g., Gouweleeuw et al. (1998a and 1998b). For a discussion of several issues concerning the method and its consequences, see e.g., De Wolf (1998).

Consider a categorical variable ξ in the original microdata file to which PRAM will be applied and denote the same categorical variable in the perturbed file by X. Moreover, assume that ξ (and hence X as well) has K categories, numbered 1, …, K. Define the transition probabilities involved in applying PRAM by $p_{kl} = \mathrm{P}(X = l \mid \xi = k)$, i.e., the probability that an original score k will be changed into the score l. PRAM is then fully described by the $K \times K$ matrix with entries p_{kl} with k, l = 1, …, K. Applying PRAM then means that, given the score k in the original file in record r, the score will be replaced by a score drawn from the probability distribution $p_{k1}, \ldots, p_{kK}$. For each record in the original file, this procedure is performed independently of the other records.

Note that p_{kk} is the probability that the score k in the original file is left unchanged. Moreover, be aware of the fact that, since PRAM involves a chance experiment, different (perturbed) microdata files will be obtained when applying the same PRAM-matrix several times to a single original microdata file (each time starting with the unperturbed original microdata file).

Since the transition probabilities are known, unbiased estimates of contingency tables are easily obtained. Other more elaborate techniques will be needed to correct for the PRAM-perturbation in more complex analyses as e.g., in loglinear models.

References on PRAM

Gouweleeuw, J.M., P. Kooiman, L.C.R.J. Willenborg and P.P. de Wolf (1998a), *Post Randomisation for Statistical Disclosure Control: Theory and Implementation*, Journal of Official Statistics, Vol. 14, 4, pp. 463 – 478.

Gouweleeuw, J.M., P. Kooiman, L.C.R.J. Willenborg and P.P. de Wolf (1998b), *The post randomisation method for protecting micropdata*, Qüestiió, Quaderns d’Estadística i Investigació Operativa, Vol. 22, 1, pp. 145 – 156.

[7] For the other values this is generally not feasible because there are too many. One should first categorize the values and apply top or bottom coding to this new, ordinal variable. Note that in practice there is no difference between a continuous variable and an ordinal variable with many categories, since we are dealing with finite populations.

Warner, S.L. (1965), *Randomized Response; a survey technique for eliminating evasive answer bias*, Journal of the American Statistical Association, Vol. 57, pp. 622 – 627.

De Wolf, P.P., J.M. Gouweleeuw, P. Kooiman and L.C.R.J. Willenborg (1998), *Reflections on PRAM*, Proceedings of the conference "Statistical Data Protection", March 25-27 1998, Lisbon, Portugal. This paper can also be found on the CASC-Website (http://neon.vb.cbs.nl/casc/ Related Papers)

2.2.5 Numerical Micro Aggregation

Microaggregation is a family of statistical disclosure control techniques for *quantitative* (numeric) microdata which belong to the substitution/perturbation category. The rationale behind microaggregation is that confidentiality rules in use allow publication of microdata sets if records correspond to groups of k or more individuals, where no individual dominates (*i.e.* contributes too much to) the group and k is a threshold value. Strict application of such confidentiality rules leads to replacing individual values with values computed on small aggregates (microaggregates) prior to publication. This is the basic principle of microaggregation.

To obtain microaggregates in a microdata set with n records, these are combined to form g groups of size at least k. For each variable, the average value over each group is computed and is used to replace each of the original averaged values. Groups are formed using a criterion of maximal similarity. Once the procedure has been completed, the resulting (modified) records can be published.

Consider a microdata set with p continuous variables and n records (*i.e.* the result of observing p variables on n individuals). A particular record can be viewed as an instance of $\boldsymbol{X'}=(X_1, \ldots , X_p)$ where the X_i are the variables. With these individuals, g groups are formed with n_i individuals in the i-th group ($n_i \geq k$ and $n = \sum_{i=1}^{g} n_i$). Denote by x_{ij} the j-th record in the i-th group; denote by $\overline{x}_i$ the average record over the i-th group, and by $\overline{x}$ the average record over the whole set of n individuals.

The optimal k-partition is defined to be the one that maximizes within-group homogeneity; the higher the within-group homogeneity, the lower the information loss, since microaggregation replaces values in a group by the group centroid. The sum of squares criterion is common to measure homogeneity in clustering. The within-groups sum of squares SSE is defined as

$$SSE = \sum_{i=1}^{g} \sum_{j=1}^{n_i} (x_{ij} - \overline{x}_i)'(x_{ij} - \overline{x}_i).$$

The lower SSE, the higher the within-group homogeneity.

The total sum of squares is

$$SST = \sum_{i=1}^{g} \sum_{j=1}^{n_i} (x_{ij} - \overline{x})'(x_{ij} - \overline{x}).$$

In terms of sums of squares, the optimal k-partition is the one that minimizes SSE. A measure L of information loss standardized between 0 and 1 can be obtained from

$$L = \frac{SSE}{SST}$$

For a microdata set consisting of many variables, these can be microaggregated together or partitioned into several groups of variables. In the latter case, independent microaggregation for each group of variables is performed. Different results are obtained depending on whether and how variables are partitioned.

Multivariate fixed-size microaggregation

The method for multivariate fixed-size microaggregation implemented in μ-Argus tries to form homogeneous groups of records by taking into account the distances between records themselves and between records and the average of all records in the data set; this method will be called MDAV (multivariate microaggregation based on Maximum Distance to Average Vector). Given a group of variables, the MDAV algorithm is as follows:

Algorithm MDAV

1. The average record $\bar{x}$ of all records in the data set is computed. The most distant record x_r to the average record $\bar{x}$ is considered (using the squared Euclidean distance).
2. The most distant record x_s from the record x_r considered in the previous step is found.
3. Two groups are formed around x_r and x_s, respectively. One group contains x_r and the *k-1* records closest to x_r. The other group contains x_s and the *k-1* records closest to x_s.
4. If there are at least *3k* records which do not belong to the two groups formed in Step 3, go to Step 1 taking as new data set the previous data set minus the groups formed in the previous instance of Step 3.
5. If there are between *3k-1* and *2k* records which do not belong to the two groups formed in Step 3, compute: a) the average record $\bar{x}$ of the remaining records; b) the most distant record x_r from $\bar{x}$; c) the group containing the *k-1* records closest to x_r and x_r; d) the other group containing the rest of records. Exit the Algorithm.
6. If there are less than *2k* records which do not belong to the groups formed in Step 3, form a new group with those records and exit the Algorithm.

The above algorithm will be applied independently to each group of variables resulting from partitioning the set of variables in the data set.

If each variable in the data set is processed independently from the other variables (*i.e.* microaggregation is performed on a variable by variable basis, which results in univariate microaggregation), a polynomial shortest-path algorithm exists to find the exact solution to optimal univariate microaggregation (Hansen and Mukherjee, 2002). Due to its high memory and computing requirements, this optimal algorithm is only recommended for small data sets.

References on microaggregation

J. Domingo-Ferrer and J. M. Mateo-Sanz (2002) "Practical data-oriented microaggregation for statistical disclosure control", *IEEE Transactions on Knowledge and Data Engineering,* vol. 14, pp. 189-201.

J. Domingo-Ferrer and V. Torra (2001) "A quantitative comparison of disclosure control methods for microdata", in *Confidentiality, Disclosure and Data Access: Theory and Practical Applications for Statistical Agencies* (eds. P. Doyle, J. Lane, J. Theeuwes and L. Zayatz), Amsterdam: North-Holland, pp. 111-133.

S. L. Hansen and S. Mukherjee (2002) "A polynomial algorithm for univariate optimal microaggregation", *IEEE Trans. on Knowledge and Data Engineering* (to appear).

2.2.6 Numerical Rank Swapping

Although originally described only for ordinal variables (Greenberg, 1987), this method can be used for any numerical variable (Moore, 1996). First, values of variable V_i are ranked in ascending order, then each ranked value of V_i is swapped with another ranked value randomly chosen within a restricted range (*e.g.* the rank of two swapped values cannot differ by more than *p*% of the total number of records). This algorithm is independently used on each variable in the original data set.

Rank swapping has been identified as a particularly well-performing methods in terms of the tradeoff between disclosure risk and information loss (see the comparison Domingo-Ferrer and Torra (2001)).

References on rank swapping

J. Domingo-Ferrer and V. Torra (2001) "A quantitative comparison of disclosure control methods for microdata", in *Confidentiality, Disclosure and Data Access: Theory and Practical Applications for Statistical Agencies* (eds. P. Doyle, J. Lane, J. Theeuwes and L. Zayatz), Amsterdam: North-Holland, pp. 111-133.

B. Greenberg (1987) "Rank swapping for ordinal data", U. S. Bureau of the Census (unpublished manuscript).

R. Moore (1996) "Controlled data swapping techniques for masking public use microdata sets", U. S. Bureau of the Census (unpublished manuscript).

2.2.7 Sullivan masking

Preface

The method has been implemented and tested at the German Federal Statistical Office. The report (Brand, 2002) describes the method and presents results of these experiments. The report concludes that, although Sullivan's masking performs very well in terms of information loss and disclosure risk, it is not easy to use. Users must be well informed, both, on the specific properties of the data set that is to be masked, and on the properties of Sullivan's method, in order to be able to choose the parameters of the method properly. If the parameter choice is not adequate, or if the input data set has some 'bad' properties, the method may fail. Therefore, the report suggests that more convenient masking algorithms (such as microaggregation and rank swapping) might be better suited for a general purpose SDC package like Argus.
The method has been integrated into the ARGUS package in spite of that, but we do recommend use of this methods only for experts on SDC, for purposes such as comparison with alternative SDC methods.
To be able to set parameters properly, we recommend for those users to at least read the report mentioned above closely.

Brief description of the method, and its parameters

Basically, the algorithm proposed by Sullivan (1989), (Fuller 1993) combines transformations with adding noise. The transformations are chosen as to ensure that the univariate distributions are preserved approximately. Additional procedures are used for correcting differences in correlations induced by transformations and mask. The transformations allow in particular for application to discrete variables. During the application of the algorithm a distance criterion is evaluated. This ensures that the distance between the masked variables and its "standard normal" counterpart is not one of the two smallest distances.

The algorithm consists of six steps (Sullivan/Fuller 1989, Fuller 1993):

1. Calculate the empirical distribution function for every variable,
2. Smooth the empirical distribution function,
3. Convert the <u>smoothed</u> empirical distribution function into a uniform random variable and this into a standard normal random variable,
4. Add noise to the standard normal variable,
5. Back-transformation to values of the distribution function,
6. Back-transformation to the original scale.

In the μ-ARGUS implementation of the method, the user is supposed to control the program by setting six parameters. In the following, we explain the six steps of the algorithm, referring to, and illustrating the use of these five parameters.
Steps 1 and 2 provide transformations for continuous and discrete variables. For a continuous variable the empirical distribution is computed in step 1. In step 2, this distribution is transformed towards the so called smoothed empirical distribution (Sullivan (1989)). These are mapped into standard normal values. Correlations of the transformed variables are nearly identical to those of the original variables as long as the variables are jointly normal. Note, that the correlations of the transformed variables may differ substantially from the original ones, when the observed variables are not normally distributed.
For details concerning the transformation of discrete variables see (Brand (2002)). It is important to note here, *that the overall total number of categories of all the discrete variables minus the number of discrete variables must not exceed the number of continuous variables!*

In the fourth step of the algorithm, the transformed variables are masked by adding noise. The relative amount of noise can be chosen setting the value of the parameter ***"Relative amount of noise"***.

Subsequently, for any masked observation, a vector of Mahalanobis-distances between masked and original observations is calculated. The criterion chosen for sufficiency is that the distance between the original record and it's masked counterpart is not one of the two smallest distances. Otherwise the mask will be repeated.

For manipulating the noise, the program offers two subroutines: If the noise is "slightly" to small, that is, if the distance of the true pair is the second smallest but the distance exceeds a certain threshold value, to be given by the user as parameter ***"Minimum difference (normal data)"***, the noise will be multiplied by a constant, specified by the user as parameter **"*Multiplier for noise 'slightly too small'*** ". Otherwise the noise will be generated completely new. Noise generation will be repeated two times at most. Even after the second time, there may still be records where the distance criterion is not satisfied. Id-numbers of those records are listed in the log-file. It is recommended that the user inspects those records manually.

In steps 5 and 6 of the algorithm, the masked values are transformed back to the original scale.
The back transformations ensure that the univariate distributions are approximately preserved. Sample means and variances are similar to those of the original data.

The correlations, however, differ, due to numerical differences and/or not jointly normally distributed original variables, or because of partial masks. In order to improve the situation, the program offers to adjust cross correlations between the variables and their masked counterparts to a robust average. The adjustment step is repeated until the observed cross correlations differ from the desired cross correlations less than a user specified amount (set parameter "***Similarity of masking for all variables***"), or until a predetermined number of iterations is exceeded. A second iterative adjustment step is included to make the off diagonal elements of the correlation matrices nearly identical.
In steps 5 and 6 of the algorithm masked values are transformed back to the original scale. This approximation can be repeated until a convergence criterion for the correlations (to be given by the user as parameter "***Allowed difference correlation matrix***") is achieved, or a certain number of iterations is exceeded.
Finally, for the method to work smoothly, the number of records on the input file (parameter **"Max batch size"**)should be below 1500. When the data input file, as supplied by the user, exceeds the maximum number of records given by this parameter, ARGUS will split the set into subsets, and will apply the method subsequently to each of the subset. Afterwards, ARGUS merges the output files into one.

References on Sullivan's Method

Brand, R. (2002): Tests of the Applicability of Sullivan's Algorithm to Synthetic Data and Real Business Data in Official Statistics, Deliverable 1.1-D1 of the EPROS Project CASC
Fuller, W.A. (1993): Masking Procedures for Microdata Disclosure Limitation, Journal of Official Statistics 9, pp. 383 – 406.

Sullivan, G.R. (1989): The Use of Added Error to Avoid Disclosure in Microdata Releases, unpublished PhD-Thesis, Iowa State University.
Sullivan, G.R. and W.A. Fuller (1989): The Use of Measurement Error to Avoid Disclosure, Proceedings of the Section on Survey Research Methods 1989, American Statistical Association, pp. 802 -- 807.

2.3 Methodology description of the individual risk approach

2.3.1 The traditional model

To be able to distinguish safe from unsafe microdata, it is necessary that a disclosure risk model is specified. Disclosure models can differ greatly in their degrees of sophistication. The basic model in μ–ARGUS is a fairly simple such model, namely one based on a thresholding rule. The understanding is that a combination of values is safe only if the (estimated) frequency of its occurrence in the population (or in the file) is above a certain threshold value. Which combinations to consider is also part of the disclosure risk model that one applies, and should be specified for μ–ARGUS by the data protector

At Statistics Netherlands we have developed some guidelines for the set of tables to be inspected. From version 3.1 of μ-Argus, a new approach for the risk has been included. This will be described in the next sections.

2.3.2 Notation

An individual risk of disclosure allows one to estimate a measure of the chance of identification of each record in the released file on the basis of the actual values observed on the public variables. In the last few years a number of proposals have been made. Fienberg and Makov (1998) and Skinner and Holmes (1998) propose, with different motivations, a log linear model for the estimation of the individual risk. Benedetti and Franconi (1998) propose a methodology for individual risk estimation based on the sampling weight which is the approach used in this version of μ-ARGUS.

In order to briefly describe the above methodology we present the notation used:

Let the released file be a random sample of size n selected from a finite population of N units. For the generic unit i in the population, we denote as w_i its probability of being included in the sample. For each record i the released file contains a set of *key* variables i.e. variables that allow identification and are accessible to the public, and the *sensitive* variables. Under the hypothesis that the key variables are discrete, a situation which is classical in household surveys and in population censuses, we can focus the analysis on each of the $k=1,\dots,K$ subpopulations defined by all the possible combinations of values of such variables; note that the maximum number of such combinations, K, can be quite high (in the order of hundred thousand). Let f_k and F_k be, respectively, the number of records in the released file and the number of units in the population with the k-th combination of categories of the key variables; F_k is unknown for each k. In the sample to be released only a subset of the total number K of combinations will be observed and only this subset, for whom $f_k>0$, is of interest to the disclosure risk estimation problem. We consider the worst case in which no measurement error in the value of the key variables occurred, an assumption which implies $F_k \geq f_k$. **For a particular record *i* in the sample to be released let *k*(*i*) be the subpopulation it belongs to and, similarly, $f_{k(i)}$ and $F_{k(i)}$ be respectively the number of records in the sample and of units in the population with the same characteristics as *i*.**
The ratio $f_{k(i)} / F_{k(i)}$ can be considered a reasonable estimate of the sampling rate relative to the subpopulation $k(i)$ in which the record i is included.

2.3.3 Risk model

To develop a quantitative model for assessing the risk of disclosure we need some assumptions on the nature of possible attacks on the privacy of an individual. We assume that the person who attempts the identification (the *intruder*) has an external data base or public register available that contains identifiers (for example name and addresses of individuals) and key variables. We express the event of the identification of an individual as the fact that record *i* in the released file and unit i^* in the register available to the intruder belongs to the same unit in the population. In **the case of independence among units** the hypothesised way in which an intruder will try to identify the target individual i^* is by comparing the combination of categories of the key variables on his/her register with the observed combinations in the released file. If only one unit in the register and one record in the released file present the same categories of the key variables then the intruder will link in a deterministic way the unit to the record and, therefore, identify the individual. Note that records in the file to be released with the same combinations of categories of the key variables will lookidentical to the intruder. For those units for whom it is not possible to find a one to one relationship between the categories of the key variables in the two files, we assume that the intruder will always try to link a record to a unit. Therefore, for those records the link will not be deterministic, but the choice will be based on probabilistic reasoning. We denote with I_i the event to make a correct link between the record *i* and the individual i^*. **We define disclosure risk r_i for the target record *i* the probability $\Pr(I_{ki})$ of establishing a correct link between record *i* and unit i^* in the population given the observed sample. Note that records in the file to be released with the same combination of categories of the key variables are identical for the intruder in terms of uncertainty to make an identification. So that the risk of each record in *k*(*i*) is equal to the risk of the record *i. So w*e define with $r_{k(i)}$ the risk of each record in the domain *k*(*i*) and we have $r_{k(i)} = r_i$ .The aim is, thus, to find for each record a reasonable estimate of such function.**

The information given by the sample in terms of identification for the record *i* consist of the frequency $f_{k(i)}$. So we have:

$$r_i^{ind} = r_{k(i)} = P(I_i \mid s) = P(I_i \mid f_{k(i)}) = \sum_{h \geq f_{k(i)}} P(I_i \mid F_{k(i)} = h, f_{k(i)}) P(F_{k(i)} = h \mid f_{k(i)})$$

Consider the worst case when the intruder has the whole population available as external information. As a consequence, if the unit *i* of the sample is unique then $\Pr(I_i \mid F_{k(i)}, f_{k(i)}) = 1$; if two units, *i** and *i'*, exist in the population presenting the same combinations of categories of the key variables and at least one of these units is included in the sample then, if we suppose that the intruder performs a probabilistic linkage, we obtain $\Pr(I_i \mid F_{k(i)} = 2, f_{k(i)}) = 1/2$. If the same probabilistic reasoning is applied to higher values of $F_{k(i)}$ we obtain:

$$r_{k(i)}^{ind} = \sum_{h \geq f_{k(i)}} \frac{1}{h} P(F_{k(i)} = h \mid f_{k(i)})$$

In order to simplify the notation in what follow we omitted the subscript *i.*

The idea is to consider $F_k|f_k$ as a random variable distributed according to a negative binomial distribution with f_k successes and probability of success p_k . Under such a framework the risk of disclosure for each record can be seen as the negative moment of order one of a negative binomial distribution. The substitution of the moment generating function of the negative binomial distribution leads to:

$$r_k^{ind} = \int_0^{\infty} \left\{ \frac{p_k \exp(-t)}{1 - q_k \exp(-t)} \right\}^{f_k} dt$$

where $q_k = 1 - p_k$. After some algebra we obtain:

$$r_k^{ind} = \left(\frac{p_k}{q_k} \right)^{f_k} \int_1^{1/f_k} \frac{(y-1)^{f_{k-1}}}{y} dy \tag{1}$$

The estimation of the parameter p_k exploiting the sampling design employed, gives as a result the following: $\hat{p}_k = f_k / \hat{F}_k = f_k / \sum_{i:k(i)=k} w_i$ **.** The substitution of $\hat{p}_k$ in the formula leads to the estimation of the risk of disclosure $\hat{r}_k$ in the case of independence.

In the formula for the risk it is also possible to include a parameter π relative to: the quality of the key variables, the probability of being included in the external file and the probability of an attempt to identify a unit in the released file. If this is the case the new risk ρ can be written as: $\hat{\rho}_k = \pi * \hat{r}_k$.

It is important to remark that if we want to evaluate the risk of disclosure of a microdata file with the function defined above, the key variables used to calculate that risk, obviously, have to be coherent with the sample design used to obtain the sample microdata.

This implies that if certain level of detail is used for a key variable in the sampling design, attention should be paid to maintain this level (or increase it) when releasing such variable in the microdata file.

For example let us consider the geographic information. If we want to release the regional variable at municipality level but the sampling design contains such variable at province level, the estimate of the risk using municipality information contained in the microdata file is not significant. As a result the estimate of the risk is not reliable. Great care should therefore be used in releasing the level of detail of the key variables contained in the sampling design.

2.4 Application of Risk model approach

2.4.1 Introduction

The purpose of the present chapter is to describe how to use the methodology of individual risk to produce a safe file. In the next section we will describe the application of this methodology at a very general, descriptive level.

In section 0 the estimation of the individual risk is presented. Section 2.4.4 explains how to evaluate the frequencies of combinations of key variables in the sample, f_k , and discusses the estima[8]tion of these frequencies in the population, $\hat{F}_k$. These two processes will be described also in the presence of missing values. In section 2.4.5 we show how to choose the threshold value necessary to the protection step. Section 2.4.6 describes the protection phase and how to produce a *safe file*.

2.4.2 An overview of the use of Individual Risk

Our approach is based on the need to handle *sample data*: the data file therefore does not include the whole population, but a subset of it, and each unit in the file represents one or more units of the population through its *individual weights*. It is therefore necessary to have this variable in the input data set.

Our method estimates the level of disclosure risk for each unit, defined as the probability of identifying an individual (for the definition of identification see the section 2.3). Once the user has chosen the key variables and selected the variable *individual weight* it is possible to calculate the individual risk. A schematic representation of the step to evaluate this risk is given in Figure 2.4.1.

After the application of the risk calculation algorithm, each record *i* will have associated its own value of the disclosure risk ρ_i. At this point, the user will input a threshold α, that he considers the maximum tolerable risk. This choice can be based on the graph representing the distribution of the individual risk in the file provide by μ-Argus.

Once α has been selected, μ-Argus can be used to apply the local suppression only to those records *i* for which $\rho_i > \alpha$, following a suppression method that minimises the number of local suppressions.

At this point μ-Argus shows a report that includes all the information relative to the protection process: the level of the threshold fixed by the user, the number of suppressions for each variable calculated by the protection algorithm and the record description of the new file .

After the protection step, the whole risk calculation algorithm can be run again, producing the current risk values and another risk graph that shows the new distribution of the risk.

At this stage, the user has two choices: a) he/she is satisfied by the result in terms of information preserved in the file and the output file is recorded as *safe file*; b) he/she discards the results, choosing to rollback to the previous risk values, e.g. in order to select another level of α.

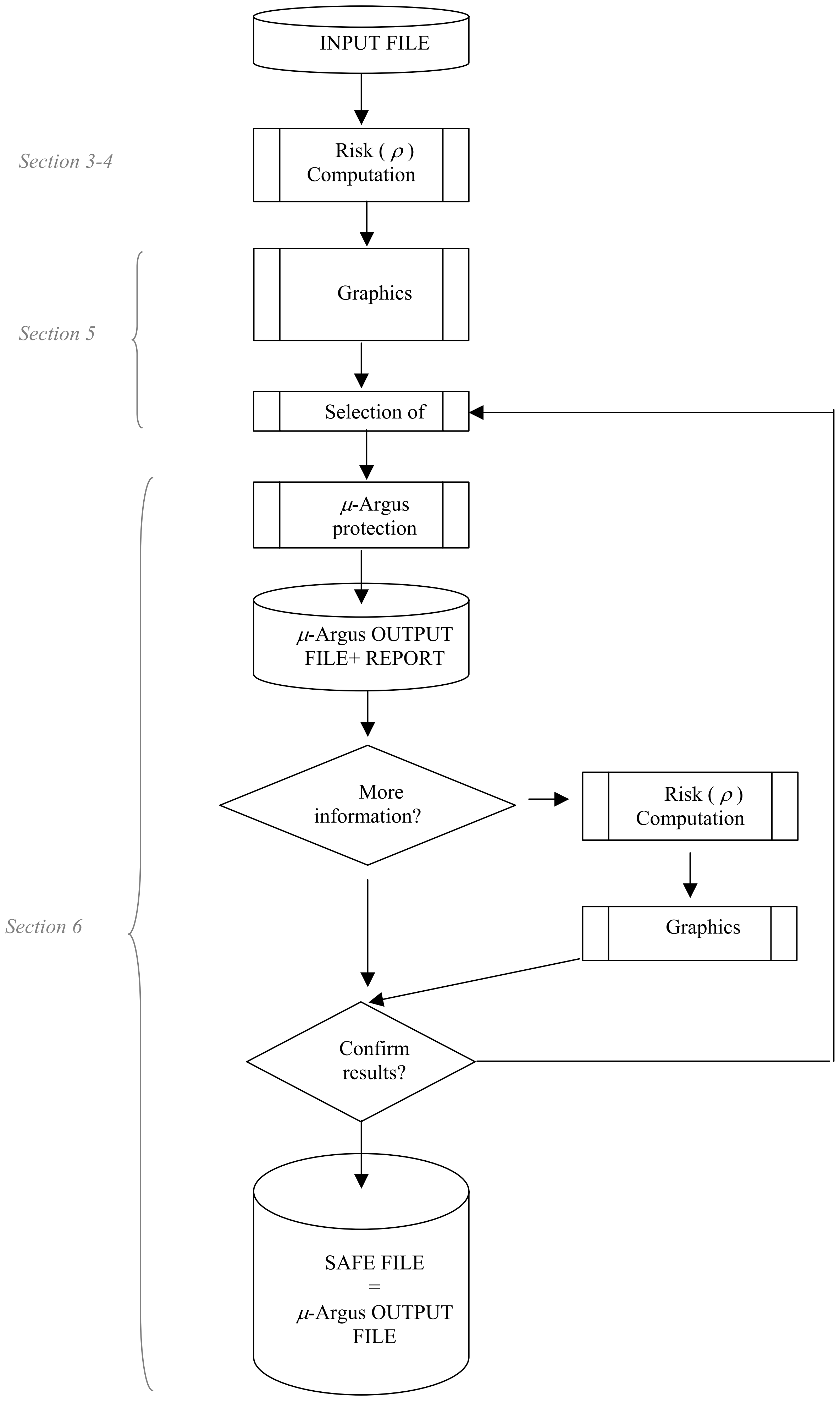


Figure 2.4.1: Process Structure

2.4.3 Base Individual Risk computation

This paragraph describes how μ-Argus calculates the base individual risk according to the methodology presented in section 2.3.3. First of all we recall that it is indispensable to have the information relative to the *individual weight.*

The individual risk, $r_i^{ind} = r_{k(i)}^{ind}$, represents the base individual risk for a unit i having combination $k(i)=k$ of key variables, and is the same for every unit belonging to the same sub-population. An approximation of the formula 1) of section 2.3.3 it is given by:

$$r_{k(i)}^{ind} = r_k^{ind} = \left(\frac{\hat{p}_k}{1-\hat{p}_k}\right)^{f_k} \left\{ A_0 \left(1 + \sum_{j=0}^{f_k-3} (-1)^{j+1} \prod_{l=0}^{j} B_l \right) + (-1)^{f_k} \log(\hat{p}_k) \right\} \tag{2}$$

where

$$\hat{p}_k = \frac{f_k}{\hat{F}_k} = \frac{f_k}{\sum_{i:k(i)=k} w_i}, \tag{3}$$

and w_i are the individual weights,

$$\text{while } B_l = \frac{(f_k - 1 - l)^2}{(l+1)(f_k - 2 - l)} \frac{\hat{p}_k^{l+2-f_k} - 1}{\hat{p}_k^{l+1-f_k} - 1} \quad \text{and} \quad A_0 = \frac{\hat{p}_k^{1-f_k} - 1}{(f_k - 1)}. \tag{4}$$

The above formulation works for $f_k \geq 3$; if $f_k = 1$ we use:

$$r_k = \frac{\hat{p}_k}{1-\hat{p}_k} \log\left(\frac{1}{\hat{p}_k}\right), \tag{4a}$$

while if $f_k = 2$:

$$r_k = \left(\frac{\hat{p}_k}{1-\hat{p}_k}\right) - \left[\left(\frac{\hat{p}_k}{1-\hat{p}_k}\right)^2 \log\left(\frac{1}{\hat{p}_k}\right) \right]. \tag{4b}$$

However, we found the task of evaluating formula (2) exceedingly heavy or even absolutely impossible when observed frequencies are too large. In these cases the introduction of a numerical approximation is convenient. We obtained satisfactory results using:

$$r_k = \frac{\hat{p}_k}{f_k - (1 - \hat{p}_k)} \tag{5}$$

This approximation is used for frequencies greater than 40. We were forced to set this value because of software limitations: however, use of a higher threshold could increase precision.

Finally, in order to consider other factors influencing the risk (such as the quality of the key variables, the intruding probability, and so on) we use a multiplying factor π so the final risk formula is given by:

$$\rho_i = \pi * r_{k(i)}^{ind} \tag{6}$$

The factor π, set to 1 as the default, may be fixed before the risk computation starts.

2.4.4 Frequencies calculation

A fundamental step for risk estimation is the computation of the frequencies f_k and $\hat{F}_k$.

First of all, we consider the population as partitioned into K sub-populations ($k = 1, \ldots, K$), defined through all the possible combinations of categories of the key variables.

It must be stressed that in the individuation of these sub-populations we use ***all*** the variables defined by the user as 'key'.

Suppose we have a file composed of 8 units:

UnitID	Key_Var1	Key_Var2	Key_Var3	Key_Var4	w_i	$f_i = f_{k(i)}$	$\hat{F}_k$
1	1	2	5	1	18	2	110
2	1	2	1	1	45,5	2	84,5
3	1	2	1	1	39	2	84,5
4	3	3	1	5	17	1	17
5	4	3	1	4	541	1	541
6	4	3	1	1	8	1	8
7	6	2	1	5	5	1	5
8	1	2	5	1	92	2	110

With $k(i) = k$ we denote the sub-population defined by the combination of categories of the key variables (*string*) in the unit i. In our example, there are 6 sub-populations, and unit 1 and 8 belong to the same sub-population identified by the string (1 , 2 , 5 ,1).

With f_k we represent the frequency (count) of units in the k^{th} sub-population that are present in the sample (i.e. in the file). The estimation of these frequencies in the population, $\hat{F}_k$, is given by the sum of the weights associated with the units belonging to that sub-population: $\hat{F}_k = \sum_{i:k(i)=k} w_i$.

In the example above, we get:

$$\text{k(1)} = \text{k(8)} = (1, 2, 5, 1) \Rightarrow \begin{cases} \hat{F}_{k(1)} = \hat{F}_{k(8)} = w_1 + w_8 = 18 + 92 = 110 \\ f_{k(1)} = f_{k(8)} = 1+1 = 2 \end{cases}$$

A problem may arise if there are missing values in the key variables.

Actually, a missing value could stand for any of the possible categories of the variable considered. Thus, in our opinion, computation of the f_k should take this into account. Consider the set of strings or combinations which are '*compatible'* with the one characterising the k^{th} sub-population, i.e. combinations which completely agree, except at most for one or more missing categories. In the presence of missing values, computation of f_k may be pursued by counting the number of units having strings compatible with the k^{th} sub-population. A similar argument can be applied to $\hat{F}_k$.

The table below shows how missing values affect computation of the relevant quantities in the context of the previous example:

UnitID	Key_Var1	Key_Var2	Key_Var3	Key_Var4	w_i	$f_i=f_{k(i)}$	$\hat{F}_k$
1	1	2	5	1	18	3	149
2	1	2	1	1	45,5	2	84,5
3	1	2	.	1	39	4	194,5
4	.	.	1	5	17	3	576
5	4	3	1	.	541	3	566
6	.	3	1	1	8	2	549
7	6	2	1	5	5	2	22
8	1	2	5	1	92	3	149

The string (1 , 2 , . , 1), associated with the UnitID 3, is compatible with the sub-populations identified by the strings (1 , 2 , 5 , 1) and (1 , 2 , 1 , 1), and, in the same way, in each of this two sub-populations it has to be counted also the unit characterised by the string (1 , 2 , . , 1).
So:

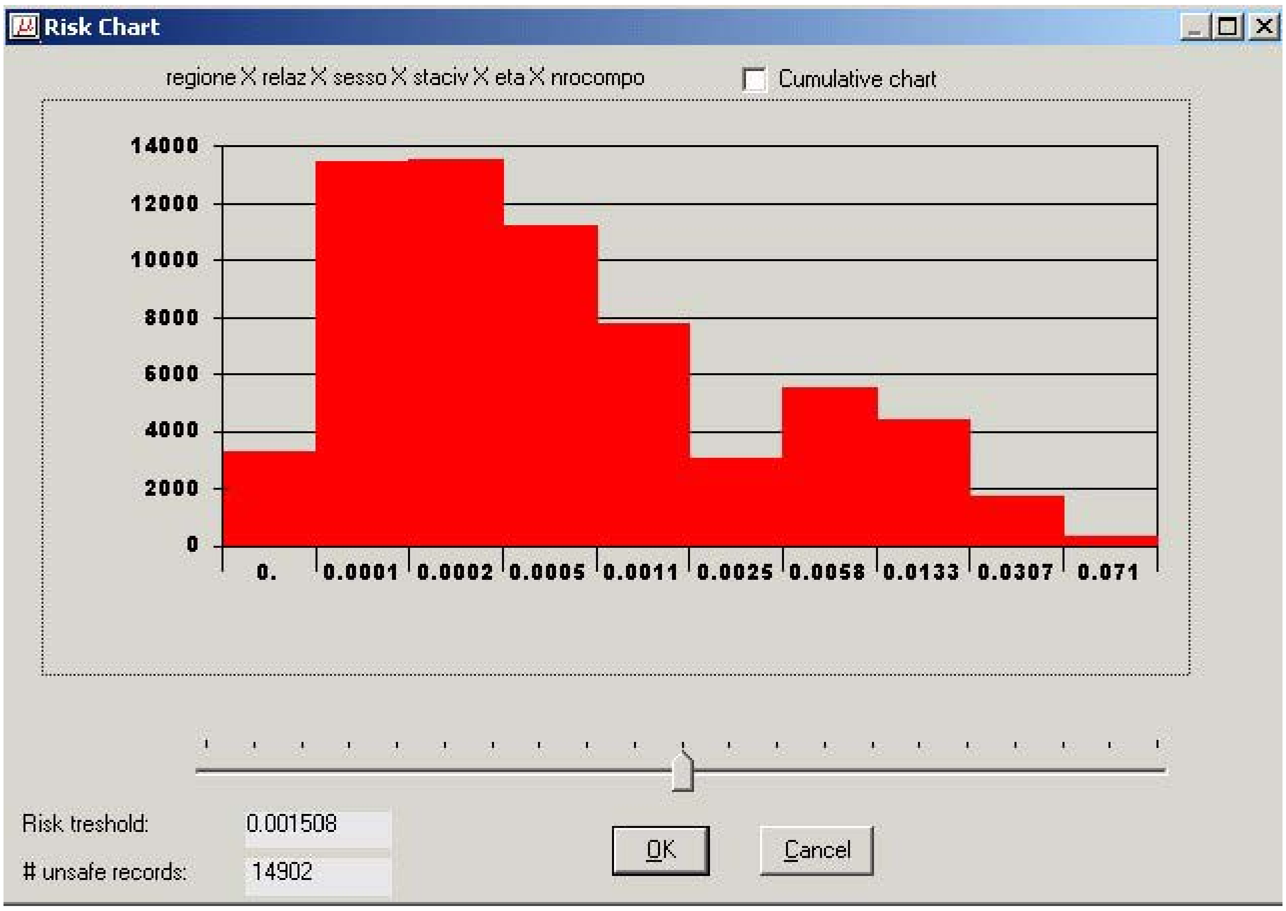
$$\hat{F}_{k(1)} = \hat{F}_{k(8)} = w_1 + w_8 + w_3 = 18 + 92 + 39 = 149,$$
$$f_{k(1)} = f_{k(8)} = 1 + 1 + 1 = 3,$$

while

$$\hat{F}_{k(3)} = w_3 + w_1 + w_8 + w_2 = 39 + 18 + 92 + 45{,}5 = 194{,}5$$
$$f_{k(3)} = 1 + 1 + 1 + 1 = 4,$$

2.4.5 Selection of the threshold

After the evaluation of the final risk ρ_i , the user has to fix the threshold α that is the maximum tolerable risk. To fix this value μ-Argus provides a graphic that shows the frequency histogram of *log*(ρ_i).



Risk treshold: 0.001508
# unsafe records: 14902

However, for the convenience of the user, the labels on the axis still report the corresponding ρ_i value, in order to better evaluate the appropriate α value. For the same goal μ-Argus shows the number of records that present a risk value greater than the threshold α.

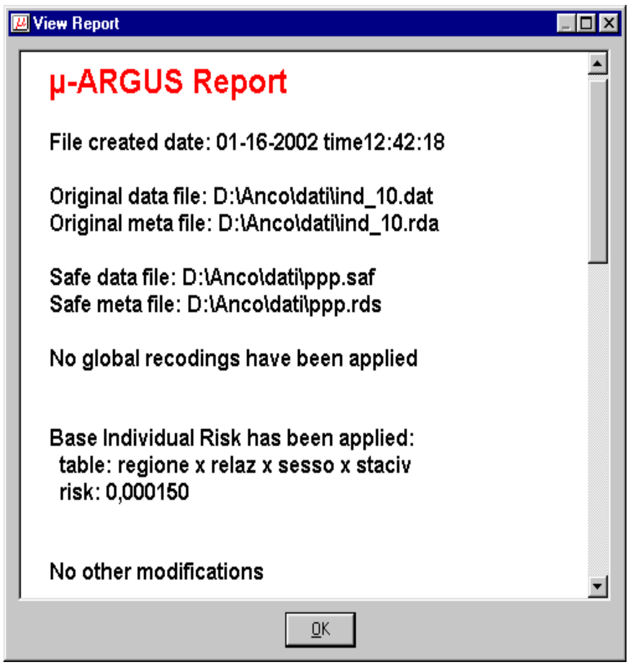
To select the threshold simply move the bar till the appropriate value shown in the Risk threshold box below.

2.4.6 Protection phase and creation of safe file

After the final risk ρ_i has been evaluated for each record and the value of α has been chosen, μ-Argus applies the protection step through use of the local suppression. For thisprotection method, an optimised procedure minimises the number of suppressions and reduces the value of the risk below the threshold α.

Once the suppression algorithm has been applied, the risk calculation can be run again on the output file produced by μ-Argus, in order to produce the new values of the risk after the protection step so that it is possible to analyse the new distribution of the risk.

At this point the user can check the information contained in the report file about the protection phase.


View Report

μ-ARGUS Report

File created date: 01-16-2002 time12:42:18

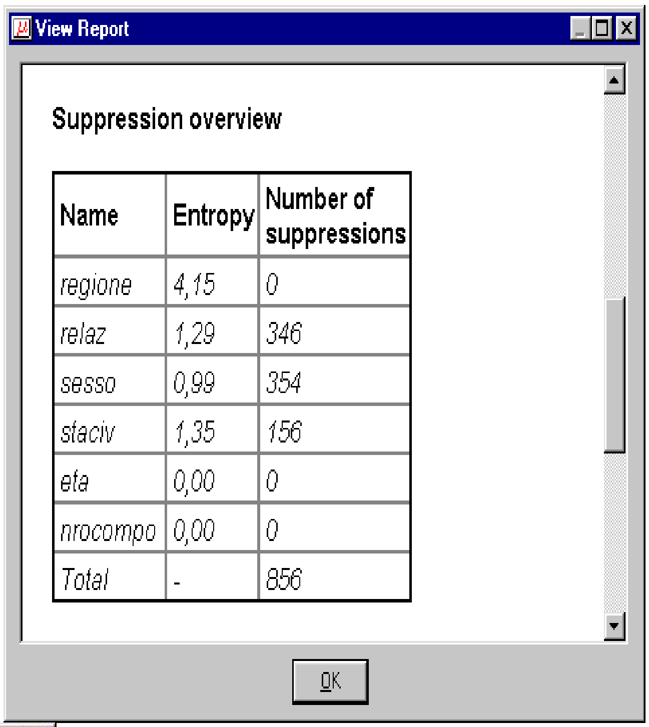
Original data file: D:\Anco\dati\ind_10.dat
Original meta file: D:\Anco\dati\ind_10.rda

Safe data file: D:\Anco\dati\ppp.saf
Safe meta file: D:\Anco\dati\ppp.rds

No global recodings have been applied

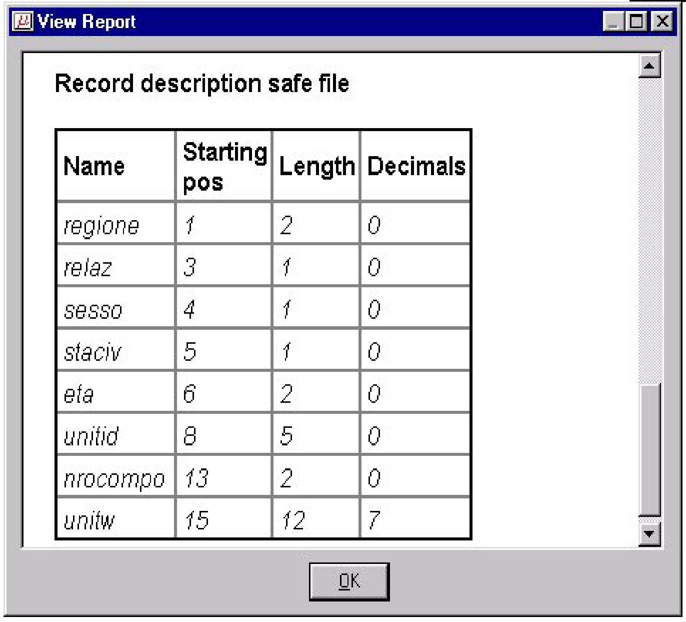
Base Individual Risk has been applied:
table: regione x relaz x sesso x staciv
risk: 0,000150

No other modifications


View Report

Suppression overview

Name	Entropy	Number of suppressions
regione	4,15	0
relaz	1,29	346
sesso	0,99	354
staciv	1,35	156
eta	0,00	0
nrocompo	0,00	0
Total	-	856


View Report

Record description safe file

Name	Starting pos	Length	Decimals
regione	1	2	0
relaz	3	1	0
sesso	4	1	0
staciv	5	1	0
eta	6	2	0
unitid	8	5	0
nrocompo	13	2	0
unitw	15	12	7

In that file it is reported the number of suppression for each variable so that it is possible to rate the information contained in the protect file.

So the user has now two options:

a) He/she is satisfied with the result. The safe file has been generated and he can leave ARGUS;

b) He/She is not satisfied by the results. He/she has two alternatives to get a better solution.
   1. Go back to the risk-specification part and specify a different α value;
   2. Go back and recode some variable and again pass through all the procedure above describe or do both i.e. recode, calculate the risk and select a new α.

References to the individual risk approach

Benedetti, R. and Franconi, L. (1998), ‘An estimation method for individual risk of disclosure based on sampling design’, submitted for publication.

Fienberg, S.E. and Makov, U.E. (1998), ‘Confidentiality. Uniqueness and disclosure Limitation for categorical data’, *Journal of Official Statistics, 14, 4, 385-397*

Skinner, C. J and Holmes D.J. (1998), ‘Estimating the Re-identification risk per record in microdata’, *Journal of Official Statistics, 14, 4, 361-372*

2.5 Information loss

A measure for disclosure risk such as a thresholding rule or the more sophisticated individual risk approach can be used as a criterion to distinguish between safe and unsafe microdata. If unsafe microdata are going to be transformed into safe microdata it is necessary to have a measure of information loss at one's disposal. This is used to limit the amount of "damage" done to the data when they are being modified. In practice, if the modifications are being carried out interactively by a data protector, this person is likely to use a crude and intuitive measure of information loss. However, if the modification is (partly) done automatically using μ–ARGUS it is necessary that the package uses some measure for the information loss. In case of applying local suppressions only, μ–ARGUS simply counts the number of local suppressions. The more suppressions the higher the information loss. In case of automatic global recoding μ–ARGUS uses an information loss measure that uses the following parameters: a valuation of the importance of an identifying variable (according to the data protector), as well as a valuation of each of the possible predefined codings for each identifying variable.

Both global recoding and local suppression lead to a loss of information, because either less detailed information is provided or some information is not given at all. A balance between global recoding and local suppression has to be found in order to make the information loss due to SDC measures as low as possible. It is recommended to start by recoding some variables globally until the number of unsafe combinations that have to be protected by local suppression is sufficiently low. The remaining unsafe combinations can then be protected by suppressing some values.

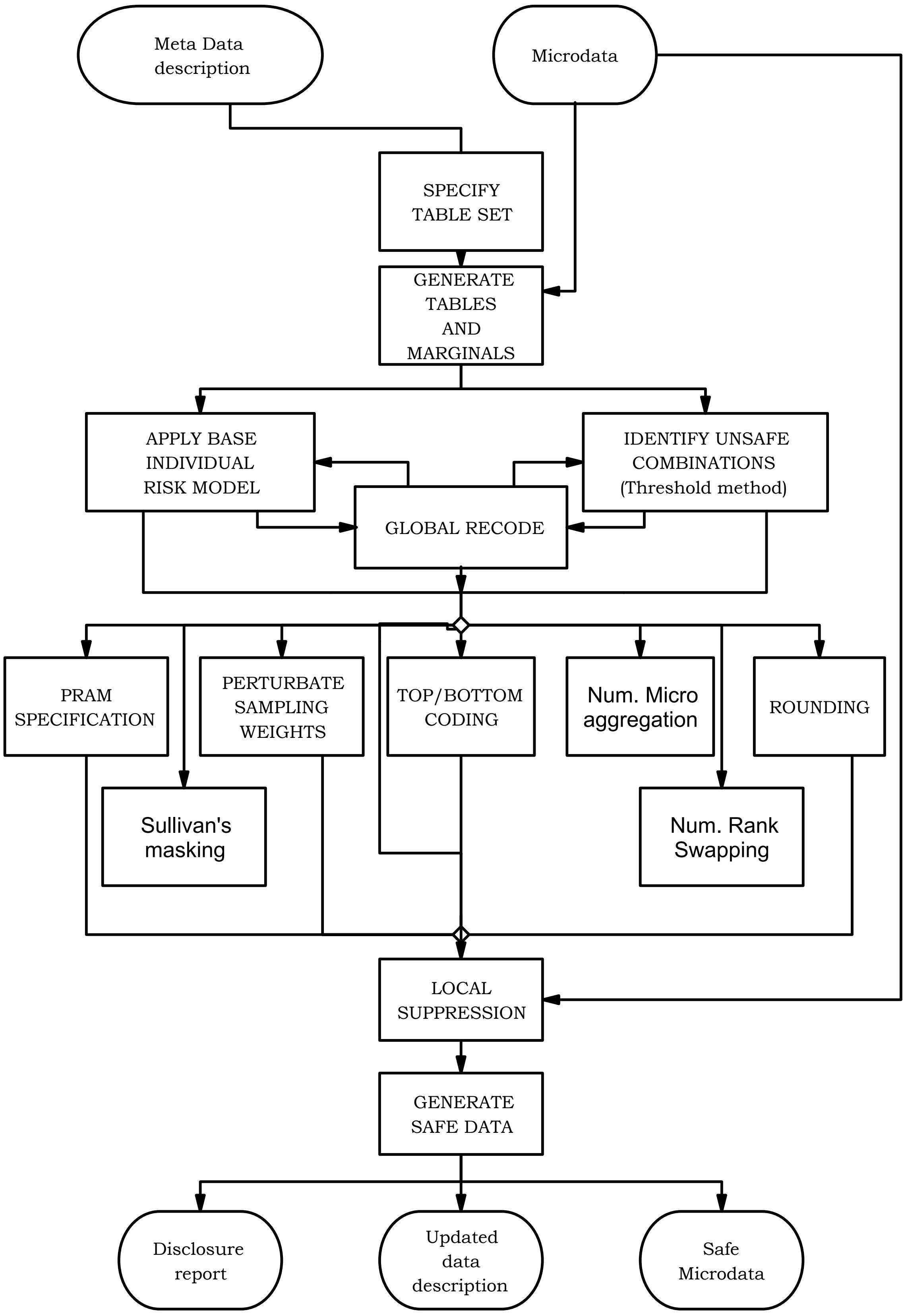
2.6 Sampling weights

So far we only discussed the most direct form of re-identification, namely that of an intruder recognising an individual. Sometimes a more indirect form of disclosure is possible. This is for instance the case if a microdata file contains certain explicit information that allows an intruder to deduce other information that a data releaser would not like to release. It is not so much the danger of a privacy threat that drives the prevention of this type of disclosure, but rather the prevention of an embarrassing situation for the data releaser. For it is somewhat embarrassing if information can be derived from the data that is not supposed to be available. An example of this is postsampling weights. Such weights are often included in a microdata file so as to correct for all sorts of defects in a sample due to selective response and over- or undersampling of certain subpopulations. However, knowledge of the procedure to calculate such weights, together with some population data (stratum frequencies) can be supposed to be generally known. With this information an intruder might in some cases be able to re-identify the stratum to which a respondent belongs. Knowing the stratum means in fact that certain attributes (that define the stratum) are known to an intruder. This could for example be a region. If it were the policy of the data releaser not to release any regional data in the file, this would be an embarrassment. This can be avoided in various ways. It is possible to draw a suitable subsample (records) from the microdata set, such that the weights for this new sample are not very different. It is also possible to add some noise to the weights, in order to hamper the stratum re-identification procedure. In μ–ARGUS it is possible to add noise to the weights.

2.7 Household variables

Some identifiers necessarily yield the same scores for all members of the same household. Such variables are called household variables. Examples of such variables are "the size of the household" and "the occupation of the head of the household". If a household variable determines the households in the population uniquely it is called a household key. Household variables in a microdata set containing records pertaining to different persons in a household, some of whom are represented in this file, may allow an intruder to group the records of persons belonging to the same household. This is an extra disclosure risk that threatens a microdata set containing such variables. The increased risk follows from the fact that households might be more easily recognisable than individual persons. μ–ARGUS is able to deal with household variables. This means that if a value of a household variable in a record is suppressed, then the corresponding values are suppressed in the records referring to the other members of the same household.

2.8 Functional design of μ-ARGUS



3. A tour of μ-ARGUS

This section will give the reader a brief introduction through the use of μ–ARGUS. Some Windows experience is assumed. In this section we will use a demo-dataset called DEMODATA, which has been supplied with the μ–ARGUS software. In section 4 a more systematic description of the different parts of μ–ARGUS will be given. Please note that all options in μ-ARGUS can be accessed via the drop-down menu, but also via the toolbar. In this tour we will use the drop-down menu.

3.1 Preparation

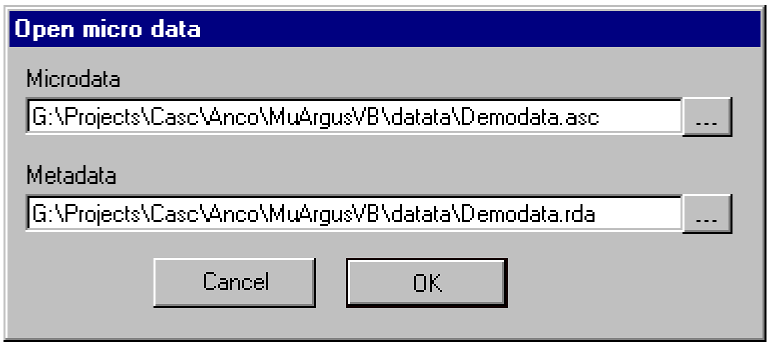
When you double click the μ–ARGUS icon the program starts. To start the disclosure control with μ–ARGUS you need to have a microdata file and the metadata describing this microdata file. Three options are now available for microdata file format:

- fixed format (as previously),
- free format (with a defined separator),
- free format with variable names in the first line (called FreeWithMeta format in μ–ARGUS). A format very convenient for e.g. SAS-users.

Here we restrict our attention to fixed format input. The other formats are discussed in Section 4.

In the methods presented in Section 3, the microdata file must be a fixed format ASCII file. If you click (File|Open microdata) you can specify the name of the microdata file. The program assumes the extension .asc, but you can use your own extension.

Open micro data

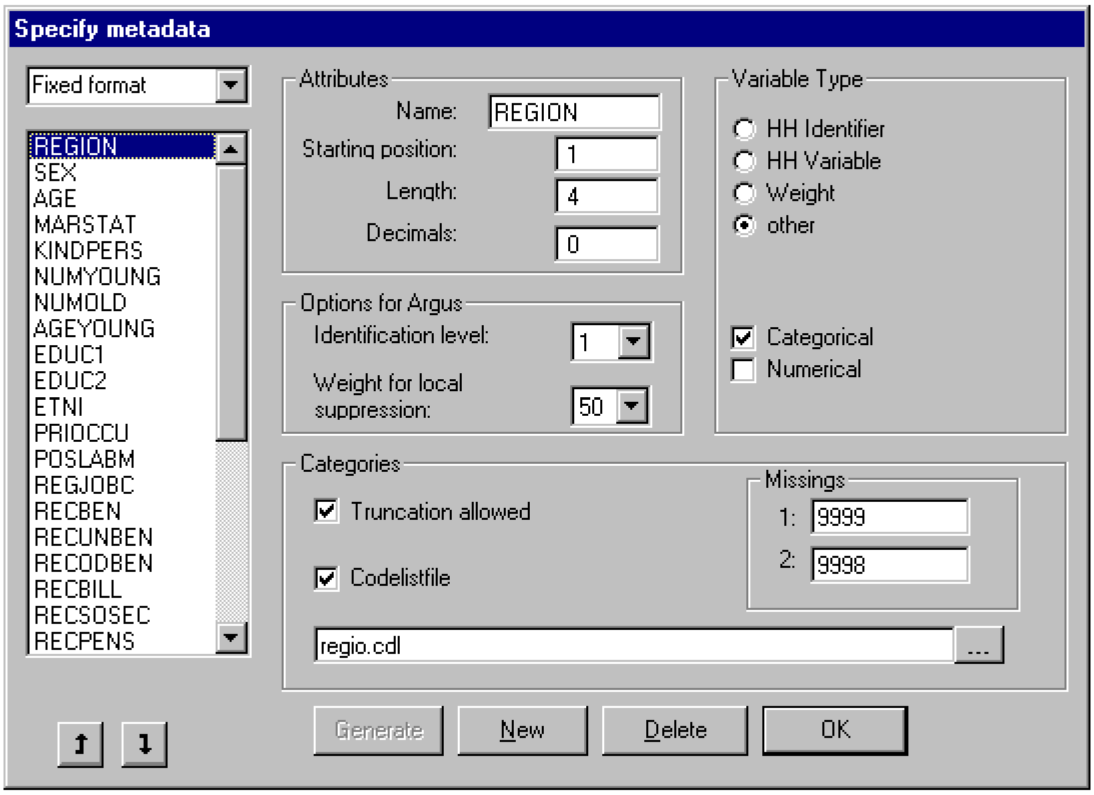

Open micro data

Microdata
G:\Projects\Casc\Anco\MuArgusVB\datata\Demodata.asc

Metadata
G:\Projects\Casc\Anco\MuArgusVB\datata\Demodata.rda

The metadata describing this ASCII file is stored in a separate file. For this file the program assumes the extension .rda (Record Description for Argus), but you can select another extension. If no metadata file is specified, the program has the facility to specify the metadata interactively via the menu option (Specify|Metadata). In this section, we will give a description of how the use of (Specify|Metadata) results in a display of the information in the metadata file for μ–ARGUS. When you enter or change the metadata interactively using μ–ARGUS the option (Specify|Metadata) will bring you to this screen:

Specify metadata


Specify metadata

Fixed format

REGION
SEX
AGE
MARSTAT
KINDPERS
NUMYOUNG
NUMOLD
AGEYOUNG
EDUC1
EDUC2
ETNI
PRIOCCU
POSLABM
REGJOBC
RECBEN
RECUNBEN
RECODBEN
RECBILL
RECSOSEC
RECPENS

Attributes
Name: REGION
Starting position: 1
Length: 4
Decimals: 0

Options for Argus
Identification level: 1
Weight for local suppression: 50

Variable Type
( ) HH Identifier
( ) HH Variable
( ) Weight
(•) other

[✓] Categorical
[ ] Numerical

Categories
[✓] Truncation allowed
[✓] Codelistfile
regio.cdl

Missings
1: 9999
2: 9998

In a first tour you should leave the fields as they are, but we will explain here the meaning of these fields.

In the left pane of the ‘Specify metadata’ window the available variables of the dataset are shown. In the other fields the various attributes, relevant for the disclosure control process are shown.

- The starting position in each record of the datafile will be familiar.
- The Variable Type:
  - HH Identifier: The unique identifier of a household
  - HH Variable: A variable that by nature has the same value for each member of a household
  - Weight: The variable is a sampling weight
  - Categorical: A categorical variable can be used as a spanning variable in a table. For μ-ARGUS this variable can be defined as identifying
  - Numerical: A numerical variable can be used for top/bottom coding and rounding
- The identification level is an option to easily generate the set of tables to be inspected in the disclosure control process. The contents of these sets are based on certain rules used by Statistics Netherlands, but they could easily be used in other situations as well. This procedure is further explained in the next section
  - **0**: an individual cannot be identified by this variable and it will not play a role in the disclosure control process.
  - **1**: the variable is most identifying.
  - **2**: the variable is more identifying.
  - **3**: the variable is identifying.
- The weight for local suppression. When at the end of a μ-ARGUS run the remaining unsafe combinations are suppressed by local suppression, one of the options is to use the user-supplied weights to select the variables that will be suppressed. These weights are specified here.
- The codelists of the variables used to span the tables is always generated by μ-ARGUS itself. However the user can supply the name of a codelist file (see example below in the metadata file example). The labels in this codelist file are then used when displaying information on this

variable. Besides the name of the codelist file it is possible to indicate whether truncation is a feasible way of recoding. This is the case when the codelist is hierarchical.

- One or two missing values can be specified per variable. Missing values play a specific role in the SDC-process, as missing values will be imputed when local suppression is applied. Note that the weight variable cannot have a missing value, but categorical variables need to have at least one missing value.

The SDC-file stores the information in the format of a keyword (enclosed by '<', '>') followed by the relevant parameters.

• <RECODABLE>	This variable may be recoded.
• <CODELIST>	Name of the codelist file.
• <IDLEVEL>	Identification level.
• <TRUNCABLE>	The codelist is hierarchical and therefore removing one or more digits is a relevant way of recoding.
• <NUMERIC>	The variable is numeric and rounding etc. is allowed.
• <DECIMALS>	The number of decimal positions for a (numeric) variable.
• <WEIGHT>	The variable contains the sample weights. Randomising (check) the weight is an option by specifying the amount of noise to add, as a percentage .
• <HOUSE_ID>	This variable is a household identification.
• <HOUSEHOLD>	A household variable contains typically the same value for each member of a household. When the suppression of the value for one member is necessary, it will be done for each member.
• <SUPPRESSWEIGHT>	Priority weight for the selection of the suppression pattern; default value = 50.

An example of a meta data file:

```
REGION 1  4  9999  9998
 <RECODABLE>
 <CODELIST>  "Regio.cdl"
 <IDLEVEL>  1
 <SUPPRESSWEIGHT>  50
SEX 5  1 9
 <RECODABLE>
 <CODELIST>  "Sex.cdl"
 <IDLEVEL>  2
 <SUPPRESSWEIGHT>  50
AGE 6  2  99
 <RECODABLE>
 <IDLEVEL>  3
 <SUPPRESSWEIGHT>  50
 <NUMERIC>
……….
COMPCODE 31  3  999
 <RECODABLE>
 <IDLEVEL>  3
 <SUPPRESSWEIGHT>  50
 <TRUNCABLE>
…………
WEIGHT 44  6
 <NUMERIC>
 <DECIMALS>  2
 <WEIGHT>
INCOME 50  8  99999999
```

```
<NUMERIC>
```

In case of as free-format data file the first line of the metadata file reads

```
<SEPARATOR> ","
```

In case of a free format file with the variable names on the first line of the datafile, the second line of the metadata file reads

```
<NAMESINFRONT>
```

An example of a codelist file of the Dutch provinces:

```
20,Groningen
21,Friesland
22,Drenthe
23,Overijssel
24,Flevoland
25,Gelderland
26,Utrecht
27,Noord-Holland
28,Zuid-Holland
29,Zeeland
30,Noord-Brabant
31,Limburg
```

3.1.1 Specify tables set.

When the metadata is ready, the set of combinations to be inspected by μ–ARGUS can be specified. Either you specify the tables manually or you use one of the two rules to generate this set. If you select the tables manually you select the variables in the left pane and move them to the middle pane (with the '>'-button). If the table is ready and the appropriate threshold has been specified you add the table to the set of tables in the right pane with the '->'-button. Each table can have a different threshold.

If you use the 'automatic specification of tables' , you will have two options, based on the rules used in Statistics Netherlands, 1: using the identification level and 2: all tables up to a given dimension.

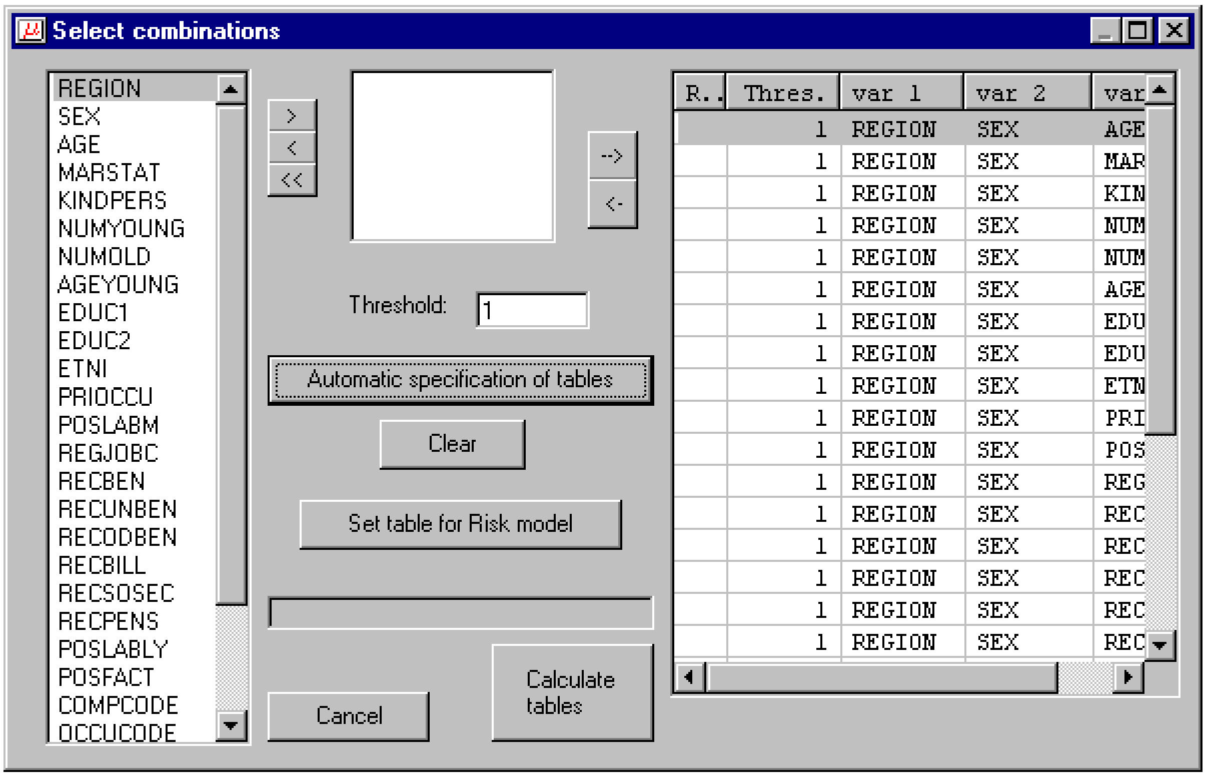
1. In the first case the set of tables is constructed as follows: For each variable the parameter identification level must be specified. The selection of variables is restricted to those variables where the identification level is >0. Each variable with a level = 1 is combined with each variable of level ≤ 2 with each variable of level ≤3 etc. At Statistics Netherlands this rule is used to produce microdata files for researchers (MUC). For all tables the same threshold must be specified.
2. In the second case all tables up to a given dimension are used. The selection of variables is restricted to those variables where the identification level is >0. For each dimension a different threshold can be specified. At Statistics Netherlands this rule is used to produce public use files (PUF).

When the set of tables is constructed using the generator, it is still possible to make adjustments. You can add additional ones or delete certain tables. If a large number of tables is being generated due to the specific parameters μ-ARGUS will ask whether it should proceed or not.

Independent of the way in which the set of tables is constructed, the threshold for the tables must be specified - separate thresholds for each number of dimensions in the table (1d, 2d, etc). This threshold is the maximum value of a cell in a table, which is still considered unsafe. Above this threshold a cell is considered safe. In case of a sample survey this value is calculated at the level of the sample. To translate this at the level of the population the sample fraction should be taken into account by the user.

If the new risk model is used, you select a table for this model by pressing the button 'set table for risk model'. A description of this model can be found in section 3.2.3. A restriction is that there cannot be an overlap between the tables used for the classical threshold method and the new risk-model. Mixing the basic model and the new risk approach makes no sense. Therefore the overlapping tables will be removed automatically.

The window for the specification of the combinations



REGION
SEX
AGE
MARSTAT
KINDPERS
NUMYOUNG
NUMOLD
AGEYOUNG
EDUC1
EDUC2
ETNI
PRIOCCU
POSLABM
REGJOBC
RECBEN
RECUNBEN
RECODBEN
RECBILL
RECSOSEC
RECPENS
POSLABLY
POSFACT
COMPCODE
OCCUCODE

Threshold: 1

R..	Thres.	var 1	var 2	var
	1	REGION	SEX	AGE
	1	REGION	SEX	MAR
	1	REGION	SEX	KIN
	1	REGION	SEX	NUM
	1	REGION	SEX	NUM
	1	REGION	SEX	AGE
	1	REGION	SEX	EDU
	1	REGION	SEX	EDU
	1	REGION	SEX	ETN
	1	REGION	SEX	PRI
	1	REGION	SEX	POS
	1	REGION	SEX	REG
	1	REGION	SEX	REC
	1	REGION	SEX	REC
	1	REGION	SEX	REC
	1	REGION	SEX	REC
	1	REGION	SEX	REC

When you are satisfied with the specified set of tables you press the button 'Calculate tables'.

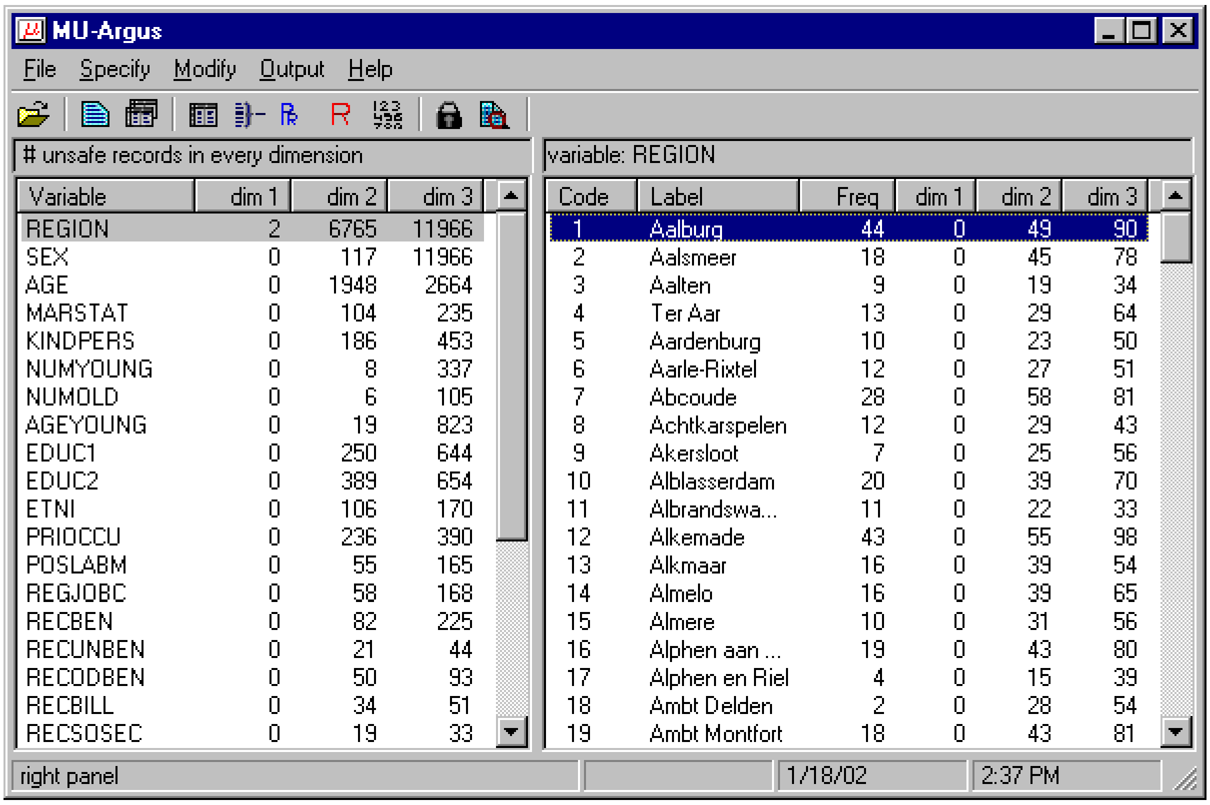
This will be done in three phases. In the **first phase** μ–ARGUS will inspect the microdata file and construct lists of existing codes for the active variables. These lists are constructed independently from the specified codelists. This is done in order to prevent from problems arising when some existing codes might be missing in the codelists. When the codelists are available μ–ARGUS will in the **second phase** calculate the tables to be inspected. In the final **third phase** the marginals of the tables will be calculated. The progress is shown by the progress bar.

After this you are ready to start the disclosure control process.

3.2 The process of Disclosure Control

When all the preparations have been made the actual process of Disclosure Control can start. The two main actions that can be performed are Global Recoding and Local Suppression. Additionally you can modify numerical variables by 'top/bottom coding', 'rounding', 'numerical Micro Aggregation', 'numerical Rank Swapping' and 'Sullivan's Masking'. The user can select and perform manually the Global Recodings (via Modify|Global Recode). The central μ-ARGUS window is the main window where for each variable the number of unsafe combinations is shown by dimension of the table. The left hand window displays the number of unsafe cells in one-dimensional tables involving the named variable; then the same information is given for tables of greater dimension up to the highest dimension created by the user. For the selected variable in the left window, the number of unsafe combinations for each category of the selected variable is shown in the right hand window. The higher the number of unsafe combinations for a variable the more likely it is that the variable needs to be globally recoded in order to achieve a safe microdata file.

The main menu for μ-ARGUS, showing the number of unsafe combinations per variable.



# unsafe records in every dimension

Variable	dim 1	dim 2	dim 3
REGION	2	6765	11966
SEX	0	117	11966
AGE	0	1948	2664
MARSTAT	0	104	235
KINDPERS	0	186	453
NUMYOUNG	0	8	337
NUMOLD	0	6	105
AGEYOUNG	0	19	823
EDUC1	0	250	644
EDUC2	0	389	654
ETNI	0	106	170
PRIOCCU	0	236	390
POSLABM	0	55	165
REGJOBC	0	58	168
RECBEN	0	82	225
RECUNBEN	0	21	44
RECODBEN	0	50	93
RECBILL	0	34	51
RECSOSEC	0	19	33

variable: REGION

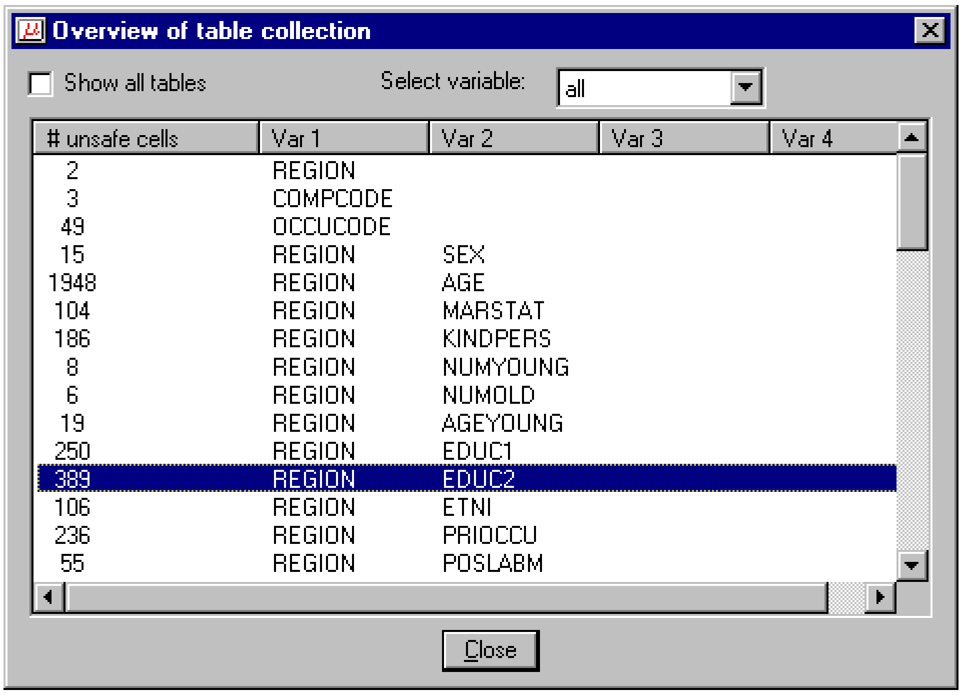
Code	Label	Freq	dim 1	dim 2	dim 3
1	Aalburg	44	0	49	90
2	Aalsmeer	18	0	45	78
3	Aalten	9	0	19	34
4	Ter Aar	13	0	29	64
5	Aardenburg	10	0	23	50
6	Aarle-Rixtel	12	0	27	51
7	Abcoude	28	0	58	81
8	Achtkarspelen	12	0	29	43
9	Akersloot	7	0	25	56
10	Alblasserdam	20	0	39	70
11	Albrandswa...	11	0	22	33
12	Alkemade	43	0	55	98
13	Alkmaar	16	0	39	54
14	Almelo	16	0	39	65
15	Almere	10	0	31	56
16	Alphen aan ...	19	0	43	80
17	Alphen en Riel	4	0	15	39
18	Ambt Delden	2	0	28	54
19	Ambt Montfort	18	0	43	81

right panel 1/18/02 2:37 PM

The unsafe combinations are calculated by checking the threshold value in each combination. Please note that if e.g. a three-dimensional combination is checked also the one and two dimensional marginal combinations are checked. In this window also the scores of these marginal tables are shown.

As the unsafe combinations are calculated for the tables specified, it is possible to inspect the number of unsafe cells per table via the menu option (Modify|Show tables). The following window is shown:

Number of unsafe cells per table

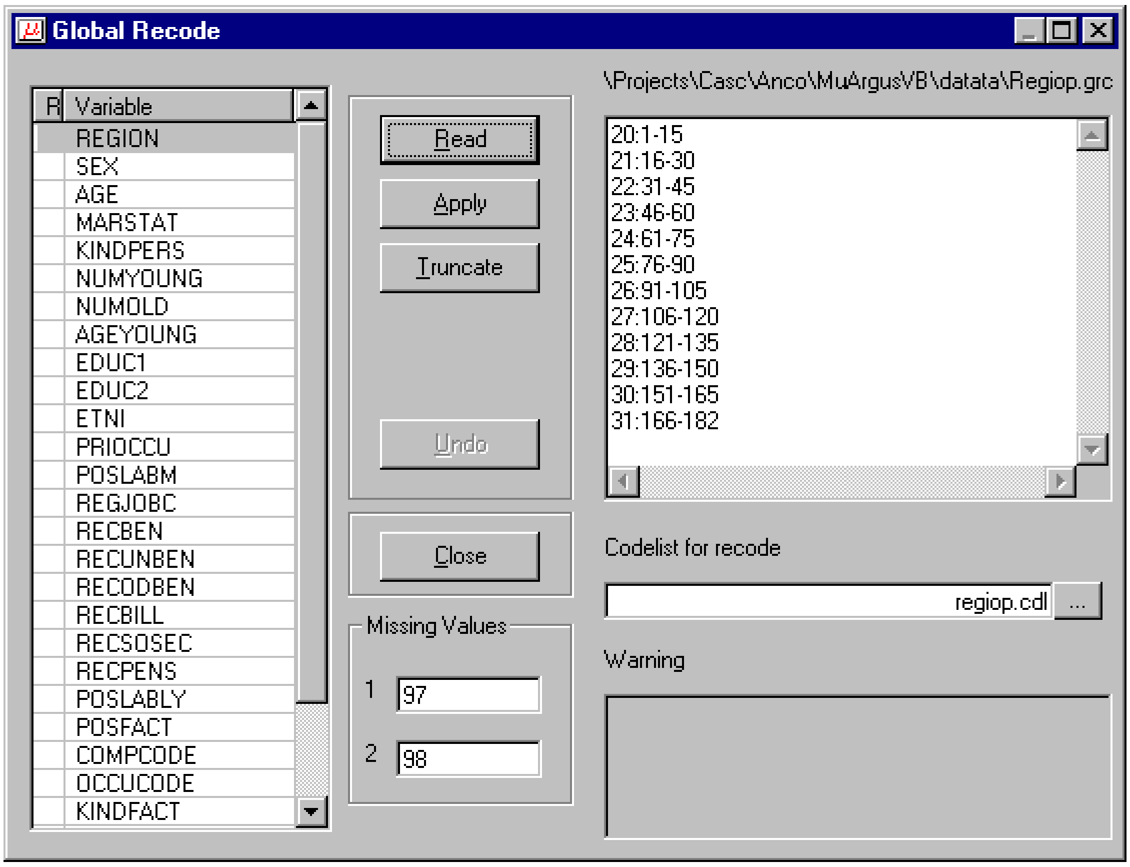


# unsafe cells	Var 1	Var 2	Var 3	Var 4
2	REGION			
3	COMPCODE			
49	OCCUCODE			
15	REGION	SEX		
1948	REGION	AGE		
104	REGION	MARSTAT		
186	REGION	KINDPERS		
8	REGION	NUMYOUNG		
6	REGION	NUMOLD		
19	REGION	AGEYOUNG		
250	REGION	EDUC1		
389	REGION	EDUC2		
106	REGION	ETNI		
236	REGION	PRIOCCU		
55	REGION	POSLABM		

3.2.1 Global Recoding

Via the μ–ARGUS main window the user decides which variable will be recoded. Double click on the variable in the main window or choosing the menu option Modify|Global recode brings you to the global recode window. In this window the global recodings for each variable can be specified.

Specifying and selecting global recodings



Global Recode

\Projects\Casc\Anco\MuArgusVB\datata\Regiop.grc

R	Variable
	REGION
	SEX
	AGE
	MARSTAT
	KINDPERS
	NUMYOUNG
	NUMOLD
	AGEYOUNG
	EDUC1
	EDUC2
	ETNI
	PRIOCCU
	POSLABM
	REGJOBC
	RECBEN
	RECUNBEN
	RECODBEN
	RECBILL
	RECSOSEC
	RECPENS
	POSLABLY
	POSFACT
	COMPCODE
	OCCUCODE
	KINDFACT

```
20:1-15
21:16-30
22:31-45
23:46-60
24:61-75
25:76-90
26:91-105
27:106-120
28:121-135
29:136-150
30:151-165
31:166-182
```

Codelist for recode: regiop.cdl

Missing Values: 1: 97; 2: 98

Warning

1. On the left you see a list of the variables. You can recode a variable by applying a recode scheme or by truncating a variable. Truncation is only possible if in the metadata description the variable has been designed as hierarchical.
2. If you apply a global recoding or truncate a variable, the colour of the variable will be changed into red and an 'R' or 'T ' will be indicated in the first column of the list-window.
3. If no global recode file is available you can make one yourself or read an existing one. You can also edit the recoding.
4. In the edit-box 'Codelist' you can specify the name of the file containing the codelist (labels) for the recoded variable.
5. It is also possible to change the codes for the missing values.
6. If you apply a global recoding, μ-ARGUS will show the result of the recoding in the warning-window. This can either be an error message showing on which line the error occurred, but also it can be a warning. For example some codes in the base code file have not been recoded. However this could be the purpose of the global recoding.

A recoding scheme

Global recoding means that certain categories of a variable are collapsed into one new category. The syntax of the recode file is as follows: each line in the file corresponds to 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.

Between the two windows there are five buttons.

1. “Read” allows you to read a previously made recode file into the program.
2. “Apply” applies the recode in the file to the tables. This will add the corresponding cells. If the cell was unsafe the new cell might exceed the “Safe Limit” and become safe.
3. “Truncate”: Variables which are preceded by an asterisk in the left window can be truncated. This means that you are allowed to chop of digits from the end of the code for the categories. When you click on “Truncate” you can specify how many digits must be chopped. This number always applies to the original codes: if you want to truncate the same variable twice, each time one digit, you have to fill in “2” the second time.
4. “Undo” will undo any recodes, truncations and replaces performed on a variable. This brings the variable to its original state.
5. “Close” will close the dialog box and bring you back to the main window of μ-ARGUS and show the updated overview of the unsafe combinations per variable.

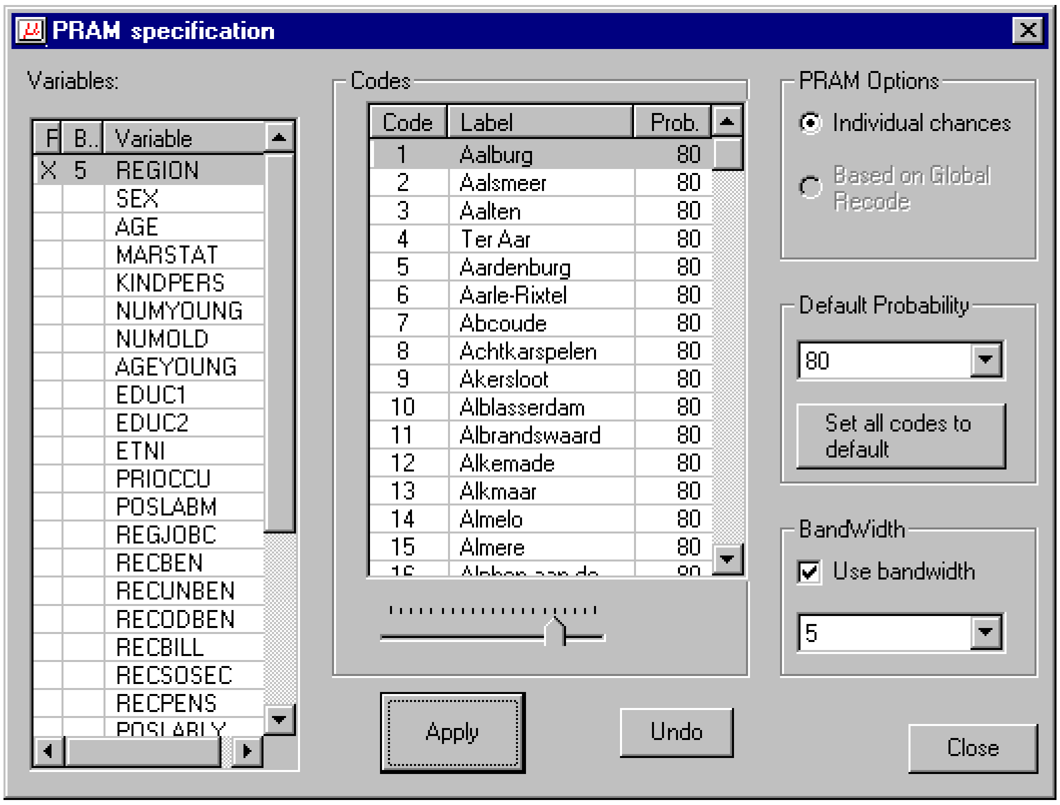
3.2.2 PRAM specification

Post Randomisation is a technique where noise is deliberately added to categorical variables. See also section 2.2.4. Additionally the parameters of the added noise can be made available to the users of a dataset. They can then make better use of the datafile. At this stage you can specify the amount of noise added to the variable. Typically this is done by specifying the chance of changing a category into an other category. In the current implementation you specify the chance of not changing a certain category. The complement is the chance of changing the value into an other category. You can limit the spread of this by specifying the bandwidth. This implies that the chance of changing a category is limited to the nearest n categories.

The individual chance per category can be changed by using the slider. Press the apply button to accept the settings.

The ’Undo’-button will reset the specification.

If a variable is PRAM-med, this is shown in the listbox by a P in the first column and an indication that the bandwidth has been used or not.


PRAM specification

Variables:

F	B..	Variable
X	5	REGION
		SEX
		AGE
		MARSTAT
		KINDPERS
		NUMYOUNG
		NUMOLD
		AGEYOUNG
		EDUC1
		EDUC2
		ETNI
		PRIOCCU
		POSLABM
		REGJOBC
		RECBEN
		RECUNBEN
		RECODBEN
		RECBILL
		RECSOSEC
		RECPENS

Codes

Code	Label	Prob.
1	Aalburg	80
2	Aalsmeer	80
3	Aalten	80
4	Ter Aar	80
5	Aardenburg	80
6	Aarle-Rixtel	80
7	Abcoude	80
8	Achtkarspelen	80
9	Akersloot	80
10	Alblasserdam	80
11	Albrandswaard	80
12	Alkemade	80
13	Alkmaar	80
14	Almelo	80
15	Almere	80

PRAM Options
[● Individual chances]
○ Based on Global Recode

Default Probability: 80

Set all codes to default

BandWidth
[✓ Use bandwidth]
5

Apply   Undo   Close

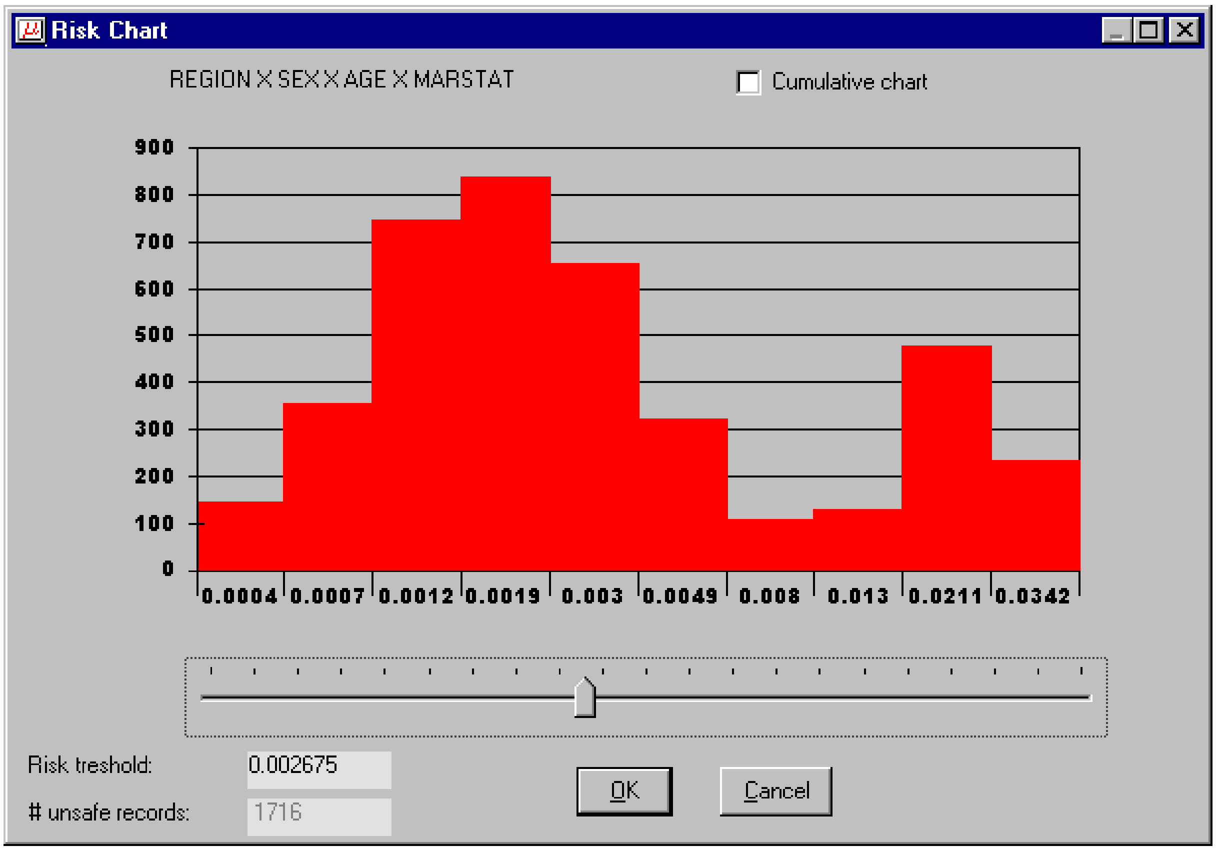
3.2.3 Risk approach

If for certain combinations the individual risk approach has been selected then you can specify the level of risk. All records with an individual risk above this level will be treated as unsafe. Local suppression will then be applied in the final stage of ARGUS. Of course it is possible to apply global recoding and re-inspect the risk chart. The level of risk acceptable for making a datafile available is of course dependent of the usage of a file, whether it goes to a research-institute or whether it is meant as a public use file.

When the graph is shown, the slider can be used to adjust the risk-level. Also the number of unsafe records is constantly adjusted as well.

If more than one risk-table had been specified, you are first asked to specify for which table you want to specify the risk-threshold.

Individual risk chart



Risk treshold: 0.002675

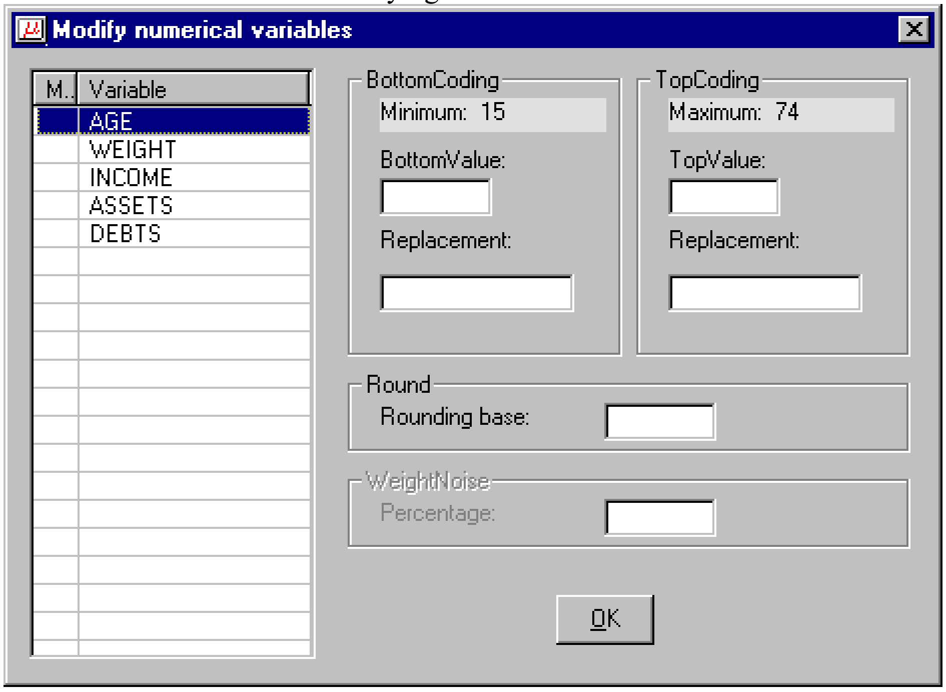
unsafe records: 1716

3.2.4 Numerical variables

For numerical variables you have the possibility of modifying them here. You can simply round the variable or apply top/bottom coding. Top/Bottom coding means that all values above/below a certain threshold are replace by a given value. For rounding the rounding base should be supplied.

Additionally for the weight variable you could add noise. All these modifications will only take place when the safe datafile is generated. At this stage the information is only stored.

Modifying numerical variables



Minimum: 15

BottomValue:

Replacement:

Maximum: 74

TopValue:

Replacement:

Rounding base:

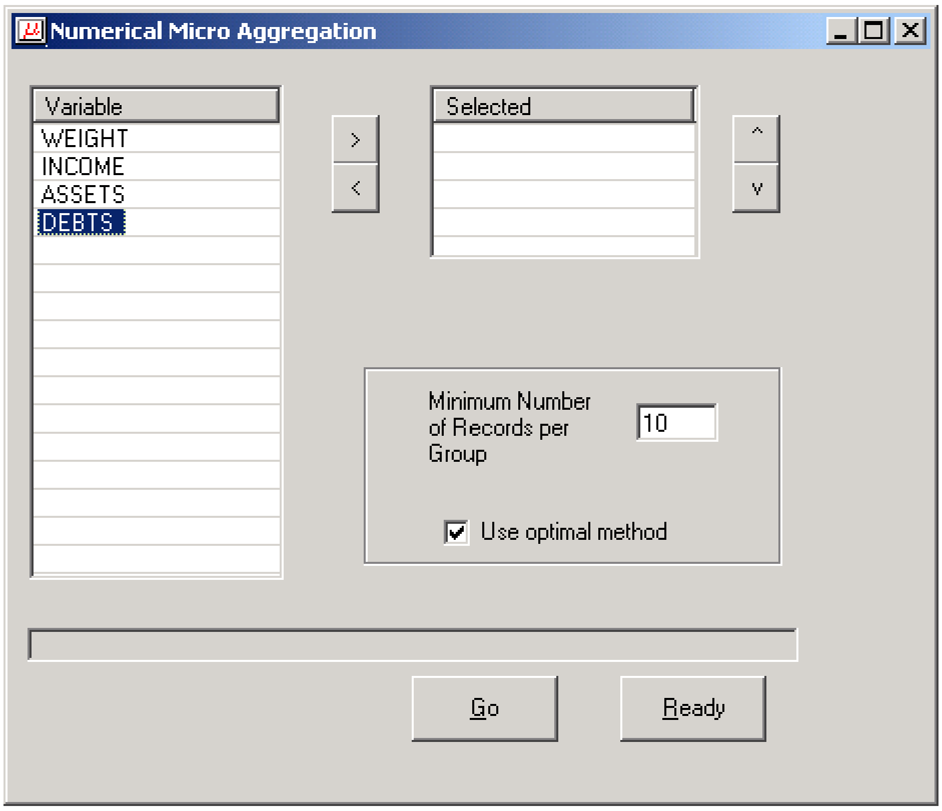
Percentage:

3.2.5 Numerical Micro Aggregation

To obtain microaggregates in a microdata set with *n* records, these are combined to form *g* groups of size at least *k*. For each variable, the average value over each group is computed and is used to replace each of the original averaged values. Groups are formed using a criterion of maximal similarity. Further details of the technique are given in Section 2.2.5.

The user can select the variable(s) required for numerical microaggregation as well as selecting the minimum number of records per group. The user can also specify that the optimal method should be used. However, the optimal method is not available for numerical micro-aggregation of a single variable.

Numerical Micro aggregation



Minimum Number of Records per Group: 10

[✓] Use optimal method

3.2.6 Numerical Rank Swapping

This technique, applied to variable V_i, involves first ranking values of V_i in ascending order, then each ranked value of V_i is swapped with another ranked value randomly chosen within a restricted range (*e.g.* the rank of two swapped values cannot differ by more than *p*% of the total number of records). This algorithm is independently used on each variable in the original data set.

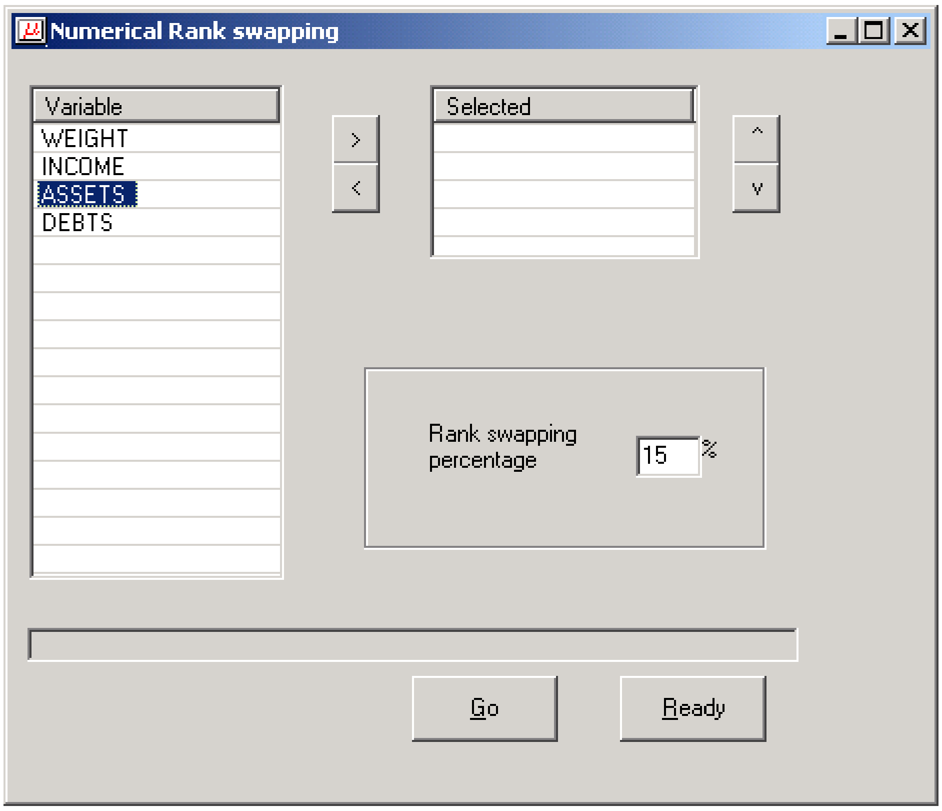
Only one parameter must be specified in order to use rank swapping:

- 'Rank swapping percentage': two values will not be swapped unless their rank difference is below this percentage of the total number of records. The value of this parameter must lie between 0 and 100. Values of *p* around 15 are recommended.

Example

If 'Rank swapping percentage' = 5, and there are 1000 rows, then a specific record will be swapped with records at a maximum rank distance of 50.

Numerical rank swapping



Rank swapping percentage 15 %

3.2.7 Sullivan Masking

This facility is included in the software for use by those with expert knowledge of Sullivan masking. Further guidance on its use can be obtained from the CASC web site. A runtime version of GAUSS, which is required for the Sullivan Masking approach, can also be downloaded from the web site.. The window displaying the input required for calling Sullivan Masking is presented after a description of the program messages.

How to read the program messages

When the program has been started, a window displaying program messages appears on the screen, starting with lines such as:

```
      » run c:\gsrun36\programs\sul_anco.prg.gcg;
      constants
      realtive amount of noise      0.50000000
      minimum difference for multiplying with constant      0.50000000
      minimum difference for sufficient mask if initial mask is not sufficient      0.50000000
      Maximum difference between cross correlations      0.10000000
      Maximum difference between elements of correlation matrices      0.10000000
  Partial mask: First column not to be masked      5.0000000
```

For the most part, these messages are also stored in a log-file. In the following, we illustrate typical messages of the log-file using an instance with two continuous, and one discrete variable, while occasionally referring to the messages displayed on the screen.

Display of input parameters

```
      constants
      realtive amount of noise 0.50000000
      minimum difference for multiplying with constant 0.50000000
      minimum difference for sufficient mask if initial mask is not sufficient 0.50000000
      Maximum difference between cross correlations 0.10000000
```

```
Maximum difference between elements of correlation matrices 0.10000000
Partial mask: First column not to be masked  --
```

Statistics for continuous variables:

- Mean and Standard Deviation for original and transformed variables,
- Correlations, Differences between original and transformed variables

```
continouus variables

descriptive statistics
means and standarddeviations

 mean                stdc
 564240.00000000 252139.95175406
 54438.44000000  25999.13278896

correlations
transformed data
1.00000000 -0.03924644
-0.03924644 1.00000000

original data
1.00000000 -0.03730867
-0.03730867 1.00000000

difference in correlations 0.00000000 -0.00193777
-0.00193777 0.00000000
```

Statistics for discrete variables:

- Mean and Standard Deviation for original and transformed variables,
- Correlations, Differences between original and transformed variables

```
binary variables
deskriptive statistics
means and standard deviations
  mean          stdc
     0.492     0.500

correlations
transformed data
 1.00000000
original data
1.00000000

Difference in correlations 2.2204460e-016
```

Record id-numbers for those records where the criterion for sufficiency of the masking is not satisfied.

```
in 32.0  cases the internal distance criterion is not fulfilled
case numbers
29.0 55.0 57.0 78.0 119. 121. 155. 162. 168. 178. 192. 216. 253. 265. 270.
273. 276. 293. 301. 343. 377. 380. 384. 386. 388. 394. 397. 429. 433. 453.
479. 494.
```

Because the sufficiency criterion is not always satisfied in this instance, the masking is repeated for part of the data set, that is, the noise is “generated completely new” for 31 records...,

```
in 31.0  cases errors will be generated completly new:
29.0 55.0 57.0 78.0 119. 121. 155. 162. 168. 178. 192. 216. 265. 270. 273.
276. 293. 301. 343. 377. 380. 384. 386. 388. 394. 397. 429. 433. 453. 479.
494.
iteration of mask 2.0
```

..., but still, the sufficiency criterion is not satisfied for 30 records,...

```
in 30.0  cases the internal distance criterion is not fulfilled
case numbers
55.0 57.0 78.0 119. 121. 155. 162. 168. 178. 192. 216. 265. 270. 273. 276.
293. 301. 343. 377. 380. 384. 386. 388. 394. 397. 429. 433. 453. 479. 494.
```

..., and the noise is “generated completely new”...,

```
in 30.0  cases errors will be generated completly new:
55.0 57.0 78.0 119. 121. 155. 162. 168. 178. 192. 216. 265. 270. 273. 276.
293. 301. 343. 377. 380. 384. 386. 388. 394. 397. 429. 433. 453. 479. 494.
iteration of mask 3.0
```

.... Although this procedure has still left some records without sufficient protection according to the distance criterion, the algorithm stops here, because the number of iterations is limited to three.

According to the internal distance criterion the cases listed above[9] are not sufficiently masked

```
55.0 57.0 78.0 119. 121. 155. 162. 168. 178. 192. 216. 265. 270. 273. 276.
293. 301. 343. 377. 380. 384. 386. 388. 394. 397. 429. 433. 453. 479. 494.
```

In this instance, no adjustment is required.

```
no adjustment required
```

If it were, it would be up to the user to decide:

```
Adjusting cross-correlations? (yes = 1)?1
```

The program would then do some adjustment, writing messages on the screen such as

```
Cross-correlations: target 0.77
Iteration 1.0
cross correlations
0.88 0.78 0.68 0.76 1.0
Iteration 2.0
cross correlations
0.78 0.77 0.73 0.77 1.0
correlations
Original
1.0 0.64 0.37 0.22 0.077
0.64 1.0 0.61 0.12 0.13
0.37 0.61 1.0 0.063 0.046
0.22 0.12 0.063 1.0 0.0090
0.077 0.13 0.046 0.0090 1.0
Masked
1.0 0.57 0.31 0.13 0.090
0.57 1.0 0.42 0.070 0.16
0.31 0.42 1.0 0.022 0.060
0.13 0.070 0.022 1.0 -0.017
0.090 0.16 0.060 -0.017 1.0
Difference
2.2e-016 0.063 0.055 0.093 -0.014
0.063 1.1e-016 0.19 0.052 -0.032
0.055 0.19 0.00 0.041 -0.014
0.093 0.052 0.041 -2.2e-016 0.026
-0.014 -0.032 -0.014 0.026 0.00
```

..., and thereafter ask the user to decide upon adjustment of the correlation matrix (see sec. 1, above):

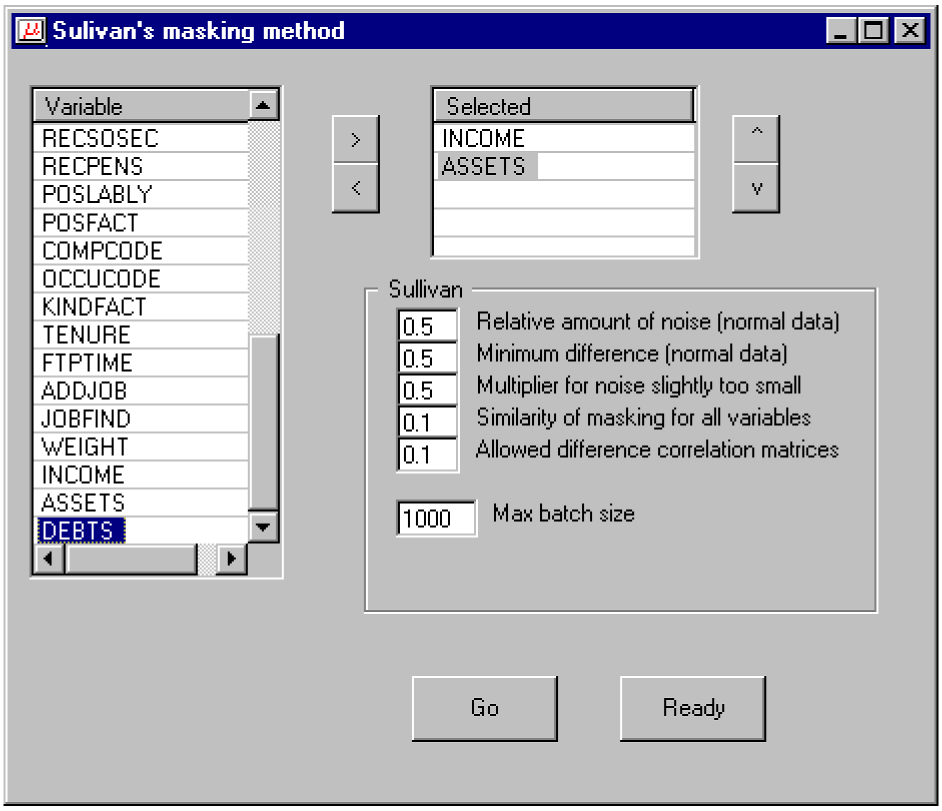
```
Adjusting correlations? (yes = 1)? 1
```

The adjustment steps produce log messages such as

```
Variable 2.0
initial correlations
0.64 1.0 0.61 0.12 0.13
0.57 1.0 0.42 0.070 0.16 Maximum difference: 0.19
5.0 iterations
comparison of correlations
0.64 1.0 0.61 0.12 0.13
0.55 1.0 0.52 0.10 0.11
Maximum difference:
0.093
```

The output is stored according to user request. The log-file concludes with warnings of records without sufficient protection:

```
saving adjusted data?, yes = 2
saving datstern and binmask
in 35.000  cases the internal distance criterion is not fulfilled
case numbers
31.000 48.000 55.000 57.000 78.000 119.000 121.000 155.000 162.000 168.000 178.000 192.000 216.000 265.000 270.000 273.000
276.000 293.000 294.000 301.000 343.000 377.000 380.000 384.000 386.000 388.000 390.000 394.000 397.000 410.000 429.000
433.000 453.000 479.000 494.000
```



Sullivan

0.5	Relative amount of noise (normal data)
0.5	Minimum difference (normal data)
0.5	Multiplier for noise slightly too small
0.1	Similarity of masking for all variables
0.1	Allowed difference correlation matrices
1000	Max batch size

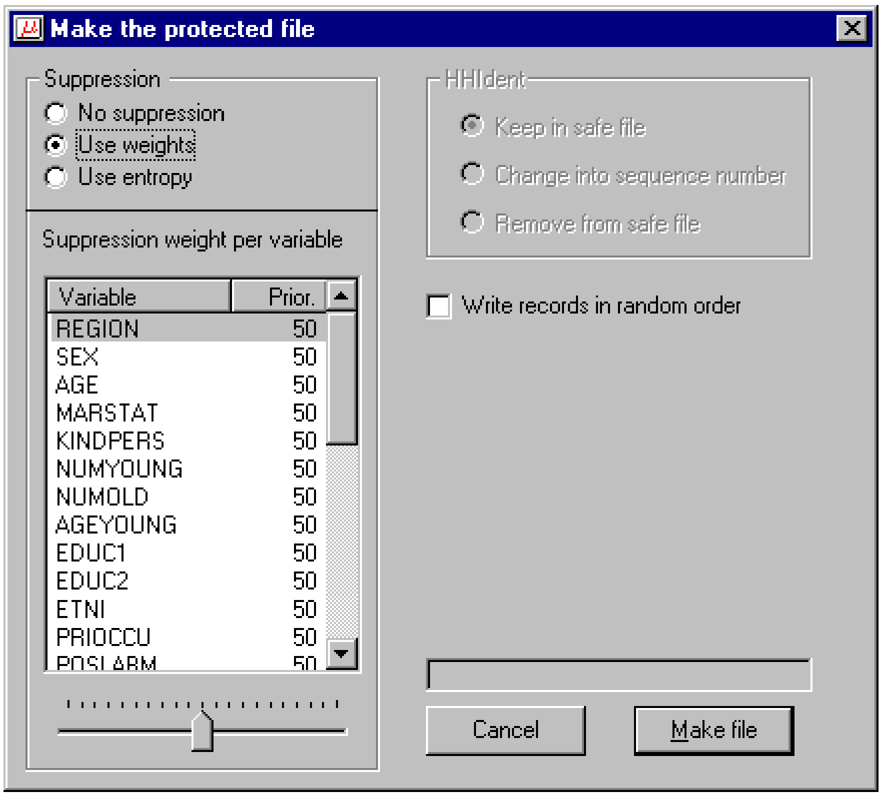
3.3 Local suppression and making a safe microdata file

When the user is satisfied with the set of global recodings, it is time to solve the remaining unsafe combinations with local suppression. For this you select the option (Output|Make suppressed file). To protect the remaining unsafe combinations μ–ARGUS will suppress certain values (i.e. impute a missing value). If there is more than one unsafe combination in a record and the unsafe combinations have a variable in common, then μ–ARGUS will suppress the common variable. If not μ–ARGUS will have to choose one of the variables minimising the loss of information. This information loss can be either based on the entropy function or a user-specified priority function.

The user-defined priority function can be used e.g. if a certain variable has been recoded already so much, or if a user thinks that for a certain variable the imputation of a missing value is very undesirable.

Other options are the treatment of the household identifier: keep it as it is, change it into a simple sequence number or remove it.

Also it is possible to change the order of the records in the output file.



Variable	Prior.
REGION	50
SEX	50
AGE	50
MARSTAT	50
KINDPERS	50
NUMYOUNG	50
NUMOLD	50
AGEYOUNG	50
EDUC1	50
EDUC2	50
ETNI	50
PRIOCCU	50
POSLARM	50

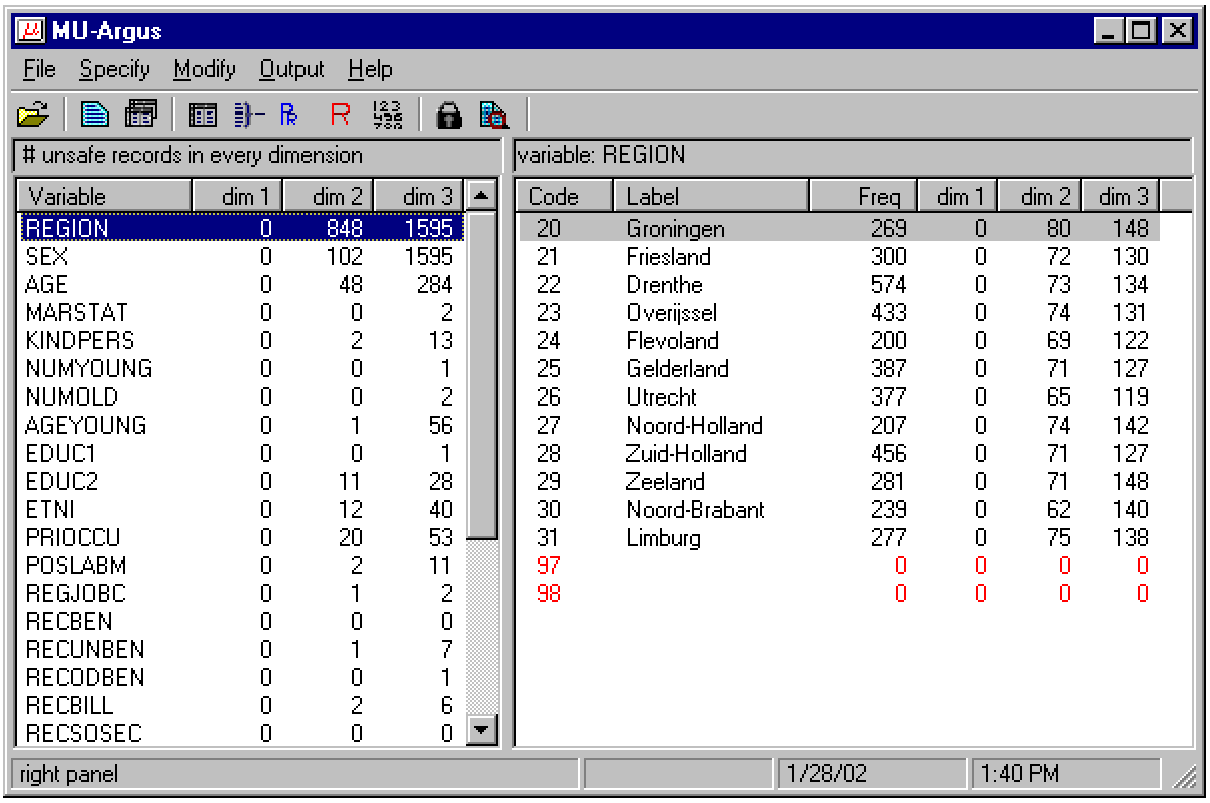
Then you press “Make File”. The program then makes new microdata, the program will apply the global recodings, replacements and truncations as ordered, and protect the last unsafe cells by locally suppressing categories in single records or sometimes, in case of a household variable, in all records belonging to the same household. If there is a weight variable it will be randomised, if requested. If there is a household identifier then there is an option to keep this identifier in the datafile, or change the household identifier into a sequence number. Please bear in mind that this variable can be very sensitive. The suggested extension of the filename is ”.saf”. It also makes a new metafile, which is the same as for the original microdata. Specific meta-information for µ-ARGUS will be removed. For the new metafile the extension is “.rds”.

When finished making the output file the program makes a report file. The format of this file is HTML. This will make it possible to view the report when ARGUS has been finished with a browser. The report file can be viewed with (Output|View report), but will also be shown automatically when ARGUS has made a safe file. This report will give a summary of the actions performed by µ-ARGUS.

4. Description of 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 μ–ARGUS is invoked.

4.1 Main window


MU-Argus
File Specify Modify Output Help
# unsafe records in every dimension
variable: REGION
right panel
1/28/02
1:40 PM

Variable	dim 1	dim 2	dim 3
REGION	0	848	1595
SEX	0	102	1595
AGE	0	48	284
MARSTAT	0	0	2
KINDPERS	0	2	13
NUMYOUNG	0	0	1
NUMOLD	0	0	2
AGEYOUNG	0	1	56
EDUC1	0	0	1
EDUC2	0	11	28
ETNI	0	12	40
PRIOCCU	0	20	53
POSLABM	0	2	11
REGJOBC	0	1	2
RECBEN	0	0	0
RECUNBEN	0	1	7
RECODBEN	0	0	1
RECBILL	0	2	6
RECSOSEC	0	0	0

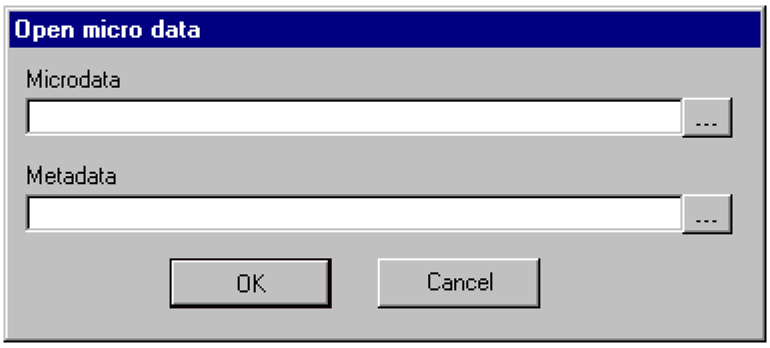
Code	Label	Freq	dim 1	dim 2	dim 3
20	Groningen	269	0	80	148
21	Friesland	300	0	72	130
22	Drenthe	574	0	73	134
23	Overijssel	433	0	74	131
24	Flevoland	200	0	69	122
25	Gelderland	387	0	71	127
26	Utrecht	377	0	65	119
27	Noord-Holland	207	0	74	142
28	Zuid-Holland	456	0	71	127
29	Zeeland	281	0	71	148
30	Noord-Brabant	239	0	62	140
31	Limburg	277	0	75	138
97		0	0	0	0
98		0	0	0	0

Overview of the menu-items:

File	Specify	Modify	Output	Help
Open microdata	Metadata	Show Table Collection	Make suppressed file	Contents
Exit	Combinations	Global recode	View report	About
		PRAM specification		
		Risk specification		
		Modify numeric var		
		Numerical Microagregation		
		Numerical Rank Swapping		
		Sullivan masking		

4.2 *The File Menu*

4.2.1 File|OpenMicrodata



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 files of types listbox. When you have selected the microdata-file a suggestion for the metafile is given but only when this file exists. The data file can be fixed format, free format (with defined separator) or 'FreeWithMeta' format. Data files in the FreeWithMeta format (with variable names specified in the headings over the columns of data) can be generated from SAS. If this input format is used, a first version of the metadata file can be generated automatically by μ–ARGUS by pressing the Generate button (in the Specify metadata window) after the data have been read into the program.

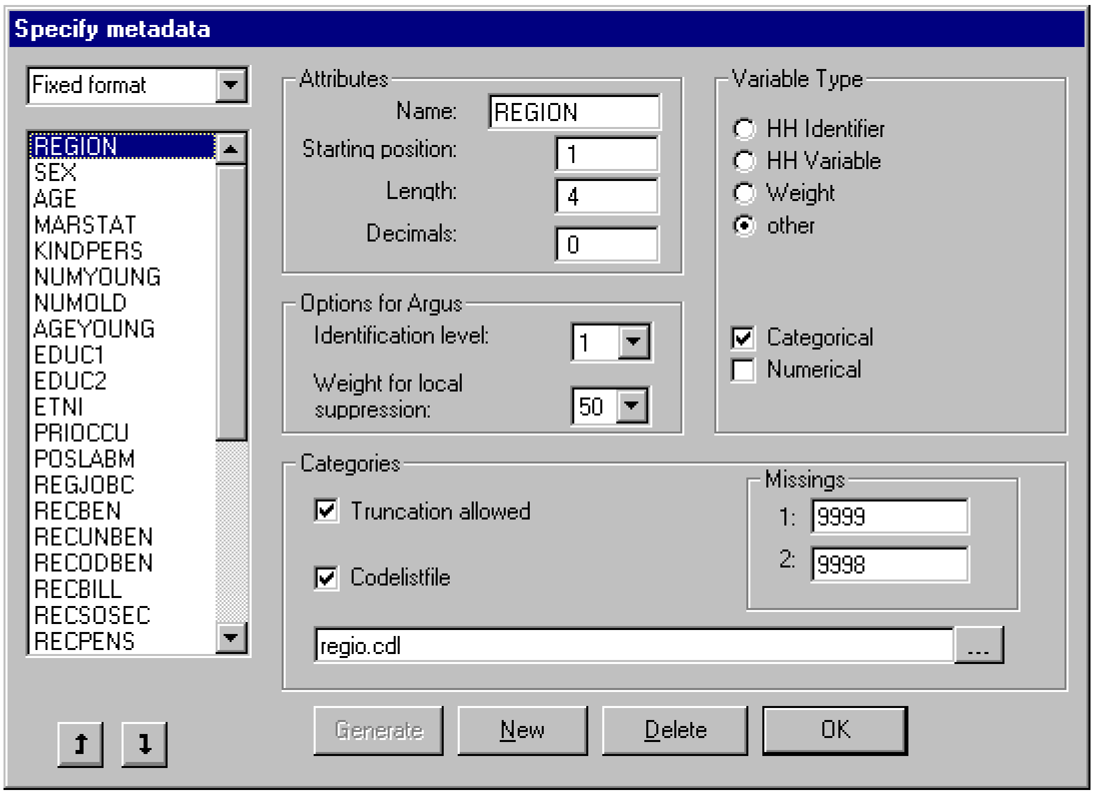
- Before you click **OK** you must have filled in the name of the microdata-file.
- If you change your mind you can cancel the whole operation by clicking **Cancel**.

4.2.2 File|Exit

Terminates the μ–ARGUS-session.

4.3 *The Specify menu*

4.3.1 Specify|MetaFile


Specify metadata

Fixed format

REGION
SEX
AGE
MARSTAT
KINDPERS
NUMYOUNG
NUMOLD
AGEYOUNG
EDUC1
EDUC2
ETNI
PRIOCCU
POSLABM
REGJOBC
RECBEN
RECUNBEN
RECODBEN
RECBILL
RECSOSEC
RECPENS

Attributes
Name: REGION
Starting position: 1
Length: 4
Decimals: 0

Variable Type
HH Identifier
HH Variable
Weight
[✓ other]

[✓ Categorical]
Numerical

Options for Argus
Identification level: 1
Weight for local suppression: 50

Categories
[✓ Truncation allowed]
[✓ Codelistfile]
regio.cdl

Missings
1: 9999
2: 9998

In this dialog box all the attributes of the identifying variables can be specified.

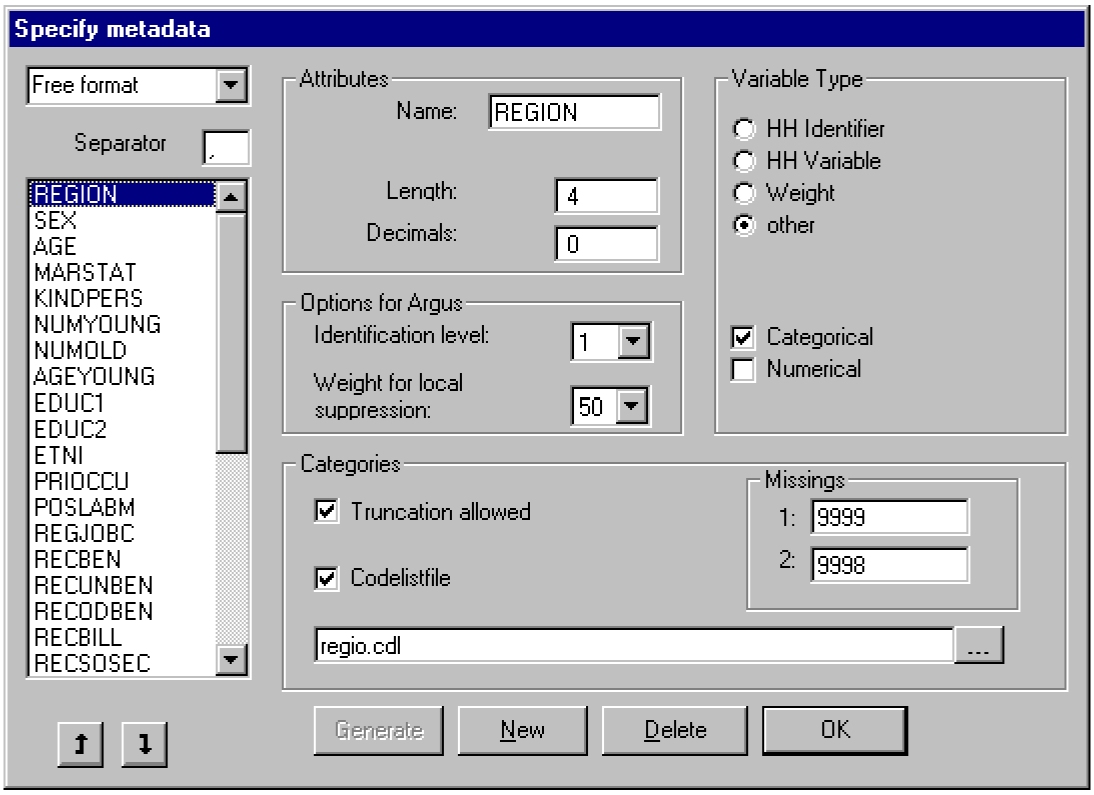
Attributes

If under File|Open microdata an 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 a default "new" is substituted. Apart from defining a new variable, an existing one can be modified or deleted.

Fixed format data file

The following attributes can be specified: name of the variable, its first position in the data file, its length and the number of decimal places. Furthermore the kind of variable can be specified: weight variable, household variable, household identifier, or none of these. A weight variable specifies the weight of the record, and is based on the (post)sampling design used. A household variable is one that yields the same score for individuals belonging to the same household. A household identifier uniquely identifies households represented in the file.

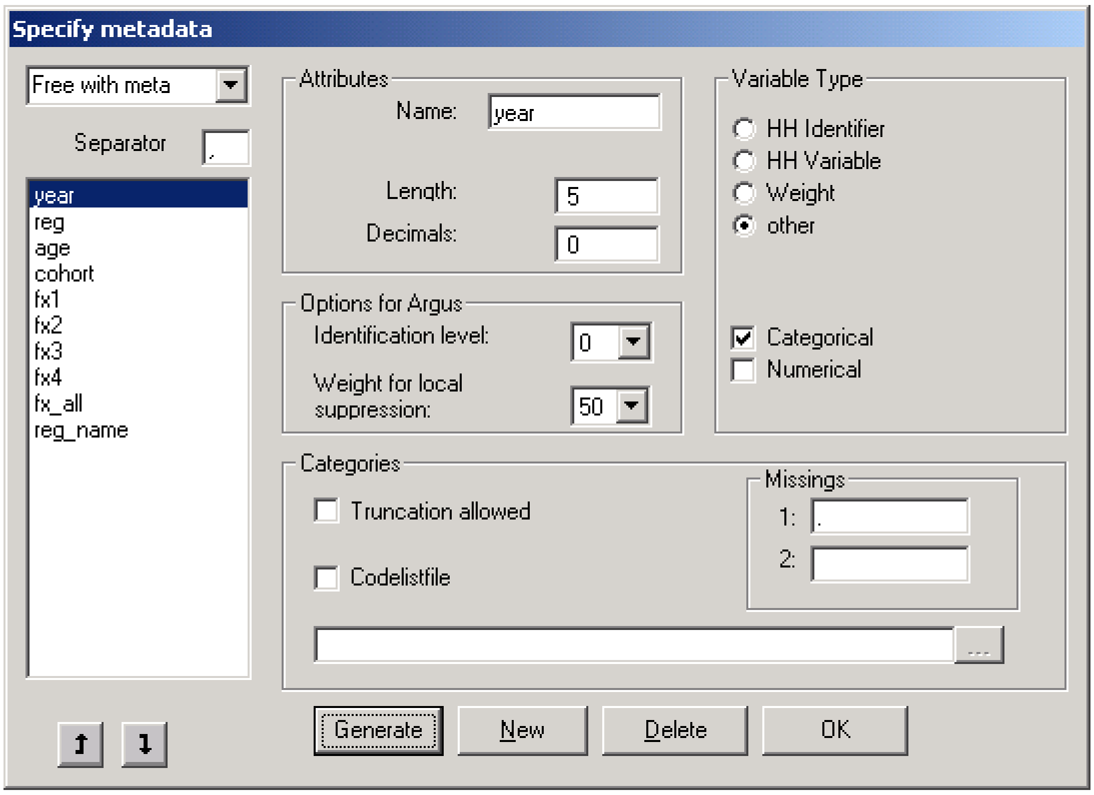
Free format data file



The following attributes can be specified: name of the variable, the character used as a separator between different data items, the length of the variable, and the number of decimal places. Furthermore the kind of variable can be specified: weight variable, household variable, household identifier, or none of these. A weight variable specifies the weight of the record, and is based on the (post)sampling design used. A household variable is one that yields the same score for individuals belonging to the same household. A household identifier uniquely identifies households represented in the file.

The 'Up' and 'Down' buttons can be used to change the order of the variables. Different from the fixed format record description the order of specification is necessarily the order in which the variables are stored in the free-format datafile.

FreeWithMeta format datafile


Specify metadata

Free with meta

Separator ,

year
reg
age
cohort
fx1
fx2
fx3
fx4
fx_all
reg_name

Attributes
Name: year
Length: 5
Decimals: 0

Options for Argus
Identification level: 0
Weight for local suppression: 50

Variable Type
HH Identifier
HH Variable
Weight
[✓ other]

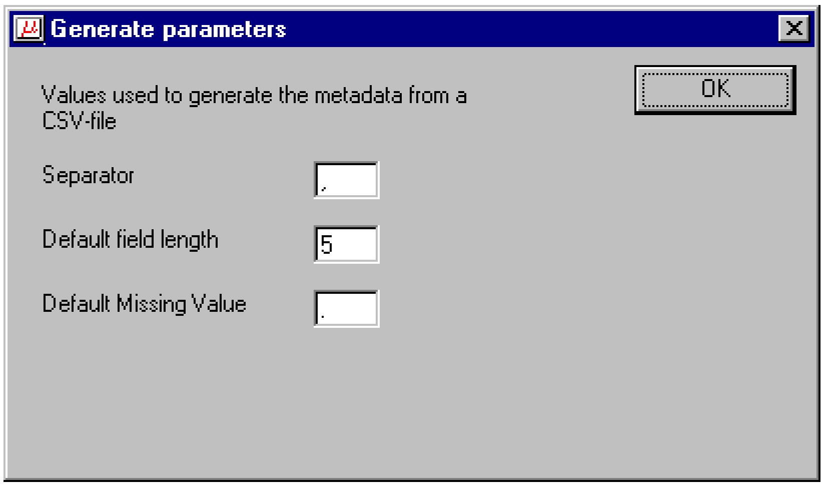
[✓ Categorical]
Numerical

Categories
Truncation allowed
Codelistfile

Missings
1: .
2:

The free format with meta format is a special variant of a free format data file. E,g, SAS users can easily generate this format. The difference with the normal free format datafile is that the first record contains the names of all the variables.

The following attributes can be specified: name of the variable, the character used as a separator between different data items, the length of the variable, and the number of decimal places. Furthermore the kind of variable can be specified: weight variable, household variable, household identifier, or none of these. A weight variable specifies the weight of the record, and is based on the (post)sampling design used. A household variable is one that yields the same score for individuals belonging to the same household. A household identifier uniquely identifies households represented in the file.


Generate parameters

Values used to generate the metadata from a CSV-file

Separator ,

Default field length 5

Default Missing Value .

If this input format is used, a first version of the metadata file can be generated automatically by μ-ARGUS by pressing the Generate button (in the Specify metadata window) after the data have been read into the program.

Options for ARGUS

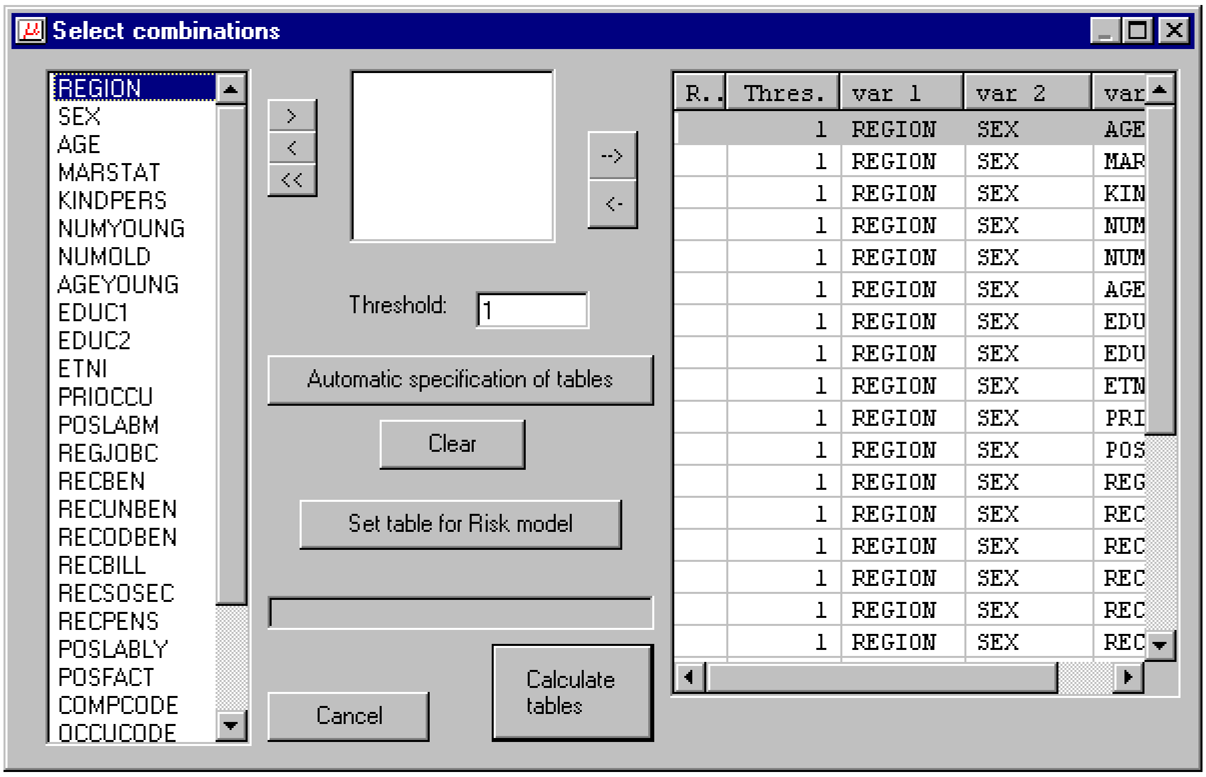
The identification level of the variable can also be specified. This level is used to let μ–ARGUS generate combinations of variables that should be checked. Identification level 0 means that the variable is not identifying, implying that it does not appear in any of the combinations to be checked. Identification level 1 is the highest level of identifiability, followed by levels 2, 3, etc. (Identification levels ≥ 4 leads to very large sets of combinations to be checked). If a variable is identifiable at level i, it is also identifiable at level i+1. In other words the identification levels (for i>0) are nested. Also the suppression priority (weight for local suppression) can be specified here already. It will play a role when the safe datafile is created at the end of a μ-ARGUS-run.

Codelist

The codelist of a variable should be known to μ–ARGUS before it can operate properly. There are two options for obtaining the codelist for each variable. In the first place they can be specified by the user, by listing them in a codelist filename. Or such a codelist (or rather the part that appears in the data) can be determined by μ–ARGUS by a run through the microdata.

Hierarchical variables can be truncated, that is, some digits of the codes can be "chopped off" and then yield a meaningful code. In this way one can replace a 'n+1' digit code by a 'n' digit code, by chopping off the least significant (n+1th) digit. A variable can have two missing values. The first one is the one that is used by μ–ARGUS when it locally suppresses a value of this variable. The second missing value is only used by μ–ARGUS to know that this value should not be used as a regular, i.e. nonmissing, value. In many surveys two missing values are used for don't know and refusal.

4.3.2 Specify|Combinations



REGION
SEX
AGE
MARSTAT
KINDPERS
NUMYOUNG
NUMOLD
AGEYOUNG
EDUC1
EDUC2
ETNI
PRIOCCU
POSLABM
REGJOBC
RECBEN
RECUNBEN
RECODBEN
RECBILL
RECSOSEC
RECPENS
POSLABLY
POSFACT
COMPCODE
OCCUCODE

Threshold: 1

R..	Thres.	var 1	var 2	var
	1	REGION	SEX	AGE
	1	REGION	SEX	MAR
	1	REGION	SEX	KIN
	1	REGION	SEX	NUM
	1	REGION	SEX	NUM
	1	REGION	SEX	AGE
	1	REGION	SEX	EDU
	1	REGION	SEX	EDU
	1	REGION	SEX	ETN
	1	REGION	SEX	PRI
	1	REGION	SEX	POS
	1	REGION	SEX	REG
	1	REGION	SEX	REC
	1	REGION	SEX	REC
	1	REGION	SEX	REC
	1	REGION	SEX	REC
	1	REGION	SEX	REC

This dialog box can be used to specify the tables that should be checked. This option can be used in one of two ways. First of all a user can interactively select variables from the listbox that should span a table. Secondly it can be used to let μ–ARGUS generate the tables. The package can do that in one of two ways:

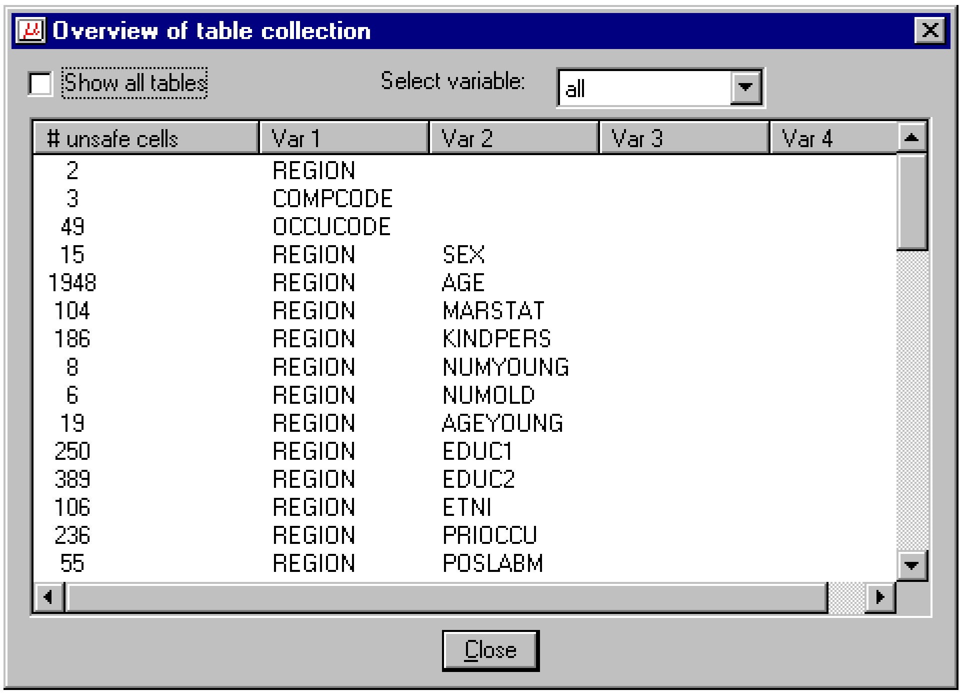
1. by generating all combinations of k variables. Only the variables with identification level >0 are used. K is to be specified by the user. For each dimension the threshold value should be specified. This option is the implementation of the Dutch approach for Public Use Files (PUF).
2. by using the identification levels of all variables, provided that they have been specified by the user. In this case μ–ARGUS generates all tables that can be obtained when variables from different levels of identification are chosen. In this case a single threshold should be specified. This option is the Dutch approach for Micro data files Under Contract. (MUC).

There are undo options that allows one to remove variables from a table or to remove tables that have been selected at an earlier stage.

Additionally there is the option of specifying that a table will be used for the new Individual risk approach. Because the traditional risk model and the new apprach cannot be applied to the same identifying variables, all overlapping combinations are removed,when a table is used for the individual risk model.

4.4 Modify

4.4.1 Modify|Show Tables



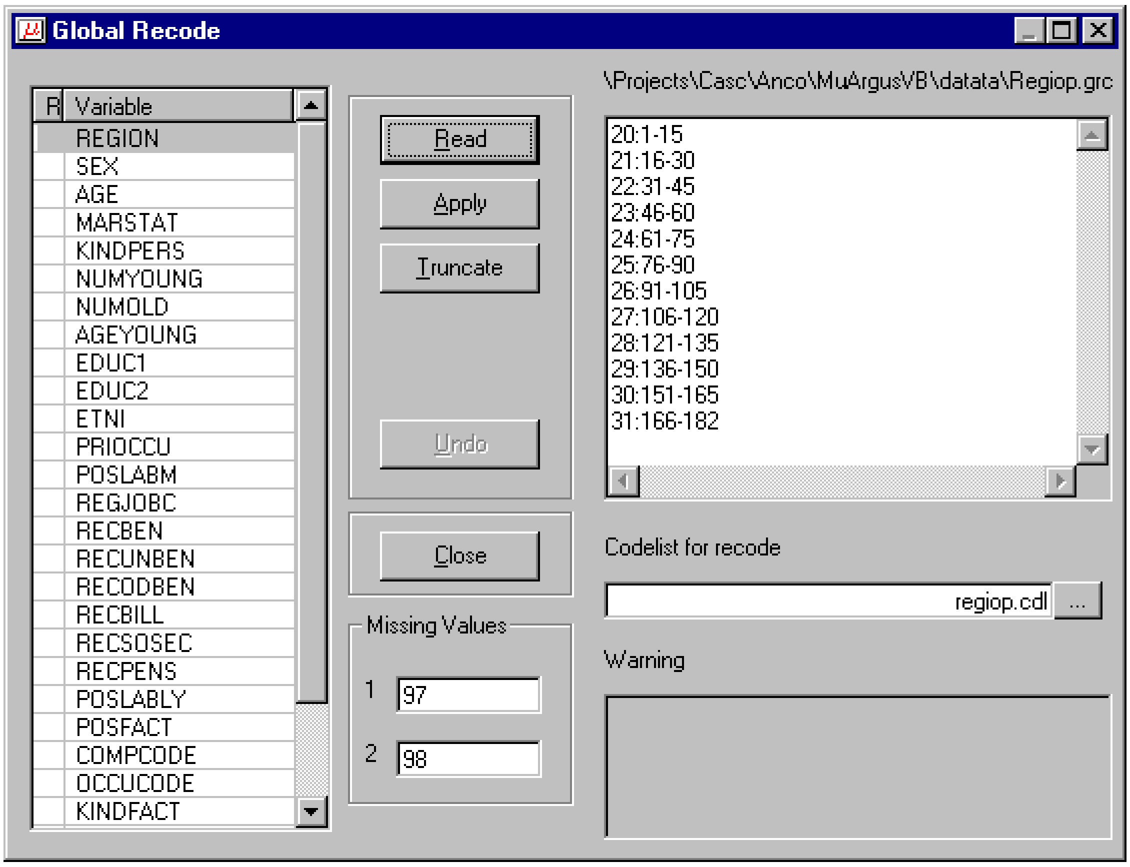
# unsafe cells	Var 1	Var 2	Var 3	Var 4
2	REGION			
3	COMPCODE			
49	OCCUCODE			
15	REGION	SEX		
1948	REGION	AGE		
104	REGION	MARSTAT		
186	REGION	KINDPERS		
8	REGION	NUMYOUNG		
6	REGION	NUMOLD		
19	REGION	AGEYOUNG		
250	REGION	EDUC1		
389	REGION	EDUC2		
106	REGION	ETNI		
236	REGION	PRIOCCU		
55	REGION	POSLABM		

In this window you can see which tables have been generated and how many unsafe cells each table has.

- On default only the tables with unsafe cells are shown. You can see all tables when you check **Show all tables**.
- If you are only interested in one variable you could use the select variable box
- The **Close** button closes the dialog box.

4.4.2 Global Recode

Global recoding (in conjunction with local suppression) is the main option for protecting individual information in a data file.



```
20:1-15
21:16-30
22:31-45
23:46-60
24:61-75
25:76-90
26:91-105
27:106-120
28:121-135
29:136-150
30:151-165
31:166-182
```

Missing Values: 1 97, 2 98

Codelist for recode: regiop.cdl

This dialog box offers two possibilities to reduce the number of unsafe cells. These are global recode and truncate.

Short description

- The left pane shows a list of all the variables that can be recoded.
- If it is possible to truncate a variable the truncate button is active.
- With **Read** you can open existing recode files.
- If you click on the right edit box, you can easily change a recoding scheme
- With **Apply** the recoding scheme will be applied to the selected variable.
- With **Truncate** the truncation will be applied to the selected variable.
- All recoding and truncations are applied on the original categories. When a recoding, truncation or replacement is applied, a **R** or **T** is placed in front of the variable, and the variable is shown in red.
- With **Undo** you can undo a global recoding or truncation of the variable which has been selected.
- The **Close** button closes the dialog box.

Global Recode

There are some rules on how you have to specify a recode scheme. All the codelists are treated as alphanumeric codes. This means that the codelist are not restricted to numbers only. However this implies that the codes '01' and ' 1' are considered different 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 (-). 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 below 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 till 182 a possible recode is then:

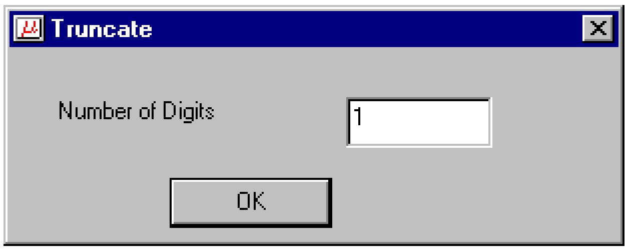
```
1:   - 49
2:   50 - 99
3:   100 - 149
4:   150 -
```

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.

Truncate
If you click **Truncate** this dialog box will pop up.



You can specify how many digits you want to truncate. If you apply a truncation and you want to truncate another digit you have to fill in 2 digits in the popup dialog box. This is because the recodings are always applied to the original categories.

Codelist
A new codelist can be selected for the recoded variable. Note that similar to the codelists in the meta data file (.RDA), this codelist is only used for enhancing the information reported on a variable in the main-window of μ-ARGUS.

Missing values
When a new recoding scheme is selected also a new set of missing values can be specified. If these boxes are empty the original missing values will still be used.

Warning
In the warning pane μ-ARGUS reports back on a recoding that has been applied. Sometimes these are only warnings, e.g. only a part of the codelist has been recoded. Sometimes a severe error has occurred, preventing μ-ARGUS from applying a recoding scheme. Typically the syntax of a recoding scheme has been violated.

When the focus is moved to another variable, or when this window is closed, μ-ARGUS will ask you whether the recoding scheme must be saved. Note that also the changed missing codes and the name of the codelist are stored in this file. A typical extension is .GRC.

Example of a .GRC-file

```
20:1-15
21:16-30
22:31-45
23:46-60
24:61-75
25:76-90
26:91-105
27:106-120
28:121-135
29:136-150
30:151-165
31:166-182

<MISSING> 97 98
```

```
<CODELIST>    regiop.cdl
```

4.4.3 PRAM specification

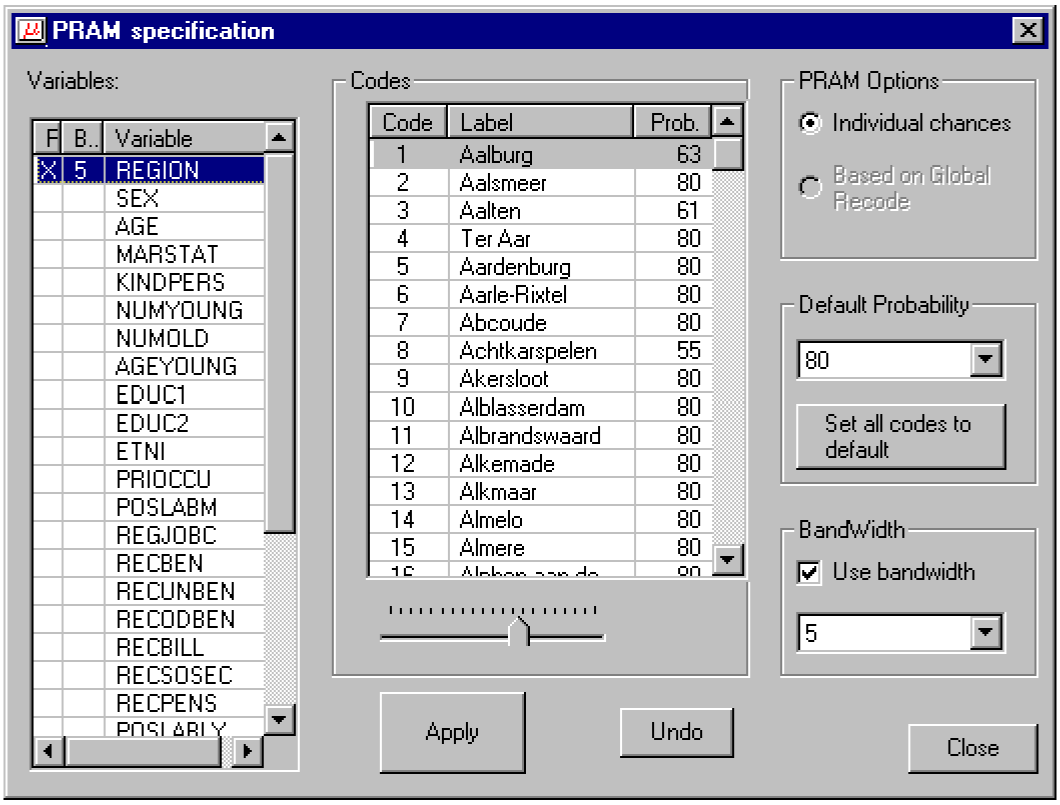
PRAM is a disclosure control technique that can be applied to categorical data. Basically, it is a form of deliberate misclassification, using a known probability mechanism. Applying PRAM means that for each record in a microdatafile, the score on one or more categorical variables is changed. See also section 2.2.4.

At this stage you can specify the probability of changing the score of a category.

First you select a variable to be PRAMmed. In the listbox in the middle you will see all the categories. The probability is the probability of not changing a score. You can set all probabilities to a certain percentage with the button 'Set all codes to default'. Further it is possible to change individual scores using the slider.

If a category will be changed, randomly another category will be selected. However if you want to restrict this, you can select a bandwidth. In that case the new category will be a neighbouring one.

Pressing the 'Apply'-button will store the information. The actual PRAMming will only be done when the new datafile is generated. It is still possible to apply global recoding and come back to re-specify your PRAM-probabilities.



F	B..	Variable
X	5	REGION
		SEX
		AGE
		MARSTAT
		KINDPERS
		NUMYOUNG
		NUMOLD
		AGEYOUNG
		EDUC1
		EDUC2
		ETNI
		PRIOCCU
		POSLABM
		REGJOBC
		RECBEN
		RECUNBEN
		RECODBEN
		RECBILL
		RECSOSEC
		RECPENS

Code	Label	Prob.
1	Aalburg	63
2	Aalsmeer	80
3	Aalten	61
4	Ter Aar	80
5	Aardenburg	80
6	Aarle-Rixtel	80
7	Abcoude	80
8	Achtkarspelen	55
9	Akersloot	80
10	Alblasserdam	80
11	Albrandswaard	80
12	Alkemade	80
13	Alkmaar	80
14	Almelo	80
15	Almere	80

[✓ Individual chances] Based on Global Recode

Default Probability: 80

[✓ Use bandwidth]

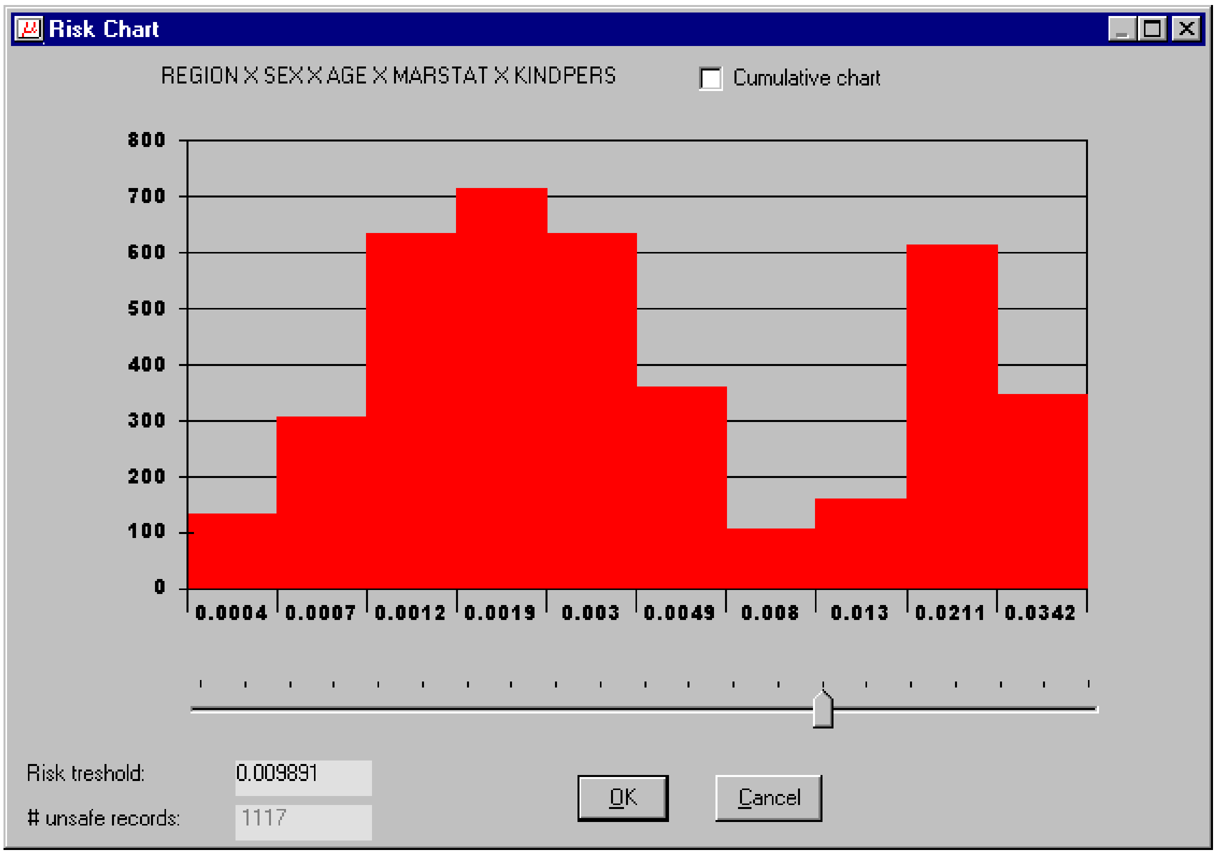
BandWidth: 5

4.4.4 Individual risk specification

A new risk model has been incorporated in μ-ARGUS. This method is based on the work by Luisa Franconi and her colleagues at Istat. A full description can be found in section 2.3 and 2.4. Here you can specify the threshold level for the risk-model. All records above the chosen risk-level are considered unsafe. When generating a safe file in the final stage of μ-ARGUS the combination of variables on which the risk model is based is considered unsafe and a missing value will be imputed.

Use the slider to set a risk-level. You will see at the same time the number of records that will be considered unsafe, when you fix this risk level.

When you have fixed a risk-level, you can still go back, select an other global recoding and inspect the risk-chart again.



Risk treshold: 0.009891

unsafe records: 1117

4.4.5 Modify numerical variables

Several modifications to a numerical variable are possible.

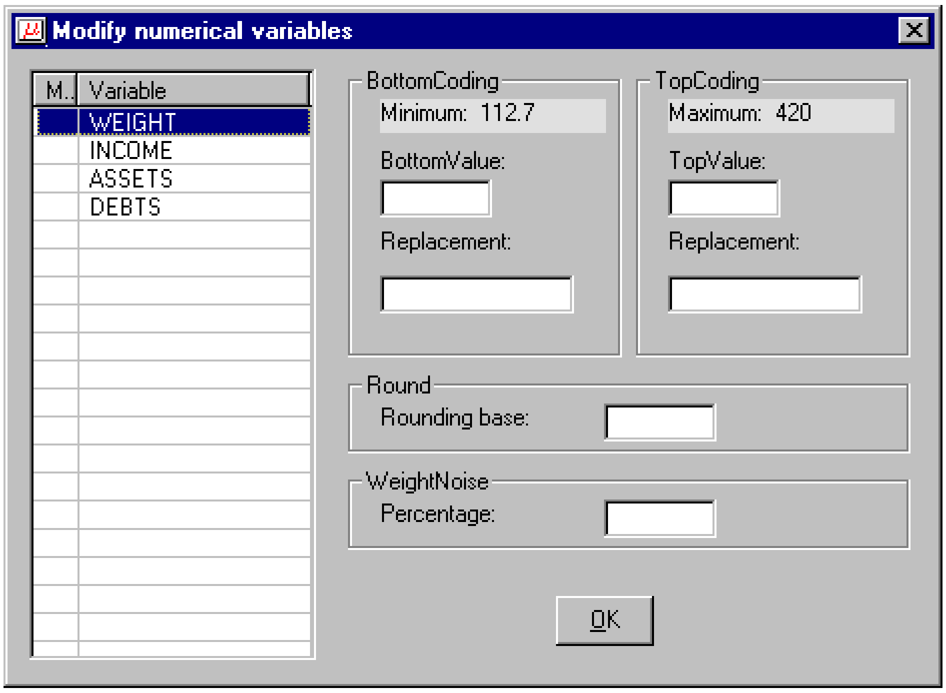
- Top/Bottom coding
- Rounding
- Add noise to the weight variable

Top and bottom coding is a technique to protect the extreme values of a distribution. Typically it is applied to numerical variables. All values above respectively below a certain threshold value will be replaced by another value. In this stage of using μ–ARGUS only the instructions will be stored. The actual top/bottom coding will be done when the safe microdata file is written.

In this window μ-ARGUS shows the real minimum and maximum of a variable. You can specify a value above/below which all values are replaced by a specified replacement value. μ-ARGUS is completely flexible what replacement you specify. The replacement value can also be alphanumeric. μ-ARGUS only prevents you from entering a top/bottom value above/below the extreme value.

Rounding can be applied to a numerical variable. It will increase the protection of the datafile, however there are no real measures to calculate the increase in protection.

Weight/Noise. As the weight variable might in some cases be identifying, it might e.g. disclose the municipality, adding noise can be a solution. At this stage you can specify the amount of noise to be added as a percentage.



Minimum: 112.7

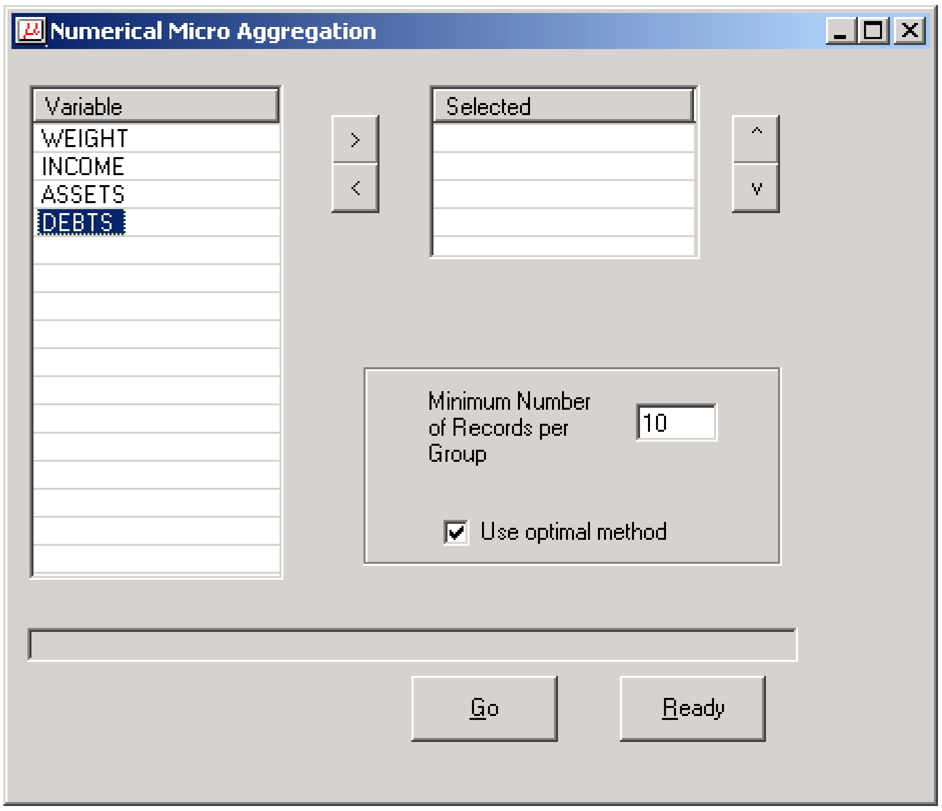
Maximum: 420

4.4.6 Numerical Micro Aggregation

Numerical Micro Aggregation can be applied to numerical variables only. The user can specify whether the optimal method should be used or not. However, the optimal method is not available for numerical micro-aggregation of a single variable.

The minimum number of records per group can also be specified

To run the program click on the 'Go' button. The 'Ready' button will take the user back to the main menu.



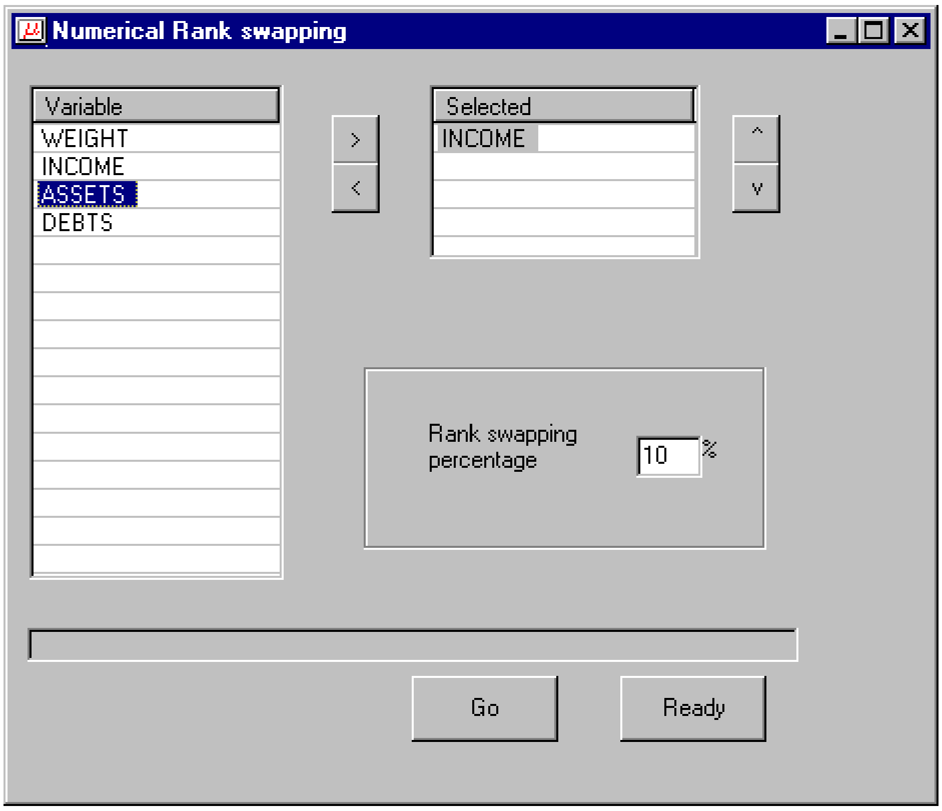
Minimum Number of Records per Group 10

[✓] Use optimal method

4.4.7 Numerical Rank Swapping

The user choices here are to select the numerical variables required for rank swapping and choose the rank swapping percentage. If 5% is selected and there are 1000 rows, then a specific record will be swapped with records at a maximum rank distance of 50.

The 'Go' Button performs the operation and the 'Ready' button takes the user back to the main menu.



Rank swapping percentage 10 %

4.4.8 Sullivan Masking

This facility is included in the software for use by those with expert knowledge of Sullivan masking. Further guidance on its use can be obtained from the CASC web site. A runtime version of GAUSS, which is required for the Sullivan Masking approach, can also be downloaded from the web site. The window displaying the input required for calling Sullivan Masking is presented here. A description of the program messages is provided in Section 3.2.7.

Sullivan masking can only be applied to numerical variables.. The options for the user are

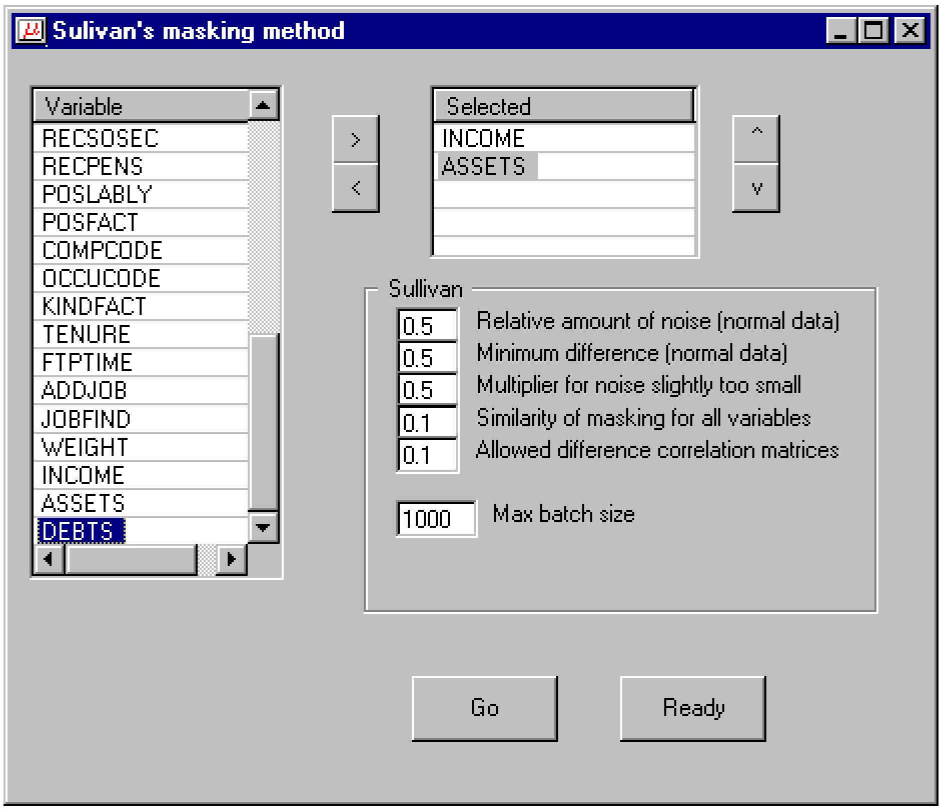
- Relative amount of noise
- Minimum difference
- Multiplier for noise slightly too small
- Similarity of masking for all variables
- Allowed difference correlation matrices

- Maximum batch size.

Please note that there are restrictions on the selected variables. These restrictions are implied by the Sullivan-methodology.

- o The number of continuous variables should be at least 2
- o The number of categories in the categorical variables should be less than the total number of variables involved in this Sullivan-run.

The ‘Go’ Button performs the operation and the ‘Ready’ button takes the user back to the main menu.



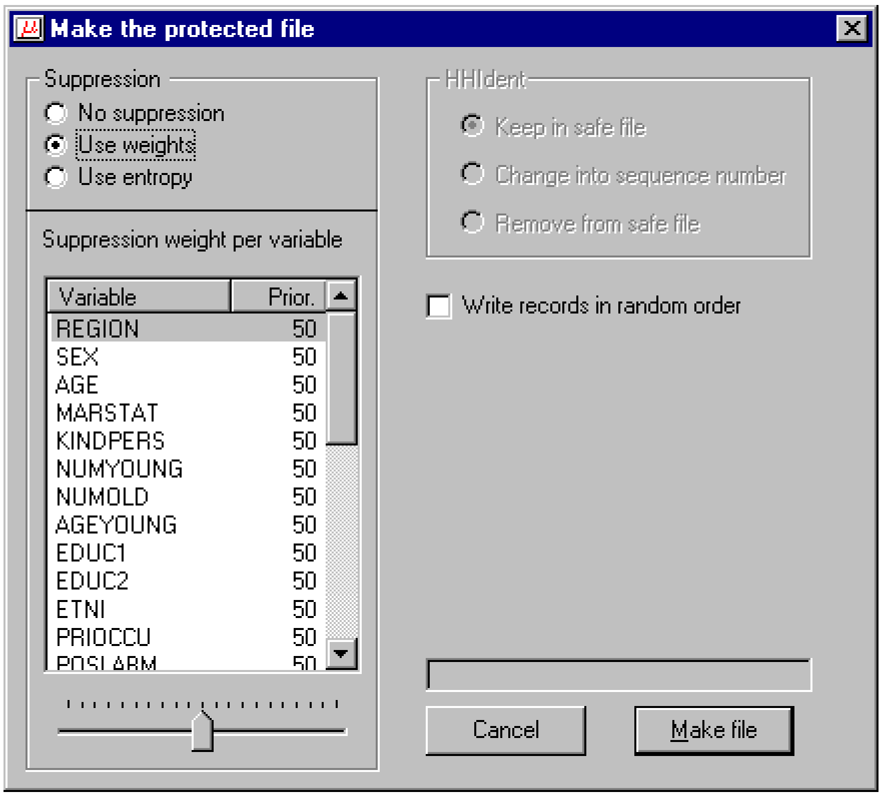
Sullivan

0.5	Relative amount of noise (normal data)
0.5	Minimum difference (normal data)
0.5	Multiplier for noise slightly too small
0.1	Similarity of masking for all variables
0.1	Allowed difference correlation matrices
1000	Max batch size

4.5 Output

4.5.1 Output|Make Suppressed File

When finally all the data modifications have been specified, Global recoding, Risk specification, PRAM-parameters, Top/bottom coding, rounding, noise added to the weight variable, it is time to write a protected file.


Make the protected file

Suppression
- ○ No suppression
- [✓ Use weights]
- ○ Use entropy

Suppression weight per variable

Variable	Prior.
REGION	50
SEX	50
AGE	50
MARSTAT	50
KINDPERS	50
NUMYOUNG	50
NUMOLD	50
AGEYOUNG	50
EDUC1	50
EDUC2	50
ETNI	50
PRIOCCU	50
POSLARM	50

HHIdent
- [✓ Keep in safe file]
- ○ Change into sequence number
- ○ Remove from safe file

☐ Write records in random order

When actually writing the safe file, the data manipulations are applied and the remaining unsafe combinations are protected by local suppression. I.e. certain values are replaced by the first missing value. Which variable(s) are selected is a small optimisation problem. If there is only one unsafe combination the variable with the lowest information loss is chosen. To calculate the information loss there are two options. Either you select the 'use weights' and you are free to assign an information loss (suppression weight) to each variable. The variable with the lowest information loss is then suppressed. The alternative option is to use an entropy function. The variable with the lowest value of the entropy function will then be suppressed. This entropy H(x) is defined as

$$H(x) = -\frac{1}{N}\sum_{x\in X} f(x)\log_2\frac{f(x)}{N}$$

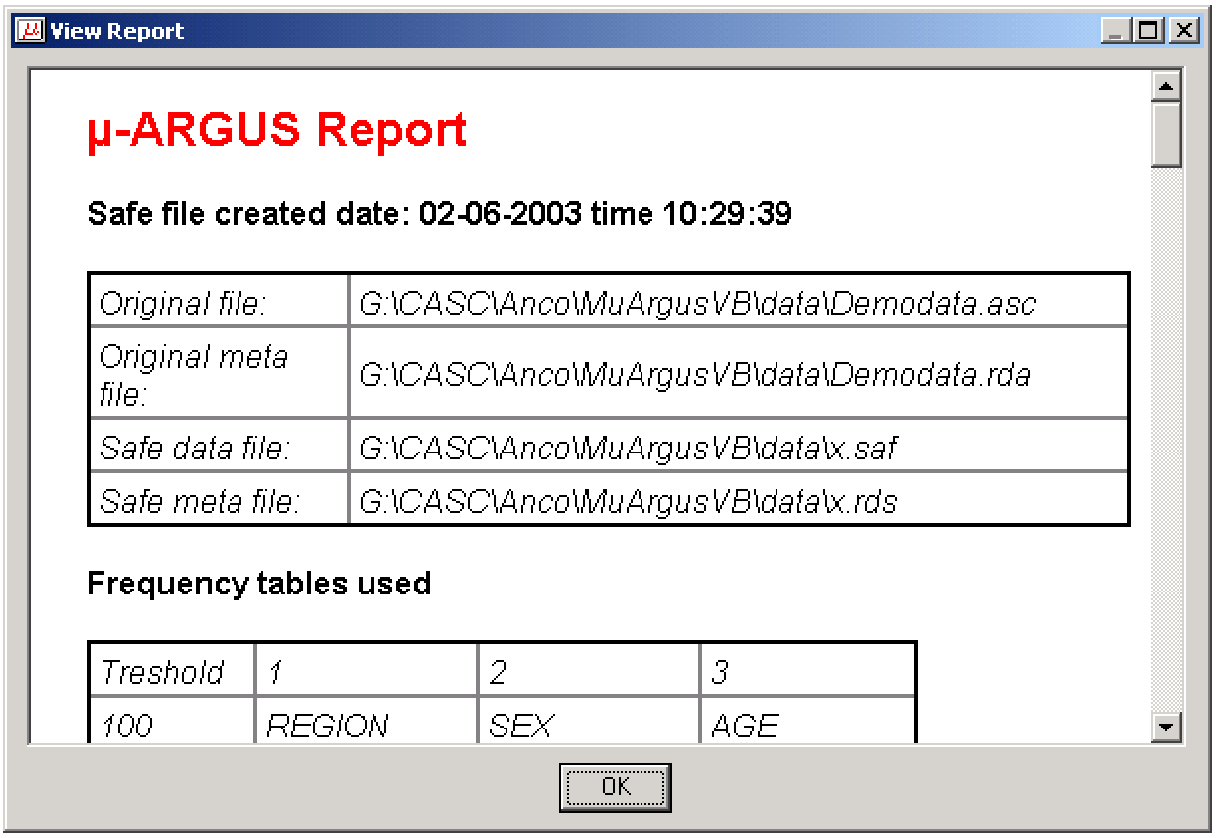
where f(x) is the frequency of category x of variable X and N the total number of records.
In the case of more than one unsafe combination in a record a set of variables must be suppressed. This set is chosen in such a way that all unsafe combinations are protected, but the total information loss is being minimised. Note that if a household variable need to be suppressed in a record it is suppressed in all records of the same household.
If there is a household identifier, there are three possibilities

- Do not change
- Change the Household identifier into a simple sequence number
- Remove the Household identifier completely from the datarecord.

Finally there is an option to write the records in a random order. This could be done to prevent easy linking to other data files.
Pressing the 'make file' button will start the writing process. First you will have the opportunity to specify the name of the output file. Automatically a report file will be written at the same time. The name of the reportfile is the same as the outputfile, but with the extension .HTML. This HTML-format allows you to inspect the report later with a browser. When the datafile is ready, however, µ-ARGUS will show the report directly.
In any case you will now have a 'safe'file, safe according to the modifications that you have applied.

4.5.2 Output|View Report

Views the report file which has been generated with Output|Make suppressed file



µ-ARGUS Report

Safe file created date: 02-06-2003 time 10:29:39

Original file:	G:\CASC\Anco\MuArgusVB\data\Demodata.asc
Original meta file:	G:\CASC\Anco\MuArgusVB\data\Demodata.rda
Safe data file:	G:\CASC\Anco\MuArgusVB\data\x.saf
Safe meta file:	G:\CASC\Anco\MuArgusVB\data\x.rds

Frequency tables used

Treshold	1	2	3
100	REGION	SEX	AGE

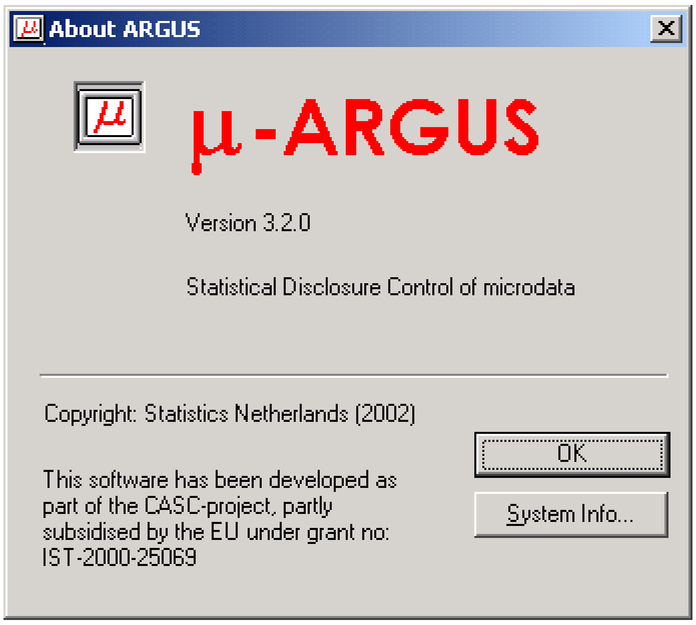
4.6 Help

4.6.1 Help|Contents

Shows the content page of the help file. This program has context-sensitive help. Whenever you press the F1, you will get some information referring to the stage of the program where you are.

4.6.2 Help|About

Shows the about box.



Version 3.2.0

Statistical Disclosure Control of microdata

Copyright: Statistics Netherlands (2002)

This software has been developed as part of the CASC-project, partly subsidised by the EU under grant no: IST-2000-25069