Journal of Official Statistics, Vol. 14, No. 4, 1998, pp. 421–435

Optimal Local Suppression in Microdata

A.G. de Waal and L.C.R.J. Willenborg[1]

In this article we assume that a safe microdata set has to be produced by a statistical office, for release to external researchers. To check the safety of such a microdata set, we assume that the statistical office checks the frequency of certain combinations of values. If a combination occurs frequently enough in the file, it is considered safe, otherwise unsafe. Unsafe combinations can be eliminated from the file by using techniques such as global recoding (= combining several categories of a variable into a single one) and local suppression (= replacing the value of a variable in a record by a missing value). In practice one first applies global recodings interactively to reduce the initial number of unsafe combinations drastically. Possible remaining unsafe combinations in the microdata set are then eliminated automatically through the application of local suppressions. The present article concentrates on this second step, i.e., the elimination of unsafe combinations by local suppressions, in an optimal way. In particular several optimal local suppression models are formulated and studied. The aim of these models is to apply local suppression in an optimal way, under various constraints. All these local suppression models turn out to be set-covering problems.

Key words: Statistical disclosure control; microdata; local suppression; integer programming; set-covering.

1. Introduction

In former days statistical offices used to publish only macrodata, i.e., tables. This was sufficient to satisfy the demands of the users of statistical data. Nowadays, however, the users of statistical data want to have data that are as detailed as possible. Not only do they want more detailed tables, but they also want to have microdata, i.e., data for individual respondents. This is mainly due to the increased power of modern computers, which enables users of statistical data to analyze these microdata by themselves. As a consequence of the demand for microdata statistical offices are put in a difficult position. On the one hand it is their duty to satisfy this demand, on the other hand they should protect the privacy of their respondents.

To achieve both aims, i.e., to satisfy the demand for microdata while protecting the privacy of the individual respondents, statistical offices apply certain protection measures. The measures are applied only when the privacy of some respondents is endangered. Information that is deemed safe is not protected in order to release as much information as possible. Examples of protection measures are *global recoding*, where several categories of a variable are combined into a single one, and *local suppression*, where a value of a

[1] Division Research and Development, Department of Statistical Methods, Statistics Netherlands, P.O. Box 4000, 2270 JM VOORBURG, The Netherlands, <twal@cbs.nl and lwlg@cbs.nl>.
Acknowledgments: The views expressed in this article are those of the authors and do not necessarily reflect the policies of Statistics Netherlands.

variable in a record is replaced by ''missing.'' In this article we explain how to determine the minimum number of local suppressions such that the resulting microdata set is considered safe.

The remainder of this article is organized as follows. In Sections 2 and 3 the approach that has been adopted by Statistics Netherlands to protect its microdata sets against statistical disclosure is sketched. In particular, a rule based on checking the frequencies of certain combinations of values of variables is described in Section 2. In Section 3 two statistical disclosure control (SDC) techniques to protect microdata sets are described in some detail, i.e., global recoding and local suppression. Both techniques are extensively used at Statistics Netherlands. More information on the view of Statistics Netherlands with respect to the protection of microdata sets can be found in De Waal and Willenborg (1996) and Willenborg and De Waal (1996). For more information on SDC in general we refer to Duncan and Lambert (1986, 1989), Fienberg (1994), Lambert (1993), Marsh, Dale and Skinner (1994), Paass (1988), and Skinner, Marsh, Openshaw, and Wymer (1994).

The problem of minimizing the number of local suppressions while protecting a microdata set is described in Section 4. This problem is a special case of a general mathematical problem. The general problem is stated in Section 5. A similar problem, namely the problem of minimizing the number of different categories that are affected by local suppressions, i.e., that are suppressed in at least one record, is discussed in Section 6. Special solutions to the problems in Sections 5 and 6 are examined in Section 7. In Section 8 solution methods for the problems considered are examined. Finally, Section 9 concludes this article with a short discussion of its main findings and some suggestions for future research.

2. Statistical Disclosure Control for Microdata

To protect the privacy of respondents a statistical office should prevent the disclosure of sensitive information on individual respondents by an intruder. As disclosure of sensitive information is possible only after an individual has been identified, a statistical office usually tries to prevent the identification of individual respondents.

Identification of individual respondents can occur when values of several so-called identifying variables are taken into consideration. The distinction between identifying variables and non-identifying variables is not clear, and has to be made on the basis of a personal judgement. Identifying variables, intuitively, concern characteristics of individuals that can be known by other people, and that could be used to track somebody down. An example of a variable that might qualify as non-identifying is ''The party for which you voted when you voted for the first time.'' Examples of identifying variables are ''Age,'' ''Sex,'' ''Domicile,'' and ''Occupation.'' The values of identifying variables can be assumed known to friends and acquaintances of a respondent. Although each identifying variable is generally not sufficient to identify an individual when considered separately, a combination of identifying variables might be sufficient. When a combination of values of identifying variables is unique, i.e., occurs only once in the population, then an intruder might identify the corresponding individual. For example, the combination ''Age = 20,'' ''Sex = Female,'' ''Domicile = Amsterdam,'' and ''Occupation = Miner'' is very likely a unique combination (if it is not an error!). So, an intruder may identify this individual, i.e., he or she may be able to determine the name of this individual. Subsequently, the intruder may be able to use the released microdata set to disclose sensitive information about this respondent.

In practice, however, it is a bad idea to prevent only the occurrence of respondents in the microdata set who are unique in the population (with respect to a certain combination of values). Firstly, because unicity in the population, in contrast to unicity in the microdata set, is hard to establish. Secondly, because an intruder may look at other combinations of values than the statistical office does. For these reasons it is better to avoid the occurrence of combinations of values in the microdata set that are rare in the population, instead of trying to avoid only the population-uniques in the microdata set.

To prevent the occurrence of rare combinations of values in the microdata set, SDC rules at Statistics Netherlands prescribe which combinations of values of identifying variables have to be checked before a microdata set can be disseminated. Moreover, the rules also prescribe how many times these combinations have to occur in order to be considered safe for release, i.e., the rules prescribe certain threshold values. For instance, the SDC rules may describe that the bivariate combinations ''Occupation × Statistician'' × ''Sex = Male'' and ''Occupation = Statistician'' × ''Nationality = non-Dutch'' each should occur at least 20 times in the microdata set to be considered safe. In case the frequency of a particular combination is at least the prescribed threshold value then this combination is considered safe. Otherwise the combination is considered unsafe and disclosure limitation measures should be applied. The threshold values may depend on the combination that has to be checked. For example, the threshold value of a bivariate combination may be different from the threshold value of a trivariate combination. For a discussion of this approach see Pannekoek and de Waal (1998) and Zaslavsky and Horton (1998).

3. Global Recoding and Local Suppression

To safeguard a microdata set against statistical disclosure two techniques are often applied at Statistics Netherlands, namely *global recoding* and *local suppression*. In the present section we discuss the meaning and application of these techniques to produce safe microdata. In case of the procedures applied at Statistics Netherlands this amounts to ''removal'' of unsafe combinations. In the present section we explain how this removal is to be interpreted in case of global recoding and in case of local suppression. Of course, the elimination of unsafe combinations should cause as little information loss (suitably quantified in one way or another) as possible. In Subsection 3.1 global recoding is considered and in Subsection 3.2 local suppression. Subsection 3.3 discusses both techniques, in particular if they are used simultaneously to eliminate unsafe combinations from a microdata file.

3.1. Global recoding

In case of global recoding several categories of a variable V are collapsed into a single one. In the corresponding microdata file the values of V appearing in the respective records are replaced by the new codes. A global recode is applied to the entire data set, not only to the unsafe part of the data set. This is done to obtain a uniform categorization of each variable. As an example consider the variable ''Age'' which can take the values 0, 1, 2, …, 99, each of which stands for the age of a person in years. Suppose ''Age'' is recoded by collapsing the original values into 10-year classes, i.e., by collapsing the original values 0, 1, …, 9 into a single category 0–9, the original values 10, 11, …, 19 into a single category 10–19, et cetera. Then each original value of the variable ''Age'' in the microdata set is replaced by the corresponding 10-year class. For instance, for each record in which the original value of ''Age'' equals 26 the new, recoded, value equals 20–29 and similarly for all other ages. The effect of globally recoding the variable ''Age'' is that less detailed information about the age of a person is released.

In the SDC package, μ-ARGUS global recoding can be applied both interactively and automatically. (Information on μ-ARGUS can be found on the web page http://www.cbs.nl/sdc.) In the former mode the user should specify which variables to recode and which categories to combine. In the latter mode, μ-ARGUS should pick a coding from a set of possible codings for the variable it wants to recode. For instance, for the variable ''Region'' the possible codings could be municipalities (in the original file), regions within provinces, provinces, and clusters of provinces. These alternative codings have been prepared prior to the SDC process, for each identifying variable present in the microdata set. Advantages of this way of global recoding, compared to the general case, is that it is easier to automate, and that it allows to control the codings that will be generated. This prevents the possibility that a coding of a variable is generated that is not used in practice.

3.2. *Local suppression*

In case of local suppression the value of a variable V in a record R is replaced by a missing value. In the corresponding microdata file the original value of V in record R is replaced by the ''missing'' value. Whereas a global recode is applied to the entire data set, a local suppression is only applied to a particular value in a particular record. This means that local suppression does not require that definitions of variables need to be changed, because it does not effect the coding of any variable.

To explain the mechanics of this elimination process, consider, for example, a file of the Dutch Labor Force Survey, containing ''Region'' (municipalities) and ''Profession.'' Suppose that, among other things, one has to check the bivariate combination ''Region'' × ''Profession.'' Assume that, for instance, the combination ''Urk'' × ''mathematician'' occurs once in the file and the threshold is set to three, which means that this combination is considered unsafe. (Urk is a small village in The Netherlands.) If we locally suppress ''Urk'' in the record in which the combination appears, it has to be checked that the remaining ''combination'' ''mathematician'' appears frequently enough in the file, i.e., at least three times. If so, the remaining combination is safe (which is likely). It would also be possible to suppress the value ''mathematician'' instead of the value ''Urk''. Then it has to be checked that the combination ''Urk'' occurs frequently enough in the file. Whatever value is locally suppressed depends on the purposes one has in mind for the analyses of the protected file, and could formally be quantified in terms of information loss. The intuitive goal that one wants to pursuit is to produce a safe file with a minimum loss of information.

It should be noted that there is an important difference between local suppression in microdata sets and cell suppression in tables. If a cell value in a table is suppressed it is often necessary to suppress some additional cell values, because usually the marginal totals of the tables are published. These marginal totals allow one in many cases to compute the value of a suppressed internal cell value if no additional cell values have been suppressed (for more information on cell suppression in tables see Willenborg and De Waal, 1996). If a value in a record in a microdata set is locally suppressed, then this value cannot be computed, because there are no marginal totals. Of course, one can compute the marginal totals oneself, but the computed marginal totals will be (partly) based on the missing values. So, the computed marginal totals cannot be used to recalculate the missing values.

3.3. Discussion

Both techniques, global recoding and local suppression, can be used to eliminate unsafe combinations, each in its own way. These techniques can be used separately or in combination. Both techniques cause a certain loss of information in the data file to which they are applied. Moreover, local suppression may induce biased estimates when it is being dealt with in a straightforward (but naive) way, such as ignoring the missing values. For that reason it should be applied only on a relatively small scale. Global recoding is the preferable technique when a large number of unsafe combinations have to be eliminated (and hence the number of required local suppressions would be large.) On the other hand, local suppressions may be preferable to global recodings when much information would be lost due to these global recodings. When the number of local suppressions is small the bias introduced in estimates is often negligible, whereas the information loss that would have occurred when the microdata set should have been protected by global recodings only would have been substantial. In practice the right balance has to be found between applying global recodings on the one hand and local suppressions on the other.

At Statistics Netherlands first some variables are globally recoded and subsequently the remaining unsafe records are protected by locally suppressing some values. The package μ-ARGUS is designed to carry out these tasks smoothly. In this article we assume that the approach outlined above is used, i.e., that the global recodings have already been carried out. Only the local suppressions remain to be determined. In the remainder of this article we explain how the number of local suppressions can be minimized.

This approach can in turn be used as a stepping stone to a more comprehensive – and in practice more useful – model in which the optimum mix of global recodings and local suppressions to eliminate a given set of unsafe combinations has to be calculated (cf. Hurkens and Tiourine, 1998a, b.) Such a model can, in turn, be seen as a precursor to a model in which the unsafety definition of microdata on the basis of a frequency criterion for certain combinations of values is replaced by a probabilistic model that yields the disclosure risk for each record in a file (cf. De Waal and Willenborg, 1996). The aim is then again to modify an unsafe microdata set by global recoding and local suppression, in such a way that a safe one is obtained, with minimum information loss. A microdata file is then considered safe if the disclosure risk for each record is below a given threshold value.

4. Optimal Local Suppression

The easiest way to determine which variable values should be locally suppressed would be to do this for each combination that has to be checked and for each record separately. Basically there are two ways of doing this. Firstly, a value may be set to ''missing'' immediately after a certain unsafe combination is identified. The resulting microdata set, with missings due to local suppression, is then used to determine whether or not other combinations, possibly in other records, are safe. Secondly, the original microdata set may be used to determine whether or not a certain combination is safe. However, both approaches lead to some practical problems.

Suppose that a value is set to ''missing'' immediately after a certain unsafe combination is identified and that the resulting microdata set, with missings due to local suppression, is used to determine whether or not other combinations, possibly in other records, are safe. The problem with this sequential approach is that some combinations may not incorrectly seem to appear frequently enough. For example, if we suppress the value ''Statistician'' in the combination ''Occupation = Statistician'' × ''Nationality = non-Dutch,'' then this may have the consequence that later on the combination ''Occupation = Statistician'' × ''Sex = Male'' might not seem to occur frequently enough. This combination would therefore be considered unsafe. However, this combination could occur frequently enough in the original microdata set. In that case it should in fact be considered safe.

Alternatively, the original microdata set can be used to determine whether or not a combination is safe. This approach may also lead to problems. Suppose, for instance, that the combination ''Occupation = Statistician'' × ''Nationality = non-Dutch'' does not occur frequently enough in the (original) file and that we decide to suppress the value of ''Nationality'', i.e., ''non-Dutch'', in each record in which this combination occurs. Suppose furthermore that the combination ''Occupation = Statistician'' × ''Sex = Female'' also does not occur frequently enough and in this case we decide to suppress the value ''Female.'' Then it is likely that we suppress too much. In case there would be records of non-Dutch female statisticians in the microdata set then it would have been better if we had suppressed the single value ''Statistician'' for these persons instead of the two values ''non-Dutch'' and ''Female.'' Here we make the assumption that the combinations obtained after suppressing the value ''Statistician,'' i.e., ''Nationality = non-Dutch'' and ''Sex = Female,'' occur frequently enough in the population. Whether this assumption is correct has to be checked, of course.

We conclude that we cannot decide for each unsafe combination and record *separately* which values should be suppressed if we want to minimize the number of local suppressions. We have to decide which values have to be suppressed for all the unsafe combinations and records *simultaneously*. In Section 5 we examine how the resulting problem can be formally stated as a 0–1 integer programming problem.

5. Models for Optimal Local Suppression

In this section we formulate ''local suppression in a microdata file'' as a general 0–1 integer programming problem. We show that the problem discussed in Section 4 is a special case of this general problem. We begin by defining the term ''minimum unsafe combination,'' or MINUC. These MINUCs will play a crucial role in the remainder of this article.

To fix our minds we suppose that the applicable SDC rules required that it is necessary to check whether certain trivariate combinations of values of identifying variables occur frequently enough. The fact that we are considering combinations of three values of identifying variables is not really restrictive. The number three could be replaced by any other number without affecting the essence of the method. In the sequel we only consider the identifying variables of the microdata set because our measures to protect a microdata set involve only such variables. In other words, whenever we refer to (a category/value of) a variable we mean (a category/value of) an identifying variable.

We start by checking all the univariates. In case a value of a variable is considered safe we check the bivariate combinations in which this value occurs. In case a value of a variable does not occur frequently enough, e.g., ''Occupation = Mayor,'' we do not check the bivariate combinations involving ''Occupation = Mayor,'' e.g., ''Occupation = Mayor'' × ''Sex = Female.'' Next we check the trivariate combinations in which only safe bivariate combinations occur. After checking the required trivariate combinations we are able to list for all the records the minimum unsafe univariate, bivariate and trivariate combinations.

A consequence of this way of constructing the MINUCs is that whenever we suppress a value in a minimum unsafe n-variate combination the resulting $(n-1)$-variate combination will be safe. Moreover, when in each MINUC of a record at least one value is suppressed then this record is considered safe. This property of the MINUCs makes it easy to find the minimum number of local suppressions.

Suppose we need to suppress some variable values in some records. For each value j in a MINUC in record i we introduce a dummy variable y_{ij}. This dummy variable is equal to 0 if value j in record i is not suppressed or if value j does not occur in record i; otherwise it is equal to 1. For each MINUC and for each record we have the constraint stating that at least one value of a MINUC in a record must be suppressed. In other words, the sum of the y_{ij}'s of the corresponding values is at least 1. As we have remarked before this constraint is necessary and sufficient in order to make this combination safe. As a target function we use a weighted sum of the y_{ij}'s.

In mathematical terms we consider the following 0–1 integer programming problem. Let the total number of unsafe records be denoted by I and the total number of different values involved in the MINUCs by J. After renumbering the records and the variables the dummy variables y_{ij} $(i = 1, \ldots, I; j = 1, \ldots, J)$ must satisfy

$$y_{ij} = \begin{cases} 1 & \text{if value } j \text{ in record } i \text{ is suppressed} \\ 0 & \text{if value } j \text{ in record } i \text{ is not suppressed} \\ & \text{or if value } j \text{ does not occur in record } i \end{cases} \tag{5.1}$$

Suppose there are K MINUCs in the microdata set. Let c_{jk} $(j = 1, \ldots, J, k = 1, \ldots, K)$ and d_{ik} $(i = 1, \ldots, I; k = 1, \ldots, K)$ be defined by

$$c_{jk} = \begin{cases} 1 & \text{if value } j \text{ occurs in MINUC } k \\ 0 & \text{otherwise} \end{cases} \tag{5.2}$$

and

$$d_{ik} = \begin{cases} 1 & \text{if MINUC } k \text{ occurs in record } i \\ 0 & \text{otherwise.} \end{cases} \tag{5.3}$$

The constraints of the problem are given by

$$\sum_{j=1}^{J} c_{jk}\, y_{ij} \geq d_{ik}, \qquad \text{for all } i = 1, \ldots, I, k = 1, \ldots, K \tag{5.4}$$

since we have to suppress at least one value in each MINUC.

We consider the following target function:

$$\sum_{i=1}^{I} \sum_{j=1}^{J} w_{ij}\, y_{ij} \tag{5.5}$$

where w_{ij} denotes the non-negative weight of value j in record i which needs to be specified by the user. Our problem is to minimize the target function (5.5) under the constraints given by (5.1) and (5.4).

Note that the problem discussed in Section 3 is obtained if we choose all the weights w_{ij} equal to one. Because the weights in the target function (5.5) may be arbitrary non-negative numbers the problem stated above is more general than the problem of Section 3. The weights allow one to indicate how important one considers a specific value in a specific record to be as far as local suppression is concerned.

Also note that the problem above can be decomposed into subproblems for each record separately. For each record i the target function (5.5) has to be replaced by the target function

$$\sum_{j=1}^{J} w_{ij}\, y_{ij} \qquad \text{for all } i = 1, \ldots, I \tag{5.6}$$

The constraints to be considered for this problem consist of all those given in (5.4) as far as they pertain to record i.

This subproblem for each record can sometimes be partitioned into a number of smaller subproblems. Consider the MINUCs of a particular record to be the vertices of a graph. Two MINUCs are joined by an edge if and only if they have a value in common. This graph may be disconnected. In that case it consists of several connected subgraphs that are mutually disconnected. Each subgraph corresponds to a subproblem, namely the problem of minimizing (5.6), under the constraints that the MINUCs corresponding to the vertices are made safe. Thus sometimes we will be able to reduce the original problem to a number of smaller subproblems. But also these subproblems may sometimes be reduced to still smaller problems, some of which are trivial.

This reduction follows from the observation that only the dummy variables corresponding to values that occur in more than one MINUC have to be considered. Combinations that are still unsafe after some of these values have been suppressed can be made safe by suppressing the values involved, taking the weights associated with each of the variables into account.

Thus in practice we can expect that the general optimization problem will be reduced to a number of small subproblems. This implies that it may even be feasible to try all possibilities in order to minimize the target function.

Example: Throughout the remainder of this article we illustrate the optimal solutions to the problems considered by means of an example. In this example there are eleven unsafe records, seven variables (V_1 to V_7) and 21 different values (A to U). These eleven records contain the following MINUCs:

Record 1: “$V_1 = A$” × “$V_2 = B$” and “$V_2 = B$” × “$V_3 = C$”
Record 2: “$V_1 = A$” × “$V_4 = D$” and “$V_1 = A$” × “$V_5 = E$”
Record 3: “$V_2 = F$” × “$V_3 = C$”
Record 4: “$V_1 = G$” × “$V_2 = H$” and “$V_2 = H$” × “$V_3 = I$”
Record 5: “$V_5 = J$” × “$V_6 = K$”
Record 6: “$V_5 = J$” × “$V_6 = L$”
Record 7: “$V_1 = M$” × “$V_2 = N$” and “$V_2 = N$” × “$V_3 = O$”
Record 8: “$V_1 = M$” × “$V_3 = O$”
Record 9: “$V_5 = P$” × “$V_6 = Q$” and “$V_6 = Q$” × “$V_7 = R$”
Record 10: “$V_5 = S$” × “$V_6 = T$”
Record 11: “$V_5 = S$” × “$V_7 = U$”

Thus Records 1, 2, 4, 7, and 9 each contain two unsafe combinations and the remaining records one each. Note that the records containing two unsafe combinations each have one overlapping variable/value, that is a variable value combination that appears in both MINUCs. For Record 1 this is “$V_2 = B$,” for Record 2 “$V_1 = A$,” for Record 4 “$V_2 = H$,” for Record 7 “$V_2 = N$,” and for Record 9 “$V_6 = Q$.”

Example: If target function (5.6) has weights w_{ij} all equal to one, then an optimal solution to the problem considered in this section is given by:

suppress in Record 1: “$V_2 = B$”
suppress in Record 2: “$V_1 = A$”
suppress in Record 3: “$V_2 = F$”
suppress in Record 4: “$V_2 = H$”
suppress in Record 5: “$V_5 = J$”
suppress in Record 6: “$V_5 = J$”
suppress in Record 7: “$V_2 = N$”
suppress in Record 8: “$V_3 = O$”
suppress in Record 9: “$V_6 = Q$”
suppress in Record 10: “$V_6 = T$”
suppress in Record 11: “$V_7 = U$”

Thus 11 values are locally suppressed and ten different categories are affected, i.e., suppressed in some record.

6. Minimizing the Number of Different Affected Categories

Instead of minimizing the total number of local suppressions the user of the data might want to minimize the number of different categories that are affected by the local suppressions, i.e., he or she might want to minimize the number of categories that is suppressed in at least one record. A rationale for this could be that he or she considers a category that is suppressed in some records to be unsuited, or hardly suited, for statistical analysis. In other words, affected categories are of no, or only limited, value to him or her.

We can formulate this second problem as follows. First we introduce some new dummy variables. For each category value j that occurs in a MINUC we introduce a dummy variable z_j, defined as

$$z_j = \begin{cases} 1 & \text{if category } j \text{ is affected, i.e., if category } j \text{ is} \\ & \text{suppressed in some record} \\ 0 & \text{if category } j \text{ is not affected} \end{cases} \tag{6.1}$$

Note that the z_j's are independent of the records. The following constraints have to be satisfied:

$$\sum_{j=1}^{J} c_{jk}\, z_j \geq 1 \qquad \text{for all } k = 1, \ldots, K \tag{6.2}$$

We consider the following target function:

$$\sum_{j=1}^{J} z_j \tag{6.3}$$

Target function (6.3) must be minimized under the constraints given by (6.2). The optimization problem that then arises is a set-covering problem.

This problem can be extended by replacing (6.3) with a weighted sum of the z_j's. This would enable the user to indicate how important he or she considers each category to be. Very important categories should be given a large weight; unimportant categories should be given a small weight. The resulting problem can in many cases be decomposed into a number of subproblems as described in Section 5.

For this problem records cannot be considered independently, as was the case for the problem of Section 5. However, in many cases the problem can be reduced to smaller subproblems, because the remarks made in Section 5 apply to this case as well. The problem can sometimes be further reduced to subproblems corresponding to connected subgraphs. In this case the MINUCs correspond to the vertices of a graph. Two vertices are joined by an edge if and only if the corresponding MINUCs have a value in common and both MINUCs occur simultaneously in at least one record.

Example (continued): We consider our example again. An optimal solution to the problem considered in this section is given by:

suppress in Record 1: ``$V_1 = A$'' and ``$V_3 = C$''
suppress in Record 2: ``$V_1 = A$''
suppress in Record 3: ``$V_3 = C$''
suppress in Record 4: ``$V_2 = H$''
suppress in Record 5: ``$V_5 = J$''
suppress in Record 6: ``$V_5 = J$''
suppress in Record 7: ``$V_1 = M$'' and ``$V_3 = O$''
suppress in Record 8: ``$V_3 = O$''
suppress in Record 9: ``$V_6 = Q$''
suppress in Record 10: ``$V_5 = S$''
suppress in Record 11: ``$V_5 = S$''

Thus 13 values are locally suppressed and eight different categories are affected.

7. Special Solutions

After the problems of Sections 5 and 6 have been solved there are usually a number of possible optimal solutions. Among these possible solutions a solution that is optimal with respect to some additional criterion can be chosen. In this section we consider some of these extended problems.

Suppose that the number of local suppressions has been minimized by solving the 0–1 integer programming problem of Section 5. Suppose furthermore that among these solutions we want to find the solution that affects a maximum number of different categories. As a result the local suppressions will probably spread more or less evenly over the categories.

This problem can be formalized as follows. Let the minimum number of local suppressions be denoted by N_{min}. This number is known because we assume that the problem of Section 5 has been solved. We want to use both the variables y_{ij} and the variables z_j in one problem. The variable z_j should be equal to one if and only if there is a y_{ij} equal to one for some i. This can be achieved by using a large number W and introducing the following constraints:

$$W z_j \geq \sum_{i=1}^{I} y_{ij} \qquad \text{for all } j = 1, \ldots, J \tag{7.1}$$

and

$$y_{ij} \geq z_j \qquad \text{for all } j = 1, \ldots, J \tag{7.2}$$

As we want the number of local suppressions to be minimal we have to add the following constraint:

$$\sum_{i=1}^{I} \sum_{j=1}^{J} y_{ij} = N_{min} \tag{7.3}$$

The target function we consider is given by (6.3). This target function must be maximized under the constraints given by (5.4), (7.1), (7.2), and (7.3).

Example (continued): We consider our example again. A solution to the problem stated above is given by:

suppress in Record 1: "$V_2 = B$"
suppress in Record 2: "$V_1 = A$"
suppress in Record 3: "$V_2 = F$"
suppress in Record 4: "$V_2 = H$"
suppress in Record 5: "$V_6 = K$"
suppress in Record 6: "$V_6 = L$"
suppress in Record 7: "$V_2 = N$"
suppress in Record 8: "$V_1 = M$"
suppress in Record 9: "$V_6 = Q$"
suppress in Record 10: "$V_6 = T$"
suppress in Record 11: "$V_7 = U$"

Thus 11 values are locally suppressed and 11 different categories are affected.

Similar to the problem above the user of the data might want to affect as few different categories as possible while at the same time suppressing as few values as possible.

In this case the target function (6.3) must be *minimized* under the constraints given by (5.4), (7.1), (7.2), and (7.3).

Example (continued): We consider our example again. A solution to the problem above is given by:

suppress in Record 1: ''$V_2 = B$''
suppress in Record 2: ''$V_1 = A$''
suppress in Record 3: ''$V_2 = F$''
suppress in Record 4: ''$V_2 = H$''
suppress in Record 5: ''$V_5 = J$''
suppress in Record 6: ''$V_5 = J$''
suppress in Record 7: ''$V_2 = N$''
suppress in Record 8: ''$V_1 = M$''
suppress in Record 9: ''$V_6 = Q$''
suppress in Record 10: ''$V_5 = S$''
suppress in Record 11: ''$V_5 = S$''

Thus 11 values are locally suppressed and nine different categories are affected.

The final problem we consider is the following. Suppose that the number of different categories affected has been minimized by solving the 0–1 integer programming problem of Section 6. Suppose furthermore that among these solutions we want to find a solution that suppresses a minimum number of values.

Let the minimum number of different categories affected be denoted by M_{min}. This number is known because we assume that the problem of Section 6 has been solved. We introduce the following constraint:

$$\sum_{j=1}^{J} z_j = M_{min} \tag{7.4}$$

In this case we have to minimize (5.5) with all w_{ij}'s equal to one under the constraints given by (5.4), (7.1), (7.2), and (7.4).

Note that the problem above is easy to solve once the problem of minimizing (6.3) with equal w_{ij}'s under the constraints (6.2) has been solved. On the one hand at least one value per MINUC should be suppressed. Thus the optimal value of the target function (5.5) is at least equal to the number of MINUCs. On the other hand it is sufficient to suppress only one value in a MINUC to make this combination safe. This implies that for a MINUC in which n values have been suppressed we can re-open any $(n-1)$ values, i.e., replace the missings by the original values. Thus the optimal value of the target function (5.5) is at most equal to the number of MINUCs. Combining both results we conclude that the optimal value of the target function equals the number of MINUCs, and that a solution to the above problem can be found by re-opening any $(n-1)$ values in each MINUC in which n values have been suppressed.

Example (continued): For the last time we consider our example. A solution to the problem above is given by:

suppress in Record 1: ``$V_1 = A$'' and ``$V_3 = C$''
suppress in Record 2: ``$V_1 = A$''
suppress in Record 3: ``$V_3 = C$''
suppress in Record 4: ``$V_2 = H$''
suppress in Record 5: ``$V_5 = J$''
suppress in Record 6: ``$V_5 = J$''
suppress in Record 7: ``$V_2 = N$''
suppress in Record 8: ``$V_1 = M$''
suppress in Record 9: ``$V_6 = Q$''
suppress in Record 10: ``$V_5 = S$''
suppress in Record 11: ``$V_5 = S$''

Thus 12 values are locally suppressed and eight different categories are affected.

8. Solution Methods

The problems of Sections 5, 6, and 7 can be solved by using a standard algorithm for solving 0–1 integer problems, such as a branch-and-bound algorithm (see for instance Nemhauser and Wolsey (1988)). A drawback of these standard algorithms, however, is that they are rather time-consuming. In practice one therefore often uses a heuristic to determine a suboptimal solution. Several of such heuristics are described in Van Gelderen (1995).

Van Gelderen (1995) describes two basic greedy algorithms. In the first one the value that is chosen to be suppressed at any stage is the value that makes the most MINUCs safe at that stage. In the second one a value is chosen that occurs in MINUCs of as few values as possible, e.g., when only bivariate and trivariate MINUCs occur in the problem then the bivariate MINUCs are protected first. Van Gelderen (1995) also describes an exchange procedure. After a partial solution has been generated, it is likely that values that entered the solution early in the selection process no longer contribute as much to the solution as they did when selected. That is, it may be advantageous to replace a value in the partial solution by a value not occurring in this partial solution. These three ingredients, the two basic greedy algorithms and the exchange procedure, are combined to construct six hybrid algorithms that have already been proposed by Vasko and Wilson (1986) for general set-covering problems.

Apart from greedy algorithms Van Gelderen (1995) examines a heuristic based on the sum of the frequencies of the values in each of the (remaining) MINUCs. The combination with the lowest sum of frequencies, i.e., a combination that is made up of relatively rare values, is chosen. From this combination the value that has the highest frequency, i.e., the value that simultaneously protects as many MINUCs as possible, is selected for suppression. This heuristic is also combined with the first basic greedy algorithm, yielding a kind of modified Vasko and Wilson algorithm.

Finally, Van Gelderen (1995) describes a probabilistic heuristic. At each stage of this heuristic the frequencies of the values in the remaining MINUCs are calculated. Based on these frequencies the probabilities for selecting a specific value for suppression are determined. The probability of suppressing a value is its frequency divided by the sum of the frequencies of all values. The exchange procedure described above is incorporated into this probabilistic heuristic in order not to miss the really important values.

These heuristics have been applied to the problem of minimizing the number of affected categories (see Section 6). The instances were generated randomly. Each value was considered equally important, i.e., target function (6.3) was used. For the largest problem considered, 1,000 unsafe combinations with 64 different categories, the optimal solution was determined. It turned out that 43 different categories had to be affected. It took 1,158 seconds of user CPU-time to solve this problem on a SPARC 10/31, and 3,479 seconds of user CPU-time to prove optimality when an exact algorithm was used. On the other hand, the heuristics each took less than half a second. The worst heuristic for this case, the first basic greedy algorithm, obtained a suboptimal solution of 47 different affected categories. The best heuristic for this case, the algorithm based on the sum of frequencies, obtained a supoptimal solution of 44 different affected categories.

For other, smaller, problems the solutions of the heuristics were only compared to each other, not to the optimal solution. For these smaller problems the heuristics were also combined, i.e., each heuristic of such a combination was used to solve the problem to suboptimality and the best solution found was selected. It turned out that these combinations of different heuristics yield very good results. A combination of three heuristics of the Vasko and Wilson kind, three heuristics of the modified Vasko and Wilson kind, and the probabilistic heuristic accounted for over 95% of the best solutions.

9. Discussion

Optimal local suppression procedures can be automated and used in many practical situations, although theoretically the problems are infeasible (NP-complete). This is particularly the case if a problem splits into several smaller subproblems, each of which can be solved efficiently. This is, for instance, the case if the total number of suppressed values is to be minimized, because then the problem can be solved record-wise. In case the number of different affected categories is to be minimized the problem – in practice – tends to be somewhat more difficult. However, in many cases the problem can be decomposed into a number of smaller, and therefore easier to solve, problems. Moreover, fast heuristics that give good results, i.e., with solutions close to the optimal one, have been constructed. Thus it is possible to solve this problem in many practical instances as well within a relatively short period of time. Algorithms to determine the minimum number of locally suppressed values have been implemented in μ-ARGUS.

Automating the global recodings is the next logical step to automating the local suppressions. Optimization models for a special form of global recoding have been developed by Hurkens and Tiourine (1998), generalizing the kind of models presented. The models these authors consider are able to deal with local suppression and global recoding simultaneously. For these models it is assumed that a data protector has, for each variable that appears in the combinations of variables checked, provided a set of alternative codings. It is also requested from this person to specify how much information is lost when a particular global recoding is applied. The idea is that a global recoding for a variable in this case is nothing but a selection of one of the predefined codings for that variable.

The optimization problem that has to be solved is to select the global recodings in such a way that the resulting microdata set is safe and the associated information loss is minimized.

10. References

De Waal, A.G. and Willenborg, L.C.R.J. (1996). A View on Statistical Disclosure Control of Microdata. Survey Methodology, 22, 95–103.

Duncan, G.T. and Lambert, D. (1986). Disclosure-limited Data Dissemination (with discussion). Journal of the American Statistical Association, 81, 10–28.

Duncan, G.T. and Lambert, D. (1989). The Risk of Disclosure for Microdata. Journal of Business and Economic Statistics, 7, 207–217.

Fienberg, S.E. (1994). Conflicts Between the Needs for Access to Statistical Information and Demands for Confidentiality. Journal of Official Statistics, 10, 115–132.

Hurkens, C.A.J. and Tiourine, S.R. (1998a). On Solving Huge Set-cover Models of the Microdata Protection Problem. Paper presented at SDP '98, 25–27 March, Lisbon, Portugal.

Hurkens, C.A.J. and Tiourine, S.R. (1998b). Models and Methods for the Microdata Protection Problem. Journal of Official Statistics, 14, 437–447.

Lambert, D. (1993). Measures of Disclosure Risk and Harm. Journal of Official Statistics, 9, 313–331.

Marsh, C., Dale, A., and Skinner, C.J. (1994). Safe Data versus Safe Settings: Access to Microdata from the British Census. International Statistical Review, 62, 32–54.

Nemhauser, G.L. and Wolsey, L.A. (1988). Integer and Combinatorial Optimization. Wiley, New York.

Paass, G. (1988). Disclosure Risk and Disclosure Avoidance for Microdata. Journal of Business and Economic Statistics, 6, 487–500.

Pannekoek, J. and de Waal, T. (1998). Synthetic and Combined Estimators in Statistical Disclosure Control. Journal of Official Statistics, 14, 399–410.

Skinner, C.J., Marsh, C., Openshaw, S., and Wymer, C. (1994). Disclosure Control for Census Microdata. Journal of Official Statistics, 10, 31–51.

Van Gelderen, R.P. (1995). ARGUS Statistical Disclosure Control of Survey Data. Report, Statistics Netherlands, Voorburg.

Vasko, F.J. and Wilson, G.R. (1986). Hybrid Heuristics for Minimum Cardinality Set-Covering Problems. Naval Research Logistics Quarterly, 33, 241–249.

Willenborg, L. and De Waal, T. (1996). Statistical Disclosure Control in Practice. Lecture Notes in Statistics, Volume 111, Springer-Verlag, New York.

Zaslavsky, A.M. and Horton, N.J. (1998). Balancing Disclosure Risk Against the Loss of Nonpublication. Journal of Official Statistics, 14, 411–419.

Received April 1997
Revised June 1998