Noname manuscript No.
(will be inserted by the editor)

Information Preserving Regression-based Tools for Statistical Disclosure Control

Øyvind Langsrud

Statistics and Computing, 2019, Volume 29, Issue ?, pp ???-???

Abstract This paper presents a unified framework for regression-based statistical disclosure control for microdata. A basic method, known as information preserving statistical obfuscation (IPSO), produces synthetic data that preserve variances, covariances and fitted values. The data are then generated conditionally according to the multivariate normal distribution. Generalizations of the IPSO method are described in the literature and these methods aim to generate data more similar to the original data. This paper describes these methods in a concise and interpretable way, which is close to efficient implementation. Decomposing the residual data into orthogonal scores and corresponding loadings is an essential part of the framework. Both QR decomposition (Gram Schmidt orthogonalization) and singular value decomposition (principal components) may be used. Within this framework, new and generalized methods are presented. In particular, a method is described by means of which the correlations to the original principal component scores can be controlled exactly. It is shown that a suggested method of random orthogonal matrix masking (ROMM) can be implemented without generating an orthogonal matrix. Generalized methodology for hierarchical categories is presented within the context of microaggregation. Some information can then be preserved at the lowest level and more information at higher levels. The presented methodology is also applicable to tabular data. One possibility is to replace the content of primary and secondary suppressed cells with generated values. It is proposed replacing suppressed cell frequencies with decimal numbers and it is argued that this can be a useful method.

Keywords microdata anonymization · synthetic data · microaggregation · hybrid microdata · cell suppression · official statistics

1 Introduction

Microdata are data sets in which each record contains several variables concerning a person or an organization. Usually a microdata data set cannot be made publicly available for reasons of confidentiality. Methods for statistical disclosure control of microdata aim to create protected data that can be released (Hundepool et al. 2012). Two main categories are non-perturbative methods (information reduction, coarsening data) and perturbative methods (changing data). The latter category is often divided into perturbative masking methods and synthetic data generators. In addition, hybrid data generators combine original and synthetic data. Synthetic data are data randomly drawn from a statistical model, often under constraints such that certain statistics or internal relationships of the original data set are preserved.

A broad class of masking methods falls into the category of matrix masking (Duncan and Pearson 1991), which consists of replacing the original data matrix, $\boldsymbol{Y}$, with $\boldsymbol{M}_1\boldsymbol{Y}\boldsymbol{M}_2 + \boldsymbol{M}_3$. Then, $\boldsymbol{M}_1$ is a record-transforming mask, $\boldsymbol{M}_2$ is a variable-transforming mask, and $\boldsymbol{M}_3$ is a displacing mask. A specific masking method is noise addition where $\boldsymbol{M}_1$ and $\boldsymbol{M}_2$ are identity matrices and $\boldsymbol{M}_3$ contains randomly generated data. As follows below, when the M-matrices are allowed to be stochastically generated, it becomes problematic to dis-

Ø. Langsrud
Statistics Norway, P.O. Box 8131 Dep., 0033 Oslo, Norway
Tel.: +47-21094425
E-mail: Oyvind.Langsrud@ssb.no

tinguish between masking methods and synthetic data generators.

Given a parametric model, making statistical inference from synthetic data the same way as from original data requires that sufficient statistics for the joint distribution of all variables in the data are preserved. This can be achieved by drawing data jointly from the conditional distribution. In general, this is difficult, and data may instead be drawn with fixed values of uncertain parameters. The distributional properties are then violated and variances in particular will be underestimated.

The problems just described are commonly handled by using multiple imputation with fully conditional specifications (Loong and Rubin 2017; Drechsler 2011; Reiter and Raghunathan 2007). Multiple imputation makes it possible to correct for the additional variance from imputation. The use of fully conditional specifications is a way of overcoming the problems of drawing from a multivariate distribution. Then, the assumed joint distribution is specified indirectly. The United States Census Bureau releases synthetic data products prepared by using multiple imputation (Benedetto et al. 2013; Jarmin et al. 2014). The conditional specifications of continuous variables involve normality and linear regression models. These specifications are consistent with the properties of a joint multivariate normal distribution. Sometimes the variables are transformed so that they have an approximately normal distribution.

When all the variables to be synthesized are continuous and assuming multivariate normality and a linear regression model, it is possible to generate synthetic data in a way that exactly preserves the distributional properties. Then there is no need for multiple imputation. Such a basic regression-based synthetic data generator is described by Burridge (2003) as information preserving statistical obfuscation (IPSO). The generated data then constitute a matrix of independent samples from the multivariate normal distribution conditioned on parameter estimates. An equivalent data generation method was described by Langsrud (2005) within the area of Monte Carlo testing. The simulation-based tests were referred to as rotation tests since the generated data can be interpreted as randomly rotated data. This means that the data can be generated as matrix masking with $\boldsymbol{M}_1$ as a random rotation matrix and with $\boldsymbol{M}_2$ being identity and $\boldsymbol{M}_3$ being zero. Strictly speaking, $\boldsymbol{M}_1$ is then a uniformly (according to the Haar measure) distributed random orthogonal matrix restricted to preserving fitted values. Given that $\boldsymbol{M}_1$ is a restricted orthogonal matrix, this ensures that the sample covariance matrix is preserved. The specific uniform distribution ensures that original and generated data are independent. Such simulation methodology can be traced back to Wedderburn (1975) in the only intercept case where fitted values are the sample means. In this case Ting et al. (2008) described generalized methodology called random orthogonal matrix masking (ROMM). Their generalization allows $\boldsymbol{M}_1$ to be generated in other ways so that the similarity to the original data can be controlled.

Authors of popular software tools for microdata protection emphasize the importance of low information loss or high data utility provided that the disclosure risk is acceptable (Hundepool et al. 2014; Templ et al. 2015). Therefore, it is beneficial to use information preserving methods that generate data more similar to the original data than the IPSO method. In addition to ROMM, such methods are described in Muralidhar and Sarathy (2008) and in Domingo-Ferrer and Gonzalez-Nicolas (2010). In these papers the distributional assumptions are discussed in detail. The IPSO method is made to preserve the sufficient statistics under multivariate normality. Even if the original data are not multivariate normal, the results of common statistical analyses relying on (multivariate) normality will be the same regardless of whether the original or the IPSO generated data are used. The results of analyses not relying on normality will, however, be different. Generating data more similar to the original data also means that the results of any analysis will tend to be more similar to the results based on the original data. The empirical distribution of the original data will also be much better approximated.

The present paper describes several information preserving methods under a common matrix decomposition framework. The aim is twofold. First, to describe existing regression-based methods in a concise and interpretable way, which is close to efficient implementation. Second, to develop new techniques and generalized methodology within the framework. All in all, an important class of tools within the area of statistical disclosure control is described. Disclosure risk and information loss are not investigated in this paper, but both are important when the methodology is applied in practise. We believe that the methodology is well suited to controlling these properties, since the flexibility enables targeted changes.

A requirement that fitted values must be preserved means that we can only change the residual data. In Section 2 the residual data are decomposed into scores and loading using an optional decomposition method (QR or SVD/PCA). Generating data with preserved information can be done by generating new scores and Section 2 formulates an algorithm for the IPSO method.

Section 3 considers the addition of synthetic values to preliminary residuals from arbitrarily predefined data. According to Muralidhar and Sarathy (2008) in particular, it is possible at the variable level to select the degree of similarity to the original data. We will also suggest another and related method that makes it possible to control exactly the correlation between generated and original scores. This method can be viewed as a modified and extended variant of the principal component approach in Calvino (2017).

Section 4 describes an approach in which the IPSO algorithm is modified by using scores from arbitrary residual data. In Section 5 we look into the topic of ROMM and discuss how other methods can be viewed within this context. We show that simulations according to Section 4 can be an efficient and equivalent alternative to doing computations via an orthogonal matrix.

Section 6 considers the generation of microdata within the context of microaggregation. This means that the IPSO method is applied to several clusters/categories as described by Domingo-Ferrer and Gonzalez-Nicolas (2010). New generalized methodology for hierarchical categories is presented. Some information can then be preserved at the lowest level and more information at higher levels.

The methodology is also directly applicable to tabular data obtained by crossing several categorical variables. Statistical disclosure control for tabular data is an important field and several methods exist (Hundepool et al. 2012; Salazar-Gonzalez 2008). In Section 7 we assume that a table suppression method has been applied. It is then possible to replace the content of suppressed (primary and secondary) cells by generated/ synthetic values. In particular, it is suggested replacing suppressed cell frequencies with decimal numbers and it is argued that this can be a useful method.

2 Information preserving statistical obfuscation (IPSO)

Assume a multivariate multiple regression model defined by

$$\boldsymbol{Y} = \boldsymbol{X}\boldsymbol{B} + \boldsymbol{E} \tag{1}$$

where $\boldsymbol{Y}$ is a $n \times k$ matrix containing n observations of k confidential variables and where the $n \times m$ matrix $\boldsymbol{X}$ contains corresponding non-confidential variables including a constant term (column of ones). The standard assumptions are that the rows of $\boldsymbol{E}$ are independent multivariate normal with zero means and a common covariance matrix.

We now want to generate, $\boldsymbol{Y}^*$, a synthetic variant of $\boldsymbol{Y}$, so that the fitted values and the sample covariance matrix are preserved. Equivalently, we impose the restrictions that ${\boldsymbol{Y}^*}^T\boldsymbol{Y}^* = \boldsymbol{Y}^T\boldsymbol{Y}$ and $\boldsymbol{X}^T\boldsymbol{Y}^* = \boldsymbol{X}^T\boldsymbol{Y}$. The synthetic data can be drawn from the multivariate regression model conditioned on the restrictions. Burridge (2003) described such simulations as information preserving statistical obfuscation. Langsrud (2005) considered the same problem within the area of Monte Carlo testing and the simulation-based tests were referred to as rotation tests. He described an algorithm based on QR decomposition which is very similar to the algorithm in Burridge (2003). Here we formulate a similar algorithm using a general decomposition:

$$\boldsymbol{Y} = \hat{\boldsymbol{Y}} + \hat{\boldsymbol{E}} = \hat{\boldsymbol{Y}} + \boldsymbol{T}\boldsymbol{W} \tag{2}$$

$$\boldsymbol{Y}_s = \hat{\boldsymbol{Y}}_s + \hat{\boldsymbol{E}}_s = \hat{\boldsymbol{Y}}_s + \boldsymbol{T}^*\boldsymbol{W}_s \tag{3}$$

$$\boldsymbol{Y}^* = \hat{\boldsymbol{Y}} + \hat{\boldsymbol{E}}^* = \hat{\boldsymbol{Y}} + \boldsymbol{T}^*\boldsymbol{W} \tag{4}$$

The data is split into fitted values and residuals. Some method is used to decompose the residuals as a matrix, $\boldsymbol{T}$, of orthogonal scores and a matrix, $\boldsymbol{W}$, of loadings (2). A simulated data matrix, $\boldsymbol{Y}_s$, whose elements are independent standard normal deviates, is decomposed similarly (3) to obtain the matrix $\boldsymbol{T}^*$ of synthetic scores orthogonal to $\boldsymbol{X}$. The synthetic data is obtained by replacing $\boldsymbol{T}$ by $\boldsymbol{T}^*$ (4).

The fitted values are computed according to the regression model (1). We allow collinear columns in $\boldsymbol{X}$. The fitted values are uniquely defined independently of how this problem is handled. Two obvious candidates for the decomposition method are QR decomposition (Gram Schmidt orthogonalization) and singular value decomposition (SVD) (Strang 1988). In the present paper we refer to a generalized QR decomposition as described in Appendix 1 ($\boldsymbol{T} = \boldsymbol{Q}$ and $\boldsymbol{W} = \boldsymbol{R}$). This decomposition is unique and dependent columns are allowed. When using SVD as dealt with in Appendix 2, we are essentially performing principal components analysis (PCA) (Jolliffe 2002), but the scores are scaled differently ($\boldsymbol{T} = \boldsymbol{U}$ and $\boldsymbol{W} = \boldsymbol{\Lambda}\boldsymbol{V}^T$).

The properties of the synthetic data are independent of the choice of a decomposition method, but this choice is important in some of the modifications below. Note that Burridge (2003) formulated the algorithm using the Choleski decomposition (see Appendix 1). Then, $\boldsymbol{W}$ is calculated from the covariance matrix estimate and $\hat{\boldsymbol{E}}^*$ is found as $\hat{\boldsymbol{E}}_s\boldsymbol{W}_s^{-1}\boldsymbol{W}$. In the only intercept case, Mateo-Sanz et al. (2004) formulated another algorithm where $\boldsymbol{W}$ is calculated by Choleski decomposition and $\boldsymbol{T}^*$ is generated by an orthogonalization algorithm.

Note that in the special case of a single confidential variable ($k = 1$), then $\boldsymbol{T}$ is simply the vector of residuals scaled to unit length and $\boldsymbol{W}$ is the scale factor (scalar). Thus, $\boldsymbol{Y}^*$ is obtained by replacing the original residuals by residuals obtained from simulated data, scaled

so that the sum of squared residuals is preserved. This yields a simplified procedure compared to the one recently published in Klein and Datta (2018).

A logical question is: Why not simply release the regression coefficients and the covariance matrix? Since $\boldsymbol{X}$ is released there is no more information in the IPSO-generated data than in these estimates. A good reason for releasing synthetic data is that this is a user-friendly product. Most users would prefer a data set which can be analyzed by ordinary tools. Sophisticated users who would prefer estimates instead can easily calculate them. Anyway, this question is relevant for IPSO, but not for the other methods described below.

3 Arbitrary residual data with a synthetic addition

Instead of drawing synthetic data directly from the model we may start with an arbitrary data set $\boldsymbol{Y}_g$ and add a synthetic matrix to preserve the required information. However, since we want to preserve the fitted values we will only use the residuals from the arbitrary data set. Then, we generate synthetic data according to

$$\boldsymbol{Y}_g = \hat{\boldsymbol{Y}}_g + \hat{\boldsymbol{E}}_g \tag{5}$$

$$\boldsymbol{Y}^* = \hat{\boldsymbol{Y}} + \hat{\boldsymbol{E}}_g + \boldsymbol{T}^*\boldsymbol{C} \tag{6}$$

Here we make use of a synthetic $\boldsymbol{T}^*$ similar to the one generated in (3). In this case we will generate $\boldsymbol{T}^*$ in a manner that it is orthogonal to both $\boldsymbol{X}$ and $\boldsymbol{Y}_g$. In practice the QR decomposition of a composite matrix (27) may be used. The matrix $\boldsymbol{C}$ is calculated from an equation (see below), but a solution may not exist.

With reference to the decomposition of $\boldsymbol{Y}$ (2) two important special cases of this approach can be expressed as

$$\boldsymbol{Y}^* = \hat{\boldsymbol{Y}} + \hat{\boldsymbol{E}}\boldsymbol{D} + \boldsymbol{T}^*\boldsymbol{C} \tag{7}$$

$$\boldsymbol{Y}^* = \hat{\boldsymbol{Y}} + \boldsymbol{T}\boldsymbol{D}\boldsymbol{W} + \boldsymbol{T}^*\sqrt{\boldsymbol{I}-\boldsymbol{D}^2}\boldsymbol{W} \tag{8}$$

where $\boldsymbol{D}$ is a diagonal matrix whose i'th diagonal element ($d_i \in [0,1]$) controls how much information from the i'th column of $\hat{\boldsymbol{E}}$ (7) or $\boldsymbol{T}$ (8) is re-used. The first method (7) was introduced by Muralidhar and Sarathy (2008) and is described in a different way in their paper. Equation (8) is written without $\boldsymbol{C}$ since, in that case, the expression for $\boldsymbol{C}$ is known. The square root of a diagonal matrix means that we take the square root of each diagonal element. Hence, we calculate new unit score vectors as $d_iT_i + \sqrt{1-d_i^2}T_i^*$ where T_i denotes the i'th column of $\boldsymbol{T}$. With a constant term included in $\boldsymbol{X}$, it is easy to show that d_i is a correlation coefficient in both equations. That is, in (7), for each variable, we control exactly the correlation between the original and the generated residuals. And in (8) we control exactly the correlation between the original score vector and the score vector of the generated data. Note that if all diagonal elements of $\boldsymbol{D}$ are set to to be equal, (7) and (8) become equivalent.

Using (8) in the cases where all d_i's are either zero or one, we are keeping some score vectors and simulating others. If in addition the decomposition method used in (2) is SVD, the method is very similar to the one described in Calvino (2017), where some PCA score vectors are swapped (randomly permuted). Using swapping instead of the above simulation leads to an approximate instead of an exact preservation of the covariance matrix. Our approach is also more general than the one in Calvino (2017) since we allow regression variables beyond the constant term. Thus, within our framework, PCA means PCA of residuals. In any case, we preserve regression fits and preserving some PCA scores is one way of generating data that is even more similar to the original data.

Working with PCA score vectors can be very useful when there are many highly correlated variables, especially when a few components account for most of the variability in the data. One may want to rotate the components for better interpretability, however. Understanding the components is of particular interest when some variables are considered more sensitive than others. Then, the correlations (d_i's) can be specified accordingly. Possible rotation of components fits into the general framework of this paper. In fact, going from PCA to QR is a rotation. If the aim is to have good control of the relationship to a few important y-variables, the QR approach would be easier. One may choose the column ordering of the y-variables to correspond to the importance. Then d_1 is the correlation between the original and the generated residuals for the first y-variable, similar to (7). Also note that, when using QR, setting some of the first d_i's equal to one is equivalent to reclassifying those y-variables as x-variables. In general, the method based on controlling components (8) is more reliable than (7) because it avoids the problem of a possible nonexistent $\boldsymbol{C}$.

To discuss the calculation of $\boldsymbol{C}$ in general we will introduce a scalar parameter α and we re-write equation (6) as

$$\boldsymbol{Y}^* = \hat{\boldsymbol{Y}} + \alpha\hat{\boldsymbol{E}}_g + \boldsymbol{T}^*\boldsymbol{C} \tag{9}$$

The matrix $\boldsymbol{C}$ can now be found from the equation

$$\boldsymbol{C}^T\boldsymbol{C} = \hat{\boldsymbol{E}}^T\hat{\boldsymbol{E}} - \alpha^2\hat{\boldsymbol{E}}_g^T\hat{\boldsymbol{E}}_g \tag{10}$$

and the Cholesky decomposition may be used to compute $\boldsymbol{C}$ when possible. An alternative is to find $\boldsymbol{C}$ via

the eigen decomposition (see Appendix 2). But a solution may not exist since negative eigenvalues of the computed right side can occur (not positive definite).

However, we will present here a way of computing the largest α that makes the equation solvable. By manipulating the characteristic eigenvalue equation it turns out that the limit is found as the square root of the smallest eigenvalue of $\hat{\boldsymbol{E}}^T\hat{\boldsymbol{E}}(\hat{\boldsymbol{E}}_g^T\hat{\boldsymbol{E}}_g)^{-1}$. A modification is needed when $(\hat{\boldsymbol{E}}_g^T\hat{\boldsymbol{E}}_g)$ is not invertible. One possibility is to find the limit as the square root of the inverse of the largest eigenvalue of $(\hat{\boldsymbol{E}}^T\hat{\boldsymbol{E}})^{-1}\hat{\boldsymbol{E}}_g^T\hat{\boldsymbol{E}}_g$. The calculation of the limit can be useful in combination with the method (7) originally described by Muralidhar and Sarathy (2008). In practice, one may choose the limit if $\alpha = 1$ cannot be chosen. Using $\alpha < 1$ means that the initial residual data are downscaled. In (7) this means that all the required correlations are multiplied by α and that the similarity to the original data is not as high as expected.

4 Using scores from arbitrary residual data

To generate $\boldsymbol{Y}^*$ in a way that preserves the information without considering its distributional properties, $\boldsymbol{Y}_s$ (3) can be replaced by anything as long as the residual data have the desired dimension. Necessary scores can then be computed. Using an arbitrary data set $\boldsymbol{Y}_g$, synthetic data can be generated by

$$\boldsymbol{Y}_g = \hat{\boldsymbol{Y}}_g + \hat{\boldsymbol{E}}_g = \hat{\boldsymbol{Y}}_g + \boldsymbol{T}_g\boldsymbol{W}_g \tag{11}$$

$$\boldsymbol{Y}^* = \hat{\boldsymbol{Y}} + \boldsymbol{T}_g\boldsymbol{W} \tag{12}$$

with $\boldsymbol{W}$ as above (2).

A possible application is when $\boldsymbol{Y}_g$ is a preliminary data set in some sense. In particular, the methodology can be used in combination with another method for anonymization of microdata. If the result of such a method ($\boldsymbol{Y}_g$) preserves the information approximately, the method here can be used as a correction to achieve exactness ($\boldsymbol{Y}^*$). In such cases one would expect $\boldsymbol{Y}^*$ to be close to $\boldsymbol{Y}_g$. This is the case when QR is used as the decomposition method in (2), but not necessarily so when SVD is used. The problem is that SVD is unstable when the difference between two singular values is small. Therefore, QR decomposition is preferable.

Another possible application is to combine original and randomly generated data and one may generate $\boldsymbol{Y}_g$ by

$$\boldsymbol{Y}_g = \boldsymbol{Y}\boldsymbol{D} + \boldsymbol{Y}_s(\boldsymbol{I} - \boldsymbol{D}) \tag{13}$$

using a diagonal matrix, $\boldsymbol{D}$, as earlier. To make the parameters in $\boldsymbol{D}$ interpretable in this case, the variables in the randomly generated data, $\boldsymbol{Y}_s$, should be scaled to have variances similar to those of the original variables. When all diagonal elements are close to one this means that $\boldsymbol{Y}_g$ is close to $\boldsymbol{Y}$. Again, it is important to use QR decomposition to avoid instability.

Since the aim is to generate scores, $\boldsymbol{Y}_g$ may be created by using original scores instead of original data. This the case for the ROMM method described below in (17).

5 Random orthogonal matrix masking (ROMM)

Data generated according to ROMM (Ting et al. 2008) can be expressed as

$$\boldsymbol{Y}^* = \boldsymbol{M}^*\boldsymbol{Y} \tag{14}$$

where $\boldsymbol{M}^*$ is a randomly generated orthogonal matrix. As described by Langsrud (2005) the basic IPSO method of Section 2 can equivalently be formulated as such a ROMM method with

$$\boldsymbol{M}^* = \boldsymbol{U}_X\boldsymbol{U}_X^T + \boldsymbol{U}_E\boldsymbol{P}^*\boldsymbol{U}_E^T \tag{15}$$

where the columns of $\boldsymbol{U}_X$ form an orthogonal basis for the column space of $\boldsymbol{X}$ and $\boldsymbol{U}_E$ form an orthonormal basis for the complement so that $[\boldsymbol{U}_X\ \boldsymbol{U}_E]$ is an orthogonal matrix. The matrix $\boldsymbol{P}^*$ is drawn as a uniformly distributed orthogonal matrix. Ting et al. (2008) stated that all methods that preserve means and the covariance matrix are special cases of ROMM. This also holds beyond the only intercept case. Regardless of the method, as long as the generated data preserve information as above, we can always write $\boldsymbol{Y}^* = \hat{\boldsymbol{Y}} + \boldsymbol{T}^*\boldsymbol{W} = \boldsymbol{M}^*\boldsymbol{Y}$ where

$$\boldsymbol{M}^* = [\,\boldsymbol{U}_X\ \boldsymbol{T}^*\ \boldsymbol{U}_A^*\,]\,[\,\boldsymbol{U}_X\ \boldsymbol{T}\ \boldsymbol{U}_A\,]^T \tag{16}$$

Here, $\boldsymbol{U}_A$ and $\boldsymbol{U}_A^*$ are additional orthogonal columns constructed so that the composite matrices are square.

A ROMM approach to generalizing the IPSO method can be expressed by (15) and by letting $\boldsymbol{P}^*$ be the result of orthogonalizing $\boldsymbol{I} + \lambda\boldsymbol{H}$ where $\boldsymbol{H}$ is a matrix filled with standard normal deviates. Since we are referring to a regression model, this is an extended version of the method (only intercept) originally proposed in Ting et al. (2008). However, in that paper, such an extension was indicated in the appendix. The QR decomposition will be used to orthogonalize $\boldsymbol{I} + \lambda\boldsymbol{H}$, and SVD will not work. The identity matrix is obtained when $\lambda = 0$ and the original data are then unchanged. An ordinary uniformly distributed orthogonal matrix is obtained when $\lambda \to \infty$ and thus we have the IPSO method. Since QR decomposition is sequential, the diagonal elements of $\boldsymbol{P}^*$ will have a decreasing tendency to approach one (sequentially phenomenon). This

is probably not an intention of the method. This means that the choice of the basis, $\boldsymbol{U}_E$, matters. To avoid this phenomenon, a possibility is to generate the basis randomly. To discuss this further we look at the special case without a constant term or any other x-variables. We also assume that $\boldsymbol{T}$ has full rank (or is temporary extended to have full rank). Now it is possible to use $\boldsymbol{P}^*$ directly as $\boldsymbol{M}^*$ so that $\boldsymbol{T}^* = \boldsymbol{P}^*\boldsymbol{T}$. Thus, the above sequentially phenomenon means that the similarity to the original data is highest for the first observation. Another possibility is to make use of (15) without $\boldsymbol{U}_X$ and with $\boldsymbol{U}_E = \boldsymbol{T}$. Then it follows that $\boldsymbol{T}^* = \boldsymbol{T}\boldsymbol{P}^*$ and now the sequentially phenomenon means that the similarity to the original data is highest for the first score vector. Since $\boldsymbol{T}$ is orthogonal it turns out that we can perform the QR needed to create $\boldsymbol{P}^*$ from $\boldsymbol{I} + \lambda\boldsymbol{H}$ after left multiplication with $\boldsymbol{T}$. In addition we have that $\boldsymbol{T}\boldsymbol{H}$ is distributed exactly like $\boldsymbol{H}$. That is, we can find $\boldsymbol{T}^*$ by orthogonalizing $\boldsymbol{T} + \lambda\boldsymbol{H}$. If we only need the first columns of $\boldsymbol{T}^*$ (others were included temporarily), we can simplify the method, without changing the result, by only generating as many columns of $\boldsymbol{H}$ as needed.

The discussion when we have x-variables is similar, but a little trickier. We need to "work" in the correct subspace. One possibility is to generate $\boldsymbol{H}$ so that it is orthogonal to $\boldsymbol{X}$. This can be done by left-multiplying a preliminarily generated matrix by $\boldsymbol{U}_E$. An easier and equivalent approach is to generate $\boldsymbol{H}$ straightforwardly and instead enforce orthogonality to $\boldsymbol{X}$ afterwards. Thus we obtain a method that is a special case of the one in Section 4 with

$$\boldsymbol{Y}_g = \boldsymbol{T} + \lambda\boldsymbol{H} \tag{17}$$

where $\boldsymbol{H}$ is a matrix filled with standard normal deviates. This method is more efficient than generating the orthogonal matrix, $\boldsymbol{M}^*$. Especially in cases with a large number of observations, ROMM will be very time and memory consuming. Using (11), (12) and (17) now solves this problem.

Note that Ting et al. (2008) have described another ROMM approach that makes use of special block diagonal matrices. We will not discuss this topic in detail. But when IPSO is performed separately within clusters, as described in the section below, this can be interpreted according to (14) where $\boldsymbol{M}^*$ is a block diagonal matrix where the blocks are independent random orthogonal matrices.

6 Generalized microaggregation

A particular method for statistical disclosure control is microaggregation (Domingo-Ferrer and Mateo-Sanz 2002; Templ et al. 2015). Then, a method is first used to group the records into clusters. The aggregates within these clusters are released instead of the individual record values. Equivalently, the microdata records can be replaces by averages within the clusters.

The clustered data can also be the starting point for the IPSO method in Section 2 and one may proceed in two ways:

MHa: Run the IPSO procedure separately within each cluster.

MHb: Add dummy variables to $\boldsymbol{X}$ corresponding to the clustering before running IPSO.

The first method is described by Domingo-Ferrer and Gonzalez-Nicolas (2010) as a hybrid method that combines microaggregation with generation of synthetic data. The notation, MHa, is chosen since this method is known as MicroHybrid. The alternative method introduced here, MHb, is also useful in combination with microaggregation. In both methods sums within clusters, the overall fitted values and the overall sample covariance matrix are preserved. In MHa the fits and the covariance matrix estimates within each cluster are also preserved. An extended variant of MHb is to cross all the original x-variables with the clusters. Then all the fits will be preserved within clusters, but not the covariance matrix estimates.

To discuss this more generally we will present an algorithm in which some regression variables are crossed with clusters and some are not. For this purpose we assume two data matrices of x-variables, $\boldsymbol{X}_A$ and $\boldsymbol{X}_B$. The matrix $\boldsymbol{X}_A$ consists of all the variables to be crossed with clusters (including the intercept). As before, we let $\boldsymbol{X}$ denote the full regression model matrix. We will now extend and partition this matrix as

$$\boldsymbol{X}_{ext} = [\,\boldsymbol{X}\ \ \boldsymbol{X}_3\,] = [\,\boldsymbol{X}_1\ \ \boldsymbol{X}_2\ \ \boldsymbol{X}_3\,] \tag{18}$$

- $\boldsymbol{X}_1$ is the regression variable matrix obtained by crossing $\boldsymbol{X}_A$ with clusters.
- $\boldsymbol{X}_2$ contains $\boldsymbol{X}_B$ adjusted for (made orthogonal to) $\boldsymbol{X}_1$.
- $\boldsymbol{X}_3$ contains the regression variable matrix obtained by crossing $\boldsymbol{X}_B$ with clusters followed by adjustment for $\boldsymbol{X}_1$ and $\boldsymbol{X}_2$.

Fitted values obtained by regressing $\boldsymbol{Y}$ onto $\boldsymbol{X}$ and the corresponding residuals are two orthogonal parts of $\boldsymbol{Y}$. According to (18) we also divide each of these parts into two orthogonal parts.

$$\boldsymbol{Y} = \hat{\boldsymbol{Y}} + \hat{\boldsymbol{E}} = \left(\hat{\boldsymbol{Y}}_1 + \hat{\boldsymbol{Y}}_2\right) + \left(\hat{\boldsymbol{E}}_3 + \hat{\boldsymbol{E}}_4\right) \tag{19}$$

Here $\hat{\boldsymbol{Y}}_1$, $\hat{\boldsymbol{Y}}_2$ and $\hat{\boldsymbol{E}}_3$ are the regression fits obtained by using $\boldsymbol{X}_1$, $\boldsymbol{X}_2$ and $\boldsymbol{X}_3$, respectively.

Performing computations via $\boldsymbol{X}$ will be very inefficient when we have several variables, observations and

clusters. We also want to (partly) preserve the covariance matrix within clusters. Therefore some of the computations will be performed within clusters and in the following we will call this the local level. As opposed to *globally*, *locally* means that we perform the computations by looping through all the clusters. In particular, we can compute the fitted values $\hat{\boldsymbol{Y}}_1$ locally and then we can use the original matrix $\boldsymbol{X}_A$ and will not need $\boldsymbol{X}_1$. Above, $\boldsymbol{X}_2$ was made orthogonal to $\boldsymbol{X}_1$. In this case it does not matter whether we do the orthogonalization globally or locally. In the algorithm below we perform this computation locally using $\boldsymbol{X}_A$ instead of $\boldsymbol{X}_1$. As mentioned above, the regression fits obtained by using $\boldsymbol{X}_2$ globally are contained in $\hat{\boldsymbol{Y}}_2$. If we instead use $\boldsymbol{X}_2$ to compute regression fits locally, the result is $\hat{\boldsymbol{Y}}_{2ext} = \hat{\boldsymbol{Y}}_2 + \hat{\boldsymbol{E}}_3$. This means that we have sketched how to calculate the four parts in (19). In order to compute synthetic data this will be combined with computations on simulated data. We can use the following algorithm for all the computations:

1. Globally simulate a data matrix $\boldsymbol{Y}_s$ with the same dimension as $\boldsymbol{Y}$.
2. Locally calculate $\hat{\boldsymbol{Y}}_1$, $\hat{\boldsymbol{Y}}_{s1}$ and $\hat{\boldsymbol{X}}_B$ by regressing $\boldsymbol{Y}$, $\boldsymbol{Y}_s$ and $\boldsymbol{X}_B$ onto $\boldsymbol{X}_A$.
3. Locally calculate $\boldsymbol{X}_2 = \boldsymbol{X}_B - \hat{\boldsymbol{X}}_B$.
4. Locally calculate $\hat{\boldsymbol{Y}}_{2ext}$ and $\hat{\boldsymbol{Y}}_{s2ext}$ by regressing $\boldsymbol{Y}$ and $\boldsymbol{Y}_s$ onto $\boldsymbol{X}_2$.
5. Locally calculate $\hat{\boldsymbol{E}}_4 = \boldsymbol{Y} - \hat{\boldsymbol{Y}}_1 - \hat{\boldsymbol{Y}}_{2ext}$ and $\hat{\boldsymbol{E}}_{s4} = \boldsymbol{Y}_s - \hat{\boldsymbol{Y}}_{s1} - \hat{\boldsymbol{Y}}_{s2ext}$.
6. Locally decompose $\hat{\boldsymbol{E}}_4 = \boldsymbol{T}_4\boldsymbol{W}_4$ and $\hat{\boldsymbol{E}}_{s4} = \boldsymbol{T}_{s4}\boldsymbol{W}_{s4}$ and calculate $\hat{\boldsymbol{E}}_4^* = \boldsymbol{T}_{s4}\boldsymbol{W}_4$.
7. Globally calculate $\hat{\boldsymbol{Y}}_2$ and $\hat{\boldsymbol{Y}}_{s2}$ by regressing $\boldsymbol{Y}$ and $\boldsymbol{Y}_s$ onto $\boldsymbol{X}_2$.
8. Globally calculate $\hat{\boldsymbol{E}}_3 = \hat{\boldsymbol{Y}}_{2ext} - \hat{\boldsymbol{Y}}_2$ and $\hat{\boldsymbol{E}}_{s3} = \hat{\boldsymbol{Y}}_{s2ext} - \hat{\boldsymbol{Y}}_{s2}$.
9. Globally decompose $\hat{\boldsymbol{E}}_3 = \boldsymbol{T}_3\boldsymbol{W}_3$ and $\hat{\boldsymbol{E}}_{s3} = \boldsymbol{T}_{s3}\boldsymbol{W}_{s3}$ and calculate $\hat{\boldsymbol{E}}_3^* = \boldsymbol{T}_{s3}\boldsymbol{W}_3$.
10. Globally calculate $\boldsymbol{Y}^* = \hat{\boldsymbol{Y}}_1 + \hat{\boldsymbol{Y}}_2 + \hat{\boldsymbol{E}}_3^* + \hat{\boldsymbol{E}}_4^*$.

In addition to this main algorithm we will discuss five possible modifications.

a) Drop item 6 and replace item 9 by: Locally decompose $\hat{\boldsymbol{E}}_3 + \hat{\boldsymbol{E}}_4 = \boldsymbol{T}\boldsymbol{W}$ and let $\hat{\boldsymbol{E}}_4^* = \boldsymbol{T}_{s4}\boldsymbol{W}$ and set $\hat{\boldsymbol{E}}_3^* = 0$.

b) Drop item 6 and instead, after item 9: Find locally $\hat{\boldsymbol{E}}_4^* = \boldsymbol{T}_{s4}\boldsymbol{C}$ where the matrix $\boldsymbol{C}$ satisfies $\boldsymbol{C}^T\boldsymbol{C} = \hat{\boldsymbol{E}}_4^T\hat{\boldsymbol{E}}_4 + \hat{\boldsymbol{E}}_3^T\hat{\boldsymbol{E}}_3 - \hat{\boldsymbol{E}}_3^{*T}\hat{\boldsymbol{E}}_3^*$. This is similar to (6).

c) This is a generalization with a) and b) as special cases. Drop item 6 and modify item 9 by $\hat{\boldsymbol{E}}_3^* = \alpha\boldsymbol{T}_{s3}\boldsymbol{W}_3$ and find $\hat{\boldsymbol{E}}_4^*$ as in b). This is similar to (9).

d) If we have groups of clusters, we may want some of the variables in $\boldsymbol{X}_B$ to be crossed with these groups. At the beginning of the algorithm we then replace $\boldsymbol{X}_B$ with a model matrix that contains such crossing.

e) If the clusters are partitioned into smaller pieces, we may want some of the variables in $\boldsymbol{X}_A$ (maybe only the constant term) to be crossed with these pieces. Before item 2 in the algorithm we will then locally replace $\boldsymbol{X}_A$ with a model matrix that contains crossing with pieces that are present in the actual cluster.

The main algorithm preserves the fitted values (or regression parameters). However, the covariance matrix is partitioned into two terms, $\hat{\boldsymbol{E}}_3^T\hat{\boldsymbol{E}}_3 + \hat{\boldsymbol{E}}_4^T\hat{\boldsymbol{E}}_4$. The last term is preserved at the local level, but the first term is only preserved at the global level. In the special case where $\boldsymbol{X}_B$ is empty, the main algorithm simplifies to MHa and $\hat{\boldsymbol{E}}_3$ vanishes.

When $\boldsymbol{X}_B$ is not empty, we can use modification a), b) or c) to preserve the covariance matrix at the local level. In a) we have a side effect regarding the parameter estimates corresponding to $\boldsymbol{X}_B$. They are preserved globally as required, but within each cluster these parameter estimates have become identical to the global estimate. This may or may not be a required property. Modification b) avoids this property. With many variables in $\boldsymbol{X}_B$ and/or few observations in each cluster, modification a) or b) may not be possible. An extra problem in b) is that the matrix $\boldsymbol{C}$ may not be found. In cases where a) is possible but not b) one may choose c). Within each cluster it is possible, using the method in Section 3, to compute the largest α that makes it possible to find $\boldsymbol{C}$. The final α can be chosen as the smallest of all these values.

Using modification d) we are only changing the input. This method can be a compromise if one is uncertain whether a variable should belong to $\boldsymbol{X}_A$ or $\boldsymbol{X}_B$.

Modification e) is especially useful if we want to preserve means in groups that are too small to be used as clusters. In practice modification e) may also be used in the special case of a single cluster. Then $\boldsymbol{X}_B$ is not needed and the method simplifies to MHb. The covariance matrix is only preserved at the top level.

Modifications d) and e) can be combined and they can also be combined with a), b) or c). All in all, the main algorithm with the possibility of the modifications is a flexible approach for generating hybrid microdata within the context of microaggregation.

We will now exemplify the methodology by discussing scenarios where the general methodology may be used in different ways, including the five modifications. We assume business data where $\boldsymbol{Y}$ consists of sensitive continuous economic variables such as employment expenses and taxes. We have several categorical

non-sensitive variables such as region, industry group and employment size classes. We will refer to the latter categorical variable as K_{ESC}. The data can be subjected to microaggregation with clusters created from the non-sensitive variables. We assume that this is done in such a way that K_{ESC} is unique within each cluster. Synthetic data can be generated by the IPSO method in two ways, MHa and MHb, as described at the beginning of this section. In both cases, $\boldsymbol{X}_B$ is empty. When MHa is used, $\boldsymbol{X}_A$ contains the intercept. The other variant, MHb, means that input to the method is a single cluster and that $\boldsymbol{X}_A$ contains dummy variables corresponding to the microaggregation clustering. The choice between MHa and MHb is a question of whether one wants to reproduce the cluster specific covariances or not. If the variability in some clusters is very low, the synthetic data will be close to the original data in those clusters when MHa is used. When MHa is used, a very high correlation between some sensitive variables in some clusters will also be reproduced. To avoid this property, method MHb may be chosen instead.

Now assume that the exact number of employees, N_{ESC}, is a non-sensitive variable. Then, one objective might be to preserve the correlations (or equivalently regression fits) between N_{ESC} and the sensitive variables. This could be done by including N_{ESC} in $\boldsymbol{X}_A$ even though K_{ESC} is already a variable used within the microaggregation clustering. This way of doing it is still a possibility within MHa since IPSO, in general, can include x-variables (beyond the intercept). This means that the correlations to N_{ESC} will be preserved within each cluster. If one company has a value of N_{ESC} very different from the others in a cluster, the sensitive values of this company will influence the correlations a lot and another method may be preferred. One possibility is to include N_{ESC} in $\boldsymbol{X}_B$ instead. The correlations between N_{ESC} and the sensitive variables will be preserved, but not within each cluster.

A possibility between including N_{ESC} in $\boldsymbol{X}_A$ and including N_{ESC} in $\boldsymbol{X}_B$ is to preserve the correlations within each employment size class (each K_{ESC} category). Since the employment size classes are groups of clusters, this mean that we use modification d). A related method is to use the employment size classes as the clusters within the method and include N_{ESC} in $\boldsymbol{X}_A$. Again the correlations are preserved within the employment size classes. This time covariances of the sensitive variables are preserved within the employment size classes, but not within the original microaggregation clustering. We can still preserve the means within the microaggregation clusters by using modification e). This means that the original microaggregation clusters are cluster pieces within the description of e).

Modifications a), b) and c) are relevant when $\boldsymbol{X}_B$ is in use. This was the case in the example above when N_{ESC} was included directly in $\boldsymbol{X}_B$ and when modification d) was in use. In these cases the standard method does not preserve covariances exactly within each cluster (as required), but this can be achieved by means of a), b) or c). In the case of a) this means that the relations, in terms of regression coefficients, between N_{ESC} and the sensitive variables are made identical in each cluster. The data may seem strange, because identical regression coefficients will never occur in real data. Therefore, the method is probably not preferable and one would prefer b). Since relations are preserved at the global level, there is some information about this in each cluster. When b) is used one can imagine that someone may misunderstand this and may use the data to analyse how relationships differ between clusters. Method a) will prevent someone publishing misleading results about differences. Although, method b) is the preferable choice. But method b) may not be possible and c) is chosen instead.

At the end of this section, we note that an alternative approach is to set all the residuals to zero. If we consider only intercepts in $\boldsymbol{X}_A$ and $\boldsymbol{X}_B$ we obtain ordinary microaggregation. The original values are replaced by means within clusters. Beyond the only intercept cases, we obtain a form of generalized microaggregation which, in some cases, can be a useful alternative to ordinary microaggregation. Based on the above exemplification one could imagine including N_{ESC} in $\boldsymbol{X}_A$ or $\boldsymbol{X}_B$. One reason could be as follows. Suppose many users always divide all numbers by N_{ESC} to obtain per employee data. From their perspective it would be better to divide all data by N_{ESC} before microaggregation so that means per employee are preserved within clusters. Producing two data sets is not an alternative, but including N_{ESC} in the method may be a solution. Means per employee will be closer to the truth at the same time as ordinary means are preserved.

7 Suppressed tabular data

It is possible to combine the IPSO methodology in Section 2 with statistical disclosure control for tabular data. As described in Section 6, $\boldsymbol{X}$ may be composed of dummy variables corresponding to a categorical variable. Sums within the categories will then be preserved. Instead of just a single categorical variable we may have several variables, which can be used to tabulate the data in various ways. When playing around with which dummy variables are to be included in $\boldsymbol{X}$, one is playing around with which sums are to be preserved. Thus, we can include in $\boldsymbol{X}$ only those variables that correspond

Table 1 The example frequency table where cells to be suppressed are marked with *.

	col1	col2	col3	col4	Total
row1	3*	11*	32	30	76
row2	1*	9*	13*	8	31
row3	12	22	2*	2*	38
row4	18*	19	16	3*	56
Total	34	61	63	43	201

to cells that are found to be safe by means of a cell suppression method.

Instead of using microdata as a starting point we can use the sums obtained by crossing all the categorical variables (cover table). This often means that most of the input data are preserved and only some new data are generated (the suppressed cells). In the following, we apply the methodology to cell frequencies as a y-variable. Even if all the original values are counts, we will generate synthetic values that are not whole numbers. As will be discussed, this can be considered as a nice property.

An example of a frequency table is given in Table 1. We assume that values below 4 cannot be published. This means that five cells in the table are primarily suppressed. To prevent the possibility of these values being calculated from other cell values, four additional cells are secondarily suppressed. These additional cells are found using cell suppression methodology. Here we have 16 inner cell frequencies (totals excluded) and 16 publishable cell frequencies (suppressed cells excluded).

Generally, we let $\boldsymbol{Y}$ be $n \times k$ consisting of all the n inner multivariate cell elements in a cover table. Furthermore we let $\boldsymbol{Z}$ be $m \times k$ consisting all the m multivariate elements of the publishable cells.

We focus in particular on the univariate special case ($k = 1$) where the values are frequencies. In Table 1 we have $n = m = 16$.

Obviously $\boldsymbol{Z}$ can be calculated from $\boldsymbol{Y}$ and this can be done via a $n \times m$ dummy matrix $\boldsymbol{X}$:

$$\boldsymbol{Z} = \boldsymbol{X}^T\boldsymbol{Y} \tag{20}$$

We have one column in $\boldsymbol{X}$ for each publishable cell. Each publishable cell is either an inner cell or a sum of several inner cells. In the first case this means that the corresponding column of $\boldsymbol{X}$ has only one element that is one (others are zero). By regressing $\boldsymbol{Y}$ onto $\boldsymbol{X}$ we can calculate fitted values by

$$\hat{\boldsymbol{Y}} = \boldsymbol{X}\left(\boldsymbol{X}^T\boldsymbol{X}\right)^{\dagger}\boldsymbol{X}^T\boldsymbol{Y} = \boldsymbol{X}^{\dagger^T}\boldsymbol{X}^T\boldsymbol{Y} = \boldsymbol{X}^{\dagger^T}\boldsymbol{Z} \tag{21}$$

Since the columns of $\boldsymbol{X}$ are collinear we use the Moore-Penrose generalized inverse here (see Appendix 2). It is clear that the fitted values can be calculated directly from the published cell values. We also have that $\boldsymbol{Z}$ is preserved in the sense that $\boldsymbol{Z} = \boldsymbol{X}^T\hat{\boldsymbol{Y}}$. Many elements of $\boldsymbol{Y}$ are also elements of $\boldsymbol{Z}$ and they are reproduced exactly in $\hat{\boldsymbol{Y}}$. Elements of $\boldsymbol{Y}$ that are suppressed are replaced in $\hat{\boldsymbol{Y}}$ by estimates computed from the publishable cells. Estimates of possible suppressed cells not included in $\boldsymbol{Y}$ can be calculated from $\hat{\boldsymbol{Y}}$. To be more specific, we extend (20) to all elements and not only the publishable ones and let $\boldsymbol{Z}_{\text{All}} = \boldsymbol{X}_{\text{All}}^T\boldsymbol{Y}$. Now we can compute corresponding estimates by

$$\hat{\boldsymbol{Z}}_{\text{All}} = \boldsymbol{X}_{\text{All}}^T\hat{\boldsymbol{Y}} \tag{22}$$

Even though the starting point is different from ordinary regression, synthetic data can be generated by the same method. The IPSO method in Section 2 preserves $\boldsymbol{X}^T\boldsymbol{Y}$ and $\boldsymbol{Y}^T\boldsymbol{Y}$. We can generate synthetic residuals, $\hat{\boldsymbol{E}}^*$, to obtain $\boldsymbol{Y}^* = \hat{\boldsymbol{Y}} + \hat{\boldsymbol{E}}^*$. Then the synthetic variant of $\boldsymbol{Z}_{\text{All}}$ is

$$\boldsymbol{Z}^*_{\text{All}} = \boldsymbol{X}^T_{\text{All}}\boldsymbol{Y}^* \tag{23}$$

It is worth noting that

- The method can be simplified by working with reduced versions of $\boldsymbol{Y}$ and $\boldsymbol{X}$. Rows corresponding to publishable cells can be removed and then redundant columns of $\boldsymbol{X}$ can also be removed.
- If preserving the covariance matrix is not a requirement, the fitted values ($\hat{\boldsymbol{Y}}$ and $\hat{\boldsymbol{Z}}_{\text{All}}$) can be an alternative to synthetic values.

Table 2 The suppressed values of the example frequency table and corresponding generated values.

(row,col)	$\boldsymbol{Y}$	$\boldsymbol{Y}^*$	$\hat{\boldsymbol{Y}}$	$\hat{\boldsymbol{Y}}_{\text{mod4}}$	$\hat{\boldsymbol{Y}}_{\text{mod10}}$
(2,1)	1	11.1562	4.5217	0.8696	4.0870
(3,3)	2	9.3549	6.6957	1.8261	2.7826
(3,4)	2	-5.3549	-2.6957	2.1739	1.2174
(1,1)	3	0.1986	4.1739	2.9565	0.6957
(4,4)	3	10.3549	7.6957	2.8261	3.7826
(2,2)	9	6.1986	10.1739	8.9565	6.6957
(1,2)	11	13.8014	9.8261	11.0435	13.3043
(2,3)	13	5.6451	8.3043	13.1739	12.2174
(4,1)	18	10.6451	13.3043	18.1739	17.2174

Now we will look at the example in Table 1. All the suppressed cells are inner cells and therefore, when considering generated values, we can refer to $\hat{\boldsymbol{Y}}$ and $\boldsymbol{Y}^*$. These values are shown in Table 2. The underlying $\hat{\boldsymbol{E}}^*$ was generated in the ordinary way.

When variance is not important, $\hat{\boldsymbol{Y}}$, may be used. However, these values are quite far from the true values. One may wish for values that are closer to the truth and still safe. One possibility is to do the calculations after a modulo operation. With 10 as the divisor, this means that we only consider the last digit of the suppressed values as unsafe. In our example, this means that we

Table 3 The example frequency table with decimal numbers in place of suppressed values.

	col1	col2	col3	col4	Total
row1	0.4449	13.5551	32	30	76
row2	4.2102	6.4449	12.3449	8	31
row3	12	22	2.6551	1.3449	38
row4	17.3449	19	16	3.6551	56
Total	34	61	63	43	201

replace the largest unsafe values, 11, 13 and 18, by 1, 3 and 8. After the calculations, 10 is added back to these cells. The fitted values obtained in this way are presented in the last column of Table 2. Since the starting point was that all values below 4 were unsafe, one may try 4 as the divisor. However, in this case, this results in model fits that are very close to the truth. In fact, the true value is the closest integer in all cases.

Note that if one combines the modulo operation with ordinary generation of synthetic values, the variance of the original values will not be preserved. A solution is to scale the residuals by a constant to achieve the required variance. Even if correct variance is not important, synthetic residuals may still be added. One reason is to increase the differences from the truth. Another reason is to ensure that none of the generated values are whole numbers. Then, a linear combination of synthetic values using integer coefficients cannot be a whole number unless it is possible to rewrite the combination as a combination of publishable values (whole numbers). Have in mind that the columns $\boldsymbol{X}_{\text{All}}$ are linearly dependent so that linear combinations can be written in several ways.

Table 3 presents the results from a method that may be used in practice. The modulo 10 method is used and the ordinary residuals are downscaled by a factor of 10. In general, if we replace the original suppressed values by values generated in a manner that includes synthetic residuals, we have the following characteristics.

1. The values of all the publishable cells are unchanged. In particular, the values add up correctly to all totals and subtotals. In practice, however, we need to take into account numerical precision error.
2. The values of all the suppressed cells, including possible subtotals, will not be whole numbers. Obtaining a value as close to a whole number as the level of the numerical precision error is very unlikely.
3. If a suppressed cell value is a whole number, this means that the suppression algorithm has failed. This can be verified by re-running the random generation part.
4. Assuming a successful suppression algorithm, any sum of values resulting in a whole number is a true and publishable value.
5. Negative values may be generated.
6. After cell suppression and generation of decimal numbers, only the inner cells may be stored. Various totals can be calculated when needed.
7. The methodology can be used when several variables are involved (not only cell frequency). The ordinary method (no modulo and ordinary residuals) may then be preferred.

As mentioned above, one reason to generate decimal numbers may be to generate values to be stored. They may never be shown and only used technically. After computation of any requested totals, the whole numbers may be shown and other numbers hidden. In practice, a numerical precision limit is needed to define whole numbers. Another reason to generate decimal numbers is to give more information to users. Without using the modulo method there is no extra information in fitted values than is already hidden in the published cells. Methods that assume positive integers and produce lower and upper bounds will provide more precise information about what is hidden in the data. However, presenting intervals is more complicated and precise information may not be the goal. Appropriate information may be achieved by combining the modulo method and scaling of residuals. A third reason to use the approach presented here may be to control the result of the suppression method. Suppressing linked tables can be especially challenging (de Wolf and Giessing 2009), and not all algorithms guarantee protection. When working with linked tables, we need to include all cells from all tables in $\boldsymbol{Z}_{\text{All}}$. A fourth reason to calculate decimal numbers is to permit calculation of arbitrary (user defined) sums of cells afterwards. A sum not included in the suppression method may be publishable even if suppressed cells are involved.

The fitted values contain only information available from publishable cells. When variance is preserved, a piece of information is provided about the suppressed cells and this may be a reason to scale the residuals. One problematic special case to be aware of is when the rank of $\boldsymbol{X}$ is $n-1$. Normally, the generated score vectors of unit length underlying $\hat{\boldsymbol{E}}^*$ are randomly orientated in the subspace orthogonal to the column space of $\boldsymbol{X}$. But in this special case, the dimension of this subspace is one. Thus, given preserved variance, $\pm\hat{\boldsymbol{E}}$ are the only two possible instances of $\hat{\boldsymbol{E}}^*$. This is typically a problem in small example data sets and not in applications. In the example here, the subspace dimension is two and this problem is avoided.

To summarise, without using the modulo method and using residuals scaled in a manner unknown to the user, replacing ordinary suppression (missing value) with decimal numbers releases no extra information. Decimal numbers may, however, be a user-friendly alternative. Switching to modulo should be done with care as the method has not been thoroughly studied.

8 Concluding remarks

This paper has focused on information preserving regression-based methods, which belong to the area of synthetic and hybrid data. Such an approach to protecting microdata is sometimes preferred, but in many applications other methods are used. In the area of statistical disclosure control, insight into the described methodology is useful regardless. Innovative tools are often created by combining aspects of several methods. In the present paper, combined methods have been described in Sections 6 and 7. In both cases, another method is utilized before data are generated by the synthetic data methodology. In the former case, clusters according to microaggregation are made and in the latter case table suppression is performed.

Since a requirement for all the methods treated in this paper is to preserve fitted values, these methods are mostly about how to generate residuals. In some cases, an alternative is to set these residuals at zero. Then we are replacing all the original data with deterministic imputations (in the sense opposed to random imputations). We discussed this approach in Sections 6 and 7.

Regression-based methods have been criticized for not being able to deal with data with outliers in a satisfactory way (Templ and Meindl 2008). This criticism can be met by combining methods. As with other techniques (Templ and Meindl 2008), one can start by dividing the data into outlying and non-outlying observations. Thereafter, one proceeds by generating data within each group. One may use methodology described in this paper in either group, or only in the non-outlying observations group. In the case of both groups, we can say that such methodology is already covered by the general description in Section 6.

The contribution of this paper has mainly been to describe and develop methods in a concise and interpretable way. Several questions arise concerning the performance of the methods and which method to prefer in specific situations. A wide-ranging comparison of the methods is beyond the scope of the present paper, but studies and performance comparisons can be found in the underlying references. The characteristics of the methods are different and the preferred choice would depend on the situation.

However, if we limit the discussion to non-combined methods (Sections 2-5), we can make some reflections. The IPSO method is the basic method to be used when the aim is only to preserve variances, covariances and fitted values. Technically, performing the computations via QR decomposition is faster than using SVD. Methodology that controls the relationship with the original data by means of a single parameter is easy to understand. The IPSO method and the original data are thus the two extremes. One such method is obtained with all diagonal elements equal in (7) or (8). These two equations are then equivalent. This method is elegant and exact control is obtained. Furthermore, this method is a special case of the one described in Muralidhar and Sarathy (2008). An easy and efficient implementation is obtained by (8) and QR decomposition. It is also easy adjustable, so that the relationship with some x-variables can be controlled better. However, the methodology in Section 3 makes use of new scores generated under certain orthogonality restrictions. In cases with many variables and few observations this may be impossible. The single parameter ROMM method may then be an alternative. This method is efficiently implemented by (11), (12) and (17) which utilize the theory in Sections 4 and 5.

Appendix 1: Generalized QR decomposition

The QR decomposition of a $n \times m$ matrix $\boldsymbol{A}$ with rank r can be written as

$$\boldsymbol{A} = \boldsymbol{Q}\boldsymbol{R} \tag{24}$$

where $\boldsymbol{Q}$ is a $n \times r$ matrix whose columns form an orthonormal basis for the column space of $\boldsymbol{A}$. This decomposition can be viewed as the matrix formulation of the Gram-Schmidt orthogonalization process. The Cholesky decomposition of $\boldsymbol{A}^T\boldsymbol{A}$ can be read from the QR decomposition of $\boldsymbol{A}$ as $\boldsymbol{R}^T\boldsymbol{R}$.

In this paper, in order to allow linearly dependent columns of $\boldsymbol{A}$ ($r < m$), we refer to a generalized variant of QR decomposition. In such cases a usual decomposition (Chan 1987) is

$$\boldsymbol{A}\tilde{\boldsymbol{P}} = \boldsymbol{Q}\tilde{\boldsymbol{R}} \tag{25}$$

where $\tilde{\boldsymbol{P}}$ is a permutation matrix that reorders the columns (pivoting) in order to make a decomposition so that $\tilde{\boldsymbol{R}}$ is upper triangular.

To make the decomposition unique, we require the diagonal entries of $\tilde{\boldsymbol{R}}$ to be positive. Furthermore, we require $\tilde{\boldsymbol{P}}$ to keep the order of the columns as close to the original order as possible (minimal pivoting). We

now have $\boldsymbol{A} = \boldsymbol{Q}\tilde{\boldsymbol{R}}\tilde{\boldsymbol{P}}^T$ and in generalized QR decomposition (24) we use

$$\boldsymbol{R} = \tilde{\boldsymbol{R}}\tilde{\boldsymbol{P}}^T \tag{26}$$

The QR decomposition of a composite matrix can be written as

$$[\boldsymbol{A}_1\ \boldsymbol{A}_2] = [\boldsymbol{Q}_1\ \boldsymbol{Q}_2]\left[\boldsymbol{R}_1^T\ \boldsymbol{R}_2^T\right]^T \tag{27}$$

Now $\boldsymbol{Q}_1$ can be computed by QR decomposition of $\boldsymbol{A}_1$. The matrix $\boldsymbol{Q}_2$ can be computed by QR decomposition of $\boldsymbol{A}_2 - \boldsymbol{Q}_1\boldsymbol{Q}_1^T\boldsymbol{A}_2$, which is the residual part after regressing $\boldsymbol{A}_2$ onto $\boldsymbol{A}_1$.

Appendix 2: The singular value decomposition

The singular value decomposition (SVD) of a $n \times m$ matrix $\boldsymbol{A}$ with rank r can be written as

$$\boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{V}^T \tag{28}$$

where $\boldsymbol{\Lambda}$ is a $r \times r$ diagonal matrix of strictly positive singular values in descending order. This is the rank-revealing version of the decomposition (Demmel et al. 1999). Other variants of SVD allow some singular values to be zero, but these can be omitted. The columns of $\boldsymbol{U}$ form an orthonormal basis for the column space of $\boldsymbol{A}$ and the columns of $\boldsymbol{V}$ form an orthonormal basis for the row space.

The singular values are the square root of the eigenvalues of of $\boldsymbol{A}^T\boldsymbol{A}$ and $\boldsymbol{A}\boldsymbol{A}^T$. The eigen decompositions of these two symmetric matrices can be read directly from the SVD of $\boldsymbol{A}$ as $\boldsymbol{V}\boldsymbol{\Lambda}^2\boldsymbol{V}^T$ and $\boldsymbol{U}\boldsymbol{\Lambda}^2\boldsymbol{U}^T$. It is also worth mentioning that an alternative to the ordinary Cholesky decomposition, $\boldsymbol{A}^T\boldsymbol{A} = \boldsymbol{R}^T\boldsymbol{R}$, is to let $\boldsymbol{\Lambda}\boldsymbol{V}^T$ play the role of $\boldsymbol{R}$.

To make the SVD unique we can require all column sums of $\boldsymbol{V}$ to be positive. In cases with equal singular values, the decomposition is not unique regardless.

There is a close relationship between SVD and PCA. In PCA, the variables are usually centered to zero means, and in many cases standardized to equal variances prior to decomposition. If $\boldsymbol{A}$ is such a centered/standardized matrix then $\boldsymbol{U}\boldsymbol{\Lambda}$ is the matrix of PCA scores and $\boldsymbol{V}$ is the matrix of PCA loadings.

The Moore-Penrose generalized inverse of $\boldsymbol{A}$ can be written as

$$\boldsymbol{A}^\dagger = \boldsymbol{V}\boldsymbol{\Lambda}^{-1}\boldsymbol{U}^T \tag{29}$$

We have

$$\boldsymbol{A}^\dagger = (\boldsymbol{A}^T\boldsymbol{A})^\dagger\boldsymbol{A}^T = \boldsymbol{A}^T(\boldsymbol{A}\boldsymbol{A}^T)^\dagger \tag{30}$$

When $\boldsymbol{A}$ is invertible, $\boldsymbol{A}^\dagger = \boldsymbol{A}^{-1}$. When $\boldsymbol{A}^T\boldsymbol{A}$ or $\boldsymbol{A}\boldsymbol{A}^T$ is invertible this means, respectively, that $\boldsymbol{A}^\dagger = (\boldsymbol{A}^T\boldsymbol{A})^{-1}\boldsymbol{A}^T$ or $\boldsymbol{A}^\dagger = \boldsymbol{A}^T(\boldsymbol{A}\boldsymbol{A}^T)^{-1}$.

References

Benedetto, G., Stinson, M.H., Abowd, J.M.: The Creation and Use of the SIPP Synthetic Beta. Tech. rep., United States Census Bureau (2013)

Burridge, J.: Information preserving statistical obfuscation. Stat. Comput. **13**(4), 321–327 (2003). doi: 10.1023/A:1025658621216

Calvino, A.: A Simple Method for Limiting Disclosure in Continuous Microdata Based on Principal Component Analysis. J. Off. Stat. **33**(1), 15–41 (2017). doi:10.1515/JOS-2017-0002

Chan, T.F.: Rank revealing QR factorizations. Linear Alg. Appl. **88-9**, 67–82 (1987). doi:10.1016/0024-3795(87)90103-0

de Wolf, P.P., Giessing, S.: Adjusting the tau-ARGUS modular approach to deal with linked tables. Data Knowl. Eng. **68**(11), 1160–1174 (2009). doi: 10.1016/j.datak.2009.06.005

Demmel, J., Gu, M., Eisenstat, S., Slapnicar, I., Veselic, K., Drmac, Z.: Computing the singular value decomposition with high relative accuracy. Linear Alg. Appl. **299**(1-3), 21–80 (1999). doi:10.1016/S0024-3795(99)00134-2

Domingo-Ferrer, J., Gonzalez-Nicolas, U.: Hybrid microdata using microaggregation. Inf. Sci. **180**(15), 2834–2844 (2010). doi:10.1016/j.ins.2010.04.005

Domingo-Ferrer, J., Mateo-Sanz, J.M.: Practical data-oriented microaggregation for statistical disclosure control. IEEE Transactions on Knowledge and Data Engineering **14**(1), 189–201 (2002)

Drechsler, J.: Synthetic Datasets for Statistical Disclosure Control. Springer, New York (2011)

Duncan, G.T., Pearson, R.W.: Enhancing Access to Microdata While Protecting Confidentiality: Prospects for the Future. Stat. Sci. **6**(3), 219–239 (1991)

Hundepool, A., de Wolf, P.P., Bakker, J., Reedijk, A., Franconi, L., Polettini, S., Capobianchi, A., Domingo, J.: mu-ARGUS user's manual, version 5.1. Tech. rep., Statistics Netherlands (2014)

Hundepool, A., Domingo-Ferrer, J., Franconi, L., Giessing, S., Nordholt, E.S., Spicer, K., de Wolf, P.P.: Statistical Disclosure Control. John Wiley & Sons, Ltd (2012). doi: 10.1002/9781118348239.ch1

Jarmin, R.S., Louis, T.A., Miranda, J.: Expanding the role of synthetic data at the U.S. Census Bureau. Statistical Journal of the IAOS **30**(1-3), 117–121 (2014)

Jolliffe, I.: Principal Component Analysis, Second Edition. Springer, New York (2002)

Klein, M.D., Datta, G.S.: Statistical disclosure control via sufficiency under the multiple linear regression model. Journal of Statistical Theory and Practice **12**(1), 100–110 (2018)

Langsrud, Ø.: Rotation tests. Stat. Comput. **15**(1), 53–60 (2005). doi:10.1007/s11222-005-4789-5

Loong, B., Rubin, D.B.: Multiply-Imputed Synthetic Data: Advice to the Imputer. J. Off. Stat. **33**(4), 1005–1019 (2017). doi:10.1515/JOS-2017-0047

Mateo-Sanz, J., Martinez-Balleste, A., Domingo-Ferrer, J.: Fast generation of accurate synthetic microdata. In: DomingoFerrer, J and Torra, V (ed.) Privacy in Statistical Databases, Proceedings, vol. 3050, pp. 298–306 (2004). Conference on Privacy in Statistical DataBases (PSD 2004), Barcelona, Spain, June 09-11, 2004.

Muralidhar, K., Sarathy, R.: Generating Sufficiency-based Non-synthetic Perturbed Data. Trans. Data Priv. **1**(1), 17–33 (2008)

Reiter, J.P., Raghunathan, T.E.: The multiple adaptations of multiple imputation. J. Am. Stat. Assoc. **102**(480), 1462–1471 (2007). doi:10.1198/016214507000000932

Salazar-Gonzalez, J.J.: Statistical confidentiality: Optimization techniques to protect tables. Comput. Oper. Res. **35**(5), 1638–1651 (2008). doi:10.1016/j.cor.2006.09.007

Strang, G.: Linear Algebra and Its Applications, 3rd edition. Harcourt Brace Jovanovich, San Diego (1988)

Templ, M., Kowarik, A., Meindl, B.: Statistical Disclosure Control for Micro-Data Using the R Package sdcMicro. J. Stat. Softw. **67**(4) (2015)

Templ, M., Meindl, B.: Robustification of microdata masking methods and the comparison with existing methods. In: Domingo-Ferrer, J., Saygın, Y. (eds.) Privacy in Statistical Databases, Proceedings, pp. 113–126. Springer Berlin Heidelberg, Berlin, Heidelberg (2008). UNESCO Chair in Data Privacy International Conference (PSD 2008), Istanbul, Turkey, September 24-26, 2008.

Ting, D., Fienberg, S.E., Trottini, M.: Random orthogonal matrix masking methodology for microdata release. International Journal of Information and Computer Security **2**(1), 86–105 (2008). doi:10.1504/IJICS.2008.016823

Wedderburn, R.W.M.: Random Rotations and Multivariate Normal Simulation. Research Report, Rothamsted Experimental Station (1975)