IN3 Working Paper Series
INTERNET INTERDISCIPLINARY INSTITUTE

ISSN 2013-8644

Triangulation-Based Multivariate Microaggregation

Juvenal Machín Casañas (jmachinc@uoc.edu)
Open University of Catalonia

Agustí Solanas Gómez (agusti.solanas@urv.cat)
Smart Health Research Group
Dept. Computer Engineering and Mathematics
Rovira i Virgili University

Working Paper

Working Paper Series WP00-000

Research group: Name
Research group coordinator: Name and Surname(s) (Affiliation)

Submitted in: month yyyy
Accepted in: month yyyy
Published in: month yyyy

Universitat Oberta de Catalonia
IN3
INTERNET INTERDISCIPLINARY INSTITUTE
http://in3.uoc.edu

Internet Interdisciplinary Institute (IN3)
http://www.in3.uoc.edu
Parc Mediterrani de la Tecnologia
Av. Carl Friedrich Gauss, 5
08860 Castelldefels
Barcelona (Espanya)
Tel. 93 4505200

Universitat Oberta de Catalunya (UOC)
http://www.uoc.edu/
Av. Tibidabo, 39-43
08035 Barcelona
Espanya
Tel. 93 253 23 00

Table of contents

Triangulation-Based Multivariate Microaggregation

Juvenal Machín Casañas (jmachinc@uoc.edu)

Open University of Catalonia (UOC)

Agustí Solanas Gómez (agusti.solanas@urv.cat)

Smart Health Research Group
Dept. Computer Engineering and Mathematics
Rovira i Virgili University

Recommended citation:

MACHIN, Juvenal; SOLANAS, Agusti (2016). “Triangulation-Based Multivariate Microaggregation” [online working paper]. (Working Paper Series; WP00-000). IN3 Working Paper Series. IN3 (UOC). [Accessed: dd/mm/yy].
<url>

Abstract

Microaggregation is a Statistical Disclosure Control technique in which similar records are clustered into groups containing a minimum of k records that are later replaced by group centroids, so that released data preserve some of their statistical properties while reducing the risk of re-identification. A fixed-size microaggregation method clusters data into groups of size *k* except perhaps one group with size between *k* and 2*k* -1, whereas a data-oriented (variable-size) method allows group size to vary between *k* and 2*k* -1.

Heuristic clustering methods are needed since the minimum information loss microaggregation problem is NP-hard (Oganian and Domingo-Ferrer, 2001).

In this paper we studied various microaggregation methods in the literature and we have proposed a new heuristic approach for multivariate fixed-size microaggregation based on the triangulation of a set of points in $\mathbb{R}^2$. A reference data set and a random generated one are used to compare the method outcomes, in terms of information loss, with other previous proposals and the results summarized.

Keywords
microaggregation, microdata protection, Statistical disclosure control, privacy, triangulation.

Introduction

Principle 6 of the United Nations Economic Commission Report Fundamental Principles for Official Statistics states that “Individual data collected by statistical agencies for statistical compilation, whether they refer to natural or legal persons, are to be strictly confidential and used exclusively for statistical purposes” (UN Statistical Commission, 1994, p. 2). So microdata - information at the level of individual respondents – should never be released to the public.

Summarizing data to an aggregate level is the fastest way to protect individual privacy; however, aggregated data is not a useful tool to do research - i.e. to explore a relationship between two variables – because of the information loss. So access to certain level of detail must be provided in order to do some practical inference. Plus, with increased computing power and new inference and processing techniques like big data and machine learning to identify individual respondents by means of their confidential microdata, it’s becoming easier than ever. So privacy risks are quickly rising in a serious way.

Statistical Disclosure Control (SDC) seeks to protect data so that sensitive information cannot be linked to specific individuals, thus allowing the information to be released without any privacy issues. SDC techniques must be applied to microdata sets prior releasing, either by masking data or generating new *synthetic* data, to avoid leaking of confidential information. But the data released must preserve its value as an analytical resource, so the information loss must be minimized during the process.

This research has been conducted in order to study the application of the triangulation of a set of points problem to the microaggregation of two-dimensional numerical data. Its main goal is to determine if there exists some point set triangulation-based method which satisfies the constraints imposed by microaggregation. And, in case a suitable method was found, a comparative study would be carried out to compare its outcomes with those of other well-known methods.

1. State of the art.

Microaggregation (Defays and Anwar, 1995) is a masking perturbative SDC technique for microdata. The main goal of microaggregation is to achieve a clustering of numerical data so that there is an upper bound for the size of groups, *k* that maximizes intra-group homogeneity. Then, microdata are replaced by their corresponding cluster centroids.

In the context of information security, microaggregation can be used as a technique for achieving Statistical Disclosure Control in microdata because it provides a *k-anonymized* version of the microdata set that is suitable for subsequent public releases while preserving data utility (minimizing information loss), so that the altered dataset can still be used for scientific or statistical research.

In microaggregation, each group has at least *k* data points. This characteristic provides the *k-anonymity* of the data been released. A *fixed-size* method divides data into groups of size k except perhaps one group with size between *k* and 2*k* -1, whereas a *data-oriented* (variable-size) method allows group sizes to vary between *k* and 2*k* - 1. While fixed-size methods tend to be more computationally efficient, data-oriented methods are more flexible and can adapt better to different data distributions and thus achieve a lower information loss than the former.

Given a parameter *k*, any optimal microaggregation has minimum information loss for that *k*. All the groups into an optimal microaggregation have at most 2*k* - 1 records, since each group with size 2*k* can be partitioned into smaller groups in order to further reduce information loss (Domingo-Ferrer and Mateo-Sanz, 2002).

Regarding the metrics to evaluate the quality of the methods, an approach has been presented in (Domingo-Ferrer and Torra, 2001; Sebé et al., 2002; Nin, Herranz and Torra, 2008) where both information loss (*IL*) and disclosure risk (*DR*) are combined into a score: *score = 0.5 IL + 0.5 DR.*

An optimal microaggregation method must minimize the information loss resulting from this replacement process because lower *IL* means less distortion and hence more utility of the data. But a trade-off between the information loss and the disclosure risk is needed.

On univariate data, a polynomial-time optimal microaggregation algorithm is given in (Hansen and Mukherjee, 2003) but, for multivariate data, the optimal microaggregation was proved to be an NP-hard problem (Oganian and Domingo-Ferrer, 2001). It cannot be solved in polynomial time, so several heuristic methods have been proposed in the literature to approximate the results.

Given a set of points *P* in the Euclidean plane, a ***triangulation*** of the set is a breakdown of its convex hull into triangles whose vertices belong to *P*, with all the points of *P* are vertices of its triangulations, and so that each pair of triangles has its inner disjoint. We will focus on two well-known triangulations:

A ***Delaunay triangulation*** for a set of points is a triangulation such that satisfies the *empty sphere test* geometric criterion (Delaunay, 1934): the circumcircle of each triangle does not contain any other point in its interior. This condition ensures that the inner minimum angle of the triangles is maximized. The Delaunay triangulation is the dual graph of the Voronoi diagram (Voronoï, 1908) and contains $O(n^{\lceil d / 2 \rceil})$ simplices (Seidel, 1995). The closest neighbor *b* to any point *p* is on an edge *bp* in the Delaunay triangulation since the *Euclidean Minimum Spanning Tree* (tree of minimum total length whose vertices are the given points) is a subgraph of the Delaunay Triangulation (Preparata and Shamos, 1985).

On the other hand, ***Minimum-Weight Triangulation*** (MWT), also called *Optimal Triangulation* or *Minimum Length Triangulation* problem, is the computational geometry optimization problem of finding, for a set of points, a triangulation that has minimal edge length. It has been shown that the MWT is an NP-hard type problem (Mulzer and Rote, 2008), although Remy and Steger showed an approximation scheme for the MWT where, for any constant $\varepsilon > 0$, a triangulation achieving the approximation ratio of $1 + \varepsilon$ can be computed in a quasi-polynomial time (Remy and Steger, 2009).

The Delaunay triangulation has an approximation ratio of Θ(n) to MWT (Kirkpatrick, 1980) and the greedy triangulation is known to have an approximation ratio of Θ(√n) (Levcopoulos and Krznaric, 1998).

The next section will offer a more technical review in the context of the state of the art in both multivariate microaggregation and triangulation methods.

1.1. Multivariate microaggregation methods

In 1998, Mateo Sanz and Domingo Ferrer proposed the first multivariate (fixed size) microaggregation method, called ***MD*** (*maximum distance*) method, where the grouping process is applied to subsets of variables of the microdata set: “The idea is to form groups of size *k without* projecting multivariate data in one-dimension. Instead, a multivariate distance is used” (Mateo-Sanz and Domingo-Ferrer, 1998, p. 518). This method was later improved in (Domingo-Ferrer et al., 2006) with a data-oriented version called *MD-MHM*, which uses an adaptation of Hansen-Mukherjee algorithm in (Hansen and Mukherjee, 2003). In 2003, a fixed-size method called ***MDAV*** - *Maximum distance to average vector* – was first proposed in (Hunderpool, et al., 2003) and implemented in the *µ-Argus* package for statistical disclosure control. This method would be later modified in (Laszlo and Mukherjee, 2005) with the name *Centroid-based fixed size microaggregation (CBFS)* and finally improved by (Domingo-Ferrer et al., 2006) with a data-oriented version which uses a multivariate version of the Hansen-Mukherjee algorithm (Hansen and Mukherjee, 2003), called *MDAV-MHM*. MDAV has become a usual reference for multivariate microaggregation methods (Solanas, 2008; Lin et al., 2010; Mortazavi, Jalili and Goharzagi, 2013). It works by first computing a square matrix of distances between all records. Then, for every iteration, the average vector of the unassigned records, *c*, is computed and two clusters of *k* records will be grown: one from the farthest record (*r*) from *c* and one from the farthest record (*s*) from *r.* Finally, the remaining records are assigned to their closest group.

An MDAV-based data-oriented method has been proposed in (Solanas, Martínez-Ballesté, and Domingo-Ferrrer, 2006) with the name **V-MDAV** (*Variable-size Maximum Distance to Average Vector*), which generates k-partitions with group sizes varying between *k* and 2*k*-1 and thus with higher within-group homogeneity. This implies improved flexibility, which can adapt well to poorly homogeneous datasets. The method uses a *gain factor* that has to be tuned "in order to improve the adaptability of V-MDAV" (Solanas and Martínez-Ballesté, 2006, p. 5) depending on the data distribution. A multivariate data-oriented microaggregation method based on V-MDAV has been proposed in (Chettri, Paul and Dutta, 2013), with the name **CV-MDAV** - *Centroid based Variable size Maximum Distance to Average Vector*. The *CV-MDAV* algorithm iterates as long as at least 3*k* records remain unassigned. For each iteration, the algorithm computes the centroid of the remaining records in the dataset and the farthest record from it, x_r*. T*hen it finds the 2*k* nearest neighbours of x_r. Current cluster, c_i is formed with the first (*k-1*) neighbours of x_r. Each of the other y_j neighbours is tested for inclusion in the currently formed cluster by computing a heuristic. This algorithm also uses a constant gain factor, *γ*, in the heuristic to *conservatively* expand the cluster. The heuristic compares two distances: the distance from y_j to the cluster centroid (d_2) and the distance from y_j to the centroid of its k-nearest neighbours (d_3). If d_2 is less than d_3 ($d_2 < \gamma d_3$) it expands the current cluster and re-computes its centroid. This test is repeated for the remaining $y_{2k\text{-}1}$ records to be included in cluster c_i so the cluster is expanded as long as it has less than 2*k*-1 records in it. When there are less than 3k records, if there is more than 2k, it will form a cluster around x_r with its nearest k-1 neighbours. And finally, a new cluster with the remaining records.

A two-stage fixed-size method, **TFRP** (*Two Fixed Reference Points*)*,* has been proposed in (Chang, Li and Huang, 2007). "In the first phase, TFRP uses a novel fixed-size algorithm to shorten the running time efficiently. In the second phase, TFRP reduces the number of groups generated by the first phase to improve the data quality". (Chang, Li and Huang, 2007, p. 1868). This second phase is intended to reduce the information loss, as the within group sum of squares (*SSE*) of the resulting groups in phase 1 is high. After applying the *TFRP*-2, if several groups contains greater than or equal to 2*k* records then the groups are broken down using any fixed-size microaggregation method. And in case the size of the closest group to which a vector x_i has to be assigned is (4*k*-1) records then the vector x_i is assigned to its second closest group. Another two-stage method called ***DBA*** (*Density-Based Algorithm*) has been proposed in (Lin et al., 2010). This microaggregation method works with the concepts of *k-neighborhood* of a record *x, $N^k(x,T)$*, defined as the set containing *x* and the *k* -1 nearest records to *x*, and the *k-density* of *x, $d^K(x,T)$*, defined as the inverse of the sum of Euclidean distance from each record in *$N^k(x,T)$* to the centroid of *$N^k(x,T)$.* The first stage partitions a data set into groups using the *k-neighborhood* of the record with the highest *k-density* among all the records still unassigned to any group, until less than *k* records remain unassigned. Then, these remaining records are assigned to their respective nearest groups. The second stage will try to fine tune the results in order to achieve low information loss by decomposing or merging the groups in accordance with the information loss been measured. If, at the end, few groups end up having more than 2*k*-1 records, then it applies MDAV algorithm to each group with size greater than 2*k*-1. This second stage is similar to the second stage in TFRP, but *TFRP*-2 disallows merging a record to a group of size over 4*k* -1.

A **Genetic Algorithm** for solving the microaggregation problem has been proposed in (Solanas, 2008). K-partitions are represented as strings (chromosomes) of length N = number of records, where each gene stores the cluster number to which that particular record is assigned to. It uses a *Fitness* function to evaluate the chromosome in the population, $Fitness = \frac{1}{SSE+1}$, thus giving the level of homogeneity of the groups in the *k*-partition represented by a given chromosome. As operator, the method uses one-point crossover and mutation. In case of large data sets, the performance of this method decreases. So, a hybrid approach method has also been proposed in (Solanas, 2008) which takes the advantage of both *MDAV* and classic GA by mixing them and produces better result in terms of SSE.

A fixed-size multivariate microaggregation method has been proposed in (Kokolakis and Fouskakis, 2009), named **IP** (*Importance Partitioning*), which basically works by iteratively building a group of *k* points around the most distant point from the unassigned data set total mean.

An **iterative optimization** method has been proposed in (Mortazavi, Jalili and Gohargazi, 2013). This method, called **IMHM** (*Iterative MHM-based microaggregation*), is based on the optimal univariate microaggregation algorithm *MHM* proposed in (Hansen and Mukherjee, 2003) and focuses on reducing the information loss. It builds the clusters using an iterative optimization method which, for each iteration, reduces SSE after calculating the assignment to centroids. It uses an improved MHM to avoid the local optimum. IMHM strategy is to reformulate the microaggregation problem as a Linear Program that the algorithm will try to minimize. The microaggregation problem is formulated as minimizing $\sum_{i=1}^{c}\sum_{j=1}^{n} w_{ij} \cdot b_{ij}$[1], with a couple of constraints regarding the

[1] The assignment of the *j*-th record to the *i*-th cluster is denoted by b_{ij} = 1 and the cost of b_{ij} is denoted by w_{ij} = || C_i – X_j ||, where C_i denotes the i-th cluster centroid and X_j is the j-th record. $b_{ij} \in \{0,1\}$

point-to-cluster assignations (total sum of b_{ij} must be equal to 1, to satisfy that all records are assigned to exactly one cluster) and the intrinsic constraints regarding data privacy and utility of microaggregation ($k \leq \sum_{j=1}^{n} b_{ij} < 2k$).

An $O(n^2 \log n)$ time method based on the **sequential minimization** of *SSE* has been proposed in (Panagiotakis and Tziritas, 2013) with the name **GSMS** (G*roup selection based on sequential minimization of SSE)*. According to this method, each point of the data set is a cluster "centre" candidate. The corresponding cluster is defined by the *k*−1 closest records to the centre. The method consists of two phases: In the first phase, the candidate cluster that minimizes the current SSE of the remaining data gets discarded in every iteration. Finally, in the second phase the remaining records are assigned to their closest cluster. In order to fast compute the closest records to the "centre" of a candidate cluster, the method uses *n* priority queues, one for each point of the data set. Authors have also presented an improved version of the algorithm, called *GSMS-T2*, by applying the Phase II of *TFRP* algorithm, as seen in (Chang, Li and Huang, 2007).

In 2014, a new approach based on the well-known NP-hard combinatorial optimization problem ***the travelling salesman problem*** (TSP) has been introduced in (Mortazavi and Jalili, 2014), with the name **FDM** (*Fast Data-oriented Microaggregation*). This TSP-based variable-size method is intended to achieve resource-efficient multivariate microaggregation method for large numerical data sets and it can produce multiple protected versions of a data set within a single load. FDM provides two approximation parameters that enable the data publisher to select a desired trade-off between data quality (in terms of information loss) and execution time. Main idea is to sequence data records in a TSP tour and then applying an adapted MHM algorithm to produce optimal partitioning in terms of SSE with respect to that tour. The tour construction algorithm uses an improved version of the *savings heuristics* proposed in (Clarke and Wright, 1964) by means of a heap data structure and an optimized sorting method that "sorts all records based on their distances to the hub node and only calculates the savings of the node pairs that their end point distances to the hub are at least half of the current maximum savings" (Mortazavi and Jalili, 2014, p. 198), so not all pairs need to be considered during tour construction. In fact, only the nearest *active*[2] neighbours are considered for pairings. If the neighbour of a node is inactivated during execution, the neighbourhood will be updated with the next nearest active node. The improved version of MHM is based on the optimal MHM proposed in (Hansen and Mukherjee, 2003), modified to efficiently partitions a sequence of records in a tour in $O(nk^2)$ time. Another TSP-based method has been proposed in (Maya and Solanas, 2015). It is a fixed-size microaggregation method in

[2] A node x_i, $i \neq 1$ is active, if its degree in the partial tour is less than 2. (Mortazavi and Jalili, 2014, p. 198).

which, following the TSP analogy, each record is represented by a city whose attributes are the position of the city in the graph.

A data-oriented microaggregation O(n^2) time method has been proposed in (Laszlo and Mukherjee, 2015) with the name **ILS** (*Iterated Local Search*). It uses a **local search** which finds a local minimum, inside an iterated local search algorithm to find the final solution. Given *P*, the collection of all *k*-partitions for fixed *k*, they define the *neighbourhood* of P as *N(P)*: $P \rightarrow 2^P$ formed by all the partitions that can be obtained either by shifting a point from some cluster *C* in *P* to another cluster in *P*, where |*C*| >*k* - *shift* - or by exchanging a pair of points between two clusters in *P* - *swap*. LS starts with a valid k-partition P and iteratively generates (*updates*) new feasible partitions from *N(P)* with monotonically decreasing information loss (lower SSE) until convergence, which occurs when the update step fails to improve the partition, that is: it can't find any lower SSE partition performing any update operations.

This update is defined in terms of the *swap* and *shift* operations seen before. Both operations are defined to satisfy the cluster size constraints but also to preserve the number of clusters in the partition.

A **tree-decomposition** approach has been proposed in (Panagiotakis and Tziritas, 2015). The algorithm, called **HTEPM** (*Hierarchical Tree Equi-Partition for Microaggregation*) is an adaptation of the HTEP algorithm in (Panagiotakis, Grinias and Tziritas, 2011) with a cardinality constraint so that the microaggregation conditions are satisfied. It is an O(N^2) multivariate micro-aggregation method and works by considering that each group is equivalent to a sub-tree, which is then iteratively evaluated and the one with the highest score (SSE) is split into two sub-trees, resulting in a hierarchical forest of trees with *almost* equal score.

Despite of the numerous microaggregation methods proposed in the literature, we haven't found any triangulation-based microaggregation method.

1.2. Triangulation methods

1.2.1. Delaunay triangulation.

The most straightforward method, ***triangle-flipping***, works by computing an arbitrary triangulation of the points in *P* and then gradually altering it by flipping edges of the triangles until no triangle is non-Delaunay. This approach is historically due to Lawson (Lawson, 1977). Triangle flipping requires O(n^2) flips in a worst-case scenario (De Berg et al., 2008).

Another approach, ***Incremental construction***, has been proposed in (Mc Lain, 1976). It builds the triangulation by successively generating simplices whose circumhyperspheres contain no points in *P.* Another approach, called ***on-line*** or ***incremental insertion***, is proposed in (Guibas, Knuth and Sharir, 1990; Edelsbrunner and Shah, 1992; Su and Drysdale, 1995). These methods are based on the results in (Joe, 1989, 1991). Starting with a simplex which contains the convex hull of the point set, these algorithms repeatedly partition the simplex by adding one vertex at a time to the triangulation. The circumsphere criterion is then recursively tested on all the simplices adjacent to the new ones and, if necessary, their faces are flipped retriangulating only the part of the graph affected by the addition of the point. The Bowyer-Watson algorithm (Bowyer, 1981; Watson, 1981) provides a non-flipping alternative by deleting, after every insertion, any triangles whose circumcircles contain the new point. This operation leaves a star-shaped polygonal hole which is then re-triangulated using the new point.

A ***Divide and Conquer (D&C)*** approach to solve the Delaunay triangulation in the two dimension case was first proposed in (Lee and Schachter, 1980), improved in (Guibas and Stolfi, 1985) and later in (Dwyer, 1987) with an O(*n* log log *n*) time algorithm. A modified D&C approach is used in (Cignoni, Montani and Scopigno, 1998) to perform a triangulation in *d* dimensions to solve the E^d case. This approach works by recursively splitting the points into two sets, computing Delaunay triangulation for each set and finally merging the sets along the splitting lines. D&C has been shown to be the fastest way of computing the Delaunay triangulation (Su and Drysdale, 1995).

A hybrid algorithm, ***Sweep-hull***, proposed in (Sinclair, 2010) uses a radially propagating sweep-hull, generated from a radially sorted set of points in two dimensions. This, coupled with a final triangle flipping step provides the Delaunay triangulation for the set of points.

1.2.2. Minimum-Weight Triangulation.

Due to its complexity and the difficulty of finding an exact solution, many heuristics have been proposed in the literature in order to approximate the solution, generally focusing on finding a good subgraph of the MWT.

β-skeletons are defined by Kirkpatrick and Radke in (Kirkpatrick and Radke, 1985) to describe the *shape* of a set of points. They are proximity graphs whose region of influence is modulated by a parameter, *β*. It has been shown that the √2-skeleton of *S* is a subgraph of the minimum weight triangulation of *S* (Keil, 1994). This result was later improved by Cheng and Xu who proved that, for β > 1/ sin k (with k ≈ π/3.1), the β-skeleton of *S* is a subgraph of a minimum weight triangulation of *S* (Cheng and Xu, 2001). Using **circle-based *β*-skeletons**, an $O(n^{k+2})$ exhaustive search algorithm is given in (Cheng, Golin and Tsang, 1995) to compute the MWT of *n* points in the plane, where *k* is the number of connected components in the planar graph - without edge crossings - consisting of the convex hull and the β-skeletons of *S*. Later, Shiyan Hu introduced *one-sided β-skeleton* and gave an algorithm for identifying subgraphs of the MWT using the one-sided ($\sqrt{2}\beta$)-skeleton (Hu, 2009). Another approach, based on the *locally minimal skeleton*, or *LMT-skeleton* for short, has been proposed in (Dickerson and Montague, 1996). This method computes the LMT subgraph of the MWT in $O(n^4)$ time, $O(n^3)$ space. This subgraph, *usually* connected, contains a set of edges that must be in every locally minimal triangulation. The remaining *untriangulated* space are simple polygons. But It has been shown that, on the average, for large data sets the number of components is linear and, therefore, the LMT-skeleton does not provide enough information in order to compute the MWT of a particular point set in polynomial time (Bose, Devroye and Evans, 2002).

A **genetic algorithm** for the MWT has been proposed in (Qin, Wang and Gong, 1997), with the name *Genetic Minimum Weight Triangulation* (*GMWT*). This method encodes a solution (string) as a lower triangular matrix M in which the element M_{ij} is 1 if the edge between points *i* and *j* is selected in the triangulation (otherwise M_{ij}=0). The fitness function is $f_i(M) = \sum_{i=0}^{n} \sum_{j=0}^{i} L_{ij}(1 - M_{ij})$, where L_{ij} is the distance between P_i and P_j. As a selection strategy, the fitness value f_i of the best string of generation *k* is compared with the fitness value f_J of the worst string of generation *k + 1* **,** if $f_i > f_J$, then string M_i*(k)* is substituted for M_J*(k + 1)*, so that the maximum fitness value of the population never decreases as the process of evolution continues. They use a new crossover operator called *polygon crossover* and a mutation operator where the probability of mutation is dynamically determined depending on the fitness values. Mutation involves the perturbation of two adjacent triangles, one of which is randomly chosen and the other is chosen to be adjacent to the first, and polygon crossover only produces *legal* triangulations. The results are way better than those of a greedy algorithm but the matrix representation requires space proportional to n^2.

A new *weighted* coding GA has been proposed in (Capp and Julstrom, 1998) which improves the space complexity of GMWT. It is called **weight-coded GA for MWT** and associates an integer-valued to each point, so the number of weights in a chromosome equals the number of points in the problem instance. The triangulation represented by a chromosome is identified by adding each point's weight to the lengths of the edges in which it participates and applying a heuristic for MWT to the modified lengths, so this heuristic is used as a *decoding* algorithm. The length of the resulting triangulation, with the original lengths, is the chromosome's fitness and the single best chromosome is preserved for the next generation. The heuristic works as follows: it identifies the points' convex hull, sorts the unused edges into ascending order of their lengths and then an iterated greedy algorithm attempts to insert the next shortest edge into the triangulation. If the edge doesn't cross any edge already in the triangulation, the heuristic includes it (otherwise, the edge is discarded). This iterates until the triangulation is complete.

A **branch-and-cut** approach of the MWT problem has been proposed in (Kyoda et al., 1997). It works by combining the branch-and-cut paradigm with the *β*-skeleton method and reformulating the MWT problem as an Integer Program: “a subset of the complete graph of the n points such that no two edges intersect with each other and the number of edges is M, a constant for any triangulation.” (Kyoda et al., 1997, p. 385). The branch-and-bound algorithm solves the LP. If the solution is integral, then is the solution for the IP. If no, it appends to the LP some cutting planes that are guaranteed to be satisfied by the optimal solution to IP. Then, it solves the LP again and iterates until the IP solution is obtained or no cutting planes violate the solution. If this happens, the algorithm selects one of the variables which are neither 0 nor 1 and branches into two cases: in case a) the variable is set to 0 and in case b) the variable is set to 1. For each case, the algorithm applies cutting planes and solves the LP obtained. Then, the minimum value of the function with integral solution is obtained.

Here, “branching” corresponds to the cases “adopting the edge” and “discarding the edge”.

An **ant colony optimization algorithm** for MWT, *ACO_MWT*, has been proposed In (Jahani, Bigham and Askari, 2010), where the process of constructing solutions is viewed as a walk on the fully connected graph whose vertices are the point set that we want to compute MWT on it.

2. Methodology.

Spiral prototyping (Boehm, 1986) has been used in the design of a new microaggregation method and tested against different toy example microdata sets to evaluate the fitness of the microaggregated set produced, and therefore the feasibility of the method. Different versions of the method have been refined based on the information loss outcomes.

Experimental tests have been conducted to measure the outcomes of the proposed method and compared to other well-known methods when fed with reference and random data set test file. The tests have been iterated for different values of the security parameter and combination of attributes and finally averaged to observe the trends in information loss.

All the necessary program modules and functions have been developed in *R* (Hornik, 2008), a programming language for statistical computing that has been widely used for developing statistical software and data analysis (Vance, 2009) with increasing popularity (Tiobe, 2016). We have implemented from scratch our microaggregation method, the *SSE*, *SST* and *Information Loss* function, as well as several testing modules to carry out the experiments.

We have used the implementation of the Delaunay triangulation in package *tripack* (Renka, et al., 2015) with a modification to allow duplicate records. This modification consisted of a pre-processing ordering of the records and a post-processing to add different triangles for the duplicate records (*vertices*).

3. Proposed method.

We propose the following fixed-size multivariate microaggregation method, based on the well-known Delaunay triangulation. It uses the triangles generated by the triangulation as a 'pre-clustering' heuristic. The triangulation is computed only once, at the beginning of the algorithm, and then post-processed to include repeated values so any vertex in a triangle can be seen as a vector of (possible) *n* duplicated points.

The method generates groups of *k* records except maybe one last group which may contain between *k* and 2*k*-1 records. Within each iteration, the algorithm calculates the two more distant triangles of the unassigned points, based on the Euclidean distance between their centroids. It will then grow two clusters of k records, alternatively starting from these two farthest triangles.

For any cluster, it chooses the points to add based on the Euclidean distance from the current – provisional - cluster centroid to the corresponding triangle centroid. Then, the points will be added to the cluster ordered by its Euclidean distance to the centroid, until *k* points are assigned.

Therefore, for each iteration the algorithm assigns 2*k* points and the process goes on until less than 2*k* points remain unassigned, building an extra cluster of *k* records if necessary.

Finally, the remaining points – if any - are assigned to the closest group by computing its centroid and the new centroid of the cluster is re-computed.

Algorithm 1 **Triangulation-Based Multivariate Microaggregation with Fixed Group Size (TBM)**

Require: *D* data set with N 2-dimensional data points

Require: *k* Minimum cardinality constraint

Ensure: *M* Microaggregated data set

```
T ← ComputeTriangulation(D)
TriangleCentroids ← GetCentroids(T)
Distance ← ComputeDistanceMatrix(TriangleCentroids)
Unassigned ← D
M ← matrix[N,2]
remain ← Length(Unassigned)
while ( remain >= 2k )
    Cent ← ComputeCentroid(Unassigned)
    CentB ← GetFarthestTriangle(Cent)
    CentA ← GetFarthestTriangle(CentB)
    Cluster , Centroid ← GrowCluster(CentA, k)
    M ← InsertCentroid(Centroid,Cluster)
    Unassigned ← Unassigned – Cluster
    Cluster , Centroid ← GrowCluster(CentB, k)
    M ← InsertCentroid(Centroid, Cluster)
    Unassigned ← Unassigned – Cluster
    remain ← remain – 2k
end while
if ( remain >= k )
    Cent ← ComputeCentroid(Unassigned)
    CentA ← GetClosestTriangle(Cent)
    Cluster, Centroid ← GrowCluster(CentA, k)
    M ← InsertCentroid(Centroid, Cluster)
    Unassigned ← Unassigned – Cluster
    remain ← remain – k
    if ( remain  > 0 )
        Cent ← ComputeCentroid(Unassigned)
        CentA ← GetClosestCluster(Cent)
        Cluster ← ExtendCluster(CentA, Unassigned)
        Centroid ← ComputeCentroid(Cluster)
        M ← InsertCentroid(Centroid, Cluster)
    end if
end if
return (M)
```

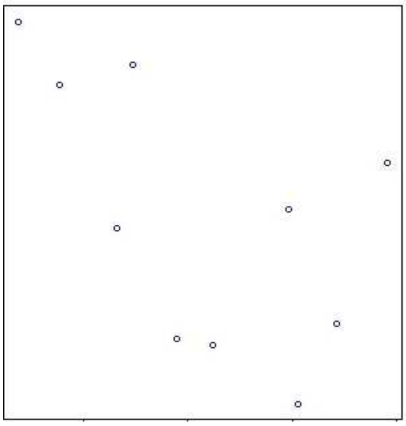
Algorithm 1.1 ***GrowCluster*(*Origin*, *k*)**

Require: *origin* initial triangle.
Require: *k* minimum cardinality constraint
Ensure: *k*-length *Cluster*

```
Cluster ← ()
P ← ()
C1← GetTriangleCentroid(origin)
ClusterCentroid ← C1
while ( Length(Cluster)<k )
    l ← 0
    while (l ==0)
        P2 ← GetTriangleVertex(origin)
        P ← P2 U P - Visited
        l ← Length(P)
        if (l ==0)
            if (Length(Cluster)==0)
                Ce ← GetTriangleCentroid(i)
            else
                Ce ← ComputeCentroid(Cluster)
            end if
            SetVisited(i)
            i ← GetClosestTriangle(Ce)
        end if
    end while
    P ← SortPfromDistanceTo(P, ClusterCentroid)
    n ← 1
    while ((Length(Cluster) < k ) and (l >0))
        cluster ← InsertIntoCluster(Cluster, P[1])
        SetVisited(P[1])
        P ← P - P[1]
        l ← Length(P)
        n ← n +1
    end while
    if (Length(P)==0)
        SetCentroidVisited(i)
    ClusterCentroid ← ComputeCentroid(Cluster)
    i ← GetClosestTriangle(ClusterCentroid)
end while
return (Cluster, ClusterCentroid)
```

A simple example with 10 random points, *k* = 4 is given below. In figure 1 we can see the input data set for our example.

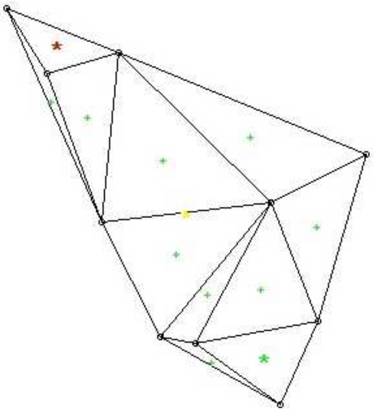
Fig. 1. Input data set



Source: Own elaboration.

In figure 2 we can see our method computes the triangulation of the set of points - in this case, a Delaunay triangulation - and calculates the baricenters of the triangles [line 2] (the green ‘plus’ signs, eleven triangles in the example) and their distance matrix [line 3]. Then, for every iteration it computes the centroid [line 8] of the unassigned points (the yellow ‘star’), the farthest triangle [line 9] from it (the red ‘star’) and the farthest triangle [line 10] from the latter (the green ‘star’). These two triangles will be the *source* triangles from which the clusters *A* and *B* will begin to grow.

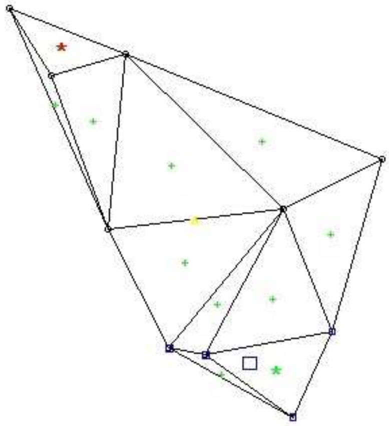
Fig. 2. Triangulation and source points



Source: Own elaboration.

The *growCluster* [lines 11] function will choose the *k* = 4 closest points to the centroid, according to the triangles proximity. The corresponding clusters centroids are recalculated every time a point is assigned. Figure 3 shows the points assigned to cluster *A* (marked as squares) and the centroid of cluster *A* (the bigger square mark).

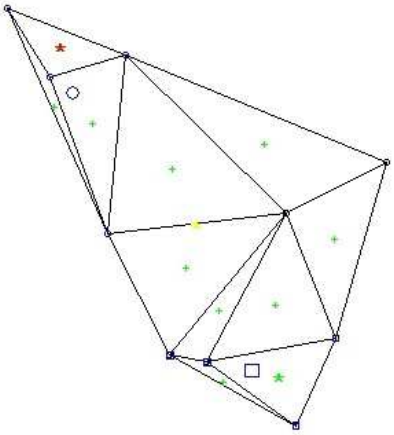
Fig. 3. Cluster A.



Source: Own elaboration.

Once cluster *A* is completed, the algorithm will grow the second cluster [line 22] from the other source point (the red star). Figure 4 shows the points assigned to cluster *B*, marked with circles.

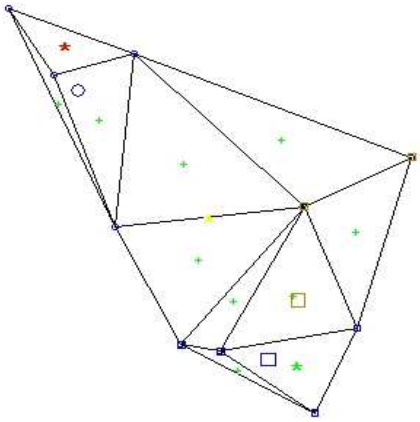
Fig. 4. Cluster B.



Source: Own elaboration.

As a result, the first group (*A*) is marked with squares and the second group (*B*) is marked with circles. In our example, after one single iteration, only two records remain unassigned (*less than k*). So the algorithm calculates their centroid and assigns these points to its closest cluster, which is the cluster *A* (squares), and its centroid is then re-calculated [lines 26:32],. Figure 5 shows this process (hence the big golden square mark which has slightly "moved" towards the upper-right side of the image).

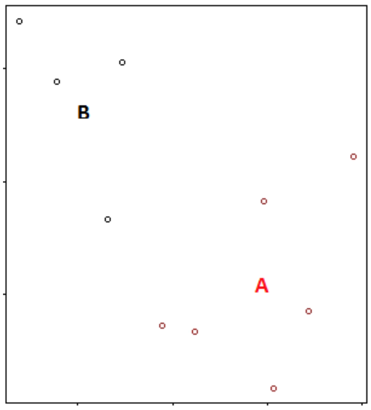
Fig. 5. Remaining points assigned to cluster A.



Source: Own elaboration.

So, finally, the algorithm has grown one cluster of six points and a second cluster of four points. Figure 6 shows the clustering outcome, with the records in cluster A (red) and cluster B (black).

Fig. 6. Clustering outcome.



Source: Own elaboration.

And, after replacing the records by their corresponding cluster centroids, we get 34.712% information loss.

4. Experimental results.

In this section, empirical results on the proposed heuristic are reported and compared with those obtained by other microaggregation methods available in the *sdcMicro* R package (Templ, Kowarik and Meind, 2015): the *simple*[3] method and state-of-the-art *mdav*.

To study the information loss of our method and compare its outcomes, we have implemented an iterative testing procedure. Two experiments have been conducted by means of this procedure: the first one having a reference data set as input and the second one using a random-generated data set.

4.1. Experiment #1

The first experiment was carried out using the *Census* data set, that has become usual reference for testing multivariate microaggregation methods (Brand, Domingo-Ferrer and Mateo-Sanz, 2002; Laszlo and Mukherjee, 2005; Solanas, 2015). This microdata set contains 1080 records with 13 numerical attributes.

The experiment involved repeating thirty times the following procedure:

- According to the goals and constraints of this research regarding the bivariate case, two different attributes from the data set were chosen on a random basis to create a reduced data set, D. (Every test using a distinct combination of attributes).
- Both attributes were standardized to have mean 0 and variance 1 before microaggregation, in order to give them equal weight regardless of their scale, getting data set D'.
- D' was then sorted by first and second attributes in ascending order to get an ordered data set D''. This was done in order to simplify the pre-processing of the Delaunay implementation, as stated in *section 2*, and to avoid a worst-case scenario for the *simple* method.
- Every method tested was fed with the same input, D'', and iterated for different values of the security parameter, *k*, in the range [2:50], to produce a *k-Anonymized* version of the original data.
- Results were de-standardized.

[3] Simple method clusters *k* records sequentially. “With method ‘simple’ one can apply microaggregation directly on the (unsorted) data. It is useful for the comparison with other methods as a benchmark […]” (Templ, Kowaric and Meindl, 2016, p. 45).

- Information loss was measured for every single method iteration.

Finally, all the tests results were averaged, to get a unique information loss value for every combination of method and security parameter. Standard deviation was also computed.

Table 1 shows the average information loss caused by each method for different values of the security parameter *k*, measured using the following expression:

$$I_{loss} = \frac{SSE}{SST} \cdot 100 \quad (1)$$

Where *SST* is the total sum of squares (sum of squared Euclidean distances from all records to the data set centroid).

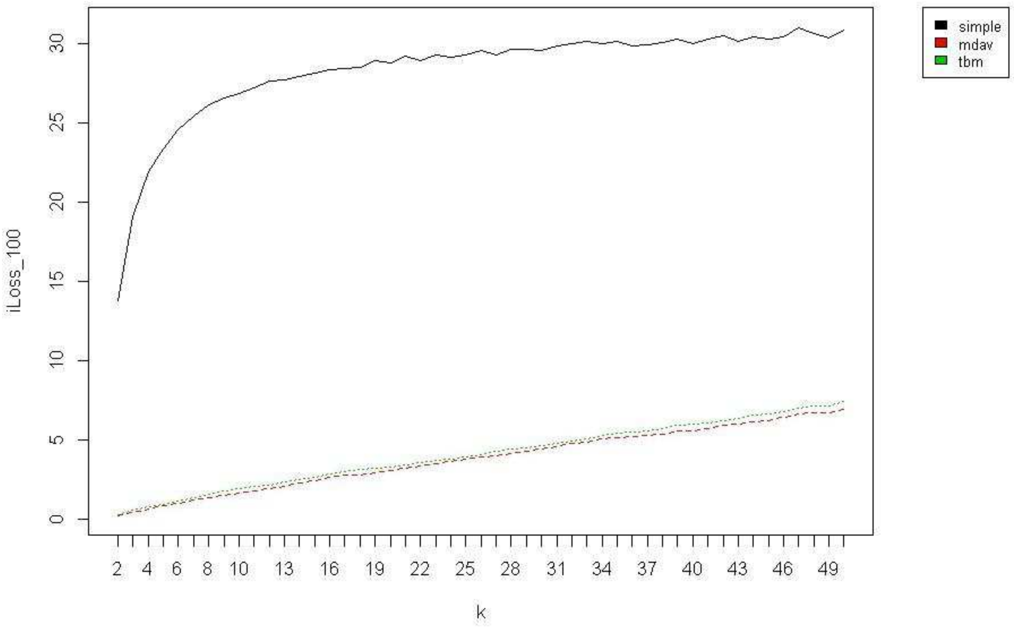
Table 1: Comparison of information loss (%). *Census* data set.

k	simple		mdav		tbm	
	mean	sd	mean	sd	mean	sd
2	13.823	15.951	0.210	0.117	0.312	0.196
3	19.074	22.096	0.451	0.292	0.591	0.349
4	21.940	25.249	0.637	0.323	0.784	0.380
5	23.385	26.626	0.854	0.406	0.963	0.451
6	24.600	28.139	1.027	0.451	1.174	0.533
7	25.427	28.911	1.192	0.496	1.348	0.585
8	26.145	29.676	1.359	0.551	1.551	0.672
9	26.628	30.236	1.497	0.559	1.783	0.752
10	26.924	30.438	1.644	0.619	1.931	0.769
20	28.805	32.177	3.075	1.012	3.315	1.049
30	29.590	32.826	4.421	1.429	4.681	1.361
40	30.011	33.023	5.590	1.744	6.035	1.825
50	30.899	33.363	6.974	2.122	7.435	2.391

Source: Own elaboration.

Figure 7 shows the graph of information loss for each *k* in the range 2 to 50.

Fig. 7. Information loss (%) graph for k in [2:50]. *Census* dataset.



4.2. Experiment #2

For the second experiment, a test was performed on 30 uniform random data sets, each data set consisting of 1080 records with 2 numerical attributes. These are referred to as *Sim* data sets. Attribute values were independently drawn from the [−10000, 10000] interval by simple random sampling.

The experiment involved repeating thirty times the following procedure:

- A *Sim* two-dimensional data set, D, was generated.
- Both attributes were standardized to have mean 0 and variance 1 before microaggregation, in order to give them equal weight, regardless of their scale, getting data set D'.

- D’ was then sorted by first and second attributes in ascending order to get an ordered data set, D’’.
- Each method tested was fed with the same input, D’’, and iterated for different values of the security parameter, *k*, in the range [2:50], to produce a *k-Anonymized* version of the original data.
- Results were de-standardized.
- Information loss was measured for every single method iteration.

Finally, all the tests results were averaged, to get a unique value of information loss for every combination of method and security parameter.

Table 2 shows the information loss caused by each method, for different values of the security parameter *k*, measured using expression (1):

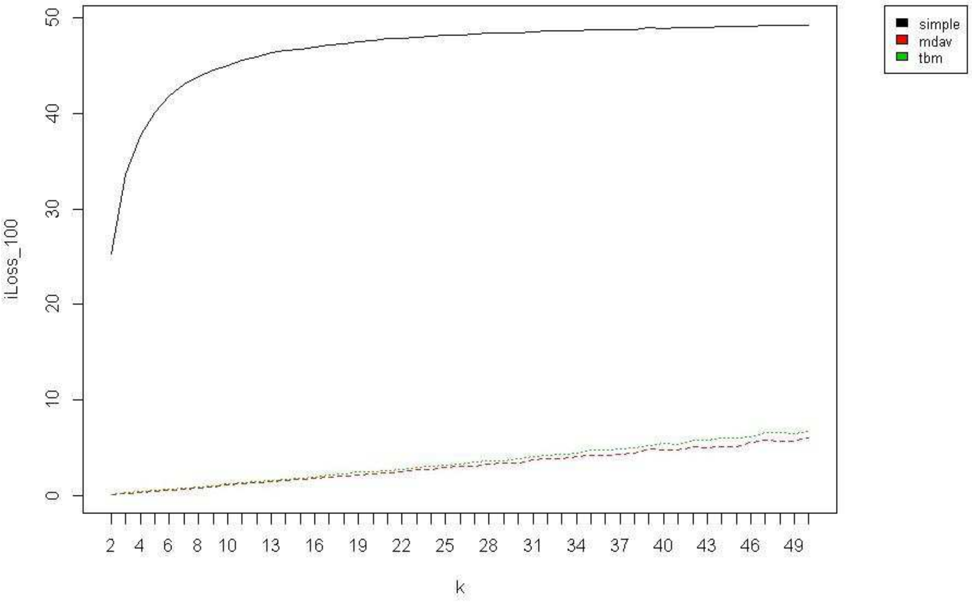
Table 2: Comparison of information loss (%). *Sim* random-generated data sets.

k	simple		mdav		tbm	
	mean	sd	mean	sd	mean	sd
2	25.269	1.082	0.113	0.003	0.165	0.015
3	33.546	0.939	0.240	0.008	0.318	0.015
4	37.680	0.846	0.365	0.014	0.444	0.024
5	40.109	1.048	0.497	0.017	0.575	0.029
6	41.743	0.989	0.624	0.018	0.697	0.030
7	43.038	0.942	0.741	0.023	0.823	0.033
8	43.791	0.908	0.861	0.030	0.958	0.030
9	44.477	0.781	0.983	0.028	1.092	0.044
10	44.947	0.931	1.107	0.029	1.229	0.051
20	47.548	0.923	2.300	0.063	2.568	0.095
30	48.389	0.941	3.474	0.115	3.893	0.217
40	48.890	0.880	4.760	0.190	5.532	0.403
50	49.241	1.014	6.099	0.237	6.731	0.530

Source: Own elaboration.

Figure 8 shows the graph of information loss for each *k* in the range 2 to 50.

Fig. 8. Information loss (%) graph for k in [2:50]. *Sim* datasets.



Source: Own elaboration.

4.3. Discussion

The results show that the performance of our method is close to that of MDAV, though slightly inferior (less than 1% information loss within the studied range of *k*).

We believe that this inferior result is due to the characteristics of the chosen triangulation (Delaunay) and the algorithm heuristic itself, which chooses the closest triangle's baricenter from a given point. Since minimizing the distance between the triangles doesn't necessarily imply minimizing the distance between their points (unless the triangles' edges are minimal), an error might be introduced every time a point is added to a cluster because the next point is not exactly the closest one.

With regard to stability, MDAV have shown to be the most stable method.

5. Conclusion and further research.

- In this paper we have shown that it is possible to use a triangulation-based heuristic method for microaggregation with satisfactory outcomes in terms of information loss.

- A new fixed-size heuristic method for multivariate microaggregation has been proposed.

- The experiments we have carried out show that our method performance is close to that of MDAV in terms of information loss.

Further work needs to be done to compare the various microaggregation methods with different reference data sets, including the trade-off between data disclosure risk and information loss.

Finally, optimizing the outcome of the method using an optimal triangulation (MWT) heuristic can be a promising line of research.

Bibliography

BOSE, P.; DEVROYE, L.; EVANS, W. (2002). Diamonds are not a minimum weight triangulation's best friend. *International Journal of Computational Geometry & Applications*, vol. 12, no. 6, p. 445-453.

BOEHM, B. (1986). A spiral model of software development and enhancement. *ACM SIGSOFT Software Engineering Notes*, vol. 11, no. 4, p. 14-24.

BOWYER, A. (1981). Computing Dirichlet tessellations. *Computer Journal*, vol. 24, no. 2, p. 162-166.

BRAND, R.; DOMINGO-FERRER, J.; MATEO-SANZ, J. M. (2002). Reference data sets to test and compare SDC methods for protection of numerical microdata. *European Project IST-2000-25069 CASC.* [online]. [Date of query: Oct. 25, 2016]. Available at: <*http://neon.vb.cbs.nl/casc/CASCrefmicrodata.pdf*.>

CAPP, K.; JULSTROM, B. A. (1998). A weight-coded genetic algorithm for the minimum weight triangulation problem. In: *Proceedings of the 1998 ACM symposium on Applied Computing, SAC'98*. New York, USA: ACM, p. 327-331. doi:10.1145/330560.330833.

CHANG, C.; LI, Y.; HUANG, W. (2007). TFRP: An efficient microaggregation algorithm for statistical disclosure control. *Journal of Systems and Software*, vol. 80, no. 11, p. 1866-1878. doi:10.1016/j.jss.2007.02.01.

CHENG, S.W.; XU, Y.F. (2001). On β-skeleton as a subgraph of the minimum weight triangulation. *Theoretical Computer Science*, vol. 262, no. 1, p. 459-471. doi:10.1016/S0304-3975(00)00318-2.

CHENG, S.W.; GOLIN, M. J.; TSANG, J.C.F. (1995). Expected case analysis of-skeletons with applications to the construction of minimum weight triangulations. In: *Proceedings of the Seventh Canadian Conference on Computational Geometry (CCCG)*. Quebec, Canada, p. 279-284.

CHETTRI, S.; PAUL, B.; DUTTA, A. K. (2013). Statistical Disclosure Control for Data Privacy Preservation. *International Journal of Computer Applications*, vol. 80, no. 10, p. 38-43.

CIGNONI, P.; MONTANI, C.; SCOPIGNO, R. (1998). DeWall: A fast divide and conquer Delaunay triangulation algorithm in E^d. *Computer-Aided Design*, vol. 30, no. 5, p. 333-341. doi:10.1016/S0010-4485(97)00082-1.

CLARKE, G. U.; WRIGHT, J. W. (1964). Scheduling of vehicles from a central depot to a number of delivery points. *Operations research*, vol. 12, no. 4, p. 568-581.

DE BERG, M.; VAN KREVELD, M.; OVERMARS, M.;SCHWARZKOPF, O. (2008). *Computational Geometry: Algorithms and Applications.* 3rd ed. Berlin, Heidelberg: Springer-Verlag.

DEFAYS, D.; ANWAR, N. (1995). Micro-aggregation: a generic method. In: *Proceedings of the 2nd International Symposium on Statistical Confidentiality.* Luxembourg: Office for Official Publications of the European Communities, p. 69-78.

DELAUNAY, B. (1934). Sur la Sphère Vide. A la memoire de Georges Voronoi. *Izvestia Akademii Nauk SSSR, Otdelenie Matematicheskih i Estestvennyka Nauk*, vol. 7, p. 793-800.

DICKERSON, M. T.; MONTAGUE, M. H. (1996). A (usually?) connected subgraph of the minimum weight triangulation. In: *Proceedings of the 12th annual ACM Symposium on Computational Geometry*. Philadelphia, USA: ACM, p. 204-213. doi:10.1145/237218.237364.

DOMINGO-FERRER, J.; MATEO-SANZ, J. M. (2002). Practical data-oriented microaggregation for statistical disclosure control. *IEEE Transactions on Knowledge and data Engineering*, vol. 14, no. 1, p. 189-201. doi:10.1109/69.979982.

DOMINGO-FERRER, J.; TORRA, V. (2001). Disclosure control methods and information loss for microdata. In: DOYLE, P.; LANE, J.I.; THEEUWES, J.J.M.; ZAYATZ L.M. (eds.). *Confidentiality, disclosure, and data access: theory and practical applications for statistical agencies*. Amsterdam: Elsevier, p. 91-110.

DOMINGO-FERRER, J.; MARTÍNEZ BALLESTÉ, A.; MATEO-SANZ, J. M.; SEBÉ, F. (2006). Efficient multivariate data-oriented microaggregation. *The VLDB Journal*, vol. 15, no. 4, p. 355-369. doi: 10.1007/s00778-006-0007-0.

DWYER, R. A. (1987). A faster divide-and-conquer algorithm for constructing Delaunay triangulations. *Algorithmica*, vol. 2, no. 1-4, p. 137-151. doi:10.1007/BF01840356.

EDELSBRUNNER, H.; SHAH, N. R. (1992). Incremental topological flipping works for regular triangulations. In: *Proceeding 8th annual ACM symposium on Computational geometry*. ACM. Berlin: ACM, p. 43-52. doi:10.1007/BF01975867.

GUIBAS, L.; STOLFI, J. (1985). Primitives for the manipulation of general subdivisions and the computation of Voronoi. *ACM transactions on graphics,* vol. 4, no. 2, p. 74-123.

GUIBAS, L. J.; KNUTH, D. E.; SHARIR, M. (1990). Randomized incremental construction of Delaunay and Voronoy diagrams. In: *International Colloquium on Automata, Languages, and Programming*. Berlin: Springer, p. 414-431. doi:10.1007/BF01758770.

HANSEN, S. L.; MUKHERJEE, S. (2003). A polynomial algorithm for optimal univariate microaggregation. *IEEE Transactions on Knowledge and Data Engineering*, vol. 15, no. 4, p. 1043-1044. doi:10.1109/TKDE.2003.1209020.

HORNIK, K. (2008). The R FAQ. [online]. [Date of query: Dec. 28, 2016]. Available at: <http://CRAN.R-project.org/doc/FAQ/R-FAQ.html>. ISBN 3-900051-08-9.

HU, S. (2009). A new asymmetric inclusion region for minimum weight triangulation. *Journal of Global* Optimization, vol. 46, no. 63, p. 63-73. doi:10.1007/s10898-009-9409-z.

HUNDERPOOL, A.; VAN DE WETERING, A.; RAMASWAMY, R.; FRANCONI, L.; CAPOBIANCHI, A.; DEWOLF, P.; DOMINGO-FERRER, J.; TORRA, V.; BRAND, R.; GIESSINGS, S. (2003). µ-ARGUS version 3.2 software and user's manual. *Statistics Netherlands, Voorburg*.

JAHANI, M.; BIGHAM, B. S.; ASKARI, A. (2010). An Ant Colony Algorithm for the Minimum Weight Triangulation. In: *International Conference on Computational Science and Its Applications (ICCSA)*. Fukuoka, Japan: IEEE, p. 81-85. doi:10.1109/ICCSA.2010.38.

JOE, B. (1989). Three-dimensional triangulations from local transformations. *SIAM Journal on Scientific and Statistical Computing*, vol. 10, no. 4, p. 718-741. doi:10.1137/0910044.

JOE, B. (1991). Construction of three-dimensional Delaunay triangulations using local transformations. *Computer Aided Geometric Design*, vol. 8, no. 2, p. 123-142. doi:10.1016/0167-8396(91)90038-D.

KEIL, J. M. (1994). Computing a subgraph of the minimum weight triangulation. *Computational Geometry*, vol. 4, no. 1, p. 18-26. doi:10.1016/0925-7721(94)90014-0.

KIRKPATRICK, D. G. (1980). A note on Delaunay and optimal triangulations. *Information Processing Letters*, vol. 10, no. 3, p. 127-128.

KIRKPATRICK, D. G.; RADKE, J. D. (1985). A Framework for Computational Morphology, In: TOUSSAINT, G.D. (eds.). *Machine Intelligence and Pattern Recognition*. North-Holland: Elsevier, vol. 2, p. 217-248.

KOKOLAKIS, G.; FOUSKAKIS, D. (2009). Importance partitioning in micro-aggregation. *Computational Statistics & Data Analysis*, vol. 53, no. 7, p. 2439-2445. doi:10.1016/j.csda.2008.09.028.

KYODA, Y.; IMAI, K.; TAKEUCHI, E.; TAJIMA, A.; (1997). A branch-and-cut approach for minimum weight triangulation. In: *8th International Symposium on Algorithms and Computation,* Singapore. Berlin: Springer, p. 384-393. http://dx.doi.org/10.1007/3-540-63890-3_41.

LASZLO, M.; MUKHERJEE, S. (2005). Minimum spanning tree partitioning algorithm for microaggregation. *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, no. 7, p. 902-911. doi:10.1109/TKDE.2005.112.

LASZLO, M.; MUKHERJEE, S. (2015). Iterated local search for microaggregation. *Journal of Systems and Software*, vol. 100, p. 15-26. doi:10.1016/j.jss.2014.10.012.

LAWSON, C.L. (1977). Software for C^1 surface interpolation. In: RICE, J. (ed.). *Mathematical Software III*. New York: Academic Press, p. 161–194.
[online]. [Date of query: Oct. 25, 2016]. Available at: <https://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/19770025881.pdf>

LEE, D. T.; SCHACHTER, B. J. (1980). Two algorithms for constructing a Delaunay triangulation. *International Journal of Computer and Information Science,* vol. 9, no. 3, p. 219-242. doi:10.1007/BF00977785.

LEVCOPOULOS, C.; KRZNARIC, D. (1998). Quasi-greedy triangulations approximating the minimum weight triangulation. *Journal of Algorithms*, vol. 27, no. 2, p. 303-338. doi:10.1006/jagm.1997.0918.

LIN, J.; WEN, T.; HSIEH, J.; CHANG, P. (2010). Density-based microaggregation for statistical disclosure control. *Expert Systems with Applications*, vol. 37, no. 4, p. 3256-3263. doi:10.1016/j.eswa.2009.09.054.

MATEO-SANZ, J.M.; DOMINGO-FERRER, J. (1998). A comparative study of microaggregation methods. *Questiio: Quaderns d'Estadistica, Sistemes, Informatica i Investigació Operativa*, vol. 22, no. 3, p. 511-526.

MAYA LÓPEZ, A.; SOLANAS, A. (2015). Multivariate microaggregation with fixed group size based TSP. Master Thesis. Open University of Catalonia. In: *Review Universitat Oberta de Catalunya Open Access* [online]. [Date of query: Oct. 10, 2016]. Available at: < http://hdl.handle.net/10609/43963 >

MCLAIN, D. H. (1976). Two dimensional interpolation from random data. *The Computer Journal*, vol. 19, no. 2, p. 178-181.

MORTAZAVI, R.; JALILI, S. (2014). Fast data-oriented microaggregation algorithm for large numerical datasets. *Knowledge-Based Systems*, vol. 67, p. 195-205. doi:10.1016/j.knosys.2014.05.011.

MORTAZAVI, R.; JALILI, S.; GOHARGAZI, H. (2013). Multivariate microaggregation by iterative optimization. *Applied intelligence*, vol. 39, no. 3, p. 529-544.

MULZER, W.; ROTE, G. (2008). Minimum-weight triangulation is NP-hard. *Journal of the ACM*, vol. 55, no. 2, p. 1-29. doi:10.1145/1346330.1346336.

NIN, J.; HERRANZ, J.; TORRA, V. (2008). Towards a more realistic disclosure risk assessment. In: DOMINGO-FERRER, J; SAYGIN, Y. (eds.). *Privacy in Statistical Databases: UNESCO Chair in Data Privacy International Conference*, PSD 2008, Istanbul, Turkey, September 24-26, 2008. Berlin: Springer, p. 152-165.

OGANIAN, A.; DOMINGO-FERRER, J. (2001). On the complexity of optimal microaggregation for statistical disclosure control. *Statistical Journal of the United Nations Economic Commission for Europe*, vol. 18, no. 4, p. 345-353.

PANAGIOTAKIS, C.; TZIRITAS, G. (2013). Successive group selection for microaggregation. *IEEE Transactions on Knowledge and Data Engineering*, vol. 25, no. 5, p. 1191-1195. doi:10.1109/TKDE.2011.242.

PANAGIOTAKIS, C.; TZIRITAS, G. (2015). A minimum spanning tree equipartition algorithm for microaggregation. *Journal of Applied Statistics*, vol. 42, no. 4, p. 846-865. doi:10.1080/02664763.2014.993361.

PANAGIOTAKIS, C.; GRINIAS, I.; TZIRITAS, G. (2011). Natural image segmentation based on tree equipartition, bayesian flooding and region merging. *IEEE Transactions on Image Processing*, vol. 20, no. 8, p. 2276-2287. doi:10.1109/TIP.2011.2114893.

PREPARATA, F. P; SHAMOS, M.I. (1985). *Computational geometry: An introduction*. First edition. Texts and Monographs in Computer Science*.* New York: Springer-Verlag.

QIN, K.; WANG, W.; GONG, M. (1997). A genetic algorithm for the minimum weight triangulation. In: *Evolutionary Computation, IEEE International Conference on*. IEEE, p. 541-546. doi:10.1109/ICEC.1997.592370.

REMY, J.; STEGER, A. (2009). A quasi-polynomial time approximation scheme for minimum weight triangulation. *Journal of the ACM,* vol. 56, no. 3, p. 15.

RENKA, R.J.; GEBHARDT, A.; EGLEN, S.; ZUYEV, S.; WHITE, D. (2016). Tripack: Triangulation of Irregularly Spaced Data (versión 1.3-8). [online]. [Date of query: Nov. 1, 2016]. Available at: <https://cran.r-project.org/web/packages/tripack/index.html>.

SEBÉ, F.; DOMINGO-FERRER, J.; MATEO-SANZ, J.M.; TORRA, V. (2002). Post-masking optimization of the tradeoff between information loss and disclosure risk in masked microdata sets. In: DOMINGO-FERRER, J. (ed.). *Inference Control in Statistical Databases*. Berlin: Springer, p. 163-171.

SEIDEL, R. (1995). The upper bound theorem for polytopes: an easy proof of its asymptotic version. *Computational Geometry*, vol. 5, no. 2, p. 115-116.

SINCLAIR, D. A. (2010). S-hull: a fast radial sweep-hull routine for delaunay triangulation. [online]. [Date of query: Dec. 7, 2016]. Available at: <http://www.s-hull.org/paper/s_hull.pdf>

SOLANAS, A. (2008). Privacy protection with genetic algorithms. In: YANG, A.; SHAN, Y.; BUI, L.T. (eds.). *Success in evolutionary computation*. Berlin: Springer, p. 215-237. http://dx.doi.org/10.1007/978-3-540-76286-7_10.

SOLANAS, A.; MARTINEZ-BALLESTE, A.; DOMINGO-FERRER, J. (2006). V-MDAV: a multivariate microaggregation with variable group size. In: RIZZI, A.; VICHI, M. (eds.). *17th COMPSTAT Symposium of the IASC*, Rome. Berlin: Springer, p. 917-925.

SU, P.; DRYSDALE, R. L. S. (1995). A comparison of sequential Delaunay triangulation algorithms. In: *Proceedings of the 12th Annual ACM Symposium on Computational geometry*. Philadelphia, USA: ACM, p. 61-70. doi:10.1145/220279.220286.

TEMPL, M.;KOWARIK, A.; MEINDL, B. (2015).Statistical Disclosure Control for Micro-Data Using the R Package sdcMicro, *Journal of Statistical Software*, vol. 67, no. 4, p. 1-36. doi:10.18637/jss.v067.i04.

TEMPL, M.;KOWARIK, A.; MEINDL, B. (2016). Package 'sdcMicro'. Statistical Disclosure Control Methods for Anonymization of Microdata and Risk Estimation. Version 4.6.1. *The Comprehensive R Archive Network*. [online]. [Date of query: Oct. 3, 2016].
Available at: <https://cran.r-project.org/web/packages/sdcMicro/sdcMicro.pdf>

TIOBE. (2016). Tiobe index for R. *R* | *TIOBE - The Software Quality Company*. [online]. [Date of query: Dec. 28, 2016].
Available at: < http://www.tiobe.com/tiobe-index/r/>

UNITED NATIONS STATISTICAL COMMISSION, et al. (1994). Fundamental principles of official statistics. *Official Records of the Economic and Social Council.*

VANCE, A. (2009). Data analysts captivated by R's power. *New York Times*. [online]. [published Jan. 6, 2009]. [Date of query: Dec. 7, 2016]. *Available at:* <http://www.nytimes.com/2009/01/07/technology/business-computing/07program.html>

VORONOÏ, G.M. (1908). Nouvelles applications des paramètres continus à la théorie des formes quadratiques. Deuxième mémoire. Recherches sur les parallélloèdres primitifs. *Journal für die reine und angewandte Mathematik*, vol. 134, p. 198-287.

WATSON, D.F. (1981). Computing the n-dimensional Delaunay tessellation with application to Voronoi polytopes. *Computer Journal*, vol. 24, no. 2, p. 167-172.

Resumen

La microgregación es una técnica para el Control de Divulgación Estadística en el que los registros similares se agregan en grupos que contienen un mínimo de k registros y son luego sustituidos por los centroides de cada grupo, de modo que los datos liberados preserven algunas de sus propiedades estadísticas al mismo tiempo que se reduzca el riesgo de re-identificación. Un método de tamaño fijo divide los datos en grupos de tamaño k, excepto tal vez un grupo con tamaño entre k y 2k-1, mientras que un método orientado a datos (de tamaño variable) permite que el tamaño de grupo esté comprendido entre k y 2k-1. Es necesario emplear métodos heurísticos, ya que el problema de la microagregación es de tipo NP-duro.

En este trabajo hemos realizado un estudio de los diversos métodos de microagregación en la literatura y proponemos un nuevo enfoque heurístico para la microagregación multivariable de tamaño fijo, basado en la triangulación del conjunto de puntos en $\mathbb{R}^2$*. Se han utilizado un conjunto de datos de referencia y otro conjunto generado aleatoriamente para comparar los resultados del método propuesto, en términos de pérdida de información, con los de otros métodos conocidos.*

Palabras clave

microagregación, protección de microdatos, control de divulgación estadística, privacidad, triangulación.

Resum

Fusce vestibulum lorem ac turpis cursus fermentum. Vivamus pharetra bibendum velit nec rutrum. Duis arcu massa, posuere vel consectetur quis, suscipit ac massa. Duis convallis rutrum justo, vitae sodales velit aliquam et. Integer enim nibh, tristique quis lacinia et, porta vitae lorem.

Sed placerat luctus erat, sed pellentesque justo gravida tristique. In magna sem, fermentum sit amet elementum sit amet, eleifend eget nulla. Vestibulum vitae ante metus, non imperdiet orci. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; Sed dolor velit, malesuada at consectetur eu, volutpat in tortor. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos himenaeos. Praesent ac pellentesque enim. Nunc elementum volutpat metus vel pharetra. Vivamus faucibus lorem non ante ultricies scelerisque

Paraules clau

vestibulum vitae, ante metus, non imperdiet orc, ipsum primis, faucibus luctus, ultrices posuere, cubilia Curae

Juvenal Machín Casañas
jmachinc@uoc.edu
Open University of Catalonia (Spain)

Juvenal Machín is B.Eng. in Computer Engineering (1999) from La Laguna University and B.Sc.(Hons) in Computer Engineering (2011) from Catholic University of Murcia. He has been the dean of the Official Association of Computer Engineers of the Canary Islands and professor-tutor at UNED University. He is founder member and area leader at Barcelonaqbit , 'the quantum information and cybersequrity think-tank' and Chief Security Officer at La Palma Health Area. His fields of activity are information security, smart health and data privacy.

Agustí Solanas Gómez
agusti.solanas@urv.cat
Department of Computer Engineering and Mathematics
Rovira i Virgily University (Spain)

Dr. Agusti Solanas is M.Sc. (2004) in Computer Engineering from Rovira i Virgili University and Ph.D. (2007) in Telematics Engineering from the Technical University of Catalonia. He has authored over 100 publications and he has delivered several invited talks worldwide. He is senior member of the Institute of Electrical and Electronics Engineers (IEEE) and member of the Association for Computing Machinery (ACM). He is the head of the Smart Health Research Group and Associate Professor in the Department of Computer Engineering and Mathematics at the Rovira I Virgili University (URV) of Tarragona, Catalonia, Spain, and the vice-president of the Computer Society Spain Chapter. His fields of activity are mobile health, smart health, cognitive health, data privacy, ubiquitous computing, and artificial intelligence.

http://www.smarthealthresearch.com