**CASC PROJECT**

Computational Aspects of Statistical Confidentiality

# NEW MICROAGGREGATION ALGORITHMS (SOFTWARE DOCUMENTATION AND RELATED PAPERS)

**(WORKPACKAGE 1.1, DELIVERABLE 1.1-D5)**

**Josep M. Mateo-Sanz, Josep Domingo-Ferrer, Francesc Sebé,**
**Antoni Martínez-Ballesté, Àngel Torres and Narcís Macià**
**Universitat Rovira i Virgili**
**{jmateo,jdomingo,fsebe,anmartin,atorres,nmacia}@etse.urv.es**

**11 June 2002**

This deliverable is intended to document the microaggregation software contributed in Deliverable 1.1-D6. The algorithms being implemented are described in two scientific papers attached to this deliverable.

The software in Deliverable 1.1-D6 consists of a single piece of portable C++ code (microaggregation.cpp). It can be compiled and run in any environment.

**User's manual**

Software usage:

```
$ microaggregation parameters_file
```

This program performs a microaggregation over a set of data. For a single variable (univariate data), the Hansen-Mukherjee polynomial exact microaggregation algorithm is used (see attached [Hans02]). For the general case (multivariate data), Domingo-Ferrer/Mateo-Sanz heuristic multivariate fixed-size microaggregation is used (see attached [Domi02]). The data set is formed by *n* records of *v* variables each, so its size is *n*·*v*. These variables are numbers read in C++ `double` format (if integers are provided, they are converted) and a row of *v* numbers is considered a record. These data are kept in the input file.

For example, let us suppose the following input file:

```
10 4 6 2 1 0  4
12 3 7 1 2 1 -1
17 2 5 1 3 1 -2
21 2 8 2 4 1 -1
9 3 3 3 5 1  -4
12 4 7 3 6 0 -3
12 4 6 3 7 0  4
14 3 7 3 6 0 -5
13 3 6 3 5 4 -1
15 3 7 5 4 3  3
17 2 6 7 3 2 -2
17 3 8 7 2 1 -2
18 4 6 7 1 1 -1
```

In the example above, there are 13 records and 7 variables.

There are different ways to perform a multivariate microaggregation. One possibility is to use the whole set of variables to form groups of records; other possibilities consist of using different subsets of variables. Variables can also be sorted in many ways, so the results of different executions could be slightly different by changing the order of the columns (only if groups of variables are defined and the columns change between groups). The sorting and partitioning of the variables must be defined in the parameter file. This is a text file, with parameters, comments (lines beginning with #) and CRs. The parameters in this file must be described strictly in the following order:

1 Verbose mode. This is used to generate a standard output list of the parameters read from file, to make shure that parameters are well defined.

2 Input and output data files. If the output file does not exist, a new one will be created. Otherwise, the results are appended to the existing output file.

3 Number of records. In the example above this parameter would be 13.

4 Number of variables. In the example above it would be 7.

5 Minimum number of records per group. This parameter defines the minimum number *k* of records per group. In the example above, if this parameter equals 4, there will be 2 groups with 4 records and a remaining one with 5. If there were 160 records, the program would generate 40 different groups with 4 elements each.

6 Number of groups of variables. This parameter describes whether the entire set of variables is used to group records (in that case the parameter is 1) or whether groups of variables must be considered. (Some example definitions are included below)

7 Number of variables per group. Number of variables in each group.

8 Order of the variables. Variables can be sorted before performing microaggregation. This parameter line defines the sorting info.

Next there is a parameter file example:

```
# ###################################################
# Parameters example file for 'microaggegation'
# (Lines with '#' are comments, CR are also skipped)
#####################################################

# Set 1 for verbose mode, 0 for 'silent' mode
1

# Input and output data files
input.dat
output.dat

# Number of records
13

# Number of variables
7

# Minimum number of records per group
3

# Number of groups of variables
1
```

In this example, all variables will be used to calculate distances and group records. In the next example, there are 7 variables (V1, V2, V3, V4, V5, V6 and V7), but two groups of variables are used to calculate distances:

```
...

# Number of variables
7

# Minimum number of records per group
3

# Number of groups of variables
2

# How many variables (columns) are in each group.
4 3

# This will define the column (variables) sorting
# for the original data file.
1 2 3 4 5 6 7
```

The first group is V1, V2, V3, V4 and the second one is V5, V6, V7. In the last example, three groups are defined: G1=(V1, V5, V7), G2=(V2, V4), G3=(V3, V6).

```
...
# Number of variables
```

```
7

# Minimum number of records per group
3

# Number of groups of variables
3

# How many variables (columns) are in each group.
3 2 2

# This will define the column (variables) sorting
# for the original data file.
1 5 7 2 4 3 6
```

Note that G1=(V1, V5, V7) will give the same output as G1=(V7, V1, V5), G1=(V5,V7, V1), and so on.

The output identifies each group of records with their mean. For example, the following output means that the records (rows) 1, 3 and 4 belong to the same group, whereas 2 and 5 form another group (note that all variables have been used to calculate distances):

```
1.2     5.76   3.4     8.2     -1.6
0.2     8.9    1.7     3.2     4.1
1.2     5.76   3.4     8.2     -1.6
1.2     5.76   3.4     8.2     -1.6
0.2     8.9    1.7     3.2     4.1
```

In another example output, where two groups of variables (G1=(V1, V4, V5) and G2=(V2, V3)) are used, the output could be as follows:

```
0.75    2.25   5.7     7.95    1.1
4.16    2.25   5.7     1.76    4.13
4.16    3.3    1.7     1.76    4.13
4.16    3.3    1.7     1.76    4.13
0.75    3.3    1.7     7.95    1.1
```

For G1 there is a classification: elements 1 and 5 are in the same class, whereas 2, 3 and 4 form the other class. For G2, elements 1 and 2 are in the same class, whereas 3, 4 and 5 form the other class.

## References (attached to this document)

[Hans02] S. L. Hansen and S. Mukherjee, “A polynomial algorithm for optimal univariate microaggregation”, IEEE Transactions on Knowledge and Data Engineering (to appear).

[Domi02] J. Domingo-Ferrer and J. M. Mateo-Sanz, “Practical Data-Oriented Microaggregation for Statistical Disclosure Control”, IEEE Transactions on Knowledge and Data Engineering, vol. 14, no. 1, pp. 189-201.