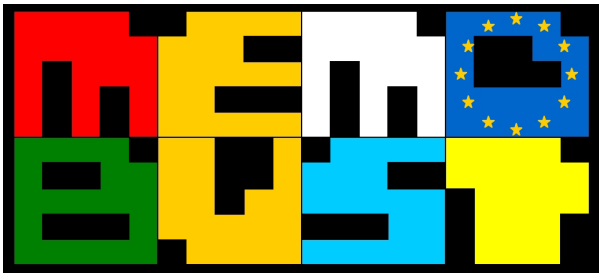

This module is part of the

Memobust Handbook

on Methodology of Modern Business Statistics

26 March 2014

Method: Weighted Matching of Object Characteristics

Contents

General section

1. Summary

Weighted matching is applied to match two data sets with many common units, on common object characteristics. The method is able to value the strength of possible (candidate) matches by using matching weights. Weighted matching can be formulated as an optimisation problem, in which the optimal (weighted) sum of matches is calculated, under certain constraints, such as that each record can appear in at most one match. The goal of the method is to find solutions to such problems, exact ones or good approximations. The reader is advised to consult the theme module "Micro-Fusion – Object Matching (Record Linkage)" prior to reading the present one. Also the reader should refer to the method module "Micro-Fusion – Unweighted Matching of Object Characteristics", which can be viewed as a special case of the matching method described in the present paper. It also introduces some concepts that are not re-introduced in the current module.

2. General description of the method

2.1 Outline

Various matching methods make use of matching weights. They can be used to differentiate between the potential matches in a matching problem. There is a variety of reasons to work with matching weights: you may want to express that not all of the variables are equally reliable, that is, that they do not have reliable scores. Or you may want to indicate that different objects corresponding with records that are matching candidates demonstrate a certain degree of similarity or dissimilarity. Or you may want to demonstrate that different objects are a certain distance apart, as measured by a certain metric. Or you want to use a probability to show that two objects are probably the same. Then a probability model is needed to quantify differences in scores on the matching key, and the resulting probabilities can be used as matching weights. The method described in this module uses weights to match records on the same object from different data sets. The module draws heavily on Willenborg and Heerschap (2012) to which the interested reader is referred for additional information. It is also the reason why several examples provided are from social statistics, rather than from business statistics. They have been retained as they illustrate certain points clearly. They are also indications that matching is not only used within the business statistics area.

We start with some preliminary material on graphs and metrics in the next subsection.

2.2 Preliminaries

2.2.1 Graphs

Graphs are convenient to describe matching. We only need a few elementary concepts from this area. These are presented in the current section, along with some notation and graphical conventions.

A graph $G = (V, E)$ consists of a finite set of points V, also called nodes or vertices, of which some pairs are connected by lines (E), also called sides, edges or branches. A graph is depicted in Figure 1.

Weights can be assigned to the lines (edges) in the form of real numbers. A graph with weights associated with points or edges is called a weighted graph. In this module the weights are associated

with the edges, and they express the strength of the match between two records. There are different ways to calculate these weights. In some applications, the smaller the weights the more similar the keys of the records are.[1]

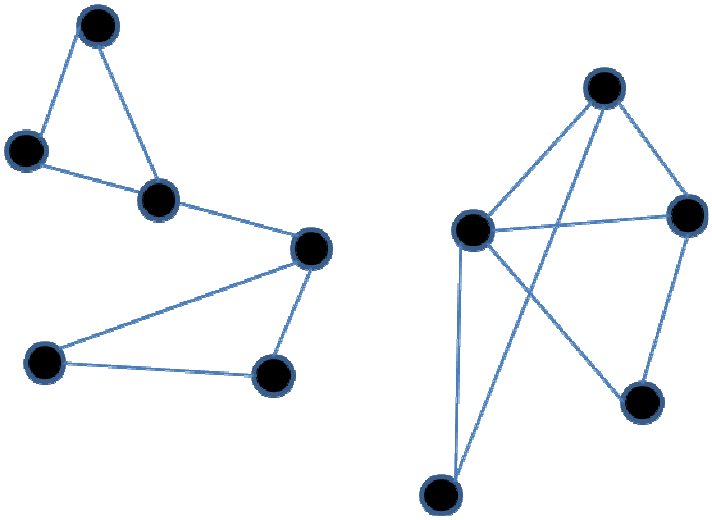

Figure 1. Example of a graph with two connectivity components

A special type of graph is the bipartite graph. See Figure 2. Here, the set of nodes *V* can be split into two disjoint sets A and B. The edges only connect nodes in A with nodes in B. Bipartite graphs are highly suited to illustrating matching and the theory behind it. Important is the MC graph, the matching candidate graph. This is a bipartite graph that represents the possible matches between records from two files. The edges may or may not be assigned matching weights. A matching candidate graph symbolises part of the constraints that apply for a matching problem.

A *path* in a graph is a succession of nodes arranged in such a way that an edge runs from each node to the following node in the row. Given a graph $G = (V, E)$ where v and w are two points of G, so $v, w \in V$. A path in G from v to w is a sequence $v_1, ..., v_k$ of points in G, such that:

1. $v_1 = v$,
2. $v_2 = w$,
3. $\{v_i, v_{i+1}\} \in E$ for all $i = 1, ..., k-1$.

If there is a path from *v* to *w* in G, then there is also one from *w* to *v* (symmetry). If there is a path in G from *u* to *v* and from *v* to *w*, then there is also one from *u* to *w* (transitivity). Here, *u*, *v* and *w* are points in G. For each point *v* in G, there is – by definition – a path from *v* to *v* (reflexivity). In other words, the relationship 'connected by a path in a given graph' is an *equivalence relationship* on the set of points of the graph, i.e., a binary relationship that is reflexive, symmetrical and transitive. If there is only one equivalence class for a graph G, it is said to be *connected*. In that case, all pairs of points can therefore be connected with each other via paths in G. If there are two or more equivalence classes for

[1] In other applications it may just be the other way round.

a graph, G is said to be disconnected. In that case, an equivalence class of this relationship corresponds with a *connected component* of G; this is a connected subgraph of G.

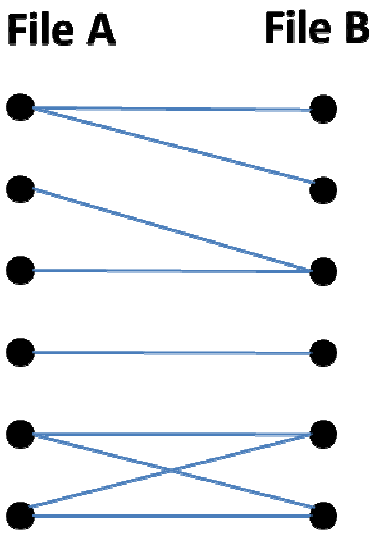

Figure 2. A bipartite graph.

2.2.2 Metrics

Metrics are an important concept for weighted matching. A metric is used in the present module to determine the matching weights. A metric is a function that defines the distance between each pair of elements of a set. Sometimes, it concerns a function that is related to that of a metric, but which deviates on several components from that of a metric. In that case, we have generalised metrics. But we will discuss metrics here first.

We assume a set X for which function $d : X \times X \rightarrow [0, \infty)$ is defined that satisfies a number of conditions:

1. $d(x, y) = 0$ if and only if $x = y$,
2. $d(x, y) = d(y, x)$ for all x, y in X (*symmetry*), and
3. $d(x, z) \leq d(x, y) + d(y, z)$ for all x, y, z in X (*triangle inequality*).

A non-negative function d that satisfies conditions 1, 2, and 3 is called a metric. The conditions for a metric are not always needed. Replacing them is sometimes necessary and yields alternative distance functions, such as pseudo-metrics or hypermetrics. But in this module we stick to metrics.

In matching and specifically in the comparison of matching keys, this concerns the measurement of the distances between the scores for the matching keys, or, in other words, determining the comparability or non-comparability.

In general, we denote by d, d_H or $d(.,.)$ a metric. We denote the scores on a matching key as a vector $(\alpha_1, ..., \alpha_n)$ for a matching key $(v_1, ..., v_n)$.

A few of the metrics we use here are so special that they are specified separately. The first one is the Hamming distance. Let α and β be two strings of equal length *n,* viewed as vectors of symbols. The Hamming distance between α and β is defined as:

$$d_H(\alpha,\beta) = d_H((\alpha_1,...,\alpha_n),(\beta_1,...,\beta_n)) = |\{i \mid \alpha_i \neq \beta_i, 1,...,n\}|,$$

i.e., the number of places in which the vectors α and β have different scores. Note that the Hamming distance can be defined for all types of variables.

To illustrate the Hamming distance suppose that there are two matching keys of four alphanumeric figures, '1034' and '1135' respectively. In this case, the Hamming distance is 2, because the figures differ in two places, which are positions 2 and 4. In other words: the smaller the Hamming distance the greater the comparability of the matching keys. The Hamming distance is equal to the number of 'elementary changes' that must be made in one key value to obtain the other key value.

The next metric that we want to introduce is the Levenshtein distance. Let α and β be two strings. The Levenshtein distance $d_L(\alpha,\beta)$ counts the minimum number of elementary operations, such as deleting a character, replacing a character, adding a character, that are necessary to transform one string into the other. If another elementary operation is added, namely interchanging neighbouring characters, then we have the so-called Levenshtein-Damerau-distance. The Levenshtein distance and the Levenshtein-Damerau distance are examples of metrics that are specifically designed for strings. There are other metrics of this type, specifically tailored to certain types of variables.

Consider the words '*apple*' and '*pear*'. Their Levenshtein distance is 4. To see this consider the following chain of elementary changes: *apple* → *pple* → *pele* → *peae* → *pear.* In less than 4 steps a transformation from *'apple'* to *'pear'* using elementary transformation associated with this distance function are not possible, as the reader in invited to check. The advantage of the Levenshtein distance, compared to the Hamming distance, is that the distance of strings of different lengths can be calculated.

More examples of metrics for strings and relevant for matching can be found in Section 4.

2.3 Calculating matching weights

There are different ways to determine matching weights that can be used in a matching problem. We will discuss several here. The list is not exhaustive, but it does provide several important examples. These matching weights are used for matching if the information about the 'matching candidacy' of two records is not represented in 'either/or' form (matching candidate? 'yes' or 'no'), but with more differentiation. The extent to which two records match can be expressed in a matching weight.

In the discussion in the sections below, we look at two data sets, A and B, that contain records, for which there are common matching variables $v_1,...,v_n$ that together form the matching key, based on which the records in the two data sets are matched. Weights are used for candidate matches, to indicate the 'strength' of a match. See Figure 3 for such a situation.

For more information and examples on MC-graphs, the interested reader should consult the method module "Micro-Fusion – Unweighted Matching of Object Characteristics".

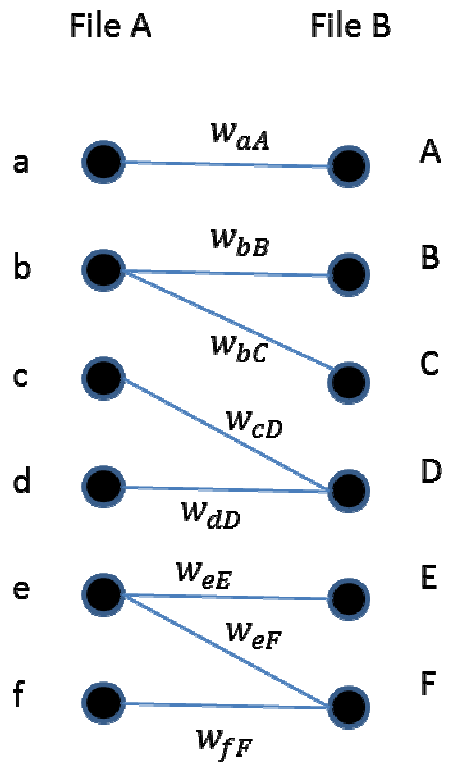

Figure 3. MC-graph with matching weights

2.3.1 Using metrics

For matching, it is important to find suitable metrics for each of the variables in a primary or secondary matching key, or rather for each type of variables. The following variables may occur in a secondary matching key: names (first names, surnames, enterprise names, street names, city names, etc.), time indications (dates of birth, ages at a certain reference time), economic activity, etc. Finding suitable metrics to be used for the secondary matching keys can be seen as a separate subfield in matching.

We take another look at strings, as they are quite important in matching. There are several aspects to strings when it comes to measuring the distance between them. This depends on how we look at them: literally, as objects built from an alphabet, or looking at other aspects such as their pronunciation (phonetics) or their meaning.

A metric can be used to calculate matching weights. These matching weights can be used to express the strength of a candidate match. We should add that, in practice, it is necessary to work with cut-off values: matches that are too weak in terms of the associated matching weight are not considered to be matching candidates. The trick is to properly establish these cut-off values: on the one hand too many irrelevant matches should be avoided but not many correct matches should be missed. In practice, this requires experimentation with various settings of the cut-off values.

All the considerations to use matching weights must be derived from the processes or mechanisms that (may) have caused differences in the data. This could be writing mistakes (‘Dickson’ instead of ‘Dixon’), alternative designations (‘Main Str.’ instead of ‘Main Street’), use of synonyms (‘shipping’ instead of ‘transporting’). It is therefore important to have thorough knowledge of the way in which

the data sets to be matched have been compiled. In addition, it is possible that not exactly the same matching variables will be used in the two data sets, or that the scores do not relate to the same moment in time. As a result, the attributes of entities (e.g., businesses, enterprises, etc.) could have changed.

2.3.2 Using probabilities

Matching weights can also be based on probability models. Stochastic methods can enter into matching for different reasons. We offer the following reasons:

1. Errors can occur in the secondary matching keys. The errors can be present for various reasons. An answer to a question in a survey could have been understood incorrectly and therefore answered incorrectly by the respondent in question; a given answer could have been incorrectly processed, for example, keyed in wrongly; errors could have been made in the coding of answers, etc. This type of error is often referred to as a non-sampling error. The first step would be to identify and model all major sources of errors using probability models. These models can then be used to calculate the probabilities that two scores match based on corresponding object characteristics from two matching data sets.
2. The reference times of the two matching data sets differ to such an extent that the effects of the dynamics of the population are noticeable on the units contained therein: values of certain scores could have been changed for some units. An enterprise could have merged, split or gone bankrupt. Therefore, if the reference times differ significantly from one another, it is not self-evident that the units and/or their scores on object characteristic variables would have remained unchanged.
3. Some comparable matching variables are not defined exactly the same way in the two data sets. The associated question can be different, or the position in the questionnaire could have been changed, or the value range of comparable variables may differ slightly. In that case, it may sometimes be unclear which scores correspond with one another. Suppose {20,21} is an age class in one matching file and 11 - 20 and 21 - 30 are age classes in the other file. The 20 and 21-year-olds are in the same age group in the first matching file, but they are in two different age categories in the second. We can also estimate which part of the people in the category (20,21) in the first file will end up in the age category 11-20 and which part will be in the age category 21-30 in the second file: $\frac{n_{20}}{n_{20}+n_{21}}$, and $\frac{n_{21}}{n_{20}+n_{21}}$ respectively, where n_{20} is the number of 20-year-olds on the measurement date and n_{21} the number of 21-year-olds at that point in time.

In practice, combinations of these causes of differences often occur. Data sets can have different reference times, there may be processing errors in the data, and the units may not be exactly comparable. Section 4 presents examples of a situation as in point 3 above, and an example with a combination of points 2 and 3 above (variables with deviating value ranges and different reference times).

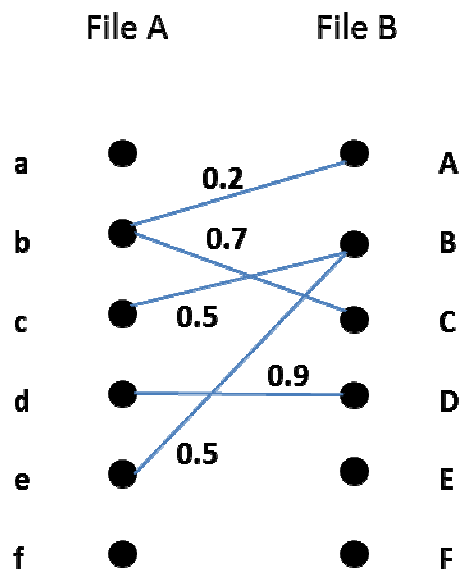

Figure 4. MC-digraph with probabilities as matching weights.

2.4 Quality of matching variables

In practice, based on the quality of the scores, we will want to differentiate between the different matching variables in the matching key. Some variables will have more reliable scores than others, and we will want to take this effect into account when determining the overall matching weight.

We consider this 'quality weight' as a subjective weight that the person performing the matching establishes based on his/her knowledge and experience with the different variables in the matching key. It is possible that experiments must first take place before a good choice can be made about these weights. These weights only have meaning in terms of the relationships between them, not in an absolute sense. Users can express the relative importance of a variable for the multivariate distance function. In this way, they can influence the effect of a certain variable in the total. If the variable has been reliably measured, then a relatively high weight is needed. If it is a variable with relatively more errors than the other variables in the matching key, then this variable should be given a lower weight.

For that matter, it is also possible to express the difference in the quality of matching variables in a different way, for example, when matching, by going through the scores in the order of the quality of the matching variables (from high to low), and then accepting certain deviations in the scores with increasing tolerance.

2.5 MC graph with matching weights

Once we have selected a method to determine matching weights, we can start calculating an MC graph with matching weights. We may have to use a cut-off value so that we do not have to include candidate matches of two records with a matching weight that is too low (they will not become edges in the MC graph).

2.6 Optimisation model

Once the MC graph with matching weights has been calculated we are almost ready to calculate the matching. What is needed in addition is a specification of an object function, and matching conditions. The object function could be the sum of the weights associated with the edges chosen for a particular match. If the weights are larger in case a match is stronger, the goal would be to find matches among the candidate matches that maximise the sum of the associated weights. The matching conditions yield the constraint for the matching. A common requirement is that a record can be in no more than one match.

The model that we get in this way is a well-known one in combinatorial optimisation, called bipartite matching. It is discussed in books like Lawler (1976, Ch.5), Papadimitriou and Steiglitz (1998, Ch.11) or Nemhauser and Wolsey (1988, Ch. III.2), to which the interested reader is kindly referred.

3. Preparatory phase

The object characteristics common to both data sets to be matched are identified. It has to be decided if they are suitable for this type of matching; the number of potential matches should not be too big. A suitable metric for these variables should be found, as well as a suitable cut-off value. This requires some experimenting: with the cut-off value the number of potential matches can be controlled. In case the matching data sets are big special measures should be taken, such as blocking to create a manageable matching problem.

Now candidate matches can be found, from which the matches are to be calculated.

4. Examples – not tool specific

4.1 First example

Given a matching key that consists of n variables that are all object characteristics. For the ith variable we have a metric d_i. For the entire matching key, we can define a metric $d = \sum_i w_i d_i$, with weights w_i, $w_i > 0$, $i = 1,...,n$.

4.2 Second example

Let δ be a 0-1-indicator function, defined as follows: $\delta(a,b) = 0$ if $a = b$ and $\delta(a,b) = 1$ if $a \neq b$, for scores a, b for a matching or other variable. For score vectors α, β , we define

$$\Delta(\alpha,\beta) = (\delta(\alpha_1,\beta_1),...,\delta(\alpha_n,\beta_n)) \in \{0,1\}^n .$$

This indicator vector plays a central role in the method described in Fellegi and Sunter (1969). See also the method module “Micro-Fusion – Fellegi-Sunter and Jaro Approach to Record Linkage” in the present handbook. Note that

$$d_H(\alpha,\beta) = \sum_{i=1}^{n} \delta(\alpha_i,\beta_i) .$$

4.3 Third example

Consider a name variable, such as first name, surname, business name, street name, place name, etc. There are several ways in which the extent to which the distance of these names, or what they stand for, can be expressed:

- **String as a sequence of symbols**. Here, you may want to express the extent to which two surnames differ from one another. The difference between 'Jansen' and 'Janssen' is smaller than the difference between 'Jansen' and 'Todd' (Jansen→Tansen→ Tonsen→ Todsen→ Todden→Todde→Todd). This concerns only the spelling of the names: the letters that are present and their order of occurrence. This can be quantified using a metric (the Levenshtein metric or Levenshtein-Damerau metric, for instance).
- **Meaning of a string**. The words 'teacher' and 'instructor' are very different from one another as strings, but in terms of meaning (concepts), they are very close, and could even be considered being synonyms.
- **Pronunciation of a string** The distance concept here relates to the meaning (semantics) associated with strings, not the way they are composed of characters from some alphabet. A similar difference is obtained if we consider pronunciation (say, in English) of strings, 'Dixon' and 'Dickson' are pronounced the same. Phonetically these strings are equal.

The last two cases are comparable, in the sense that we do not measure the distance of the literal strings, but on some associated attribute (interpretation / meaning or pronunciation).

Let d be a metric on S, and D a metric on T. Then $d(s,t)$ measures the distance between the strings s and t and $D(f(s),f(t))$ the distance between the meaning of s and t, or their pronunciation. This would be a distance between two points in a classification (a tree), which could, e.g., be the length of the shortest path (in the tree) connecting these points.

4.4 Fourth example (Soundex algorithm)

Comparing strings taking the phonetic characteristics of English into account could be done by using a so-called Soundex algorithm. This algorithm maps alphanumeric strings to Soundex strings (consisting of a letter followed by three numerical digits): the letter is the first letter of the name, and the digits encode the remaining consonants. Similar sounding consonants share the same digit.

A string (name) is mapped to a Soundex string using the following rules:

1. Retain the first letter of the name; drop all occurrences of a, e, I, o, u, y, h, w.
2. Replace consonants with digits as follows (after the first letter)
   - b,f,p,v→1
   - c,g,j,k,q,s,x,z→2
   - d,t→3
   - l→4
   - m,n→5

- r→6

3. If two or more letters with the same number are adjacent in the original name (before step 1), only retain the first letter; also two letters with the same number separated by ‘h’ or ‘w’ are coded as a single number, whereas such letters separated by a vowel are coded twice. This rule also applies to the first letter.
4. Iterate the previous step until you have one letter and three numbers. If you have too few letters in your word that you can’t assign three numbers, append with zeros until there are three numbers. If you have more than 3 letters, just retain the first 3 numbers.

Applying these rule to ‘Rupert’ and ‘Robert’ yields the same Soundex string R163.

4.5 Fifth example (Trigrams)

The names Hendriks, Hendricks, Hendrickx, Hendriksz, Hendrikx, Hendrix are all pronounced the same (in Dutch, at least), while all are different as strings. In this example we look at two of them and see how they differ if we look at trigrams. We consider two of them, ‘Hendriksz’ and ‘Hendrix’. For both names we consider the extended versions, which we obtain by adding a space (‘_’) at the start and end of each name. We then get: ‘_Hendriksz_‘ and ‘_Hendrix_‘ . The trigrams for the first string are (we write everything in lower case letters): (_he, hen, end, ndr, dri, rik, iks, ksz, sz_) and for the second string (_he, hen, end, ndr, dri, rix, ix_) . They have 5 trigrams in common, 5 out of 9 for the first string and 5 out of 7 for the second one.

5. Examples – tool specific

6. Glossary

For definitions of terms used in this module, please refer to the separate “Glossary” provided as part of the handbook.

7. References

Fellegi, I. P. and Sunter, A. B. (1969), A theory for record linkage. *Journal of the American Statistical Association* **64**, 1183–1200.

Lawler, E. L. (1976), *Combinatorial Optimization: Networks and Matroids*. Holt, Rinehart, Winston.

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

Papadimitriou, C. H. and Steiglitz, K. (1998), *Combinatorial Optimization: Algorithms and Complexity*. Dover, Mineola (NY).

Willenborg, L. and Heerschap, N. (2012), *Matching*. Contribution to Methods Series. Statistics Netherlands, The Hague.

Specific section

8. Purpose of the method

The purpose is adding variables to a microdata set Ds-input1 from a second microdata set Ds-input2 for the same objects in both data sets. Records from two microdata sets are combined using a set of common object characteristics. It can be viewed as a more general case of the unweighted matching method (described in the method module "Micro-Fusion – Unweighted Matching of Object Characteristics").

9. Recommended use of the method

1. In case object identifiers of good quality in both matching data sets are not available weighted matching may be considered as an option, under certain conditions.
2. Common object characteristic values of good quality should be present in both matching data sets. Also if similar variables are present in both data sets (with a different, but almost the same domain) this method can be considered, depending on how much the domains differ. Observation errors can occur in the scores of these variables.
3. The unweighted matching method can be characterised as being 'black and white': two records are either matching candidates or they are not. There is no room for any differentiation. However, there are situations where this is desirable. Some spelling mistakes or alternative designations are more likely than others.
4. In addition, it is possible that not exactly the same matching variables have been used in the two data sets, or that the scores do not relate to the same moment in time. As a result, the attributes of an entity (individual, business, etc.) could have changed. Also in this case the method aims at matching records for the same object.

10. Possible disadvantages of the method

1. It can be too slow, as compared to unweighted matching.
2. Values of tuning parameters require some experimentation or specialist knowledge.

11. Variants of the method

The text in the general section of the module places the emphasis on the basic variant for matching with matching weights, where the matches are 1:1. As stated earlier, there are also situations in which 1:n, m:1 and even n:m matches are possible. This is the case for composite units such as businesses which, over time, can split or merge into other units. Formally, this means that the conditions under which matches are possible must be adapted. Also they do not relate to the same units, but to combinations of units that produce comparable entities.

In the discussion we have so far assumed that all the scores on object characteristics are present. In practice, however, this is not always necessarily the case, and scores can also be erroneously missing. Calculating matching weights is more difficult in this situation, because the missing values cannot just be omitted: they must be replaced by stochastic variables, with a known assumed distribution. In such cases, the unknown parameter values must be estimated using, for example, the EM algorithm. For

information about the EM algorithm, see Wikipedia (http://en.wikipedia.org/wiki/EM_algorithm) and the references provided there.

We can summarise the available variants as follows:

1. Number of records in the output dataset:

   1.1 Each record of Ds-input1 is part of the output dataset (left outer join), or only matching records occur in the output dataset.

   1.2 Each record of Ds-input1 can occur more than once in the output dataset, or can occur at most once in the output dataset.

2. Duplicate records may be present in Ds-input1 or in Ds-input2.
3. One or more blocking variables can be used to divide the datasets for matching.
4. Missings in the object characteristics may be present in the input data sets.

12. Input data

1. Ds-input1. This is the primary input data set. It is a microdata set, to which additional variables will be added.
2. Ds-input2. This auxiliary input data set contains the variables that will be added to Ds-input1.

13. Logical preconditions

1. Missing values
   1. The object characteristic values used in the matching may contain missing values, but not too many, as they negatively influence the matching performance.
2. Erroneous values
   1. Errors in the object characteristic values are allowed, but it should still be possible to use them for matching. With certain assumptions on the cause of the errors, they must be are usable owing to a small distance to the correct values.
3. Other quality related preconditions
   1.
4. Other types of preconditions
   1. Enough object characteristic variables must be available in both input data sets to identify objects in the population. Otherwise more than one record with smallest distance remains, and an arbitrary choice should be made from them, with a high risk on Type I errors.

14. Tuning parameters

1. Optimisation function.
2. Matching weight.
3. Cut-off values.

15. Recommended use of the individual variants of the method

1.

16. Output data

1. Ds-output1: a microdata set containing all variables of Ds-input1, with variables added from Ds-input2.
2. Optional Ds-output2 containing all non-matching records from Ds-input1.
3. Optional Ds-output3 containing all non-matching records from Ds-input2.

17. Properties of the output data

1. The output data set contains all variables from Ds-input1, but with additional variables from Ds-input2, presumably for the same objects.

18. Unit of input data suitable for the method

Processing full data sets (internally blocking variables can divide a data set in smaller parts).

19. User interaction - not tool specific

1. Before matching the tuning parameters must be set by analysing the results for different values.
2. No user interaction during matching.
3. After matching the number of mismatches must be evaluated, and quality indicators (Type1 and Type 2 errors).

20. Logging indicators

1. Number of non-matching records from Ds-input1.
2. Number of non-matching records from Ds-input2.
3. Time used.

21. Quality indicators of the output data

1. The number of mismatches or missed matches and the number of missed matches can be used as quality indicators. The quality of the matching method can be assessed based on the inspection of matches of test files. It is a labour intensive job to carry out. You must examine not only the matching candidates and the matches ultimately selected, but also any missed matches under various parameter settings. The quality indicators are influenced by the way that the weights are calculated, the use of cut-off values and the use of blocking variables to stratify large data sets.

22. Actual use of the method

1.

Interconnections with other modules

23. Themes that refer explicitly to this module

1. Micro-Fusion – Object Matching (Record Linkage)
2. Micro-Fusion – Probabilistic Record Linkage

24. Related methods described in other modules

1. Micro-Fusion – Object Identifier Matching
2. Micro-Fusion – Unweighted Matching of Object Characteristics
3. Micro-Fusion – Fellegi-Sunter and Jaro Approach to Record Linkage

25. Mathematical techniques used by the method described in this module

1.

26. GSBPM phases where the method described in this module is used

1. 5.1 Integrate data

27. Tools that implement the method described in this module

1.

28. Process step performed by the method

Adding variables to microdata set

Administrative section

29. Module code

Micro-Fusion-M-Weighted Matching

30. Version history

Version	Date	Description of changes	Author	Institute
0.1	21-04-2012	first version	Leon Willenborg, Rob van de Laar	CBS (Netherlands)
0.2	02-07-2012	second version	Leon Willenborg, Rob van de Laar	CBS (Netherlands)
0.3	11-07-2013	third version	Leon Willenborg	CBS (Netherlands)
0.4	09-08-2013	revised version (using review comments)	Leon Willenborg	CBS (Netherlands)
0.5	17-11-2013	revised version (using EB review comments)	Leon Willenborg	CBS (Netherlands)
0.5.1	19-11-2013	preliminary release		
1.0	26-03-2014	final version within the Memobust project		

31. Template version and print date

Template version used	1.0 p 4 d.d. 22-11-2012
Print date	21-3-2014 17:58