STATISTICS IN MEDICINE, VOL. 14, 491–498 (1995)

# PROBABILISTIC LINKAGE OF LARGE PUBLIC HEALTH DATA FILES

MATTHEW A. JARO

*Match Ware Technologies, Inc., 14637 Locustwood Lane, Silver Spring, MD 20905, U.S.A.*

SUMMARY

Probabilistic linkage technology makes it feasible and efficient to link large public health databases in a statistically justifiable manner. The problem addressed by the methodology is that of matching two files of individual data under conditions of uncertainty. Each field is subject to error which is measured by the probability that the field agrees given a record pair matches (called the *m* probability) and probabilities of chance agreement of its value states (called the *u* probability). Fellegi and Sunter pioneered record linkage theory. Advances in methodology include use of an EM algorithm for parameter estimation, optimization of matches by means of a linear sum assignment program, and more recently, a probability model that addresses both *m* and *u* probabilities for all value states of a field. This provides a means for obtaining greater precision from non-uniformly distributed fields, without the theoretical complications arising from frequency-based matching alone. The model includes an iterative parameter estimation procedure that is more robust than pre-match estimation techniques. The methodology was originally developed and tested by the author at the U.S. Census Bureau for census undercount estimation. The more recent advances and a new generalized software system were tested and validated by linking highway crashes to Emergency Medical Service (EMS) reports and to hospital admission records for the National Highway Traffic Safety Administration (NHTSA).

## 1. INTRODUCTION

Record linkage has a number of important applications in the areas of public health and injury surveillance. Examples include linking cohort studies to avoid the cost of new surveys, linking morbidity data to mortality data, vital statistics linking, tracking cases longitudinally, and survey frame development. Record linkage projects are often implemented in an *ad-hoc* manner, with the researcher empirically deciding which combinations of matched fields constitute matched records. This technique generally involves a large number of passes through the data, provides no statistical justification for the process, and is difficult to repeat.

This paper discusses concepts and recent advances in record linkage methodology. The application of the AUTOMATCH Generalized Record Linkage System software[1] in matching highway safety crashes is presented.

### 1.1. Objective of record linkage

Individual record linkage involves two files: file A and file B with records pertaining to individual cases. Each file consists of fixed fields, which contain the information to be matched. Obviously, one or more fields on file A must have equivalent fields on file B. For example, in order to match on surname and age, both files A and B must contain fields containing this information. In order

CCC 0277–6715/95/050491–08

to analyse the record linkage problem, consider the set of all possible record pairs. The first pair would be record 1 from file A with record 1 from file B. The next pair would be record 1 from file A and record 2 from file B, until $n \times m$ pairs were formed (where $n$ is the number of records on file A and $m$ is the number of records on file B).

The objective of the record linkage process is to classify each pair as belonging to one of two sets: the set of matched record pairs M, and the set of unmatched record pairs, U. For example, if we were to inspect, say the pair created from record 123 on file A with record 217 on file B, we must be able to say it is not a match (and belongs in set U) or it is a match and belongs in set M.

Obviously, there are many times the number of unmatched pairs as there are matched pairs. To illustrate this, consider two files with 1000 records each. There are 1,000,000 possible record pairs, but only 1000 possible matches (if there are no duplicates on the files). Thus, set M will contain at most 1000 pairs and set U will contain the remaining 999,000 pairs.

In order for a record linkage application to be feasible, it should be possible for an analyst to examine the match fields for any record on file A and the equivalent fields for any record on file B, and declare with reasonable certainty that the record pair examined is a match or a non-match. One way to determine if a record linkage application is feasible is to multiply the number of unique values in each field and then to compare this product with the total number of records in both files. If the product is much greater than the number of records, the application is probably feasible.

### 1.2. Blocking

For any files of reasonable size it is not feasible to compare all record pairs since the number of possible pairs is the product of the number of records on each file. Even a case with two small files of 1000 records each has 1,000,000 possible pairs to examine, with 999,000 unmatched pairs. If there were a way to look at pairs of records having a high probability of being matches and ignoring all pairs with very low probabilities, then it would become computationally feasible to conduct the linkage with large files.

Fortunately, the concept of blocking provides a method of limiting the number of pairs being examined. If one were to partition both files into mutually exclusive and exhaustive subsets and only search for matches within a subset, then the process of linkage becomes manageable.

To understand the concept of blocking, consider a field such as age. If there are 100 possible ages, then this field partitions a file into 100 subsets. The first subset is all people with an age of zero, the next is those with an age of 1, etc. These subsets are called blocks (or pockets in some systems). Suppose that the age values were uniformly distributed. If this were so, then out of our 1000 record file, there would be ten records for people of age 0 on each file, ten records for people of age 1, etc.

The pairs of records to be compared are taken from records in the same block. The first block would consist of all persons of age 0 on files A and B. This would be $10 \times 10$ or 100 record pairs. The second block would consist of all persons on files A and B with an age of 1. When the process is complete, we would have compared 100 (blocks) $\times$ 100 (pairs in a block) = 10,000 pairs, rather than the 1,000,000 record pairs required without blocking.

Blocking causes all records having the same value in the blocking fields to be compared. One consequence of this is that records not matching on the blocking fields will automatically be classified as non-matched. In fact, if our blocking field were age, and age was in error on one of the files, then the records involved are considered to be unmatched. To get around this problem, multiple passes are used.

Suppose we were to run a match in which age is a blocking field. Any records that do not match can be rematched using another blocking scheme, say postal code of residence. If a record did not match on age in pass 1, then it still has an opportunity to match on postal code in pass 2. It is only those cases that have errors on both the age and postal code fields that will not be matched. If this is a major problem, then a third pass can be run with different blocking fields. Errors on all three blocking fields are unlikely.

It should be obvious from the example above that smaller blocks are many times more efficient than large blocks. It is much better to use very restrictive blocking schemes (especially in the first pass). Since most of the records will match on the first pass, a second pass match has much fewer records to process, and can be less restrictive.

A field such as age alone does not constitute a good blocking strategy, since age is generally not uniformly distributed (some ages may be much more prevalent in the files than others) and partitioning a large file into 100 categories still leaves many records in each block.

The blocking strategies for each pass should be independent to the extent possible. For example, if a pair of files had surname, given name, sex, and birthdate (year, month and day) fields, then the first pass could be blocked on surname, sex, and birth year. The second pass could be blocked on birth month, birth day, and given name. Errors in surname, for example, would be unmatched in pass 1, but would be likely to be matched in pass 2.

The fields that are the best blocking fields are those with the most number of values possible and the highest reliability. For example, sex alone is a poor choice, since it only divides the file into two subsets. Similarly, fields subject to a great probability of error should be avoided. For example, apartment number is generally misreported or omitted, and hence would not make a good blocking field. In mathematical terms, the fields with the highest weights make the best blocking fields.

### 1.3. Weights

The information contained in the fields to be matched helps to determine which record pairs are matches and which are non-matches. Each field provides some information. Taken together, all the fields should determine the status of the pair being examined.

Some fields provide more information more reliably than others. For example, it would be absurd to sort both files by sex and assert that if the sex agrees, the record pair represents the same individual. However, it would not be so silly to sort both files on Social Security Number and assert that if this number agrees then the record pair represents the same individual.

Each field has two probabilities associated with it. These are called the $m$ and $u$ probabilities. The $m$ probability is the probability that a field agrees given that the record pair being examined is a matched pair. This is effectively one minus the error rate of the field. For example, in a sample of matched records, if sex disagrees 10 per cent of the time due to a transcription error, or due to being misreported, then the $m$ probability for this field is $0{\cdot}9(1 - 0{\cdot}1)$. The more reliable a field, the greater the $m$ probability will be.

The $u$ probability is the probability that a field agrees given that the record pair being examined is an unmatched pair. Since there are so many more unmatched pairs possible than matched pairs, this probability is effectively the probability that the field agrees at random. For example, if both files contained 1000 records, there are at most 1,000,000 possible record pairs but only a maximum of 1000 matches, leaving 999,000 unmatched pairs. The contribution of the matched pairs is negligible in this size file, and even smaller in larger files.

The weight for a field is computed as the logarithm to the base two of the ratio of $m$ and $u$. To see how this translates into actual values, let us examine our example of the sex and the Social Security Number fields.

Assume that sex has a 10 per cent error rate and Social Security Number has a 40 per cent error rate. The $m$ probability for sex is 0·9. The $u$ probability is 0·5 in situations with an equal number of males and females. Thus, the weight for sex is: $\log_2(m/u) = \ln(m/u)/\ln(2) = \ln(0{\cdot}9/0{\cdot}5)/\ln(2) = 0{\cdot}85$. Assume that the probability of chance agreement of Social Security Number is one in ten million. Given $m$ as 0·6 (40 per cent error rate in matched pairs), then the weight for Social Security is $\log_2(0{\cdot}6/0{\cdot}0000001) = 22{\cdot}51$. Thus, the weight for a match on sex is 0·85 and a match on Social Security Number is worth 22·51. The weights have captured what we know intuitively about the fields.

For each record pair compared, a composite weight also maybe computed as the sum of the individual weights for all field comparisons. If a field agrees in the pair being compared, the agreement weight, as computed above, is used. If a field disagrees in the pair being compared, the disagreement weight is computed as: $\log_2[(1-m)/(1-u)]$. This results in field disagreements receiving negative weights. Thus, agreements add to the composite weight and disagreements subtract from the composite weight. Obviously, the higher the score, the greater the agreement. The $m$ probability must always be greater than the u probability. If this is not the case, the probability of chance agreement is greater than the probability that the field agrees in a matched pair, and thus the field does not aid in the discrimination of matched and unmatched record pairs and should be discarded.

The distribution of the composite weights is generally bimodal. Many values have highly negative weights, since most cases are unmatched pairs. There is another mode at highly positive weights for the matched cases.

## 2. THE AUTOMATCH GENERALIZED RECORD LINKAGE SYSTEM

### 2.1. Historical background

Newcombe and Kennedy[2] discussed the concept of weights based on the probabilities of chance agreement of component value states. Fellegi and Sunter[3] extended and formalized the mathematical concepts. More importantly, they presented an optimal decision procedure whereby cutoff threshold weights can be computed by deciding on acceptable probabilities for false matches and false non-matches. Since there are $2^n$ possible match/non-match configurations of $n$ fields, then given the $m$ and $u$ probabilities one can sum the probabilities for agreement and disagreement until the selected error threshold is reached. This establishes two threshold weights. All composite weights higher than the high threshold weight are considered matched. Weights between the two thresholds are clerical review cases, and weights below the low threshold are non-matches (see Figure 1).

Notice, in Figure 1, that there is an area under the curve in the M region that represents an extension of the U distribution. This area cannot be seen in an actual histogram, but can be imagined by extrapolation. This area represents pairs that are assigned the status of match but are in reality non-matches. If the clerical review region is very wide, then the probability of incorrect assignment is low. If it is narrow, then the probability increases. Similarly, the extension of the M curve into the U region represents those matches which are assigned the status of non-match.

The Fellegi–Sunter model requires that the $m$ probabilities and the $u$ probabilities act independently. This means that errors in fields must not be correlated, and values in one field are not correlated to values in any other field. In practice, the independence assumption required by the Fellegi–Sunter model does not entirely hold and match/non-match configurations are too simplistic for actual use. For example, if two fields differ by one character, one would not wish to say that they totally disagreed. Consequently, cutoff weights are generally determined by

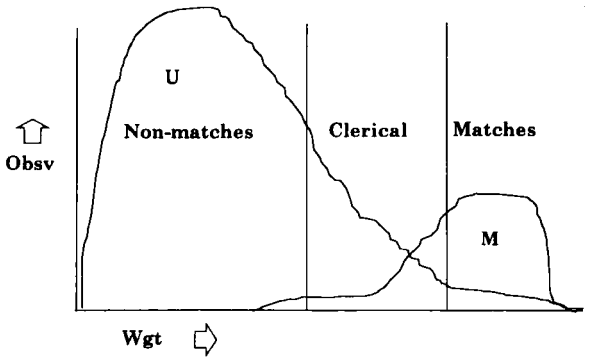


Figure 1. Distribution of weight for a typical matching process

a combination of an examination of the weight histogram and by an observation of actual results.

Jaro[4] extended the concepts of record linkage theory by developing a linear sum assignment approach to matching, extending the probability model, and using the estimation of maximum likelihood model[5] for $m$ probability estimation. The new matching has been implemented in the AUTOMATCH Record Linkage System[3] and the system represented significant improvements over the original UNIMATCH system[6].

### 2.2. Matching algorithm

The AUTOMATCH Record Linkage System[1] uses the following algorithm to conduct the matching:

1. Blocks are formed by reading the records with common blocking keys on file A and file B.
2. Weights are computed from the $m$ and $u$ probabilities, as follows:

$$m_i = \text{prob}\{\text{field } i \text{ agrees} \mid r \in \text{M}\}$$

$$u_i = \text{prob}\{\text{field } i \text{ agrees} \mid r \in \text{U}\}$$

$$w_i = \log_2 \frac{m_i}{u_i} \text{ if field agrees, else}$$

$$w_i = \log_2 \frac{(1 - m_i)}{(1 - u_i)}$$

$$W = \sum_i w_i$$

where $r$ is any given record pair and $W$ is the composite weight for the record pair.

3. Composite weight are found for all record pairs in the block being considered, using $[m_i/u_i]$ when field $i$ agrees on both records or $[(1 - m_i)/(1 - u_i)]$ if not. The contribution of fields that match to the composite weight is positive while the contribution of mismatching fields is negative.

4. A matrix is formed of all of the composite weights from records in both file A B. A linear sum assignment procedure is used to compute an optimal assignment as follows:

$$\text{Maximize to } Z_b = \sum_{i=1}^{p} \sum_{j=1}^{q} C_{ij} X_{ij}, \text{ subject to}$$

$$\sum_{j=1}^{q} X_{ij} = 1, i = 1, 2, \ldots, p$$

and

$$\sum_{i=1}^{p} X_{ij} = 1, j = 1, 2, \ldots, q,$$

where $C_{ij}$ is the cost (weight) of matching record $i$ on file A with record $j$ on file B, $X_{ij}$ is an indicator variable that is 1 if record $i$ is assigned to record $j$ and 0 if it is not, $p$ is the number of records in the block belonging to file A, and $q$ is the number of records in the block belonging to file B. $Z_b$ is the maximum weight for each block. This process is similar to that of optimally assigning employees to tasks in order to maximize the efficiency of the organization given that certain employees have propensities toward certain tasks. Each employee can be assigned one and only one task. Jaro first used the linear sum assignment model in matching for census undercount determination in the census pretest of Tampa, Florida, in 1985.[4]

The results of the linear sum assignment identify both the matched pairs and the duplicates on either file. If one row or column has more than one element above the match cutoff weight, then that record is a potential duplicate match to the record assigned for that row or column. Potential duplicates (or multiple records on either file) can be inspected with the interactive clerical review program provided with the AUTOMATCH system and dropped if they are not true multiples, or swapped with the record actually assigned to be the match. Such multiples can be retained on the same file as the matched pairs or copied to separate files.

A single file can be unduplicated using AUTOMATCH and the principles discussed for separate files apply to internal linkages.

### 2.3. Balanced probability model

Record linkage techniques have often taken advantage of the distribution of the value states in a field. For example, a match on a name like *Padereski* is more likely to represent the same individual than a match on a name like *Smith*. Consequently, frequency tables of value states have been used to alter the $u$ probability. For example, if there are 1000 records and *Smith* appears 200 times the $u$ probability for the field given *Smith* is 0·2. Unfortunately, using frequency tables in this manner creates an incorrect probability model, since the numerator of the likelihood ratio ($m$) is not conditioned upon specific values but the denominator ($u$) is so conditioned.

In the AUTOMATCH system, frequency tables are used, however, there are entries for the number of times a value has agreed in a matched pair and the number of times that value has participated in the matched pair. Thus, there is both a different $m$ and $u$ probability possible for each value state of each field. The user of the system must make an initial estimate of the global $m$ probability for each field. A match is then executed. Since most of the matching decisions will be correct despite errors in parameter estimation, a good sample of matched pairs is obtained. This sample is important since the $m$ probabilities require matched pairs. The matched pairs after

a matching pass are then inspected to update the frequency tables by value state. This process very quickly converges to obtain a stable set of $m$ probabilities.

## 3. CONDUCTING A RECORD LINKAGE PROJECT

The process of conducting a record linkage project involves more than just the matching algorithm. An important application of the AUTOMATCH system was matching highway crashes to Emergency Medical Service (EMS) records for the National Highway Traffic Safety Administration (NHTSA) and the National Association of Governors' Highway Safety Representatives (NAGHSR). The entire record linkage process can be summarized as follows:

1. Prepare the data files for matching. This preprocessing step involves standardizing the representation of the fields in both files, standardizing names and addresses, etc. Name and postal address standardization can be accomplished using the AUTOSTAN Generalized Standardization System.[7]
2. Prepare data dictionaries describing the files to be matched.
3. Prepare matching specifications. This involves deciding on blocking strategies, fields to be matched, etc.
4. Index both of the files to be matched. Since data files are often very large, it is not desirable to maintain multiple copies of these files. Traditional methods would involve taking the two input files and sorting them by the blocking fields (now we have four files: the two original files and the two new sorted files), then performing a match (now we have non-matched records, duplicates, and matched pair files, etc.) and repeating the process for each matching pass. AUTOMATCH operates by indexing the original files so the user records may be retrieved in sorted order. All programs operate by using pointers to the original records, so that the entire data management is integrated. It is only after all passes are complete that the user may want to extract information from the two original files.
5. Perform a frequency analysis on both files to construct tables for estimating the u probabilities for each field by value.
6. Run the matching algorithm. Pointer files are created that indicate which records matched and which records are residuals (non-matched records) on either file. The residuals only are processed in subsequent match passes. A histogram of the match results is printed so that the user can establish reliable cutoff weights.
7. Perform a clerical review. This program allows the user to inspect marginal matches and duplicates so that they can be matched (or unmatched) if they were treated incorrectly by the matcher.
8. Steps 4 to 7 are repeated in subsequent passes. All steps are integrated by means of scripts in the AUTOMATCH system.

The NHTSA project involved matching highway crashes to ambulance runs in the state of Maine. The same match was conducted previously using *ad-hoc* methodology involving over 100 passes, three months of time, and manual database queries. These results were replicated using the new AUTOMATCH system in two passes in several hours on a personal computer.

Specialized software is being developed to aid states in conducting highway crash matching for estimating Emergency Medical System (EMS) response times, fatality rates, and extent of injuries. This includes matching highway crashes to ambulance runs, highway–crash–ambulance–run matched pairs to hospital admissions, and finally highway crashes alone to hospital admissions. The first and second matches provide a means of tracking a case from crash to hospital discharge.

The record linkage software has been used in a number of health organizations including numerous cancer registries. Applications beside highway crashes have included vital statistics matching, cancer cohort studies, cancer record unduplication, geographic coding, address matching, database cleansing, tracking abused and neglected children, criminal behaviour studies, and epidemiological studies.

## REFERENCES

1. Match Ware Technologies, Inc., *AUTOMATCH Generalized Record Linkage System*, Silver Spring, MD. (1992).
2. Newcombe, H. B. and Kennedy, J. M. 'Record linkage', *Communications of the Association for Computing Machinery*, **5**, 563–566 (1962).
3. Fellegi, I. P. and Sunter, A. B. 'A theory for record linkage', *Journal of the American Statistical Association*, **64**, 1183–1210 (1969).
4. Jaro, M. A. 'Advances in record-linkage methodology as applied to matching the 1985 Census of Tampa, Florida', *Journal of the American Statistical Association*, **84**, 414–420 (1989).
5. Dempster, A. P., Laird, N. M. and Rubin, D. B. 'Maximum likelihood from Incomplete data via the EM algorithm', *Journal of the Royal Statistical Society*, **39**, 1–38 (1977).
6. Jaro, M. A. 'UNIMATCH: A computer system for general record linkage under conditions of uncertainty', *American Federation of Information Processing Societies (AFIPS) Conference Proceedings*, **40**, 523–530 (1972).
7. Match Ware Technologies, Inc., *AUTOSTAN Generalized Standardization System*, Silver Spring, Maryland (1993).