An introduction to fastLink for probabilistic record linkage

Anders Alexandersson
Florida Cancer Data System
Introduction
Why fastLink?

R package for fast probabilistic record linkage

- Correct: Outperforms deterministic methods
- Fast: Exponentially faster than R package RecordLinkage
- Reproducible: Easily integrated with R (D’oh!) and Stata – unlike Match*Pro
- Measures the linkage quality: Creates “confusion tables” – unlike Match*Pro
Syntax

fastLink(dfA, dfB, varnames [, options])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dfA</td>
<td>Dataset A - to be matched with Dataset B</td>
</tr>
<tr>
<td>dfB</td>
<td>Dataset B - to be matched with Dataset A</td>
</tr>
<tr>
<td>varnames</td>
<td>Names of matching variables</td>
</tr>
<tr>
<td>options</td>
<td>Optional arguments, e.g., return.all</td>
</tr>
</tbody>
</table>
A simple example

```r
> library(fastLink)
> data(samplematch)
>
> fastLink(
+   dfA = dfA, dfB = dfB,
+   varnames = c("firstname", "lastname", "city", "birthyear"),
+   )

===============

fastLink(): Fast Probabilistic Record Linkage
===============

If you set return.all to FALSE, you will not be able to calculate a confusion table as a summary statistic.
Calculating matches for each variable.
Getting counts for zeta parameters.
    Parallelizing calculation using OpenMP. 1 threads out of 8 are used.
Running the EM algorithm.
Getting the indices of estimated matches.
    Parallelizing calculation using OpenMP. 1 threads out of 8 are used.
Deduping the estimated matches.
Getting the match patterns for each estimated match.
$matches
  inds.a  inds.b
  1     64     3
  2    153     6
```
Correct: Outperforms deterministic methods

Advantage of probabilistic record linkage:

Can use poor-quality matching variables such as address, DOB, gender and name

Disadvantage:

Must estimate match and non-match values ("m- and u-values")
Fast: Exponentially faster than R package RecordLinkage

1) Get Match Patterns
Hashing: Compute match probability for each of the *observed* agreement patterns (not for all *possible* agreement patterns)

2) Count Unique Match Patterns
Parallelization: Count the patterns in parallel using OpenMP

3) Get Matched Pairs
Implemented in C++
fastLink is a regular R package.

The R package haven loads and saves Stata datasets.

Pre- and post-processing can be done in R or Stata.

Markdown and LaTeX let you combine code with annotations.

This presentation is done in Stata Markdown with the command markstat.
Measures the linkage quality: Create a confusion table

```r
> out <- fastLink(
+   dfA = dfA, dfB = dfB,
+   varnames = c("firstname", "lastname", "city", "birthyear"),
+   return.all = TRUE
+ )
```

fastLink(): Fast Probabilistic Record Linkage

Calculating matches for each variable.
Getting counts for zeta parameters.
Parallelizing calculation using OpenMP. 1 threads out of 8 are used.
Running the EM algorithm.
Getting the indices of estimated matches.
Parallelizing calculation using OpenMP. 1 threads out of 8 are used.
Deduping the estimated matches.
Getting the match patterns for each estimated match.
> confusion(out)
$confusion.table

<table>
<thead>
<tr>
<th></th>
<th><code>True</code> Matches</th>
<th><code>True</code> Non-Matches</th>
</tr>
</thead>
<tbody>
<tr>
<td>Declared Matches</td>
<td>50.0</td>
<td>0.0</td>
</tr>
<tr>
<td>Declared Non-Matches</td>
<td>0.3</td>
<td>299.7</td>
</tr>
</tbody>
</table>

$addition.info

<table>
<thead>
<tr>
<th>Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>Max Number of Obs to be Matched</td>
</tr>
<tr>
<td>Sensitivity (%)</td>
</tr>
<tr>
<td>Specificity (%)</td>
</tr>
<tr>
<td>Positive Predicted Value (%)</td>
</tr>
<tr>
<td>Negative Predicted Value (%)</td>
</tr>
<tr>
<td>False Positive Rate (%)</td>
</tr>
<tr>
<td>False Negative Rate (%)</td>
</tr>
<tr>
<td>Correctly Classified (%)</td>
</tr>
<tr>
<td>F1 Score (%)</td>
</tr>
</tbody>
</table>
fastLink details
Fellegi-Sunter model

Ivan Fellegi and Alan Sunter
“A Theory for Record Linkage”
(1969)

Calculate a
match weight =
\[ \ln(m/u)/\ln(2) \]
or a
non-match weight =
\[ \ln((1-m)/(1-u))/\ln(2) \].

Figure 1: Ivan Fellegi
fastLink step by step (optional functions)

1) Calculate agreement variable by variable: `gammapar*()`

2) Count unique agreement patterns: `tableCounts()`

3) Calculate Fellegi-Sunter weights: `emlinkMARmov()`

4) Find the matches: `matchesLink()`

5) Dedupe the matches: `dedupeMatches()`
Meet the developers from Princeton University

- Ted Enamorado; PhD 2018 (Expected) in Politics
- Ben Fifield; PhD 2018 (Expected) in Politics
- Kosuke Imai; Professor in Politics and Center for Statistics and Machine Learning

Figure 2: Kosuke Imai
TBT: Bayes’ Theorem

Bayes’ theorem (1763) is used to update our beliefs.

\[ P(A|B) \propto P(A) \times P(B|A) \]

The posterior is proportional to the prior times the likelihood.

**Figure 3:** Thomas Bayes?
A complex example: Read Stata datasets in R

The R package `haven` reads Stata datasets:

```r
> library(haven)
>
> patient_male <- read_dta("patient_male.dta")
> requestor_male <- read_dta("requestor_male.dta")
>
> patient_other <- read_dta("patient_other.dta")
> requestor_other <- read_dta("requestor_other.dta")
```
A complex example: Linkage for females, and hide the output

```r
> hide <- capture.output(out2 <- fastLink(
+   dfA = patient_other, dfB = requestor_other,
+   varnames = c("ssn", "zip", "stnum", "dob", "dblmf", "dblml"),
+   return.all = TRUE))
```

The R function `capture.output()` hides the default output.

The variable constructs are SSN + (Address + DOB + Names).

For example, “dblmf” is Double Metaphone of First Name.
A complex example: Confusion table for females

There are 658 links (declared matches) for females out of 10,603 possible links at the default matching threshold of 0.85 (not shown). The default threshold is for research, not for service provision. There are 646 links at the higher threshold of 0.98:

```r
> confusion(out2, threshold = 0.98)
$confusion.table

```

<table>
<thead>
<tr>
<th></th>
<th>'True' Matches</th>
<th>'True' Non-Matches</th>
</tr>
</thead>
<tbody>
<tr>
<td>Declared Matches</td>
<td>645.95</td>
<td>0.05</td>
</tr>
<tr>
<td>Declared Non-Matches</td>
<td>9.31</td>
<td>9947.69</td>
</tr>
</tbody>
</table>

```

$addition.info

```r
$results

Max Number of Obs to be Matched: 10603.00
Sensitivity (%): 98.58
Specificity (%): 100.00
Positive Predicted Value (%): 99.99
Negative Predicted Value (%): 99.91
False Positive Rate (%): 0.00
False Negative Rate (%): 1.42
Correctly Classified (%): 99.91
F1 Score (%): 99.28
```
Slide 5, “A simple example”, ended with:

$matches

  inda.a  inda.b
1     64     3
2    153     6

There were 50 declared matches (links) at the default matching threshold 0.85.

How to extract the matches as R dataframes:

```
> matchesA_other <- patient_other[out2$matches$inds.a,]
> matchesB_other <- requestor_other[out2$matches$inds.b,]
```
Post-processing
Create variable of matching pattern

The `getPatterns()` function stores the matching pattern as string variable `$patterns`. In most cases it is not needed to split the string variable.

```r
> matches_other <- matchesB_other
> matches_other$pattern <- do.call(paste, out2$patterns)
```
Add posterior and pid variables, and subset to matches

The posterior probabilities are stored in `$posterior`. The dataset should have the PID variables from both source datasets. Subset to matches at or above matching threshold 0.98.

```r
matches_other$posterior <- out2$posterior
matches_other$patient_id_number_n20 <- matchesA_other$pid
matches_other <- matches_other[out2$posterior >= .98,
+ c("patient_id_number_n20", "pid", "posterior", "pattern")]
```
Create a combined Stata dataset

Save R dataframe with matched females as Stata dataset:

```r
> save(matches_other, file="matches_other.Rda")
> tmp <-"matches_other.dta"
> write_dta(matches_other, tmp)
```

Repeat for males (not shown), and then combine in Stata:

```stata
. use matches_male, clear
. qui append using matches_other
. qui save matches, replace
```
Determine the “possibles”

Add confusion tables with different matching thresholds (posteriors) to do a sensitivity analysis.

Assume we are interested in posteriors 0.98-0.99 (aka credible range 98-99%). Then we can tabulate the matching patterns:

```
. use matches, clear
. tab pattern if inrange( posterior, 0.98, 0.99 ), mi

<table>
<thead>
<tr>
<th>pattern</th>
<th>Freq.</th>
<th>Percent</th>
<th>Cum.</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 2 2 0 2 2</td>
<td>272</td>
<td>95.10</td>
<td>95.10</td>
</tr>
<tr>
<td>2 0 0 0 0 2</td>
<td>11</td>
<td>3.85</td>
<td>98.95</td>
</tr>
<tr>
<td>2 NA NA 0 0 2</td>
<td>2</td>
<td>0.70</td>
<td>99.65</td>
</tr>
<tr>
<td>NA 2 NA 2 2 0</td>
<td>1</td>
<td>0.35</td>
<td>100.00</td>
</tr>
<tr>
<td>Total</td>
<td>286</td>
<td>100.00</td>
<td></td>
</tr>
</tbody>
</table>
```

So there are 286 “possibles” which may require clerical review.
- Vignette with many practical examples
- Aggregated confusion tables
- Better blocking features