Package ‘fastLink’

October 13, 2022

Type Package

Title Fast Probabilistic Record Linkage with Missing Data

Version 0.6.0

Date 2020-04-25

Description
Implements a Fellegi-Sunter probabilistic record linkage model that allows for missing data
and the inclusion of auxiliary information. This includes functionalities to conduct a merge of two
datasets under the Fellegi-Sunter model using the Expectation-
Maximization algorithm. In addition,
tools for preparing, adjusting, and summarizing data merges are included. The package implements methods
described in Enamorado, Fifield, and Imai (2019) "Using a Probabilistic Model to Assist Merging of
Large-scale Administrative Records", American Political Science Review and is available
at <http://imai.fas.harvard.edu/research/linkage.html>.

License GPL (>= 3)

Imports Matrix, parallel, foreach, doParallel, gtools, data.table,
stringdist, stringr, stringi, Rcpp (>= 0.12.7), FactoClass,
adagio, dplyr, plotrix, grDevices, graphics

Depends R (>= 2.14.0)

LinkingTo RcppArmadillo, Rcpp, RcppEigen

Encoding UTF-8

LazyData true

BugReports https://github.com/kosukeimai/fastLink/issues

RoxygenNote 7.0.2

Suggests testthat

NeedsCompilation yes

Author Ted Enamorado [aut, cre],
Ben Fifield [aut],
Kosuke Imai [aut]

Maintainer Ted Enamorado <ted.enamorado@gmail.com>

Repository CRAN

Date/Publication 2020-04-29 17:00:03 UTC

R topics documented:

`fastLink-package` *Fast Probabilistic Record Linkage with Missing Data*

Description

`fastLink` implements methods developed by Enamorado, Fifield, and Imai (2018) ”Using a Probabilistic Model to Assist Merging of Large-scale Administrative Records”, to probabilistically merge large datasets using the Fellegi-Sunter model while allowing for missing data and the inclusion of auxiliary information. The current version of this package conducts a merge of two datasets under the Fellegi-Sunter model, using the Expectation-Maximization Algorithm. In addition, tools for conducting and summarizing data merges are included.

Details

Package:	fastLink
Type:	Package
Version:	0.6.0-
Date:	2020-04-25
License:	GPL (>= 3)

Author(s)

Ted Enamorado `<ted.enamorado@gmail.com>`, Ben Fifield `<benfifield@gmail.com>`, and Kosuke Imai `<imai@harvard.edu>`

Maintainer: Ted Enamorado `<ted.enamorado@gmail.com>`

References

Enamorado, Ted, Ben Fifield and Kosuke Imai. (2019) "Using a Probabilistic Model to Assist Merging of Large-scale Administrative Records." American Political Science Review. Vol. 113, No. 2. Available at `https://imai.fas.harvard.edu/research/files/linkage.pdf`.

`aggconfusion` *aggconfusion*

Description

Aggregate confusion tables from separate runs of fastLink() (UNDER DEVELOPMENT)

Usage

```
aggconfusion(object)
```

Arguments

`object`	A list of confusion tables.

Value

'aggconfusion()' returns two tables - one calculating the confusion table, and another calculating a series of additional summary statistics.

Author(s)

Ted Enamorado <ted.enamorado@gmail.com> and Ben Fifield <benfifield@gmail.com>

`aggregateEM` *Aggregate EM objects for use in 'summary.fastLink()'*

Description

`aggregateEM` aggregates EM objects for easy processing by 'summary.fastLink()'

Usage

```
aggregateEM(em.list, within.geo)
```

Arguments

`em.list`	A list of 'fastLink' or 'fastLink.EM' objects that should be aggregate in 'summary.fastLink()'
`within.geo`	A vector of booleans corresponding to whether each object in 'em.list' is a within-geography match or an across-geography match. Should be of equal length to 'em.list'. Default is NULL (assumes all are within-geography matches).

`blockData` *blockData*

Description

Contains functionalities for blocking two data sets on one or more variables prior to conducting a merge.

Usage

```
blockData(dfA, dfB, varnames, window.block, window.size,
kmeans.block, nclusters, iter.max, n.cores)
```

Arguments

`dfA`	Dataset A - to be matched to Dataset B
`dfB`	Dataset B - to be matched to Dataset A
`varnames`	A vector of variable names to use for blocking. Must be present in both dfA and dfB
`window.block`	A vector of variable names indicating that the variable should be blocked using windowing blocking. Must be present in varnames.
`window.size`	The size of the window for window blocking. Default is 1 (observations +/- 1 on the specified variable will be blocked together).
`kmeans.block`	A vector of variable names indicating that the variable should be blocked using k-means blocking. Must be present in varnames.
`nclusters`	Number of clusters to create with k-means. Default value is the number of clusters where the average cluster size is 100,000 observations.
`iter.max`	Maximum number of iterations for the k-means algorithm to run. Default is 5000
`n.cores`	Number of cores to parallelize over. Default is NULL.

Value

A list with an entry for each block. Each list entry contains two vectors — one with the indices indicating the block members in dataset A, and another containing the indices indicating the block members in dataset B.

Examples

```
## Not run:
block_out <- blockData(dfA, dfB, varnames = c("city", "birthyear"))

## End(Not run)
```

`calcMoversPriors` *calcMoversPriors*

Description

calcMoversPriors calculates prior estimates of in-state and cross-state movers rates from the IRS SOI Migration data, which can be used to improve the accuracy of the EM algorithm.

Usage

```
calcMoversPriors(geo.a, geo.b, year.start, year.end,
county, state.a, state.b, matchrate.lambda, remove.instate)
```

Arguments

`geo.a`	The state code (if state = TRUE) or county name (if state = FALSE) for the earlier of the two voter files.
`geo.b`	The state code (if state = TRUE) or county name (if state = FALSE) for the later of the two voter files.
`year.start`	The year of the voter file for geography A.
`year.end`	The year of the voter file for geography B.
`county`	Whether prior is being calculated on the county or state level. Default is FALSE (for a state-level calculation).
`state.a`	If `county = TRUE` (indicating a county-level match), the state code of `geo.a`. Default is NULL.
`state.b`	If `county = TRUE` (indicating a county-level match), the state code of `geo.b`. Default is NULL.
`matchrate.lambda`	If TRUE, then returns the match rate for lambda (the expected share of observations in dataset A that can be found in dataset B). If FALSE, then returns the expected share of matches across all pairwise comparisons of datasets A and B. Default is FALSE
`remove.instate`	If TRUE, then for calculating cross-state movers rates assumes that successful matches have been subsetted out. The interpretation of the prior is then the match rate conditional on being an out-of-state or county mover. Default is TRUE.

Value

`calcMoversPriors` returns a list with estimates of the expected match rate, and of the expected in-state movers rate when matching within-state.

Author(s)

Ben Fifield <benfifield@gmail.com>

Examples

```
calcMoversPriors(geo.a = "CA", geo.b = "CA", year.start = 2014, year.end = 2015)
```

`clusterMatch` *clusterMatch*

Description

Creates properly sized clusters for matching, using either alphabetical or word embedding clustering. If using word embedding, the function first creates a word embedding out of the provided vectors, and then runs PCA on the matrix. It then takes the first `k` dimensions (where `k` is provided by the user) and k-means is run on that matrix to get the clusters.

Usage

```
clusterMatch(vecA, vecB, nclusters, max.n, word.embed, min.var,
weighted.kmeans, iter.max)
```

Arguments

`vecA`	The character vector from dataset A
`vecB`	The character vector from dataset B
`nclusters`	The number of clusters to create from the provided data. Either nclusters = NULL or max.n = NULL.
`max.n`	The maximum size of either dataset A or dataset B in the largest cluster. Either nclusters = NULL or max.n = NULL
`word.embed`	Whether to use word embedding clustering. Default is FALSE.
`min.var`	The minimum amount of explained variance (maximum = 1) a PCA dimension can provide in order to be included in k-means clustering when using word embedding. Default is .20.
`weighted.kmeans`	Whether to weight the k-means algorithm features by the explained variance of the included principal component when using word embedding clustering. Default is FALSE.
`iter.max`	Maximum number of iterations for the k-means algorithm.

Value

`clusterMatch` returns a list of length 3:

`clusterA`	The cluster assignments for dataset A
`clusterB`	The cluster assignments for dataset B
`n.clusters`	The number of clusters created
`kmeans`	The k-means object output.
`pca`	The PCA object output.
`dims.pca`	The number of dimensions from PCA used for the k-means clustering.

Author(s)

Ben Fifield <benfifield@gmail.com>

Examples

```
data(samplematch)
cl <- clusterMatch(dfA$firstname, dfB$firstname, nclusters = 3)
```

confusion — *Get confusion table for fastLink objects*

Description

Calculate confusion table after running fastLink().

Usage

```
confusion(object, threshold)
```

Arguments

`object`	A 'fastLink' object or list of fastLink objects. Can only be run if 'return.all = TRUE' in 'fastLink().'
`threshold`	The matching threshold above which a pair is a true match. Default is .85

Value

'confusion()' returns two tables - one calculating the confusion table, and another calculating a series of additional summary statistics.

Author(s)

Ted Enamorado <ted.enamorado@gmail.com> and Ben Fifield <benfifield@gmail.com>

Examples

```
## Not run:
 out <- fastLink(
 dfA = dfA, dfB = dfB,
 varnames = c("firstname", "middlename", "lastname"),
 stringdist.match = c("firstname", "middlename", "lastname"),
 partial.match = c("firstname", "lastname", "streetname"),
 return.all = TRUE)

 ct <- confusion(out)

## End(Not run)
```

`countyfips` *County-level FIPS Codes*

Description

This data maps county names to FIPS codes for use in calculating prior movers rates.

Usage

```
countyfips
```

Format

A dataframe containing 3235 observations.

`countyinflow` *County-level inflow rates by state*

Description

This data compiles and cleans county-level movers inflow rates by county, from the IRS Statistics on Income dataset.

Usage

```
countyinflow
```

Format

A dataframe containing 423752 observations.

`countyoutflow` *County-level outflow rates by state*

Description

This data compiles and cleans county-level movers outflow rates by county, from the IRS Statistics on Income dataset.

Usage

```
countyoutflow
```

Format

A dataframe containing 424475 observations.

`dedupeMatches` *dedupeMatches*

Description

Dedupe matched dataframes.

Usage

```
dedupeMatches(matchesA, matchesB, EM,
matchesLink, patterns, linprog)
```

Arguments

`matchesA`	A dataframe of the matched observations in dataset A, with all variables used to inform the match.
`matchesB`	A dataframe of the matched observations in dataset B, with all variables used to inform the match.
`EM`	The EM object from `emlinkMARmov()`
`matchesLink`	The output from `matchesLink()`
`patterns`	The output from `getPatterns()`.
`linprog`	Whether to implement Winkler's linear programming solution to the deduplication problem. Default is false.

Value

`dedupeMatches()` returns a list containing the following elements:

`matchesA`	A deduped version of matchesA
`matchesB`	A deduped version of matchesB
`EM`	A deduped version of the EM object

Author(s)

Ted Enamorado <ted.enamorado@gmail.com> and Ben Fifield <benfifield@gmail.com>

`dfA` *Sample dataset A*

Description

This data is a randomized and anonymized sample dataset to display features of fastLink.

Usage

```
dfA
```

Format

A dataframe containing 500 observations.

`dfB` *Sample dataset B*

Description

This data is a randomized and anonymized sample dataset to display features of fastLink.

Usage

```
dfB
```

Format

A dataframe containing 350 observations.

`emlinklog` *emlinklog*

Description

Expectation-Maximization algorithm for Record Linkage allowing for dependencies across linkage fields

Usage

```
emlinklog(patterns, nobs.a, nobs.b, p.m, p.gamma.j.m, p.gamma.j.u,
iter.max, tol, varnames)
```

Arguments

`patterns`	table that holds the counts for each unique agreement pattern. This object is produced by the function: tableCounts.
`nobs.a`	Number of observations in dataset A
`nobs.b`	Number of observations in dataset B
`p.m`	probability of finding a match. Default is 0.1
`p.gamma.j.m`	probability that conditional of being in the matched set we observed a specific agreement pattern.
`p.gamma.j.u`	probability that conditional of being in the non-matched set we observed a specific agreement pattern.
`iter.max`	Max number of iterations. Default is 5000
`tol`	Convergence tolerance. Default is 1e-05
`varnames`	The vector of variable names used for matching. Automatically provided if using `fastLink()` wrapper. Used for clean visualization of EM results in summary functions.

Value

`emlinklog` returns a list with the following components:

`zeta.j`	The posterior match probabilities for each unique pattern.
`p.m`	The probability of finding a match.
`p.u`	The probability of finding a non-match.
`p.gamma.j.m`	The probability of observing a particular agreement pattern conditional on being in the set of matches.
`p.gamma.j.u`	The probability of observing a particular agreement pattern conditional on being in the set of non-matches.
`patterns.w`	Counts of the agreement patterns observed, along with the Felligi-Sunter Weights.
`iter.converge`	The number of iterations it took the EM algorithm to converge.
`nobs.a`	The number of observations in dataset A.
`nobs.b`	The number of observations in dataset B.

Author(s)

Ted Enamorado <ted.enamorado@gmail.com> and Benjamin Fifield

Examples

```
## Not run:
## Calculate gammas
g1 <- gammaCKpar(dfA$firstname, dfB$firstname)
g2 <- gammaCKpar(dfA$middlename, dfB$middlename)
g3 <- gammaCKpar(dfA$lastname, dfB$lastname)
g4 <- gammaKpar(dfA$birthyear, dfB$birthyear)
```

```
## Run tableCounts
tc <- tableCounts(list(g1, g2, g3, g4), nobs.a = nrow(dfA), nobs.b = nrow(dfB))

## Run EM
em.log <- emlinklog(tc, nobs.a = nrow(dfA), nobs.b = nrow(dfB))

## End(Not run)
```

`emlinkMARmov` *emlinkMARmov*

Description

Expectation-Maximization algorithm for Record Linkage under the Missing at Random (MAR) assumption.

Usage

```
emlinkMARmov(patterns, nobs.a, nobs.b, p.m, iter.max,
tol, p.gamma.k.m, p.gamma.k.u, prior.lambda, w.lambda,
prior.pi, w.pi, address.field, gender.field, varnames)
```

Arguments

`patterns`	table that holds the counts for each unique agreement pattern. This object is produced by the function: tableCounts.
`nobs.a`	Number of observations in dataset A
`nobs.b`	Number of observations in dataset B
`p.m`	probability of finding a match. Default is 0.1
`iter.max`	Max number of iterations. Default is 5000
`tol`	Convergence tolerance. Default is 1e-05
`p.gamma.k.m`	probability that conditional of being in the matched set we observed a specific agreement value for field k.
`p.gamma.k.u`	probability that conditional of being in the non-matched set we observed a specific agreement value for field k.
`prior.lambda`	The prior probability of finding a match, derived from auxiliary data.
`w.lambda`	How much weight to give the prior on lambda versus the data. Must range between 0 (no weight on prior) and 1 (weight fully on prior)
`prior.pi`	The prior probability of the address field not matching, conditional on being in the matched set. To be used when the share of movers in the population is known with some certainty.
`w.pi`	How much weight to give the prior on pi versus the data. Must range between 0 (no weight on prior) and 1 (weight fully on prior)

`address.field`	Boolean indicators for whether a given field is an address field. Default is NULL (FALSE for all fields). Address fields should be set to TRUE while non-address fields are set to FALSE if provided.
`gender.field`	Boolean indicators for whether a given field is for gender. If so, exact match is conducted on gender. Default is NULL (FALSE for all fields). The one gender field should be set to TRUE while all other fields are set to FALSE if provided.
`varnames`	The vector of variable names used for matching. Automatically provided if using `fastLink()` wrapper. Used for clean visualization of EM results in summary functions.

Value

`emlinkMARmov` returns a list with the following components:

`zeta.j`	The posterior match probabilities for each unique pattern.
`p.m`	The probability of a pair matching.
`p.u`	The probability of a pair not matching.
`p.gamma.k.m`	The matching probability for a specific matching field.
`p.gamma.k.u`	The non-matching probability for a specific matching field.
`p.gamma.j.m`	The probability that a pair is in the matched set given a particular agreement pattern.
`p.gamma.j.u`	The probability that a pair is in the unmatched set given a particular agreement pattern.
`patterns.w`	Counts of the agreement patterns observed, along with the Felligi-Sunter Weights.
`iter.converge`	The number of iterations it took the EM algorithm to converge.
`nobs.a`	The number of observations in dataset A.
`nobs.b`	The number of observations in dataset B.

Author(s)

Ted Enamorado <ted.enamorado@gmail.com> and Kosuke Imai

Examples

```
## Not run:
## Calculate gammas
g1 <- gammaCKpar(dfA$firstname, dfB$firstname)
g2 <- gammaCKpar(dfA$middlename, dfB$middlename)
g3 <- gammaCKpar(dfA$lastname, dfB$lastname)
g4 <- gammaKpar(dfA$birthyear, dfB$birthyear)

## Run tableCounts
tc <- tableCounts(list(g1, g2, g3, g4), nobs.a = nrow(dfA), nobs.b = nrow(dfB))

## Run EM
em <- emlinkMARmov(tc, nobs.a = nrow(dfA), nobs.b = nrow(dfB))

## End(Not run)
```

`emlinkRS` *emlinkRS*

Description

Calculates Felligi-Sunter weights and posterior zeta probabilities for matching patterns observed in a larger population that are not present in a sub-sample used to estimate the EM.

Usage

```
emlinkRS(patterns.out, em.out, nobs.a, nobs.b)
```

Arguments

`patterns.out`	The output from 'tableCounts()' or 'emlinkMARmov()' (run on full dataset), containing all observed matching patterns in the full sample and the number of times that pattern is observed.
`em.out`	The output from 'emlinkMARmov()', an EM object estimated on a smaller random sample to apply to counts from a larger sample
`nobs.a`	Total number of observations in dataset A
`nobs.b`	Total number of observations in dataset B

Value

`emlinkMARmov` returns a list with the following components:

`zeta.j`	The posterior match probabilities for each unique pattern.
`p.m`	The posterior probability of a pair matching.
`p.u`	The posterior probability of a pair not matching.
`p.gamma.k.m`	The posterior of the matching probability for a specific matching field.
`p.gamma.k.u`	The posterior of the non-matching probability for a specific matching field.
`p.gamma.j.m`	The posterior probability that a pair is in the matched set given a particular agreement pattern.
`p.gamma.j.u`	The posterior probability that a pair is in the unmatched set given a particular agreement pattern.
`patterns.w`	Counts of the agreement patterns observed, along with the Felligi-Sunter Weights.
`iter.converge`	The number of iterations it took the EM algorithm to converge.
`nobs.a`	The number of observations in dataset A.
`nobs.b`	The number of observations in dataset B.

Author(s)

Ted Enamorado <ted.enamorado@gmail.com> and Ben Fifield <benfifield@gmail.com>

Examples

```
## Not run:
## -------------
## Run on subset
## -------------
dfA.s <- dfA[sample(1:nrow(dfA), 50),]; dfB.s <- dfB[sample(1:nrow(dfB), 50),]

## Calculate gammas
g1 <- gammaCKpar(dfA.s$firstname, dfB.s$firstname)
g2 <- gammaCKpar(dfA.s$middlename, dfB.s$middlename)
g3 <- gammaCKpar(dfA.s$lastname, dfB.s$lastname)
g4 <- gammaKpar(dfA.s$birthyear, dfB.s$birthyear)

## Run tableCounts
tc <- tableCounts(list(g1, g2, g3, g4), nobs.a = nrow(dfA.s), nobs.b = nrow(dfB.s))

## Run EM
em <- emlinkMAR(tc, nobs.a = nrow(dfA.s), nobs.b = nrow(dfB.s))

## ------------------
## Apply to full data
## ------------------

## Calculate gammas
g1 <- gammaCKpar(dfA$firstname, dfB$firstname)
g2 <- gammaCKpar(dfA$middlename, dfB$middlename)
g3 <- gammaCKpar(dfA$lastname, dfB$lastname)
g4 <- gammaKpar(dfA$birthyear, dfB$birthyear)

## Run tableCounts
tc <- tableCounts(list(g1, g2, g3, g4), nobs.a = nrow(dfA), nobs.b = nrow(dfB))

em.full <- emlinkRS(tc, em, nrow(dfA), nrow(dfB)

## End(Not run)
```

`fastLink` *fastLink*

Description

Run the fastLink algorithm to probabilistically match two datasets.

Usage

```
fastLink(dfA, dfB, varnames, stringdist.match,
stringdist.method, numeric.match, partial.match,
cut.a, cut.p, jw.weight,
cut.a.num, cut.p.num,
priors.obj, w.lambda, w.pi,
address.field, gender.field, estimate.only, em.obj,
dedupe.matches, linprog.dedupe,
reweight.names, firstname.field, cond.indep,
n.cores, tol.em, threshold.match, return.all, return.df, verbose)
```

Arguments

`dfA`	Dataset A - to be matched to Dataset B
`dfB`	Dataset B - to be matched to Dataset A
`varnames`	A vector of variable names to use for matching. Must be present in both dfA and dfB
`stringdist.match`	A vector of variable names indicating which variables should use string distance matching. Must be a subset of 'varnames' and must not be present in 'numeric.match'.
`stringdist.method`	String distance method for calculating similarity, options are: "jw" Jaro-Winkler (Default), "jaro" Jaro, and "lv" Edit
`numeric.match`	A vector of variable names indicating which variables should use numeric matching. Must be a subset of 'varnames' and must not be present in 'stringdist.match'.
`partial.match`	A vector of variable names indicating whether to include a partial matching category for the string distances. Must be a subset of 'varnames' and 'stringdist.match'.
`cut.a`	Lower bound for full string-distance match, ranging between 0 and 1. Default is 0.94
`cut.p`	Lower bound for partial string-distance match, ranging between 0 and 1. Default is 0.88
`jw.weight`	Parameter that describes the importance of the first characters of a string (only needed if stringdist.method = "jw"). Default is .10
`cut.a.num`	Lower bound for full numeric match. Default is 1
`cut.p.num`	Lower bound for partial numeric match. Default is 2.5
`priors.obj`	A list containing priors for auxiliary movers information, as output from calcMoversPriors(). Default is NULL
`w.lambda`	How much weight to give the prior on lambda versus the data. Must range between 0 (no weight on prior) and 1 (weight fully on prior). Default is NULL (no prior information provided).
`w.pi`	How much weight to give the prior on pi versus the data. Must range between 0 (no weight on prior) and 1 (weight fully on prior). Default is NULL (no prior information provided).
`address.field`	The name of the address field. To be used when 'pi.prior' is included in 'priors.obj'. Default is NULL (no matching variables should have address prior applied). Must be present in 'varnames'.

`gender.field`	The name of the field indicating gender. If provided, the exact-matching gender prior is used in the EM algorithm. Default is NULL (do not implement exact matching on gender). Must be present in 'varnames'.
`estimate.only`	Whether to stop running the algorithm after the EM step (omitting getting the matched indices of dataset A and dataset B). Only the EM object will be returned. Can be used when running the match on a random sample and applying to a larger dataset, or for out-of-sample prediction of matches. Default is FALSE.
`em.obj`	An EM object from a prior run of 'fastLink' or 'emlinkMARmov'. Parameter estimates will be applied to the matching patterns in 'dfA' and 'dfB'. If provided. 'estimate.only' is set to FALSE. Often provided when parameters have been estimated on a smaller sample, and the user wants to apply them to the full dataset. Default is NULL (EM will be estimated from matching patterns in 'dfA' and 'dfB').
`dedupe.matches`	Whether to dedupe the set of matches returned by the algorithm. Default is TRUE.
`linprog.dedupe`	If deduping matches, whether to use Winkler's linear programming solution to dedupe. Default is FALSE.
`reweight.names`	Whether to reweight the posterior match probabilities by the frequency of individual first names. Default is FALSE.
`firstname.field`	The name of the field indicating first name. Must be provided if reweight.names = TRUE.
`cond.indep`	Estimates for the parameters of interest are obtained from the Fellegi-Sunter model under conditional independence. Default is TRUE. If set to FALSE parameters estimates are obtained from a model that allows for dependencies across linkage fields.
`n.cores`	Number of cores to parallelize over. Default is NULL.
`tol.em`	Convergence tolerance for the EM Algorithm. Default is 1e-04.
`threshold.match`	A number between 0 and 1 indicating either the lower bound (if only one number provided) or the range of certainty that the user wants to declare a match. For instance, threshold.match = .85 will return all pairs with posterior probability greater than .85 as matches, while threshold.match = c(.85, .95) will return all pairs with posterior probability between .85 and .95 as matches.
`return.all`	Whether to return the most likely match for each observation in dfA and dfB. Overrides user setting of `threshold.match` by setting `threshold.match` to 0.0001, and automatically dedupes all matches. Default is TRUE.
`return.df`	Whether to return the entire dataframe of dfA and dfB instead of just the indices. Default is FALSE.
`verbose`	Whether to print elapsed time for each step. Default is FALSE.

Value

`fastLink` returns a list of class 'fastLink' containing the following components if calculating matches:

`matches`	An nmatches X 2 matrix containing the indices of the successful matches in `dfA` in the first column, and the indices of the corresponding successful matches in `dfB` in the second column.
`EM`	A list with the output of the EM algorithm, which contains the exact matching patterns and the associated posterior probabilities of a match for each matching pattern.
`patterns`	A matrix with the observed matching patterns for each successfully matched pair.
`nobs.a`	The number of observations in dataset A.
`nobs.b`	The number of observations in dataset B.
`zeta.name`	If reweighting by name, the posterior probability of a match for each match in dataset A and B.

If only running the EM and not returning the matched indices, `fastLink` only returns the EM object.

Author(s)

Ted Enamorado <ted.enamorado@gmail.com>, Ben Fifield <benfifield@gmail.com>, and Kosuke Imai

Examples

```
## Not run:
fl.out <- fastLink(dfA, dfB,
varnames = c("firstname", "lastname", "streetname", "birthyear"),
n.cores = 1)

## End(Not run)
```

`gammaCK2par` *gammaCK2par*

Description

Field comparisons for string variables. Two possible agreement patterns are considered: 0 total disagreement, 2 agreement. The distance between strings is calculated using a Jaro-Winkler distance.

Usage

```
gammaCK2par(matAp, matBp, n.cores, cut.a, method, w)
```

Arguments

`matAp`	vector storing the comparison field in data set 1
`matBp`	vector storing the comparison field in data set 2
`n.cores`	Number of cores to parallelize over. Default is NULL.
`cut.a`	Lower bound for full match, ranging between 0 and 1. Default is 0.92
`method`	String distance method, options are: "jw" Jaro-Winkler (Default), "jaro" Jaro, and "lv" Edit
`w`	Parameter that describes the importance of the first characters of a string (only needed if method = "jw"). Default is .10

Value

`gammaCK2par` returns a list with the indices corresponding to each matching pattern, which can be fed directly into `tableCounts` and `matchesLink`.

Author(s)

Ted Enamorado <ted.enamorado@gmail.com>, Ben Fifield <benfifield@gmail.com>, and Kosuke Imai

Examples

```
## Not run:
g1 <- gammaCK2par(dfA$firstname, dfB$lastname)

## End(Not run)
```

`gammaCKpar` *gammaCKpar*

Description

Field comparisons for string variables. Three possible agreement patterns are considered: 0 total disagreement, 1 partial agreement, 2 agreement. The distance between strings is calculated using a Jaro-Winkler distance.

Usage

```
gammaCKpar(matAp, matBp, n.cores, cut.a, cut.p, method, w)
```

Arguments

`matAp`	vector storing the comparison field in data set 1
`matBp`	vector storing the comparison field in data set 2
`n.cores`	Number of cores to parallelize over. Default is NULL.
`cut.a`	Lower bound for full match, ranging between 0 and 1. Default is 0.92
`cut.p`	Lower bound for partial match, ranging between 0 and 1. Default is 0.88
`method`	String distance method, options are: "jw" Jaro-Winkler (Default), "jaro" Jaro, and "lv" Edit
`w`	Parameter that describes the importance of the first characters of a string (only needed if method = "jw"). Default is .10

Value

`gammaCKpar` returns a list with the indices corresponding to each matching pattern, which can be fed directly into `tableCounts` and `matchesLink`.

Author(s)

Ted Enamorado <ted.enamorado@gmail.com>, Ben Fifield <benfifield@gmail.com>, and Kosuke Imai

Examples

```
## Not run:
g1 <- gammaCKpar(dfA$firstname, dfB$lastname)

## End(Not run)
```

`gammaKpar` *gammaKpar*

Description

Field comparisons: 0 disagreement, 2 total agreement.

Usage

```
gammaKpar(matAp, matBp, gender, n.cores)
```

Arguments

`matAp`	vector storing the comparison field in data set 1
`matBp`	vector storing the comparison field in data set 2
`gender`	Whether the matching variable is gender. Will override standard warnings of missingness/nonvariability. Default is FALSE.
`n.cores`	Number of cores to parallelize over. Default is NULL.

Value

`gammaKpar` returns a list with the indices corresponding to each matching pattern, which can be fed directly into `tableCounts` and `matchesLink`.

Author(s)

Ted Enamorado <ted.enamorado@gmail.com>, Ben Fifield <benfifield@gmail.com>, and Kosuke Imai

Examples

```
## Not run:
g1 <- gammaKpar(dfA$birthyear, dfB$birthyear)

## End(Not run)
```

`gammaNUMCK2par` *gammaNUMCK2par*

Description

Field comparisons for numeric variables. Two possible agreement patterns are considered: 0 total disagreement, 2 agreement. The distance between numbers is calculated using their absolute distance.

Usage

```
gammaNUMCK2par(matAp, matBp, n.cores, cut.a)
```

Arguments

`matAp`	vector storing the comparison field in data set 1
`matBp`	vector storing the comparison field in data set 2
`n.cores`	Number of cores to parallelize over. Default is NULL.
`cut.a`	Lower bound for full match. Default is 1

Value

`gammaNUMCK2par` returns a list with the indices corresponding to each matching pattern, which can be fed directly into `tableCounts` and `matchesLink`.

Author(s)

Ted Enamorado <ted.enamorado@gmail.com>, Ben Fifield <benfifield@gmail.com>, and Kosuke Imai

Examples

```
## Not run:
g1 <- gammaNUMCK2par(dfA$birthyear, dfB$birthyear)

## End(Not run)
```

---

gammaNUMCKpar *gammaNUMCKpar*

Description

Field comparisons for numeric variables. Three possible agreement patterns are considered: 0 total disagreement, 1 partial agreement, 2 agreement. The distance between numbers is calculated using their absolute distance.

Usage

```
gammaNUMCKpar(matAp, matBp, n.cores, cut.a, cut.p)
```

Arguments

`matAp`	vector storing the comparison field in data set 1
`matBp`	vector storing the comparison field in data set 2
`n.cores`	Number of cores to parallelize over. Default is NULL.
`cut.a`	Lower bound for full match. Default is 1
`cut.p`	Lower bound for partial match. Default is 2

Value

`gammaNUMCKpar` returns a list with the indices corresponding to each matching pattern, which can be fed directly into `tableCounts` and `matchesLink`.

Author(s)

Ted Enamorado <ted.enamorado@gmail.com>, Ben Fifield <benfifield@gmail.com>, and Kosuke Imai

Examples

```
## Not run:
g1 <- gammaNUMCKpar(dfA$birthyear, dfB$birthyear)

## End(Not run)
```

`getMatches` *getMatches*

Description

Subset two data frames to the matches returned by `fastLink()` or `matchesLink()`. Can also return a single deduped data frame if dfA and dfB are identical and fl.out is of class 'fastLink.dedupe'.

Usage

```
getMatches(dfA, dfB, fl.out, threshold.match, combine.dfs)
```

Arguments

`dfA`	Dataset A - matched to Dataset B by `fastLink()`.
`dfB`	Dataset B - matches to Dataset A by `fastLink()`.
`fl.out`	Either the output from `fastLink()` or `matchesLink()`.
`threshold.match`	A number between 0 and 1 indicating the lower bound that the user wants to declare a match. For instance, threshold.match = .85 will return all pairs with posterior probability greater than .85 as matches. Default is 0.85.
`combine.dfs`	Whether to combine the two data frames being merged into a single data frame. If FALSE, two data frames are returned in a list. Default is TRUE.

Value

`getMatches()` returns a list of two data frames:

`dfA.match`	A subset of `dfA` subsetted down to the successful matches.
`dfB.match`	A subset of `dfB` subsetted down to the successful matches.

Author(s)

Ben Fifield <benfifield@gmail.com>

Examples

```
## Not run:
fl.out <- fastLink(dfA, dfB,
varnames = c("firstname", "lastname", "streetname", "birthyear"),
n.cores = 1)
ret <- getMatches(dfA, dfB, fl.out)

## End(Not run)
```

`getPatterns` *getPatterns*

Description

Get the full matching patterns for all matched pairs in dataset A and dataset B

Usage

```
getPatterns(
  matchesA,
  matchesB,
  varnames,
  stringdist.match,
  numeric.match,
  partial.match,
  stringdist.method = "jw",
  cut.a = 0.92,
  cut.p = 0.88,
  jw.weight = 0.1,
  cut.a.num = 1,
  cut.p.num = 2.5
)
```

Arguments

`matchesA`	A dataframe of the matched observations in dataset A, with all variables used to inform the match.
`matchesB`	A dataframe of the matched observations in dataset B, with all variables used to inform the match.
`varnames`	A vector of variable names to use for matching. Must be present in both matchesA and matchesB.
`stringdist.match`	A vector of booleans, indicating whether to use string distance matching when determining matching patterns on each variable. Must be same length as varnames.
`numeric.match`	A vector of booleans, indicating whether to use numeric pairwise distance matching when determining matching patterns on each variable. Must be same length as varnames.
`partial.match`	A vector of booleans, indicating whether to include a partial matching category for the string distances. Must be same length as varnames. Default is FALSE for all variables.
`stringdist.method`	String distance method for calculating similarity, options are: "jw" Jaro-Winkler (Default), "jaro" Jaro, and "lv" Edit

`cut.a`	Lower bound for full string-distance match, ranging between 0 and 1. Default is 0.92
`cut.p`	Lower bound for partial string-distance match, ranging between 0 and 1. Default is 0.88
`jw.weight`	Parameter that describes the importance of the first characters of a string (only needed if stringdist.method = "jw"). Default is .10
`cut.a.num`	Lower bound for full numeric match. Default is 1
`cut.p.num`	Lower bound for partial numeric match. Default is 2.5

Value

`getPatterns()` returns a dataframe with a row for each matched pair, where each column indicates the matching pattern for each matching variable.

Author(s)

Ted Enamorado <ted.enamorado@gmail.com> and Ben Fifield <benfifield@gmail.com>

`getPosterior` *getPosterior*

Description

Get the posterior probability of a match for each matched pair of observations

Usage

```
getPosterior(matchesA, matchesB, EM, patterns)
```

Arguments

`matchesA`	A dataframe of the matched observations in dataset A, with all variables used to inform the match.
`matchesB`	A dataframe of the matched observations in dataset B, with all variables used to inform the match.
`EM`	The EM object from `emlinkMARmov()`
`patterns`	The output from `getPatterns()`.

Value

`getPosterior` returns the posterior probability of a match for each matched pair of observations in matchesA and matchesB

Author(s)

Ben Fifield <benfifield@gmail.com>

`inspectEM` *inspectEM*

Description

Inspect EM objects to analyze successfully and unsuccessfully matched patterns.

Usage

```
inspectEM(object, posterior.range, digits)
```

Arguments

`object`	The output from either `fastLink` or `emlinkMARmov`.
`posterior.range`	The range of posterior probabilities to display. Default is c(0.85, 1).
`digits`	How many digits to include in inspectEM dataframe. Default is 3.

Value

`inspectEM` returns a data frame with information about patterns around the provided threshold.

Author(s)

Ben Fifield <bfifield@princeton.edu>

`matchesLink` *matchesLink*

Description

matchesLink produces two dataframes that store all the pairs that share a pattern that conforms to the an interval of the Fellegi-Sunter weights

Usage

```
matchesLink(gammalist, nobs.a, nobs.b, em, thresh, n.cores = NULL)
```

Arguments

`gammalist`	A list of objects produced by either gammaKpar or gammaCKpar.
`nobs.a`	number of observations in dataset 1
`nobs.b`	number of observations in dataset 2
`em`	parameters obtained from the Expectation-Maximization algorithm under the MAR assumption. These estimates are produced by emlinkMARmov
`thresh`	is the interval of posterior zeta values for the agreements that we want to examine closer. Ranges between 0 and 1. Can be a vector of length 1 (from specified value to 1) or 2 (from first specified value to second specified value).
`n.cores`	Number of cores to parallelize over. Default is NULL.

Value

`matchesLink` returns an nmatches X 2 matrix with the indices of the matches rows in dataset A and dataset B.

Author(s)

Ted Enamorado <ted.enamorado@gmail.com>, Ben Fifield <benfifield@gmail.com>, and Kosuke Imai

Examples

```
## Not run:
## Calculate gammas
g1 <- gammaCKpar(dfA$firstname, dfB$firstname)
g2 <- gammaCKpar(dfA$middlename, dfB$middlename)
g3 <- gammaCKpar(dfA$lastname, dfB$lastname)
g4 <- gammaKpar(dfA$birthyear, dfB$birthyear)

## Run tableCounts
tc <- tableCounts(list(g1, g2, g3, g4), nobs.a = nrow(dfA), nobs.b = nrow(dfB))

## Run EM
em <- emlinkMAR(tc)

## Get matches
ml <- matchesLink(list(g1, g2, g3, g4), nobs.a = nrow(dfA), nobs.b = nrow(dfB),
em = em, thresh = .95)

## End(Not run)
```

`nameReweight` *nameReweight*

Description

Reweights posterior probabilities to account for observed frequency of names. Downweights posterior probability of match if first name is common, upweights if first name is uncommon.

Usage

```
nameReweight(dfA, dfB, EM, gammalist, matchesLink,
varnames, firstname.field, patterns, threshold.match, n.cores)
```

Arguments

`dfA`	The full version of dataset A that is being matched.
`dfB`	The full version of dataset B that is being matched.
`EM`	The EM object from `emlinkMARmov()`
`gammalist`	The list of gamma objects calculated on the full dataset that indicate matching patterns, which is fed into `tableCounts()` and `matchesLink()`.
`matchesLink`	The output from `matchesLink()`.
`varnames`	A vector of variable names to use for matching. Must be present in both matchesA and matchesB.
`firstname.field`	A vector of booleans, indicating whether each field indicates first name. TRUE if so, otherwise FALSE.
`patterns`	The output from `getPatterns()`.
`threshold.match`	A number between 0 and 1 indicating either the lower bound (if only one number provided) or the range of certainty that the user wants to declare a match. For instance, threshold.match = .85 will return all pairs with posterior probability greater than .85 as matches, while threshold.match = c(.85, .95) will return all pairs with posterior probability between .85 and .95 as matches.
`n.cores`	Number of cores to parallelize over. Default is NULL.

Value

`nameReweight()` returns a list containing the following elements:

`zetaA`	The reweighted zeta estimates for each matched element in dataset A.
`zetaB`	The reweighted zeta estimates for each matched element in dataset B.

Author(s)

Ted Enamorado <ted.enamorado@gmail.com> and Ben Fifield <benfifield@gmail.com>

`plot.fastLink` *Plot matching patterns of the EM object by posterior probability of match*

Description

`plot.fastLink()` plots the matching patterns of the EM object, ordering the matching patterns by the posterior probability of the match.

Usage

```
## S3 method for class 'fastLink'
plot(x, posterior.range, ...)
```

Arguments

`x`	Either a `fastLink` or `fastLink.EM` object to be plotted.
`posterior.range`	The range of posterior probabilities to display. Default is c(0.85, 1).
`...`	Further arguments to be passed to `plot.fastLink()`.

`preprocText` *preprocText*

Description

Preprocess text data such as names and addresses.

Usage

```
preprocText(text, convert_text, tolower, soundex,
usps_address, remove_whitespace, remove_punctuation, convert_text_to)
```

Arguments

`text`	A vector of text data to convert.
`convert_text`	Whether to convert text to the desired encoding, where the encoding is specified in the 'convert_text_to' argument. Default is TRUE
`tolower`	Whether to normalize the text to be all lowercase. Default is TRUE.
`soundex`	Whether to convert the field to the Census's soundex encoding. Default is FALSE.
`usps_address`	Whether to use USPS address standardization rules to clean address fields. Default is FALSE.

`remove_whitespace`	Whether to remove leading and trailing whitespace, and to convert multiple spaces to a single space. Default is TRUE.
`remove_punctuation`	Whether to remove punctuation from a string. Default is TRUE.
`convert_text_to`	Which encoding to use when converting text. Default is 'Latin-ASCII'. Full list of encodings in the `stri_trans_list()` function in the `stringi` package.

Value

`preprocText()` returns the preprocessed vector of text.

Author(s)

Ben Fifield <benfifield@gmail.com>

`print.inspectEM` *print.inspectEM*

Description

Print information from the EM algorithm to console.

Usage

```
## S3 method for class 'inspectEM'
print(x, ...)
```

Arguments

`x`	An `inspectEM` object
`...`	Further arguments to be passed to `print.fastLink()`.

`statefips` *State-level FIPS Codes*

Description

This data maps state names to FIPS codes for use in calculating prior movers rates.

Usage

```
statefips
```

Format

A dataframe containing 54 observations.

`stateinflow` *State-level inflow rates by state*

Description

This data compiles and cleans state-level movers inflow rates by state, from the IRS Statistics on Income dataset.

Usage

```
stateinflow
```

Format

A dataframe containing 11321 observations.

`statemove` *In-state movers rates by state*

Description

This data collects in-state movers rates by state, for imputation where within-county movers rates are not available.

Usage

```
statemove
```

Format

A dataframe containing 51 observations.

`stateoutflow` *State-level outflow rates by state*

Description

This data compiles and cleans state-level movers outflow rates by state, from the IRS Statistics on Income dataset.

Usage

```
stateoutflow
```

Format

A dataframe containing 11320 observations.

`stringSubset` *stringSubset*

Description

Removes as candidate matches any observations with no close matches on string-distance measures.

Usage

```
stringSubset(vecA, vecB, similarity.threshold, stringdist.method,
jw.weight, n.cores)
```

Arguments

`vecA`	A character or factor vector from dataset A
`vecB`	A character or factor vector from dataset B
`similarity.threshold`	Lower bound on string-distance measure for being considered a possible match. If an observation has no possible matches above this threshold, it is discarded from the match. Default is 0.8.
`stringdist.method`	The method to use for calculating string-distance similarity. Possible values are 'jaro' (Jaro Distance), 'jw' (Jaro-Winkler), and 'lv' (Levenshtein). Default is 'jw'.
`jw.weight`	Parameter that describes the importance of the first characters of a string (only needed if stringdist.method = "jw"). Default is .10.
`n.cores`	Number of cores to parallelize over. Default is NULL.

Value

A list of length two, where the both entries are a vector of indices to be included in the match from dataset A (entry 1) and dataset B (entry 2).

Examples

```
## Not run:
subset_out <- stringSubset(dfA$firstname, dfB$lastname, n.cores = 1)
fl_out <- fastLink(dfA[subset_out$dfA.block == 1,], dfB[subset_out$dfB.block == 1,],
varnames = c("firstname", "lastname", "streetname", "birthyear"), n.cores = 1)

## End(Not run)
```

summary.fastLink *Get summaries of fastLink() objects*

Description

summary.fastLink() calculates and outputs FDR, FNR, match counts, and match rates for estimated matches from a fastLink() object.

Usage

```
## S3 method for class 'fastLink'
summary(object, num.comparisons = 1,
thresholds = c(.95, .85, .75), weighted = TRUE, digits = 3, ...)
```

Arguments

object	Either a single ‘fastLink‘ or ‘fastLink.EM‘ object, or a list of ‘fastLink‘ or ‘fastLink.EM‘ objects to be aggregated together produced by ‘aggregateEM‘.
num.comparisons	The number of comparisons attempted for each observation in the across-geography match step. A correction factor to avoid multiple-counting. Default is NULL
thresholds	A vector of posterior probabilities to calculate the summary statistics.
weighted	Whether to weight the cross-geography matches on FDR and FNR.
digits	How many digits to include in summary object. Default is 3.
...	Further arguments to be passed to summary.fastLink().

tableCounts *tableCounts*

Description

Count pairs with the same pattern in the cross product between two datasets.

Usage

```
tableCounts(gammalist, nobs.a, nobs.b, n.cores)
```

Arguments

gammalist	A list of objects produced by gammaKpar, gammaCK2par, or gammaCKpar.
nobs.a	number of observations in dataset 1
nobs.b	number of observations in dataset 2
n.cores	Number of cores to parallelize over. Default is NULL.

Value

`tableCounts` returns counts of all unique mathching patterns, which can be fed directly into `emlinkMAR` to get posterior matching probabilities for each unique pattern.

Author(s)

Ted Enamorado <ted.enamorado@gmail.com>, Ben Fifield <benfifield@gmail.com>, and Kosuke Imai

Examples

```
## Not run:
## Calculate gammas
g1 <- gammaCKpar(dfA$firstname, dfB$firstname)
g2 <- gammaCKpar(dfA$middlename, dfB$middlename)
g3 <- gammaCKpar(dfA$lastname, dfB$lastname)
g4 <- gammaKpar(dfA$birthyear, dfB$birthyear)

## Run tableCounts
tc <- tableCounts(list(g1, g2, g3, g4), nobs.a = nrow(dfA), nobs.b = nrow(dfB))

## End(Not run)
```

Index