

# Module X: fastlink with blocking

Rebecca C. Steorts

# Reading

- ▶ Binette and Steorts (2020)
- ▶ Edmorando et al. (2020)
- ▶ Fellegi and Sunter (1969)

# Agenda

- ▶ We continue with our exploration of `fastLink` by adding blocking.
- ▶ We investigate this using the `RLdata10000` data set

# Load Packages

```
# load libraries  
library(fastLink)
```

# Load RLdata10000

```
# load RLdata10000
records <-
  read.table("data/RLdata10000.csv",
             sep=";", header=TRUE)
head(records, 4)
```

##	fname_c1	fname_c2	lname_c1	lname_c2	by	bm	bd	rec_id	ent_id
## 1	FRANK	<NA>	MUELLER	<NA>	1967	9	27	1	3606
## 2	MARTIN	<NA>	SCHWARZ	<NA>	1967	2	17	2	2560
## 3	HERBERT	<NA>	ZIMMERMANN	<NA>	1961	11	6	3	3892
## 4	HANS	<NA>	SCHMITT	<NA>	1945	8	14	4	329

## RLdata10000

```
# Number of unique records  
length(unique(records$ent_id))
```

```
## [1] 9000
```

# Linkage Fields

```
# linkage fields  
linkageFields <- c("fname_c1", "lname_c1", "by", "bm", "bd")
```

## Add Numeric Fields

```
# We can add numeric comparisons using dissimilarity  
numericMatchFields <- c("by")  
  
# Make sure these are class numeric  
records$by <- as.numeric(records$by)
```



# Preparation

```
# linkage fields
linkageFields <- c("fname_c1",
                  "lname_c1", "by", "bm", "bd")

# string distance fields
stringDistFields <- c("fname_c1", "lname_c1")

# partial distance fields (fields where we allow
# for agree, disagree, and partially agree)
partialMatchFields <- c("fname_c1", "lname_c1")
```

## Run fastLink

```
out2 <- fastLink(dfA = records,  
                 dfB = records,  
                 varnames = linkageFields,  
                 stringdist.match = stringDistFields,  
                 partial.match = partialMatchFields,  
                 cut.a.num = 1.5,  
                 cut.a = 0.94, cut.p = 0.84,  
                 dedupe = FALSE)
```

##

## =====

## fastLink(): Fast Probabilistic Record Linkage

## =====

##

## If you set return.all to FALSE, you will not be able to

## dfA and dfB are identical, assuming deduplication of a s

## Setting return.all to FALSE.

##

## Calculating matches for each variable

## How did we do?

```
recordsfL2 <- getMatches(dfA = records,  
                        dfB = records,  
                        fl.out = out2)  
  
# number of unique individuals that  
# fastLink finds  
length(unique(recordsfL2$dedupe.ids))  
  
## [1] 8983
```

# Blocking

1. We can use traditional, deterministic blocking, which is simple and easier but does not allow us to propagate error
2. We can also use probabilistic blocking, which allows us propagate (some) of the record linkage error

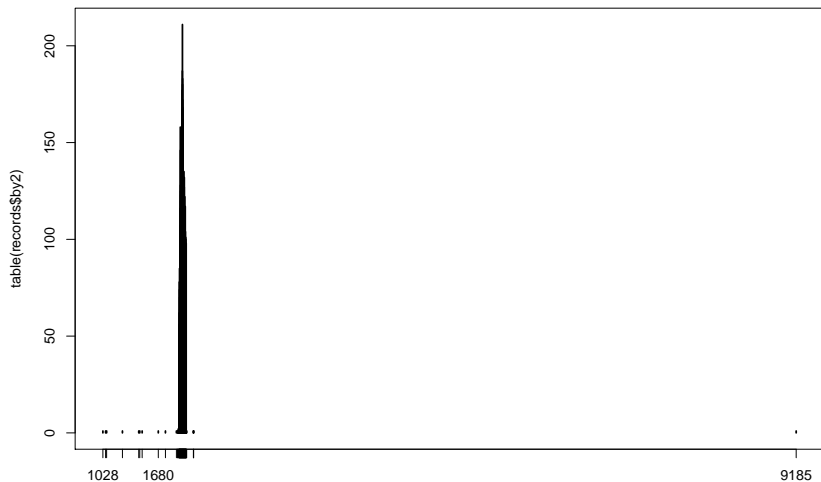
# Traditional Blocking

We will create a simplistic blocking scheme based upon date of birth year.

```
records$by2 <- records$by
```

# Plot

```
plot(table(records$by2))
```



# Traditional Blocking

We will filter out records that are not typical for date of birth year for computational reasons.

```
head(records$by2[records$by < 1924] <- 1923)
```

```
## [1] 1923
```

```
head(records$by2[records$by > 2008] <- 2009)
```

```
## [1] 2009
```

## Traditional Blocking

```
blockby <- blockData(records, records, varnames = "by2")

##
## =====
## blockData(): Blocking Methods for Record Linkage
## =====
##
## Blocking variables.
##     Blocking variable by2 using exact blocking.
##
## Combining blocked variables for final blocking assignment
```



## Traditional Blocking

```
# modify the list of linkage fields  
# birth year is of no use for  
# merging within a traditional block  
linkageFields2 <- c("fname_c1", "lname_c1", "bm", "bd")
```

## Traditional Blocking

```
# store the results from each block
results <- list()
for(j in 1:length(blockby)) {
  # subset original data to form block
  records.temp <- records[blockby[[j]]$dfA.inds, ]
  # fastLink applied to a block
  out.temp <- fastLink(dfA = records.temp, dfB =
    records.temp,
    varnames = linkageFields2,
    stringdist.match = stringDistFields,
    partial.match = partialMatchFields,
    cut.a = 0.92, cut.p = 0.84,
    threshold.match = 0.90,
    dedupe = FALSE)
  # get the data
  records.temp <-
    getMatches(dfA = records.temp,
      dfB = records.temp,
      stringdist.match = stringDistFields,
      partial.match = partialMatchFields,
      cut.a = 0.92, cut.p = 0.84,
      threshold.match = 0.90,
      dedupe = FALSE)
```