

## Module 8: Bayesian Fellegi and Sunter

Rebecca C. Steorts

# BDD package

- ▶ Marchant, Rubinstein, and Steorts (2021) have provided a package for implementing Sadinle (2014), which is referred to as BDD (Bayesian Duplicated Detection).

# RLdata500 data set

- We investigate it on the RLdata500 data set.

```
library(magrittr)      # pipe operator (must be loaded before BDD)
library(comparator)    # normalized Levenshtein distance
library(clevr)         # evaluation functions
library(BDD)
RLdata500[['rec_id']] <- seq.int(length.out=nrow(RLdata500))
head(RLdata500)
```

##	fname_c1	fname_c2	lname_c1	lname_c2	by	bm	bd	rec_id
## 1	CARSTEN	<NA>	MEIER	<NA>	1949	7	22	1
## 2	GERD	<NA>	BAUER	<NA>	1968	7	27	2
## 3	ROBERT	<NA>	HARTMANN	<NA>	1930	4	30	3
## 4	STEFAN	<NA>	WOLFF	<NA>	1957	9	2	4
## 5	RALF	<NA>	KRUEGER	<NA>	1966	1	13	5
## 6	JUERGEN	<NA>	FRANKE	<NA>	1929	7	4	6

# Distance functions

```
scoring_fns <- list(  
  fname_c1 = Levenshtein(normalize = TRUE),  
  lname_c1 = Levenshtein(normalize = TRUE),  
  by = function(x, y) abs(x - y),  
  bm = function(x, y) abs(x - y),  
  bd = function(x, y) abs(x - y)  
)
```

# Breaks

For each scoring function above, we provide a breaks vectors, which specifies the discrete levels of agreement (from 'high' agreement to 'low').

```
scoring_breaks <- list(  
  fname_c1 = c(-Inf, .05, .15, .3, Inf),  
  lname_c1 = c(-Inf, .05, .15, .3, Inf),  
  by = c(-Inf, 0, 1, 3, Inf),  
  bm = c(-Inf, 0, 1, 3, Inf),  
  bd = c(-Inf, 0, 2, 7, Inf)  
)
```

## Comparison vectors

- ▶ Now we are ready to compute the attribute comparison scores for the record pairs.
- ▶ Since this is a small data set, we consider all pairs using the `pairs_all` function.
- ▶ For larger data sets, blocking/indexing is recommended using `pairs_hamming`, `pairs_fuzzyblock`, or a custom blocking function.

## Comparison vectors

```
pairs <- pairs_all(RLdata500$rec_id) %>%  
  compute_scores(RLdata500, scoring_fns, id_col = 'rec_id')  
  discretize_scores(scoring_breaks)
```

# Speeding up inference

- ▶ To speed up inference, we only consider a subset of the pairs as candidate matches.
- ▶ Specifically, we consider pairs that have a strong agreement on name (accounting for missing names).



# Speeding up inference

To speed up inference, we only consider a subset of the pairs as candidate matches.

Specifically, we consider pairs that have a strong agreement on name (accounting for missing names).

```
pairs[['candidate']] <- (pairs$fname_c1 < 4) & (pairs$lname_c1 < 4) |  
  is.na(pairs$fname_c1) | is.na(pairs$lname_c1)
```

## priors on $m$ and $u$ probabilities

- ▶ Next we specify the priors on the  $m^*$  and  $u^*$  probabilities for each attribute and agreement level.
- ▶ `lambda` contains the lower truncation points for the truncated Beta priors on the  $m^*$  probabilities. `alpha1` and `beta1` are the shape parameters for the truncated Beta distributions to the BDD function below.
- ▶ The Beta priors on the  $u^*$  probabilities are uniform by default but can be adjusted using the `alpha0` and `beta0` arguments.

priors on  $m$  and  $u$  probabilities

```
lambda <- list(  
  fname_c1 = c(0.8,0.85,0.99),  
  lname_c1 = c(0.8,0.85,0.99),  
  by = c(0.8,0.85,0.99),  
  bm = c(0.8,0.85,0.99),  
  bd = c(0.8,0.85,0.99)  
)
```

## Initialization and Gibbs sampler

Finally we initialize the model and run inference using Markov chain Monte Carlo.

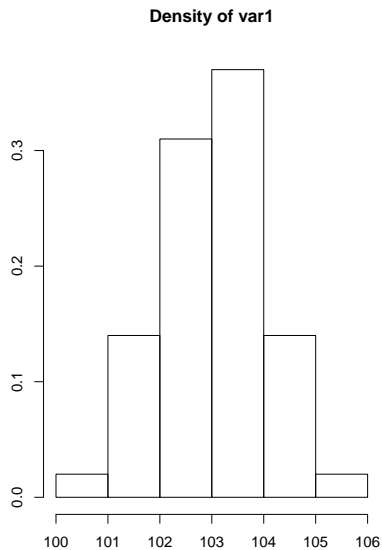
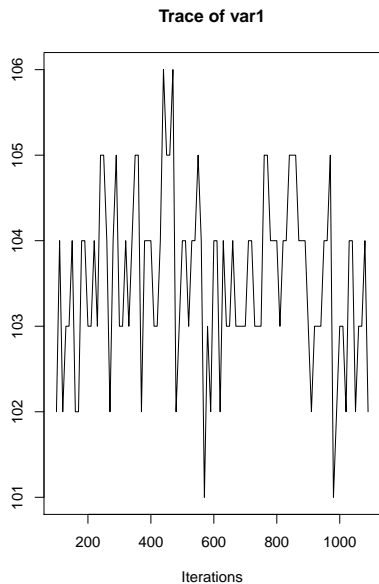
```
model <- BDD(pairs, lambda,  
            id_cols = c("rec_id.x", "rec_id.y"),  
            candidate_col = "candidate")  
fit <- run_inference(model, 100,  
                    thin_interval = 10,  
                    burnin_interval = 100)  
# Completed sampling in 0.7 seconds
```

## Posterior samples

- ▶ The posterior samples of the linkage structure can be accessed by calling `extract(fit, "links")`.
- ▶ However, these samples only cover the records that were considered as candidate pairs.
- ▶ We can obtain samples of the complete linkage structure (for all records) using the following function.

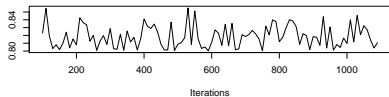
```
links_samples <-  
  complete_links_samples(fit, RLdata500$rec_id)
```

# Traceplot of the cluster sizes

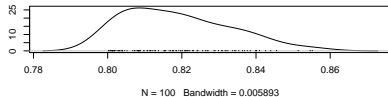


# Traceplot of the m probabilities

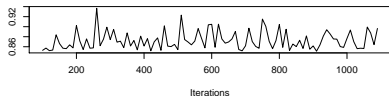
Trace of fname\_c1[1]



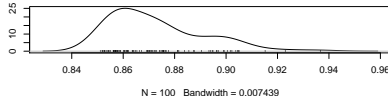
Density of fname\_c1[1]



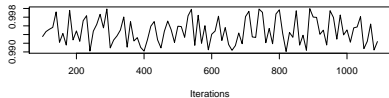
Trace of fname\_c1[2]



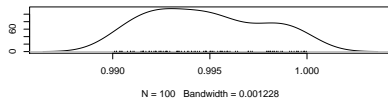
Density of fname\_c1[2]



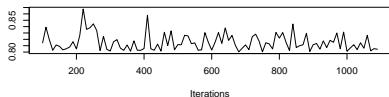
Trace of fname\_c1[3]



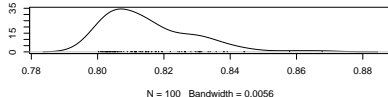
Density of fname\_c1[3]



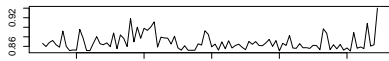
Trace of lname\_c1[1]



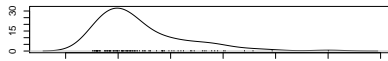
Density of lname\_c1[1]



Trace of lname\_c1[2]

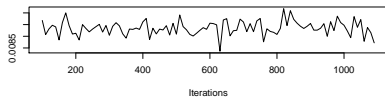


Density of lname\_c1[2]

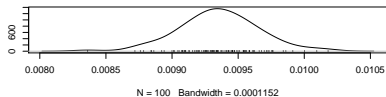


# Traceplot of the u probabilities

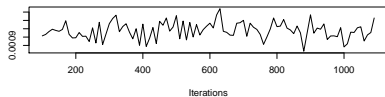
Trace of fname\_c1[1]



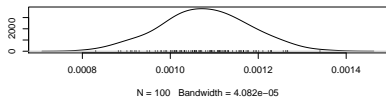
Density of fname\_c1[1]



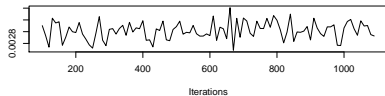
Trace of fname\_c1[2]



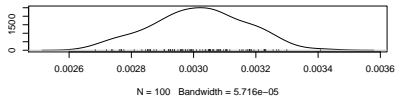
Density of fname\_c1[2]



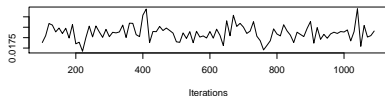
Trace of fname\_c1[3]



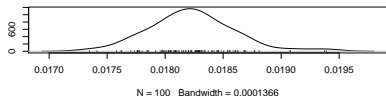
Density of fname\_c1[3]



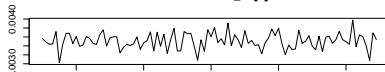
Trace of lname\_c1[1]



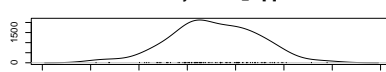
Density of lname\_c1[1]



Trace of lname\_c1[2]

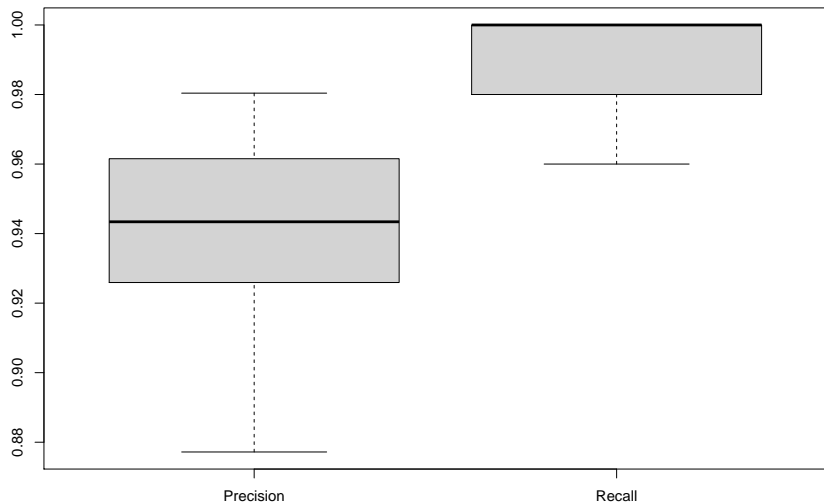


Density of lname\_c1[2]





# Pairwise Evaluation Metrics (Boxplots)



# Your Turn and Discussions

- ▶ What other evaluation metrics would you look at?
- ▶ Work on these either with a partner or in your free time?
- ▶ What types of evaluations can we look at given the fact that this is an unsupervised problem?

# Your Turn and Discussions

- ▶ You might think about how you would try and replicate the analysis that Sadinle did in his 2014 paper given this package.
- ▶ Can you think of a simulation study to study the sensitivity of the method?
- ▶ If you like this method, go read Sadinle (2017) and Sadinle (2018)!