Module 9: Bayesian Graphical Entity Resolution

Rebecca C. Steorts

Reading

- Binette and Steorts (2020)
- Steorts, Hall, Fienberg (2016)
- Steorts (2015)

What is "Bayesian"?

1. Setting up a *full probability model* – a joint probability distribution for all observable and unobservable quantities

$$\begin{aligned} p(\boldsymbol{x}|\boldsymbol{\theta}) - &\ \text{likelihood} \\ p(\boldsymbol{\theta}) - &\ \text{prior} \end{aligned}$$

2. Conditioning on observed data – calculating and interpreting the appropriate *posterior distribution*

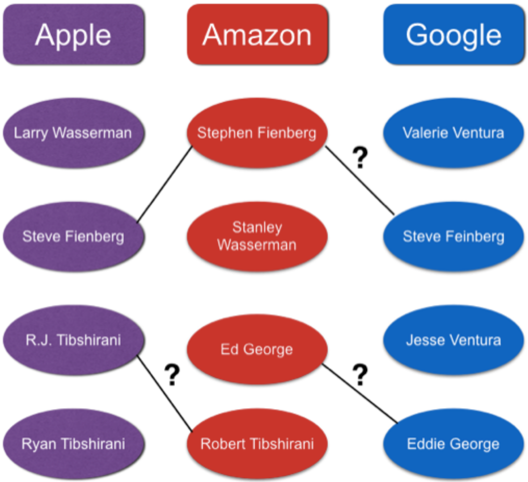
$$p(\boldsymbol{\theta}|\boldsymbol{x}) = \frac{p(\boldsymbol{x}, \boldsymbol{\theta})}{p(\boldsymbol{x})} = \frac{p(\boldsymbol{x}|\boldsymbol{\theta})p(\boldsymbol{\theta})}{p(\boldsymbol{x})} \propto p(\boldsymbol{x}|\boldsymbol{\theta})p(\boldsymbol{\theta})$$

Why Bayesian Entity Resolution

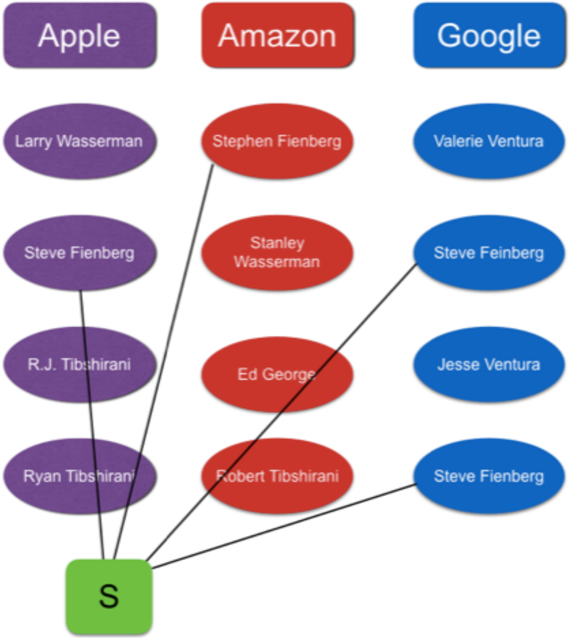
1. Entity resolution can be treated as a clustering problem.
2. Records are clustering to a latent entity.
3. This results in the model becoming a bipartite graph, which allows one to estimate latent individuals across multiple high dimensional databases.
4. The Bayesian paradigm naturally allows uncertainty quantification of the entity resolution process, a full posterior distribution, credible intervals, etc.
5. Theoretical properties have recently been explored for latent variable models, supporting the above approach.

[Copas and Hilton (1990), Tancredi and Liseo (2011), Steorts, Barnes, Neiswanger (2017), Zanella et al. (2016), Betancourt et al. (2020)]

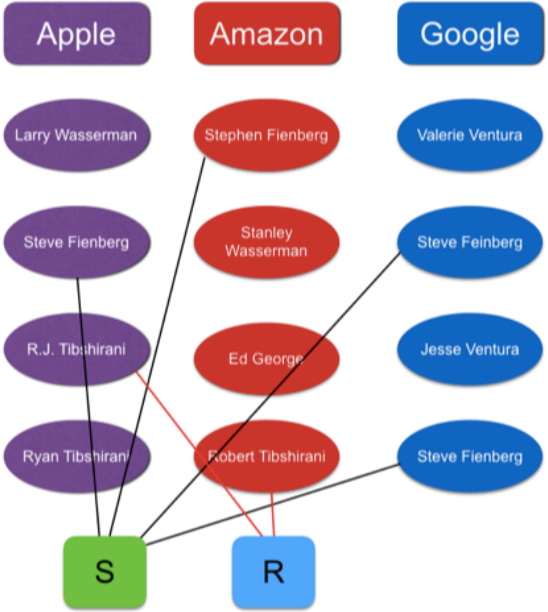
The entity resolution graph



The latent variable approach



The latent variable approach

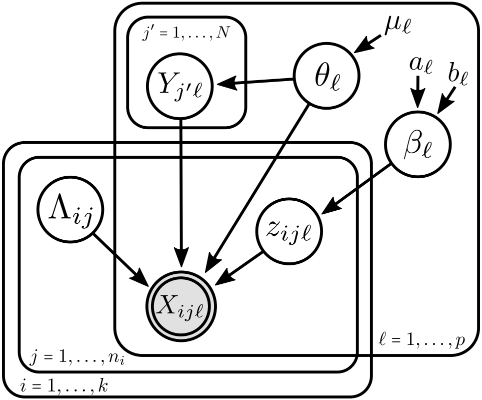


Notation

- $X_{ij\ell}$: observed value of the ℓth field for the jth record in the ith data set, $1 \leq i \leq k$ and $1 \leq j \leq n_i$.
- $Y_{j'\ell}$: true value of the ℓth field for the j'th latent individual.
- λ_{ij}: latent individual to which the jth record in the ith list corresponds. $\mathbf{\Lambda}$ is the collection of these values..
  - e.g. Five records in one list $\mathbf{\Lambda} = \{1, 1, 2, 3, 3\} \rightarrow 3$ latent entities or clusters.
- $z_{ij\ell}$: indicator of whether a distortion has occurred for record field value $X_{ij\ell}$

Graphical Record Linkage

Graphical model representation of Steorts et al. (2016):



- Λ_{ij} represents the linkage structure $\rightarrow$ uniform prior.
- Requires information about the number of latent entities a priori and it is very informative.

Bayesian Entity Resolution

Previous literature: not balanced regarding modeling, handling high-dimensional data, and uncertainty of multiple databases.

- Bayesian model: simultaneously links and de-duplicates.
- Assume records are noisy, distorted.
- We have a novel representation $(\boldsymbol{\Lambda})$: linkage structure.
- The strength of a Bayesian approach is that transitivity of the linked records is nearly automatic.
- Our data structure provides uncertainty estimates of linked records that can be propagated into later analyses.

[Steorts, Hall, and Fienberg (2014), Steorts, Hall, and Fienberg (2016)]

Empirically Motivated Priors

- The major weakness of Steorts, Hall, and Fienberg (2016) is the fact that it did not handle text (string) data.
- Steorts (2015) overcomes this issue by taking an empirical Bayesian approach, and making extensive comparisons to supervised methods.

Model Specification: String model

- The distortion of string-valued variables is modeled using a probabilistic mechanism based on some measure of distance between the true and distorted strings.

$$P(X_{ij\ell} = w|\lambda_{ij}, Y_{\lambda_{ij}\ell}, z_{ij\ell}) = \frac{\alpha_\ell \exp[-cd(w, Y_{\lambda_{ij}\ell})]}{\sum_{w \in S_l} \alpha_\ell \exp[-cd(w, Y_{\lambda_{ij}\ell})]}$$

where c is a parameter that needs to be specified and d represents a string metric distance e.g. Levenshtein or Jaro-Winkler.

Model Specification: Likelihood Function

$$X_{ij\ell} = w|\lambda_{ij}, Y_{\lambda_{ij}\ell}, z_{ij\ell} \overset{iid}{\sim} \begin{cases} \delta(Y_{\lambda_{ij}\ell}), & \text{if } z_{ij\ell} = 0 \\ F_\ell(Y_{\lambda_{ij}\ell}), & \text{if } z_{ij\ell} = 1 \text{ and } \ell \leq p_s \\ G_\ell, & \text{if } z_{ij\ell} = 1 \text{ and } \ell > p_s \end{cases}$$

- $z_{ij\ell} = 0$, then $X_{ij\ell} = Y_{\lambda_{ij\ell}}$
- F_ℓ is the string model in the last slide.
- G_ℓ is the empirical distribution function of the categorical data.

Model Specification: Hierarchical Model

$$Y_{\lambda_{ij}\ell} \overset{iid}{\sim} G_\ell$$

$$z_{ij\ell}|\beta_{i\ell} \overset{iid}{\sim} \text{Bernoulli}(\beta_{i\ell})$$

$$\beta_{i\ell} \overset{iid}{\sim} \text{Beta}(a, b)$$

$$\lambda_{ij} \overset{iid}{\sim} \text{DiscreteUniform}(1, \ldots, \text{N})$$

where a, b, N are unknown parameters that must be estimated or fixed.

- $\beta_{i\ell}$ represent the distortion probabilities of the fields.
- The parameters a and b for the Beta prior need to be specified.
- The number of latent entities or clusters needs to be specified in advance.

blink package

R package that removes duplicate entries from multiple databases using the empirical Bayes graphical method:

```
install.packages("blink")
```

- Formatting data for use with blink
- Tuning parameters
- Running the Gibbs sampler (estimate model parameters)
- Output

RLdata500 data

We will continue with the RLdata500 dataset in the blink package consisting of 500 records with 10% duplication.

```
library("blink")
```

```
## Loading required package: stringdist
```

```
## Loading required package: plyr
```

```
data("RLdata500") # load data
head(RLdata500) # take a look
```

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

Formatting the data

```r
# categorical variables
X.c <- as.matrix(RLdata500[, c("by","bm","bd")])

# string variables
X.s <- as.matrix(RLdata500[, c("fname_c1", "lname_c1")])
```

`X.c` and `X.s` include all files stacked on top of each other, for categorical and string variables respectively

```r
# keep track of which rows of are in which files
file.num <- rep(c(1, 2, 3), c(200, 150, 150))
```

Tuning parameters

Hyperparameters

```r
# Subjective choices for distortion probability prior
# parameters of a Beta(a,b)
a <- 1
b <- 999
```

Distortion

```r
# string distance function example
d <- function(s1, s2) {
  adist(s1, s2) # approximate string distance
}

# steepness parameter
c <- 1
```

Running the Gibbs sampler

```r
lam.gs <- rl.gibbs(file.num = file.num, # file
                   X.s = X.s, X.c = X.c, # data
                   num.gs = 1000, # iterations
                   a = a, b = b, # prior params
                   c = c, d = d, # distortion
                   M = 500) # max # latents
```

Evaluation

```r
# estimated pairwise links
est_links_pair <- pairwise(links(lam.gs))

# true pairwise links
true_links_pair <- pairwise(links(matrix(identity.RLdata500, nrow = 1)))

#comparison
comparison <- links.compare(est_links_pair, true_links_pair,
                            counts.only = TRUE)

# precision
precision <- comparison$correct/(comparison$incorrect + comparison$correct)

# recall
recall <- comparison$correct/(comparison$correct + comparison$missing)


# results
c(precision, recall)
```

```
## [1] 1.00 0.14
```

Your turn

Using the `title`, `authors`, `year`, and `journal` columns in the `cora` dataset from the `RLdata` package. First, we will load the necessary packages and data sets.

```
library(devtools)
```

```
## Loading required package: usethis
```

```
install_github("cleanzr/RLdata")
```

```
## Skipping install of 'RLdata' from a github remote, the S
##   Use `force = TRUE` to force installation
```

```
library(RLdata)
library(igraph)
```

```
##
## Attaching package: 'igraph'
```

```
## The following objects are masked from 'package:stats':
##
```

Your turn

1. Let's only use data with **complete cases** (for simplicity):

```
not_missing <-
  complete.cases(cora[, c("year", "journal", "title", "
```

2. Format the data to use with `blink`

   - Which columns are string vs. categorical?
   - Of the remaining data, assume that the rows 1-200 are from database 1, 201-400 are from database 2, and 401-560 are from database 3

3. Create tuning parameters for your model

   - Think about prior hyperparameters as well as the string distortion function

4. Run the Gibbs sampler 50 times to update the linkage structure

5. **Extra:** Evaluate your estimated linkage structure using precision and recall

Your turn (solution)

```
# 1. complete cases of data onl
# load data
not_missing <-
  complete.cases(cora[, c("year", "journal", "title", "authors")])

# 2. formatting data
# categorical variables
X.c <- as.matrix(cora[not_missing, c("year","journal")])

# string variables
X.s <- as.matrix(cora[not_missing, c("title", "authors")])

# keep track of which rows of are in which files
file.num <- rep(c(1, 2, 3), c(200, 200, 160))
```

Your turn (solution, cont'd)

```r
# 3. hyperparameters

# Subjective choices for distortion probability prior
# parameters of a Beta(a,b)
a <- 1
b <- 999

# string distance function example
d <- function(s1, s2) {
  adist(s1, s2) # approximate string distance
}

# steepness parameter
c <- 1

# 4. run the gibbs sampler
lam.gs <- rl.gibbs(file.num = file.num, # file
                   X.s = X.s, X.c = X.c, # data
                   num.gs = 10, # iterations
                   a = a, b = b, # prior params
                   c = c, d = d, # distortion
                   M = nrow(X.s)) # max # latents
```

Your turn (extra solution)

```r
# 5. (extra) evaluation

# estimated pairwise links
est_links_pair <- pairwise(links(lam.gs))

# true pairwise links, only for those included
# get true number
included_idx <- seq_len(sum(not_missing))
names(included_idx) <- cora[not_missing, "id"]
included_pairs <- subset(cora_gold, id1 %in% names(included_idx) & id2 %in% names(included_idx))

# get index for graphmaking
included_pairs <- apply(included_pairs, 2, function(col) {
  names <- as.character(col)
  included_idx[names]
})

# need list of pairs to compare
true_links_pair <- split(included_pairs, seq(nrow(included_pairs)))

#comparison
comparison <- links.compare(est_links_pair, true_links_pair, counts.only = TRUE)

# precision
precision <- comparison$correct/(comparison$incorrect + comparison$correct)

# recall
recall <- comparison$correct/(comparison$correct + comparison$missing)

# results
c(precision, recall)
```

```
## [1] 1.000000000 0.002596921
```

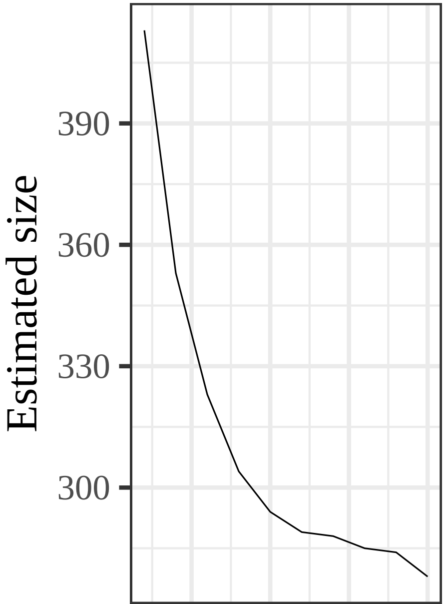
Your turn (extra solution, cont'd)

```r
# count how many unique latent individuals
size_est <- apply(lam.gs, 1, function(x) {
  length(unique(x))
  })


# get true number of individuals by using graph clusters
g <- make_empty_graph(length(included_idx))
g <- add_edges(g, as.vector(t(included_pairs)))
clust <- components(g, "weak")
```

Your turn (extra solution, cont'd)

```
## `stat_bin()` using `bins = 30`. Pick better value with `
```



Discussion

- Fully Bayesian approach for merging multiple databases simultaneously
- Handles transitive closures
- Beats semi-supervised approaches as illustrated in Steorts (2015)
- Speeds up can be handled in Marchant et al. (2021)
- Subjective priors instead of the uniform prior as discussed in Zanella et al. (2016) and Betancourt et al. (2021)