Efficient and Scalable Bipartite Matching with Fast Beta Linkage (fabl)

Brian Kundinger[a] Jerome Reiter[a] Rebecca C. Steorts [b]

[a] Department of Statistical Science, Duke University
[b] Department of Statistical Science and Computer Science, Duke University
Principal Mathematical Statistician, United States Census Bureau

October 4, 2021

Abstract

In this paper, we propose the *fast beta linkage* (`fabl`) methods, which extends a recent Bayesian Fellegi Sunter model to allow for parallel computing. In addition, we propose using hashing techniques to hasten calculations and to reduce the overall computational complexity. To achieve a scalable model, we propose independent priors in order to create computational speeds and gains, where we relax the one-to-one matching assumption. Second, we resolve the one-to-one matching issue via a post processing step. Third, we derive conditional distributions, a Gibbs sampler, and corresponding Bayes estimates for our proposed model. Fourth, we explore other computational speeds up based upon storage efficient indexing. Finally, we explore a sensitivity analysis and revisit the case study of the El Salvadoran conflict to understand the practical ramifications regarding our proposed modeling choices, where we provide comparisons to the BRL package in CRAN.

In this paper, we propose *fast beta linkage* (`fabl`), which extends a recent Bayesian Fellegi Sunter model for increased efficiency and scalability. Specifically, we relax the one-to-one matching requirement of the Beta Record Linkage method of (Sadinle, 2017) and propose independent priors over the matching space. This allows us to employ hashing techniques that hasten calculations and reduce computational complexity, complete pairwise record comparisons over large datasets through parallel computing, and reduce memory costs through a new technique called *storage efficient indexing*. Through simulation studies and a case study of homicides from the El Salvadoran Civil War, we show that our method has markedly increased speed with minimal loss of accuracy.

QUESTION: Does this abstract necessarily need to read like an outline?

Keywords: record linkage, bipartite record linkage, Bayesian methods, Gibbs sampling, hashing techniques, Markov chain Monte carlo, parallel/distributed computing

1 INTRODUCTION

Record linkage is the task of identifying duplicate records across multiple data sources, often in the absence of a unique identifier (Christen, 2012). Record linkage is an increasingly important task in "data cleaning," that is used for inferential and predictive analyses in fields such as statistics, computer science, machine learning, political science, economics, precision medicine, official statistics, and others. Many statistical record linkage methods and case studies are extensions of seminal work of Fellegi and Sunter (1969) and Newcombe et al. (1959). Specifically, Fellegi and Sunter (1969) created comparison vectors for each pair of records in the data and independently classified those pairs as a match or a non-match using a likelihood ratio test (or hypothesis test). Recent work in the statistical literature has extended the aforementioned work (Winkler and Thibaudeau, 1991; Fair, 2004; Wagner et al., 2014; Gill and Goldacre, 2003).

In this paper, we consider bipartite record linkage, which is commonly known as merging two databases that contain duplication across the two databases, but do not contain any duplications within each database (Sadinle, 2017). Much of the statistical literature focuses on bipartite record linkage (Fellegi and Sunter, 1969; Jaro, 1989; Winkler, 1988; Belin and Rubin, 1995; Larsen and Rubin, 2001; Tancredi and Liseo, 2011; Herzog et al., 2007b; Gutman et al., 2013; Sadinle, 2017).

The Fellegi-Sunter method and its extensions are widely popular mainly due to its simplicity and computational scalability. Despite this, like any method, it has limitations, that are well known in the literature. Specifically, the assumption of no duplications within a database implies a maximum one-to-one constraint. That is, a record from one database can be linked with only one record in the other database. Modern methods of Fellegi-Sunter either ignore this restriction (Winkler, 1988; Belin and Rubin, 1995; Larsen and Rubin, 2001), include this as a post-processing step (Jaro, 1989), or include it directly in the modeling framework (Sadinle, 2017). It is important to note that Fellegi and Sunter (1969) ignored this restriction as well. In practice, it seems that one should either directly include it in the modeling framework or correct for this in a post-processing step.

To our knowledge, Sadinle (2017) was the first to propose correcting the Fellegi and

Sunter model for the one-to-one constraint. Specifically, the authors propose a Bayesian variant of the original model that satisfies two goals. First, it allows one to quantify uncertainty regarding match or non-match status. In addition, posterior distributions are readily available to help estimate point estimates of interest (and quantity such estimates). Second, under the Bayesian paradigm, the one-to-one matching constraint is easily imbedded. The downside to this constraint is an added computational cost. The goal of our paper is proposing an extension of Sadinle (2017) that scales to large databases, but does not compromise much in terms of accuracy on the original case study considered by the author.

Our Contribution In this paper, we propose a Bayesian extension of Sadinle (2017) (BRL) through a modification to the model specification that allows for parallel computing of the linkage parameter, using hashing techniques to hasten calculations, and to reduce computational complexity. Specifically, Sadinle (2017) proposes a prior distribution that strictly enforces one-to-one matching. Instead, we propose independent priors (breaking the one-to-one matching) in order to create computational speeds and gains. Second, we resolve the one-to-one matching issue via a post processing step. Third, we derive conditional distributions, a Gibbs sampler, and corresponding Bayes estimates for our proposed model. Fourth, we explore other computational speeds up based upon storage efficient indexing. Finally, we explore a sensitivity analysis and revisit the case study of the El Salvadoran conflict to understand the practical ramifications regarding our proposed modeling choices. To recognize the lineage from the original `BRL` method, we name our method *fast beta linkage* (`fabl`, pronounced "fable")

The remainder of the paper is as follows. Section **??** reviews bipartite record linkage; section **??** reviews the traditional approach of the Fellegi and Sunter framework before introducing our proposed work.

2 BIPARTITE RECORD LINKAGE AND RELATED WORK

As already mentioned, bipartite record linkage is the traditional set up studied in record linkage, where we assume there are two databases $\boldsymbol{X}_1$ and $\boldsymbol{X}_2$ such that there are duplications across the databases but not within each databases. We assume that each database has n_1

and n_2 records, and that $n_1 \leq n_2$. Denote the number of entities simultaneously recorded by *both databases* by n_{12} such that $0 \leq n_{12} \leq n_2$. Given this set up, as explained by Steorts et al. (2016) and Sadinle (2017), this can be viewed as bipartite record linkage. This is important as it allows one to view the matching matrix or linkage structure in the following way:

$$\Delta_{ij} = \begin{cases} 1 & \text{if records } i \in \boldsymbol{X}_1 \text{ and } j \in \boldsymbol{X}_2 \text{ refer to the same entity;} \\ 0 & \text{otherwise.} \end{cases} \tag{1}$$

Bipartite matching can also be viewed as a matching labeling $\boldsymbol{Z} = (Z_1, \ldots, Z_{n_2})$ for the records in database $\boldsymbol{X}_2$ such that

$$Z_j = \begin{cases} i & \text{if records } i \in \boldsymbol{X}_1 \text{ and } j \in \boldsymbol{X}_2 \text{ refer to the same entity;} \\ n_1 + j & \text{if records } j \in \boldsymbol{X}_2 \text{ does not have a match in database } \boldsymbol{X}_1. \end{cases} \tag{2}$$

Depending on which representation is the most convenient, we can go back and forth between the two using $\Delta_{ij} = I(Z_j = i)$, where $I(\cdot)$ is the indicator function.

There are many approaches to bipartite record linkage, and we review some of the most common approaches below.

Fellegi and Sunter As already mentioned, the most common approach is the original approach of Fellegi and Sunter (1969). Enamorado et al. (2019a) have extended this approach using computational speeds ups to allow for a much more scalable and practical approach, providing open source code for the community.

Directly Modeling the Data Instead of relying on comparison vectors, some authors will incorporate the data directly into their bipartite record linkage model (Fortini et al., 2001; Matsakis, 2010; Liseo and Tancredi, 2011; Tancredi and Liseo, 2011; Gutman et al., 2013; Shan et al., 2020). These approaches have led to extensions regarding downstream tasks in the literature or joint models, such as regression approaches (Dalzell and Reiter, 2018; Steorts et al., 2018; Tancredi et al., 2020). There have been extensions to directly model the data in this literature where authors consider merging multiple databases (more

than two) (Steorts et al., 2016; Steorts, 2015; Zanella et al., 2016; Marchant et al., 2019; Tancredi and Liseo, 2015). We do not review this literature given that it is not directly/fairly comparable with the case of bipartite record linkage.

Classification Approaches One common way of approaching the record linkage task, especially in computer science and machine learning is using a classification approach. One uses a model (such as random forests) to classify records into matches/non-matches. When access to unique identifiers is available, one can create a training data (or reference data) set of record pairs (where the true match status is known). Finally, one can train a classifier using the reference data set to predict the match status on any record pairs that were not included in the reference data set (Bilenko et al., 2006; Christen, 2008; Ventura et al., 2015).

3 REVIEW OF THE FELLEGI SUNTER METHOD

In this section, we review notation, terminology, and the classical approach of Fellegi and Sunter.

Consider ordered record pairs $\boldsymbol{X}_1 \times \boldsymbol{X}_2$, where the size of the databases are n_1 and n_2, respectively. Denote the set of matches by $\boldsymbol{M} = \{(i,j) : i \in \boldsymbol{X}_1, j \in \boldsymbol{X}_2, \Delta_{ij} = 1\}$. Denote the set of non-matches $\boldsymbol{U} = \{(i,j) : i \in \boldsymbol{X}_1, j \in \boldsymbol{X}_2, \Delta_{ij} = 0\}$. An alternative way of merging two databases and removing duplications can be viewed as identifying the sets of matches/non-matches, or rather $\boldsymbol{M}, \boldsymbol{U}$. In the literature, it is well known that record pairs that are estimated as matches are known as links; record pairs that are estimated as non-matches are known as non-links. Fellegi and Sunter (1969) proposed using all-to-all pairwise comparison of records to estimate the matches/non-matches.

3.1 Comparison Data

In this section, we review the notion of *comparison data*. Intuitively, records that are co-referent or rather refer to the same entity should be similar; records that are non co-referent and refer to different entities should not be similar. A common way to encode this is using a comparison vector $\boldsymbol{\gamma}_{ij}$ which is computed for each record pair (i,j) in $\boldsymbol{X}_1 \times \boldsymbol{X}_2$. Denote F as

the criteria for compare the records such that $\boldsymbol{\gamma}_{ij} = (\gamma_{ij}^1, \gamma_{ij}^2, \ldots, \gamma_{ij}^f, \ldots \gamma_{ij}^F)$. In most cases, F represents to a single comparison per feature that the databases share, such as gender. The simplest way to compare two records is check for agreement or disagreement, and this is commonly used for nominal variables such as gender. For more complex measurements, such as textual or numeric feature information, we can take into account partial agreement patterns using common string metrics, such as the Edit, Jaro, or Jaro-Winkler distance functions. This allows us to divide our records into a set of similarity values into different levels of disagreement (Bilenko and Mooney, 2006; Elmagarmid et al., 2007).

Let $\mathcal{S}_f(i, j)$ denote a general similarity measure for feature f of records i and j, where the range of $\mathcal{S}_f$ can be divided into $L_f + 1$ intervals denoted by $I_{f0}, I_{f1}, \ldots, I_{fL_f}$, representing disagreement levels. Following convention, I_{f0} represents the highest level of agreement (inclusive of complete agreement) and I_{fL_f} represents the highest level of disagreement (including complete disagreement). Thus, we can construct comparison vectors in the following way:

$$\gamma_{ij} = \ell \text{ if } \mathcal{S}_f(i, j) \in I_{f\ell}.$$

The choice of $I_{f\ell}$ are application specific, which we discuss in our simulation and case study.

3.2 Blocking

In practice, it is not feasible to make all-to-all record comparisons as the computationally complexity is of the order $O(n_1 \times n_2)$. The most common solution is utilize blocking, which places similar records into partitions, clusters, or "blocks" to reduces this computational burden. Blocking can be deterministic, such as reducing the dimensionality of the comparison space on a highly reliable feature, such a gender. Blocking can be probabilistic, such as utilizing machine learning methods, known as hashing, which seek to use probabilistic functions (with low computational cost) to place similar records into the same bin. We propose hashing base methods for Bayesian bipartite record linkage, illustrating their ability to improve upon computational scalability. We emphasize that depending on the size of the problem, our proposed methodology can be used with or without hashing.

3.3 Fellegi-Sunter Model

In this section, we review the Fellegi-Sunter model. While we have introduced comparison data, $\boldsymbol{\gamma}_{ij}$, this is not sufficient to determine whether a record is a match/non-match given errors that naturally occur in the data. This motivated Fellegi and Sunter (1969) to consider the following likelihood ratio

$$w_{ij} = \log \frac{P(\gamma_{ij} \mid \Delta_{ij} = 1)}{P(\gamma_{ij} \mid \Delta_{ij} = 0)} \tag{3}$$

as a weight to estimate if record pairs are a match/non-match. Assume $\boldsymbol{\Gamma}_{ij}$ is a random vector, whose distribution depends on Δ_{ij}. Then equation 3 is a representative value of $\boldsymbol{\Gamma}_{ij} \mid \Delta_{ij}$. The ratio will be large if we favor the pair being a match. This is a problem in practice as transitive closures are violated. For example, suppose records i and j in $\boldsymbol{X}_1$ are extremely similar but are known to be non-matches. Suppose that both records i and j in $\boldsymbol{X}_1$ are very similar to record k in $\boldsymbol{X}_2$. By transitive closures, this implies that i and j are a match, which cannot be true by assumption.

Jaro (1989) proposed addressing this issue via an optimization approach as follows:

$$\begin{aligned} \max_{\Delta} \sum_{i=1}^{n_1} \sum_{j=1}^{n_2} w_{ij}\Delta_{ij} \quad &\text{subject to} \quad \Delta_{ij} \in \{0,1\}; \\ \sum_{i=1}^{n_1} \Delta_{ij} \leq 1, j = 1, 2, \ldots n_2; \; &\text{and} \\ \sum_{i=1}^{n_2} \Delta_{ij} \leq 1, j = 1, 2, \ldots n_1. & \end{aligned} \tag{4}$$

Equation 4 ensures that each record of $\boldsymbol{X}_2$ only matches with at most one record in $\boldsymbol{X}_1$ (and the reverse condition). This is commonly known as the *maximum-weight bipartite matching problem* or a *linear sum assignment problem*. The output of this algorithm is a bipartite matching that maximizes the sum of the weights among matched pairs, where the pairs are not matched are assumed to be non-matches. Jaro (1989) provided no theoretical justification for using this approach, however, Sadinle (2017) recently showed that under certain conditions, equation 4 is the maximum likelihood estimate.

3.4 Model Estimation

In our review, we have assumed one knows $P(\cdot \mid \Delta_{ij} = 1)$ and $P(\cdot \mid \Delta_{ij} = 0)$, however, these are typically not known in practice and must be estimated. Assume the comparison vectors are independent given the bipartite matching and assume that the match status of the record pairs are independent. It then follows that we can model the comparison data using mixture models in the following way:

$$\begin{aligned} \Gamma_{ij} = \gamma_{ij} \mid \Delta_{ij} = 1 &\overset{iid}{\sim} \mathcal{M}(\boldsymbol{m}), \\ \Gamma_{ij} = \gamma_{ij} \mid \Delta_{ij} = 0 &\overset{iid}{\sim} \mathcal{U}(\boldsymbol{u}), \\ \Delta_{ij} &\overset{iid}{\sim} \text{Bernoulli}(p), \end{aligned} \tag{5}$$

where p is the proportion of records that match. Note that $\boldsymbol{m}$ and $\boldsymbol{u}$ are vectors of the match/non-match parameters. One common way to estimate the unknown parameters of the aforementioned model is using the EM algorithm.

One criticism of using mixture models as proposed by Jaro (1989) is the fact that the decision rule will lead to "many-to-many assignments (breaking transitive closures). One way of resolving this issue to incorporate the constraint directly into the model (Fortini et al., 2001; Matsakis, 2010; Liseo and Tancredi, 2011; Tancredi and Liseo, 2011; Larsen, 2005, 2012; Gutman et al., 2013; Sadinle, 2017), which is appealing as one then obtains uncertainty quantification under the Bayesian paradigm, but often has to sacrifice scaling to very large databases. On the other hand, one can turn to post-processing methods in practical situations. In this paper, we attempt to bridge the best of both approaches. Specifically, we propose a model in the spirit of Sadinle (2017) and Enamorado et al. (2019b). We relax the one-to-one matching constraint proposed by Sadinle (2017) for purely computational reasons, and resolve this in a post-processing step. In addition, we propose our own computational speeds up and those in the spirit of Enamorado et al. (2019b). Our framework is then embarassingly parallel, available in the fabl package, and evaluated on a real case study and simulation studies illustrating that any loss in accuracy due to removing the constraint is minimal.

4 PROPOSED METHODOLOGY

In this section, we propose the fast beta linkage method (fabl) for bipartite record linkage, extending the work of Sadinle (2017). Specifically, we review the general approach of Sadinle (2017) and then provide our extensions for scalability.

4.1 Beta Linkage Model

In this section, we review the Beta Linkage Model, which can be written as follows:

$$\begin{aligned} \boldsymbol{\Gamma}_{ij} \mid Z_j = i &\overset{iid}{\sim} \mathcal{M}(\boldsymbol{m}) \\ \boldsymbol{\Gamma}_{ij} \mid Z_j \neq i &\overset{iid}{\sim} \mathcal{U}(\boldsymbol{u}) \\ \boldsymbol{Z} &\sim \mathcal{B}, \end{aligned} \tag{6}$$

where $\mathcal{B}$ represents a prior on the space of bipartite matchings and $\mathcal{M}(\boldsymbol{m}), \mathcal{M}(\boldsymbol{u})$ are models for the comparison vectors of matches and non-matches.

4.2 New Model Specification

In this section, we provide the specification for the models of $\mathcal{M}(\boldsymbol{m})$ and $\mathcal{U}(\boldsymbol{u})$, which follows from assuming that the features are conditionally independent of the level of agreement of another. The likelihood can be written as follows:

$$\mathcal{L}(\boldsymbol{Z}, \Phi \mid \boldsymbol{\Gamma}) = \prod_{j=1}^{n_1} \prod_{i=1}^{n_2} \left[\prod_{f=1}^{F} \prod_{l=1}^{L_f} m_{fl}^{I(Z_j=i)} u_{fl}^{I(Z_j \neq i)} \right]^{I(\gamma_{ij}^f = l)}, \tag{7}$$

where $m_{f\ell} = P(\Gamma_{ij}^f = \ell \mid Z_j = i)$ denotes the probability of a match have level ℓ of disagreement in feature f and $u_{f\ell} = P(\Gamma_{ij}^f = \ell \mid Z_j \neq i)$ represents the same probability for non-matches. In addition, denote $\boldsymbol{m}_f = (m_{f1}, \ldots, m_{fL_f})$, $\boldsymbol{u}_f = (u_{f1}, \ldots, u_{fL_f})$, $\boldsymbol{m} = (m_1, \ldots m_F)$, $\boldsymbol{u} = (u_1, \ldots u_F)$, and $\Phi = (\boldsymbol{m}, \boldsymbol{u})$.

Next, we allow for missing values or rather missing comparisons for corresponding record pairs. As commonly done in the record linkage literature, we assume that these occur missing at random [CITE: Little and Rubin (2002), Sadinle (2017), Steorts et al (2016)].

The likelihood becomes

$$\mathcal{L}(\boldsymbol{Z}, \Phi \mid \boldsymbol{\gamma}^{obs}) = \prod_{j=1}^{n_1} \prod_{i=1}^{n_2} \left[\prod_{f=1}^{F} \prod_{l=1}^{L_f} m_{fl}^{a_{f\ell}(\boldsymbol{Z})} u_{fl}^{a_{f\ell}(\boldsymbol{Z})} \right], \tag{8}$$

where

$$a_{f\ell}(\boldsymbol{Z}) = \sum_{i,j} I_{obs}(\gamma_{ij}^f) I(\gamma_{ij}^f = \ell) I(Z_j = i) \tag{9}$$

$$b_{f\ell}(\boldsymbol{Z}) = \sum_{i,j} I_{obs}(\gamma_{ij}^f) I(\gamma_{ij}^f = \ell) I(Z_j \neq i), \tag{10}$$

where I_{obs} denotes if a value is observed or not. Next, for a particular matching configuration $\boldsymbol{Z}$, $a_{f\ell}(\boldsymbol{Z})$ and $b_{f\ell}(\boldsymbol{Z})$ denote the number of matches and non-matches with observed disagreement level ℓ for comparison feature f. Using this parameterization, we specify independent conjugate priors for all features f:

$$\boldsymbol{m}_f \sim \text{Dirichlet}(\alpha_{f0}, \ldots, \alpha_{fL_f}) \tag{11}$$

$$\boldsymbol{u}_f \sim \text{Dirichlet}(\beta_{f0}, \ldots, \beta_{fL_f}). \tag{12}$$

4.3 Fast Beta Prior

In this section, we deviate from the approach of Sadinle (2017), and propose a more scalable prior on the matching labelings $\boldsymbol{Z} = (Z_1, Z_2, \ldots Z_{n_2})$, where recall $Z_j \in \{1, 2, 3, n_1, n_1 + j\}$.

While the rest of the model has been proposed previously, to our knowledge, this particular prior has not been proposed before.

Brian: could you please lay out your proposal here more clear regarding the notation above and add as much details as possible regarding your approach. More detail is better than less. It will be important to contrast this with the prior that Sadinle uses and note that you're dropping the constraint. (Make sure to include any proofs/derivations that might not be obvious to a reader given this is a major contribution to the paper).

$$Z_j|\lambda = \begin{cases} \frac{1}{n_1}\lambda & z_j \leq n_1; \\ 1 - \lambda & z_j = n_1 + j \end{cases}$$

$$\lambda \sim \text{Beta}(\alpha_\lambda, \beta_\lambda)$$

Intuitively, this set of priors says that record $j \in \boldsymbol{X}_2$ has some match in $\boldsymbol{X}_1$ with probability λ, and that each record $i \in \boldsymbol{X}_1$ is equally likely to be that match. λ itself is given a prior distribution to reflect prior beliefs about the overall rate of matchingness in the data. One choice of uninformative prior might be $\lambda \sim \text{Beta}(1, 1)$, which corresponds to a prior belief that nonmatches and matches are equally likely, and another might be $\lambda \sim \text{Beta}\left(1, \frac{1}{n_1}\right)$, which corresponds to a uniform prior on the labeling of $\mathbf{Z}$.

There is a clear relationship between our proposed model and that of Sadinle (2017). Sadinle (2017) constructs a prior distribution on the entire $\mathbf{Z}$ vector, called the *beta distribution for bipartite matching*, given by

$$P(\mathbf{Z}|\alpha_\pi, \beta_\pi) = \frac{(n_1 - n_{12}(\mathbf{Z}))!}{n_1!} \frac{\text{B}(n_{12}(\mathbf{Z}) + \alpha_\pi, n_2 - \mathbf{Z}) - \beta_\pi])}{\text{B}(\alpha_\pi, \beta_\pi)}$$

where $\text{B}(\cdot, \cdot)$ represents the Beta function. This prior induces a Gibbs sampler that strictly enforces one-to-one matching, previously matched records from the set of candidate records when sampling Z_j. This creates a dependency that makes the sampler *inherently serial*.

On the other hand, we propose independent priors for each Z_j, creating a sampler that is *embarassingly parallel*, allowing for significant computational gains. More importantly, since only the agreement pattern of Z_j is used for calculations within the Gibbs sampler, and not the particular record label, this sampling is only needed at the level of the unique agreement patterns, which offers additional computational savings. In doing so however, we thereby weaken the one-to-one requirement from `BRL`; our sampler does ensure that each record in $\boldsymbol{X}_2$ can be matched to at most one record in $\boldsymbol{X}_1$, but allows for the possibility that multiple records in $\boldsymbol{X}_2$ match to the same record in $\boldsymbol{X}_1$. We resolve contradictions when calcuating the Bayes estimate, which is explained in Section 4.5

NOTE TO BEKA: I think it will be best to do the "post-processing step" along with the Bayes Estimate. Its equivalent to minimizing the posterior loss subject to the constraint that $Z_j \neq Z_{j'}, \forall j \neq j'$.

4.4 Gibbs Sampler

We work with the following factorization of the joint posterior distribution:

Brian: The Gibbs sampler has some minor typos. It should be written in terms of $\boldsymbol{Z}, \Phi, \lambda \mid \gamma_{obs}$. Also, just make the rest of the notation match please:)

$$\begin{aligned} p(\mathbf{Z}, \mathbf{m}, \mathbf{u}, \lambda | \Gamma) &\propto p(\Gamma | \mathbf{Z}, \mathbf{m}, \mathbf{u}) p(\mathbf{Z} | \lambda) p(\mathbf{m}, \mathbf{u}) p(\lambda) \\ &\propto \prod_{j=1}^{n_B} \prod_{i=1}^{n_A} \left[\prod_{f=1}^{F} \prod_{l=1}^{L_f} m_{fl}^{I(Z_j = i)} u_{fl}^{I(Z_j \neq i)} \right] \\ &\times \prod_{f=1}^{F} \prod_{l=1}^{L_f} m_{fl}^{\alpha_{fl}-1} \times \prod_{f=1}^{F} \prod_{l=1}^{L_f} u_{fl}^{\beta_{fl}-1} \\ &\times \prod_{j=1}^{n_B} \left[I(Z_j \leq n_A) \frac{1}{n_A} \lambda + I(Z_j = n_A + 1)(1 - \lambda) \right] \end{aligned}$$

This factorization leads to following Gibbs Sampler:

Sample $\mathbf{m}^{(s+1)}$ $\mathbf{u}^{(s+1)} | \Gamma, \mathbf{Z}^{(s)}$: The $\mathbf{m}$ and $\mathbf{u}$ parameters are updated through standard multinomial-dirichlet mechanics. Thus we have

$$\mathbf{m}_f | \mathbf{Z}, \Gamma \sim \text{Dirichlet}(\alpha_{f1}(\mathbf{Z}), \ldots, \alpha_{fL_f}(\mathbf{Z}))$$

$$\mathbf{u}_f | \mathbf{Z}, \Gamma \sim \text{Dirichlet}(\beta_{f1}(\mathbf{Z}), \ldots, \beta_{fL_f}(\mathbf{Z}))$$

where $\alpha_{fl}(\mathbf{Z}) = \alpha_{fl} + \sum_{i,j} I(\gamma_{ij}^f = l) I(z_j = i)$ and $\beta_{fl}(\mathbf{Z}) = \beta_{fl} + I(\gamma_{ij}^f = l) I(z_j \neq i)$.

Sample $\lambda^{(s+1)} | \mathbf{Z}^{(s)}$: As a function of λ, the linkage structure parameter $\mathbf{Z}$ is sequence of successes (when $z_j < n_A + 1$) and failures (when $z_j = n_A + 1$), and therefore $p(\mathbf{Z}|\lambda) = \mathcal{L}(\lambda | \mathbf{Z})$ is determined only by the number of duplicates $D = \sum_{i=1}^{n_B} \mathbf{1}_{z_j < n_A + 1}$ encoded by $\mathbf{Z}$. Thus we have

$$\begin{aligned} p(\lambda | \mathbf{Z}) &\propto p(\mathbf{Z} | \lambda) p(\lambda) \\ &\propto \lambda^D (1 - \lambda)^{n_B - D} \lambda^{\alpha_\lambda - 1} (1 - \lambda)^{\beta_\lambda - 1} \\ &\propto \lambda^{D + \alpha_\lambda - 1} (1 - \lambda)^{n_B - D + \beta_\lambda - 1} \end{aligned}$$

$$\implies \lambda^{(s+1)}|\mathbf{Z}^{(s+1)} \sim \text{Beta}(D + \alpha_\lambda, n_B - D + \beta_\lambda)$$

Sample $\mathbf{Z}^{(s+1)}|\Gamma, \mathbf{m}^{(s+1)}, \mathbf{u}^{(s+1)}, \lambda^{(s+1)}$: Because we sample Z_j independently of all other $Z_{j'}$, we use only the full conditional for an individual Z_j. Let $\Gamma_{\cdot j}$ denote the set of n_A comparison vectors with $j \in B$, and note that as a function of Z_j, the likelihood $p(\Gamma_{\cdot j}|Z_j, \mathbf{m}, \mathbf{u}) = \mathcal{L}(Z_j|\Gamma_{\cdot j}, \mathbf{m}, \mathbf{u})$ is a discrete distribution with probabilities proportional to

$$\begin{aligned} p(\Gamma_{\cdot j}|Z_j = z_j, \mathbf{m}, \mathbf{u}) &\propto \prod_{i=1}^{n_A} \left[\prod_{f=1}^{F} \prod_{l=1}^{L_f} m_{fl}^{I(Z_j=i)} u_{fl}^{I(Z_j \neq i)} \right]^{I(\gamma_{ij}^f = l)} \\ &\propto \prod_{i=1}^{n_A} \left(\prod_{f=1}^{F} \prod_{l=1}^{L_f} \frac{m_{fl}}{u_{fl}} \right)^{I(z_j=i, \gamma_{ij}^f = l)} \\ &= \begin{cases} w_{ij} & z_j \leq n_A; \\ 1 & z_j = n_A + 1 \end{cases} \end{aligned}$$

where $w_{ij} = \left(\frac{\prod_{f=1}^{F} \prod_{l=1}^{L_f} m_{fl}}{\prod_{f=1}^{F} \prod_{l=1}^{L_f} u_{fl}} \right)^{I(\gamma_{ij}^f = l)} = \frac{P(\boldsymbol{\gamma_{ij}}|Z_j=i)}{P(\boldsymbol{\gamma_{ij}}|Z_j \neq i)}$. The interested reader should note that these are precisely the likelihood ratios used in the Fellegi-Sunter model to classify matches and non-matches, and we therefore refer to w_{ij} as the *Fellegi Sunter weights*.

With the likelihood in this form, we can derive the full conditional

$$p(Z_j|\Gamma_{\cdot j}, \mathbf{m}, \mathbf{u}, \lambda) \propto p(\Gamma_{\cdot j}|Z_j, \mathbf{m}, \mathbf{u}) P(Z_j|\lambda)$$

$$\begin{aligned} &\propto \left(\sum_{i=1}^{n_A} w_{ij} \mathbf{1}_{z_j=i} + \mathbf{1}_{z_j=n_A+1} \right) \left(\lambda \sum_{i=1}^{n_A} \frac{1}{n_A} \mathbf{1}_{z_j=i} + (1-\lambda) \mathbf{1}_{z_j=n_A+1} \right) \\ &= \frac{\lambda}{n_A} \sum_{i=1}^{n_A} w_{ij} \mathbf{1}_{z_j=i} + (1-\lambda) \mathbf{1}_{z_j=n_A+1} \\ \implies Z_j^{(s+1)}|\mathbf{m}, \mathbf{u}, \Gamma, \lambda &\propto \begin{cases} \frac{\lambda}{n_A} w_{ij} & z_j \leq n_A; \\ 1-\lambda & z_j = n_A + 1 \end{cases} \end{aligned}$$

In order to make fair comparisons against the (Sadinle, 2017) model, we integrate over the posterior of λ and rearrange terms to produce the final full conditional:

$$p\left(Z_j^{(s+1)} = i | \mathbf{m}, \mathbf{u}, \mathbf{Z^{(s)}}\right) \propto \begin{cases} w_{ij} & i \leq n_A; \\ n_A \frac{n_B - D + \beta_\lambda}{D + \alpha_\lambda} & i = n_A + 1 \end{cases}$$

4.5 Bayes Estimate

We calculate a Bayes estimate $\hat{\mathbf{Z}}$ for the linkage parameter $\mathbf{Z}$ by assigning different positive losses to different types of errors, and minimizing posterior expected loss. We adopt the loss function proposed in Sadinle 2017, in which $\hat{Z}_j \in \{1, \ldots, n_A + 1, R\}$, with R representing the option to leave the matching undetermined by the model. Specifically, we minimize the quantity $L(\hat{\mathbf{Z}}, \mathbf{Z}) = \sum_{j=1}^{n_B} L(\hat{Z}_j, Z_j)$ where

$$L(\hat{Z}_j, Z_j) = \begin{cases} 0 & \text{if } Z_j = \hat{Z}_j; \\ \theta_R, & \text{if } \hat{Z}_j = R; \\ \theta_{10}, & \text{if } Z_j \leq n_A, \hat{Z}_j = n_A + 1; \\ \theta_{01}, & \text{if } Z_j = n_A + 1, \hat{Z}_j \leq n_A; \\ \theta_{11}, & \text{if } Z_j, \hat{Z}_j \leq n_A, Z_j \neq \hat{Z}_j; \end{cases}$$

Here, θ_R is the loss from not making a decision on the linkage status, θ_{10} is the loss from a false non-match, θ_{01} is the loss from a false match, and θ_{11} is the loss from the special case of a false match in which the record has a true match other than the one estimated by the model. This loss function leads to closed form decision rules for minimizing posterior expected loss. In this paper, we adopt losses $\theta_R = \infty, \theta_{10} = 1, \theta_{01} = 1, \theta_{11} = 2$, inducing the intuitive decision rule

$$\hat{Z}_j = \begin{cases} i, & \text{if } P(Z_j = i | \Gamma) > \frac{1}{2}; \\ 0, & \text{otherwise}; \end{cases}$$

For a more in-depth explanation of this function and the induced Bayes estimate, see (Sadinle, 2017).

Since our Gibbs procedure does not strictly enforce one-to-one matching, it is possible for the final Bayes estimate to link multiple records in B to one record in A. The modeler can either report both such matches (with their respective posterior match probabilities), or resolve these conflicts by accepting only the match with highest posterior probability. A similar approach can be see in the most probable maximal matching sets used by (Steorts, 2013) to match records to latent entities. Such a resolution procedure indeed is equivalent to minimizing posterior risk under the restriction of one-to-one matching, and as thus theoretically justified within the Bayesian framework.

5 EFFICIENT AND SCALABLE IMPLEMENTATION

In this section, we propose efficient and scalable proposals using hashing based methods that are similar in spirit to (**?**), a fast and scalable implementation of the Fellegi-Sunter model. Specifically, one way we can improve our computational efficiency is by recognizing that record pairs contribute to posterior calculations only through the agreement pattern of the γ_{ij} vector. To make this more precise, let $\mathcal{H}$ be the set of unique agreement patterns in the data, let $P = |\mathcal{H}|$ denote the total number of unique agreement patterns. Observe that P is bounded above by $\prod_{f=1}^{F} L_f$, and that this bound does not depend on n_1 or n_2. Prior to processing the data, we identify all P patterns in $\mathcal{H}$ and enumerate them as follows: $h_1, \ldots, h_P$, which are called *hashed values.* Next, we map record pairs to their corresponding hashed value. More specifically, when the record pair (i, j) exhibits the p^{th} agreement pattern, we say $(i, j) \in h_p$. Whenever possible, we conduct calculations over these P agreement patterns, instead of the typical $n_1 \times n_2$ record pairs.

5.1 Data Representation, Hashing, and Storage

In this section, we describe the hashing function we propose.

First, we hash record pairs of the same agreement pattern to unique integer values. (**?**) accomplished this efficiently through the hashing function

$$\tilde{\gamma}_{ij} = \sum_{f=1}^{F} I(\gamma_{ij}^{f} > 0) 2^{\gamma_{ij}^{f} + I(k>1) \sum_{e=1}^{k-1} (L_e - 1)}$$

This function maps each agreement pattern to a unique integer, allowing us to store a scalar quantity instead of an entire vector for each record pair. For computational ease, we then map these integers to sequential integers from $1, \ldots, P$.

The classic Fellegi Sunter method represents the γ_{ij} comparison as a vector of length F, with each component γ_{ij}^f taking on values in $\{1, \ldots, L_f\}$. To ease our computational burden, we instead use a *one hot encoding* of the comparison vector. For example, if $L_1 = L_2 = 2$ and $L_3 = 3$, then $\gamma_{ij} = (2, 1, 3)$ under the classical framework becomes $\gamma_{ij} = (0, 1, 1, 0, 0, 0, 1)$ under our framework. This is a bijective transformation that does not change the meaning of the data, but this representation eases calculations and posterior updates. This is also the form the data takes in the `BRL` package in `R`.

In the classic Fellegi Sunter framework, $\boldsymbol{\Gamma}$ is a $n_1 n_2 \times F$ matrix, with each row providing the comparison vector for a different (i, j) pair. In contrast, we do not store these comparison vectors themselves, but instead only the hashed value h_p corresponding to the agreement pattern of the (i, j) pair. We store this information in a nested list $\tilde{\boldsymbol{\Gamma}}$ where the p^{th} component of the j^{th} list contains a vector of records in $\boldsymbol{X}_1$ that share agreement pattern p with record $j \in \boldsymbol{X}_2$. For each p, we also calculate $H_p = \sum_{i=1}^{n_1} \sum_{j=1}^{n_2} \mathbf{1}_{(i,j) \in h_p}$, the total instances of agreement pattern p throughout the data, and also for each j, we calculate $H_{p_j} = \sum_{i=1}^{n_1} \mathbf{1}_{(i,j) \in h_p}$ the instances of agreement pattern p among the comparison vectors between record $j \in \boldsymbol{X}_2$ and each of the n_1 records in $\boldsymbol{X}_1$.

The hashing procedure described above considerably reduces the memory needed to store the comparison information. That is, instead of storing $n_1 \times n_1$ comparison vectors, which are relatively long under either the Fellegi Sunter or our modified framework, we only store the P unique vectors, and then $n_1 \times n_2$ scalar quantities relating record pairs to those vectors. However, even storing these $n_1 \times n_2$ scalar labels can become burdensome with large data. Worse, the overwhelming majority of these labels relate to record pairs that are clear non-matches.

To address this issue, we propose a new method called *storage efficient indexing* (SEI). In standard indexing, one decides a certain criteria that they expect all true matching pairs to satisfy. In addition, one decides a priori to label any record pairs that do not meet

that criteria as non-matches. For example, one might only consider pairs with a certain similarity score on a field deemed to be important (like first name), or only pairs with exact matching on a specified number of fields. While generally chosen to be be quite loose, establishing these criteria requires knowledge of the problem and invites room for human error. To improve upon this, we propose to reduce the comparison space and reduce the storage requirements while avoiding the drawbacks of standard indexing.

Observe that all records of the same agreement pattern have the same probability when sampling Z_j. Therefore we know that records belonging to an h_p such that H_{p_j} is large are very unlikely to be sampled consistently enough to be deemed a match through the Bayes estimate, even without considering the form of the agreement pattern itself.

In SEI, rather than store all of these unlikely record labels, we choose to store only a small number R of them. Posterior calculations still attribute the appropriate weight to all records through the summary statistics H_p, and H_{p_j}. Rather than storing $n_1 \times n_2$ record labels, SEI allows us to store at most $n_2 \times P \times R$ labels, regardless of how large n_1 might be.

Lastly, for large data, we can partition the two datasets $\boldsymbol{X}_1$ and $\boldsymbol{X}_2$ into smaller blocks $\{\boldsymbol{X}_{1m}\}$ and $\{\boldsymbol{X}_{2m}\}$ for more manageable computations. On a single machine, we can read-in data sequentially, conduct hashing, compress information through SEI, and delete the original data from memory before continuing with the next chunk of data. With multiple cores or multiple machines, this can be done in parallel. Thus, the combination of hashing, SEI, and partitioning allows us to conduct linkage tasks over much larger datasets.

Brian: how large can we go and how fast can we go?

6 EFFICIENT POSTERIOR INFERENCE

Updating $\mathbf{m}$ and $\mathbf{u}$: After receiving matching statuses from $\mathbf{Z}$, the Sadinle method calculates $\alpha_{fl}(\mathbf{Z})$ and $\beta_{fl}(\mathbf{Z})$ for each field and level. This constitutes $2 \times \sum L_f$ many summations over $n_A \times n_B$ quantities, and becomes computationally burdensome with large data. In contrast, we recognize that each unique agreement pattern contributes to the posterior $\alpha(\mathbf{Z})$ and $\beta(\mathbf{Z})$ vectors in the same way. In fact, if we denote $H_p^m = \sum_{j=1}^{n_B} \mathbf{1}_{(Z_j,j)\in h_p}$ to be the

number of matching record pairs with agreement pattern p, then the contribution of pairs of pattern p to the $\alpha(\mathbf{Z})$ vector is simply $H_p^m \times h_p$. Thus our posterior update for the α vector is simply $\alpha(\mathbf{Z}) = \alpha_0 + \sum_{p=1}^{P} H_p^m \times h_p$. Then, we can easily calculate H_p^u, the number of nonmatching record pairs of agreement pattern p, by subtracting the number of matching pairs from the total present in the data; that is $H_p^u = H_p - H_p^m$. From this, we can update our β parameter through $\beta(\mathbf{Z}) = \beta_0 + \sum_{p=1}^{P} H_p^u \times h_p$. Note that these constitute P many summations over n_B quantities, and thus avoid the $n_A \times n_B$ summation from the original method.

Updating $\mathbf{Z}$: Although sampling Z_j from a the full conditional provided earlier is conceptually straightforward, it becomes computational burdensome when n_A is larger. The reader can confirm that sampling a value from a large set of unequal probabilities becomes difficult in most programming languages. To speed up computation, we break this sampling step into two simpler steps. First, we calculate the Fellegi Sunter weight w_p associated with each unique pattern and sample the agreement pattern between j and its potential match. Second, we sample the record label uniformly among records associated with that agreement pattern for that particular $j \in B$. More concretely, define $h(Z_j)$ to be the agreement pattern between j and its potential match, and say $h(Z_j) = h_{P+1}$ when $Z_j = n_A + 1$. Then,

$$P\left(h\left(Z_j^{(s+1)}\right) = p | \mathbf{m}, \mathbf{u}, \mathbf{Z}^{(\mathbf{s})}\right) \propto \begin{cases} w_p \times H_{p_j} & p \leq P; \\ n_A \frac{n_B - D + \beta_\lambda}{D + \alpha_\lambda} & p = P + 1 \end{cases}$$

$$P\left(Z_j^{(s+1)} = i \middle| \; h\left(Z_j^{(s+1)}\right) = p\right) = \begin{cases} \frac{1}{H_{p_j}} & (i,j) \in h_p \\ 0 & \text{otherwise} \end{cases}$$

Lastly, we recognize that all posterior updates are governed by the agreement patterns of the record pairs rather than the record labels themselves. Thus we complete the entire Gibbs procedure first at the level of the P agreement patterns with the first equation above. After, we can back-fill the records corresponding to the agreement patterns through the second equation. Sampling uniformly is computationally simple even for large sets of candidate records, but this step can also be parallelized when working with large data.

To aid the reader, we provide summary of the `fabl` method through pseudocode:

Algorithm 1 Summary of fabl algorithm

procedure HASHING AND PREPROCESSING
    Partition files A and B into chunks $\{A_I\}, \{B_J\}$
    for each I, J **do**
        Create comparison vectors between A_I and B_J
        Hash results and calculate summary statistics
        Use SEI to reduce memory usage
    end for
    Synthesize results across pairings
end procedure
procedure GIBBS SAMPLING
    Initialize m, u, and Z parameters
    for $t \in \{1, \ldots, T\}$ **do**
        Sample $m^{t+1}|Z^t, \Gamma$ and $u^{t+1}|Z^t, \Gamma$
        Sample $H\left(Z^{t+1}\right)|m^{t+1}, u^{t+1}, \Gamma$ ▷ Sample agreement pattern, not record
    end for
    Sample $Z|H(Z), \Gamma$ ▷ Fills in record label based on agreement pattern
end procedure

RCS: I stopped here.

7 SIMULATION STUDIES

In this section, we consider a simulation study, nearly identical to that in **?**. Brian: can you explain how it's the same versus different than what it is given in Sadinle as this would be the most helpful. Do you utilize Mauricio's exact files and approach and then do something else? If so, I would state this for clarity.

Brian: If you look in Sadinle (2017), did you use Table 2 or something else? Can you construct something similar to Figure 3 and Figure 4 as it's likely a reviewer will look at this paper and ask why you didn't do this exact comparison.

We first compare `fabl` against `BRL` on several simulated datasets with varying amounts of error and overlap. We use first name, last name, age, and occupation for this linkage, and create comparison vectors according to the default settings of the `compareRecords` function from the `BRL` package. Each simulation identifies duplicated individuals between two datasets, each with 500 records. We conduct linkage when matching records exhibit 1, 2, and 3 errors across the four fields, and when there are 50, 250, and 450 individuals in common across datasets. We use flat priors for all m and u parameters, run the Gibbs Sampler for 1000 iterations, and discard the first 100 as burn-in. This is a near exact replication of the simulation study provided by (**?**).

7.1 Precision, Recall, and F-measure

In this section, we compare `fabl` to `BRL` in terms of precision, recall, and f-measure, which are commonly defined record linkage evaluation metrics that are defined in (Christen, 2012). In cases when there are only one or two errors in matching records, and in cases with low to moderate duplication across records, we see that `fabl` provides approximately equivalent accuracy as `BRL`. We find that our method only has weakened performance in the most extreme scenario of very high errors and very high overlap across files. In these situations, `BRL` is removing large numbers of records from consideration throughout the Gibbs Sampler, making its implementation most different from `fabl`. However, such extreme linkage tasks,

which such high amounts of errors and overlap, are extremely rare in practice.

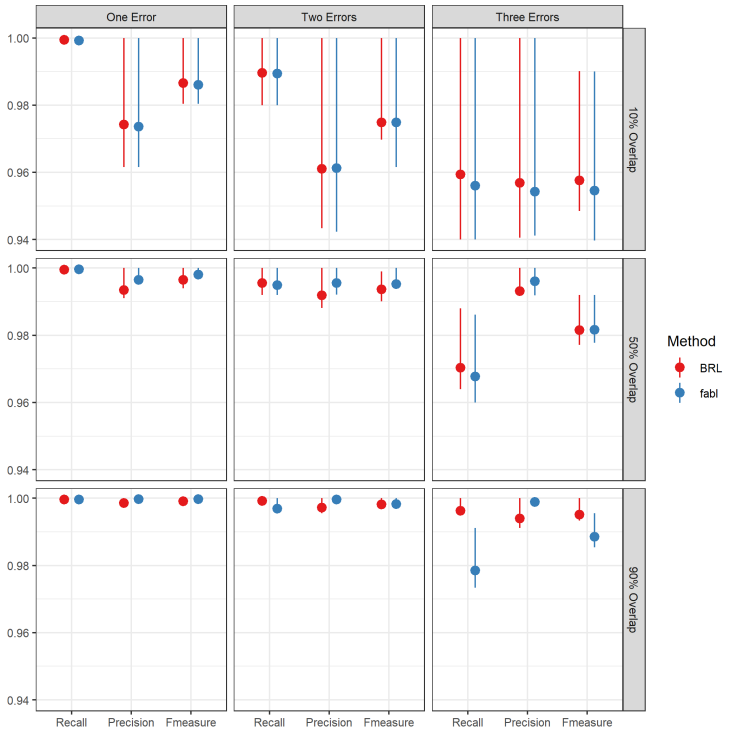

Figure 1: Posterior means and credible intervals for accuracy metrics under the replication of simulation study from Sadinle 2017. For each level of overlap and each level of error, we have 100 paired sets of 500 records.

7.2 Speed

Brian: I have some general questions here. Are you using blocking for BRL to scale? Could you use the hashing proposal for BRL. It's important that the scale comparison be as fair as possible or there will be some pushback here. My suggestion around all this is to look at the computational complexity of both methods as it's hard to attack this.

To demonstrate speed, we generate comparison vectors from pre-specified distributions so that we can easily increase the size of the linkage problem. Distributions are meant to emulate the behavior of similarity scores across first name, last name, and day, month. For example, $u^{\text{month, 1}} = P(\text{Records have same birth-month} \mid \text{Nonmatch}) = \frac{1}{12}$. For simplicity,

	m	u
fname	$\left(\frac{19}{20}, \frac{1}{20}\right)$	$\left(\frac{1}{100}, \frac{99}{100}\right)$
lname	$\left(\frac{19}{20}, \frac{1}{20}\right)$	$\left(\frac{1}{100}, \frac{99}{100}\right)$
day	$\left(\frac{19}{20}, \frac{1}{20}\right)$	$\left(\frac{1}{30}, \frac{29}{30}\right)$
month	$\left(\frac{19}{20}, \frac{1}{20}\right)$	$\left(\frac{1}{12}, \frac{11}{12}\right)$
year	$\left(\frac{19}{20}, \frac{1}{20}\right)$	$\left(\frac{1}{12}, \frac{11}{12}\right)$

Table 1: Distributions used for m and u probabilties in simulation studies

we consider only exact matching, so a vector (1, 0) corresponds to agreement and (0, 1) to disagreement. We simulate these data for different values of n_A and n_B, and compare the run-time of `fabl` against `BRL`. Note that the number of unique patterns P is bounded above by $2^5 = 32$, a bound which is consistently attained in the larger simulations.

We see that at low data size, `BRL` outperforms, but that `fabl` is significantly faster at handling larger data. In particular, run-time for `BRL` seems to grow quadratically (or linearly with the size of both A and B) while run-time for `fabl` seems to grow linearly (in the size of B).

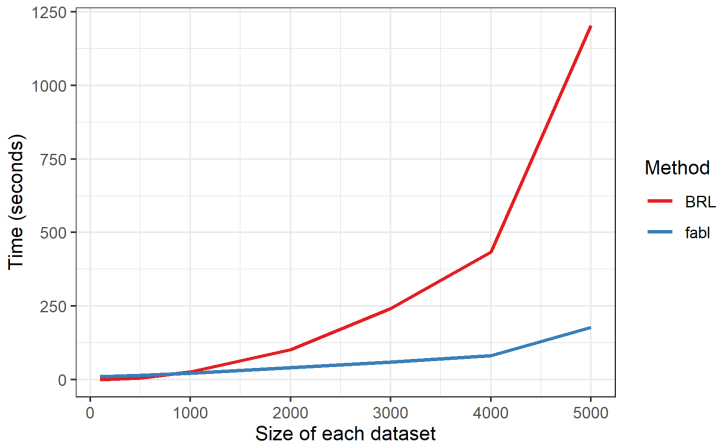

Figure 2: Run-time for BRL and fabl to run 1000 Gibbs iterations, including hashing step for fabl, for increasing values of both n_A and n_B. We see near quadratic growth in runtime for BRL, and near linear growth for fabl.

The above discussion suggests that for fixed n_B, computation time should remain mostly

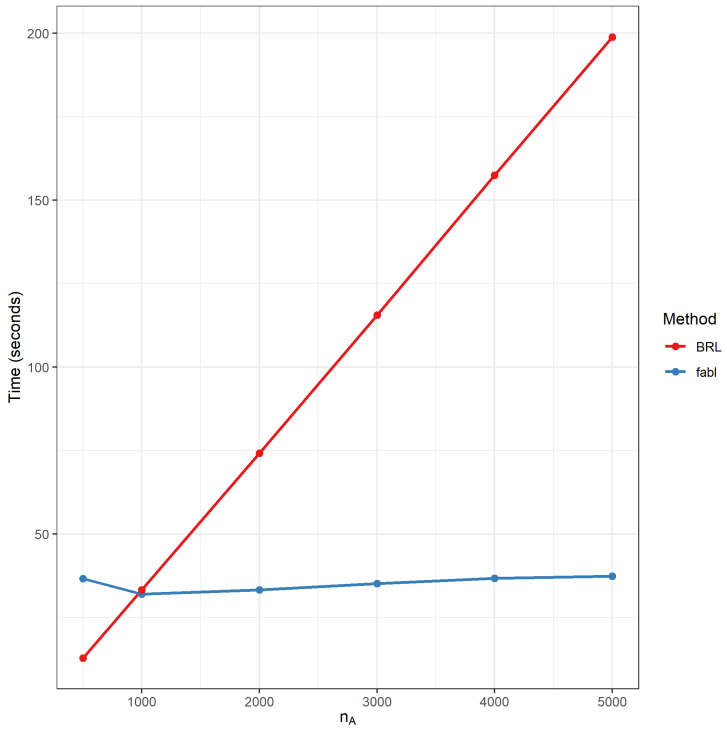

Figure 3: Run-time for BRL and fabl to run 1000 Gibbs iterations, including hashing step for fabl, with increasing n_A, and n_B fixed at 500. We see linear growth in runtime for BRL, and near constant runtime for fabl.

constant with growing n_A. Simulation study suggests that this is true. In the plot below, fixing $n_B = 500$, we see linear growth for the run-time under `BRL` as n_A increases, with much more static run-time under `fabl`. The slight increases in run-time that we do see are due primarily to the hashing step, which again can be run in parallel for large data.

We note here that `BRL` is coded in C, which makes for unfair comparison against `fabl`, currently only built in R. Additionally, although `fabl` is amenable to parallelization, this simulation was run on a single core. Running `fabl` in C++ with paralellization for the hashing step and sampling the matching status of the record pairs should lead to even more computational gains.

7.3 Scaling to a Larger Simulation Study

In this section, we investigate scaling to larger data sets using our proposed methodology.

Specifically, we demonstrate how partitioning and hashing the data through `fabl` method allows us undertake significantly larger linkage tasks. We concatenate two sets of 40 simulated datasets from the original simulation studies of to create two larger files, each of 20,000 records (40,000 records total). Working on one machine, we sequentially compare records across chunks, hash results, and then synthesize summary statistics for the entire simulation. Under standard Fellegi Sunter procedures, this would require 400,000,000 comparison vectors, each consisting for four integers, resulting a final comparison matrix about 6.4 GB in size.

For simplicity, we partition one dataset into 20 smaller chunks, and leave the second dataset fully intact. We compare records, hash results, and then synthesize summary statistics for all 20 chunk comparisons. The resulting data object is now only 90 MB, about 1% the size of the object required under the standard method. Executed sequentially, these comparisons and the Gibbs sampler take about one hour to run; using distributed computing. However, this could be sped up.

Turning to evaluation metrics, this simulation achieved 96.5% recall and 97.7% precision, with an overall F-measure of 97.1% F-measure. The reader will note that this slightly worse performance than witnessed in the smaller simulation studies; this is expected because it is naturally more difficult to link more files with the same amount of information. With more linkage fields `fabl` maintains the high accuracy seen above. Brian: you should look to see if you fix these in the post-processing step as this would be interesting/important.

How long does it take on the same machine to run BRL. Can you run this? This can close this section out!

8 CASE STUDIES

We now demonstrate the power of our method through a case study of documented identifiable deaths (DID) from the El Salvadoran Civil War. Though the data files used here are small, this study shows how the computational complexity of `fabl` depends on the number of unique agreement patterns, and how significant computational gains can be achieved by simplifying the construction of the comparison vectors. Secondly, the case study

reveals the impact of independently sampling Z_j rather than strictly enforcing one-to-one matching as done by `BRL`.

8.1 El Salvadoran Civil War

The country of El Salvador was immersed in civil war from 1980 to 1991, and throughout the time, several organizations attempted to document casualties of the conflict. When estimating the total number of casualties, one cannot simply sum the numbers recorded by each organization, as it is likely that the same individuals are recorded in multiple casualty lists. To obtain a more accurate estimate of the casualties then, we follow Sadinle's 2017 paper and link records of El Salvadoran civilian casualties from two sources: El Rescate - Tutela Regal (ERTL) and the Salvadoran Human Rights Commission (CDHES, by its acronym in Spanish). The ERTL dataset consists of digitized reports that had been published throughout the conflict. The CDHES dataset consists of casualties that had been reported directly to the organization, and later digitized.

There are several challenges with working with such data. Firstly, both datasets have been automatically digitized, which inherently leads to some degree of typographical error. Secondly, the CDHES records are all second hand accounts reported by individuals, which can result in additional errors. Lastly, the only fields recorded are given name, last name, date of death, and place of death; it is relatively common for a parent and child to share the same given name, resulting in indistinguishable records for two different individuals. This last point nearly breaks the earlier mentioned assumption that there are no duplicates within files, and reveals a key difference between `BRL` and our proposed method.

We only utilized records with nonmissing entries for given and last name, results in $n_A = 4420$ files in CHDES and $n_B = 1323$ files in ERTL. The names were standardized to account for common misspellings in the Spanish language, and then compared using a modified Levenstein distance to account for the fact that second names are often omitted. Place of birth is recorded by municipality and department within that municipality; however, since department was missing in 95% of records in CHDES and 80% of records in ERTL, we excluded it from our linkage process. Thus we conduct linkage using given name, last

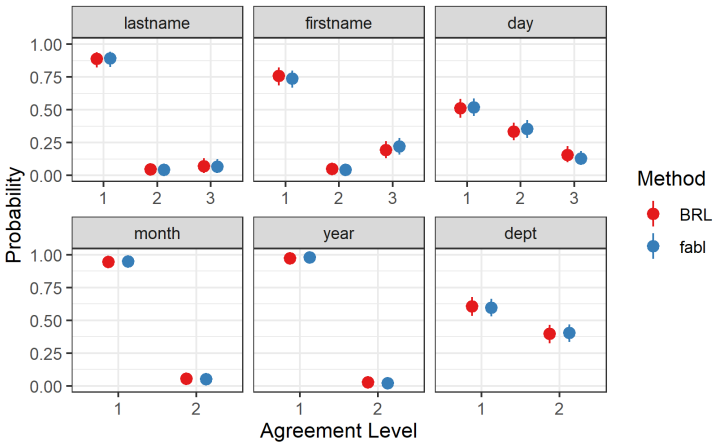

Figure 4: Posterior estimates of m parameters with 95% credible intervals

name, municipality, and day, month, and year of death. We again use flat priors for the **m** and **u** parameters.

To mirror the original implementation, we constructed the comparison vectors using 4 levels of agreement for each field, according to the thresholds provided in Figure XX. This took 422 seconds for our proposed method and 240 for `BRL`. However, we observed that posterior distributions of several levels of the **m** and **u** parameters were nearly identical, and that of the $4^5 \times 2 = 2048$ possible agreement patterns, only 1173 are realized in the data. This leads us to believe that such high number of agreement levels creates unnecessary distinctions in the data and makes the comparison vectors less interpretable. Therefore we re-ran our analysis with fewer agreement levels for each field (see Figure XX), and obtained analogous results. With 216 possible agreement patterns, 159 were realized in the data, and our proposed method became much faster, finishing in 124 seconds. Meanwhile `BRL` took 239 seconds, relatively unchanged from the first implementation. This demonstrates the way that the computational complexity of our method depends on the number of unique agreement patterns, and how significant computational gains can be made by simplifying the construction of the comparison vectors. We also note that estimates of the **m** and **u** parameters under each method are very similar, as shown in Figure 4.

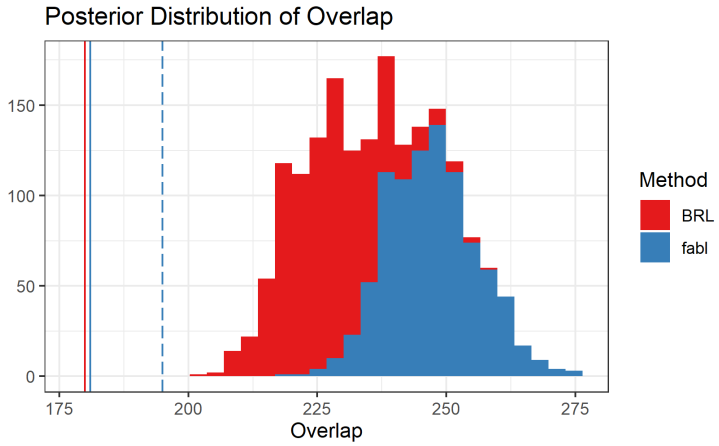

Figure 5: Posterior distribution of overlap across the two files. The solid lines show the Bayes estimate for the amount of overlap, and the dashed line is the Bayes estimate under fabl before resolving violations of one-to-one matching

Violations of One-to-One Matching Figure 5 shows the posterior distribution for D, the number of duplicates found across file for both `fabl` and `BRL`. We see that `fabl` consistently overmatches within each Gibbs iteration when compared to `BRL`, which is to be expected because `BRL` explicitly prevents matches that violate one-to-one matching throughout the entire sampler. Most of such matchings are due to the randomness in the sampling procedure, and they occur sporadically throughout the sampler in such a way that does not measurably influence the eventual Bayes estimate.

Overall, both methods presented similar results. `fabl` yielded an initial Bayes estimate of 195 matches found across files, and after resolving matches that violated one-to-one requirements, yielded 180 matches across files. This is acceptably close to the 181 matches found by `BRL`. The main reason for this discrepancy is the difference in how each model handles situations in which one file in B has multiple plausible matches in A.

The records shown in Figure 6 provide one such instance. Note these records present a near violation of our assumption that there are no duplications within files; we continue to assume that records 825 and 826 in ERTL correspond to different individuals, but their records are nearly identical. Additionally, using the modified Levenstein distance from Sadinle 2017, the comparison vectors $\gamma_{2776,825}$ and $\gamma_{2776,826}$ are exactly identical.

	lastname	firstname	dataset	day	month	year	dept	muni
825	PINEDA	ROSA	CDHES	6	4	1984	NA	NA
826	PINEDA	ROSA MARIA	CDHES	6	4	1984	NA	NA
2776	PINEDA	ROSA MARIA	ER-TL	4	4	1984	CUSCATLAN	NA

Figure 6: Example of linkage situation with multiple plausible matches

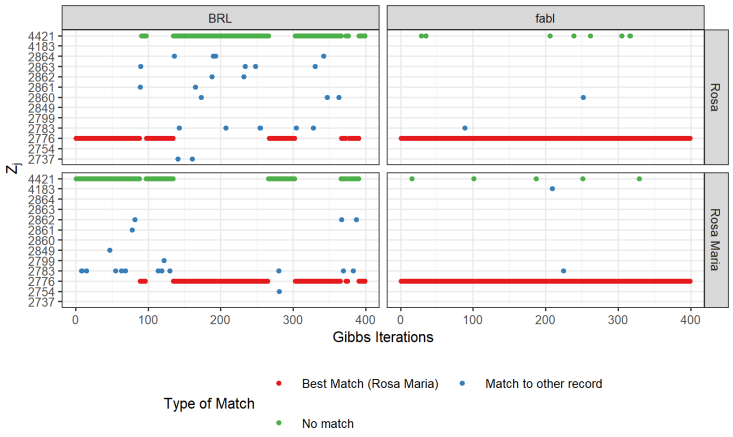

Figure 7: Gibbs sampling in situation with multiple plausible matches.

Figure 7 shows the values that Z_{825} and Z_{826} take on throughout the Gibbs sampler, and demonstrates how each method handles this situation. Under `fabl`, both records in B match to the same record in A throughout the Gibbs process, creating consistent violations of one-to-one matching. Under `BRL`, the Gibbs process creates one matching configuration stays there for a while. However, if one pair "unmatches," then the other record has a chance to latch on. Then, the Gibbs process is stuck with that matching status for a while, resulting in a Gibbs process with poor mixing. Additionally, `fabl` allows the modeler to inspect records with multiple plausible matches, and if they desire, to then choose the record pairing with the highest posterior probability. `BRL` in contrast, in strictly enforcing one-to-one matching throughout the sampler, can lead to situations where none of the plausible matches reach the threshold to be identified through the Bayes estimate.

9 DISCUSSION

We have presented a method for Bayesian record linkage that is feasible for large datasets. In particular, our hashing procedure and model assumptions allow for a linkage procedure whose computational complexity does not scale with the size of the larger dataset, making this method particularly powerful in linking records when one datafile is substantially smaller than the other.

In our case study, we included an exploration of how to conduct record linkage when modeling assumptions are not met in practice. We explored "one-to-many" scenarios in which one record in A has multiple plausible matches in B, and showed how both `fabl` and `BRL` demonstrated undesirable qualities. Other issues arise under "many-to-one" scenarios, where one record in B has multiple plausible matches in A, and "many-to-many" scenarios in which there is duplication both across and within datasets. Tuning `fabl` for use in these scenarios is one potential avenue for future work.

10 APPENDIX

REFERENCES

Belin, T. R. and Rubin, D. B. (1995), "A Method for Calibrating False-Match Rates in Record Linkage," *Journal of the American Statistical Association*, 90, 694–707.

Bilenko, M. and Mooney, R. (2006), "Riddle: Repository of Information on Duplicate Detection, Record Linkage, and Identity Uncertainty," Online; retrieved July 29, 2020.

Bilenko, M., Kamath, B., and Mooney, R. J. (2006), "Adaptive blocking: Learning to scale up record linkage," *Proceedings - IEEE International Conference on Data Mining, ICDM*, pp. 87–96.

Christen, P. (2008), "Automatic Record Linkage Using Seeded Nearest Neighbour and Support Vector Machine Classification," in *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, p. 151–159, New York, NY, USA, Association for Computing Machinery.

Christen, P. (2012), "A survey of indexing techniques for scalable record linkage and deduplication," *IEEE Transactions on Knowledge and Data Engineering*, 24, 1537–1555.

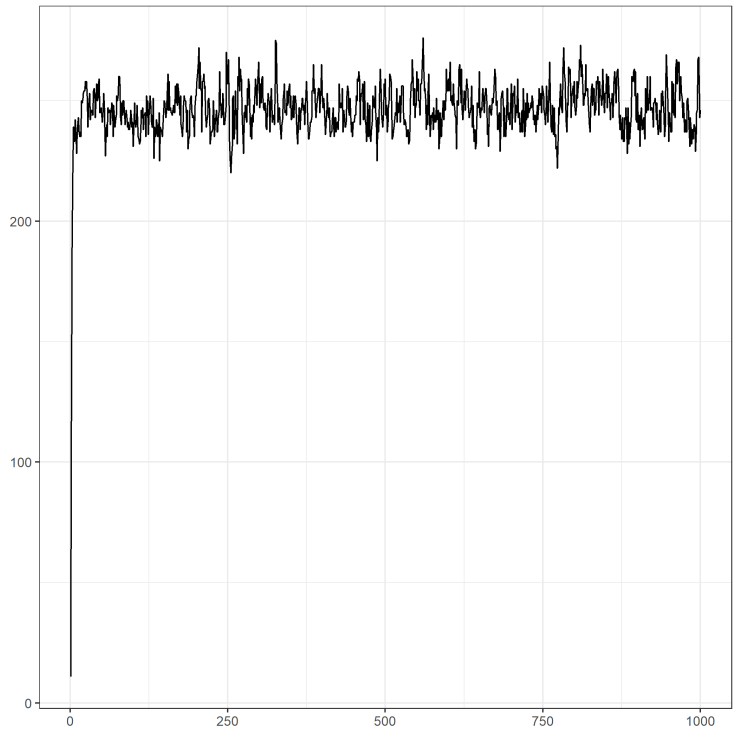

Figure 8: Traceplot for number of matches found across datasets in El Salvador case study

Christophides, V., Efthymiou, V., Palpanas, T., Papadakis, G., and Stefanidis, K. (2021), "An Overview of End-to-End Entity Resolution for Big Data," *ACM Computing Surveys*, 53.

Dalzell, N. M. and Reiter, J. P. (2018), "Regression modeling and file matching using possibly erroneous matching variables," *Journal of Computational and Graphical Statistics*, 27, 728–738.

Elmagarmid, A. K., Ipeirotis, P. G., and Verykios, V. S. (2007), "Duplicate Record Detection: A Survey," *IEEE Transactions on Knowledge and Data Engineering*, 19, 1–16.

Enamorado, T., Fifield, B., and Imai, K. (2019a), "Using a Probabilistic Model to Assist Merging of Large-Scale Administrative Records," *American Political Science Review*, 113, 353–371.

Enamorado, T., Fifield, B., and Imai, K. (2019b), "Using a Probabilistic Model to Assist Merging of Large-Scale Administrative Records," *American Political Science Review*, 113, 353–371.

Fair, M. (2004), "Generalized record linkage system–Statistics Canada's record linkage software," *Austrian Journal of Statistics*, 33, 37–53.

Fellegi, I. P. and Sunter, A. B. (1969), "A Theory for Record Linkage," *Journal of the American Statistical Association*, 64, 1183–1210.

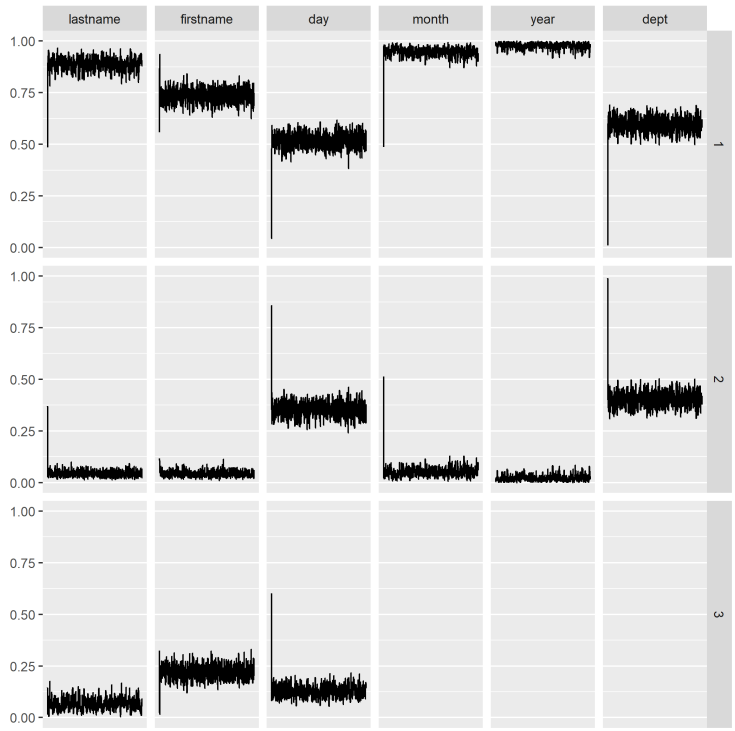

Figure 9: Traceplot for m parameter in El Salvador case study

Fortini, M., Liseo, B., Nuccitelli, A., and Scanu, M. (2001), "On Bayesian Record Linkage," *Research in Official Statistics*, 4, 185–198.

Gill, L. and Goldacre, M. (2003), "English national record linkage of hospital episode statistics and death registration records," *Report to the department of health*.

Gutman, R., Afendulis, C. C., and Zaslavsky, A. M. (2013), "A Bayesian Procedure for File Linking to Analyze End-of-Life Medical Costs," *Journal of the American Statistical Association*, 108, 34–47.

Herzog, T., Scheuren, F., and Winkler, W. (2007a), *Data Quality and Record Linkage Techniques*, Springer, New York.

Herzog, T. N., Scheuren, F. J., and Winkler, W. E. (2007b), *Data quality and record linkage techniques*, Springer Science & Business Media.

Jaro, M. A. (1989), "Advances in Record-Linkage Methodology as Applied to Matching the 1985 Census of Tampa, Florida," *Journal of the American Statistical Association*, 84, 414–420.

Jaro, M. A. (1995), "Probabilistic linkage of large public health data files," *Statistics in Medicine*, 14, 491–498.

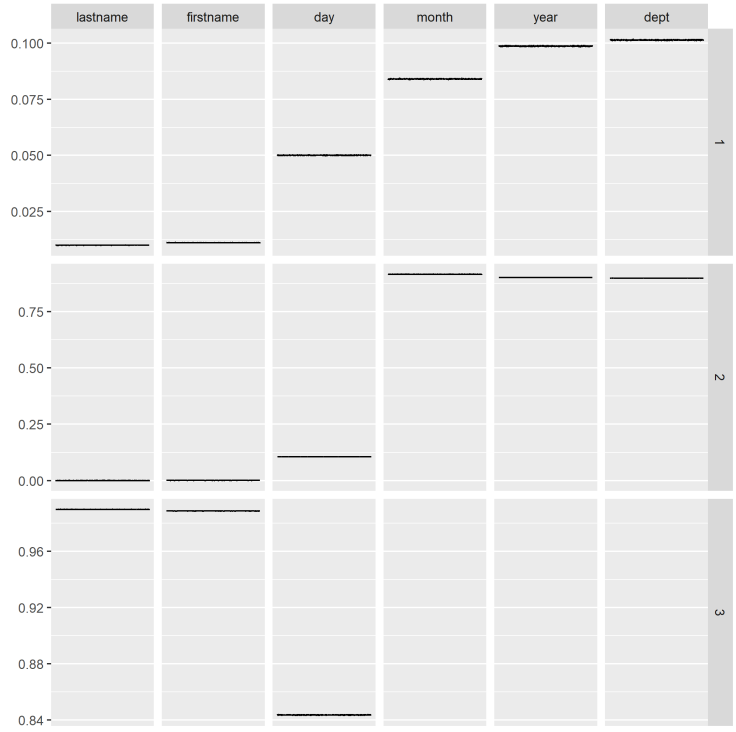

Figure 10: Traceplot for u parameter in El Salvador case study

Jurek-Loughrey, A. and P, D. (2019), *Semi-supervised and Unsupervised Approaches to Record Pairs Classification in Multi-Source Data Linkage*, pp. 55–78, Springer International Publishing, Cham.

Larsen, M. D. (2005), "Advances in Record Linkage Theory: Hierarchical Bayesian Record Linkage Theory," in *Proceedings of the Joint Statistical Meetings, Section on Survey Research Methods*, pp. 3277–3284, The American Statistical Association.

Larsen, M. D. (2012), "An Experiment with Hierarchical Bayesian Record Linkage," arXiv e-prints, arxiv:1212.5203.

Larsen, M. D. and Rubin, D. B. (2001), "Iterative Automated Record Linkage Using Mixture Models," *Journal of the American Statistical Association*, 96, 32–41.

Liseo, B. and Tancredi, A. (2011), "Bayesian estimation of population size via linkage of multivariate Normal data sets," *Journal of Official Statistics*, 27, 491–505.

Liseo, B. and Tancredi, A. (2013), "Some advances on Bayesian record linkage and inference for linked data," .

Marchant, N. G., Steorts, R. C., Kaplan, A., Rubinstein, B. I. P., and Elazar, D. N. (2019), "d-blink: Distributed End-to-End Bayesian Entity Resolution," *arXiv e-prints*, arxiv:1909.06039.

Matsakis, N. E. (2010), "Active Duplicate Detection with Bayesian Nonparametric Models," Ph.D. thesis, Massachusetts Institute of Technology.

Newcombe, H. B., Kennedy, J. M., Axford, S. J., and James, A. P. (1959), "Automatic Linkage of Vital Records," *Science*, 130, 954–959.

Sadinle, M. (2017), "Bayesian Estimation of Bipartite Matchings for Record Linkage," *Journal of the American Statistical Association*, 112, 600–612.

Shan, M., Thomas, K., and Gutman, R. (2020), "A Bayesian Multi-Layered Record Linkage Procedure to Analyze Functional Status of Medicare Patients with Traumatic Brain Injury," *arXiv e-prints*, arxiv:2005.08549.

Steorts, R. (2013), "Divergence losses under benchmarking for small area estimation," *Working Paper*.

Steorts, R. C. (2015), "Entity Resolution with Empirically Motivated Priors," *Bayesian Analysis*, 10, 849–875.

Steorts, R. C., Hall, R., and Fienberg, S. E. (2016), "A Bayesian Approach to Graphical Record Linkage and Deduplication," *Journal of the American Statistical Association*, 111, 1660–1672.

Steorts, R. C., Tancredi, A., and Liseo, B. (2018), "Generalized Bayesian Record Linkage and Regression with Exact Error Propagation," in *International Conference on Privacy in Statistical Databases*, pp. 297–313, Springer.

Tancredi, A. and Liseo, B. (2011), "A Hierarchical Bayesian Approach to Record Linkage and Size Population Problems," *Annals of Applied Statistics*, 5, 1553–1585.

Tancredi, A. and Liseo, B. (2015), "Regression analysis with linked data: problems and possible solutions," *Statistica*, 75, 19–35.

Tancredi, A., Steorts, R., and Liseo, B. (2020), "A Unified Framework for De-Duplication and Population Size Estimation (with Discussion)," *Bayesian Analysis*, 15, 633–682.

Ventura, S. L., Nugent, R., and Fuchs, E. R. (2015), "Seeing the Non-Stars:(Some) Sources of Bias in Past Disambiguation Approaches and a New Public Tool Leveraging Labeled Records," *Research Policy*, 44, 1672–1701.

Wagner, D., Lane, M., et al. (2014), "The person identification validation system (PVS): applying the Center for Administrative Records Research and Applications'(CARRA) record linkage software," Tech. rep., Center for Economic Studies, US Census Bureau.

Winkler, W. E. (1988), "Using the EM Algorithm for Weight Computation in the Fellegi-Sunter Model of Record Linkage," in *Proceedings of the Section on Survey Research Methods*, pp. 667–671, American Statistical Association.

Winkler, W. E. (2014), "Matching and Record Linkage," *Wiley Interdisciplinary Reviews: Computational Statistics*, 6, 313–325.

Winkler, W. E. and Thibaudeau, Y. (1991), *An Application of the Fellegi-Sunter Model of Record Linkage to the 1990 US Decennial Census*.

Zanella, G., Betancourt, B., Wallach, H., Miller, J., Zaidi, A., and Steorts, R. C. (2016), "Flexible Models for Microclustering with Application to Entity Resolution," in *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NIPS'16, pp. 1425–1433, NY, USA, Curran Associates Inc.