Bayesian Matching with Relaxed Duplication Assumptions through Dirchlet Record Linkage

Brian Kundinger

August 29, 2024

Abstract

Probabilistic record linkage is the use of statistical methods to identify unique entities within and across databases. Bayesian methods can adopt different prior distributions for various assumptions on the nature of record duplications in the linkage task, but become computationally expensive as the size of the linkage task grows. In this paper, we propose Dirichlet Record Linkage (`DRL`), a method for linking a duplicate-free reference file to a target file that may have internal duplications. We use a Dirichlet process prior for the parameter governing the number of matches a record in the reference file has in the target file. We decompose this parameter into a sequence of conditional probabilities, and use these parameters in a computationally efficient Gibbs sampler to conduct the linkage task. We demonstrate the speed and accuracy of our approach relative to other recent Bayesian methods through simulations and case studies. In particular, we show that `DRL` exhibits strong performance using default, non-informative hyperparameters.

1 INTRODUCTION

Probabilistic record linkage is the use of statistical methods to identify unique entities within and across databases, generally without the use of unique identifiers. This is an increasingly important task in "data cleaning," either for its own sake, or as a preliminary step before conducting subsequent analysis. These techniques are used in government, non-profit organizations, healthcare, industry, and their use continues to grow (e.g., Herzog et al. 2007, Christen 2012, Papadakis et al. 2021, Binette & Steorts 2022). In this paper, we propose a method for conducting record linkage between a duplicate free reference file and some other file that may have internal duplications.

Many probabilistic record linkage methods are based on the foundational work of Fellegi & Sunter (1969). In these Fellegi-Sunter (FS) methods, the data analyst first creates comparison vectors for each pair of records in the data files. These vectors indicate how similar the records are on a set of variables measured in both files, known as the linkage variables. Using these comparison vectors, the analyst classifies each pair as a match or non-match. An alternative paradigm is to model the linkage variables directly (e.g., Tancredi & Liseo 2011, Steorts et al. 2016, Marchant et al. 2021, Betancourt et al. 2022). In this article, we build on the contributions to the comparison vector approach.

In the original FS formulation, the matching status of each record pair is considered independent from the matching status of all other record pairs. This can be suboptimal when we know that each file is internally free of duplications or when we desire a set of declared matches that respects transitivity. These concerns can be addressed through a Bayesian framework through different prior distributions on the linkage structure to formalize different assumptions on the type of matchings between files. For example, Sadinle (2017) proposed a prior distribution on the one-to-one matchings between two files, and Aleshin-Guendel & Sadinle (2023) proposed a prior on the partitions of records corresponding to unique entities found in arbitrarily many files. These more tailored record linkage methods have the advantage of producing coherent sets of matches that respect transitivity without the need for post-processing, and are generally more accurate than the standard FS approach. However, they also face limited scalability due to the computation required to model more

complex dependencies between matching statuses.

Given these challenges, much work has been done to increase the speed and scalability of FS methods. One approach is to use probabilistic or deterministic methods to reduce the number of record pairs considered as possible matches. With "blocking," the analyst uses some feature in the data to allocate records into smaller partitions (or blocks), and runs a linkage algorithm independently on each block (Christen 2012). Blocking on an unreliable field, however, can lead to missed matches, making this form of blocking often undesirable (Steorts et al. 2014). "Filtering," is a similar technique which uses some deterministic or probabilistic criteria to set the match probability for certain comparison vectors to zero after they have been created (Murray 2016, McVeigh et al. 2019). Without reducing the number of comparison vectors in the model, Enamorado et al. (2019) used of a lower dimensional representation of the set of comparison vectors that allowed for much faster parameter estimation of the standard FS model. Kundinger, Reiter & Steorts (2024) and Kundinger, Aleshin-Guendel & Steorts (2024) then adopted this approach for a faster implementation of the bipartite model of Sadinle (2017).

In this work, we continue to build on the these advances to provide an efficient and scalable record linkage method without the stringent assumption that each file is free of duplicates. We introduce Dirichlet Record Linkage (`DRL`), an extension of the comparison vector record linkage framework for the scenario of linking one duplicate-free reference file to another file with unknown amounts of internal duplications. To do so, we model the parameter governing the amount of matches between files as a Dirichlet process. Since the resulting set of possible matches for each record is extremely high dimensional, we propose an innovative Gibbs sampler utilizing the stick-breaking representation of the Dirichlet process for tractable posterior inference. We adopt the dimension reduction techniques of Kundinger, Reiter & Steorts (2024) for additional gains in speed and scalability.

In what follows, Section 2 reviews the work of Fellegi & Sunter (1969) and a selection of Bayesian extensions to contextualize our intervention. Section 3 introduces `DRL`, and describes our approach for efficient posterior sampling. Sections 4 and 5 demonstrate the speed and accuracy of our method relative to several other FS techniques in a series

of simulations and a case study of voter registration records in North Carolina. Finally, Section 6 discusses areas for further research.

2 THE FELLEGI-SUNTER FRAMEWORK FOR RECORD LINKAGE

Consider two data files A and B comprising n_A and n_B records, respectively, and including F linkage variables measured in both files. Let $[x] := \{1, \ldots, x\}$ denote the sequence of integers from 1 to x. For $i \in [n_A]$, let record i be given by $A_i = (A_{i1}, \ldots, A_{iF})$, so that $A = (A_i : i \in [n_A])$. Similarly, for $j = 1, \ldots, n_B$, let record j be given by $B_j = (B_{j1}, \ldots, B_{jF})$, so that $B = (B_j : j \in [n_B])$. By convention, we label the two files such that $n_A \geq n_B$.

In record linkage tasks, records that refer to the same entity should be similar, and records that refer to different entities should be dissimilar. To represent this, Fellegi & Sunter (1969) proposed using the comparison vector $\gamma_{ij} = (\gamma_{ij}^1, \ldots, \gamma_{ij}^F)$, where γ_{ij}^f is a comparison for field f between records A_i and B_j. Binary comparisons are commonly used due to their simplicity; partial agreement or more complicated comparisons can be used through similarity metrics or distance functions (Winkler 1990, Bilenko et al. 2003, Elmagarmid et al. 2007). We assume that each field's comparison is discretized, and we let L_f denote the number of levels for field f. We collect all of the comparison vectors as $\boldsymbol{\gamma} = \{\gamma_{ij}\}_{i=1,j=1}^{n_A,n_B}$.

Consider records from A and B as a disjoint set of nodes. We have an edge between two records if they are coreferent (or matching). Our parameter of interest is the collection of edges representing coreferent records. This parameter can be expressed in various ways depending on the model assumptions for a particular approach to record linkage. For example, Fellegi & Sunter (1969) used a coreferent matrix $\Delta \in \{0, 1\}^{n_A \times n_B}$, where

$$\Delta_{ij} = \begin{cases} 1, & \text{if records } A_i \text{ and } B_j \text{ refer to the same entity;} \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

This sparse matrix representation can become cumbersome for large linkage tasks. In the setting where each record in B can match to at most one record in A, Sadinle (2017)

proposed using the more compact vector $\boldsymbol{Z} = (Z_1, \ldots, Z_{n_B})$ for the records in B such that

$$Z_j = \begin{cases} i, & \text{if records } A_i \text{ and } B_j \text{ refer to the same entity;} \\ n_A + j, & \text{if record } B_j \text{ does not have a match in } A. \end{cases}$$

When presenting the theory in this paper, we use either representation for convenience depending on the context. We can go back and forth between the two using $\Delta_{ij} = I(Z_j = i)$, where $I(\cdot) = 1$ when the expression inside the parentheses is true, and $I(\cdot) = 0$ otherwise.

2.1 Models for Comparison Vector Based Record Linkage

Record linkage models within the FS framework consist of a model for the comparison vectors Γ, and a model for the linkage structure that classifies the record pairs as matches and non-matches. In all such methods, the model for the comparison vectors takes the form

$$\Gamma_{ij} \mid \Delta_{ij} = 1 \overset{iid}{\sim} \mathcal{M}(\boldsymbol{m}),$$
$$\Gamma_{ij} \mid \Delta_{ij} = 0 \overset{iid}{\sim} \mathcal{U}(\boldsymbol{u}).$$

Here, $\mathcal{M}$ and $\mathcal{U}$ are the distributions for matching and non-matching record pairs, and $\boldsymbol{m}$ and $\boldsymbol{u}$ are their respective sets of parameters. When using comparison vectors with discrete agreement levels, $\mathcal{M}$ and $\mathcal{U}$ are collections of independent multinomial distributions for each linkage feature. Accordingly, $\boldsymbol{m} = (\boldsymbol{m}_1, \ldots, \boldsymbol{m}_F)$, where $\boldsymbol{m}_f = (m_{f1}, \ldots, m_{fL_f})$ and $m_{fl} = \mathbb{P}(\Gamma_{ij}^f = l | \Delta_{ij} = 1)$ for all fields f and agreement levels l. The $\boldsymbol{u}$ parameters are defined similarly, with $u_{fl} = \mathbb{P}(\Gamma_{ij}^f = l | \Delta_{ij} = 0)$.

The original approach of Fellegi & Sunter (1969) models the matching status of each record pair independently from all other record pairs. Accordingly, they model Δ_{ij} through independent, identically distributed Bernoulli random variables. We generally assume that data in the records is missing completely at random, and is thus ignorable (Little & Rubin 2002). Letting $I_{obs}(\cdot)$ denote an indicator of whether a record field is observed, the likelihood function after marginalizing out the missing data is

$$\mathcal{L}(\Delta, m, u \mid \boldsymbol{\gamma}) = \prod_{i=1}^{n_A} \prod_{j=1}^{n_B} \prod_{f=1}^{F} \prod_{l=1}^{L_f} \left[m_{fl}^{I(\Delta_{ij}=1)} u_{fl}^{I(\Delta_{ij}=0)} \right]^{I(\gamma_{ij}^f = l) I_{obs}(\gamma_{ij}^f)}. \tag{2}$$

The independence assumption at the level of the individual record pairs makes this model straightforward to fit through the EM algorithm (Sariyar & Borg 2010). Recently, much work as been done to use this model for increasingly large linkage tasks, such as through the `fastLink` package in `R` by Enamorado et al. (2019), and the `splink` library in Python by Linacre et al. (2022). However, the full independence assumption of the record pair matching tends to produce a large number of false matches, and post-processing is required to get a more accurate estimate of the linkage structure Jaro (1989).

Results tend to improve when prior knowledge about the linkage structure is incorporated into the model. For example, there are many situations where practitioners know that there are no duplications within each file, and this can be incorporated into the model in a Bayesian framework through a prior distribution on the linkage structure. For this setting, Sadinle (2017) proposed the "beta distribution for bipartite matching," given by

$$\begin{aligned} \mathbb{P}(\boldsymbol{Z} \mid \pi) &= \frac{[n_A - n_{12}(\boldsymbol{Z})]!}{n_A!} \pi^{n_{12}(\boldsymbol{Z})} (1-\pi)^{n_B - n_{AB}(\boldsymbol{Z})}, \\ \pi &\sim \text{Beta}(\alpha_\pi, \beta_\pi), \end{aligned} \tag{3}$$

where $n_{AB} = \sum_{j=1}^{n_B} I(Z_j < n_A + j)$ is the number of links between files A and B, and α_π and β_π are fixed and known. This prior induces a Gibbs sampler that strictly enforces one-to-one matching, and has has been shown to outperform the base FS method when model assumptions are satisfied. This model is implemented in `R` through the `BRL` package, but has computational complexity that grows quadratically with the size of the linkage task, and is therefore limited to small to moderate data files (Sadinle 2020).

More recently, Aleshin-Guendel & Sadinle (2023) extended the FS framework for the setting of arbitrary amount of duplications within and across an arbitrary number of files. To do so, they presented a generative process for multifile partitions, and parameterized each step of this generative process. This leads to a structured prior for multifile partitions consisting of

1. A prior for the number of unique entities exhibited across files. By default, this is taken to be uniform from 1 to the total number of records in the linkage task.

2. Given the total number of unique entities, a prior on the number of unique entities

exhibited in each subset of files. This prior is given by a multinomial-Dirichlet distribution.

3. Given the number of unique entities exhibited in each file, a prior for the number of records in each file associated with each unique entity. By default, this is taken to be a Poisson distribution truncated between 1 and a user specified upper bound.

4. Given the number of records in each file associated with each unique entity, a prior for the within-file partitions of records to entities. This is taken to be uniform over the labellings of such assignments.

5. Given the overlap table and within-file partitions, a prior for the matching between files. This is taken to be uniform over all possible such matchings.

This structured prior induces a Gibbs sampler on the partition of records corresponding to the unique entities in the record linkage task. Similarly the beta prior for bipartite matching however, this Gibbs sampler becomes infeasible for larger linkage tasks.

2.1.1 Fast Beta Linkage

Kundinger, Reiter & Steorts (2024) proposed to model each element of $\boldsymbol{Z}$ independently, relaxing the assumption that there are no duplicates within each database. Under this relaxation, they introduced the fast beta linkage (`fabl`) prior:

$$\mathbb{P}(Z_j = i \mid \pi) = \begin{cases} \pi/n_A, & \text{if } i \leq n_A, \\ 1 - \pi, & \text{if } i = n_A + j, \end{cases} \tag{4}$$

$$\pi \sim \text{Beta}(\alpha_\pi, \beta_\pi). \tag{5}$$

This prior says that a record B has some match in A with probability π, and that each record in A is equally likely to be that match. The `fabl` prior closely mimics the beta prior for bipartite matchings in (3) without enforcing the bipartite matching restriction within the Gibbs sampler. This prior was originally proposed for efficient MCMC sampling, but we recently showed that this prior was conducive to variational inference. This implementation

of the model, dubbed "Variational Beta Linkage" (or `vabl`) was shown to conduct linkage 50-250 times fast than the original MCMC implementation in `fabl`(Kundinger, Aleshin-Guendel & Steorts 2024). If desired, a bipartite matching can be acquired through a simple post-processing step after computing the Bayes estimate.

3 DIRICHLET RECORD LINKAGE

Although the `fabl` framework is effective and efficient when A has no internal duplications, significant problems arise when a record in B has multiple matching records in A. In particular, since matching probability is normalized among all records in A, posterior probability can be split among multiple matching records such that none of the matches achieves high enough posterior probability to be identified through the Bayes estimate. This amounts to a paradox: the more matches that record B_j has in A, the less likely the algorithm is to find a match. Here, we attempt to resolve this paradox by extending the `fabl` framework to handle these internal duplications in A. We emphasize that we are not interested in deduplication within A for its own sake, but rather aim to conduct record linkage in light the problems posed by these duplications. Therefore, duplications within files that have no duplications across files will go undetected; we believe this acceptable for many practical applications.

We provide updated notation to describe matching in this setting. We drop the convention of labelling datasets such that $n_A \geq n_B$, and instead let B denote the file that we assume is free of internal duplicates. In this setting, we refer to the duplicate free file B as the reference file, and the other file A as the target file. Let $Z_j = (Z_{j,1}, \ldots)$ be a set containing the indices for all of the records in A that are a match with record B_j, and let $\boldsymbol{Z} = \{Z_j | j \in [n_B]\}$ denote the collection of all such sets for all records in B. Let $|Z_j|$ denote the number of records in A that are linked to B_j. We use $Z_j = \emptyset$ to denote when B_j has no match in A.

Define a vector of probabilities $\boldsymbol{\pi} = (\pi_0, \ldots)$ where π_k is the probability that some record B_j has exactly k matches in A. To avoid setting a maximum number of matches for any

B_j, we assign $\boldsymbol{\pi}$ a Dirichlet process prior. That is,

$$f(\boldsymbol{\pi}) = \sum_{k=0}^{\infty} \pi_k \delta_{\pi_k}(\boldsymbol{\pi}),$$

where δ_{π_k} is the indicator function which evaluates to zero everywhere, except for $\delta_{\pi_k}(\pi_k) = 1$. We model each π_k as a product of conditional probabilities: let η_k be the probability that some record in B has at least k matches, given that it has at least $k-1$ matches. This gives us the stick breaking representation

$$\pi_k = (1 - \eta_{k+1}) \prod_{c=1}^{k} \eta_c, \tag{6}$$

where η_k are independent random variables from a Beta$(\alpha_\eta, \beta_\eta)$ distribution. We use notation for an arbitrary α_η as it makes future derivations more readable, but we fix $\alpha_\eta = 1$ as is done in a typical Dirichlet process prior.

Conditional on B_j having k matches, we construct a prior specification on $\boldsymbol{Z}$ such that each matching q of length $|q|$ is equally likely. Since there are $\binom{n_A}{|q|} = \frac{n_A!}{(n_A-|q|)!|q|!}$ ways to select $|q|$ matching records out of all n_A possible records, we use

$$\mathbb{P}(Z_j = q|\boldsymbol{\pi}) = \frac{(n_A - |q|)!|q|!}{n_A!}\pi_{|q|}. \tag{7}$$

In Appendix 7.1, we show that the full conditional for Z_j in this setting is given by

$$p\left(Z_j = q|\boldsymbol{\gamma}, \boldsymbol{m}, \boldsymbol{u}, \boldsymbol{\pi}\right) \propto \frac{(n_A - |q|)!|q|!}{n_A!}\pi_{|q|} \prod_{i \in q} w_{ij}, \tag{8}$$

where for all $i \in [n_A]$ and $j \in [n_B]$,

$$w_{ij} = \prod_{f=1}^{F} \prod_{l=1}^{L_f} \left(\frac{m_{fl}}{u_{fl}}\right)^{I(\gamma_{ij}^f = l) I_{obs}(\gamma_{ij}^f)}.$$

Unfortunately, while the full conditional in (8) has a closed form, sampling from it directly is infeasible for almost any record linkage task. In particular, there are 2^{n_A} possible matchings for each B_j,such that even enumerating these possible matchings would be computationally demanding. One could reduce this set of possible matchings by setting a maximum number of matches, K, per record in B, but this would still require $\sum_{k=1}^{K} \frac{n_A!}{(n_A-k)!k!}$ possible options for the set Z_j, and would be prohibitive for most values of n_A seen in record linkage problems. With MCMC however, we can sequentially sample individual components of Z_j at a much lower computational cost, and learn K from the data, rather than setting it ahead of time.

3.1 Sequential Sampler

We use the stick breaking representation in (6) to generalize the fast beta prior in (4), producing a sequence of priors that allows for each record in B to match to multiple records in A. In each iteration of the Gibbs sampler, we sample an initial set of links using η_1. For each record in B that was found to have a link, we remove the linked record in A from consideration, and then sample another potential link with η_2. We continue, using η_k in the k^{th} matching step, until no new links are found, at which we point the matching phase terminates. Crucially, there is no need to specify a maximum number of links per record, as this estimated through the model.

When B_j has been linked to $k-1$ records, we use a prior that says that the probability that B_j has a k^{th} match is η_k, and that all remaining records in A are equally likely to be linked. We use

$$\mathbb{P}(Z_{j,k} = q_k|\eta_k) = \begin{cases} \frac{\eta_k}{n_A-(k-1)}, & q_k \notin A_{j,k}, \\ 1-\eta_k, & q_k = \emptyset; \end{cases}$$

where $A_{j,k} = [n_A] \setminus (Z_{j,1}, \ldots, Z_{j,k-1})$ is the set of records in A that are available to be matched with B_j during matching step k. This sequence of priors leads to sequence of posteriors that can be used to sample arbitrarily many links for record B_j. These posteriors are given by

$$\mathbb{P}(Z_{j,k} = q_k|Z_{j,k-1}, \eta_k, \boldsymbol{m}, \boldsymbol{u}, \boldsymbol{\gamma}) \propto \begin{cases} \frac{\eta_k}{n_A-(k-1)} w_{q_k,j}, & q_k \in A_{j,k}, \\ 1-\eta_k, & q_k = \emptyset. \end{cases} \tag{9}$$

The derivation of this posterior is analogous to the one provided in Appendix 7.1 for the joint distribution of the set Z_j.

This sequential sampler produces an output $Z_j = q = (q_1, \ldots, q_k)$ when $Z_{j,c} = q_c$ for steps $c \in [k]$, and the $k+1$ step produces $Z_{j,k+1} = \emptyset$. While this vector is necessarily ordered, all reorderings of the same elements are equivalent for the purposes of record linkage. That is, $Z_j = (i, i')$ and $Z_j = (i', i)$ both communicate that record B_j is matched to records A_i and $A_{i'}$. Let $\sigma(q)$ denote all $|q|!$ possible orderings of the elements of q. Marginalizing over

such orderings, we have

$$\begin{aligned}
\mathbb{P}(Z_{j,k+1} = \emptyset|\boldsymbol{\gamma}, \boldsymbol{m}, \boldsymbol{u}, \boldsymbol{\eta}) \sum_{q' \in \sigma(q)} \prod_{c=1}^{k} \mathbb{P}(Z_{j,c} = q'_c|\boldsymbol{\gamma}, \boldsymbol{m}, \boldsymbol{u}, \boldsymbol{\eta}) \\
&\propto (1 - \eta_{k+1}) \sum_{q' \in \sigma(q)} \prod_{c=1}^{k} \frac{\eta_c}{n_A - (c-1)} \prod_{c=1}^{k} w_{q'_c,j} \\
&= (1 - \eta_{k+1}) k! \prod_{c=1}^{k} \frac{\eta_c}{n_A - (c-1)} \prod_{c=1}^{k} w_{q_c,j} \\
&= \frac{(n_A - k)! k!}{n_A!} (1 - \eta_{k+1}) \prod_{c=1}^{k} \eta_c \prod_{c=1}^{k} w_{q_c,j} \\
&= \frac{(n_A - k)! k!}{n_A!} \pi_k \prod_{c=1}^{k} w_{q_c,j} \\
&= p\left(Z_j = q|\boldsymbol{\gamma}, \boldsymbol{m}, \boldsymbol{u}, \boldsymbol{\pi}\right).
\end{aligned}$$

Thus, the probability that the sequential process samples each of the components of q (in any order) is equal to the joint probability of q as expressed in the joint distribution in (8).

We take a moment to emphasize the computational advantages to this approach. Sampling from the full conditional shown in (8) would have complexity $O(2^{n_A} n_B)$, and would be nearly impossible for any reasonable record linkage task. Using the sequential sampler shown in (9) in Gibbs iteration s has complexity $O\left(n_A(n_B + \sum_{k=1}^{K} n_{k-1}^{(s)})\right)$, where $n_k^{(s)} = \sum_{j=1}^{n_B} I\left(|Z_j^{(s)}| \geq k\right)$ is the number of records in B linked to at least k records in A. Even with the sequential sampler, this complexity would still grow quadratically as the size of the linkage task grows. However, we follow the methodology from Kundinger, Reiter & Steorts (2024) to conduct sampling at the level of the unique instances of γ_{ij} in the linkage task, instead of all $n_A n_B$ comparison vectors to greatly reduce computational complexity. We review this methodology in Appendix 7.2, and provide an adaptation of (9) using this methology in Appendix 7.3.1. Using this approach, the sampler has complexity $O\left(P(n_B + \sum_{k=1}^{K} n_{k-1}^{(s)})\right)$, which grows linearly in the size of the base dataset. Thus, we have produced a computationally efficient sampler with speed comparable to the $O(P n_B)$ complexity of `fabl`.

3.2 Bayes Estimate

Many of the loss functions that have been proposed for different record linkage models unfortunately are not amenable to `DRL`. For example, Sadinle (2017) proposed a loss function that ensured a Bayes estimate that respected one-to-one matching and allowed certain components of the estimate to remain undeclared and left for clerical review. However, this loss function relies on making exactly one linkage decision per record in B, and is not amenable to our scenario where we do not set a maximum number of links per record. More recently, Aleshin-Guendel & Sadinle (2023) proposed a loss function that estimated a partition, and thus respected transitivity, and also allowed for leaving parts of the Bayes estimate undeclared. The implementation of this loss function however relies on each MCMC sample itself being a partition, which does not hold true in our relaxed setting.

For this reason, we instead use a generalization of the absolute number of errors loss function from Tancredi & Liseo (2011). It is more convenient here to use the matrix representation of the linkage structure Δ as in (1), rather than the $\boldsymbol{Z}$ notation used throughout Section 3. Specifically, we use

$$L(\Delta, \hat{\Delta}) = \sum_{i=1}^{n_A} \sum_{j=1}^{n_B} \tau \hat{\Delta}_{ij}(1 - \Delta_{ij}) - \Delta_{ij}(1 - \hat{\Delta}_{ij}),$$

where $\tau > 0$ is a penalization term specified by the user to weigh the importance of false matches relative to false non-matches. This loss function is equivalent to

$$L(\Delta, \hat{\Delta}) = \sum_{i=1}^{n_A} \sum_{j=1}^{n_B} \tau \hat{\Delta}_{ij} - (1 + \tau)\hat{\Delta}_{ij}\Delta_{ij} + \Delta_{ij}.$$

Minimizing $L(\Delta, \hat{\Delta})$ is equivalent to maximizing the quantity

$$W(\hat{\Delta}) = \hat{\Delta}_{ij}[(1 + \tau)\Delta_{ij} - \tau].$$

The expectation of this quantity with respect to the linkage structure is given by

$$E[W(\hat{\Delta})] = \hat{\Delta}_{ij}[(1 + \tau)\mathbb{P}(\Delta_{ij} = 1) - \tau],$$

which is maximized by the decision rule

$$\hat{\Delta}_{ij} = \begin{cases} 1, & \text{if } \mathbb{P}(\Delta_{ij} = 1) > \frac{\tau}{1+\tau}, \\ 0 & \text{otherwise.} \end{cases} \tag{10}$$

When the user specifies $\tau = 1$, indicating that false matches and false non-matches are weighted equally, we recover the decision rule of Tancredi & Liseo (2011) with a threshold of 0.5 to declare matches. This decision rule also functions similarly to that of Sadinle (2017) under certain choices of the losses.

Since the `DRL` prior as specified in in (7) does not strictly enforce the requirement that there are no duplicates in B, it is possible for this Bayes estimate to link multiple records in B to the same record in A. To obtain a Bayes estimate that corresponds to our model assumptions, we minimize the expected loss subject to the constraint that $\hat{Z}_{j,k} \neq \hat{Z}_{j',k}$ for all $j \neq j'$ and all k. This means that when multiple records in B are declared as matching to the same record in A in the initial Bayes estimate, we simply accept the one with highest posterior probability, and declare all other pairs as non-matching. We make the simplifying assumption that all records that match to the same B_j also match to each other, but this is not verified in the model. We believe this assumption is well suited for many practical applications, and is worth the computational advantages of the `DRL` approach.

4 SIMULATIONS

We demonstrate the accuracy of `DRL` through an adaptation of the simulation study from Aleshin-Guendel & Sadinle (2023). We compare the performance of `DRL` against models implemented in `fastLink`, `vabl`, and `multilink`. Results from `BRL` and `fabl` are similar to those of `vabl`, and are therefore omitted.

For each simulation, we construct two data files A and B of size $n_A = n_B = 500$. File B is constructed to have no internal duplicates, and we designate 10% of the records in B (or 50 records) to have some number of matches in A. The number of matches these records have in A is sampled from a Poisson distribution truncated to the interval [1, 5]. We use Poisson distributions with means 0.1, 1, and 2, and denote these settings as low, medium, and high duplication settings respectively. Each of these duplicate records has up to three altered fields relative to the original record in A. After accounting for the records in A that have a match in B, the rest of the records in A are comprised of unique entities. We follow the methodology of Christen & Vatsalan (2013) to generate original records for the

		Level of Disagreement		
Fields	Similarity	1	2	3
First and Last Name	Levenstein	0	(0, .25]	(.5, .1]
Age, Postal Code, and Occupation	Binary	Agree	Disagree	

Table 1: Construction of comparison vectors for accuracy study with simulated data files of Section‘4.

unique entities by simulating first name, last name, age, postal code, and occupation from frequency tables based on demographic information in Australia. Fields are then chosen for distortion uniformly at random. Names are subject to string insertions, deletions and substitutions, as well as common keyboard, phonetic, and optical recognition errors. Age and occupation are distorted through keyboard errors and missingness.

We create comparison vectors for these linkage tasks according to the thresholds described in Table 1. We use uniform priors for $\boldsymbol{m}$ and $\boldsymbol{u}$, with $\alpha_{fl} = \beta_{fl} = 1$ for all f and l. We use uniform priors for π for `vabl`, and also uniform priors for each η_k in the sequential sampler for `DRL`. For `multilink`, we correctly designate that there are no duplicates in B and that A can have up to 5 records that are linked together (and to one record in B). We use the default setting of a Poisson distribution with mean 1 for the number of records clustered together in A. In consultation with the authors of `multilink`, we also implement the method with filtering, requiring that both first and last name have an agreement level of 1 or 2 in order for the record pair to be considered for matching. For `DRL` and `multilink`, we use a Gibbs sampler with 1000 iterations and discard the first 100 as burn-in. For `fastLink` and `vabl`, we run the algorithm with a convergence tolerance of 1e-04, in line with default settings. `fastLink` results are provided without the Jaro correction that enforces one-to-one matching.

We compute point estimates of the linkage structure using comparable decision rules for each method. For each Bayesian method, we use losses reflecting equal importance for false matches and false non-matches. For `vabl` and `DRL`, this results in a probability threshold of 0.5 to declare record pairs as a match. We choose this threshold for `fastLink` as well. For each of these, we obtain an initial point estimate, then post-process so that no two records

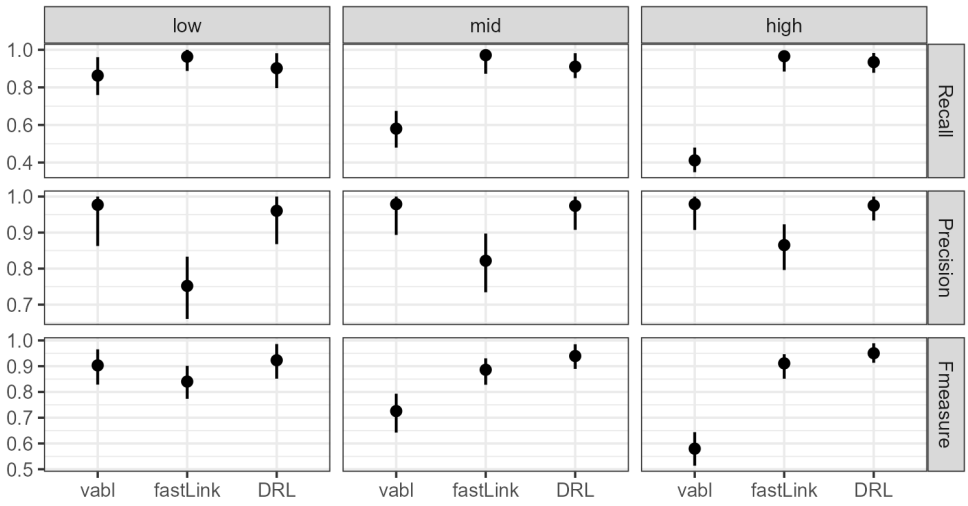

Figure 1: Results from simulation study of described in Section 4 for `vabl`, `fastLink`, and `DRL`. For each method and simulation setting, dots show the median, and bars show the 2.5^{th} and 97.5^{th} percentiles of recall and precision of Bayes estimates across 100 simulated datasets.

in B match to same record in A. For `multilink`, Aleshin-Guendel & Sadinle (2023) do not provide closed form expressions for the decision rule for their Bayes estimate, but we select analogous losses.

To compare the accuracy of each method, we use recall, precision, and F-measure as defined by Christen (2012). Recall is the proportion of true matches found by the estimate; given by

$$\theta_{\text{Recall}} = \frac{\sum_{i=1}^{n_A}\sum_{j=1}^{n_B}\Delta_{ij}\hat{\Delta}_{ij}}{\sum_{i=1}^{n_A}\sum_{j=1}^{n_B}\Delta_{ij}}.$$

Precision is the proportion of links found by the estimate that are true matches, given by

$$\theta_{\text{Precision}} = \frac{\sum_{i=1}^{n_A}\sum_{j=1}^{n_B}\Delta_{ij}\hat{\Delta}_{ij}}{\sum_{i=1}^{n_A}\sum_{j=1}^{n_B}\hat{\Delta}_{ij}}.$$

Lastly, we compute F-measure, a pseudo-average of recall and precision, given by

$$\theta_{\text{F-measure}} = 2\left(\frac{\theta_{\text{Recall}} \times \theta_{\text{Precision}}}{\theta_{\text{Recall}} + \theta_{\text{Precision}}}\right).$$

In Figure 1, we see that in the low duplication setting, `vabl` performs comparatively with `DRL` in terms of recall and precision. However, as the number of internal duplications

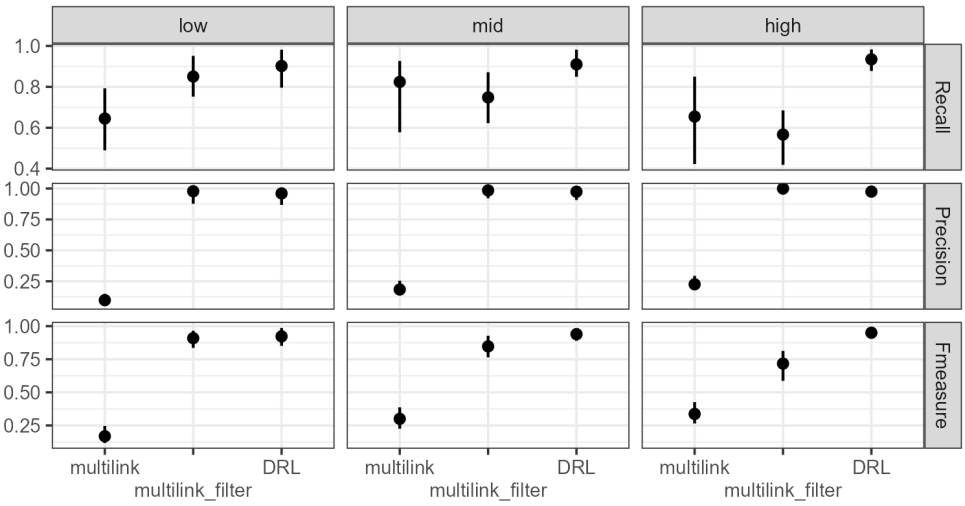

Figure 2: Results from simulation study of described in Section 4 for `multilink` (with and without filtering) and `DRL`.

in A grows, recall suffers. This is because `vabl` is constrained to identify at most one match for each record in B, and in many cases, posterior probability is split among several truly matching record pairs such that none of them reach the threshold to be declared a match. In contrast, `fastLink` maintains higher recall at all duplication settings, and precision improves as the rate of duplication grows. Upon closer analysis, the actual number of false positive matches remains similar across the duplication settings, but the larger number of true positives results in higher precision.

The results from `multilink` require a more nuanced interpretation. While `DRL` only evaluates the record pairs between each record in A and one particular B_j, `multilink` must also consider the comparison vectors between record pairs in A. Notably, each record in A can have up to three errors relative to the original record in B, which means that records in A can have up to six errors relative to each other. This means that `multilink` is engaging in a more difficult matching process, and in this sense, is not directly comparable to `DRL`. With that caveat mentioned, we see in Figure 2 that `multilink` without filter has poor precision under these simulation settings. With filtering however, the precision greatly improves. Nonetheless, as the rate of duplication increases, recalls suffers for both implementations of `multilink`. This because 5% to 10% of truly matching record pairs get removed from

consideration by our chosen filtering scheme, and because of the difficulty of matching large clusters as described above.

In contrast, `DRL` maintains strong performance at all duplication rates considered. Notably, `DRL` has only slightly poorer precision than `vabl` even in the low duplication setting where the model assumptions of `vabl` are closest to being satisfied. Additionally, `DRL` is consistently able to match records in B to several records in A without needing to set a maximum number of links through the prior distribution, as is needed in `multilink`.

We note that `vabl` is incredibly fast, reaching convergence in around 0.01 seconds on average. Due to the design of the `fastLink` package in `R`, we cannot directly measure the speed of the EM algorithm separately from the construction of the comparison vectors, but we assume that it is even faster than `vabl`. When we use filtering, `multilink` completes 1000 iterations of the Gibbs sampler in about 0.5 seconds. Without filtering however, it takes around 1000 seconds. Lastly, `DRL` completes the 1000 Gibbs iterations in about 15 seconds.

We conducted another simulation with 30% of records in B having matches in A, with all other settings held constant. Under these settings, `fastLink` and `multilink` perform better, but `DRL` still outperforms. We include these results in Appendix 7.4.

5 CASE STUDY

We demonstrate our method on two snapshots of the North Carolina Voter Registration (`NCVR`) database taken two months apart (Christen 2014). The snapshots are filtered to include only those voters whose details changed over the two-month period, so there are no matching records with full agreement on all fields. We use first name, middle name, last name, as string fields, and street address and age as categorical fields. Unique voter registration numbers are provided, however they are known to contain some errors. The `NCVR` dataset is not publicly available due to sensitive information. However, we have permission to utilize it for publication by its owner.

Using the voter registration numbers, we can see that each file has internal duplication rates of about 1%. In this analysis, we deduplicate file B, and leave A with duplicates. In

Method	Recall	Precision	F-measure
DRL	0.9891623	0.9735309	0.9812843
vabl	0.9743739	0.9798799	0.9771191
fastLink	0.9988472	0.8494865	0.9181321
fastLink (with Jaro)	0.9514490	0.9664229	0.9588775

Table 2: Accuracy results for NCVR, showing the DRL has the highest F-measure.

practice, such low amount of internal duplication may not warrant the use of DRL since vabl is considerably faster. However, we demonstrate here that even with such few internal duplications, DRL is effective at identifying these multiple matches and does not declare too many false matches.

We compare four approaches: vabl, DRL, fastLink, and fastLink with the Jaro correction. This task is too large for running multilink in its current implementation in R, and is thus omitted. We use a threshold of 0.5 to declare matches, and post-process so no two records in B match to the same record in A, as done in Section 4. Results are in Table 2.

We see that fastLink without the one-to-one post-processing results in the highest recall, but leads to an undesirable amount of false positives. Since the posterior match probability of each record pair is computed independently, we expect such behavior on large linkage tasks such as this. When we use the Jaro post-processing to achieve a one-to-one matching, we get considerably better results. However, we lose the possibility of identifying cases where one record in B matches to two records in A, and we still have worse recall and precision than attained under vabl.

In Figure 3, we show the precision, recall, and F-measure under various choices of match probability thresholds, corresponding to different choices for τ in the decision rule in (10). We see that vabl and DRL produce more refined match probability estimates, allowing the accuracy metrics to vary smoothly across the different thresholds. In contrast, under fastLink, all record pairs of the same agreement pattern have the same posterior probability, leading to much more coarse posterior distribution and more erratic curves.

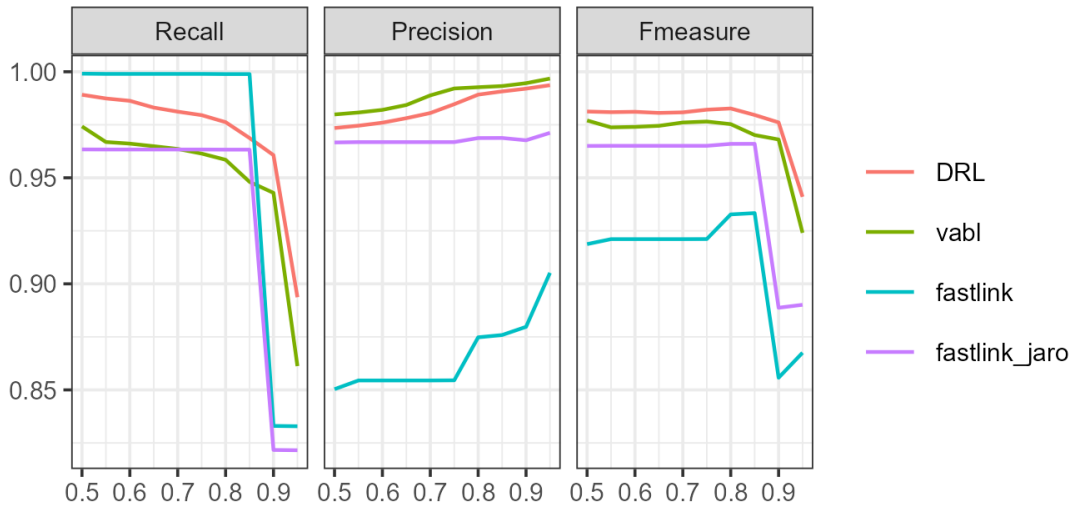

Figure 3: Accuracy metrics for NCVR data at various match probability thresholds. We see that `DRL` maintains the strongest F-measure for all thresholds considered.

We see that `vabl` maintains the highest precision at all thresholds, but that `DRL` is able to identify enough more additional true matches that `DRL` maintains the highest overall F-measure.

Lastly, we note a limitation in using preexisting software for record linkage tasks. In its currently implementation, `fastLink` allows users to define normalized Levenshtein distance thresholds for coding field comparisons into three discrete levels. In this data however, it is common for a record to contain a middle initial (like "E"), rather than a full middle name (like "Elizabeth"). Using Levenshtein distance thresholds, the comparison field for middle name for such a pair of records could be coded as a full disagreement, when it should be more reasonably given its own agreement level. Likewise, when two records have a middle initial that happens to match, this is coded as a full agreement, even though it is possible these initials represent different names. In this analysis, 27.6% of the false matches under `DRL` at the 0.5 probability threshold included at least one record containing a middle initial rather than middle name. A more careful construction of the comparison vectors is likely to improve results for all methods considered.

6 CONCLUSION

In this paper, we have introduced Dirichlet Record Linkage, a new Bayesian method for matching one duplicate free reference file to another file that may have duplicates. We have discussed the trade offs between model complexity and speed across several comparison vector based methods. If a practitioner is sure that each file has no (or reasonably few) internal duplicates, then `vabl` allows for high accuracy and low computation time. If a practitioner is interested in deduplicating both datasets simultaneously, the additional computation of `multilink` may be worth the ability to needs to more carefully model the partition structure of the data. When a practitioner is confident that one dataset is relatively duplicate free, `DRL` offers an intermediate method that is robust to internal duplications in the other file while maintaining reasonable speed.

In future work, we seek to adapt the methodology presented here to more closely approximate the model implemented by `multilink`. At current, `DRL` either post-processes a Bayes estimate such that there are no implicit matchings between two records in the "duplicate-free" data file, or refrains from post-processing altogether, leaving a Bayes estimate that may not respect transitive closure. While we have argued this acceptable for many practical purposes, we acknowledge that it is not acceptable for others. Additionally, while the MCMC sampler derived in this paper was shown to scale to sizeable linkage tasks, we also acknowledge the inherent scalability limitations of MCMC methods. We hope to explore the utility of variational inference in addressing these limitations(as in Kundinger, Aleshin-Guendel & Steorts (2024)), and develop new methods for the challenges of modeling the joint distribution of clusters of matching records of varying sizes.

REFERENCES

Aleshin-Guendel, S. & Sadinle, M. (2023), 'Multifile partitioning for record linkage and duplicate detection', *Journal of the American Statistical Association* **118**(543), 1786–1795.

Betancourt, B., Sosa, J. & Rodríguez, A. (2022), 'A prior for record linkage based on allelic partitions', *Computational Statistics & Data Analysis* **172**, 107 – 474.

Bilenko, M., Mooney, R. J., Cohen, W. W., Ravikumar, P. & Fienberg, S. E. (2003), 'Adaptive Name Matching in Information Integration', *IEEE Intelligent Systems* **18**(5), 16–23.

Binette, O. & Steorts, R. C. (2022), '(Almost) all of entity resolution', *Science Advances* **8**(12), eabi8021.

Christen, P. (2012), *Data Matching: Concepts and Techniques for Record Linkage, Entity Resolution, and Duplicate Detection*, Data-Centric Systems and Applications, Springer-Verlag, Berlin Heidelberg.

Christen, P. (2014), Preparation of a real temporal voter data set for record linkage and duplicate detection research, Technical report, Australian National University.
URL: *http://users.cecs.anu.edu.au/ christen/publications/ncvoter-report-29june2014.pdf*

Christen, P. & Vatsalan, D. (2013), Flexible and extensible generation and corruption of personal data, *in* 'Proceedings of the 22nd ACM International Conference on Information and Knowledge Management', CIKM '13, Association for Computing Machinery, New York, NY, USA, p. 1165–1168.

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

Enamorado, T., Fifield, B. & Imai, K. (2019), 'Using a Probabilistic Model to Assist Merging of Large-Scale Administrative Records', *American Political Science Review* **113**(2), 353–371.

Fellegi, I. P. & Sunter, A. B. (1969), 'A Theory for Record Linkage', *Journal of the American Statistical Association* **64**(328), 1183–1210.

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

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**(406), 414–420.

Kundinger, B., Aleshin-Guendel, S. & Steorts, R. C. (2024), 'Variational Beta Linkage', *TBD* .

Kundinger, B., Reiter, J. & Steorts, R. C. (2024), 'Efficient and Scalable Bipartite Matching with Fast Beta Linkage (fabl)', *Bayesian Analysis (Accepted)* .

Linacre, R., Lindsay, S., Manassis, T., Slade, Z., Hepworth, T., Kennedy, R. & Bond, A. (2022), 'Splink: Free software for probabilistic record linkage at scale.', *International Journal of Population Data Science* **7**(3).
URL: *https://ijpds.org/article/view/1794*

Little, R. J. A. & Rubin, D. B. (2002), *Statistical Analysis with Missing Data*, Wiley.

Marchant, N. G., Kaplan, A., Elazar, D. N., Rubinstein, B. I. & Steorts, R. C. (2021), 'd-blink: Distributed end-to-end bayesian entity resolution', *Journal of Computational and Graphical Statistics* **30**(2), 406–421.

McVeigh, B. S., Spahn, B. T. & Murray, J. S. (2019), 'Scaling bayesian probabilistic record linkage with post-hoc blocking: An application to the california great registers', *arXiv preprint arXiv:1905.05337* .

Murray, J. S. (2016), 'Probabilistic record linkage and deduplication after indexing, blocking, and filtering', *Journal of Privacy and Confidentiality* **7**(1), 3–24.

Papadakis, G., Ioannou, E., Thanos, E. & Palpanas, T. (2021), 'The Four Generations of Entity Resolution', *Synthesis Lectures on Data Management* **16**, 1–170.

Sadinle, M. (2017), 'Bayesian Estimation of Bipartite Matchings for Record Linkage', *Journal of the American Statistical Association* **112**(518), 600–612.

Sadinle, M. (2020), *BRL: Beta Record Linkage*. R package version 0.1.0.
URL: *https://CRAN.R-project.org/package=BRL*

Sariyar, M. & Borg, A. (2010), 'The RecordLinkage Package: Detecting Errors in Data', *The R Journal* **2**(2), 61–67.
URL: *https://doi.org/10.32614/RJ-2010-017*

Steorts, R. C., Hall, R. & Fienberg, S. E. (2016), 'A Bayesian Approach to Graphical Record Linkage and Deduplication', *Journal of the American Statistical Association* **111**(516), 1660–1672.

Steorts, R. C., Ventura, S. L., Sadinle, M. & Fienberg, S. E. (2014), A Comparison of Blocking Methods for Record Linkage, *in* 'Privacy in Statistical Databases', Lecture Notes in Computer Science, Springer, Cham, pp. 253–268.

Tancredi, A. & Liseo, B. (2011), 'A Hierarchical Bayesian Approach to Record Linkage and Size Population Problems', *The Annals of Applied Statistics* **5**(2B), 1553–1585.

Winkler, W. E. (1990), String Comparator Metrics and Enhanced Decision Rules in the Fellegi–Sunter Model of Record Linkage, *in* 'Proceedings of the Section on Survey Research Methods', American Statistical Association, pp. 354–359.

Wortman, J. P. H. (2019), Record linkage methods with applications to causal inference and election voting data, PhD thesis, Duke University.

7 APPENDIX

7.1 Full Conditional for the Set Z_j

We express the likelihood in (2) for the model assumptions of `DRL` as

$$\mathcal{L}(\boldsymbol{Z}, m, u \mid \boldsymbol{\gamma}) = \prod_{i=1}^{n_A}\prod_{j=1}^{n_B}\prod_{f=1}^{F}\prod_{l=1}^{L_f}\left[m_{fl}^{I(i\in Z_j)}u_{fl}^{I(i\notin Z_j)}\right]^{I(\gamma_{ij}^f=l)I_{obs}(\gamma_{ij}^f)}.$$

We follow the proof technique introduced by Wortman (2019) and elaborated by Kundinger, Reiter & Steorts (2024). When B_j does not link to any record in A (such that $|Z_j| = 0$) the contribution to the likelihood is simply a product of u parameters, which we will call c_j:

$$\mathbb{P}(\Gamma_{\cdot j}|\boldsymbol{m}, \boldsymbol{u}, \boldsymbol{\pi}, Z_j = \emptyset) = \prod_{i=1}^{n_A}\prod_{f=1}^{F}\prod_{l=1}^{L_f}u_{fl}^{I(\gamma_{ij}^f=l)I_{obs}(\gamma_{ij}^f)} = c_j.$$

When $Z_j = q = (q_1, \ldots, q_k)$ for some $|q| > 0$, we have

$$\mathbb{P}(\Gamma_{\cdot j}|\boldsymbol{m}, \boldsymbol{u}, \boldsymbol{\pi}, Z_j = q) = \prod_{i\in q}\prod_{f=1}^{F}\prod_{l=1}^{L_f}m_{fl}^{I(\gamma_{ij}^f=l)I_{obs}(\gamma_{ij}^f)}\prod_{i\notin q}\prod_{f=1}^{F}\prod_{l=1}^{L_f}u_{fl}^{I(\gamma_{ij}^f=l)I_{obs}(\gamma_{ij}^f)}.$$

We multiply and divide by the u parameters for the matching record pairs to obtain

$$\begin{aligned}\mathbb{P}(\Gamma_{\cdot j}|\boldsymbol{m}, \boldsymbol{u}, \boldsymbol{\pi}, Z_j = q) &= \prod_{i\in q}\prod_{f=1}^{F}\prod_{l=1}^{L_f}\left(\frac{m_{fl}}{u_{fl}}\right)^{I(\gamma_{ij}^f=l)I_{obs}(\gamma_{ij}^f)}\prod_{i=1}^{n_A}\prod_{f=1}^{F}\prod_{l=1}^{L_f}u_{fl}^{I(\gamma_{ij}^f=l)I_{obs}(\gamma_{ij}^f)}\\ &= c_j\prod_{i\in q}w_{ij}.\end{aligned}$$

Lastly, we multiply the likelihood by the prior in (7) to obtain the posterior distribution. We have

$$\begin{aligned}p\left(Z_j = q|\boldsymbol{\gamma}, \boldsymbol{m}, \boldsymbol{u}, \boldsymbol{\pi}\right) &= \frac{\frac{(n_A-k)!|k|!}{n_A!}\pi_k c_j\prod_{i\in q}w_{ij}}{\sum_{h\in\mathcal{Z}}\frac{(n_A-|h|)!|h|!}{n_A!}\pi_{|h|}c_j\prod_{i\in h}w_{ij}}\\ &= \frac{\frac{(n_A-k)!|k|!}{n_A!}\pi_k\prod_{i\in q}w_{ij}}{\sum_{h\in\mathcal{Z}}\frac{(n_A-|h|)!|h|!}{n_A!}\pi_{|h|}\prod_{i\in h}w_{ij}}\\ &\propto \frac{(n_A-k)!|k|!}{n_A!}\pi_k\prod_{i\in q}w_{ij}.\end{aligned}$$

Importantly, the constant c_j is not found in the final expression because the probability mass associated with every potential value for Z_j shares the same c_j. This does not occur due to proportionality.

7.2 Hashing for Efficient Posterior Inference

Following Kundinger, Reiter & Steorts (2024) we use hashing to reduce the computational complexity of the Gibbs sampler. Since each component γ_{ij}^f of the comparison vector is discrete, there are only finitely many possible realizations of the comparison vector γ_{ij}. Let P be the number of unique agreement patterns observed in $\boldsymbol{\gamma}$. This number is bounded above by $P^* = \prod_{f=1}^{F}(L_f + 1)$, where the addition of 1 to L_f for each field accounts for the possibility of missing values. This upper bound does not scale with n_A or n_B, but rather is determined by F and L_f.

We adopt a hashing function to map the P unique agreement patterns present in the comparison vectors to the integers in $[P]$. Let

$$h_f^{(i,j)} = I_{obs}(\gamma_{ij}^f) 2^{\gamma_{ij}^f + I(f>1)\times\sum_{e=1}^{f-1}(L_e-1)}$$

denote a hashed value for field f for record pair (i, j). Summing over the fields for the agreement pattern of record pair (i, j) gives the hash value $h^{(i,j)} = \sum_{f=1}^{F} h_f^{(i,j)}$. We enumerate the unique hashed agreement patterns from 1 to P. Denote each agreement pattern as $h_p = (h_p^1, \ldots, h_p^F)$, so that when record pair (i, j) exhibits agreement pattern p, we write $\gamma_{ij} = h_p$. Collect the agreement patterns as $\mathcal{P} = \{h_p \mid p \in [P]\}$.

Let $e(h_p)$ denote the $\sum_{f=1}^{F} L_f$ length vector in which the $l + \sum_{k=1}^{f-1} L_k$ component is 1 when $h_p^f = l$, and 0 otherwise. Observe that $e(h_p)$ represents a long version of h_p, which is useful for computational purposes. Let $N_{p_j} = \sum_{i=1}^{n_B} I(\gamma_{ij} = h_p)$ denote the number of records in A with which record j in B has agreement pattern p. Collect these counts in $\mathcal{N} = \{N_{p_j} \mid j \in [n_B], p \in [P]\}$. Let $N_p = \sum_{j=1}^{n_B} N_{p_j}$ denote the total number of record pairs with agreement pattern p. Finally, let $r_{p_j} = \{i \in [n_A] \mid \gamma_{ij} = h_p\}$ be the set of records in A with which record j in B has agreement pattern p, and collect these sets as $\mathcal{R} = \{r_{p_j} \mid p \in [P], j \in [n_B]\}$.

The set $\tilde{\boldsymbol{\gamma}} = \{\mathcal{P}, \mathcal{R}, \mathcal{N}\}$ fully characterizes the information in $\boldsymbol{\gamma}$ at a reduced storage cost. In particular, by letting

$$m_p = \prod_{f=1}^{F} \prod_{l=1}^{L_f} m_{fl}^{I(h_p^f = l) I_{obs}(h_p^f)}$$

and

$$u_p = \prod_{f=1}^{F} \prod_{l=1}^{L_f} u_{fl}^{I(h_p^f = l) I_{obs}(h_p^f)},$$

the likelihood function in (2) can be written as

$$\mathcal{L}(\boldsymbol{Z}, m, u \mid \tilde{\Gamma}) = \prod_{j=1}^{n_B} \prod_{p=1}^{P} \prod_{i \in r_{p_j}} m_p^{I(i \in Z_j)} u_p^{I(i \notin Z_j)}. \tag{11}$$

Thus this hashing procedure allows for exact inference on the original model, without any approximation.

7.2.1 Batched Computation of Comparison Vectors

When it becomes infeasible to store all $n_A n_B$ comparison vectors at one time, we partition A and B into t_1 and t_2 smaller disjoint batches, as proposed by Kundinger, Reiter & Steorts (2024). Let $\{A^1, \ldots, A^{t_1}\}$ be a partition of A such that

$$\cup_{a=1}^{t_1} A^a = A \quad \text{and} \quad A^a \cap A^{a'} = \emptyset \quad \text{for all} \quad a \neq a'.$$

Likewise, let $\{B^1, \ldots, B^{t_2}\}$ be a partition of B such that

$$\cup_{b=1}^{t_2} B^b = B \quad \text{and} \quad B^b \cap B^{b'} = \emptyset \quad \text{for all} \quad b \neq b'.$$

For each a and b, we compute comparison vectors for all records in $A^a \times B^b$, creating the comparison matrix $\boldsymbol{\gamma}^{ab}$.

We then perform hashing on the batch of comparison data, obtain a compressed $\tilde{\boldsymbol{\gamma}}^{ab}$, and delete the memory intensive matrix $\boldsymbol{\gamma}^{ab}$ before continuing with the next batch of data. That is, we calculate

$$\begin{aligned} r_{p_j}^{ab} &= \{i \in [n_A] \mid \gamma_{ij} = h_p, i \in A^a, j \in B^b\}, \\ N_{p_j}^{ab} &= |r_{p_j}^{ab}|. \end{aligned}$$

We compute these quantities for each batch in parallel. Finally, summary statistics from each pairwise batch comparison are combined to recover summary statistics for the full

comparison matrix $\boldsymbol{\gamma}$ through

$$r_{p_j} = \{r_{p_j}^{ab}\}_{a=1,b=1}^{t_1,t_2} \tag{12}$$
$$\text{and } N_{p_j} = \sum_{a=1}^{t_1}\sum_{b=1}^{t_2} N_{p_j}^{ab}.$$

We can further reduce memory costs through storage efficient indexing (SEI) (Kundinger, Reiter & Steorts 2024). All records i in A that share agreement pattern p with record B_j have the same weight w_p (as described in Appendix 7.2). These records have the same probability to be identified as the link for record B_j. Thus, records $i \in r_{p_j}$ with large N_{p_j} are unlikely to be sampled consistently enough to be deemed a match in the Bayes estimate. Rather than store all of these record labels, we store only a small number S. For each $r_{p_j}^{ab}$, we sample S indices without replacement to form $\text{SEI}(r_{p_j}^{ab})$. We collect these memory-reduced lists to form $\text{SEI}(r_{p_j})$ as in (12), and collect these to form $\text{SEI}(\mathcal{R})$.

Lastly, when using SEI along with batching in this manner, we implicitly restrict the sequential sampler described in Section 3.1. If $|\text{SEI}(r_{p_j})| < N_{p_j}$ after conducting SEI, there is a non-zero probability we can sample an agreement pattern in (13) and not have a corresponding record label for it when sampling in (14). Therefore, when using SEI, we must limit the amount of sequential sampling steps to some number $K < S$.

7.3 Gibbs Sampler

We initialize $\boldsymbol{Z}$ to reflect no matches across data files; that is, $Z_j = \emptyset$ for all $j \in [n_B]$. We then iteratively sample $\boldsymbol{m}$, $\boldsymbol{u}$, $\boldsymbol{\eta}$, and $\boldsymbol{Z}$ for a prespecified number of Gibbs iterations. A computationally efficient method to sample $\boldsymbol{Z}$ using the hashing described in Appendix 7.2 is provided in Appendix 7.3.1. The the full conditional for $\boldsymbol{\eta}$ is provided in Appendix 7.3.2, and full conditionals for $\boldsymbol{m}$ and $\boldsymbol{u}$ are provided in Appendix 7.3.3.

7.3.1 Efficient Sequential Sampling for Components $Z_{j,k}$

Let $\tilde{\boldsymbol{\gamma}} = \{\mathcal{P}, \mathcal{R}, \mathcal{N}\}$ denote the set of summary statistics described in Appendix 7.2. We provide a computationally efficient implementation of the sampler in (9) by using, and updating, quantities in $\mathcal{R}$ and $\mathcal{N}$ as the sequential sampler progresses. Let $r_{p_j}(k) =$

r_{p_j} $(Z_{j,1}, \ldots Z_{j,k-1})$ be the set of records of agreement pattern p available for matching at the k^{th} matching step, and let $N_{p_j}(k) = |r_{p_j}(k)|$ be the number of such records. We first sample among $P+1$ options for the agreement pattern between B_j and its potential link. Define r as an arbitrary set of records. We have

$$p\left(Z_{j,k} \in r \mid \tilde{\boldsymbol{\gamma}}, \boldsymbol{m}, \boldsymbol{u}, \eta_k\right) \propto \begin{cases} \frac{\eta_k N_{p_j}(k)}{n_A - (k-1)} w_p, & r = r_{p_j}(k); \\ 1 - \eta_k, & r = \emptyset. \end{cases} \tag{13}$$

Since all remaining records in A sharing the same agreement pattern with B_j are equally likely, we then sample among candidate records uniformly using

$$p\left(Z_{j,k} = q \mid Z_{j,k} \in r, \boldsymbol{m}, \boldsymbol{u}, \eta_k\right) = \begin{cases} \frac{1}{N_{p_j}(k)}, & r = r_{p_j}(k) \text{ and } q \in r; \\ 1, & r = \emptyset \text{ and } q = \emptyset. \end{cases} \tag{14}$$

Sampling from (13) and (14) is mathematically equivalent to sampling from the full conditional for $Z_{j,k}$ in (9).

7.3.2 Sampling Match Rate η_k

Define $n_k = \sum_{j=1}^{n_B} I\left(|Z_j| \geq k\right)$ to be the number of records in B that are linked to at least k records in A (with $n_0 = n_B$). Note that step k of the sequential sampler acts on n_{k-1} records in B and identifies n_k links. As such, the full conditional for η_k is a standard Beta-Binomial update. For each η_k, we have

$$\begin{aligned} &\mathbb{P}\left(\eta_k \middle| \boldsymbol{\gamma}, \boldsymbol{Z}, \boldsymbol{m}, \boldsymbol{u}\right) \\ &\propto \mathbb{P}\left(\boldsymbol{Z} | \eta_k\right) \mathbb{P}(\eta_k) \\ &= \prod_{j=1}^{n_B} \mathbb{P}\left(Z_j | \eta_k\right) \mathbb{P}(\eta_k) \\ &\propto \prod_{j=1}^{n_B} \left[\eta_k^{I(|Z_j| \geq k)} (1-\eta_k)^{1 - I(|Z_j| \geq k)}\right]^{I(|Z_j| \geq k-1)} \eta_k^{\alpha_\eta - 1} (1-\eta_k)^{\beta_\eta - 1} \\ &\propto \eta_k^{n_k + \alpha_\eta - 1} (1-\eta_k)^{n_{k-1} - n_k + \beta_\eta - 1}. \end{aligned}$$

Thus we have

$$\eta_k | \boldsymbol{Z}, \boldsymbol{m}, \boldsymbol{u}, \boldsymbol{\gamma} \sim \text{Beta}\left[n_k + \alpha_\eta, n_{k-1} - n_k + \beta_\eta\right].$$

7.3.3 Full Conditional for m and u

The $\boldsymbol{m}$ and $\boldsymbol{u}$ parameters are updated through standard multinomial-Dirichlet distributions. For a particular m_{fl}, we have

$$\mathcal{L}(m_{fl}|\boldsymbol{\gamma}, \boldsymbol{u}, \boldsymbol{Z}, \boldsymbol{\eta}) \propto \prod_{i=1}^{n_A}\prod_{j=1}^{n_B} m_{fl}^{I(i\in Z_j)I(\gamma_{ij}^f=l)I_{obs}(\gamma_{ij}^f)} m_{fl}^{\alpha_{fl}-1} = m_{fl}^{\alpha_{fl}(\boldsymbol{Z})-1},$$

where $\alpha_{fl}(\boldsymbol{Z}) = \alpha_{fl} + \sum_{i=1}^{n_A}\sum_{j=1}^{n_B} I_{obs}(\gamma_{ij}^f)I(\gamma_{ij}^f = l)I(i \in Z_j)$. Analogous procedures lead to $\mathcal{L}(u_{fl}|\boldsymbol{\gamma}, \boldsymbol{m}, \boldsymbol{Z}, \boldsymbol{\eta}) \propto u_{fl}^{\beta_{fl}(\boldsymbol{Z})-1}$, where $\beta_{fl}(\boldsymbol{Z}) = \beta_{fl} + \sum_{i=1}^{n_A}\sum_{j=1}^{n_B} I_{obs}(\gamma_{ij}^f)I(\gamma_{ij}^f = l)I(i \notin Z_j)$. Thus, for the vectors of parameters $\boldsymbol{m}_f$ and $\boldsymbol{u}_f$, we have

$$\begin{aligned}
&\boldsymbol{m}_f|\boldsymbol{\gamma}, \boldsymbol{Z}, \boldsymbol{u}, \boldsymbol{\eta} \sim \text{Dirichlet}(\alpha_{f1}(\boldsymbol{Z}), \ldots, \alpha_{fL_f}(\boldsymbol{Z}),\\
&\boldsymbol{u}_f|\boldsymbol{\gamma}, \boldsymbol{Z}, \boldsymbol{m}, \boldsymbol{\eta} \sim \text{Dirichlet}(\beta_{f1}(\boldsymbol{Z}), \ldots, \beta_{fL_f}(\boldsymbol{Z})),\\
&\text{where } \alpha_{fl}(\boldsymbol{Z}) = \alpha_{fl} + \sum_{i=1}^{n_A}\sum_{j=1}^{n_B} I_{obs}(\gamma_{ij}^f)I(\gamma_{ij}^f = l)I(i \in Z_j), \qquad (15)\\
&\text{and } \beta_{fl}(\boldsymbol{Z}) = \beta_{fl} + \sum_{i=1}^{n_A}\sum_{j=1}^{n_B} I_{obs}(\gamma_{ij}^f)I(\gamma_{ij}^f = l)I(i \notin Z_j). \qquad (16)
\end{aligned}$$

Using the hashing techniques provided in Appendix 7.2 and likelihood as expressed in (11), we can perform these updates in a more efficient manner. Let $\boldsymbol{\alpha_0} = (\alpha_{11}, \ldots, \alpha_{FL_F})$ and $\boldsymbol{\beta_0} = (\beta_{11}, \ldots, \beta_{FL_F})$ be concatenated vectors of prior parameters for the $\boldsymbol{m}$ and $\boldsymbol{u}$ distributions respectively. The terms needed for the posterior updates for the $\boldsymbol{m}$ and $\boldsymbol{u}$ parameters are given by the appropriate components of the vectors

$$\boldsymbol{\alpha}(\boldsymbol{Z}) = \boldsymbol{\alpha_0} + \sum_{p=1}^{P} n_p(\boldsymbol{Z}) \times e(h_p), \qquad (17)$$

$$\boldsymbol{\beta}(\boldsymbol{Z}) = \boldsymbol{\beta_0} + \sum_{p=1}^{P} (N_p - n_p(\boldsymbol{Z})) \times e(h_p), \qquad (18)$$

where $n_p(\boldsymbol{Z}) = \sum_{i=1}^{n_A}\sum_{j=1}^{n_B} I(i \in Z_j)I(\gamma_{ij} = h_p)$ is the number of matching record pairs with agreement pattern h_p. Specifically, the $l + \sum_{k=1}^{f-1} L_k$ components of (17) and (18) provide the posterior updates for level l and field f in (15) and (16). These vectorized summations can be computed more efficiently than summing over $n_A n_B$ record pairs for each field and agreement level.

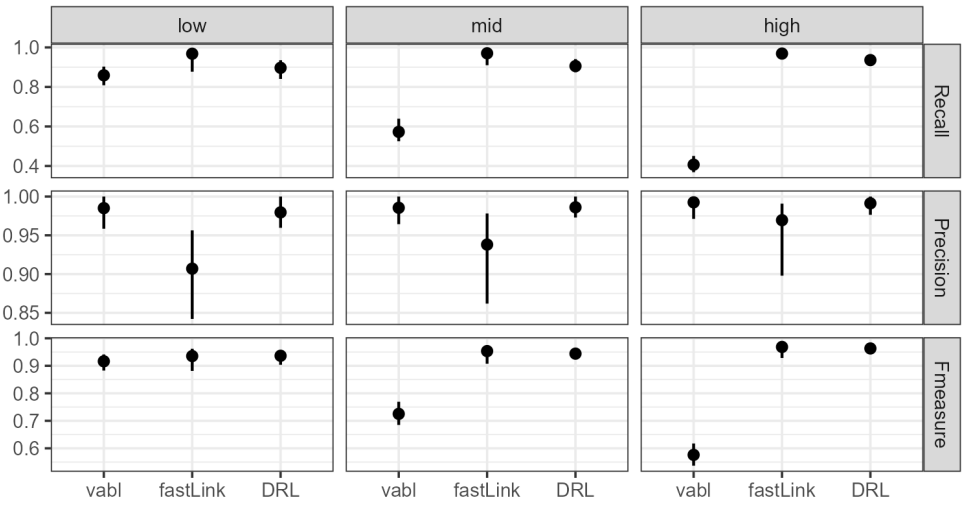

Figure 4: Results from simulation study of described in Appendix 7.4 for `vabl`, `fastLink`, and `DRL`.

7.4 Additional Simulation

We repeat the simulation from Section 4 with a larger proportion of the records in B having matches in A. Specifically, we increase the number of such records to 150, rather than 50. As shown in Figures 4 and 5, the precision for `multilink` without filtering and for `fastLink` is much better in this setting, but still is not competitive with `DRL`.

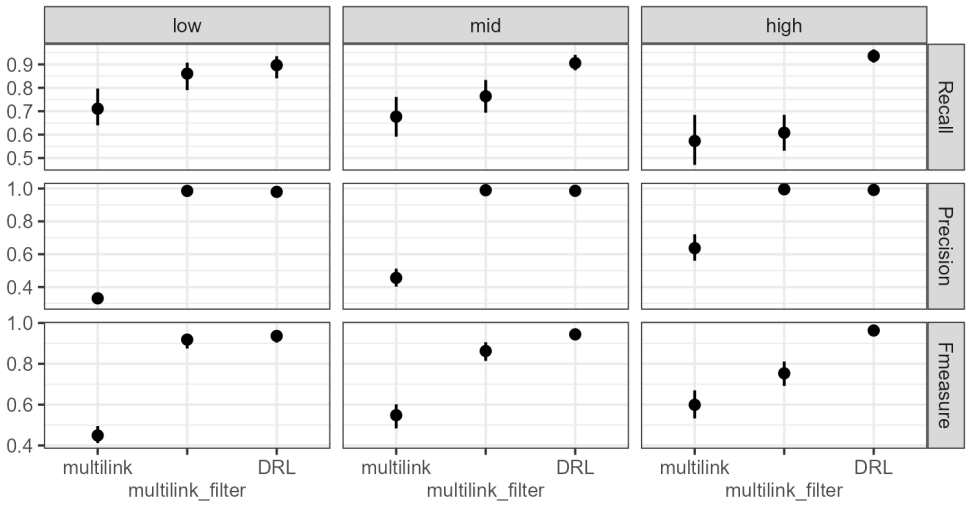

Figure 5: Results from simulation study of described in Appendix 7.4 for `multilink` (with and without filtering) and `DRL`.