Multifile Partitioning for Record Linkage and Duplicate Detection

Serge Aleshin-Guendel, Mauricio Sadinle*

Department of Biostatistics, University of Washington, Seattle, Washington, U.S.A.

October 11, 2021

Abstract

Merging datafiles containing information on overlapping sets of entities is a challenging task in the absence of unique identifiers, and is further complicated when some entities are duplicated in the datafiles. Most approaches to this problem have focused on linking two files assumed to be free of duplicates, or on detecting which records in a single file are duplicates. However, it is common in practice to encounter scenarios that fit somewhere in between or beyond these two settings. We propose a Bayesian approach for the general setting of multifile record linkage and duplicate detection. We use a novel partition representation to propose a structured prior for partitions that can incorporate prior information about the data collection processes of the datafiles in a flexible manner, and extend previous models for comparison data to accommodate the multifile setting. We also introduce a family of loss functions to derive Bayes estimates of partitions that allow uncertain portions of the partitions to be left unresolved. The performance of our proposed methodology is explored through extensive simulations. Code implementing the methodology is available at `https://github.com/aleshing/multilink`.

Keywords: data matching; data merging; entity resolution; microclustering.

*Serge Aleshin-Guendel is a Ph.D. student, Department of Biostatistics, University of Washington, Seattle, WA 98195 (e-mail: aleshing@uw.edu); and Mauricio Sadinle is an Assistant Professor, Department of Biostatistics, University of Washington, Seattle, WA 98195 (e-mail: msadinle@uw.edu). This research was supported by the NSF under grant SES-1852841 and by the NIH–NIDA under award R21DA051756. The authors thank Jorge A. Restrepo for providing the Colombian homicide data.

1 Introduction

When information on individuals is collected across multiple datafiles, it is natural to merge these to harness all available information. This merging requires identifying *coreferent* records, i.e., records that refer to the same entity, which is not trivial in the absence of unique identifiers. This problem arises in many fields, including public health (Hof et al. 2017), official statistics (Jaro 1989), political science (Enamorado et al. 2019), and human rights (Sadinle 2014, 2017, Ball & Price 2019).

Most approaches in this area have thus far focused on one of two settings. *Record linkage* has traditionally referred to the setting where the goal is to find coreferent records across two datafiles, where the files are assumed to be free of duplicates. *Duplicate detection* has traditionally referred to the setting where the goal is to find coreferent records within a single file. In practice, however, it is common to encounter problems that fit somewhere in between or beyond these two settings. For example, we could have multiple datafiles that are all assumed to be free of duplicates, or we might have duplicates in some files but not in others. In these general settings, the data collection processes for the different datafiles possibly introduce different patterns of duplication, measurement error, and missingness into the records. Further, dependencies among these data collection processes determine which specific subsets of files contain records of the same entity. We refer to this general setting as *multifile record linkage and duplicate detection*.

Traditional approaches to record linkage and duplicate detection have mainly followed the seminal work of Fellegi & Sunter (1969), by modeling comparisons of fields between pairs of records in a mixture model framework (Winkler 1994, Jaro 1989, Larsen & Rubin 2001). These approaches work under, and take advantage of, the intuitive assumption that coreferent records will look similar, and non-coreferent records will look dissimilar. However, these approaches output independent decisions for the coreference status of each pair of records, necessitating the use of ad hoc post-processing steps to reconcile incompatible decisions that ignore the logical constraints of the problem.

Our approach to multifile record linkage and duplicate detection builds on previous Bayesian approaches where the parameter of interest is defined as a partition of the records. These Bayesian approaches have been carried out in two frameworks. In the *direct-modeling* framework, one directly models the fields of information contained in the records (Matsakis 2010, Tancredi & Liseo 2011, Liseo & Tancredi 2011, Steorts

2015, Steorts et al. 2016, Tancredi et al. 2020, Marchant et al. 2021, Enamorado & Steorts 2020), which requires a custom model for each type of field. While this framework can provide a plausible generative model for the records, it can be difficult to develop custom models for complicated fields like strings, so most approaches are limited to modeling categorical data, with some exceptions (Liseo & Tancredi 2011, Steorts 2015). In the *comparison-based* framework, following the traditional approaches, one models comparisons of fields between pairs of records (Fortini et al. 2001, Larsen 2005, Sadinle 2014, 2017). By modeling comparisons of fields, instead of the fields directly, a generic modeling approach can be taken for any field type, as long as there is a meaningful measure of similarity for that field type.

Sadinle & Fienberg (2013) generalized Fellegi & Sunter (1969) by linking $K > 2$ files with no duplicates. However, in addition to inheriting the issues of traditional approaches, their approach does not scale well in the number of files or the file sizes encountered in practice. Steorts et al. (2016) presented a Bayesian approach in the direct-modeling framework for the general setting of multifile record linkage and duplicate detection, which has been extended by Steorts (2015) and Marchant et al. (2021). This approach uses a flat prior on arbitrary labels of partitions, which incorporates unintended prior information.

In light of the shortcomings of existing approaches, we propose an extension of Bayesian comparison-based models that explicitly handles the setting of multifile record linkage and duplicate detection. We first present in Section 2 a parameterization of partitions specific to the context of multifile record linkage and duplicate detection. Building on this parameterization, in Section 3 we construct a structured prior for partitions that can incorporate prior information about the data collection processes of the files in a flexible manner. As a by-product, a family of priors for K-partite matchings is constructed. In Section 4 we construct a likelihood function for comparisons of fields between pairs of records that accommodates possible differences in the datafile collection processes. In Section 5 we present a family of loss functions that we use to derive Bayes estimates of partitions. These loss functions have an *abstain option* which allow portions of the partition with large amounts of uncertainty to be left unresolved. Finally, we explore the performance of our proposed methodology through simulation studies in Section 6. In the Appendix we present an application of our proposed approach to link three Colombian homicide record systems.

2 Multifile Partitioning

Consider K files $\boldsymbol{X}_1, \cdots, \boldsymbol{X}_K$, each containing information on possibly overlapping subsets of a population of entities. The goal of multifile record linkage and duplicate detection is to identify the sets of records in $\boldsymbol{X}_1$, $\cdots$, $\boldsymbol{X}_K$ that are coreferent, as illustrated in Figure 1. Identifying coreferent records across datafiles represents the goal of record linkage, and identifying coreferent records within each file represents the goal of duplicate detection.

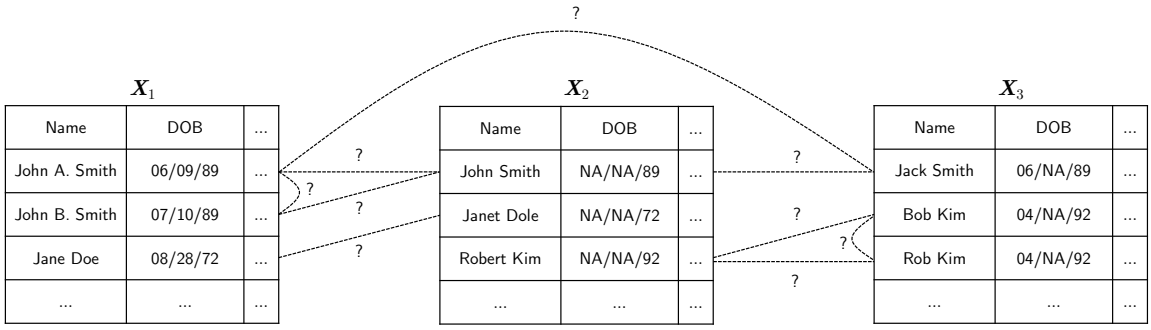


Figure 1: A toy example of the multifile record linkage and duplicate detection problem.

We denote the number of records contained in datafile $\boldsymbol{X}_k$ as r_k, and the total number of records across all files as $r = \sum_{k=1}^{K} r_k$. We label the records in all datafiles in a consecutive order, that is, those in $\boldsymbol{X}_1$ as $R_1 = (1, \cdots, r_1)$, those in $\boldsymbol{X}_2$ as $R_2 = (r_1 + 1, \cdots, r_1 + r_2)$, and so on, finally labeling the records in $\boldsymbol{X}_K$ as $R_K = (\sum_{k=1}^{K-1} r_k + 1, \cdots, r)$. We denote $[r] = (1, \cdots, r)$, where it is clear that $[r] = (R_1, \cdots, R_K)$, which represents all the records coming from all datafiles.

Formally, multifile record linkage and duplicate detection is a partitioning problem. A partition of a set is a collection of disjoint subsets, called clusters, whose union is the original set. In this context, the term *coreference partition* refers to a partition $\mathcal{C}$ of all the records in $\boldsymbol{X}_1$, $\cdots$, $\boldsymbol{X}_K$, or equivalently a partition $\mathcal{C}$ of $[r]$, such that each cluster $c \in \mathcal{C}$ is exclusively composed of all the records generated by a single entity (Matsakis 2010, Sadinle 2014). This implies that there is a one-to-one correspondence between the clusters in $\mathcal{C}$ and the entities represented in at least one of the datafiles. Estimating $\mathcal{C}$ is the goal of multifile record linkage and duplicate detection.

2.1 Multifile Coreference Partitions

In the setting of multifile record linkage and duplicate detection, the datafiles are the product of K data collection processes, which possibly introduce different patterns of duplication, measurement error, and missingness. This indicates that records coming from different datafiles should be treated differently. To take this into account, we introduce the concept of a *multifile coreference partition* by endowing a coreference partition $\mathcal{C}$ with additional structure to preserve the information on where records come from. Each cluster $c \in \mathcal{C}$ can be decomposed as $c = c^1 \cup \cdots \cup c^k \cup \cdots \cup c^K$, where c^k is the subset of records in cluster c that belong to datafile $\boldsymbol{X}_k$, which leads us to the following definition.

Definition 1. *The* multifile coreference partition *of datafiles $\boldsymbol{X}_1, \cdots, \boldsymbol{X}_K$ is obtained from the coreference partition $\mathcal{C}$ by expressing each cluster $c \in \mathcal{C}$ as a K-tuple $(c^1, \cdots, c^K)$, where c^k represents the records of c that come from datafile $\boldsymbol{X}_k$.*

For simplicity we will continue using the notation $\mathcal{C}$ to denote a multifile coreference partition, although technically this new structure is richer and therefore different from a coreference partition that does not preserve the datafile membership of the records. The multifile representation of partitions is useful for decoupling the features that are important for within-file duplicate detection or for across-files record linkage.

For duplicate detection, the goal is to identify coreferent records within each datafile. This can be phrased as estimating the *within-file coreference partition* $\mathcal{C}_k$ of each datafile $\boldsymbol{X}_k$. Clearly, these $\mathcal{C}_k$ can be obtained from the multifile partition $\mathcal{C}$ by extracting the kth entry of each cluster $c = (c^1, \ldots, c^K) \in \mathcal{C}$. Two useful summaries of a given within-file partition $\mathcal{C}_k$ are the number of within-file clusters $n_k = |\mathcal{C}_k|$, which is the number of unique entities represented in datafile $\boldsymbol{X}_k$, and the within-file cluster sizes $\boldsymbol{d}^k = \{|c^k| : c^k \in \mathcal{C}_k\}$, which represent the number of records associated with each entity in datafile $\boldsymbol{X}_k$.

On the other hand, in record linkage the goal is to identify coreferent records across datafiles. Given the within-file partitions, $\mathcal{C}_1, \cdots, \mathcal{C}_K$, the goal can be phrased as identifying which clusters across these partitions represent the same entities. This across-datafiles structure can be formally represented by a K-*partite matching*. Given K sets $V_1, \cdots, V_K$, a K-partite matching $\mathcal{M}$ is a collection of subsets from $\cup_{k=1}^K V_k$ such that each $m \in \mathcal{M}$ contains maximum one element from each V_k. If we think of each V_k as the set of clusters $\mathcal{C}_k$ representing the entities in datafile $\boldsymbol{X}_k$, then it is clear that the goal is to identify the K-partite

matching $\mathcal{M}$ that puts together the clusters that refer to the same entities across datafiles. This structure can be extracted from a multifile coreference partition $\mathcal{C}$, given that each element $c = (c^1, \ldots, c^K) \in \mathcal{C}$ contains the coreferent clusters across all within-file partitions. Indeed, a multifile coreference partition can be thought of as a K-partite matching of within-file coreference partitions.

A useful summary of the across-datafile structure is the amount of entity-overlap between datafiles, represented by the number of clusters $c = (c^1, \ldots, c^K) \in \mathcal{C}$ with records in specific subsets of the files. We can concisely summarize the entity-overlap of the datafiles through a contingency table. In particular, consider a 2^K contingency table with cells indexed by $\boldsymbol{h} \in \{0,1\}^K$ and corresponding cell counts $n_{\boldsymbol{h}}$. Here, $\boldsymbol{h}$ represents a pattern of inclusion of an entity in the datafiles, where a 1 indicates inclusion and a 0 exclusion. For instance, if $K = 3$, n_{011} is the number of clusters $c = (c^1, c^2, c^3) \in \mathcal{C}$ representing entities with records in datafiles 2 and 3 but without records in datafile 1. We let $\mathcal{H} = \{0,1\}^K \setminus \{0\}^K$ and denote the (incomplete) contingency table of counts as $\boldsymbol{n} = \{n_{\boldsymbol{h}}\}_{\boldsymbol{h} \in \mathcal{H}}$, which we refer to as the *overlap table*. We ignore the cell $\{0\}^K$ which would represent entities that are not recorded in any of the K files. This cell is not of interest in this article, although it is the parameter of interest in population size estimation (see e.g. Bird & King 2018).

Example. To illustrate the concept of a multifile partition, consider two files with five and seven records respectively, so that $\boldsymbol{X}_1$ contains records $1-5$ and $\boldsymbol{X}_2$ contains records $6-12$. Suppose the coreference partition is $\{\{1,9\},\{2\},\{3,8,10,11\},\{4,5,7\},\{6\},\{12\}\}$. The corresponding multifile partition is $\mathcal{C} = \{(\{1\},\{9\}),(\{2\},\emptyset),(\{3\},\{8,10,11\}),(\{4,5\},\{7\}),(\emptyset,\{6\}),(\emptyset,\{12\})\}$. As illustrated in Figure 2, the within-file partitions can be extracted as $\mathcal{C}_1 = \{\{1\},\{2\},\{3\},\{4,5\}\}$ and $\mathcal{C}_2 = \{\{6\},\{7\},\{9\},\{8,10,11\},\{12\}\}$, and the within-file cluster sizes are $\boldsymbol{d}^1 = (1,1,1,2)$ and $\boldsymbol{d}^2 = (1,1,1,3,1)$. The overlap table in this case is $\{n_{11}, n_{10}, n_{01}\}$, indicating that $n_{11} = 3$ entities are represented in both datafiles, $n_{10} = 1$ entity is represented only in the first datafile, and $n_{01} = 2$ entities are represented only in the second datafile. In total, there are $n_1 = |\mathcal{C}_1| = n_{11} + n_{10} = 4$ unique entities represented in $\boldsymbol{X}_1$, $n_2 = |\mathcal{C}_2| = n_{11} + n_{01} = 5$ unique entities represented in $\boldsymbol{X}_2$, and $n = |\mathcal{C}| = n_{11} + n_{10} + n_{01} = 6$ entities among both datafiles.

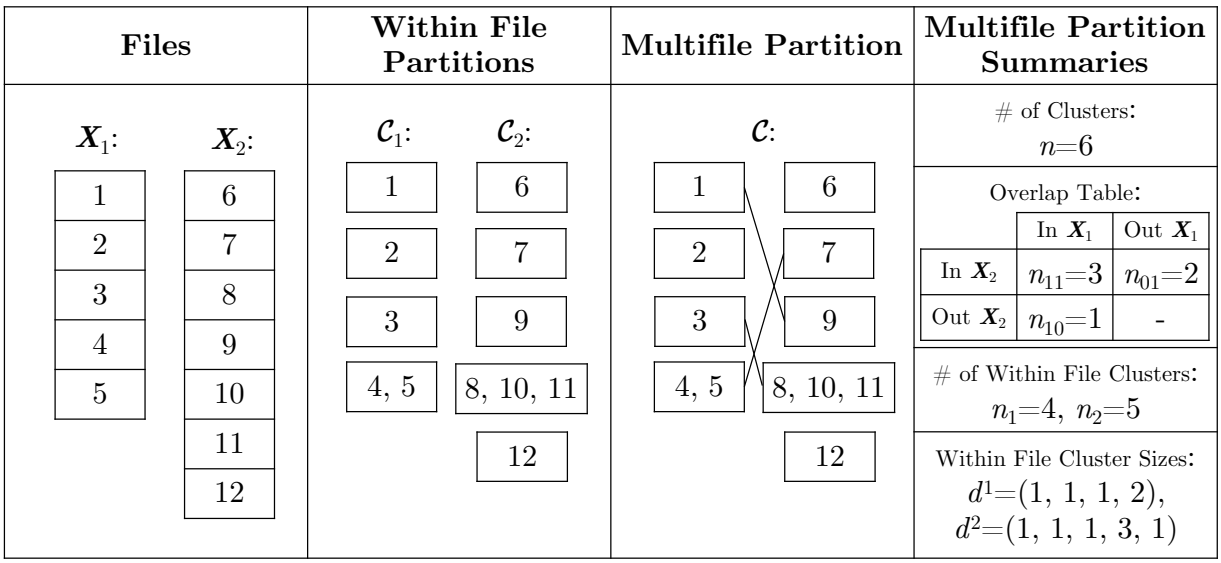


Figure 2: An illustration of a multifile partition of [12], where $\boldsymbol{X}_1$ contains records $1-5$ and $\boldsymbol{X}_2$ contains records $6-12$.

3 A Structured Prior for Multifile Partitions

Bayesian approaches to multifile record linkage and duplicate detection require prior distributions on multifile coreference partitions. We present a generative process for multifile partitions, building on our representation introduced in Section 2.1. The idea is to generate a multifile partition by first generating summaries that characterize it, as follows:

1. Generate the number of unique entities n represented in the datafiles, which also corresponds to the number of clusters of the multifile partition.
2. Given n, generate an overlap table $\boldsymbol{n} = \{n_{\boldsymbol{h}}\}_{\boldsymbol{h}\in\mathcal{H}}$ so that $n = \sum_{\boldsymbol{h}\in\mathcal{H}} n_{\boldsymbol{h}}$, where $\mathcal{H} = \{0,1\}^K \setminus \{0\}^K$. From $\boldsymbol{n}$ we can derive the number of entities in datafile $\boldsymbol{X}_k$ as $n_k = \sum_{\boldsymbol{h}\in\mathcal{H}} h_k n_{\boldsymbol{h}}$, where h_k is the kth entry of $\boldsymbol{h}$.
3. For each $k = 1, \cdots, K$, given n_k, independently generate a set of counts $\boldsymbol{d}^k = \{d_i^k\}_{i=1}^{n_k}$, representing the number of records associated with each entity in file $\boldsymbol{X}_k$. From $\boldsymbol{d}^k$, we can derive the number of records in file $\boldsymbol{X}_k$ as $r'_k = \sum_{i=1}^{n_k} d_i^k$. Index the r'_k records as $R'_k = (\sum_{l=1}^{k-1} r'_l + 1, \cdots, \sum_{l=1}^{k} r'_l)$.
4. For each $k = 1, \cdots, K$, given $\boldsymbol{d}^k$, induce a within-file partition $\mathcal{C}_k$ by randomly allocating R'_k into n_k clusters of sizes $d_1^k, \cdots, d_{n_k}^k$.
5. Given the overlap table $\boldsymbol{n}$ and within-file partitions $\{\mathcal{C}_1, \cdots, \mathcal{C}_K\}$, generate a K-partite matching of

the within-file partitions by selecting uniformly at random from the set of all K-partite matchings with overlap table $\boldsymbol{n}$. By definition, the result is a multifile coreference partition.

By parameterizing each step of this generative process, we can construct a prior distribution for multifile partitions, as we now show.

3.1 Parameterizing the Generative Process

Prior for the Number of Entities or Clusters. In the absence of substantial prior information, we follow a simple choice for the prior on the number of clusters, by taking a uniform distribution over the integers less than some upper bound, U, i.e. $\mathbb{P}(n) = U^{-1} I(n \in \{1, \cdots, U\})$. In practice, we set U to be the actual number of records across all datafiles r, which is observed. More informative specifications are discussed in Appendix A.

Prior for the Overlap Table. Conditional on n, we use a Dirichlet-multinomial distribution as our prior on the overlap table $\boldsymbol{n} = \{n_{\boldsymbol{h}}\}_{\boldsymbol{h} \in \mathcal{H}}$. Given a collection of positive hyperparameters for each cell of the overlap table, $\boldsymbol{\alpha} = \{\alpha_{\boldsymbol{h}}\}_{\boldsymbol{h} \in \mathcal{H}}$, and letting $\alpha_0 = \sum_{\boldsymbol{h} \in \mathcal{H}} \alpha_{\boldsymbol{h}}$, the prior for the overlap table under this choice is $\mathbb{P}(\boldsymbol{n} \mid n) = [(n!)\Gamma(\alpha_0)/\Gamma(n + \alpha_0)] \prod_{\boldsymbol{h} \in \mathcal{H}} [\Gamma(n_{\boldsymbol{h}} + \alpha_{\boldsymbol{h}})/(n_{\boldsymbol{h}}!)\Gamma(\alpha_{\boldsymbol{h}})]$. Due to conjugacy, $\boldsymbol{\alpha}$ can be interpreted as prior cell counts, which can be used to incorporate prior information about the overlap between datafiles. In the absence of substantial prior information, when the number of files is not too large and the overlap table is not expected to be sparse, we recommend setting $\boldsymbol{\alpha} = (1, \cdots, 1)$. In Appendix A we discuss alternative specifications when the overlap table is expected to be sparse.

Prior for the Within-File Cluster Sizes. Given the number of entities in datafile $\boldsymbol{X}_k$, $n_k = \sum_{\boldsymbol{h} \in \mathcal{H}} h_k n_{\boldsymbol{h}}$, we generate the within-file cluster sizes $\boldsymbol{d}^k = \{d_i^k\}_{i=1}^{n_k}$ assuming that $d_1^k, \cdots, d_{n_k}^k \mid n_k \overset{iid}{\sim} p_k(\cdot)$. Here $p_k(\cdot)$ represents the probability mass function of a distribution on the positive integers, so that $\mathbb{P}(\boldsymbol{d}^k \mid n_k) = \prod_{i=1}^{n_k} p_k(d_i^k)$. We do not expect a-priori many duplicates per entity, and therefore we expect the counts in $\boldsymbol{d}^k$ to be mostly ones or to be very small (Miller et al. 2015, Zanella et al. 2016). We therefore use a similar approach to Klami & Jitta (2016), and use distributions truncated to the range $\{1, \cdots, U^k\}$, where U^k is a file-specific upper bound on cluster sizes. We further use distributions where prior mass is concentrated at small values. A default specification is to use a Poisson distribution with mean 1 truncated to $\{1, \cdots, U^k\}$,

i.e. $p_k(d_i^k) \propto (d_i^k!)^{-1} I(d_i^k \in \{1, \cdots, U^k\})$. More informative options could be used for $p_k(\cdot)$ by using any distribution on $\{1, \cdots, U^k\}$, where this could vary from file to file if some files were known to have more or less duplication.

Prior for the Within-File Partitions. Given the within-file cluster sizes $\boldsymbol{d}^k$, the number of ways of assigning $d_1^k, \cdots, d_{n_k}^k$ records to clusters $1, \cdots, n_k$, respectively, is given by the multinomial coefficient $r_k'! / \prod_{i=1}^{n_k} d_i^k!$, with $r_k' = \sum_{i=1}^{n_k} d_i^k$. However, the ordering of the clusters is irrelevant for constructing the within-file partition $\mathcal{C}_k$ of R_k'. There are $n_k!$ ways of ordering the n_k clusters of $\mathcal{C}_k$, which leads to $r_k'! / (n_k! \prod_{i=1}^{n_k} d_i^k!)$ partitions of R_k' into clusters of sizes $d_1^k, \cdots, d_{n_k}^k$. We then have $\mathbb{P}(\mathcal{C}_k \mid \boldsymbol{d}^k) = (n_k! / r_k'!) \prod_{i=1}^{n_k} d_i^k!$.

Prior for the K-Partite Matching. Given the overlap table $\boldsymbol{n}$ and the within-file partitions $\{\mathcal{C}_1, \cdots, \mathcal{C}_K\}$, our prior over K-partite matchings of the within-file partitions is uniform. Thus we just need to count the number of K-partite matchings with overlap table $\boldsymbol{n}$. This is taken care of by Proposition 1, proven in Appendix A.

Proposition 1. *The number of K-partite matchings that have the same overlap table, $\boldsymbol{n} = \{n_{\boldsymbol{h}}\}_{\boldsymbol{h} \in \mathcal{H}}$, is $\prod_{k=1}^K n_k! / \prod_{\boldsymbol{h} \in \mathcal{H}} n_{\boldsymbol{h}}!$. Thus $\mathbb{P}(\mathcal{C} \mid \{\mathcal{C}_k\}_{k=1}^K, \boldsymbol{n}) = \prod_{\boldsymbol{h} \in \mathcal{H}} n_{\boldsymbol{h}}! / \prod_{k=1}^K n_k!$.*

The Structured Prior for Multifile Partitions. Letting quantities followed by $(\mathcal{C})$ mean they are computable from $\mathcal{C}$, the density of our structured prior for multifile partitions is

$$\begin{aligned}
\mathbb{P}(\mathcal{C}) &= \mathbb{P}(n)\mathbb{P}(\boldsymbol{n} \mid n) \prod_{k=1}^K \left[\mathbb{P}(\boldsymbol{d}^k \mid n_k)\mathbb{P}(\mathcal{C}_k \mid \boldsymbol{d}^k)\right] \mathbb{P}(\mathcal{C} \mid \{\mathcal{C}_k\}_{k=1}^K, \boldsymbol{n}) \\
&= \mathbb{P}(n(\mathcal{C})) \frac{n(\mathcal{C})!\, \Gamma(\alpha_0)}{\Gamma(n(\mathcal{C}) + \alpha_0)} \prod_{\boldsymbol{h} \in \mathcal{H}} \left[\frac{\Gamma(n_{\boldsymbol{h}}(\mathcal{C}) + \alpha_{\boldsymbol{h}})}{\Gamma(\alpha_{\boldsymbol{h}})}\right] \prod_{k=1}^K \left[\frac{1}{r_k'(\mathcal{C})!} \prod_{c_k \in \mathcal{C}_k} [|c_k|!\; p_k(|c_k|)]\right]. \qquad (1)
\end{aligned}$$

3.2 Comments and Related Literature

The structured prior for multifile partitions allows us to incorporate prior information about the total number of clusters, the overlap between files, and the amount of duplication in each file. If we restrict the prior for the within-file cluster sizes to be $p_k(d_i^k) = I(d_i^k = 1)$ for a given datafile $\boldsymbol{X}_k$, then we enforce the assumption that there are no duplicates in that file. Imposing this restriction for all datafiles leads to the special case of a prior for K-partite matchings, which is of independent interest, as we are not aware of any such constructions outside of the bipartite case (Fortini et al. 2001, Larsen 2005, Sadinle 2017).

Our prior construction, where priors are first placed on interpretable summaries of a partition and then a uniform prior is placed on partitions which have those summaries, mimics the construction of the priors on bipartite matchings of Fortini et al. (2001), Larsen (2005) and Sadinle (2017), and the Kolchin and allelic partition priors of Zanella et al. (2016) and Betancourt, Sosa & Rodríguez (2020). While the Kolchin and allelic partition priors could both be used as priors for multifile partitions, these do not incorporate the datafile membership of records. Using these priors in the multifile setting would imply that the sizes of clusters containing records from only one file have the same prior distribution as the sizes of clusters containing records from two files, which should not be true in general.

Miller et al. (2015) and Zanella et al. (2016) proposed the *microclustering property* as a desirable requirement for partition priors in the context of duplicate detection: denoting the size of the largest cluster in a partition of $[r]$ by M_r, a prior satisfies the microclustering property if $M_r/r \to 0$ in probability as $r \to \infty$. A downside of priors with this property is that they can still allow the size of the largest cluster to go to ∞ as r increases. For this reason Betancourt, Sosa & Rodríguez (2020) introduced the stronger *bounded microclustering property*, which we believe is more practically important: for any r, M_r is finite with probability 1. Our prior satisfies the bounded microclustering property as $M_r \leq \sum_{k=1}^{K} U^k$.

While our parameter of interest is a partition $\mathcal{C}$ of r records, the prior developed in this section is a prior for a partition of a random number of records. In practice we condition on the file sizes, $\{r_k\}_{k=1}^K$, and use the prior $\mathbb{P}(\mathcal{C} \mid \{r_k\}_{k=1}^K) \propto \mathbb{P}(\mathcal{C}) I(r'_k(\mathcal{C}) = r_k(\mathcal{C}) \text{ for all } k)$, which alters the interpretation of the prior. A similar problem occurs for the Kolchin partition priors of Zanella et al. (2016). This motivated the exchangeable sequences of clusters priors of Betancourt, Zanella & Steorts (2020), which are similar to Kolchin partition priors, but lead to a directly interpretable prior specification. It would be interesting in future work to see if an analogous prior could be developed for our structured prior for multifile partitions. Despite this limitation, we demonstrate in simulations in Section 6 that incorporating strong prior information into our structured prior for multifile partitions can lead to improved frequentist performance over a default specification.

4 A Model for Comparison Data

We now introduce a comparison-based modeling approach to multifile record linkage and duplicate detection, building on the work of Fellegi & Sunter (1969), Jaro (1989), Winkler (1994), Larsen & Rubin (2001), Fortini et al. (2001), Larsen (2005) and Sadinle (2014, 2017). Working under the intuitive assumption that coreferent records will look similar, and non-coreferent records will look dissimilar, these approaches construct statistical models for comparisons computed between each pair of records.

There are two implications of the multifile setting described in Section 2 that are important to consider when constructing a model for the comparison data. First, models for the comparison data should account for the fact that the distribution of the comparisons between record pairs might potentially change across different pairs of files. For example, if files $\boldsymbol{X}_k$ and $\boldsymbol{X}_{k'}$ are not accurate, whereas files $\boldsymbol{X}_q$ and $\boldsymbol{X}_{q'}$ are, then the distribution of comparisons between $\boldsymbol{X}_k$ and $\boldsymbol{X}_{k'}$ will look very different compared with the distribution of comparisons between $\boldsymbol{X}_q$ and $\boldsymbol{X}_{q'}$. Second, the fields available for comparison will vary across pairs of files. For example, files $\boldsymbol{X}_k$ and $\boldsymbol{X}_{k'}$ may have collected information on a field that file $\boldsymbol{X}_q$ did not. In this scenario, we would like a model that is able to utilize this extra field when linking $\boldsymbol{X}_k$ and $\boldsymbol{X}_{k'}$, even though it is not available in $\boldsymbol{X}_q$. In this section we introduce a Bayesian comparison-based model that explicitly handles the multifile setting by constructing a likelihood function that models comparisons of fields between different pairs of files separately. The separate models are able to adapt to the level of noise of each file pair, and the maximal number of fields are able to be compared for each file pair.

4.1 Comparison Data

We construct comparison vectors for pairs of records to provide evidence for whether they correspond to the same entity. For $k \leq k'$, let $\mathcal{P}_{kk'} = \{(i,j) : i < j, i \in \boldsymbol{X}_k, j \in \boldsymbol{X}_{k'}\}$ denote the set of all record pairs between files $\boldsymbol{X}_k$ and $\boldsymbol{X}_{k'}$, and let F be the total number of different fields available from the K files. For each file pair (k,k'), $k \leq k'$, and record pair $(i,j) \in \mathcal{P}_{kk'}$, we compare each field $f = 1, \ldots, F$ using a similarity measure $\mathcal{S}_f(i,j)$, which will depend on the data type of field f. For unstructured categorical fields such as race, $\mathcal{S}_f$ can be a binary comparison which checks for agreement. For more structured fields containing strings or numbers, $\mathcal{S}_f$ should be able to capture partial agreements. For example, string fields can be compared using

a string metric like the Levenshtein edit distance (see e.g. Bilenko et al. 2003), and numeric fields can be compared using absolute differences. Comparison $\mathcal{S}_f(i,j)$ will be missing if field f is not recorded in record i or record j, which includes the case where field f is not recorded in datafiles $\boldsymbol{X}_k$ or $\boldsymbol{X}_{k'}$.

While we could directly model the similarity measures $\mathcal{S}_f(i,j)$, this would require a custom model for each type of comparison, which inherits similar problems to the direct modeling of the fields themselves. Instead, we follow Winkler (1990) and Sadinle (2014, 2017) in dividing the range of $\mathcal{S}_f$ into $L_f + 1$ intervals $I_{f0}, I_{f1}, \cdots, I_{fL_f}$ that represent varying levels of agreement, with I_{f0} representing the highest level of agreement, and I_{fL_f} representing the lowest level of agreement. We then let $\gamma_{ij}^f = l$ if $\mathcal{S}_f(i,j) \in I_{fl}$, where larger values of γ_{ij}^f represent larger disagreements between records i and j in field f. Finally, we form the comparison vector $\boldsymbol{\gamma}_{ij} = (\gamma_{ij}^1, \cdots, \gamma_{ij}^F)$. Constructing the comparison data this way allows us to build a generic modeling approach. In particular, extending Fortini et al. (2001), Larsen (2005) and Sadinle (2014, 2017), our model for the comparison data is

$$\boldsymbol{\gamma}_{ij} \mid \mathcal{C}(i) = \mathcal{C}(j), (i,j) \in \mathcal{P}_{kk'} \overset{iid}{\sim} \mathsf{M}_{kk'}(\boldsymbol{m}_{kk'}),$$
$$\boldsymbol{\gamma}_{ij} \mid \mathcal{C}(i) \neq \mathcal{C}(j), (i,j) \in \mathcal{P}_{kk'} \overset{iid}{\sim} \mathsf{U}_{kk'}(\boldsymbol{u}_{kk'}),$$

where $\mathcal{C}$ is a multifile partition, $\mathcal{C}(i)$ denotes record i's cluster in $\mathcal{C}$, $\mathcal{C}(i) = \mathcal{C}(j)$ indicates that records i and j are coreferent, $\mathsf{M}_{kk'}(\boldsymbol{m}_{kk'})$ is a model for the comparison data among coreferent record pairs from the file pair $\boldsymbol{X}_k$ and $\boldsymbol{X}_{k'}$, $\mathsf{U}_{kk'}(\boldsymbol{u}_{kk'})$ is a model for the comparison data among non-coreferent record pairs from datafile pair $\boldsymbol{X}_k$ and $\boldsymbol{X}_{k'}$, and $\boldsymbol{m}_{kk'}$ and $\boldsymbol{u}_{kk'}$ are vectors of parameters.

In the next section we make two further assumptions that simplify the model parameterization. Before doing so, we note a few limitations of our comparison-based model. First, computing comparison vectors scales quadratically in the number of records. Second, comparison vectors for different record pairs are not actually independent conditional on the partition (see Section 2 of Tancredi & Liseo 2011). Third, modeling discretized comparisons of record fields represents a loss of information. While the first limitation is computational and unavoidable in the absence of blocking (see Appendix B), the other two limitations are inferential. Despite these limitations, we find in Section 6 that the combination of our structured prior for multifile partitions and our comparison-based model can produce linkage estimates with satisfactory frequentist performance.

4.2 Conditional Independence and Missing Data

Under the assumptions that the fields in the comparison vectors are conditionally independent given the multifile partition of the records and that missing comparisons are ignorable (Sadinle 2014, 2017), the likelihood of the observed comparison data, $\boldsymbol{\gamma}^{obs}$, becomes

$$\mathcal{L}(\mathcal{C}, \Phi \mid \boldsymbol{\gamma}^{obs}) = \prod_{k \leq k'} \prod_{f=1}^{F} \prod_{l=0}^{L_f} (m_{kk'}^{fl})^{a_{kk'}^{fl}(\mathcal{C})} (u_{kk'}^{fl})^{b_{kk'}^{fl}(\mathcal{C})}. \tag{2}$$

Here $m_{kk'}^{fl} = \mathbb{P}(\gamma_{ij}^f = l \mid \mathcal{C}(i) = \mathcal{C}(j), (i,j) \in \mathcal{P}_{kk'})$, $u_{kk'}^{fl} = \mathbb{P}(\gamma_{ij}^f = l \mid \mathcal{C}(i) \neq \mathcal{C}(j), (i,j) \in \mathcal{P}_{kk'})$, $a_{kk'}^{fl}(\mathcal{C}) = \sum_{(i,j) \in \mathcal{P}_{kk'}} I_{obs}(\gamma_{ij}^f) I(\gamma_{ij}^f = l) I(\mathcal{C}(i) = \mathcal{C}(j))$, $b_{kk'}^{fl}(\mathcal{C}) = \sum_{(i,j) \in \mathcal{P}_{kk'}} I_{obs}(\gamma_{ij}^f) I(\gamma_{ij}^f = l) I(\mathcal{C}(i) \neq \mathcal{C}(j))$, $I_{obs}(\cdot)$ is an indicator of whether its argument was observed, and $\Phi = (\boldsymbol{m}, \boldsymbol{u})$ where $\boldsymbol{m}$ collects all of the $\boldsymbol{m}_{kk'}^f = (m_{kk'}^{f0}, \cdots, m_{kk'}^{fL_f})$ and $\boldsymbol{u}$ collects all of the $\boldsymbol{u}_{kk'}^f = (u_{kk'}^{f0}, \cdots, u_{kk'}^{fL_f})$. For a given multifile partition $\mathcal{C}$, $a_{kk'}^{fl}(\mathcal{C})$ represents the number of record pairs in $\mathcal{P}_{kk'}$ that belong to the same cluster with observed agreement at level l in field f, and $b_{kk'}^{fl}(\mathcal{C})$ represents the number of record pairs in $\mathcal{P}_{kk'}$ that do not belong to the same cluster with observed agreement at level l in field f.

5 Bayesian Estimation of Multifile Partitions

Bayesian estimation of the multifile coreference partition $\mathcal{C}$ is based on the posterior distribution $p(\mathcal{C}, \Phi \mid \boldsymbol{\gamma}^{obs}) \propto \mathbb{P}(\mathcal{C}) p(\Phi) \mathcal{L}(\mathcal{C}, \Phi \mid \boldsymbol{\gamma}^{obs})$, where $\mathbb{P}(\mathcal{C})$ is our structured prior for multifile partitions (1), $\mathcal{L}(\mathcal{C}, \Phi \mid \boldsymbol{\gamma}^{obs})$ is the likelihood from our model for comparison data (2), and $p(\Phi)$ represents a prior distribution for the $\Phi = (\boldsymbol{m}, \boldsymbol{u})$ model parameters. We now specify this prior $p(\Phi)$, outline a Gibbs sampler to sample from $p(\mathcal{C}, \Phi \mid \boldsymbol{\gamma}^{obs})$, and present a strategy to obtain point estimates of the multifile partition $\mathcal{C}$.

5.1 Priors for $\boldsymbol{m}$ and $\boldsymbol{u}$

We will use independent, conditionally conjugate priors for $\boldsymbol{m}_{kk'}^f$ and $\boldsymbol{u}_{kk'}^f$, namely $\boldsymbol{m}_{kk'}^f \sim \mathsf{Dirichlet}(\mu_{kk'}^{f0}, \cdots, \mu_{kk'}^{fL_f})$ and $\boldsymbol{u}_{kk'}^f \sim \mathsf{Dirichlet}(\nu_{kk'}^{f0}, \cdots, \nu_{kk'}^{fL_f})$. In this article we will use a default specification of $(\mu_{kk'}^{f0}, \cdots, \mu_{kk'}^{fL_f}) = (\nu_{kk'}^{f0}, \cdots, \nu_{kk'}^{fL_f}) = (1, \cdots, 1)$. We believe this prior specification is sensible for the $\boldsymbol{u}$ parameters, following the discussion in Section 3.2 of Sadinle (2014), as comparisons amongst non-coreferent records are likely to be highly variable and it is more likely than not that eliciting meaningful priors for them is too difficult. For

the $\boldsymbol{m}$ parameters, it might be desirable in certain applications to introduce more information into the prior. For example, one could set $\mu_{kk'}^{f0} > \cdots > \mu_{kk'}^{fL_f}$ to incorporate the prior belief that higher levels of agreement should have larger prior probability than lower levels of agreement. Another route would be to use the sequential parameterization of the $\boldsymbol{m}$ parameters, and the associated prior recommendations, described in Sadinle (2014).

5.2 Posterior Sampling

In Appendix B we outline a Gibbs sampler that produces a sequence of samples $\{\mathcal{C}^{[t]}, \Phi^{[t]}\}_{t=1}^{T}$ from the posterior distribution $p(\mathcal{C}, \Phi \mid \boldsymbol{\gamma}^{obs})$, which we will use to obtain Monte Carlo approximations of posterior expectations involved in the derivation of point estimates $\hat{\mathcal{C}}$, as presented in the next section. In Appendix B, we discuss the computational complexity of the Gibbs sampler, how computational performance can be improved through the usage of indexing techniques, and the initialization of the Gibbs sampler.

5.3 Point Estimation

In a Bayesian setting, one can obtain a point estimate $\hat{\mathcal{C}}$ of the multifile partition using the posterior $\mathbb{P}(\mathcal{C} \mid \boldsymbol{\gamma}^{obs}) = \int p(\mathcal{C}, \Phi \mid \boldsymbol{\gamma}^{obs}) d\Phi$ and a loss function $L(\mathcal{C}, \hat{\mathcal{C}})$. The Bayes estimate is the multifile partition $\hat{\mathcal{C}}$ that minimizes the expected posterior loss $\mathbb{E}[L(\mathcal{C}, \hat{\mathcal{C}}) \mid \boldsymbol{\gamma}^{obs}] = \sum_{\mathcal{C}} L(\mathcal{C}, \hat{\mathcal{C}}) \mathbb{P}(\mathcal{C} \mid \boldsymbol{\gamma}^{obs})$, although in practice such expectations are approximated using posterior samples. Previous examples of loss functions for partitions included Binder's loss (Binder 1978) and the variation of information (Meilă 2007), both recently surveyed in Wade & Ghahramani (2018). The quadratic and absolute losses presented in Tancredi & Liseo (2011) are special cases of Binder's loss, and Steorts et al. (2016) drew connections between their proposed maximal matching sets and the losses of Tancredi & Liseo (2011).

In many applications there may be much uncertainty on the linkage decision for some records in the datafiles. For example, in Figure 1 it is unclear which of the records with last name "Smith" are coreferent. It is thus desirable to leave decisions for some records unresolved, so that the records can be hand-checked during a clerical review, which is common in practice (see e.g. Ball & Price 2019). In the classification literature, leaving some decisions unresolved is done through a *reject option* (see e.g. Herbei & Wegkamp

2006), which here we will refer to as an *abstain option*. We will refer to point estimates with and without abstain option as *partial estimates* and *full estimates*, respectively. We now present a family of loss functions for multifile partitions which incorporate an abstain option, building upon the family of loss functions for bipartite matchings presented in Sadinle (2017).

5.3.1 A Family of Loss Functions with an Abstain Option

For the purpose of this section we will represent a multifile partition $\mathcal{C}$ as a vector $\boldsymbol{Z} = (Z_1, \cdots, Z_r)$ of labels, where $Z_i \in \{1, \ldots, r\}$, such that $Z_i = Z_j$ if $\mathcal{C}(i) = \mathcal{C}(j)$. We represent a Bayes estimate here as a vector $\hat{\boldsymbol{Z}} = (\hat{Z}_1, \ldots, \hat{Z}_r)$, where $\hat{Z}_i \in \{1, \ldots, r, A\}$, with A representing an abstain option intended for records whose linkage decisions are not clear and need further review. We assign different losses to using the abstain option and to different types of matching errors. We propose to compute the overall loss additively, as $L(\boldsymbol{Z}, \hat{\boldsymbol{Z}}) = \sum_{i=1}^{r} L_i(\boldsymbol{Z}, \hat{\boldsymbol{Z}})$. To introduce the expression for the ith-record-specific loss $L_i(\boldsymbol{Z}, \hat{\boldsymbol{Z}})$, we use the notation $\Delta_{ij} = I(Z_i = Z_j)$, and likewise $\hat{\Delta}_{ij} = I(\hat{Z}_i = \hat{Z}_j)$.

The proposed individual loss for record i is

$$L_i(\boldsymbol{Z}, \hat{\boldsymbol{Z}}) = \begin{cases} \lambda_A, & \text{if } \hat{Z}_i = A, \\ 0, & \text{if } \Delta_{ij} = \hat{\Delta}_{ij} \text{ for all } j \text{ where } \hat{Z}_j \neq A, \\ \lambda_{\text{FNM}}, & \text{if } \hat{Z}_i \neq A, \quad \sum_{j \neq i} \hat{\Delta}_{ij} = 0, \quad \sum_{j \neq i} \Delta_{ij} > 0, \\ \lambda_{\text{FM1}}, & \text{if } \hat{Z}_i \neq A, \quad \sum_{j \neq i} \hat{\Delta}_{ij} > 0, \quad \sum_{j \neq i} \Delta_{ij} = 0, \\ \lambda_{\text{FM2}}, & \text{if } \hat{Z}_i \neq A, \quad \sum_{j \neq i} \hat{\Delta}_{ij} > 0, \quad \sum_{j \neq i} (1 - \hat{\Delta}_{ij}) \Delta_{ij} > 0. \end{cases} \tag{3}$$

That is, λ_A represents the loss from abstaining from making a decision; λ_{FNM} is the loss from a false non-match (FNM) decision, that is, deciding that record i does not match any other record ($\sum_{j \neq i} \hat{\Delta}_{ij} = 0$) when in fact it does ($\sum_{j \neq i} \Delta_{ij} > 0$); λ_{FM1} is the loss from a type 1 false match (FM1) decision, that is, deciding that record i matches other records ($\sum_{j \neq i} \hat{\Delta}_{ij} > 0$) when it does not actually match any other record ($\sum_{j \neq i} \Delta_{ij} = 0$); and λ_{FM2} is the loss from a type 2 false match (FM2), that is, a false match decision when record i is matched to other records ($\sum_{j \neq i} \hat{\Delta}_{ij} > 0$) but it does not match all of the records it should be matching ($\sum_{j \neq i} (1 - \hat{\Delta}_{ij}) \Delta_{ij} > 0$).

The posterior expected loss is $\mathbb{R}(\hat{\boldsymbol{Z}}) = \sum_{i=1}^{r} \mathbb{E}[L_i(\boldsymbol{Z}, \hat{\boldsymbol{Z}}) \mid \boldsymbol{\gamma}^{obs}]$, where

$$\mathbb{E}[L_i(\boldsymbol{Z}, \hat{\boldsymbol{Z}}) \mid \boldsymbol{\gamma}^{obs}] = \begin{cases} \lambda_A, & \text{if } \hat{Z}_i = A, \\ \lambda_{\text{FNM}}\ \mathbb{P}(\sum_{j \neq i} \Delta_{ij} > 0 \mid \boldsymbol{\gamma}^{obs}), & \text{if } \hat{Z}_i \neq A, \sum_{j \neq i} \hat{\Delta}_{ij} = 0, \\ \lambda_{\text{FM1}}\ \mathbb{P}(\sum_{j \neq i} \Delta_{ij} = 0 \mid \boldsymbol{\gamma}^{obs}) + & \\ \lambda_{\text{FM2}}\ \mathbb{P}(\sum_{j \neq i} (1 - \hat{\Delta}_{ij}) \Delta_{ij} > 0 \mid \boldsymbol{\gamma}^{obs}), & \text{if } \hat{Z}_i \neq A, \sum_{j \neq i} \hat{\Delta}_{ij} > 0, \end{cases} \tag{4}$$

and quantities computed with respect to the posterior distribution, $\mathbb{P}(\boldsymbol{Z} \mid \boldsymbol{\gamma}^{obs})$, can all be approximated using posterior samples. While this presentation is for general positive losses $\lambda_{\text{FNM}}, \lambda_{\text{FM1}}, \lambda_{\text{FM2}}$ and λ_A, these only have to be specified up to a proportionality constant (Sadinle 2017). If we do not want to allow the abstain option, then we can set $\lambda_A = \infty$ and the derived full estimate $\hat{\boldsymbol{Z}}$ will have a linkage decision for all records. Although we have been using partition labelings $\boldsymbol{Z}$, the expressions in (3) and (4) are invariant to different labelings of the same partition. In the two-file case, Sadinle (2017) provided guidance on how to specify the individual losses $\lambda_{\text{FNM}}, \lambda_{\text{FM1}}, \lambda_{\text{FM2}}$ and λ_A in cases where there is a notion of false matches being worse than false non-matches or vise versa. Sadinle (2017) also gave recommendations for default values of these losses that lead to good frequentist performance in terms not over- or under-matching across repeated samples. In Appendix C, we discuss how our proposed loss function differs from the loss function of Sadinle (2017) and propose a strategy for approximating the Bayes estimate.

6 Simulation Studies

To explore the performance of our proposed approach for linking three duplicate-free files, as in the application to the Colombian homicide record systems of Appendix E, we present two simulation studies under varying scenarios of measurement error and datafile overlap. The two studies correspond to scenarios with equal and unequal measurement error across files, respectively. Both studies present results based on full estimates. In Appendix D we further explore the performance of our proposed approach for linking three files with duplicates, with results based on full and partial estimates.

6.1 General Setup

We start by describing the general characteristics of the simulations. For each of the simulation scenarios we conduct 100 replications, for each of which we generate three files as follows. For each of $n = 500$ entities, $\boldsymbol{h} \in \mathcal{H}$ is drawn from a categorical distribution with probabilities $\{p_{\boldsymbol{h}}\}_{\boldsymbol{h}\in\mathcal{H}}$, where $\boldsymbol{h}$ represents the subset of files the entity appears in, and so we change the values of $\{p_{\boldsymbol{h}}\}_{\boldsymbol{h}\in\mathcal{H}}$ across simulation scenarios to represent varying amounts of file overlap. Files are then created by generating the implied number of records for each entity. In the additional simulations considered in Appendix D, the generated number of records for each entity depends not only on $\boldsymbol{h}$, but also on the duplication mechanism.

All records are generated using a synthetic data generator developed in Tran et al. (2013), which allows for the incorporation of different forms of measurement error in individual fields, along with dependencies between fields we would expect in applications. The data generator first generates clean records before distorting them to create the observed records. In particular, each observed record will have a fixed number of erroneous fields, where errors selected uniformly at random from a set of field dependent errors displayed in Table 3 of Sadinle (2014) (reproduced in Appendix D), with a maximum of two errors per field. We generate records with seven fields of information: sex, given name, family name, age, occupation, postal code, and phone number.

For each simulation replicate, we construct comparison vectors as given in Table 4 of Sadinle (2014) (reproduced in Appendix D). We use the model for comparison data proposed in Section 4 with flat priors on $\boldsymbol{m}$ and $\boldsymbol{u}$ as discussed in Section 5.1, and the structured prior proposed in Section 3 with a uniform prior on the number of clusters and $\boldsymbol{\alpha} = (1, \cdots, 1)$ as described in Section 3.1. Using the Gibbs sampler presented in Appendix B we obtain $1,000$ samples from the posterior distribution of multifile partitions, and discard the first 100 as burn-in. In Appendix D we discuss convergence of the Gibbs sampler, present running times of the proposed approach, and present an extra simulation exploring the running time of the approach with a larger number of records. We then approximate the Bayes estimate $\hat{\boldsymbol{Z}}$ for multifile partitions using the loss function described in Section 5.3.1 as described in Appendix C. For full estimates, we use the default values of $\lambda_{\text{FNM}} = \lambda_{\text{FM1}} = 1$ and $\lambda_{\text{FM2}} = 2$ recommended by Sadinle (2017). In Appendix D we explore alternative specifications of the loss function.

We will assess the performance of the Bayes estimate using *precision* and *recall* with respect to the true coreference partition $\boldsymbol{Z}$. Let $\mathcal{P}$ be the set of all record pairs. Using notation from Section 5.3.1, let $TM(\boldsymbol{Z}, \hat{\boldsymbol{Z}}) = \sum_{(i,j)\in\mathcal{P}} \Delta_{ij}\hat{\Delta}_{ij}$ be the number of true matches (record pairs correctly declared coreferent), $FM(\boldsymbol{Z}, \hat{\boldsymbol{Z}}) = \sum_{(i,j)\in\mathcal{P}}(1-\Delta_{ij})\hat{\Delta}_{ij}$ be the number of false matches (record pairs incorrectly declared coreferent), and $FNM(\boldsymbol{Z}, \hat{\boldsymbol{Z}}) = \sum_{(i,j)\in\mathcal{P}} \Delta_{ij}(1-\hat{\Delta}_{ij})$ the number of false non-matches (record pairs incorrectly declared non-coreferent). Then *precision* is $TM(\boldsymbol{Z}, \hat{\boldsymbol{Z}})/[TM(\boldsymbol{Z}, \hat{\boldsymbol{Z}}) + FM(\boldsymbol{Z}, \hat{\boldsymbol{Z}})]$, the proportion of record pairs declared as coreferent that were truly coreferent, and *recall* is $TM(\boldsymbol{Z}, \hat{\boldsymbol{Z}})/[TM(\boldsymbol{Z}, \hat{\boldsymbol{Z}}) + FNM(\boldsymbol{Z}, \hat{\boldsymbol{Z}})]$, the proportion of record pairs that were truly coreferent that were correctly declared as coreferent. Perfect performance corresponds to precision and recall both being 1. In the simulations, we computed the median, 2nd, and 98th percentiles of these measures over the 100 replicate data sets. Additionally, in Appendix D, we assess the performance of the Bayes estimate when estimating the number of entities, n.

6.2 Duplicate-Free Files, Equal Errors Across Files

In this simulation study we explore the performance of our methodology by varying the number of erroneous fields per record over $\{1, 2, 3, 5\}$, and also varying $\{p_{\boldsymbol{h}}\}_{\boldsymbol{h}\in\mathcal{H}}$, which determines the amount of overlap, over four scenarios:

- High Overlap: $p_{001} = p_{010} = p_{100} = 0.4/3, p_{011} = p_{101} = p_{110} = 0.15, p_{111} = 0.15$,
- Medium Overlap: $p_{001} = p_{010} = p_{100} = 0.7/3, p_{011} = p_{101} = p_{110} = 0.05, p_{111} = 0.15$,
- Low Overlap: $p_{001} = p_{010} = p_{100} = 0.8/3, p_{011} = p_{101} = p_{110} = 0.05/3, p_{111} = 0.15$,
- No-Three-File Overlap: $p_{001} = p_{010} = p_{100} = 0.55/3, p_{011} = p_{101} = p_{110} = 0.15, p_{111} = 0$.

These are intended to represent a range of scenarios that could occur in practice. In the high overlap scenario 60% of the entities are expected to be in more than one datafile, in the low overlap scenario 80% of the entities are expected to be represented in a single datafile, and in the no-three-file overlap scenario no entities are represented in all datafiles.

To implement our methodology, in addition to the general set-up described in Section 6.1, we restrict the prior for the within-file cluster sizes so that they have size one with probability one, incorporating the

assumption of no-duplication within files (see Section 3.2). Imposing this restriction for all datafiles leads to a prior for tripartite matchings. To illustrate the impact of using our structured prior, we compare with the results obtained using our model for comparison data with a flat prior on tripartite matchings.

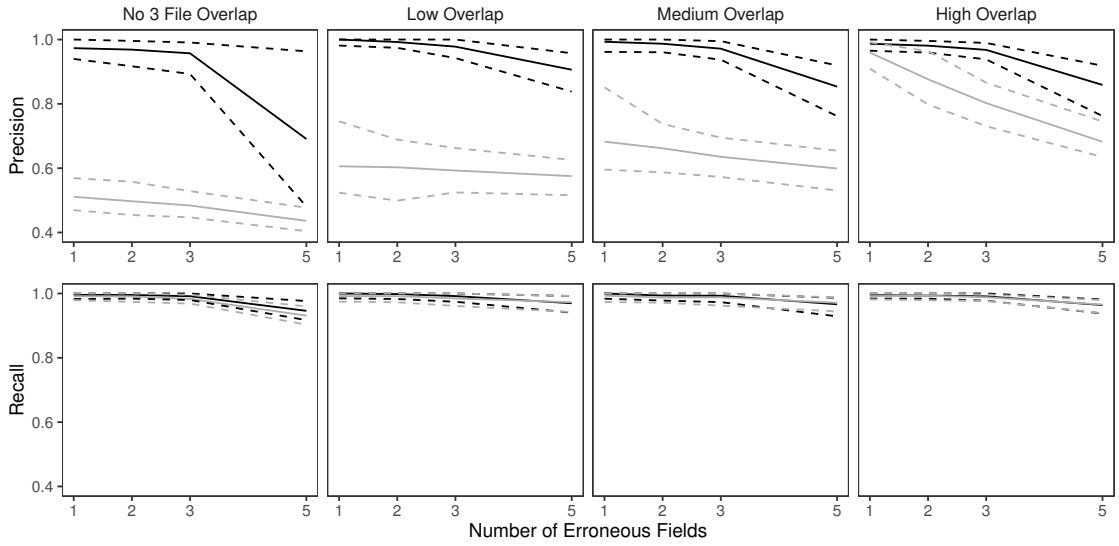


Figure 3: Performance comparison for simulation with equal measurement error across files. Black lines refer to results under our structured prior, grey lines to results under the flat prior, solid lines show medians, and dashed lines show 2nd and 98th percentiles.

The results of the simulation are seen in Figure 3. We see that our proposed approach performs consistently well across different settings, with the exception of the no-three-file overlap setting under high measurement error, where the precision decreases dramatically. The approach using a flat prior on tripartite matchings has poor precision in comparison, and it is particularly low when the amount of overlap is low. This suggests that our structured prior improves upon the flat prior by protecting against over-matching (declaring noncoreferent record pairs as coreferent). In Appendix D we demonstrate how the performance in the no-three-file overlap setting can be improved through the incorporation of an informative prior for the overlap table through $\boldsymbol{\alpha}$.

6.3 Duplicate-Free Files, Unequal Errors Across Files

In this simulation study we have different patterns of measurement error across the three files. Rather than each field in each record having a chance of being erroneous according to Table 3 of Sadinle (2014), we will

use the following measurement error mechanism to generate the data. For each record in the first file, age is missing, given name has up to seven errors, and all other fields are error free. For each record in the second file, sex and occupation are missing, last name has up to seven errors, and all other fields are error free. For each record in the third file, phone number and postal code have up to seven errors and all other fields are error free. Under this measurement error mechanism, there is enough information in the error free fields to inform pairwise linkage of the files. We further vary $\{p_{\boldsymbol{h}}\}_{\boldsymbol{h}\in\mathcal{H}}$ over the no-three-file and high overlap settings from Section 6.2.

Our goal in this study is to demonstrate that having both the structured prior for partitions and the separate models for comparison data from each file-pair can lead to better performance than not having these components. We will compare our model as described in Section 6.2 to both our model for comparison data with a flat prior on tripartite matchings (as in Section 6.2) and a simplification of our model for comparison data using a single model for all file-pairs but with our structured prior for partitions.

The results of the simulation are given in Figure 4. We see that our proposed approach outperforms both alternative approaches in both precision and recall in both overlap settings. This suggests that both the structured prior for tripartite matchings and the separate models for comparison data from each file-pair can help improve performance over alternative approaches. We note that in the no-three-file (high) overlap setting the precision of the proposed approach is greater than or equal to the precision of the approach using a single model for all file-pairs in 98 (100) of the 100 replications.

7 Discussion and Future Work

The methodology proposed in this article makes three contributions. First, the multifile partition parameterization, specific to the context of multifile record linkage and duplicate detection, allows for the construction of our structured prior for partitions, which provides a flexible mechanism for incorporating prior information about the data collection processes of the files. This prior is applicable to any Bayesian approach which requires a prior on partitions, including direct-modeling approaches such as Steorts et al. (2016). We are not aware of any priors for K-partite matchings when $K > 2$, so we hope our construction will lead to more development in this area. The second contribution is an extension of previous comparison-based models

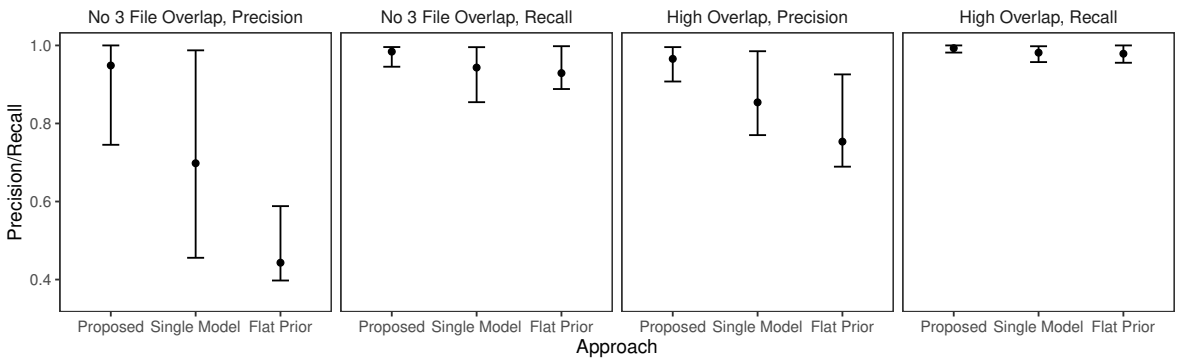


Figure 4: Performance comparison for simulation with unequal measurement error across files. "Proposed" refers to our proposed approach, "Single Model" refers to the approach using a single model for all file-pairs and our structured prior for partitions, and "Flat Prior" refers to the approach using our model for comparison data with a flat prior on tripartite matchings. Dots show medians, and bars show 2nd and 98th percentiles.

that explicitly handles the multifile setting. Allowing separate models for comparison data from each file pair leads to higher quality linkage. The third is a novel loss function for multifile partitions which can be used to derive Bayes estimates with good frequentist properties. Importantly, the loss function allows for linkage decisions to be left unresolved for records with large matching uncertainty. As with our structured prior on partitions, the loss function is applicable to any Bayesian approach which requires point estimates of partitions, including direct-modeling approaches.

There are a number of directions for future work. One direction is the modeling of dependencies between the comparison fields (see e.g. Larsen & Rubin 2001), which should further improve the quality of the linkage. Another direction is the development of approaches to jointly link records and perform a downstream analysis, thereby propagating the uncertainty from the linkage. See Section 7.2 of Binette & Steorts (2020) for a recent review of such joint models. In this direction, a natural task to consider next is population size estimation, where the linkage of the datafiles plays a central role (Tancredi & Liseo 2011, Tancredi et al. 2020).

Supplementary Appendices for Multifile Partitioning for Record Linkage and Duplicate Detection

A Structured Prior Appendix

In this appendix, we prove Proposition 1 from the main text and provide additional guidance for the specification of the structured prior for multifile partitions.

A.1 Proof of Proposition 1

In this section, we restate and prove Proposition 1 from the main text.

Proposition 1. *The number of K-partite matchings that have the same overlap table, $\boldsymbol{n} = \{n_{\boldsymbol{h}}\}_{\boldsymbol{h}\in\mathcal{H}}$, is $\prod_{k=1}^{K} n_k! / \prod_{\boldsymbol{h}\in\mathcal{H}} n_{\boldsymbol{h}}!$, where $n_k = \sum_{\boldsymbol{h}\in\mathcal{H}} h_k n_{\boldsymbol{h}}$ is the number of entities in datafile $\boldsymbol{X}_k$. Thus $\mathbb{P}(\mathcal{C} \mid \{\mathcal{C}_k\}_{k=1}^{K}, \boldsymbol{n}) = \prod_{\boldsymbol{h}\in\mathcal{H}} n_{\boldsymbol{h}}! / \prod_{k=1}^{K} n_k!$.*

Proof. Let us first count all of the ways that we can place the clusters in file $\boldsymbol{X}_k$ into the overlap table cells that $\boldsymbol{X}_k$ is included in, $\mathcal{H}_k = \{\boldsymbol{h} \in \mathcal{H} : h_k = 1\}$. This is just a multinomial coefficient, $n_k!/(\prod_{\boldsymbol{h}\in\mathcal{H}_k} n_{\boldsymbol{h}}!)$. Thus the number of ways we can place all of the clusters from all of the files into the cells in $\mathcal{H}$ is $\prod_{k=1}^{K} n_k!/(\prod_{\boldsymbol{h}\in\mathcal{H}_k} n_{\boldsymbol{h}}!) = (\prod_{k=1}^{K} n_k!)/[\prod_{h\in\mathcal{H}}(n_{\boldsymbol{h}}!)^{\sum_{k=1}^{K} h_k}]$. Given that there are $n_{\boldsymbol{h}}$ clusters from each file with $h_k = 1$ in cell $\boldsymbol{h}$, now all we have to count is how many distinct complete matchings are possible between them, which is just $(n_{\boldsymbol{h}}!)^{(\sum_{k=1}^{K} h_k)-1}$. Thus the number of K-partite matchings is $[(\prod_{k=1}^{K} n_k!)/(\prod_{\boldsymbol{h}\in\mathcal{H}}(n_{\boldsymbol{h}}!)^{\sum_{k=1}^{K} h_k})][\prod_{\boldsymbol{h}\in\mathcal{H}}(n_{\boldsymbol{h}}!)^{\sum_{k=1}^{K} h_k}/n_{\boldsymbol{h}}!] = \prod_{k=1}^{K} n_k! / \prod_{\boldsymbol{h}\in\mathcal{H}} n_{\boldsymbol{h}}!$. □

A.2 Prior Specification Guidance

In this section we provide additional guidance for the specification of the structured prior for multifile partitions described in Section 3 of the main text. In particular, we further discuss the priors for the number of clusters, the overlap tables, and the within-file cluster sizes.

Prior for the Number of Entities or Clusters. In the main text we recommended, in the absence of substantial prior information, to use a uniform prior on $\{1, \cdots, U\}$ for the number of clusters, for some upper bound

U. Our default recommendation was to set $U = r$, i.e. the total number of records. If one has substantive prior information about the number of clusters, this could instead be incorporated using other distributions on the positive integers. Miller et al. (2015) and Zanella et al. (2016) both suggest to use a negative-binomial distribution with parameters $a > 0$ and $q \in (0, 1)$ truncated to the positive integers, i.e. $\mathbb{P}(n) \propto \frac{\Gamma(n+a)}{(n!)\Gamma(a)}(1-q)^a q^n I(n \in \mathbb{N})$. Zanella et al. (2016) further suggest a weakly informative specification for this Negative-binomial prior where a and q are selected such that $\mathrm{E}[n] = \sqrt{\mathrm{Var}(n)} = r/2$. We follow Miller et al. (2015) and Zanella et al. (2016) and suggest a negative-binomial prior for the number of clusters n when incorporating substantive prior information.

Prior for the Overlap Table. In the main text we recommended using a Dirichlet-multinomial prior for the overlap table, specified by a collection of positive hyperparameters. In the absence of substantial prior information we recommended setting $\boldsymbol{\alpha} = (1, \cdots, 1)$. Due to conjugacy of the Dirichlet distribution with the multinomial, $\boldsymbol{\alpha}$ can be interpreted as prior cell counts, and thus our recommendation amounts to incorporating a prior count of 1 to each cell and an overall prior sample size of $2^K - 1$.

In our simulations and application, we found across a variety of overlap settings that this default prior performed satisfactorily. However, in the no-three-file-overlap setting, our approach struggled when there was a large amount of measurement error. We find in Appendix D.2 that we can improve performance in this setting by using informative prior cell counts, rather than using our default specification. What sets this no-three-file-overlap simulation setting apart from the other settings is that it is sparse, i.e. in this setting the count for the three-file-overlap cell of the overlap table is truly 0.

When linking a large number of files K, the size of the overlap table, $2^K - 1$, becomes large very quickly, which makes it likely that the true overlap table is sparse. Our default prior specification may not be appropriate in these settings as using a prior cell count of 1 for each cell may be incorporating prior information that is too strong, as illustrated in Example 1.4 of Berger et al. (2015). One possible alternative as a default specification when the overlap table is potentially sparse, would be to set $\boldsymbol{\alpha} = (1/(2^K - 1), \cdots, 1/(2^K - 1))$ (see Section 3.2 of Berger et al. (2015) for justification). If one has prior information concerning which cells of the overlap table are likely to be sparse, based on the results in Appendix D.2, we recommend attempting to incorporate this information into the prior. For example, if

it is believed that some combination of files are likely not to have collected information on the same set of entities, one can incorporate this information by making the corresponding prior cell counts close to 0.

Another route one could take would be to replace the Dirichlet-multinomial prior on $\boldsymbol{n}$ with a multinomial prior on $\boldsymbol{n}$, with a non-Dirichlet prior on the multinomial cell probabilities. For example, one could use the tensor-factorization priors of Dunson & Xing (2009) and Zhou et al. (2015), which have been shown to have lead to estimates of cell probabilities with good performance in large sparse contingency tables.

Prior for the Within-File Cluster Sizes. Given that in a joint record linkage and duplicate detection scenario we do not expect there to be many duplicates per entity in any given file, in the main text we recommended specifying i.i.d. priors for the sizes of the within-file clusters, i.e. $d_1^k, \cdots, d_{n_k}^k \mid n_k \overset{iid}{\sim} p_k(\cdot)$ for a given file $\boldsymbol{X}_k$. Here $p_k(\cdot)$ represents the probability mass function of a distribution on $\{1, \cdots, U^k\}$, where U^k is a file-specific upper bound on cluster sizes.

When a given file $\boldsymbol{X}_k$ is assumed to have no duplicates, in the main text we recommended enforcing this restriction that there are no duplicates in that file by setting $U^k = 1$ and $p_k(d_i^k) = I(d_i^k = 1)$. When a given file $\boldsymbol{X}_k$ is assumed to have duplicates, in the main text we recommended a Poisson distribution with parameter $\lambda = 1$, i.e. $p_k(d_i^k) \propto (d_i^k!)^{-1} I(d_i^k \in \{1, \cdots, U^k\})$. This prior places most of the prior mass close to 1, where various properties such as prior mean and standard deviation can be computed numerically (e.g. when $U^k = 10$ the prior mean is 1.58). If one has information on the average amount of duplication in file k, given a specified upper bound U^k, one could specify the parameter λ of the Poisson prior such that the prior mean is equal to the average amount of duplication. Alternatively, one could place a hyperprior on λ. This is similar approach to Zanella et al. (2016), who used a negative-binomial distribution with parameters $r > 0$ and $p \in (0, 1)$ truncated to the positive integers, with a gamma hyperprior for r and a beta hyperprior for p. Indeed, the negative-binomial prior of Zanella et al. (2016) can be seen as a generalization of our Poisson prior, based on well-known connections between the Poisson and negative-binomial, and could be used instead of our Poisson recommendation if desired.

In the simulations presented in Appendix D.3 we explore using a Poisson prior with parameter λ varying over $\{0.1, 1, 2\}$, when the within-file cluster sizes are generated from a Poisson with parameter λ varying over $\{0.1, 1, 2\}$. We find in these simulations that when there is medium or high duplication (i.e. the within-file

cluster sizes are generated from a Poisson with mean in $\{1, 2\}$), the results are not sensitive to λ, whereas when there is low duplication (i.e. the within-file cluster sizes are generated from a Poisson with mean 0.1), the results are sensitive to λ. This suggests that model performance is more sensitive to the specification of the prior distribution for within-file cluster sizes when there is a low amount of duplication, and that care should be taken when specifying the prior for the within-file cluster sizes for files which are expected to have very little duplication.

B Posterior Inference Appendix

In this appendix, we first derive full conditional distributions of our structured prior for partitions, and then use the full conditional distributions to derive a Gibbs sampler for posterior inference in our model. We then discuss the computational complexity of our approach to posterior inference, how computational performance can be improved through the usage of indexing techniques, and the initialization of the Gibbs sampler.

B.1 Conditional Assignment Probabilities

In this section we use the form of the prior distribution in Equation (1) of the main text to derive the conditional probability for assigning a record j from file $\boldsymbol{X}_k$ to a given cluster of an existing multifile partition $\mathcal{C}_{-j}$ of the other records. Specifically, we derive $\mathbb{P}(j \to c \mid \mathcal{C}_{-j})$, where $j \to c$ denotes adding record j to a cluster $c \in \mathcal{C}_{-j}$ or to an empty cluster. Let a quantity followed by $(\mathcal{C}_{-j})$ denote that it is derived from $\mathcal{C}_{-j}$ analogously to in Section 3.1 of the main text. Let $\mathbf{1}_k$ denote the inclusion pattern indicating inclusion only in file $\boldsymbol{X}_k$, that is, $\mathbf{1}_k$ is a vector of zeroes except for its kth entry which equals 1. Further, let $\mathbf{1}_c$ denote the inclusion pattern of the cluster $c \in \mathcal{C}_{-j}$, that is, the lth entry of $\mathbf{1}_c$ is $I(c^l \neq \emptyset)$. Finally, let $\mathbf{1}_{c \cup j}$ denote the inclusion pattern of the cluster $c \in \mathcal{C}_{-j}$ after adding record j to it. Then the conditional assignment probability is

$$\mathbb{P}(j \to c \mid \mathcal{C}_{-j}) \propto$$
$$\begin{cases} \left[\frac{\mathbb{P}(n(\mathcal{C}_{-j})+1)}{\mathbb{P}(n(\mathcal{C}_{-j}))}\right]\left[\frac{(n(\mathcal{C}_{-j})+1)(n_{\mathbf{1}_k}(\mathcal{C}_{-j})+\alpha_{\mathbf{1}_k})}{n(\mathcal{C}_{-j})+\alpha_0}\right] p_k(1) & , \text{ if } c = (\emptyset, \cdots, \emptyset) \\ \left[\frac{n_{\mathbf{1}_{c\cup j}}(\mathcal{C}_{-j})+\alpha_{\mathbf{1}_{c\cup j}}}{n_{\mathbf{1}_c}(\mathcal{C}_{-j})+\alpha_{\mathbf{1}_c}-1}\right] p_k(1) & , \text{ if } c \neq (\emptyset, \cdots, \emptyset), |c^k| = 0 \\ (|c^k|+1)\left[\frac{p_k(|c^k|+1)}{p_k(|c^k|)}\right] & , \text{ if } |c^k| > 0. \end{cases}$$

B.2 Gibbs Sampler

We will now derive a Gibbs sampler to explore the posterior of Φ and $\mathcal{C}$. Suppose we are at iteration $t+1$ of the sampler, with current samples $\Phi^{[t]} = (\boldsymbol{m}^{[t]}, \boldsymbol{u}^{[t]})$ and $\mathcal{C}^{[t]}$. Then we obtain the samples for iteration $t+1$ through the following steps:

1. For $k \leq k'$ and $f \in \{1, \ldots, F\}$, sample

$$\boldsymbol{m}_{kk'}^{f[t+1]} \mid \mathcal{C}^{[t]}, \boldsymbol{\gamma}^{obs} \sim \mathsf{Dirichlet}(a_{kk'}^{f0}(\mathcal{C}^{[t]}) + \mu_{kk'}^{f0}, \cdots, a_{kk'}^{fL_f}(\mathcal{C}^{[t]}) + \mu_{kk'}^{fL_f})$$

and

$$\boldsymbol{u}_{kk'}^{f[t+1]} \mid \mathcal{C}^{[t]}, \boldsymbol{\gamma}^{obs} \sim \mathsf{Dirichlet}(b_{kk'}^{f0}(\mathcal{C}^{[t]}) + \nu_{kk'}^{f0}, \cdots, b_{kk'}^{fL_f}(\mathcal{C}^{[t]}) + \nu_{kk'}^{fL_f}).$$

Call these samples $\Phi^{[t+1]}$.

2. We now sample the cluster assignment for each record $j \in [r]$ sequentially. Suppose we have sampled the first $j-1$ records, and are sampling the cluster assignment for record j from file $\boldsymbol{X}_k$. Let $\mathcal{C}_{-j}^{[t]}$ denote the current partition of $[r]$, without record j, after sampling the first $j-1$ records. Then we sample the cluster assignment for record j according to the following probabilities:

$$\mathbb{P}(j \to c \mid \mathcal{C}_{-j}^{[t]}, \Phi^{[t+1]}, \boldsymbol{\gamma}^{obs}) \propto$$
$$\begin{cases} \left[\frac{\mathbb{P}(n(\mathcal{C}_{-j}^{[t]})+1)}{\mathbb{P}(n(\mathcal{C}_{-j}^{[t]}))}\right]\left[\frac{(n(\mathcal{C}_{-j}^{[t]})+1)(n_{\mathbf{1}_k}(\mathcal{C}_{-j}^{[t]})+\alpha_{\mathbf{1}_k})}{n(\mathcal{C}_{-j}^{[t]})+\alpha_0}\right] p_k(1) & , \text{ if } c = (\emptyset, \cdots, \emptyset) \\ \left[\prod_{k'=1}^{K}\prod_{i\in c^{k'}} \mathcal{L}_{ij}^{[t+1]}\right]\left[\frac{n_{\mathbf{1}_{c\cup j}}(\mathcal{C}_{-j}^{[t]})+\alpha_{\mathbf{1}_{c\cup j}}}{n_{\mathbf{1}_c}(\mathcal{C}_{-j}^{[t]})+\alpha_{\mathbf{1}_c}-1}\right] p_k(1) & , \text{ if } |c^k| = 0, c \neq (\emptyset, \cdots, \emptyset) \\ \left[\prod_{k'=1}^{K}\prod_{i\in c^{k'}} \mathcal{L}_{ij}^{[t+1]}\right](|c^k|+1)\left[\frac{p_k(|c^k|+1)}{p_k(|c^k|)}\right] & , \text{ if } |c^k| > 0, \end{cases}$$

where, letting k' denote the file that record i is in,

$$\mathcal{L}_{ij}^{[t+1]} = \prod_{f=1}^{F} \left[\prod_{l=0}^{L_f} \left(\frac{m_{kk'}^{fl[t+1]}}{u_{kk'}^{fl[t+1]}} \right)^{I(\gamma_{ij}^f = l)} \right]^{I_{obs}(\gamma_{ij}^f)}$$
$$= \exp \left[\sum_{f=1}^{F} I_{obs}(\gamma_{ij}^f) \sum_{l=0}^{L_f} \log \left(\frac{m_{kk'}^{fl[t+1]}}{u_{kk'}^{fl[t+1]}} \right) I(\gamma_{ij}^f = l) \right].$$

B.3 Computational Complexity

The computational complexity of posterior inference in our proposed approach can be broken up into the complexity of pre-computing comparison vectors, and the complexity of individual steps of the Gibbs sampler presented in Appendix B.2.

- The computational complexity of pre-computing comparison vectors is $\mathcal{O}(\texttt{rp} * F)$, where $\texttt{rp}$ is the number of valid record pairs. To be more specific, when we assume there are duplicates in every file, $\texttt{rp} = r(r-1)/2$, and when we assume there are no duplicates in each file, $\texttt{rp} = \sum_{k<k'} r_k r_{k'}$. For in between situations where we assume there are no duplicates in some files and duplicates in the remaining files, it can be shown that $\sum_{k<k'} r_k r_{k'} < \texttt{rp} < r(r-1)/2$. Thus in the most general case, pre-computing comparison vectors scales quadratically in the number of records. We discuss in the following section how the cost of this step can be reduced through the usage of blocking.

- The computational complexity of step 1 of the Gibbs sampler presented in Appendix B.2, i.e. sampling the $\boldsymbol{m}$ and $\boldsymbol{u}$ parameters, is $\mathcal{O}(\texttt{rp} * \texttt{fl} + \texttt{fp} * \texttt{fl})$, where $\texttt{fp}$ is the number of valid file pairs, and $\texttt{fl} = \sum_{f=1}^{F}(L_f+1)$ is the total number of agreement levels across all fields. We have that $\texttt{fp} = \binom{K}{2} + K_d$, where K_d is the number of files that are assumed to have duplicates. This follows as the a and b summaries of the partition can be calculated from a matrix multiplication of a $\texttt{fl} \times \texttt{rp}$ matrix and a $\texttt{rp} \times 1$ matrix, and given these a and b summaries the complexity of sampling the $\boldsymbol{m}$ and $\boldsymbol{u}$ parameters from their full conditionals is $\mathcal{O}(\texttt{fp} * \texttt{fl})$.

- The computational complexity of step 2 of the Gibbs sampler presented in Appendix B.2, i.e. sampling the partition $\mathcal{C}$, is difficult to analyze in general. In the most general case, where we assume there are duplicates in each file, in the worst case scenario, each record could be placed in its own cluster.

The complexity of sampling the cluster assignment for a single record would then be $\mathcal{O}(r)$, and the complexity of sampling the cluster assignment for all records would then be $\mathcal{O}(r^2)$. However, the number of clusters potentially changes whenever a new cluster assignment is sampled, which complicates this analysis. Further, once introduces constraints on the partition space, either through assuming there are no duplicates in some files, or using indexing as described in the next section, the number of clusters available for a specific record's cluster assignment step will depend on these constraints. In general the best we can say is that this step will be faster when each record has on average (with respect to the posterior) a small number of clusters to which it can be assigned, and slower when each record has on average a large number of clusters to which it can be assigned.

In our current implementation of the proposed approach, we have found that even though both steps of the Gibbs sampler scale quadratically in the number of records in the worst case, the cost of sampling the partition generally dominates the cost of sampling the $\boldsymbol{m}$ and $\boldsymbol{u}$ parameters, and is the main bottleneck of our approach. We note that the sampling of the partition will essentially have the same computational complexity regardless of whether one uses a comparison-based model for records, as we have proposed in the main text, or one uses a direct-modeling approach, as in Steorts et al. (2016).

B.4 Blocking, Indexing, and Scalability

As described in the previous section, there are two main bottlenecks to scalability in our proposed approach: pre-computing the comparison vectors and sampling the partition from its full conditional in the Gibbs sampler presented in Appendix B.2. Both of these bottlenecks can be sped up through the use of indexing techniques, which declare certain pairs of records non-coreferent a priori based on comparisons of a small number of fields (Christen 2012, Steorts et al. 2014, Murray 2015). This both reduces the number of record pairs under consideration, and reduces on average the number of clusters to which each record can be assigned in the Gibbs sampler presented in Appendix B.2.

In particular, if $\mathcal{P} = \cup_{k \leq k'} \mathcal{P}_{kk'}$ is the set of all possible record pairs, indexing techniques generate a set $\mathcal{P}^* \subset \mathcal{P}$, such that $|\mathcal{P}^*| \ll |\mathcal{P}|$, where record pairs in $\mathcal{P}^*$ are candidate coreferent pairs, and record pairs in $\mathcal{P} \setminus \mathcal{P}^*$ are fixed as non-coreferent. Thus when performing posterior inference, this truncates our prior on

multifile partitions to the set $\{\mathcal{C} : \mathcal{C}(i) \neq \mathcal{C}(j), \forall (i,j) \in \mathcal{P} \setminus \mathcal{P}^*\}$.

We briefly review two common indexing techniques from Murray (2015), blocking and indexing by disjunction. Blocking declares pairs of records to be non-coreferent when they disagree on a set of error-free fields. The use of error-free fields guarantees that the candidate coreferent pairs output from blocking are transitive, so that $\mathcal{P}^*$ forms a partition of the records. Indexing by disjunction declares pairs of records to be non-coreferent when they disagree at a certain threshold for each field in a given set of reliable fields. Candidate coreferent pairs output from indexing by disjunction are not guaranteed to be transitive.

When a set of error-free fields are available, we recommend blocking. Blocking schemes can be implemented without constructing comparison vectors for each record pair, thus reducing the cost of pre-computing comparison vectors for all record pairs to just the cost of pre-computing comparison vectors for record pairs within each block. Our proposed approach can then be run independently in each block, drastically reducing on average the number of clusters to which each record can be assigned in the Gibbs sampler presented in Appendix B.2.

Within blocks, there is no further way to reduce cost of the pre-computing the comparison vectors. However, it is still possible to reduce the cost of sampling the partition from its full conditional in the Gibbs sampler presented in Appendix B.2 through the use of indexing by disjunction. By fixing certain pairs of records to be non-coreferent, one reduces on average the number of clusters to which each record can be assigned in the Gibbs sampler presented in Appendix B.2. Note that the comparisons for record pairs fixed as non-coreferent in $\mathcal{P} \setminus \mathcal{P}^*$ still contribute to the model through the $b_{kk'}^{fl}$ term in the likelihood in Equation (2) of the main text, avoiding many of the issues presented in Murray (2015).

However, the non-transitivity of $\mathcal{P}^*$ output from indexing by disjunction can be problematic, as it suggests that the thresholds used in indexing by disjunction are too stringent, and that they may be excluding true coreferent pairs. Non-transitivity can also cause problems for Markov chain Monte Carlo samplers (like our Gibbs sampler in Appendix B.2), as it can make traversing the constrained space of multifile partitions difficult. To avoid the issue of non-transitivity, we propose to use the transitive closure of the candidate coreferent pairs, $\mathcal{P}^*$, generated by indexing by disjunction, which we refer to as *transitive indexing*. Transitive indexing has been used before in the post-hoc blocking methodology of McVeigh et al. (2019) for two-file

record linkage.

B.5 Initialization

Due to the nature of the Gibbs sampler in Appendix B.2, we can initialize the multifile partition $\mathcal{C}$ without needing to initialize Φ. A simple initialization for $\mathcal{C}$ is to let each record belong to its own cluster, which works well when indexing is used. However, we observed during some preliminary simulations that when sampling K-partite matchings without using indexing, the sampler can take a large number of iterations to mix if we initialize $\mathcal{C}$ in this way, where the number of iterations depends on the partition the data was simulated from. This problem can not be avoided as in Appendix B.4, as the constraints on the space of K-partite matchings cannot be relaxed. In this case, we constructed a simple alternative initialization. The idea is to use an indexing scheme for initialization, even if indexing is not being used to reduce the number of candidate coreferent pairs. In particular, we first generate a set of record pairs $\mathcal{P}^* \subset \mathcal{P}$ through transitive indexing as described in Appendix B.4. For each block of records in $\mathcal{P}^*$, we sample a random K-way matching of records in that block. We then initialize $\mathcal{C}$ such that each record belongs to its own cluster, except for the sampled K-way matchings.

C Point Estimation Appendix

In this appendix we discuss how our proposed loss function differs from the loss function of Sadinle (2017), and propose a strategy for approximating the Bayes estimate under our proposed loss function.

C.1 Comparison to Sadinle (2017)

Unlike our loss function construction, in the two-file set-up Sadinle (2017) constructed the loss function from individual losses for the records in the smaller datafile only. Such construction however leads to an asymmetry in the loss function that is arbitrary. Consider an example of two datafiles, where the first file has records a and b, and the second file has records c and d. In that case the role of the datafiles can be arbitrarily interchanged. If the true matching has a link between a and c but the matching estimate has a link between b and c, the loss will be $\lambda_{\text{FNM}} + \lambda_{\text{FM1}}$ if file two is chosen not to contribute to the loss, but it

will be λ_{FM2} if file one is chosen not to contribute to the loss. Our new construction presented in the main text does not lead to such issues.

C.2 Approximating the Bayes Estimate

Finding a partition $\hat{\boldsymbol{Z}}$ such that $\mathbb{R}(\hat{\boldsymbol{Z}})$ is minimized corresponds to an optimization problem closely related to graph partitioning problems (e.g., Brandes et al. 2007, Lancichinetti & Fortunato 2009, Newman 2013) or correlation clustering (e.g., Bansal et al. 2004, Demaine et al. 2006), both of which are known to be NP-complete. Thus, unlike in Sadinle (2017), we cannot minimize $\mathbb{R}(\hat{\boldsymbol{Z}})$ exactly in general. All approaches for graph partitioning problems or for correlation clustering instead rely on heuristic algorithms whose performance is evaluated empirically via simulation studies and benchmark datasets. We will follow a similar approach.

We take advantage of the fact that in practice a large number of record pairs will have zero or close to zero posterior probability of matching $\mathbb{P}(\Delta_{ij} = 1 \mid \boldsymbol{\gamma}^{obs})$. Based on this, we propose to threshold $\mathbb{P}(\Delta_{ij} = 1 \mid \boldsymbol{\gamma}^{obs})$ at a small value δ to create a graph where an edge represents a non-negligible probability of matching between two records. We then break the records up into connected components of this graph, each component representing groups of records that are more likely to be coreferent. We then find the Bayes estimate by minimizing $\mathbb{R}(\hat{\boldsymbol{Z}})$ separately within each of these connected components. We can think of δ as a way of trading-off between accuracy of the Bayes estimate and computational tractability: larger values of δ decrease the size of the resulting connected components, making the minimization more tractable within each component, while smaller δ make the resulting approximation more accurate as using the threshold $\delta = 0$ is no longer an approximation. We recommend setting δ as the smallest probability such that the largest connected component is smaller than some pre-specified upper bound that captures a computational budget. To minimize $\mathbb{R}(\hat{\boldsymbol{Z}})$ within the connected components, we propose to do so over posterior samples, $\{\boldsymbol{Z}^{[t]}\}_{t=1}^{T}$, and find the sample which minimizes $\mathbb{R}(\boldsymbol{Z}^{[t]})$. As this minimization is happening separately within each connected component, the final Bayes estimate of the partition of all r records does not itself have to be a posterior sample.

To minimize $\mathbb{R}(\hat{\boldsymbol{Z}})$ when searching for partial estimates, let Ω denote the power set of $[r]$, and let

$\boldsymbol{Z}_{\omega}^{[t]}$ denote the posterior draw $\boldsymbol{Z}^{[t]}$ where the records in $\omega \in \Omega$ are set to abstain, A. Then in order to accommodate partial estimates, we can minimize $\mathbb{R}(\hat{\boldsymbol{Z}})$ over $\{\boldsymbol{Z}_{\omega}^{[t]} \mid t \in [T], \omega \in \Omega\}$.

Unless stated otherwise, in all simulations and in the application, for full (partial) estimates, we find the Bayes estimate separately within connected components of records with posterior probability larger than δ of matching, where δ is the smallest probability such that the largest connected component is smaller than 50 (12).

D Simulation Appendix

D.1 Tables 3 and 4 from Sadinle (2014)

Table 3 of Sadinle (2014) is reproduced in Table 1. In this table, edit errors are insertions, deletions, or substitutions of characters in a string, OCR errors are optical character recognition errors, keyboard errors are typing errors that rely on a certain keyboard layout, and phonetic errors are errors using a list of predefined phonetic rules. Table 4 of Sadinle (2014) is reproduced in Table 2.

Table 1: Types of errors per field in the simulation studies.

	Type of error					
Field	**Missing values**	**Edits**	**OCR**	**Keyboard**	**Phonetic**	**Misspelling**
Given name		✓	✓	✓	✓	
Family name		✓	✓	✓	✓	✓
Age interval	✓					
Sex	✓					
Occupation	✓					
Phone number	✓	✓	✓	✓		
Postal code	✓	✓	✓	✓		

Table 2: Construction of levels of disagreement for the simulation studies.

		Levels of disagreement			
Field	**Similarity measure**	**0**	**1**	**2**	**3**
Given name	Levenshtein	0	$(0, 0.25]$	$(0.25, 0.5]$	$(0.5, 1]$
Family name	Levenshtein	0	$(0, 0.25]$	$(0.25, 0.5]$	$(0.5, 1]$
Age interval	Binary comparison	Agree	Disagree		
Sex	Binary comparison	Agree	Disagree		
Occupation	Binary comparison	Agree	Disagree		
Phone number	Levenshtein	0	$(0, 0.25]$	$(0.25, 0.5]$	$(0.5, 1]$
Postal code	Levenshtein	0	$(0, 0.25]$	$(0.25, 0.5]$	$(0.5, 1]$

D.2 Prior Sensitivity Analysis for Simulation with Duplicate-Free Files, Equal Errors Across Files

In Section 6.2 of the main text, we saw that our proposed approach struggled in the no-three-file overlap setting when there was high measurement error. In practice, if we knew that no entity is represented in all three datafiles we could enforce that restriction just like we enforce that there are no duplicates in given files, which would likely lead to better performance. While it is reasonable to assume in some applications that there are no duplicates in a given file (for example the application considered in Section 7 of the main text), it is less reasonable to assume with absolute certainty that there is no entity represented in all three datafiles. Thus we want to instead incorporate the weaker information that there is a low amount of three way overlap. We can achieve this through an informative specification of $\boldsymbol{\alpha}$.

In the no-three-file overlap setting, the overlap table was generated from a multinomial distribution with probability vector $\boldsymbol{p} = (p_{001}, p_{010}, p_{011}, p_{100}, p_{101}, p_{110}, p_{111})$ where $p_{001} = p_{010} = p_{100} = 0.55/3, p_{011} = p_{101} = p_{110} = 0.15, p_{111} = 0$. Note that our Dirichlet-multinomial prior can be motivated as the result of first drawing $\{q_{\boldsymbol{h}}\}_{\boldsymbol{h} \in \mathcal{H}}$ from a Dirichlet distribution with hyperparameters $\boldsymbol{\alpha}$, then drawing $\boldsymbol{n}$ from a multinomial distribution of size n with probabilities $\{q_{\boldsymbol{h}}\}_{\boldsymbol{h} \in \mathcal{H}}$. As discussed in Appendix A.2, $\boldsymbol{\alpha}$ can be interpreted as prior cell counts, which can be used to incorporate prior information about the amount of overlap between

datafiles. Thus when specifying an informative $\boldsymbol{\alpha}$, we want the Dirichlet prior for $\{q_{\boldsymbol{h}}\}_{\boldsymbol{h}\in\mathcal{H}}$ to be centered roughly around $\boldsymbol{p}$. We can accomplish this by setting $\boldsymbol{\alpha} = \kappa \times (p_{001}, p_{010}, p_{011}, p_{100}, p_{101}, p_{110}, 1/\kappa)$. $\kappa + 1$ represents the sum of prior cell counts. As κ increases, the Dirichlet prior for $\{q_{\boldsymbol{h}}\}_{\boldsymbol{h}\in\mathcal{H}}$ becomes more concentrated near $\boldsymbol{p}$.

We repeated the no-three-file overlap setting simulation from Section 6.2 of the main text using this informative specification with $\kappa \in \{49, 99\}$. The results are presented in Figure 5. We see that when there is high measurement error, these more informative specifications improve upon the performance of the default specification of $\boldsymbol{\alpha} = (1, \cdots, 1)$.

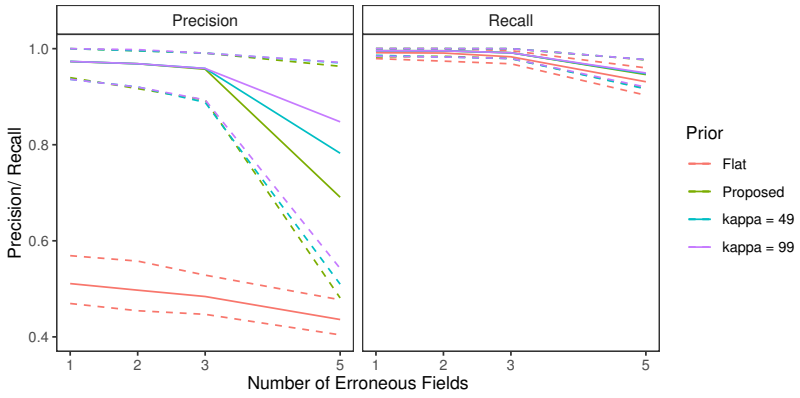


Figure 5: Performance comparison for no-three-file overlap simulation with more informative settings of $\boldsymbol{\alpha}$. Solid lines show medians, and dashed lines show 2nd and 98th percentiles. "Flat" refers to a flat prior on tripartite matchings, "Proposed" refers to our structured prior for partitions when $\boldsymbol{\alpha} = (1, \cdots, 1)$, and "kappa = 49" and "kappa = 99" refer to the more informative specifications of $\boldsymbol{\alpha}$ with $\kappa \in \{49, 99\}$.

D.3 Files with Duplicates, Full Estimates

This simulation study consists of linkage and duplicate detection for three datafiles, so that the target of inference is a general multifile partition. We conduct this study with probabilities fixed at $p_{001} = p_{010} = p_{100} = 0.3, p_{011} = p_{101} = p_{110} = 0.025, p_{111} = 0.025$, representing a very low overlap setting, which can be challenging as seen in Section 6.2 of the main text. For each entity represented in the datafiles we generated a within-file cluster size from a Poisson distribution with mean λ truncated to $\{1, \cdots, 5\}$. In this study, in addition to varying the number of erroneous fields per record over $\{1, 2, 3, 5\}$ to explore different amounts of

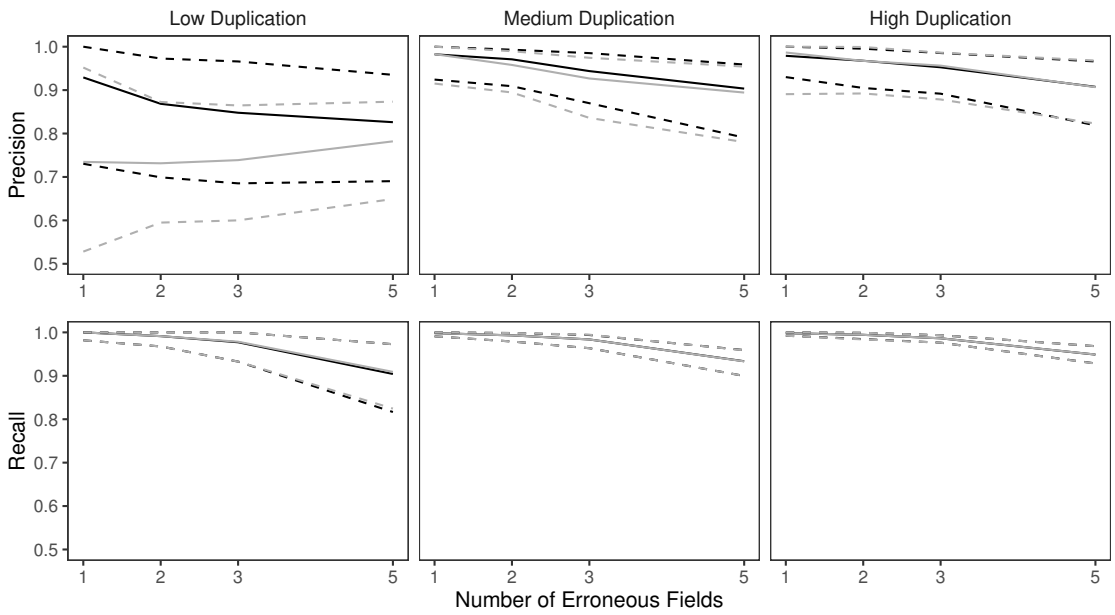


Figure 6: Performance comparison for simulation with datafiles with duplicates and full estimates. Black lines refer to results under our structured prior, grey lines refer to the approach of Sadinle (2014), solid lines show medians, and dashed lines show 2nd and 98th percentiles.

measurement error, and we vary λ over $\{0.1, 1, 2\}$ to explore low, medium, and high amounts of duplication.

To implement our methodology, in addition to the general set-up described in Section 6.1 of the main text, we use a Poisson prior with mean 1 truncated to $\{1, \cdots, 10\}$ on the within-file cluster sizes. For comparison we use the comparison-based model of Sadinle (2014), which treats all of the records as coming from one file and uses a flat prior on partitions. For both models we use transitive indexing as in described Appendix B.4 to reduce the number of comparisons, where the initial indexing scheme declares record pairs as non-coreferent if they disagree in either given or family name at the highest level (according to Table 4 of Sadinle (2014)).

The results of the simulation are seen in Figure 6. In the medium and high duplication settings, the models have similarly good performance. We believe that the similar performance between models in these settings is due to the use of the indexing, which significantly reduces the size of the space of possible multifile partitions, so that the influence of the structured prior is minimized. However, in the low duplication setting we see that the precision of the proposed model is better across the varying measurement error settings than the model of Sadinle (2014). This suggests that in low duplication settings, our approach once again

improves upon an approach that uses flat priors for partitions by protecting against over-matching.

We now explore the sensitivity of our approach to changes in the prior for the number within-file cluster sizes, and demonstrate how the performance in the low duplication setting can be further improved through the incorporation of an informative prior for the within-file cluster sizes. In the simulation that was just described, we used a Poisson prior with mean $\lambda = 1$ truncated to $\{1, \cdots, 10\}$ for the within-file cluster sizes. In the simulation, the within-file cluster sizes were generated from Poisson distributions with mean λ truncated to $\{1, \cdots, 5\}$, where λ varied over $\{0.1, 1, 2\}$. We now repeat the same simulation using Poisson priors with mean λ truncated to $\{1, \cdots, 10\}$ for the within-file cluster sizes, where $\lambda \in \{0.1, 1, 2\}$. The results are presented in Figure 7. The results for the medium and high duplication settings are very robust to the within-file cluster size prior specification. In the low duplication setting we see that the performance among the different within-file cluster size prior specifications is best when $\lambda = 0.1$, and worst when $\lambda = 2$ (but still better than the approach of Sadinle (2014)). This behavior is expected as the within-file cluster size prior with $\lambda = 0.1$ is informative in the low duplication setting.

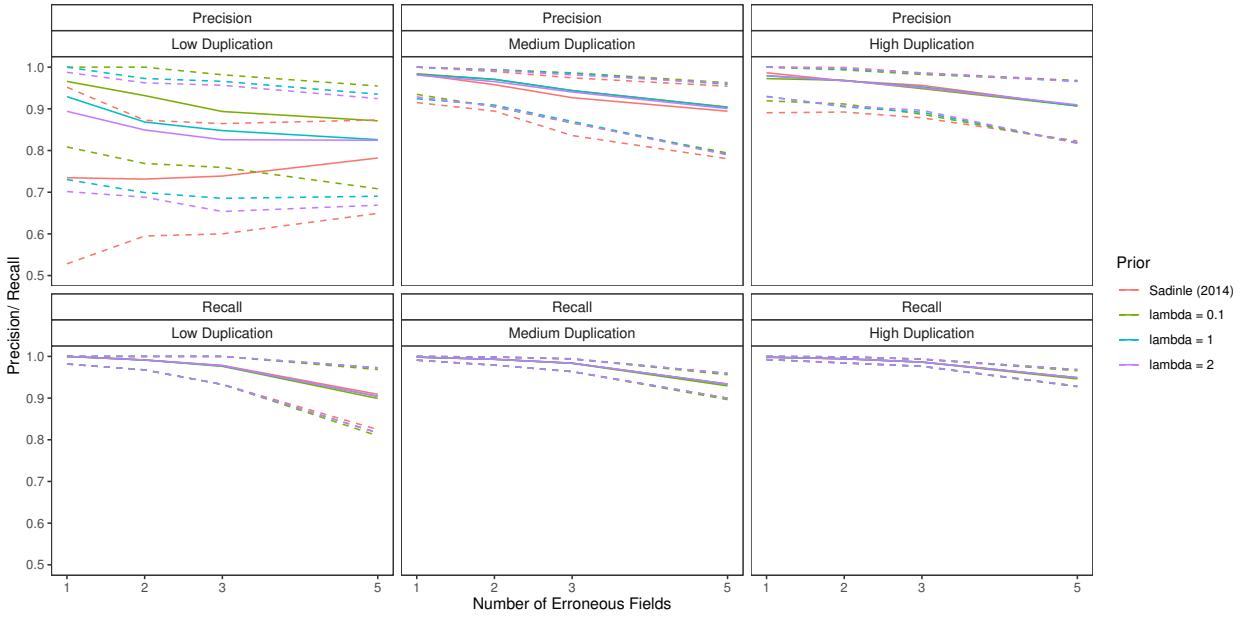


Figure 7: Performance comparison for simulation with datafiles with duplicates and full estimates, with varying priors for the within-file cluster sizes. Solid lines show medians, and dashed lines show 2nd and 98th percentiles.

D.4 Files with Duplicates, Partial Estimates

We now examine the performance of partial estimates in the low duplication setting of the simulation presented in the previous section, where both the proposed approach and the approach of Sadinle (2014) struggled the most. For partial estimates, we use $\lambda_{\text{FNM}} = \lambda_{\text{FM1}} = 1$, $\lambda_{\text{FM2}} = 2$, and $\lambda_A = 0.1$, so that abstaining from making a linkage decision is 10% as costly as making a false non-match. We will assess the performance of the Bayes estimate using precision and the *abstention rate*, $\sum_{i=1}^{r} I(\hat{Z}_i = A)/r$, the proportion of records which the Bayes estimate abstained from making a linkage decision. Recall is no longer useful when using partial estimates, as we are not trying to find all true matches.

In Figure 8 we see that, for both approaches, using partial estimates leads to improved precision in comparison with full estimates, while maintaining a relatively low abstention rate. This result is expected, as the records to which our partial estimate assigns the abstain option are the most ambiguous in terms of which records they should be linked to, and therefore they are the most likely to lead to false matches which decrease the precision. Using partial estimates with an abstain option is therefore a good way of compromising between automated and manual linkage: records for which linkage decisions are difficult are left to be handled via clerical review.

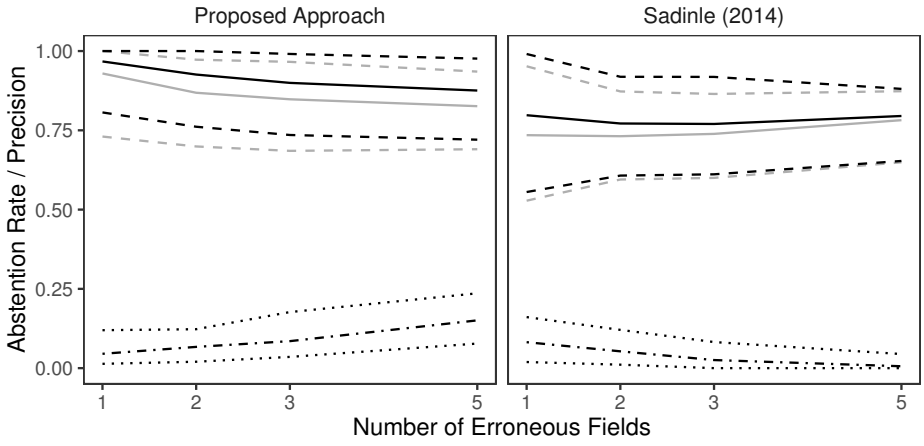


Figure 8: Performance comparison for simulation with datafiles with duplicates and partial estimates. Black solid and dashed lines refer to precision for partial estimates, grey solid and dashed lines refer to precision for full estimates, and dot-dashed and dotted lines refer to the abstention rate for partial estimates. Solid and dot-dashed lines show medians, and dashed and dotted lines show 2nd and 98th percentiles.

D.5 Results for an Alternative Metric

When evaluating the performance of the full estimates in the simulations thus far, we have focused on the metrics of precision and recall, which are global measures of how well the true partition is being estimated. One could also be interested in how well other summaries of the partition are being estimated, e.g. the number of entities (i.e. the number of clusters), the sizes of the clusters, the overlap table, etc. In this section we report the performance of the full estimates in the previous simulations when estimating the number of entities.

For each replicate data set in each simulation scenario considered thus far, we obtained a full estimate of the partition, which can be used to derive an estimate of the number of entities. For a given simulation scenario, let n_0 denote the true number of entities (in all the scenarios considered thus far, $n_0 = 500$), and let $\hat{n}_s$ denote the estimate of the number of entities based on the full estimate of the partition for replicate data set $s \in \{1, \cdots, 100\}$. For each simulation scenario, we can thus estimate the bias of these estimates, $\sum_{s=1}^{100} \hat{n}_s - n_0$, and the mean-squared error of these estimates, $\sum_{s=1}^{100} (\hat{n}_s - n_0)^2$.

D.5.1 Duplicate-Free Files, Equal Errors Across Files

The results for estimating the number of latent entities in the simulations conducted in Section 6.2 of the main text and Appendix D.2 are seen in Figures 9 and 10. We see that that across the different simulation settings the proposed approach has a slight negative bias, and the approach using a flat prior has a very large negative bias. In the no-three-file overlap settings, we see that the more informative prior specifications are less biased than the proposed approach when there are a larger number of erroneous fields, which mirrors the results from Appendix D.2. Across all approaches, the bias increases as the number of erroneous fields increases. The results for the mean-squared error estimates are very similar to the results for the bias estimates.

D.5.2 Duplicate-Free Files, Unequal Errors Across Files

The results for estimating the number of latent entities in the simulation conducted in Section 6.3 of the main text are seen in Figures 9 and 10. We see that that across the two simulation settings all approaches

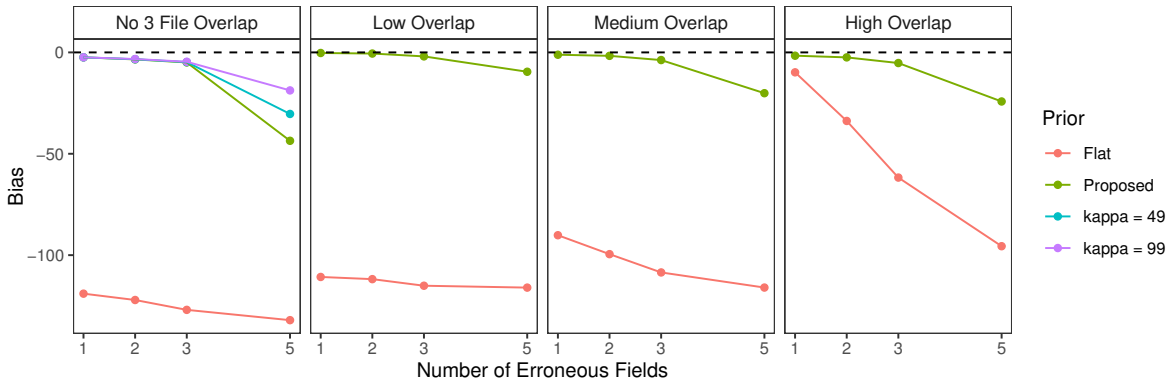


Figure 9: Bias estimates for simulation with duplicate-free files and equal errors across files. "Flat" refers to a flat prior on tripartite matchings, "Proposed" refers to our structured prior for partitions when $\boldsymbol{\alpha} = (1, \cdots, 1)$, and "kappa = 49" and "kappa = 99" refer to the more informative specifications of $\boldsymbol{\alpha}$ discussed in Appendix D.2.

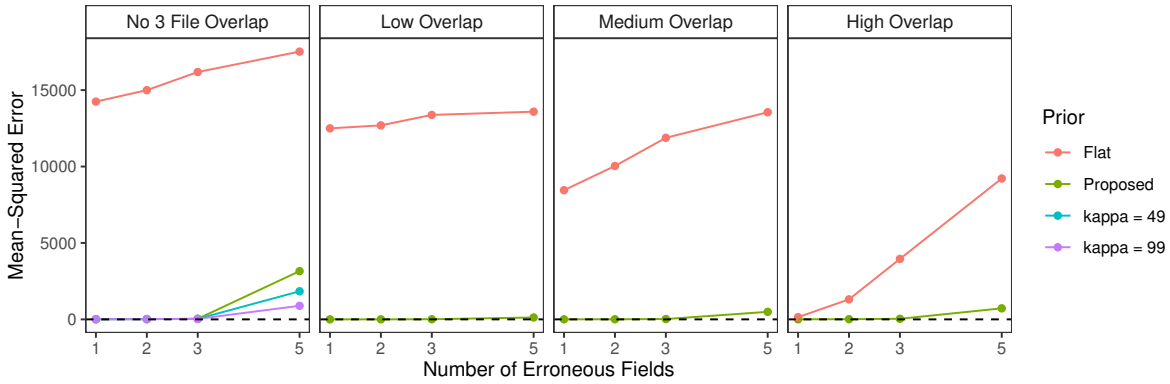


Figure 10: Mean-squared error estimates for simulation with duplicate-free files and equal errors across files. "Flat" refers to a flat prior on tripartite matchings, "Proposed" refers to our structured prior for partitions when $\boldsymbol{\alpha} = (1, \cdots, 1)$, and "kappa = 49" and "kappa = 99" refer to the more informative specifications of $\boldsymbol{\alpha}$ discussed in Appendix D.2.

have a negative bias, with the proposed approach having the smallest bias and the approach using a flat prior having the largest bias in both scenarios. The results for the mean-squared error estimates are very similar to the results for the bias estimates.

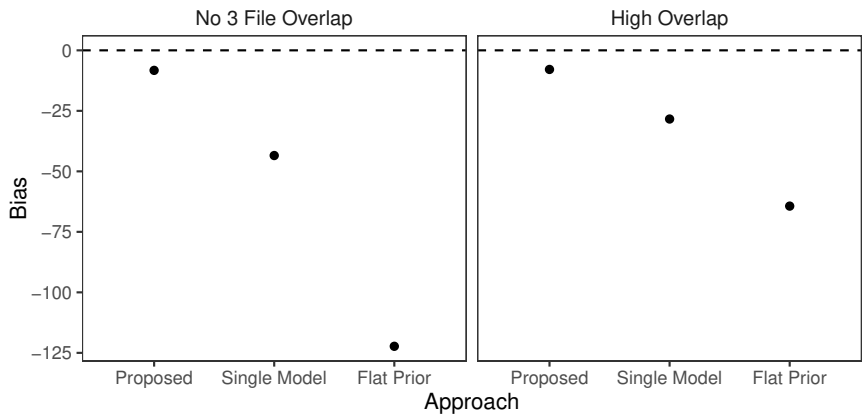


Figure 11: Bias estimates for simulation with duplicate-free files and unequal errors across files. "Proposed" refers to our proposed approach, "Single Model" refers to the approach using a single model for all file-pairs and our structured prior for partitions, and "Flat Prior" refers to the approach using our model for comparison data with a flat prior on tripartite matchings.

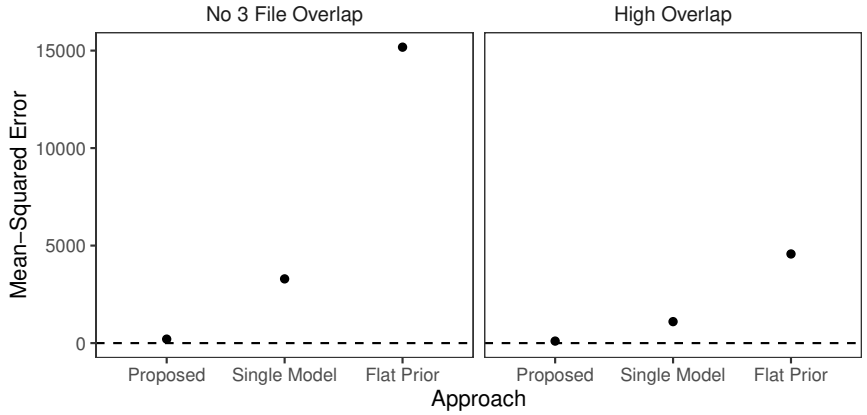


Figure 12: Mean-squared error estimates for simulation with duplicate-free files and unequal errors across files. "Proposed" refers to our proposed approach, "Single Model" refers to the approach using a single model for all file-pairs and our structured prior for partitions, and "Flat Prior" refers to the approach using our model for comparison data with a flat prior on tripartite matchings.

D.5.3 Files with Duplicates, Full Estimates

The results for estimating the number of latent entities in the simulation conducted in Appendix D.3 are seen in Figures 13 and 14. In the low duplication setting, we see that the proposed approach with $\lambda = 0.1$ has the smallest bias across error settings, followed by the proposed approach with $\lambda = 1$, then the proposed

approach with $\lambda = 2$, and then the approach of Sadinle (2014) with the largest bias across error settings. This mirrors the results from Appendix D.3. In the medium and high duplication settings we see that all approaches have a slight negative bias when there are a small number of erroneous fields, and a slight positive bias when there are a large number of erroneous fields. The different variants of the proposed approach all have similar performance in these settings. The proposed approach performs best when there are a small number of erroneous fields, and the approach of Sadinle (2014) performs best when there are a large number of erroneous fields. The results for the mean-squared error estimates are very similar to the results for the bias estimates.

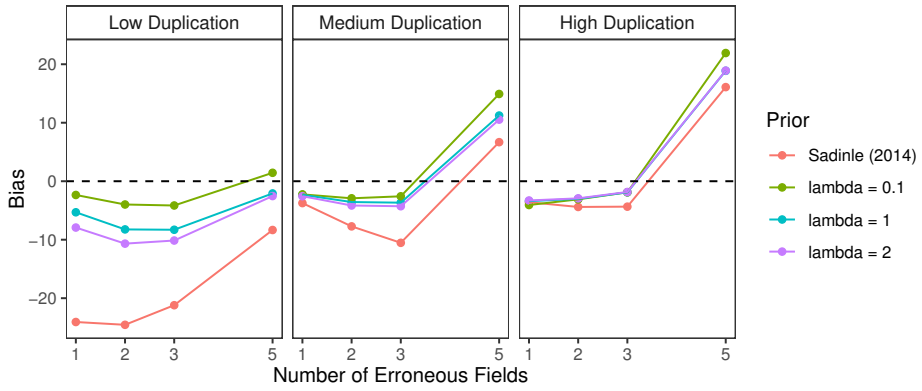


Figure 13: Bias estimates for simulation with files with duplicates and equal errors across files. "Sadinle (2014)" refers to the approach of Sadinle (2014) and "lambda=..." refers to the proposed approach, varying the prior over within-file cluster sizes.

D.6 Simulation Running Times

In this section we present the average running time of our proposed Gibbs sampler across the various simulations settings (i.e. the average time it takes to draw 1000 samples for each simulation setting). The running times are based on the implementation in the R package multilink which can be downloaded on GitHub at `https://github.com/aleshing/multilink`, with the Gibbs sampler described in Appendix B.2 written in C++, running on a laptop with a 3.1 GHz processor.

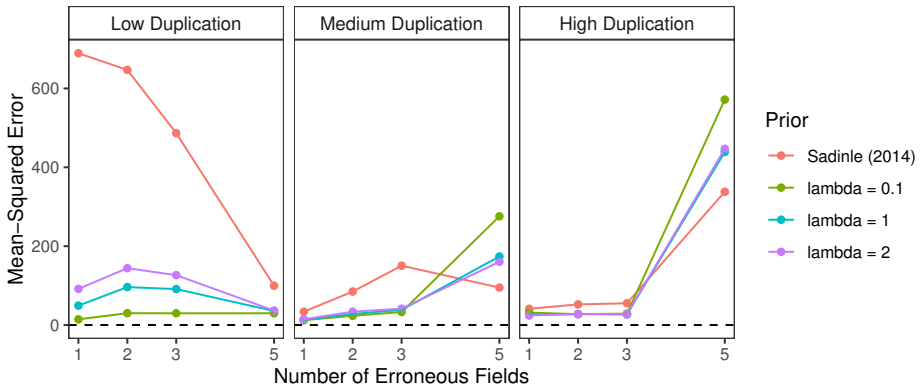


Figure 14: Mean-squared error estimates for simulation with files with duplicates and equal errors across files. "Sadinle (2014)" refers to the approach of Sadinle (2014) and "lambda=..." refers to the proposed approach, varying the prior over within-file cluster sizes.

D.6.1 Duplicate-Free Files, Equal Errors Across Files

The average running times of our approach in the simulations conducted in Section 6.2 of the main text are presented in Table 3. The average number of records was 725 in the settings with no-three-file overlap, 676 in the settings with low overlap, 725 in the settings with medium overlap, and 875 in the settings with high overlap.

Table 3: Average running time in seconds for proposed approach in simulations with duplicate-free files and equal errors across files.

Number of Erroneous Fields	No 3 File Overlap	Low Overlap	Medium Overlap	High Overlap
1	111.0	77.6	83.7	108.6
2	112.9	77.7	83.7	109.2
3	111.8	77.4	83.0	109.4
5	95.7	73.8	77.3	101.1

D.6.2 Duplicate-Free Files, Unequal Errors Across Files

The average running time of our proposed approach in the simulations conducted in Section 6.3 of the main text was 121.8 seconds in the no-three-file overlap setting and 104.1 seconds in the high overlap setting. The

average number of records was 725 in the settings with no-three-file overlap and 875 in the settings with high overlap.

D.6.3 Files with Duplicates, Full Estimates

The average running times of our approach in the simulations conducted in Appendix D.3 are presented in Table 4. The average number of records was 590 in the settings with low duplication, and 889 in the settings with medium duplication, and 1260 in the settings with high duplication. We note here that in this simulation, compared to the simulations with duplicate-free files, we used indexing, which sped up the running time.

Table 4: Average running time in seconds for proposed approach in with files with duplicates and equal errors across files.

Number of Erroneous Fields	Low Duplication	Medium Duplication	High Duplication
1	1.5	8.0	20.9
2	1.1	7.7	20.8
3	1.0	7.4	21.0
5	0.9	6.6	19.2

D.7 Larger Sample Size Simulation

All of the simulations thus far had fixed the true number of latent entities, n, to 500. To explore the running time of our proposed approach further, we now present an additional set of simulations where the number of latent entities varies over $\{100, 500, 1000, 2500\}$. For concreteness we will focus on the simulation setting with duplicate-free files, equal errors across files, medium overlap, and 1 erroneous field per record (i.e. one of the settings from the simulation conducted in Section 6.2 of the main text). For this chosen simulation setting, we repeat the simulation as conducted in Section 6.2 of the main text, varying the number of latent entities over $\{100, 500, 1000, 2500\}$. For the $n = 2500$ setting we conduct 25, rather than 100, replications (due to how long each replication takes). The average number of records was 146 when $n = 100$, 725 when

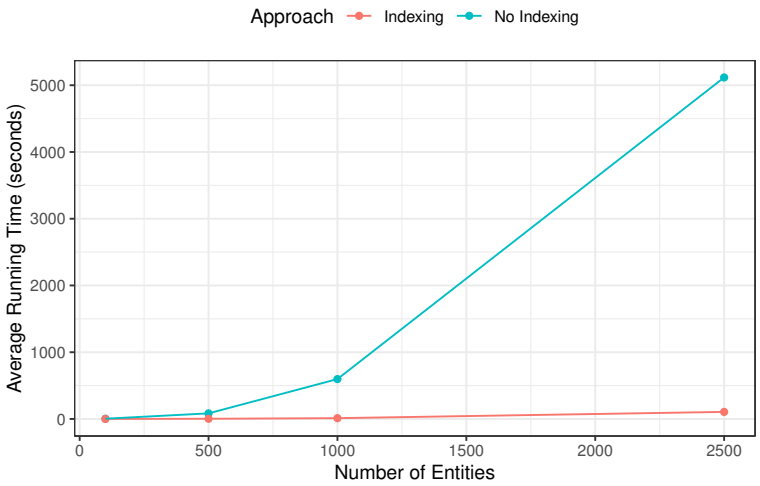


Figure 15: Average running time for simulation varying the number of latent entities.

$n = 500$, 1452 when $n = 1000$, and 3629 when $n = 2500$. In addition to fitting the proposed approach without indexing as in Section 6.2 of the main text, we additionally fit the proposed approach using the indexing scheme described in Appendix D.3 to demonstrate the utility of indexing for improving the running time or our proposed approach, as discussed in Appendix B.4.

The average running time for each setting of the number of entities, n, is presented in Figure 15. Our proposed approach without indexing runs relatively quickly in the settings with $n \in \{100, 500, 1000\}$, for example it only takes around 10 minutes on average to draw 1000 samples when $n = 1000$. However, when $n = 2500$ our proposed approach without indexing runs takes roughly an hour and a half on average to draw 1000 samples. While this running time is manageable, it indicates that the running time in settings with more records than this would prohibitively slow. When looking at the running time of our proposed approach using indexing, we see that the average running is reduced drastically. For example, when $n = 2500$, the average running time is only around 100 seconds.

These results suggest that our approach without indexing can be run in a manageable amount of time for thousands of records, but would be prohibitively slow when the number of records is much larger than a few thousand. Our proposed approach with indexing however can scale to larger file sizes, potentially in the tens of thousands. Using our approach, with or without indexing, in conjunction with blocking, as suggested in Appendix B.4, can help to further scale our approach to large file sizes.

We note that the precision and recall in these simulations, across the different settings of the number of entities, were comparable to the results for the medium overlap, 1 erroneous field per record setting from the simulation results in Section 6.2 of the main text.

D.8 Alternative Loss Function Specifications

In this section explore the impact of varying the specification of the losses $\lambda_{\text{FNM}}, \lambda_{\text{FM1}}, \lambda_{\text{FM2}}$, and λ_A on the performance of our proposed approach across the different simulation settings. For full estimates, we follow Sadinle (2017) and consider the following specifications of the losses $\lambda_{\text{FNM}}, \lambda_{\text{FM1}}$, and λ_{FM2} (with $\lambda_A = \infty$).

A) $\lambda_{\text{FNM}} = 1, \lambda_{\text{FM1}} = 1, \lambda_{\text{FM2}} = 2$. This is the specification used in all of the simulations thus far.

B) $\lambda_{\text{FNM}} = \lambda_{\text{FM1}} = \lambda_{\text{FM2}} = 1$. Compared to specification A, this specification does not penalize type 2 false matches as heavily.

C) $\lambda_{\text{FNM}} = 4, \lambda_{\text{FM1}} = \lambda_{\text{FM2}} = 1$. This specification penalizes false non-matches more heavily than false matches.

D) $\lambda_{\text{FNM}} = 1, \lambda_{\text{FM1}} = 3, \lambda_{\text{FM2}} = 5$. This specification penalizes false matches more heavily than false non-matches, and type 2 false matches more heavily than type 1 false matches.

E) $\lambda_{\text{FNM}} = 1, \lambda_{\text{FM1}} = 2, \lambda_{\text{FM2}} = 3$. Compared to specification E, this specification does not penalizes false matches as heavily.

F) $\lambda_{\text{FNM}} = \lambda_{\text{FM1}} = 1, \lambda_{\text{FM2}} = 4$. Compared to specification A, this specification penalizes type 2 false matches more heavily.

For partial estimates, we consider combining $\lambda_A \in \{0.05, 0.1, 0.25\}$ with the six specifications of $\lambda_{\text{FNM}}, \lambda_{\text{FM1}}$, and λ_{FM2} that we have just introduced.

D.8.1 Duplicate-Free Files, Equal Errors Across Files

The performance of our proposed approach in the simulations conducted in Section 6.2 of the main text, using the different loss function specifications, are seen in Figure 16. So that the figure is easier to scrutinize,

we do not plot the results under specifications E and F, as the results under specifications E are very similar to the results under specification D, and the results under specifications F are very similar to the results under specifications A and B.

We see that when there are a low number of erroneous fields, the results are fairly robust to the loss function specification. When there are a high number of erroneous fields, we see that specification D leads to the highest precision and the lowest recall, specification C leads to the lowest precision and the highest recall, and specifications A and B are somewhere in between. These results are expected, as specification C penalizes false non-matches much more than false matches, and will thus decide to match records more often than the other specifications, and specification D penalizes false matches much more than false non-matches, and thus will decide to match records less often than the other specifications.

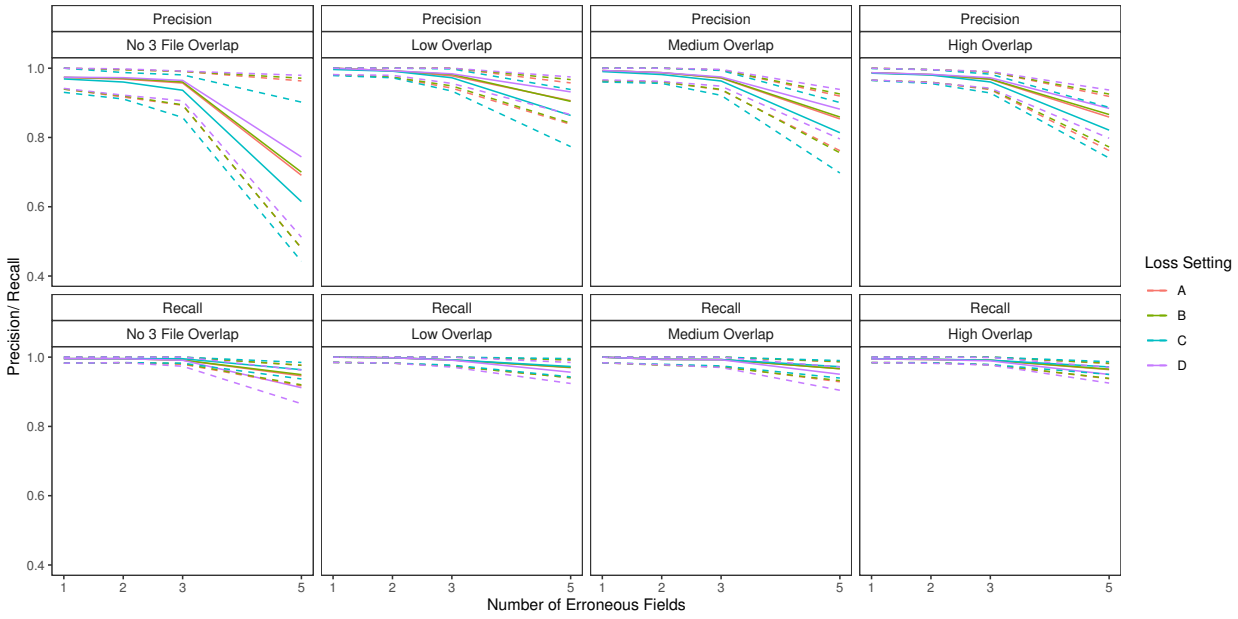


Figure 16: Performance comparison across different loss function specifications for simulation with duplicate-free files and equal measurement error across files. Solid lines show medians, and dashed lines show 2nd and 98th percentiles.

D.8.2 Duplicate-Free Files, Unequal Errors Across Files

The performance of our proposed approach in the simulations conducted in Section 6.3 of the main text, using the different loss function specifications, are seen in Figure 17. Overall the results are fairly robust to

the loss function specification. The results under specifications A, B and F are very similar and the results under specifications D and E are very similar. Similar to the last section, specifications D and E lead to the highest precision and the lowest recall, specification C leads to the lowest precision and the highest recall, and specifications A, B, and F are somewhere in between.

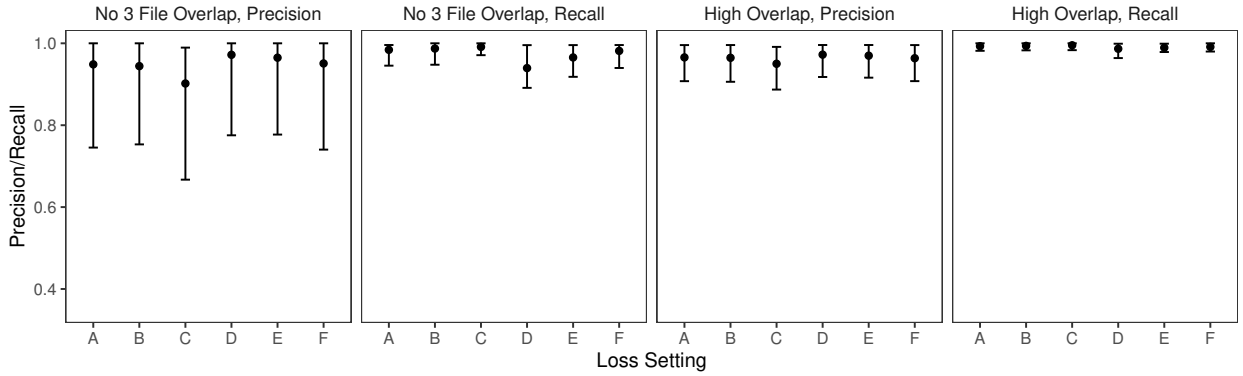


Figure 17: Performance comparison across different loss function specifications for simulation with duplicate-free files and unequal measurement error across files. Solid lines show medians, and dashed lines show 2nd and 98th percentiles.

D.8.3 Files with Duplicates, Full Estimates

The performance of our proposed approach in the simulations conducted in Appendix D.3, using the different loss function specifications, are seen in Figure 16. So that the figure is easier to scrutinize, we do not plot the results under specifications E and F, as the results under specifications E are very similar to the results under specification D, and the results under specifications F are very similar to the results under specifications A and B.

We see that when there is medium and high duplication, the results are fairly robust to the loss function specification. When there is low duplication, we see that specification D leads to the highest precision and the lowest recall, specification C leads to the lowest precision and the highest recall, and specifications A and B are somewhere in between. These results are expected, as described in the previous sections.

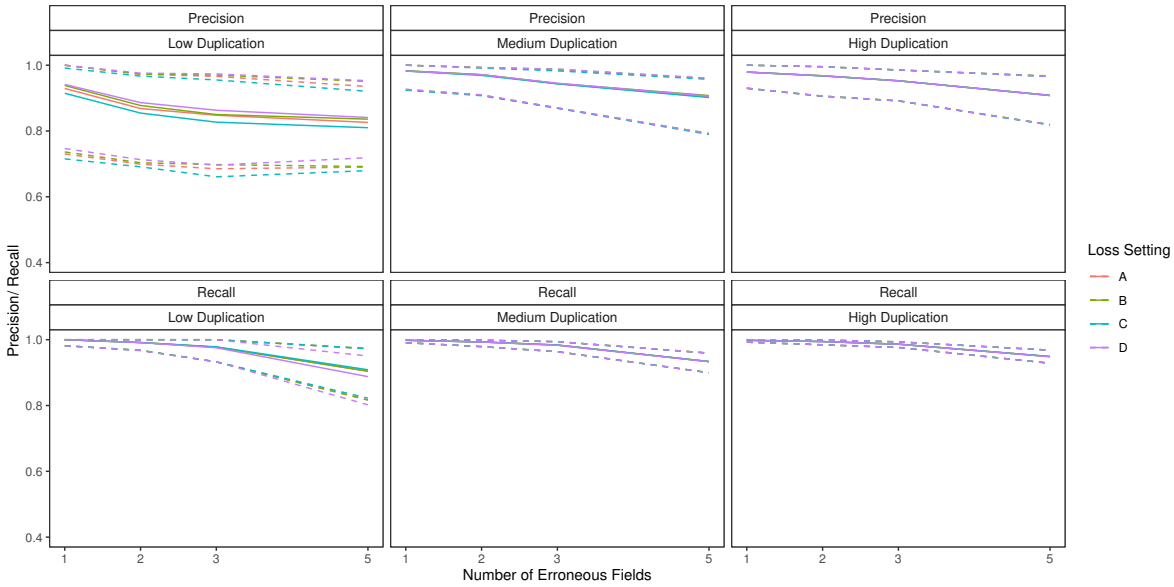


Figure 18: Performance comparison across different loss function specifications for simulation with files with duplicates and equal measurement error across files, when using full estimates. Solid lines show medians, and dashed lines show 2nd and 98th percentiles.

D.8.4 Files with Duplicates, Partial Estimates

The performance of our proposed approach in the simulations conducted in Appendix D.4, using the different loss function specifications, are seen in Figure 19. So that the figure is easier to scrutinize, we do not plot the results under specifications F, as the results under specifications F are very similar to the results under specifications A and B.

We see that as we increase λ_A, the abstention rate decreases and the precision decreases across the different specifications of $\lambda_{\text{FNM}}, \lambda_{\text{FM1}}$, and λ_{FM2}. Further, for each loss specification the abstention rate increases as the number of erroneous fields increases, as there is more uncertainty in the linkage. For a given setting of λ_A, the precision is fairly robust to the different specifications of $\lambda_{\text{FNM}}, \lambda_{\text{FM1}}$, and λ_{FM2}, with specifications D and E having slightly higher precision that specifications A, B and C. For a given setting of λ_A, the abstention rate is highest under specifications D and E, lowest under specifications A and B, with specification C somewhere in between.

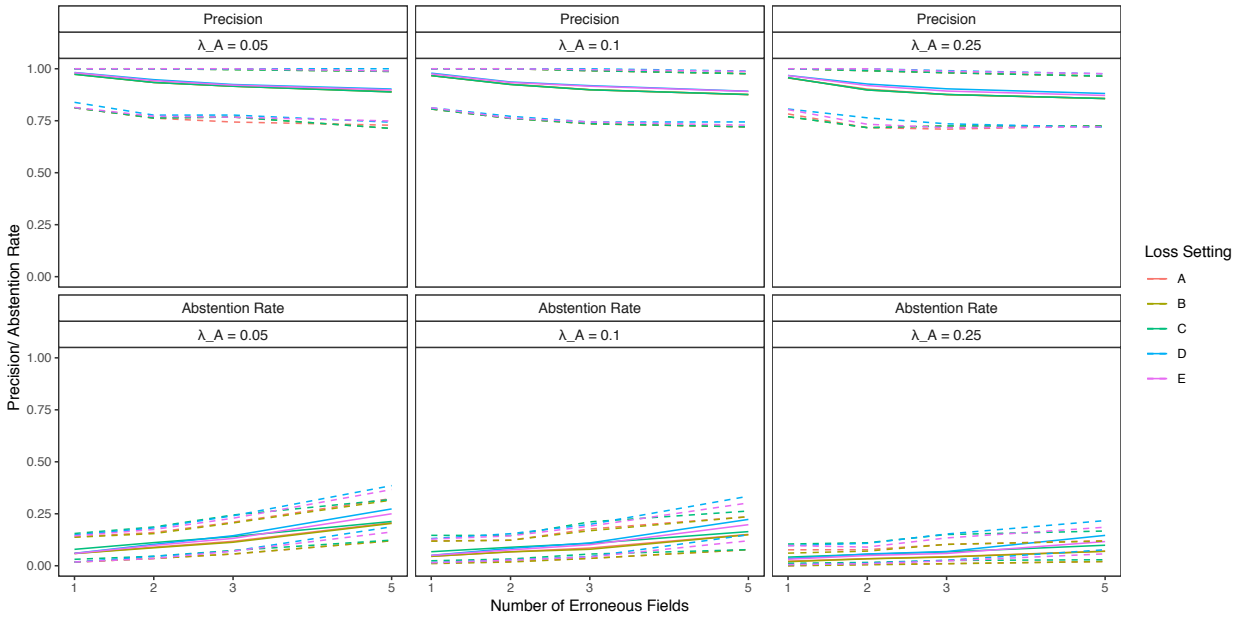


Figure 19: Performance comparison across different loss function specifications for simulation with files with duplicates and equal measurement error across files, when using partial estimates. Solid lines show medians, and dashed lines show 2nd and 98th percentiles.

D.9 Convergence Diagnostics

For all simulations we ran the Gibbs sampler described in Appendix B.2 for $1,000$ iterations, discarding the first 100 samples as burn-in. We initially came up with these sampling and burn-in lengths based on a small number of test runs for each simulation scenario. In particular, for each simulation scenario we ran a small number of test runs and examined the trace plots for the number of entities, n. As the chains for n appeared to converge quickly based on these trace plots, we determined that a burn-in period of 100 samples was appropriate. To illustrate this procedure, for each simulation scenario we now present trace plots for the number of entities, n, for a small number of runs.

D.9.1 Duplicate-Free Files, Equal Errors Across Files

For the simulations conducted in Section 6.2 of the main text with 3 erroneous fields per record, for each overlap setting we present the trace plots for n for the last 5 of the 100 simulation runs in Figure 20. The chains for the other scenarios with $1, 2$, and 5 erroneous fields per record converged similarly quickly.

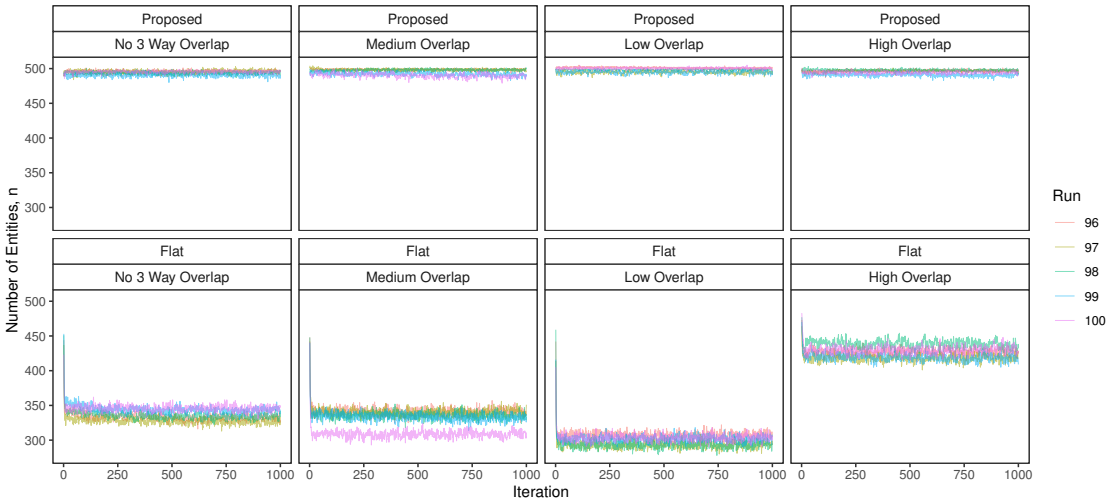


Figure 20: Trace plots for n for the last 5 of 100 runs for the simulation with duplicate-free files and equal measurement error across files. "Proposed" refers to the proposed approach and "Flat" refers to the approach using a flat prior for tripartite matchings.

D.9.2 Duplicate-Free Files, Unequal Errors Across Files

For the simulations conducted in Section 6.3 of the main text, for each overlap setting we present the trace plots for n for the last 5 of the 100 simulation runs in Figure 21.

D.9.3 Files with Duplicates, Full Estimates

For the simulations conducted in Appendix D.3 with 3 erroneous fields per record, for each overlap setting we present the trace plots for n for the last 5 of the 100 simulation runs in Figure 22. The chains for the other scenarios with $1, 2$, and 5 erroneous fields per record converged similarly quickly.

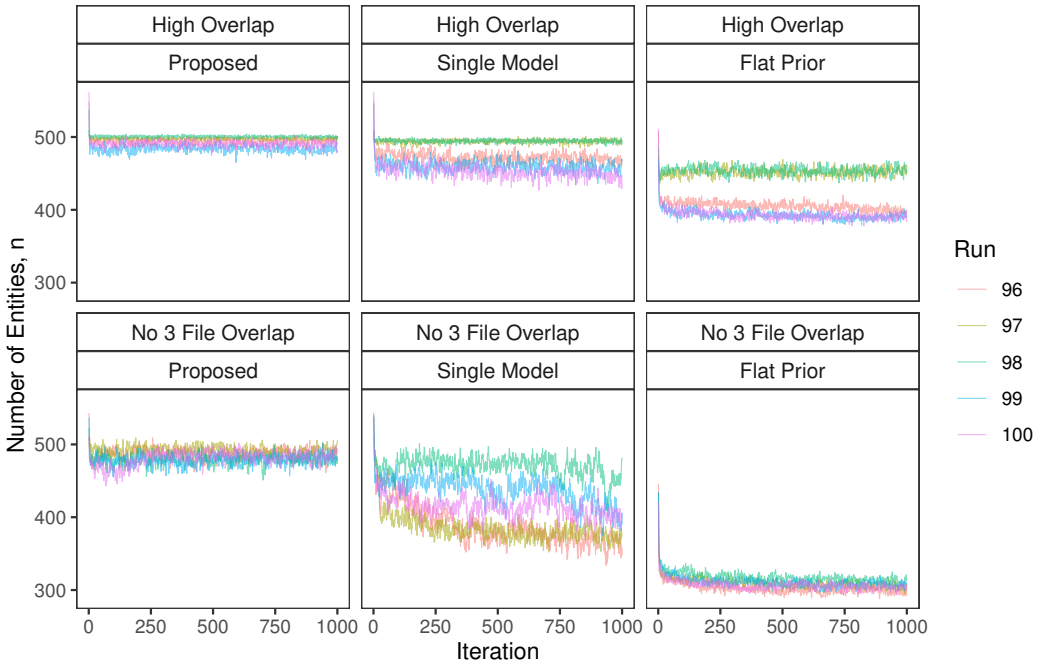


Figure 21: Trace plots for n for the last 5 of 100 runs for the simulation with duplicate-free files and unequal measurement error across files. "Proposed" refers to our proposed approach, "Single Model" refers to the approach using a single model for all file-pairs and our structured prior for partitions, and "Flat Prior" refers to the approach using our model for comparison data with a flat prior on tripartite matchings.

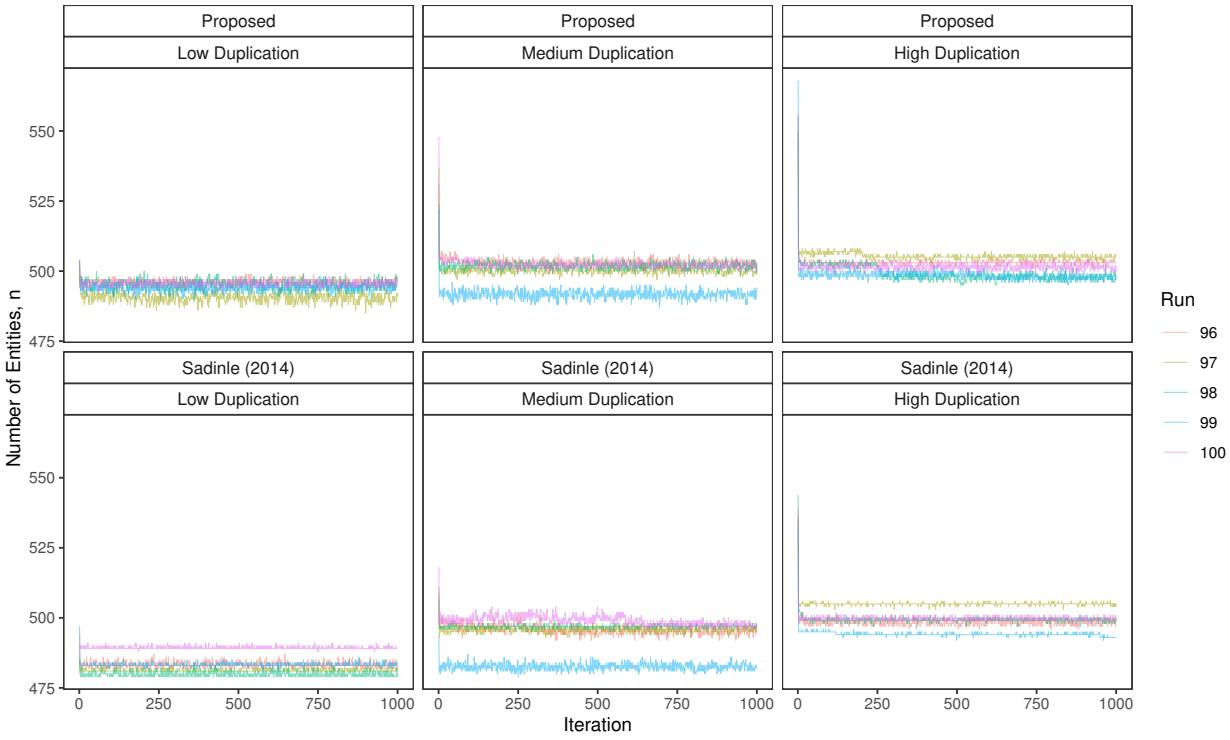


Figure 22: Trace plots for n for the last 5 of 100 runs for the simulation with files with duplicates and equal measurement error across files, when using full estimates. "Proposed" refers to the proposed approach and "Sadinle (2014)" refers to the approach of Sadinle (2014).

E Colombia Application

In this appendix we re-examine the three record systems containing information on homicides from 2004 in the Quindio province of Colombia, previously studied in Sadinle & Fienberg (2013). These record systems, provided by the Conflict Analysis Resource Center (CERAC), were maintained by the Colombian National Statistics Office (Departamento Administrativo Nacional de Estadistica, DANE), the Colombian National Police (Policia Nacional de Colombia, PN), and the Colombian Forensics Institute (Instituto Nacional de Medicina Legal y Ciencias Forenses, ML). While the purpose of DANE is to record all homicides occurring in Colombia, PN and ML only record homicides obtained from their daily activities (Departamento Administrativo Nacional de Estadisticas, DANE 2009, Restrepo & Aguirre 2007). Linking DANE to PN and ML is thus an important step in assessing the coverage of DANE and arriving at better estimates of the number of homicides in Colombia. Previously the records were linked by hand, which gives us a ground truth to assess the performance of our proposed approach. The linkage methodology of Sadinle & Fienberg (2013) did not scale to a large number of records, so the authors restricted their analysis to the 162 records from the last three months of 2004. We will now use our proposed approach to link all the 769 records from 2004.

E.1 Implementation Details

The three record systems are believed to be free of duplicates, so the target of inference is a tripartite matching. The fields available from all three record systems are municipality and date of the homicide, whether the location of the homicide was urban or rural, and the age, sex, and marital status of the victim. Additionally, educational status of the victim is available in DANE and ML, which we are able to use despite it being missing in PN, as explained in Section 4. Although we have seven fields available for the linkage, none of them provide a large amount of discriminative information, which comparison-based approaches rely on. Thus we expect there to be significant uncertainty in the linkage, making the proposed approach particularly relevant.

There are $r_1 = 323$ records in DANE, $r_2 = 157$ records in ML, and $r_3 = 289$ records in PN, so there are $189,431$ record pairs for which we construct comparison data, according to Table 5. We use transitive indexing as described in Appendix B, where the initial indexing scheme declares record pairs as non-coreferent

Table 5: Construction of levels of disagreement for the Colombian homocide record systems.

		Levels of disagreement			
Field	**Similarity measure**	**0**	**1**	**2**	**3**
Date	Absolute Difference	0	$1-2$	$3-7$	8+
Age	Absolute Difference	0	$1-2$	$3-9$	10+
Other Fields	Binary comparison	Agree	Disagree		

if they disagree in either municipality, sex, date by more than 60 days, or age by more than 9 years. This reduces the number of candidate coreferent record pairs down to $60,324$.

We present results from our approach under two prior specifications. The first is the default specification used in the simulations in Sections 6.2 and 6.3. The second specification differs from the default by placing a more informative prior on the overlap table through $\boldsymbol{\alpha}$. In particular, based on characteristics of the record systems described in Restrepo & Aguirre (2007), we specify a prior that captures the beliefs that: 1) if a homicide is recorded in PN or ML, it is highly likely to also be recorded in DANE, and 2) DANE and PN are expected to have a high coverage of homicides. This prior is described in more detail in the following section. We used the same loss function specification as outlined in Section 6.1 and Appendix D. We ran $3,000$ iterations of the Gibbs sampler presented in Appendix B, discarding the first $1,000$ as burn-in. In Section E.5 we discuss convergence of the Gibbs sampler for this application.

E.2 An Informative Prior Specification

We will guide our prior specification using the following two passages from Restrepo & Aguirre (2007) (translated to English):

- "According to DANE, 'The differences in the Legal Medicine and DANE data are mainly due to the fact that the latter organization receives, in addition to the death certificates sent by Legal Medicine (which are sent to DANE after a technical examination), the homicide reports made by police inspectors, nurses or health promoters - in places where there are no legal doctors - who arrive at the site where the body is found and register the cases as homicide without a technical examination and according

to the picture that presents the corpse'. So we find here a reason for the increased coverage of vital statistics" (Restrepo & Aguirre 2007, p. 331).

- "The National Police assures to have coverage at the national level, making an institutional presence in all the municipalities of the country: 'We have a presence throughout the country and we know all the cases; Legal Medicine does not have the same coverage and thus their data is not exact' " (Restrepo & Aguirre 2007, p. 330).

Note that our Dirichlet-Multinomial prior for the overlap table can be motivated as the result of first drawing $\{q_{\boldsymbol{h}}\}_{\boldsymbol{h}\in\mathcal{H}}$ from a Dirichlet distribution with hyperparameters $\boldsymbol{\alpha}$, then drawing $\boldsymbol{n}$ from a multinomial distribution of size n with probabilities $\{q_{\boldsymbol{h}}\}_{\boldsymbol{h}\in\mathcal{H}}$. Based on these passages we specify $\boldsymbol{\alpha}$ as follows, referring in our notation to DANE as list 1, ML as list 2, and PN as list 3:

- If a homicide is going to be recorded by one of the three record systems, it is very likely that it will be known by DANE. Therefore, we choose $\alpha_{1++} = \alpha_{100}+\alpha_{101}+\alpha_{110}+\alpha_{111}$ and $\alpha_{0++} = \alpha_{001}+\alpha_{010}+\alpha_{011}$ such that $\text{mode}(q_{1++}) = 0.95$ and $\mathbb{P}(q_{1++} > 0.9) = 0.95$, where $q_{1++} = q_{100} + q_{101} + q_{110} + q_{111} \sim \mathsf{Beta}(\alpha_{1++}, \alpha_{0++})$ is the prior probability of a homicide being recorded in DANE given it is recorded in one of the three systems.

- If a homicide is recorded by PN, then it is very likely that it will be recorded by DANE. Therefore, we choose $\alpha_{1+1} = \alpha_{101} + \alpha_{111}$ and $\alpha_{0+1} = \alpha_{001} + \alpha_{011}$ such that $\text{mode}(q_{1+1}) = 0.95$ and $P(q_{1+1} > 0.9) = 0.9$, where $q_{1+1} = q_{101} + q_{111} \sim \mathsf{Beta}(\alpha_{1+1}, \alpha_{0+1})$ is the prior probability of a homicide being recorded in DANE given it is recorded in PN.

- If a homicide is recorded by ML, then it is very likely that it will be known by DANE. Therefore, we choose $\alpha_{11+} = \alpha_{110} + \alpha_{111}$ and $\alpha_{01+} = \alpha_{010} + \alpha_{011}$ such that $\text{mode}(q_{11+}) = 0.95$ and $P(q_{11+} > 0.9) = 0.9$, where $q_{11+} = q_{110} + q_{111} \sim \mathsf{Beta}(\alpha_{11+}, \alpha_{01+})$ is the prior probability of a homicide being recorded in DANE given it is recorded in ML.

- The above induce six constraints, which determine α_{011}, α_{001}, and α_{010}, but we need one extra constraint to determine the remaining α. We thus choose the configuration of α_{100}, α_{101}, α_{110}, and α_{111}

that maximizes α_{101}, which controls the prior probability of a homicide being jointly recorded by DANE and PN, given that these systems that are supposed to have the largest coverage of homicides.

Under this specification we arrive at the setting $\alpha_{001} = 1.63, \alpha_{010} = 1.63, \alpha_{011} = 2.94, \alpha_{100} = 0, \alpha_{101} = 30.96, \alpha_{110} = 30.96, \alpha_{111} = 37.78$. For the sake of propriety, we set $\alpha_{100} = 0.1$.

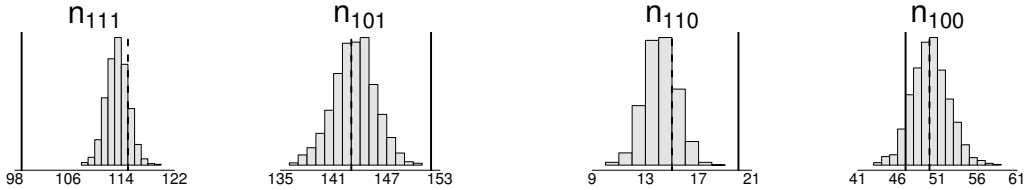
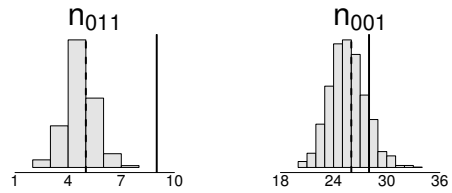
E.3 Results

Under the default prior specification the precision and recall of the full estimate are 0.90 and 0.93 respectively. Recall is no longer useful when using partial estimates, as we are not trying to find all true matches. Thus we will assess the performance of the partial estimates using precision and the *abstention rate*, the proportion of records which the Bayes estimate abstained from making a linkage decision. The partial estimate has an abstention rate of 11%, and improves the precision of the estimate to 0.93. Under the informative prior specification the precision and recall of the full estimate are 0.93 and 0.96 respectively. The partial estimate has an abstention rate of 11%, and improves the precision of the estimate to 0.95. Due to the performance difference, we will focus on the results under the informative prior specification for the rest of this section. Analogous results under the default specification are provided in the next section.

The total number of homicides based on the hand labelling is 383. Under the informative prior specification a 95% credible interval for the number of unique homicides n is $[372, 383]$, with an estimate based on the full estimate of the tripartite matching of 376. In Table 6 we display the posterior distribution for the overlap table and the overlap table derived from the full estimate, along with the overlap table derived from the ground truth hand labelling. We can see that n_{111}, the number of homicides recorded in all three files, and n_{100}, the number of homicides recorded in just DANE, are overestimated, and the remaining cells of the overlap table (and n) are underestimated.

While the performance of the estimated matching is fairly good, with precision of both the full and partial estimate (before clerical review) above 0.9, the overestimation of n_{111} and the underestimation of n is indicative of over-matching. We believe this over-matching occurrs due to the low amount of discriminative information provided by the fields. In particular, the only fields we believe can be fully trusted are municipality of the homicide (of which there are 12) and sex of the victim (which is coded as binary), which can only

Table 6: Posterior distribution of the overlap table for the Colombian record systems, under the informative prior specification. Black lines indicate the ground truth, dotted lines indicate quantities derived from the full estimate of the tripartite matching.

	In PN		Out PN	
DANE	In ML	Out ML	In ML	Out ML
In				
Out			–	

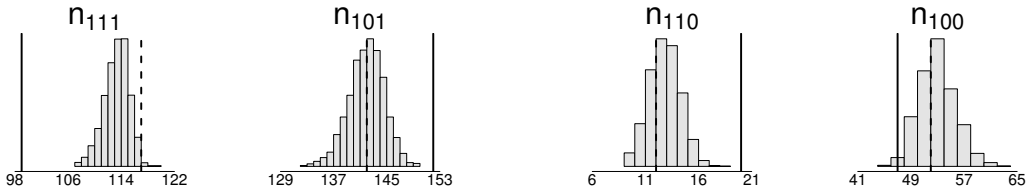
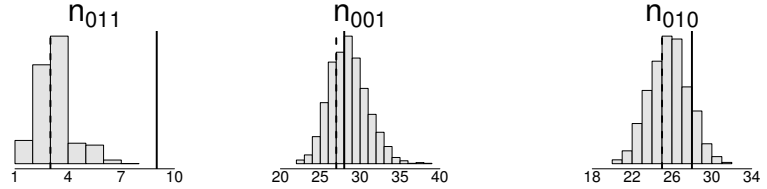
partition the records into 22 blocks of candidate coreferent records (there are no records of female homicide victims in two municipalities). The remaining fields all have some amount of error and do not provide highly discriminative information since they are either low dimensional categorical fields (urban/rural location of the homicide has two categories, marital status has five categories, and educational status has six categories) or numeric fields that are essentially ordinal categorical (date of homicide and age of victim). Therefore, records of different homicides may look similar based on the comparisons of these fields, causing them to be mistakenly matched. In these low-information settings, clerical review becomes especially important for the record linkage workflow, which our proposed approach of using partial estimates incorporates by design.

E.4 Results Under Default Prior

Under the default prior specification a 95% credible interval for the number of unique homicides n is $[376, 389]$, with an estimate based on the full estimate of the tripartite matching of 378. Thus we see that n is better estimated under the default prior compared to the informative prior (though the point estimate is still an underestimate). In Table 7 we display the posterior distribution for the overlap table and the overlap table

derived from the full estimate, along with the overlap table derived from the ground truth hand labelling. We can see that, as with the informative prior specification, n_{111} and n_{100} are overestimated, and the remaining cells of the overlap table (and n) are underestimated.

Table 7: Posterior distribution of the overlap table for the Colombian record systems, under the default prior specification. Black lines indicate the ground truth, dotted lines indicate quantities derived from the full estimate of the tripartite matching.

	In PN		Out PN	
DANE	In ML	Out ML	In ML	Out ML
In				
Out				–

E.5 Convergence Diagnostics

In the application, we ran the the Gibbs sampler described in Appendix B.2 for $3,000$ iterations, discarding the first $1,000$ samples as burn-in, under a default and an informative prior specification. In Figure 23 we present the the trace plots for the number of entities, n, under each of these prior specifications. The chains for n appear to converge quickly based on these trace plots. For each of these chains we computed Geweke's convergence diagnostic as implemented in the `R` package `coda` (Plummer et al. 2006). The Geweke's Z-scores indicated it was reasonable to treat these chains as drawn from their stationary distributions.

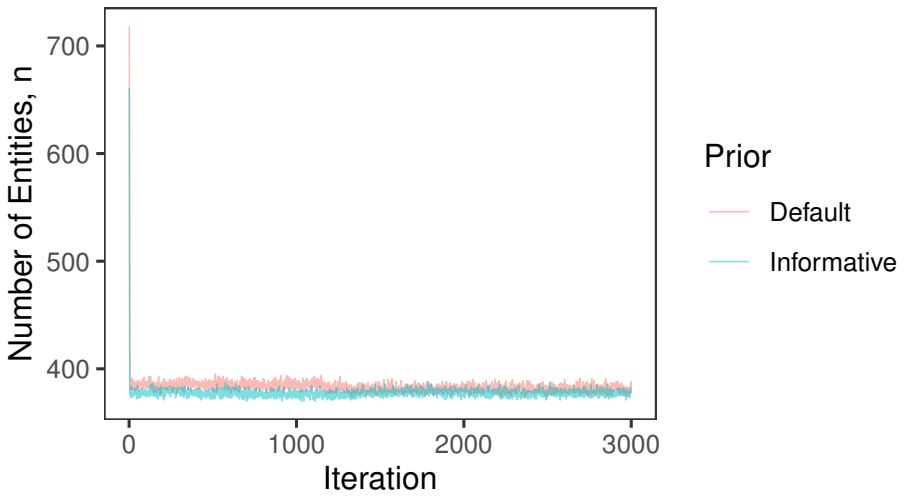


Figure 23: Trace plots for n in Colombia application.

References

Ball, P. & Price, M. (2019), 'Using statistics to assess lethal violence in civil and inter-state war', *Annual review of statistics and its application* **6**, 63–84.

Bansal, N., Blum, A. & Chawla, S. (2004), 'Correlation Clustering', *Machine Learning* **56**(1), 89–113.

Berger, J. O., Bernardo, J. M. & Sun, D. (2015), 'Overall objective priors', *Bayesian Analysis* **10**(1), 189–221.

Betancourt, B., Sosa, J. & Rodríguez, A. (2020), 'A Prior for Record Linkage Based on Allelic Partitions', *arXiv preprint arXiv:2008.10118* .

Betancourt, B., Zanella, G. & Steorts, R. C. (2020), 'Random Partition Models for Microclustering Tasks', *arXiv preprint arXiv:2004.02008* .

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.

Binder, D. A. (1978), 'Bayesian cluster analysis', *Biometrika* **65**(1), 31–38.

Binette, O. & Steorts, R. C. (2020), '(Almost) All of Entity Resolution', *arXiv preprint arXiv:2008.04443* .

Bird, S. M. & King, R. (2018), 'Multiple systems estimation (or capture-recapture estimation) to inform public policy', *Annual review of statistics and its application* **5**, 95–118.

Brandes, U., Delling, D., Gaertler, M., Görke, R., Hoefer, M., Nikoloski, Z. & Wagner, D. (2007), On Finding Graph Clusterings with Maximum Modularity, *in* 'Graph-Theoretic Concepts in Computer Science', Springer Berlin Heidelberg, pp. 121–132.

Christen, P. (2012), 'A Survey of Indexing Techniques for Scalable Record Linkage and Deduplication', *IEEE Transactions on Knowledge and Data Engineering* **24**(9), 1537–1555.

Demaine, E. D., Emanuel, D., Fiat, A. & Immorlica, N. (2006), 'Correlation clustering in general weighted graphs', *Theoretical Computer Science* **361**(2), 172 – 187.

Departamento Administrativo Nacional de Estadisticas, DANE (2009), Metodologia Estadisticas Vitales, Technical Report 82, Direccion de Censos y Demografia.
URL: *http://www.dane.gov.co/files/investigaciones/fichas/Estadisticas_vitales.pdf*

Dunson, D. B. & Xing, C. (2009), 'Nonparametric Bayes modeling of multivariate categorical data', *Journal of the American Statistical Association* **104**(487), 1042–1051.

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.

Enamorado, T. & Steorts, R. C. (2020), Probabilistic Blocking and Distributed Bayesian Entity Resolution, *in* 'International Conference on Privacy in Statistical Databases', Springer, pp. 224–239.

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

Fortini, M., Liseo, B., Nuccitelli, A. & Scanu, M. (2001), 'On Bayesian record linkage', *Research in Official Statistics* **4**(1), 185–198.

Herbei, R. & Wegkamp, M. H. (2006), 'Classification with reject option', *Canadian Journal of Statistics* **34**(4), 709–721.

Hof, M. H., Ravelli, A. C. & Zwinderman, A. H. (2017), 'A probabilistic record linkage model for survival data', *Journal of the American Statistical Association* **112**(520), 1504–1515.

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.

Klami, A. & Jitta, A. (2016), Probabilistic size-constrained microclustering, *in* 'Proceedings of the Thirty-Second Conference on Uncertainty in Artificial Intelligence', AUAI Press, pp. 329–338.

Lancichinetti, A. & Fortunato, S. (2009), 'Community detection algorithms: A comparative analysis', *Phys. Rev. E* **80**, 056117.

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

Larsen, M. D. & Rubin, D. B. (2001), 'Iterative Automated Record Linkage Using Mixture Models', *Journal of the American Statistical Association* **96**(453), 32–41.

Liseo, B. & Tancredi, A. (2011), 'Bayesian Estimation of Population Size via Linkage of Multivariate Normal Data Sets', *Journal of Official Statistics* **27**(3), 491–505.

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* pp. 1–16.

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

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* .

Meilă, M. (2007), 'Comparing clusterings—an information based distance', *Journal of multivariate analysis* **98**(5), 873–895.

Miller, J., Betancourt, B., Zaidi, A., Wallach, H. & Steorts, R. C. (2015), 'Microclustering: When the cluster sizes grow sublinearly with the size of the data set', *arXiv preprint arXiv:1512.00792* .

Murray, J. S. (2015), 'Probabilistic record linkage and deduplication after indexing, blocking, and filtering', *Journal of Privacy and Confidentiality* **7**(1).

Newman, M. E. J. (2013), 'Spectral methods for community detection and graph partitioning', *Phys. Rev. E* **88**, 042822.

Plummer, M., Best, N., Cowles, K. & Vines, K. (2006), 'CODA: convergence diagnosis and output analysis for MCMC', *R news* **6**(1), 7–11.

Restrepo, J. A. & Aguirre, K. (2007), 'Homicidios y Muertes Violentas: Un Analisis Comparativo de las Fuentes en Colombia', *Forensis: Datos Para La Vida* .

Sadinle, M. (2014), 'Detecting duplicates in a homicide registry using a Bayesian partitioning approach', *The Annals of Applied Statistics* **8**(4), 2404–2434.

Sadinle, M. (2017), 'Bayesian estimation of bipartite matchings for record linkage', *Journal of the American Statistical Association* **112**(518), 600–612.

Sadinle, M. & Fienberg, S. E. (2013), 'A Generalized Fellegi-Sunter Framework for Multiple Record Linkage With Application to Homicide Record Systems', *Journal of the American Statistical Association* **108**(502), 385–397.

Steorts, R. C. (2015), 'Entity Resolution with Empirically Motivated Priors', *Bayesian Analysis* **10**(4), 849–875.

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* 'International Conference on Privacy in Statistical Databases', Springer, pp. 253–268.

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

Tancredi, A., Steorts, R. & Liseo, B. (2020), 'A Unified Framework for De-Duplication and Population Size Estimation (with Discussion)', *Bayesian Analysis* **15**(2), 633–682.

Tran, K.-N., Vatsalan, D. & Christen, P. (2013), GeCo: an online personal data generator and corruptor, *in* 'Proceedings of the 22nd ACM International Conference on Information & Knowledge Management', pp. 2473–2476.

Wade, S. & Ghahramani, Z. (2018), 'Bayesian cluster analysis: Point estimation and credible balls (with discussion)', *Bayesian Analysis* **13**(2), 559–626.

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.

Winkler, W. E. (1994), Advanced Methods for Record Linkage, *in* 'Proceedings of the Section on Survey Research Methods', American Statistical Association, pp. 467–472.

Zanella, G., Betancourt, B., Miller, J. W., Wallach, H., Zaidi, A. & Steorts, R. C. (2016), Flexible models for microclustering with application to entity resolution, *in* 'Advances in Neural Information Processing Systems', pp. 1417–1425.

Zhou, J., Bhattacharya, A., Herring, A. H. & Dunson, D. B. (2015), 'Bayesian factorizations of big sparse tensors', *Journal of the American Statistical Association* **110**(512), 1562–1576.