1 DIRICHLET RECORD LINKAGE

We provide updated notation to allow us to describe one record in B having multiple matches in A. Let Z_j be a set containing the indices for all of the records in A that are a match with record B_j, and let $Z = \{Z_j | j = 1, \ldots, n_B\}$ denote the collection of all such set. Let $|Z_j| = \sum_{k=1}^{\infty} I(Z_{j,k}) > 0$ denote the number of links associated with record B_j.

We can allow each record in B to match to multiple records in A through a Dirichlet process prior. Define a vector of probabilities $\boldsymbol{\pi} = (\pi_0, \ldots)$ where π_k is the probability that some record in B has exactly k matches in A. In implementation, we model each π_k as a product of conditional probabilities: let η_k be the probability that some record in B has at least k matches, given that it has at least $k-1$ matches. This gives us the stick breaking representation

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

where η_k are independent random variables from a Beta$(\alpha_\eta, \beta_\eta)$ distribution.

Similar to fabl, we adopt a prior specification on Z so that each matching Z_j of length $|Z_j| = k$ is equally likely. Formally, we have

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

$$\boldsymbol{m}_f \sim \text{Dirichlet}(\alpha_{f1}, \ldots, \alpha_{fL_f}), \forall f = 1, \ldots, F, \tag{2b}$$

$$\boldsymbol{u}_f \sim \text{Dirichlet}(\beta_{f1}, \ldots, \beta_{fL_f}), \forall f = 1, \ldots, F, \tag{2c}$$

$$p(Z_j = q | \boldsymbol{\pi}) = \begin{cases} \frac{(n_A - k)!}{n_A!} \pi_k, & q \neq 0, \\ \pi_0, & q = 0; \end{cases} \tag{2d}$$

$$\pi_k = (1 - \eta_{k+1}) \prod_{c=1}^{k} \eta_c, \tag{2e}$$

$$\eta_k \sim \text{Beta}(\alpha_\eta, \beta_\eta). \tag{2f}$$

1.1 Derivation of Joint Distribution

We first derive the Gibbs sampler for a the joint distribution of the set Z_j. Following the observation of Wortman (2019) and elaborated by Kundinger et al. (2024), when B_j does not link to any record in A (such that $|Z_j| = 0$) the contribution to the likelihood is simply a product of u parameters, which we will call c_j:

$$p(\Gamma_{\cdot j}|\boldsymbol{m}, \boldsymbol{u}, \pi, Z_j = n_A + j) = \prod_{i=1}^{n_A} \prod_{f=1}^{F} \prod_{l=1}^{L_f} u_{fl}^{I(\gamma_{ij}^f = l) I_{obs}(\gamma_{ij}^f)} = c_j. \tag{3}$$

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

$$p(\Gamma_{\cdot j}|\boldsymbol{m}, \boldsymbol{u}, \pi, Z_j = q) = \prod_{i \in q} \prod_{f=1}^{F} \prod_{l=1}^{L_f} m_{fl}^{I(\gamma_{ij}^f = l) I_{obs}(\gamma_{ij}^f)} \prod_{i \notin q} \prod_{f=1}^{F} \prod_{l=1}^{L_f} u_{fl}^{I(\gamma_{ij}^f = l) I_{obs}(\gamma_{ij}^f)}. \tag{4}$$

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

$$p(\Gamma_{\cdot j}|\boldsymbol{m}, \boldsymbol{u}, \pi, Z_j = q) = \prod_{i \in q} \prod_{f=1}^{F} \prod_{l=1}^{L_f} \left(\frac{m_{fl}}{u_{fl}} \right)^{I(\gamma_{ij}^f = l) I_{obs}(\gamma_{ij}^f)} \prod_{i=1}^{n_A} \prod_{f=1}^{F} \prod_{l=1}^{L_f} u_{fl}^{I(\gamma_{ij}^f = l) I_{obs}(\gamma_{ij}^f)} \tag{5}$$

$$= c_j \prod_{i \in q} w_{ij}. \tag{6}$$

Lastly, we multiply the likelihood by the prior π_K and divide by c_j to obtain the full conditional

$$p\left(Z_j^{(s+1)} = q | \gamma, \boldsymbol{m}^{(s+1)}, \boldsymbol{u}^{(s+1)}, \pi^{(s+1)}\right) \propto \begin{cases} \frac{(n_A - k)!}{n_A!} \pi_k^{(s+1)} \prod_{i \in q} w_{ij}^{(s+1)}, & |q| = k; \\ \pi_0^{(s+1)}, & |q| = 0. \end{cases} \tag{7}$$

1.2 Sequential Sampler

Sampling this joint distribution is computational prohibitive as the number of records in A grows. In particular, when allowing B_j to match to up to K records, there are $\sum_{k=1}^{K} \frac{n_A!}{(n_A - K)!}$ possible options for the set Z_j. Furthermore, if we were to sample this joint distribution directly, we would need to choose K ahead of time in order to probably enumerate these potential options. Through Gibbs sampling however, we can break this joint distribution into a sequence of more simple conditional univariate distributions. This allows for a more

computationally efficient sampler, and allows us to learn K from the data, rather than set it ahead of time.

We generalize the fast beta prior from Kundinger et al. (2024) to sequence of priors that allow for multiple matchings. When B_j has been linked to $k-1$ records, we say that the probability that B_j has a k^{th} match is η_k, and that all remaining records in A are equally likely to be linked. We use

$$p(Z_{j,k} = q_k|\eta_k) = \begin{cases} \frac{\eta_k}{n_{j,k}}, & q_k \in N_{j,k}, \\ 1-\eta_k, & z_{j,k} = \emptyset; \end{cases} \tag{8}$$

$$\eta_k \sim \text{Beta}(\alpha_\eta, \beta_\eta) \tag{9}$$

where $N_{j,k} = [n_A] \setminus (Z_{j,1}, \ldots, Z_{j,k-1})$ is the set of records in A that are available to be matched with B_j, and $n_{j,k} = |N_{j,k}| = n_A - (k+1)$ is the number of such records.

This sequence of priors leads to sequence of posteriors that can be used to sample arbitrarily many links for record B_j. Specifically,

$$p(Z_{j,k} = q_k|Z_{j,k-1}, \eta_k, \boldsymbol{m}, \boldsymbol{u}, \gamma) = \begin{cases} \frac{\eta_k}{n_{j,k}} w_{q_k,j}, & q_k \in N_{j,k}, \\ \frac{1-\eta_k}{,} & q_k = \emptyset; \end{cases} \tag{10}$$

$$\eta_k \sim \text{Beta}(\alpha_\eta + n_k(Z), \beta_\eta + n_{k-1}(Z) - n_k(Z)) \tag{11}$$

where $n_k(Z) = \sum_{j=1}^{n_B} I(|Z_j| \geq k)$ is the number of records in B that have at least k matches in A. Note that $n_0(Z) = n_B$, and that for each k, we can view $n_{k-1}(Z)$ as a number of trials, and $n_k(Z)$ as a number of successes, and thus update η_k as a straightforward Beta random variable.

Observe that the joint posterior for the set Z_j provided in (7) is proportional to the

product of sequential posteriors provided in (10). For $|q| = k$, we have

$$
\begin{aligned}
p\left(Z_j^{(s+1)} = q | \gamma, \boldsymbol{m}^{(s+1)}, \boldsymbol{u}^{(s+1)}, \pi^{(s+1)}\right) &\propto \frac{(n_A - k)!}{n_A!} \pi_k^{(s+1)} \prod_{i \in q} w_{ij}^{(s+1)} \\
&= \prod_{c=1}^{k} \frac{1}{n_A - (c+1)} (1 - \eta_{k+1}) \prod_{c=1}^{k} \eta_k \prod_{c=1}^{k} w_{q_c,j} \\
&= (1 - \eta_{k+1}) \prod_{c=1}^{k} \frac{\eta_k}{n_A - (c+1)} \prod_{c=1}^{k} w_{q_c,j} \\
&= p(Z_{j,k+1} = \emptyset) \prod_{c=1}^{k} p(Z_{j,c} = q_c | \eta_c) p(\Gamma_{\cdot j} | \boldsymbol{m}, \boldsymbol{u}, \boldsymbol{\eta}, q_c \in Z_j) \\
&\propto p(Z_{j,k+1} = \emptyset | \Gamma_{\cdot j}, \boldsymbol{m}, \boldsymbol{u}, \boldsymbol{\eta}) \prod_{c=1}^{k} p(Z_{j,c} = q_c | \Gamma_{\cdot j}, \boldsymbol{m}, \boldsymbol{u}, \boldsymbol{\eta}).
\end{aligned}
$$

Thus, this sequence of priors induces an extension of `fabl` with an iterative matching phase. In each iteration of the Gibbs sampler, we sample an initial set of links using η_1. For each record in B that was found to have a link, we remove the linked record in A from consideration, and then sample another potential link with η_2. We continue, using η_k in the k^{th} matching step, until no new links are found, at which we point the matching phase terminates. The $\boldsymbol{\eta}, \boldsymbol{m}$, and $\boldsymbol{u}$ parameters are estimated based on all of the links identified. Crucially, there is no need to specify a maximum number of links per record, as this estimated through the model.

2 SIMULATIONS

We demonstrate the accuracy of the multiple match approach at various levels of overlap between files and errors between matching records through an adaptation of the simulation study from Sadinle (2017) and Kundinger et al. (2024).

For each simulation, we construct two data files A and B such that there are in which there are 25, 125, or 225 records in B that have matching records in A. The matching record in A exhibits 1, 2, or 3 errors accross the five fields used for linkage. Then, every record in A that has a matching record in B is duplicated, such that each simulation has 50, 250, or 450 matching record pairs in total. We use uniform priors for the $\boldsymbol{m}$ and $\boldsymbol{u}$, with $\alpha_{fl} = \beta_{fl} = 1$ for all f and l. We use uniform priors for π for standard `fabl`, and also

uniform priors for each η_k for multiple match. (We correctly specify the prior for multilink. (Need to write provide more description of model to make this succinct)) We run the Gibbs sampler for 1000 iterations, and discard the first 100 as burn-in.

In these simulations, standard `fabl` drastically under performs, so much so that results are omitted from the graph. For each record in B with matching records in A, the posterior match probability is split between the two matching records. Due to the randomness of the Gibbs sampler, one of the records occassionally has a posterior probability over 0.5 and is thus identified through the Bayes estimate, but often, both records have posterior probability below 0.5. In through `vabl`, the posterior probability of the two records is cut precisely in half, and thus `vabl` did not identify any matching record pairs in any of the simulations.

The performance of `multilink` is more varied. When there is a moderate or high amount of matching records, `multilink` performs comparably with multiple match. However, in the setting where the number of matching records is low, `multilink` under performs. This may be because the model being fit is more complex, and so it is difficult to learn the parameters from such few observations. Noteably however, the performance of `multilink` is highly sensitive to prior specification. In FIGURE (to be added), we show accuracy suffers under various forms of mispecification.

The "fabl swap" approach here is also strong. It generally has higher recall, but lower precision. This makes sense, because it doesn't need to consecutive matching algorithim in order to find the "multiple matches", but it also doesn't employ any post processing to clean up erroneous matches. In all simulations, multiple match and fabl swap have similar f-measure. (Not sure exactly how I want to analyze that. In NCVR, there is more of a difference.)

We see that multiple match is a strong alternative to multiple match in this setting. In particular, the average computation time for 1000 iterations of the Gibbs sampler for multiple match was around 30 seconds, while it was around 1000 seconds for `multiple match`.

NOTE: I've included fastlink (without Jaro) in this image for the time being. However,

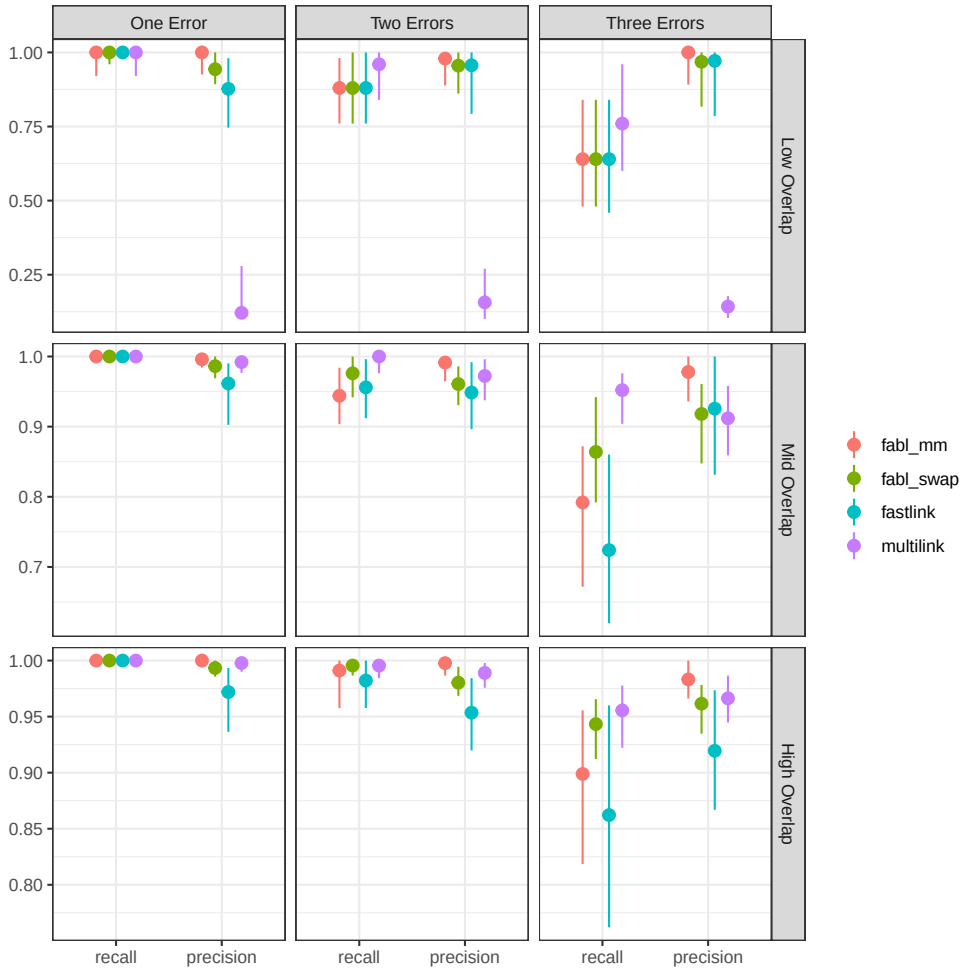

Figure 1: Sadinle Simulation

there is currently no way through fastlink to produce estimates that respect the 2-to-1 matchings while enforcing transitivity.

3 CASE STUDY

We demonstrate our method on the North Carolina Voter Registration (`NCVR`) database taken two months apart (**?**). The snapshots are filtered to include only those voters whose details changed over the two-month period, so there are matching records with full agreement on all fields. We use first name, middle name, last name, as string fields, and street address and age as categorical fields. Unique voter registration numbers are provided,

method	recall	precision	f_measure
fabl	0.9743739	0.9798799	0.9771191
fabl_mm	0.9891623	0.9735309	0.9812843
fastLink	0.9988472	0.8494865	0.9181321
fastLink_jaro	0.9514490	0.9664229	0.9588775

Table 1: Accuracy results for `NCVR`, showing the `DRL` has the highest f-measure.

however they are known to contain some errors. The `NCVR` dataset is not publicly available due to sensitive information. However, we have permission to utilize it for publication by its owner.

Using the voter registration numbers, we can see that each file has internal duplication rates of about 1%. In this analysis, we deduplicate file B, and left A with duplicates. In practice, such low amount of internal duplication may not warrant the use of Dirichlet Beta Linkage since `vabl` is considerably faster. However, we demonstrate here that even with such few internal duplications, `DBL` is effective at identifying this multiple matches and does not declare too many false matches.

I compare four approaches: standard `fabl`, fabl with the `DBL` method, `fastLink`, and `fastLink` with the Jaro correction. For each method, links are declared with a 0.5 probability cut off, and fabl, fabl_mm, use the post processing to ensure transitivity. Results are in Table 1.

We see that `DRL` has the highest f-measure, followed by fabl. When we use `fastLink` with no post-processing, we end up with an undesirable amount of false positives. Since the posterior match probability of each record pair is computed independently, we expect such behavior on large linkage tasks such as this. When we use the Jaro post-processing to achieve a one-to-one matching, we get considerably better results. However, we lose the possibility of identifying cases where one record in B matches to two records in A, and we still have worse recall and precision than attained under standard `fabl`.

REFERENCES

Aleshin-Guendel, S. & Sadinle, M. (2023), 'Multifile partitioning for record linkage and duplicate detection', *Journal of the American Statistical Association* **0**(0), 1–10.
URL: *https://doi.org/10.1080/01621459.2021.2013242*

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

Marchant, N. G., Kaplan, A., Elazar, D. N., Rubinstein, B. I. P. & Steorts, R. C. (2021), 'd-blink: Distributed end-to-end bayesian entity resolution', *Journal of Computational and Graphical Statistics* **30**(2), 406–421.
URL: *https://doi.org/10.1080/10618600.2020.1825451*

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

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

4 APPENDIX