Global update to $q(u)$. Due to non-conjugacy, our surrogate objective $\mathcal{L}$ has no closed-form update to $q(u)$. Instead, we employ numerical optimization to update vectors $\hat{\rho}, \hat{\omega}$ simultaneously:

$$\arg\max_{\hat{\rho},\hat{\omega}} \mathcal{L}_{\text{hdp-local}}(\hat{\rho}, \hat{\omega}, \hat{\theta}, \hat{s}) + \mathcal{L}_{\text{hdp-global}}(\hat{\rho}, \hat{\omega}) \quad \text{subject to } \hat{\omega}_k > 0, \hat{\rho}_k \in (0,1) \text{ for } k = 1, 2 \dots K.$$

Details are in the supplement. The update to $q(u)$ requires expectations under $q(\pi)$, and vice versa, so it can be useful to iteratively optimize $q(\pi)$ and $q(u)$ several times given fixed local statistics.

To handle large datasets, we can adapt these updates to perform *stochastic variational inference* (SVI) [19]. Stochastic algorithms perform local updates on random subsets of sequences (batches), and then perturb global parameters by following a noisy estimate of the natural gradient, which has a simple closed form. SVI has previously been applied to non-sticky HDP-HMMs with point-estimated β [7], and can be easily adapted to our more principled objective. One drawback of SVI is the requirement of a learning rate schedule, which must typically be tuned to each dataset.

3.4 Memoized Variational Inference

We now outline a memoized algorithm [13] for our sticky HDP-HMM variational objective. Before execution, each sequence is randomly assigned to one of B batches. The algorithm repeatedly visits batches one at a time in random order; we call each full pass through the complete set of B batches a *lap*. At each visit to batch b, we perform a local step for all sequences n in batch b and then a global step. With $B = 1$ batches, memoized inference reduces to the standard full-dataset algorithm, while with larger B we have more affordable local steps and faster overall convergence. With just one lap, memoized inference is equivalent to the synchronous version of *streaming variational inference*, presented in Alg. 3 of Broderick et al. [20]. We focus on regimes where dozens of laps are feasible, which we demonstrate dramatically improves performance.

Affordable, but exact, batch optimization of $\mathcal{L}$ is possible by exploiting the additivity of statistics M, S. For each statistic we track a batch-specific quantity M^b, and a whole-dataset summary $M \triangleq \sum_{b=1}^{B} M^b$. After a local step at batch b yields $\hat{s}^b, \hat{r}^b$, we update $M^b(\hat{s}^b)$ and $S^b(\hat{r}^b)$, *increment* each whole-dataset statistic by adding the new batch summary and subtracting the summary stored in memory from the previous visit, and store (or *memoize*) the new statistics for future iterations. This update cycle makes M and S consistent with the most recent assignments for *all* sequences. Memoization does require $\mathcal{O}(BK^2)$ more storage than SVI. However, this cost does not scale with the number of sequences N or length T. Sparsity in transition counts M may make storage cheaper.

At any point during memoized execution, we can evaluate $\mathcal{L}$ exactly for all data seen thus far. This is possible because nearly all terms in Eq. (6) are functions of only global parameters $\hat{\rho}, \hat{\omega}, \hat{\theta}, \hat{\tau}$ and sufficient statistics M, S. The one exception that requires local values $\hat{s}, \hat{r}$ is the entropy term $\mathcal{L}_{\text{entropy}}$. To compute it, we track a $(K+1) \times K$ matrix H^b at each batch b:

$$H^b_{0\ell} = -\sum_n \hat{r}_{n1\ell} \log \hat{r}_{n1\ell}, \quad H^b_{k\ell} = -\sum_n \sum_{t=1}^{T_n - 1} \hat{s}_{ntk\ell} \log \tfrac{\hat{s}_{ntk\ell}}{\hat{r}_{ntk}}, \tag{12}$$

where the sums aggregate sequences n that belong to batch b. Each entry of H^b is non-negative, and given the whole-dataset entropy matrix $H = \sum_{b=1}^{B} H^b$, we have $\mathcal{L}_{\text{entropy}} = \sum_{k=0}^{K} \sum_{\ell=1}^{K} H_{k\ell}$.

4 State Space Adaptation via Birth, Merge, and Delete Proposals

Reliable nonparametric inference algorithms must quickly identify and create missing states. Split-merge samplers for HDP topic models [10, 11] are limited because proposals can only split an existing state into two new states, require expensive traversal of all data points to evaluate an acceptance ratio, and often have low acceptance rates [12]. Some variational methods for HDP topic models also dynamically create new topics [16, 21], but do not guarantee improvement of the global objective and can be unstable. We instead interleave stochastic birth proposals with delete and merge proposals, and use memoization to efficiently verify proposals via the exact full-dataset objective.

Birth proposals. Birth moves can create many new states at once while maintaining the monotonic increase of the whole-dataset objective, $\mathcal{L}$. Each proposal happens within the local step by trying to improve $q(z_n)$ for a single sequence n. Given current assignments $\hat{s}_n, \hat{r}_n$ with truncation K, the move proposes new assignments $\hat{s}'_n, \hat{r}'_n$ that include the K existing states and some new states with index $k > K$. If $\mathcal{L}$ improves under the proposal, we accept and use the expanded set of states for all remaining updates in the current lap. To compute $\mathcal{L}$, we require candidate global parameters $\hat{\rho}', \hat{\omega}', \hat{\theta}', \hat{\tau}'$. These are found via a global step from candidate summaries M', S', which combine