amplified to be at full-dataset scale: $\hat{\lambda}_k^* = \lambda_0 + \frac{N}{|\mathcal{B}_t|} s_k(\mathcal{B}_t)$. Then, we interpolate between this and the previous global parameters to arrive at the final result: $\hat{\lambda}_k^{(t)} \leftarrow \rho_t \hat{\lambda}_k^* + (1-\rho_t)\hat{\lambda}_k^{(t-1)}$. The learning rate ρ_t controls how "forgetful" the algorithm is of previous values; if it decays at appropriate rates, stochastic inference provably converges to a *local* optimum of the global objective $\mathcal{L}(q)$ [1].

This online approach has clear computational advantages and can sometimes yield higher quality solutions than the full-data algorithm, since it conveys information between local and global parameters more frequently. However, performance is extremely sensitive to the learning rate decay schedule and choice of batch size, as we demonstrate in later experiments.

3 Memoized online variational inference

Generalizing previous incremental variants of the *expectation maximization* (EM) algorithm [10], we now develop our *memoized online variational inference* algorithm. We divide the data into B fixed batches $\{\mathcal{B}_b\}_{b=1}^B$. For each batch, we maintain *memoized* sufficient statistics $S_k^b = [\hat{N}_k(\mathcal{B}_b), s_k(\mathcal{B}_b)]$ for each component k. We also track the full-dataset statistics $S_k^0 = [\hat{N}_k, s_k(\mathbf{x})]$. These compact summary statistics allow guarantees of correct full-dataset analysis while processing only one small batch at a time. Our approach hinges on the fact that these sufficient statistics are *additive*: summaries of an entire dataset can be written exactly as the addition of summaries of distinct batches. Note that our memoization of deterministic analyses of batches of data is distinct from the stochastic memoization, or "lazy" instantiation, of random variables in some Monte Carlo methods [11, 12].

Memoized inference proceeds by visiting (in random order) each distinct batch once in a full pass through the data, incrementally updating the local and global parameters related to that batch b. First, we update local parameters for the current batch ($q(z_n \mid \hat{r}_n)$ for $n \in \mathcal{B}_b$) via Eq. (9). Next, we update cached global sufficient statistics for each component: we subtract the old (cached) summary of batch b, compute a new batch-level summary, and add the result to the full-dataset summary:

$$S_k^0 \leftarrow S_k^0 - S_k^b, \quad S_k^b \leftarrow \Big[\sum_{n \in \mathcal{B}_b} \hat{r}_{nk}, \sum_{n \in \mathcal{B}_b} \hat{r}_{nk} t(x_n) \Big], \quad S_k^0 \leftarrow S_k^0 + S_k^b. \tag{11}$$

Finally, given the new full-dataset summary S_k^0, we update global parameters exactly as in Eq. (10). Unlike stochastic online algorithms, memoized inference is guaranteed to improve the full-dataset ELBO at every step. Correctness follows immediately from the arguments in [10]. By construction, each local or global step will result in a new q that strictly increases the objective $\mathcal{L}(q)$.

In the limit where $B = 1$, memoized inference reduces to standard full-dataset updates. However, given many batches it is far more scalable, while maintaining all guarantees of batch inference. Furthermore, it generally converges faster than the full-dataset algorithm due to frequently interleaving global and local updates. Provided we can store memoized sufficient statistics for each batch (*not* each observation), memoized inference has the same computational complexity as stochastic methods while avoiding noise and sensitivity to learning rates. Recent analysis of convex optimization algorithms [13] demonstrated theoretical and practical advantages for methods that use cached full-dataset summaries to update parameters, as we do, instead of stochastic current-batch-only updates.

This memoized algorithm can compute the full-dataset objective $\mathcal{L}(q)$ exactly at any point (after visiting all items once). To do so efficiently, we need to compute and store the assignment entropy $H_k^b = -\sum_{n \in \mathcal{B}_b} r_{nk} \log r_{nk}$ after visiting each batch b. We also need to track the full-data entropy $H_k^0 = \sum_{b=1}^B H_k^b$, which is additive just like the sufficient statistics and incrementally updated after each batch. Given both H_k^0 and S_k^0, evaluation of the full-dataset ELBO in Eq. (8) is exact and rapid.

3.1 Birth moves to escape local optima

We now propose additional *birth* moves that, when interleaved with conventional coordinate ascent parameter updates, can add useful new components to the model and escape local optima. Previous methods for changing variational truncations create just one extra component via a "split" move that is highly-specialized to particular likelihoods. Wang and Blei [15] explore truncation levels via a local collapsed Gibbs sampler, but samplers are slow to make large changes. In contrast, our births add *many* components at once and apply to *any* exponential family mixture model.