$$\begin{aligned} U_{i\ell} &= \sum_{j=1}^{N} \mathbb{E}_q[s_{ij\ell}] + \sum_{j=1}^{N} \mathbb{E}_q[r_{ji\ell}] \\ &= \sum_{j=1}^{N} \sum_{m=1}^{K} (\hat{\eta}_{ij\ell m} + \hat{\eta}_{jim\ell}) \end{aligned} \tag{28}$$

Again, the usage statistic $U_{i\ell}$ has a natural interpretation as the expected number of times $\pi_{i\ell}$ was "used": it is the expected sum over how often node i was in community ℓ as either a source or a receiver.

Computational Efficiency Although the assortative MMSB is a slightly less expressive model than the full MMSB, it is significantly more computationally efficient. In both the assortative and full MMSB, the matrix $\hat{\eta}$ is of size $O(N^2K^2)$, which can be prohibitively expensive to compute and store. However, due to the fact that the aMMSB only has K "real" community interactions (i.e. those for which $\ell = m$), we can achieve linear complexity in K by only computing terms $\hat{\eta}_{ij\ell m}$ for which $\ell = m$. The details of how this changes the above computations are given in Appendix C.

6 Variational Inference Algorithms

Section 5 outlines the basic variational updates and objective function calculations for our chosen variational approximations. We now introduce specific algorithms that contain various interleavings and variations of these updates.

6.1 Single Batch Variational Inference

Single batch variational inference is the most immediate application of the results from Section 5. We consider the entire dataset at once, and iterate between updating "local" parameters $q(\mathcal{Z})$ and "global" parameters $q(\pi)$, $q(\phi)$, and $q(u)$. We see that the local step updates those factors which scale with the amount of observed data, while the global step scales with the number of hidden states K (albeit $q(\pi)$ scales with the number of nodes in the graph for the HDP-aMMSB). Intuitively, information only "flows" between local variables through the global parameters. This can make batch algorithms require many local steps to converge, which can be prohibitively expensive for a large dataset.

6.2 Memoized Online Variational Inference

To address this issue, we use *memoized online* variational inference [12], which divides the dataset into many smaller batches. After performing a local step on a single batch, we update the global parameters of the entire model and move on to the next batch. Although this does not change the runtime of a single pass through the dataset, it decreases the overall number of passes required by promoting more rapid exchange of information between batches.

Batch Definitions. To use memoized inference, we need a way of segmenting data into batches. For the HDP-HMM, we set each batch to be a collection of sequences, where a local step corresponds to running the forward-backward algorithm on each sequence in the batch. Although we do not do so here, it is possible to define batches as small subsets of larger sequences [7].

For the HDP-aMMSB, we consider a batch to be some subset of nodes and all their outgoing relationships. That is, for a set of indicies $I \subseteq \{1, \ldots, N\}$, a batch is all pairs $\{(i, j) : i \in I, j \in \{1, \ldots, N\}, j \neq i\}$. This definition is primarily for convenience of implementation; more sophisticated strategies that define batches as arbitrary subsets of edges have previously found success [10, 14].

Merge and Delete Moves. Although not discussed here, a major benefit of memoized algorithms is their ability to vary the truncation level K as the algorithm progresses [12, 11]. This often allows them to find higher quality and more compact models than competing methods.