Meta-Learning MCMC Proposals Supplementary Materials

Tongzhou Wang[*]
Facebook AI Research
tongzhou.wang.1994@gmail.com

Yi Wu
University of California, Berkeley
jxwuyi@gmail.com

David A. Moore[†]
Google
davmre@gmail.com

Stuart J. Russell
University of California, Berkeley
russell@cs.berkeley.edu

S-1 Experiment Details

As mentioned in main paper Sec. 4, parametrizing MDNs in all experiments have elu activation and two hidden layers each of size $\lambda \max\{\text{input size}, \text{output size}\}$, where $4 \leq \lambda \leq 5$ depending on the task, and output the proposal distribution as a mixture of $4 \leq m \leq 16$ components.

S-1.1 General Binary-Valued Grid Models

For directed binary-valued grid models, we use higher amount of MDN mixtures than other experiments because more variables are proposed and general discrete BNs can have highly multi-modal conditionals.

Architecture The underlying MDN has 106-480-480-120 network structure, mapping the CPTs of all 23 motif variables and 14 conditioning variable values to the proposal distribution of 9 proposed variables as a mixture of 12 components.

Additional Sample Runs We provide additional sample runs on four grid models with various percentages of deterministic relations in Fig. 1.

S-1.2 Gaussian Mixture Model with Unknown Number of Components

Formal Model Formally, the model can be written as

$$\begin{aligned}
M &\sim \text{Unif}\{1, 2, \ldots, m\} \\
\mu_j &\sim \mathcal{N}(0, \sigma_\mu^2 I) && j = 1, \ldots, m \\
\mathbf{v}|M &\sim \text{Unif}\{a \in \{0,1\}^m : \sum_j a_j = M\} \\
z_i|\mathbf{v} &\sim \text{Unif}\{j : v_j = 1\} && i = 1, \ldots, n \\
x_i|z_i, \boldsymbol{\mu} &\sim \mathcal{N}(\mu_{z_i}, \sigma^2 I) && i = 1, \ldots, n,
\end{aligned}$$

where m and n are model parameters, and $\sigma_\mu^2 = 4$ and $\sigma^2 = 0.1$ are fixed constants.

[*]Work done while the author was at the University of California, Berkeley
[†]Work done while the author was at the University of California, Berkeley

32nd Conference on Neural Information Processing Systems (NeurIPS 2018), Montréal, Canada.

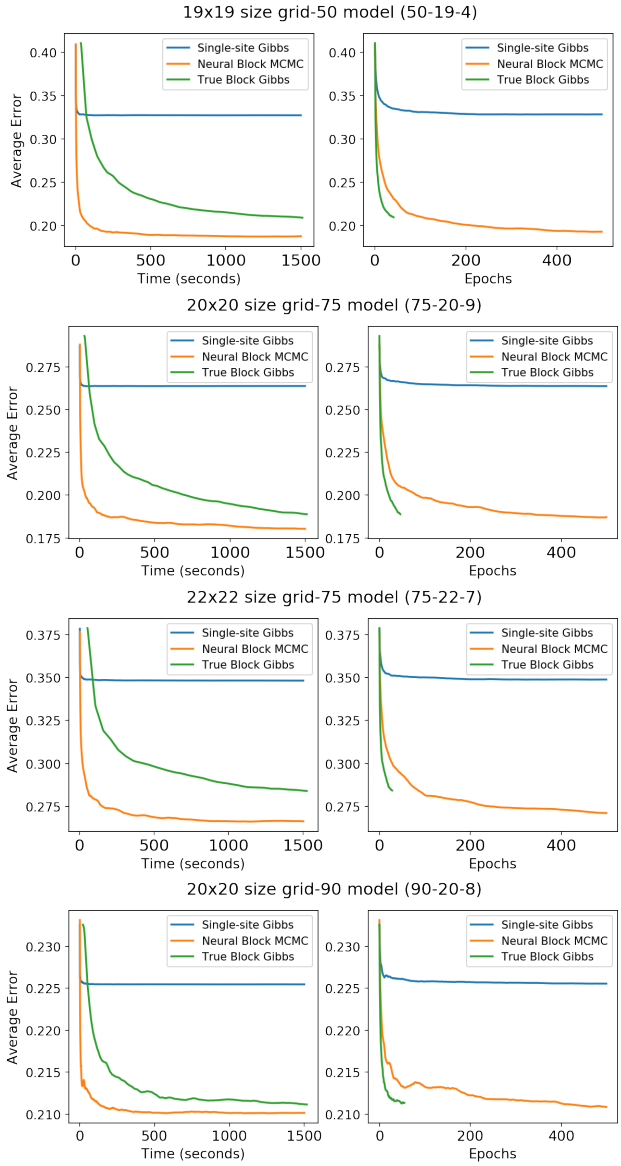

Figure 1: Additional sample runs with single-site Gibbs, neural block MCMC, and block Gibbs with true conditionals on UAI 2008 grid models. These results are obtained in same setting as Fig. 4 of main paper. For each model, we compute 10 random initializations and run three algorithms for 1500s on each one. Plots show average error for each algorithm. Epochs plots are cut off at 500 epochs to better show the comparison. "grid-k" represents that the model has $k\%$ deterministic relations.

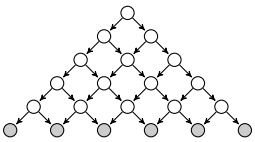

Figure 2: Small version of the triangle grid model in experiment Sec. S-2.1. Evidence nodes (shaded) are at bottom layer. The actual network has 15 layers and 120 nodes.

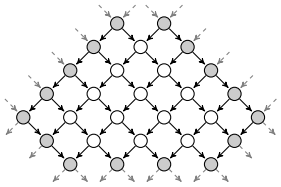

Figure 3: Motif for the model in Fig. 2. Conditioning variables (shaded) form the Markov blanket of proposed variables (white). Dashed gray arrows are possible irrelevant dependencies.

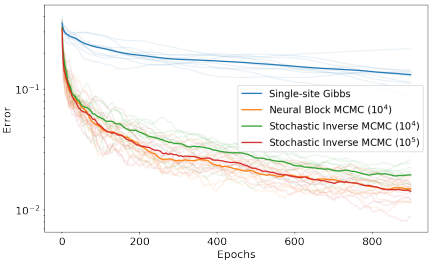

Figure 4: Error w.r.t. epochs on the triangle model in Fig. 2. Semitransparent lines show individual MCMC runs. Opaque lines show average error over 10 MCMC runs for each algorithm. Numbers in parentheses are the amounts of training data.

Architecture Our neural block proposal is trained using the GMM with $m = 8$ max mixtures and $n = 60$ data points. It proposes two mixture components (μ_j, v_j)s through underlying MDN of 156-624-624-36 network structure. The MDN's inputs include 60 observed points $\mathbf{x} = \{x_i\}_i$, 8 component means $\boldsymbol{\mu} = \{\mu_j\}_j$ and component active indicators $\mathbf{v} = \{v_j\}_j$ with values for the two proposed mixtures replaced by zeros. Orders for $\mathbf{x}$, $\boldsymbol{\mu}$ and $\mathbf{v}$ are such that they are sorted along the first principle component of $\mathbf{x}$ to break symmetry. In addition, the inputs also contain the principle component and the indicators of which components are being proposed. The MDN outputs a proposal distribution over the two (μ_j, v_j)s as a mixture of 4 MDN components.

Breaking the Symmetry In training, such mixture models have symmetries that must be broken before being used as input to the neural network [2]. In particular, the mixtures $\{(v_j, \mu_j)\}_j$ can be permuted in $m!$ ways and the points $\{(z_i, x_i)\}_i$ in $n!$ ways. Following a similar procedure by Le et al. [1], we sort these values according the first principal component of $\mathbf{x}$, and also feed the first principal component vector into the MDN.

Proposal and Inference with z Collapsed In order to avoid nearly-deterministic relations, *e.g.*, $p(v_j = 0, z_i = j) = 0$, and still train a general proposal unconstrained by n, we choose to consider the collapsed model without $\mathbf{z}$. We first experiment on the intuitive approach which adds a resampling step for $\mathbf{z}$ in the proposal. At each proposal step, trained proposal q_θ is first used to propose new mixtures $\boldsymbol{\mu}'$ and $\mathbf{v}'$, and then $\mathbf{z}$ is proposed from $p(\mathbf{z}|\boldsymbol{\mu}', \mathbf{v}', \mathbf{x})$. The MH rule is applied lastly to either accept or reject all proposed values. While this method gives good performance in small models, it suffers greatly from low acceptance ratio as n, the number of observed points $\mathbf{z}$, grows large. Therefore, we eventually choose the approach described in main paper Sec. 4.2, *i.e.*, applying the MH rule on the *collapsed* model with $\mathbf{z}$ resampled from $p(\mathbf{z}|\boldsymbol{\mu}, \mathbf{v}, \mathbf{x})$ afterwards. Since the acceptance ratio no longer depends on n, this approach behaves much more scalable than the first one in our experiments. It outperforms SDDS split-merge MCMC in GMMs of various sizes, as shown in Fig. 7 of main paper.

S-1.3 NER Tagging

Architecture The underlying MDN has two hidden layers each of size $4 \times \max\{\text{input size}, \text{output size}\}$, with output size varying according to the number of proposed variables. It maps local CRF parameters of all motif variables and conditioning variable values to the NER tag proposal for consecutive proposed variables as a mixture of 4 components.

S-2 Additional Experiment

S-2.1 Comparison with Inverse MCMC

Neural block proposals can also be used model-specifically by training only on instantiations within a particular model. In this subsection, we demonstrate that our method achieves comparable performance with a more complex task-specific MCMC method, Inverse MCMC [3].

Figure 2 illustrates the triangle grid network used in this experiment, which is identical to what Stuhlmüller et al. [3] used to evaluate Inverse MCMC. For our method, we chose the motif shown in Fig. 3. The proposal is trained on all instantiations in this triangle model.

Inverse MCMC is an algorithm that builds auxiliary data structures offline to speed up inference. Given an inference task, it trains an inverse graph for each latent variable where the latent variable is at bottom and evidence variables are at top. These graphs are then used as MCMC proposals.

It is difficult to compare these two methods w.r.t. time. While both methods require offline training, Inverse MCMC needs to train from scratch if the set of evidence nodes changes, yet neural block sampling only needs one-time training for different inference tasks on this model. In this experiment, for each inference epoch, both methods propose about 10.5 values on average per latent variable. Figure 4 shows a more meaningful comparison of error w.r.t. epochs. Our learned neural block proposal, trained using 10^4 samples, achieves comparable performance with Inverse MCMC, which is trained using 10^5 samples and builds model-specific data structures (inverse graphs).

Architecture The underlying MDN has 161-1120-1120-224 network structure, mapping the CPTs of all 29 motif variables and 16 conditioning variable values to the proposal distribution of 13 proposed variables as a mixture of 16 components.

Inverse MCMC Setting In this experiment, we run Inverse MCMC with frequency density estimator trained with posterior samples, proposal block size up to 20 and Gibbs proposals precomputed, following the original approach of Stuhlmüller et al. [3].

References

[1] Tuan Anh Le, Atilim Gunes Baydin, and Frank Wood. Inference compilation and universal probabilistic programming. In *Artificial Intelligence and Statistics*, pages 1338–1348, 2017.

[2] Robert Nishihara, Thomas Minka, and Daniel Tarlow. Detecting parameter symmetries in probabilistic models. *arXiv preprint arXiv:1312.5386*, 2013.

[3] Andreas Stuhlmüller, Jacob Taylor, and Noah Goodman. Learning stochastic inverses. In *Neural Information Processing Systems*, 2013.