We thank the reviewers for their insightful feedback; we address each review below.

<u>**R1:**</u> **"...it is known how to solve the BFE optimisation problem by double loop algorithms"** Our understanding is that double-loop algorithms (Yuille, 2001) converge to a local minimum, but not necessarily a global one. Further, double-loop algorithms outperform LBP generally when LBP does not converge, but in our experiments LBP converged reliably, and we thus anticipate that double-loop optimization will not necessarily improve further. We also note that despite the superiority of double loop algorithms, LBP remains popular, and so amortizing it seems useful in our view.

"...what is meant by 'they are run once...'" A Transformer or RNN is run over a sequence of embeddings corresponding to the nodes in the graph to obtain a sequence of representations. We do *not* then further update these Transformer/RNN representations with message-passing style updates as in much recent work in graph neural networks.

"...meaningless for pairwise marginals..." Agreed. We included the pairwise marginals just for completeness.

"Ising model...expect that the estimation quality will degrade with the (average) interaction strength." Thank you for this suggestion! We tested our approach in settings where the interaction strength is higher by sampling pairwise potentials from $\mathcal{N}(0, 3)$ (unary potentials are still sampled from $\mathcal{N}(0, 1)$). For a 10x10 grid we obtain correlations of 0.460, 0.641, and 0.770 for Mean Field, Loopy BP, and the inference network, respectively. For a 15x15 grid we analogously obtain 0.435, 0.665, and 0.738. When sampling from $\mathcal{N}(0, 5)$ we obtain 0.358, 0.525, 0.619 for these 3 approaches in the 10x10 case, and 0.362, 0.459, 0.564 in the 15x15 case.

"The experiments have in my view a preliminary character" We agree our experiments are on small datasets. Our contribution is mostly methodological and a first step toward learning undirected models in this way, in line with early work for learning directed models with VAEs. We also note that we consider 3 fairly different archetypal settings.

"some details of them are missing" We apologize for the missing details, which had to be relegated to the supplementary materials due to space. We will make sure to include them should the paper be accepted.

<u>**R2:**</u> **"...having some higher order factors"...** Having too many higher order factors could be slow (since marginals scale exponentially in the order) but having a few may improve performance; we will investigate this!

"...optimizing in the subspace..." We explored this approach in preliminary experiments, and while it can succeed it is much less scalable: it requires (pre)computing the SVD of a large matrix (i.e., $O(|\mathcal{F}|^2)$) and we need an SVD for each graph topology (c.f., HMMs, which have different lengths per sentence). We will add more discussion around this point.

"...hardware..." We ran our experiments on 1080 Tis, using pytorch 1.1.

"Why are the inference networks different..." Empirically we observed small differences between architectures for different tasks in preliminary experiments, and we chose the best; see answer to R3 as well.

<u>**R3:**</u> **"The marginal constraints can also be enforced by matrix operations.."** See response to R2.

"The numbers in Table 1 and Figure 2 seem inconsistent..." The figures in Table 1 are for both unary and pairwise marginals. While LBP does slightly better for unary marginals, the inference network does much better for pairwise, and when averaged across both in Table 1, the inference network does better overall; we will clarify this further.

"...I wonder whether some architecture of inference networks are better than others...." We experimented with both Transformers and RNNs, and picked the architecture with the best performance in preliminary experiments. For HMMs, we find Transformers give an NLL of ≈ 116, slightly worse than RNNs; we will include these results.

"The form of discrepancy used ... is not clear." We apologize for the lack of clarity: we used L2 for the synthetic experiments and Jensen-Shannon divergence for the rest, though L2 was only slightly worse. We will add this detail.

"...random seed as a hyperparameter..." In our experience discrete latent variable models can be quite sensitive to initialization, and so to fairly compare all model variants we randomly sample the same number of seeds for each; we do not regard this as a hyperparameter but as giving each model/method equal opportunity in experiments.

"...did not get numbers similar..." The following reproduces our numbers on a 1080 Ti, using pytorch 1.1, by epoch 9: `python pen_uhmm.py -cuda -K 30 -bsz 32 -dropout 0.3 -ilr 0.0003 -infarch rnnnode -init 0.001 -just_diff -lemb_size 64 -loss alt3 -lr 0.0001 -markov_order 3 -max_len 30 -not_inf_residual -optalg adam -penfunc js -q_layers 1 -qemb_size 150 -qinit 0.001 -seed 21442 -vemb_size 64 -epochs 10`. The code prints out *perplexity* rather than the NLL reported in the paper, and thus looks higher. We apologize for the potential confusion. (To convert, take $\log(\text{perplexity}) \times 50509/2747$).

"...the authors probably ran over 100k seeds..." We did not run 100k seeds; all methods were given 100 random configurations, as the code (line 609 in pen_uhmm.py) makes clear. Regarding stability, we calculated the std. deviation of best NLL obtained per run for each method in Table 4, giving (from top to bottom): 5.3, 0.7, 1.1, 0.5, 6.4, 2.8, 1.1. Thus amortized BFE is more stable than exact inference and LBP, but less than Mean-Field+BL and 1st Order.