

Sampling From a Discrete Posterior

Bar Weinstein

December 24, 2024

1 Setup

We have posterior

$$\begin{aligned} p(\eta, \theta, \gamma, A^* | Y, Z, X, A) &\propto p(Y | Z, A^*, X, \eta) p(\eta) \\ &\times p(A | A^*, X, \gamma) p(\gamma) \\ &\times p(A^* | X, \theta) p(\theta), \end{aligned} \tag{1}$$

where η, θ, γ are continuous and A^* is discrete with large dimension $O(N^2)$.

First approach is to use the cut-posterior

$$p_{\text{cut}}(\eta, \theta, \gamma, A^* | Y, Z, X, A) \propto p(\eta | Y, Z, A^*, X) p(\theta, \gamma, A^* | A, X), \tag{2}$$

and sample from it by first sample θ, γ , then generate A^* samples, and finally sample η either via plug-in or multi-stage sampling. This approach is fine but other than plug-in sampling it takes a while. We can maybe do better!

Gibbs sampling requires sampling from the full conditional of each. Write $O = (Y, Z, X, A)$ for the observed data. Then the full conditionals are

$$\begin{aligned} \gamma &\sim p(\gamma | O, \eta, \theta, A^*) \propto p(A | A^*, X, \gamma) p(\gamma) \\ \theta &\sim p(\theta | O, \eta, \gamma, A^*) \propto p(A^* | X, \theta) p(\theta) \\ \eta &\sim p(\eta | O, \theta, \gamma, A^*) \propto p(Y | Z, A^*, X, \eta) p(\eta) \\ A^* &\sim p(A^* | O, \eta, \theta, \gamma) \propto p(Y | Z, A^*, X, \eta) p(A | A^*, X, \gamma) p(A^* | X, \theta). \end{aligned} \tag{3}$$

Sampling γ, θ, η is easy, but sampling A^* is hard. The continuous parameters can be updated with gradient-based methods such as MALA or HMC. I'll now present methods for sampling A^* .

2 Methods for Discrete Sampling

Assume that \mathcal{A}^* is the space of all possible A^* . Naively, we can sample A^* by sampling all edges independently. However, A^* may have global impact on Y so flipping one edge A_{ij}^* can impact the the likelihood of Y_k where $k \neq i, j$. That will complicate the naive A^* updates, e.g., via MH.

Write the log conditional posterior of A^* as

$$\begin{aligned} f(A^*) &= \log p(A^* | O, \eta, \theta, \gamma) \\ &\propto \log p(Y | Z, A^*, X, \eta) + \log p(A | A^*, X, \gamma) + \log p(A^* | X, \theta). \end{aligned} \tag{4}$$

1. **The Hamming Ball Sampler.**¹ Variant of slice sampling for discrete variables. Idea is to have an auxiliary variable that constraint the sampling space to a subset of \mathcal{A}^* . At each iteration we create a “slice” (the auxiliary variable), sample A^* from this slice subset, etc. Example of application is to restrict the number of changes of A^* proposal in each iteration and sample from that the Hamming ball of radius d . Effectively, it is local updates constraints to the number of new changes.
2. **Informed Proposals.** Basic idea is that in updating A^* we propose a new value from a kernel $q(A^{*'}|A^*)$. In fact, we propose each edge separately so it can be written as $q(A^{*'}|A^*) = \sum_e q(A^{*'}|A^*, e)q(e)$, that is, sample edges with $q(e)$ and then “flip” the edge with $q(A^{*'}|A^*, e) = I(A_{ij}^{*'} = A_{ij}^*)$, $\forall (i, j) \neq e$. Key idea here is what edges e we select to flip. Naive random-walk just select these edges uniformly, but we can do better. That is the main idea behind Zanella (2020)² paper on Informed Proposals.

Assume for now that we only “flip” one edge at a time (can be extended for block or sequential updates in each iteration). One option is to inform the edge selection by its impact on $f(A^*)$ values, i.e., on the gradients. Since A^* is discrete we don’t have gradients. As before, write $A^{*'}$ as A^* where only edge e is “flipped” ($0 \rightarrow 1, 1 \rightarrow 0$). Then the score is $d(A^{*'}, A^*) = f(A^{*'}) - f(A^*)$. We can use this score to inform the edge selection by taking proposals $q(A^{*'}|A^*) \propto \exp(d(A^{*'}, A^*)/t)$ where t is a “temperature” controlling the locality of the updates. By Zanella (2020), optimal t is $t = 2$ in the binary case (so called “Barker choice”). The scores are effectively the tempered likelihood ratios. However, computing $d(A^{*'}, A^*)$ is expensive since we need to evaluate f twice for each possible edges, i.e., at least $O(N^2)$ per iteration. Grathwohl et al. (2021)³ proposed to approximate this score by taking a gradient: $\tilde{d}(A^{*'}, A^*) = (A_e^{*'} - A_e^*) \nabla f_{A_e^*}(A_e^*)$. That is, $\tilde{d}(A^{*'}, A^*) \approx d(A^{*'}, A^*) = f(A^{*'}) - f(A^*)$, where the gradient of $f()$ is taken w.r.t. A_e^* as if A_e^* is continuous⁴. They have some results about this approximation. The proposed pseudo-algorithm for A^* updates is thus:

Algorithm 1 Gibbs With Gradients

Input: unnormalized log-prob $f(\cdot)$, current sample x
 Compute $\tilde{d}(x)$ {Eq. 3 if binary, Eq. 4 if categorical}
 Compute $q(i|x) = \text{Categorical}\left(\text{Softmax}\left(\frac{\tilde{d}(x)}{2}\right)\right)$
 Sample $i \sim q(i|x)$
 $x' = \text{flipdim}(x, i)$
 Compute $q(i|x') = \text{Categorical}\left(\text{Softmax}\left(\frac{\tilde{d}(x')}{2}\right)\right)$
 Accept with probability:

$$\min\left(\exp(f(x') - f(x)) \frac{q(i|x')}{q(i|x)}, 1\right)$$

¹<https://www.tandfonline.com/doi/full/10.1080/01621459.2016.1222288>

²<https://www.tandfonline.com/doi/full/10.1080/01621459.2019.1585255?src=recsys>

³<https://proceedings.mlr.press/v139/grathwohl21a.html>

⁴It can be performed automatically in torch via `torch.autograd.grad` or in JAX via `jax.grad`.

Namely, compute scores for edge selection, sample edge with probabilities proportional to the scores, flip its edge, compute new scores, and accept/reject with MH. This can be extended for either K edges “flips” in a block or K sequential flips in each iteration. The cool thing is that don’t really have to flip all edges in each iteration.

Limitations. To compute the scores $d(A^*, A^*)$ we need to evaluate f twice for each edge flip, i.e., $O(N^2)$ evaluations. If A^* and A are edge independent then, evaluating $\log p(A|A^*, X, \gamma) + \log p(A^*|X, \theta)$ is easy since each flip only affects one edge. However, the outcome model can possibly depend on global network statistics, e.g., eigenvector centrality, so flipping one edge can impact the outcomes of many units! That will be computationally expensive to evaluate the scores using the outcomes model. One option is to neglect the outcome model in the scores computation, or just use them occasionally (e.g., in every L iterations). That resemble cut-posterior sampling, but not exactly as we sample everything together. Need to think if using the gradient with \tilde{d} will assist even with the outcome model.

Multiple updates. In Grathwohl et al. (2021) approach, in each iteration we “flip” one edge. It can be updated by sampling K edges from $q(i|A^*)$ and in the accept/reject step write accordingly the new network A^* with all the edges flips, and instead of $q(i|A^*)$ write the product $\prod_k q(i_k|A^*)$.

Discrete MALA. Zhang et al. (2022)⁵ proposed to use *discrete Metropolis-adjusted Langevin algorithm (DMALA)*. Their approach builds on Grathwohl et al. (2021). Their idea is that if we can factorize the discrete parameter space $\mathcal{A}^* = \prod_e \mathcal{A}_e^* = \{0, 1\}^D$ with $D = N(N-1)/2$, then we can suggest an update for all edges in each iteration, instead of the single update in Grathwohl et al. (2021). Their methods begins with computing the gradient $\nabla f(A^*)$ as approximation for the difference (likelihood ratio) as before. Then we “flip” edge e with probability $\frac{\exp(\frac{1}{2}\nabla f(A^*)_e(A_e^* - A_e) - \frac{1}{2\alpha})}{\exp(\frac{1}{2}\nabla f(A^*)_e(A_e^* - A_e) - \frac{1}{2\alpha}) + 1}$ where α is the step size. Namely, we compute once the gradient for all edges, and then propose update (flip) for each edge at once. We accept/reject the update with MH step with the log-probabilities f and the product of selection probabilities.

Algorithm 2 DULA and DMALA with Binary Variables.

given: Stepsize α .

loop

compute $P(\theta) = \frac{\exp(-\frac{1}{2}\nabla U(\theta) \odot (2\theta - 1) - \frac{1}{2\alpha})}{\exp(-\frac{1}{2}\nabla U(\theta) \odot (2\theta - 1) - \frac{1}{2\alpha}) + 1}$

sample $\mu \sim \text{Unif}(0, 1)^d$

$I \leftarrow \text{dim}(\mu \leq P(\theta))$

$\theta' \leftarrow \text{flipdim}(I)$

▷ Optionally, do the MH step

compute $q(\theta'|\theta) = \prod_i q_i(\theta'_i|\theta) = \prod_{i \in I} P(\theta)_i \cdot \prod_{i \notin I} (1 - P(\theta)_i)$

compute $P(\theta') = \frac{\exp(-\frac{1}{2}\nabla U(\theta') \odot (2\theta' - 1) - \frac{1}{2\alpha})}{\exp(-\frac{1}{2}\nabla U(\theta') \odot (2\theta' - 1) - \frac{1}{2\alpha}) + 1}$

compute $q(\theta|\theta') = \prod_i q_i(\theta_i|\theta') = \prod_{i \in I} P(\theta')_i \cdot \prod_{i \notin I} (1 - P(\theta')_i)$

set $\theta \leftarrow \theta'$ with probability

$$\min \left(1, \exp(U(\theta') - U(\theta)) \frac{q(\theta|\theta')}{q(\theta'|\theta)} \right)$$

end loop

output: samples $\{\theta_k\}$

⁵<https://proceedings.mlr.press/v162/zhang22t.html>

References