CPSC 661: Sampling Algorithms in Machine Learning *Out: March 1, 2021*

Problem Set 1

Lecturer: Andre Wibisono *Due: March 19, 2021*

Instruction

Solve at least 3 of the following problems (feel free to solve as many as you'd like). Each problem has equal worth, so you can choose the ones that are most interesting to you. Collaboration is allowed and encouraged, but please write your own solution and acknowledge your collaborators. Submit the solution in PDF format via Canvas. If there are questions, please post a discussion on Canvas or email `andre.wibisono@yale.edu`.

1 Relations between distances

Let ρ and ν be probability distributions on $\mathcal{X} = \mathbb{R}^n$. Assume ρ has a density $h = \frac{d\rho}{d\nu} \colon \mathbb{R}^n \to \mathbb{R}$ with respect to ν. Recall the total variation distance is $\mathrm{TV}(\rho, \nu) = \frac{1}{2}\mathbb{E}_\nu[|h(X) - 1|]$, the χ^2-divergence is $\chi^2_\nu(\rho) = \mathbb{E}_\nu[(h(X) - 1)^2]$, the KL divergence is $H_\nu(\rho) = \mathbb{E}_\nu[h(X) \log h(X)]$, and the warmness is $M^\infty_\nu(\rho) = \sup_{x \in \mathbb{R}^n} |h(x) - 1|$.

Show that:

$$2\,\mathrm{TV}(\rho, \nu)^2 \;\le\; H_\nu(\rho) \;\le\; \chi^2_\nu(\rho) \;\le\; 2\mathrm{TV}(\rho, \nu) \cdot M^\infty_\nu(\rho).$$

2 Distances between Gaussians

Let $\rho = \mathcal{N}(0, \alpha I)$ for some $\alpha > 0$, and $\nu = \mathcal{N}(0, I)$ on $\mathbb{R}^n$.

(a) Compute $\chi^2_\nu(\rho)$, $H_\nu(\rho)$, and $M^\infty_\nu(\rho)$.

(b) Compare the behavior and dependence of $\mathrm{TV}(\rho, \nu)$, $\chi^2_\nu(\rho)$, $H_\nu(\rho)$, and $M^\infty_\nu(\rho)$ on n and α.

3 Linearization

Suppose ρ is close to ν, in the sense that $\rho(x) = (1 + \epsilon g(x))\nu(x)$ for some $\epsilon > 0$ and $g \colon \mathbb{R}^n \to \mathbb{R}$. Show that as $\epsilon \to 0$, the KL divergence recovers the χ^2-divergence:

$$H_\nu(\rho) = \frac{\epsilon^2}{2}\,\chi^2_\nu(\rho) + O(\epsilon^3).$$

4 Random walk on bipartite graph

Let P be the random walk Markov chain on an undirected unweighted graph G. Show that $\lambda = -1$ is an eigenvalue of P if and only if G is bipartite.

5 Ornstein-Uhlenbeck

Recall the Ornstein-Uhlenbeck (OU) algorithm on $\mathbb{R}^n$ is:

$$X_{k+1} = e^{-\eta} X_k + \sqrt{(1 - e^{-2\eta})}\, Z_k \tag{1}$$

where $\eta > 0$ is step size, and $Z_k \sim \mathcal{N}(0, I)$ is an independent Gaussian random variable in $\mathbb{R}^n$.

(a) Show that $\nu = \mathcal{N}(0, I)$ is the stationary distribution by checking that if $X_k \sim \nu$, then $X_{k+1} \sim \nu$.

(b) Show that $\nu = \mathcal{N}(0, I)$ is the stationary distribution by checking that the Markov chain corresponding to OU is reversible with respect to ν.

(c) *(Optional)* If you want to sample from $\nu = \mathcal{N}(\mu, \Sigma)$ for some $\mu \in \mathbb{R}^n$, $\Sigma \succ 0$, how should you change the OU algorithm?

6 Convergence of Ornstein-Uhlenbeck

Consider the Ornstein-Uhlenbeck algorithm (1) on $\mathbb{R}^n$ above. Suppose we start at a deterministic point $X_0 = x_0$ for some $x_0 \in \mathbb{R}^n$, so the initial distribution is a point mass: $\rho_0 = \delta_{x_0}$.

(a) Compute the distribution ρ_k of X_k explicitly.

(b) Compute the mixing time in χ^2-divergence and KL divergence.

7 Approximate Ornstein-Uhlenbeck

An approximate version of the Ornstein-Uhlenbeck algorithm on $\mathbb{R}^n$ is:

$$X_{k+1} = X_k - \eta\, X_k + \sqrt{2\eta}\, Z_k$$

where $\eta > 0$ is step size, and $Z_k \sim \mathcal{N}(0, I)$ is an independent Gaussian random variable in $\mathbb{R}^n$.

(a) Show that $\nu = \mathcal{N}(0, I)$ is *not* the stationary distribution.

(b) Compute the stationary distribution ν_η for any $\eta > 0$.

(c) Compute the distance between ν_η and ν in χ^2-divergence and KL divergence.

8 Metropolis vs. Barker filters

Recall in the Metropolis-Hastings algorithm, from a point x and for a proposed point y, the Metropolis filter accepts y with probability

$$\min\left\{1, \frac{\nu(y)P_y(x)}{\nu(x)P_x(y)}\right\}$$

while the Barker filter accepts with probability

$$\frac{\nu(y)P_y(x)}{\nu(y)P_y(x)+\nu(x)P_x(y)}.$$

Implement the Metropolis filter and the Barker filter for the following random walks:

1. $P =$ Brownian motion with step size η.
2. $P =$ Unadjusted Langevin Algorithm with step size η.

Take the target distribution to be Gaussian: $\nu = \mathcal{N}(0, \Sigma)$ on $\mathbb{R}^n$ where $\Sigma = \mathrm{diag}(\lambda_1, \ldots, \lambda_n)$ is a diagonal matrix with $\lambda_1, \ldots, \lambda_n$ equally spaced in between $\lambda_1 = 1$ and $\lambda_n = 4$.

(a) Plot the average acceptance probability of the two filters as you increase n and vary η.

(b) Plot the approximate mixing time of the two filters as you increase n and vary η. Do you see the difference in performance between the two filters?

(*Note:* See the experimental setup in [Dwivedi et al., 2019], Sections 4.1 and 4.4, and see Figures 1(a) and 10(a,c) for sample outputs. The code for [Dwivedi et al., 2019] is available online; you may take it as a starting point and modify it, or you may implement your own code.)

9 Sampling non-log-concave distribution

Consider sampling from the mixture of Gaussian on $\mathbb{R}^n$:

$$\nu = \frac{1}{2}\mathcal{N}(-a, I) + \frac{1}{2}\mathcal{N}(a, I)$$

If $\|a\|_2 \le 1$, then ν is $(1 - \|a\|_2^2)$-strongly log-concave; but if $\|a\|_2 > 1$, then ν is not log-concave. Note that ν is 1-log-smooth. Take $n = 2$ and start from initial distribution $\rho_0 = \mathcal{N}(0, I)$.

Implement and compare the performances of the following algorithms:

1. Unadjusted Langevin Algorithm (ULA).
2. Metropolis Random Walk (MRW).

3. Metropolis-Adjusted Langevin Algorithm (MALA).

Take a few different values of a ranging from $\|a\|_2 < 1$ (ν log-concave) to $\|a\|_2 > 1$ (ν not log-concave), and vary the step size η. Do you see a difference in how the performances of the algorithms degrade as $\|a\|_2$ increases?

(See [Dwivedi et al., 2019], Section 4.2 and Figure 4(a) for a sample output.)

References

[Dwivedi et al., 2019] Dwivedi, R., Chen, Y., Wainwright, M. J., and Yu, B. (2019). Log-concave sampling: Metropolis-Hastings algorithms are fast. *Journal of Machine Learning Research*, 20:1–42.