Let's take a walk!
Sampling a distribution by Random Walk algorithms

Kunal Gupta

University of California, San Diego

k5gupta@eng.ucsd.edu

September 29, 2020

Common Techniques for sampling a distribution

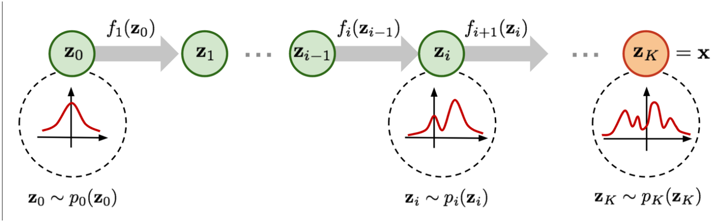
- **GANs:** Generate $P_\theta(X|z)$ given $z \sim \mathcal{N}(0,1)$ and train with discriminator $D_\phi(\{0,1\}|X)$.
  - Doesn't require calculating log-likelihoods.
  - No need for encoder network.
- **VAEs[1]:** Generate $P_\theta(X|z)$ given $z \sim E(z|X)$. However, calculating $E(z|X)$ is hard and is approximated by a parametric $Q_\phi(z|X)$. Assuming some prior over z i.e. $P(z)$ we solve for

$$\arg\max_{\theta,\phi} \; ELBO(\theta,\phi,X) = \mathbb{E}[logP_\theta(X|z)] - D_{KL}[Q_\phi(z|X)||P(z)]$$

$$\implies Q_{\phi^*}(z|X) \approx E(z|X)$$

  Can calculate only a lower-bound of log-likelihood
- **Normalizing Flows :** Exact calculation of log-likelihood possible
- **MCMC :** Exact calculation for gradient of log-likelihood

What is Normalizing Flows[2,3] ?



Given a random variable $z \sim \pi(z)$, we create another random variable $x = f(z)$ s.t. $z = f^{-1}(x)$ exists. Technique to infer $p(x)$ from $\pi(z)$ is called normalizing flows.

Intuitively, $\int p(x)dx = \int \pi(z)dz = 1 \iff p(x)|dx| = \pi(z)|dz|$ i.e. we scale the two distributions by the size of their respective rectangle. For multivariate case, $\implies p(x) = \pi(z)\left|det\frac{dz}{dx}\right| = \pi(f^{-1}(x))\left|det\frac{df^{-1}}{dx}\right|$

If $x = z_k = f_k \circ f_{k-1} \circ \cdots \circ f_1(z_0)$

$logp(x) = log\pi_k(z_k) = log\pi_0(z_0) - \sum_{i=1}^{k} log\left|det\frac{df_i}{dz_{i-1}}\right|$

Problem Statement

Problem[4] : Sample from a distribution $p(x) \propto e^{-f(x)}$ given black box access to f and ∇f

- **Latent modeling :** Suppose we have a data generation model $p_\theta(x|h)$ and a prior over latent variables $p_\theta(h)$ then we obtain a distribution over latent variables as

$$p_\theta(h|x) \propto p_\theta(x|h)p(h)$$

- **Energy based models :** In a dynamic system, a particle is most likely to be present in region where energy is minimum. It's probability distribution is defined as

$$p(x) \propto e^{-E(x)}$$

Markov Chains [5]

Let's see the cow surface distribution example for some motivation.

Definition

Let D be a finite set. A random process $X_1, X_2, \cdots$ with values in D is called a *Markov chain* if
$P\{X_{n+1} = x_{n+1} | X_n = x_n, \cdots, X_0 = x_0\} = P\{X_{n+1} = x_{n+1} | X_n = x_n\}$

Definition

The matrix $\pi = (p_{ij})_{i,j \in D}$ is called the transition probability matrix. p_{ij} is the probability of transition from state i to state j $\forall n$.

For example, $P\{X_n = j | X_0 = i\} = P_i\{X_n = j\} = (\pi^n)_{ij}$

Definition

A Markov chain X_n is called ergodic if the limit $\Gamma(j) = lim_{n \to \infty} P_i\{X_n = j\}$ exists for every state j and does not depend on the initial state i. The D-vector Γ is called the *stationary probability*. This implies $\Gamma = \Gamma \pi$

Metropolis(-Hastings) algorithm [6]

- We wish to draw samples from some probability distribution without knowing its exact height at any point.
- Key Idea [7] : "Wander around" on that distribution in such a way that the amount of time spent in each location is proportional to the height of the distribution.

Consider a probability distribution $P(x)$ (a.k.a target distribution) that can only be evaluated only upto a scale by a function $f(x)$ i.e. $f(x) \propto P(x)$

- **Initialize** $x_t \sim Q(x|y) = \mathcal{N}(y, \sigma)$
- **For**
  - Generate a candidate $x^{'} \sim Q(x^{'}|x_t)$
  - Calculate acceptance ratio $\alpha = \frac{f(x^{'})}{f(x_t)} \left(= \frac{P(x^{'})}{P(x)} \right)$

    This is used to decide whether to accept or reject the new candidate.
  - Generate a uniform random number $u \in [0, 1]$
  - If $u \leq \alpha$ then $x_{t+1} = x^{'}$ else, $x_{t+1} = x_t$
- **End for**

Why Metropolis-Hastings work [7] ?

- We wish to construct a transition matrix π that yields a stationary distribution $\Gamma \approx P \propto f$ which is same as our target distribution P
- Recall that $\Gamma(y) = \Gamma(x)\pi(x, y)$ for stationary distribution Γ
- This must also be true for $\Gamma(x) = \Gamma(y)\pi(y, x)$
- Therefore π should be such that $\Gamma(x)\pi(x, y) = \Gamma(y)\pi(y, x)$ is true
- Choosing $\pi(x, y) = min\left(1, \frac{\Gamma(y)}{\Gamma(x)}\right)$ does the trick
- Note: Instead of evaluating $\Gamma(.)$ we use $f(.)$ as $\Gamma \propto f$
- In summary, if the probability of $x \to y$ is higher than we reduce it to match the probability of $y \to x$

Metropolis-Adjusted Langevin Algorithm (MALA)

- We wish to generate candidates more intelligently.
- Instead generate candidate $x^{'} = \nabla log\ f(x_t) + \epsilon,\ \epsilon \sim \mathcal{N}(x_t, \sigma_t)$
- Note: Since $f \propto P \implies \nabla logf = \nabla logP$. No need to calculate normalizing factor!
- However, there is an interesting alternative to this!
- Consider the set of equations

$$x_{t+1} = x_t - \eta \nabla f(x_t)\ (GradientDescent)$$

$$x_{t+1} = x_t - \eta \nabla f(x_t) + \epsilon,\ \epsilon \sim \mathcal{N}(x_t, \sigma_t)\ (Langevin - MC)$$

- What happens as $\nabla f \rightarrow 0$ (optimum) ?
- While GD reaches optimum, LM reaches the target distribution.
- It can be shown that with small η MH almost always selects the candidate[8]. There we can avoid MH step altogether!

Learning a Score function

Let's assume that $q(x)$ is the true distribution and $p(x;\theta)$ is a parameterized function (neural network) that approximated $q(x)$ upto a scale. The core principle of score matching is to learn θ so that $\psi(x;\theta) = \frac{\partial logp(x;\theta)}{\partial x}$ best matches the corresponding score of the true distribution i.e. $\frac{\partial logq(x)}{\partial x}$. We therefore aim to minimize the following objective:

$$J_{ESM}(\theta) = \mathbb{E}_{q(x)}\left[\frac{1}{2}||\psi(x;\theta) - \frac{\partial logq(x)}{\partial x}||^2\right]$$

We can show that this objective is equivalent to

$$J_{ISM}(\theta) = \mathbb{E}_{q(x)}\left[tr(\nabla\psi(x;\theta)) + \frac{1}{2}||\psi(x;\theta)||^2\right] + C$$

Provided that $q(x)\psi(x;\theta) \rightarrow 0, x \rightarrow \pm\infty$. Another intuition can be that the gradient $\psi(x;\theta)$ of the log density at some corrupted point $\tilde{x}$ should ideally move us towards the clean sample x.

References

1. https://wiseodd.github.io/techblog/2016/12/10/variational-autoencoder
2. https://lilianweng.github.io/lil-log/2018/10/13/flow-based-deep-generative-models.html
3. https://youtu.be/i7LjDvsLWCg
4. http://www.offconvex.org/2020/09/19/beyondlogconvavesampling/
5. http://gregorygundersen.com/blog/2019/10/28/romantic-markov-chains/
6. http://gregorygundersen.com/blog/2019/11/02/metropolis-hastings/
7. https://stats.stackexchange.com/questions/165/how-would-you-explain-markov-chain-monte-carlo-mcmc-to-a-layperson/12657
8. Welling, Max, and Yee W. Teh. "Bayesian learning via stochastic gradient Langevin dynamics." Proceedings of the 28th international conference on machine learning (ICML-11). 2011.

Some more references

- Song, Yang, and Stefano Ermon. "Generative modeling by estimating gradients of the data distribution." Advances in Neural Information Processing Systems. 2019.
- Du, Yilun, and Igor Mordatch. "Implicit generation and modeling with energy based models." Advances in Neural Information Processing Systems. 2019.
- Dalalyan, Arnak S. "Theoretical guarantees for approximate sampling from smooth and log-concave densities." arXiv preprint arXiv:1412.7392 (2014).
- Hyvärinen, Aapo. "Estimation of non-normalized statistical models by score matching." Journal of Machine Learning Research 6.Apr (2005): 695-709.
- Vincent, Pascal. "A connection between score matching and denoising autoencoders." Neural computation 23.7 (2011): 1661-1674.
- Song, Yang, and Stefano Ermon. "Improved techniques for training score-based generative models." arXiv preprint arXiv:2006.09011 (2020).

Some more references

- Sliced Score Matching: A Scalable Approach to Density and Score Estimation
- Yang, Guandao, et al. "Pointflow: 3d point cloud generation with continuous normalizing flows." Proceedings of the IEEE International Conference on Computer Vision. 2019.
- Chen, Ricky TQ, et al. "Neural ordinary differential equations." Advances in neural information processing systems. 2018.
- Cai, Ruojin, et al. "Learning Gradient Fields for Shape Generation." arXiv preprint arXiv:2008.06520 (2020).

The End