Sampling

Krishna Balasubramanian

University of California, Davis

STA 243: Spring 2020

Sampling

A large part of statistics depends on computing expectations. As a few examples consider:

- In statistics, we assume the data set (denoted as $\mathcal{D}$ and consisting of n samples) is assumed to come from some probability distribution with parameter $\boldsymbol{\theta}$. In Bayesian statistics, it is further assumed that the parameter $\boldsymbol{\theta}$ is also drawn from another distribution. Often times, we are interested in computing some statistic ($g(\cdot)$) of the random vector $\boldsymbol{\theta}$ or its expected value. In this case, the posterior distribution is given by $\mathbb{E}(g(\boldsymbol{\theta})|\mathcal{D}) = \int_{\boldsymbol{\theta}} g(\boldsymbol{\theta}) p(\boldsymbol{\theta}|\mathcal{D})\, d\boldsymbol{\theta}$.
- Marginalizing over missing data $p(\mathbf{x}) = \int p(\mathbf{x}, \mathbf{z})\, d\mathbf{z} = \mathbb{E}_{p(\mathbf{z})}[p(\mathbf{x}|\mathbf{z})]$
- Computing probabilities $p(\mathbf{x} \in C) = E_p(\mathbf{I}_C)$, where $\mathbf{I}_C$ is the indicator set.

Basic Idea

When the expectation (integral or summation) does not have a closed form we need to resort to approximation techniques. One class of approximation methods is numerical integration techniques such as the trapezoid or Simpson's method. A second class of approximation methods, which we will concentrate on, are based on Monte Carlo sampling and the law of large numbers:

$$\frac{1}{m}\sum_{i=1}^{m} g(\mathsf{x}_i) \approx \mathbb{E}_{p(\mathsf{x})}[g(\mathsf{x})] \quad \text{if} \quad \mathsf{x}_1, \ldots, \mathsf{x}_m \sim p(\mathsf{x}).$$

Basic Idea

The above estimator is **unbiased** and has **variance**:

$$\frac{\mathrm{Var}\left(g(\mathbf{x})\right)}{m}$$

The convergence above is quite stable and rapid (as indicated by the uniform law of large numbers and large deviation theory) and does not depend on the dimensionality of $\mathbf{x}$. Slow convergence may occur, however, if $g(\cdot)$ is high where $p(\cdot)$ is low and vice verse.

Basic Idea

The benefit of sampling methods over numerical integration methods is that they work better in high dimensional cases. But it depends crucially on our ability to sample from $p(\mathbf{x})$. Some high dimensional models are easy to sample from where as in other high dimensional models such as exponential family models or Markov random fields, sampling is not straightforward. We now look at some sampling techniques.

Histogram Method

- In general, we will assume that we can sample from a uniform $U([0,1])$ distribution.
- Sampling from a uniform distribution has been widely studied and many efficient methods for doing so exist.
- To sample from a discrete one dimensional RV $\mathbf{x} \in \mathbb{R}$, we can just generate a $r \sim U([0,1])$ random number and compare it with cdf F_X and sample x_i for which $r_i \in [F_X(x_i), F_X(x_{i+1})]$.
- The above method can be applied to continuous RVs by discretizing them (approximating a continuous RV by its discrete histogram). The method works well for one dimensional RVs but suffers greatly in high dimensional cases.

Transformation Method

- We focus on the case of one dimensional distribution. Extensions to high dimensional models are straightforward.
- The transformation $\mathbf{x} \mapsto F_{\mathbf{x}}(\mathbf{x})$ results in a uniform RV

  $$\begin{aligned} P(F_{\mathbf{x}}(\mathbf{x}) \leq r) &= P(F_{\mathbf{x}}^{-1}(F_X(\mathbf{x})) \leq F_{\mathbf{x}}^{-1}(r)) \\ &= P(\mathbf{x} \leq F_{\mathbf{x}}^{-1}(r)) \\ &= F_{\mathbf{x}}(F_{\mathbf{x}}^{-1}(r)) = r. \end{aligned}$$

  As a result transforming the uniform samples by $r \mapsto F^{-1}(r)$ results in samples from X.
- Technically, there is a problem with the method as stated above if the pdf or pmf of X is not strictly positive (F_X is not invertible). A more careful method should resolve that difficulty. In low dimensional cases, the above transformation works well. The basic problem of computing F_X and inverting it become difficult in high dimensions and other methods are necessary.

Discrete random variables: Gumbel Trick

It turns out that there is a neat trick based on Gumbel distribution that could be used to sample from a discrete random variable $\mathbf{x}$ taking values in $\{1, \ldots, K\}$ with probability proportional to $p_1, \ldots, p_k$ (note just known up to a normalization constant is enough for p_k.). A random variable $\mathbf{g} \in \mathbb{R}$ is called as standard Gumbel distribution if $\mathbf{g} = -\log(-\log(r))$ where $r \sim U[0, 1]$. We now have the following fact.

Fact

Let $\mathbf{x}$ be a discrete random variable taking values in $\{1, \ldots, K\}$ with $P(\mathbf{x} = k) \propto p_k$ and let $\mathbf{g}_k$ be a i.i.d. sequence of standard Gumbel random variables. Then

$$\mathbf{x} = \arg\max_k (\log(p_k) + \mathbf{g}_k)$$

Discrete random variables: Gumbel Trick

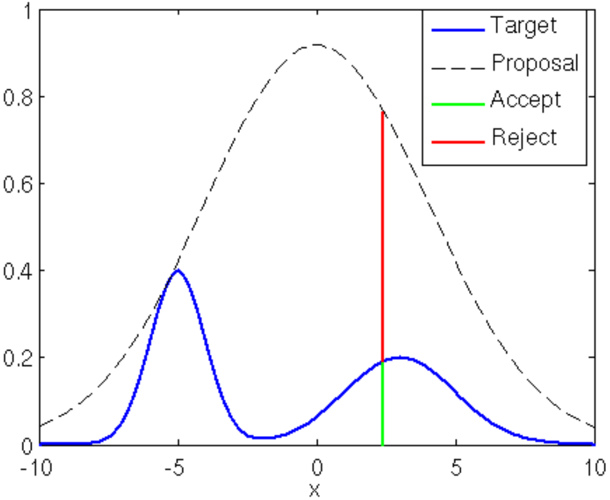
Hence, we have the following method for sampling:

1. Draw K i.i.d. standard Gumbel random variables, $\mathbf{g}_1, \ldots, \mathbf{g}_K$.
2. Add $\log(p_k)$ to the Gumbel random variables.
3. Take the value of $k \in \{1, \ldots, K\}$ that produces the maximum.

Rejection Sampling

- Again, we start with a one dimensional formulation. Assume that we want to sample from p, but we can sample from q instead, and we know further that $p(\mathbf{x}) \leq kq(\mathbf{x})$ everywhere for some constant k.
- Sampling $\mathbf{x}_i \sim q$ and then $r_i \sim U([0, kq(\mathbf{x}_i)])$ would give a pair $(\mathbf{x}_i, r_i)$ which would be uniformly distributed over the graph of the function (area under the function) kq.
- By rejecting the pair if $r_i \geq p(\mathbf{x})$ we ensure that the remaining sample pairs are uniformly distributed over the graph of $p(\mathbf{x})$. We can then discard the r_i and keep the $\mathbf{x}_i$ samples which constitute a sample from p.
- Rejection sampling can be modified for use if p is known up to a constant $p = c\tilde{p}$ (its normalization term is not easily computable). In this case we find q such that $\tilde{p} \leq kq$ and proceed as before.

Rejection Sampling



Adaptive Rejection Sampling

- Adaptive rejection sampling is way of adaptively computing q and k for distributions p whose logarithm is concave.
- In this case, we can upper bound the $\log p$ with a piecewise linear function (envelope) computed based on the derivative $\nabla \log p$ at different points.
- The distribution itself p is then upper bounded by a piecewise exponential function which constitutes the proposal q. As samples get rejected, they are added to the computation of the envelope and the quality of the upper bound improves. The difficulty here is as before in cases of high dimensionality.

Importance Sampling

Importance sampling directly estimates the expectation $\mathbb{E}_p(g)$ by noticing that

$$\mathbb{E}_p(g) = \int g(\mathsf{x}) \frac{p(\mathsf{x})}{q(\mathsf{x})} q(\mathsf{x}) \, d\mathsf{x} = \mathbb{E}_q(g \, p/q)$$

which is approximated by averaging $g(\mathsf{x}) \, p(\mathsf{x})/q(\mathsf{x})$ over samples from $\mathsf{x}_1, \ldots, \mathsf{x}_m \sim q$. A useful trick is performing importance sampling when we can't evaluate the normalization terms of p and q. In this case $p = \tilde{p}/Z_p$ and $q = \tilde{q}/Z_q$ and

$$\mathbb{E}_p(g) = \frac{Z_q}{Z_p} \int g(\mathsf{x}) \frac{\tilde{p}(\mathsf{x})}{\tilde{q}(\mathsf{x})} q(\mathsf{x}) \, d\mathsf{x} \approx \frac{Z_q}{Z_p} \frac{1}{m} \sum_{i=1}^{m} \frac{\tilde{p}(\mathsf{x}_i)}{\tilde{q}(\mathsf{x}_i)} g(\mathsf{x}_i)$$

where x_i are samples from q.

Importance Sampling

The factor Z_q/Z_p may be approximated as follows

$$\frac{Z_p}{Z_q} = \frac{1}{Z_q}\int \tilde{p}(\mathsf{x})\,d\mathsf{x} = \int \frac{\tilde{p}(\mathsf{x})}{\tilde{q}(\mathsf{x})} q(\mathsf{x})\,d\mathsf{x} \approx \frac{1}{m}\sum_{i=1}^{m} \frac{\tilde{p}(\mathsf{x}_i)}{\tilde{q}(\mathsf{x}_i)}$$

where $\mathsf{x}_i \sim q$. Putting all this together gives

$$\mathbb{E}_p(g) \approx \sum_{i=1}^{m} w_i g(\mathsf{x}_i) \qquad w_i = \frac{\tilde{p}(\mathsf{x}_i)/\tilde{q}(\mathsf{x}_i)}{\sum_{i=1}^{m} \tilde{p}(\mathsf{x}_i)/\tilde{q}(\mathsf{x}_i)} \qquad \mathsf{x}_i \sim q.$$

As before, the main problem is high dimensions. If p, q are high dimensional, weights $p(\mathsf{x}_i)/q(\mathsf{x}_i)$ become smaller rapidly. If q is low where pg is high, the estimator will be highly inaccurate since it may take a long time to obtain a meaningful sample.

Markov Chain Monte Carlo (MCMC)

We previously saw how samples can be used to approximate expectations

$$\frac{1}{m}\sum_{i=1}^{m} g(\mathsf{x}_i) \approx \mathbb{E}_p(g(\mathsf{x})) \quad \text{where} \quad \mathsf{x}_1, \ldots, \mathsf{x}_m \sim p.$$

We also saw a number of techniques for producing samples from a distribution *p* such as the histogram and transformation methods and rejection and importance sampling. Markov chain Monte Carlo (MCMC) is a collection of sampling methods that are based on following random walks on Markov chains.

Brief Introduction to Markov Chains

- Homogenous Markov chain $X_0, X_1, X_2, \ldots$ is a random processes that is completely characterized by the transition probabilities $P(X_n = y | X_{n-1} = z) = T(z, y)$ and initial probabilities $\pi_0(z) = P(X_0 = z)$.
- To simplify the notation we will assume that X_i are discrete and finite $X_i \in \{1, \ldots, k\}$ and we will consider π and T as a (row) vector and matrix of probabilities.
- For homogenous Markov processes conditional distribution of X_n given X_{n-1} is independent of $X_1, \ldots, X_{n-2}$.
- As a result, we have $P(X_1) = \pi_1 = \pi_0 T$. Similarly, $\pi_k = \pi_0 T^k$ and for large k, π_k tends to a unique stationary distribution π satisfying $\pi T = \pi$ (regardless of π_0) if the Markov chain characterized by T is ergodic.
- In other words, no matter what is the initial distribution π_0 (or where we start from) the resulting position distribution after k steps tends to the stationary distribution π for large k.

MCMC

- The idea of MCMC is to generate a random sample from p by following a random walk of k steps on a Markov chain T, for which p equals its stationary distribution π.
- Thus, no matter where we start, if we follow a random walk for a long period (called burn-in time) we will end up with a sample from its stationary distribution.
- If we want several samples, we can either (i) repeat the process several times (ii) take consecutive samples after the burn in time or (iii) follow a random walk and record every l-step as a sample.
- Approach (ii) will not produce independent samples and approach (iii) will result in approximately independent samples from π if l is sufficiently large.

MCMC

- To sample from p using MCMC, we need to design a Markov chain T whose stationary distribution π is p.
- To ensure that, it suffices to show that T satisfies the detailed balance property with respect to p

$$p_i T_{ij} = p_j T_{ji} \quad \forall\, i, j$$

since then

$$[pT]_i = \sum_j p_j T_{ji} = \sum_j p_i T_{ij} = p_i \sum_j T_{ij} = p_i \quad \Rightarrow \quad pT = p.$$

- We also need to ensure that the Markov chain described by T is ergodic so there will be a unique stationary distribution.
- One simple way to ensure ergodicity of T is to have $T_{ij} > 0$ for all i, j. It is useful to know (and easy to verify) that if we have several Markov chains $T_1, \ldots, T_l$ that satisfy the detailed balance property then a linear combination of them $\sum_i \alpha_i T_i$ would also satisfy it.

3 MCMC Algorithms

Based on the above introduction, we will discuss the following three types of MCMC sampling algorithms:

- Metropolis-Hastings
- Gibbs Sampling
- Langevin Monte Carlo

Basic Idea

Reference