COMP 480/580 — Probabilistic Algorithms and Data Structure Mar 7, 2018

Lecture 16

Lecturer: Anshumali Shrivastava *Scribe By: Anthony Chen*

1 Overview: Sampling Techniques

In the previous lecture, we introduced basic ideas and formulations for sampling. We answered the question: How do we sample from a distribution with CDF $F(x) = Pr(X \leq x)$ with access to only a uniform distribution $U \sim [0, 1]$? We covered inversion sampling, Monte Carlo estimation, importance sampling and rejection sampling. In this lecture, we will dive deeper into these methods and have a better understanding of the intuition behind sampling algorithms. We can roughly divide sampling techniques into the following three categories:

- **Transformation Based**
  - Inverse Transform
  - Box-Muller Method/Transform
  - Composition Transform
- **Rejection Based**
  - Rejection Sampling
- **Importance Sampling** (strictly speaking, this is an estimation technique not a sampling technique)

We will go over these different methods below.

2 Transformation Based Method

2.1 Inverse Transformation

First, we revisit the inverse transformation method we talked about in the last lecture. Inverse transformation method is a transformation-based method that uses the target function's inverse.

Inverse Transformation:

Goal: Sample x from a distribution that has CDF $F(x)$

Algorithm:

1. Sample $u \sim U[0, 1]$
2. Return $F^{-1}(u)$

The difficulty for the inverse transformation method is that we are not always able to compute F^{-1}. F^{-1} is application-dependent, and some important distributions, such as the Gaussian distribution, do not even have a closed form formula for CDF $F(x)$, let alone a closed-form solution for F^{-1}. Also, in the previous lecture we did not discuss discrete distributions. For discrete probability mass functions, we cannot calculate F^{-1} as we did in the continuous case, but the algorithm is still very simple and straightforward.

Inverse Transformation (Discrete):

Goal: Sample x from $(x_1, x_2, ..., x_n)$ proportional to probability $(p_1, p_2, ..., p_n)$.

Algorithm:

1. Sample $u \sim U[0,1]$
2. Return x_j if $\sum_{i=1}^{j-1} p_i \leq u \leq \sum_{i=1}^{j} p_i$

It should be noted that as $j \to \infty$, the term $\sum_{i=1}^{j-1} p_i \leq u \leq \sum_{i=1}^{j} p_i$ becomes a CDF. This CDF is (exactly) F^{-1}, where F is the (discrete) probability distribution we want to sample from. Thus, the discrete version of the inverse transformation method is (unsurprisingly) done in the same way as the continuous version of the inverse transformation method.

2.2 Box-Muller Transform

The Box-Muller Transform, proposed by George Edward Pelham Box and Mervin Edgar Muller, is a well-known sampling method for generating pairs of independent Gaussian variables with zero mean and unit variance. This method is important because sampling from a Gaussian distribution can be difficult. Since there is no closed form formula for the CDF of Gaussian distribution, we can't apply inverse transformation to sample from it. The Box-Muller Transform, on the other hand, manages to make the sampling feasible by using the following fact:

Consider a circle in polar coordinates as shown in **Figure 1**. Then, **Theorem 1** tells us that we can easily generate a pair of independent Gaussian variables if we can sample from the uniform distribution and the exponential distribution. Here, we will state **Theorem 1** without proof.

Theorem 1 *Let $X_1 = R \cdot \cos\theta$, $X_2 = R \cdot \cos\theta$ in polar coordinates. Then,*

$$X_1, X_2 \sim N(0,1) \Longleftrightarrow \theta \sim U[0, 2\pi], R^2 \sim Expo(\frac{1}{2})$$

Using the results from **Theorem 1**, we can easily generate Gaussian distributed random variables, as described in Box-Muller Transform below:

Box-Muller Transform:

Goal: Sample x_1, x_2 from joint Gaussian distribution with zero means, unit variance and zero covariance (independent).

Algorithm:

1. Sample $u_1, u_2 \sim U[0,1]$

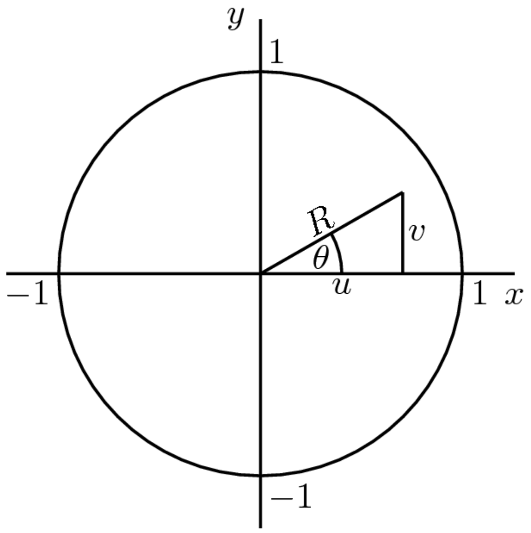

Figure 1: Box-Muller Transform in Polar Coordinates

2. Let $R = \sqrt{-2\log(u_1)}$, $\theta = 2\pi u_2$
3. Return $x_1 = R\cos\theta$, $x_2 = R\sin\theta$

2.3 Composition Transform

The Composition Transform can be used when the CDF we want to sample from is composed of other functions that we can sample from. Consider CDFs ($F(x)$) of the following form:

$$F(x) = \sum_{i=1}^{N} p_i \cdot F_i(x), \qquad f(x) = \sum_{i=1}^{N} p_i \cdot f_i(x)$$

where $0 \leq p_i \leq 1, \sum_i^N p_i = 1$. $F(x)$ can be interpreted as a linear combination of different CDF functions. This kind of distribution arises when we have a set of distributions $F_1, F_2, ...F_i, ...F_N$, and we generate a value X by choosing from the distributions with probability $p_1, p_2, ...p_i, ...p_N$. One example is the hyperexponential family. A hyperexponential distribution has the following PDF:

$$f(x) = \sum_{i=1}^{n} \alpha_i \lambda_i e^{-\lambda_i x}$$

How can we sample from a CDF in such form? We use the composition transform

Composition Transform:

Goal: Sample x from compositional CDF $F(x) = \sum_{i=1}^{N} p_i \cdot F_i(x)$

Algorithm:

1. generate I from $\{1, 2, ..., N\}$ with $Pr(I = j) = p_j$
2. generate a sample from F_I and return it

Proving that the algorithm is correct is straightforward. Consider the probability of our sample $y = k$:

$$Pr(y = k) = \sum_{i=1}^{N} Pr(I = i)F_i(k) = F(k)$$

3 Rejection Based Method

3.1 Rejection Sampling

In the previous lecture, we covered the rejection sampling method

Rejection Sampling:

Goal: Sample from $f(x)$

Given: Ability to sample from $f(x)$ such that $\forall x, f(x) \leq M \cdot g(x), M \geq 1$

Algorithm:

1. sample $x \sim g(x)$
2. sample $u \sim U[0, 1]$
3. accept x if $u \leq \frac{f(x)}{M \cdot g(x)}$

However, we did not prove that rejection sampling generates samples $\sim f(x)$. Here we show the proof.

Proof: We want to show

$$Pr(X|\text{Accepted}) = f(x)$$

Using the Bayesian Rule:

$$Pr(X|\text{Accepted}) = \frac{Pr(X, \text{Accepted})}{Pr(\text{Accepted})} = \frac{Pr(\text{Accepted}|X) \cdot Pr(X)}{Pr(\text{Accepted})}$$

We can make the following substitutions:

$$Pr(X) = g(x)$$

$$Pr(\text{Accepted}|X) = Pr\left(u < \frac{f(x)}{M \cdot g(x)}|x\right) = \frac{f(x)}{M \cdot g(x)}$$

Here, we suppose that x is a discrete random variable, but the proof works for continuous random variables by replacing the summation with an integral.

$$Pr(\text{Accepted}) = \sum_x Pr(\text{Accepted}|X = x) \cdot Pr(X = x)$$

$$= \sum_x \frac{f(x)}{M \cdot g(x)} \cdot g(x) = \sum_x \frac{f(x)}{M} = \frac{1}{M}$$

Thus,

$$Pr(X|\text{Accepted}) = \frac{Pr(\text{Accepted}|X) \cdot Pr(X)}{Pr(\text{Accepted})} = \frac{f(x)}{M \cdot g(x)} \cdot g(x) \cdot M = f(x)$$

which completes the proof.

4 Sampling Application: Monte Carlo Integration

4.1 Comparison to numerical methods

One of the most important practical uses for sampling is estimation. For many applications of practical interest, we wish to estimate the integral of a "difficult" function that does not have a clean closed-form solution. Rather than estimate the integral using methods based on Riemann sums and approximations, we can use rejection-based sampling or importance sampling to estimate the Monte Carlo integration of a function instead. An example would be using Monte Carlo method to estimate the value of π. Here, we integrate to find the area of a circle of known radius. We can then use the value of integral to compute an approximation $\hat{\pi}$. Using rejection sampling, we draw random points from a circle inside a square and only accept the point if it falls into the circle.

However, we already have existing deterministic methods based on the Riemann sum to estimate integrals. Why should we use Monte Carlo methods, which are inherently random? As dimension of the problem goes up, the complexity of deterministic numerical approximation goes up exponentially. In high-dimensional space, all of the deterministic numerical methods for integral approximation become inefficient.

4.2 Curse of Dimensionality

However, other issues arise when we use sampling in high-dimensional space. Consider the above example of estimating π by sampling points in a circle: As the dimension goes up, we are no longer sampling from a circle but rather from a high-dimensional sphere in a high-dimensional cube[1]. It turns out that as the dimensionality $d \longrightarrow \infty$ the fraction of the sphere volume to the cube volume $\frac{V(sphere)}{V(cube)} \longrightarrow 0$. This result (and other similar results) are collectively called the curse of dimensionality, which is a phenomenon that arises in high-dimensional spaces. In general, our everyday intuition that applies very well to low-dimensional spaces does not generalize to high-dimensional spaces. Our algorithms tend to behave in strange, counterintuitive, and unpleasant ways. Methods which are optimal in low dimensions must be replaced by methods for high-dimensional versions of the problem. In the case of rejection sampling, the curse of dimensionality implies that when we try to filter points in high-dimensional space (as in our example), we will almost always get points that are in the hypercube but not in the hypersphere. This means we reject many, many points - making our sampling procedure inefficient. The expectation of our estimation is still correct, but the variance of rejection sampling will be very bad. As a result, in high-dimensional spaces, both numerical and sampling-based methods have difficulty with estimation, but the hardness comes in different ways.

[1]Editor's note: We will sometimes refer to a high-dimensional sphere as a *hypersphere* and a high-dimensional cube as a *hypercube*.