COMP 480/580 — Probabilistic Algorithms and Data Structure Oct 22, 2024

Lecture 14

Lecturer: Anshumali Shrivastava Scribe By: Marko Tanevski (mt102), Samek Rangarajan (sr118), Daniel Zhang (dfz1), Pranav Suryadevara (ps102)

1 Sampling

Consider the scenario in which you have access to a uniform pseudo-Random Number Generator (pRNG) $U[0,1]$ that chooses a real number between 0 and 1 with uniform probability. Given this generator, is it possible to sample a number from a standard Gaussian distribution? In general, given any distribution D with a PDF f, is it possible to use $U[0,1]$ to sample an $x \sim D$, assuming we can compute $f(x)$ for any x?

1.1 Inversion Method

Let's first look at how to generate any distribution with an invertible CDF from $U[0,1]$. Our goal is, at each step k, to generate an $x_k \sim D$ for some distribution D where $f(x)$ is its PDF, and its CDF is the invertible $F(x)$. In particular, if we were to sample x_k 50 times and D was a standard Gaussian distribution, we would like to be able to generate x_k's something like the points in the picture below:

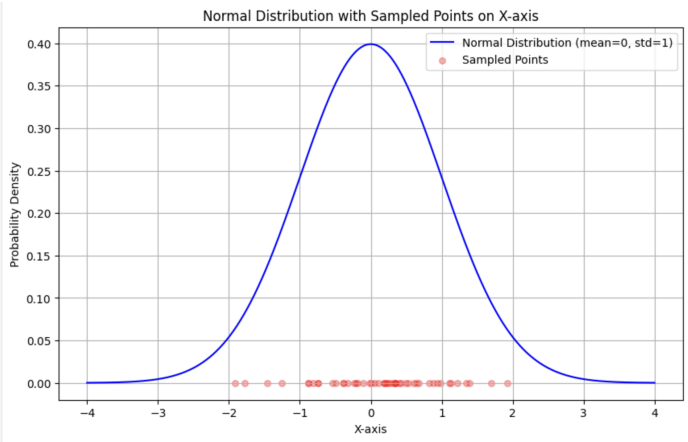

Figure 1: Sampling 50 Points from a Normal Distribution

How can we do this? Well, at step k, generate a $y_k \sim U[0,1]$, and then return $x_k = F^{-1}(y_k)$. Why is does this work? Well, in continuous cases

$$F(a) = \int_{-\infty}^{a} f(x)dx$$

which is equivalent to

$$P(x \leq a) = F(a), \quad x \sim D.$$

If we make the assumption that this F has a nice form (e.g., Gaussian, Laplacian), we should be able to approximate its inverse, as reasoned above. However, many of the F's we like do not actually have a form, we will see this in the following lectures. Being able to approximate $F^{-1}(x)$, while doable for simple functions, is almost impossible for more complex ones. This aside, if we can calculate $F^{-1}(x)$, then below is the proof for the aforementioned described sampling algorithm:

$$P(x_k < a) = P[F(x_k) < F(a)] = P[y_k < F(a)] = F(a)$$

as $y_k \sim U[0, 1]$. Note, the first equality holds true because the CDF is a monotonic function. Thus, we have shown that the values of x_k follow the definition of the CDF of f, and so we can say x_k are sampled from f. Note that in practice F is not invertible so this method is not very useful in practice.

1.2 Sampling Goal

The primary goal of sampling in computational contexts is estimation. Sampling allows us to estimate quantities that are otherwise too complex or computationally expensive to calculate directly.

For example, consider estimating the energy of a molecular configuration. If we want to approximate the potential energy of all molecules, we could use the energy function

$$\frac{1}{|\text{Molecule}|} \sum_{m \in \text{Molecule}} \mathcal{P}(m)$$

where $\mathcal{P}(m)$ represents the potential energy function for each molecule. Since this sum can be large and computationally intensive, we can instead sample from this distribution. By drawing samples m_i such that $m_i \propto \mathcal{P}(m_i)$, we can approximate the expected energy of a molecule and estimate the sum above.

Another powerful application of sampling is in estimating the hypervolume of a shape, as shown in Figure 2, where we are estimating the area of the chaotic figure. Even without an exact formula for the hypervolume, we can use Monte Carlo methods to estimate it accurately by leveraging fast checks for whether a point lies inside the shape. By sampling from a well-chosen distribution, we can gain insight into complex spaces with relative ease. Of course, we do need to be able to check whether a point lies inside a shape or not, but this is often much easier to do than computing its hypervolume.

An important prerequisite for estimation is the ability to sample uniformly from the space of interest, especially when dealing with high-dimensional or complex spaces. Uniform sampling ensures that the estimates we generate are accurate.

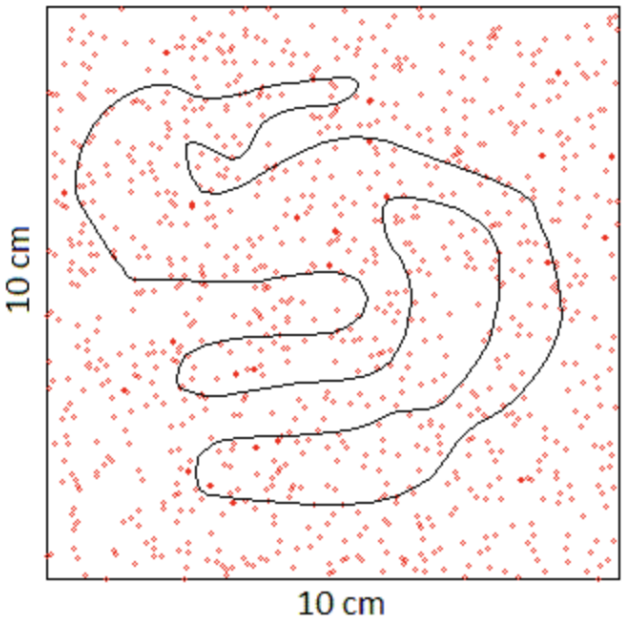

Figure 2: Monte Carlo Hypervolume Estimation [2]

For example, by sampling $n = 1,000$ points above with k points inside, we can estimate the area of the shape as $\frac{k}{n} \cdot V$ where $V = 100cm^2$ is the hypervolume of the entire space.

The power of this is that the estimation accuracy is independent of n (although this is not the case for estimation variance or standard error, which is dependent on n).

1.3 Rejection Sampling

Rejection sampling is another sampling method that is useful in multidimensional sampling. Multidimensional sampling is useful in Markov chains because the state space is multidimensional.

In rejection sampling, the goal is to sample $x \sim f$ (where f is the target distribution). However, assume we can't sample from f, and unlike in the Inversion Method, assume that we don't actually have the CDF of f at hand. This is often the case. Rejection sampling now tells us "don't worry about it" as long as we sample from $x \sim g$ from any proposed distribution g we can sample from. The only condition is that $\exists M > 0$ such that

$$f(x) < Mg(x) \quad \forall x$$

which is identical to saying g can't be 0 if f is nonzero or that support$(f) \subseteq$ support(g).

Our rejection algorithm is as follows:

1. Choose $u \sim U[0, 1]$ and $x_k \sim f$.
2. If $u < \frac{f(x_k)}{Mg(x_k)}$ return x_k, else return to 1.

Thus we have created rejection / acceptance criteria for whether we reject or we accept a value of $x_k \sim f$ or not. See Figure 3 below to visualize this.

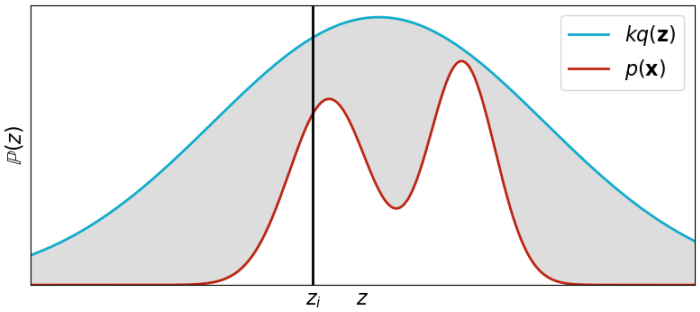

Figure 3: Rejection sampling [1]. The intuition here would be is as follows. Let $q(z)$ and $p(x)$ be our distributions, where $p(x)$ is the one we want to sample from, but cannot. Sample z_i from $kq(z)$, where k is some constant. The black line is where our z_i lies. Now, sample uniformly $u \in [0, 1]$. Then, mark u on the line of z_i. If that marked point lies below the red line of $p(x)$, then return z_i. Otherwise, repeat. Essentially, we are generating a point (z_i, u), and if that point lies in the area under the curve of $p(x)$, then we accept that point as something that could have been generated by $p(x)$, and thus accept z_i.

We will now prove the correctness of rejection sampling. We want to show that the random variable outputted by the algorithm is equal the CDF of the distribution we want to sample from.

Let f be the distribution we want to sample from with CDF of F, and let g be the distribution we can sample from, with some $M \geq 1$ satisfying $f(x) < Mg(x)$ for all x. Furthermore, let $X \sim f$, let $Y \sim g$, and let $U \sim [0, 1]$. The random variable returned by rejection sampling is $Y \mid U < \frac{f(Y)}{Mg(Y)}$; that is, it returns Y given that U is smaller than $\frac{f(Y)}{Mg(Y)}$. We want to show that $P\big(Y \leq y \mid U \leq \frac{f(Y)}{Mg(Y)}\big) = F(y)$. The steps are below.

$$
\begin{aligned}
P\left(Y \leq y \mid U \leq \frac{f(Y)}{Mg(Y)}\right) &= \frac{P\left(Y \leq y, U \leq \frac{f(Y)}{Mg(Y)}\right)}{P\left(U \leq \frac{f(Y)}{Mg(Y)}\right)} \\
&= \frac{\int_{-\infty}^{y} \left(\int_0^{\frac{f(t)}{Mg(t)}} 1 \cdot du \right) g(t)\, dt}{\int_{-\infty}^{\infty} \left(\int_0^{\frac{f(t)}{Mg(t)}} 1 \cdot du \right) g(t)\, dt} \\
&= \frac{\int_{-\infty}^{y} \left(\frac{f(t)}{Mg(t)} \right) g(t)\, dt}{\int_{-\infty}^{\infty} \left(\frac{f(t)}{Mg(t)} \right) g(t)\, dt} \\
&= \frac{\int_{-\infty}^{y} \frac{f(t)}{M} dt}{\int_{-\infty}^{\infty} \frac{f(t)}{M} dt} \\
&= \frac{\int_{-\infty}^{y} f(t)\, dt}{\int_{-\infty}^{\infty} f(t)\, dt} \\
&= \frac{\int_{-\infty}^{y} f(t)\, dt}{1} \\
&= \int_{-\infty}^{y} f(t)\, dt = F(t)
\end{aligned}
$$

Therefore, the CDF of the output of the rejection sampling algorithm is the same as that of f, so it is correct.

Again, note the difference between being able to sample f and being able to calculate it. Here, we can evaluate f and g, but we are unable to invert F (and as such sample it). This is very different.

Note this will work for any f if g is the normal distribution. However, as seen in Figure 3, the probability of accepting is very low, if f and g are very different; imagine if the gray area was much larger, how much more unlikely it would be to accept the result. Thus, ideally, when performing rejection sampling we want f and g with similar distributions.

We can calculate how good our choice of g is by whether we are rejecting a lot or not. If we are rejection very rarely, then we've definitely picked a good g.

Note, this is kind of like the chicken and the egg problem: we want to sample from f, but it is very hard to sample from f without being able to sample from g similar to f, but if we have such a g we might as well be able to sample from f. Regardless, it is a useful tool that sometimes can be very beneficial even for exotic distributions f. People do their entire PhDs in statistical physics on identifying such f's and g's.

2 Monte Carlo Estimation

The goal of Monte Carlo estimation is to estimate the expected value of a function $Q(x)$ over a probability distribution $p(x)$, where direct computation of the integral is challenging. The expectation we want to estimate is:

$$I = \mathbb{E}[Q(x)] = \int Q(x)p(x)dx$$

For instance, say we want to compute:

$$I = \int_0^1 xe^{-x^3}dx = \mathbb{E}[xe^{-x^3}] \quad \text{for} \quad x \sim U[0,1].$$

This integral can be approximated using Monte Carlo estimation by taking random samples $x_i \sim U[0,1]$ and averaging the function values $Q(x_i) = x_i e^{-x_i^3}$.

In general, given a probability distribution $p(x)$, a function whose expected value we wish to compute $Q(x)$, and n number of samples drawn from $p(x)$, the Monte Carlo estimator for $\mathbb{E}[Q(x)]$ is given by:

$$\hat{I}_n = \frac{1}{n}\sum_{i=1}^{n} Q(x_i).$$

The proof is presented below:

- **Unbiasedness:** We show that $\hat{I}_n$ is an unbiased estimator for $\mathbb{E}[Q(x)]$. The expected value of $\hat{I}_n$ is:

  $$\mathbb{E}[\hat{I}_n] = \mathbb{E}\left[\frac{1}{n}\sum_{i=1}^{n} Q(x_i)\right].$$

  Using the linearity of expectation:

  $$\mathbb{E}[\hat{I}_n] = \frac{1}{n}\sum_{i=1}^{n} \mathbb{E}[Q(x_i)].$$

Since the x_i's are i.i.d., $\mathbb{E}[Q(x_i)] = \mathbb{E}[Q(x)]$, and thus:

$$\mathbb{E}[\hat{I}_n] = \mathbb{E}[Q(x)].$$

Therefore, $\hat{I}_n$ is an unbiased estimator of $\mathbb{E}[Q(x)]$.

- **Variance:** The variance of $\hat{I}_n$ is given by:

$$\text{Var}(\hat{I}_n) = \text{Var}\left(\frac{1}{n}\sum_{i=1}^{n} Q(x_i)\right).$$

Since the x_i's are i.i.d., we can use the property of the variance of a sum:

$$\text{Var}(\hat{I}_n) = \frac{1}{n^2}\sum_{i=1}^{n} \text{Var}(Q(x_i)) = \frac{1}{n}\text{Var}(Q(x)).$$

Thus, the variance decreases proportionally to $1/n$, meaning that as n increases, the estimator becomes more accurate.

- **Law of Large Numbers:** By the law of large numbers, the Monte Carlo estimator $\hat{I}_n$ converges to the true expected value as $n \to \infty$. Specifically:

$$\lim_{n\to\infty} \mathbb{P}\left(\left|\hat{I}_n - \mathbb{E}[Q(x)]\right| > \varepsilon\right) = 0 \quad \text{for any} \quad \varepsilon > 0.$$

This guarantees that as more samples are taken, the Monte Carlo estimate will approach the true expected value.

- **Conclusion:** Monte Carlo estimation provides a flexible and unbiased method for approximating integrals, especially when they are difficult to compute analytically. As the number of samples increases, the estimator becomes more accurate, and the variance of the estimator decreases. For integrals like $\int_0^1 xe^{-x^3}dx$, Monte Carlo estimation offers a practical approach to obtaining numerical approximations.

2.1 Importance Sampling

The goal of importance sampling is to estimate the expectation of a function $w(x)$ under a PDF $g(x)$ that is difficult to sample from directly. Let this expectation be defined as I

$$I = \mathbb{E}[w(x)]_{x\sim g} = \int h(x)w(x)dx$$

For example, imagine $g(x)$ is the probability distribution for molecules in a container and $w(x)$ measures the energy of each molecule. Our goal is to compute the average of $w(x)$ over all molecular configurations according to $g(x)$. Sampling from the configurations may be infeasible because of the number of molecules, so we turn to importance sampling to get an accurate approximation without direct sampling. In a discrete setting, the expectation would look like this:

$$\sum_{i=1}^{n} w(i) \times g(i) \quad \text{where } g(i) = P(X = i)$$

Now, consider the continuous setting. We have the following assumptions:

- $g(x)$ is the target distribution that we can't sample from.
- $w(x)$ is a function we want to take the expectation of under $g(x)$.
- $f(x)$ is another PDF with the following conditions
  1. $f(x) > 0$ for all x where $g(x) > 0$
  2. $f(x)$ is easier to sample from than $g(x)$

Instead of sampling from $g(x)$, we sample from $f(x)$ and apply an importance weight to each sample to adjust for the fact that we're not sampling from $g(x)$ directly. The importance weight, $\frac{g(x)}{f(x)}$, adjusts for the difference between $g(x)$ and $f(x)$ at each point x. Thus our estimator is

$$\frac{1}{n}\sum_{i=1}^{n} w(x_i) \times \frac{g(x_i)}{f(x_i)} \quad x_i \sim f.$$

The proof that this works is presented below:

- Start with the expectation under $g(x)$

$$I = \mathbb{E}[w(x)]_{x\sim g} = \int w(x) \times g(x)dx$$

- Because $f(x) > 0$ whenever $g(x) > 0$, we can multiply and divide by $f(x)$ inside the integral without changing the value

$$= \int w(x) \times \frac{g(x)}{f(x)} \times f(x)dx$$

- If we observe the integral as an expectation with respect to $f(x)$

$$= \mathbb{E}\left[w(x) \times \frac{g(x)}{f(x)}\right]_{x\sim f}$$

  Therefore,

$$\mathbb{E}[w(x)]_{x\sim g} = \mathbb{E}\left[w(x) \times \frac{g(x)}{f(x)}\right]_{x\sim f}$$

This shows that sampling from an easier distribution $f(x)$ and applying the importance weight can accurately approximate expectations with respect to $g(x)$. This makes estimation more efficient when $g(x)$ is complex.

References

[1] Gregory Gunderson. Sampling: Two basic algorithms, 2019. Last accessed 26 October 2024.

[2] Math on Web. Finding areas using the monte carlo method, 2024. Last accessed 29 October 2024.