12 Monte Carlo Methods

Monte Carlo is an umbrella term referring to a set of numerical techniques for solving one or both of these problems:

1. Approximating expected values that cannot be solved in closed-form
2. Sampling from distributions for which is a simple sampling algorithm is not available.

Recall that expectation of a function $\phi(\mathbf{x})$ of a continuous variable $\mathbf{x}$ with respect to a distribution $p(\mathbf{x})$ is defined as:

$$E_{p(\mathbf{x})}[\phi(\mathbf{x})] \equiv \int p(\mathbf{x})\phi(\mathbf{x})d\mathbf{x} \tag{1}$$

Monte Carlo methods approximate this integral by drawing N samples from $p(\mathbf{x})$

$$\mathbf{x}_i \sim p(\mathbf{x}) \tag{2}$$

and then approximating the integral by the weighted average:

$$E_{p(\mathbf{x})}[\phi(\mathbf{x})] \approx \frac{1}{N}\sum_{i=1}^{N}\phi(\mathbf{x}_i) \tag{3}$$

Estimator properties. This estimate is unbiased:

$$E_{p(\mathbf{x}_{1:N})}\left[\frac{1}{N}\sum_i \phi(\mathbf{x}_i)\right] = \frac{1}{N}\sum_i E_{p(\mathbf{x}_i)}[\phi(\mathbf{x}_i)] = \frac{1}{N}NE_{p(\mathbf{x})}[\phi(\mathbf{x})] = E_{p(\mathbf{x})}[\phi(\mathbf{x})] \tag{4}$$

Furthermore, the variance of this estimate is inversely proportional to the number of samples:

$$\text{var}_{p(\mathbf{x}_{1:N})}\left[\frac{1}{N}\sum_i \phi(\mathbf{x}_i)\right] = \frac{1}{N^2}\sum_i \text{var}_{p(\mathbf{x}_{1:N})}[\phi(\mathbf{x}_i)] = \frac{1}{N^2}N\text{var}_{p(\mathbf{x}_i)}[\phi(x_i)] = \frac{1}{N}\text{var}_{p(\mathbf{x})}[\phi(\mathbf{x})] \tag{5}$$

Hence, the more samples we get, the better our estimate will be; in the limit, the estimator will converge to the true value.

Dealing with unnormalized distributions. We often wish to compute the expected value of a distribution for which evaluating the normalization constant is difficult. For example, the posterior distribution over parameters $\mathbf{w}$ given data D is:

$$p(\mathbf{w}|D) = \frac{p(D|\mathbf{w})p(\mathbf{w})}{p(D)} \tag{6}$$

The posterior mean and covariance ($\bar{\mathbf{w}} = E[\mathbf{w}]$ and $E[(\mathbf{w} - \bar{\mathbf{w}})(\mathbf{w} - \bar{\mathbf{w}})^T]$) can be useful to understand this posterior, i.e., what we believe the parameter values are "on average," and how much uncertainty there is in the parameters. The numerator of $p(\mathbf{w}|D)$ is typically easy to compute, but $p(D)$ entails an integral which is often intractable, and thus must be handled numerically.

Most generally, we can write the problem as computing the expected value with respect to a distribution $p(\mathbf{x})$ defined as

$$p(\mathbf{x}) \equiv \frac{1}{Z} P^*(\mathbf{x}), \qquad Z = \int P^*(\mathbf{x}) d\mathbf{x} \tag{7}$$

Monte Carlo methods will allow us to handle distributions of this form.

12.1 Sampling Gaussians

We begin with algorithms for sampling from a Gaussian distribution.

For the simple 1-dimensional case, $x \sim \mathcal{N}(0, 1)$, there is well known algorithm called the Box-Muller Method that is based on an approach called rejection sampling. It is implemented in Matlab in the command `randn`.

For a general 1D Gaussian, $x \sim \mathcal{N}(\mu, \sigma^2)$, we sample a variable $z \sim \mathcal{N}(0, 1)$, and then set $x = \sigma z + \mu$. You should be able to show that x has the desired mean and variance.

For the multi-dimensional case, $\mathbf{x} \sim \mathcal{N}(0, \mathbf{I})$, each element is independent and Gaussian: $x_i \sim \mathcal{N}(0, 1)$ and so each element can be sampled with `randn`.

To sample from a Gaussian with general mean vector $\boldsymbol{\mu}$ and variance matrix $\boldsymbol{\Sigma}$ we first sample $\mathbf{z} \sim \mathcal{N}(0, \mathbf{I})$, and then set $\mathbf{x} = \mathbf{L}\mathbf{z} + \boldsymbol{\mu}$, where $\boldsymbol{\Sigma} = \mathbf{L}\mathbf{L}^T$. We can compute $\mathbf{L}$ from $\boldsymbol{\Sigma}$ by the Cholesky Factorization of $\boldsymbol{\Sigma}$, which must be positive definite. Then we have

$$E[\mathbf{x}] = E[\mathbf{L}\mathbf{z} + \boldsymbol{\mu}] = \mathbf{L}E[\mathbf{z}] + \boldsymbol{\mu} = \boldsymbol{\mu} \tag{8}$$

and

$$E[(\mathbf{z} - \boldsymbol{\mu})(\mathbf{z} - \boldsymbol{\mu})^T] = E[\mathbf{L}\mathbf{z}(\mathbf{L}\mathbf{z})^T] = \mathbf{L}E[\mathbf{z}\mathbf{z}^T]\mathbf{L}^T = \mathbf{L}\mathbf{L}^T = \boldsymbol{\Sigma} \tag{9}$$

12.2 Importance Sampling

In some situations, it may be difficult to sample from the desired distribution $p(\mathbf{x})$; however, we can sample from a similar distribution $q(\mathbf{x})$. Importance sampling is a technique that allows one to approximate expectation with respect to $p(\mathbf{x})$ by sampling from $q(\mathbf{x})$. The only requirement on q is that it have the same support as p, i.e., q is nonzero everywhere that p is nonzero.

Importance sampling is based on the following equality:

$$E_{q(\mathbf{x})}\left[\frac{p(\mathbf{x})}{q(\mathbf{x})}\phi(\mathbf{x})\right] = \int \frac{p(\mathbf{x})}{q(\mathbf{x})}\phi(\mathbf{x})q(\mathbf{x})d\mathbf{x} \tag{10}$$

$$= \int \phi(\mathbf{x})p(\mathbf{x})dx \tag{11}$$

$$= E_{p(\mathbf{x})}\left[\phi(\mathbf{x})\right] \tag{12}$$

In other words, we can compute the desired expectation by sampling values $\mathbf{x}_i$ from $q(\mathbf{x})$, and then computing

$$E_q\left[\frac{p(\mathbf{x})}{q(\mathbf{x})}\phi(\mathbf{x})\right] \approx \frac{1}{N}\sum_i \frac{p(\mathbf{x}_i)}{q(\mathbf{x}_i)}\phi(\mathbf{x}_i) \tag{13}$$

It often happens that p and/or q are known only up to multiplicative constants. That is,

$$p(\mathbf{x}) \equiv \frac{1}{Z_p}P^*(\mathbf{x}) \tag{14}$$

$$q(\mathbf{x}) \equiv \frac{1}{Z_q}Q^*(\mathbf{x}) \tag{15}$$

where P^* and Q^* are easy to evaluate but the constants Z_p and Z_q are not.

Then we have:

$$E_{p(\mathbf{x})}[\phi(\mathbf{x})] = \int \frac{\frac{1}{Z_p}P^*(\mathbf{x})}{\frac{1}{Z_q}Q^*(\mathbf{x})}\phi(\mathbf{x})q(\mathbf{x})dx = \frac{Z_q}{Z_p}E_{q(\mathbf{x})}\left[\frac{P^*(\mathbf{x})}{Q^*(\mathbf{x})}\phi(\mathbf{x})\right] \tag{16}$$

and so it remains to approximate $\frac{Z_q}{Z_p}$. If we substitute $\phi(\mathbf{x}) = 1$, the above formula states that

$$\frac{Z_q}{Z_p}E_{q(x)}\left[\frac{P^*(x)}{Q^*(x)}\right] = 1 \tag{17}$$

and so $\frac{Z_p}{Z_q} = E_{q(\mathbf{x})}[\frac{P^*(\mathbf{x})}{Q^*(\mathbf{x})}]$. Thus we have:

$$E_{p(\mathbf{x})}[\phi(\mathbf{x})] = \frac{E_{q(\mathbf{x})}\left[\frac{P^*(\mathbf{x})}{Q^*(\mathbf{x})}\phi(\mathbf{x})\right]}{E_{q(\mathbf{x})}\left[\frac{P^*(\mathbf{x})}{Q^*(\mathbf{x})}\right]} \tag{18}$$

Hence, the importance sampling algorithm is:

1. Sample N values $\mathbf{x}_i \sim q(\mathbf{x}_i)$
2. Compute
$$w_i = \frac{P^*(\mathbf{x}_i)}{Q^*(\mathbf{x}_i)} \tag{19}$$
3. Estimate the expected value
$$E[\phi(\mathbf{x})] \approx \frac{\sum_i w_i\phi(\mathbf{x}_i)}{\sum_i w_i} \tag{20}$$

The importance sampling algorithm will only work well when $q(\mathbf{x})$ is sufficiently similar to the function $p(\mathbf{x})|\phi(\mathbf{x})|$. Put more concretely, the variance of the estimator grows as the dissimilarity between $q(\mathbf{x})$ and $p(\mathbf{x})|\phi(\mathbf{x})|$ grows (and is minimized when they are equal). An alternative is to use the MCMC algorithm to draw samples directly from $p(\mathbf{x})$, as described below.

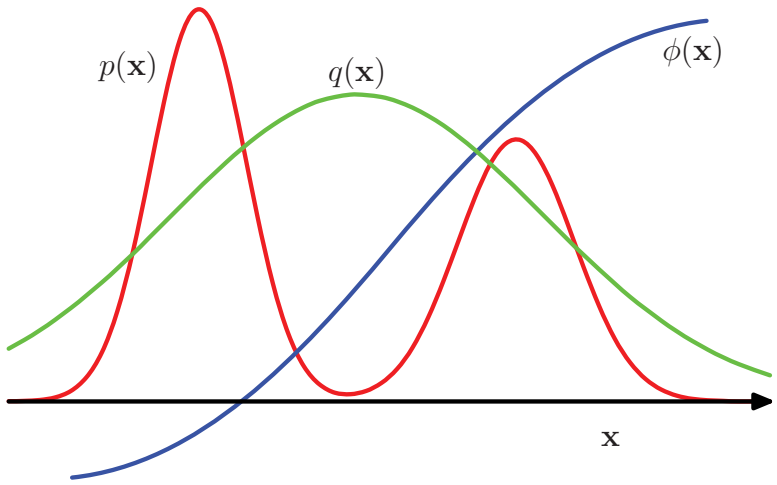

Figure 1: Importance sampling may be used to sample relatively complicated distributions like this bimodal $p(\mathbf{x})$ by instead sampling simpler distributions like this unimodal $q(\mathbf{x})$. Note that in this example, sampling from $q(\mathbf{x})$ will produce many samples that will be given a very low weight since $q(\mathbf{x})$ has a lot of mass where $p(\mathbf{x})$ is near zero (in the center of the plot). On the other hand, $q(\mathbf{x})$ has ample mass around the two modes of $p(\mathbf{x})$ and so it is a relatively good choice. If $q(\mathbf{x})$ had very little mass around one of the modes of $p(\mathbf{x})$, the estimate given by importance sampling would have a very high variance (unless $|\phi(\mathbf{x})|$ was small enough there to compensate for the difference). (Figure from *Pattern Recognition and Machine Learning* by Chris Bishop.)

12.3 Markov Chain Monte Carlo (MCMC)

MCMC is a very general algorithm for sampling from any distribution. For example, there is no simple method for sampling models $\mathbf{w}$ from the posterior distribution except in specialized cases (e.g., when the posterior is Gaussian).

MCMC is an iterative algorithm that, given a sample $\mathbf{x}_t \sim p(\mathbf{x})$, modifies that sample to produce a new sample $\mathbf{x}_{t+1} \sim p(\mathbf{x})$. This modification is done using a proposal distribution $q(\mathbf{x}'|\mathbf{x})$, that, given a $\mathbf{x}$, randomly selects a "mutation" to $\mathbf{x}$. This proposal distribution may be almost anything, and it is up to the user of the algorithm to choose this distribution; a common choice would be simply a Gaussian centered at $\mathbf{x}$: $q(\mathbf{x}'|\mathbf{x}) = \mathcal{N}(\mathbf{x}'|\mathbf{x}, \sigma^2\mathbf{I})$.

The entire algorithm is:

select initial point $\mathbf{x}_1$
$t \leftarrow 1$
loop
  Sample $\mathbf{x}' \sim q(\mathbf{x}'|\mathbf{x}_t)$
  $\alpha \leftarrow \frac{P^*(\mathbf{x}')}{P^*(\mathbf{x}_t)} \frac{q(\mathbf{x}_t|\mathbf{x}')}{q(\mathbf{x}'|\mathbf{x}_t)}$
  Sample $u \sim \text{Uniform}[0, 1]$
  if $u \leq \alpha$ **then**
    $\mathbf{x}_{t+1} \leftarrow \mathbf{x}'$
  else
    $\mathbf{x}_{t+1} \leftarrow \mathbf{x}_t$
  end if
  $t \leftarrow t + 1$
end loop

Amazingly, it can be shown that, if $\mathbf{x}_1$ is a sample from $p(\mathbf{x})$, then every subsequent $\mathbf{x}_t$ is also a sample from $p(\mathbf{x})$, if they are considered in isolation. The samples are correlated to each other via the Markov Chain, but the marginal distribution of any individual sample is $p(\mathbf{x})$.

So far we assumed that $\mathbf{x}_1$ is a sample from the target distribution, but, of course, obtaining this first sample is itself difficult. Instead, we must perform a process called **burn-in**: we initialize with any $\mathbf{x}_1$, and then discard the first T samples obtained by the algorithm; if we pick a large enough value of T, we are guaranteed that the remaining samples are valid samples from the target distribution. However, there is no exact method for determining a sufficient T, and so heuristics and/or experimentation must be used.

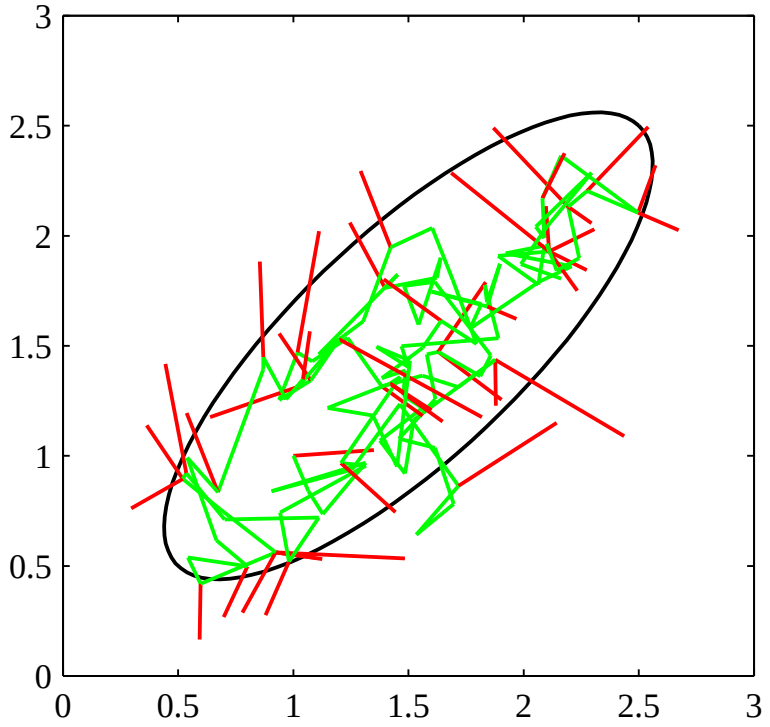

Figure 2: MCMC applied to a 2D elliptical Gaussian with a proposal distribution consisting of a circular Gaussian centered on the previous sample. Green lines indicate accepted proposals while red lines indicate rejected ones. (Figure from *Pattern Recognition and Machine Learning* by Chris Bishop.)

