Importance Sampling

Gianluca Bencomo

August 6, 2023

Abstract

Importance Sampling forms the conceptual springboard for many Markov Chain Monte Carlo (MCMC) techniques. In particular, particle filters require a strong understanding of the topics discussed here. This note treats importance sampling in a commonly seen formalization and its critical variation: *sampling importance resampling*. The goal is to provide the reader with an alternative treatment of importance sampling that may assist students reading more classical texts on the topic.

1 Introduction

In Bayesian inference, we can commonly reduce problems to an expectation of the following form:

$$\mathbb{E}\left[\boldsymbol{f}(\boldsymbol{x})\right] = \int \boldsymbol{f}(\boldsymbol{x})p(\boldsymbol{x})d\boldsymbol{x}$$

where $\boldsymbol{f} : \mathbb{R}^N \to \mathbb{R}^M$ is some arbitrary function and $p(\boldsymbol{x})$ is a probability density function over $\boldsymbol{x}$ that is often a posterior distribution of interest. For example, suppose we have some neural network $\boldsymbol{f}_{\boldsymbol{\theta}} : \mathcal{X} \to \mathcal{Y}$ that maps inputs $\boldsymbol{x} \in \mathcal{X}$ to labels $\boldsymbol{y} \in \mathcal{Y}$ and is parameterized by $\boldsymbol{\theta}$. Given some dataset $\mathcal{D} = \{(\boldsymbol{x}_1, \boldsymbol{y}_1), \ldots, (\boldsymbol{x}_N, \boldsymbol{y}_N)\}$, a Bayesian approach to neural networks looks to compute

$$p(\boldsymbol{\theta}|\mathcal{D}) = \frac{p(\mathcal{D}|\boldsymbol{\theta})p(\boldsymbol{\theta})}{p(\mathcal{D})}.$$

When deploying $\boldsymbol{f}_{\boldsymbol{\theta}}$, a sensible choice for $\boldsymbol{\theta}$ might be

$$\boldsymbol{\theta}^* = \mathbb{E}\left[\boldsymbol{\theta}|\mathcal{D}\right] = \int \boldsymbol{\theta}p(\boldsymbol{\theta}|\mathcal{D})d\boldsymbol{\theta}.$$

Alternatively, if we wanted to produce an ensemble classifier, we could construct something on the order of

$$\mathbb{E}\left[\boldsymbol{y}|\boldsymbol{x}\right] = \int \boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{x})p(\boldsymbol{\theta}|\mathcal{D})d\boldsymbol{\theta}.$$

In both cases, we will require the ability to sample from $p(\boldsymbol{\theta}|\mathcal{D})$ to perform Monte Carlo integration if the integral is not available in closed-form. Often,

sampling from posteriors, or any arbitrary distribution, is impossible to do directly. This is where the machine learning practioner might begin thinking about importance sampling.

For the remainder of this text, we will be discussing importance sampling and its critical variation: *sampling importance resampling (SIR)*. I personally enjoy working with particle filters, which we do not discuss here, but benefit from a strong understanding of these two techniques. However, before we begin, let us construct a simple example that we will use throughout this text and offer a few more words as to *why* we care about this topic.

1.1 Example

Assume the following probability density function (pdf):

$$p(x) = \begin{cases} x \exp(x^2)/(\frac{1}{2}(e-1)) & 0 \leq x \leq 1 \\ 0 & \text{otherwise} \end{cases}$$

It is easy to show that $p(x)$ integrates to 1. If we were interested in understanding $p(x)$, the following expectation would probably be useful:

$$\mathbb{E}[x] = \int xp(x)dx = \frac{1}{\frac{1}{2}(e-1)} \int_0^1 x^2 \exp(x^2)dx.$$

We can evaluate this expectation in closed-form, rather painfully:

$$\mathbb{E}[x] = \frac{1}{\frac{1}{2}(e-1)} \left(-\frac{\sqrt{\pi}\,\text{erfi}(1) - 2\text{e}}{4} + \frac{\sqrt{\pi}\,\text{erfi}(0)}{4} \right) \approx 0.730747,$$

where $\text{erfi}(\cdot)$ is the imaginary error function, also known as the Faddeeva function. Alternatively, we could understand $p(x)$ by observing examples from it. By just drawing N independent random samples $x_i \sim p(x), i = 1, \ldots, N$, we could probably understand $p(x)$ qualitatively. If we still wanted to know $\mathbb{E}[x]$, we could estimate this expectation as

$$\mathbb{E}[x] \approx \frac{1}{N} \sum_{i=1}^{N} x_i.$$

For many people, including myself, this approximation would have been easier than deriving the closed-form solution, and our *only* option when no closed-form solution is available. However, it is not immediately obvious how to sample from $p(x)$ to begin with. This is the problem we will treat in the coming sections using the $p(x)$ defined here. Figure 1 shows the pdf of $p(x)$ for reference.

2 Importance Sampling

Importance sampling assumes the ability to evaluate some $p(\boldsymbol{x})$, but not the ability to sample from it directly. As we showed in Section 1.1, it is possible to specify some $p(\boldsymbol{x})$ that is easy to evaluate but non-trivial to solve for some expectations. Monte Carlo integration is our only tool, in many cases, but it

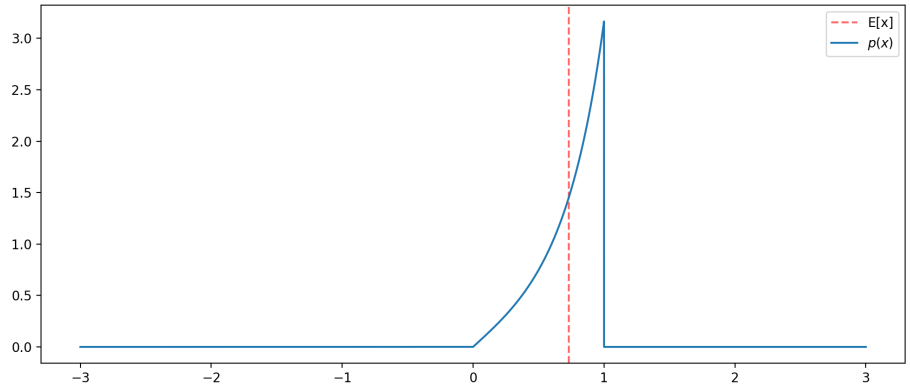

Figure 1: Probability density function $p(x)$ and the respective value for $\mathbb{E}[x]$, plotted as a vertical line.

requires the ability to sample. Importance sampling sidesteps this problem by sampling from a surrogate distribution $q(\boldsymbol{x})$ instead.

At its crux, importance sampling is just remarkably simple trick:

$$\mathbb{E}\left[\boldsymbol{f}(\boldsymbol{x})\right] = \int \boldsymbol{f}(\boldsymbol{x})p(\boldsymbol{x})d\boldsymbol{x} = \int \boldsymbol{f}(\boldsymbol{x})\left[\frac{q(\boldsymbol{x})}{q(\boldsymbol{x})}\right]p(\boldsymbol{x})d\boldsymbol{x} = \int \boldsymbol{f}(\boldsymbol{x})\left[\frac{p(\boldsymbol{x})}{q(\boldsymbol{x})}\right]q(\boldsymbol{x})d\boldsymbol{x},$$

where we have transformed an expectation with respect to $p(\boldsymbol{x})$ into an expectation with respect to $q(\boldsymbol{x})$. If $q(\boldsymbol{x})$ is easy to sample from, and has *support* greater than or equal to the support of $p(\boldsymbol{x})$, then we can perform a Monte Carlo integration by drawing N samples from $q(\boldsymbol{x})$:

$$\boldsymbol{x}_i \sim q(\boldsymbol{x}), i = 1, \ldots, N$$

and approximating as follows:

$$\mathbb{E}\left[\boldsymbol{f}(\boldsymbol{x})\right] \approx \frac{1}{N}\sum_{i=1}^{N}\left[\frac{p(\boldsymbol{x}_i)}{q(\boldsymbol{x}_i)}\right]\boldsymbol{f}(\boldsymbol{x}_i) = \frac{1}{N}\sum_{i=1}^{N}\boldsymbol{w}_i\boldsymbol{f}(\boldsymbol{x}_i).$$

The term $\boldsymbol{w}_i$ allows us to correct the importance distribution $q(\boldsymbol{x})$ such that we receive weighted samples from $p(\boldsymbol{x})$.

The catch is that the importance distribution and the target distribution cannot be *too* dissimilar. To illustrate this, let us consider two cases: $q(x) = \mathcal{N}(1.0, 0.5)$ and $q(x) = \text{Beta}(3.0, 1.0)$. These two cases are shown in Figure 2, where $\text{Beta}(3.0, 1.0)$ very closely approximates $p(x)$ and $\mathcal{N}(1.0, 0.5)$ has a lot of probability mass outside of support for $p(x)$.

Suppose we took 100,000 samples from both $q(x) = \mathcal{N}(1.0, 0.5)$ and $q(x) = \text{Beta}(3.0, 1.0)$. In Figure 3, we show the original, equally weighted, samples in orange for both distributions. Transforming $q(x)$ to some approximate density over $p(x)$ is a matter of applying $\boldsymbol{w}_i$ to every $x_i \sim q(x)$. After this transformation, we receive our desired samples for $p(x)$ shown in blue. As the number of samples $N \to \infty$, we should have some convergence guarentees as shown empirically in Figure 3.

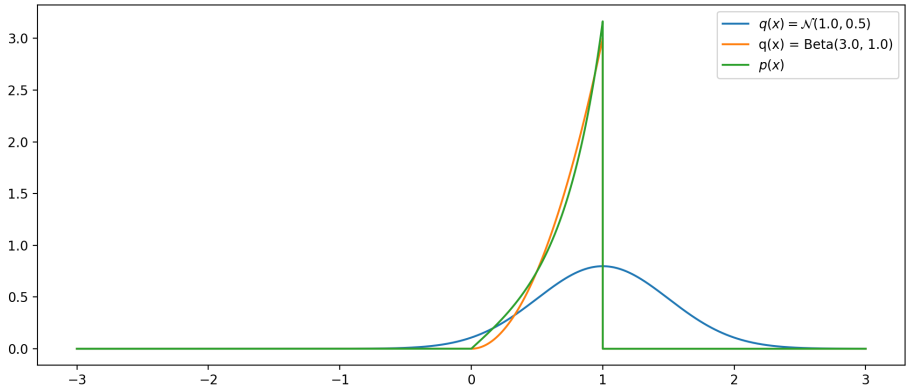

Figure 2: Probability density function $p(x)$ and two different importance distributions.

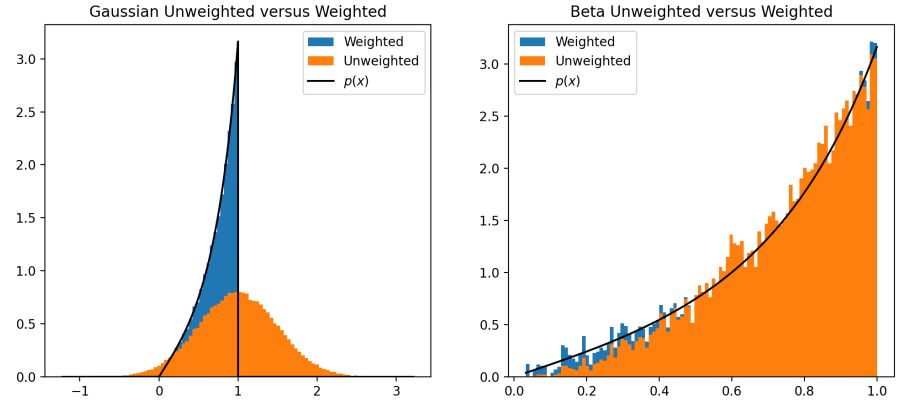

Figure 3: 100,000 samples before weights are applied versus after weights are applied for $q(x) = \mathcal{N}(1.0, 0.5)$ and $q(x) = \text{Beta}(3.0, 1.0)$.

We have promising results at high sample counts N. However, Figure 4 illustrates the need for a good choice of importance distribution. There, we are approximating $\mathbb{E}[x]$ for $p(x)$ given $q(x) = \mathcal{N}(1.0, 0.5)$ and $q(x) = \text{Beta}(3.0, 1.0)$, over a range of sample sizes N. It is clear to see that the $\text{Beta}(3.0, 1.0)$ is significantly more sample efficient, and converges to the true $\mathbb{E}[x]$ much faster. This is accomplished because $\boldsymbol{w}_i \approx 1 \ \forall \ i \in \{1, \ldots, N\}$. When $q(x)$ is a bad choice, as it is for $\mathcal{N}(1.0, 0.5)$, more samples are required to accurately perform Monte Carlo integration. We still see convergence, just at a slower rate.

Now, direct importance sampling requires normalized densities but recall that Bayesian posteriors are almost always known only up to some constant c. This presents a problem. However, using some arbitrary posterior distribution,

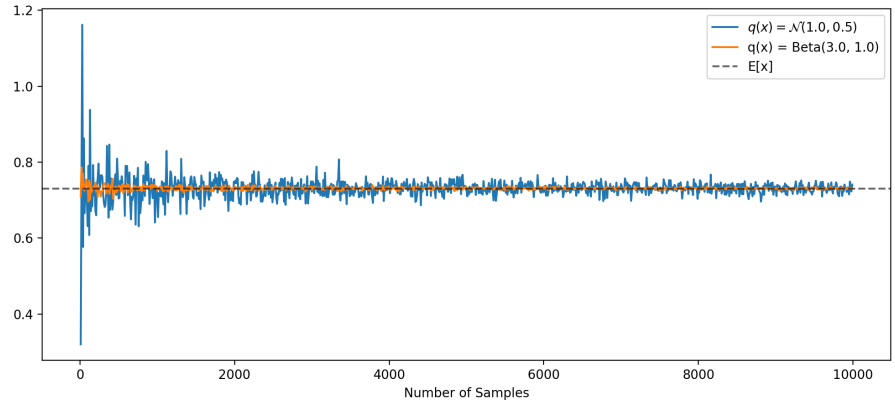

Figure 4: Convergence towards $\mathbb{E}[x]$ for $q(x) = \mathcal{N}(1.0, 0.5)$ and $q(x) = \text{Beta}(3.0, 1.0)$.

$p(\boldsymbol{x}|\mathcal{D})$, we can treat this problem as follows:

$$\begin{aligned}
\mathbb{E}\left[\boldsymbol{x}|\mathcal{D}\right] &= \int \boldsymbol{f}(\boldsymbol{x}) p(\boldsymbol{x}|\mathcal{D}) d\boldsymbol{x} \\
&= \frac{\int \boldsymbol{f}(\boldsymbol{x}) p(\mathcal{D}|\boldsymbol{x}) p(\boldsymbol{x}) d\boldsymbol{x}}{\int p(\mathcal{D}|\boldsymbol{x}) p(\boldsymbol{x}) d\boldsymbol{x}} \\
&= \frac{\int \left[\boldsymbol{f}(\boldsymbol{x}) \frac{p(\mathcal{D}|\boldsymbol{x}) p(\boldsymbol{x})}{q(\boldsymbol{x})}\right] q(\boldsymbol{x}) d\boldsymbol{x}}{\int \left[\frac{p(\mathcal{D}|\boldsymbol{x}) p(\boldsymbol{x})}{q(\boldsymbol{x})}\right] q(\boldsymbol{x}) d\boldsymbol{x}} \\
&\approx \frac{\frac{1}{N} \sum_{i=1}^{N} \boldsymbol{f}(\boldsymbol{x}_i) \frac{p(\mathcal{D}|\boldsymbol{x}_i) p(\boldsymbol{x}_i)}{q(\boldsymbol{x}_i)}}{\frac{1}{N} \sum_{j=1}^{N} \frac{p(\mathcal{D}|\boldsymbol{x}_j) p(\boldsymbol{x}_j)}{q(\boldsymbol{x}_j)}} \\
&= \sum_{i=1}^{N} \left[\frac{\frac{p(\mathcal{D}|\boldsymbol{x}_i) p(\boldsymbol{x}_i)}{q(\boldsymbol{x}_i)}}{\sum_{j=1}^{N} \frac{p(\mathcal{D}|\boldsymbol{x}_j) p(\boldsymbol{x}_j)}{q(\boldsymbol{x}_j)}}\right] \boldsymbol{f}(\boldsymbol{x}_i) = \sum_{i=1}^{N} \boldsymbol{w}_i \boldsymbol{f}(\boldsymbol{x}_i)
\end{aligned}$$

where we have used an importance sampling approximation to also approximate the normalization constant. This gives us importance sampling, as before, except we normalize the weights as an intermediate step.

3 Importance Sampling Resampling

The importance distribution, $q(\boldsymbol{x})$, can be hard to choose. Given a bad choice of $q(\boldsymbol{x})$, one mitigation is to use *Importance Sampling Resampling*.

Given $\boldsymbol{x}_i \sim q(\boldsymbol{x}), i = 1, \ldots, N$, we compute normalized weights

$$\boldsymbol{w}_i = \frac{p(\boldsymbol{x}_i)}{q(\boldsymbol{x}_i)} \quad \rightarrow \quad \hat{\boldsymbol{w}}_i = \frac{\boldsymbol{w}_i}{\sum_{j=1}^{N} \boldsymbol{w}_j},$$

and then resample $i = 1, \ldots, N$ according to

$$\boldsymbol{y}_i = \begin{cases} \boldsymbol{x}_1 & \text{w.p } \hat{\boldsymbol{w}}_1 \\ \vdots & \\ \boldsymbol{x}_N & \text{w.p } \hat{\boldsymbol{w}}_N \end{cases}$$

where we use $\boldsymbol{y}$ as follows:

$$\mathbb{E}\left[\boldsymbol{f}(\boldsymbol{x})\right] \approx \frac{1}{N} \sum_{i=1}^{N} \boldsymbol{f}(\boldsymbol{y}_i).$$

The idea here is to create a collection of samples that have a greater effective sample size. By bootstrapping the original dataset such that we get duplicates of $\boldsymbol{x}_i$'s that exist in high probability regions of $p(\boldsymbol{x})$ and fewer $\boldsymbol{x}_i$'s from lower probability regions, we accomplish this goal. Figure 5 shows the effect of resampling for $q(x) = \mathcal{N}(1.0, 0.5)$ and the importance of resampling is further realized in the time-varying setting with particle filters. Sampling importance resampling clearly has a positive effect, despite the choice of $q(\boldsymbol{x})$ being suboptimal.

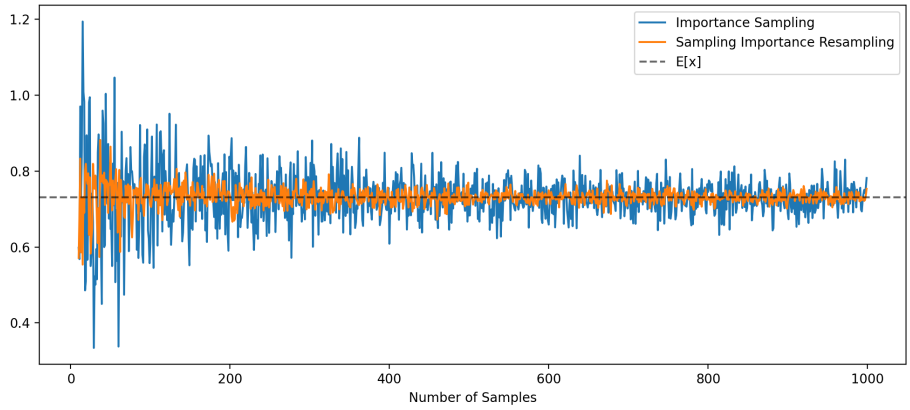

Figure 5: Convergence towards $\mathbb{E}[x]$ for importance sampling versus sampling importance resampling using $q(x) = \mathcal{N}(1.0, 0.5)$.

4 Acknowledgements

I heavily referenced [Bishop and Nasrabadi, 2006, Särkkä and Svensson, 2023] when generating this text. The reader is encouraged to read both texts for excellent expositions on sampling techniques and many other topics.

References

[Bishop and Nasrabadi, 2006] Bishop, C. M. and Nasrabadi, N. M. (2006). *Pattern recognition and machine learning*, volume 4. Springer.

[Särkkä and Svensson, 2023] Särkkä, S. and Svensson, L. (2023). *Bayesian filtering and smoothing*, volume 17. Cambridge university press.