# Sampling Methods for Approximate Inference

Amir Globerson

Thus far we considered variational methods for inference, resulting in algorithms like belief propagation and mean field. We now turn to a different approach, which is often empirically effective and has been studied extensively. The approach, known as sampling methods or Monte Carlo or particle based methods, uses the fact that expected values of random variables can be approximated via an average of a sample (i.e., the law of large numbers). The book by Koller and Friedmann provides a detailed introduction. You can also look at the paper [1]

## 1 Sampling Methods - The Basic Idea

The inference task is usually to calculate certain marginals of a distribution $p(x_1, \ldots, x_n)$. Say we are interested in $p_1(0)$ (namely the probability that $X_1 = 0$), and assume binary variables for simplicity. Then this is the same as calculating the expected value of the following "indicator" random variable $f(X_1, \ldots, X_n)$:

$$f(X_1, \ldots, X_n) = \begin{cases} 0 & X_1 = 0 \\ 1 & X_1 = 1 \end{cases} \tag{1}$$

We now see that:

$$\mathbb{E}\left[f(X_1, \ldots, X_n)\right] = \sum_{x_1,\ldots,x_n} p(x_1, \ldots, x_n) f(x_1, \ldots, x_n) = \sum_{x_1=1,x_2,\ldots,x_n} p(x_1, \ldots, x_n) = p_1(1) \tag{2}$$

Other marginals can be calculate similarly. The main message here is that if we knew how to calculate expected values of random variables $f(X_1, \ldots, X_n)$ we could also calculate marginals. But recall that the main difficulty is we don't know how to efficiently calculate sums like Equation 2 for our models of interest.

So here's another idea. For simplicity lets write $X$ instead of $X_1, \ldots, X_n$. We want to calculate the expected value $\mathbb{E}\left[f(X)\right]$. Assume we could generate samples of $X$ from the distribution $p(x)$. Denote $X^{(1)}, \ldots, X^{(T)}$ a set of $T$ such independent samples. Then the law of large numbers tells us that:

$$\frac{1}{T} \sum_{i=1}^{T} f(X^{(i)}) \to \mathbb{E}\left[f(X)\right] \tag{3}$$

as $T \to \infty$. Note that the RHS is a scalar whereas the LHS is a random variable. There are precise mathematical definitions for what this limit means (see weak and strong laws of large numbers). Intuitively, it means that as $T \to \infty$ the average on the LHS will be arbitrarily close to the expected value we are interested in.

This is the starting point of sampling methods: if we could generate IID samples from $p(x)$ we could calculate the expected value of any function of $X$ given enough samples. There are two obvious potential difficulties with this. The first is the question of how many samples $T$ we will need. The second is the fact it may not be easy to sample from $p(x)$. Indeed for general Markov networks, sampling is just as hard as inference. Our main focus will be the latter problem of sampling. Below we briefly comment on the issue of the required sample size $T$.

### 1.1 Sample sizes for sampling methods

How large must $T$ be to get a reasonable estimate of marginals. Let's again focus on a binary variables $X$, and assume we can generate IID samples of $X$ (note that even if we have variables other than $X$, since we are only interested in the marginal of $X$ it is enough to sample $X$ itself). Put differently if you have a coin with success probability $c$, how many samples do you need to estimate $c$ to within a given accuracy. In the language of statistics, this is the question of a confidence interval.

An analysis using Hoeffding's bound (e.g., similar to what you've seen in the context of PAC learning) yields the following guarantee. For any $\epsilon, \delta > 0$ if you use a sample $T \geq \frac{\ln \frac{2}{\delta}}{2\epsilon^2}$ the probability that:

$$\frac{1}{T} \sum_i X^{(i)} \notin [c - \epsilon, c + \epsilon] \tag{4}$$

is less than $\delta$. In other words you need order of $\epsilon^{-2}$ samples to guarantee an additive approximation of $\epsilon$ with high probability. A bound for multiplicative approximation can be obtained via Chernoff's bound.

## 2 Sampling Methods when $X$ cannot be sampled

As mentioned above, in most cases we will not be able to generated IID samples of $X$ (indeed if we could, the above sample size argument says we would have a polytime approximation method for marginals, which is known to be NP hard). It turns out that sampling can still be used to obtain effective approximations, and this has been a very active field of research, resulting in practical methods. We only scratch the surface below, providing two examples of approaches. In particular we do not discuss Markov

Chain Monte Carlo (MCMC) methods, which are a very important family of algorithms (Gibbs sampling discussed below is an instance of these).

## 2.1 Importance Sampling

In the spirit of mean field inference, we can approach the problem by relying on some other distribution $q(x)$ from which we do know how to sample. In what follows, we use the notation:

$$\mathbb{E}_q[X] = \sum_x q(x)x \tag{5}$$

Namely, $\mathbb{E}_q[X]$ is the expected value of the random variable $X$, when its distribution is $q(x)$. We now notice that we can write $\mathbb{E}_p[X]$ (which we want to calculate) by taking expectation with respect to $q$. The following is clearly true (assuming both $p, q$ are strictly positive):

$$\mathbb{E}_p[X] = \sum_x p(x)\frac{q(x)}{q(x)}x = \sum_x q(x)\frac{p(x)}{q(x)}x = \mathbb{E}_q\left[\frac{p(X)}{q(X)}X\right] \tag{6}$$

This immediately suggests the following approximation. Denote by $X^{(1)}, \ldots, X^{(T)}$ IID variables generated from $q(x)$. Consider the approximation:

$$\frac{1}{T}\sum_{i=1}^{T}\frac{p(X^{(i)})}{q(X^{(i)})}X^{(i)} = \frac{1}{T}\sum_{i=1}^{T} W_i X^{(i)} \tag{7}$$

where we define $W_i$ to be the corresponding ratio. Note that this assumes we can calculate $p(x)$ for a given $x$, which is not true for Markov networks, but we can address this via a similar trick. As $T \to \infty$. We do not study this further here, but it can be shown (as is intuitively clear), that it's most important to sample in regions where $f(x)p(x)$ is large, since these are those that are important for calculating the expectation.

In the case of Markov networks, we know that $p(x) = cg(x)$ but we do not know $c$ (namely the inverse of the partition function). We can use a simple trick to overcome this, by consider the following estimator:

$$\frac{\sum_{i=1}^{T} W_i X^{(i)}}{\sum_{i=1}^{T} W_i} \tag{8}$$

Since $W_i$ appears in both numerator and denominator, the $c$ factor cancels out, and we can just use $g(x)$. Furthermore, as $T \to \infty$ the denominator converges to $1$ and we are left with the estimator we wanted.

Of course whether or not the method will work closely depends on what $q(x)$ we choose (this is also known as the proposal distribution). This is discussed in length in the references above.

## 2.2 Gibbs Sampling

An alternative approach to hardness of sampling is to generate samples that will be "approximately" IID from $p(x)$. One very elegant way of achieving this is to construct a sequence of random variables $X^{(t)}$ such that for some large enough $T$, the variable $X^{(T)}$ will have a distribution close to that of $X$. The key thing here is that for small $t$ this may not be the case. Namely, it takes $X^{(t)}$ some time to "burn in" and start looking like the $X$ we want. If we manage to generate such $X^{(T)}$ a reasonable estimate of $\mathbb{E}[X]$ is to generate $T_2$ IID instances of $X^{(T)}$ and average them.

The most general formulation of this approach is a class of methods known as Markov Chain Monte Carlo. Here we will consider a simple instance of these, known as the Gibbs Sampling Algorithm.

Consider again the case where our variable $X$ is actually $n$ variables $X_1, \ldots, X_n$. We assume that sampling from $p(x)$ is hard, but that sampling from $p(x_1|x_2, \ldots, x_n)$ is easy. Similarly sampling $x_i$ given the other variables is assumed to be easy. This is indeed the case for many models, including MRFs. This suggests a very simple sampling algorithm: start by sampling $n$ variables $X_1^{(0)}, \ldots, X_n^{(0)}$ (e.g., sample each uniformly, or set all of them to zero). Now sample $X_1^{(t)}, \ldots, X_n^{(t)}$ recursively as follows: sample $X_j^{(t)}$ from the distribution $p(x_j|X_1^{(t)}, \ldots, X_{j-1}^{(t)}, X_{j+1}^{(t-1)}, \ldots, X_n^{(t-1)})$.

The indexing looks a bit complicated by the idea is simple: keep a single "copy" of your $n$ variables. At each time step, choose the next coordinate to sample (in cyclic order) and sample it given the current value of the other coordinates.

It turns out that under fairly general conditions, as $t \to \infty$ we have that the distribution of $X_1^{(t)}, \ldots, X_n^{(t)}$ approaches that of $X_1, \ldots, X_n$.

Let's analyze it for the case of two variables $X_1, X_2$. To keep notation clear, let's denote the distribution of the two variable by $p^*$ (i.e., the distribution we want to sample from). It turns out that the tricky thing to show is that the limit $P(X_1^{(t)} = x_1, X_2^{(t)} = x_2)$ exists an is unique (i.e., independent of the distribution of $X_1^{(0)}, X_2^{(0)}$). Let's assume that this is indeed the case and denote the limit distribution by $\pi(x_1, x_2)$. We will want to show that $\pi = p^*$. This is equivalent to requiring that if we start by setting $P[X_1^{(0)}, X_2^{(0)}]$ to $\pi$ we will have $X_1^{(1)}, X_2^{(1)} = \pi$. Below we'll see that $\pi = p^*$ satisfies this:

$$P[X_1^{(1)} = x_1, X_2^{(1)} = x_2] = P[X_1^{(1)} = x_1] p^*(x_2|x_1) \tag{9}$$

Now:

$$P[X_1^{(1)} = x_1] = \sum_{x_2} P[X_2^{(0)} = x_2] p^*(x_1|x_2) \tag{10}$$

We can see that setting $P[X_1^{(0)}, X_2^{(0)}]$ to $p^*$ yields $P[X_2^{(0)} = x_2] = p^*(x_2)$

and therefore $P[X_1^{(1)} = x_1] = p^*(x_1)$. Thus for Equation 9 we get:

$$P[X_1^{(1)} = x_1, X_2^{(1)} = x_2] = p^*(x_1)p^*(x_2|x_1) = p^*(x_1, x_2) \tag{11}$$

So starting with $p^*$ at time $0$ yields $p^*$ at time $1$. Therefore $p^*$ is the unique limit of the process.

## References

[1] C. Andrieu, N. de Freitas, A. Doucet, and M. I. Jordan. An introduction to mcmc for machine learning. *Machine Learning*, 50(1):5–43, 2003.