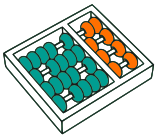

Monte Carlo inference [1]

Lesson No. 11

Gustavo de J. Merli - 262948

1 Introduction

This chapter discusses an alternative class of algorithms based on the idea of Monte Carlo approximation. The idea is very simple: generate some (unweighted) samples from the posterior, $x_s \sim p(x|D)$, and then use these to compute any quantity of interest, such as a posterior marginal, $p(x_1|D)$, or the posterior of the difference of two quantities, $p(x_1 - x_2|D)$, or the posterior predictive, $p(y|D)$, etc. All of these quantities can be approximated by $\mathbb{E}[f|D] \approx \frac{1}{S}\sum_{s=1}^{S} f(x^s)$ for some suitable function f.

2 Sampling from standard distributions

Discussion about sampling from 1 or 2 dimensional distributions of standard form.

2.1 Using the cdf

The simplest method for sampling from a univariate distribution is based on the inverse probability transform. Let F be a cdf of some distribution we want to sample from, and let F^{-1} be its inverse. Then we have the following result.

$$\text{if } U \sim U(0,1) \text{ is a uniform rv, then } F^{-1}(U) \sim F. \tag{1}$$

With this, it is possible to use $u \sim U(0,1)$ to represent the height up the y axis. Then, for an inverse function F^{-1}, the x value can be computed as $x = F^{-1}(u)$.

2.2 Sampling from a Gaussian (Box-Muller method)

Sample $z_1, z_2 \in (-1,1)$, uniformly, and then discard pairs that do not satisfy $z_1^2 + z_2^2 \leq 1$. The result will be points uniformly distributed inside the unit circle, so $p(z) = \frac{1}{\pi}\mathbb{I}(z \text{ inside circle})$. Now define

$$x_i = z_i \left(\frac{-2lnr^2}{r^2}\right)^{\frac{1}{2}} \tag{2}$$

for $i = 1:2$, where $r^2 = z_2^1 + z_2^2$. Using the multivariate change of variables formula, we have

$$p(x_1, x_2) = p(z_1, z_2)\left|\frac{\partial(z_1, z_2)}{\partial(x_1, x_2)}\right| = \left[\frac{1}{\sqrt{2\pi}} exp(-\frac{1}{2}x_1^2)\right]\left[\frac{1}{\sqrt{2\pi}} exp(-\frac{1}{2}x_2^2)\right] \tag{3}$$

Hence x_1 and x_2 are two independent samples from a univariate Gaussian. This is known as **Box-Muller** method.

To sample from a multivariate Gaussian, first compute the Cholesky decomposition of its covariance matrix, $\Sigma = LL^T$, where L is lower triangular. Next sample $x \sim N(0, I)$ using the Box-Muller method. Finally set $y = Lx + \mu$.

3 Rejection sampling

When the inverse cdf method cannot be used, one simple alternative is to use rejection sampling.

3.1 Basic idea

In rejection sampling, create a **proposal distribution** $q(x)$ which satisifes $Mq(x) \geq \tilde{p}(x)$, for some constant M, where $\tilde{p}(x)$ is an unnormalized version of $p(x)$ (i.e., $p(x) = \tilde{p}(x)/Z_p$ for some possibly unknown constant Z_p). The function $Mq(x)$ provides an upper envelope for $\tilde{p}$. Then sample $x \sim q(x)$, which corresponds to picking a random x location, and then sample $u \sim U(0,1)$, which corresponds to picking a random height (y location) under the envelope. If $u > \frac{\tilde{p}(x)}{Mq(x)}$, reject the sample, otherwise accept it.

Since it's generated with probability $q(x)$ and accepted with probability $\frac{\tilde{p}(x)}{Mq(x)}$, the probability of acceptance is

$$p(accept) = \int \frac{\tilde{p}(x)}{Mq(x)} q(x) dx = \frac{1}{M} \int \tilde{p}(x) dx \tag{4}$$

Hence the smallest M as possible while satisfying $Mq(x) \geq \tilde{p}(x)$ is the one to choose.

3.2 Example

Sampling from a Gamma distribution.

$$Ga(x|\alpha,\lambda) = \frac{1}{\Gamma(\alpha)} x^{\alpha-1} \lambda^{\alpha} exp(-\lambda x) \tag{5}$$

Using rejection sampling using a $Ga(k,\lambda-1)$ distribution as a proposal, where $k = \lfloor \alpha \rfloor$. The ratio has the form

$$\frac{p(x)}{q(x)} = \frac{\Gamma(k)\lambda^{\alpha}}{\Gamma(\alpha)(\lambda-1)^k} x^{\alpha-k} exp(-x) \tag{6}$$

The ration attains its maximum when $x = \alpha - k$

$$M = \frac{Ga(\alpha-k|\alpha,\lambda)}{Ga(\alpha-k|k,\lambda-1)} \tag{7}$$

3.3 Application to Bayesian statistics

Suppose wanting to draw (unweighted) samples from the posterior, $p(\theta|D) = p(D|\theta)p(\theta)/p(D)$. It is possible to use rejection sampling with $\tilde{p}(\theta) = p(D|\theta)p(\theta)$ as the target distribution and $q(\theta) = p(\theta)$ as the proposal, and $M = p(D|\hat{\theta})$, where $\hat{\theta} = argmax p(D|\theta)$ is the MLE. Accepting points with probability

$$\frac{\tilde{p}(\theta)}{Mq(\theta)} = \frac{p(D|\theta)}{p(D|\hat{\theta}} \tag{8}$$

This procedure is very inefficient if the prior is vague and the likelihood is informative.

3.4 Adaptive rejection sampling

The idea is to upper bound the log density with a piecewise linear function. The initial locations for the pieces are based on a fixed grid over the support of the distribution. Then evaluate the gradient of the log density at these locations, and make the lines be tangent at these points.

$$q(x) = M_i \lambda_i exp(-\lambda_i(x - x_{i-1})),\ x_{i-1} < x \leq x_i \tag{9}$$

where x_i are the grid points. . If the sample x is rejected, create a new grid point at x, and thereby refine the envelope. This is known as **adaptive rejection sampling** (ARS).

3.5 Rejection sampling in high dimensions

It is hard to achieve a close upper bound in high dimensions. To see this, consider sampling from $p(x) = \mathcal{N}(0, \sigma_p^2 I)$ using as a proposal $q(x) = \mathcal{N}(0, \sigma_q^2 I)$. Obviously $\sigma_q^2 \geq \sigma_p^2$ in order to be an upper bound. In D dimensions, $M = (\sigma_q/\sigma_p)^D$. The acceptance rate is $1/M$. For example, if σ_q exceeds σ_p by just 1%, then in 1000 dimensions the acceptance ratio will be about 1/20000. This a weakness of rejection sampling.

4 Importance sampling

This section describe the method **importance sampling** for approximating integrals of the form

$$I = \mathbb{E}[f] = \int f(x)p(x)dx \tag{10}$$

4.1 Basic idea

The idea is to draw samples x in regions which have high probability, $p(x)$, but also where $|f(x)|$ is large. The result can be super efficient, meaning it needs less samples than if were to sample from the exact distribution $p(x)$. The reason is that the samples are focussed on the important parts of space.

Importance sampling samples from any proposal, $q(x)$. It then uses these samples to estimate

$$\mathbb{E}[f] \approx \frac{1}{S}\sum_{s=1}^{S} w_s f(x^s) = \hat{I} \tag{11}$$

where $w_s = \frac{p(x^s)}{q(x^s)}$ are the **importance weights**. Unlike the rejection sampling, all the samples are used.

The lower bound is obtained when the optimal importance distribution is used

$$q^*(x) = \frac{|f(x)|p(x)}{\int |f(x')|p(x')dx'} \tag{12}$$

4.2 Handling unnormalized distributions

It is frequently the case that that is possible to evaluate the unnormalized target distribution, $\tilde{p}(x)$, but not its normalization constant, Z_p. For this, use an unnormalized proposal, $\tilde{q}(x)$, with possibly unknown normalization constant Z_q.

$$\mathbb{E}[f] \approx \frac{Z_q}{Z_p}\frac{1}{S}\sum_{s=1}^{S} \tilde{w}_s f(x^s) \tag{13}$$

where $\tilde{w}_s = \frac{\tilde{p}(x^s)}{\tilde{q}(x^s)}$ is the unnormalized importance weight.

$$\frac{Z_p}{Z_q} \approx \frac{1}{S}\sum_{s=1}^{S} \tilde{w}_s \tag{14}$$

Hence

$$\hat{I} = \sum_{s=1}^{S} w_s f(x^s) \tag{15}$$

where

$$w_s = \frac{\tilde{w}_s}{\sum_{s'} \tilde{w}_{s'}} \tag{16}$$

are the normalized importance weights.

4.3 Importance sampling for a DGM: likelihood weighting

This section describes a way to use importance sampling to generate samples from a distribution which can be represented as a directed graphical model.

If it has no evidence, it is possible to sample from the unconditional joint distribution of a DGM $p(x)$ as follows: first sample the root nodes, then sample their children, then sample their children, etc. This is known as **ancestral sampling**.

Sampling unobserved variables with ancestral sampling, conditional on their parents. But don't sample observed variables; instead just use their observed values. This is equivalent to using a proposal of the form

$$q(x) = \prod_{t \notin E} p(x_t | x_{pa(t)}) \prod_{t \in E} \delta_{x_t^*}(x_t) \tag{17}$$

where E is the set of observed nodes, and x_t^* is the observed value for node t. The importance weight is as follows:

$$w(x) = \frac{p(x)}{q(x)} = \prod_{t \notin E} \frac{p(x_t|x_{pa(t)})}{p(x_t|x_{pa(t)})} \prod_{t \in E} \frac{p(x_t|x_{pa(t)})}{1} = \prod_{t \in E} p(x_t|x_{pa(t)}) \tag{18}$$

This technique is known as **likelihood weighting**.

4.4 Sampling importance resampling (SIR)

It is possible to draw unweighted samples from $p(x)$ by first using importance sampling (with proposal q) to generate a distribution of the form

$$p(x) \approx \sum_s w_s \delta_{x^s}(x) \tag{19}$$

where w_s are the normalized importance weights. Then sample with replacement from equation 19, where the probability that x^s is picked is w_s. Let this procedure induce a distribution denoted by $\hat{p}$. This is known as **sampling importance resampling** (SIR). The result is an unweighted approximation of the form

$$p(x) \approx \frac{1}{S'} \sum_{s=1}^{S'} \delta_{x^s}(x) \tag{20}$$

with $S' << S$.

5 Particle filtering

Particle filtering (PF) is a Monte Carlo, or **simulation based**, algorithm for recursive Bayesian inference. It is very widely used in many areas, including tracking, time-series forecasting, online parameter learning, etc.

5.1 Sequential importance sampling

The basic idea is to appproximate the belief state (of the entire state trajectory) using a weighted set of particles:

$$p(z_{1:t}|y_{1:t}) \approx \sum_{s=1}^{S} \hat{w}_t^s \delta_{z_{1:t}^s}(z_{1:t}) \tag{21}$$

where $\hat{w}_t^s$ is the normalized weight of sample s at time t. From this representation, it is possible to compute the marginal distribution over the most recent state, $p(z_t|y_{1:t})$, by simply ignoring the previous parts of the trajectory, $z_{1:t-1}$.

Assume that $q(z_t|z_{1:t-1}, y_{1:t}) = q(z_t|z_{t-1}, y_t)$, then it only needs to keep the most recent part of the trajectory and observation sequence, rather than the whole history, in order to compute the new sample. In this case, the weight becomes

$$w_t^s \propto w_{t-1}^s \frac{p(y_t|z_t^s) p(z_t^s|z_{t-1}^s)}{q(z_t^s|z_{t-1}^s, y_t)} \tag{22}$$

Hence it is possible to approximate the posterior filtered density using

$$p(z_t|y_{1:t}) \approx \sum_{s=1}^{S} \hat{w}_t^s \delta_{z_t^s}(z_t) \tag{23}$$

5.2 The degeneracy problem

The basic sequential importance sampling algorithm fails after a few steps because most of the particles will have negligible weight. This is called the **degeneracy problem**, and occurs because we are sampling in a high-dimensional space (in fact, the space is growing in size over time), using a myopic proposal distribution.

The approximated **effective sample size** can be computed as follows

$$\hat{S}_{eff} = \frac{1}{\sum_{s=1}^{S}(w_t^s)^2} \tag{24}$$

There are two main solutions to the degeneracy problem: adding a resampling step, and using a good proposal distribution.

5.3 The resampling problem

The main improvement to the basic SIS algorithm is to monitor the effective sampling size, and whenever it drops below a threshold, to eliminate particles with low weight, and then to create replicates of the surviving particles. In particular, generate a new set $\{z_t^{s*}\}_{s=1}^{S}$ by sampling with replacement S times from the weighted distribution

$$p(z_t|y_{1:t}) \approx \sum_{s=1}^{S} \hat{w}_t^s \delta_{z_t^s}(z_t) \tag{25}$$

where the probability of choosing particle j for replication is w_t^j. The result is an iid *unweighted* sample from the discrete density equation 25, so the new weghts are set to $w_t^s = 1/S$.

5.4 The proposal distribution

The simplest and most widely used proposal distribution is to sample from the prior:

$$q(z_t|z_{t-1}^s, y_t) = p(z_t|z_{t-1}^s) \tag{26}$$

In this case, thethe weight update simplifies to

$$w_t^s \propto w_{t-1}^s p(y_t|z_t^s) \tag{27}$$

Sample values from the dynamic model, and then evaluate how good they are after seen the data. This is the approach used in the **condensation** algorithm (which stands for "conditional density propagation") used for visual tracking.

It is much better to actually look at the data y_t when generating a proposal. In fact, the optimal proposal distribution has the following form:

$$q(z_t|z_{t-1}^s, y_t) = p(z_t|z_{t-1}^s, y_t) = \frac{p(y_t|z_t)p(z_t|z_{t-1}^s)}{p(y_t|z_{t-1}^s)} \tag{28}$$

Using this proposal, the weight is given by

$$w_t^s \propto w_{t-1}^s p(y_t|z_{t-1}^s) = w_{t-1}^s \int p(y_t|z_t')p(z_t'|z_{t-1}^s)dz_t' \tag{29}$$

However, there are two cases when the optimal proposal distribution can be used. The first setting is when z_t is discrete, so the integral becomes a sum. Of course, if the entire state space is discrete, it is possible to use an HMM filter instead, but in some cases, some parts of the state are discrete, and some continuous. The second setting is when $p(z_t|z_{t-1}^s, y_t)$ is Gaussian. This occurs when the dynamics are nonlinear but the observations are linear.

5.5 Application: robot localization

Consider a mobile robot wandering around an office environment. Assume that it already has a map of the world, represented in the form of an **occupancy grid**, which just specifies whether each grid cell is empty space or occupied by an something solid like a wall. The goal is for the robot to estimate its location. This can be solved optimally using an HMM filter, since it was assumed that the state space is discrete. However, since the number of states, K, is often very large, the $O(K^2)$ time complexity per update is prohibitive. It is possible to use a particle filter as a sparse approximation to the belief state. This is known as **Monte Carlo localization**.

5.6 Application: visual object tracking

The next example is concerned with tracking an object (in this case, a remote-controlled helicopter) in a video sequence. The method uses a simple linear motion model for the centroid of the object, and a color histogram for the likelihood model, using **Bhattacharya distance** to compare histograms. The proposal distribution is obtained by sampling from the likelihood.

5.7 Application: time series forecasting

Assume that the model is a linear-Gaussian state-space model. There are many models which are either nonlinear and/or non-Gaussian. For example, **stochastic volatility** models, which are widely used in finance, assume that the variance of the system and/or observation noise changes over time. Particle filtering is widely used in such settings.

6 Rao-Blackwellised particle filtering (RBPF)

In some models, it is possible to partition the hidden variables into two kinds, q_t and z_t, such that it is possible to analytically integrate out z_t provided knowing the values of $q_{1:t}$. This means it only has sample $q_{1:t}$, and can represent $p(z_t|q_{1:t})$ parametrically. Thus each particle s represents a value for $q^s_{1:t}$ and a distribution of the form $p(z_t|y_{1:t}, q^s_{1:t})$. These hybrid particles are sometimes called **distributional particles** or **collapsed particles**.

The advantage of this approach is that dimensionality of the space is reduced in which it is sampled, which reduces the variance of the estimate. Hence this technique is known as **Rao-Blackwellised particle filtering** or **RBPF** for short.

6.1 RBPF for switching LG-SSMs

A canonical example for which RBPF can be applied is the switching linear dynamical system (SLDS) model.

If propose from the prior, $q(q_t = k|q^s_{t-1})$, the weight update becomes

$$w^s_t \propto w^s_{t-1} p(y_t|q_t = k, q^s_{1:t-1}, y_{1:t-1}) = w^s_{t-1} L^s_{t,k} \tag{30}$$

where

$$L^s_{tk} = \int p(y_t|q_t = k, z_t) p(z_t|q_t = k, y_{1:t-1} q^s_{1:t-1}) dz_t \tag{31}$$

The quantity L^s_{tk} is the predictive density for the new observation y_t conditioned on $q_t = k$ and the history $q^s_{1:t-1}$. In the case of SLDS models, this can be computed using the normalization constant of the Kalman filter.

6.2 Application: tracking a maneuvering target

The application of SLDS is to track moving objects that have piecewise linear dynamics. For example, suppose one wants to track an airplane or missile; q_t can specify if the object is flying normally or is taking evasive action. This is called **maneuvering target tracking**.

6.3 Fast SLAM

The problem of **simultaneous localization and mapping** or **SLAM** for mobile robotics. The main problem with the Kalman filter implementation is that it is cubic in the number of landmarks. However, conditional on knowing the robot's path, $q_{1:t}$, where $q_t \in \mathbb{R}^2$, the landmark locations $z \in R^{2L}$ are independent. (Assume the landmarks don't move, so drop the t subscript). That is, $p(z|q_{1:t}, y_{1:t}) = \prod_{l=1}^{L} p(z_l|q_{1:t}, y_{1:t})$. Consequently it is possible to use RBPF, where the robot's trajectory is sampled, $q_{1:t}$, and L independent 2d Kalman filters are run inside each particle. This takes $O(L)$ time per particle. Fortunately, the number of particles needed for good performance is quite small (this partly depends on the control / exploration policy), so the algorithm is essentially linear in the number of particles. This technique has the additional advantage that it is easy to use sampling to handle the data association ambiguity, and that it allows for other representations of the map, such as occupancy grids.

References

[1] K. P. Murphy, *Machine Learning : A Probabilistic Perspective*, 1st. Cambridge, Mass. [u.a.]: MIT Press, 2013.