Monte Carlo inference

Jiali Lin

Virginia Tech

January 15, 2017

Outline

Introduction

Sampling from inverse cdf

Rejection sampling

Importance sampling

Particle filtering

Introduction

- We typically can draw sample $\boldsymbol{x}^s \overset{i.i.d}{\sim} f(\boldsymbol{x})$.
- Now, we are unable to have this step.
- **Idea:** based on non-iterative algorithms of **Monte Carlo approximation**.
  - Generate some (unweighted) samples from the posterior, $\boldsymbol{x}^s \sim p(\boldsymbol{x}|\mathcal{D})$ (where s means "sample").
  - Use these to compute any quantity of interest, such as a posterior marginal, $p(x_1 - x_2|\mathcal{D})$.
- Useful when we have complicated posterior distribution.

Outline

Sampling from inverse cdf

- Let F be a cdf of target distribution and F^{-1} be its inverse.
- We can sample from any univariate distribution, by evaluating its inverse cdf.
- Let u represent the height up the y axis. $u \sim U(0, 1)$ using a pseudo random number generator.
- **Idea:** "slide along" the x axis until intersecting the F curve, "drop down" to return the corresponding x.
- **Problems**: impossible to derive cdf $F(x)$.

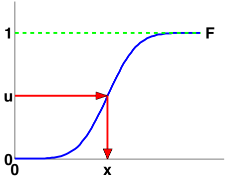

Figure: Sampling using an inverse CDF. Figure generated by `CdfSamplingDemo`.

Outline

Rejection sampling

- **Goal:** reject samples from $q(x)$ such that they are sampled from $p(x)$.
- $q(x)$ must "cover" or envelop the distribution $p(x)$ (i.e. $cq(x) > p(x)$ for all x).
- The samples are accepted if $\frac{p(x)}{cq(x)} > u$ where $u \sim Unif(0,1)$, and rejected otherwise.
- If the ratio is close to one, then $p(x)$ must have a large amount of probability mass around x and that sample should be more likely accepted.
- If the ratio is small, then it means that $p(x)$ has low probability mass around x and we should be less likely to accept the sample.

Illustration of rejection sampling

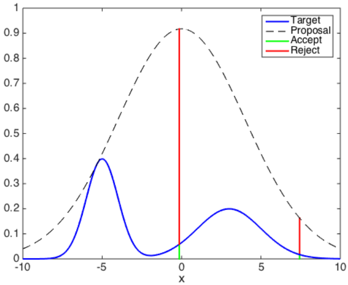

Figure: Rejection sampling from a mixture of two normal distributions using a proposal of normal distribution, where $c = \texttt{max(f(x)./q(x))}$. Figure generated by `R ejectionSampling`.

Comments:

- Hard to define proposal $q(x)$ before we know $p(x)$.
- $q(x)$ has to be closer to $p(x)$. Otherwise, it would be bad efficiency (low acceptance ratio).
- Fails in high dimensional space.

Outline

Importance sampling

- We can evaluate $p(\boldsymbol{x})$ easily for any given value of $\boldsymbol{x}$.
- Proposal distribution $q(\boldsymbol{x})$ is easy to draw samples.
- **Ideas:** uses these samples to estimate the integral

$$I = E[f] = \int f(\boldsymbol{x}) \frac{p(\boldsymbol{x})}{q(\boldsymbol{x})} q(\boldsymbol{x}) d\boldsymbol{x} \approx \frac{1}{S} \sum_{s=1}^{S} w_s f(\boldsymbol{x}^s) = \hat{I}$$

  where $w_s = \frac{p(\boldsymbol{x}^s)}{q(\boldsymbol{x}^s)}$ are the importance weights that can be computed.
- Weights correct the bias introduced by sampling from the wrong distribution.
- Unlike rejection sampling, *all the samples are used and focused on the important parts of space*.
- The latent sequence, $\boldsymbol{x}_{1:t}$ is determined by weight $\boldsymbol{w}_{1:t}$.

Importance sampling (cont'd)

- Usually $p(\boldsymbol{x}) = \tilde{p}(\boldsymbol{x})/Z_p$, where $p(\boldsymbol{x})$ can be evaluated easily, whereas Z_p is unknown. Then

$$E[f] = \frac{Z_q}{Z_p} \int f(\boldsymbol{x}) \frac{\tilde{p}(\boldsymbol{x})}{\tilde{q}(\boldsymbol{x})} q(\boldsymbol{x}) d\boldsymbol{x} \approx \frac{Z_q}{Z_p} \frac{1}{S} \sum_{s=1}^{S} \tilde{w}_s f(\boldsymbol{x}^s)$$

  where $\tilde{w}_s = \frac{\tilde{p}(\boldsymbol{x}^s)}{\tilde{q}(\boldsymbol{x}^s)}$ is the unormalized importance weight.
- We can use the same set of samples to evaluate the ratio

$$\frac{Z_p}{Z_q} = \frac{1}{Z_p} \int \tilde{p}(\boldsymbol{x}) d\boldsymbol{x} = \int \frac{\tilde{p}(\boldsymbol{x})}{\tilde{q}(\boldsymbol{x})} \approx \frac{1}{S} \sum_{s=1}^{S} \tilde{w}_s$$

- Hence

$$\hat{I} = \frac{\frac{1}{S} \sum_s \tilde{w}_s f(\boldsymbol{x}^s)}{\frac{1}{S} \sum_s \tilde{w}_s} = \sum_{s=1}^{S} w_s f(\boldsymbol{x}^s)$$

  where

$$w_s = \frac{\tilde{w}_s}{\sum_s' \tilde{w}_{s'}}$$

Outline

Introduction

Sampling from inverse cdf

Rejection sampling

Importance sampling

Particle filtering

Linear (Gaussian) State Space Models

- HMM structure with Gaussian conditional distributions:

$$
\begin{aligned}
p(z_1) &= N(z_1|0, Q_0) \\
z_{t+1} &= Az_t + w_t, \quad w_t \sim N(0, Q) \\
x_t &= Wz_t + v_t, \quad N(0, R)
\end{aligned}
$$

- Hidden states and observations are jointly Gaussian, so all marginals are Gaussian (parameterized by mean & covariance)/
- Posterior distribution of state at any time, given observations at any subset of other times, is Gaussian.

Particle filtering

- **Particle filtering (PF)**: a recursive Bayesian inference Monte Carlo algorithm.
- Estimate the posterior density of the **state space model**.
- Observations arrive sequentially in time.
- Given $p(\boldsymbol{z}_1), \quad p(\boldsymbol{z}_t|\boldsymbol{z}_{t-1}), \quad p(\boldsymbol{y}_t|\boldsymbol{z}_t)$.
- Goal: performing inference on-line $p(\boldsymbol{z}_{1:t}|\boldsymbol{y}_{1:t})$.

Particle filtering

- **Idea:** approximate the belief state (of the entire state trajectory) using a weighted set of particles

$$p(\boldsymbol{z}_{1:t}|\boldsymbol{y}_{1:t}) \approx \sum_{s=1}^{S} \hat{w}_t^s \delta_{\boldsymbol{z}_{1:t}^s}(\boldsymbol{z}_{1:t})$$

- Update the marginal distribution over $p(\boldsymbol{z}_t|\boldsymbol{y}_{1:t})$, by ignoring the previous parts of the trajectory, $\boldsymbol{z}_{1:t-1}$.
- If the proposal has the form $q(\boldsymbol{z}_{1:t}^s|\boldsymbol{y}_{1:t})$, then the importance weights are given by

$$w_t^s \propto \frac{p(\boldsymbol{z}_{1:t}^s|\boldsymbol{y}_{1:t})}{q(\boldsymbol{z}_{1:t}^s|\boldsymbol{y}_{1:t})}$$

  which can be normalized

$$\hat{w}^s = \frac{w_t^s}{\sum_{s'} w_t^{s'}}$$

Particle filtering (cont'd)

- We can rewrite the numerator recursively

$$\begin{aligned}
p(\boldsymbol{z}_{1:t}|\boldsymbol{y}_{1:t}) &= \frac{p(\boldsymbol{y}_t|\boldsymbol{z}_{1:t}, \boldsymbol{y}_{1:t-1})p(\boldsymbol{z}_{1:t}|\boldsymbol{y}_{1:t-1})}{p(\boldsymbol{y}_t|\boldsymbol{y}_{1:t-1})} \\
&= \frac{p(\boldsymbol{y}_t|\boldsymbol{z}_t)p(\boldsymbol{z}_t|\boldsymbol{z}_{1:t-1}, \boldsymbol{y}_{1:t-1})p(\boldsymbol{z}_{1:t-1}|\boldsymbol{y}_{1:t-1})}{p(\boldsymbol{y}_t|\boldsymbol{y}_{1:t-1})} \\
&\propto p(\boldsymbol{y}_t|\boldsymbol{z}_t)p(\boldsymbol{z}_t|\boldsymbol{z}_{t-1})p(\boldsymbol{z}_{1:t-1}|\boldsymbol{y}_{1:t-1})
\end{aligned}$$

  where we have made the usual Markov assumptions.
- We will restrict attention to proposal densities

$$q(\boldsymbol{z}_{1:t}|\boldsymbol{y}_{1:t}) = q(\boldsymbol{z}_t|\boldsymbol{z}_{1:t-1}, \boldsymbol{y}_{1:t})q(\boldsymbol{z}_{1:t-1}|\boldsymbol{y}_{1:t-1})$$

- We can "grow" the trajectory by adding the new state $\boldsymbol{z}_t$ to the end

$$\begin{aligned}
w_t^s &\propto \frac{p(\boldsymbol{y}_t|\boldsymbol{z}_t^s)p(\boldsymbol{z}_t^s|\boldsymbol{z}_{t-1}^s)p(\boldsymbol{z}_{1:t-1}^s|\boldsymbol{y}_{1:t-1})}{q(\boldsymbol{z}_t^s|\boldsymbol{z}_{1:t-1}^s, \boldsymbol{y}_{1:t})q(\boldsymbol{z}_{1:t-1}^s|\boldsymbol{y}_{1:t-1})} \\
&= w_{t-1}^s \frac{p(\boldsymbol{y}_t|\boldsymbol{z}_t^s)p(\boldsymbol{z}_t^s|\boldsymbol{z}_{t-1}^s)}{q(\boldsymbol{z}_t^s|\boldsymbol{z}_{1:t-1}^s, \boldsymbol{y}_{1:t})}
\end{aligned}$$

Particle filtering (cont'd)

- We further assume that $q(\boldsymbol{z}_t|\boldsymbol{z}_{1:t-1}, \boldsymbol{y}_{1:t}) = q(\boldsymbol{z}_t|\boldsymbol{z}_{t-1}, \boldsymbol{y}_t)$

$$w_t^s \propto w_{t-1}^s \frac{p(\boldsymbol{y}_t|\boldsymbol{z}_t^s)p(\boldsymbol{z}_t^s|\boldsymbol{z}_{t-1}^s)}{q(\boldsymbol{z}_t^s|\boldsymbol{z}_{t-1}^s, \boldsymbol{y}_t)}$$

- Hence we can approximate the posterior filtered density using

$$p(\boldsymbol{z}_t|\boldsymbol{y}_{1:t}) \approx \sum_{s=1}^{S} \hat{w}_t{}^s \delta_{z_t^s}(\boldsymbol{z}_t)$$

Potential problems of Particle filtering

- **Comment # 1**: Choice of proposal distribution.
  - *Solutions*: Let $q(\boldsymbol{z}_t|\boldsymbol{z}_{1:t-1}, \boldsymbol{y}_t) = p(\boldsymbol{z}_t|\boldsymbol{z}_{1:t-1})$ (known as **Condensational Filter**).

- **Comment # 2**: **Particle degeneracy:** Pfs fails after a few steps because most of the particles will have negligible weight.
  - *Solutions*: Introducing re-sampling: break those big particle into smaller ones, from the "re-sampling" step.

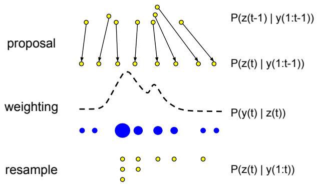

Figure: Illustration of particle filtering.

Particle filtering algorithm

The basic algorithm is now very simple: for each old samples, propose an extension using $\boldsymbol{z}_t^s \sim q(\boldsymbol{z}_t | \boldsymbol{z}_{t-1}^s, \boldsymbol{y}_t)$, and give this new particle weight w_t^s.

---

for $s = 1 : S$ **do**

1. Draw $\boldsymbol{z}_t^s \sim q(\boldsymbol{z}_t | \boldsymbol{z}_{t-1}^s, \boldsymbol{y}_t)$.
2. Compute weight $w_t^s \propto w_{t-1}^s \frac{p(\boldsymbol{y}_t | \boldsymbol{z}_t^s) p(\boldsymbol{z}_t^s | \boldsymbol{z}_{t-1}^s)}{q(\boldsymbol{z}_t^s | \boldsymbol{z}_{t-1}^s, \boldsymbol{y}_t)}$.

given x^k, λ^{k-1}, and parameter $\beta \in (0, 1)$.

Normalize weights: $\hat{w}^s = \frac{w_t^s}{\sum_{s'} w_t^{s'}}$.
Compute $\hat{S}_{\text{eff}} = \frac{1}{\sum_{s=1}^S} (w_t^s)^2$

If $\hat{S}_{\text{eff}} < S_{\text{min}}$ **then**

1. Resample S indcies $\pi \sim w_t$.
2. $\boldsymbol{z}_t = \boldsymbol{z}_t^\pi$.
3. $w_t^s = 1/S$.

return $p(z_{1:t} | \boldsymbol{y}_{1:t})$.
