1 Sampling Importance Resampling (SIR)

Let $f$ be the density of a random variable $X$, $h$ is a function. We are to calculate $\mu = \mathbb{E}\{h(X)\}$. In most cases, when $h$ or $f$ is complicated, we can not calculate the expectations directly. Or put it simple, we cannot directly calculate the integral,

$$\mu = \int h(x)f(x)dx.$$

To address this problem, we use a sampling method. We firstly draw $X_1, \ldots, X_n$ i.i.d from the pdf $f$. Then, from the law of large number, we can approximate the expectation via the sample mean, that is,

$$\mu \approx \frac{1}{n}\sum_{i=1}^{n} h(X_i)$$

Although the mentioned approximation is theoretically sound for approximation every integrals, a practical difficulty is that we are incapable to draw samples i.i.d from arbitrary probability functions. For example, Gaussian or uniform distribution can be readily found in a MATLAB or R package, but for the other obsolete pdf’s, we have no idea how to generate samples, not mention to guarantee the independence. This point, actually motivates the technique of Importance Sampling, where we incorporate known pdf with readily sampling methods to generate samples via arbitrary pdf. To start with, we give the algorithm of SIR in Algorithm 1.

Here we try to illustrate the correctness of Algorithm 1 in subsequent context, except for some technical details.

For an arbitrary probability space $\mathcal{A} \subseteq \Omega$, we first consider this following conditional probability,

$$P(X \in \mathcal{A}|Y_1, \ldots, Y_m) = \frac{\sum_{i=1}^{m} \mathbb{1}_{\{Y_i \in \mathcal{A}\}} w^*(Y_i)}{\sum_{i=1}^{m} w^*(Y_i)} \quad (1)$$

**Algorithm 1** Sampling Importance Resampling

1. Draw sample candidates $Y_1, \ldots, Y_m$ iid from $g$
2. Calculate the standardized importance weights, $w(Y_i) = \frac{f(Y_i)/g(Y_i)}{\sum_j f(Y_j)/g(Y_j)}$ for $i = 1, \ldots, m$
3. Resample $X_1, \ldots, X_n$ from $Y_1, \ldots, Y_m$ with replacement of probability $w(Y_1), \ldots, w(Y_m)$ respectively.
where $1_{\{x \in A\}}$ is an indicator function, which takes on value 1 when $x \in A$ and 0 otherwise. For the denominator of (1), when $m \to \infty$, we have,
\[
\lim_{m \to \infty} \frac{1}{m} \sum_{i=1}^{m} 1_{\{Y_i \in A\}} w^*(Y_i) = \mathbb{E}(1_{\{Y \in A\}} w^*(Y)) = \int_A w^*(y) g(y) dy
\]

For the numerator of (1), when $m \to \infty$, we have,
\[
\lim_{m \to \infty} \frac{1}{m} \sum_{i=1}^{m} w^*(Y_i) = \mathbb{E}(w^*(Y)) = \int f(y) dy = 1
\]

Thus combining these two results, we have
\[
\lim_{m \to \infty} P(x \in A|Y_1, \ldots, Y_m) = \int_A w^*(y) g(y) dy = \int_A f(y) dy.
\]

Also, note the identity $P(X \in A) = \mathbb{E} (P(X \in A|Y_1, \ldots, Y_m))$ holds. This identity can be proved via definition, because the right side yields,
\[
\mathbb{E} (P(X \in A|Y_1, \ldots, Y_m)) = \int \int p(x|y_1, \ldots, y_m) p(y_1, \ldots, y_m) dy \int_{y \in A} \int_{x \in A} p(x) dx = P(X \in A)
\]

Since we already demonstrated that $P(X \in A|Y_1, \ldots, Y_m)$ is a constant, say, $\int_A f(y) dy$, the expectation of $P(X \in A|Y_1, \ldots, Y_m)$ is the same. So when $m \to \infty$, we have $P(x \in A) = \int_A f(x) dx$. This is equivalent to say that, the samples $X_1, \ldots, X_n$ we generated via the SIR algorithm conforms a distribution with pdf $f(x)$.

Moreover, the referred Importance Sampling is worth noting. Actually, we are considering the following presentation of our object function $\mu(x)$,
\[
\mu(x) = \int h(x) f(x) dx = \int h(x) \frac{f(x)}{g(x)} g(x) dx = \int h(x) \frac{f(x)}{g(x)} g(x) dx \int \frac{f(x)}{g(x)} g(x) dx
\]

Thus in our application, when the probability we concerned is not normalized to 1, or in other words, only known to be in the form of $f'(x) \propto f(x)$, (2) still applies. We only need to calculate both the denominator and numerator of (2) via the sampling methods described.

8 - 2
2 The Particle Filter

2.1 Basic Algorithm

The particle filter is an alternative nonparametric implementation of the Bayes filter. The key idea of the particle filter is to represent the posterior \( \text{bel} (x_t) \) by a set of random state samples drawn from this posterior.

In particle filter, the samples of a posterior distribution are called \textit{particles} and are denoted

\[ X_t := x_t^{(1)}, x_t^{(2)}, \ldots, x_t^{(M)} \]

Each particle \( x_t^{(m)} \) (with \( 1 \leq m \leq M \)) is a hypothesis of the state at time \( t \). Here \( M \) denotes the number of particles in the particle set \( X_t \).

The idea behind particle filters is to approximate the belief \( \text{bel} (x_t) \) by the set of particles \( X_t \). Ideally, the likelihood for a state hypothesis \( x_t \) to be included in the particle set \( X_t \) shall be proportional to its Bayes filter posterior \( \text{bel} (x_t) \):

\[ x_t^{(m)} \sim \text{bel} (x_t) = p(x_t \mid z_{1:t}, u_{1:t}) \]

As a result, the denser a subregion of the state space is populated by samples, the more likely it is that the true state falls into this region. It is absolutely true when \( M \rightarrow \infty \). For finite \( M \), particles are drawn from a slightly different distribution. The difference is negligible when \( M \) is not too small.

All Bayes filter algorithms construct \( \text{bel} (x_t) \) recursively from \( \text{bel} (x_{t-1}) \) one time earlier. For particle filters, it means to construct the particle set \( X_t \) recursively from the set \( X_{t-1} \). The algorithm is illustrated as below:

**Algorithm 2 The Particle Filter**

1. Input: \( x_{t-1}, u_t, z_t \)
2. initialize \( \overline{X}_t = X_t = \emptyset \)
3. for \( m = 1 \) to \( M \) do
   4. sample \( x_t^{(m)} \sim p(x_t \mid u_t, x_{t-1}^{(m)}) \)
   5. \( w_t^{(m)} = p(z_t \mid x_t^{(m)}) \)
   6. \( \overline{X}_t = \overline{X}_t + <x_t^{(m)}, w_t^{(m)}> \)
4. endfor
5. for \( m = 1 \) to \( M \) do
   6. draw \( m \) with probability proportional to \( w_t^{(m)} \)
   7. add \( x_t^{(m)} \) to \( X_t \)
6. endfor
7. return \( X_t \)

2.2 Mathematical Derivation of the PF

To derive particle filters mathematically, it shall prove useful to think of particles as samples of state sequences

\[ x_{0:t}^{(m)} = x_0^{(m)}, x_1^{(m)}, \ldots, x_t^{(m)} \]
We modify this algorithm by simply appending to the particle $x_i^{(m)}$ the sequence of state samples from which it was generated $x_0^{(m)}$. This particle filter then calculates $bel(x_0,t) = p(x_0,t \mid u_1,t, z_1,t)$ instead of the belief $bel(x_t) = p(x_t \mid u_1,t, z_1,t)$. This definition serves only as the means to derive the particle filter algorithm.

The posterior $bel(x_0,t)$ is obtained similarly to the derivation of $bel(x_t)$:

$$p(x_0,t \mid z_{1:t}, u_{1:t}) = \eta p(z_t \mid x_0,t, z_{1:t-1}, u_{1:t}) p(x_0,t \mid z_{1:t-1}, u_{1:t})$$

$$= \eta p(z_t \mid x_t) p(x_t \mid x_0,t, z_{1:t-1}, u_{1:t}) p(x_0,t \mid z_{1:t-1}, u_{1:t})$$

$$= \eta p(z_t \mid x_t) p(x_t \mid x_{t-1}, u_t) p(x_0,t \mid z_{1:t-1}, u_{1:t-1})$$

The derivation is now carried out by induction. The initial condition is trivial to verify, assuming that our first particle set is obtained by sampling the prior $p(x_0)$. Let us assume that the particle set at time $t - 1$ is distributed according to $bel(x_0,t-1)$. For the $m$-th particle $x_0^{(m)}$ in this set, the sample $x_i^{(m)}$ is generated from the proposal distribution:

$$p(x_t \mid x_{t-1}, u_t) bel(x_0,t-1) = p(x_t \mid x_{t-1}, u_t) p(x_0,t-1 \mid z_{1:t-1}, u_{1:t-1})$$

It is obvious that $x_i^{(m)} \sim p(x_t \mid x_{i-1}^{(m)}, u_t)$

With the definition of $w^{(m)}$ in SIR, we have:

$$w_i^{(m)} = \frac{\text{target distribution}}{\text{proposal distribution}}$$

$$= \frac{\eta p(z_t \mid x_t) p(x_t \mid x_{t-1}, u_t) p(x_0,t-1 \mid z_{1:t-1}, u_{1:t-1})}{p(x_t \mid x_{t-1}, u_t) p(x_0,t-1 \mid z_{1:t-1}, u_{1:t-1})}$$

$$= \eta p(z_t \mid x_t)$$

The constant $\eta$ plays no role since the resampling takes place with probabilities proportional to the importance weights $w_i^{(m)}$. The resulting particles are exactly distributed according to the product of the proposal and the importance weights $w_i^{(m)}$:

$$\eta w_i^{(m)} p(x_t \mid x_{t-1}, u_t) p(x_0,t-1 \mid z_{1:t-1}, u_{1:t-1}) = bel(x_0,t)$$

We’ve proved that $x_0^{(m)}$ is distributed according to $bel(x_0,t)$. Then the state sample $x_i^{(m)}$ is distributed according to $bel(x_t)$.

### 3 Sequential Importance Sampling

In this section, we continue to investigate nonparametric filtering algorithms with target and proposal functions same as previous sections. The sequential importance sampling (SIS) is a popular methods for sequential data, or performing online-processing. Assume the sequential data we are dealing with is $x = (x_1, \ldots, x_p)$. At time $t$, the data at hand is denoted as $x_i = (x_1, \ldots, x_i)$, where $i \leq p$. We assume the target distribution $f(x)$ can be factorized as follows,

$$f(x) = f(x_1)f(x_2 \mid x_1:1)f(x_3 \mid x_2:2) \ldots f(x_p \mid x_1:p-1).$$
Remember that we can not directly draw samples from $f(x)$. To overcome this problem, we consider the proposal function can be factorized in a corresponding convention, namely,

$$g(x) = g(x_1)g(x_2|x_{1:1})g(x_3|x_{1:2}) \ldots g(x_p|x_{1:p-1}).$$

Note a trivial choice of $g$ is Gaussian, since products of Gaussians are still Gaussian, which have readily sampling procedures. With this factorization of the proposal and target function, we can summarize the algorithm of SIS in Algorithm 3

**Algorithm 3 Sequential Importance Sampling**

1. Draw sample candidates $X_1$ from $g(x_1)$
2. Set $\hat{w}_1^*(X_1) = \frac{f(X_1)}{g(X_1)}$ and $i = 2.$
3. Given $X_{1:i-1}$, sample $X_i \sim g(x_i)$.
4. Set $X_{1:i} = (X_{1:i-1}, X_i)$ and define $\hat{w}_i^*(X_i) = \hat{w}_1^*(X_{i-1}) \frac{f(X_{1:i})}{f(X_{1:i-1})g(X_i|X_{1:i-1})}$