Sequential Monte Carlo Samplers

Antoine Carnec, Alexandra Málaga, Maxim Fedotov

March 28, 2022

Enumeration of Latin Squares

Let $\mathbf{M}_d$ be a set of matrices M such that each row of M is a permutation of numbers: $\{0, \ldots, d-1\}$. A latin square, S, is a square matrix such that every row and column is a permutation of $\{0, \ldots, d-1\}$ and at every position, the corresponding row and column does not contain the number at that index.
For example the following is a Latin Square

$$\begin{bmatrix} 0 & 1 & 2 \\ 2 & 0 & 1 \\ 1 & 2 & 0 \end{bmatrix}$$

The known number of Latin Squares for increasing d is

$$1, 2, 12, 576, 161280, 812851200, 61479419904000, \ldots$$

Largest known $d = 11$, $\geq 10^{43}$.

The Idea

- We want to sample from a distribution ; eg. posterior in bayesian inference.
- Idea: Sample from some easy distribution and use importance sampling to transform initial sample to the desired distribution.
- Sequence $\pi_0, \ldots, \pi_T$. π_0 is easy to sample from and π_T is a distribution of interest. Go from π_t to π_{t-1} using importance sampling (with extra steps).

Importance Sampling

It is a method to sample from a distribution $\pi(x) = \tilde{\pi}(x)/Z$ where Z is a normalizing constant. It can be done in two steps:

1. Given a proposal distribution $q(x)$, simulate N samples.

$$X_n \sim q(x), \qquad n = 1, ..., N$$

2. Assign an importance weight to each sample.

$$w_n = \frac{\pi(x_n)}{q(x_n)}, \qquad n = 1, ..., N$$

Where the estimator of the normalizing constant is:

$$\tilde{Z} = \frac{1}{N} \sum_{n=1}^{N} w_n$$

And the normalized importance weights are: $\tilde{w}_n = \frac{w_n}{\tilde{Z}}$.

Notation

- At time t we have a set of particles, which we notate as $\{X_n^t\}_{n=1}^N$.
- Designate the weight of particle n at time t as w_n^t.
- Normalised weights W_n^t.
- Sequence of distributions $\pi_1, \dots, \pi_T$, such that $\pi_t = \frac{1}{L_t}\tilde{\pi}_t$.

The Algorithm

Algorithm 1: Basic SMC Sampler

Input: Distributions $\pi_0, \ldots, \pi_T$ where $\pi_t(x) = \frac{1}{L_t} m(x) \tilde{\pi}_t(x)$;

Markov Kernel M_t

(n indicates that action is done for $n = 1, \ldots, N$)

$X_n^0 \sim \pi_0$

$w_n^0 \leftarrow \tilde{\pi}_0(X_n^0)$

$W_n^0 \leftarrow \frac{w_n^0}{\sum_{j=1}^n w_j^0}$

for $t = 1, \ldots, T$ **do**

$\hat{X}_n^t \leftarrow$ resample X_n^{t-1} with Multinomial(W_n^{t-1})

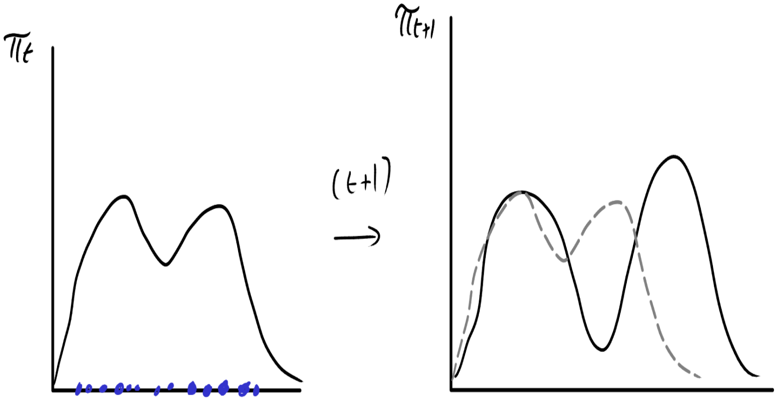
$X_n^t \sim M_t(\hat{X}_n^t)$

$w_n^t \leftarrow \frac{\tilde{\pi}_t(X_n^t)}{\tilde{\pi}_{t-1}(X_n^t)}$

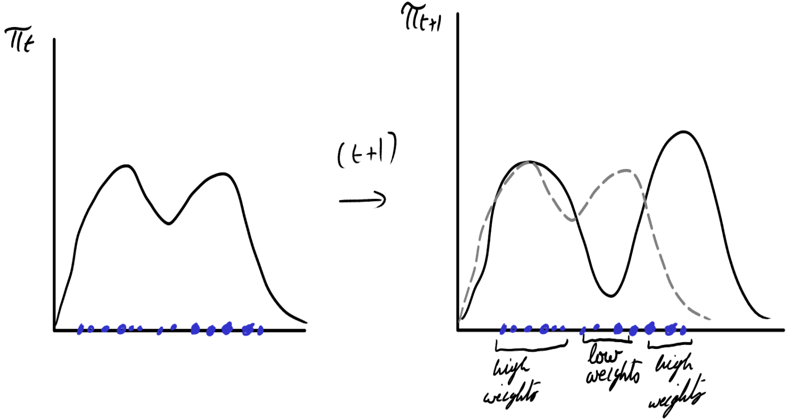
$W_n^t \leftarrow \frac{w_n^t}{\sum_{j=1}^n w_j^t}$

end

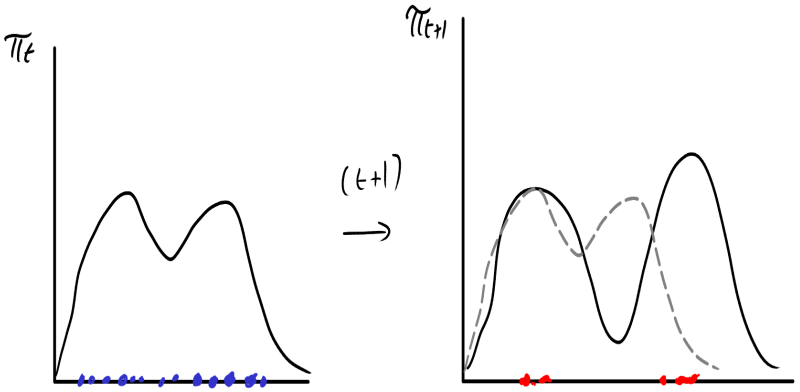
A picture



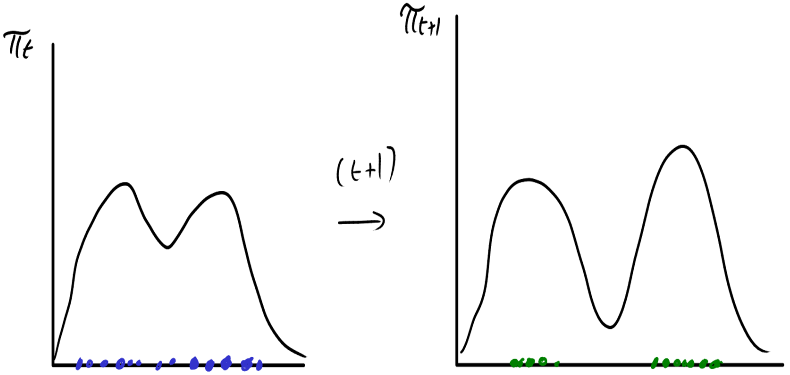
Getting weights



Resampling



Applying MCMC



ESS

It is a measure of the necessary number of samples from a dependent sampling algorithm to collect the same information as n samples from an independent sampling algorithm. For weighted samples, the effective sample size is:

$$ESS = \frac{(\sum_{i=1}^{N} w_i)^2}{\sum_{i=1}^{N} w_i^2}$$

Properties:

- $ESS \in [1, N]$.
- $N/ESS - 1$ converges to the chi-square (pseudo-) distance of your empirical distribution to your theoretical distribution (π_t). So higher ESS implies that your empirical approximation is good.
- Intuitively, you can think about ESS as the *diversity* of your sample.

Making things adaptive: How to choose $\{\pi_t\}$ in practice?

- Define

$$\pi_t(x) = \frac{1}{L_t} m(x) \exp\{-\lambda_t V(x)\},$$

- Every λ_t defines a probability distribution.
- For example in the context of bayesian inference, $m(x)$ is the prior, $\exp\{-V(x)\}$ is the likelihood.
- Idea: Choose λ_t such that new distribution is not too far from the previous one.

How to find λ_t

Solve for λ_t in

$$ESS = ESS_{\min}$$

where ESS in this case can be expressed as

$$\begin{aligned} ESS &= \frac{\left(\sum_{n=1}^{N} w_n\right)^2}{\sum_{n=1}^{N} w_n^2} \\ &= \frac{\left(\sum_{n=1}^{N} \exp\{-(\lambda_t - \lambda_{t-1})V(x)\}\right)^2}{\sum_{n=1}^{N} \exp\{-2(\lambda_t - \lambda_{t-1})V(x)\}}. \end{aligned}$$

Adaptive Tempering

Algorithm 2: Adaptive Tempering SMC Sampler

Input: π_0 ; $\overline{\lambda}$; $V(x)$; $ESS_{\min}$; Markov Kernel M_t

(n indicates that action is done for $n = 1, \ldots, N$)

$\lambda_{-1} \leftarrow 0$

$t \leftarrow -1$

while $\lambda_t < \overline{\lambda}$ **do**

  $t \leftarrow t + 1$

  if $t = 0$ **then**

    $X_n^t \sim \pi_0$

  else

    $\hat{X}_n^t \leftarrow$ resample X_n^{t-1} from Multinomial(W_n^{t-1})

    $X_n^t \sim M_t(\hat{X}_n^t)$

  end

  $\lambda_t \leftarrow$ Solve for λ_t in $ESS = ESS_{\min}$

  $w_n^t \leftarrow \exp\{-(\lambda_t - \lambda_{t-1})V(X_n^t)\}$

  $W_n^t \leftarrow \frac{w_n^t}{\sum_{j=1}^n w_j^t}$

end

How to count with sampling

- Again take

$$\pi_t(x) = \frac{1}{L_t} m(x) \exp\{-\lambda_t V(x)\}$$

- Fix d.
- $m(x)$ is a uniform distribution over matrices which rows are permutations of numbers $0, 1, \ldots, d-1$. $m(x) = \frac{1}{(d!)^d}$.
- $V(x) = \sum_{j=1}^{d} \left\{ \sum_{l=1}^{d} \left(\sum_{i=1}^{d} \mathbb{1}(x[i,j] = l) \right)^2 - d \right\}$
- We also need to specify the depth of resampling and a number of particles that we are sampling at each iteration.

Normalising Constant

- Let $\lambda_t = \infty$
- Denote set of all Latin squares of size $d \times d$ by $\mathbf{S}_d$. Consider

$$\sum_{x \in \mathbf{S_d}} \pi(x) = \frac{1}{L_t^{\mathbf{S}_d}} \sum_{x \in \mathbf{S_d}} m(x) e^{-\lambda_t V(x)}$$
$$\implies L_t \cdot (d!)^d = |\mathbf{S_d}|$$

  where $L_t^{\mathbf{S}_d}$ is a normalising constant and $|\mathbf{S_d}|$ is the number of Latin squares of size $d \times d$.

- You can show that $|\mathbf{S_d}|$ is within an ϵ of the number of Latin Squares once $\lambda_t > log(\frac{(d!)^d}{\epsilon})$

Normalising Constant

The standard SMC estimator for a normalizing constant, L_t, of the elements of sequence of target distributions $\{\pi_0, \pi_1, \ldots\}$ based on re-samples of size N is

$$L_t^N = \prod_{s=0}^{t} \frac{1}{N} \sum_{n=1}^{N} w_n^s,$$

where w_n^s are weights assigned to particles on s-th iteration of resampling.