An Overview of High-Dimensional Sampling Methods

Genghis Luo

NYU Shanghai

May 19, 2025

Outline

Outline

Problem Set-up I

Suppose we wish to evaluate:

$$u = \int_{\mathcal{X} \subseteq \mathbb{R}^d} f(x)\pi(x)dx = \mathbb{E}_\pi[f] \tag{1}$$

with an oracle accessing evaluation of $f(x) \in C_0^b(\mathcal{X}), \pi(x) \in \mathcal{P}(\mathcal{X})$ for $\forall x \in \mathcal{X}$.

Problem Set-up II.1: Numerical perspective [DH03]

Gauss Quadrature via polynomial interpolation (d=1)

- **Equidistant nodes**: Newton-Cotes formula [1]

$$I(F) := \int_a^b F(t)dt, \quad \hat{I}_n(F) = \sum_{i=0}^{n} \lambda_i F(t_i)$$

with $h_i = h = \frac{b-a}{n}$, $t_i = a + ih$, $i \in [n]$ and
$\lambda_{in} = \frac{1}{b-a} \int_a^b \prod_{i \neq j} \frac{t-t_i}{t_i-t_j} dt = \frac{1}{n} \int_0^n \prod_{i \neq j} \frac{s-j}{i-j} ds$
where nodes are chosen independently and weights can be pre-calculated based on different rules within each mesh:

n	$\lambda_{0n}, \ldots, \lambda_{nn}$	Error	Name
1	$\frac{1}{2}, \frac{1}{2}$	$h^3 F''(\tau)/12$	Trapezoidal rule
2	$\frac{1}{6}, \frac{4}{6}, \frac{1}{6}$	$h^5 F^{(4)}(\tau)/90$	Simpson's rule (Kepler's barrel rule)
3	$\frac{3}{8}, \frac{9}{8}, \frac{9}{8}, \frac{3}{8}$	$3h^5 F^{(4)}(\tau)/80$	Newton's 3/8-rule
4	$\frac{7}{90}, \frac{32}{90}, \frac{12}{90}, \frac{32}{90}, \frac{7}{90}$	$8h^7 F^{(6)}(\tau)/945$	Milne's rule

[1] need smoothness assumption up to constant

Problem Set-up II.2: Numerical perspective

Gauss Quadrature via polynomial interpolation (d=1)

- **Non-equidistant nodes**: Gauss-(Chebyshev, Laguerre, Hermite, Legendre,...) quadrature [2]

Consider quadrature of weighted integrals, with a positive weight function $\omega(t) > 0$:

$$I^w(F) := \int_a^b \omega(t)F(t)dt, \quad \hat{I}_n^w(F) := \sum_{i=0}^{n} \lambda_{in} F(\tau_{in})$$

with uniquely determined nodes $\tau_{0n}, \ldots, \tau_{nn}$ and weights $\lambda_{0n}, \ldots, \lambda_{nn}$.

$\omega(t)$	Interval $I = [a, b]$	Orthogonal polynomials
$\frac{1}{\sqrt{1-t^2}}$	$[-1, 1]$	Chebyshev polynomials T_n
e^{-t}	$[0, \infty)$	Laguerre polynomials L_n
e^{-t^2}	$(-\infty, \infty)$	Hermite polynomials H_n
1	$[-1, 1]$	Legendre polynomials P_n

[2] better guarantees while requiring stricter smoothness depending on n

Problem Set-up II.3: Numerical perspective

Quadrature in higher dimensions

A separable integral can be integrated dimension-wise:

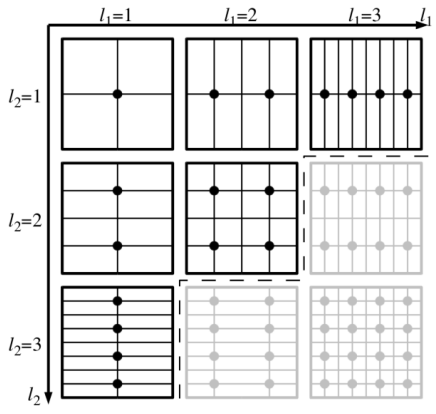
$$I_L = \int_a^b \int_a^b \phi(x,y)\mathrm{d}x\ \mathrm{d}y = \int_a^b \phi^{(x)}(x)\mathrm{d}x \int_a^b \phi^{(y)}(y)\mathrm{d}x \qquad (2)$$

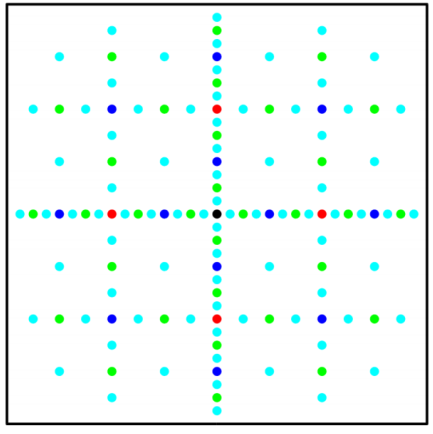
where $\phi(x,y) = \phi^{(x)}(x)\phi^{(y)}(y)$. So we consider choosing a basis $\phi_1(x,y), \ldots, \phi_n(x,y)$ and approximate $f(x,y) \approx \sum_{i=1}^n c_i \phi_i(x,y)$ with $\phi_i(x,y)$ such that (2) holds. Then integrate

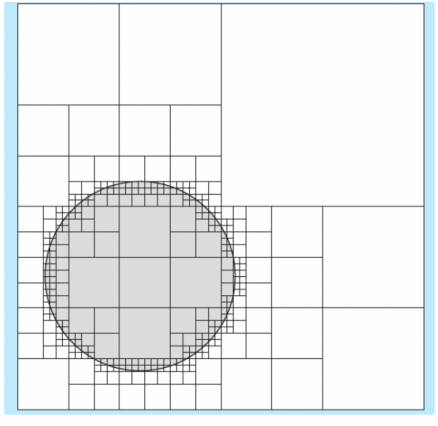
$$I(f) \approx \sum_{i=1}^n c_i \hat{I}\left(\phi_i^{(x)}\right) \hat{I}\left(\phi_i^{(y)}\right)$$

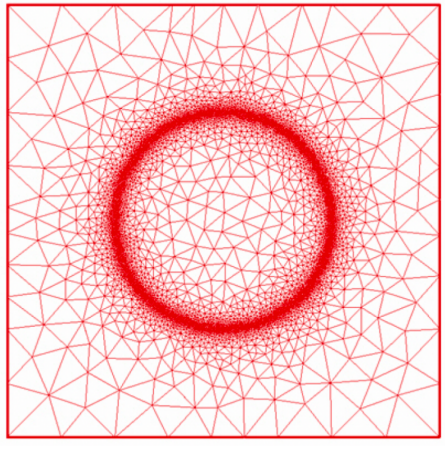
- Full grids: Consider mesh width $h_\ell = 2^{-\ell}$ with grid points $x_{\ell,i} = ih_\ell = i2^{-\ell}$ and basis functions $\phi_{\ell,i}(x) = \phi\left(\frac{x - x_{\ell,i}}{h_\ell}\right)$, where $\phi(x) = \max\{1 - |x|, 0\}$.
- Sparse grids: Construct hierarchical basis $\widetilde{V}_n = \bigoplus_{\ell=1}^n W_\ell$ where $W_\ell := \text{span}\left\{\phi_{\ell,i}\,;\, i \in I_\ell := \left\{j : 1 \le j < 2^\ell,\, j \text{ is odd}\right\}\right\}$ and truncate diagonally to get $V_n = \bigoplus_{|\ell|_1 \le n+d-1} W_\ell$.

Problem Set-up II.4: Numerical perspective









Full-grid v.s. Sparse-grid

The number of grid points of a sparse grid grows as $\mathcal{O}\left(2^n n^{d-1}\right)$ in contrast to $\mathcal{O}\left(2^{nd}\right)$ of a full grid, drastically reduced points in higher dimensions d, while sparse-grid space achieves:

$$\left\| u - u_n^{(SG)} \right\|_2 \in \mathcal{O}\left(2^{-2n} n^{d-1}\right)$$

whereas a full-grid space achieves:

$$\left\| u - u_n^{(FG)} \right\|_2 \in \mathcal{O}\left(2^{-2n}\right)$$

Problem Set-up II.5: Numerical perspective

Several remarks to make:

- curse of dimensionality
- deterministic instead of probabilistic/statistic

Problem Set-up III: Monte Carlo integration

If we can draw N i.i.d. samples $X^{(i)}$ from $\pi(x)$, then we can use the golden Monte Carlo estimator

$$\hat{\pi}_N(x) = \frac{1}{N}\sum_{i=1}^{N}\delta_{X^{(i)}}(x)$$

to approximate the target integration:

$$\hat{f} := \frac{1}{N}\sum_{i=1}^{N} f\left(X^{(i)}\right) = \mathbb{E}_{\hat{\pi}}[f] \approx \mathbb{E}_{\pi}[f]$$

It is apparent that the Monte Carlo estimator is unbiased:

$$\mathbb{E}[\hat{f}] = \mathbb{E}_{\pi}[f]$$

with variance:

$$\mathrm{Var}(\hat{f}) = \frac{\mathrm{Var}(f)}{N}$$

Therefore, the problem is reduced to sampling from a target distribution $\pi(x)$.

Outline

Outline

Inverse Sampling I

The cumulative density function (CDF) of $\pi(x)$ is:

$$F_X(x) = \Pr(X \leq x) = \int_{-\infty}^{+\infty} \pi(u) \mathbb{1}_{\{u \leq x\}} du = \int_{-\infty}^{x} \pi(u) du$$

Suppose we have access to the inverse CDF and we can successfully sample from uniform distribution, we arrive at the simplest algorithm for exact sampling:

Algorithm 1 Inverse Transform Sampling

Require: CDF $F_X(x) = \int_{-\infty}^{x} \pi(u)\, du$

1: Sample $u \sim \mathcal{U}(0, 1)$
2: Compute $X \leftarrow F_X^{-1}(u)$
3: **return** X

Proof of correctness:

$$\Pr(F_X^{-1}(u) \leq x) = \Pr\left(u \leq F_X(x)\right) = F_X(x)$$

Inverse Sampling II

Several remarks to make:

- very limited to cases where the inverse cdf has an analytical form that can be tabulated.
- very limited to low dimensions (usually d=1)
- analog: transformation method (e.g., Box Muller algorithm for sampling gaussians)

In many problems, we only know $\pi(x)$ up to a normalizing constant as

$$\gamma(x) = \mathcal{Z}\,\pi(x).$$

where the *normalizing constant*

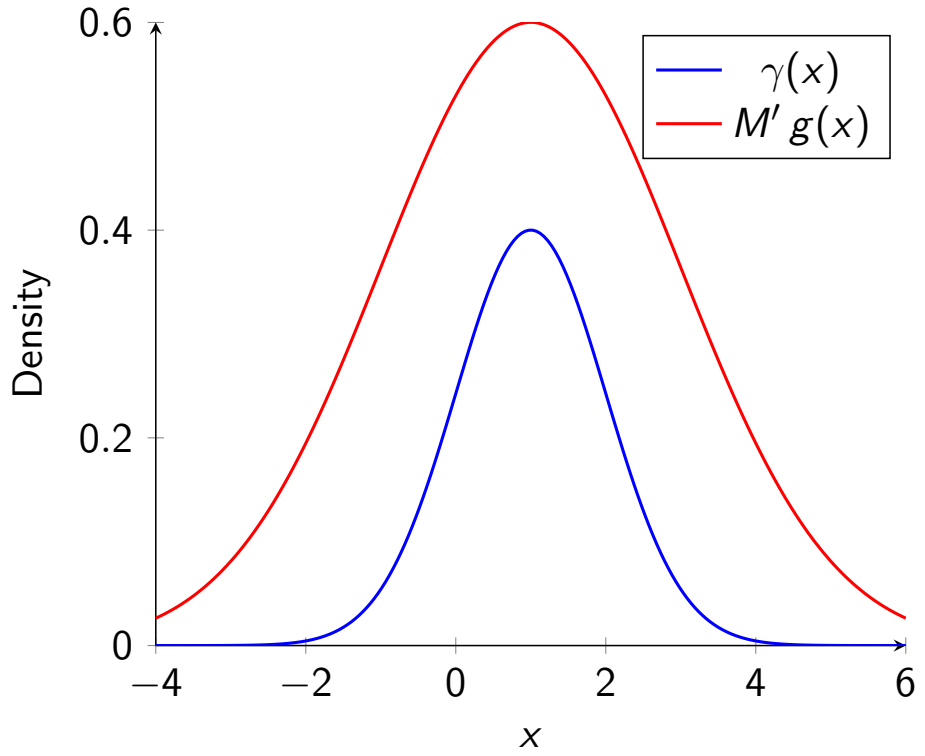
$$\mathcal{Z} := \int_{\mathcal{X}} \gamma(x)\,dx$$

is often intractable.

Outline

Rejection Sampling I: motivation

Suppose we have a proposal distribution $q(x)$ that is easy to sample and also known as $g(x)$ up to a normalizing constant, such that $\gamma(x) < M'g(x)$ for all x for some $M' \geq M$ where $M = \sup_{x \in X} \frac{\gamma(x)}{g(x)} < +\infty$. This implies that $\gamma(x) > 0 \Rightarrow g(x) > 0$, and also that the tails of $g(x)$ must be thicker than the tails of $\gamma(x)$.



Rejection Sampling II: algorithm

Algorithm 2 Rejection Sampling

Require: Unnormalized target density $\gamma(x)$; Unnormalized proposal density $g(x)$; Constant $M > 0$ such that $\gamma(x) \leq M' g(x)$ for all x; Desired sample size N

for $i = 1, \ldots, N$ **do**
  repeat
    Sample $Y \sim g(y)$
    Sample $u \sim \mathcal{U}(0, 1)$
  if $u \leq \dfrac{\gamma(Y)}{M' g(Y)}$ **then**
    Set $X^{(i)} \leftarrow Y$
    break
  end if
  until a sample is accepted
end for
return $\{X^{(1)}, \ldots, X^{(N)}\}$

Rejection Sampling III: correctness

Correctness of rejection sampling:

$$\Pr(\mathrm{Y} \leq x \text{ and Y accepted}) = \int_{-\infty}^{x} \frac{\gamma(y)}{M' g(y)} q(y) dy = \frac{\int_{-\infty}^{x} \gamma(y) dy}{M' \int_{\mathcal{X}} g(y) \mathrm{d}y}$$

$$\boxed{\gamma := \Pr[Y \text{ accepted}] = \frac{\int_{\mathcal{X}} \gamma(y) \mathrm{d}y}{M' \int_{\mathcal{X}} g(y) \mathrm{d}y}}$$

$$\implies \Pr(\mathrm{Y} \leq x \mid \mathrm{Y} \text{ accepted}) = \frac{\Pr(\mathrm{Y} \leq x \text{ and Y accepted })}{\Pr(\mathrm{Y} \text{ accepted })}$$

$$= \frac{\frac{\int_{-\infty}^{x} \gamma(y) dy}{M' g(y) \mathrm{d}y}}{\frac{\int_{\mathcal{X}} \gamma(y) dy}{M' \int_{\mathcal{X}} g(y) \mathrm{d}y}} = \int_{-\infty}^{x} \pi(y) dy$$

Rejection Sampling IV: remarks

Indeed, the number of trials before a candidate sample is accepted follows a geometric distribution:

$$\Pr\left(k^{\text{th}} \text{ proposal is accepted}\right) = (1-\gamma)^{k-1}\gamma$$

So the number of trials before success is an unbiased estimate of $1/\gamma$.
Several remarks to make:

- A tractable good proposal density $g(x)$ is often not feasible in high dimensions.
- The bounded constant M a.s. diverges as dimension grows, i.e., the acceptance rate $\gamma \to 0$ as d grows, i.e., the number of trials before success $1/\gamma \to \infty$ as d grows. [3]

Can we have a softer version of rejection?

[3] e.g., even for high-dimensional gaussians

Outline

Importance sampling I.1: motivation

Motivated by:

$$\mathbb{E}_\pi[f] = \int_{\mathcal{X}} f(x)\pi(x)dx = \int_{\mathcal{X}} f(x)\frac{\pi(x)}{q(x)}q(x)\mathrm{dx} = \mathbb{E}_q[w(X)f(X)]$$

where $w(x) := \pi(x)/q(x)$, we transform the original problem from sampling $\pi(x)$ into sampling $q(x)$ and reweighting. This corresponds to the following Monte Carlo estimator of $\pi(x)$:

$$\hat{\pi}_N(x) = \frac{1}{N}\sum_{i=1}^{N} w\left(X^{(i)}\right)\delta_{X^{(i)}}(x), \text{ where } X^{(i)} \overset{i.i.d.}{\sim} q$$

Though this Monte Carlo estimator is unbiased, we would usually prefer a normalized version which is biased yet asymptotically unbiased, due to (1) we then only need to know $w(X^{(i)})$ as $\widetilde{w}(X^{(i)})$ up to a division of normalizing constants; (2) a smaller MSE (see in [Liu08]):

$$\hat{\pi}_N(x) = \sum_{i=1}^{N} W^{(i)}\delta_{X^{(i)}}(x), \text{ where } W^{(i)} = \frac{\widetilde{w}\left(X^{(i)}\right)}{\sum_{j=1}^{N}\widetilde{w}\left(X^{(j)}\right)}$$

Importance sampling I.2: motivation

The normalized version can also be used in the usual setting where we only know $\pi(x)$ as $\gamma(x)$ up to a normalizing constant, and we only have a proposal distribution $q(x)$ that is easy to sample but also known as $g(x)$ up to a normalizing constant, then we have:

$$\pi(x) = \frac{\gamma(x)}{\int_{\mathcal{X}} \gamma(x)dx} = \frac{\frac{\gamma(x)}{g(x)}g(x)}{\int_{\mathcal{X}} \frac{\gamma(x)}{g(x)}g(x)dx} = \frac{\widetilde{w}(x)q(x)}{\int_{\mathcal{X}} \widetilde{w}(x)q(x)dx}$$

where $\widetilde{w}(x) := \gamma(x)/g(x)$. We can even estimate the normalizing constant $\mathcal{Z} := \int_{\mathcal{X}} \gamma(x)\, dx$ as long as we have access to $q(x)$ by:

$$\hat{Z}_N = \frac{1}{N}\sum_{i=1}^{N} \frac{\gamma\left(X^{(i)}\right)}{q\left(X^{(i)}\right)}, \text{ where } X^{(i)} \overset{\text{i.i.d.}}{\sim} q$$

Importance sampling II: algorithm

Algorithm 3 Importance Sampling

Require: Unnormalized target density $\gamma(x)$; unnormalized proposal $g(x)$ (or normalized density $q(x)$); number of samples N; function $f(x)$ whose expectation is to be estimated.

for $i = 1, \ldots, N$ **do**
  Sample $X^{(i)} \sim q(x)$.
  Compute the unnormalized weight: $\widetilde{w}(X^{(i)}) \leftarrow \frac{\gamma(X^{(i)})}{g(X^{(i)})}$.
end for
Compute the sum $S \leftarrow \sum_{j=1}^{N} \widetilde{w}(X^{(j)})$.
for $i = 1, \ldots, N$ **do**
  Set the normalized weight: $W^{(i)} \leftarrow \frac{\widetilde{w}(X^{(i)})}{S}$.
end for
Estimate the expectation: $\hat{\mathbb{E}}[f] \leftarrow \sum_{i=1}^{N} W^{(i)} f(X^{(i)})$.
if $q(x)$ is provided **then**
  Compute the normalizing constant: $\hat{Z}_N \leftarrow \frac{1}{N} \sum_{i=1}^{N} \frac{\gamma\left(X^{(i)}\right)}{q\left(X^{(i)}\right)}$
end if
return $\{X^{(i)}, W^{(i)}\}_{i=1}^{N}$, $\hat{\mathbb{E}}[f]$, $\hat{Z}_N$

Importance sampling III.1: example

Consider the following integral as an example (see here for a MATLAB implementation):

$$I = \iint_{[-1,1]\times[-1,1]} f(x,y)dxdy$$

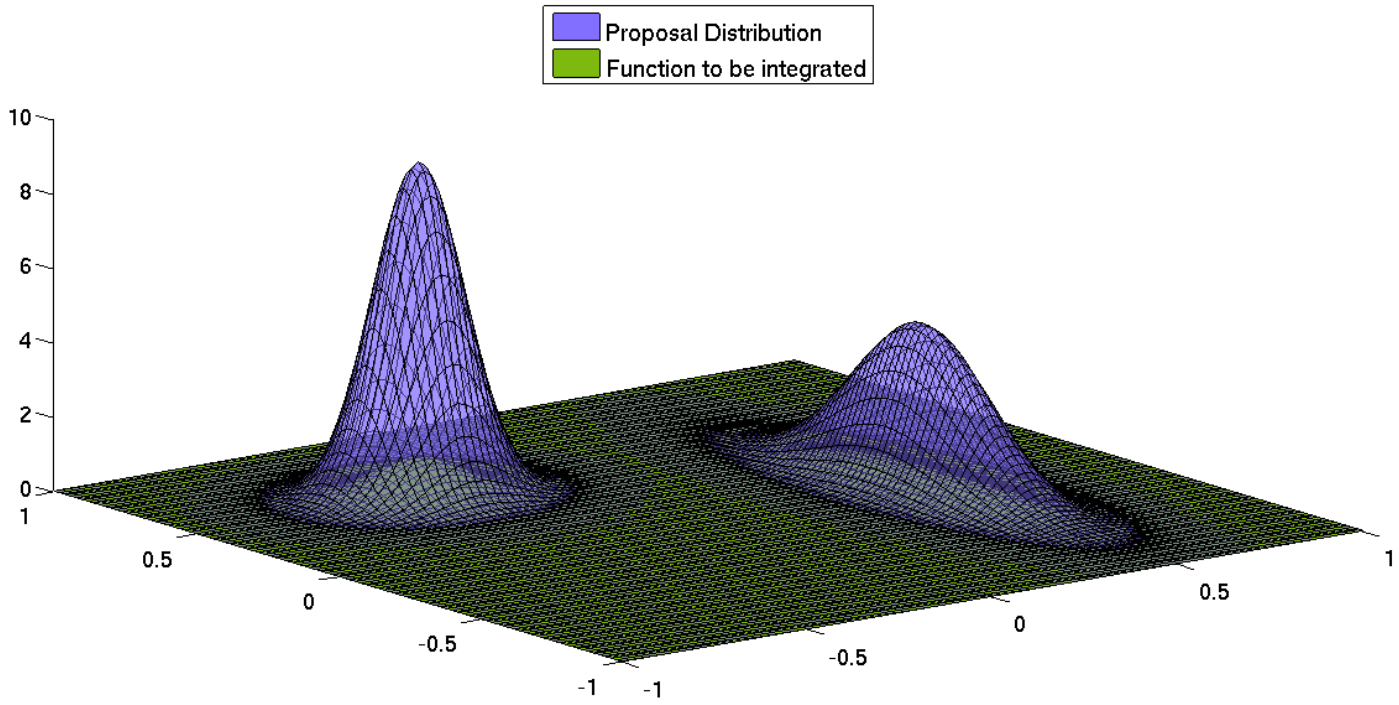
where

$$f(x,y) = 0.5e^{-90(x-0.5)^2-45(y+0.1)^2} + e^{-45(x+0.4)^2-60(y-0.5)^2}$$

Our proposal distribution is:

$$q(x,y) = 0.46 \cdot \mathcal{N}\left(\begin{bmatrix} 0.5 \\ -0.1 \end{bmatrix}, \begin{bmatrix} 1/180 & 0 \\ 0 & 1/20 \end{bmatrix}\right)$$
$$+ 0.54 \cdot \mathcal{N}\left(\begin{bmatrix} -0.4 \\ 0.5 \end{bmatrix}, \begin{bmatrix} 1/90 & 0 \\ 0 & 1/120 \end{bmatrix}\right)$$

Importance sampling III.2: example



Importance sampling III.3: example

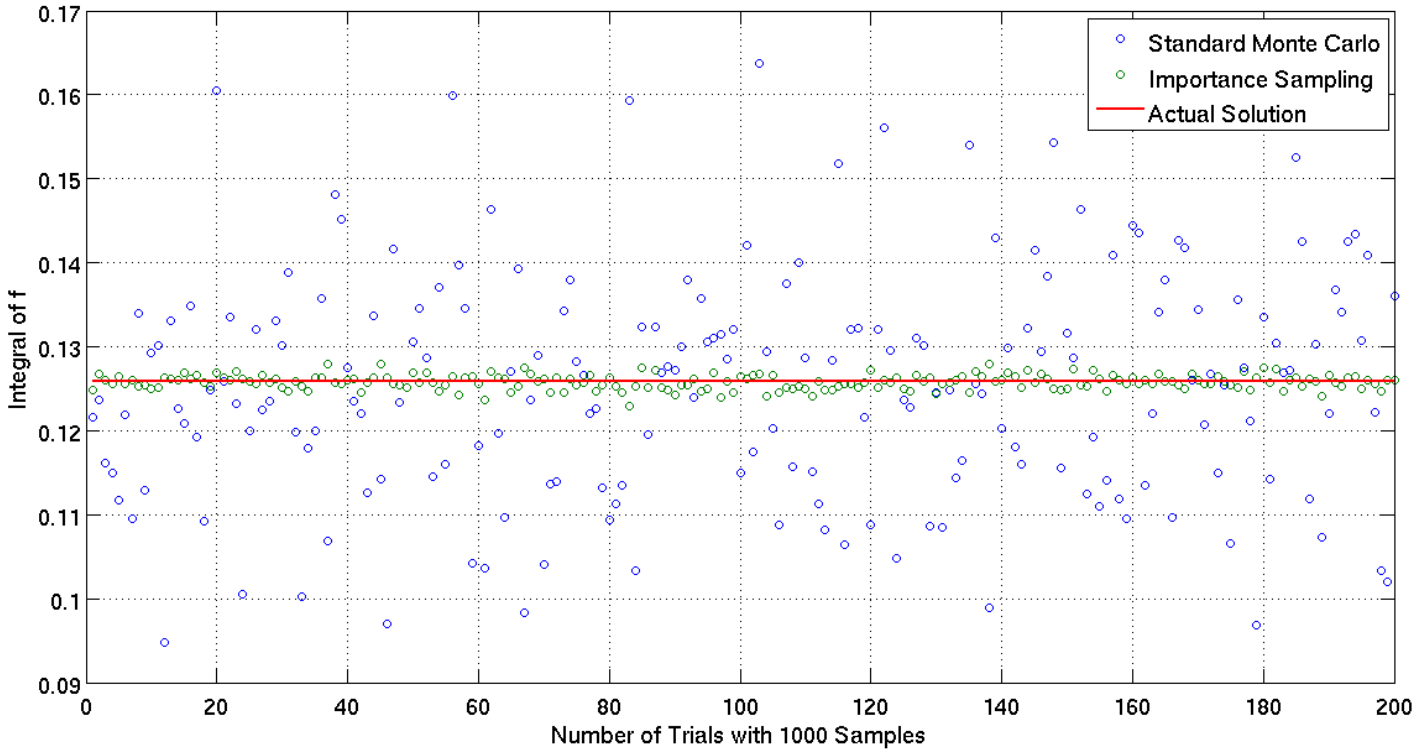


Figure 1: $N = 1000$, count $= 200$ (we take 1000 random sample points per run and run the simulation 200 times)

Importance sampling III.4: example

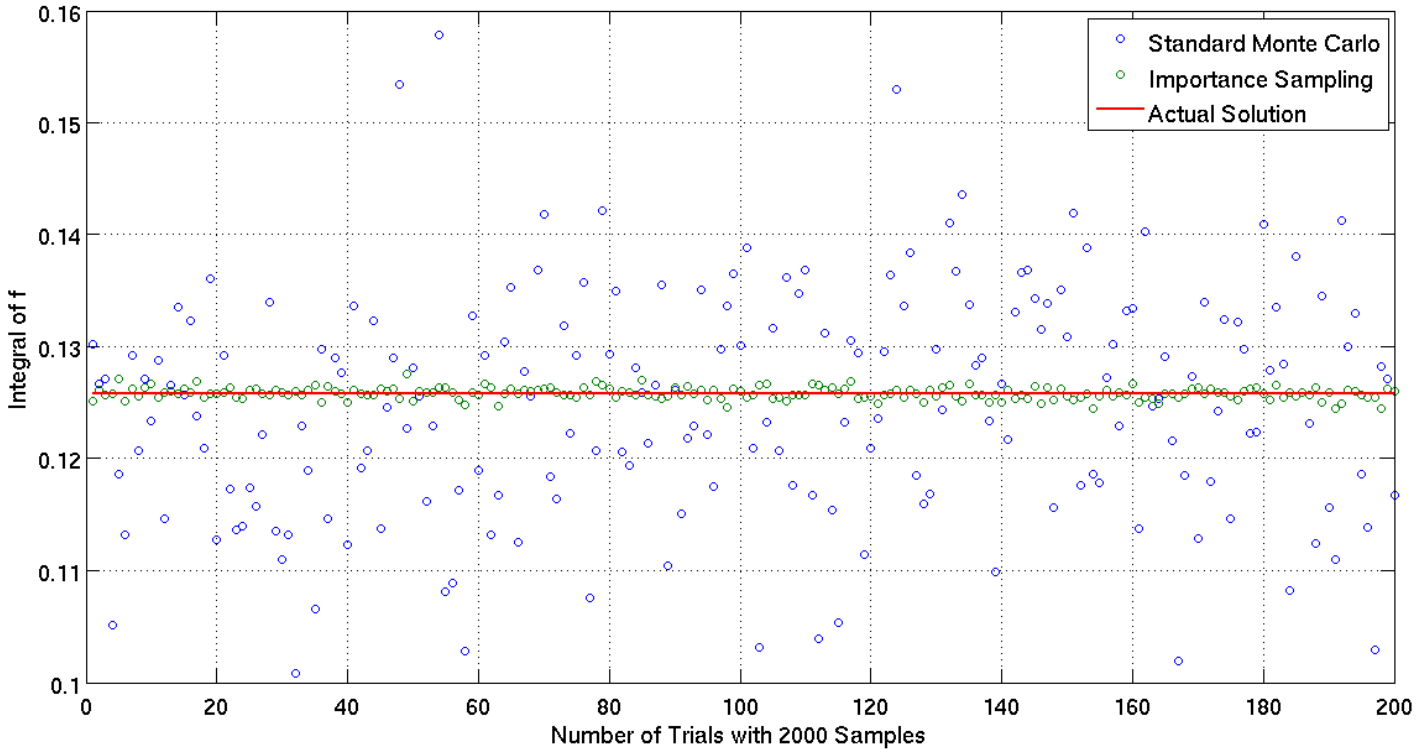


Figure 2: $N = 2000$, count $= 200$ (we take 2000 random sample points per run and run the simulation 200 times)

Importance sampling IV: remarks

Several remarks to make:

- The general idea behind importance sampling is to find properly weighted samples, but it is difficult and often impossible to select the proper $q(x)$ in high dimensions.
- The variance of the weights $\mathrm{Var}_q(w)$ increases exponentially fast with dimensionality despite a good choice of $q(x)$, i.e., the *Effective Sample Size ESS* $:= m/(1 + \mathrm{Var}_q(w))$ decreases exponentially to 0 as d grows. [4]

[4] albeit there exists few high-dimensional scenarios where IS succeeds as $q(x)$ successfully tends to $\pi(x)$, this is then a chicken-egg problem as thus $\pi(x)$ itself is not hard to sample at all.

Outline

MCMC I.0: motivation

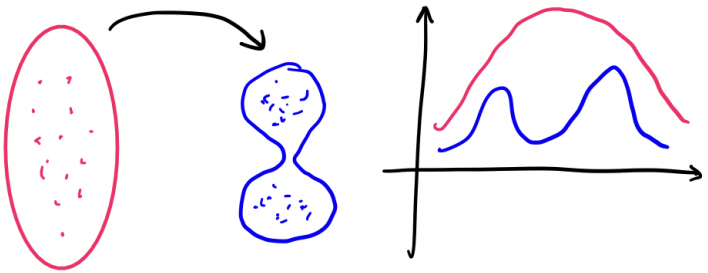


Figure 3: Importance Sampling

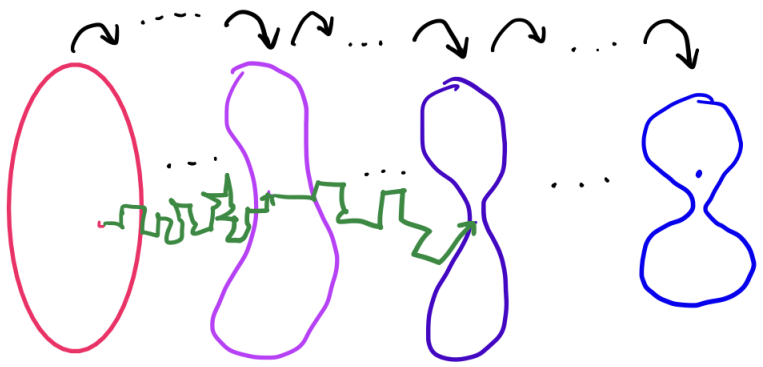


Figure 4: Markov Chain Monte Carlo

MCMC I.1: motivation

Definition: Markov Chain

A homogeneous Markov chain is a sequence of random variables $\{X_n, n \in \mathbb{N}\}$ defined on $(X, \mathcal{B}(X))$ such that for any $A \in \mathcal{B}(X)$ the following probability condition is satisfied:

$$\mathbb{P}(X_n \in A \mid X_0, \ldots, X_{n-1}) = \mathbb{P}(X_n \in A \mid X_{n-1})$$

with homogeneous transition kernel: $K(x, A) := \mathbb{P}(X_n \in A \mid X_{n-1} = x)$

Markov Chain Monte Carlo (MCMC): Given a target distribution π, we need to design a Markov transition kernel K such that asymptotically

$$\frac{1}{N}\sum_{n=1}^{N} f(X_n) \xrightarrow{N\to\infty} \int f(x)\pi(x)dx \text{ a.s. and/or } X_n \sim \pi \text{ (stricter)}$$

Luckily, (1) It is easy to simulate the Markov Chain even if π is complex; (2) Under mild conditions (irreducible, aperiodic, invariant), such an estimator is consistent. Under additional conditions (Lyapunov condition and minorization property), the CLT also holds with a convergence rate $\mathcal{O}(1/\sqrt{N})$.

MCMC I.2: motivation

In principle, we need the chain to satisfy three key properties [5] [6]:

I. The desired distribution π is an **invariant** [7] distribution of the Markov chain, i.e.

$$\int_x \pi(x)P(x,y)dx = \pi(y)$$

II. The chain is **irreducible** if for $\forall x, y \in \mathcal{X}$:

$$\Pr(X_t = x | X_0 = y) > 0 \text{ for some } t > 0$$

III. The chain is **aperiodic** if $\exists t_0 > 0$ s.t.:

$$\Pr(X_{t_0} = x | X_0 = x) > 0 \text{ for } \forall t > t_0$$

---

[5] Indeed, (I)+(II)+(III) implies **ergodicity**, saying $\forall x \in \mathcal{X}$, $\lim_{t\to\infty} \left\| P^t(x,\cdot) - \pi(\cdot) \right\|_{\mathrm{TV}} = 0$

[6] Indeed, (II) and (III) imply **primitivity**, saying $\exists t_0 > 0$ s.t. for $\forall x, y \in \mathcal{X}$: $\Pr(X_t = x | X_0 = y) > 0$ for $\forall t > t_0$

[7] A stricter condition is the detailed balance condition(**reversibility**): for $\forall x, y \in \mathcal{X}$, $\pi(x)P(x,y) = \pi(y)P(y,x)$

Outline

Gibbs Sampling I: RSGS algorithm

Algorithm 4 Random-Scan Gibbs Sampler

Require: Initial state $X^{(0)} = (X_1^{(0)}, X_2^{(0)}, \ldots, X_d^{(0)})$; normalized target density $\pi(x)$; number of iterations N.

1: **for** $t = 1, 2, \ldots, N$ **do**

2: Randomly select an index $i_t \sim \text{Uniform}(\{1, 2, \ldots, d\})$.

3: Update the i_t-th coordinate by sampling from its full conditional distribution:

$$X_{i_t}^{(t)} \sim \pi\left(x_{i_t} \,\middle|\, X_{-i_t}^{(t-1)}\right),$$

where $X_{-i_t}^{(t-1)}$ represents the vector of current values for all coordinates except x_{i_t}.

4: For every $j \neq i_t$, set

$$X_j^{(t)} \leftarrow X_j^{(t-1)}.$$

5: **end for**

6: **return** $\{X^{(t)}\}_{t=0}^{N}$.

Gibbs Sampling II: SSGS algorithm

Algorithm 5 Systematic-scan Gibbs Sampler

Require: Initial state $X^{(0)} = \left(X_1^{(0)}, X_2^{(0)}, \ldots, X_d^{(0)}\right)$; normalized target density $\pi(x)$; number of iterations N.

1: **for** $t = 1, 2, \ldots, N$ **do**

2: **for** $i = 1, 2, \ldots, d$ **do**

3: Update the ith coordinate by sampling from its full conditional distribution:

$$X_i^{(t)} \sim \pi\left(x_i \,\middle|\, X_1^{(t)}, \ldots, X_{i-1}^{(t)}, X_{i+1}^{(t-1)}, \ldots, X_d^{(t-1)}\right).$$

4: For indices $j < i$, use the updated values $X_j^{(t)}$; for indices $j > i$, retain the previous iteration's values $X_j^{(t-1)}$.

5: **end for**

6: **end for**

7: **return** $\{X^{(t)}\}_{t=0}^{N}$.

Gibbs Sampling III: remarks

Several remarks to make:

- One can verify that the Markov chain in Gibbs sampling (for both schemes) satisfies irreducibility, aperiodicity, and $\pi(x)$ as the unique invariant distribution.
- The motto of Gibbs sampling is to fix some components of the chain at each step and preserves the conditional density of $\pi(x)$ for the remaining components therefore leaves $\pi(x)$ invariant. It is generally the **partial resampling principle**.
- Gibbs sampling only succeeds when sampling from conditionals is sometimes feasible even when sampling from the joint is impossible.
- The choice of coordinates(ordering) such that the conditional distribution is simple can be often very hard, and even in special cases which this is possible, fixing the choice of coordinates can lead to slow convergence rate due to poor conditioning of $\pi(x)$ in those coordinates.

Gibbs Sampling III: example

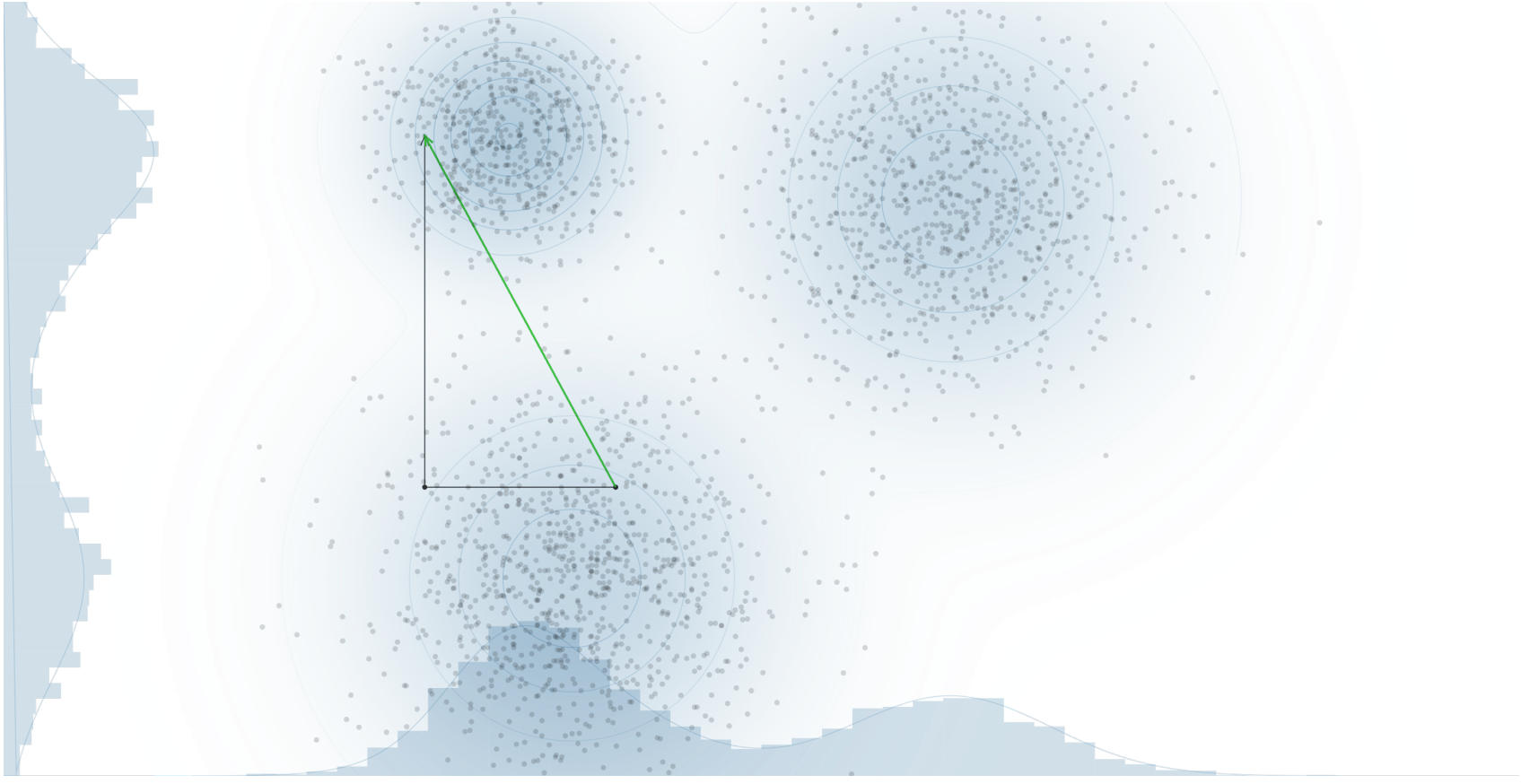


Figure 5: Gibbs Sampling [github]

Outline

Metropolis-Hastings Algorithm I: algorithm

Algorithm 6 Metropolis-Hastings Algorithm

Require: Unnormalized target density $\gamma(x)$; proposal distribution $q(x, x')$; initial state $X^{(0)}$; number of iterations N.

1: **for** $i = 1, 2, \ldots, N$ **do**
2: Draw a proposal X^* from $q\left(X^{(i-1)}, \cdot\right)$.
3: Compute the acceptance probability

$$\alpha\left(X^{(i-1)}, X^*\right) \leftarrow \min\left\{1, \frac{\gamma\left(X^*\right)\, q\left(X^*, X^{(i-1)}\right)}{\gamma\left(X^{(i-1)}\right)\, q\left(X^{(i-1)}, X^*\right)}\right\}.$$

4: Draw $u \sim \mathcal{U}[0, 1]$.
5: **if** $u \leq \alpha\left(X^{(i-1)}, X^*\right)$ **then**
6: Set $X^{(i)} \leftarrow X^*$.
7: **else**
8: Set $X^{(i)} \leftarrow X^{(i-1)}$.
9: **end if**
10: **end for**
11: **return** $\{X^{(i)}\}_{i=0}^{N}$.

Metropolis-Hastings Algorithm II.1: remarks

Several remarks to make:

- It is indeed that the Markov chain in Metropolis-Hastings algorithm is a.s. irreducible and aperiodic [Tie94], and the M-H algorithm converges under very weak assumptions to the target distribution π.
- Different proposals result to different algorithms, e.g. symmetric proposal leads to the earliest Metropolis algorithm; independent proposal $q(x, x') = q(x')$ with bounded constraints leads to **independent M-H algorithm**, Gaussian proposal leads to **random-walk M-H algorithm**, etc.
- There are generalizations of proposal forms, namely, mixture of proposals, e.g. **Multiple-Try Metropolis** by multiple independent proposals with its different forms in various contexts. We can also combine MH-step with Gibbs sampling to enhance performance as a **hybrid algorithm**.
- To implement the Metropolis scheme we only need to know the target density $\pi(x)$ up to a constant $\gamma(x)$.

Metropolis-Hastings Algorithm II.2: remarks

Several more remarks to make:

- It is intuitively clear that if we choose the proposal to be nearly reversible (not symmetric), we nearly get a probability 1 acceptance, i.e., the chain converges fast, but it is expected to be very hard in high dimensions. In fact, as d grows, the average acceptance probability grows exponentially close to 0.
- Because the samples, $X^{(k)}$, generated by a typical MCMC scheme are only asymptotically distributed according to the target distribution π, so for finite N the MCMC estimator $\bar{f}_N$ will have a bias. In fact, with some assumptions on how the chain is generated, the final sample $X^{(N)}$ can be reweighted so that it can be used to compute unbiased (or nearly unbiased) averages against π even when N is finite.
- The idea of M-H algorithm is incredibly flexible, there is still a huge potential combining it with other methods, while in general it still suffers from a low convergence rate in high dimensions and vague non-asymptotic guarantees.

Metropolis-Hastings Algorithm III: example

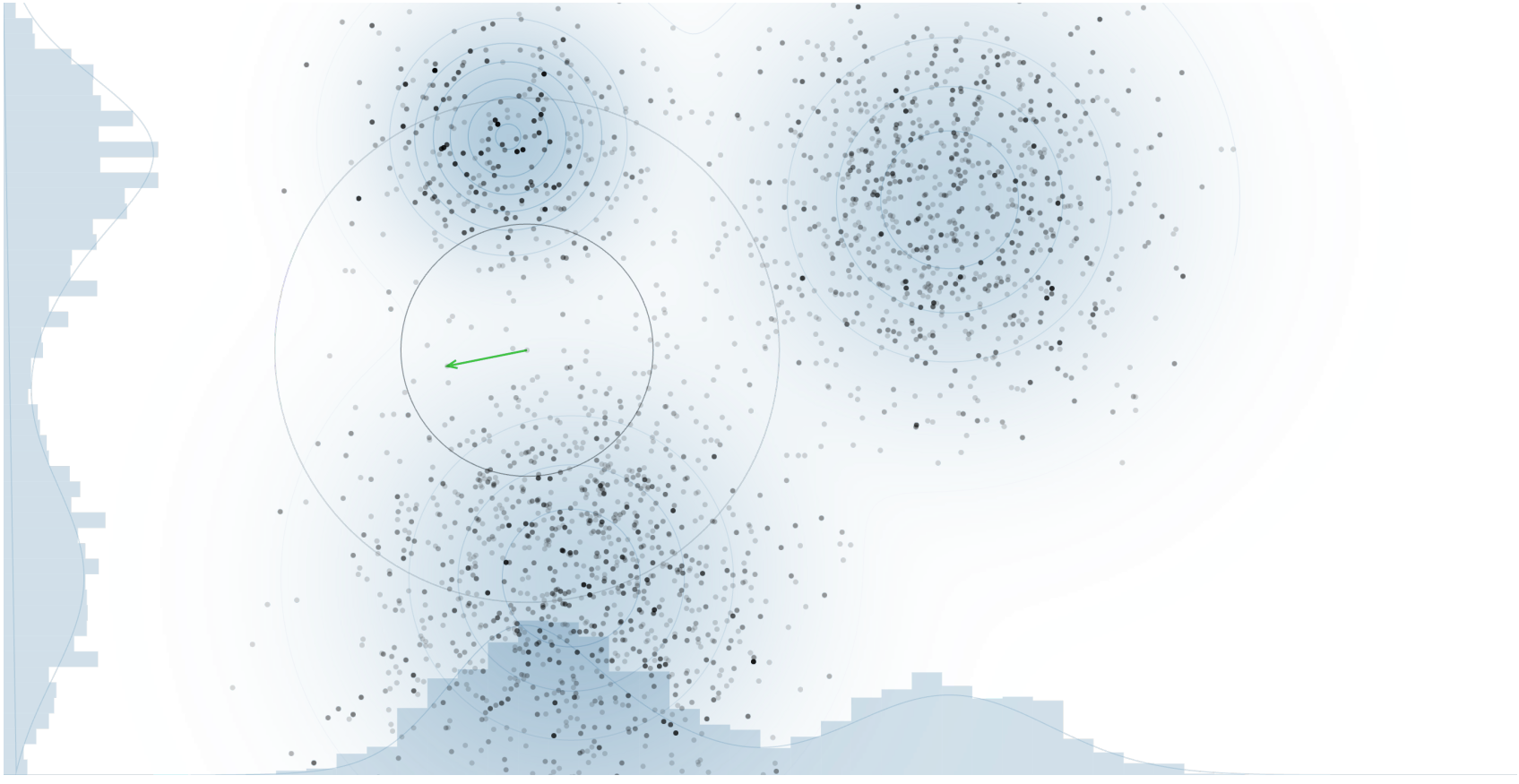


Figure 6: Random Walk Metropolis-Hastings [github]

Outline

Langevin Dynamics I.1: motivation

Let's impose some structures on the target distribution $\pi(x) = e^{-U(x)}$!

Langevin Dynamics I.2: motivation

Langevin dynamics can produce samples from a probability density $\pi(x)$ using only the score function $\nabla_x \log \pi(x) = -\nabla_x U(x)$. Given a fixed step size $\epsilon > 0$, and an initial value $X_0 \sim \pi_0(x)$ with $\pi_0(x)$ being any prior distribution, the Langevin dynamics recursively compute the following noisy gradient descent step for $t \in [T]$:

$$X_t = X_{t-1} + \frac{\epsilon}{2} \nabla_x \log p\left(X_{t-1}\right) + \sqrt{\epsilon} Z_t \tag{3}$$

where $Z_t \sim \mathcal{N}(0, I_d)$. The distribution of X_T equals $\pi(x)$ when $\epsilon \to 0$ and $T \to \infty$, in which case X_T becomes an exact sample from $\pi(x)$ under some regularity conditions [WT11]. When $\epsilon > 0$ and $T < \infty$, a Metropolis-Hastings update is needed to correct the error of (3), but it can often be ignored in practice [CFG14] [DM19] [NHH+19]. The error is negligible when ϵ is small and T is large.

Langevin Dynamics I.3: motivation

Note that sampling from (3) only requires the score function $\nabla_x \log \pi(x) = -\nabla_x U(x)$. In the score-based generative modeling setting, we do not have access to the target distribution, thus no knowledge of the potential $U(x)$, so parameterizing this score function by a Neural Network is thoroughly proposed by Song et al. in 2020 [SSDK$^+$20] as the basis of score-based generative modeling frameworks. In our statistical sampling setting, we do have the access to $U(x)$, so we can run Langevin Dynamics immediately, while the problem is that it can be very slow when $U(x)$ is nonconvex and multimodal.

Langevin Dynamics II: algorithm

Algorithm 7 (Unadjusted) Langevin Dynamics

Require: Target density $\pi(x) = e^{-U(x)}$; initial state $X^{(0)}$; step size ϵ; number of iterations N.

1: **for** $i = 1, \ldots, N$ **do**

2: Compute the gradient $\nabla U(X^{(i-1)})$.

3: Sample $Z^{(i)} \sim \mathcal{N}(0, I_d)$

4: Update the state:

$$X^{(i)} \leftarrow X^{(i-1)} - \frac{\epsilon}{2}\nabla U(X^{(i-1)}) + \sqrt{\epsilon}\, Z^{(i)}.$$

5: **end for**

6: **return** $\{X^{(i)}\}_{i=0}^{N}$

Langevin Dynamics III.1: remarks

Several remarks to make:

- The reason of choosing the step size for diffusion $\widetilde{\eta}$ as $\sqrt{2\eta}$ of the step size for drifting, can be justified easily in order to have $\Theta(1)$ quantities in the RHS of (3).
- Langevin Diffusion is a special instance of Ito diffusion where a Markov semi-group $(K_t\phi)(x) := \mathbb{E}[\phi(X_t) \mid X_0 = x]$ can be associated, followed by an infinitesimal generator $\mathcal{L}\phi := \lim_{t\to 0^+} \frac{K_t\phi - \phi}{t}$ for any test function ϕ. Thus the associated *Kolmogorov's backward equation* can be justified as:

$$\partial_t K_t\phi = \mathcal{L}K_t\phi = K_t\mathcal{L}\phi$$

and a forward equation named as *Fokker-Planck equation* can also be justified as:

$$\partial t\pi_t = \mathcal{L}^*\pi_t$$

Langevin Dynamics III.2: remarks

Several more remarks to make:

- In fact, Langevin dynamics can be understood as a gradient flow of the relative entropy functional w.r.t. the Wasserstein metric.
- With the assumption of *Bakry-Émery Criterion* (or a more general version, *Holley-Stroock Perturbation Principle*) saying $U(x)$ is λ-strongly convex, we can show that the convergence of Langevin dynamics satisfies the *log-Sobolev inequality*:

$$KL(\pi_t \| \pi) \leqslant e^{-2\lambda t} KL(\pi_0 \| \pi)$$

- For discretization algorithms for (3), the state-of-the-art result for log-concave sampling gives a guarantee of the iteration step T of form $T \gtrsim \Theta\left(k\sqrt{d}\log\varepsilon^{-1}\right)$ to reach accuracy ε where $k := \beta/\alpha$ with $\alpha I \leqslant \nabla^2 f \leqslant \beta I$ for $0 < \alpha \leqslant \beta$. See [AC23].
- For nonconvex potential $U(x)$, we need $\widetilde{\Theta}\left(\varepsilon^{-2}\right)$ to find ε-approximate local minimum.
- See everything in more details through the book "Log-Concave Sampling" by Chewi.

Langevin Dynamics III.3: remarks

Several more more remarks to make:

- The special choice of $U(x) = \frac{1}{2}\|x\|^2$ leads to the so-called *Ornstein–Uhlenbeck (OU)* process as to be *variance-preserving*, formally it corresponds to a canonical choice of the *forward process* in diffusion models, however this just happens coincidentally as in diffusion models we choose the OU process to be a noising schedule of the unknown target distribution. A broader class of "noising schedules" including OU process hold *reverse processes*, where the score function shows up again to be a key component. It seems to have some relations to Langevin dynamics, but again this is a coincidence. Yang et al. invented the first original idea to combine these two coincidences together, termed as "Annealed Langevin Dynamics" for sampling, and started the field of score-based generative modeling. See their paper [SSDK+20] and Song's blog `https://yang-song.net/blog/2021/score/`.

Langevin Dynamics IV: example

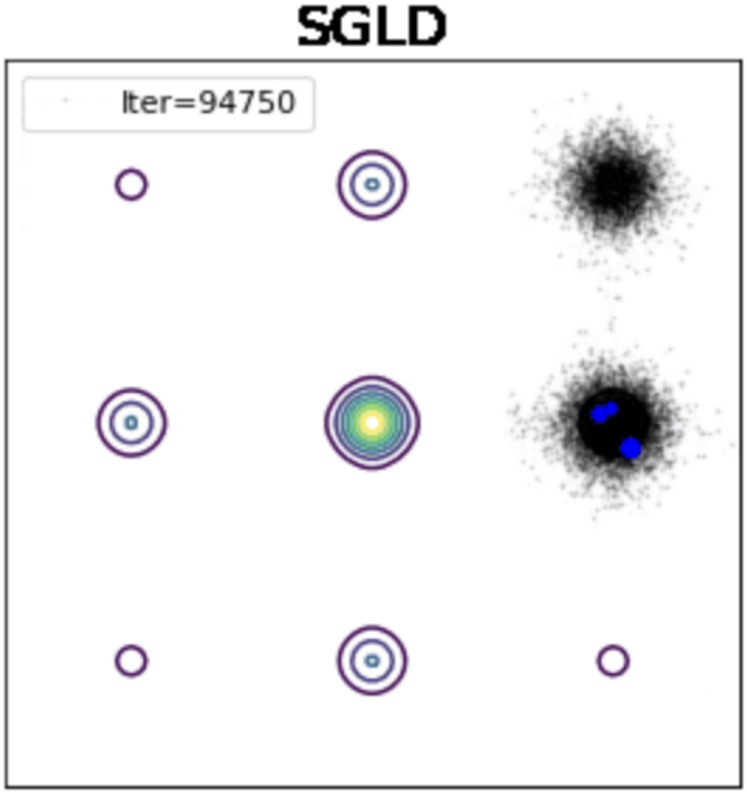


Figure 7: (Stochastic Gradient) Langevin Dynamics [github]

MCMC: final remarks

Several last remarks to make:

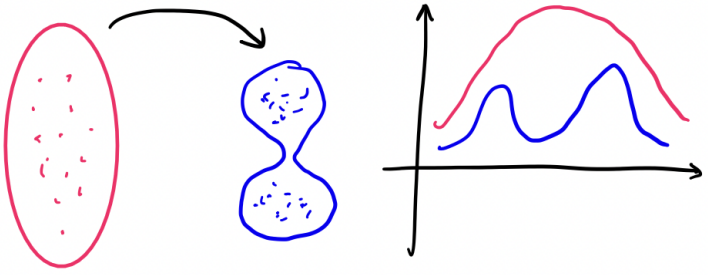
- Importance Sampling: Parallelization(efficiency) but lack Exploration(local and stable)
- MCMC: *Exploration*(local and stable) but lack *Parallelization*(efficiency)

> Can we combine their strengths together?

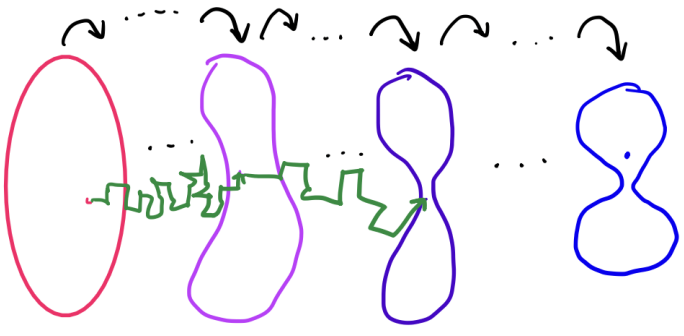
From equilibrium sampling to non-equilibrium sampling by introducing bridges/paths...

Outline

Sequential Monte Carlo I: motivation



(a) Importance Sampling



(b) Markov Chain Monte Carlo

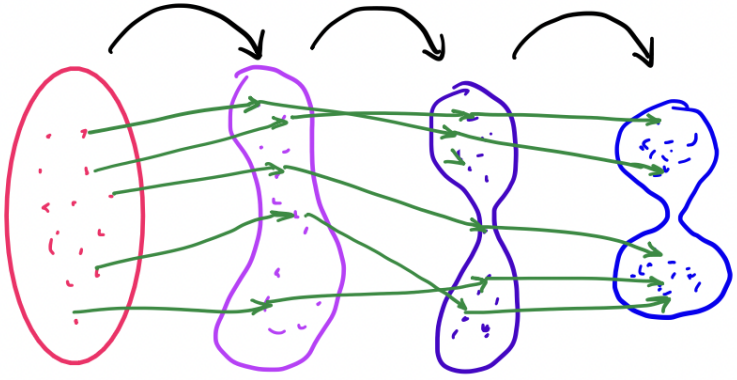


Figure 9: Sequential Monte Carlo (Sequential Importance Sampling + Resampling)

Sequential Monte Carlo II.1: SIS motivation

It is nontrivial to design a good trial distribution for IS in high dimensional problems. One of the most useful strategies is to build up the trial density sequentially, in the sense of breaking hard problems into manageable pieces. Learning sequentially is a natural task for many problems (e.g., state space models such as **hidden markov models**).

Suppose we are situated in a high-dimensional setting: $\mathbf{x} = (x_1, \ldots, x_d)$. By Bayes rule, we can always decompose target distribution as:

$$\pi(\mathbf{x}) = \pi(x_1)\,\pi(x_2 \mid x_1) \cdots \pi(x_d \mid x_1, \ldots, x_{d-1})$$

Consider building our reference distribution $q(\mathbf{x})$ sequentially:

$$q(\mathbf{x}) = q_1(x_1)\,q_2(x_2 \mid x_1) \cdots q_d(x_d \mid x_1, \ldots, x_{d-1})$$

Then the importance weights in standard IS turns out to be:

$$w(\mathbf{x}) = \frac{\pi(x_1)\,\pi(x_2 \mid x_1) \cdots \pi(x_d \mid x_1, \ldots, x_{d-1})}{q_1(x_1)\,q_2(x_2 \mid x_1) \cdots q_d(x_d \mid x_1, \ldots, x_{d-1})} \tag{4}$$

Sequential Monte Carlo II.2: SIS motivation

Let $\mathbf{x}_t = (x_1, \ldots, x_t)$. Then (5) can be calculated recursively:

$$w_t(\mathbf{x}_t) = w_{t-1}(\mathbf{x}_{t-1}) \frac{\pi(x_t \mid \mathbf{x}_{t-1})}{q_t(x_t \mid \mathbf{x}_{t-1})}$$

so that $w_d(\mathbf{x}_d) = w(\mathbf{x})$. It is apparent that the difficulty is computing $\pi(x_t \mid \mathbf{x}_{t-1}) = \frac{\pi(\mathbf{x}_t)}{\pi(\mathbf{x}_{t-1})}$ as we do not have marginals, therefore we need to find a sequence of "auxiliary distributions" $\pi_t(\mathbf{x}_t), t \in [d]$, as an approximation to the marginal distributions $\pi(\mathbf{x}_t)$ so that $\pi_d(\mathbf{x}) = \pi(\mathbf{x})$.

Sequential Monte Carlo II.3: SIS algorithm

Algorithm 8 Sequential Importance Sampling (SIS)

Require: d; sample size N; auxiliary distributions $\{\pi_t\}_{t=0}^{d}$ (with $\pi_0(\cdot) \equiv 1$); proposal kernels $\{q_t\}_{t=1}^{d}$.

Initialization: Set $w_0 \leftarrow 1$ and let $\mathbf{x}_0$ be empty.
for $i = 1, 2, \ldots, N$ **do**
  for $t = 1, 2, \ldots, d$ **do**
    Draw $X_t \sim q_t\big(\cdot \mid \mathbf{X}_{t-1}\big)$.
    Update trajectory: $\mathbf{X}_t \leftarrow \big(\mathbf{X}_{t-1}, X_t\big)$.
    Compute the incremental weight:

$$u_t \leftarrow \frac{\pi_t\big(\mathbf{X}_t\big)}{\pi_{t-1}\big(\mathbf{X}_{t-1}\big)\, q_t\big(x_t \mid \mathbf{X}_{t-1}\big)}.$$

    Update weight: $w_t \leftarrow w_{t-1}\, u_t$.
  end for
  Output: Assign $(\mathbf{X}^{(i)}, w^{(i)}) \leftarrow (\mathbf{X}_d, w_d)$.
end for
return $\{(\mathbf{X}^{(i)}, w^{(i)}) : i = 1, \ldots, N\}$.

Sequential Monte Carlo II.4: general SIS

Finding the "auxiliary distributions" as reference for marginals is as hard as the original problem, we can expect from IS, that SIS fails for large d unless q happens to be a very good approximation of π due to the a.s. occurrence of high-variance weights (instability).
The key to bypass this instability issue is to introduce the idea of **resampling**. We will cover this idea soon after we first introduce a more general version of SIS.
Notice that marginals are not the only path we can choose to bridge between distributions, generally, our task is to approximate a bridging sequence of target distributions $(\widetilde{\pi}_t)$ by a sequence of proposal kernels (q_t). More specifically, each target distribution:

$$\tilde{\pi}_t\left(dx_{0:t}\right) = \tilde{\gamma}_t\left(x_{0:t}\right) dx_{0:t} / \tilde{Z}_t$$

is defined on the product space $\left(\mathcal{X}^{t+1}, \mathscr{X}^{t+1}\right)$, where $\tilde{\gamma}_t\left(x_{0:t}\right)$ is an unnormalized density and $\tilde{Z}_t = \int_{\mathcal{X}^{t+1}} \tilde{\gamma}_t\left(x_{0:t}\right) dx_{0:t}$ is a normalizing constant (with $\tilde{Z}_0 = 1$).

Sequential Monte Carlo II.5: general SIS

As input, it requires a sequence of proposal kernels (q_t) on $(\mathcal{X}, \mathscr{X})$. At step t, this defines the proposal distribution:

$$\tilde{q}_t\left(dx_{0:t}\right) = \tilde{\pi}_0\left(dx_0\right) \prod_{s=1}^{t} q_s\left(x_{s-1}, dx_s\right).$$

The weight function can be written as:

$$\tilde{w}_t\left(x_{0:t}\right) = \tilde{\gamma}_t\left(x_{0:t}\right) / \tilde{q}_t\left(x_{0:t}\right) = \prod_{s=1}^{t} w_s\left(x_{0:s}\right),$$

where the incremental weight is:

$$w_t\left(x_{0:t}\right) = \frac{\tilde{\gamma}_t\left(x_{0:t}\right)}{\tilde{\gamma}_{t-1}\left(x_{0:t-1}\right) q_t\left(x_{t-1}, x_t\right)} \tag{5}$$

As output, the algorithm returns weighted particles $\left(w_t^n, x_{0:t}^n\right)_{n\in[N]}$ approximating $\tilde{\pi}_t$ as $N \to \infty$, and an unbiased estimator $\tilde{Z}_t^N$ of $\tilde{Z}_t$ which is consistent as $N \to \infty$.

Sequential Monte Carlo III.1: Resampling

As we have mentioned, the key problem is high-variance weights (instability). The key is to introduce resampling at each step, that is, instead of carrying samples with very low weight, we replace low weight samples with copies of high weight samples in a statistically consistent manner.

Theorem

Given samples with importance weights $\left\{\left(\mathbf{x}^{(i)}, w^{(i)}\right) : i \in [m]\right\}$, *if we resample with replacement* $\mathbf{x}^{(*i)}$ *from* $\left\{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(m)}\right\}$ *with probabilities proportional to the importance weights, i.e.*

$$\mathbb{P}\left[\mathbf{x}^{(*i)} = \mathbf{x}^{(k)} \mid \left\{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(m)}\right\}\right] = \frac{w^{(k)}}{\sum_j w^{(j)}}$$

then the distribution of $\left\{\mathbf{x}^{(*1)}, \ldots, \mathbf{x}^{(*m)}\right\}$ *is approximately the target distribution when* m *is large.*

Sequential Monte Carlo III.2: Resampling

Proof. Let $p_*(\cdot)$ be pdf of $\mathbf{x}^{(*i)}$. For $\mathbf{x} \in \left\{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(m)}\right\} \equiv \mathbf{X}$, write $\mathbf{x}_o = \mathbf{X} \backslash \{\mathbf{x}\}$. Then by the resampling procedure,

$$\begin{aligned} p_*(\mathbf{x}) &= \binom{m}{1} q(\mathbf{x}) \int q(\mathbf{x}_o) \frac{w(\mathbf{x})}{\sum_j w^{(j)}} d\mathbf{x}_o \\ &= q(\mathbf{x}) w(\mathbf{x}) \int \frac{q(\mathbf{x}_o)}{\sum_j w^{(j)}/m} d\mathbf{x}_o \to \frac{1}{Z_\pi} \pi(\mathbf{x}) \end{aligned}$$

since $\sum_j w^{(j)}/m \xrightarrow{\text{a.s.}} Z_\pi = \int \pi(\mathbf{y}) d\mathbf{y} (= 1$ if π is normalized).

□

Several remarks to make:

- With more work, and a few more assumptions, we could show quantitative bounds for the error in SIS with (multinomial/bernoulli/systematic) resampling that is independent of d, and depends on large sample size N, something that would not typically be possible for direct IS.
- See [GSS93], [GCW17], [WLH16] for further discussions.

Sequential Monte Carlo IV.1: SMC algorithm

Algorithm 9 General Sequential Monte Carlo Methods (SIS+resampling) [DHJa22]

Require: Sequence of distributions $(\tilde{\pi}_t)$, proposal Markov kernels (q_t), resampling distribution $r\left(\cdot \mid w^{1:N}\right)$ on $[N]^N$ where $w^{1:N}$ is an N-vector of probabilities.

1: **Initialization:**

- Sample particle x_0^n from $\pi_0(\cdot)$ for $n \in [N]$ independently.
- Set $w_0^n = N^{-1}$ for $n \in [N]$.

2: **for** $t = 1, 2, \ldots, T$ **do**

3: **(a)** Sample ancestor indices $\left(a_{t-1}^n\right)_{n\in[N]}$ from $r\left(\cdot \mid w_{t-1}^{1:N}\right)$, and define $\check{x}_{0:t-1}^n = x_{0:t-1}^{a_{t-1}^n}$ for $n \in [N]$.

4: **(b)** Sample particle $x_t^n \sim q_t\left(\check{x}_{0:t-1}^n, \cdot\right)$ and set $x_{0:t}^n = \left(\check{x}_{0:t-1}^n, x_t^n\right)$ for $n \in [N]$.

5: **(c)** Compute weights $w_t\left(x_{0:t}^n\right)$ for $n \in [N]$, and set $w_t^n \propto w_t\left(x_{0:t}^n\right)$ such that $\sum_{n\in[N]} w_t^n = 1$.

6: **end for**

7: **Output:** Weighted particles $\left(w_t^n, x_t^n\right)_{n\in[N]}$ approximating π_t as $\pi_t^N(f) = \sum_{n\in[N]} w_t^n f\left(x_t^n\right)$, and estimator $Z_t^N = \prod_{s=1}^t N^{-1} \sum_{n\in[N]} w_s\left(x_{s-1}^n, x_s^n\right)$ of Z_t for $t \in [T]$.

Sequential Monte Carlo IV.2: Particle Filter

In the context of state space models, specifically hidden Markov models, we consider a latent Markov chain $(x_t)_{t\geq 0}$ defined on $(\mathcal{X}, \mathscr{X})$, initialized as $x_0 \sim \pi_0$ and evolving for each time step $t \geq 1$ according to a Markov kernel f, i.e. $x_t \mid x_{t-1} \sim f(x_{t-1}, \cdot)$. We assume access to Y -valued observations $(y_t)_{t\geq 1}$ that are modeled as conditionally independent given $(x_t)_{t\geq 0}$, with observation density g on $(\mathcal{Y}, \mathscr{Y})$, i.e. $y_t \mid x_t \sim g(x_t, \cdot)$. Given observations collected up to time t, sequential state inference is based on the posterior distribution

$$p(dx_{0:t} \mid y_{1:t}) = \frac{p(dx_{0:t})\, p(y_{1:t} \mid x_{0:t})}{p(y_{1:t})} \tag{6}$$

where the joint distribution of the states is $p(dx_{0:t}) = \pi_0(dx_0) \prod_{s=1}^{t} f(x_{s-1}, dx_s)$ and the conditional likelihood of the observations is $p(y_{1:t} \mid x_{0:t}) = \prod_{s=1}^{t} g(x_s, y_s)$. We will also be interested in the marginal likelihood $p(y_{1:t}) = \int_{\mathcal{X}^{t+1}} p(dx_{0:t}, y_{1:t})$ when there are unknown parameters in the model to be inferred. From (6), we can derive other quantities of interest such as the filtering distribution $p(dx_t \mid y_{1:t})$, defined as the last marginal of $p(dx_{0:t} \mid y_{1:t})$, and the state predictive distribution $p(dx_{t+1} \mid y_{1:t}) = \int_{\mathcal{X}} f(x_t, dx_{t+1})\, p(dx_t \mid y_{1:t})$.

Sequential Monte Carlo IV.3: Particle Filter

Particle filters can be understood as specific cases of SMC methods to sequentially approximate the posterior distribution $\tilde{\pi}_t\left(dx_{0:t}\right) = p\left(dx_{0:t} \mid y_{1:t}\right)$ (with $\tilde{\pi}_0 = \pi_0$) and the marginal likelihood $\tilde{Z}_t = p\left(y_{1:t}\right)$. In this setting, the incremental weight function in (5) reduces to

$$w_t\left(x_{t-1}, x_t\right) = \frac{f\left(x_{t-1}, x_t\right) g\left(x_t, y_t\right)}{q_t\left(x_{t-1}, x_t\right)}$$

Different choices of proposal kernels (q_t) give rise to distinct SMC methods. For example, the **bootstrap particle filter** of Gordon et al. [GSS93] corresponds to general SMC methods with $q_t\left(x_{t-1}, dx_t\right) = f\left(x_{t-1}, dx_t\right)$ and $w_t\left(x_t\right) = g\left(x_t, y_t\right)$ for all t.

Sequential Monte Carlo IV.4: SMC Sampler motivation

SMC Samplers as a framework introduced by Moral et al. in 2006 [DMDJ06] is another instance of general SMC methods, but it obtains huge flexibility and provides great framework for sampling. Given a sequence of target distributions (π_t) and backward Markov kernels (L_t) on $(\mathrm{X}, \mathscr{X})$, the target distribution in is

$$\tilde{\pi}_t\left(dx_{0:t}\right) = \pi_t\left(dx_t\right) \prod_{s=1}^{t} L_{s-1}\left(x_s, dx_{s-1}\right), \tag{7}$$

with $\tilde{\pi}_0 = \pi_0$. Note that (7) has π_t as the marginal distribution on x_t and the normalizing constant is $\tilde{Z}_t = Z_t$. In this case, the proposal kernels (q_t) correspond to the forward kernels (M_t) defined on $(\mathrm{X}, \mathscr{X})$ and the incremental weight function (5) reduces to the weight function

$$w_t\left(x_{t-1}, x_t\right) = \frac{\gamma_t\left(x_t\right) L_{t-1}\left(x_t, x_{t-1}\right)}{\gamma_{t-1}\left(x_{t-1}\right) M_t\left(x_{t-1}, x_t\right)}.$$

Sequential Monte Carlo IV.5: SMC Sampler algorithm

Algorithm 10 Sequential Monte Carlo Sampler (SMCS) [DMDJ06]

Require: Sequence of unnormalized distributions (γ_t), forward Markov kernels (M_t), backward Markov kernels (L_t), resampling distribution $r\left(\cdot \mid w^{1:N}\right)$ on $[N]^N$ where $w^{1:N}$ is an N-vector of probabilities.

1: **Initialization:**

- Sample particle x_0^n from $\pi_0(\cdot)$ for $n \in [N]$ independently.
- Set $w_0^n = N^{-1}$ for $n \in [N]$.

2: **for** $t = 1, 2, \ldots, T$ **do**

3: **(a)** Sample ancestor indices $\left(a_{t-1}^n\right)_{n\in[N]}$ from $r\left(\cdot \mid w_{t-1}^{1:N}\right)$, and define $\check{x}_{t-1}^n = x_{t-1}^{a_{t-1}^n}$ for $n \in [N]$.

4: **(b)** Sample particle $x_t^n \sim M_t\left(\check{x}_{t-1}^n, \cdot\right)$ for $n \in [N]$.

5: **(c)** Compute weights $w_t\left(\check{x}_{t-1}^n, x_t^n\right) = \frac{\gamma_t(x_t^n) L_{t-1}\left(x_t^n, \check{x}_{t-1}^n\right)}{\gamma_{t-1}\left(\check{x}_{t-1}^n\right) M_t\left(\check{x}_{t-1}^n, x_t^n\right)}$ for $n \in [N]$, and set $w_t^n \propto w_t\left(\check{x}_{t-1}^n, x_t^n\right)$ such that $\sum_{n\in[N]} w_t^n = 1$.

6: **end for**

7: **Output:** Weighted particles $\left(w_t^n, x_t^n\right)_{n\in[N]}$ approximating π_t as $\pi_t^N(f) = \sum_{n\in[N]} w_t^n f\left(x_t^n\right)$, and estimator $Z_t^N = \prod_{s=1}^t N^{-1} \sum_{n\in[N]} w_s\left(x_{s-1}^n, x_s^n\right)$ of Z_t for $t \in [T]$.

Sequential Monte Carlo IV.6: SMC Sampler

Several remarks to make:

- Adding a few more weak assumptions and with hard work, one can show that SMCS has the propagated error bounded independent of the step size. So that parallelize computation across chains is achievable, in the sense of having a large number of runs and a small number of steps can give good samples and estimations. See in [DHJa22] for quantitative guarantees.
- There are batch of alternatives to multinomial/categorical distribution for resampling (e.g., Bernoulli, systematic).

Sequential Monte Carlo IV.7: SMC Sampler

SMCS is an extremely rich and flexible sampling framework, key components within this framework are:

- Paths of Distributions
  - Geometric paths (annealed importance sampling [Nea01])
  - Gaussian convolutions (forward and backward SDE processes in score-based generative models [SSDK$^+$20])
  - See other paths like path of partial posteriors, path of truncated distributions, etc. in [DHJa22].
- Forward and Backward Markov Kernels
  - Exact MCMC moves, e.g. design M_t as π_t-invariant kernel, or design L_{t-1} to be the time reversal of M_t as $L_{t-1}(x_t, x_{t-1}) = \pi_t(x_{t-1}) M_t(x_{t-1}, x_t) / \pi_t(x_t)$, suggested by Jarzynski and Neal [Jar97][Nea01].
- Parallelization
  - See detailed guarantees and principled guides in [DHJa22].
- Parameters
  - step sizes, preconditioning matrices, inverse temperatures in geometric annealing path, etc.

Something we can talk about later

- Song et al.'s paper on "Score-based generative modeling through stochastic differential equations." [SSDK+20] with Song's blog `https://yang-song.net/blog/2021/score/`, to understand how sampling can be done if we have correct data at first but do not access the distribution (learning task), and how two regimes can be combined.
- Preliminaries of Score-based diffusion models: "variance-preserving" diffusion scheme with OU semigroup and "variance-exploding" scheme with heat semigroup, Fokker-Plank equation for the diffusion process, learning the score $\nabla \log \pi_t(x_t)$ is equivalent to learning the denoising oracle $\mathbb{E}[X_0 \mid X_t = x]$ by Tweedie's formula, Log-Sobolev Inequality for convergence rate guarantee. See all for a short review in [BH24].

Outline


Jason M. Altschuler and Sinho Chewi.
Faster high-accuracy log-concave sampling via algorithmic warm starts .
In *2023 IEEE 64th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 2169–2176, Los Alamitos, CA, USA, November 2023. IEEE Computer Society.


Joan Bruna and Jiequn Han.
Provable posterior sampling with denoising oracles via tilted transport.
In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang, editors, *Advances in Neural Information Processing Systems*, volume 37, pages 82863–82894. Curran Associates, Inc., 2024.


Tianqi Chen, Emily Fox, and Carlos Guestrin.
Stochastic gradient hamiltonian monte carlo.
In Eric P. Xing and Tony Jebara, editors, *Proceedings of the 31st International Conference on Machine Learning*, volume 32

of *Proceedings of Machine Learning Research*, pages 1683–1691, Bejing, China, 22–24 Jun 2014. PMLR.


Peter Deuflhard and Andreas Hohmann.
Numerical Analysis in Modern Scientific Computing: An Introduction.
Springer-Verlag, Berlin, Heidelberg, 2 edition, 2003.


Chenguang Dai, Jeremy Heng, Pierre E. Jacob, and Nick Whiteley and.
An invitation to sequential monte carlo samplers.
Journal of the American Statistical Association, 117(539):1587–1600, 2022.


Yilun Du and Igor Mordatch.
Implicit generation and generalization in energy-based models.
ArXiv, abs/1903.08689, 2019.


Pierre Del Moral, Arnaud Doucet, and Ajay Jasra.
Sequential monte carlo samplers.
Journal of the Royal Statistical Society Series B: Statistical Methodology, 68(3):411–436, May 2006.

_eprint: https://academic.oup.com/jrsssb/article-pdf/68/3/411/49795343/jrsssb_68_3_411.pdf.


Mathieu Gerber, Nicolas Chopin, and Nick Whiteley.
Negative association, ordering and convergence of resampling methods.
Working Papers 2017-36, Center for Research in Economics and Statistics, July 2017.


N.J. Gordon, D.J. Salmond, and A.F.M. Smith.
Novel approach to nonlinear/non-gaussian bayesian state estimation.
IEE Proceedings F (Radar and Signal Processing), 140:107–113, 1993.


C. Jarzynski.
Nonequilibrium equality for free energy differences.
Phys. Rev. Lett., 78:2690–2693, Apr 1997.


Jun S. Liu.
Monte Carlo Strategies in Scientific Computing.
Springer Publishing Company, Incorporated, 2008.


Radford M. Neal.
Annealed importance sampling.
11(2):125–139, April 2001.


Erik Nijkamp, Mitch Hill, Tian Han, Song-Chun Zhu, and Ying Nian Wu.
On the anatomy of mcmc-based maximum likelihood learning of energy-based models.
In *AAAI Conference on Artificial Intelligence*, 2019.


Yang Song, Jascha Sohl-Dickstein, Diederik Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole.
Score-based generative modeling through stochastic differential equations.
11 2020.


Luke Tierney.
Markov chains for exploring posterior distributions.
The Annals of Statistics, 22(4):1701–1728, 1994.


Nick Whiteley, Anthony Lee, and Kari Heine.

On the role of interaction in sequential monte carlo algorithms.

Bernoulli, 22(1):494–529, 2016.

Max Welling and Yee Whye Teh.
Bayesian learning via stochastic gradient langevin dynamics.
In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, ICML'11, page 681–688, Madison, WI, USA, 2011. Omnipress.