Lecture 7-9: Selected Topics on Advanced MCMC Methods

STAT 6474
Instructor: Hongxiao Zhu
Outline

• Slice sampler

• Simulated/Parallel Tempering

• Sequential MC
1 Slice Sampler
Slice sampler

• Motivation

Suppose you want to sample \( \theta \) from distribution \( f(\theta) \). Suppose the black curve is the graph of \( f(\theta) \). If you were to uniformly sample \( \theta \), each value would have the same likelihood of being sampled, and your distribution would be \( f(\theta) \propto c \), for some constant \( c \) instead of some non-uniform \( f(\theta) \). Instead of the black line, your distribution would look more like the blue line.

In order to sample \( \theta \) in a manner which will retain the distribution \( f(\theta) \), some sampling techniques must be used which takes into account the varied likelihoods in \( f(\theta) \).
Slice sampler

- **Fundamental theorem of simulation**

Simulating $$\theta \sim f(\theta)$$

is equivalent to simulating

$$(\theta, u) \sim \mathcal{U}\{(\theta, u) : 0 < u < f(\theta)\}.$$  

One can sample from the subgraph $$\phi(f) = \{(\theta, u) : 0 \leq u \leq f(\theta)\}.$$  

**Example 1:**
sample from Beta(2.7, 6.3) by sampling uniformly from the envelope and throw away the points above $$f(\theta)$$.

This is equivalent to sampling from a uniform distribution:

$$p(\theta, u) \propto 1_{\phi(f)}(\theta, u).$$
• It is not always easy to sample uniformly from a subgraph... (why)

• How about using a Markov chain?

➢ i.e., sample \( u \mid \theta \) first, then \( \theta \mid u \)?

1. \( (u \mid \theta) \sim \text{Unif}[0, f(\theta)]. \)
2. \( (\theta \mid u) \sim \text{Unif}_{A(u)}, \ A(u) = \{\theta : u \leq f(\theta)\}. \)

Repeat steps 1-2 \( N \) times.

A Gibbs sampling scheme with uniform full conditionals \( p(\theta, u) \propto 1_{\phi(f)}(\theta, u). \)
Example 2. Using simple slice sampler to sample from $f(\theta) = \frac{1}{2} e^{-\sqrt{\theta}}$.

See board derivation.
The slice sampler remains valid if one replaces $f(\theta)$ with an unnormalized version $f_1(\theta)$:

$$f(\theta) \propto f_1(\theta) \quad (why?)$$

In practice, sampling from a horizontal slice of a multimodal distribution may be difficult. There is a tension between obtaining a large sampling region and thereby making possible large moves in the distribution space, and obtaining a simpler sampling region to increase efficiency.
One solution: the “stepping-out” procedure

First, a width $w$ is used to define the area containing the given value. Each endpoint of this area is tested to see if it lies outside the given slice. If not, the region is extended by $w$ until the both endpoints lie outside the slice.

A candidate sample is selected uniformly from within this region. If the candidate sample lies inside of the slice, then it is accepted as the new sample. If it lies outside of the slice, the candidate point becomes the new boundary for the region. A new candidate sample is taken uniformly.

The process repeats until the candidate sample is within the slice.

Another solution: a more general slice sampler

• If one can factorize

\[ f(\theta) \propto \prod_{i=1}^{k} f_i(\theta) \]

where \( f_i \)'s are positive functions, but not necessarily densities (e.g. \( f_i \) could be the likelihood of each sample in a Bayesian framework with flat prior.)

1. \( (u_1 \mid \theta) \sim \text{Unif}([0, f_1(\theta)]) \)
   
    :  

k. \( (u_k \mid \theta) \sim \text{Unif}([0, f_k(\theta)]) \)

k+1. \( (\theta \mid u_1, \ldots, u_k) \sim \text{Unif}_{A(u_1,\ldots,u_k)}(\theta), \)

\[ A(u_1, \ldots, u_k) = \{ \theta : f_i(\theta) \geq u_i, i = 1, \ldots, k \}. \]
Example 3. Using general slice sampler to generate samples from

\[ f(\theta) = (1 + \sin^2(3\theta))(1 + \cos^4(5\theta)) \exp\{-\theta^2/2\}. \]

See board demonstration.
Multivariate Case

• Treating each variable independently, sampling each variable in turn repeatedly.
  Using the slice sampling within a Gibbs.

• Hyper-rectangle slice sampling.
  In the stepping out procedure, using a hyperrectangle to substitute the w region.
II Simulated/Parallel Tempering
Simulated Tempering vs. Parallel Tempering

• What is tempering? Why?

MH achieves bad mixing in situations like:

It is easier to move from one mode to another if

\[ f_T(\theta|X) \propto \pi(\theta|X)^{1/T} \]

when \( T > 1 \).

\( T \) is called temperature.
Simulated Tempering

• Goal: let a MCMC scheme move more freely in the state space.
• Marinari and Parisi (1992), Geyer and Thompson (1995)

• Idea: construct a family of distributions

\[ \Pi = \{ \pi_i(X), i \in I \}, \quad \pi_i(X) \propto \pi^{\frac{1}{T_i}}(X). \]

by varying only the temperature \( T_1 > T_2 > \ldots > T_I \).

• A new target distribution \( \pi(X, i) \propto c_i \pi^{\frac{1}{T_i}}(X) \) is defined on the augmented space \( (X, i) \in \mathcal{X} \times I \).
Simulated Tempering – cont’d

• A MCMC sampler is used to sample from $\pi(X, i)$.

1. initialize $i = i$ and $X \in \mathcal{X}$.

2. with probability $\alpha_0$, stay with the current temperature $i = i$, and draw $X$ from a MCMC transition $P_i(X, \tilde{X})$ that leaves $\pi_i$ invariant.

3. with probability $1 - \alpha_0$, propose a level transition $i \rightarrow i'$, from a transition function $\alpha(i, i')(\text{usually a nearest-neighbor simple random walk with reflecting boundary})$, and let $i = i'$ with probability

$$\min \left\{ 1, \frac{c_{i'} \pi_{i'}(X) \alpha(i', i)}{c_i \pi_i(X) \alpha(i, i')} \right\},$$

otherwise let $i = i$.

The normalizing constants $\{c_i\}$ are unknown and need to be estimated prior the simulation.
Methods to estimate $c_{i'}$ and $c_{i}$:

1. stochastic approximation.

2. reverse logistic regression

3. trial and error: starts with a few hot distributions, running and tuning the normalizing constants of these distributions such that each distribution is visited with an approximately equal frequency; and then a few more distributions are added to the distribution sequence, and running and tuning the normalizing constants of all distributions in the current distribution sequence such that each distribution is visited with an approximately equal frequency. This process is repeated till all distributions have been added to the sequence.

Reference:
About $c_i$ and $c_i'$: Reference: section 10.3 Liu’s book.

Here, the $c_i$ are constants that can be controlled by the user and they should be tuned so that each tempered distribution in the system should have a roughly equal chance to be visited. Ideally, the $c_i$ should be proportional to the reciprocal of the $i$th partition function, $Z_i = \int \exp\{-h(x)/T_i\}$. But in practice, one needs to tune these parameters via some pilot studies. An interesting tuning procedure called the reverse logistic regression is described in Geyer and Thompson (1995).

Here $h(x) = -\log \pi(X)$, so $Z_i = \int \pi_i(x)$. 
Parallel Tempering


- **Idea:** Instead of augmenting $\mathcal{X}$ to $\mathcal{X} \times I$, Geyer (1991) suggested directly dealing with the product space

  $\mathcal{X}_1 \times \ldots \times \mathcal{X}_I$,

  where $\mathcal{X}_i$ are identical copies of $\mathcal{X}$.

Suppose $(x_1, \ldots, x_I) \in \mathcal{X}_1 \times \cdots \times \mathcal{X}_I$, and for the family of distributions $\Pi = \{\pi_i, i = 1, \ldots, I\}$, define a joint probability distribution as

$$\pi(x_1, \ldots, x_I) = \prod_{i \in I} \pi_i(x_i).$$
Parallel Tempering – Cont’d

- Let the current state be \((x_1^{(t)}, \ldots, x_I^{(t)})\); draw \(u \sim \text{Uniform}[0,1]\).

- If \(u \leq \alpha_0\), we conduct the parallel step; that is, we update every \(x_i^{(t)}\) to \(x_i^{(t+1)}\) via their respective MCMC scheme.

- If \(u > \alpha_0\), we conduct the swapping step; that is, we randomly choose a neighboring pair, say \(i\) and \(i+1\), and propose “swapping” \(x_i^{(t)}\) and \(x_{i+1}^{(t)}\). Accept the swap with probability

\[
\min \left\{ 1, \frac{\pi_i(x_{i+1}^{(t)})\pi_{i+1}(x_i^{(t)})}{\pi_i(x_i^{(t)})\pi_{i+1}(x_{i+1}^{(t)})} \right\}.
\]
Example 4

Using parallel tempering to sample from the following bivariate density

\[ f(x_1, x_2) = w_1 N(\mu_1, \Sigma_1) + (1 - w_1) N(\mu_2, \Sigma_2), \]

where \( \mu_1 = (0, 0)^T \), \( \mu_2 = (10, 0)^T \) and \( \sigma_1 = 0.8, \sigma_2 = 0.3 \), \( w_1 = 0.5 \)

\[ \Sigma_1 = \sigma_1^2 I, \quad \Sigma_2 = \sigma_2^2 I. \]
Density plots

Contour plot of the log density of bivariate mixed normal

T1=100, x2=0.202

T2=3, x2=0.202

original, T3=1, x2=0.202

T4=0.5, x2=0.202
Histogram at the 4 temperatures sliced at the samples with $|x_2|<0.1$
Scatter plot of posterior samples

Scatter plot of posterior samples at T=100

Scatter plot of posterior samples at T=3

Scatter plot of posterior samples at T=1

Scatter plot of posterior samples at T=0.5
III Sequential Monte Carlo
What can Sequential Monte Carlo do?

• Sequential Monte Carlo (also called Particle filters by engineers) are a set of on-line posterior density estimation algorithms that estimate the posterior density of the state-space by directly implementing the Bayesian recursion equations.

• It is applied to situations when observations arrive sequentially in time and one is interested in performing inference on-line.

• State estimation and Tracking

http://www.youtube.com/watch?v=O1FZyWz_yj4
State-space models
(also called Hidden Markov Model especially when state space is discrete)

- \( \{X_k\}_{k \geq 1} \) hidden \( \mathcal{X} \)-valued Markov process with
  \[
  X_1 \sim \mu(x_1) \quad \text{and} \quad X_k | (X_{k-1} = x_{k-1}) \sim f(x_k | x_{k-1}).
  \]

- \( \{Y_k\}_{k \geq 1} \) observed \( \mathcal{Y} \)-valued process with observations conditionally independent given \( \{X_k\}_{k \geq 1} \) with
  \[
  Y_k | (X_k = x_k) \sim g(y_k | x_k).
  \]

- **Main Objective:** Estimate \( \{X_k\}_{k \geq 1} \) given \( \{Y_k\}_{k \geq 1} \) online/offline.
A graphical representation of the state space model
Inference in state-space models

- Given observations $y_{1:n} := (y_1, y_2, \ldots, y_n)$, inference about $X_{1:n} := (X_1, \ldots, X_n)$ relies on the posterior

\[
p (x_{1:n} \mid y_{1:n}) = \frac{p (x_{1:n}, y_{1:n})}{p (y_{1:n})}
\]

where

\[
p (x_{1:n}, y_{1:n}) = \mu (x_1) \prod_{k=2}^{n} f (x_k \mid x_{k-1}) \prod_{k=1}^{n} g (y_k \mid x_k),
\]

\[
p (y_{1:n}) = \int \cdots \int p (x_{1:n}, y_{1:n}) \, dx_{1:n}
\]
Inference in state-space models – cont’d

- We want to compute $p(x_{1:n} | y_{1:n})$ and $p(y_{1:n})$ sequentially in time $n$.

- For non-linear non-Gaussian models, numerical approximations are required.
Inference in state-space models – cont’d

- Assume you can generate $X_{1:n}^{(i)} \sim p\left(x_{1:n} \mid y_{1:n}\right)$ where $i = 1, \ldots, N$ then MC approximation is

$$\hat{p}\left(x_{1:n} \mid y_{1:n}\right) = \frac{1}{N} \sum_{i=1}^{N} \delta_{X_{1:n}^{(i)}}(x_{1:n})$$

- Integration is straightforward

$$\int \phi_n\left(x_{1:n}\right) \hat{p}\left(x_{1:n} \mid y_{1:n}\right) dx_{1:n} = \frac{1}{N} \sum_{i=1}^{N} \phi_n\left(X_{1:n}^{(i)}\right).$$

- Marginalisation is straightforward

$$\hat{p}\left(x_{k} \mid y_{1:n}\right) = \int \hat{p}\left(x_{1:n} \mid y_{1:n}\right) dx_{1:k-1} dx_{k+1:n} = \frac{1}{N} \sum_{i=1}^{N} \delta_{X_{k}^{(i)}}(x_{k})$$
Basics of Sequential Monte Carlo Methods

- **Problem**: Sampling from $p(x_{1:n} | y_{1:n})$ is impossible in general cases.

- **Divide and conquer strategy**: Break the problem of sampling from $p(x_{1:n} | y_{1:n})$ into a collection of simpler subproblems. First approximate $p(x_1 | y_1)$ at time 1, then $p(x_{1:2} | y_{1:2})$ at time 2 and so on.

- Each target distribution is approximated by a cloud of random samples termed *particles* evolving according to *importance sampling* and *resampling* steps.
Importance Sampling (weighted sampling)

- Assume you have at time $n-1$
  
  $$\hat{p}(x_{1:n-1}|y_{1:n-1}) = \frac{1}{N} \sum_{i=1}^{N} \delta_{\tilde{X}_{1:n-1}^{(i)}}(x_{1:n-1}).$$

- By sampling $\tilde{X}_{n}^{(i)} \sim f(x_n|x_{n-1}^{(i)})$ and setting $\tilde{X}_{1:n}^{(i)} = (x_{1:n-1}^{(i)}, \tilde{X}_{n}^{(i)})$
  
  then

  $$\hat{p}(x_{1:n}|y_{1:n-1}) = \frac{1}{N} \sum_{i=1}^{N} \delta_{\tilde{X}_{1:n}^{(i)}}(x_{1:n}).$$

- Our target at time $n$ is

  $$p(x_{1:n}|y_{1:n}) = \frac{g(y_n|x_n)p(x_{1:n}|y_{1:n-1})}{\int g(y_n|x_n)p(x_{1:n}|y_{1:n-1}) \, dx_n \, dx_{\{1:n-1\}}}$$

  so by substituting $\hat{p}(x_{1:n}|y_{1:n-1})$ to $p(x_{1:n}|y_{1:n-1})$ we obtain

  $$\tilde{p}(x_{1:n}|y_{1:n}) = \sum_{i=1}^{N} W_n^{(i)} \delta_{\tilde{X}_{1:n}^{(i)}}(x_{1:n}), \quad W_n^{(i)} \propto g(y_n|\tilde{X}_{1:n}^{(i)})$$

Prediction of $p(x_{1:n}|y_{1:n-1})$ based on $\hat{p}(x_{1:n-1}|y_{1:n-1})$.

Weighted approximation.
Re-sampling

- We have a “weighted” approximation $\tilde{\rho}(x_{1:n} \mid y_{1:n})$ of $\rho(x_{1:n} \mid y_{1:n})$

$$\tilde{\rho}(x_{1:n} \mid y_{1:n}) = \sum_{i=1}^{N} W^{(i)}_{n} \delta_{X^{(i)}_{1:n}}(x_{1:n}).$$

- To obtain $N$ samples $X^{(i)}_{1:n}$ approximately distributed according to $\rho(x_{1:n} \mid y_{1:n})$, we just resample

$$X^{(i)}_{1:n} \sim \tilde{\rho}(x_{1:n} \mid y_{1:n})$$

to obtain

$$\hat{\rho}(x_{1:n} \mid y_{1:n}) = \frac{1}{N} \sum_{i=1}^{N} \delta_{X^{(i)}_{1:n}}(x_{1:n}).$$

- Particles with high weights are copied multiples times, particles with low weights die.
Bootstrap Filter (sequential importance resampling -- SIR)

At time $n = 1$
- Sample $\tilde{X}_1^{(i)} \sim \mu(x_1)$ then

\[
\tilde{p}(x_1|y_1) = \sum_{i=1}^{N} W_1^{(i)} \delta_{\tilde{X}_1^{(i)}}(x_1), \quad W_1^{(i)} \propto g(y_1|\tilde{X}_1^{(i)}).
\]

- Resample $X_1^{(i)} \sim \tilde{p}(x_1|y_1)$ to obtain $\hat{p}(x_1|y_1) = \frac{1}{N} \sum_{i=1}^{N} \delta_{X_1^{(i)}}(x_1)$.

At time $n \geq 2$
- Sample $\tilde{X}_n^{(i)} \sim f(x_n|X_{n-1}^{(i)})$, set $\tilde{X}_{1:n}^{(i)} = (X_{1:n-1}^{(i)}, \tilde{X}_n^{(i)})$ and

\[
\tilde{p}(x_{1:n}|y_{1:n}) = \sum_{i=1}^{N} W_n^{(i)} \delta_{\tilde{X}_{1:n}^{(i)}}(x_{1:n}), \quad W_n^{(i)} \propto g(y_n|\tilde{X}_n^{(i)}).
\]

- Resample $X_{1:n}^{(i)} \sim \tilde{p}(x_{1:n}|y_{1:n})$ to obtain

\[
\hat{p}(x_{1:n}|y_{1:n}) = \frac{1}{N} \sum_{i=1}^{N} \delta_{X_{1:n}^{(i)}}(x_{1:n}).
\]

Sequential importance sample (SIS) is the same as SIR, but without the resampling stage. What would happen?
SMC output

- At time $n$, we get

$$
\hat{p}(x_{1:n} | y_{1:n}) = \frac{1}{N} \sum_{i=1}^{N} \delta_{X_{1:n}^{(i)}}(x_{1:n}).
$$

- The marginal likelihood estimate is given by

$$
\hat{p}(y_{1:n}) = \prod_{k=1}^{n} \hat{p}(y_k | y_{1:k-1}) = \prod_{k=1}^{n} \left( \frac{1}{N} \sum_{i=1}^{N} g(y_k | \tilde{X}_k^{(i)}) \right).
$$

- Computational complexity is $O(N)$ and memory requirements $O(nN)$. 

SMC output – cont’d

- If we are only interested in \( p(x_n|y_{1:n}) \) or \( p(s_n(x_{1:n})|y_{1:n}) \) where 
  \[ s_n(x_{1:n}) = \Psi_n(x_n, s_{n-1}(x_{1:n-1})) \] 
is fixed-dimensional then memory requirements \( \mathcal{O}(N) \).
SMC on path-space figures

Figure: $p(x_1 | y_1)$ and $\hat{E} [X_1 | y_1]$ (top) and particle approximation of $p_x(x_1 | y_1)$. 
Figure: $p (x_1 | y_1), p (x_2 | y_{1:2})$ and $\hat{E} [X_1 | y_1], \hat{E} [X_2 | y_{1:2}]$ (top) and particle approximation of $p (x_{1:2} | y_{1:2})$ (bottom)
Figure: $p(x_k | y_{1:k})$ and $\hat{E}[X_k | y_{1:k}]$ for $k = 1, 2, 3$ (top) and particle approximation of $p(x_{1:3} | y_{1:3})$ (bottom)
Figure: $p(x_k|y_{1:k})$ and $\hat{X}_k|y_{1:k}$ for $k = 1,...,10$ (top) and particle approximation of $p(x_{1:10}|y_{1:10})$ (bottom)
All particles share the same ancestors! Why?

SMC will never be able to “look back” well! The more resampling you do, the more diversity you lose.

Figure: \( p(x_k | y_{1:k}) \) and \( \hat{E}[X_k | y_{1:k}] \) for \( k = 1, ..., 24 \) (top) and particle approximation of \( p(x_{1:24} | y_{1:24}) \) (bottom)
Indeed, if you fix $k$ and look at the SMC approximation of $p(x_{1:k}|y_{1:n})$, if you increase $n$ large enough, then the marginal distribution $p(x_j|y_{1:n}), j = 1, \ldots, k$ will be approximated by one single data mass! Too Bad!

This is due the fact that you have been using resampling again and again.

Therefore, SMC (particle filter) will be good at approximating marginal distribution $p(x_n|y_{1:n})$, but it is not going to perform well on the joint distribution $p(x_{1:n}|y_{1:n})$. 
Illustration of the Degeneracy problem

- **Degeneracy problem.** For any $N$ and any $k$, there exists $n(k, N)$ such that for any $n \geq n(k, N)$

\[ \hat{p}(x_{1:k} | y_{1:n}) = \delta_{x_{1:k}}^n(x_{1:k}) . \]

\[ \hat{p}(x_{1:n} | y_{1:n}) \] is an unreliable approximation of $p(x_{1:n} | y_{1:n})$ as $n \nearrow$.

**Figure:** Exact calculation of $\frac{1}{n} \mathbb{E} [ \sum_{k=1}^{n} X_k | y_{1:n} ]$ via Kalman (blue) vs SMC estimate (red) for $N = 1000$. As $n$ increases, the SMC estimate deteriorates.

To avoid these problems, new methods such as Smoothing approximation were proposed. Details skipped.
Example 5 : Linear Gaussian State-Space model

\[ x_1 \sim N(0, 1) \]
\[ x_n = \phi x_{n-1} + v_n, \quad v_n \sim N(0, s_v^2) \]
\[ y_n = x_n + w_n, \quad w_n \sim N(0, s_w^2) \]

We will derive the posterior distribution directly (Kalman filter solution) and apply SMC using SIR algorithm.

See board derivation and R code.
Kalman filter model: $p(x_k | y(1:k))$

Sequential Monte Carlo using SIR algorithm

Particles for $p(x_n | y_{1:n})$
Example 6: Stochastic volatility model

\[ x_1 \sim N(0, \frac{\sigma^2}{1 - \phi^2}) \]

\[ x_n = \phi x_{n-1} + v_n, \quad v_n \sim N(0, \sigma^2), \quad |\phi| < 1 \]

\[ y_n = \exp\{\gamma + x_n\} w_n, \quad w_n \sim N(0, 1) \]

Leave as exercise.
Reference

• Video lectures of Arnaud Doucet

http://videolectures.net/nips09_doucet_freitas_smc/