Lecture Notes for STAT546: Computational Statistics

—Lecture 6: Monte Carlo

Faming Liang

Purdue University

August 28, 2024

An actively pursued research direction for alleviating the local-trap problem suffered by the Metropolis-Hastings (MH) algorithm is the population-based MCMC, where a population of Markov chains are run in parallel, each equipped with possibly different but related invariant distributions. Information exchange between different chains provides a means for the target chains to learn from past samples, and this in turn improves the convergence of the target chains.

Mathematically, the population-based MCMC may be described as follows. In order to simulate from a target distribution $f(x)$, one simulates of an augmented system with the invariant distribution

$$f(x_1, \ldots, x_N) = \prod_{i=1}^{N} f_i(x_i), \tag{1}$$

where $(x_1, \ldots, x_N) \in \mathcal{X}^N$, N is called the population size, $f(x) \equiv f_i(x)$ for at least one $i \in \{1, 2, \ldots, N\}$, and those different from $f(x)$ are called the trial distributions in terms of importance sampling. Different ways of specification of the trial distributions and updating the population of Markov chains lead to different algorithms, such as the adaptive direction sampling (Gilks *et al.*, 1994), conjugate gradient Monte Carlo (Liu, Liang and Wong, 2000), parallel tempering (Geyer, 1991; Hukushima and Nemoto, 1996), evolutionary Monte Carlo (Liang and Wong, 2000, 2001), sequential parallel tempering (Liang, 2003), and equi-energy sampler (Kou, Zhou and Wong, 2006).

Adaptive direction sampling

Adaptive direction sampling (ADS) (Gilks *et al.*, 1994) is an early population-based MCMC method, in which each distribution $f_i(x)$ is identical to the target distribution, and at each iteration, one sample is randomly selected from the current population to undergo an update along a direction toward another sample randomly selected from the remaining set of the current population. An important form of the ADS is the *snooker algorithm*.

Snooker Algorithm

1. Select one individual, say $x_c^{(t)}$, at random from the current population $\boldsymbol{x}^{(t)}$. The $x_c^{(t)}$ is called the current point.
2. Select another individual, say $x_a^{(t)}$, from the remaining set of the current population, i.e., $\{x_i^{(t)} : i \neq c\}$, and form a direction $e_t = x_c^{(t)} - x_a^{(t)}$. The individual $x_a^{(t)}$ is called the anchor point.
3. Set $y_c = x_a^{(t)} + r_t e_t$, where r_t is a scalar sampled from the density
$$f(r) \propto |r|^{d-1} f(x_a^{(t)} + r_t e_t), \tag{2}$$
where d is the dimension of x, and the factor $|r|^{d-1}$ is derived from a transformation Jacobian (Roberts and Gilks, 1994).
4. Form the new population $\boldsymbol{x}^{(t+1)}$ by replacing $x_c^{(t)}$ by y_c and leaving all other individuals unchanged (i.e., set $x_i^{(t+1)} = x_i^{(t)}$ for $i \neq c$).

To show the sampler is proper, we need to show that at the equilibrium the new sample y_c is independent of the $x_i^{(t)}$ for $i \neq a$ and is distributed as $f(x)$. This fact follows directly from the following lemma, which is a generalized version of Lemma 3.1 of Roberts and Gilks (1994) and was proved by Liu, Liang and Wong (2000).

Lemma 1

(Liu, Liang and Wong, 2000) Suppose $x \sim \pi(x)$ and y is any fixed point in a d-dimensional space. Let $e = x - y$. If r is drawn from distribution $f(r) \propto |r|^{d-1}\pi(y + re)$, then $x' = y + re$ follows the distribution $\pi(x)$. If y is generated from a distribution independent of x, then x' is independent of y.

Conjugate gradient Monte Carlo (Liu, Liang and Wong, 2000)

Let $\boldsymbol{x}^{(t)} = (x_1^{(t)}, \ldots, x_N^{(t)})$ denote the current population of samples. One iteration of the CGMC sampler consists of the following steps.

1. Select one individual, say $x_c^{(t)}$, at random from the current population $\boldsymbol{x}^{(t)}$.
2. Select another individual, say $x_a^{(t)}$, at random from the remaining set of the population, i.e. $\{x_i^{(t)} : i \neq c\}$. Starting with $x_a^{(t)}$, conduct a deterministic search, using the conjugate gradient method or the steepest descent method, to find a local mode of $f(x)$. Denote the local mode by $z_a^{(t)}$, which is called the anchor point.

3. Set $y_c = z_a^{(t)} + r_t e_t$, where $e_t = x_c^{(t)} - z_a^{(t)}$, and r_t is a scalar sampled from the density

$$f(r) \propto |r|^{d-1} f(z_a^{(t)} + r_t e_t), \tag{3}$$

where d is the dimension of x, and the factor $|r|^{d-1}$ is derived from the transformation Jacobian.

4. Form the new population $\boldsymbol{x}^{(t+1)}$ by replacing $x_c^{(t)}$ by y_c and leaving other individuals unchanged (i.e., set $x_i^{(t+1)} = x_i^{(t)}$ for $i \neq c$).

The gradient-based optimization procedure performed in step 2 can be replaced by some other optimization procedures, for example, a short run of simulated annealing (Kirkpatrick *et al.*, 1983). Since the local optimization step is usually expensive in computation, Liu, Liang and Wong (2000) proposed the multiple-try MH algorithm for the line sampling step, which enables effective use of the local modal information of the distribution and thus improve the convergence of the algorithm.

Sample MH Algorithm

(Lewandowski and Liu, 2008)
In adaptive direction sampling and conjugate gradient Monte Carlo, when updating the population, one first selects an individual from the population and then updates the selected individual using the standard Metropolis-Hastings procedure. If the candidate state is of high quality relative to the whole population, one certainly wants to keep it in the population. However, the acceptance of the candidate state depends on the quality of the individual that is selected for updating. To improve the acceptance rate of high quality candidates and to improve the set $\{x_i^{(t)} : i = 1, \ldots, N\}$ as a sample of size N from $f(x)$, Lewandowski and Liu (2008) proposed the sampling Metropolis-Hastings (SMH) algorithm.

Sample MH Algorithm

- Take one candidate draw $x_0^{(t)}$ from a proposal distribution $g(x)$ on $\mathcal{X}$, and compute the acceptance probability

$$\alpha_0^{(t)} = \frac{\sum_{i=1}^{N} \frac{g(x_i^{(t)})}{f(x_i^{(t)})}}{\sum_{i=0}^{N} \frac{g(x_i^{(t)})}{f(x_i^{(t)})} - \min_{0 \le k \le N} \frac{g(x_k^{(t)})}{f(x_k^{(t)})}}.$$

- Draw $U \sim Unif\,0, 1$, and set

$$\begin{aligned} \mathcal{S}_{t+1} &= \left\{ x_1^{(t+1)}, \ldots, x_n^{(t+1)} \right\} \\ &= \begin{cases} \mathcal{S}_t, & \text{if } U > \alpha_0^{(t)}; \\ \left\{ x_1^{(t)}, \ldots, x_{i-1}^{(t)}, x_0^{(t)}, x_{i+1}^{(t)}, \ldots, x_n^{(t)} \right\}, & \text{if } U \le \alpha_0^{(t)}, \end{cases} \end{aligned}$$

where i is chosen from $(1, ..., n)$ with the probability weights

$$\left(\frac{g(x_1^{(t)})}{f(x_1^{(t)})}, ..., \frac{g(x_n^{(t)})}{f(x_n^{(t)})} \right).$$

Thus, $\boldsymbol{x}_{t+1}$ and $\boldsymbol{x}_t$ differ by one element at most.

It is easy to see that in the case of $N = 1$, SMH reduces to the traditional MH with independence proposals. The merit of SMH is that to accept a candidate state, it compares the candidate with the whole population, instead of a single individual randomly selected from the current population. Lewandowski and Liu (2008) show that SMH will converge under mild conditions to the target distribution $\prod_{i=1}^{N} f(x_i)$ for $\{x_1, ..., x_N\}$, and can be more efficient than the traditional MH and adaptive direction sampling.

Parallel tempering(Geyer, 1991)

Parallel tempering simulates in parallel a sequence of distributions

$$f_i(x) \propto \exp(-H(x)/T_i), \quad i = 1, \ldots, n, \tag{4}$$

where T_i is the temperature associated with the distribution $f_i(x)$. The temperatures form a ladder $T_1 > T_2 > \cdots > T_{n-1} > T_n \equiv 1$, so $f_n(x) \equiv f(x)$ corresponds to the target distribution. The idea underlying this algorithm can be explained as follows: Raising temperature flattens the energy landscape of the distribution and thus eases the MH traversal of the sample space, the high density samples generated at the high temperature levels can be transmitted to the target temperature level through the exchange operations, and this in turn improves convergence of the target Markov chain.

Let $\boldsymbol{x}^{(t)} = (x_1^{(t)}, \ldots, x_N^{(t)})$ denote the current population of samples. One iteration of parallel tempering consists of the following steps.

1. *Parallel MH step*: Update each $x_i^{(t)}$ to $x_i^{(t+1)}$ using the MH algorithm.
2. *State swapping step*: Try to exchange $x_i^{(t+1)}$ with its neighbors: Set $j = i - 1$ or $i + 1$ according to probabilities $q_e(i, j)$, where $q_e(i, i+1) = q_e(i, i-1) = 0.5$ for $1 < i < N$ and $q_e(1, 2) = q_e(N, N-1) = 1$, and accept the swap with probability

$$\min\left\{1, \exp\left(\left[H(x_i^{(t+1)}) - H(x_j^{(t+1)})\right]\left[\frac{1}{T_i} - \frac{1}{T_j}\right]\right)\right\}. \quad (5)$$

Evolutionary Monte Carlo

The genetic algorithm (Holland, 1975) has been successfully applied to many hard optimization problems, such as the traveling salesman problem, protein folding, machine learning, among others. It is known that its crossover operator is the key to the power of the genetic algorithm, which makes it possible to explore a far greater range of potential solutions to a problem than conventional optimization algorithms. Motivated by the genetic algorithm, Liang and Wong (2000, 2001) proposed the evolutionary Monte Carlo algorithm (EMC), which incorporates most attractive features of the genetic algorithm into the framework of Markov chain Monte Carlo.

EMC works in a fashion similar to parallel tempering: A population of Markov chains are simulated in parallel with each chain having a different temperature. The difference between the two algorithms is that EMC includes a genetic operator, namely, the crossover operator in its simulation. The numerical results indicate that the crossover operator improves the convergence of the simulation and that EMC can outperform parallel tempering in almost all scenarios.

Suppose the target distribution of interest is written in the form

$$f(x) \propto \exp\{-H(x)\}, \quad x \in \mathcal{X} \subset \mathbb{R}^d,$$

where the dimension $d > 1$, and $H(x)$ is called the fitness function in terms of genetic algorithms. Let $\boldsymbol{x} = \{x_1, \ldots, x_N\}$ denote a population of size N with x_i from the distribution with density

$$f_i(x) \propto \exp\{-H(x)/T_i\}.$$

In terms of genetic algorithms, x_i is called a chromosome or an individual, each element of x_i is called a gene, and a realization of the element is called a genotype. As in parallel tempering, the temperatures form a decreasing ladder $T_1 > T_2 > \cdots > T_N \equiv 1$, with $f_N(x)$ being the target distribution.

Mutation

The mutation operator is defined as an additive Metropolis-Hastings move. One chromosome, say x_k, is randomly selected from the current population $\boldsymbol{x}$. A new chromosome is generated by adding a random vector e_k so that

$$y_k = x_k + e_k, \tag{6}$$

where the scale of e_k is chosen such that the operation has a moderate acceptance rate, e.g., 0.2 to 0.5, as suggested by Gelman, Roberts and Gilks (1996). The new population $\boldsymbol{y} = \{x_1, \cdots, x_{k-1}, y_k, x_{k+1}, \cdots, x_N\}$ is accepted with probability $\min(1, r_m)$, where

$$r_m = \frac{f(\boldsymbol{y})}{f(\boldsymbol{x})} \frac{T(\boldsymbol{x}|\boldsymbol{y})}{T(\boldsymbol{y}|\boldsymbol{x})} = \exp\left\{-\frac{H(y_k) - H(x_k)}{T_k}\right\} \frac{T(\boldsymbol{x}|\boldsymbol{y})}{T(\boldsymbol{y}|\boldsymbol{x})}, \tag{7}$$

and $T(\cdot|\cdot)$ denotes the transition probability between populations.

Crossover

One type of crossover operators that works for the real-coded chromosomes is the so-called "real crossover", which includes the k-point and uniform crossover operators. They are called real crossover by Wright (1991) to indicate that they are applied to real-coded chromosomes.
In addition to the real crossover, Liang and Wong (2001a) proposed the snooker crossover operator, which works as follows:

1. Randomly select one chromosome, say x_i, from the current population $\boldsymbol{x}$.
2. Select the other chromosome, say x_j, from the sub-population $\boldsymbol{x} \setminus \{x_i\}$ with a probability proportional to $\exp\{-H(x_j)/T_s\}$, where T_s is called the selection temperature.
3. Let $e = x_i - x_j$, and $y_i = x_j + re$, where $r \in (-\infty, \infty)$ is a random variable sampled from the density

$$f(r) \propto |r|^{d-1} f(x_j + r\mathbf{e}). \tag{8}$$

4. Construct a new population by replacing x_i with the "offspring" y_i, and replace $\boldsymbol{x}$ by $\boldsymbol{y}$.

Exchange

This operation is the same as that used in parallel tempering (Geyer, 1991; Hukushima and Nemoto, 1996). Given the current population $\boldsymbol{x}$ and the temperature ladder $\boldsymbol{t}$, $(\boldsymbol{x}, \boldsymbol{t}) = (x_1, T_1, \cdots, x_N, T_N)$, one tries to make an exchange between x_i and x_j without changing the t's. The new population is accepted with probability $\min(1, r_e)$,

$$r_e = \frac{f(\boldsymbol{x}')}{f(\boldsymbol{x})} \frac{T(\boldsymbol{x}|\boldsymbol{x}')}{T(\boldsymbol{x}'|\boldsymbol{x})} = \exp\left\{ (H(x_i) - H(x_j)) \left(\frac{1}{T_i} - \frac{1}{T_j} \right) \right\}, \quad (9)$$

Typically, the exchange is only performed on neighboring temperature levels.

The Algorithm

Based on the operators described above, the algorithm can be summarized as follows. Given an initial population $\boldsymbol{x} = \{x_1, \cdots, x_N\}$ and a temperature ladder $\boldsymbol{t} = \{T_1, T_2, \cdots, T_N\}$, EMC iterates between the following two steps:

1. Apply either mutation or crossover operator to the population with probability q_m and $1 - q_m$, respectively. The q_m is called the mutation rate.
2. Try to exchange x_i with x_j for N pairs (i, j) with i being sampled uniformly on $\{1, \cdots, N\}$ and $j = i \pm 1$ with probability $q_e(i, j)$, where $q_e(i, i+1) = q_e(i, i-1) = 0.5$ and $q_e(1, 2) = q_e(N, N-1) = 1$.

Consider simulating from a 2D mixture normal distribution

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma} \sum_{k=1}^{20} w_k \exp\{-\frac{1}{2\sigma^2}(x-\mu_k)'(x-\mu_k)\}, \qquad (10)$$

where $\sigma = 0.1$, $w_1 = \cdots = w_{20} = 0.05$. The mean vectors μ_1, μ_2, $\cdots$, μ_{20} (given in Table 1) are uniformly drawn from the rectangle $[0, 10] \times [0, 10]$. Among them, components 2, 4, and 15 are well separated from the others. The distance between component 4 and its nearest neighboring component is 3.15, and the distance between component 15 and its nearest neighboring component (except component 2) is 3.84, which are 31.5 and 38.4 times of the standard deviation, respectively. Mixing the components across so long distances puts a great challenge on EMC.

Table 1: Mean vectors of the 20 components of the mixture normal distribution (Liang and Wong, 2001).

k	μ_{k1}	μ_{k2}	k	μ_{k1}	μ_{k2}	k	μ_{k1}	μ_{k2}	k	μ_{k1}	μ_{k2}
1	2.18	5.76	6	3.25	3.47	11	5.41	2.65	16	4.93	1.50
2	8.67	9.59	7	1.70	0.50	12	2.70	7.88	17	1.83	0.09
3	4.24	8.48	8	4.59	5.60	13	4.98	3.70	18	2.26	0.31
4	8.41	1.68	9	6.91	5.81	14	1.14	2.39	19	5.54	6.86
5	3.93	8.82	10	6.87	5.40	15	8.33	9.50	20	1.69	8.11

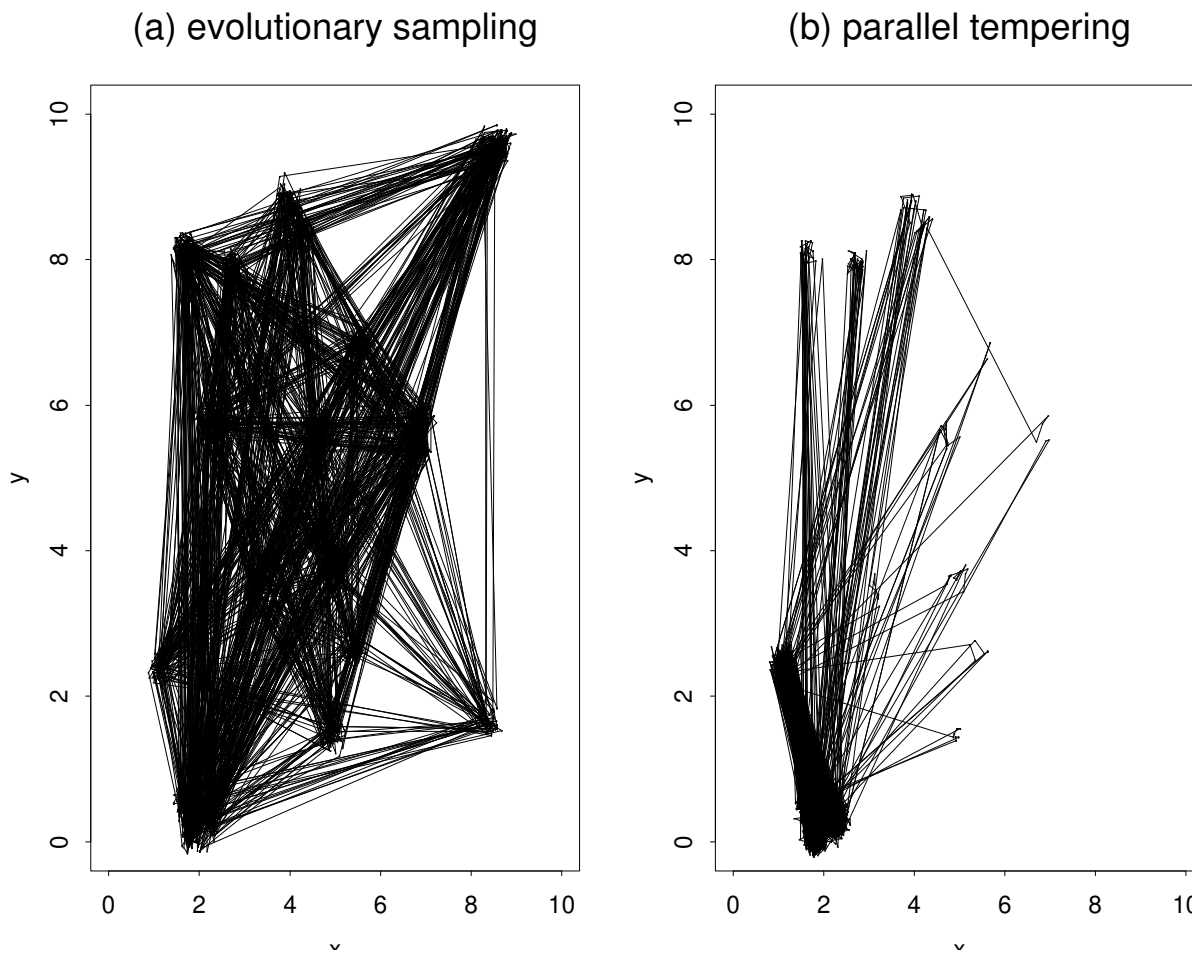

Figure 1: The sample path of the first 10000 iterations at temperature $t = 1$. (a) EMC. (b) Parallel tempering. (Liang and Wong, 2001a)

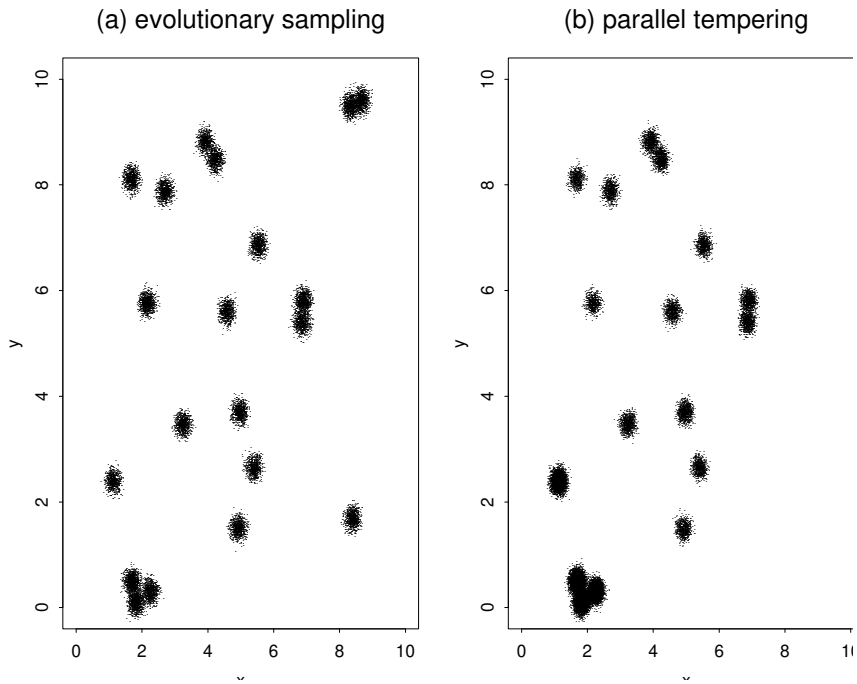

Figure 2: The plot of whole samples. (a) EMC. (b) Parallel tempering. (Liang and Wong, 2001a)

Table 2: Comparison of EMC and parallel tempering for the mixture normal example. (Liang and Wong, 2001)

parameter	true value	EMC-A		EMC-B		PT	
		est.	SD	est.	SD	est.	SD
μ_1	4.48	4.48	0.004	4.44	0.026	3.78	0.032
μ_2	4.91	4.91	0.008	4.86	0.023	4.34	0.044
Σ_{11}	5.55	5.55	0.006	5.54	0.051	3.66	0.111
Σ_{22}	9.86	9.84	0.010	9.78	0.048	8.55	0.049
Σ_{12}	2.61	2.59	0.011	2.58	0.043	1.29	0.084