ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Models, Algorithms and Data (MAD): Introduction to Computing

Spring semester 2019

Prof. Dr. Jens Honore Walther
Dr. Julija Zavadlav
ETH Zentrum, CLT
CH-8092 Zürich

Set 12

Issued: 17.05.2019; Due: 26.05.2019

Let x denote the position of a one dimensional particle that can randomly move due to thermal fluctuations under the action of a potential $V(x)$. The probability that the particle will be located at the position x, for a given temperature T, is given by the Gibbs distribution,

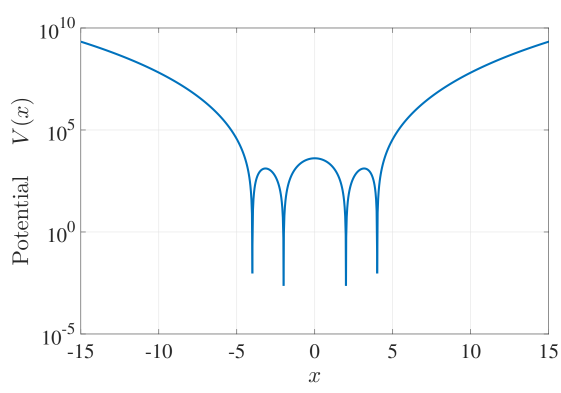
$$p(x) = \frac{1}{Z(T)} \mathrm{e}^{-\frac{1}{k_B T} V(x)}, \tag{1}$$

where k_B is the Boltzmann constant and $Z(T)$ is a normalization constant called "partition function" and is defined as $Z(T) = \int \mathrm{e}^{-\frac{1}{k_B T} V(x)} \mathrm{d}x$. Notice that for a general function V this integral does not have an analytical solution. For simplicity, we choose units such that $k_B = 1$.

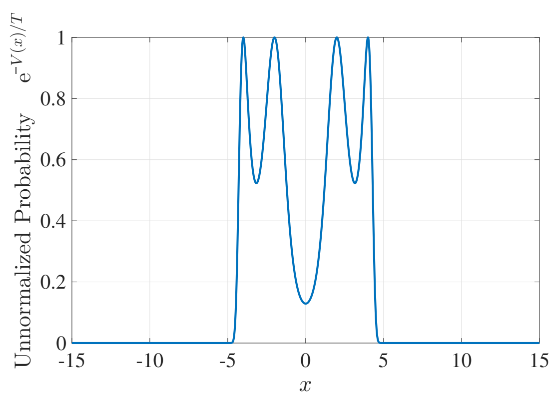
Consider the special case

$$V(x) = (x-4)^2(x-2)^2(x+2)^2(x+4)^2 . \tag{2}$$

This potential is shown in logarithmic scale is Figure 1a. Notice that the particle has high probability to be located at the positions where the potential is minimized, i.e., $x = \pm 2, \pm 4$. If $T = 0$ the particle can only be located at these points but if $T \neq 0$ the particle can be in any other place due to random thermal fluctuations. The probability of finding the particle at position x at temperature $T = 2000$ is shown in Figure 1b.



(a)



(b)

Figure 1: (a) The potential function given by (2) plotted in logarithmic scale. (b) The unnormalized Gibbs distribution (1) for temperature $T = 2000$ and potential given in (2).

The goal of this exercise is to obtain samples from the probability distribution (1) using the *Rejection Sampling* (RS) and the *Markov Chain Monte Carlo* (MCMC) algorithm. One important property of these algorithms is that they can work without knowledge of the normalization constant Z. Once a set of samples, $\{X^{(k)}\}_{k=1}^{M}$, is obtained from the distribution p one can approximate integrals of functions with respect to p, i.e.,

$$\int f(x)\, p(x)\, \mathrm{d}x \approx \frac{1}{M} \sum_{k=1}^{M} f(X^{(k)})\,. \tag{3}$$

For example, if $f(x) = x$ the expected value of the probability distribution can be estimated.

For the rest of the exercise, we assume that p is a distribution that is known up to a normalizing constant.

Rejection Sampling Assume that there exists a distribution $q(x)$ that is easy to be sampled with the property that

$$p(x) \leq Lq(x)\,, \tag{4}$$

for some positive constant L. Let M be the number of iterations. Then the following algorithm can provide samples X_k, from p:

1. Set $k = 0$. For $i = 1, \ldots, M$ do the following:
2. Generate a sample Y from q and a sample U from $\mathcal{U}[0, 1]$,
3. If $U \leq \frac{p(Y)}{Lq(Y)}$, accept Y, increment k and set $X_k = Y$.
4. Go to 2.

Here, $\mathcal{U}[a, b]$ is the uniform distribution in the interval $[a, b]$. At the end of the algorithm the vector X contains samples from the distribution p. The number of accepted samples is given by k and the *acceptance ratio* is given by k/M. The higher the acceptance ratio the more efficient the sampling algorithm is.

Notice that the efficiency of the algorithm depends on L; the smaller L is the higher the acceptance rate can be.

Markov Chain Monte Carlo Let $\mathcal{N}(y, \sigma^2)$ be the normal distribution with mean y and variance σ^2. Fix M. The following algorithm gives M samples from p:

1. Set $k = 0$. Initialize X_0 (e.g., $X_0 = 0$). For $i = 1, \ldots, M$ do the following:
2. Generate a sample Y from $\mathcal{N}(X_{i-1}, \sigma^2)$ and a sample U from $\mathcal{U}[0, 1]$,
3. If $U \leq \frac{p(Y)}{p(X_{i-1})}$ increment k and set $X_i = Y$,
4. Otherwise set $X_i = X_{i-1}$.
5. Go to 2.

As in the RS algorithm, the vector X contains the samples from distribution p and the *acceptance ratio* is given by k/M. The parameter σ of the normal distribution in Step 2 determines the efficiency of the algorithm. Notice that in this algorithm we obtain M samples, in contrast with the RS algorithm where we obtain k samples.

The normal distribution in Step 2 is usually referred as the *proposal distribution*. The choice of the normal distribution we made here is not unique. Other choices can be made for the *proposal distribution* leading to a family of proposal schemes under the name of Metropolis-Hastings algorithms.

Question 1: Rejection versus MCMC Sampling

In this question, we will implement the Rejection Sampling and the MCMC algorithm for the sampling of the Gibbs distribution (1) for temperature $T = 2000$. For the rejection sampling algorithm we will consider will consider $q = \mathcal{U}[-15, 15]$, where for the MCMC algorithm we take $\sigma = 4$ in the proposal distribution and $X_0 = 0$ for the starting point. As a baseline use the skeleton code in `ex12.py`.

a) Implement the potential function, the gibbs distribution and a wrapper to sample from a normal distribution with given mean and standard deviation.

b) For the rejection sampling algorithm, find the smallest value L such that inequality (4) is satisfied. Then implement the Rejection Sampling algorithm.

c) Implement the MCMC algorithm in python. Play with the parameter by choosing values $\sigma = 0.1, 1, 4, 8$ and evaluate the acceptance rate. What if you change the temperature to $T = 100$?

d) Obtain $M = 10^5$ samples by running your code. How do the obtained distributions compare to the wished one?

e) Also compute the acceptance rate for rejection sampling. How does it compare to MCMC? How would you improve the acceptance ratio for rejection sampling?