SI252 Reinforcement Learning: Homework #03

Due on April 6, 2025 at 11:59 a.m.(CST)

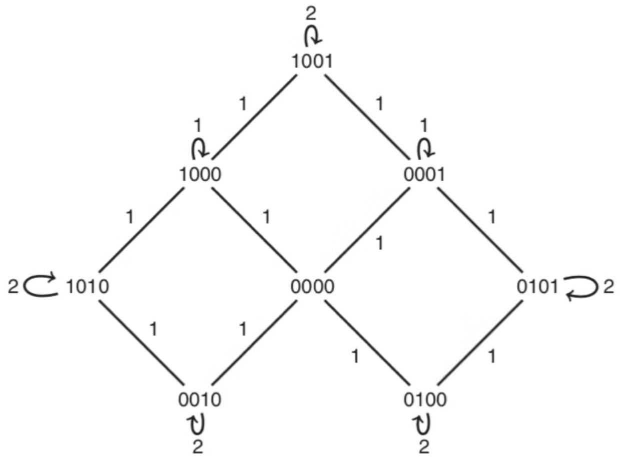
Name: **Zhou Shouchen**
Student ID: 2021533042

Problem 1

Binary Sequences with No Adjacent 1s: compute the expected number of 1s in a good sequence if all good sequences are equal likely ($m = 150$)
(a) Find the analytical solution
(b) Implement a DTMC based MCMC algorithm for approximate computing
(c) Implement a CTMC based MCMC algorithm for approximate computing
Solution
(a) This can be modeled as a dynamic programming problem. Let f_n donotes the number of good sequences whose length is n, and g_n donates the number of 1s among all good sequences whose length is n. Then for the border case, we can obviously get that $f_0 = 1, f_1 = 2, g_0 = 0, g_1 = 1$. And for $n \geq 2$, we can consider 2 cases:
1. If a good sequence with length n, ends with 0, then the $(n-1)$-th bit can be 0 or 1. And these sequences has number of f_{n-1}, contributes g_{n-1} 1s to the total number of 1s.
2. If a good sequence with length n, ends with 1, then the $(n-1)$-th bit must be 0, which is the same meaning that the $n-2$-th bit can be 0 or 1. And these sequences has number of f_{n-2}, contributes $g_{n-2} + f_{n-2}$ 1s to the total number of 1s.
So above all, we can get the recursive equation:

$$\begin{aligned} f_n &= f_{n-1} + f_{n-2} &&, n \geq 2 \\ g_n &= g_{n-1} + g_{n-2} + f_{n-2} &&, n \geq 2 \end{aligned}$$

We can compute the f_m and g_m to get the analytical solution. Since there are total f_m good sequences, and g_m total number of 1s, so the expectede number of 1s in a godd sequence is $\frac{g_m}{f_m}$.
The analytical solution of $m = 150$ is 41.6117667420032.
(b) We can design a Markov Chain. Each state X is a good sequence with length m. For each transition, we uniformly choose one of its m components, and flip that bit to generate a new state. If the new state is a good sequence, then translate to the new state, otherwise translate to the old state itself. An example of $m = 4$ is as follows:



1. Finite state: Since the number of good sequence with length m would not exceed 2^m, so the Markov Chain is finite state.
2. Irreducible: Since for each state X, it requires at most m steps to translate to the state that is all 0. So for any two state in the Markov Chain requires at most $2m$ step to translate to each other, so the Markov Chain is irreducible.
3. Aperiodic: Since for the good sequences with at least one 1, it must have a transition to itself if $m > 1$,

Problem 1 continued on next page...

so the Markov Chain is aperiodic.
So the Markov Chain is finite state, irreducible and aperiodic, which means it's ergodic, so it has a unique stationary distribution.
Since for two state i, j, if the Hamming distance between i and b is j, then $p_{i,j} = p_{j,i} = \frac{1}{m}$, and otherwise $p_{i,j} = p_{j,i} = 0$. So the transition matrix is symmetric, which means it is a double stochastic matrix. So the unique stationary distribution is the uniform distribution, which is exactly what we want.
Define $r(X)$ be the number of 1s for the state X. From the Strong Law of Large Numbers for Markov Chains, we have

$$\mathbb{E}\left[r(X)\right] = \lim_{n\to\infty} \frac{r(X_1) + \ldots + r(X_n)}{n}$$

So we can simulate by calculating the right hand size and get the expected number of 1s in a good sequence. The simulated approximated solution using DTMC based MCMC of $m = 150$ is 41.606899. Which is quite close to the analytical solution.
(c) Similarly with the construction in (b), but the time becomes continuous. Consider the state i, let $\mathcal{N}_i$ be the neighbors of state i, which means if state $j \in \mathcal{N}_i$, then the Hamming distance between state i and state j is 1, and state j is a good sequence. Then the transiton probability of the embedded chain is $p^e_{i,j} = \frac{1}{|\mathcal{N}_i|}$.
From the detailed balance equation: $\forall i, j$, we have

$$\pi_i q_{i,j} = \pi_j q_{j,i}$$

Since we want the stationary distribution to be a uniform distribution, and $q_{i,j} = p^e_{i,j} v_i$, thus we need to set the holding time of state i to be $T_i \sim \text{Expo}(|\mathcal{N}_i|)$.
To simulate the CTMC based MCMC, we can easily apply the Gillespie algorithm. Suppose the state transition occurs at time $0 = t_0 < t_1 < \ldots < t_N = T$, then we can expand the Strong Law of Large Numbers for Markov Chains to be:

$$\mathbb{E}\left[r(X)\right] = \lim_{T\to+\infty} \int_0^T \frac{r(X_t)}{T} \mathrm{d}t = \lim_{N\to\infty} \sum_{n=1}^{N} \frac{r(X_{t_n}) \cdot (t_n - t_{n-1})}{T}$$

The simulated approximated solution using CTMC based MCMC of $m = 150$ is 41.59466945403054. Which is quite close to the analytical solution. And we can see that both DTMC and CTMC based MCMC have the similar results, both quite close to the analytical solution. But CTMC cost more times compared to DTMC in the same steps. This is because CTMS need to get all the neigbors of the state, which requires additionally $O(m)$ time.

Problem 2

Power-Law Distribution: Implement a Metropolis-Hastings algorithm to simulate from the power-law distributions shown in the lecture.

Solution

The power-law distribution with constant $S = -\dfrac{3}{2}$ is defined as follows:

$$\pi_i = \frac{i^S}{\sum\limits_{k=1}^{\infty} k^S} = \frac{i^{-\frac{3}{2}}}{\sum\limits_{k=1}^{\infty} k^{-\frac{3}{2}}} \propto i^{-\frac{3}{2}}, \quad \text{for } i = 1, 2, \ldots$$

Since the state space is $\{1, 2, \ldots\}$, which are the positive integers, so the proposal Markov Chain could be constructed with a single reflecting bound:

$$\begin{tikzcd}
\mathbf{1} \arrow[r, bend left, "1"] & \mathbf{2} \arrow[l, bend left, "\frac{1}{2}"] \arrow[r, bend left, "\frac{1}{2}"] & \mathbf{3} \arrow[l, bend left, "\frac{1}{2}"] \arrow[r, bend left, "\frac{1}{2}"] & \mathbf{4} \arrow[l, bend left, "\frac{1}{2}"] \arrow[r, bend left, "\frac{1}{2}"] & \cdots \arrow[l, bend left, "\frac{1}{2}"]
\end{tikzcd}$$

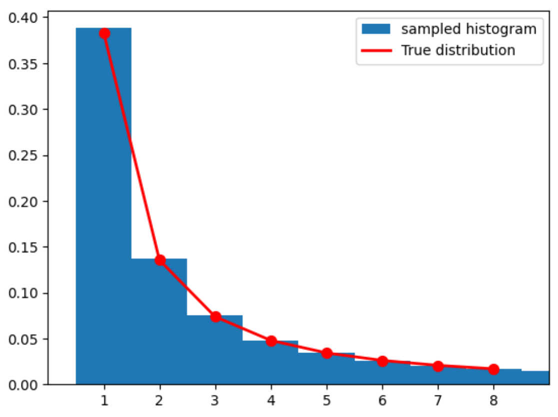
This is obviously an irreducible Markov Chain, and the transition probability is

$$p_{i,j} = \begin{cases} \frac{1}{2} & \text{if } j = i \pm 1 \\ 1 & \text{if } i = 1, j = 2 \\ 0 & \text{otherwise} \end{cases}$$

Thus we can apply the Metropolis-Hastings algorithm.
For each step, the transition from i to j has the accept rate:

$$a_{i,j} = \min\left(\frac{\pi_j p_{j,i}}{\pi_i p_{i,j}}, 1\right) = \min\left(\frac{j^{-\frac{3}{2}} p_{j,i}}{i^{-\frac{3}{2}} p_{i,j}}, 1\right) \Rightarrow \begin{cases} a_{1,2} = 2^{-\frac{5}{2}} \\ a_{i,i+1} = \left(\dfrac{i}{i+1}\right)^{\frac{3}{2}} \\ a_{i+1,i} = 1 \end{cases}$$

The number of transition is set to be 1000000 steps, and the first 10000 are used as burn in to ensure the state distribution enters the stationary distribution. And the curve for the true distribution and the sampled distribution is shown as follows:



Problem 2 continued on next page...

As for convinence to show, we rounded the numbers into 3 decimal places. The normalize factor used first 1000000 terms to add up. And the numbers of the exact probability and simulated results are shown as follows:

i	1	2	3	4	5	6	7	8	≥ 9
Exact	0.383	0.135	0.074	0.048	0.034	0.026	0.021	0.017	0.262
Simulation	0.388	0.137	0.075	0.048	0.034	0.026	0.021	0.017	0.254

And we can see that the simulated results are quite close to the exact results.

Problem 3

Knapsack Problem: $m = 50, w = 100$, each g_j is sampled from the discrete uniform distribution over the supporting set $\{3, 4, 5, 6, 7\}$, each w_j is chosen random from the discrete uniform distribution over the supporting set $\{2, 4, 6, 8\}$.
(a) Find the maximization of the total worth of the treasure by using DP method
(b) Find the maximization of the total worth of the treasure by using Metropolis-Hastings(MH) algorithm
(c) Discuss the choice of β & function of $V_1(x)$ in MCMC, verify your argument with numerical experiments
(d) Discuss the pros and cons of both MCMC(MH) and DP methods
Solution
(a) The i-th item weights w_i, and worths g_i. Define $f_{i,j}$ represent the maximum worth of the treasure that can be obtained by using the first i items and the knapsack capacity is j. Then we can get the Bellman equation:

$$f_{i,j} = \max\{f_{i-1,j}, f_{i-1,j-w_i} + g_i | \, j \geq w_i\}$$

where the boarder conditions are $f_{0,j} = 0, \forall j \in \mathbb{Z}$. And the optimal solution is $f_{m,w}$.
We can fix the sample seed, the random seed is set to be np.random.seed(0), and the optimal solution is 150.
(b) Let the state $x = (x_1, \ldots, x_m)$ be the 01 binary sequence, representing whether each item is selected or not. And the transition is a random flip. If the new state y is a valid state, i.e.

$$\sum_{i=1}^{m} y_i w_i \leq w$$

then it has equal probability to transition to all states, which means that

$$p_{x,y} = p_{y,x} = \frac{1}{m}$$

Otherwise, it will stay at the same state. It is obvious that the Markov Chain is reducible as all states can transition to the state $(0, \ldots, 0)$ in m steps.
To generate a distribution that have a bigger probability at the states with higher values, the distribution can set to be

$$\pi_x \propto e^{\beta V_1(x)}$$

Where

$$V_1(x) = \sum_{i=1}^{m} x_i g_i$$

is the value of state x, and β is a positive constant for adjusting the distribution.
So above all, we have constructed a reducible proposal Markov Chain, and a disired distribution $s(x)$. And we can calculate the acceptance rate of the transition is

$$\begin{aligned} a_{x,y} &= \min\left(\frac{\pi_y p_{y,x}}{\pi_x p_{x,y}}, 1\right) \\ &= \min\left(\frac{e^{\beta V_1(y)} \cdot \frac{1}{m}}{e^{\beta V_1(x)} \cdot \frac{1}{m}}, 1\right) \\ &= \min\left(e^{\beta[V_1(y) - V_1(x)]}, 1\right) \end{aligned}$$

So we can apply Metropolis-Hastings algorithm to generate the samples. The state with the maximum probability representing the optimal solution is the state with the maximum value of $V_1(x)$.
(c) From the analysis above, we can setting $\beta = 0.01, 0.1, 1, 10, 100$ respectively. Additionally, the β is set to be a function of t: $\beta(t) = C \cdot \log(t + 1)$. Where $C = \dfrac{1}{\log\left(\frac{T}{2}\right)}$ to ensure the in middle time, $\beta = 1$.

Problem 3 continued on next page...

We can also setting a new value function $V_2(x) = \frac{\sum_{i=1}^m x_i g_i}{\sum_{i=1}^m x_i w_i}$, which represent to the cost-effectiveness. The simulations results for different β and value functions are as follows:

	$\beta = 0.01$	$\beta = 0.1$	$\beta = 1$	$\beta = 10$	$\beta = 100$	$\beta = \log(t+1)$
$V_1^*(x)$	98	81	142	103	102	**144**
$V_2^*(x)$	99	110	109	7	7	91

We can see that V_1 value function has better performance.
As $\beta \gg 1$: Suppose that x is the strictly local optimal state, and y are the states that x can transition to. Thus it must have $V_1(y) < V_1(x)$, thus the acceptance rate from x to y is $\lim\limits_{\beta \to +\infty} e^{\beta[V_1(y)-V_1(x)]} = 0$. Thus it would stay in the local optimal state, it lacks of exploration, only exploration.
If $\beta \ll 1$: Then $\lim\limits_{\beta \to 0^+} e^{\beta[V_1(y)-V_1(x)]} = 1$, so it would always explore, and it lacks of exploitation.
And $\beta(t) = C \cdot \log(1+t)$ make the exploration-exploitation trade-off better, it explore more at the beginning, and exploite more later. So it has a better performance.
(d) 1. Metropolis-Hastings / MCMC Methods
Advantages: Suitable for large-scale and high-dimensional problems, when the state space is enormous, it can find high-value solutions through random walks. Hyperparameters (such as β) can be flexibly adjusted to balance exploration and exploitation, which helps in escaping local optima. It can accommodate complex objective functions; the design is quite flexible as long as the acceptance rate is properly constructed.
Disadvantages: The result is an approximate solution, it requires sufficiently long sampling and an appropriate burn-in period to approach the optimal solution, and the convergence speed and sample autocorrelation may be suboptimal. Selecting the hyperparameters (such as β) is challenging and may require extensive experimentations.
2. Dynamic ProgrammingMethod
Advantages: It can generate the optimal solution, which means the quality of the solution does not rely on randomness. The dynamec programming method is often efficient and straightforward to implement.
Disadvantages: The time and space complexity of the DP method depends on the knapsack capacity, which is a NP-complete problem, could not be solved in polynomial time. DP algorithms are typically applicable only to structured problems; for unstructured or approximate optimization problems, it is difficult to apply them directly.

Problem 4

Standard Normal Distribution: generate samples from the standard normal distribution
(a) Implement a Metropolis-Hastings algorithm.
(b) Implement a Hamiltonian Monte Carlo algorithm.
(c) Implement with the Box-Muller Method.
(d) Compare the above three algorithms with corresponding pros and cons.

Solution

(a) Let $X \sim \mathcal{N}(0,1)$, and $Y = |X|$. So $X \in (-\infty, +\infty)$, $Y \in (0, +\infty)$. We can get the PDF of the stationary distribution:

$$\pi_i = \frac{2}{\sqrt{2\pi}} e^{-\frac{1}{2}i^2}, i > 0$$

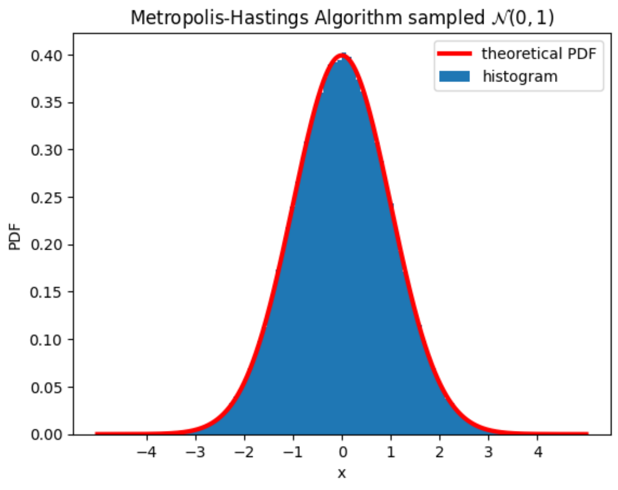
And we can choose the proposal distribution to be Expo(1), which means that the one-step transition probability density from state x to y is $f_{x,y} = f_{X_{n+1}|X_n}(y|x) = e^{-y}, y > 0$, thus we can get that the acceptance rate:

$$a_{i,j} = \min\left(\frac{\pi_j f_{j,i}}{\pi_i f_{i,j}}, 1\right) = \min\left(e^{-\frac{1}{2}j^2 + j + \frac{1}{2}i^2 - i}, 1\right)$$

Then we can do the Discrete time continuous state Metropolis-Hastings algorithm to sample the distribution of Y.

To sample on $X \sim \mathcal{N}(0,1)$, since $Y = |X|$, so we can generate $U \sim \text{Unif}(0,1)$, and let $X = \begin{cases} Y, & U \le \frac{1}{2} \\ -Y, & U > \frac{1}{2} \end{cases}$.

Thus the $\mathcal{N}(0,1)$ is sampled with Metropolis-Hastings algorithm in 1000000 samples, and the burn in is set to be 10000 samples. The result is as follows:



(b) Since π_x is a the stationary distribution $\mathcal{N}(0,1)$, and we can ignore the constant form, so the potential energy is

$$U(x) = -\log(\pi_x) = \frac{1}{2}x^2 + \frac{1}{2}\log(2\pi) \overset{\text{ignore const}}{\Rightarrow} U(x) = \frac{1}{2}x^2$$

And let the mass be 1, then the Kinetic energy is

$$V(\omega) = \frac{1}{2}\omega^2, \quad \text{where } \omega \sim \mathcal{N}(0,1)$$

Thus the Hamiltonian energy is

$$H(x,\omega) = U(x) + V(\omega) = \frac{1}{2}x^2 + \frac{1}{2}\omega^2$$

Initially, set $x_0 = 0, \omega_0 \sim \mathcal{N}(0,1)$, apply the Leapfrog method to sample:

$$\omega_{t+\frac{\delta}{2}} = \omega_t - \frac{\delta}{2} \cdot \frac{\mathrm{d}U(x_t)}{\mathrm{d}x_t} = \omega_t - \frac{\delta}{2} \cdot x_t$$

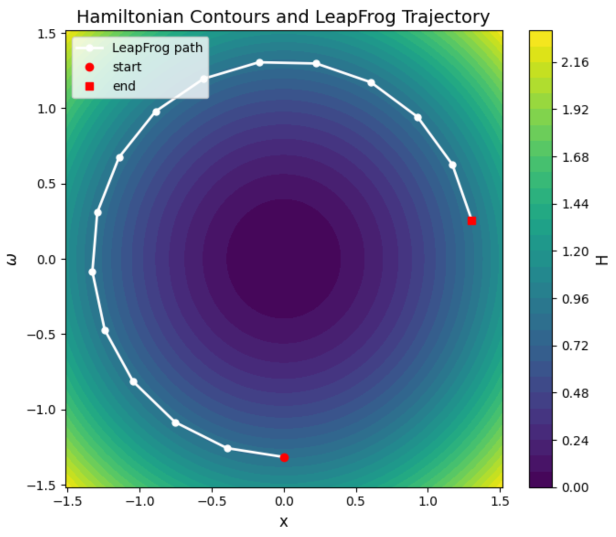
$$x_{t+\delta} = x_t + \delta \cdot \frac{\mathrm{d}V(\omega_{t+\frac{\delta}{2}})}{\mathrm{d}\omega_{t+\frac{\delta}{2}}} = x_t + \delta \cdot \omega_{t+\frac{\delta}{2}}$$

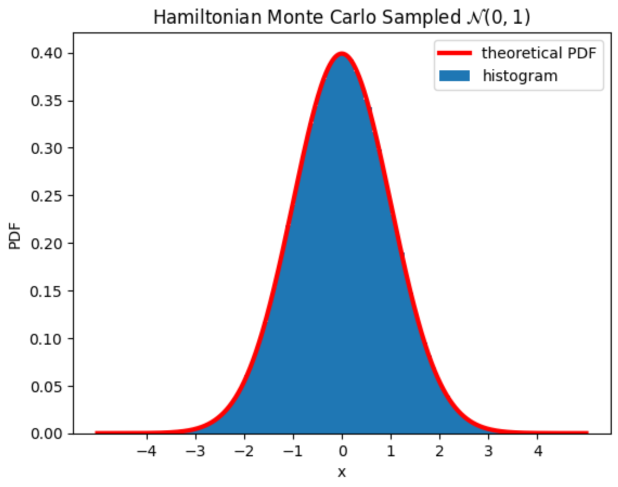
$$\omega_{t+\delta} = \omega_{t+\frac{\delta}{2}} - \frac{\delta}{2} \cdot \frac{\mathrm{d}U(x_{t+\delta})}{\mathrm{d}x_{t+\delta}} = \omega_{t+\frac{\delta}{2}} - \frac{\delta}{2} \cdot x_{t+\delta}$$

After Leapfrog L steps, we can get the state (x_L, ω_L). And set $(x_L, -\omega_L)$ to be the proposal distribution. And the accept rate for the Hamiltonian Monte Carlo algorithm is

$$a_{(x_0,\omega_0),(x_L,-\omega_L)} = \min\left(\frac{\exp\left(-H(x_L,-\omega_L)\right)}{\exp\left(-H(x_0,\omega_0)\right)}, 1\right) = \min\left(\frac{\exp\left(-U(x_L) - V(-\omega_L)\right)}{\exp\left(-U(x_0) - V(\omega_0)\right)}, 1\right)$$

We step the stepsize to be $\delta = 0.3, L = 15$, and the sample trajectory of one step's LeapFrog and sample results are as follows:





(c) Let $U \sim \text{Unif}(0, 2\pi)$, so $f_U(u) = \dfrac{1}{2\pi}, u \in [0, 2\pi]$, which is easy to sample. Let $T \sim \text{Expo}(1)$, so $f_T(t) = e^{-t}, t \in (0, +\infty)$. To sample on T, we can use the Universality of Uniform: the Exponential

Problem 4 continued on next page...

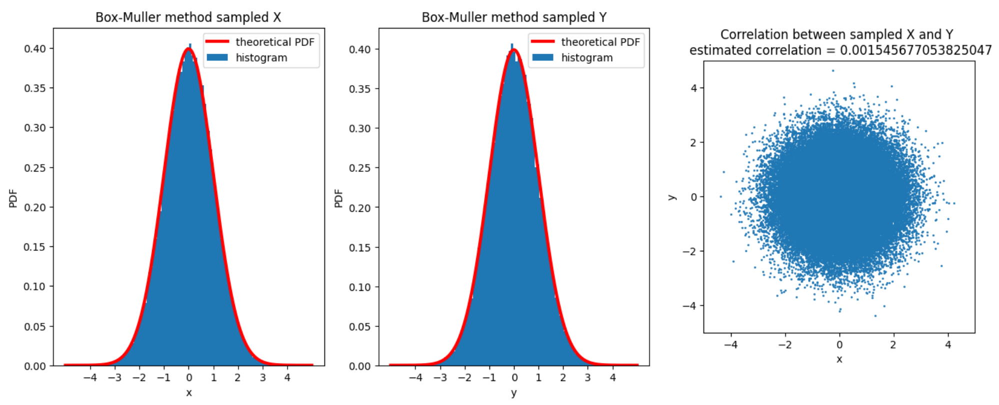
distribution has CDF

$$F_T(t) = 1 - e^{-t}, \forall t > 0$$

So the inverse funciton of its CDF is $F_T^{-1}(x) = -\ln(1-x)$. So let $U_2 \sim \text{Unif}(0,1)$, then

$$T = F^{-1}(U_2) = -\ln(1-U_2)$$

Let $X = \sqrt{2T}\cos(U), Y = \sqrt{2T}\sin(U)$, from Box-Muller method, we can get that X, Y are i.i.d. $N(0,1)$. The sampled results are as follows:



(d) 1. Metropolis-Hastings Algorithm:
Advantages: It is highly general-purpose, could applicable to a wide variety of distributions, and it does not require the normalization constant of the target distribution.
Disadvantages: Samples are often requires select a suitable distribution, and requires steps to burn up.
2. Hamiltonian Monte Carlo Algorithm:
Advantages: Due to the conservation of energy, the acceptance rate should be 1. But there exists numerical error, however quite close to 1, which means it has quite high acceptance rate. The samples are efficient, especially in high-dimensional problems.
Disadvantages: Requires computation of gradients, increasing implementation complexity. Sensitive to parameters: step size δ and number of steps for LeapFrog L.
3. Box-Muller Method:
Advantages: Direct and simple method to generate independent standard normal samples, which is easy to implement.
Disadvantages: It would sample the normal distribution only.

Problem 5

Beta Normal Distribution: generate samples from the beta distribution Beta$(5,5)$
(a) Implement a Metropolis-Hastings algorithm.
(b) Implement a Hamiltonian Monte Carlo algorithm.
(c) Implement with the Acceptance-Rejection Method.
(d) Compare the above three algorithms with corresponding pros and cons.

Solution

Let $X \sim$ Beta$(5,5)$. We can calculate the beta integral:

$$\beta(5,5) = \frac{\Gamma(5)\cdot\Gamma(5)}{\Gamma(10)} = \frac{4!\cdot 4!}{9!} = \frac{1}{630}$$

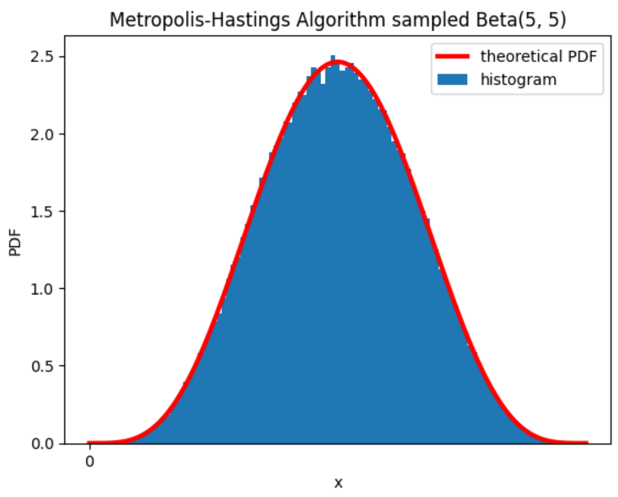
Then we can calculate the PDF of X:

$$f(x) = f_X(x) = \frac{1}{\beta(5,5)}x^{5-1}(1-x)^{5-1} = 630x^4(1-x)^4, 0 < x < 1$$

(a) Since the support of Beta distributio is $[0,1]$, so we can choose the proposal distribution to be Unif(1), which means that the one-step transition probability density from state x to y is $f_{x,y} = 1$, thus we can get that the acceptance rate:

$$a_{x,y} = \min\left(\frac{\pi_j f_{j,i}}{\pi_i f_{i,j}}, 1\right) = \min\left(\frac{j^4(1-j)^4}{i^4(1-i)^4}, 1\right)$$

Then we can do the Discrete time continuous state Metropolis-Hastings algorithm to sample the distribution Beta$(5,5)$ in 1000000 samples, and the burn in is set to be 10000 samples. The result is as follows:



(b) Since π_x is a the stationary distribution Beta$(5,5)$, and we can ignore the constant form, so the potential energy is

$$U(x) = -\log(\pi_x) = -\log(630) - 4\log(x(1-x)) \overset{\text{ignore const}}{\Rightarrow} U(x) = -4\log(x(1-x))$$

And let the mass be 1, then the Kinetic energy is

$$V(\omega) = \frac{1}{2}\omega^2, \quad \text{where } \omega \sim \mathcal{N}(0,1)$$

Thus the Hamiltonian energy is

$$H(x,\omega) = U(x) + V(\omega) = -4\log(x(1-x)) + \frac{1}{2}\omega^2$$

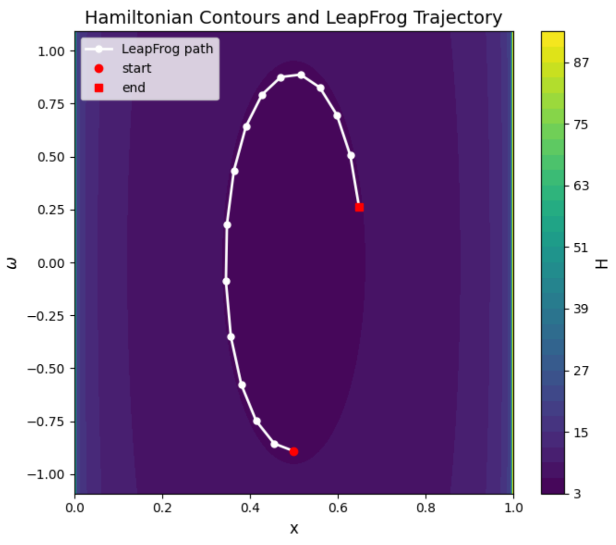
Initially, set $x_0 = 0.5, \omega_0 \sim \mathcal{N}(0,1)$, apply the Leapfrog method to sample:

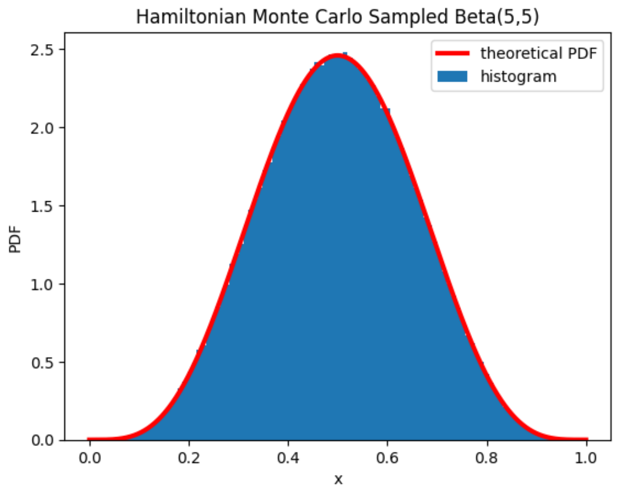
$$\omega_{t+\frac{\delta}{2}} = \omega_t - \frac{\delta}{2} \cdot \frac{\mathrm{d}U(x_t)}{\mathrm{d}x_t} = \omega_t - \frac{\delta}{2} \cdot \left(-\frac{4}{x_t} + \frac{4}{1-x_t}\right)$$
$$x_{t+\delta} = x_t + \delta \cdot \frac{\mathrm{d}V(\omega_{t+\frac{\delta}{2}})}{\mathrm{d}\omega_{t+\frac{\delta}{2}}} = x_t + \delta \cdot \omega_{t+\frac{\delta}{2}}$$
$$\omega_{t+\delta} = \omega_{t+\frac{\delta}{2}} - \frac{\delta}{2} \cdot \frac{\mathrm{d}U(x_{t+\delta})}{\mathrm{d}x_{t+\delta}} = \omega_{t+\frac{\delta}{2}} - \frac{\delta}{2} \cdot \left(-\frac{4}{x_{t+\delta}} + \frac{4}{1-x_{t+\delta}}\right)$$

Due to the range limitation of $x \in (0,1)$, we can add a reflection bound at $x = 0$ and $x = 1$, i.e. when applying Leapfrog, if $x < 0$, then set $x \leftarrow -x, \omega \leftarrow -\omega$, if $x > 1$, then set $x \leftarrow 2-x, \omega \leftarrow -\omega$. After Leapfrog L steps, we can get the state (x_L, ω_L). And set $(x_L, -\omega_L)$ to be the proposal distribution. And the accept rate for the Hamiltonian Monte Carlo algorithm is

$$a_{(x_0,\omega_0),(x_L,-\omega_L)} = \min\left(\frac{\exp\left(-H(x_L,-\omega_L)\right)}{\exp\left(-H(x_0,\omega_0)\right)}, 1\right) = \min\left(\frac{\exp\left(-U(x_L)-V(-\omega_L)\right)}{\exp\left(-U(x_0)-V(\omega_0)\right)}, 1\right)$$

We step the stepsize to be $\delta = 0.05, L = 15$, additionally, since the support is $(0,1)$, so for numerical accuracy, we clip x into the range $[10^{-10}, 1-10^{-10}]$. One trajectory and sample results are as follows:



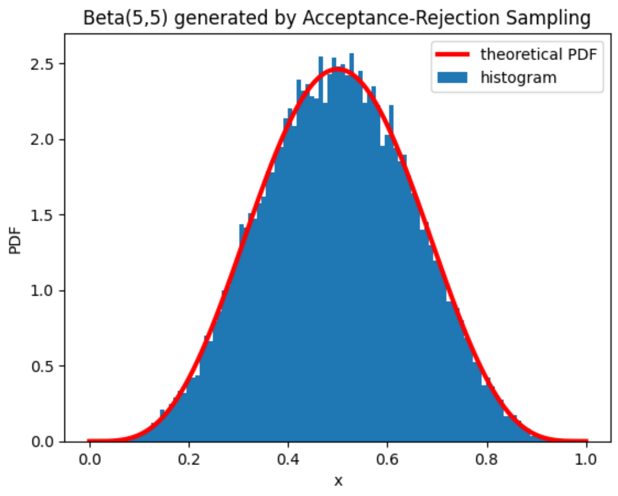


Problem 5 continued on next page...

(c) As for the acceptance-rejection algorithm, since the support of Beta distribution is $(0,1)$, so we can take $Y \sim \text{Unif}(0,1)$. And its PDF is $g(y) = f_Y(y) = 1$.
Let c donate a constant:

$$c = \sup_y \frac{f(y)}{g(y)} = \sup_y \frac{20y(1-y)^3}{1} = 630x^4(1-x)^4\Big|_{y=\frac{1}{2}} = \frac{315}{128}$$

Then we can apply the Acceptance-Rejection algorithm to generate the samples on $X \sim \text{Beta}(5,5)$:



(d) 1. Metropolis-Hastings Algorithm:
Advantages: It is highly general-purpose, could applicable to a wide variety of distributions, and it does not require the normalization constant of the target distribution.
Disadvantages: Samples are often requires select a suitable distribution, and requires steps to burn up.
2. Hamiltonian Monte Carlo Algorithm:
Advantages: Due to the conservation of energy, the acceptance rate should be 1. But there exists numerical error, however quite close to 1, which means it has quite high acceptance rate. The samples are efficient, especially in high-dimensional problems.
Disadvantages: Requires computation of gradients, increasing implementation complexity. Sensitive to parameters: step size δ and number of steps for LeapFrog L.
3. Acceptance-Rejection Method:
Advantages: It produces independent samples, and is simple to implement. If the good envelope function is suitable, it is efficient.
Disadvantages: Efficiency depends heavily on the choice of proposal distribution, for example, ours implement has a quite low acceptance rate, which is only 0.0038838.

Problem 6

Normal-Normal Conjugacy: Implement a Metropolis-Hastings algorithm to find the posterior mean and variance of θ after observing the value of $Y = y$. The parameter setting: $y = 3, \mu = 0, \sigma^2 = 1, \tau^2 = 4$, and $d = 0.1, 0.5, 1, 5, 10, 100$.

Solution

Let $Y|\Theta = \theta \sim \mathcal{N}(\theta, \sigma^2)$, where σ^2 is known but θ is unknown, and its prior is $\theta \sim \mathcal{N}(\mu, \tau^2)$. We can get the posterior distribution of θ with the Normal-Normal conjugate:

$$\Theta|Y = y \sim \mathcal{N}\left(\frac{\frac{1}{\sigma^2}}{\frac{1}{\sigma^2}+\frac{1}{\tau^2}}y + \frac{\frac{1}{\tau^2}}{\frac{1}{\sigma^2}+\frac{1}{\tau^2}}\mu, \frac{1}{\frac{1}{\sigma^2}+\frac{1}{\tau^2}}\right)$$

$$f_{\Theta|Y}(\theta|Y=y) \propto f_{Y|\Theta}(y|\theta)(y|\theta) \cdot f_{\Theta}(\theta) \propto e^{-\frac{1}{2\sigma^2}(y-\theta)^2} \cdot e^{-\frac{1}{2\tau^2}(\theta-\mu)^2}$$

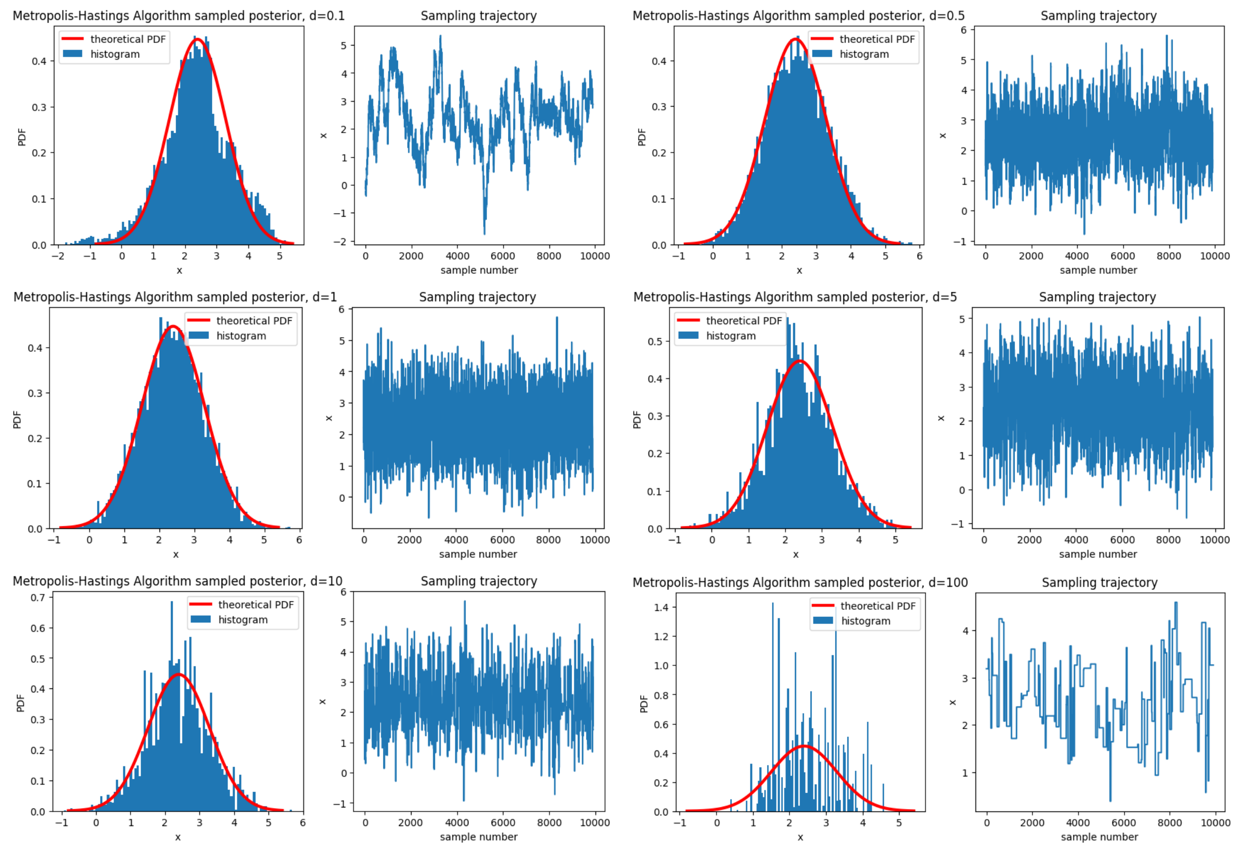
Then we can apply the Random Walk Metropolis:
For the state x, generate the new state $x' = x + d \cdot \mathcal{N}(0,1)$. Since $\mathcal{N}(0,1)$ is systemic, thus

$$x = x' - d \cdot \mathcal{N}(0,1) \sim x' + d \cdot \mathcal{N}(0,1)$$

which means that the one-step transition probability density has the property $f(x,x') = f(x',x)$.
The acceptance rate is

$$a_{x,x'} = \min\left(\frac{\pi_{x'} f_{x',x}}{\pi_x f_{x,x'}}, 1\right) = \min\left(\frac{\pi_{x'}}{\pi_x}, 1\right) = \min\left(e^{-\frac{1}{2\sigma^2}(y-x')^2 - \frac{1}{2\tau^2}(x'-\mu)^2 + \frac{1}{2\sigma^2}(y-x)^2 + \frac{1}{2\tau^2}(x-\mu)^2}, 1\right)$$

And the simulated results are as follows: When d is too small, the exploration is not enough, causing the



restricted mobility. And when d is too large, there are many flatten regions, which means the acceptance rate is too small. So the optimal choice is $d = 1$.

Problem 6 continued on next page...

And the estimated mean and variance of different d are as follows:

d	0.1	0.5	1	5	10	100
mean	2.376	2.454	2.377	2.355	2.382	2.521
variance	1.149	0.869	0.806	0.797	0.794	0.701

The theoritical mean is 2.4, and variance is 0.8. From the estimated mean and variance, we could also discover that $d = 1$ is the optimal choice.

Problem 7

Bivariate Standard Normal Distribution: Implement a Gibbs sampler to generate samples from a bivariate standard normal distribution with correlation $\rho = 0.6, -0.6$.

Solution

Suppose that the correlation of the bivariate standard normal distribution is $\rho \in (-1, 1)$, which means that the covariance matrix is $\Sigma = \begin{bmatrix} 1 & \rho \\ \rho & 1 \end{bmatrix}$. Thus the joint PDF is

$$f_{X,Y}(x, y) = \frac{1}{2\pi\sqrt{1-\rho^2}} e^{-\frac{x^2 - 2\rho xy + y^2}{2(1-\rho^2)}}$$

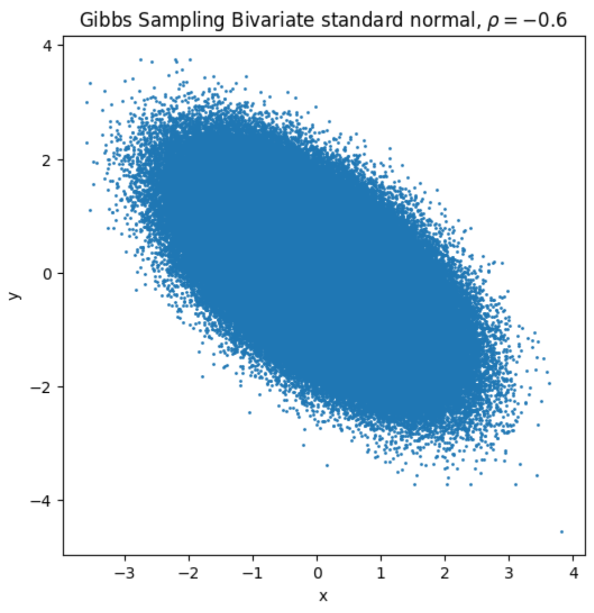
So we can derive that the conditional PDF of X given $Y = y$ is

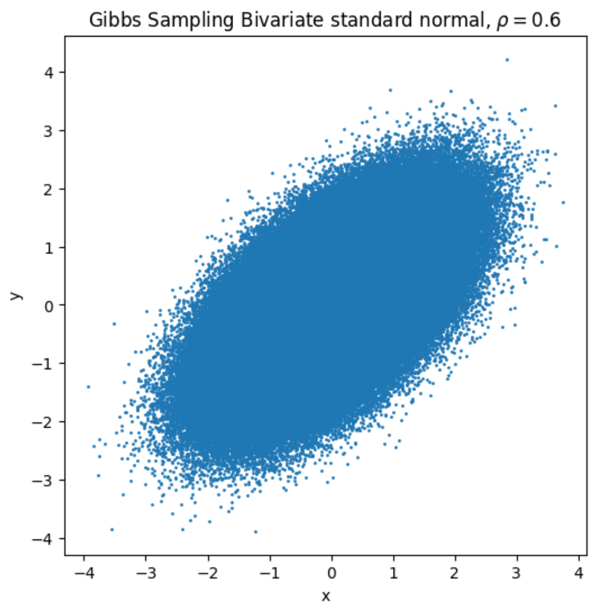
$$f_{X|Y=y}(x|Y = y) \propto e^{-\frac{(x-\rho y)^2}{2(1-\rho^2)}} \sim \mathcal{N}(\rho y, 1 - \rho^2)$$

Similarly, the conditional PDF of Y given $X = x$ is

$$f_{Y|X=x}(y|X = x) \propto e^{-\frac{(y-\rho x)^2}{2(1-\rho^2)}} \sim \mathcal{N}(\rho x, 1 - \rho^2)$$

Then we can apply the Gibbs sampler to generate samples from the bivariate standard normal distribution with correlation $\rho = 0.6, -0.6$ in 1000000 samples, and the burn in is set to be 10000 samples. The results are as follows:





Problem 8

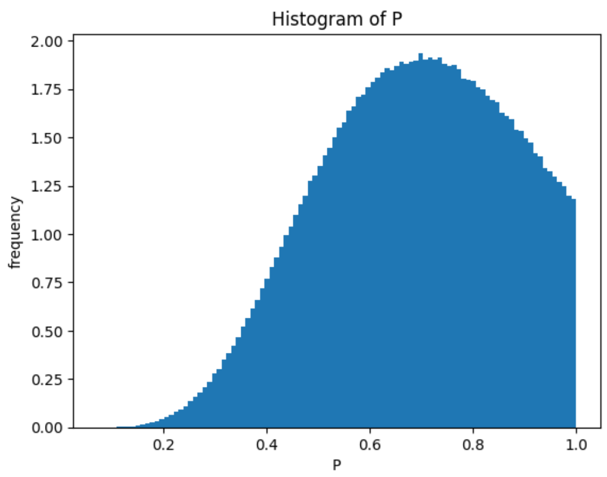
Chicken-Egg with Unknown Parameters: The parameter setting: $\lambda = 10, a = b = 1, x = 7$.
(a) Implement a Gibbs sampler to find the posterior mean and the variance of p after observing x hatched eggs.
(b) Implement a Metropolis-Hastings algorithm to find the posterior mean and variance of p after observing x hatched eggs.
(c) Compare such two methods of MCMC.
Solution
Let $N \sim \text{Pois}(\lambda)$ donates number of eggs.
The prior distribution of the hatch probability is $P \sim \text{Beta}(a, b)$.
$X|P = p \sim \text{Pois}(\lambda p)$ is the number of hatched eggs, $N - X|P = p \sim \text{Pois}(\lambda(1-p))$ is the number of unhatched eggs.
(a) Directly sample the conditional probability above is complex, so we consider the conditional joint distribution of $X, N|P$:
Since $X|N = n, P = p \sim \text{Bin}(n, p)$, and we have the prior $P \sim \text{Beta}(a, b)$, from the Beta-Binomial conjugate, we can get that

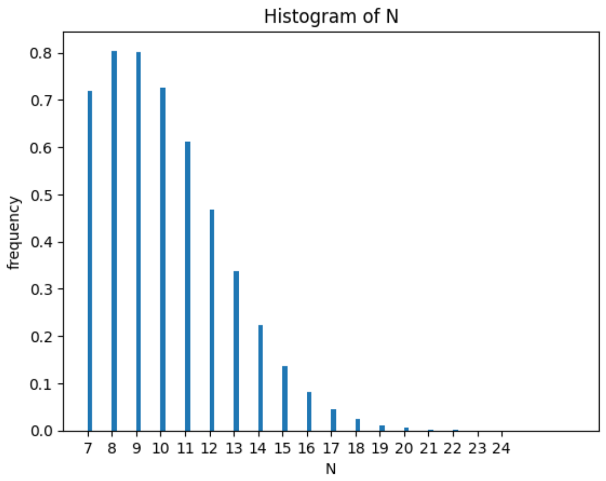
$$P|N = n, X = x \sim \text{Beta}(a + x, b + n - x)$$

And from the definition of N and X, we have

$$f_{N|P,X}(n|P = p, X = x) \sim x + \text{Pois}(\lambda(1-p))$$

The conditional distributions are all easy to sample, so we can use the Gibbs sampling to sample the joint distribution of $P, N|X$, then the marginal distribution of $P|X = x$ and $N|X = x$ can be calculated. Set the initial state to be $p_0 = 0.5, n_0 = 7$. The sampling results are as follows:





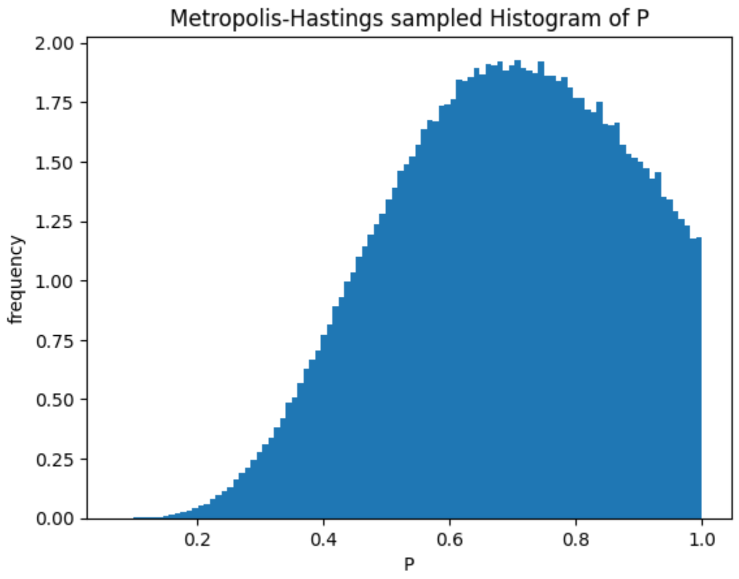
The estimated P given $X = 7$ has mean: 0.68449, variance: 0.03196.
(b) Directly calculate the posterior distribution of P with Metropolis-Hastings algorithm:

$$\begin{aligned} f_{P|X}(p|X = x) &\propto P(X = x|P = p) f_P(p) \\ &= e^{-(\lambda p)} \frac{(\lambda p)^x}{x!} \cdot \frac{1}{\beta(a, b)} p^{a-1} (1-p)^{b-1} \\ &\propto e^{-\lambda p} (\lambda p)^x \cdot p^{a-1} (1-p)^{b-1} \end{aligned}$$

Since the support of P is $[0,1]$, thus we can $\text{Unif}(0,1)$ to be the proposal distribution, i.e. the one-step transition probability density from state x to y is $f_{i,j} = f_{X_{n+1}|X_n}(j|i) = 1, j > 0$, so we can get that the acceptance rate:

$$a_{i,j} = \min\left(\frac{\pi_j f_{j,i}}{\pi_i f_{i,j}}, 1\right) = \min\left(\frac{e^{-\lambda j}(\lambda j)^x \cdot j^{a-1}(1-j)^{b-1}}{e^{-\lambda i}(\lambda i)^x \cdot i^{a-1}(1-i)^{b-1}}, 1\right)$$

The sampling results are as follows: The estimated P given $X = 7$ has mean: 0.68468, variance: 0.03198.



(c) We can see that both Gibbs sampling and Metropolis-Hastings algorithm with same sample number as burn in number have a similar estimation.
Metropolis-Hastings:
Advantages: It can sample many different distributions with suitable proposal distribution, without requiring the conditional distribution is easy to sample.
Disadvantages: It is actually a accept-reject method, if the simulation steps are not enough, it may reject many times and stay at the same state, also, it may need more calculations.
Gibbs sampling: Advantages: It need less calculations, do not need to reject samples, thus it is efficient.
Disadvantages: It need to sample the conditional distribution, which required to be easy to sample. If the conditional distributions are complex, it still require Metropolis-Hastings algorithm or other sampling methods to generate a single sample, which is not efficient.

Problem 9

Three-dimensional Joint Distribution: Implement a Gibbs sampler to generate samples from the three-dimensional joint distribution shown in the lecture.

Solution

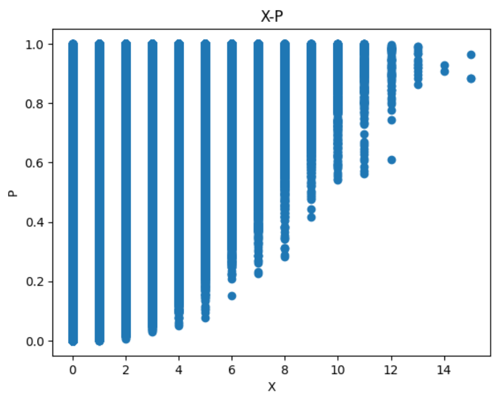
Random variables X, P, N have joint density

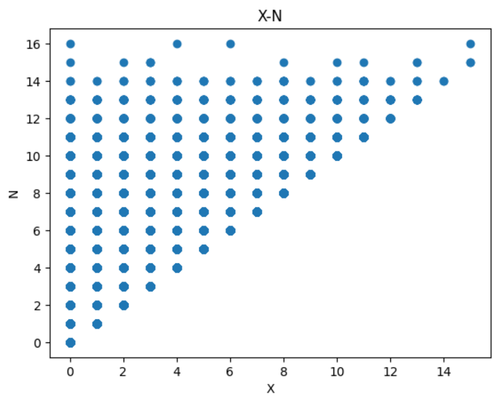
$$\pi(x,p,n) \propto \binom{n}{x} p^x (1-p)^{n-1} \frac{4^n}{n!}$$

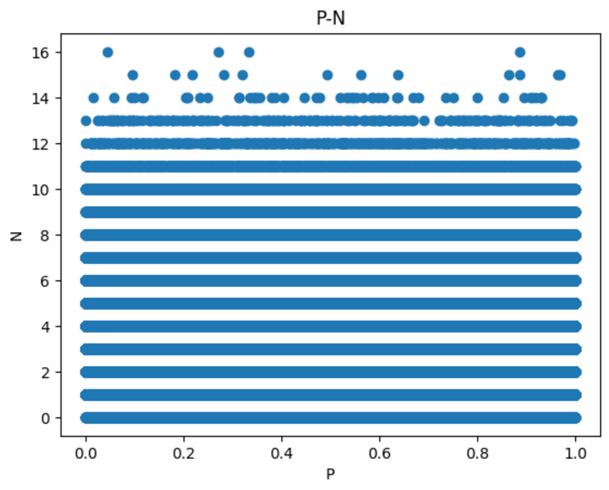
for $x = 0, 1, 2, \dots, n$, $0 < p < 1$, and $n = 0, 1, 2, \dots$. The p variable is continuous, x and n are discrete. We can get the conditional PDF and PMF for the corresponding variables as follows:

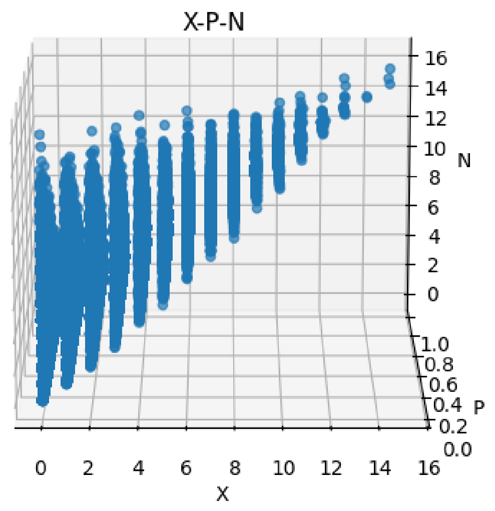
$$\begin{aligned} P_{X|N,P}(X = x|N = n, P = p) &\propto \binom{n}{x} p^x (1-p)^{n-x} \sim \text{Bin}(n,p) \\ f_{P|X,N}(p|X = x, N = n) &\propto p^x (1-p)^{n-x} \sim \text{Beta}(x+1, n-x+1) \\ P_{N|X,P}(N = n|X = x, P = p) &\propto \binom{n}{x} p^x (1-p)^{n-x} = x + Z \end{aligned}$$

We can get that the last PMF is a shifted Poisson, where $Z \sim \text{Pois}(4(1-p))$.
Initially, we set $(x_0, p_0, n_0) \leftarrow (1, 0.5, 2)$, and we can use Gibbs sampling to generate the samples.
The simulated results are as follows:

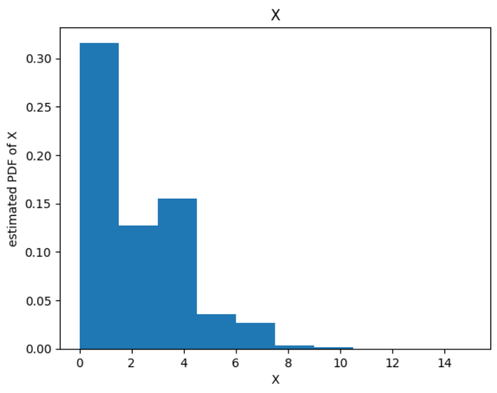


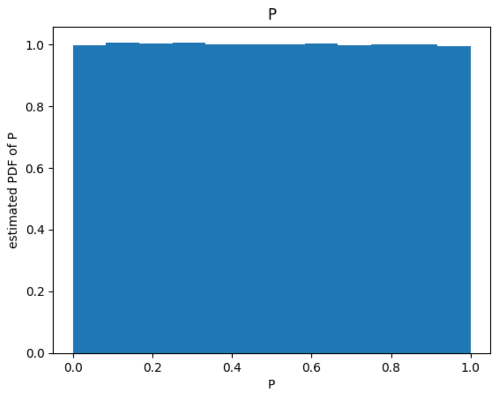


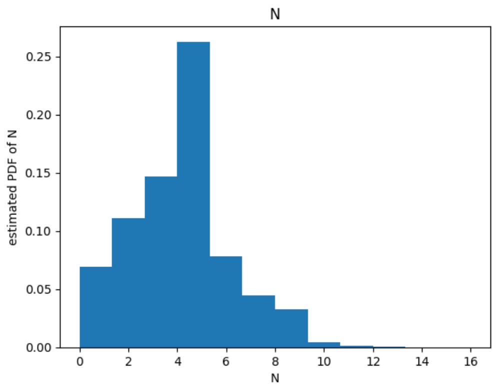




The estimated marginal distribution are as follows:







Problem 10

Example I in the Lecture, Section 4
(a) Implement a Metropolis-Hastings algorithm.
(b) Implement a Hamiltotinn Monte Carlo algorithm
(c) Implement with the Aoceptance-Rejection Method.
(d) Compare the above three algorithms with corresponding pros and cons.
Solution
The target distribution's PDF is

$$f(x) = \frac{8}{3\pi\sqrt{5}}\left(1+\frac{1}{5}x^2\right)^{-3}, -\infty < x < \infty$$

Since the support of the distribution is $-\infty < x < \infty$, so we first introduce the distribution Y be the deformation of Cauchy distribution. And its PDF and CDF are:

$$\begin{aligned}g(x) &= \frac{1}{\pi\sqrt{5}(1+\frac{1}{5}x^2)}\\ G(x) &= \frac{1}{\pi\sqrt{5}}\int_{-\infty}^{x}\frac{1}{1+\frac{1}{5}y^2}\mathrm{d}x = \frac{1}{\pi}\arctan\left(\frac{x}{\sqrt{5}}\right)\end{aligned}$$

Since $G(x)$ is strictly increasing, by calculating its the inverse function, and sample $U \sim \text{Unif}(0,1)$, we can get the samples Y by:

$$G^{-1}(y) = \sqrt{5}\tan(\pi y), y \in [-0.5, 0.5] \Rightarrow XY = \sqrt{5}\tan(\pi(U-0.5))$$

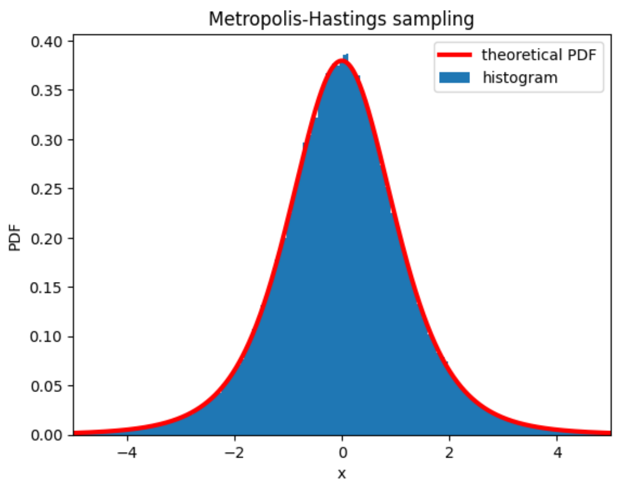
(a) Take the deformation of Cauchy distribution as the proposal distribution, i.e. the one step transition PDF is

$$f_{x,y} = \frac{1}{\pi\sqrt{5}(1+\frac{1}{5}y^2)}$$

So the acceptance rate is:

$$a_{x,y} = \min\left(\frac{\pi_y f_{y,x}}{\pi_x f_{x,y}}, 1\right) = \min\left(\left(\frac{1+\frac{1}{5}x^2}{1+\frac{1}{5}y^2}\right)^2, 1\right)$$

Apply the Metropolis-Hastings algorithm to generate samples as follows:



(b) Since π_x is a the stationary of the target distribution, and we can ignore the constant form, so the potential energy is

$$U(x) = -\log(\pi_x) = -\log\left(\frac{8}{3\pi\sqrt{5}}\right) + 3\log\left(1+\frac{1}{5}x^2\right) \overset{\text{ignore const}}{\Rightarrow} U(x) = 3\log\left(1+\frac{1}{5}x^2\right)$$

And let the mass be 1, then the Kinetic energy is

$$V(\omega) = \frac{1}{2}\omega^2, \quad \text{where } \omega \sim \mathcal{N}(0,1)$$

Thus the Hamiltonian energy is

$$H(x,\omega) = U(x) + V(\omega) = 3\log\left(1+\frac{1}{5}x^2\right) + \frac{1}{2}\omega^2$$

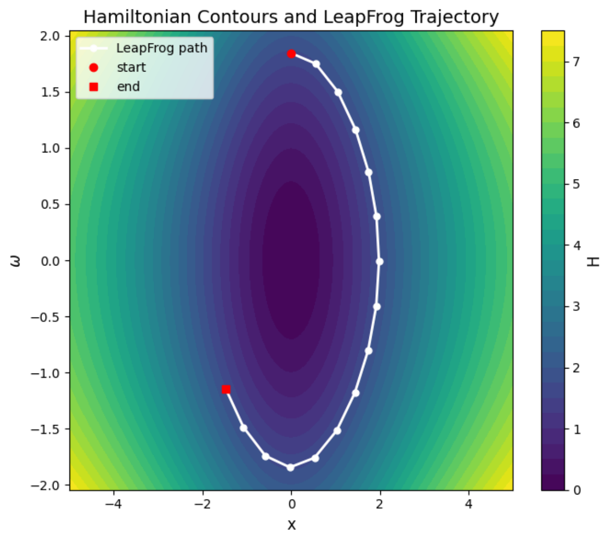
Initially, set $x_0 = 0, \omega_0 \sim \mathcal{N}(0,1)$, apply the Leapfrog method to sample:

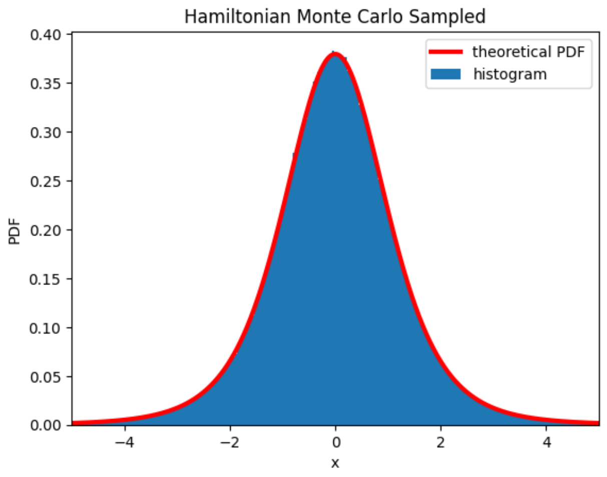
$$\begin{aligned}
\omega_{t+\frac{\delta}{2}} &= \omega_t - \frac{\delta}{2}\cdot\frac{\mathrm{d}U(x_t)}{\mathrm{d}x_t} = \omega_t - \frac{\delta}{2}\cdot\frac{6x_t}{5+x_t^2} \\
x_{t+\delta} &= x_t + \delta\cdot\frac{\mathrm{d}V(\omega_{t+\frac{\delta}{2}})}{\mathrm{d}\omega_{t+\frac{\delta}{2}}} = x_t + \delta\cdot\omega_{t+\frac{\delta}{2}} \\
\omega_{t+\delta} &= \omega_{t+\frac{\delta}{2}} - \frac{\delta}{2}\cdot\frac{\mathrm{d}U(x_{t+\delta})}{\mathrm{d}x_{t+\delta}} = \omega_{t+\frac{\delta}{2}} - \frac{\delta}{2}\cdot\frac{6x_{t+\delta}}{5+x_{t+\delta}^2}
\end{aligned}$$

After Leapfrog L steps, we can get the state (x_L, ω_L). And set $(x_L, -\omega_L)$ to be the proposal distribution. And the accept rate for the Hamiltonian Monte Carlo algorithm is

$$a_{(x_0,\omega_0),(x_L,-\omega_L)} = \min\left(\frac{\exp\left(-H(x_L,-\omega_L)\right)}{\exp\left(-H(x_0,\omega_0)\right)}, 1\right) = \min\left(\frac{\exp\left(-U(x_L) - V(-\omega_L)\right)}{\exp\left(-U(x_0) - V(\omega_0)\right)}, 1\right)$$

We step the stepsize to be $\delta = 0.3, L = 15$. One trajectory and sample results are as follows:

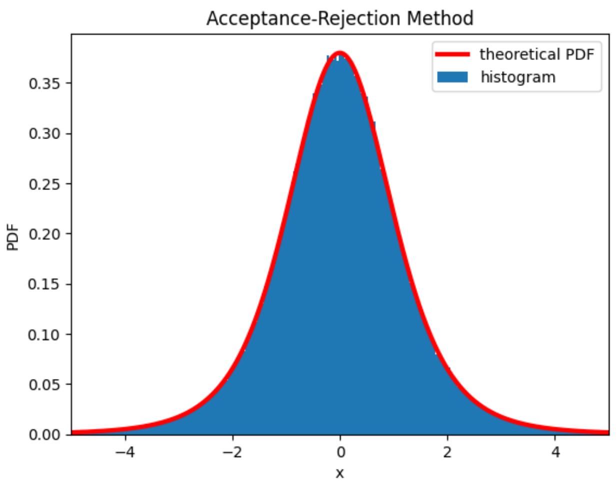




(c) As for the acceptance-rejection algorithm, we also use the deformation of Cauchy distribution as the proposal distribution. Let c donate a constant:

$$c = \sup_y \frac{f(y)}{g(y)} = \sup_y \frac{\frac{8}{3\pi\sqrt{5}}\left(1+\frac{1}{5}y^2\right)^{-3}}{\frac{1}{\pi\sqrt{5}(1+\frac{1}{5}y^2)}} = \frac{8}{3}\left(1+\frac{1}{5}y^2\right)^{-2}\Bigg|_{y=0} = \frac{8}{3}$$

After sampling the deformation of Cauchy distribution, we can apply the Acceptance-Rejection algorithm to generate the samples:



(d) 1. Metropolis-Hastings Algorithm:
Advantages: It is highly general-purpose, could applicable to a wide variety of distributions, and it does not require the normalization constant of the target distribution.
Disadvantages: Samples are often requires select a suitable distribution, and requires steps to burn up.
2. Hamiltonian Monte Carlo Algorithm:
Advantages: Due to the conservation of energy, the acceptance rate should be 1. But there exists numerical error, however quite close to 1, which means it has quite high acceptance rate. The samples are efficient, especially in high-dimensional problems.
Disadvantages: Requires computation of gradients, increasing implementation complexity. Sensitive to parameters: step size δ and number of steps for LeapFrog L.
3. Acceptance-Rejection Method:
Advantages: It produces independent samples, and is simple to implement. If the good envelope function is suitable, it is efficient.
Disadvantages: Efficiency depends heavily on the choice of proposal distribution, for example, ours implement has a low acceptance rate, which is only 0.375255.

Problem 11

Example II in the Lecture, Section 4
(a) Implement a Metropolis-Hastings algorithm.
(b) Implement a Hamiltonian Monte Carlo algorithm
(c) Implement with the Acceptance-Rejection Method.
(d) Compare the above three algoxithms with corresponding pros and cons.
Solution
The target distribution's PDF is

$$f(x) = 0.4 \cdot \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}(x-2)^2} + 0.6 \cdot \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}(x+2)^2}, -\infty < x < \infty$$

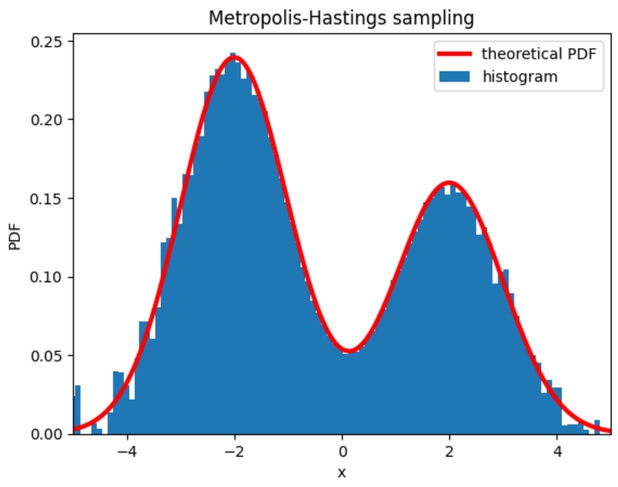
(a) Take the $\mathcal{N}(0,1)$ as the proposal distribution, i.e. the one step transition PDF is

$$f_{x,y} = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}y^2}$$

So the acceptance rate is:

$$a_{x,y} = \min\left(\frac{\pi_y f_{y,x}}{\pi_x f_{x,y}}, 1\right) = \min\left(\frac{2e^{2y-2} + 3e^{-2y-2}}{2e^{2x-2} + 3e^{-2x-2}}, 1\right)$$

Apply the Metropolis-Hastings algorithm to generate samples as follows:



(b) Since π_x is a the stationary of the target distribution, and we can ignore the constant form, so the potential energy is

$$U(x) = -\log(\pi_x) \overset{\text{ignore const}}{\Rightarrow} U(x) = -\log\left(2e^{-\frac{1}{2}(x-2)^2} + 3e^{-\frac{1}{2}(x+2)^2}\right)$$

And let the mass be 1, then the Kinetic energy is

$$V(\omega) = \frac{1}{2}\omega^2, \quad \text{where } \omega \sim \mathcal{N}(0,1)$$

Thus the Hamiltonian energy is

$$H(x,\omega) = U(x) + V(\omega) = -\log\left(2e^{-\frac{1}{2}(x-2)^2} + 3e^{-\frac{1}{2}(x+2)^2}\right) + \frac{1}{2}\omega^2$$

Problem 11 continued on next page...

Initially, set $x_0 = 0, \omega_0 \sim \mathcal{N}(0,1)$, apply the Leapfrog method to sample:

$$\omega_{t+\frac{\delta}{2}} = \omega_t - \frac{\delta}{2}\cdot\frac{\mathrm{d}U(x_t)}{\mathrm{d}x_t} = \omega_t - \frac{\delta}{2}\cdot\frac{2(x_t-2)e^{-\frac{1}{2}(x_t-2)^2}+3(x_t+2)e^{-\frac{1}{2}(x_t+2)^2}}{2e^{-\frac{1}{2}(x_t-2)^2}+3e^{-\frac{1}{2}(x_t+2)^2}}$$

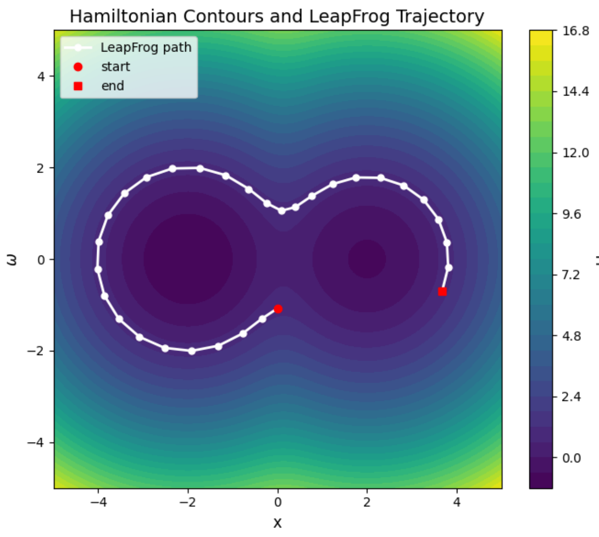
$$x_{t+\delta} = x_t + \delta\cdot\frac{\mathrm{d}V(\omega_{t+\frac{\delta}{2}})}{\mathrm{d}\omega_{t+\frac{\delta}{2}}} = x_t + \delta\cdot\omega_{t+\frac{\delta}{2}}$$

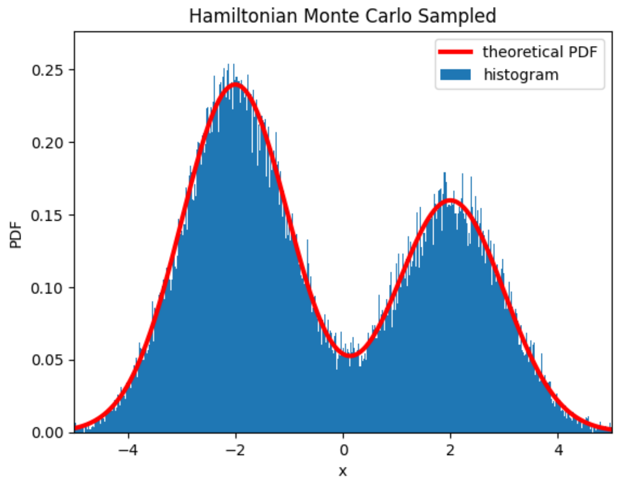
$$\omega_{t+\delta} = \omega_{t+\frac{\delta}{2}} - \frac{\delta}{2}\cdot\frac{\mathrm{d}U(x_{t+\delta})}{\mathrm{d}x_{t+\delta}} = \omega_{t+\frac{\delta}{2}} - \frac{\delta}{2}\cdot\frac{2(x_{t+\delta}-2)e^{-\frac{1}{2}(x_{t+\delta}-2)^2}+3(x_{t+\delta}+2)e^{-\frac{1}{2}(x_{t+\delta}+2)^2}}{2e^{-\frac{1}{2}(x_{t+\delta}-2)^2}+3e^{-\frac{1}{2}(x_{t+\delta}+2)^2}}$$

After Leapfrog L steps, we can get the state (x_L, ω_L). And set $(x_L, -\omega_L)$ to be the proposal distribution. And the accept rate for the Hamiltonian Monte Carlo algorithm is

$$a_{(x_0,\omega_0),(x_L,-\omega_L)} = \min\left(\frac{\exp(-H(x_L,-\omega_L))}{\exp(-H(x_0,\omega_0))}, 1\right) = \min\left(\frac{\exp(-U(x_L)-V(-\omega_L))}{\exp(-U(x_0)-V(\omega_0))}, 1\right)$$

We step the stepsize to be $\delta = 0.3, L = 30$. One trajectory and sample results are as follows:





(c) We can use acceptance-rejection method to sample on $\mathcal{N}(0,1)$. Let $Z \sim \mathcal{N}(0,1)$, and $W = |Z|$. So $Z \in (-\infty, +\infty)$, $W \in (0, +\infty)$.

Problem 11 continued on next page...

We can calculate the PDF of W:

$$h(x) = f_W(x) = \frac{2}{\sqrt{2\pi}} e^{-\frac{1}{2}x^2}, x > 0$$

And we can choose that $Y \sim \text{Expo}(1)$, and its PDF is $g(y) = f_Y(y) = e^{-y}, y > 0$, thus we can get that

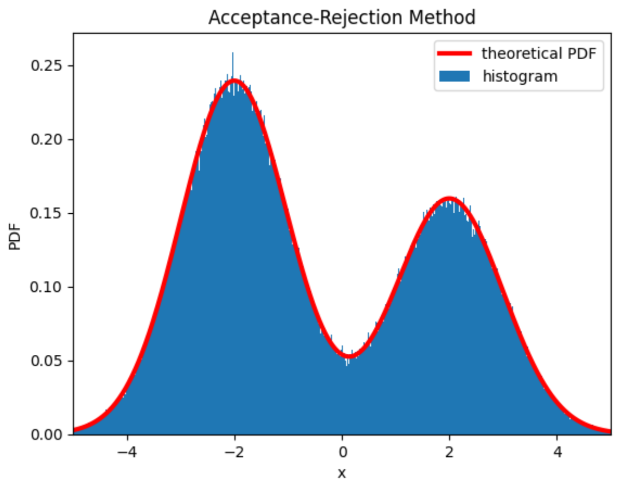
$$c = \sup_y \frac{h(y)}{g(y)} = \sup_y \frac{\frac{2}{\sqrt{2\pi}} e^{-\frac{1}{2}y^2}}{e^{-y}} = \frac{2}{\sqrt{2\pi}} e^{-\frac{1}{2}y^2+y}\bigg|_{y=1} = \sqrt{\frac{2e}{\pi}}$$

Then we can do the acceptance-rejection to sample the distribution of W.

To sample on $Z \sim \mathcal{N}(0,1)$, since $W = |Z|$, so we can generate $U_1 \sim \text{Unif}(0,1)$, and let $Z = \begin{cases} W, & U_1 \le \frac{1}{2} \\ -W, & U_1 > \frac{1}{2} \end{cases}$.

After that, we can sample the distribution of $Z \sim \mathcal{N}(0,1)$ with acceptance-rejection method.

Then for the true distribution X that we want to sample, it could be regard as $X = 0.4(Z+2) + 0.6(Z-2)$, so we can generate $U_2 \sim \text{Unif}(0,1)$, and let $X = \begin{cases} Z+2, & U_2 \le 0.4 \\ Z-2, & U_2 > 0.6 \end{cases}$. This correctness could be easily proved by using LOTP. Thus, we applied the Acceptance-Rejection algorithm to generate the samples:



(d) 1. Metropolis-Hastings Algorithm:

Advantages: It is highly general-purpose, could applicable to a wide variety of distributions, and it does not require the normalization constant of the target distribution.

Disadvantages: Samples are often requires select a suitable distribution, and requires steps to burn up.

2. Hamiltonian Monte Carlo Algorithm:

Advantages: Due to the conservation of energy, the acceptance rate should be 1. But there exists numerical error, however quite close to 1, which means it has quite high acceptance rate. The samples are efficient, especially in high-dimensional problems.

Disadvantages: Requires computation of gradients, increasing implementation complexity. Sensitive to parameters: step size δ and number of steps for LeapFrog L.

3. Acceptance-Rejection Method:

Advantages: It produces independent samples, and is simple to implement. If the good envelope function is suitable, it is efficient.

Disadvantages: Efficiency depends heavily on the choice of proposal distribution, for example, ours implement has a not very high acceptance rate, which is 0.760735, but it is somehow acceptable.

Problem 12

How to solve a general combinatorial optimization probkem with MCMC method?

Solution

To solve a combinatorial optimization problem with a global maximum using an MCMC method, we convert the optimization problem into a sampling problem by defining a target probability distribution.

- Constructing the target distribution:
  The combinatorial optimization problem is usually formulated as

  $$\max_{x\in\mathcal{C}} V(x) \Rightarrow \max_{x,\pi(x)} \pi(x)V(x) \text{ ,where } \pi(x) = \begin{cases} 1 & \text{if } x \in \mathcal{C} \\ 0 & \text{otherwise} \end{cases}$$

  But the constrain is a Dirac delta function, which is not suitable for sampling in MCMC. So we can consider applying the entropy regularization as a penalty to the objective function:

  $$\max_{\pi(x)} \pi(x)V(x) - \frac{1}{\beta}\sum_{x\in\mathcal{C}} \pi(x)\log\pi(x)$$

  IT could be solve that the optimal value, aka the target distribution is:

  $$\pi(x) = \frac{e^{\beta V(x)}}{\sum\limits_{y} e^{\beta V(y)}} \propto e^{\beta\, V(x)}$$

  Where β is an inverse temperature parameter. The simulated annealing usually set it to be $\beta = \frac{1}{T}$, where the temperature $T = \frac{C}{\log(t+1)}$, t is the iteration number. In this formulation, solutions with higher $V(x)$ receive exponentially higher probability.

- MCMC Sampling Process

  1. State Space: Define the set of all possible solutions x in the feasible region. (e.g., binary vectors, permutations).
  2. Proposal new state: Generate a new state y from the current state x.(e.g. randomly flipping bits or swapping elements).
  3. Acceptance rate: For current state x and new state y, under the current β, compute the acceptance rate:
     $$a(x, y) = \min\left(1,\, e^{\beta\,(V(y)-V(x))}\right)$$
  4. Update: Flip a coin with probability $a(x, y)$, transition to the new state y if heads; otherwise, retain x.

Thus, using an MCMC algorithm to sample from this distribution while dynamically lowering β over time. In the early stage, a low β (high temperature) drives the system to greedily search for high-$V(x)$ regions; later, it increases β (low down the temperature). This approach increases the chance of finding the global optimum or a near-optimal solution. After a sufficient number of iterations, select the solution with the highest $V(x)$ as an approximation of the global optimum.

Problem 13

Illustrate the relationship between sampling and optimization.

Solution

1. Convert optimization problem to sampling problem.

An optimization problem is usually formulated as

$$\begin{aligned} \max_x \quad & f(x) \\ \text{s.t.} \quad & f_i(x) \le 0 \end{aligned}$$

The constrained optimization problem is not suitable for sampling in MCMC. So we can consider applying the penalty to the objective function:

$$\tilde{f}(x) = f(x) + \lambda \cdot f_i(x) \cdot \mathbb{I}_{f_i(x)>0}$$

where $\lambda > 0$ and $f_i(x) \cdot \mathbb{I}_{f_i(x)>0}$ measures constraint violations.

To get the optimal value, we can apply the Metropolis random walk. For each step, translate from the current state x to a new state x' by

$$x' = x + \epsilon, \quad \epsilon \sim \mathcal{N}(0, \sigma^2 I)$$

with acceptance rate

$$a_{x,x'} = \min\left(1, e^{\beta\left(\tilde{f}(x') - \tilde{f}(x)\right)}\right)$$

β could be selected similarly to the Simulated Annealing method. Then the sampled distribution with maximum $\tilde{f}(x)$ could be approximated regarded as the optimal solution.

2. Convert sampling problem to optimization problem.

Suppose that we have a target distribution π, which is difficult to sample. We could try to use q_θ to approximate it, where θ is the parameters. Thus, we can minimize the KL-divergence $KL(q_\theta||\pi)$ to let the estimated distribution converge to the target distribution, and then sample from q_θ, which is easier to sample. Thus the sampling problem could be converted to the optimization problem.

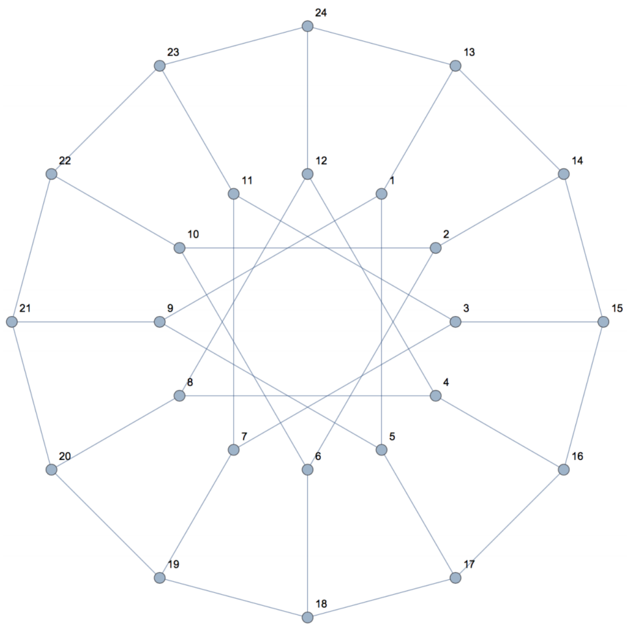
Usually, the optimization problem use the Variational Inference to approximate the target distribution. i.e. get the evidence lower bound (ELBO) of the KL-divergence, make the optimization problem easier.

Reference: https://zhuanlan.zhihu.com/p/88336614

Problem 14

Markov Chain Monte Carlo for Wireless Networks. Given a wireless network with 24 links and $0-1$ interference model, i.e., any two links are either intefere with each other or not. To describe the interference relationship between wireless links, we introduce the conflict graph model. In such model, the vertex of the conflict graph represents the wircless link. An edge between two vertioes means corresponding two links interfere with each other. The following Figure shows the corresponding conflict graph for 24-link wireless network. You are required to find the maximum independent set of the conflict graph, i.e., the largest set of wireless links that can simultancously transmit without interferences. Design the algorithm by MCMC method and evaluate your algorithm.

- Show your MCMC Design with a discrete Markow chain. Use both theory and simulation results to justify your algorithms.
- Show your MCMC Design with a continnous Markov chain. Use both theory and simulation results to justify your algorithms.
- Discuss the pros and cons for such chains.



Solution
(a) Discrete time Markov Chain:
The independent set problem is similar to the Knapsack problem: use a binary sequence to represent whether choose the link or not. And each time we randomly flip a bit in the sequence, if the new sequence does not satisfy the independent set constraint, stay at the current state, otherwise go to the new state, this is obviously a reducible Markov chain, and its stationary distribution π is the uniform distribution.
Then, let the value function $V(x)$ be the number of selected links of the state x. According to the Metropolis-Hastings algorithm, the acceptance rate for the transition to y is

$$a_{x,y} = e^{\beta(V(y)-V(x))}$$

Then we just need to check the samples with largest value, i.e. the maximum indenpendent set.
The initial state is set to be an all 0 sequence, the factor is set to be

$$\beta(t) = C \cdot \log(t+1)$$

Problem 14 continued on next page...

Where $C = \dfrac{1}{\log\left(\frac{T}{2}\right)}$, T is the total number of steps.
After simulation 10000 steps, the sampled maximum size of the independent set is 9, and it sampled 65 different sequences. The first sampled sequence is: 0100101000011010010101100, and it is verified to be a correct solution.
(b) Continuous time Markov Chain:
It is mostly same to the discrete time Markov chain, but the time is continuous, and the embedded chain has no self-transition. Thus, when considering the transition for state x , we need to find out all valid transitions $y \in \mathcal{N}(x)$, then randomly select a transition from $\mathcal{N}(x)$. And there is no other difference to the discrete time Markov chain. Actually, we should set a transition rate to get the transition time intervals, but we do not need to use the time, so it is ignored.
After simulation 10000 steps, the sampled maximum size of the independent set is 9, and it sampled 99 different sequences. The first sampled sequence is: 1100000100100010101010101001, and it is verified to be a correct solution.
(c) 1. Discrete time Markov Chain:
Advantages: It is simple to implement, at each step, a single bit is randomly flipped. One only needs to check if the new state satisfies the independent set constraint and then directly apply the Metropolis-Hastings acceptance criterion.
Disadvantages: It may have a low acceptance rate, and stay in the origin state, the chain may explore the valid state space slowly, leading to a relatively small number of distinct sampled states. In a total of 10000 samples, only 65 different optimal states is observed.
2. Continuous time Markov Chain:
Advantages: The embedded chain allows transitions only to valid neighboring states chosen at random, ensuring that each transition results in a state change. In a total of 10000 samples, 99 different optimal states were observed, which is beneficial for finding the global optimum.
Disadvantages: It requires enumerating all valid neighboring states of the current state and then randomly selecting one among them, which can be slightly more complex in terms of coding and computation.