A Primer on Markov Chain Monte Carlo Algorithms

Pramook Khungurn

February 7, 2022

This note is a primer on Markov chain Monte Carlo (MCMC) algorithms. I take materials from [Andrieu et al., 2003], [Roberts and Rosenthal, 2004], and [Neal, 2011].

1 Introduction

- We are given an unnormalized density function π_u defined on a measurable space (Ω, Σ) such that $0 < \int_\Omega \pi_u(x)\,\mathrm{d}x < \infty$.
- The above density gives rise to a probability measure P, which is given by

  $$P(A) = \frac{\int_A \pi_u(x)\,\mathrm{d}x}{\int_\Omega \pi_u(x)\,\mathrm{d}x}$$

  for any $A \in \Sigma$. The corresponding probability density function is

  $$\pi(x) = \frac{\pi_u(x)}{\int_\Omega \pi_u(x)\,\mathrm{d}x}.$$

  We collectively call P and π the **target distribution**.
- MCMC algorithms allow us to perform the following two following tasks:
  - Sample elements from Ω according to P.
  - Compute an estimate of

    $$\pi(f) = E_{x\sim\pi}[f(x)] = \int_\Omega f(x)\pi(x)\,\mathrm{d}x = \frac{\int_\Omega f(x)\pi_u(x)\,\mathrm{d}x}{\int_\Omega \pi_u(x)\,\mathrm{d}x}.$$

- There are a number of motivations for doing this.
  - In statistictical mechanics, the probability density of state x is given by

    $$\pi(x) = \frac{1}{Z}\exp\left(-\frac{E(x)}{kT}\right)$$

    where $E(x)$ is the Hamiltonian of s, k is the Boltzmann's constant, and T is the temperature of the system. The constant Z is called the **partition function**, and it is given by

    $$Z = \int_\Omega \exp\left(-\frac{E(x)}{kT}\right)\mathrm{d}x.$$

    We see that this setting is exactly the same as the one we just discuss after taking $\pi_u(x) = e^{-E(x)/(kT)}$. So, MCMC is useful for computing expectations arising from these probabilities.

– In Bayesian parameter estimation, we are given data x, and we want to estimate parameter θ of our model, which includes the how to compute the prior $p(\theta)$, and the likelihood $p(x|x)$. The posterior is given by

$$p(\theta|x) = \frac{p(x|\theta)p(\theta)}{\int p(x|\theta)p(\theta)\,\mathrm{d}\theta}$$

In many cases, we also want to compute expectations such as

$$E_{\theta\sim p(\theta|x)}[f(\theta)] = \int f(\theta)p(\theta|x)\,\mathrm{d}\theta = \frac{\int f(\theta)p(x|\theta)p(\theta)\,\mathrm{d}\theta}{\int p(x|\theta)p(\theta)\,\mathrm{d}\theta}.$$

So, if we let Θ be the set of the θ's, then we may set $\Omega = \{(\theta, x) : \theta \in \Theta\}$, and $\pi_u(\theta, x) = p(x|\theta)p(\theta)$ be our unnormalized density.

- In the above situations, what prevents us from computing probabilities and integrals is the difficulty of computing the integral $\int_\Omega \pi_u(x)\,\mathrm{d}x$.
- MCMC gives a way to sample x according to the target distribution without explicity normalizing $\pi_u(x)$. The idea is to constructs a Markov chain $\{X_i\}_{i=0}^\infty$ whose stationary distribution is the target distribution $\pi(x)$. To do so, we need to find a transition kernel K, whose density is k, such that

$$\pi(y) = \int_\Omega \pi(x)k(x, y)\,\mathrm{d}x.$$

- Then, we can run the Markov chain for a long time (starting from somewhere). When n is large, the distribution of X_n will be approximately π. We can now restart the Markov chain from X_n, run the chain for M iterations to obtain M samples x_1, x_2, …, x_M, and then compute the estimate

$$\int_\Omega f(x)\pi(x)\,\mathrm{d}x \approx \widehat{\pi}(f) = \frac{1}{M}\sum_{i=1}^{M} f(x_i).$$

This is an unbiased estimate, and the error $(\pi(f) - \widehat{\pi}(f))/\sqrt{M}$ has Gaussian distribution. We have this property despite the fact that the samples are correlated.[1]

- We say that a Markov chain $\{X_i\}_{i=0}^\infty$ with kernel density function k is **reversible** with respect to probability density function π if

$$\pi(x)k(x, y) = \pi(y)k(y, x) \tag{1}$$

for all $x, y \in \Omega$. The condition (1) is called **detailed balance**.

- Note that when π satisfies detailed balance, we have that π is a stationary distribution. This is because

$$\int_\Omega \pi(x)k(x, y)\,\mathrm{d}x = \int_\Omega \pi(y)k(y, x)\,\mathrm{d}x = \pi(y)\int_\Omega k(y, x)\,\mathrm{d}x = \pi(y).$$

- Hence, the goal would be to find a kernel that satisfies detailed balance with respect to the target distribution. The most popular way to do so is the Metropolis–Hastings algorithm.

[1] `https://en.wikipedia.org/wiki/Markov_chain_central_limit_theorem`

2 The Metropolis–Hastings Algorithm

- The algorithm presupposes a Markov chain over Ω. Let Q denote its transition kernel, and let q denote the transition kernel's density. This density can be unnormalized, just like π_u, but we require a way to sample y according to $q(x, y)$ for any x. We often call q the **proposal distribution**.
- The algorithm proceeds as follows.
  1. Choose some X_0 as the starting point.
  2. Given X_n, generates a proposal Y_{n+1} according to $q(X_n, \cdot)$.
  3. Compute $\alpha(X_n, Y_{n+1})$ where
  $$\alpha(x, y) = \min\left(1, \frac{\pi_u(y)q(y, x)}{\pi_u(x)q(x, y)}\right)$$
  is the **acceptance probability**. Here, if $\pi(x)q(x, y) = 0$, we set $\alpha(x, y) = 1$.
  4. Sample ξ uniformly from $[0, 1)$.
  5. If $\xi < \alpha(x, y)$, then set $X_{n+1} = Y_{n+1}$. Otherwise, set $X_{n+1} = X_n$.
  6. Repeat Step 2 to Step 5 until you are satisfied.
- The algorithm creates a Markov chain whose transition density is given by $k(x, y) = q(x, y)\alpha(x, y)$.
- To see that π is the stationary distribution of this above Markov chain, we need to show that k satisfies detailed balance with respect to π. In other words,
$$\pi(x)k(x, y) = \pi(y)k(y, x).$$
To see this, let $\pi(x) = C^{-1}\pi_u(x, y)$ where $C = \int_\Omega p_u(x)\,\mathrm{d}x$ is the normalizing constant. We have that
$$\begin{aligned}\pi(x)k(x, y) &= \pi(x)q(x, y)\alpha(x, y)\\ &= C^{-1}\pi_u(x)q(x, y)\min\left(1, \frac{\pi_u(y)q(y, x)}{\pi_u(x)q(x, y)}\right)\\ &= C^{-1}\min\Big(\pi_u(x)q(x, y), \pi_u(y)q(y, x)\Big).\end{aligned}$$
Notice that the expression on the RHS is symmetric in x and y. Hence, we have that $\pi(x)k(x, y) = \pi(y)k(y, x)$ as required.
- The main question for employing the Metropolis–Hastings algorithm is how to choose the proposal distribution $q(\cdot, \cdot)$.
- Of course, q needs to be such that the Markov chain is irreducible and aperiodic so that it would converge to the stationary distribution regardless of where we start.
- Note, though, that transition kernel k allows for rejection, so the chain is aperiodic by default. As a result, we only need to make sure that it covers the whole support of π. In other words, if $\pi_u(x) > 0$ and $\pi_u(y) > 0$, then $q(x, y)$ should be greater than 0 too.
- There are many approaches to construct the proposal distribution.
  - **Symmetric Metropolis algorithm.** Here, q is chosen such that $q(x, y) = q(y, x)$. The acceptance probability simplifies to
  $$\alpha(x, y) = \min\left(1, \frac{\pi_u(y)}{\pi_u(x)}\right).$$
  One way to arrange for this is to use $y \sim \mathcal{N}(x, \sigma^2)$.

- **Random walk Metropolis algorithm.** Here, $q(x, y) = q(y - x)$. For example, we may have $y \sim \text{Uniform}(x - 1, x + 1)$.
- **Independence sampler.** Here, $q(x, y) = q(y)$. The acceptance probability thus reduces to

$$\alpha(x, y) = \min\left(1, \frac{\pi_u(y)q(x)}{\pi_u(x)q(y)}\right) = \min\left(1, \frac{w(y)}{w(x)}\right)$$

where $w(x) = \pi_u(x)/q(x)$ for all x.

- **Langevin algorithm.** The proposal distribution is given by

$$\mathbf{y} \sim \mathcal{N}\Big(\mathbf{x} + \frac{\sigma^2}{2}\nabla \log \pi_u(\mathbf{x}), \sigma^2 I\Big)$$

for some small $\sigma > 0$.

- Heuristics on the standard deviation of the proposal distribution.
  - If it is too low (i.e., the proposal distribution is too narrow), then only a few modes of the target distribution $\pi(\cdot)$ would be visited.
  - If it is too wide, then there's a high chance of new samples being rejected, resulting in high correlation between samples.
- If all modes of the target distribution is visited while the acceptance probability is high, we say that the chain **mixes well**.

3 Combining Markov Chains

3.1 Mixture MCMC

- If transition kernel density k_1 and k_2 both result in the same stationary distribution π, then so does the kernel

$$\nu k_1 + (1 - \nu)k_2$$

for any $0 \leq \nu \leq 1$.

- The update step of the MCMC algorithm that uses the mixed kernel would be as follows:

  Sample $\xi \sim \text{Uniform}(0, 1)$.

  if $\xi < \nu$ **then**

  Generate the next state X_{n+1} using kernel k_1.

  else

  Generate the next state X_{n+1} using kernel k_2.

  end if

- Mixing more than two kernels is, of course, possible.
- Mixture MCMC is useful when the target distribution has many narrow peaks. In this case, we can have a "global" kernel that has large variance and a "local" kernel that has small variance. The global proposal would jump between peaks, and the local proposals would allow one to explore the space around each peak.

3.2 Cycle MCMC

- If we have multiple transition kernel densities k_1, k_2, $\ldots$, k_m that have π as the stationary distribution of their Markov chains, then the Markov chain obtained by applying the kernels in the round robin fashion, would also have π as its stationary distribution.
- Another use of this type of MCMC algorithm is when a sample $\mathbf{x}$ can be divided into m blocks $\mathbf{x} = (\mathbf{x}^{(1)}|\mathbf{x}^{(2)}|\cdots|\mathbf{x}^{(m)})$. If the kernel k_i only change the $\mathbf{x}^{(i)}$ block, then applying all the kernels in succession would give us a kernel that change all components of the sample, and this can be engineered to have π as the statinoary distribution.
- The update step of the cycle MCMC algorithm be as follows:

  Use k_1 to get a state $Y_{n+1}^{(1)}$ from X_n.
  Use k_2 to get a state $Y_{n+1}^{(2)}$ from $Y_{n+1}^{(1)}$.
  $\vdots$
  Use k_m to get a state $Y_{n+1}^{(m)}$ from $Y_{n+1}^{(1)}$.
  Set X_{n+1} to $Y_{n+1}^{(m)}$.

- The expression for the overall kernel k would be

$$k(x,y) = \int\int\cdots\int k_1(x,x^{(1)})k_2(x^{(1)},x^{(2)})\cdots k_m(x^{(m-1)},y)\,\mathrm{d}x^{(1)}\mathrm{d}x^{(2)}\cdots\mathrm{d}x^{(m)}.$$

  However, if k_i only changes the ith block independent of the other blocks, then we can write

$$k(\mathbf{x},\mathbf{y}) = \prod_{i=1}^{m} k_i(\mathbf{x}^{(i)},\mathbf{y}^{(i)}).$$

  Weirdly, all the papers that I read write the overall kernel as

$$k = k_1k_2\cdots k_m.$$

4 The Gibbs Sampler

- Suppose that $\Omega = \mathbb{R}^d$. So, we denote an elements of $\mathbb{R}^d$ by $\mathbf{x} = (x_1, x_2, \ldots, x_d)$.
- Let $\mathbf{x}_{-i}$ denote $\mathbf{x}$ with the component x_i removed. In other words,

$$\mathbf{x}_{-i} = (x_1,\ldots,x_{i-1},x_{i+1},\ldots,x_d) = (\mathbf{x}[1:i-1],\mathbf{x}[i+1:d]).$$

- Suppose that, in addition to an expression for $\pi_u(\mathbf{x})$, we are also given, for each i, an algorithm for sampling x_i given all the other components (i.e., given $\mathbf{x}_{-i}$) according to the conditional probability distribution

$$\pi(x_i|\mathbf{x}_{-i}) = \frac{\pi_u(\mathbf{x})}{\int \pi_u(\mathbf{x}[1:i-1],x,\mathbf{x}[i+1,d])\,\mathrm{d}x}.$$

- For further derivatavation, let us denote the marginal distribution of $\mathbf{x}_{-i}$ by

$$\pi(\mathbf{x}_{-i}) = \frac{1}{C}\pi_u(\mathbf{x}_{-i}) = \frac{1}{C}\int \pi_u(\mathbf{x}[1:i-1],x,\mathbf{x}[i+1,d])\,\mathrm{d}x.$$

where $C = \int \pi_u(\mathbf{x}')\,\mathrm{d}\mathbf{x}'$ is the normalizing constant. It follows that

$$\pi(\mathbf{x}) = \pi(x_i|\mathbf{x}_{-i})\pi(\mathbf{x}_{-i}),$$

which is just the standard Bayes' rule. Moreover,

$$\pi_u(\mathbf{x}) = C\pi(\mathbf{x}) = C\pi(x_i|\mathbf{x}_{-i})\pi(\mathbf{x}_{-i}) = \pi(x_i|\mathbf{x}_{-i})\pi_u(\mathbf{x}_{-i}).$$

- The ith component Gibbs sampler is a kernel k_i that leaves all components besides x_i unchanged and replaces x_i by a sample drawn from the above expression. In other words, the proposal distribution $q_i(\mathbf{x}, \mathbf{y})$ is defined only when $\mathbf{x}_{-i} = \mathbf{y}_{-i}$, and it is given by

$$q(\mathbf{x}, \mathbf{y}) = \pi(y_i|\mathbf{x}_{-i}).$$

- When used with the Metropolis–Hastings algorithm, the acceptance probability is

$$\begin{aligned}\alpha_i(\mathbf{x}, \mathbf{y}) &= \min\left(1, \frac{\pi_u(\mathbf{y})q_i(\mathbf{y}, \mathbf{x})}{\pi_u(\mathbf{x})q_i(\mathbf{x}, \mathbf{y})}\right) = \min\left(1, \frac{\pi_u(\mathbf{y})\pi(x_i|\mathbf{y}_{-i})}{\pi_u(\mathbf{x})\pi(y_i|\mathbf{x}_{-i})}\right) \\ &= \min\left(1, \frac{\pi_u(\mathbf{y}_{-i})\pi(y_i|\mathbf{y}_{-i})\pi(x_i|\mathbf{y}_{-i})}{\pi_u(\mathbf{x}_{-i})\pi(x_i|\mathbf{x}_{-i})\pi(y_i|\mathbf{x}_{-i})}\right)\end{aligned}$$

  Again, this is defined only when $\mathbf{x}_{-i} = \mathbf{y}_{-i}$. So,

$$\alpha_i(\mathbf{x}, \mathbf{y}) = \min\left(1, \frac{\pi_u(\mathbf{x}_{-i})\pi(y_i|\mathbf{x}_{-i})\pi(x_i|\mathbf{x}_{-i})}{\pi_u(\mathbf{x}_{-i})\pi(x_i|\mathbf{x}_{-i})\pi(y_i|\mathbf{x}_{-i})}\right) = 1.$$

  This means that $k_i(\mathbf{x}, \mathbf{y}) = q_i(\mathbf{x}, \mathbf{y}) = \pi(y_i|\mathbf{x}_{-i})$ given that $\mathbf{x}_{-i} = \mathbf{y}_{-i}$.

- With the above property, we have that, if $\mathbf{x}_{-i} = \mathbf{y}_{-i}$,

$$\begin{aligned}\pi(\mathbf{x})k_i(\mathbf{x}, \mathbf{y}) &= \pi(x_i|\mathbf{x}_{-i})\pi(\mathbf{x}_{-i})\pi(y_i|\mathbf{x}_{-i}) = \pi(x_i|\mathbf{y}_{-i})\pi(\mathbf{y}_{-i})\pi(y_i|\mathbf{y}_{-i}) = \pi(x_i|\mathbf{y}_{-i})\pi(\mathbf{y}) \\ &= \pi(\mathbf{y})k_i(\mathbf{y}, \mathbf{x}).\end{aligned}$$

  So, the kernel k_i is reversiable with respect to the conditional probability distribution $\pi(\mathbf{x}|\mathbf{x}_{-i})$.

- The above derivation implies that k_1, k_2, …, k_d can be combined to obtain a kernel k that has π as the stationary distribution. There are at least two ways to combine them.
  - The **deterministic-scan Gibbs sampler** is

$$k = k_1 k_2 \cdots k_d.$$

  - The **random-scan Gibbs sampler** is

$$k = \frac{k_1 + k_2 + \cdots + k_d}{d}.$$

  We note that the random-scan Gibbs sampler is reversible, but the deterministic-scan one is not. Moreover, for the deterministic scan one, X_n is usable only when d divides n.

5 Hamiltonian Monte Carlo

- Hamiltonian Monte Carlo (HMC) is an algorithm that uses the gradient of the target distribution to create proposals, which allows a new proposal Y_{n+1} to be far away from X_n while still attaining high acceptance probability.
- The algorithm uses ideas from analytical mechanics and statistical physics.

5.1 Hamiltonian Dynamics

- In analytical mechanics, the state of a system can be described by its **position vector** $\mathbf{x} \in \mathbb{R}^d$ and its **momentum vector** $\mathbf{p} \in \mathbb{R}^d$.
- The set of all $(\mathbf{x}, \mathbf{p})$ pairs are called the **phase space**.
- The **Hamiltonian**, denoted by $H(\mathbf{x}, \mathbf{p})$, is a scalar function of $\mathbf{x}$ and $\mathbf{p}$. It is normally defined to be the total energy of the system: the sum of **potential energy** $U(\mathbf{x}, \mathbf{p})$ and **kinetic energy** $K(\mathbf{x}, \mathbf{p})$.

$$H(\mathbf{x}, \mathbf{p}) = U(\mathbf{x}, \mathbf{p}) + K(\mathbf{x}, \mathbf{p}).$$

- However, for our use case, we shall require that the potential energy depends only on $\mathbf{x}$, so we can write $U(\mathbf{x})$ instead of $U(\mathbf{x}, \mathbf{p})$.
- We will also use a specific form of kinetic energy

$$K(\mathbf{x}, \mathbf{p}) = K(\mathbf{p}) = \frac{1}{2}\mathbf{p}^T M^{-1}\mathbf{p}$$

  where M is a symmetric positive-definite matrix called the **mass matrix**. The mass matrix is typically diagonal and a scalar multiple of the identity matrix: $M = mI$. In such a case, we have that

$$K(\mathbf{p}) = \frac{\|\mathbf{p}\|^2}{2m},$$

  which agrees with the standard definition of kinetic energy in physics (i.e., $K = m\|\mathbf{v}\|^2/2$).
- Given the Hamiltonian, the dynamics of the system is determined by **Hamilton's equations**:

$$\frac{\mathrm{d}x_i}{\mathrm{d}t} = \frac{\partial H}{\partial p_i},$$
$$\frac{\mathrm{d}p_i}{\mathrm{d}t} = -\frac{\partial H}{\partial x_i}.$$

  Hamilton's equations are equivalent to Newton's laws, given that the energies included in the Hamiltonian account for all the forces in system.
- Plugging in $H(\mathbf{x}, \mathbf{p}) = U(\mathbf{x}) + \|\mathbf{p}^2\|/(2m)$, we have

$$\frac{\mathrm{d}x_i}{\mathrm{d}t} = \frac{p_i}{m},$$
$$\frac{\mathrm{d}p_i}{\mathrm{d}t} = -\frac{\partial U}{\partial x_i}.$$

- For any $t \geq 0$, Hamilton's equations give rise to the "time evolution" function $\mathcal{E}_t : \mathbb{R}^{2d} \to \mathbb{R}^{2d}$ where $\mathcal{E}_t((\mathbf{x}, \mathbf{p}))$ gives the state of the system after letting it evolve according to Hamilton's equations for duration t, starting at state $(\mathbf{x}, \mathbf{p})$.
- Hamiltonian dynamics has several nice properties.
  - It is reversible in the sense that

$$\mathcal{E}_t((\mathbf{x}, \mathbf{p})) = (\mathbf{x}', \mathbf{p}') \iff \mathcal{E}_t((\mathbf{x}', -\mathbf{p}')) = (\mathbf{x}, -\mathbf{p}).$$

    In other words, we can run the dynamics backward by starting at the final position and negating the momentum.

* This property implies that the time evolution function $\mathcal{E}_t$ for any fixed t is a bijection. The inverse function is given by

$$\mathcal{E}_t^{-1}((\mathbf{x}, \mathbf{p})) = \text{flip}(\mathcal{E}_t(\text{flip}((\mathbf{x}, \mathbf{p}))))$$

where flip is a function that negates the momentum of the input: $\text{flip}(\mathbf{x}, \mathbf{p}) = (\mathbf{x}, -\mathbf{p})$.

– The Hamiltonian does not change in time as the result of the dynamics. In other words,

$$\mathcal{E}_t((\mathbf{x}, \mathbf{p})) = (\mathbf{x}', \mathbf{p}') \implies H(\mathbf{x}, \mathbf{p}) = H(\mathbf{x}', \mathbf{p}').$$

If we imagine the phase space as being partition into level sets of H, we have that Hamiltonian dynamics preserves them: a point is always mapped to another point in the same level set.

– It perserves volume in phase space. In other words, A and B be Lebesgue-measurable set in the phase space, and let v denote the Lebesgue (i.e., volume) measure. Then, by abusing the notation a little, we have that

$$\mathcal{E}_t(A) = B \implies v(A) = v(B).$$

* This property can be restated as

$$\det(D\mathcal{E}_t) = 1$$

where D is the derivative opeartor, and so $D\mathcal{E}_t$ is the Jacobian of $\mathcal{E}_t$.

- The last property has an important implication when we use Hamiltonian dynamics as a way to create proposals in the Metropolis–Hastings algorithm.
- To discuss the above implication, however, let us recall a fact regarding transformations and probability densities. Let π be a probability density defined on the phase space, and let $f : \mathbb{R}^{2d} \times \mathbb{R}^{2d}$ be a bijection on the phase space. Consider the following random process:
  – Sample $(\mathbf{x}, \mathbf{p})$ according to π.
  – Compute $(\mathbf{x}', \mathbf{p}') \leftarrow f((\mathbf{x}, \mathbf{p}))$.

  If we let π' denote the probability density of the resulting $(\mathbf{x}', \mathbf{p}')$, it follows that

$$\pi'((\mathbf{x}', \mathbf{p}')) = \frac{\pi((\mathbf{x}, \mathbf{p}))}{|\det(Df((\mathbf{x}, \mathbf{p})))|}.$$

  You can find more details about this at [Khungurn, 2022b].
- Now, if we replace the bijection f with $\mathcal{E}_t$, we have that $\pi'((\mathbf{x}', \mathbf{p}')) = \pi(\mathbf{x}, \mathbf{p})$ because $\det(D\mathcal{E}_t) = 1$. In other words, when we transform a point in phase space using time evolution according to Hamiltonian dynamics, we do not need to include any correction factors in our calculation.

5.2 The HMC Algorithm

- The HMC algorithm uses time evolution according to Hamiltonian dynamics to create proposals in the Metropolis–Hastings algorithm. In other words, we want to use $\mathcal{E}_t$ to construct the proposal distribution q.
- However, the construction is not that straightforward. To do so, we must solve the following problems.
  (a) Recall that we are given only π_u, which is a function of $\mathbf{x}$. How should we define $\mathbf{p}$? Also, what would be the probability distribution of $(\mathbf{x}, \mathbf{p})$?
  (b) How exactly should we define the proposal distribution $q(\cdot, \cdot)$?
  (c) How do we compute $\mathcal{E}_t$? In other words, how do we simulate Hamiltonian dynamics?

5.2.1 Where Does the Momentum Come From?

- To design an MCMC algorithm, we first imagine the stationary distribution that we want the Markov chain to be converge to.
- Here, we borrow an idea from statistical mechanics. When a system with state $\mathbf{x}$ of energy $E(\mathbf{x})$ is connected to a heat bath of temperature T, the probability density of observing a state $\mathbf{x}$ is given by the **canonical distribution**

$$\pi(\mathbf{x}) = \frac{1}{Z} \exp\left(\frac{-E(\mathbf{x})}{k_B T}\right)$$

  where k_B is the Boltzmann's constant, and $\mathbf{Z}$ is the partition function

$$Z = \int \exp\left(\frac{-E(\mathbf{x})}{k_B T}\right) \mathrm{d}\mathbf{x},$$

  which is a constant needed to normalize the probability density so that it integrates to 1 over the domain.
- For simplicty, let us assume that $k_B T = 1$ from now on.
- The Hamiltonian $H(\mathbf{x}, \mathbf{p})$ is the total energy of the system. So, we can just plug it into the canonical distribution to get a joint probability distributon of $\mathbf{x}$ and $\mathbf{p}$.

$$\pi(\mathbf{x}, \mathbf{p}) = \frac{1}{Z} \exp(-H(\mathbf{x}, \mathbf{p}))$$

- Taking $H(\mathbf{x}, \mathbf{p}) = U(\mathbf{x}) + K(\mathbf{p})$, we have that

$$\pi(\mathbf{x}, \mathbf{p}) = \frac{1}{Z} \exp(-U(\mathbf{x})) \exp(-K(\mathbf{p})). \tag{2}$$

  We can see from this that the marginal probability distributions for $\mathbf{x}$ and $\mathbf{p}$ are independent.
- What should $U(\mathbf{x})$ be then? Recall that we are interested in sampling from a probability distribution $\pi(\mathbf{x})$ that is proportional to $\pi_u(\mathbf{x})$. We can set the potential energy $U(\mathbf{x})$ to be $-\log \pi_u(\mathbf{x})$, which gives

$$\exp(-U(\mathbf{x})) = \exp(-(-\log \pi(\mathbf{x}))) = \pi_u(\mathbf{x}).$$

  Now, we can see that the marginal distribution of $\mathbf{x}$ would be proportional to $\pi_u(\mathbf{x})$, which means that it would be exactly $\pi(\mathbf{x})$, our target distribution.
- Equation (2) also gives us a hint where $\mathbf{p}$ should come from: we can just sample $\mathbf{p}$ out of nowhere because $\mathbf{x}$ and $\mathbf{p}$ are independent of each other!
  - This is possible because there's nothing in our problem statement that tells us what the distribution of $\mathbf{p}$ should be, so we can choose a distribution that is easy to sample from.
  - So, we can pick $K(\mathbf{p}) = \|\mathbf{p}\|^2/2m$, which makes that the marginal distribution of $\mathbf{p}$ the Gaussian distribution $\mathcal{N}(\mathbf{0}, mI)$.
- Moreover, because $\mathbf{x}$ and $\mathbf{p}$ are independent, we can drop $\mathbf{p}$ at any time, and we would not lose any information on the distribution of $\mathbf{x}$.
- The above reasoning suggests an MCMC algorithm of the following form.
  - Start with an arbitrary position $\mathbf{x}_0$.

- **for** $n = 0, 1, 2, \ldots$ **do**
  - Sample $\mathbf{p}_n$ according to $\mathcal{N}(0, mI)$.
  - Transition $(\mathbf{x}_n, \mathbf{p}_n)$ to $(\mathbf{x}'_n, \mathbf{p}'_n)$ in such a way that satisfies detailed balance with the canonical distribution (2).
  - Set $\mathbf{x}_{n+1} \leftarrow \mathbf{x}'_n$.

  end for

- We can see that the Markov chain above would have the target distribution $\pi(\mathbf{x})$ as the stationary distribution because the transition $(\mathbf{x}_n, \mathbf{p}_n) \to (\mathbf{x}'_n, \mathbf{p}'_n)$ would preserve $\pi(\mathbf{x}, \mathbf{p})$.
- Moreover, because we sample a new momentum in every step and because $\mathbf{N}(0, mI)$ has infinite support, we can say that the Markov chain is irreducible and aperiodic with respect to $\mathbf{p}$, which is a good thing.
  - On the other hand, if we decide to keep maintain $\mathbf{p}_n$ along with $\mathbf{x}_n$, we would have to create some rules to let $\mathbf{p}_n$ vary through all of its domain, which would make the algorithm somewhat more complicated.
- Note that our MCMC algorithm has an interesting behavior. Our Markov chain generate points $\mathbf{x}_0, \mathbf{x}_1, \ldots$ in position space. However, in order to generate the next position, we "lift" the position to a random point in phase space. We then transition to another point in the phase space and then "drop" the point down to position space again.

5.2.2 How to Define the Proposal Distribution

- Given a position $(\mathbf{x}, \mathbf{p})$ in phase space, we need to figure out how to transition it to another point $(\mathbf{x}', \mathbf{p}')$ in such a way that satisfies detailed balance with the canonical distribution

$$H(\mathbf{x}, \mathbf{p}) = \frac{1}{Z} \exp(-U(\mathbf{x})) \exp(-V(\mathbf{p})) = \frac{1}{Z} \pi_u(\mathbf{x}) \exp\left(\frac{-\|\mathbf{p}\|^2}{2m}\right).$$

- To do so, we employ the Metropolis–Hastings proposal+acceptance step.
- The idea of HMC is to fix a duration t and propose $(\mathbf{x}', \mathbf{p}') = \mathcal{E}_t((\mathbf{x}, \mathbf{p}))$.
- Because $\mathcal{E}_t$ is deterministic, we have that the transition kernel would be given by

$$q((\mathbf{x}, \mathbf{p}), (\mathbf{x}', \mathbf{p}')) = \delta((\mathbf{x}', \mathbf{p}') - \mathcal{E}_t((\mathbf{x}, \mathbf{p})))$$

  where $\delta(\cdot)$ is Dirac's delta function. Note that there's no correction term on the RHS. This is because $\mathcal{E}_t$ preserves volume in phase space.

- However, this turns out to not be a good idea. Let's look at the acceptance probability

$$\alpha((\mathbf{x}, \mathbf{p}), (\mathbf{x}', \mathbf{p}')) = \frac{H(\mathbf{x}', \mathbf{p}') q((\mathbf{x}', \mathbf{p}'), (\mathbf{x}, \mathbf{p}))}{H(\mathbf{x}, \mathbf{p}) q((\mathbf{x}, \mathbf{p}), (\mathbf{x}', \mathbf{p}'))}$$

  Taking $(\mathbf{x}', \mathbf{p}') = \mathcal{E}_t((\mathbf{x}, \mathbf{p}))$, we would have that the nominator would almost always be 0 because we cannot expect that we would arrive back at $(\mathbf{x}, \mathbf{p})$ if we start simulating the dynamics from $(\mathbf{x}', \mathbf{p}')$. So, the Metropolis–Hastings algorithm would not accept any proposal!

- To solve the above problem, we propose

$$(\mathbf{x}', \mathbf{p}') = \text{flip}(\mathcal{E}_t((\mathbf{x}, \mathbf{p}))).$$

  In other words, we run the Hamiltonian dynamics, and we simply negate the momentum after we finish.

- The great thing about the above proposal is that, if we take the end result of the above process $(\mathbf{x}', \mathbf{p}')$ and try to get a proposal out of it, we come back to the starting point $(\mathbf{x}, \mathbf{p})$! Symbolically,

$$(\mathbf{x}', \mathbf{p}') = \text{flip}(\mathcal{E}_t((\mathbf{x}, \mathbf{p}))) \implies (\mathbf{x}, \mathbf{p}) = \text{flip}(\mathcal{E}_t((\mathbf{x}', \mathbf{p}'))). \tag{3}$$

  This is the case because

$$\mathcal{E}_t^{-1}(\cdot) = \text{flip}(\mathcal{E}_t(\text{flip}(\cdot))).$$

  So,

$$\text{flip}(\mathcal{E}_t((\mathbf{x}', \mathbf{p}'))) = \text{flip}(\mathcal{E}_t(\text{flip}(\mathcal{E}_t((\mathbf{x}, \mathbf{p}))))) = \mathcal{E}_t^{-1}(\mathcal{E}_t((\mathbf{x}, \mathbf{p}))) = (\mathbf{x}, \mathbf{p}).$$

- The transition kernel for $\text{flip}(\mathcal{E}_t((\mathbf{x}, \mathbf{p})))$ is

$$q((\mathbf{x}, \mathbf{p}), (\mathbf{x}', \mathbf{p}')) = \delta((\mathbf{x}', \mathbf{p}') - \text{flip}(\mathcal{E}_t((\mathbf{x}, \mathbf{p})))).$$

  Note that there's no scalar correction factor here either. This is because the Jacobian of $\text{flip}(\cdot)$ has determinant $(-1)^d$, so the absolute value is 1.

- Now, the acceptance probability becomes

$$\begin{aligned}
\alpha((\mathbf{x}, \mathbf{p}), (\mathbf{x}', \mathbf{p}')) &= \min\left(1, \frac{H(\mathbf{x}', \mathbf{p}')q((\mathbf{x}', \mathbf{p}'), (\mathbf{x}, \mathbf{p}))}{H(\mathbf{x}, \mathbf{p})q((\mathbf{x}, \mathbf{p}), (\mathbf{x}', \mathbf{p}'))}\right) \\
&= \min\left(1, \frac{H(\mathbf{x}', \mathbf{p}')\delta((\mathbf{x}', \mathbf{p}') - \text{flip}(\mathcal{E}_t((\mathbf{x}, \mathbf{p}))))}{H(\mathbf{x}, \mathbf{p})\delta((\mathbf{x}, \mathbf{p}) - \text{flip}(\mathcal{E}_t((\mathbf{x}', \mathbf{p}'))))}\right) \\
&= \min\left(1, \frac{H(\mathbf{x}', \mathbf{p}')}{H(\mathbf{x}, \mathbf{p})}\right).
\end{aligned}$$

  Here, the last line follows from (3).

- If we can compute $\mathcal{E}_t$ perfectly, the acceptance probability would be 1 because the Hamiltonian would be perfectly preserved. However, this does not happen in real computation, so we keep the H terms there.

- So, now, the algorithm is as follows.
  - Start the simulation from some $\mathbf{x}_0$.
  - **for** $n = 0, 1, 2, \ldots$ **do**
    - Sample $\mathbf{p}_n$ from $\mathcal{N}(\mathbf{0}, mI)$.
    - Starting from the phase space point $(\mathbf{x}_n, \mathbf{p}_n)$, simulate the for a fixed time t to obtain $(\mathbf{x}_n^*, \mathbf{p}_n^*) = \mathcal{E}_t((\mathbf{x}_n, \mathbf{p}_n))$.
    - Flip the momentum by setting

$$(\mathbf{x}_n', \mathbf{p}_n') \leftarrow (\mathbf{x}_n^*, -\mathbf{p}_n^*).$$

    - Compute the acceptance propability

$$\alpha(\mathbf{x}_n, \mathbf{x}_n') = \min\left(1, \frac{\exp(-H(\mathbf{x}_n', \mathbf{p}_n'))}{\exp(-H(\mathbf{x}_n, \mathbf{p}_n))}\right)$$

    - With probability $\alpha(\mathbf{x}_n, \mathbf{x}_n')$, set $\mathbf{x}_{n+1} \leftarrow \mathbf{x}_n'$. Otherwise, set $\mathbf{x}_{n+1} \leftarrow \mathbf{x}_n$.

  end for

5.2.3 How to Simulate Hamiltonian Dynamics

- We now have to figure out how to simulate Hamiltonian dynamics. There are many ways to do so, including Euler's method, midpoint method, and Runge–Kutta method.
- The HMC algorithm as discussed in [Neal, 2011] uses an integration scheme called **leapfrog** to simulate the Hamiltonian dynamics.
- The leapfrog method has several nice properties that make it very suitable as a component of an MCMC algorithm. More specifically, let $\mathcal{L}_{\Delta t}((\mathbf{x}, \mathbf{p}))$ denote the result of running the leapfrog method starting from state space point $(\mathbf{x}, \mathbf{p})$ for a single time step Δt. We have that:
  - The leapfrog method is a second-order method, meaning that, if we simulate the dynamics with step size Δt, the error would be of order $(\Delta t)^2$. In other words,
  $$\mathcal{L}_{\Delta t}((\mathbf{x}, \mathbf{p})) = \mathcal{E}_{\Delta t}((\mathbf{x}, \mathbf{p})) + O((\Delta t)^2).$$
  - It is reversible in time in the same exact sense that Hamiltonian dynamics is reversible.
  $$\mathcal{L}_{\Delta t}((\mathbf{x}, \mathbf{p})) = (\mathbf{x}', \mathbf{p}') \implies \mathcal{L}_{\Delta t}((\mathbf{x}', -\mathbf{p}')) = (\mathbf{x}, -\mathbf{p}).$$
  - It also preserves volume in the phase space.
  $$\mathcal{L}_{\Delta t}(A) = B \implies v(A) = v(B).$$

  See [Young, 2014] for justification of these properties.
- The last two properties allow us to show that the transition kernel q defined by $\mathcal{L}_{\Delta t}$ instead of $\mathcal{E}_{\Delta t}$ satisfies the property
  $$q((\mathbf{x}, \mathbf{p}), (\mathbf{x}', \mathbf{p}')) = \delta((\mathbf{x}', \mathbf{p}') - \text{flip}(\mathcal{L}_{\Delta t}((\mathbf{x}, \mathbf{p})))).$$
  In other words, the RHS is just the Dirac's delta function without any scalar correction factor. Hence, the acceptance probability is simply
  $$\alpha((\mathbf{x}, \mathbf{p}), (\mathbf{x}', \mathbf{p}')) = \min\left(1, \frac{H(\mathbf{x}', \mathbf{p}')}{H(\mathbf{x}, \mathbf{p})}\right)$$
- Here, we describe the leapfrog algorithm, running for one timestep of duration Δt. The input is $(\mathbf{x}_0, \mathbf{p}_0)$, and the output will be $(\mathbf{x}_1, \mathbf{p})$. The algorithm would first calculate the half way momentum, use it to compute $\mathbf{x}_1$, and then compute the momentum $\mathbf{p}_1$. Assuming that $H(\mathbf{x}, \mathbf{p}) = U(\mathbf{x}) + K(\mathbf{p})$, the algorithm goes as follow.
  $$\begin{aligned} \mathbf{p}_{1/2} &\leftarrow \mathbf{p}_0 - \frac{\Delta t}{2}\frac{\partial U}{\partial \mathbf{x}}(\mathbf{x}_0) \\ \mathbf{x}_1 &\leftarrow \mathbf{x}_0 + \Delta t \frac{\partial K}{\partial \mathbf{p}}(\mathbf{p}_{1/2}) \\ \mathbf{p}_1 &\leftarrow \mathbf{p}_{1/2} - \frac{\Delta t}{2}\frac{\partial U}{\partial \mathbf{x}}(\mathbf{x}_1) \end{aligned}$$
- However, when running the leapfrog algorithm for multiple time steps, it is not necessary to calculate the momentum at interger indices except for the first and the last one.

  Start with input phase space point $(\mathbf{x}_0, \mathbf{p}_0)$.

  Compute $\mathbf{p}_{1/2} \leftarrow \mathbf{p}_0 - \frac{\Delta t}{2}\frac{\partial U}{\partial \mathbf{x}}(\mathbf{x}_0)$.

for $n = 0, 1, 2, \ldots, N-1$ **do**

Compute $\mathbf{x}_{n+1} \leftarrow \mathbf{x}_n + \Delta t \frac{\partial K}{\partial \mathbf{p}}(\mathbf{p}_{n+1/2})$.

Compute $\mathbf{p}_{n+3/2} \leftarrow \mathbf{p}_{n+1/2} - \Delta t \frac{\partial U}{\partial \mathbf{x}}(\mathbf{x}_{n+1})$.

end for

Compute $\mathbf{p}_N \leftarrow \mathbf{p}_{N+1/2} + \frac{\Delta t}{2} \frac{\partial U}{\partial \mathbf{x}}(\mathbf{x}_N)$

return $(\mathbf{x}_N, \mathbf{p}_N)$.

5.2.4 Putting It All Together

- The HMC algorithm is as follows.
  - Start the simulation from some $\mathbf{x}_0$.
  - **for** $n = 0, 1, 2, \ldots$ **do**
    - Sample $\mathbf{p}_n$ from $\mathcal{N}(\mathbf{0}, mI)$.
    - Starting from point $(\mathbf{x}_n, \mathbf{p}_n)$, simulate Hamiltonian dynamics with the leapfrog algorithm for N step of size Δt to obtain $(\mathbf{x}_n^*, \mathbf{p}_n^*)$.
    - Flip the momentum by setting

$$(\mathbf{x}_n', \mathbf{p}_n') \leftarrow (\mathbf{x}_n^*, -\mathbf{p}_n^*).$$

    - Compute the acceptance propability

$$\alpha(\mathbf{x}_n, \mathbf{x}_n') = \min\left(1, \frac{\exp(-H(\mathbf{x}_n', \mathbf{p}_n'))}{\exp(-H(\mathbf{x}_n, \mathbf{p}_n))}\right)$$

    - With probability $\alpha(\mathbf{x}_n, \mathbf{x}_n')$, set $\mathbf{x}_{n+1} \leftarrow \mathbf{x}_n'$. Otherwise, set $\mathbf{x}_{n+1} \leftarrow \mathbf{x}_n$.

  end for

- Conceptually, the Markov chain simulates movement of a particle. It above keeps the current position $\mathbf{x}_n$ of the particle in memory. To get to the text positive, it gives the partical a random kick $\mathbf{p}_n$ and simulate Hamiltonian dynamics to see where the particle lands after time $N\Delta t$.

6 Langevin Algorithm

- Previously in Section 2, we said that the Langevin algoirthm is a special case of the Metropolis–Hastings algorithm where the proposal $\mathbf{y}$ is given by

$$\mathbf{y} \sim \mathcal{N}\left(\mathbf{x} + \frac{\sigma^2}{2}\nabla \log \pi_u(\mathbf{x}), \sigma^2 I\right),$$

  which gives the proposal distribution

$$q(\mathbf{x}, \mathbf{y}) = \frac{1}{(\sqrt{2\pi}\sigma)^d} \exp\left(-\frac{\|\mathbf{y} - \mathbf{x} - \sigma^2 \nabla \log \pi_u(\mathbf{x})/2\|^2}{2\sigma^2}\right).$$

  We will look further into this algorithm in this section.

6.1 Connection to Hamiltonian Monte Carlo

- According to [Neal, 2011], the Langevin algorithm can be viewed as a special case of HMC where only one time step is taken when simulating Hamiltonian dynamics.
- To see this, set $U(\mathbf{x}) = -\log \pi_u(\mathbf{x})$ and $K(\mathbf{p}) = \|\mathbf{p}\|^2/2$ (i.e. $m = 1$). We have that

$$\frac{\partial U}{\partial \mathbf{x}} = -\nabla \log \pi_u(\mathbf{x})$$
$$\frac{\partial K}{\partial \mathbf{p}} = \mathbf{p}$$

  Let us run the leapfrog method for one iteration. We have that

$$\mathbf{p}_{1/2} = \mathbf{p}_0 + \frac{\Delta t}{2} \nabla \log \pi_u(\mathbf{x}_0)$$
$$\mathbf{x}_1 = \mathbf{x}_0 + \Delta t\, \mathbf{p}_{1/2} = \mathbf{x}_0 + \frac{(\Delta t)^2}{2} \nabla \log \pi_u(\mathbf{x}_0) + \Delta t\, \mathbf{p}_0$$

  Rewriting $\mathbf{x}_0$ as $\mathbf{x}$, $\mathbf{x}_1$ as $\mathbf{y}$, Δt as σ, we have that

$$\mathbf{y} = \mathbf{x} + \frac{\sigma^2}{2} \nabla \log \pi_u(\mathbf{x}) + \sigma \mathbf{p}_0. \tag{4}$$

  Moreover, if we take $\mathbf{p}_0$ so that it is distributed according to the standard normal distribution $\mathcal{N}(\mathbf{0}, I)$, then we have that

$$\mathbf{y} \sim \mathcal{N}\left(\mathbf{x} + \frac{\sigma^2}{2} \nabla \log \pi_u(\mathbf{x}), \sigma^2 I\right),$$

  which is exactly the proposal distribution of the Langevin dynamics algorithm.

- The difference between HMC and Langevin dynamics is when we calculate the acceptance propability. In HMC, we use the ratio

$$\alpha((\mathbf{x}, \mathbf{p}_0), (\mathbf{y}, \mathbf{p}_1)) = \min\left(1, \frac{\exp(-H(\mathbf{y}, \mathbf{p}_1))}{\exp(-H(\mathbf{x}, \mathbf{p}_0))}\right).$$

  However, in Langevin dynamics, the momentum does not exist, so we have to use the full Metropolis–Hastings acceptance probability

$$\alpha(\mathbf{x}, \mathbf{y}) = \min\left(1, \frac{\pi_u(\mathbf{y}) q(\mathbf{y}, \mathbf{x})}{\pi_u(\mathbf{x}) q(\mathbf{x}, \mathbf{y})}\right).$$

6.2 Langevin Equation

- Langevin dynamics gets its name from a stochastic differential equation (SDE) called the **Langevin equation**

$$\mathrm{d}\mathbf{x} = \frac{1}{2} \nabla \log \pi_u(\mathbf{x})\, \mathrm{d}t + \mathrm{d}\mathbf{W}.$$

  where $\mathbf{W}$ is the standard Brownian motion in $\mathbb{R}^d$. The initial condition is $\mathbf{x}(0) = \mathbf{x}_0$ where $\mathbf{x}_0$ is an input.

- The Langevin equation is related to the problem at hand because the "stationary distribution" of its solution is exactly the target distribution $\pi(\mathbf{x})$.

- More precisely, if the stochastic process $\{\mathbf{x}(t) : 0 \leq t \leq \infty\}$ is a solution of the Langevin equation, then the probability of density $\nu(\mathbf{x}, t) = \nu(\mathbf{x}(t))$ of $\mathbf{x}(t)$ must satisfy the **Fokker–Planck equation**:

$$\begin{aligned}\frac{\partial \nu(\mathbf{x}, t)}{\partial t} &= -\sum_{i=1}^{d} \frac{\partial}{\partial x_i}\left(\frac{1}{2}\frac{\partial \log \pi_u(\mathbf{x})}{\partial x_i}\nu(\mathbf{x}, t)\right) + \frac{1}{2}\sum_{i=1}^{d}\frac{\partial^2 \nu(\mathbf{x}, t)}{\partial x_i^2} \\ &= -\frac{1}{2}\sum_{i=1}^{d} \frac{\partial}{\partial x_i}\left(\frac{1}{\pi_u(\mathbf{x})}\frac{\partial \pi_u(\mathbf{x})}{\partial x_i}\nu(\mathbf{x}, t)\right) + \frac{1}{2}\sum_{i=1}^{d}\frac{\partial^2 \nu(\mathbf{x}, t)}{\partial x_i^2}.\end{aligned}$$

  (Again, see [Khungurn, 2022a] for more details.)

  A stationary distribution does not change in time. In other words, there is no dependence on t, and $\nu(\mathbf{x}, t) = \nu(\mathbf{x})$. The Fokker–Planck equation becomes

$$0 = -\frac{1}{2}\sum_{i=1}^{d} \frac{\partial}{\partial x_i}\left(\frac{1}{\pi_u(\mathbf{x})}\frac{\partial \pi_u(\mathbf{x})}{\partial x_i}\nu(\mathbf{x})\right) + \frac{1}{2}\sum_{i=1}^{d}\frac{\partial^2 \nu(\mathbf{x})}{\partial x_i^2}.$$

  We have that the target distributon $\pi(\mathbf{x}) = C\pi_u(\mathbf{x})$ satisfies it. This can be checked by plugging substituting $\nu(\mathbf{x})$ with $C\pi_u(\mathbf{x})$.

$$\begin{aligned}&-\frac{1}{2}\sum_{i=1}^{d} \frac{\partial}{\partial x_i}\left(\frac{1}{\pi_u(\mathbf{x})}\frac{\partial \pi_u(\mathbf{x})}{\partial x_i}C\pi_u(\mathbf{x})\right) + \frac{1}{2}\sum_{i=1}^{d}\frac{\partial^2 (C\pi_u(\mathbf{x}))}{\partial x_i^2} \\ &= -\frac{C}{2}\sum_{i=1}^{d} \frac{\partial}{\partial x_i}\left(\frac{\partial \pi_u(\mathbf{x})}{\partial x_i}\right) + \frac{C}{2}\sum_{i=1}^{d}\frac{\partial^2 \pi_u(\mathbf{x})}{\partial x_i^2} \\ &= -\frac{C}{2}\sum_{i=1}^{d} \frac{\partial^2 \pi_u(\mathbf{x})}{\partial x_i^2} + \frac{C}{2}\sum_{i=1}^{d}\frac{\partial^2 \pi_u(\mathbf{x})}{\partial x_i^2} \\ &= 0.\end{aligned}$$

- What this means is that, if we let $t \to \infty$, we would expect that the probability distribution $\nu(\mathbf{x}, t)$ would reach equilibrium and will not change in time. So, the distribution $\lim_{t\to\infty} \nu(\mathbf{x}, t)$ would be given by $\pi(\mathbf{x})$. (Note that this is not at all a rigourous argument.)
- Now that we have intuited that the stationary distribution of the dynamics described by the Langevin equation is $\pi(\mathbf{x})$, we would expect that the distribution of its discrete approximation would approach $\pi(\mathbf{x})$ as well if we do things the right way.

6.3 Langevin Algorithm as Discrete Approximation to Langevin Dynamics

- An approximate trajectory of the stochastic process $\{\mathbf{x}(t) : 0 \leq t \leq T\}$ satisfying the Langevin equation can be obtained by the Euler–Maruyama method. (See [Khungurn, 2022a] for more details.) If we take only one time step, the calculation goes as follows:
  1. Sample $\mathbf{W}_0$ from $\mathcal{N}(\mathbf{0}, T)$.
  2. Compute $\mathbf{x}_1 \leftarrow \mathbf{x}_0 + \frac{T}{2}\nabla \log \pi_u(\mathbf{x}_0) + \mathbf{W}_0$.

  In the above process, we have that $\mathbf{W}_0$ can be rewritten as $\sqrt{T}\boldsymbol{\xi}$ where $\boldsymbol{\xi} \sim \mathcal{N}(0, I)$, which means that

$$\mathbf{x}_1 = \mathbf{x}_0 + \frac{T}{2}\nabla \log \pi_u(\mathbf{x}_0) + \sqrt{T}\boldsymbol{\xi}.$$

  Notice that the above equation is exactly the same as Equation (4) if we substitute $\mathbf{x}_1$ with $\mathbf{y}$, $\mathbf{x}_0$ with $\mathbf{x}$, and T with σ^2. This shows that the Langevin algorithm is a discrete approximation of the dynamics governed by the Langevin equation.

- In [Roberts and Tweedie, 1996], Roberts and Tweedie analyzed convergence of of the Langevin algorithm, for which they distinguished between several variants.
  - The *unadjusted Langevin algorithm* (ULA) accepts the proposal
$$\mathbf{y} \sim \mathcal{N}\left(\mathbf{x} + \frac{\sigma^2}{2}\nabla \log \pi_u(\mathbf{x}), \sigma^2 I\right)$$
with probabilty one.
  - The *Metropolis-adjusted Langevin algorithm* (MALA) is where the proposal is accepted with the acceptance probability
$$\alpha(\mathbf{x}, \mathbf{y}) = \min\left(1, \frac{\pi_u(\mathbf{y}q(\mathbf{y}, \mathbf{x}))}{\pi_u(\mathbf{x})q(\mathbf{x}, \mathbf{y})}\right)$$
This is the algorithm we have been discussing all along.
  - The *Metropolis-adjusted Langevin truncated algorithm* (MALTA) is where the proposal is given by
$$\mathbf{y} \sim \mathcal{N}\left(\mathbf{x} + \frac{\delta}{\max(\delta, \|\nabla \log \pi_u(\mathbf{x})\|)}\nabla \log \pi_u(\mathbf{x}), \sigma^2 I\right)$$
for some fixed constant $\delta > 0$. Here, the displacement of $\mathbf{x}$ is bounded by a fixed distance δ. The proposal is accepted with probability $\alpha(\mathbf{x}, \mathbf{y})$ as in MALA.
- In their analysis, they also consider two notions of "ergodicity."
  - **Definition 1.** *We say that a Markov chain is* **ergodic** *if it is aperiodic and Harris-recurrent.*
  - **Definition 2.** *We say that a Markov chain* $\{\mathbf{x}_t : t = 0, 1, 2, \dots\}$ *with kernal transition function* K *is* **geometrically ergodic** *or* **exponentially ergodic** *if it converges to the stationary measure* Π *at geometric rate with multiplicative constant depending on the start point. In other words, there is a function* $V : \mathbb{R}^d \to \mathbb{R}$ *such that*
$$d_{\mathrm{TV}}(K^n(\mathbf{x}_0, \cdot), \Pi(\cdot)) \leq V(\mathbf{x}_0)\gamma^n$$
for some constant $0 < \gamma < 1$.
  - Note that we know that, for an ergodic Markov chain that has a stationary measure Π, the transition probabilities $K^n(\mathbf{x}_0, \cdot)$ will converge to the Π in the total variation norm.
$$\lim_{n\to\infty} d_{\mathrm{TV}}(K^n(\mathbf{x}_0, \cdot), \Pi(\cdot)) = 0.$$
Here, $K^n(\cdot, \cdot)$ denotes the kernel density of applying the Markov chain transition n times, and the probability measure Π is given by
$$\Pi(A) = \int_A \pi(\mathbf{x})\,\mathrm{d}\mathbf{x}.$$
  - We shall assume that all the Markov chains in this section (ULA, MALA, MALTA) has stationary distributions, which would be the target distribution Π. So, "ergodic" would mean "converges to staionary distribution."
  - On the other hand, exponential ergodicity is a much stronger notion than ergodicity. It means that the Markov chain converges exponentially fast to the target distribution.
- For ULA:

- When a ULA chain is ergodic, it generally converges to another distribution which is not the target distribution π.
- The paper proves results on convergence of the ULA chain in 1D. To do so, they make use of the following limits.

$$S_\alpha^+ := \lim_{x\to\infty} \frac{\sigma^2}{2x^\alpha} \frac{\mathrm{d}(\log \pi_u(x))}{\mathrm{d}x}$$
$$S_\alpha^- := \lim_{x\to-\infty} \frac{\sigma^2}{2|x|^\alpha} \frac{\mathrm{d}(\log \pi_u(x))}{\mathrm{d}x}.$$

- **Theorem 3.** *A ULA chain on $\mathbb{R}$ is exponentially ergodic if one of the following conditions hold:*
  - *For some $\alpha \in [0,1)$, both S_α^+ and S_α^- exist with $S_\alpha^+ < 0$ and $S_\alpha^- > 0$.*
  - *For $\alpha = 1$, both S_α^+ and S_α^- exist with $S_\alpha^+ < 0$, $S_\alpha^- > 0$, and $(1 + S_\alpha^+)(1 - S_\alpha^-) < 1$.*
- **Theorem 4.** *A ULA chain on $\mathbb{R}$ is ergodic but not exponentially ergodic if, for some $\alpha \in (-1, 0)$, both S_α^+ and S_α^- exist with $S_\alpha^+ < 0$ and ($S_\alpha^- > 0$.*
- **Theorem 5.** *A ULA chain on $\mathbb{R}$ not ergodic and even transient if one of the following conditions hold.*
  - *For some $\alpha > 1$, both S_α^+ and S_α^- exist with $S_\alpha^+ < 0$, and $S_\alpha^- > 0$.*
  - *For $\alpha = 1$, both S_α^+ and S_α^- exist with $S_\alpha^+ < -2$, and $S_\alpha^- > 2$.*

- For MALA:
  - Let $A(\mathbf{x})$ denote the "acceptance region" of $\mathbf{x}$. In other words, $A(\mathbf{x})$ is the set of proposal $\mathbf{y}$ that is always accepted. In other words,

$$A(\mathbf{x}) = \{\mathbf{y} : \pi_u(\mathbf{y})q(\mathbf{y}, \mathbf{x}) \geq \pi_u(\mathbf{x})q(\mathbf{x}, \mathbf{y})\}$$

  - We say that $A(\cdot)$ **converges inwards in** q if

$$\lim_{\mathbf{x}\to\infty} \int_{A(\mathbf{x})-I(\mathbf{x})} q(\mathbf{x}, \mathbf{y})\, \mathrm{d}\mathbf{y} = 0$$

    where $I(\mathbf{x}) = \{\mathbf{y} : \|\mathbf{y}\| \leq \mathbf{x}\}$.
  - **Theorem 6.** *Let $\mathbf{c}(\mathbf{x}) = \mathbf{x} + \sigma^2 \nabla \log \pi_u(\mathbf{x})/2$ denote the "next candidate position." Let*

$$\eta = \liminf_{\|\mathbf{x}\|\to\infty} (\|\mathbf{x}\| - \|\mathbf{c}(\mathbf{x})\|).$$

    If $\eta > 0$ and $A(\cdot)$ converges inwards in q, then the MALA chain is exponentially ergodic with $V(\mathbf{x}) = e^{s\|\mathbf{x}\|}$ for some $s < 2\eta\sigma^2$.
  - The above result is hard to interpret due to the cryptic way the conditionas are written down. Intuitively though, we can say that, if $\pi_u(\mathbf{x}) \to 0$ fast as $\|\mathbf{x}\| \to \infty$, then the MALA chain is exponentially ergodic.
  - On the other hand, when $\mathbf{x}$ has large tails, then the MALA chain is not exponentially ergodic.
  - **Theorem 7.** *If π_u is bounded and*

$$\liminf_{\|\mathbf{x}\|\to\infty} \frac{\|\nabla \log \pi_u(\mathbf{x})\|}{\|\mathbf{x}\|} > \frac{4}{\sigma^2},$$

    then the MALA chain is not exponentially ergodic.
  - **Theorem 8.** *If $\|\nabla \log \pi_u(\mathbf{x})\| \to 0$ as $\|\mathbf{x}\| \to \infty$, then the MALA chain is not geometricaly ergodic.*

- For MALTA, the authors mentione that its exponential ergodicity is much more robust than that of MALA, but they did not give much details in the paper.

References

[Andrieu et al., 2003] Andrieu, C., de Freitas, N., Doucet, A., and Jordan, M. I. (2003). An introduction to MCMC for machine learning. *Machine Learning*, 50(1–2):5–43.

[Khungurn, 2022a] Khungurn, P. (2022a). A primer on stochastic differential equations. `https://pkhungurn.github.io/notes/notes/math/sde-primer/sde-primer.pdf`. Accessed: 2022-02-06.

[Khungurn, 2022b] Khungurn, P. (2022b). Probability density under transformation. `https://pkhungurn.github.io/notes/notes/gfx/pdf-transform/pdf-transform.pdf`. Accessed: 2022-02-01.

[Neal, 2011] Neal, R. M. (2011). MCMC using Hamiltonian dynamics. *Handbook of Markov Chain Monte Carlo*, 54:113–162.

[Roberts and Rosenthal, 2004] Roberts, G. O. and Rosenthal, J. S. (2004). General state space Markov chains and MCMC algorithms. *Probability Surveys*, 1(none).

[Roberts and Tweedie, 1996] Roberts, G. O. and Tweedie, R. L. (1996). Exponential convergence of Langevin distributions and their discrete approximations. *Bernoulli*, 2(4):341–363.

[Young, 2014] Young, P. (2014). The leapfrog method and other "symplectic" algorithms for integrating Newton's laws of motion. `https://young.physics.ucsc.edu/115/leapfrog.pdf`. Accessed: 2022-02-04.