Lecture 3: Markov Chains (II): Detailed Balance, and Markov Chain Monte Carlo (MCMC)

Readings

Recommended:

- Grimmett and Stirzaker (2001) 6.5 (Detailed balance), 6.14 (MCMC).

Optional:

- Sokal (1989). A classic manuscript on Monte Carlo methods.
- Diaconis (2009). An introduction to MCMC methods, that particularly discusses some of the theoretical tools available to analyze them.
- Madras (2002). A short, classic set of notes on Monte Carlo methods.

In this lecture we'll consider exclusively a time-homogeneous Markov chain with transition matrix P, and a finite state space, $|S| = N$.

3.1 Detailed balance

Detailed balance is an important property of certain Markov Chains that is widely used in physics and statistics.

Definition. Let $X_0, X_1, \ldots$ be a Markov chain with stationary distribution π. The chain is said to be *reversible with respect to* π or to satisfy *detailed balance with respect to* π if

$$\pi_i P_{ij} = \pi_j P_{ji} \qquad \forall i, j \in S. \tag{1}$$

Equations (1) are called the *detailed balance equations*.

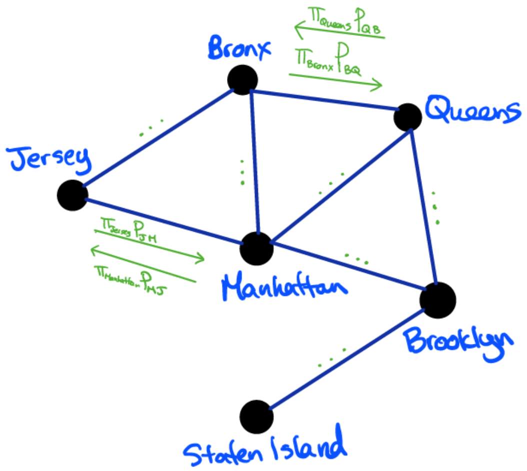
What do equations (1) represent physically? Suppose we start a chain in the stationary distribution, so that $X_0 \sim \pi$. Then $\pi_i P_{ij}$ is the amount of probability that flows down edge $i \to j$ in one time step. If (1) holds, then the amount of probability flowing from i to j, equals the amount that flows from j to i. Therefore, there is no *net* flux of probability along the edge $i \leftrightarrow j$ during one time step.

Compare (1) with the condition for a stationary distribution:

$$\pi_i = \sum_j \pi_j P_{ji}\,, \qquad \forall i \in S. \tag{2}$$

Equations (2) require solving a system of N equations, whereas the detailed balance equations (1) require solving a system of N^2 equations. Therefore detailed balance is a much *stronger* condition than the condition that π be a stationary distribution; a general Markov chain won't satisfy detailed balance. Indeed, equations (2) for the stationary distribution say that the flow of probability is balanced globally: at each *node*, the amount of probability flowing in in one step, equals the amount flowing out. The detailed balance equations say the flow of probability is balanced locally: at each *edge*, the amount of probability that flows across in one direction in one step, equals the amount that flows in the opposite direction.

A nice analogy[1] comes from thinking about traffic flow in New York City and surroundings: let each borough or suburb be represented by a node of a Markov chain, and join two nodes if there is a road, bridge, or tunnel connecting the boroughs directly. For example, the node corresponding to Manhattan would be connected to Jersey City (via the Holland tunnel), to Weekawken (via the Lincoln tunnel), to Fort Lee (via the George Washington bridge), to Queens (via the Queensboro bridge), etc:

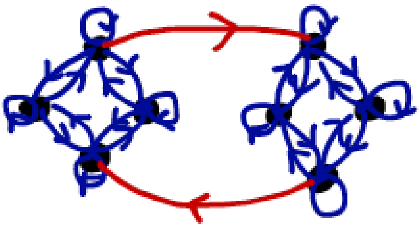


Suppose that cars driving around represent little elements of probability, and the transition matrix describes what fraction of cars cross from one borough to another each day. The traffic is in the stationary distribution, if the number of cars in Manhattan, and in all other nodes, doesn't change each day. This is possible even when cars are constantly driving around in circles, such as if all cars leave Manhattan across the Holland tunnel, drive up to Fort Lee, and enter Manhattan via the George Washington bridge. As long as the number of cars per unit time leaving Manhattan across the Holland tunnel, equals the number per unit time entering Manhattan via the George Washington bridge, the number of cars in Manhattan doesn't change, and the traffic can be in the stationary distribution.

The traffic is in detailed balance, only if the number of cars that leaves *each bridge or each tunnel* per unit time, equals the number that enter that *same* bridge or tunnel. For example, the flux of cars entering Manhattan through the Holland tunnel, must equal the flux of cars exiting Manhattan through the Holland tunnel, and similarly for every other bridge or tunnel. Not only does the number of cars in Manhattan remain constant with time, but the fluxes of cars across each single bridge or tunnel separately must be equal in each direction.

Sometimes we can tell directly from the graph structure of a Markov chain, whether it is possible for it to be in detailed balance. Here is a chain that does not satisfy detailed balance, no matter what probabilities are put on the (directed) edges:

[1] Thanks to Oliver Bühler for this idea.



There must be a net flux of probability through the red edges, because probability cannot move along them in the opposite direction.

Detailed balance is a powerful concept. It gives a tractable way to find a stationary distribution, and, it allows us to construct a system has a desired stationary distribution.

Theorem. *Let P be the transition matrix of a Markov chain X_n, and suppose there exists a distribution π such that $\pi_i P_{ij} = \pi_j P_{ji}$ for all $i, j \in S$. Then π is a stationary distribution of the chain, and X_n is reversible with respect to π.*

Proof. Suppose that π satisfies the conditions of the theorem. Then

$$\sum_i \pi_i P_{ij} = \sum_i \pi_j P_{ji} = \pi_j \sum_i P_{ji} = \pi_j.$$

Therefore $\pi = \pi P$, so π is a stationary distribution. By definition, X_n is reversible with respect to π. □

This theorem gives a useful way of finding the stationary distribution, since it is often easier to solve the detailed balance equations (1), which are local, than the equations for the stationary distribution (2), which are global. Here is an example to illustrate how one uses detailed balance to find a stationary distribution.

Example 3.1 (Ehrenfest model of diffusion) Consider a container with a membrane in the middle and a total of m particles distributed in some say between the left and right sides of the container. At each step we pick one particle at random and move it to the other side. Let X_n be the number of particles in the left side at time n. Then X_n is a Markov chain, with transition probabilities $P_{i,i+1} = 1 - \frac{i}{m}$, $P_{i,i-1} = \frac{i}{m}$. What is the stationary distribution of this chain?

Let's look for a probability distribution π that satisfies (1). If we find a solution, we know by the theorem above that it is a stationary distribution. We also know it's the unique such stationary distribution, since the transition matrix P is irreducible (see Lecture 2). Notice that all edges are between nodes $i \leftrightarrow i+1$ for $i = 0, 1, \ldots, m-1$ so we only have to solve the equations

$$\pi_i P_{i,i+1} = \pi_{i+1} P_{i+1,i} \quad \Leftrightarrow \quad \pi_i \left(1 - \frac{i}{m}\right) = \pi_{i+1} \left(\frac{i+1}{m}\right) \quad \Leftrightarrow \quad \pi_{i+1} = \pi_i \frac{m-i}{i+1}.$$

One way to solve this is to (temporarily) set $\pi_0 = 1$, and then solve recursively for $\pi_1, \pi_2, \ldots$. It is not hard to derive the (again, temporary) formula $\pi_i = \binom{m}{i}$. Now normalize π to make it a probability distribution, to obtain

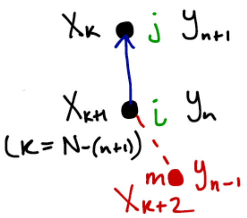
$$\pi_i = \frac{1}{2^m} \binom{m}{i}, \qquad i = 0, 1, \ldots, m.$$

Therefore the stationary distribution for the number of particles in one side of the container is Binomial$(m, \frac{1}{2})$.

Why is a Markov chain that satisfies the detailed balance equations called *reversible*? Recall the question the Homework 2 where we ran a chain backwards in time: we took a regular Markov chain, started it in the stationary distribution, and let it run, to get $X_0 \sim \pi, X_1 \sim \pi, \ldots, X_N \sim \pi$. Let $Y_n = X_{N-n}$ be the "reversed" chain. Then, you showed that $Y_0, Y_1, \ldots, Y_N$ is a Markov chain with transition probabilities

$$P(Y_{n+1} = j | Y_n = i) = \frac{\pi_j}{\pi_i} P_{ji}.$$

When are the transition probabilities for Y_n the same as those for X_n? Exactly when X_n satisfies detailed balance! In this case the chain is statistically indistinguishable whether it is run forward or backward in time: given a trajectory, no statistical test can be applied to tell if it came from running the chain X_n or the chain Y_n.



Detailed balance is a critical concept in physics and chemistry. If a system satisfies detailed balance, then it is called an *equilibrium system*, or sometimes *reversible in equilibrium*. Otherwise, it is called a *non-equilibrium system*.[2] If a system does not satisfy detailed balance, then there are non-zero fluxes in steady-state, which is only possible if the system has forces acting on it. The statistical mechanics of such forced systems is vastly more complicated, and an active area of research.[3]

If you like fluid dynamics, one way to picture detailed balance is to imagine a turbulent fluid in a box. If the fluid is forced and dissipated isotropically, (e.g. by some external heat bath), then the mean flow over long enough timescales should be zero. However, if we stir the fluid in one direction, then even if it is very turbulent there will be a mean circulation in the box.

Here are some examples of physical systems that do / do not satisfy detailed balance:

Detailed balance

- passive particle diffusing in a fluid
- protein binding and unbinding to a receptor on a cell
- Ising model with dynamics above
- ideal gas in an insulated box
- system in contact with one heat bath
- covered, insulated coffee cup with liquid/vapour equilibrium
- crystal with no external forces

No detailed balance

- self-propelled (e.g. swimming) particles, e.g. collection of bacteria
- molecular motor
- atmospheric circulation patterns
- plasma with non-Maxwellian velocities
- system in contact with two heat baths, at different temperatures
- snowflake melting in a coffee cup
- sheared crystal

[2] Being an equilibrium system is dfferent from being in equilibrium. A system is in equilibrium if its probability distribution is the stationary distribution, i.e. it is in steady-state. A system is an equilibrium system if, in addition to being in equilibrium, it satisfies detailed balance with respect to its stationary distribution.

[3] One simple example of a non-equilibrium system that is still very poorly understood is a conducting rod, that is maintained at a hot temperature at one end and a cold one at the other. We all know the steady-state is a linear temperature distribution in the rod, but deriving this from microscopic interactions is still a research topic.

3.2 Spectral decomposition for a Markov Chain that satisfies detailed balance

If P satisfies detailed balance (meaning the Markov chain whose transition probabilities are described by P satisfies detailed balance), then it can be symmetrized by a similarity transformation. Let

$$V = \Lambda P \Lambda^{-1}, \qquad \text{where} \quad \Lambda = \begin{pmatrix} \sqrt{\pi_1} & & & \\ & \sqrt{\pi_2} & & \\ & & \ddots & \\ & & & \sqrt{\pi_N} \end{pmatrix}. \tag{3}$$

All off-diagonal elements of Λ are 0. We claim that when P satisfies detailed balance, then V is symmetric. Let's check this:

$$V_{ij} = \frac{\sqrt{\pi_i}}{\sqrt{\pi_j}} P_{ij} = \underbrace{\frac{\sqrt{\pi_i}}{\sqrt{\pi_j}} \frac{\pi_j}{\pi_i} P_{ji}}_{\text{detailed balance}} = \frac{\sqrt{\pi_j}}{\sqrt{\pi_i}} P_{ji} = V_{ji}.$$

This is useful, since we can use our knowledge of the spectrum of symmetric matrices, to characterize the spectrum of P. Since V is symmetric, it has a full set of real eigenvalues $\lambda_j \in \mathbb{R}$, and an orthonormal set of eigenvectors w_j that are both the left and right eigenvectors.[4] Therefore, P has the same eigenvalues λ_j, and it has eigenvectors

$$\begin{array}{llcl} \text{left eigenvectors} & \psi_j = \Lambda w_j & \Leftrightarrow & (\psi_j)_i = (w_j)_i \sqrt{\pi_i} \\ \text{right eigenvectors} & \phi_j = \Lambda^{-1} w_j & \Leftrightarrow & (\phi_j)_i = (w_j)_i (\sqrt{\pi_i})^{-1} \end{array}$$

This relationship implies that

$$\psi_j = \Lambda^2 \phi_j \,,$$

i.e. the left eigenvector equals the right eigenvector multiplied (component-wise) by π. We already knew this was true for the eigenvectors corresponding to $\lambda_1 = 1$, but for a chain that satisfies detailed balance it is true for all the other eigenvectors as well.

Suppose the eigenvectors are normalized so that $|w_j| = 1$. Then we can write P using its spectral decomposition as

$$P = \sum_{k=1}^{N} \lambda_k \phi_k \psi_k^T = \sum_{k=1}^{N} \lambda_k \phi_k \phi_k^T \Lambda^2. \tag{4}$$

In components: $P_{ij} = \sum_{k=1}^{N} \lambda_k (\phi_k)_i (\psi_k)_j = \sum_{k=1}^{N} \lambda_k (\phi_k)_i (\phi_k)_j \pi_j$. From this decomposition, one can see that the n-step transition matrix P^n will be mostly formed from the eigenvectors corresponding to eigenvalues that are close to 1; these eigenvectors therefore are related to the long-time dynamics of the chain.

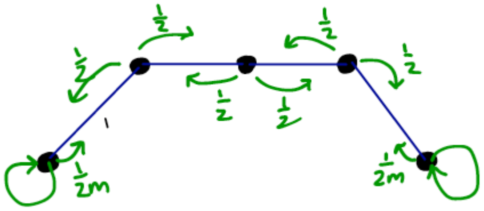
Remark. The spectral decomposition (4) is sometimes used to approximate P by truncating the sum after a small number of eigenvectors, which effectively keeps only the dynamics that evolve on longer time scales, as we will see shortly. However, if you truncate (4), then you typically don't get a stochastic matrix back. In fact, if you evolve the probability with the truncated matrix, the probability can even become negative. Therefore the spectral decomposition hides the fact that P is stochastic.

[4] We think of both left and right eigenvectors as columns vectors in this section.

The spectral decomposition gives insight into the timescales associated with the Markov chain. Let's take a look at an example.

Example 3.2 Consider the transition matrix

$$P = \begin{pmatrix} 1-\frac{1}{2m} & \frac{1}{2m} & 0 & 0 & 0 \\ \frac{1}{2} & 0 & \frac{1}{2} & 0 & 0 \\ 0 & \frac{1}{2} & 0 & \frac{1}{2} & 0 \\ 0 & 0 & \frac{1}{2} & 0 & \frac{1}{2} \\ 0 & 0 & 0 & \frac{1}{2m} & 1-\frac{1}{2m} \end{pmatrix}$$



This Markov chain describes a particle moving on a state space with 5 sites. It undergoes an unbiased random walk along the middle 3 nodes, but when it hits an endpoint, it tends to stay there for a time proportional to parameter m, before escaping to possibly visit the other endpoint. We will be interested in the dynamics for large values of m.

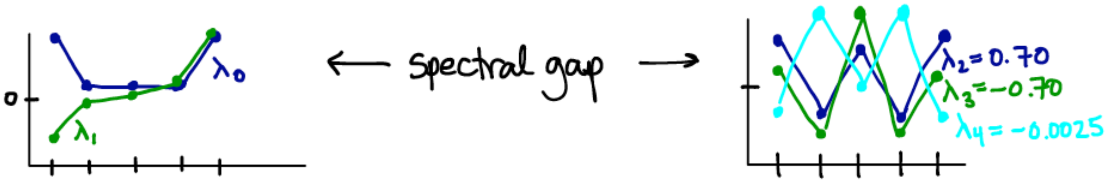
The stationary distribution is $\pi = Z^{-1}(m,1,1,1,m)$, where $Z = 2m+3$. You can check that the chain satisfies detailed balance.

Let's calculate the eigenvalues:

$$\begin{array}{lllllll} m=2: & \lambda = \{1, & 0.89, & \text{-0.75}, & 0.5, & \text{-0.14}\} \\ m=5: & \lambda = \{1, & 0.95, & \text{-0.72}, & 0.62, & \text{-0.05}\} \\ m=20: & \lambda = \{1, & 0.99, & \text{-0.71}, & 0.68, & \text{-0.01}\} \\ m=100: & \lambda = \{1, & 0.999, & \text{-0.71}, & 0.70, & \text{-0.03}\} \end{array}$$

There is a spectral gap between the second eigenvalue λ_2, which approaches 1 as $m \to \infty$, and the third eigenvalue λ_3, which appears bounded in absolute value away from 1.

What do the left eigenvectors look like? Here is a sketch (for large enough m):



The eigenvector corresponding to λ_1, is the stationary distribution (as expected), while the eigenvector corresponding to λ_2 captures transitions between the endpoints; it corrects the stationary distribution to account for the imbalance of probability between the endpoints. The timescale of transition is controlled by $-\log|\lambda_2|$. The other eigenvectors describe the diffusive, random-walk-like motion across the flat region in the middle.

3.3 Markov Chain Monte Carlo

Markov Chain Monte Carlo (MCMC) refers to a collection of algorithms to generate a random variable X whose probability distribution is π:

$$X \sim \pi.$$

You might already know about methods for generating independent samples from a probability distribution, such as the inverse transform method, the acceptance/rejection method, etc. However, these methods become essentially useless when the dimensionality of the system becomes very high. Markov Chain Monte Carlo algorithms that are effective even when π is defined on a very high-dimensional space. It is an extremely powerful algorithm; the Metropolis-Hasting algorithm (a particular MCMC method) has been named on of the top ten most influential algorithms of the 20th century.[5]

Diaconis (2009) says that

> To someone working in my part of the world, asking about applications of Markov chain Monte Carlo is a little like asking about applications of the quadratic formula.

Yet, one of the leading Monte Carlo researchers, Alan Sokal, starts his popular set of lecture notes with this warning (Sokal, 1989):

> Monte Carlo is an extremely *bad* method; it should be used only when all alternative methods are worse.

Hopefully by the end of this short lecture and homework set you will also understand the power and the pitfalls of MCMC. For now, let's start by looking at a few examples where "all else is worse."

Examples 3.3

(1) (Ising model): This is a model that was initially invented to study magnetism, but it has since been been used in an enormous variety of applications, ranging from ice to gasses to spin glasses to cancer to ion channels to neuroscience to urban segregation.

Consider a lattice, say a 2d lattice with $N = m \times m$ sites (the magnet), where each square on the lattice is assigned a "spin" $\sigma_j \in \{+1, -1\}$, called up spins and down spins. Each configuration of spins $\sigma = (\sigma_1, \dots, \sigma_N)$ has an energy

$$H(\sigma) = -\sum_{\langle i,j\rangle} \sigma_i \sigma_j, \tag{5}$$

where $\langle i, j\rangle$ means i, j are neighbours on the lattice. The energy is lower when neighbouring spins are the same.

The stationary distribution is known from statistical mechanics to be the Boltzmann distribution:

$$\pi(\sigma) = Z^{-1} e^{-\beta H(\sigma)}. \tag{6}$$

[5] See `http://www.siam.org/news/news.php?id=637`, and also Nick Hingham's blog `https://nickhigham.wordpress.com/2016/03/29/the-top-10-algorithms-in-applied-mathematics/` . The other 9 include the simplex method of linear programming, Krylov subspace iteration methods, the decompositional approach to matrix computations, the Fortran optimizing compiler, the QR algorithm for computing eigenvalues, Quicksort, Fast Fourier transform, Integer relation detection, and the Fast multipole method (invented in part by Courant's own Leslie Greengard!)

Here Z is a normalization constant, which is almost never known, and β is a parameter, called the *inverse temperature*. It is related to the actual temperature T as $\beta = (k_B T)^{-1}$, where k_B is Boltzmann's constant.

For large β (low temperature), $\pi(\sigma)$ is bimodal: it is highly peaked near configurations with mostly $+1$s or mostly -1s. Physically, the system is magnetized. For small β (high temperature), $\pi(\sigma)$ gives most weight to systems with nearly equal $+1$s and -1s; the system is disordered and loses its magnetization. One question of interest is: at which temperature (which β^{-1}) does this transition occur?

We can answer this by calculating the average magnetization $M = |\frac{1}{N}\sum_{i=1}^{N}\sigma_i|$ when the system is in equilibrium (has reached its stationary distribution). That is, we calculate $\mathbb{E}_\pi M = \sum_{\sigma\in S}\pi_\sigma f(\sigma)$. To do this we must generate random variables whose distribution is π, and average their corresponding values of M. As β increases, $\langle |M| \rangle_\pi$ should transition from 0 to 1, with a sharper transition as the system gets larger.

(2) (Particles interacting with a pairwise potential) A commonly-studied model is a collection of point particles that interact with a pairwise potential $V(r)$, where r is the distance between a pair. The total energy of a system of n particles is the sum over all the pairwise interactions:

$$U(x) = \sum_{i,j} V(|x_i - x_j|),$$

where $x = (x_1, x_2, \ldots, x_n) \in \mathbb{R}^3$ is the $3n$-dimensional vector of particle positions, and $x_i \in \mathbb{R}^3$ is the position of the ith particle. We may also restrict positions to a lattice, so that $x \in \mathbb{Z}^3$. Such models arise in numerous systems, such to describe the interactions between atoms in a crystal or noble gas cluster, interactions between amino acids in a protein, studies of particles jamming, even certain studies of pedestrian flows.

The stationary distribution is again the Boltzmann distribution: $\pi(x) = Z^{-1}e^{-\beta U(x)}$, where again, β is the inverse temperature, and we rarely know the normalization constant Z.

Depending on β, the system could prefer to be in a number of different states such as a solid, crystal, liquid, gas, or other phase. To calculate the phase diagram we must generate samples from $\pi(x)$, and compute averages of some function that characterizes the system's phase.

(3) (Likelihood functions) In Bayesian statistics one often wants to generate samples $\theta = (\theta_1, \theta_2, \ldots, \theta_p)$ from a Likelihood function $L(\theta|x)$. The vector θ represents parameter values, and the vector $x = (x_1, x_2, \ldots, x_M)$ is the observed data. The likelihood function can be calculated from a probability model as $L(\theta|x) = p_\theta(x)$, where $(p_\theta(x))_\theta$ is a family of probability densities for observing data x, or, it can be calculated through a Bayesian procedure. The form of L can often be very complicated. Inference problems of this type arise in a great many places; one example is using radial velocity data of a star to determine how many planets are in its orbit.[6]

Here are several difficulties in sampling π for these examples and the vast number of related ones:

- The size of the state space is often HUGE!

[6] See for example Hou, Fengji, Jonathan Goodman, and David W. Hogg. "The Probabilities of Orbital-Companion Models for Stellar Radial Velocity Data." *arXiv:1401.6128* (2014).

  - The $m \times m$ Ising model has $N = 2^{m^2}$ elements in the state space. Even for small m, say $m = 10$, we have over 10^{30} configurations... there is no way we can list them all.
  - Even a small system of 100 particles lives in 300-dimensional space. There is no way we are going to adequately sample every region in this space.

- Often we don't know the normalization constant for π, only a function it's proportional to.
  - The Boltzmann distribution $Z^{-1}e^{-\beta U(x)}$ arises frequently. We usually know the potential energy $U(x)$ but can almost never calculate Z analytically or though any deterministic procedure (you can calculate Z in some cases by a randomized procedure.[7])
  - Likelihood functions are similarly complicated and impossible to integrate analytically or using deterministic numerics.

- While we could simulate the *dynamics* of the system until it reaches steady-state, we may not know the system's true dynamics, or these may have widely separated time scales so cannot be efficiently simulated. Yet, we still often know the steady-state distribution π and must find a way to sample from it.

  For the examples above, here is an illustration of why the dynamics are hard to simulate:

  - Ising model – We need quantum mechanics to describe the true dynamics of a magnet, which cannot be numerically integrated for such large systems.
  - Particles in a box – the heat bath that thermalizes the system often has much faster timescales than the ones we are interested in. For example, a protein that folds is thermal because of the fluctuations in the solvent (usually water), but the dynamics of water atoms occur on timescales many times faster than the timescale of protein folding, and additionally, there are far to many water atoms to simulate all of them.
  - Likelihood functions – there are no dynamics we could simulate here; and they are almost never related to known probability transforms with nice procedures to generate samples from them.

To overcome these challenges, MCMC generates samples from π by creating a Markov chain that has π as its stationary distribution, and then simulating this Markov chain instead. The most common MCMC algorithm is the Metropolis-Hastings algorithm.

Metropolis-Hasting Algorithm. Let π be a discrete probability distribution. The Metropolis-Hastings algorithm constructs a Markov chain $X_0, X_1, \ldots$ with stationary distribution π as follows.

Let H be the transition matrix for *any* irreducible Markov chain, whose state space contains the support of π. Suppose $X_n = i$. We generate X_{n+1} in the following steps:

1. Choose a *proposal state* Y according to the probability distribution given by the ith row of H, so that $P(Y = j|X_n = i) = H_{ij}$.

[7] See Frenkel, Daan, and Anthony JC Ladd. "New Monte Carlo method to compute the free energy of arbitrary solids. Application to the fcc and hcp phases of hard spheres." *The Journal of chemical physics* 81.7 (1984): 3188-3193, or Zappa, Emilio, Miranda Holmes-Cerfon, and Jonathan Goodman. "Monte Carlo on manifolds: sampling densities and integrating functions." *Communications on Pure and Applied Mathematics* 71.12 (2018): 2609-2647.

2. Calculate the *acceptance probability*[8]

$$a_{ij} = \min\left(1, \frac{\pi_j H_{ji}}{\pi_i H_{ij}}\right). \tag{7}$$

3. Accept the proposal $Y = j$ with probability a_{ij}. That is, let $U \sim \text{Uniform}([0,1])$, and:
   - If $U < a_{ij}$ then *accept* the move: set $X_{n+1} = Y$;
   - If $U > a_{ij}$ then *reject* the move: set $X_{n+1} = X_n$.

The final induced Markov chain has transition probabilities

$$P_{ij} = \begin{cases} H_{ij}a_{ij} & (i \neq j) \\ 1 - \sum_{i\neq j} H_{ij}a_{ij} & (i = j) \end{cases} \tag{8}$$

Lemma. *The Markov chain constructed by the Metropolis-Hastings algorithm has the stationary distribution π.*

Proof. We prove this by showing the chain satisfies detailed balance with respect to π.

Suppose that $\pi_j H_{ji} \leq \pi_i H_{ij}$. Then $P_{ij} = H_{ij} \min\left(1, \frac{\pi_j H_{ji}}{\pi_i H_{ij}}\right) = \frac{\pi_j}{\pi_i} H_{ji}$, and $P_{ji} = H_{ji} \min\left(1, \frac{\pi_i H_{ij}}{\pi_j H_{ji}}\right) = H_{ji}$. Therefore

$$\pi_i P_{ij} = \pi_i \frac{\pi_j}{\pi_i} H_{ji} = \pi_j H_{ji} = \pi_j P_{ji}.$$

A similar calculation holds if $\pi_j H_{ji} > \pi_i H_{ij}$. Therefore the chain satisfies detailed balance with respect to π, so by a previous theorem, π is the stationary distribution. □

There are several reasons why this algorithm is extremely powerful.

- You don't need to know π, the normalized probability distribution, but rather only a function $g(x)$ it is proportional to: $g(x) = Z\pi(x)$, where $Z = \int_x g(x)$. The normalization constant Z cancels out in the acceptance ratio (7), so you don't need to calculate it.

- The proposal matrix H can be absolutely anything, as long as it is irreducible and has the right state space. This gives a lot of freedom to choose proposal dynamics based on what is convenient for the problem at hand. The Metropolis-Hastings step "corrects" the proposal chain H, to make it have the right stationary distribution.

Here are some possible proposal matrices for the examples in 3.3:

(1) (Ising model)

   - pick a spin, flip it
   - pick a pair of spins, exchange their values
   - pick a spin or a cluster of spins, change the value(s) depending on the environment surrounding them, e.g. set the value to be the one that minimizes the energy in the spin's local neighbourhood

(2) (Particles)

[8] This is the Metropolis-Hastings part; other formulas would give rise to other MCMC algorithms.

  - pick a single particle, move it some random displacement in a random direction
  - move all particles in the direction of the gradient of the potential, plus some random amount

While you are free to choose any proposal matrix H, some choices will work better than others. Choosing a *good* proposal matrix is an art. If proposals are too close to the current point, then almost every move is accepted, but it takes a long time to explore the whole space. If proposals are too far from the current point, then almost all moves are rejected, which also slows convergence. You should propose moves that are large enough to explore the whole space quickly, but not so large that they are often rejected.

A rule of thumb is to choose your proposals to achieve a desired average acceptance ratio. Results for idealized theoretical problems suggest this should be $\approx$25-50%, depending on the dimension of the system. In practice, for a complex problem, one doesn't usually know the optimal acceptance ratio, but rather would vary parameters in a proposal matrix until a desired level of convergence is achieved. Good Monte Carlo methods will have a proposal that is adapted to the problem at hand, taking into account the structure of individual problems.

Here are a few things to be aware of when implementing the Metropolis-Hastings algorithm in practice:

- The chain may take some time from its starting point before it actually reaches the stationary distribution. The length of the initial transient data is called the *burn-in time*. Some practitioners throw out some number of initial data steps, however if you run the chain for long enough this initial transient till not significantly affect the measured statistics.
- There are other ways to calculate the acceptance probability a_{ij}. You will be asked to examine other possibilities on the homework. The Metropolis-Hastings formula can be shown to be statistically optimal, in the sense that it has the smallest correlation time for a class of statistics f (see (9) below) (see e.g. Liu (2004), Section 13.3.1).
- This algorithm works equally well in a continuous state space, provided we interpret the transition probability h as a density: $h(y|x)$ is the probability density of jumping to y, given we start at x. The Metropolis-Hastings acceptance ratio is $a(y|x) = \min\left(1, \frac{\pi(y)h(x|y)}{\pi(x)h(y|x)}\right)$.
- How long does it take the chain to reach its stationary distribution? From the spectral decomposition of P, we know this time is controlled by $-\log|\lambda_2|$, where λ_2 is the eigenvalue of P with the second-largest norm. In practice, we almost never know λ_2, so one must determine whether the chain has converged through empirical measures. One measure is to simply plot a statistic you are estimating as a function of time, and observe when it has settled down and doesn't change much. A better measure is to estimate the correlation time for a statistic (see below).
- A common way to measure how well the algorithm works is the correlation time of a statistic. That is, if you are interested in measuring $\mathbb{E}_\pi f(x)$ for some function $f(x)$, then the correlation time of f is defined as

$$\tau_f = \frac{1}{C_f(0)} \sum_{t=-\infty}^{\infty} C_f(t)dt, \qquad \text{where} \quad C_f(t) = \mathbb{E}_\pi f(X_t)f(X_0) - (\mathbb{E}_\pi f(X_t))^2. \tag{9}$$

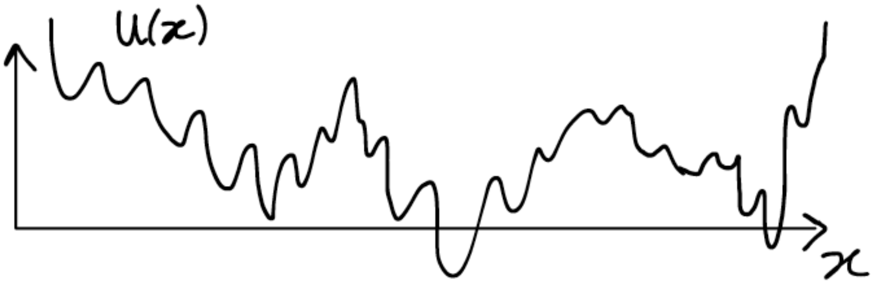
  The function $C_f(t)$ is the covariance function of $f(X_t)$, assuming $X_0 \sim \pi$. (Don't worry too much about this definition now; we will come back to it later in the course.) The quantity τ_f measures the time it takes the chain X_t to "forget" its given state: when τ_f is large, we need to generate many more steps of the chain before we forget the last step, whereas when τ_f is small, we don't need to generate as many points. Therefore we would like to choose H so the resulting Markov chain has as small a correlation time as possible.

- Because the points generated by a Markov chain are correlated, the total "effective" number of samples used to estimate a statistic is less than the actual number of points generated. Given a sequence $X_1, X_2, \ldots, X_n$ generated from a Markov Chain with $X_1 \sim \pi$, and an estimator $\hat{f}_n = (f(X_1) + \cdots + f(X_n))/n$ for $\mathbb{E}_\pi f$, the variance of this estimator when n is large is approximately

$$\mathrm{Var}(\hat{f}_n) \to \frac{\mathrm{Var}_\pi(f)}{n_{\mathrm{eff}}} \quad \text{as } n \to \infty, \qquad \text{where } n_{\mathrm{eff}} = \frac{n}{\tau_f}$$

is the number of "effective" points in the sample. That is, we require τ_f more points to estimate a statistic to a given precision, as we do with independent samples.

3.4 Monte-Carlo methods in optimization



Suppose you have a non-convex, possibly very rugged, function $U(x)$, e.g. as shown above. How can you find the global minimum?

Deterministic methods (e.g. steepest-descent, Gauss-Newton, Levenberg-Marquadt, BFGS, etc) are very good at finding a *local* minimum. To find a *global* minimum, or one that is close to optimal, one must often search the landscape stochastically.

One way to do this is to create a stationary distribution π that puts high probability on the lowest-energy parts of the landscape. A common choice is the Boltzmann distribution $Z^{-1}e^{-\beta U(x)}$ for some inverse temperature β. Then, one constructs a Markov chain to sample this stationary distribution, and keeps track of the smallest value the chain has seen.

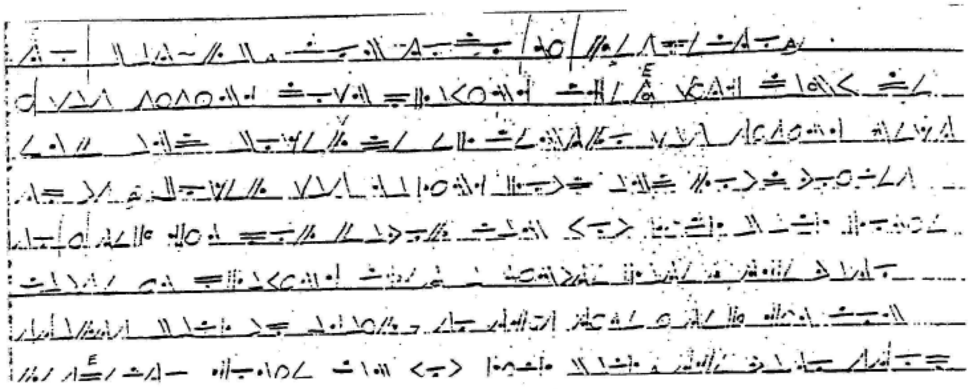
How should β be chosen to have some hope of finding a global minimum? If β is large, the global minimum will be the most likely place to be in equilibrium, but it will take a very long time to reach equilibrium, because the system will get trapped in local minima. If β is small, the chain moves about on the landscape much more quickly, but doesn't spend as much time near the local minima.

Therefore, we must change β, to find a global minimum. One way to change β is by *simulated annealing*. This is a technique that slowly increases β with time t, for example as $\beta = \log t$ or $\beta = (1.001)^t$. As $t \to \infty$, it can be shown that the stationary distribution converges to a delta-function at the global minimum Kirkpatrick et al. (1983). In practice, it takes exponentially long to do so, but this method still gives good results for many problems.

Example 3.4 (Lennard-Jones clusters) A Lennard-Jones cluster is a set of n points interacting with a pair potential $U(r) = \varepsilon\left[\left(\frac{\sigma}{r}\right)^6 - \left(\frac{\sigma}{r}\right)^{12}\right]$ for some parameters σ, ε. This is a model for many atoms, molecules, or other interacting components with reasonably short-range interactions. The energy landscape is very rugged, with a great many local minima. To explore the landscape, and/or to find the lowest minima, one method is to construct a Markov chain on the set of local minima. This works as follows: at each step in the chain, perturb the positions of the points $x_1, x_2, \ldots, x_n$ by some random amount, then find the nearest local minimum using a deterministic method, then check the energy of this local minimum, and either accept it or reject it using the Metropolis-Hastings criterion.

This method can be used to solve optimal packing problems, where now the "energy" is the (negative) packing density, or other quantity to be optimized.

Example 3.5 (Cryptography) Another use of Monte-Carlo methods in optimization is in cryptography (Diaconis, 2009). A cipher is a function $\phi : S \to A$, where S is a set of symbols (e.g. a permutation of the alphabet, the Greek letters, a collection of squiggles, etc), and $A = \{\text{a,b,c,}\ldots\}$ is a set of letters and other symbols used in writing. A code is a string of symbols $x_1x_2x_3\ldots x_n$, where $x_i \in S$. Here is an example of a code, used by inmates in a prison in California (this and the next figure are from Diaconis (2009)):



How can we decipher this code? To start one could look at the frequencies of each symbol, and compared them to the known frequencies of letters in a particular language. Suppose $f_1(a_i)$ is the frequency of letter $a_i \in A$, which could be found empirically by counting the frequency of letters in a particular novel. One can then construct a likelihood function for seeing a particular code as

$$L_1(x_1, x_2, \ldots, x_n; \phi) = \prod_{i=1}^{n} f_1(\phi(x_i)).$$

When the likelihood function $L_1(\phi)$ is high, this means that the code obtained from cipher ϕ contains letter frequencies that are similar to those in the language it is written in. Therefore if we define a probability distribution by

$$\pi(\phi) = \frac{L(\phi)}{\sum_\phi L(\phi)},$$

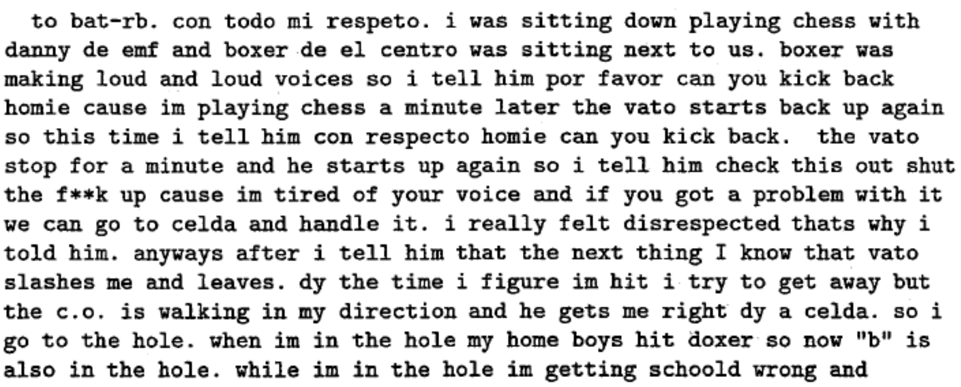
then $\pi(\phi)$ will be large for ciphers that give more ”likely” strings of letters, and small for codes that give pretty unlikely strings of letters. One strategy for deciphering a code is find the cipher ϕ that gives the largest value of $L_1(\phi)$.

Optimizing ϕ by evaluating it for all possible ciphers ϕ is impossible – even if the set of letters contains only lower-case letters and the space, there are $27! \approx 10^{28}$ possible ciphers. However, one can generate samples from $\pi(\phi)$ using an MCMC method, and keep track of the running best value ϕ.

This method was used by statisticians at Stanford to decode the prisoners' message above. It didn't work initially; the output was nonsense. So the statisticians tried again using a more sophisticated likelihood function, that included information about the frequencies of pairs of letters $f_2(a_i, a_j)$ which can also be estimated empirically. The pair likelihood function for seeing a particular code is

$$L_2(x_1, x_2, \ldots, x_n; \phi) = \prod_{i=1}^{n} f_2(\phi(x_i), \phi(x_{i+1})).$$

Optimizing L_2 over all codes using an MCMC method *did* work, and the researchers learned about daily life in prison (in a mixture of English, Spanish, and prison-slang):



```
  to bat-rb. con todo mi respeto. i was sitting down playing chess with
danny de emf and boxer de el centro was sitting next to us. boxer was
making loud and loud voices so i tell him por favor can you kick back
homie cause im playing chess a minute later the vato starts back up again
so this time i tell him con respecto homie can you kick back.  the vato
stop for a minute and he starts up again so i tell him check this out shut
the f**k up cause im tired of your voice and if you got a problem with it
we can go to celda and handle it. i really felt disrespected thats why i
told him. anyways after i tell him that the next thing I know that vato
slashes me and leaves. dy the time i figure im hit i try to get away but
the c.o. is walking in my direction and he gets me right dy a celda. so i
go to the hole. when im in the hole my home boys hit doxer so now "b" is
also in the hole. while im in the hole im getting schoold wrong and
```

References

Diaconis, P. (2009). The Markov Chain Monte Carlo revolution. *Bulletin of the American Mathematical Society*, 46:179–205.

Grimmett, G. and Stirzaker, D. (2001). *Probability and Random Processes*. Oxford University Press.

Kirkpatrick, S., Gelatt, C. D., and Vecchi, M. P. (1983). Optimization by Simulated Annealing. *Science*, 220(4598):671–680.

Liu, J. S. (2004). *Monte Carlo Strategies in Scientific Computing*. Springer-Verlag.

Madras, N. (2002). *Lectures on Monte Carlo Methods*. American Mathematical Society.

Sokal, A. D. (1989). Monte carlo methods in statistical mechanics: foundations and new algorithms. In *Cours de Troisieme Cyle de la Physique en Suisse Romande.*, Lausanne.