http://www.math.nyu.edu/faculty/goodman/teaching/MonteCarlo20/

Class notes: Monte Carlo methods
Week 3, Markov chain Monte Carlo
Jonathan Goodman
September 29, 2020

# 1 Introduction to Markov chain Monte Carlo

Markov chain Monte Carlo, called *MCMC* is part of most Monte Carlo calculations. If you have a probability distribution in more than one or two variables that comes from a real application, it's likely that MCMC is the best way to create samples. My original plan for this course was to do more classes before introducing MCMC, but most interesting examples rely on it.

A *direct sampler* is an algorithm that uses a few uniform random numbers and creates an independent sample of $X$. Some complex objects have direct samplers, such as the path $X_t$ of Exercise 6 of Week 1. Every time you call the path routine, you get a path that is independent of the ones you got before. Suppose $\rho(x)$ is a PDF that we want samples of. We call this the *target* distribution. MCMC produces a sequence $X_k$ that have $X_k \sim \rho_k(x)$. Usually $\rho_n \neq \rho$, but

$$\rho_k \to \rho \;, \quad \text{as } k \to \infty \;. \tag{1}$$

These $X_k$ are approximate samples of the target distribution, but not exact. They can be used in place of exact samples for many purposes. For example, suppose you want

$$B = \mathrm{E}_\rho[V(X)] \;.$$

[We write $B$ instead of $A$ because $A$ is something else this week.] The approximate samples lead to estimators

$$\widehat{B}_N = \frac{1}{N}\sum_{k=1}^{N} V(X_k) \;. \tag{2}$$

If $X_k$ are independent samples of $\rho$, then this estimator is unbiased. Otherwise

$$\mathrm{E}\Big[\widehat{B}_N\Big] = \frac{1}{N}\sum_{k=1}^{N} \mathrm{E}_{\rho_k}[V(X)] \;. \tag{3}$$

The right side converges to $B$ as $N \to \infty$ because the $\rho_k$ converge to $\rho$ (1). But the left side is not equal to $B$ for any fixed $N$. The estimate is biased.

An issue that can be more serious than bias is statistical error. The error bar of the MCMC estimate depends on

$$\sigma_N^2 = \mathrm{var}\Big(\widehat{B}_n\Big) \;. \tag{4}$$

The $X_k$ are not independent, so

$$\sigma_N^2 \neq \frac{1}{N^2}\sum_{k=1}^{N} \mathrm{var}(V(X_k)) \ . \tag{5}$$

The best MCMC known method may leave the right side much larger than the left side. The true formula is

$$\sigma_N^2 = \frac{1}{N^2}\sum_{j=1}^{N}\sum_{k=1}^{N} \mathrm{cov}(V(X_j), V(X_k)) \ . \tag{6}$$

We can re-write this to separate out the *diagonal terms* with $j = k$ and the *off diagonal* terms with $j \neq k$. The covariance is symmetric, so $\mathrm{cov}(V(X_j), V(X_k)) = \mathrm{cov}(V(X_k), V(X_j))$. Therefore,

$$\sigma_N^2 = \frac{1}{N^2}\left[\sum_{k=1}^{N} \mathrm{var}(V(X_k)) + 2\sum_{j=1}^{N}\sum_{k=j+1}^{N} \mathrm{cov}(V(X_j), V(X_k))\right] \ . \tag{7}$$

The second sum on the right involves *auto-correlations* or *auto-covariances*, which are covariances between the same variable at different times. We will spend a lot of time talking about the *auto-correlation time*, which measures how important these auto-correlations are. All of this may seem complicated and unpleasant, but there are many distributions where this is the best way to sample.

The method is called *Markov chain Monte Carlo* because it the $X_k$ are steps in a Markov chain. [Andrey Andreyevich Markov was a brilliant Russian mathematician from the late 1800's and early 1900's. In Russian, including the middle name is a well deserved sign of respect. Aside from probability, Markov made important contributions to number theory.] A *Markov chain* is a sequence of random variables $X_1, X_2, \ldots$ that is defined by an *initial state*, $X_0$, and *transition probabilities*, $R$. The chain is *stationary* (or *homogeneous*) if the transition distributions $R_k$ do not depend on $k$. In this course, we will understand *Markov chain* to mean *stationary* Markov chain. Once $X_0, X_1, \ldots, X_k$ are given, $R$ determines the distribution of $X_{k+1}$ from $X_k$. We write

$$X_{k+1} \sim R(\cdot|X_k) \ . \tag{8}$$

This means that $R(\cdot|y)$ is a probability distribution as a function of the first variable for every $y$. The distribution is different for each $y$.

A simple Python implementation would involve a function `Rsamp(y, rg)` that would return a sample of the distribution $R(\cdot|y)$ using the random number generator `rg`. The code will make `X[j,k]` be component $j$ of $X_k$. This assumes tt X0 $= X_0$ is given and the generator `rg` has been instantiated.

```
X      = np.array([d,(n+1)])              # allocate a d by (n+1) numpy array
X[:,0] = X0                               # copy the given initial state
for k in range(n):                        # (k+1) goes from 1 to n
    X[:,(k+1)] = Rsamp( X[:,k], rg)       # take one MCMC step
```

The command `X[:,0] = X0` copies the $d$ components of the one index array `X0` to the $d$ components `X[0,0], ..., X[(d-1),0]`. This uses the `slice` mechanism in Python, which you should look up if you're not familiar with it.

A probability distribution $\rho$ is *invariant* under $R$ if

$$X_k \sim \rho \implies X_{k+1} \sim \rho \; . \tag{9}$$

We also say that $R$ *preserves* $\rho$. The first fundamental theorem of MCMC (called the *Perron Frobenius* theorem) is that if $R$ is *nondegenerate* (definition below), and $X_k \sim \rho_k$, then (1) holds. This says that if $R$ preserves $\rho$, then the distribution of $X_k$ converges to $\rho$ as $k \to \infty$. This convergence holds for any initial state or distribution $X_0 \sim \rho_0$. The second fundamental theorem of MCMC (called the *ergodic theorem for Markov chains*) says that the MCMC estimator (2) converges to the right answer:

$$\widehat{B}_N \;\to\; \mathrm{E}_\rho[V(X)] \; , \;\; \text{as } N \to \infty \; . \tag{10}$$

This goes beyond convergence of the probability distribution $\rho_k$ because convergence of the estimators depends on different samples becoming distinct samples of $\rho$. To see what this means, suppose $R$ is the trivial transition distribution $X_{k+1} = X_k$. This preserves $\rho$, because $X_k \sim \rho \Longrightarrow X_{k+1} \sim \rho$ (trivially). This trivial chain is not "nondegenerate" ("not nondegenerate" means "degenerate") in the sense we have not yet given. But this chain has

$$\widehat{B}_N = V(X_0) \;\; \text{for all } N \; .$$

The MCMC convergence (10) obviously does not hold. To summarize, for a nondegenerate Markov chain that preserves $\rho$, the distribution of $X_k$ converges to $\rho$ as $k \to \infty$ (Perron Frobenius) and the samples "wander around the sample distribution" enough that it samples the whole distribution in the sense of (10) (ergodic theorem).

After all this introduction, here is the main point. Suppose you have a target distribution $\rho$ and there is no practical direct sampler. It is likely that there is a practical MCMC sampler. That means, there is a transition distribution $R$ that is practical and preserves $\rho$. Many (or most?) interesting distributions are sampled in this way.

MCMC typically (but not always!) have variance (7) that is much larger than it would be for independent samples. For independent samples, we saw that

$$\sigma_N^2 = \frac{\mathrm{var}(V(X))}{N} \; . \tag{11}$$

One of the most unfortunate facts in MCMC is that for large $N$

$$\sigma_N^2 \approx \frac{\tau}{N} \mathrm{var}(V(X)) \; . \tag{12}$$

That is, the MCMC estimator has variance that is larger than a direct estimator (if that were possible) by a factor of $\tau$, which is called the *auto-correlation time.*

This may be written as

$$\sigma_N^2 \approx \frac{1}{N_{\text{eff}}} \text{var}(V(X)) \ , \quad \text{where } N_{\text{eff}} = \frac{N}{\tau} \ . \tag{13}$$

This says that the "effective" number of samples, $N_{\text{eff}}$ is smaller than the actual number of samples by a factor of the suto-correlation time. The auto-correlation time is the number of MCMC steps needed to produce one sufficiently independent sample. Someone doing MCMC often is happy to achieve $\tau$ as small as 10. For $\tau = 10$ you need ten times the number of MCMC steps as independent samples to achieve a target accuracy. This can be frustrating.

Reducing $\tau$ is a major theme of MCMC research with many beautiful and powerful methods. This is an active and fast moving area.

An MCMC computation is not serious unless it comes with an estimate of $\tau$. We saw that it is easy to estimate the *static variance*, which is $\text{var}(V(X))$. This gives an error bar for a direct sampler. It is harder to estimate $\tau$ for MCMC samplers. It can be challenging to find reliable error bars for MCMC computations. Still, an MCMC computation without an error bar is not serious.

As a example, suppose the target distribution for a one component random variable is $\rho = \mathcal{N}(0, v)$. [I put $v$ instead of $\sigma^2$ for the variance because there will be other variances in this discussion.] Define the MCMC algorithm to be

$$X_{k+1} = aX_k + bZ_k \ , \quad Z_k \sim \mathcal{N}(0,1) \ \text{ i.i.d.} \tag{14}$$

This preserves the target $\rho$ if $X_k \sim \mathcal{N}(0, v) \implies X_{k+1} \sim \mathcal{N}(0, v)$. If $X_k$ is Gaussian, then $X_{k+1}$ is Gaussian, being the sum of independent Gaussians. We only need the mean and variance, assuming $\text{E}[X_k] = 0$ and $\text{var}(X_k) = v$. The mean is automatic, because

$$\text{E}[X_{k+1}] = a\text{E}[X_k] + b\text{E}[Z_k] = a \cdot 0 + b \cdot 0 = 0 \ .$$

The variance calculation uses the fact that $X_k$ and $Z_k$ are independent. Therefore

$$\begin{aligned} \text{var}(X_{k+1}) &= \text{var}(aX_k) + \text{var}(bZ_k) \\ &= a^2 \text{var}(X_k) + b^2 \text{var}(Z_k) \\ &= a^2 v + b^2 \ . \end{aligned}$$

Then $\text{var}(X_{k+1}) = v$ leads to the equation

$$v = a^2 v + b^2 \ .$$

We can choose any $a$ with $|a| < 1$ and choose

$$b = \sqrt{v}\,\sqrt{1 - a^2} \ .$$

The distribution of $X_k$ will converge to $\mathcal{N}(0, v)$ as $k \to \infty$. If you start with $X_0 = 0$ then

$$X_1 = bZ_1 \ , \quad X_2 = abZ_1 + bZ_2 \ , \ldots X_k = a^{k-1} bZ_1 + \cdots + bZ_k \ .$$

The variance is a geometric series

$$\begin{aligned}\mathrm{var}(X_k) &= a^{2(k-1)}b^2 + \cdots + b^2 \\ &= \frac{1-a^{2k}}{1-a^2}v(1-a^2) \\ &= (1-a^{2k})v \;.\end{aligned}$$

This converges to $v$ as $k \to \infty$. To be clear: this is a *model problem* to illustrate MCMC. This is not the best way to sample $\mathcal{N}(0,v)$. Direct sampling is better.

# 2 Detailed balance and Metropolis

*Detailed balance* is the trick behind many MCMC samplers. Let us suppose that $R$ is given by probability densities so there is a function $R(x|y)$. This may be called the *transition probability density* or the *transition kernel.* Then if $X_k \sim \rho_k$, then

$$\rho_{k+1}(x) = \int R(x|y)\rho_k(y)\,dy \;. \tag{15}$$

The condition that $\rho$ preserved by $R$ may be written

$$\rho(x) = \int R(x|y)\rho(y)\,dy \;. \tag{16}$$

The problem of MCMC is: given $\rho$, to find an $R(x|y)$ so that

- $R(\cdot|y)$ may be sampled in a practical way. For any $y$, $R(x|y)$ is a probability distribution as a function of $x$.
- $R$ satisfies (16).

The integral equation (16) is hard to check because it involves a $d$ variable integral. If you could calculate integrals like that, then you would not need Monte Carlo to calculate

$$\mathrm{E}[V(X)] = \int V(x)\rho(x)\,dx \;.$$

The balance condition (16) is an integral condition and is hard to check.

Detailed balance is an algebraic condition that implies the balance condition (16) and is easy to check. The condition is written informally as a “balance” between transition rates

$$\mathrm{Pr}_\rho(x \to y) = \mathrm{Pr}_\rho(y \to x) \;. \tag{17}$$

In order to have an $x \to y$ transition, you first have to be at $X_k = x$. This has probability density $\rho(x)$, which is what $\mathrm{Pr}_\rho(\cdot)$ means. Then you have to choose to go to $y$. This has probability density $R(y|x)$. Together, the probability density of an $X_k = x, X_{k+1} = y$, which is the probability density for the pair $(x, y)$, is

$$\rho(x)R(y|x) \ .$$

The detailed balance condition (17) may be expressed more formally as

$$\rho(x)R(y|x) = \rho(y)R(x|y) \ , \quad \text{for all } y \neq x \ . \tag{18}$$

Section 3 gives more motivation for the terms “balance” and “detailed balance”. “Balance” is for the PDF at a single point $x$ and transitions to or from all other states $y$. “Detailed balance” is a balance condition that requires that from $x$ transitions to and from $y$ should balance separately for each $y$. The term comes from physics (statistical physics). It is a deep physical principle that systems in “equilibrium” satisfy not only balance, but detailed balance.

The detailed balance condition (18) implies the balance condition (16). You assume that detailed balance holds and integrate (18) over $y$

$$\int_y \rho(x)R(y|x)\, dy = \int_y \rho(y)R(x|y)\, dy \ .$$

The left side integral simplifies because $R(y|x)$ is a probability density as a function of $y$ for every $x$, so $\int R(y|x)dy = 1$ and

$$\int_y \rho(x)R(y|x)\, dy = \rho(x) \int_y R(y|x)\, dy = \rho(x) \ . \tag{19}$$

The right side is the right side of (16). It seems too easy.

*Metropolis rejection*, also called *Metropolis Hastings* rejection. is a way to create a transition $R$ that satisfies detailed balance. The overall $X_k \rightarrow X_{k+1}$ transition has a *proposal* and an *accept/reject* step. The proposal is any transition probability $P(y|x)$ that is a probability distribution as a function of $y$ for every $x$. The accept/reject step modifies $P$ so get something that satisfies detailed balance. Here are the details. You are at $X_k$ and you want to generate a random $X_{k+1}$. First you create a random *proposal* $Y \sim P(\cdot|X_k)$. Then you choose, at random, whether to *accept* or *reject* $Y$. If you accept $Y$, then $X_{k+1} = Y$. If you reject $Y$, then $X_{k+1} = X_k$ and $Y$ is forgotten. The *acceptance probability* is $A(y|x)$. It is a probability, so $0 \leq A(y|x) \leq 1$ for all $x$ and $y$. For $y \neq x$, the probability density to propose and then accept $y$ is

$$R(y|x) = P(y|x)A(y|x) \ . \tag{20}$$

The acceptance probability is choses so that $R$ satisfies detailed balance. The condition (18) applied to (20) is

$$\rho(y)P(x|y)A(x|y) = \rho(x)P(y|x)A(y|x) \ .$$

This leads to

$$\frac{A(x|y)}{A(y|x)} = \frac{\rho(x)P(y|x)}{\rho(y)P(x|y)} \ . \tag{21}$$

This one equation determines both $A(x|y)$ and $A(y|x)$.

Here's how. You use the extra condition that the acceptance probabilities should be as large as possible. This is supposed to help you generate really different samples more quickly and have a smaller auto-correlation time. That is a wish, a heuristic, but it is the basis of the Metropolis Hastings formulas below. The right side of (21) is either larger or smaller than 1. If the right side is larger than 1, we choose $A(x|y) = 1$ and then $A(y|x) < 1$ so the fraction on the right also is larger than 1. In the other case, where the right side is smaller than 1, we choose $A(y|x) = 1$ and $A(x|y) < 1$. If the right side is exactly equal to 1 (unlikely but not impossible), we take $A(y|x) = A(x|y) = 1$. All of this is contained in the *Metropolis Hastings formula*

$$A(x|y) = \min\left( \frac{\rho(x)P(y|x)}{\rho(y)P(x|y)} \; , \; 1 \right) \; . \tag{22}$$

The fraction on the right is the *Metropolis Hastings ratio.* Repeating what we just said, if the ratio is less than 1, we choose $A(x,y)$ to be the ratio and $A(y|x) = 1$. You can see that the Metropolis Hastings ratio for $A(y|x)$ is the reciprocal and therefore larger than 1. If the ratio is larger than 1, we choose $A(x|y) = 1$ and satisfy the detailed balance condition (21) by taking $A(y|x) < 1$.

You need three things to program the a Metropolis Hastings MCMC algorithm.

1. A routine to evaluate $\rho(x)$ for any $x$
2. A routine to generate a random sample $X \sim R(\cdot|y)$ for any $y$.
3. A routine to evaluate $P(x|y)$ for any pair $(x, y)$

Here is how a simple Metropolis Hastings sampler might look. The code uses a sampler for the proposal distribution, called `Psamp(x, rg)`, and an evaluator of the proposal density, called `P(x,y)`. You might notice that the two `return Y` commands can be combined. If $MH > 1$, then $U < MH$, because $U \leq 1$.

```
def Rsamp(X, rg):     # one step of a MH sampler using proposal P
   Y = Psamp(X, rg)  # sample the proposal distribution
   rhoX = rho(X)     # evaluate the target density at X
   rhoY = rho(Y)     # evaluate the target density at the proposal
   PXY  = P(X,Y)     # evaluate the proposal densities for ...
   PYX  = P(Y,X)     # ... forward and backward steps
   MH   = ( rhoX*PYX )/( rhoY*PXY )     # Metropolis Hastings ratio
   if ( MH > 1.):    #  A = 1 means accept with probability 1
       return Y      #  return the proposal if accept
   U = rg.random()
   if ( U < MH ):    #  if A < 1, accept with probability A
       return Y      #  if accept, return the proposal
   return X          #  if reject, return X
```

It often happens, particularly in statistical applications, that most of the computational work goes into evaluating the target density $\rho(x)$. In Bayesian statistics, this is there the data and the data model are used. The code above does twice as much work as it needs to because it evaluates both $\rho(X)$ and $\rho(Y)$. But it would not be evaluating $\rho(X)$ is $X$ had not been proposed (and been a $Y$) before. You can make the code run twice as fast by remembering $\rho(X)$ rather than re-computing it.

The need to evaluate $P(x|y)$ is a serious drawback to the Metropolis Hastings approach. Some proposal ideas involve generating intermediate random variables, first $Z$ from $X$ then the proposal $Y$ from $X$ and $Z$. In this case, the probability density of $Y$ from $X$ could involve an integral over $z$ that cannot be calculated analytically (because most integrals are impossible) or numerically (curse of dimensionality).

A good Metropolis Hastings sampler comes from a good proposal distribution. One possibility, which is sometimes called *the* Metropolis Hastings method, is to take $Y$ to be a Gaussian about $X$ with a certain characteristic distance $r$. That means

$$Y \sim \mathcal{N}(X, r^2 I) \ . \tag{23}$$

This makes the proposal density

$$P(y|x) = \frac{1}{(2\pi r^2)^{\frac{d}{2}}} e^{-\frac{1}{2r^2}|y-x|^2} \ . \tag{24}$$

Your code does not have to evaluate the prefactor because that cancels in the Metropolis Hastings ratio (22). But it does have to evaluate the exponential part, and to generate the proposal, as in

```
for j in range(d):
    Y[j] = X[j] + r*rg.normal()
```

We sometimes call this *vanilla* Metropolis (vanilla ice cream is simple and plain and maybe not as good as fancier flavors). Good MCMC codes usually have more sophisticated proposal distributions, typically distributions that depends on the problem in a deeper way.

The Gaussian proposal (23) has a symmetric proposal density: $P(x|y) = P(y|x)$, which you can see in (24). If the proposal is symmetric then it cancels in the ratio (22), which simplifies to

$$A(x|y) = \min\left(\frac{\rho(x)}{\rho(y)}\ ,\ 1\right) \ . \tag{25}$$

The original Metropolis method has a symmetric proposal and simplified acceptance probability (25). The name "Metropolis" if from the paper with authors Metropolis, Rosenbluth, Rosenbluth, Teller, and Teller. Insiders sometimes call it the MR$^2$T$^2$ algorithm, and they argue about which of the authors deserves the most credit (hint, not the first one). The two Rosenbluths were father and son. The two Tellers were wife and husband. Hastings suggested un-symmetric proposals and found the more complex formula (22). We often call this the "Metropolis" method even though it was discovered by Hastings.

The vanilla proposal distribution 23 has a free parameter, $r$, which is the *proposal step size*. This is chosen to make the resulting MCMC method as effective as possible. If $r$ is very small then proposals are likely to be accepted (which is good), but the steps are small (which is bad because it takes many steps to move a significant distance). If $r$ is very large, then most proposals will land in regions where $\rho$ is small and therefore be rejected. There is a MCMC "rule of thumb" that says you should adjust $r$ so the acceptance probability is about 40%. If the acceptance probability is too high, then you could have taken larger steps. If the acceptance probability is to low, then you never go anywhere because most of your proposals are rejected.

## 3 Discrete state space

You get more insight into MCMC and detailed balance from thinking about what happens on a discrete state space. Suppose there are $n$ states in a state space $\mathcal{S} = \{1, \ldots, n\}$. The states are probably more complicated than just one integer, but they can be labeled by integers. Then there are $n$ probabilities $\rho_j$ for $j = 1, \ldots, n$. These form the components of a *row vector*

$$\rho = (\rho_1, \ldots, \rho_n) \; .$$

The Markov chain moves around the state space, so $X_k \in \mathcal{S}$. The transition distribution is given by a *transition matrix* $R$ with entries

$$R_{ij} = \Pr(X_{k+1} = j | X_k = i) \; .$$

This may be written less formally as

$$R_{ij} = \Pr(i \to j) \; .$$

The Markov chain probability dynamics may be written in matrix/vector form. This is useful in part because we can use eigenvalue/eigenvector analysis to understand the algorithm. The probability distribution of $X_k$ will be represented by a row vector $\rho_k = (\rho_{k,1}, \ldots, \rho_{k,n})$. We calculate the dynamics of $\rho_k$ using basic probability

$$\begin{aligned} \rho_{k+1,j} &= \Pr(X_{k+1} = j) \\ &= \sum_{i=1}^{n} \Pr(X_{k+1} = j | X_k = i) \Pr(X_k = i) \\ &= \sum_{i=1}^{n} R_{ij} \, \rho_{k,i} \\ &= \left(\rho_k R\right)_j \quad \text{(entry } j \text{ of the row vector } \rho_k R) \; . \end{aligned}$$

The matrix/vector form is

$$\rho_{k+1} = \rho_k R \; . \tag{26}$$

The condition that the MCMC dynamics preserves $\rho$ is

$$\rho = \rho R \; .$$

This says that $\rho$ is a left eigenvector of $R$ with eigenvalue one.

The fact that $\rho_k \to \rho$ as $k \to \infty$ is related to the other eigenvalues of $R$. Here is a quick review. The matrix $R$ is $n \times n$, so there is a basis consisting of $n$ eigenvectors or generalized eigenvectors. We assume for simplicity that the Jordan form of $R$ has no Jordan blocks bigger than $1 \times 1$, which is the same as assuming that there is a basis consisting of all eigenvectors. Exercise 4 shows that this is true if $R$ satisfies detailed balance. Let the left eigenvectors (row vectors) be $l_1, \ldots, l_n$. The corresponding eigenvalue equation is

$$l_j R = \lambda_j l_j \; , \;\; \text{for } j = 1, \ldots, n \; .$$

Any row vector may be expressed as a linear combination of left eigenvectors. In particular we may write

$$\rho_k = \sum_{j=1}^{n} a_{jk} l_j \; .$$

The "expansion coefficients" for $\rho_k$ in terms of the eigenvectors $l_j$ are $a_{1k}, \ldots, a_{nk}$. We know $\rho$ is an eigenvector, so we take $l_1 = \rho$ and $\lambda_1 = 1$. The convergence $\rho_k \to rho$

The evolution equation (26) for $\rho_k$ gives evolution equations for the expansion coefficients

$$a_{j,k+1} = \lambda_j a_{jk} \; .$$

In particular, if $a_{j,1} \neq 0$, then $a_{jk} \to 0$ as $k \to \infty$ if and only of $|\lambda_j| < 1$.

Eigenvalues are one way to understand the convergence of $\rho_k$ to $\rho$. A good Markov chain for MCMC (definition next week) has a transition matrix $R$ with these properties

- $\lambda_1 = 1$, and $\lambda = 1$ is a simple eigenvalue with a one dimensional eigenspace spanned by $l_1 = \rho$. The invariant distribution is the eigenvector corresponding to $\lambda = 1$.
- $|\lambda_j| < 1$ for $j \neq 1$. All other eigenvalues are "inside the unit circle".

These two facts imply that there is a number I will write as $g$ (for *spectral gap*, but not exactly, details next week.) so that

$$\|\rho_k - \rho\| \leq M(1-g)^k \; . \tag{27}$$

Here is a very vague explanation, with details next week. It illustrates a style of reasoning math people will be familiar with and others may find odd at first. We reason with inequalities that involve some number (like $M$ and $g$) whose value we pretend not to care much about. No matter what $M$ and $g$ are, as long as $g > 0$ and $g < 1$, the inequality (27) implies that $\rho_k$ converges to $\rho$ at least fast as a geometric series.

The reasoning here depends on a few mathematical facts. One fact is that if you have finitely many numbers and all of them are less than one, then the maximum of them is less than one. Another fact is that if you multiply to add numbers, you get a number. Finally, if you add a finite collection of positive numbers, the sum is not more than the number of terms multiplied by the maximum of the numbers. You will see these principles used here. Define

$$\mu = \max_{j \neq 1} |\lambda_j| \ .$$

We can call this number the *reduced spectral radius*, and note that $0 \leq \mu < 1$. [The full *spectral radius* of any square matrix $A$ is the absolute value of the largest eigenvalue. This is the radius of the smallest disk in the complex plane that contains all the eigenvalues. Eigenvalues of a matrix are called *spectrum.*] The formulas above imply that

$$\rho_k = a_1 \rho + \sum_{j=2}^{n} a_{j,0} \lambda_j^k \ .$$

We will see (next week) that $a_1 = 1$. The sum on the right, therefore, is $\rho_k - \rho$. Take $1 - g = \mu$, or $g = 1 - \mu$ to be the spectral gap. This is the “gap” between the largest of the rest of the eigenvalues and the unit circle. Then take

$$m = \max_{j \neq 1} |a_j| \ .$$

Now, $\left|\lambda_j^2\right| \leq (1-g)^2$ if $j \neq 1$. This implies that

$$\begin{aligned} \left| \sum_{j=2}^{n} a_{j,0} \lambda_j^k \right| &\leq n \max j \neq 1 \left|a_{j,0}\right| (1-g)^2 \\ &\leq nm(1-g)^2 \\ &= M(1-g)^2 \ . \end{aligned}$$

This is (27). What we “proved” (full proof next week) is “there is a number $M$ and a $g$ with $0 \leq g < 1$ so that $\cdots$ .

# 4 Exercises

**Corrections**

**Exercise 4** corrected to talk about column vectors, not row vectors. The last formula corrected to have $\langle M_W^* u, v \rangle$ (the $u$ was left out).

**Exercise 5** The proposal distribution originally given does not work (is reducible). Instead, take the proposal distribution uniform in $[x - r, x + r]$.

1. Write a formula for $R(x|y)$ for the linear auto-regressive Markov chain (14). Verify by direct integration that this satisfies the balance formula (16). Verify by algebra that it satisfies the detailed balance formula (18). Show that $\phi_n(y) = \partial_y^n \rho(y)$ is an eigenfunction of the kernel $R$ in the sense that
$$\lambda_n \phi_n(x) = \int R(x|y)\phi_n(y)\, dy \ .$$
[*Hint.* You already did the case $n = 0$. Try $\phi_1 \propto y e^{-\frac{1}{2v}y^2}$ and see how it works. The general trick uses integration by parts.] The *spectral gap* is
$$g = 1 - \max_{n \neq 0} |\lambda_n| \ .$$
Show that the spectral gap is small when $a$ is close to 1. Explain why this should be.

2. Many probability distributions are given in the form
$$\rho(x) = \frac{1}{Z} e^{-\phi(x)} \ . \qquad (28)$$
We saw this in the Bayesian posterior distribution. It is also true in many physical applications. The Gaussian with a known explicit normalization constant $Z$ is the exception. Show that the Metropolis Hastings algorithm does not require you to know the normalization constant $Z$.

3. The *MALA* algorithm (for "Metropolis adjusted Langevin algorithm") has proposals of the form
$$Y = x - a\nabla\phi(x) + \mathcal{N}(0, rI) \ .$$
   (a) Write a formula for the proposal density and show that it is not symmetric.

   (b) (Harder) Find a relationship between $a$ and $r$ that maximized (approximately) the acceptance probability when $a$ is small. If you know stochastic calculus, this is related to the stochastic differential equation
   $$dX_t = -\nabla\phi(X_t)dt + CdW_t \ .$$
   with an appropriate $C$, which leaves $\rho$ invariant.

4. Consider the vector space of $n$ component column vectors. If $W$ is a symmetric positive definite matrix, we can define the $W$ *inner product* as
$$\langle u, v\rangle_W = u^t W v \ .$$
Suppose $W$ is a diagonal matrix with $W_{jj} = \rho_j$ (the probability distribution). Then we write
$$\langle u, v\rangle_\rho = \sum_{j=1}^n u_j \rho_j v_j \ .$$

If $M$ is any $n \times n$ matrix, then the *adjoint* of $M$ *with respect to* $W$ is written $M_W^*$ and is defined by

$$\langle u, Mv \rangle_W = \langle M_W^* u, v \rangle_W \ , \quad \text{for all } u, v \ .$$

Find a formula for the entries of $M_\rho^*$ of the form $M_{\rho,ij}^* = (**) M_{ji}$. Show that if there is a positive definite $W$ so that $M_W^* = M$, then all the eigenvalues of $M$ are real. Show that $R_\rho^* = R$ if and only if $R$ satisfies detailed balance with respect to $\rho$.

5. Write a Metropolis Hastings MCMC sampler for the model problem distribution $\rho = \mathcal{N}(0, v)$. Use the proposal distribution that $Y$ is uniform in the interval $[x - r, x + r]$, $r > 0$ is the parameter of the algorithm. Apply the method to estimating a moment $B = \mathrm{E}[X^{2q}]$. Start with $X_0 = 0$. For various values of $N$, generate $M$ independent MCMC paths and generate that many independent values of $\widehat{B}_N$. Use these to estimate the bias and variance of $\widehat{B}_N$ as a function of $N$. Demonstrate in this way that the bias and variance go to zero as $N$ goes to infinity. Your code will take a very long time to run if you choose parameters carelessly. Try to identify $\tau$ from the computational data. Experiment with various values of $q$ to see how the method works on harder problems. Experiment with various values of $a$ and $r$ to see which ones converge well. As always, format your output and put it into easy to read tables.