These notes draw heavily on *Numerical Recipes*, a valuable resource for entry-level understanding of numerical methods relevant to physics.

1. Statistical Inference and MCMC

Markov Chain Monte Carlo (MCMC) is a way of drawing inferences from data. If you have some data $\vec{d}$ and a set of model parameters $\vec{\theta}$ within some model M, the output of MCMC are a whole bunches of choices of $\vec{\theta}_i$, which (if everything is done correctly) are distributed according to something called the *posterior distribution.* Roughly speaking, the distribution of points will be peaked near the "best" choice of parameters, and their distribution will be distributed according to our uncertainties about the correct parameters.

Note that as we talk about MCMC and its statistical properties, we are *not* going to be talking about "whether M is correct" or "how 'good a fit' M is under model parameters $\vec{\theta}$." All of this assumes M is the correct model—i.e. there is some set of parameters $\vec{\theta}$ under M which describes the real world.

Mathematically, it is helpful to consider the following distribution:

$$P(\vec{d}, \vec{\theta}), \tag{1}$$

which is the probability distribution of a given set of model parameters $\vec{\theta}$ being correct *and simultaneously* us getting the data $\vec{d}$ from our experiment.

The philosophy of Bayesian statistics is that we want to know the probability distribution of the real θ given the results of our experimental results:

$$P(\vec{\theta}|\vec{d}) \tag{2}$$

and we can clearly write:

$$P(\vec{\theta}, \vec{d}) = P(\vec{\theta}|\vec{d})P(\vec{d}) = P(\vec{d}|\vec{\theta})P(\vec{\theta}), \tag{3}$$

which is Bayes' Theorem. Then we can write:

$$P(\vec{\theta}|\vec{d}) = \frac{P(\vec{d}|\vec{\theta})P(\vec{\theta})}{P(\vec{d})} \tag{4}$$

This is the probability distribution—called the "posterior"—that we want! How can we calculate it? If we have a fully specified model then we might know $P(\vec{d}|\vec{\theta})$, which is called the "likelihood." Note that we might also *not* be able to calculate it easily, which is an interesting situation which there are techniques to handle, like simulation-based inference.

But the other quantities are trickier! $P(\vec{\theta})$ is the "distribution" of model parameters under model M, which is called the "prior." What can this distribution possibly mean?? I'm not 100% sure, but I think it is more-or-less true that it is only truly meaningful if your experiment is done within some well-known external situation. For example, say I am a traffic app trying to estimate the speed of traffic (this is $\vec{\theta}$) on a particular road on a particular day from real-time phone location data (this is $\vec{d}$). I may have made that same measurement lots of times in the past. I *should* have a prior $P(\vec{\theta})$ that is based on all of those previous measurements

On the other hand, what if I am a cosmologist trying to infer or put a limit on the cross-section of annihilation of dark matter particles within some dark matter model (this cross-section is $\vec{\theta}$) based on X-ray data at the center of a cluster of galaxies (this is $\vec{d}$). I'm not even sure whether the model of dark matter is correct, let alone what I should expect for the distribution of cross-sections to annihilation. There's only one value of that in the one observable universe—there may not be a "distribution" of the quantity even in principle. So priors are weird when it comes to constraining physical law. Again, I am not sure, but I think the best we can hope for is that the prior does not matter much.

Meaningless or not, however, we *have* to specify a prior if we want the posterior. We just can't say anything about the posterior otherwise.

The last quantity is the model evidence $P(\vec{d})$, which is the distribution of data sets we could get over *all* choices of $\vec{\theta}$ (when distributed as the prior). It can be calculated as:

$$P(\vec{d}) = \int \mathrm{d}^N\vec{\theta} P(\vec{d}|\vec{\theta})P(\vec{\theta}) \tag{5}$$

This is most definitely very hard to calculate. It is important in comparing different models M_1 and M_2, where the ratio of evidences is called the *Bayes factor* and is a measure of the odds of one model to another. It is closely related to the likelihood ratio in frequentist statistics. As in that case, although the evidence is a probability distribution, one can imagine integrals that would lead to "goodness-of-fit" or significance measures based on $P(\vec{d})$ (in the simplest frequentist case, this is usually expressed as "under the model such-and-so fraction of experiments are expected to yield χ^2 greater than observed" (the p-value).

A Bayesian analysis seeks the posterior distribution of $\vec{\theta}$ given the obtained data $\vec{d}$, under the model M. If the likelihood can be calculated, then it is pretty straightforward to calculate something proportional to the posterior:

$$P(\vec{\theta}|\vec{d}) \propto P(\vec{d}|\vec{\theta})P(\vec{\theta}) \tag{6}$$

I could make a very big, very fine grid over all of my choices $\vec{\theta}$ covered by a non-zero prior in all the parameter dimensions, and then I'd have a great description of the posterior. I could even estimate the value of $P(\vec{d})$ for the $\vec{d}$ I obtained, since it would just be the integral of the right-hand side of the proportionality (since the left-hand side is a normalized probability distribution). I could find the mode (the most probable value of the parameters), the mean:

$$\left\langle\vec{\theta}\right\rangle = \int \mathrm{d}^N\vec{\theta} P(\vec{\theta}|\vec{d})\vec{\theta} \tag{7}$$

the variances and covariances around the mean:

$$C_{ij} = \langle \theta_i \theta_j \rangle = \int \mathrm{d}^N \vec{\theta} P(\vec{\theta}|\vec{d}) \left(\theta_i - \langle\theta_i\rangle\right) \left(\theta_j - \langle\theta_j\rangle\right) \tag{8}$$

and all the other moments.

That's actually a pretty sensible approach for two and sometimes three-dimensional models. But . . . not for much larger than that! If I have 20 parameters in my model there's no way I'm making a grid to cover it. Also, doing the above integrals is a pain. For high-ish dimensional work MCMC gives a good approach.

Please note that the value of MCMC is when you need this distribution—if you just need the *peak*, just optimize for the peak using BFGS or something like that! Also note that although we use the language of Bayesian inference in this section, you can also do pretty much exactly the same set of calculations and reinterpret it as a frequentist result (that is, characterizing the likelihood). In fact, as we will see MCMC really is just a method of generating any distribution with some desired distribution (i.e. it doesn't need to be a posterior probability) so this method can be used, for example, in Monte Carlo integration for importance sampling and similar applications.

2. MCMC Concept

MCMC uses a "Markov chain." A Markov chain is a sequence of points $\vec{\theta}_n$ where the distribution of point $n+1$ *only* depends on point n; i.e. we can write this distribution as $p(\vec{\theta}_{n+1}|\vec{\theta}_n)$.

We want this sequence to have the property that it is distributed according to the posterior. Usually this is referred to as "ergodicity," a term which confuses non-mathematicians and as far as I can tell isn't really a well-defined enough term for the mathematicians.

For any desired distribution $\pi(\vec{\theta})$, we can get a Markov chain that will be distributed as π if:

$$\pi(\vec{\theta}_1)p(\vec{\theta}_2|\vec{\theta}_1) = \pi(\vec{\theta}_2)p(\vec{\theta}_1|\vec{\theta}_2) \tag{9}$$

This can be shown by asking, for a set of points $\vec{\theta}_1$ distributed as π, what is the distribution of their subsequent points $\vec{\theta}_2$? This distribution can be calculated as follows if the Markov chain has the property in Equation 9:

$$\begin{aligned} p(\vec{\theta}_2) &= \int \mathrm{d}^N\vec{\theta}_1 p(\vec{\theta}_2|\vec{\theta}_1)\pi(\vec{\theta}_1) \\ &= \int \mathrm{d}^N\vec{\theta}_1 p(\vec{\theta}_1|\vec{\theta}_2)\pi(\vec{\theta}_2) \\ &= \int \pi(\vec{\theta}_2)\mathrm{d}^N\vec{\theta}_1 p(\vec{\theta}_1|\vec{\theta}_2) \\ &= \int \pi(\vec{\theta}_2) \end{aligned} \tag{10}$$

So, if the Equation 9 holds, then the distribution π reproduces itself, i.e. is an equilibrium distribution. This means I can start at any point $\vec{\theta}_0$ and *eventually* the resulting Markov chain will have the posterior distribution. Note that there is a question of "burn-in" time that has to be dealt with, if you happen to start at a very unlikely point in the distribution, which generally you will.

3. Metropolis-Hastings

The Metropolis-Hastings algorithm is designed to produce a Markov chain with the properties above. At each step we have $\vec{\theta}_i$ at which we have evaluated the posterior $\pi(\vec{\theta}_i)$. Then we can:

- Generate a candiate $\vec{\theta}_{i+1}$ by drawing from a "proposal distribution" $q(\vec{\theta}_{i+1}|\vec{\theta}_i)$. A Gaussian distribution is a common choice.
- Calculate an acceptance probability:

$$\alpha(\vec{\theta}_{i+1},\vec{\theta}_i) = \min\left(1, \frac{\pi(\vec{\theta}_{i+1})q(\vec{\theta}_i|\vec{\theta_{i+1}})}{\pi(\vec{\theta}_i)q(\vec{\theta}_{i+1}|\vec{\theta}_i)}\right) \tag{11}$$

- With a probability α, accept the candidate and move on; otherwise toss it, and try again with a new candidate $\vec{\theta}_{i+1}$.

How does this generate a distribution that satisfies Equation 9? First note that distribution of $\vec{\theta}_{i+1}$ is:

$$p(\vec{\theta}_{i+1}|\vec{\theta}_i) = q(\vec{\theta}_{i+1}|\vec{\theta}_i)\alpha(\vec{\theta}_{i+1},\vec{\theta}_i) \tag{12}$$

Then we can write:

$$\pi(\vec{\theta}_i)q(\vec{\theta}_{i+1}|\vec{\theta}_i)\alpha(\vec{\theta}_{i+1},\vec{\theta}_i) = \min\left(\pi(\vec{\theta}_{i+1})q(\vec{\theta}_i|\vec{\theta_{i+1}}), \pi(\vec{\theta}_i)q(\vec{\theta}_{i+1}|\vec{\theta}_i)\right) \tag{13}$$

and we can ask what the same quantity would be if we reverse i and $i+1$:

$$\pi(\vec{\theta}_{i+1})q(\vec{\theta}_i|\vec{\theta_{i+1}})\alpha(\vec{\theta}_i,\vec{\theta}_{i+1}) = \min\left(\pi(\vec{\theta}_i)q(\vec{\theta}_{i+1}|\vec{\theta}_i), \pi(\vec{\theta}_{i+1})q(\vec{\theta}_i|\vec{\theta_{i+1}})\right) \tag{14}$$

So these are the same:

$$\pi(\vec{\theta}_i)q(\vec{\theta}_{i+1}|\vec{\theta}_i)\alpha(\vec{\theta}_{i+1},\vec{\theta}_i) = \pi(\vec{\theta}_{i+1})q(\vec{\theta}_i|\vec{\theta_{i+1}})\alpha(\vec{\theta}_i,\vec{\theta}_{i+1}) \tag{15}$$

And then using Equation 12:

$$\pi(\vec{\theta}_i)p(\vec{\theta}_{i+1}|\vec{\theta}_i) = \pi(\vec{\theta}_{i+1})p(\vec{\theta}_i|\vec{\theta}_{i+1}) \tag{16}$$

i.e. the detailed balance Equation 9.

4. Why use MCMC?

One uses MCMC when you have a large enough number of parameters, when your model is nonlinear in its parameters, and when you need to understand the error distribution extremely well. For example, when constraining cosmological parameters, we have all of these conditions—a dozen parameters, a complex physical model interposed between the model parameters and the data, and a desire to know the uncertainties and their covariances. Please note that if your model happens to be linear, you don't need to need MCMC!

The utility of the MCMC becomes clear when you think about measuring the error in the $\vec{\theta}$ values. The posterior distribution is often some complex shape in N-dimensions, and it is the probability density in all coordinates. Often we are interested in just one parameter, say θ_0. To calculate its posterior we need this integral:

$$P(\theta_0|\vec{d})P(\theta_0) = \int \mathrm{d}^{N-1}\vec{\theta}_{N-1} P(\vec{\theta}|\vec{d})P(\vec{\theta}) \tag{17}$$

which could be quite annoying. With MCMC results though, it is trivial: just look at your θ_0 samples!

In fact, you can do this for *any function of* $\vec{\theta}$ $f(\vec{\theta})$*!!*

5. Burn-in and the autocorrelation time

In the implementation of an MCMC, you have to start somewhere. Wherever you start will bias the distribution that you get. If you start far away from the peak of the distribution, the MCMC will wander towards the peak along some particular path, through areas of very low values of the probability distribution. The points along that path are highly overrepresented! But even if you start near the peak of the distribution, for a while the points will be near (sort of "remembering") the starting point, and this too is biased.

Finally, this discussion makes it clear that the MCMC points are not independent, so (for example) if we want to know the error in our MCMC determination of the mean of the posterior along some quantity (*note this is not the error in the quantity!*).

We need to be able to determine how many steps we need to take for the burn-in, and how many truly "independent" steps we have taken. The usual tool is the autocorrelation function and the autocorrelation time.

6. Proposal distribution tuning

An important issue to deal with in MCMC is the proposal distribution. In the simplest case of Metropolis-Hastings and an isotropic Gaussian distribution in the parameters, a number of issues can occur. The most obvious is that the steps (the width of the Gaussian) need to be about the right length. If you take very large steps, the acceptance probability will be extremely low and you can spend a long time waiting for your next step to be accepted. If you take very small steps, most of them will be accepted but this is because you are only very slowly updating. So you will only find the peaks of the distribution very slowly, and if you have a multimodal distribution your exploration will take so long it will be hard to traverse from peak to peak. The acceptance probability you want is of order 0.5, though as far as I know there is no hard-and-fast known criterion that is "optimal" in some sense.

There is however another issue, which is that in the general case an isotropic proposal distribution is highly inefficient. If π has a narrow ridge (aligned or not with the axes) the isotropic proposals will either be too big in the narrow dimension of the ridge, or too small in the long dimension of the ridge.

There are various ways to deal with this situation. The most obvious idea is to have an adaptive covariance in the proposal distribution that tunes to the distribution as you go in order to get the right acceptance probability. However, if you just write down a version of that adaptive method that looks like it would work heuristically, it is highly unlikely to still satisfy the detailed balance equations—you are building in a dependence of q on the distribution π.

There are methods for doing this that depend on the amount of adaptation essentially vanishing over the long haul; the simplest algorithms use the current estimates of the covariances of the distribution to set the size and shape of the proposal distribution.

Often practicioners will do something like this "by hand." Two common approaches:

- Perform a set of burn-in experiments to tune the step sizes based on the acceptance probabilities, and then use a particular (fixed) choice for the full MCMC.
- Optimize the posterior and estimate the Hessian at the optimum, start near the optimum to reduce burn-in time (though you still need burn-in!), and use the Hessian to set the covariance of the Gaussian for the proposals.

In a small number of dimensions these issues are not so bad—but in the typical case of 10 dimensions they are pretty bad.

7. Affine Invariant MCMC

However, there are methods where there are only one or two parameters that need to be tuned no matter what the number of dimensions. One class of methods is known as *affine invariant MCMC*, and is the basis of `emcee`, written by one of your ancestors as PhD students at NYU, Daniel Foreman-Mackey.

The idea here is to create a process whose steps which would be the same even if you transformed $\vec{\theta}$ under some general affine transformation. Mathematical details aside, an affine transformation is just this:

$$\vec{\theta'} = \mathbf{A} \cdot \vec{\theta} + \vec{b} \tag{18}$$

i.e. a linear transformation plus a shift. This can scale, shear, rotate, whatever.

Specifically we want it to be the case that our procedure for choosing the next step commmutes with any affine transformation of coordinates. If we start one step n by transforming $\vec{\theta}_n$ to $\vec{\theta'}_n$, and then take the step to $\vec{\theta'}_{n+1}$, that should be the same as taking the step to $\vec{\theta}_{n+1}$ and performing the same affine transformation. If this is true then it cannot possibly matter what the (local) shape of the posterior distribution is, since the affine transformation can always distort it to something isotropic.

Note that clearly our isotropic Gaussian does not satify this condition! Under the affine transformation that distribution will clearly be distorted.

It might seem hard to guarantee this property! Indeed with a single "walker" I have a hard time conceiving of doing so. And somehow you *also* have to make sure that the steps obey the ergodic properties necessary.

The procedure, due to Goodman & Weare (2010), is to use K walkers. The proposed step for each walker is based on the location of the other $K-1$ walkers. Specifically, if we are considering a walker k, we draw another walker at random j.

$$\vec{\theta}^{(k)}_{i+1} = \vec{\theta}^{(j)}_i + Z\left[\vec{\theta}^{(k)}_i - \vec{\theta}^{(j)}_i\right] \tag{19}$$

with a random variable Z, where we will see in a second that Z will range between ~ 0.7–1.4. That is, we just move walker k a little closer or further away from walker j.

This proposal is clearly affine-invariant—if I linearly map everything, it doesn't matter if I do it before or after the proposal. Also, you can see that *if* the walkers have the property that they are exploring the posterior in a reasonable way, the proposal steps will be along the locally covariant directions of the posterior. This means that the acceptance criterion will be much more efficient.

If we choose Z from a distribution $g(z)$ such that $g(z^{-1}) = zg(z)$ then the proposal distribution is symmetric between i and $i+1$. We can see this with the slight rearrangement:

$$\begin{aligned} \vec{\theta}^{(k)}_{i+1} &= \vec{\theta}^{(j)}_i + Z\left[\vec{\theta}^{(k)}_i - \vec{\theta}^{(j)}_i\right] \\ \left[\vec{\theta}^{(k)}_{i+1} - \vec{\theta}^{j}_i\right] &= Z\left[\vec{\theta}^{(k)}_i - \vec{\theta}^{(j)}_i\right] \end{aligned} \tag{20}$$

So if Z is distributed as $g(z)$ then:

$$q(\vec{\theta}^{i+1}_k | \vec{\theta}^{i}_k) = \frac{1}{\Delta_i} g(z) \tag{21}$$

where Δ_i is the distance between j and k at step i. To be symmetric we must have:

$$q(\vec{\theta}_k^i|\vec{\theta}_k^{i+1}) = \frac{1}{\Delta_{i+1}} g(z^{-1}) = \frac{1}{\Delta_i} z^{-1} g(z^{-1}) \tag{22}$$

Somewhat less easy to show is that the acceptance criterion turns into:

$$\alpha = \min\left(1, Z^{N-1} \frac{\pi(\vec{\theta}_{i+1})}{\pi(\vec{\theta}_i)}\right) \tag{23}$$

This relies on a deeper understanding of the partial resampling technique, of which this is an example, which works if the *conditional distribution* along the line is preserved. The conditional distribution is proportional to Z^{N-1} (like the density on a circle goes as $1/R$, and the density on a sphere like $1/R^2$); remember the line originates at point j so there are not worries about the divergence at zero. The key point is that this procedure yields detailed balance.

It is very important that the steps not be taken *all* in parallel. That, it turns out, destroys the detailed balance. But you can take *half* of the walkers, and make steps based on the locations of the other half (chosen at random, with replacement), and then do the opposite. This allows the walkers to be updated with parallelization.

The usual choice is:

$$g(z) \propto \frac{1}{\sqrt{z}} \tag{24}$$

for $1/a < z < a$, with $a \sim 2$, and zero otherwise.