AIML 168

Algorithmic Trading

Lecture 3:

Recall: Conjugate Priors

MCMC

MLE and MAP

The Metropolis Algorithm

Weiqing Gu

https://algorithm-trading.github.io/

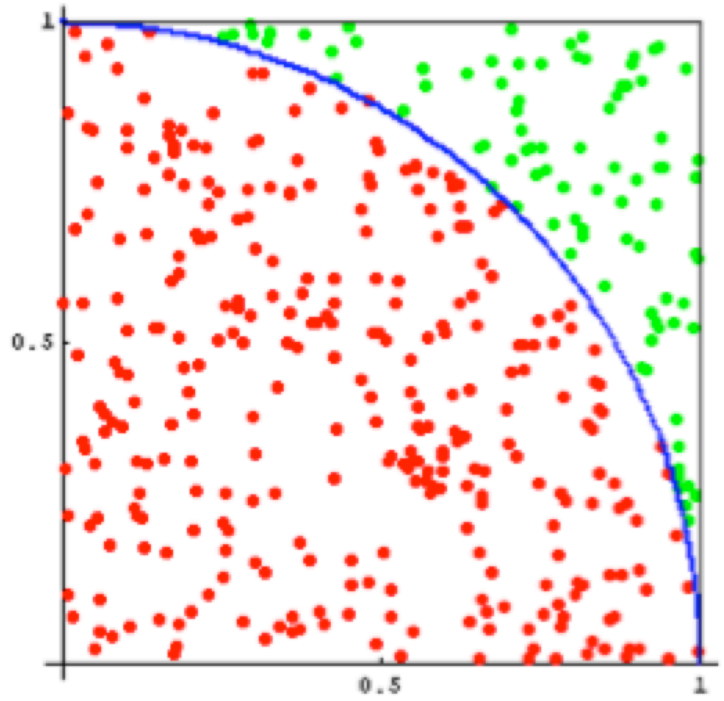
Today's topics

(closely follow our textbook)

- Recall: Bayesian inference and **Conjugate Priors**
- Markov Chain Monte Carlo (MCMC)
- MLE and MAP
- The Metropolis Algorithm
- Coding Session

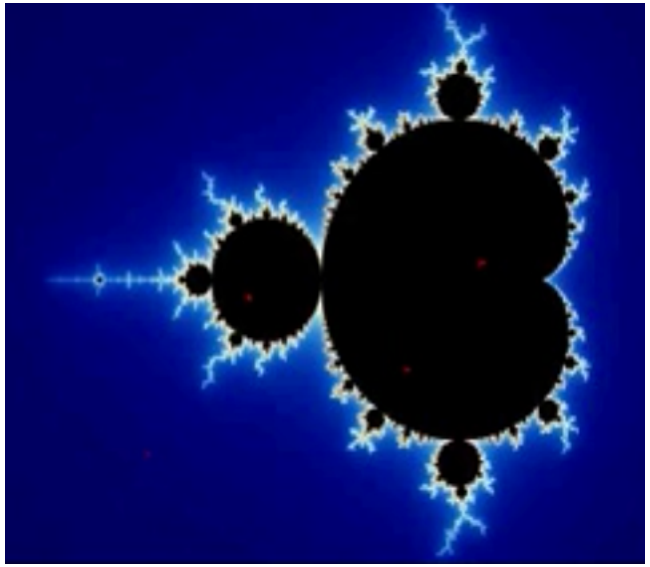
Idea of approximation for intractable large sums & integrals.

- The approximation and sampling methods started first from Monte Carlo Approximation.
- A ground broken idea:



Example: Estimate the area of the Mandelbrot set.

- Let A = Mandelbrot set (the black area).
- Let S = [-2,2] x [-2, 2] be
  the square.
- Put rice in the square box and
- Count the total number of
  rice in S & in A separately.
- Find m(A)/m(S).



Recall: Bayesian Inference $P(\theta|D) = P(D|\theta)\ P(\theta)\ /\ P(D)$

Our goal in carrying out Bayesian Statistics is to *produce quantitative trading strategies based on Bayesian models*. However, in order to reach that goal we need to consider a reasonable amount of Bayesian Statistics theory. So far we have:

- Introduced the philosophy of Bayesian Statistics, making use of Bayes' Theorem to update our prior beliefs on probabilities of outcomes based on new data
- Used conjugate priors as a means of simplifying computation of the posterior distribution in the case of inference on a binomial proportion

Today we are going to discuss MCMC (Markov Chain Monte Carlo) as a means of computing the posterior distribution when conjugate priors are not applicable.

Recall: Why is a Beta Prior Conjugate to the Bernoulli Likelihood?

We can actually use a simple calculation to prove why the choice of the beta distribution for the prior, with a Bernoulli likelihood, gives a beta distribution for the posterior.

As mentioned above, the probability density function of a beta distribution, for our particular parameter θ, is given by:

$$P(\theta|\alpha,\beta) = \theta^{\alpha-1}(1-\theta)^{\beta-1}/B(\alpha,\beta) \tag{3.12}$$

You can see that the form of the beta distribution is similar to the form of a Bernoulli likelihood. In fact, if you multiply the two together (as in Bayes' rule), you get:

$$\theta^{\alpha-1}(1-\theta)^{\beta-1}/B(\alpha,\beta) \times \theta^{k}(1-\theta)^{1-k} \propto \theta^{\alpha+k-1}(1-\theta)^{\beta+k} \tag{3.13}$$

Notice that the term on the right hand side of the proportionality sign has the same form as our prior (up to a normalising constant).

Multiple Ways to Specify a Beta Prior

At this stage we've discussed the fact that we want to use a beta distribution in order to specify our prior beliefs about the fairness of the coin. However, we only have two parameters to play with, namely α and β.

How do these two parameters correspond to our more intuitive sense of "likely fairness" and "uncertainty in fairness"?

Well, these two concepts neatly correspond to the *mean* and the *variance* of the beta distribution. Hence, if we can find a relationship between these two values and the α and β parameters, we can more easily specify our beliefs.

It turns out that the mean μ is given by:

$$\mu = \frac{\alpha}{\alpha + \beta} \tag{3.14}$$

While the standard deviation σ is given by:

$$\sigma = \sqrt{\frac{\alpha\beta}{(\alpha + \beta)^2(\alpha + \beta + 1)}} \tag{3.15}$$

Hence, all we need to do is re-arrange these formulae to provide α and β in terms of μ and σ. α is given by:

$$\alpha = \left(\frac{1-\mu}{\sigma^2} - \frac{1}{\mu} \right) \mu^2 \tag{3.16}$$

While β is given by:

$$\beta = \alpha \left(\frac{1}{\mu} - 1 \right) \tag{3.17}$$

Note that we have to be careful here, as we should not specify a $\sigma > 0.289$, since this is the standard deviation of a uniform density (which itself implies no prior belief on *any particular* fairness of the coin).

Using Bayes' Rule to Calculate a Posterior

We are finally in a position to be able to calculate our posterior beliefs using Bayes' rule.

Bayes' rule in this instance is given by:

$$P(\theta|z,N) = P(z,N|\theta)P(\theta)/P(z,N) \tag{3.18}$$

This says that the posterior belief in the fairness θ, given z heads in N flips, is equal to the *likelihood* of seeing z heads in N flips, given a fairness θ, multiplied by our *prior belief* in θ, normalised by the *evidence*.

If we substitute in the values for the likelihood function calculated above, as well as our prior belief beta distribution, we get:

$$\begin{aligned} P(\theta|z,N) &= P(z,N|\theta)P(\theta)/P(z,N) && (3.19)\\ &= \theta^z(1-\theta)^{N-z}\theta^{\alpha-1}(1-\theta)^{\beta-1}/\left[B(\alpha,\beta)P(z,N)\right] && (3.20)\\ &= \theta^{z+\alpha-1}(1-\theta)^{N-z+\beta-1}/B(z+\alpha,N-z+\beta) && (3.21) \end{aligned}$$

The denominator function $B(.,.)$ is known as the **Beta function**, which is the correct normalising function for a beta distribution, as discussed above.

If our prior is given by $\text{beta}(\theta|\alpha, \beta)$ *and we observe* z *heads in* N *flips subsequently, then the posterior is given by* $\text{beta}(\theta|z + \alpha, N - z + \beta)$.

This is an incredibly straightforward (and useful!) updating rule. All we need do is specify the mean μ and standard deviation σ of our prior beliefs, carry out N flips and observe the number of heads z and we automatically have a rule for how our beliefs should be updated.

As an example, suppose we consider the same prior beliefs as above for θ with $\mu = 0.5$ and $\sigma = 0.1$. This gave us the prior belief distribution of $\text{beta}(\theta|12, 12)$.

Now suppose we observe $N = 50$ flips and $z = 10$ of them come up heads. How does this change our belief on the fairness of the coin?

We can plug these numbers into our posterior beta distribution to get:

$$\begin{aligned} \text{beta}(\theta|z + \alpha, N - z + \beta) &= \text{beta}(\theta|10 + 12, 50 - 10 + 12) && (3.22) \\ &= \text{beta}(\theta|22, 52) && (3.23) \end{aligned}$$

At this stage we can compute the mean and standard deviation of the posterior in order to produce estimates for the fairness of the coin. In particular, the value of μ_{post} is given by:

$$\mu_{\text{post}} = \frac{\alpha}{\alpha + \beta} \tag{3.24}$$

$$= \frac{22}{22 + 52} \tag{3.25}$$

$$= 0.297 \tag{3.26}$$

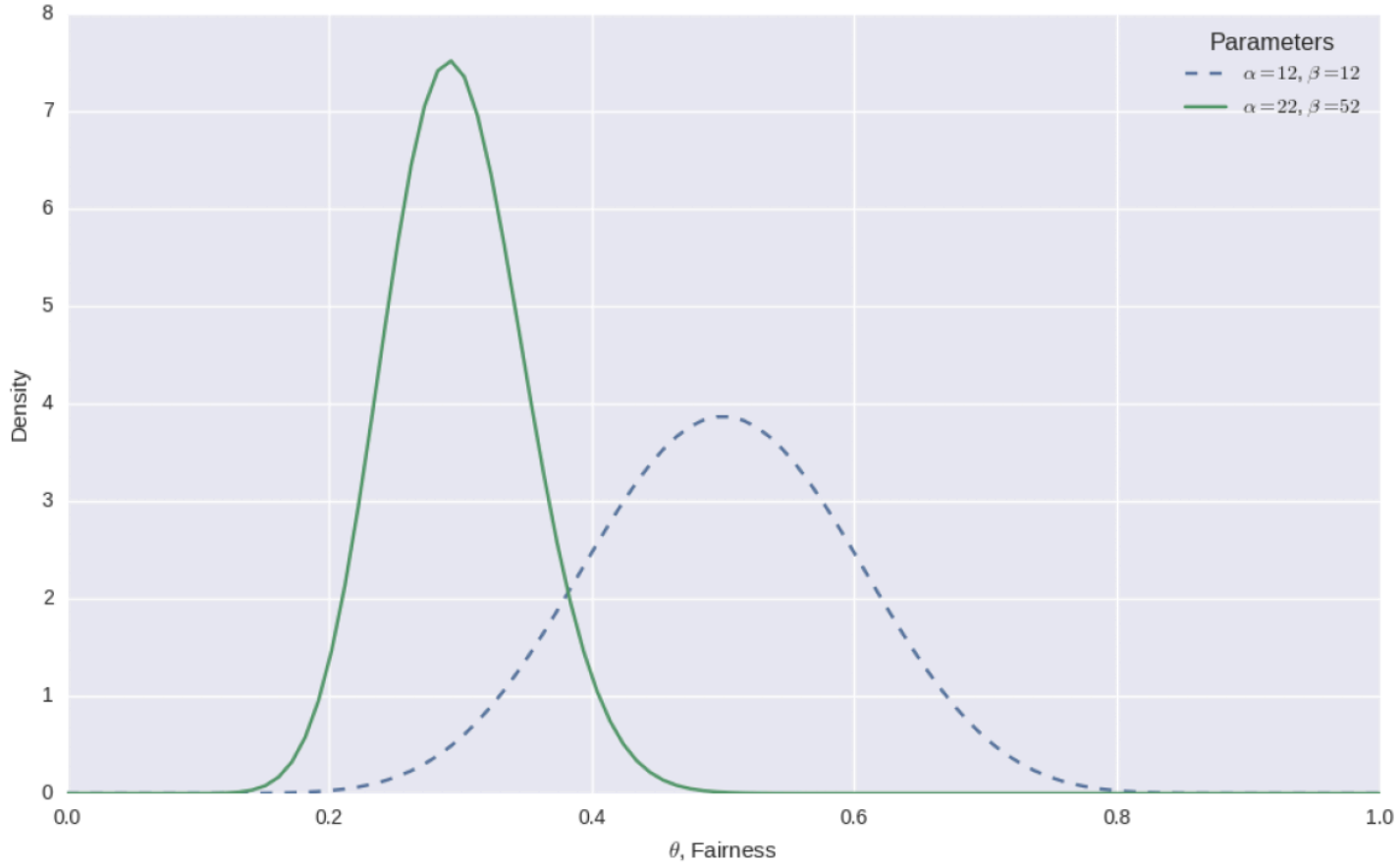
$$\tag{3.27}$$

While the standard deviation σ_{post} is given by:

$$\sigma_{\text{post}} = \sqrt{\frac{\alpha\beta}{(\alpha + \beta)^2(\alpha + \beta + 1)}} \tag{3.28}$$

$$= \sqrt{\frac{22 \times 52}{(22 + 52)^2(22 + 52 + 1)}} \tag{3.29}$$

$$= 0.053 \tag{3.30}$$

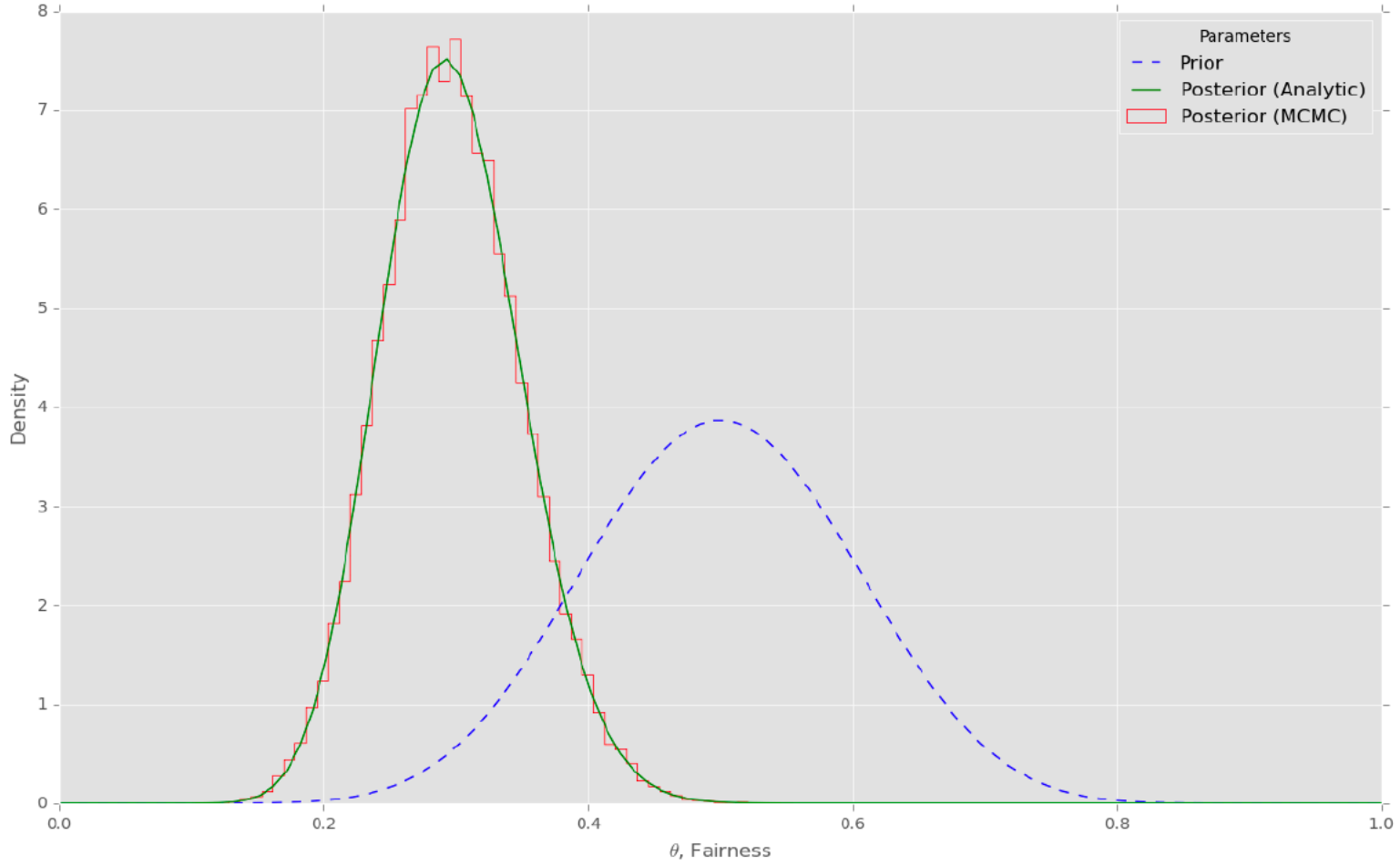


A blue dotted line for the prior belief and a green solid line for the posterior.

Figure 3.3: The prior and posterior belief distributions about the fairness θ.

Notice how the peak shifts dramatically to the left since we have only observed 10 heads in 50 flips. In addition, notice how the width of the peak has shrunk, which is indicative of the fact that our belief in the certainty of the particular fairness value has also increased.

We also can use MCMC to approximate the posterior probability distribution. (The green solid curve).



Why Markov Chain Monte Carlo for Approximate a Posterior?

We saw that conjugate priors gave us a significant mathematical "short-cut" to calculating the posterior distribution in Bayes' Rule.

Q: why do we need MCMC at all if we can simply use conjugate priors?

A: **Not all models can be succinctly stated in terms of conjugate priors**.

In particular, many more complicated modelling situations, particularly those related to hierarchical models with hundreds of parameters, which we will consider in later, are completely intractable using analytical methods.

If we recall Bayes' Rule:

$$P(\theta|D) = \frac{P(D|\theta)P(\theta)}{P(D)} \tag{4.1}$$

We can see that we need to calculate the *evidence* $P(D)$. In order to achieve this we need to evaluate the following integral, which integrates over all possible values of θ, the parameters:

$$P(D) = \int_{\Theta} P(D, \theta) \mathrm{d}\theta \tag{4.2}$$

The fundamental problem is that we are often unable to evaluate this integral analytically and so we must turn to a numerical approximation method instead.

Another problem: Curse of Dimensionality

- An additional problem is that our models might require a large number of parameters.
- This means that our prior distributions could potentially have a large number of dimensions.
- This in turn means that our posterior distributions will also be high dimensional.
- Hence, we are in a situation where we have to numerically evaluate an integral in a potentially very large dimensional space.
- This means we are in a situation often described as the Curse of Dimensionality.
- Informally, this means that the volume of a high-dimensional space is so vast that any available data becomes extremely sparse within that space.

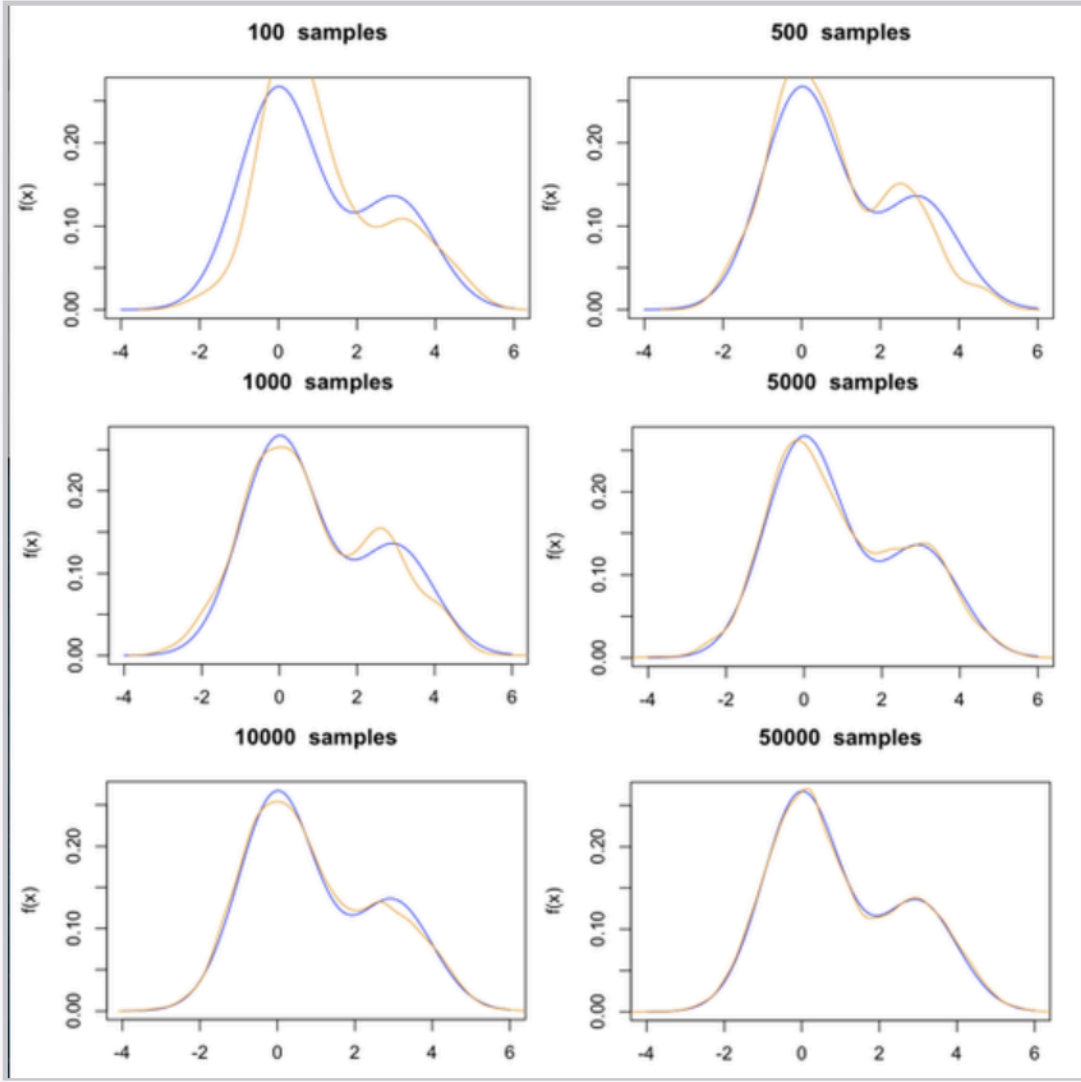
Such problems are often extremely difficult to tackle unless they are approached in an intelligent manner. The motivation behind Markov Chain Monte Carlo methods is that they perform an intelligent search within a high dimensional space and thus Bayesian Models in high dimensions become tractable.

The basic idea is to sample from the posterior distribution by combining a "random search" (the Monte Carlo aspect) with a mechanism for intelligently "jumping" around, but in a manner that ultimately doesn't depend on where we started from (the Markov Chain aspect). Hence Markov Chain Monte Carlo methods are memoryless searches performed with intelligent jumps.

As an aside, MCMC is not just for carrying out Bayesian Statistics. It is also widely used in computational physics and computational biology as it can be applied generally to the approximation of any high dimensional integral.

MCMC methods are primarily used for calculating numerical approximations of multi-dimensional integrals, for example in Bayesian statistics, computational physics,[1] computational biology[2] and computational linguistics.[3][4]

In Bayesian statistics, the recent development of MCMC methods has made it possible to compute large hierarchical models that require integrations over hundreds to thousands of unknown parameters.[5]



Convergence of the Metropolis–Hastings algorithm. Markov chain Monte Carlo attempts to approximate the blue distribution with the orange distribution.

Markov Chain Monte Carlo is a family of algorithms

- Markov Chain Monte Carlo is a family of algorithms, rather than one particular method.
- In this section we are going to concentrate on a particular method known as the **Metropolis Algorithm**.
- Later, we will consider
  - Metropolis-Hastings
  - the Gibbs Sampler
  - Hamiltonian MCMC
  - the No-U-Turn Sampler (NUTS)
- The latter is actually incorporated into **PyMC3**, the software we'll be using to conduct numerically inference.

The Metropolis Algorithm

The first MCMC algorithm considered in this chapter is due to Metropolis[36], which was developed in 1953. Hence it is not a recent method! While there have been substantial improvements on MCMC sampling algorithms since, it will suffice for this section. The intuition gained on this simpler method will help us understand more complex samplers in later sections and chapters.

The basic recipes for most MCMC algorithms tend to follow this pattern (see Davidson-Pilon[19] for more details):

1. Begin the algorithm at the *current* position in parameter space (θ_{current})
2. Propose a "jump" to a new position in parameter space (θ_{new})
3. Accept or reject the jump probabilistically using the prior information and available data
4. If the jump is accepted, move to the new position and return to step 1
5. If the jump is rejected, stay where you are and return to step 1
6. After a set number of jumps have occurred, return all of the *accepted* positions

The main difference between MCMC algorithms occurs in *how you jump* as well as *how you decide whether to jump*.

The Metropolis algorithm uses a normal distribution to propose a jump. This normal distribution has a mean value μ which is equal to the current position and takes a "proposal width" for its standard deviation σ.

This proposal width is a parameter of the Metropolis algorithm and has a significant impact on convergence. A larger proposal width will jump further and cover more space in the posterior distribution, but might miss a region of higher probability initially. However, a smaller proposal width won't cover as much of the space as quickly and thus could take longer to converge.

A normal distribution is a good choice for such a proposal distribution (for continuous parameters) as, by definition, it is more likely to select points nearer to the current position than further away. However, it will occassionally choose points further away, allowing the space to be explored.

Once the jump has been proposed, we need to decide (in a probabilistic manner) whether it is a good move to jump to the new position. How do we do this? We calculate the ratio of the proposal distribution of the *new* position and the proposal distribution at the *current* position to determine the probability of moving, p:

$$p = P(\theta_{\text{new}})/P(\theta_{\text{current}}) \tag{4.3}$$

We then generate a uniform random number on the interval $[0, 1]$. If this number is contained within the interval $[0, p]$ then we accept the move, otherwise we reject it.

While this is a relatively simple algorithm it isn't immediately clear why this makes sense and how it helps us avoid the intractable problem of calculating a high dimensional integral of the evidence, $P(D)$.

As Thomas Wiecki[50] points out in his article on MCMC sampling, we're actually dividing the posterior of the proposed parameter by the posterior of the current parameter. Utilising Bayes' Rule this eliminates the evidence, $P(D)$ from the ratio:

$$\frac{P(\theta_{\text{new}}|D)}{P(\theta_{\text{current}}|D)} = \frac{\frac{P(D|\theta_{\text{new}})P(\theta_{\text{new}})}{P(D)}}{\frac{P(D|\theta_{\text{current}})P(\theta_{\text{current}})}{P(D)}} = \frac{P(D|\theta_{\text{new}})P(\theta_{\text{new}})}{P(D|\theta_{\text{current}})P(\theta_{\text{current}})} \tag{4.4}$$

The right hand side of the latter equality contains only the likelihoods and the priors, both of which we can calculate easily. Hence by dividing the posterior at one position by the posterior at another, we're sampling regions of higher posterior probability more often than not, in a manner which fully reflects the probability of the data.

Metropolis algorithm as one of MCMC

Jacky is going to illustrate Metropolis algorithm, using PyMC3 we will consider more sophisticated samplers and then apply them to more complex models.

Ultimately, we will arrive at the point where our models are useful enough to provide insight into asset returns prediction.

At that stage we will be able to begin building a trading model from our Bayesian analysis.

MLE = Maximum Likelihood Estimate

Assume that we want to estimate an unobserved population parameter θ on the basis of observations x. Let f be the sampling distribution of x, so that $f(x \mid \theta)$ is the probability of x when the underlying population parameter is θ. Then the function:

$$\theta \mapsto f(x \mid \theta)$$

is known as the likelihood function and the estimate:

$$\hat{\theta}_{\mathrm{MLE}}(x) = \arg\max_{\theta} f(x \mid \theta)$$

is the maximum likelihood estimate of θ.

MAP = Maximum A Posteriori estimation

Now assume that a prior distribution g over θ exists. This allows us to treat θ as a random variable as in Bayesian statistics. We can calculate the posterior distribution of θ using Bayes' theorem:

$$\theta \mapsto f(\theta \mid x) = \frac{f(x \mid \theta)\, g(\theta)}{\displaystyle\int_{\Theta} f(x \mid \vartheta)\, g(\vartheta)\, d\vartheta}$$

where g is density function of θ, Θ is the domain of g.

The method of maximum a posteriori estimation then estimates θ as the mode of the posterior distribution of this random variable:

$$\hat{\theta}_{\mathrm{MAP}}(x) = \arg\max_{\theta} f(\theta \mid x) = \arg\max_{\theta} \frac{f(x \mid \theta)\, g(\theta)}{\displaystyle\int_{\Theta} f(x \mid \vartheta)\, g(\vartheta)\, d\vartheta} = \arg\max_{\theta} f(x \mid \theta)\, g(\theta).$$

The denominator of the posterior distribution (so-called marginal likelihood) is always positive and does not depend on θ and therefore plays no role in the optimization. Observe that the MAP estimate of θ coincides with the ML estimate when the prior g is uniform (that is, a constant function).

Coding Session on MCMC

- By our TA Jacky Lee

Monte Carlo Approximation (or Method)

- Key: Mathematically, we can view the procedure on the previous slide as finding E[f(X)], for certain f and X.
- How?
- *Work out details with students on the board*.