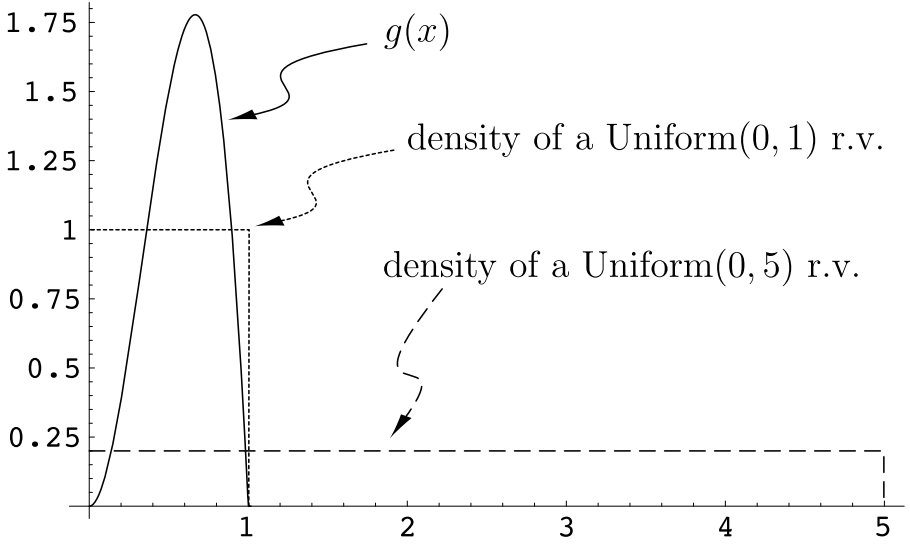
Importance Sampling and $(MC)^3$

The goals of this lecture are:

- Introduce *importance sampling*—a Monte Carlo variance reduction technique
- Discuss MCMC in simple pedigrees and discuss problems of reducibility
- Discuss poor mixing more generally and describe Metropolis-coupled MCMC

The opposite of importance sampling—"minimal relevance" sampling:

Let's say you wanted to estimate the area under the curve $g(x)$ below by Monte Carlo:



Note that we are back to "vanilla" Monte Carlo (not MCMC) for the next five slides or so. . .

- We could simulate random variables $U \sim \text{Uniform}(0, 1)$

$$\int_0^1 g(x)dx = \int_0^1 g(x)1dx = \mathbb{E}[g(U)] \approx \frac{1}{n}\sum_{i=1}^{n} g(u^{(i)})$$

And that would work pretty well.

- We could also simulate random variables $W \sim \text{Uniform}(0, 5)$

$$\int_0^1 g(x)dx = 5\int_0^1 g(x)\frac{1}{5}dx = 5\mathbb{E}[g(W)] \approx \frac{5}{n}\sum_{i=1}^{n} g(w^{(i)})$$

- Each method is correct, but the second is clearly inefficient. Any values of $w^{(i)}$ greater than 1 don't provide us with any information about the area under $g(x)$!!

- The second method has larger Monte Carlo variance.

Importance sampling:

- Importance sampling is choosing a good distribution from which to simulate one's random variables for Monte Carlo.
- It involves multiplying the integrand by 1 (usually dressed up in a "tricky fashion") to yield an expectation of a quantity that varies less than the original integrand over the region of integration.
- For example, say we wish to approximate $\int_{x \in \mathcal{A}} g(x)dx$ and we can "dream up" a distribution $h(x)$.

$$\int_{x \in \mathcal{A}} g(x)dx = \int_{x \in \mathcal{A}} g(x)\frac{h(x)}{h(x)}dx = \int_{x \in \mathcal{A}} \frac{g(x)}{h(x)}h(x)dx$$

which is

$$\mathbb{E}_h \left[\frac{g(X)}{h(X)} \right]$$

where $\mathbb{E}_h$ denotes the expectation with respect to the density h, so long as $h(x) > 0$ for all values x for which $g(x) \neq 0$.

So we could approximate $\int_{x \in \mathcal{A}} g(x)dx$ by:

$$G_n^h = \frac{1}{n} \sum_{i=1}^{n} \frac{g(x^{(i)})}{h(x^{(i)})}$$

where each $x^{(i)}$ is simulated from the density h.

- Using the Cauchy-Schwarz inequality, it is not too difficult to show that the Monte Carlo variance of G_n^h is minimized when $h(x) \propto |g(x)|$.
- Even more straightforward, if we consider only the case where $g(x) > 0$, it is immediately apparent that the Monte Carlo variance is minimized (is zero, in fact) when $h(x) \propto g(x)$.
- Of course, if you could simulate independent r.v.'s from a distribution exactly proportional to $g(x)$, that implies you know the value of the integral anyway. So this is a little circular—but it does emphasize that importance sampling works best when $h(x)$ is close to proportional to $g(x)$.

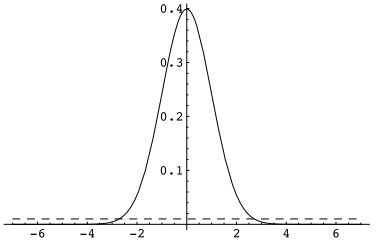
Choosing a good importance sampling distribution:

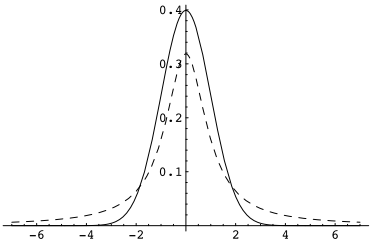
A good importance sampling function $h(x)$ should have the following properties:

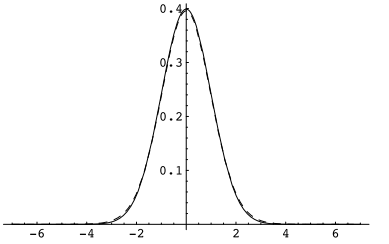
1. $h(x) > 0$ whenever $g(x) \neq 0$
2. $h(x)$ should be close to being proportional to $|g(x)|$
3. it should be easy to simulate values from $h(x)$
4. it should be easy to compute the density $h(x)$ for any value x that you might realize.

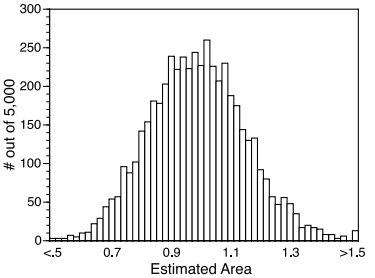
Fulfilling this wish-list in high dimensional space (where Monte Carlo techniques are most useful) is quite often a tall task.

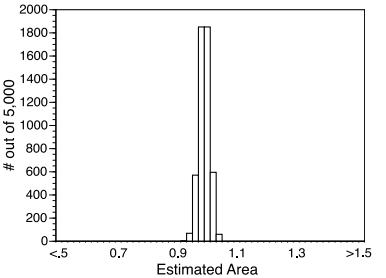
Simple example: Estimating the area under a Normal(0, 1) curve with $n = 1,000$:

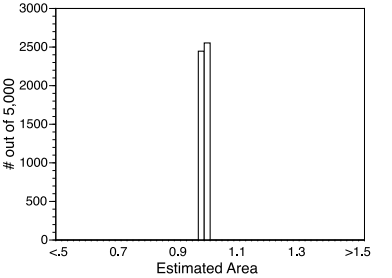












$h(x) \equiv$ Uniform $h(x) \equiv t_1$ Dist. $h(x) \equiv t_{30}$ Dist.

Examples of importance sampling in genetics:

- Griffiths and Tavare (1994) developed a "recursion" method for simulating genealogies under the coalescent model conditional on genetic data at the tips of the branches (but not *exactly* from their distribution given the data).
- This method simulates independent genealogies that can be used in Monte Carlo.
- It wasn't until somewhat later that Joe Felsenstein and his colleagues were able to dissect the method and show that it is an example of importance sampling.
- Viewing the method as an importance sampling task, Matt Stephens and Peter Donnelly (2000) were able to improve upon the method considerably.
- Anderson and Garza (2006): importance sampling to estimate very small false positive rates when doing parentage inference with SNPs.

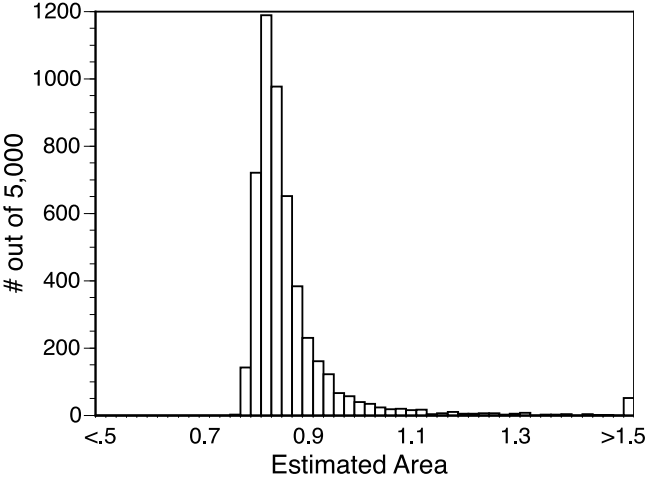
A common pitfall of importance sampling:

- Whatever is going on in the "tails" of the distribution is, unfortunately, very important...
- If $h(x)$ is thinner-tailed than $g(x)$ there could be problems: if for a rarely-realized value $x^{(i)}$, $h(x^{(i)})$ is very small, but $g(x^{(i)})$ is not terribly small, then $g(x^{(i)})/h(x^{(i)})$ could be much larger than typical ratios of $g(x)/h(x)$, leading to an unstable estimator.
- Consequently, assessing the convergence of importance sampling estimators can, at times, be quite difficult.

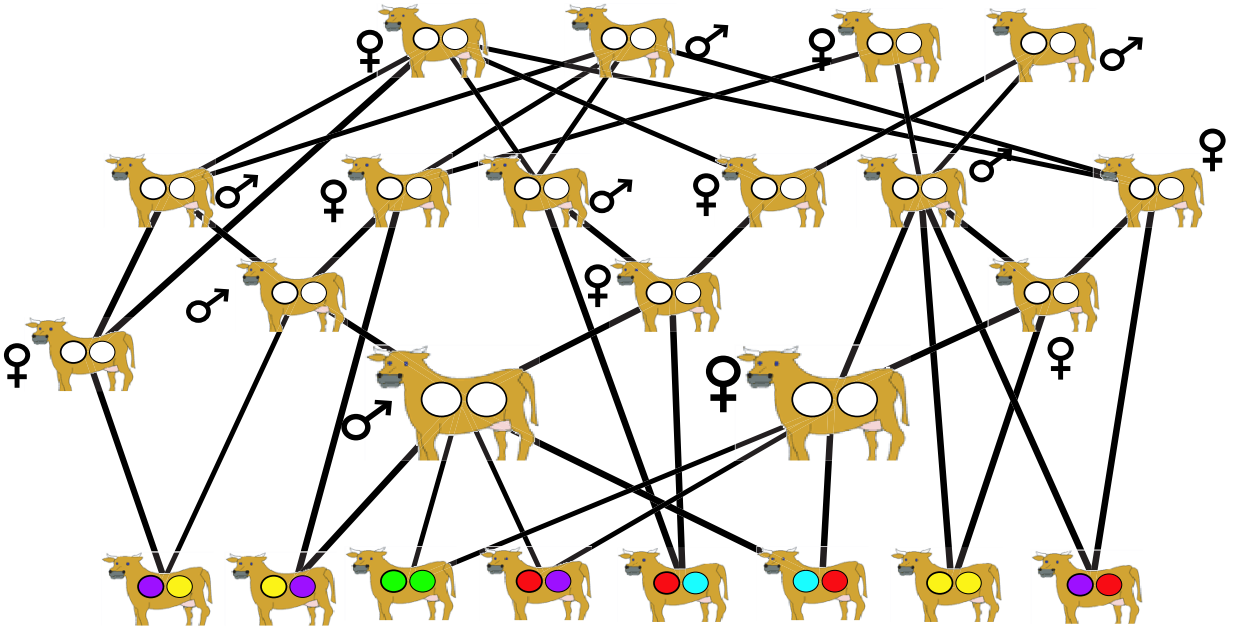
An example of a problematic thin-tailed importance sampling distribution:

Approximate the area under a thick-tailed Cauchy (t_1) distribution using a thin-tailed $\mathrm{Normal}(0, 1)$ density as the importance sampling distribution, and $n = 1,000$:

The distribution of Monte Carlo estimates appears to the right. The correct area is 1.0. Notice the number of estimates that are greater than 1.5.



Switching gears—a simple example of MCMC on a pedigree with "data at the bottom":



The colored balls represent observed allelic types at a locus.

Asking questions about the ancestors:

Simple questions:

- What is the probability that only n copies of the red allele occurred in the founders?
- What is the probability that ancestor Z carried exactly one copy of the purple allele?

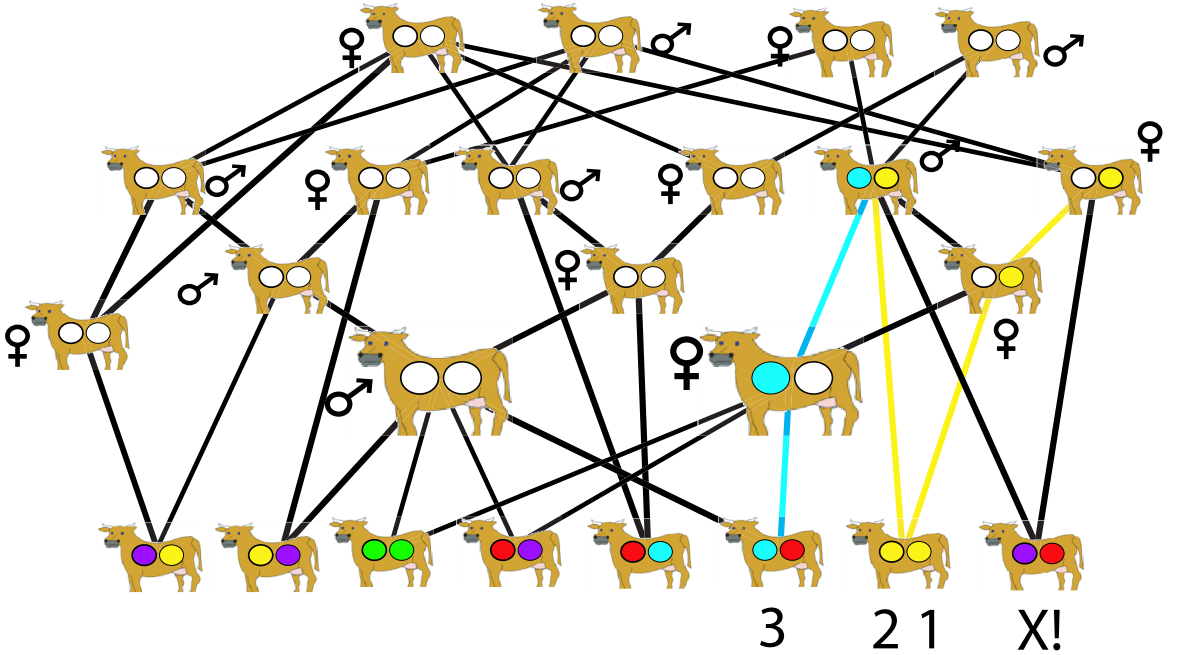
More complex questions:

- What is the probability of the observed data at two loci, given that the recombination fraction between those loci is θ?

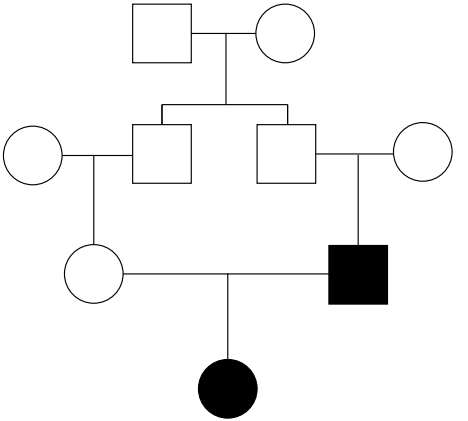
All such questions can be written as a sum over the unobserved (latent) genotypes of the ancestors, and hence they can be expressed as expectations and approximated by Monte Carlo.

Simulating ancestral genotypes naively will fail:

You can't just "flip-a-coin" your way up the pedigree the way that Wright and McPhee (1925) could to estimate inbreeding:

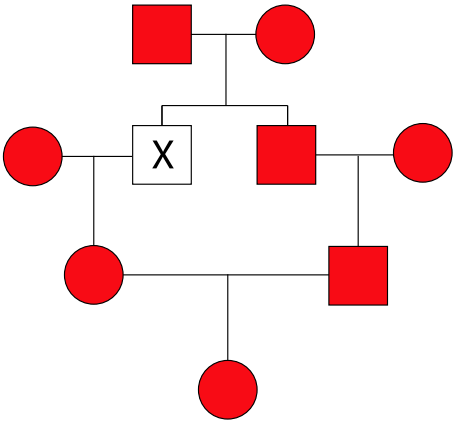


Another graph-like notation—pedigrees:



Males are squares. Females are circles. Black nodes mean they have observed data (*i.e.*,they are part of $\boldsymbol{Y}$).

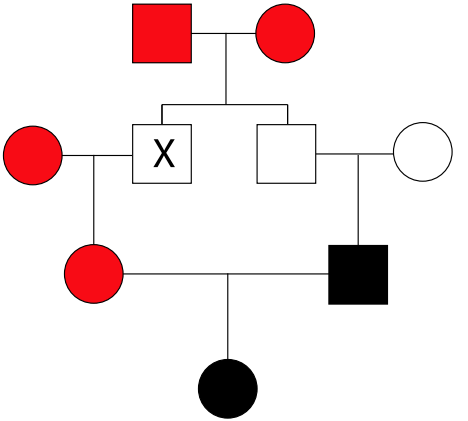
The full conditional distribution for an individual's genotype at a locus. . . :



(Red nodes represent the elements that are conditioned on to determine the full conditional for the node labeled "X".)

. . . depends only on its neighbors. . .

These are the people in your neighborhood:



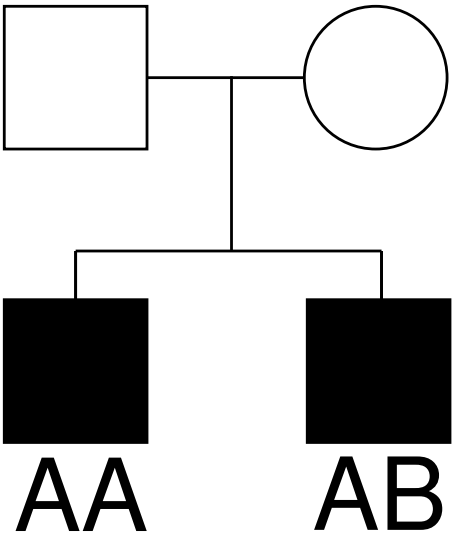
Parents, children, and spouses, (the red ones) are neighbors of node "X".

So assuming you can find a valid starting configuration, you can buzz around and do Gibbs updates to each individual's genotypes, and thus simulate the latent ancestor genotypes ($\boldsymbol{X}$) conditional on the observed data $\boldsymbol{Y}$.

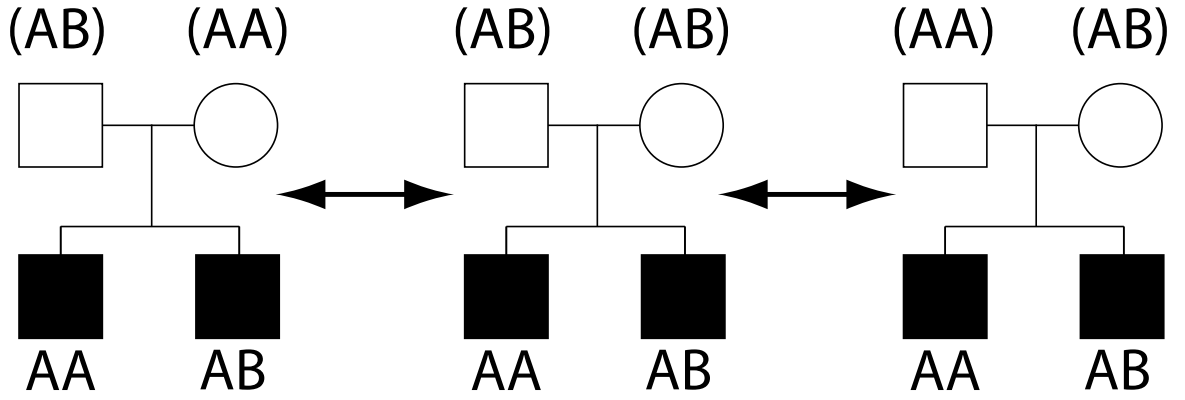
Irreducibility of single-site Gibbs sampler with two alleles:

Demonstrating this principle on an even simpler pedigree,

If the data look like:

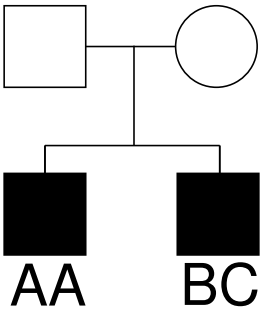


Then there are only three possible configurations of genotypes in the unshaded individuals, and these states are reachable from one another in a finite number of steps of single-site Gibbs updating. Hence this sampler is irreducible here.

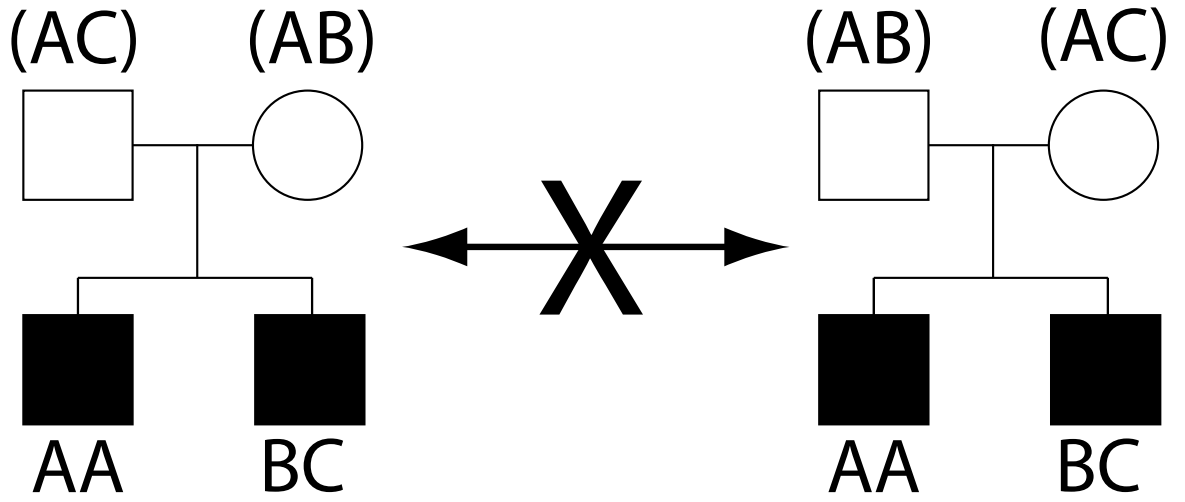


With multiple alleles, however, the single-site Gibbs sampler is reducible:

For example, with data like. . .



. . . There are only two possible states for the genotypes of the unshaded individuals and they do not communicate:



So, the single-site Gibbs sampler is not irreducible for most pedigree problems of consequence. It leads to a reducible Markov chain.

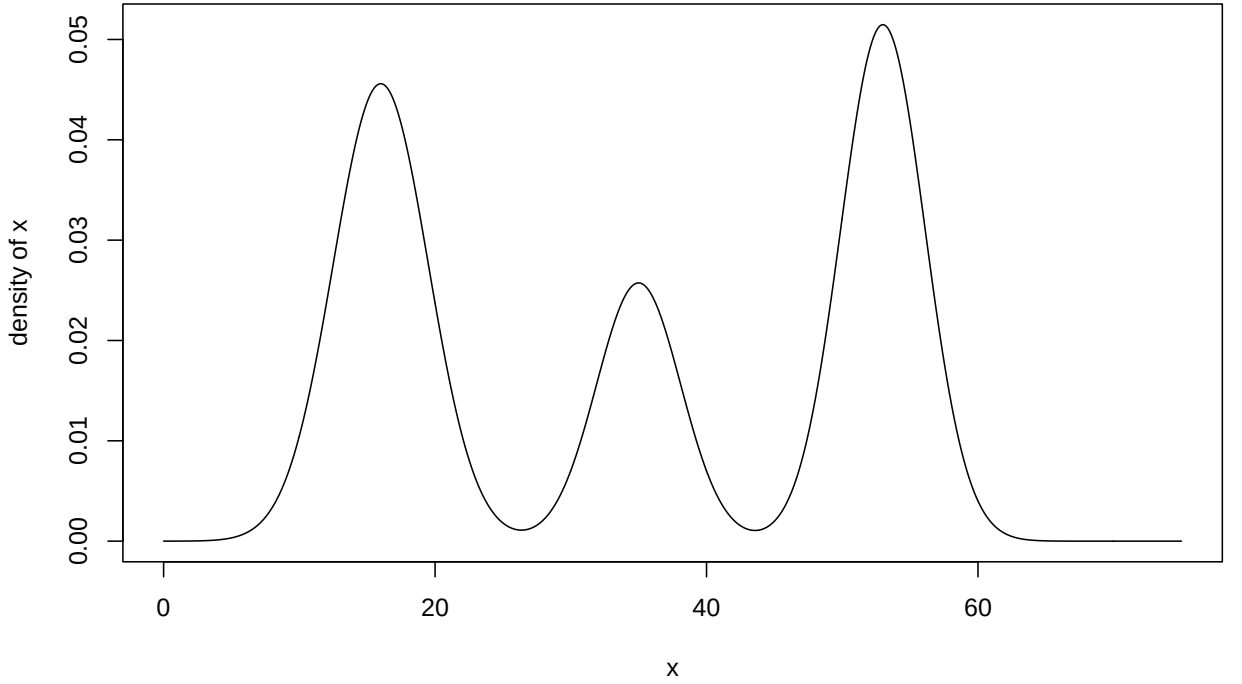
Irreducibility ain't the only thing:

While irreducibility is essential for MCMC, it is often possible to have an irreducible sampler, but have *poor mixing*.

i.e., you may be able to show theoretically that your sampler is irreducible, but *practically* it may not reach all areas of high probability.

Poor mixing means that the chain does not adequately sample all regions of the space.

Consider the following one-dimensional example—a mixture of three normal densities:



Computer Demo

Designing samplers to mix well:

This is often the hardest challenge in MCMC, especially in high dimensional space with bumpy or multimodal likelihood surfaces.

Obvious improvements could come from:

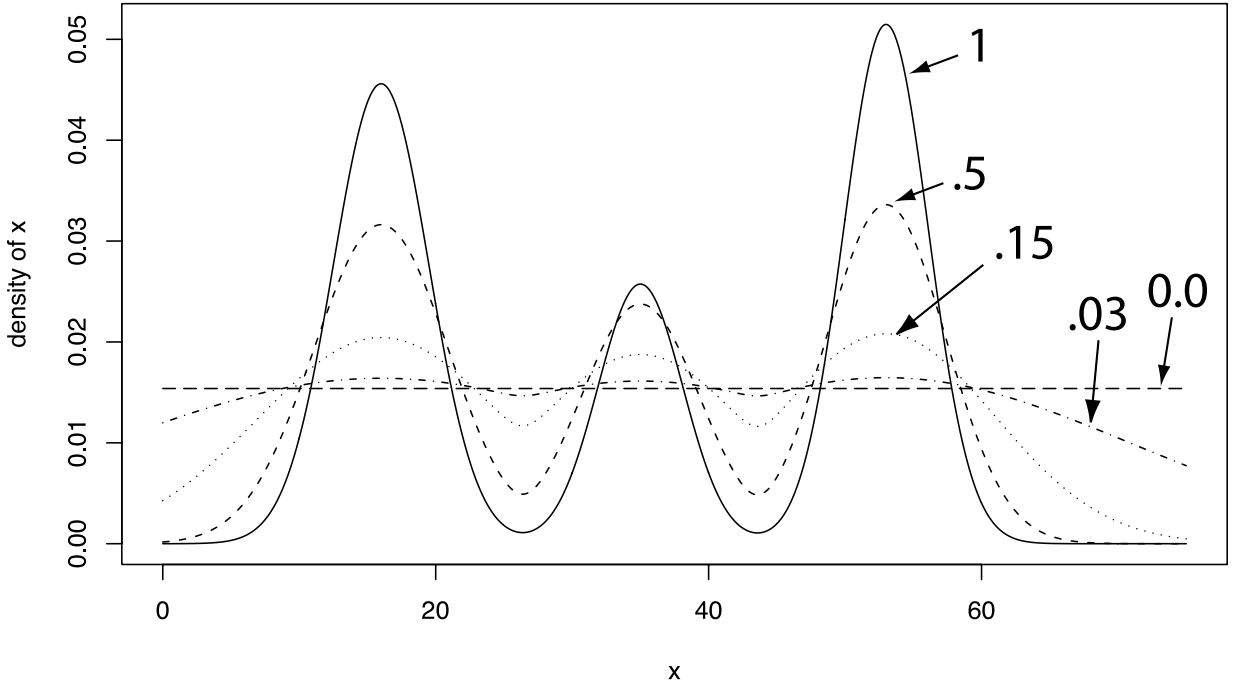
- Tweaking the proposal distribution (but note that if the variance of the proposal distibution is very high, the acceptance rate may $\downarrow$)
- Updating several variables jointly. In Gibbs sampling, in particular, this can be useful. (However, the computations necessary may require considerable extra time).

One general method for improving mixing is now called *Metropolis-coupled* MCMC (Geyer 1991).

This is used extensively in the Bayesian phylogenetic analysis package MR BAYES.

Metropolis-coupled MCMC:

MCMCMC takes advantage of the fact that by raising a probability density to a power β between zero and one, the relative heights of peaks, and the relative depths of valleys, may be made less extreme:



Recipe for Metropolis-coupled MCMC:

- Let X denote all the different variables being sampled in the MCMC
- Run one chain, labelled 1, as you normally would with limiting distribution $f(X)$ being the distribution you wish to sample from. Note: $\beta_1 = 1$.
- Run $n-1$ other chains (chains $2, \ldots, n$) in the same way, except run them so that the limiting distributions are "flattened out" versions of f. *i.e.*, the i^{th} chain ($i = 2, \ldots, n$) uses a Hastings ratio that looks like:

$$\frac{q(X|X')}{q(X'|X)} \frac{[f(X')]^{\beta_i}}{[f(X)]^{\beta_i}}$$

  where each β_i is between 0 and 1. These chains are often called the "hot" chains, because they move around more easily—much like a heated molecule.
- In addition to the normal MCMC updates, also perform "chain-swapping" updates:

1. Choose a pair of chains i and j ($1 \leq i < j \leq n$) at random and propose swapping their current states ($X_{[i]}$ and $X_{[j]}$, respectively).
2. Accept or reject the proposed swap using the Hastings ratio:

$$\alpha = \min \left\{ 1, \frac{[f(X_{[i]})]^{\beta_j}}{[f(X_{[i]})]^{\beta_i}} \frac{[f(X_{[j]})]^{\beta_i}}{[f(X_{[j]})]^{\beta_j}} \right\}$$

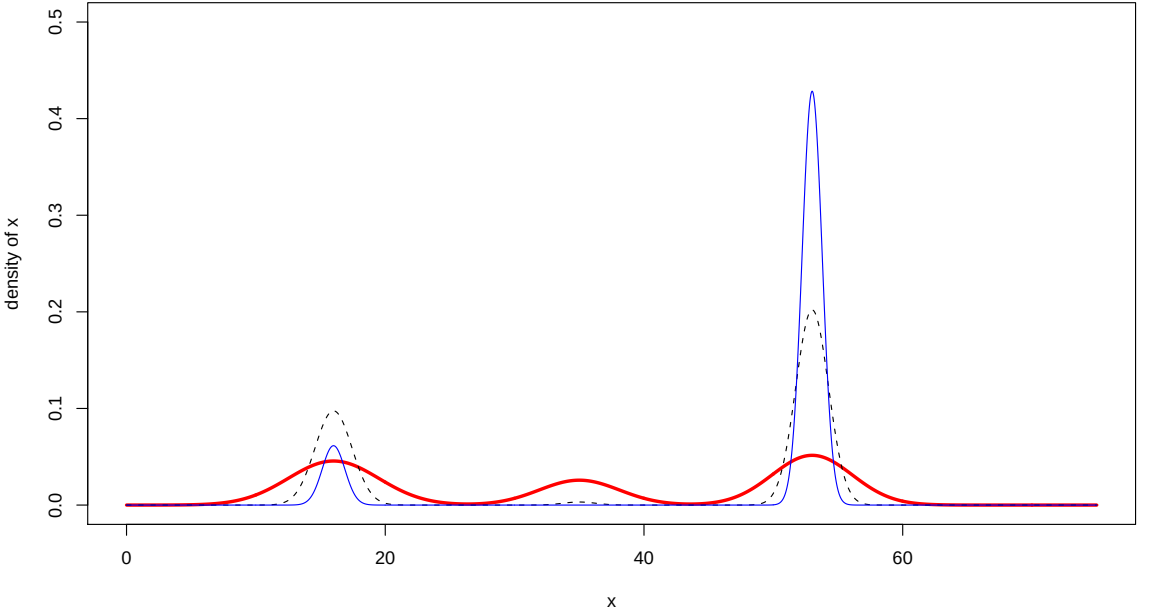
- The Monte Carlo sample itself is *only drawn from chain* 1!
- Derivation of the Hastings ratio is easy: given i and j and the current states $X_{[i]}$ and $X_{[j]}$, the proposal distribution is symmetrical, so those terms cancel out. The bits that remain are just the ratio of limiting distributions of the proposed *vs.* the original states.
- Computationally, you don't have to swap the states (which could mean copying a lot of memory around). Instead you just swap β's.

Computer Demo: `mc3demo`

Simulated Annealing:

• Simulated annealing is an optimization technique that relies on "tricks" like those employed in Metropolis-coupled MCMC

• Consider what happens if $\beta > 1$. (Thick red line is $\beta = 1$. Dotted black line is $\beta = 6$. Thin solid blue line is $\beta = 16$.)



Recipe for simple simulated annealing:

Imagine $f(x)$ is an objective function you wish to maximize. That is, you want to find the value of x that maximizes $f(x)$. To do so by simulated annealing you...

- Imagine that $f(x)$ is an unnormalized probability density or mass function
- Design an M-H sampler to sample x's from target distribution $f(x)$
- Run a single chain using this M-H sampler while raising $f(x)$ in the M-H ratio to the power of β:

$$\frac{q(X|X')}{q(X'|X)} \frac{[f(X')]^\beta}{[f(X)]^\beta}$$

- Start with β small (less than 1, typically), then progressively make β larger.

- When β is quite large and the values of x are changing little each step, or you've run out of computer time, call the current value of x the optimum.
- This is not guaranteed to be a global maximum.
- In this simple case, there are not multiple chains being run.
- In simulated annealing, you are not approximating an expectation, but you are using the machinery of MCMC to tackle an optimization problem.