Markov Chain Monte Carlo

Allan Zhang

Roadmap

- Examining the motivations and background of Markov Chain Monte Carlo (MCMC)
  - Monte Carlo – random sampling of an unknown distribution
  - Markov Chain – Stationary distributions and convergence
- Why previous methods are insufficient
  - Inverse CDF
  - Rejection Sampling
- Key insights/Formal derivation of widely used MCMC techniques
  - Metropolis and Metropolis-Hastings
  - Gibbs Sampling
- Surface-level exploration of advanced MCMC techniques
  - Simulated Annealing
  - Hamiltonian MCMC and the NUTS sampler
  - Interacting Particle

I’m assuming you know...

- Linear algebra
- Probability theory (distributions)

Bonus if you are familiar with:

- Markov Chain/Stochastic Processes
- Sampling techniques

MCMC – A random sampling technique

- Markov Chain Monte Carlo is a family of algorithms used to draw samples from a probability distribution

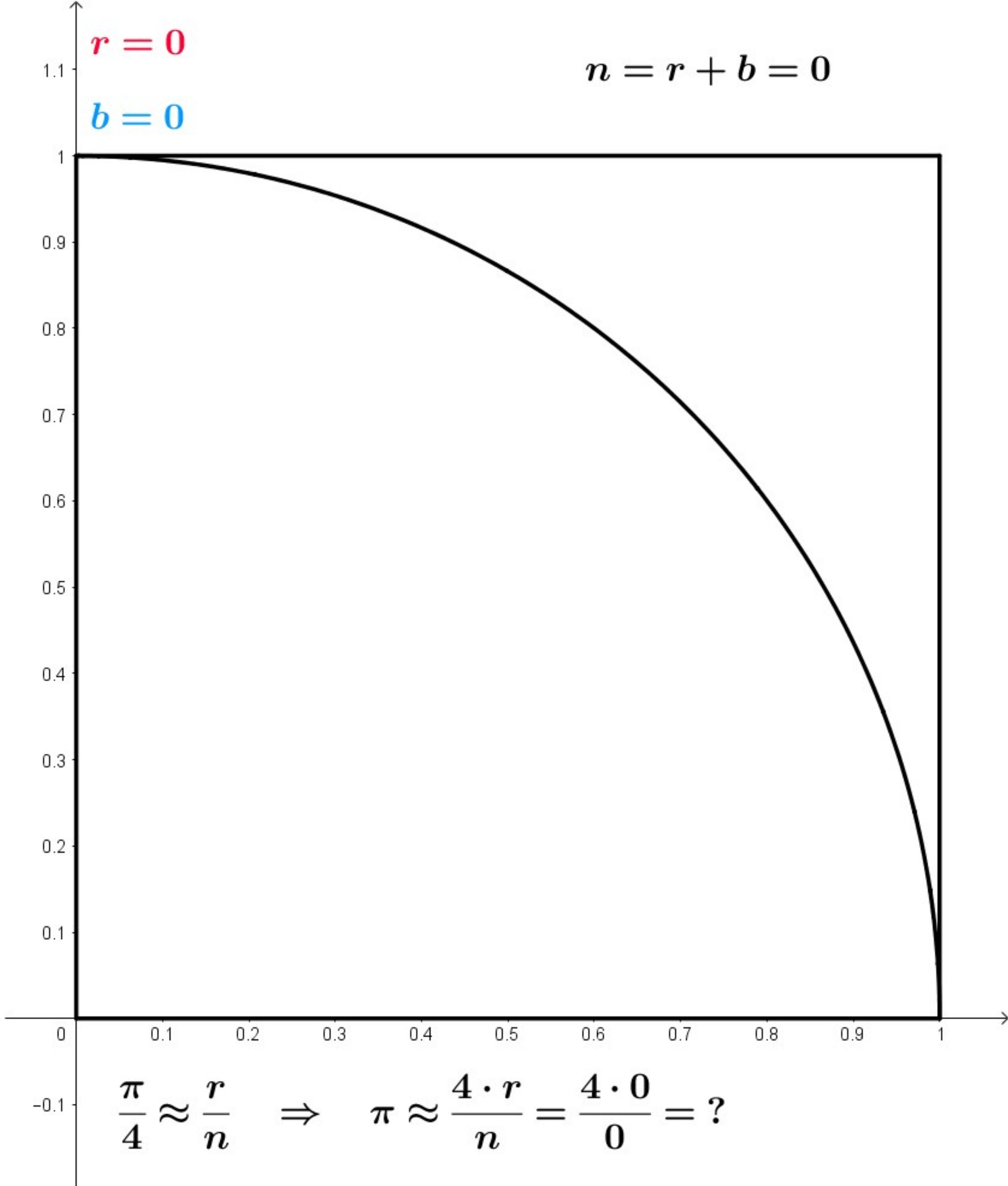
- Best applied to distributions are are highly complex or high-dimensional

Monte Carlo – “approximate” solutions can be better than exact ones

- Monte Carlo is the method of exploring a process/distribution/system via random sampling (or simulating the random process)
  - The philosophy is that an analytical/exact solution may be hard to compute, but by randomly sampling from a distribution over the domain and performing a computation on the outputs, the aggregate of the outputs approaches the exact solution
- In other words, given global parameter θ and sample parameter θ', $\theta \rightarrow \theta'$ as sample size increases.
- Seems completely obvious to us now, but Monte Carlo methods have always been closely tied to computational processing power: we take fast simulation for granted!

Monte Carlo Examples

- Ex 1: Approximating π



$r = 0$
$b = 0$
$n = r + b = 0$
1.1
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
-0.1
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
$\frac{\pi}{4} \approx \frac{r}{n} \quad \Rightarrow \quad \pi \approx \frac{4 \cdot r}{n} = \frac{4 \cdot 0}{0} = ?$

Wikipedia

Monte Carlo Examples

- Ex 2: A wacky distribution – see notebook!

Markov Chains – Only “now” matters

- Markov Chains are stochastic processes describing a sequence of events
- Given sequence $(a_n)=\{a_1,a_2,a_3,\ldots\}$

  Markov property:

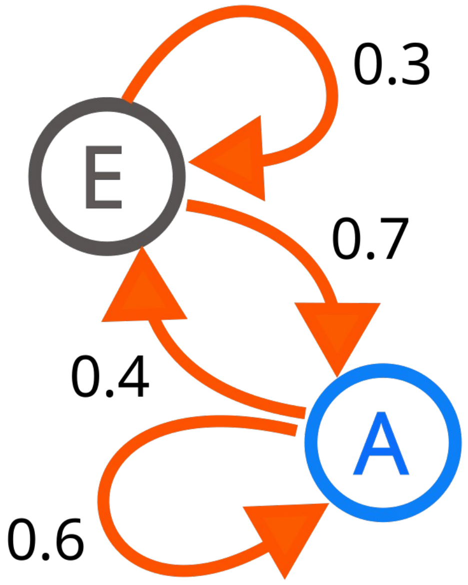
  $P(a_n=x_n|a_{n-1}=x_{n-1})=P(a_n=x_n|a_{n-1}=x_{n-1},a_{n-2}=x_{n-2},a_{n-3}=x_{n-3},\ldots)$
- In other words, Markov Chains are memoryless

Markov Chains

- Homogeneous Markov Chains can often be described by a transition matrix:
  - Let be the s initial state of the chain
  - Let $D(t,s)$ be the state of the chain given initial state s at time $t \geq 0$

$$D(t,s) = D(t-1,s)P = \ldots = D(0,s)P^t$$

$$P = \begin{bmatrix} 0.6 & 0.4 \\ 0.7 & 0.3 \end{bmatrix}$$



Wikipedia

Walking along a Markov Chain

- Observe that given a transition matrix, one can determine the exact probability of some “walker” being in some state after t periods.
- See notebook demo!
- Now what if we extended some single “walker” into a more general notion?
- Observe that we can replace the notion of a “walker” instead with an initial distribution, and observe how the “density” of each node with respect to the total population changes over time.

Markov Chain Key insight

- You might have noticed something interesting: no matter what initial distribution we started with, after a while, the distribution will converge.
- This is a key property that arises if we construct Makov chains in a certain way, which we will now formally define.

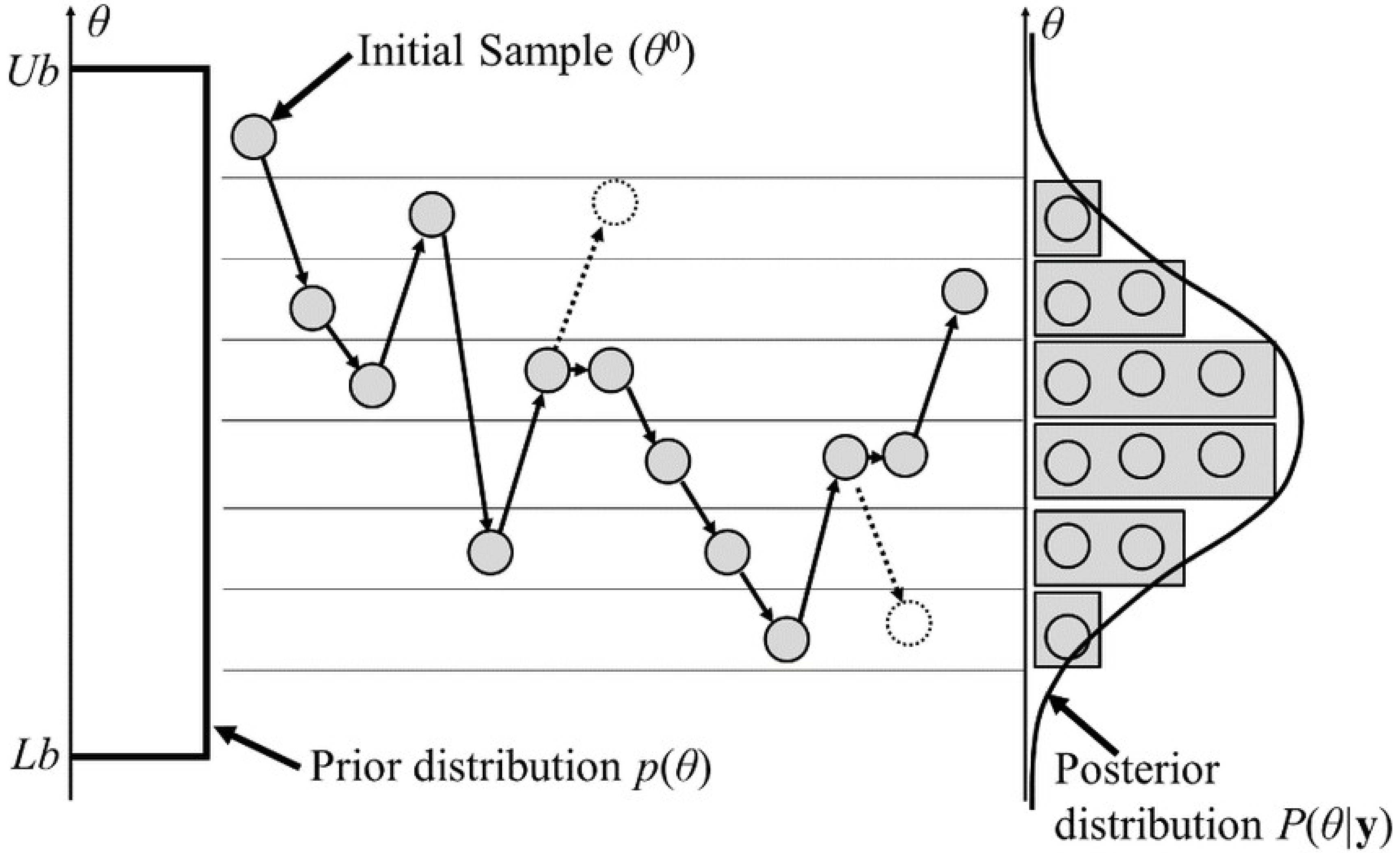
Fundamental Theorem of Markov Chains

- Consider a Markov chain that is irreducible, positive-recurrent, and aperiodic:
  - Positive-recurrent and irreducible: for all pairs of states/nodes s_i,s_j it is possible to eventually reach s_j if one starts in s_i
  - Aperiodic: the chain has no directed-cycle
- Then for all states s, $\lim_{t \to \infty} D(t, s) = \pi$

  Where $\pi = P\pi$ denotes the **stationary distribution**, and is independent of time and initial state.

Why we love stationary distributions

- Stationary distributions are essentially the reason why MCMC algorithms work
- Consider again the “single walker”: if we have a Markov chain that has already converged to stationary distribution π, then “walking along” the chain is equivalent to drawing random samples from π
- This is also an application of Bayesian statistics


θ
Ub
Lb
Initial Sample (θ^0)
Prior distribution $p(\theta)$
θ
Posterior distribution $P(\theta|\mathbf{y})$

Wikipedia

Why the fuss about MCMC?

- We already have many existing methods of generating random samples from distributions
- Inverse CDF: works only when the distribution has a closed form and is invertible
- Rejection sampling: suffers the curse of dimensionality
- See notebook demo!
- Finally, the simplified nature of MCMC walkers allows for many chains to be run in parallel
  - Especially good with modern processing

MCMC Visualizer

- https://chi-feng.github.io/mcmc-demo/

Metropolis-Hastings

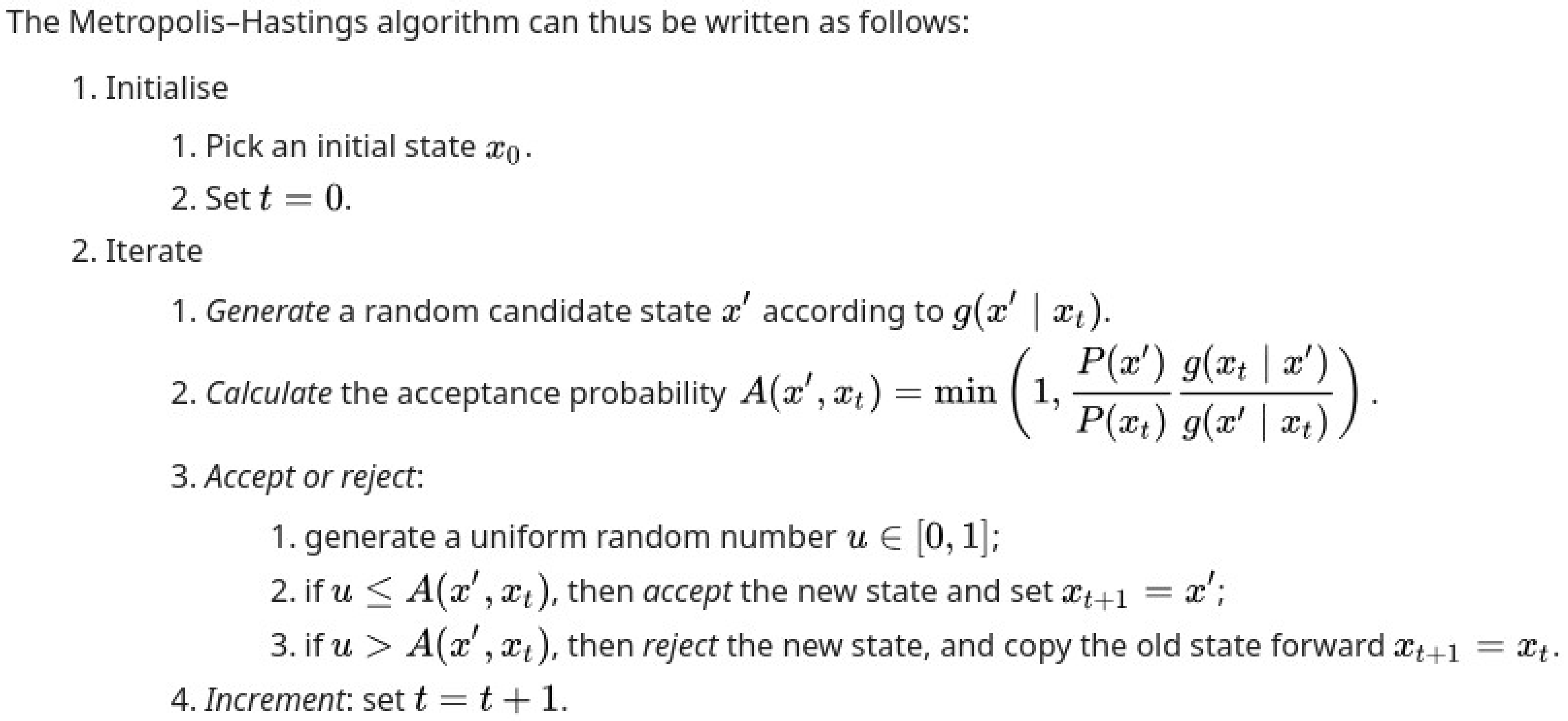
- Goal: create a Markov-Chain that admits a stationary distribution
- Issue: our parameter space is continuous, not discrete, and a transition matrix is not well-defined.
- Solution: Define a way for a random walker to “move” around the parameter space such that positive-reccurence, irreducibility, and aperiodicity properties hold.

Detailed Balance is the key!

- Positive-reccurence, irreducibility, and aperiodicity holds if for any node/state j in our state space S:
- Global balance, or π-invariance
  - The proportion of “particles” leaving j is equal to the proportion of “particles entering j
  - $\pi(j) = \Sigma_{k \in S} \pi(k) P(k \to j)$
  - Designing a proposal method satisfying the above equation is usually difficult or intractable
- Detailed balance
  - For any node/state k, the proportion of “particles” going from j to k is equal to the proportion of particles going from k to j
  - $\pi(j) P(j \to k) = \pi(k) P(k \to j)$
  - This is a stronger condition than global balance, and it is easier to design a proposal algorithm satisfying detailed balance

Metropolis Hastings Algorithm

Let P be our desired distribution that we wish to sample from. In fact P can be proportional to our desired distribution.



The Metropolis–Hastings algorithm can thus be written as follows:

1. Initialise
   1. Pick an initial state x_0.
   2. Set $t = 0$.
2. Iterate
   1. *Generate* a random candidate state x' according to $g(x' \mid x_t)$.
   2. *Calculate* the acceptance probability $A(x', x_t) = \min\left(1, \frac{P(x')}{P(x_t)} \frac{g(x_t \mid x')}{g(x' \mid x_t)}\right)$.
   3. *Accept or reject*:
      1. generate a uniform random number $u \in [0, 1]$;
      2. if $u \leq A(x', x_t)$, then *accept* the new state and set $x_{t+1} = x'$;
      3. if $u > A(x', x_t)$, then *reject* the new state, and copy the old state forward $x_{t+1} = x_t$.
   4. *Increment*: set $t = t + 1$.

Remark: The distribution of g heavily affects acceptance probability, and thus convergence to P. Best results are obtained when g is a similar “shape” to P.

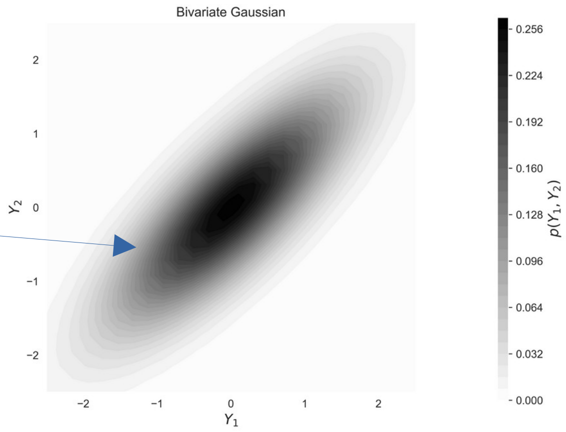
Metropolis-Hastings: Pros and Cons

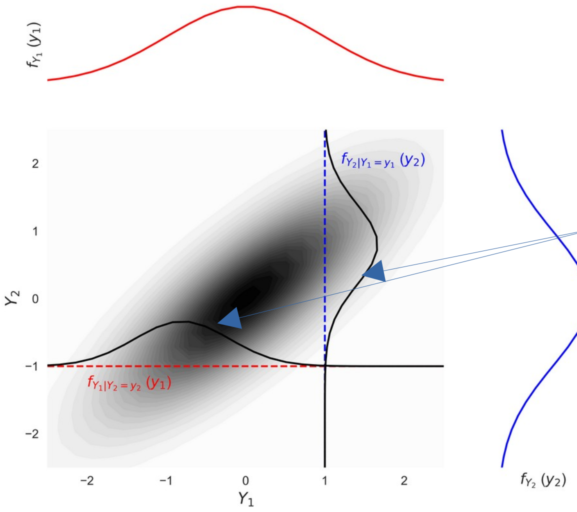
- Pros
  - Autocorrelation between sampled points ensures areas of high density are explored (“sampled”) more thoroughly
  - Relative acceptance rule means stationary distribution need not be a “real” distribution – just proportional to one. No need for annoying scalar constant
- Cons
  - Bad choice of proposal method leads to poor convergence
  - Sensitive to initialization values – If starting far from a high density area, it can be hard to reach it!
  - Locality trap – Once in a high density region, it can be hard to escape it

Gibbs sampler

- A special case of Metropolis-Hastings in which you ALWAYS accept the proposed point
- Key insight: sampling from a multivariate joint distribution is hard. What if we instead sample multiple uni-variate conditional distributions instead?

Sampling this is hard!





Sampling these is a lot easier!

https://geostatisticslessons.com/lessons/multigaussian

The Gibbs Sampler Algorithm

Suppose we want to obtain k samples of a n-dimensional random vector $\mathbf{X} = (X_1, \ldots, X_n)$

1. Begin with initial value $X^0 = (x_1, ..., x_n)$
2. Given sample $X^t = (x_1^t, ..., x_n^t)$, we obtain X^{t+1} by sampling each component x_i^{t+1} conditioned on all other components:

$$P(x_j | x_1^{t+1}, ..., x_{j-1}^{t+1}, x_{j+1}^t, ..., x_n^t)$$

3. Once all components are sampled, we have new sample $X^{t+1} = (x_1^{t+1}, ..., x_n^{t+1})$

Gibbs Sampler – Pros and Cons

- Pros
  - Doesn’t require an acceptance step
  - If conditional distributions are simple, calculation can be very fast
  - Less vulnerable to locality traps than M-H
- Cons
  - Conditional distributions must be known and tractable
  - High-dimensional data can significantly increase workload

Advanced MCMC Techniques

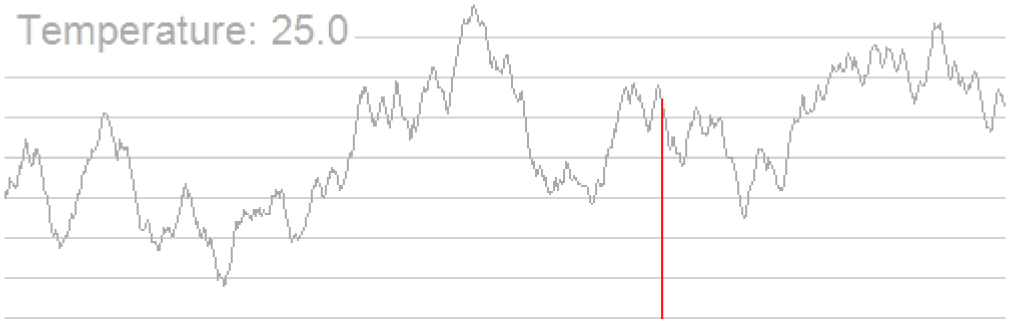
- We can see that the MCMC technique is very flexible: as long as the Markov Chain is properly designed, many things about it can be streamlined/tweaked for efficiency
- Usually the best efficiency improves are made by making a “smarter” proposal algorithm
  - Spending less time in low density areas
  - Properly exploring all local optima
- This usually leads to higher acceptance rate for new proposed points
  - Shorter time to convergence
  - Less rejected (wasted) points

Hamiltonian MCMC – roll a ball, and it’ll stay near wells

- Imposes a Hamiltonian physics constraint on the walker
- At each step, push the “ball” in a random direction with random momentum
- Higher density areas have more “gravity”, so proposal points stay near these regions
- Automatic differentiation makes gradient computation realistically fast
- No U-Turn Sampler (NUTS) – Halts the rolling ball if it backtracks/orbits
- Still struggles with multimodal/ridge densities (AKA topographies!)
- See visualizer!

Simulated Annealing – Decrease the jump length over time

- Originally not an MCMC algorithm but an optimization one
- Key insight: too long of a proposal jump and we can't stay in high density areas. Too short of a proposal jump and we can't properly explore the entire space
- Make the algorithm accept "worse" points with high probability in the beginning, but lower that over time

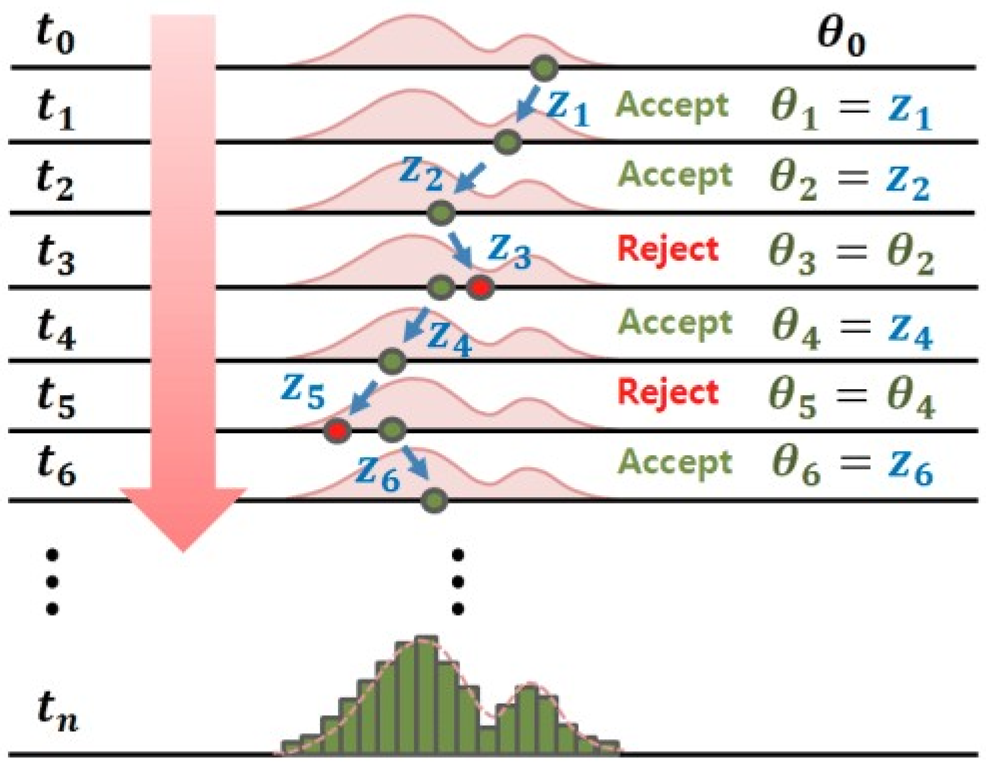


Wikipedia

And much much more...

- Slice sampling
  - “Slice” the distribution curve horizontally, then uniformly choose a point within that
- Multiple try MC
  - Allow multiple proposal tries at each jump
- Reversible jump
  - Proposals can change the dimensionality of the space
- Nested sampling
  - Subdivide the space into smaller and smaller regions and based on likelihood and sample from each region
  - Kind of like a Bayesian interpretation of Lebesgue integration

Thank you!



Markov Chain Monte Carlo:

- A random sampling technique
- More resilient to high dimensionality
- Can be parallelized
- Lots of potential for further exploration/optimization
- Widespread application

DOI:10.1080/15732479.2019.1628077