# CPSC 440: Advanced Machine Learning

## Markov Chain Monte Carlo

Mark Schmidt

University of British Columbia

Winter 2022

## Last Time: "Stupid MCMC"

- Consider finding the expected value of a fair di:
  - For a 6-sided di, the expected value is 3.5.
- Consider the following "stupid MCMC" algorithm:
  - Start with some initial value, like "4".
  - At each step, roll the di and generate a random number $u$:
    - If $u < 0.5$, "accept" the roll and take the roll as the next sample.
    - Othewise, "reject" the roll and take the old value ("4") as the next sample.
- Stationary distribution of is $\pi(c) = 1/6$, so

$$\pi(x) = p(x),$$

which is the key feature underlying MCMC methods.
  - If you run it a really long time then stop, it will look like a sample from $p$.

## Markov Chain Monte Carlo (MCMC)

- Markoc chain Monte Carlo (MCMC):
  - Design a Markov chain that has $\pi(x) = p(x)$.
    - For large enough $k$, a sample $x^k$ from the chain will be distributed according to $p(x)$.
  - Use the Markov chain samples within a Monte Carlo estimator,

$$\mathbb{E}[g(x)] \approx \frac{1}{n}\sum_{t=1}^{n} g(x^i).$$

- Law of large numbers can be generalized to show this converges as $n \to \infty$.
  - "Ergodic theroem".
  - But convergence is slower since we're generating dependent samples.
- A popular way to design the Markov chain is Metropolis-Hastings algorithm.
  - Oldest algorithm among the "10 Best Algorithms of the 20th Century".

## Special Case of Metropolis Algorithm

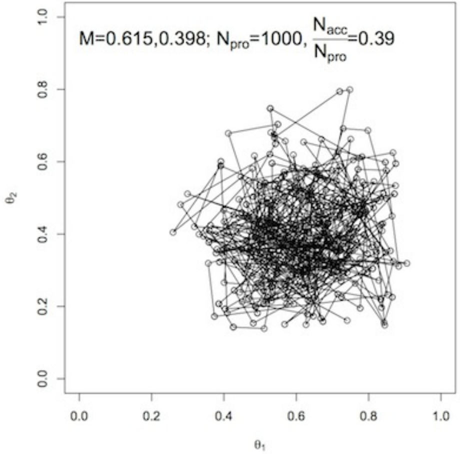
- The Metropolis algorithm for sampling from a continuous target $p(x)$:
  - Assumes we can evaluate $p$ up to a normalizing constant, $p(x) = \tilde{p}(x)/Z$.
  - Start with some initial value $x^0$.
  - On each iteration add zero-mean Gaussian noise to $x^t$ to give proposal $\hat{x}^t$.
    - And generate a $u$ uniformly between $0$ and $1$.
  - "Accept" the proposal and set $x^{t+1} = \hat{x}^t$ if

$$u \leq \frac{\tilde{p}(\hat{x}^t)}{\tilde{p}(x^t)}, \qquad \frac{\text{(probability of proposed)}}{\text{(probability of current)}}$$

  - Otherwise "reject" the sample and use $x^t$ again as the next sample $x^{t+1}$.
    - Proposals that increase probability are always accepted.
    - Proposals that decrease probability might be accepted or rejected.

  - A random walk, but sometimes rejecting steps that decrease probability:
    - A valid MCMC algorithm on continuous densities, but convergence may be slow.
    - You can implement this even if you don't know normalizing constant.

## Metropolis Algorithm in Action



Pseudo-code:

```
eps = randn(d,1)
xhat = x + eps
u = rand()
if u < ( p(xhat) / p(x) )
 set x = xhat
otherwise
  keep x
```

## Metropolis Algorithm Analysis

- Markov chain with transitions $q_{ss'} = q(x^t = s' \mid x^{t-1} = s)$ is reversible if
$$\pi(s)q_{ss'} = \pi(s')q_{s's},$$
for some distribution $\pi$ (this condition is called detailed balance).

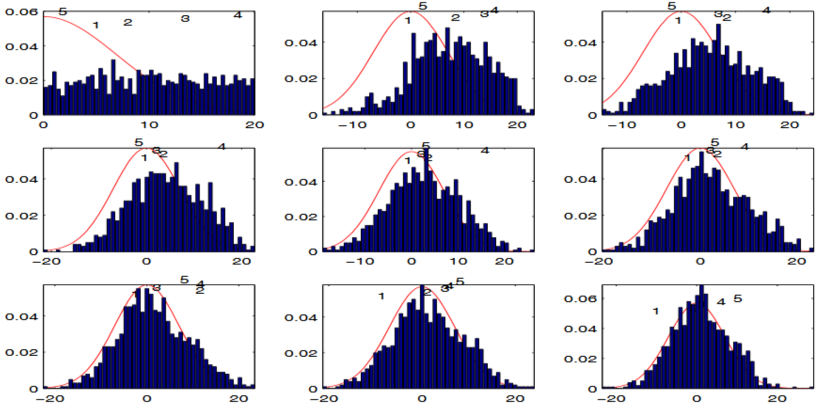
- Reversibility implies $\pi$ is a stationary distribution,
$$\sum_s \pi(s)q_{ss'} = \sum_s \pi(s')q_{s's} \qquad \text{(sum reversibility over } s \text{ values)}$$
$$\sum_s \pi(s)q_{ss'} = \pi(s')\underbrace{\sum_s q_{s's}}_{=1}$$
$$\sum_s \pi(s)q_{ss'} = \pi(s') \qquad \text{(stationary condition).}$$

- Metropolis is reversible with $\pi = p$ (bonus slide) so $p$ is stationary distribution.
  - And positive transition probabilities mean $\pi$ exsists, and is unique/reached.

## Markov Chain Monte Carlo

MCMC sampling from a Gaussian:

From top left to bottom right: histograms of 1000 independent Markov chains with a normal distribution as target distribution.



http://www.cs.ubc.ca/~arnaud/stat535/slides10.pdf

## MCMC Implementation Issues

- In practice, we often don't take all samples in our Monte Carlo estimate:
  - Burn in: throw away the initial samples when we haven't converged to stationary.
  - Thinning: only keep every $k$ samples, since they will be highly correlated.

- Two common ways that MCMC is applied:
  1. Sample from a huge number of Markov chains for a long time, use final states.
     - Great for parallelization.
     - No need for thinning, since you throw all but last samples.
     - Need to worry about burn in.
  2. Sample from one Markov chain for a really long time, use states across time.
     - Less worry about burn in.
     - Need to worry about thinning.

- It can very hard to diagnose if we have reached stationary distribution.
  - It is P-space hard (*not* polynomial-time even if P=NP).
  - Various heuristics exist.

## Metropolis-Hastings

- Metropolis algorithm is a special case of Metropolis-Hastings.
  - Uses a proposal distribution $q(\hat{x} \mid x)$, giving probability of proposing $\hat{x}$ at $x$.
    - In Metropolis, $q$ is a Gaussian with mean $x$.

- Metropolis-Hastings accepts a proposed $\hat{x}^t$ if

$$u \leq \frac{\tilde{p}(\hat{x}^t)q(x^t \mid \hat{x}^t)}{\tilde{p}(x^t)q(\hat{x}^t \mid x^t)},$$

  where extra terms ensures reversibility for asymmetric $q$:
  - E.g., if you are more likely to propose to go from $x^t$ to $\hat{x}^t$ than the reverse.

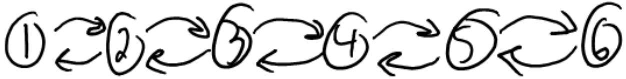
- This works under very weak conditions, such as $q(\hat{x}^t \mid x^t) > 0$.
  - But you can make performance much better/worse with an appropriate $q$.

## Metropolis-Hastings Example: Rolling Dice with Coins

- Suppose we want to sample from a fair 6-sided di.
  - p(x=1) = p(x=2) = p(x=3) = p(x=4) = p(x=5) = p(x=6) = 1/6.
  - But don't have a di or a computer and can only flip coins.

- Consider the following random walk on the numbers 1-6:
  - If $x = 1$, always propose $2$.
  - If $x = 2$, 50% of the time propose 1 and 50% of the time propose 3.
  - If $x = 3$, 50% of the time propose 2 and 50% of the time propose 4.
  - If $x = 4$, 50% of the time propose 3 and 50% of the time propose 5.
  - If $x = 5$, 50% of the time propose 4 and 50% of the time propose 6.
  - If $x = 6$, always propose $5$.

- "Flip a coin: go up if it's heads and go down it it's tails".
  - The PageRank "random surfer" applied to this graph:



## Metropolis-Hastings Example: Rolling Dice with Coins

- "Roll a di with a coin" by using random walk as transitions $q$ in Metropolis-Hastings to:
  - $q(\hat{x} = 2 \mid x = 1) = 1$, $q(\hat{x} = 1 \mid x = 2) = \frac{1}{2}$, $q(\hat{x} = 2 \mid x = 3) = 1/2, \ldots$
  - If $x$ is in the "middle" (2-5), we'll always accept the random walk.
    - If $x = 3$ and we propose $\hat{x} = 2$, then:

$$u < \frac{p(\hat{x} = 2)}{p(x = 3)} \frac{q(x = 3 \mid \hat{x} = 2)}{q(\hat{x} = 2 \mid x = 3)} = \frac{1/6}{1/6} \frac{1/2}{1/2} = 1.$$

    - If $x = 2$ and we propose $\hat{x} = 1$, then we test $u < 2$ which is also always true.
    - If $x$ is at the end ($1$ or $6$), you accept with probability $1/2$:

$$u < \frac{p(\hat{x} = 2)}{p(x = 1)} \frac{q(x = 1 \mid \hat{x} = 2)}{q(\hat{x} = 2 \mid x = 1)} = \frac{1/6}{1/6} \frac{1/2}{1} = \frac{1}{2}.$$

## Metropolis-Hastings Example: Rolling Dice with Coins

- So Metropolis-Hastings modifies random walk probabilities:
  - If you're at the end (1 or 6), stay there half the time.
  - This accounts for the fact that 1 and 6 have only one neighbour.
    - Which means they aren't visited as often by the random walk.
- Could also be viewed as a random surfer in a different graph:

$$\begin{tikzcd} 1 \arrow[loop left] \arrow[r, bend left] & 2 \arrow[l, bend left] \arrow[r, bend left] & 3 \arrow[l, bend left] \arrow[r, bend left] & 4 \arrow[l, bend left] \arrow[r, bend left] & 5 \arrow[l, bend left] \arrow[r, bend left] & 6 \arrow[l, bend left] \arrow[loop right] \end{tikzcd}$$

- You can think of Metropolis-Hastings as the modification that "makes the random walk have the right probabilities".
  - For any (reasonable) proposal distribution $q$.

## Special Case of Gibbs Sampling

- An important special case of Metropolis-Hastings is Gibbs sampling.
  - Method to sample from a multi-dimensional distribution.
  - Probably the most common multi-dimensional sampler.

- Gibbs sampling starts with some $x$ and then repeats:
  1. Choose a variable $j$ uniformly at random.
  2. Update $x_j$ by sampling it from its conditional,

$$x_j \sim p(x_j \mid x_{-j}),$$

  where $x_{-j}$ means "all variables except $x_j$".

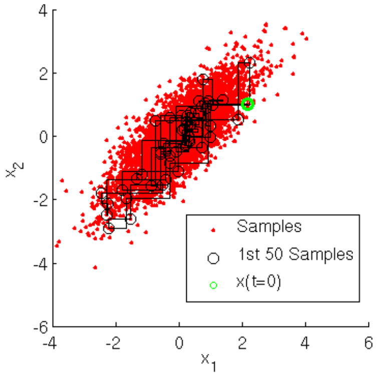
- A common variation is to cycle through the variables in order.

## Gibbs Sampling in Action

- Start with some initial value: $x^0 = \begin{bmatrix} 2 & 2 & 3 & 1 \end{bmatrix}$.
- Select random $j$ like $j = 3$.
- Sample variable $j$: $x^1 = \begin{bmatrix} 2 & 2 & 1 & 1 \end{bmatrix}$.
- Select random $j$ like $j = 1$.
- Sample variable $j$: $x^2 = \begin{bmatrix} 3 & 2 & 1 & 1 \end{bmatrix}$.
- Select random $j$ like $j = 2$.
- Sample variable $j$: $x^3 = \begin{bmatrix} 3 & 2 & 1 & 1 \end{bmatrix}$.
- $\ldots$
- Use the samples to form a Monte Carlo estimator.

## Gibbs Sampling in Action: Multivariate Gaussian

- Gibbs sampling works for general distributions.
  - E.g., sampling from multivariate Gaussian by univariate Gaussian sampling.



`https://theclevermachine.wordpress.com/2012/11/05/mcmc-the-gibbs-sampler`

- Video: `https://www.youtube.com/watch?v=AEwY6QXWoUg`

## Sampling from Conditionals

- For discrete $x_j$ the conditionals needed for Gibbs sampling have a simple form,

$$p(x_j = c \mid x_{-j}) = \frac{p(x_j = c, x_{-j})}{p(x_{-j})} = \frac{p(x_j = c, x_{-j})}{\sum_{x_j = c'} p(x_j = c', x_{-j})} = \frac{\tilde{p}(x_j = c, x_{-j})}{\sum_{x_j = c'} \tilde{p}(x_j = c', x_{-j})}$$

  where we use unnormalized $\tilde{p}$ since $Z$ is the same in numerator/denominator.
  - Note that this expression is easy to evaluate: just summing over values of $x_j$.

- For continuous $x_j$ replace the sum by an integral.
  - May be able to figure out quantile function for inverse transform sampling.
  - May need to use rejection sampling, especially in non-conjugate cases.

## Gibbs Sampling as a Markov Chain

- The "Gibbs sampling Markov chain" if $p$ is over 4 binary variables:
  - The states are the possible configurations of the four variables:
    - $s = [0\ 0\ 0\ 0], s = [0\ 0\ 0\ 1], s = [0\ 0\ 1\ 0]$, etc.
  - The initial probability $q$ is set to $1$ for the initial state, and 0 for the others:
    - If you start at $s = [1\ 1\ 0\ 1]$, then $q(x^1 = [1\ 1\ 0\ 1]) = 1$ and $q(x^1 = [0\ 0\ 0\ 0]) = 0$.
  - The transition probabilities $q$ are based on variable we choose and target $p$:
    - If we are at $s = [1\ 1\ 0\ 1]$ and choose coordinate randomly we have:

$$q(x^{t+1} = [0\ 0\ 1\ 1] \mid x^t = [1\ 1\ 0\ 1]) = 0 \quad \text{(Gibbs only updates on variable)}$$

$$q(x^{t+1} = [1\ 0\ 0\ 1] \mid x^t = [1\ 1\ 0\ 1]) = \underbrace{\frac{1}{d}}_{\text{uniform}} \underbrace{p(x_2 = 0 \mid x_1 = 1, x_3 = 0, x_4 = 1)}_{\text{from target distribution } p}.$$

  - Not homogeneous if cycling, but homogeneous if add "last variable" to state.

- Can show Gibbs sampling is a special case of Metropolis-Hastings.
  - In this case the acceptance rate is $1$ so we never reject.

## Metropolis-Hastings

- Common choices for proposal distribution $q$ in Metropolis-Hastings:
  - Metropolis originally used random walks: $x^t = x^{t-1} + \epsilon$ for $\epsilon \sim \mathcal{N}(0, \Sigma)$.
  - Hastings originally used independent proposal: $q(x^t \mid x^{t-1}) = q(x^t)$.
    - Usually not a good choice in high dimensions.
  - Gibbs sampling updates single variable based on conditional.
  - Block Gibbs sampling:
    - If you can sample multiple variables at once Gibbs sampling tends to work better.
  - Collapsed Gibbs sampling (Rao-Blackwellization):
    - MCMC provably works better at sampling marginals of a joint distribution.
    - "Try to integrate over variables you do not care about."
  - Bonus slides survey some other advanced MCMC methods.

- Unlike rejection sampling, high acceptance rate is not always good:
  - High acceptance rate may mean we're not moving very much.
  - Low acceptance rate definitely means we're not moving very much.
  - Designing good proposals $q$ is an "art".

## Outline

## Higher-Order Markov Models

- Markov models use a density of the form

$$p(x) = p(x_1)p(x_2 \mid x_1)p(x_3 \mid x_2)p(x_4 \mid x_3) \cdots p(x_d \mid x_{d-1}).$$

- They support efficient computation but Markov assumption is strong.
- A more flexible model would be a second-order Markov model,

$$p(x) = p(x_1)p(x_2 \mid x_1)p(x_3 \mid x_2, x_1)p(x_4 \mid x_3, x_2) \cdots p(x_d \mid x_{d-1}, x_{d-2}),$$

or even a higher-order models.

- General case is called directed acyclic graphical (DAG) models:
    - They allow dependence on any subset of previous features.

## DAG Models

- As in Markov chains, DAG models use the chain rule to write

$$p(x_1, x_2, \dots, x_d) = p(x_1)p(x_2 \mid x_1)p(x_3 \mid x_1, x_2) \cdots p(x_d \mid x_1, x_2, \dots, x_{d-1}).$$

- We can alternately write this as:

$$p(x_1, x_2, \dots, x_d) = \prod_{j=1}^{d} p(x_j \mid x_{1:j-1}).$$

- In Markov chains, we assumed $x_j$ only depends on previous $x_{j-1}$ given past.
- In DAGs, $x_j$ can depend on any subset of the past $x_1, x_2, \dots, x_{j-1}$.

## DAG Models

- We often write joint probability in DAG models as

$$p(x_1, x_2, \ldots, x_d) = \prod_{j=1}^{d} p(x_j \mid x_{\text{pa}(j)}),$$

where $\text{pa}(j)$ are the "parents" of feature $j$.
  - For Markov chains the only "parent" of $j$ is $(j-1)$.
  - If we have $k$ parents we only need $2^{k+1}$ parameters (for binary states).

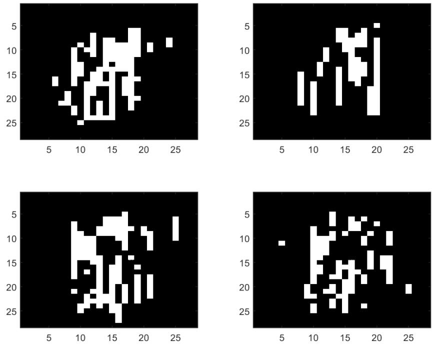
- This corresponds to a set of conditional independence assumptions,

$$p(x_j \mid x_{1:j-1}) = p(x_j \mid x_{\text{pa}(j)}),$$

that we're independent of previous non-parents given the parents.

## MNIST DIgits with Markov Chains

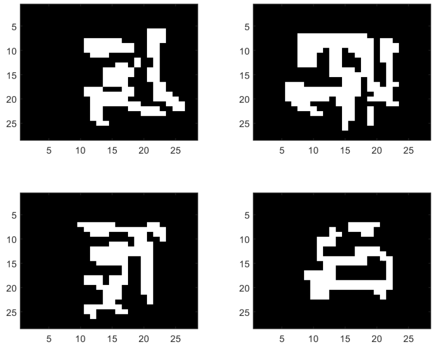
- Recall trying to model digits using an inhomogeneous Markov chain:



Only models dependence on pixel above, not on 2 pixels above nor across columns.

## MNIST Digits with DAG Model (Sparse Parents)

- Samples from a DAG model with 8 parents per feature:



Parents of $(i, j)$ are 8 other pixels in the neighbourhood ("up by 2, left by 2"):

$$\{(i{-}2, j{-}2), (i{-}1, j{-}2), (i, j{-}2), (i{-}2, j{-}1), (i{-}1, j{-}1), (i, j{-}1), (i{-}2, j), (i{-}1, j)\}.$$

## Summary

- Markov chain Monte Carlo (MCMC) approximates complicated expectations.
  - Generate samples from a Markov chain that has $p$ as stationary distribution.
  - Use these samples within a Monte Carlo approximation.
- Metropolis-Hastings: MCMC method allowing arbitrary "proposals".
  - By accepting/rejecting samples based on proposal and target probabilities.
- Gibbs sampling: Samples each variable conditioned on all others.
  - Special case of Metropolis-Hastings MCMC method.
- DAG models factorize joint distribution into product of conditionals.
  - Usually we assume conditionals depend on small number of "parents".

- Next time: conditional independence in DAGs.
  (I am not going to pretend this is exciting, but its is useful.)

## Metropolis Algorithm Analysis

- Metropolis algorithm has $q_{ss'} > 0$ (sufficient to guarantee stationary distribution is unique and we reach it) and satisfies detailed balance with target distribution $p$,
$$p(s)q_{ss'} = p(s')q_{s's}.$$
- We can show this by defining transition probabilities
$$q_{ss'} = \min\left\{1, \frac{\tilde{p}(s')}{\tilde{p}(s)}\right\},$$
and observing that
$$\begin{aligned} p(s)q_{ss'} &= p(s)\min\left\{1, \frac{\tilde{p}(s')}{\tilde{p}(s)}\right\} = p(s)\min\left\{1, \frac{\frac{1}{Z}\tilde{p}(s')}{\frac{1}{Z}\tilde{p}(s)}\right\} \\ &= p(s)\min\left\{1, \frac{p(s')}{p(s)}\right\} = \min\left\{p(s), p(s')\right\} \\ &= p(s')\min\left\{1, \frac{p(s)}{p(s')}\right\} = p(s')q_{s's}. \end{aligned}$$

## Advanced Monte Carlo Methods

- "Adaptive MCMC": tries to update $q$ as we go: needs to be done carefully.
- "Particle MCMC": use particle filter to make proposal.

- Auxiliary-variable sampling: introduce variables to sample bigger blocks:
  - E.g., introduce $z$ variables in mixture models.
  - Also used in Bayesian logistic regression (beginning with Albert and Chib).

- Trans-dimensional MCMC:
  - Needed when dimensionality of problem can change on different iterations.
  - Most important application is probably Bayesian feature selection.

- Hamiltonian Monte Carlo:
  - Faster-converging method based on Hamiltonian dynamics.

- Population MCMC:
  - Run multiple MCMC methods, each having different "move" size.
  - Large moves do exploration and small moves refine good estimates.