Markov Chain Monte Carlo

CPSC 440/550: Advanced Machine Learning

cs.ubc.ca/~dsuth/440/23w2

University of British Columbia, on unceded Musqueam land

2023-24 Winter Term 2 (Jan–Apr 2024)

Markov Chains for Monte Carlo Estimation

- We've been discussing inference in Markov chains
  - Sampling, marginals, stationary distribution, decoding, conditionals
  - Usually (for discrete chains) there are dynamic programming algorithms
- We can also use Markov chains for inference in other models
  - Most common way to do this is Markov chain Monte Carlo (MCMC)
  - Widely used for approximate inference, e.g. in Bayesian logistic regression
- High-level idea of MCMC:
  - We want to use Monte Carlo estimates with a distribution p
    - But we don't know how to generate iid samples from p
  - Design a homogeneous Markov chain whose stationary distribution is p
    - This is usually surprisingly easy to do
  - Use ancestral sampling to sample from a long version of this Markov chain
  - Use the Markov chain samples within the Monte Carlo approximation

Degenerate Example: "Pointless MCMC"

- Consider finding the expected value of a fair die:
  - For a 6-sided die, the expected value is 3.5
- Consider the following "pointless MCMC" algorithm:
  - Start with some initial value, like "4"
  - At each step, roll the die and generate a random number u:
    - If $u < 0.5$, "accept" the roll and take the roll as the next sample
    - Otherwise, "reject" the roll and take the old value (e.g. "4") as the next sample
- Generates samples from a Markov chain with this transition probability:

$$q(x_{t-1} \to x_t) = \frac{1}{2}\,\mathbb{1}(x_t = x_{t-1}) + \frac{1}{2} \cdot \frac{1}{6} = \begin{cases} 7/12 & x_t = x_{t-1} \\ 1/12 & x_t \neq x_{t-1} \end{cases}$$

  - $q(s \to s')$ is a "proposal" distribution over s'

Degenerate Example: "Pointless MCMC"

- Pointless MCMC in action:
  - Start with "4", so record "4"
  - Roll a "6" and generate 0.234, so record "6"
  - Roll a "3" and generate 0.612, so record "6"
  - Roll a "2" and generate 0.523, so record "6"
  - Roll a "3" and generate 0.125, so record "3"
  - Roll a "2" and generate 0.433, so record "2"
- So our samples are 4,6,6,6,3,2...

- If you run this long enough, you will spend 1/6 of the time on each number
- Stationary distribution is uniform: if we start at a uniform number, either staying there or going to a uniformly random number is still uniform
- So the stationary distribution of the chain is p (the uniform distribution)
  - This is the key feature of MCMC methods

- It is "pointless" since it assumes we can generate IID samples from p
  - If you can do that, don't use this algorithm for approximate samples!

Markov Chain Monte Carlo (MCMC)

- Markov 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 $p(x)$
    - Changing notation a bit: $x^{(1)}$ is the first sampled state, $x^{(2)}$ the second, $\ldots$, $x^{(n)}$ last
  - Use the Markov chain samples within a Monte Carlo estimator,

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

- Generalization of the law of large numbers ("ergodic theorem") shows: as $n \to \infty$, $\frac{1}{n}\sum_{t=1}^{n} g(x^{(t)}) \to \mathbb{E}[g(x)]$ (almost surely)
  - But convergence is slower since we're generating dependent samples
  - e.g. the variance is higher than $\mathrm{Var}[g(x)]/n$, since samples aren't iid

- A popular way to design the Markov chain is Metropolis-Hastings algorithm.
  - Oldest algorithm out of the "10 Best Algorithms of the 20th Century"

Special Case: Metropolis Algorithm

- The Metropolis algorithm for sampling from a continuous target $p(x)$:
- Assumes we can evaluate p up to a normalizing constant, $p(x) = \tilde{p}(x)/Z$

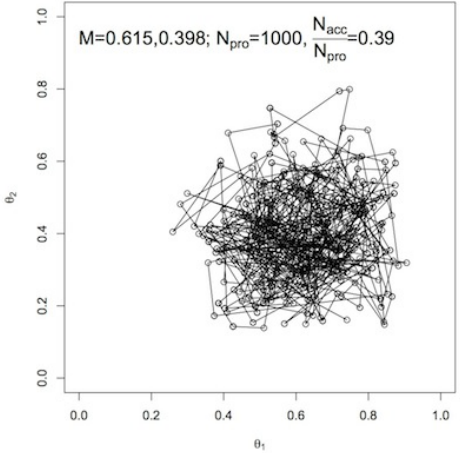
- Start with some initial value $x^{(0)}$
- Until we get bored:
  - Add zero-mean Gaussian noise to $x^{(t-1)}$ to give proposal $\hat{x}^{(t)}$
  - Generate a u uniformly between 0 and 1
  - "Accept" the proposal and set $x^{(t)} = \hat{x}^{(t)}$ if

$$u \leq \frac{\tilde{p}(\hat{x}^{(t)})}{\tilde{p}(x^{(t-1)})} \qquad \frac{\text{(probability of proposed)}}{\text{(probability of current)}}$$

  - Otherwise "reject" the sample and use $x^{(t-1)}$ again as the next sample $x^{(t)}$

- Proposals that increase probability density are always accepted
- Proposals that decrease probability density might be accepted or rejected
- Always converges for continuous densities, but might be really slow
- You can implement this even if you don't know normalizing constant

Metropolis Algorithm in Action




```
while True:
  xhat = x + \
    rs.multivariate_normal(cov=Sigma)
  u = rs.random()
  if u < p(xhat) / p(x):
      x = xhat
  yield x
```

http://www.columbia.edu/~cjd11/charles_dimaggio/DIRE/styled-4/

Metropolis Algorithm Analysis

- Markov chain with transitions $q(s \to s')$ is reversible if

$$\pi(s)\, q(s \to s') = \pi(s')\, q(s' \to s)$$

for some distribution π; this condition is called detailed balance

- Reversibility implies π is a stationary distribution:

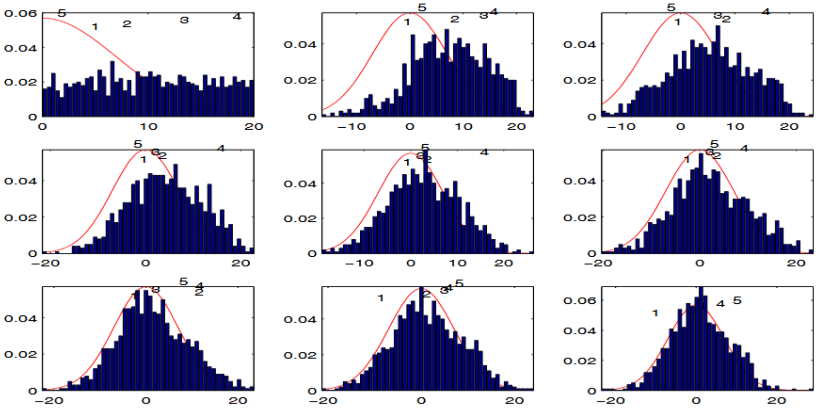
$$\begin{aligned}
\pi^{+}(s) = \sum_{s'} \pi(s')\, q(s' \to s) &= \sum_{s'} \pi(s)\, q(s \to s') && \text{(detailed balance for each term)} \\
&= \pi(s) \underbrace{\sum_{s'} q(s \to s')}_{1} \\
&= \pi(s) && \text{exactly the stationarity condition}
\end{aligned}$$

- Metropolis is reversible, with p its stationary distribution (bonus slide)
  - And positive transition probabilities mean π exists, 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 use *all* the samples in our Monte Carlo estimate
- Burn-in: throw away early samples, when we're far from the stationary dist
- Thinning: only keep every k samples, since they'll 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
     - Like an extreme form of thinning: only use one sample per chain
     - Need to worry about burn-in for each chain
  2. Sample from one Markov chain for a really long time, use states across time
     - Less worry about burn in
     - May need to thin, since samples will be correlated

- It can very hard to diagnose if we have reached stationary distribution
  - Formally, it's PSPACE-hard – even harder than NP-hard
  - Various heuristics exist

Outline

1. Metropolis algorithm
2. Metropolis-Hastings and Gibbs

Metropolis-Hastings

- Metropolis algorithm is a special case of Metropolis-Hastings
  - General version uses a general proposal distribution $q(\hat{x}^{(t+1)} \mid x^{(t)}) = q(x^{(t)} \to \hat{x}^{(t+1)})$
  - In Metropolis, q is a Gaussian with mean $x^{(t)}$

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

$$u \leq \frac{\tilde{p}(\hat{x}^t)}{\tilde{p}(x^{t-1})} \cdot \frac{q(\hat{x}^t \to x^{t-1})}{q(x^{t-1} \to \hat{x}^t)}$$

- These extra terms ensures reversibility (detailed balance) for asymmetric q
  - If you're more likely to propose $x^{(t-1)} \to \hat{x}^{(t)}$ than the other way, less likely to accept

- Eventually converges under very weak conditions, e.g. all $q(x^{(t)} \to \hat{x}^{(t+1)}) > 0$
  - But practical convergence can change a lot with different q

Metropolis-Hastings Example: Rolling Dice with Coins

- Say we want to sample from a fair 6-sided die
  - $\Pr(X = c) = \frac{1}{6}$ for each $c \in \{1, \ldots, 6\}$
  - But we don't have a die, or a computer, just coins
    - and don't want to do rejection sampling...

- 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 you can), go down it it's tails (and you can)
  - A random walk on this graph:

Metropolis-Hastings Example: Rolling Dice with Coins

- "Roll a die with a coin" by using random walk as transitions q in M-H:
  - $q(1 \to 2) = 1$, $q(2 \to 1) = \frac{1}{2}$, $q(2 \to 3) = \frac{1}{2}$, $\dots$, $q(6 \to 5) = 1$

- If $x = 3$ and we propose $\hat{x} = 2$, then we always accept: check is

$$u < \frac{p(2)}{p(3)} \cdot \frac{q(2 \to 3)}{q(3 \to 2)} = \frac{1/6}{1/6} \cdot \frac{1/2}{1/2} = 1$$

- Same for any x in the "middle" (2 to 5)
- If $x = 2$ and we propose $\hat{x} = 1$, we also always accept: check is

$$u < \frac{p(1)}{p(2)} \cdot \frac{q(1 \to 2)}{q(2 \to 1)} = \frac{1/6}{1/6} \cdot \frac{1}{1/2} = 2$$

- If x is at the end (1 or 6), you accept with probability $1/2$:

$$u < \frac{p(2)}{p(1)} \cdot \frac{q(2 \to 1)}{q(1 \to 2)} = \frac{1/6}{1/6} \cdot \frac{1/2}{1} = \frac{1}{2}$$

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:

- 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: Gibbs Sampling

- An important special case of Metropolis-Hastings is Gibbs sampling
  - Method to sample from a multi-dimensional distribution
  - Maybe 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 resampling it from its conditional distribution given everything else:

$$x_j^{(t)} \sim p\left(x_j \mid x_1^{(t-1)}, \ldots, x_{j-1}^{(t-1)}, x_{j+1}^{(t-1)}, \ldots, x_d^{(t-1)}\right)$$

     Keep other variables the same

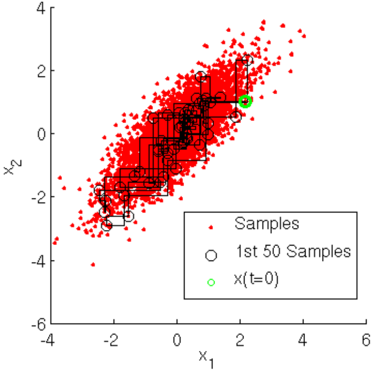
- Common variation: resample x_1, then x_2, ..., then x_d, then x_1, then x_2, ...

Gibbs Sampling in Action

- Start with some initial value: $x^{(0)} = \begin{bmatrix} 2 & 2 & 3 & 1 \end{bmatrix}$
- Select random index: $j = 3$
- Sample variable j: $x^{(1)} = \begin{bmatrix} 2 & 2 & 1 & 1 \end{bmatrix}$
- Select random index: $j = 1$
- Sample variable j: $x^{(2)} = \begin{bmatrix} 3 & 2 & 1 & 1 \end{bmatrix}$
- Select random index: $j = 2$
- Sample variable j: $x^{(3)} = \begin{bmatrix} 3 & 2 & 1 & 1 \end{bmatrix}$
- ...
- 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
- Using $x_{\neg j}$ to mean "everything but x_j":

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

  using unnormalized $\tilde{p}$ since Z is the same in numerator/denominator
  - Last expression is easy to evaluate: just sum over all values of x_j

- For continuous x_j, replace the sum by an integral
  - Might have an easy form (e.g. conditionally Gaussian)
  - Might be able to figure out the (inverse) cdf, for inverse transform sampling
  - Might 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:
    - $[0\ 0\ 0\ 0], [0\ 0\ 0\ 1], [0\ 0\ 1\ 0]$, etc (there are $2^4 = 16$ of them)
  - The initial probability $\pi^{(0)}$ is a "point mass" for the initial state:
    - If you start at $[1\ 1\ 0\ 1]$, then $\pi^{(0)}([1\ 1\ 0\ 1]) = 1$ and $\pi^{(0)}([0\ 0\ 0\ 0]) = 0$
  - The transition probabilities q are based on the variable we choose and target p:
    - If we are at $[1\ 1\ 0\ 1]$ and choose coordinate randomly we have:

$$q([1\ 1\ 0\ 1] \to [0\ 0\ 1\ 1]) = 0 \quad \text{(Gibbs only updates one variable)}$$

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

  - Not homogeneous if cycling, but can "hack it": add "last updated variable" to state

Gibbs is Metropolis-Hastings

- For random coordinates, proposal is $q(x \to \hat{x}) = \frac{1}{d}\sum_{j=1}^{d} \mathbb{1}(\hat{x}_{\neg j} = x_{\neg j}) p(\hat{x}_j \mid x_{\neg j})$
- For a proposal with $\hat{x}_{\neg j} = x_{\neg j}$, acceptance probability is min of 1 and

$$
\begin{aligned}
\frac{p(\hat{x})}{p(x)} \cdot \frac{q(\hat{x} \to x)}{q(x \to \hat{x})} &= \frac{p(\hat{x}_j \mid \hat{x}_{\neg j}) p(\hat{x}_{\neg j})}{p(x_j \mid x_{\neg j}) p(x_{\neg j})} \cdot \frac{\frac{1}{d}\, p(x_j \mid \hat{x}_{\neg j})}{\frac{1}{d}\, p(\hat{x}_j \mid x_{\neg j})} \\
&= \frac{p(\hat{x}_j \mid x_{\neg j}) p(x_{\neg j})}{p(x_j \mid x_{\neg j}) p(x_{\neg j})} \cdot \frac{p(x_j \mid x_{\neg j})}{p(\hat{x}_j \mid x_{\neg j})} \qquad \text{(since } x_{\neg j} = \hat{x}_{\neg j}\text{)} \\
&= 1
\end{aligned}
$$

- Detailed balance is satisfied; also need ergodicity for unique stationary dist

Metropolis-Hastings

bonus!

- Common choices for proposal distribution q in Metropolis-Hastings:
  - Metropolis et al. originally used random walks: $x^{(t)} = x^{(t-1)} + \epsilon$ for $\epsilon \sim \mathcal{N}(0, \mathbf{\Sigma})$
  - Hastings originally used independent proposal: $q(x^{(t-1)} \to x^{()}) = q(x^{(t)})$
    - Usually not a good choice in high dimensions
  - Gibbs sampling updates a 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 don't care about"

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

Advanced Monte Carlo Methods

bonus!

- "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:
  - For example, 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 (using $\nabla \log p$)

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

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
  - Burn-in period, and samples are highly correlated (sometimes thin them)
- Metropolis: add Gaussian noise, maybe "reject" if it decreases density
- Metropolis-Hastings: general MCMC method allowing arbitrary "proposals"
  - Accept/reject samples based on proposal and target probabilities
- Gibbs sampling: Samples each variable conditioned on all others
  - Special case of Metropolis-Hastings MCMC method

- Next time: a very quick tour of fancier probabilistic models

Metropolis Algorithm Analysis

bonus!

- Metropolis algorithm has $q(s \to s') > 0$ for all s, s'
  - This ensures stationary distribution is unique, and that we reach it
- Also has detailed balance with target distribution p, $p(s)q_{s \to s'} = p(s')q(s' \to s)$
- We can show this by defining the transition probabilities as

$$c_{s-s'} = \frac{\exp\left(-\frac{1}{2}(s-s')\Sigma^{-1}(s-s')\right)}{\left(2\pi\left|\Sigma\right|\right)^{d/2}} \qquad q_{s \to s'} = c_{s-s'} \min\left\{1, \frac{\tilde{p}(s')}{\tilde{p}(s)}\right\}$$

and observing that

$$\begin{aligned}
p(s)q(s \to s') &= c_{s-s'}p(s)\min\left\{1, \frac{\tilde{p}(s')}{\tilde{p}(s)}\right\} = c_{s-s'}p(s)\min\left\{1, \frac{\frac{1}{Z}\tilde{p}(s')}{\frac{1}{Z}\tilde{p}(s)}\right\} \\
&= c_{s-s'}p(s)\min\left\{1, \frac{p(s')}{p(s)}\right\} = c_{s-s'}\min\left\{p(s), p(s')\right\} \\
&= p(s')c_{s'-s}\min\left\{1, \frac{p(s)}{p(s')}\right\} = p(s')q(s' \to s)
\end{aligned}$$