CPSC 440: Advanced Machine Learning

Metropolis-Hastings

Mark Schmidt

University of British Columbia

Winter 2021

Last Time: A Simple Example of Metropolis-Hastings

- Consider a loaded di that rolls a 6 half the time (all others equally likely).
  - So $p(x = 6) = 1/2$ and $p(x = 1) = p(x = 2) = \cdots = p(x = 5) = 1/10$.

- Consider the following "less stupid" MCMC algorithm:
  - At each step, we start with an old state x.
  - Generate a random number x uniformly between 1 and 6 (roll a fair di), and generate a random number u in the interval $[0, 1]$.
  - "Accept" this roll

$$u < \frac{p(\hat{x})}{p(x)} = \frac{\tilde{p}(\hat{x})}{\tilde{p}(x)},$$

    and otherwise "reject" the roll and keep x on the next iteration.
  - So if we roll $\hat{x} = 6$, we accept it: $u < 1$ (''always move to higher probability").
  - If $x = 2$ and roll $\hat{x} = 1$, accept it: $u < 1$ ("always move to same probability").
  - If $x = 6$ and roll $\hat{x} = 1$, we accept it with probability $1/5$.
    - We prefer high probability states, but sometimes move to low probability states.

- Markov chain spends half its time in state 6, 10% in state 1, 10% in state 2.,. . .

Metropolis Algorithm

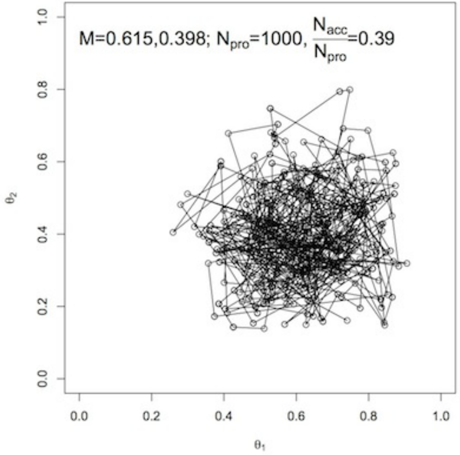
- The Metropolis algorithm for sampling from a continuous target $p(x)$:
  - On each iteration add zero-mean Gaussian noise to x^t to give proposal $\hat{x}^t$.
  - Generate u uniformly between 0 and 1.
  - "Accept" the sample and set $x^{t+1} = \hat{x}^t$ if

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

  - Otherwise "reject" the sample and use x^t again as the next sample x^{t+1}.

- 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 π (this condition is called detailed balance).

- Reversibility implies π is a stationary distribution,

$$\begin{aligned}
\sum_s \pi(s)q_{ss'} &= \sum_s \pi(s')q_{s's} && \text{(sum reversibility over } s \text{ values)}\\
\sum_s \pi(s)q_{ss'} &= \pi(s')\underbrace{\sum_s q_{s's}}_{=1}\\
\sum_s \pi(s)q_{ss'} &= \pi(s') && \text{(stationary condition).}
\end{aligned}$$

- Metropolis is reversible with $\pi = p$ (bonus slide) so p is stationary distribution.
  - Though we still need extra assumptions to ensure it's unique and we reach it.

Metropolis-Hastings

- Gibbs and Metropolis are special cases 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:
  - $q(\hat{x} = 2 \mid x = 1) = 1$, $q(\hat{x} = 1 \mid x = 2) = \frac{1}{2}$, $q(\hat{x} = 2 \mid x = 3) = 1/2, \dots$
  - 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:

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

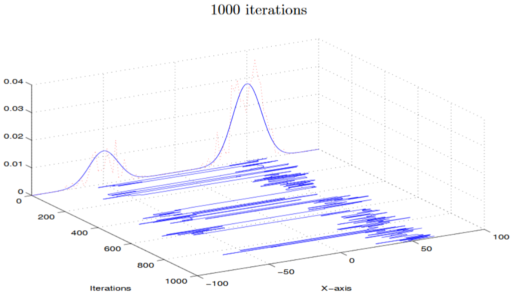
Metropolis-Hastings

- Simple choices for proposal distribution q:
  - Metropolis originally used random walks: $x^t = x^{t-1} + \epsilon$ for $\epsilon \sim \mathcal{N}(0, \Sigma)$.
  - Hastings originally used independent proposal: $q(x^t \mid x^{t-1}) = q(x^t)$.
  - Gibbs sampling updates single variable based on conditional:
    - In this case the acceptance rate is 1 so we never reject.
  - Mixture model for q: e.g., between big and small moves.
  - "Adaptive MCMC": tries to update q as we go: needs to be done carefully.
  - "Particle MCMC": use particle filter to make proposal.

- Unlike rejection sampling, we don't want acceptance rate as high as possible:
  - High acceptance rate may mean we're not moving very much.
  - Low acceptance rate definitely means we're not moving very much.
  - Designing q is an "art".

Mixture Proposal Distribution

Metropolis-Hastings for sampling from mixture of Gaussians:



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

- With a random walk q we may get stuck in one mode.
- We could have proposal be mixture between random walk and "mode jumping".
  - Bonus slides discuss some more-advanced MCMC methods.

Outline

Learning for Structured Prediction (Big Picture)

3 types of classifiers discussed in CPSC 340/440:

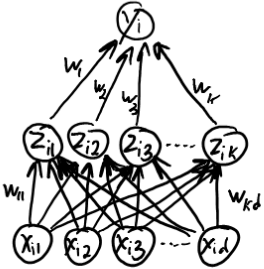
Model	"Classic ML"	Structured Prediction
Generative model $p(y, x)$	Naive Bayes, GDA	UGM (or "MRF")
Discriminative model $p(y \mid x)$	Logistic regression	CRF
Discriminant function $y = f(x)$	SVM	Structured SVM

- Discriminative models don't need to model x.
  - Don't need "naive Bayes" or Gaussian assumptions.

- Discriminant functions don't even worry about probabilities.
  - Based on decoding, which is different than inference in structured case.
  - Useful when inference is hard but decoding is easy.
  - Examples include "attractive" graphical models, matching problems, and ranking.
  - I put my material on structured SVMs here:
    - `https://www.cs.ubc.ca/~schmidtm/Courses/540-W19/L28.5.pdf`

Feedforward Neural Networks

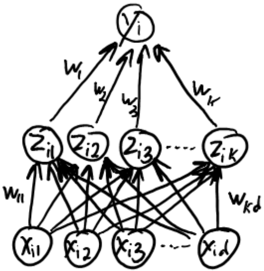
- In 340 we discussed feedforward neural networks for supervised learning.
- With 1 hidden layer the classic model has this structure:



- Motivation:
  - For some problems it's hard to find good features.
  - This learns features z that are good for particular supervised learning problem.

Neural Networks as DAG Models

- It's a DAG model but there is an important difference with our previous models:
  - In neural nets we make latent variables z_c are deterministic functions of the x_j.



- Makes inference given x trivial: if you observe all x_j you also observe all z_c.
  - In this case y is the only random variable.

Summary

- Metropolis-Hastings: MCMC method allowing arbitrary "proposals".
  - With good proposals works much better than Gibbs sampling.
- 3 types of structured prediction:
  - Generative models, discriminative models, discriminant functions.
- Neural networks learn features for supervised learning.
  - DAG model with deterministic conditionals, which makes inference easy.
- Next time: why don't neural networks just wildly overfit?

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

- Some other more-powerful MCMC methods:
  - Block Gibbs sampling improves over single-variable Gibb sampling.
  - Collapsed Gibbs sampling (Rao-Blackwellization): integrate out variables that are not of interest.
    - E.g., integrate out hidden states in Bayesian hidden Markov model.
    - E.g., integrate over different components in topic models.
    - Provably decreases variance of sampler (if you can do it, you should do it).
  - 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).

Advanced Monte Carlo Methods

- 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.
    - With mechanism to exchange samples between chains.