Markov-Chain Monte Carlo

CSE586 Computer Vision II
Penn State Univ

Recall: Problem
Sampling in High-dimensional Spaces

Standard methods fail:
• Rejection Sampling
  – Rejection rate increase with \( N \to 100\%
• Importance Sampling
  – Same problem: vast majority weights \( \to 0\)

Intuition: In high dimension problems, the "Typical Set" (volume of nonnegligible prob in state space) is a small fraction of the total space.

High-Dimensional Spaces
consider ratio of volumes of hypersphere inscribed inside hypercube

\[ V_{sphere} = \frac{\pi^{d/2}}{\Gamma(d/2 + 1)} \]
\[ V_{cube} = 2^d \]

Asymptotic behavior:
most of volume of the hypercube lies outside of hypersphere as dimension \( d \) increases

High Dimensional Spaces
Segmentation Example

• Binary Segmentation of image
  each pixel has two states: on and off

Probability of a Segmentation

• Very high-dimensional
• 256\(^2\) pixels = 65536 pixels
• Dimension of state space \( N = 65536 \) !!!!

• # binary segmentations = finite, but...
• \( 2^{65536} = 2 \times 10^{1972} \gg 10^{79} \) = atoms in universe

Representation \( P(\text{Segmentation}) \)

• Histogram? No!
• Assume pixels independent?
  \[ P(x_1, x_2, x_3, ...) = P(x_1)P(x_2)P(x_3) \]... ignores neighborhood structure of pixel lattice and empirical evidence that images are "smooth"

• Approximate solution: samples !!!
Recall: Markov Chains

Markov Chain:

- A sequence of random variables $Y_1, Y_2, Y_3, ...$
- Each variable has a distribution over states ($X_1, X_2, X_3, ...$)
- Transition probability of going to next state only depends on the current state. e.g., $P(Y_{n+1} = X_j | Y_n = X_i)$

Transition probs can be arranged in an $N \times N$ table of elements $k_{ij} = P(Y_{n+1} = X_j | Y_n = X_i)$ where the rows sum to one.

General Idea: MCMC Sampling

Start in some state, and then run the simulation for some number of time steps. After you have run it "long enough" start keeping track of the states you visit.

\[ \{ ... X_1 X_2 X_1 X_3 X_3 X_2 X_1 X_2 X_1 X_1 X_3 X_3 X_2 ... \} \]

These are samples from the distribution you want, so you can now compute any expected values with respect to that distribution empirically.

The Theory Behind MCMC Sampling

If the Markov chain is positive recurrent, there exists a stationary distribution. If it is positive recurrent and irreducible, there exists a unique stationary distribution. Then, the average of a function over samples of the Markov chain is equal to the average with respect to the stationary distribution.

We can compute this empirically as we generate samples. This is what we want to compute, and is infeasible to compute in any other way.

A simple Markov chain

\[
K = \begin{bmatrix}
0.1 & 0.6 & 0.3 \\
0.1 & 0.5 & 0.3 \\
0.6 & 0.2 & 0.3 \\
0.3 & 0.3 & 0.1 \\
\end{bmatrix}
\]

$K^T$ is transpose of transition prob table ($k_{ij}$; enb sum to one. We do this for computational convenience; (next slide)
Question:

Assume you start in some state, and then run the simulation for a large number of time steps. What percentage of time do you spend at X1, X2 and X3?

Eigen-analysis

\[ K = \begin{bmatrix} 0.1 & 0.5 & 0.6 \\ 0.6 & 0.2 & 0.3 \\ 0.3 & 0.3 & 0.1 \end{bmatrix} \]

\[ KE = ED \]

in matlab: \( [E, D] = \text{eigs}(K) \)

Eigenvalue \( \lambda_1 \) always 1

Stationary distribution \( \pi = e_i / \text{sum}(e_i) \)

i.e. \( K \pi = \pi \)

File: CSE586_PennPSU_UvA_stanley_100812.pdf

The Web as a Markov Chain

The PageRank of a webpage as used by Google is defined by a Markov chain. It is the probability to be at page \( i \) in the stationary distribution on the following Markov chain on all (known) webpages. If \( N \) is the number of known webpages, and a page \( i \) has \( k_i \) links then it has transition probability \((1-q)/k_i + q/N\) for all pages that are linked to and \( q/N \) for all pages that are not linked to. The parameter \( q \) is taken to be about 0.15.

Another Question:

Assume you want to spend a particular percentage of time at X1, X2 and X3. What should the transition probabilities be?

Thought Experiment

Consider only two states. What transition probabilities should we use so that we spend roughly equal time in each of the two states? (i.e. 50% of the time we are in state 1 and 50% of the time we are in state 2)

Detailed Balance

- In practice, you just propose some transition probabilities.
- They typically will NOT satisfy detailed balance (unless you are extremely lucky).
- Instead, you "fix them" by introducing a computational fudge factor

Detailed balance:
\[ a \cdot L(r) q(r,s) = L(s) q(s,r) \]
Solve for \( a \):
\[ a = \frac{L(s) q(s,r)}{L(r) q(r,s)} \]

Metropolis-Hastings Algorithm

This leads to the following algorithm:

1. Start with \( x^{(0)} \), then iterate:
2. propose \( x' \) from \( q(x^{(0)}, x') \)
3. calculate ratio
\[ \alpha = \frac{\pi(x')q(x', x^{(i)})}{\pi(x^{(i)})q(x^{(i)}, x')} \]
4. if \( \alpha > 1 \) accept \( x^{(i+1)} = x' \)
5. else accept with probability \( \alpha \)
6. if rejected: \( x^{(i+1)} = x^{(i)} \)

\[ \alpha \geq 1 \]
Note: you can just make up transition probability \( q \) on-the-fly, using whatever criteria you wish.

Proposition Density \( q(x, x') \)

Note: the transition probabilities \( q(x', x) \) can be arbitrary distributions. They can depend on the current state and change at every time step, if you want.

Metropolis Hastings Example

\( P(x_1) = .2 \)
\( P(x_2) = .3 \)
\( P(x_3) = .5 \)

Proposal distribution
\[ q(x_i, (x_{i-1})_{mod 3}) = .4 \]
\[ q(x_i, (x_{i+1})_{mod 3}) = .6 \]
Variants of MCMC

- there are many variations on this general approach, some derived as special cases of the Metropolis-Hastings algorithm.

The Metropolis Algorithm

When $q$ is symmetric, i.e., $q(x,x') = q(x',x)$: e.g. Gaussian

1. Start with $x^{(0)}$, then iterate:
   1. propose $x'$ from $q(x^{(t)},x')$
   2. calculate ratio
      \[ \alpha = \frac{\pi(x')}{\pi(x^{(t)})} \frac{q(x^{(t)},x')}{q(x',x^{(t)})} \]
   3. if $\alpha > 1$ accept $x^{(t+1)} = x'$
      else accept with probability $\alpha$
      if rejected: $x^{(t+1)} = x^{(t)}$

Gibbs Sampler

Special case of MH with acceptance ratio always 1 (so you always accept the proposal).

\[ q(x, y) = \prod_{i=1}^{k} q(x_i, y_i) \quad y_i = x_i, \ i = 1, \ldots, k, \text{ otherwise.} \]

With this proposal, the corresponding acceptance probability is given by

\[ \alpha = \frac{\pi(y)q(y,x)}{\pi(x)q(x,y)} = \prod_{i=1}^{k} \frac{\pi(y_i|x_i)}{\pi(x_i|y_i)} \]

since $y_i = x_i$, by definition of conditional probability for $\theta = (\theta_1, \theta_2)$.

S.Brooks, "Markov Chain Monte Carlo and its Application".

Simulated Annealing

- introduce a "temperature" term that makes it more likely to accept proposals early on. This leads to more aggressive exploration of the state space.
- Gradually reduce the temperature, causing the process to spend more time exploring high likelihood states.
- Rather than remember all states visited, keep track of the best state you’ve seen so far. This is a method that attempts to find the global max (MAP) state.
Trans-dimensional MCMC

- Exploring alternative state spaces of differing dimensions (example, when doing EM, also try to estimate number of clusters along with parameters of each cluster).
- Green’s reversible-jump approach (RJMCMC) gives a general template for exploring and comparing states of differing dimension.

Example: People counting

Problem statement: Given a foreground image, and person-sized bounding box*, find a configuration (number and locations) of bounding boxes that cover a majority of foreground pixels while leaving a majority of background pixels uncovered.

*note: height, width and orientation of the bounding boxes may depend on image location – we determine these relationships beforehand through a calibration procedure.


Searching for the Max

The space of configurations is very large. We can’t exhaustively search for the max likelihood configuration. We can’t even really uniformly sample the space to a reasonable degree of accuracy.

\[
\text{config} = \{[x_1, y_1, w_1, h_1, \theta_1], [x_2, y_2, w_2, h_2, \theta_2], \ldots, [x_k, y_k, w_k, h_k, \theta_k]\}
\]

Let \( N \) = number of possible locations for \((x,y)\) in a k-person configuration.

Size of config = \( N^k \)

And we don’t even know how many people there are...

Size of config space = \( N^4 + N^3 + N^2 + \ldots \)

If we also wanted to search for width, height and orientation, this space would be even more huge.

Likelihood Score

To measure how “good” a proposed configuration is, we generate a foreground image from it and compare with the observed foreground image to get a likelihood score.

\[
\text{config} = \{[x_1, y_1, w_1, h_1, \theta_1], [x_2, y_2, w_2, h_2, \theta_2], \ldots, [x_k, y_k, w_k, h_k, \theta_k]\}
\]

generated foreground image

compared with

observed foreground image

Likelihood Score

Bernoulli distribution model

\[
\begin{align*}
\theta_0 &= \text{prob of observing background given a label of background} \\
\theta_1 &= \text{prob of observing foreground given a label of foreground} \\
\theta_2 &= \text{prob of observing background given a label of foreground} \\
\theta_3 &= \text{prob of observing foreground given a label of background} \\
\end{align*}
\]

\[
\text{log likelihood} \quad \log(L(X)) = \sum \log(p_{c}(c|n)) = \sum \log(p_{c}^{m_{c}}(1-p_{c})^{n_{c}-m_{c}})
\]

\[
= \sum \log(m_{c}!/(n_{c}!) \cdot ((1-p_{c})^{n_{c}} / p_{c}^{m_{c}}))
\]

Number of pixels that disagree!
Proposals

• Add a rectangle (birth)

current configuration
  \[\text{add}\]

proposed configuration

• Remove a rectangle (death)

current configuration
  \[\text{remove}\]

proposed configuration

• Move a rectangle

current configuration
  \[\text{move}\]

proposed configuration

Searching for the Max

• Naïve Acceptance
  – Accept proposed configuration if it has a larger likelihood score, i.e.
    \[
    a = \frac{L(\text{proposed})}{L(\text{current})}
    \]

    Accept if \(a > 1\)

    – Problem: leads to hill-climbing behavior that gets stuck in local maxima

MCMC Sampling

• Metropolis Hastings algorithm
  Propose a new configuration
  \[
  a = \frac{L(\text{proposed}) q(\text{current} | \text{proposed})}{L(\text{current}) q(\text{proposed} | \text{current})}
  \]

  Accept if \(a > 1\)

  Else accept anyways with probability \(a\)

Difference from Naïve algorithm

Searching for the Max

• The MCMC approach
  – Generates random configurations from a distribution proportional to the likelihood!
Searching for the Max

- The MCMC approach
  - Generates random configurations from a distribution proportional to the likelihood!
  - This searches the space of configurations in an efficient way.
  - Now just remember the generated configuration with the highest likelihood.

MCMC in Action

Max likelihood configuration

looking good!

Examples