MCMC: Markov Chain Monte Carlo

Yuzhen Ye
School of Informatics and Computing
Indiana University, Bloomington
Spring 2013
Contents

- Review of Markov Chains
- Monte Carlo simulation
- Introduction of MCMC
  - Motivating problems
  - MCMC updating schemes
- Practical implementation issues
  - The choice of *transition mechanism* for the chain
  - The number of chains to be run and their length
- MCMC algorithms & applications
  - Gibbs sampler (motif finding problem)
An integer time stochastic process, consisting of a set of $m>1$ states $\{s_1, ..., s_m\}$ and
1. An $m$ dimensional initial distribution vector $(p(s_1), ..., p(s_m))$
2. An $m \times m$ transition probabilities matrix $M = (a_{s_is_j})$

For example, for DNA sequence:
the states are $\{A, C, T, G\}$ ($m=4$)
p($A$) the probability of $A$ to be the 1st letter
$a_{AG}$ the probability that $G$ follows $A$ in a sequence.
Motivating problems for MCMC

- The integration operation that plays a fundamental role in Bayesian statistics
  - For calculating the normalizing constant
  - Marginal distribution
  - Expectation

- MCMC, first introduced by Metropolis (1953), provides an alternative whereby we sample from the posterior directly, and obtain sample estimates of the quantities of interest
Sampling and optimization

- To maximize a function, $f(x)$:
  - Brute force method: try all possible $x$
  - Sample method: sample $x$ from probability distribution: $p(x) \sim f(x)$
  - Idea: suppose $x_{\text{max}}$ is a maximum of $f(x)$, then it is also maximum of $p(x)$, thus we have a high probability of sampling $x_{\text{max}}$
Monte Carlo simulation

- The idea of Monte Carlo simulation is to draw an i.i.d set of N samples from a target density p(x) defined on a high-dimensional space X.

\[ I_N(f) = \frac{1}{N} \sum_{i=1}^{N} f(x^{(i)}) N \rightarrow \infty I(f) = \int_X f(x)p(x)dx \]

- The N samples can also be used to obtain a maximum of the objective function p(x)
Rejection sampling

- Sample from a distribution $p(x)$, which is known up to a proportionality constant, by sampling from another easy-to-sample proposal distribution $q(x)$ that satisfies $p(x) \leq Mq(x)$, using an accept/reject procedure.

Ref: An introduction to MCMC for Machine Learning, 2003
**MCMC algorithms**

When samples cannot be drawn from \( p(x) \) directly but \( p(x) \) can be evaluated up to a normalizing constant, MCMC can be used, which is a strategy for generating samples \( x \) while exploring the state space \( X \) using a Markov chain mechanism.

\[
T = \begin{bmatrix}
0 & 1 & 0 \\
0 & 0.1 & 0.9 \\
0.6 & 0.4 & 0
\end{bmatrix}
\]

\[
\mu(x^{(1)}) = (0.5, 0.2, 0.3)
\]

\[
\mu(x^{(1)})T = (0.2, 0.6, 0.2)
\]

\( \mu(x^{(1)})T^t \) converges to \( p(x) = (0.2, 0.4, 0.4) \)
MCMC: basics

- Any Markov chain which is irreducible and aperiodic will have a unique stationary distribution.
  - Irreducibility: from any state of the Markov chain, there is a positive probability of visiting all other states (i.e., the transition matrix cannot be reduced to separate smaller matrices).
  - Aperiodicity: the chain should not get trapped in cycles.

- From any starting point, the chain will converge to the invariant distribution \( p(x) \), as long as \( T \) is a stochastic transition matrix that have the two properties: irreducibility & aperiodicity.

- MCMC samplers are irreducible and aperiodic Markov chains that have the target distribution as the invariant distribution.
MCMC approaches

- The Metropolis-Hastings (MH) algorithm
  - The MH algorithm is the most popular MCMC method
  - Most practical MCMC algorithms can be interpreted as special cases or extensions of this algorithm
- Simulated annealing for global optimization
- Mixtures and cycles of MCMC kernels
  - It is possible to combine several samplers into mixtures and cycles of the individual samplers
- The Gibbs sampler
The motif finding problem

- Given a set of DNA sequences:

  cctgatagacgctatctggctatccacgtacgtaggtcctctctgtgcgaatctatgcgtttccaaccat
  agtactgtgtacatttgatacgtagcgtacaccggcaacctgaaacaaacgctcagaaccagaagtgc
  aaacgtacgtgcaccctctttcttctgtggctcttgccacgccggctgatgtataagcgaatatttt
  agcctccgatgtaagtcataagtactgtgtaactattacctgccaccctctattcattccgtacgtataca
  ctgttataacaacgcgtcatggcgggtatgctttttgtctgtctgcgtacgctgtagttaacgtacgtc

- Find the motif in each of the individual sequences
The motif finding problem

- If starting positions \( s = (s_1, s_2, \ldots, s_t) \) are given, finding consensus is easy because we can simply construct (and evaluate) the profile to find the motif.

- But… the starting positions \( s \) are usually not given. How can we find the “best” profile matrix?
  - Gibbs sampling
  - Expectation-Maximization algorithm
Notations

- Set of symbols: $\Sigma$
- Sequences: $S = \{S_1, S_2, \ldots, S_N\}$
- Starting positions of motifs: $A = \{a_1, a_2, \ldots, a_N\}$
- Motif model ($\theta$): $q_{ij} = P(\text{symbol at the } i\text{-th position} = j)$
- Background model ($\theta_0$): $p_j = P(\text{symbol} = j)$
- Count of symbols in each column: $c_{ij} = \text{count of symbol } j \text{ in the } i\text{-th column in the aligned motif instances}$
Motif finding problem

- Problem: find starting positions and model parameters simultaneously to maximize the posterior probability:

\[ \max_{\theta, A} P(\theta, A \mid S) \]

- This is equivalent to maximizing the likelihood by Bayes’ Theorem, assuming a uniform prior distribution over different models:

\[ \max_{\theta, A} P(S \mid A, \theta) \]
Equivalent scoring function

- Maximize the log-odds ratio:

\[
P(S \mid A, \theta) = \prod_{i=1}^{W} \prod_{j=1}^{\Sigma} q_{ij}^{c_{ij}} \quad P(S \mid A, \theta_0) = \prod_{i=1}^{W} \prod_{j=1}^{\Sigma} p_j^{c_{ij}}
\]

Motif model \((\theta)\) : \(q_{ij} = \text{P(symbol at the i-th position = j)}\)
Background model \((\theta_0)\): \(p_j = \text{P(symbol = j)}\)
\(c_{ij}\): \(\text{ #(symbol j at position i)}\)

\[
F = \log \frac{P(S \mid A, \theta)}{P(S \mid A, \theta_0)} = \sum_{i=1}^{W} \sum_{j=1}^{\Sigma} c_{ij} \log \frac{q_{ij}}{p_j}
\]

Log of the ratio
Gibbs sampling

- Idea: a joint distribution in a high dimension may be hard to sample from, but it may be easy to sample from the conditional distributions where all variables are fixed except one.
- To sample from $p(x_1, x_2, \ldots x_n)$, let each state of the Markov chain represent $(x_1, x_2, \ldots x_n)$, the probability of moving to a state $(x_1, x_2, \ldots x_n)$ is: $p(x_i | x_1, \ldots x_{i-1}, x_{i+1}, \ldots x_n)$. It is a algorithm in a class of sampling techniques called *Markov Chain Monte Carlo (MCMC)* method.
• Start with random motif locations and calculate a motif model
• Randomly select a sequence, remove its motif and recalculate temporary model
• With temporary model, calculate probability of motif at each position on sequence
• **Select new position based on this distribution**
• Update model and Iterate

ETC…
Initialization, $t=0$, sample

\[ \begin{pmatrix} x_1^{(0)}, x_2^{(0)}, x_3^{(0)}, \ldots, x_n^{(0)} \end{pmatrix} \]

\[ x_1^{(t+1)} \sim P \left( x_1 \mid x_2^{(t)}, x_3^{(t)}, \ldots, x_n^{(t)} \right) \]

\[ x_2^{(t+1)} \sim P \left( x_2 \mid x_1^{(t+1)}, x_3^{(t)}, \ldots, x_n^{(t)} \right) \]

\[ x_3^{(t+1)} \sim P \left( x_3 \mid x_1^{(t+1)}, x_2^{(t+1)}, x_4^{(t)}, \ldots, x_n^{(t)} \right) \]

\[ \ldots \]

\[ x_{n-1}^{(t+1)} \sim P \left( x_{n-1} \mid x_1^{(t+1)}, x_3^{(t+1)}, \ldots, x_{n-2}^{(t+1)} \right) \]

\[ x_n^{(t+1)} \sim P \left( x_n \mid x_1^{(t+1)}, x_3^{(t+1)}, \ldots, x_{n-1}^{(t+1)} \right) \]
Estimator of $\theta$

- Given an alignment $A$, i.e. the starting positions of motifs, $\theta$ can be estimated by its MLE with prior probabilities (e.g. Dirichlet prior with parameter $b_j$):

$$ q_{ij} = \frac{c_{ij} + b_j}{N - 1 + B} $$

where $B = \sum_j b_j$
Finding TF binding sites

Gene 1
CACGTGT

Gene 2
CACGTGA

Gene 3
CAAGTGA

Gene 4
CAGGTGA

Transcription factor binding site, or motif instances
Sampling motifs on trees

Using both the overpresentation property and the evolutionary conservation property of motifs

Ref: Sampling motifs on phylogenetic trees, Li & Wong, PNAS, 2005
Initialization

**Parameters** are sampled using prior distributions;

**Motif instances in current species** are sampled from sequences directly for each current species;

**Motif instances in ancestral species** are randomly assigned with one of its immediate child motif instances.
Motif instance updating

Updating motif instances in \textit{ancestral} species

\[
\Pr(A_1^{(0)} \mid A_{[1]}^{(0)}, A_1^{(1)}, A_1^{(2)}, \Theta_0, p_1, p_2, w, M_{11}, M_{12})
\]

Updating motif instances in \textit{current} species

\[
\Pr(A_1^{(1)} \mid A_1^{(0)}, S, \Theta_0, p_1, w, M_{11})
\]
Motif instance updating

Updating motif instance in ancestral species

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>.036</td>
<td>.892</td>
<td>.036</td>
<td>.036</td>
<td>.036</td>
<td>.036</td>
<td>.892</td>
<td>.036</td>
<td>.036</td>
</tr>
<tr>
<td>C</td>
<td>.892</td>
<td>.036</td>
<td>.892</td>
<td>.036</td>
<td>.036</td>
<td>.036</td>
<td>.036</td>
<td>.75</td>
<td>.75</td>
</tr>
<tr>
<td>G</td>
<td>.036</td>
<td>.036</td>
<td>.036</td>
<td>.892</td>
<td>.036</td>
<td>.892</td>
<td>.036</td>
<td>.036</td>
<td>.036</td>
</tr>
<tr>
<td>T</td>
<td>.036</td>
<td>.036</td>
<td>.036</td>
<td>.036</td>
<td>.892</td>
<td>.036</td>
<td>.036</td>
<td>.178</td>
<td>.178</td>
</tr>
</tbody>
</table>

M_{11} -> M_{12} 
CCCGTGACC -> CACGTGAAC

2th position
A: 0.932...
C: 0.067
G: 8.4e-6
T: 2.5e-4
Motif instance updating

Updating motif instances for current species

Updated ancestral motif instance
CACTTGAAC

$M_{11}$

$M_{12}$

...CACACCACGTCGAGCTT...

...CACATCACGTGAACCTT...
Parameter sampling step

- Metropolis-Hasting algorithm is used to increase or decrease the width $w$ of the motif by 1 from the left or right side
Other applications of Gibbs sampling

- Biclustering microarray data by Gibbs sampling
  - Microarray data is discretized
  - Bioinformatics, 2003
- Assignment of ambiguously mapped reads
  - Bioinformatics, 2010
Practical implementation issues

- How many iterations?
- One run or many?