Lecture 4
Jump Diffusion MCMC
Pattern Recognition and Machine Learning

Dr. Guoliang Fan
School of Electrical and Computer Engineering
Oklahoma State University
Goal

- To review a few important sampling methods involved in Monte Carlo methods.

- To present the jump diffusion MCMC (JD-MCMC) method for a disconnected parameter space.

- To apply JD-MCMC for object detection.
Suppose that it is hard to sample \( p(x) \) but that it is possible to “walk around” in \( X \) using only local state transitions.

**Insights:**

- We can use a “random walk” to help us draw random samples from \( p(x) \).
- What we need is a proposal function that allows us to move locally based on the previous sample.
- A new sample is accepted or rejected (the old one is recorded again) according to the ratio between the evaluation of the present sample over that of previous one.
- The way that samples are evaluated determines the final sample distribution.
The general idea of the algorithm is to generate a series of samples that are correlated in a Markov chain and that can be used to match the unknown distribution $p(x)$.

$$\left(x^{(1)}, x^{(2)}, \ldots, x^{(\tau)}\right) \xrightarrow{\tau \to \infty} p(x)$$

What ensures that final samples will follow the unknown distribution?
Gibbs Sampling

- It is an algorithm to generate a sequence of samples from the joint probability distribution of two or more random variables. The purpose is to approximate the joint distribution of multiple variables.
Jump-Diffusion MCMC

- Jump-diffusion provide a mixed mechanism to draw samples from a disconnected state space where both discrete and continuous state variables exist.
  - **Jump** contributes in sampling over the parameter size.
    - Can be controlled by a probability
  - **Diffusion** contributes in sampling over the parameter values.
    - Can be managed by a random walk.
Application to Object Detection

- There are two kinds of parameters
  - The number of objects, $k$, 
  - The location of each object \( \Theta_k = \{(x_i, y_i) | i = 1, \ldots, k\} \)

- Two probabilistic functions
  - The prior probability of the model order $k$ is assumed to be Poisson with parameter $\lambda$
    \[
p(K^* = k) = \frac{\lambda^k}{e^\lambda k!}
    \]
  - Given the input image $I$, the likelihood function of the hypothesized $k$ objects with locations $\Theta_k$ is defined as
    \[
p(I | \Theta_k) \propto \exp\left(-\frac{||I - J(\Theta_k)||^2}{\sigma^2}\right)
    \]
Objective Function

- Given a prior probability of model order $k$ and an observed image $Y$, the solution of object detection (i.e., $\Theta$: object locations) is represented by the joint posterior probability density as:

$$p(\Theta, k | I) = \frac{p(\Theta, k, I)}{p(I)} = \frac{p(I | \Theta, k) p(\Theta, k)}{p(I)}$$

- Unknown posterior density of the object number and locations

- $\propto p(I | \Theta, k) p(\Theta, k)$

- $\propto p(I | \Theta, k) p(k)$

- Data likelihood given model $\Theta$

- Prior for the model order $k$
## Jump Diffusion MCMC Algorithm

- Initialize locations of $k$ hypothesized objects and the maximum order $K_{\text{max}}$.
- for $i=1:N$
  - Draw a sample $a \sim U(0,1)$
  - If $a < 0.33$ and $k > 1$ (jump by -1)
    - $k = k - 1$
    - MCMC Gibbs sampling
    - Accept or reject by Metropolis Sampling
  - else if $a < 0.66$ and $k < K_{\text{max}}$ (jump by +1)
    - $k = k + 1$
    - MCMC Gibbs sampling
    - Accept or reject by Metropolis Sampling
  - else (no jump)
    - MCMC diffusion (Gibbs sampling)
    - Accept or reject by Metropolis Sampling
- End
- Select samples after $M$ iterations (burn-in);
- Obtain a set of samples with certain step size.
- Compute the mean estimate of object number and the location of each one.

$$p(\Theta, k \mid Y) \propto p(Y \mid \Theta) p(k)$$

PDF of interest used for evaluation

$$\alpha = \min \left( \frac{p(Y \mid \Theta^{(B)}) p(k^{(B)})}{p(Y \mid \Theta^{(A)}) p(k^{(A)})}, 1 \right)$$

(acceptance probability for jump)

$$\beta = \left( \frac{p(Y \mid \Theta^{(B)})}{p(Y \mid \Theta^{(A)})}, 1 \right)$$

(acceptance probability for no jump)
How to get the final solution?

- After enough sampling, we can use “burn-in” to throw away $M$ samples in the beginning, and only use the later samples with step size $N$ to compute the solution.

- Then we do a mean estimation for selected samples to find the unique deterministic solution.

$$k^* = \text{Round} \left( \frac{1}{L} \sum_{i=1}^{L} k_{(M+Ni)} \right) \quad (k_i : \text{the } i\text{th sample of the object number})$$

$$\bar{x}^{(l)} = \text{Average} \left( x_{k_i=k^*}^{(l)} \right) \quad (x_{k_i=k^*}^{(l)} : \text{the } l\text{th object' position of the } i\text{th sample with } k^* \text{ order})$$

(Note: It is necessary to re-order all objects in the list of each sample to make sure the average is correct)
Object Location Re-ordering

- To make sure the mean estimation for object position is correct, a re-ordering is needed for the position list associated with each sample (with order $k^*$) before computing the average.
  \[ \overline{x}^{(l)} = \text{Average}(x^{(l)}_{k_i=k^*}) \] (the $l$th object's position of the $i$th sample with $k^*$ order)

- This can be done by setting certain rule of object ordering:
  - According to the distance to the top-left or bottom-right corner.
JD-MCMC for Object Detection

\[ k_0 = 2 \quad \text{and} \quad k_0 = 30 \]