Dicinging

Chen Yunchuan
2015.4.29
Outline

- Sampling
  - Random Variable Generation
  - Rejection Sampling
  - Importance Sampling
  - Markov Chain Monto Carlo (MCMC)
- Restricted Boltzmann Machine Revisit
  - Learning algorithms
- Imagination: Represent Word with Random Vectors
Example of Monte Carlo: Estimate $\pi$

**Procedure direct-pi**

$N_{\text{hits}} \leftarrow 0$ (initialize)

for $i = 1, \ldots, N$ do

\[
\begin{cases}
  x \leftarrow \text{ran}(-1, 1) \\
  y \leftarrow \text{ran}(-1, 1) \\
  \text{if } (x^2 + y^2 < 1) \text{ then } N_{\text{hits}} \leftarrow N_{\text{hits}} + 1
\end{cases}
\]

output $N_{\text{hits}}$

- Estimate $\pi$

\[
\pi \approx \frac{4N_{\text{hits}}}{N}
\]

<table>
<thead>
<tr>
<th>Run</th>
<th>$N_{\text{hits}}$</th>
<th>Estimate of $\pi$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3156</td>
<td>3.156</td>
</tr>
<tr>
<td>2</td>
<td>3150</td>
<td>3.150</td>
</tr>
<tr>
<td>3</td>
<td>3127</td>
<td>3.127</td>
</tr>
<tr>
<td>4</td>
<td>3171</td>
<td>3.171</td>
</tr>
<tr>
<td>5</td>
<td>3148</td>
<td>3.148</td>
</tr>
</tbody>
</table>
Estimating $\pi$

Draw i.i.d. set of samples $\{\mathbf{x}^{(i)}\}_{i=1}^{N}$ from a target density $p(\mathbf{x} \mid \cdot)$

$$
\int_{\mathcal{X}} f(\mathbf{x}) p(\mathbf{x} \mid \cdot) d\mathbf{x} \approx \frac{1}{N} \sum_{i=1}^{N} f(\mathbf{x}^{(i)})
$$

$$
\pi = \frac{S_C}{S_D} = \frac{\iint_{\mathcal{C}} dxdy}{\iint_{\mathcal{D}} dxdy} = \frac{\iint_{\mathcal{C}} p(x, y) dxdy}{\iint_{\mathcal{D}} p(x, y) dxdy}
$$

$$
= \iint_{\mathcal{D}} \mathbb{1}\{ (x, y) \in \mathcal{C} \} p(x, y) dxdy
$$

$$
\approx \frac{1}{N} \sum_{i=1}^{N} \mathbb{1}\{ (x^{(i)}, y^{(i)}) \in \mathcal{C} \} = \frac{N_{\text{hit}}}{N}
$$
Another Way to Estimate $\pi$

\[ \pi \approx \frac{2lN}{aN_{\text{hits}}} \]
Estimating $\pi$

$$p_{\text{hit}} = \frac{S_g}{S_G} = \frac{\int_0^\pi \frac{l}{2} \sin \varphi \, d\varphi}{\frac{1}{2} a \pi} = \frac{2l}{\pi a}$$

$$p_{\text{hit}} \approx \frac{N_{\text{hits}}}{N}$$

$$\pi \approx \frac{2lN}{a N_{\text{hits}}}$$
Estimating $\pi$

\[
\frac{2l}{\pi a} = \frac{S_g}{S_G} = \frac{\iint g \, dx \, dy}{\iint G \, dx \, dy} = \frac{\iint g \, p(x, y) \, dx \, dy}{\iint G \, p(x, y) \, dx \, dy} = \iint_G \mathbb{1}\{(x, y) \in g\} p(x, y) \, dx \, dy = \mathbb{E}[\mathbb{1}\{(x, y) \in g\}] = \frac{1}{N} \sum_{i=1}^{N} \mathbb{1}\{(x, y) \in g\} = \frac{N_{\text{hits}}}{N}
\]
Monte Carlo: Motivation

- Estimation
- Optimization
- Model Selection
Random Variable Generation

- if $U \sim \mathcal{U}_{[0,1]}$, then the random variable $F^{-}(U)$ has the distribution $F$
  
  where $F^{-}(u) = \inf\{x : F(x) \geq u\}$

- The theorem above is theoretical beautiful, but unpractical in real application
  
  - It is hard to compute the generalized inverse function
  
  - High dimensional cases?
Rejection Sampling

Set $i = 1$
Repeat until $i = N$

1. Sample $x^{(i)} \sim q(x)$ and $u \sim \mathcal{U}(0,1)$.

2. If $u < \frac{p(x^{(i)})}{Mq(x^{(i)})}$ then accept $x^{(i)}$ and increment the counter $i$ by 1. Otherwise, reject.

Drawbacks?
Importance Sampling

\[
\int f(x)p(x) \, dx = \int f(x) \frac{p(x)}{q(x)} q(x) \, dx \\
\approx \frac{1}{N} \sum_{i=1}^{N} f(x^{(i)}) \frac{p(x^{(i)})}{q(x^{(i)})}
\]
• Does the i.i.d. property necessary for estimating an integral?
Another $\pi$ Estimation Story
Stationary Stochastic Process

\[ \mathbb{E}[X_t] = \text{constant} = \lim_{T \to +\infty} \frac{1}{T} \int_0^T X(t) \, dt \]

\[ \mathbb{E}[f(X_t)] \neq \lim_{T \to +\infty} \frac{1}{T} \int_0^T f(X(t)) \, dt \]
Why \( \mathbb{E}[f(X_t)] = \lim_{T \to +\infty} \frac{1}{T} \int_0^T f(X(t)) \, dt \) \( \times \)?
• How to make the equation below hold?

\[ \mathbb{E}[f(X_t)] = \lim_{T \to +\infty} \frac{1}{T} \int_0^T f(X(t)) \, dt \]
Markov Chain Monte Carlo (MCMC): Intuitions
MCMC: Motivation

- Bayesian inference and learning

  \[ p(x|y) = \frac{p(y|x)p(x)}{\int_x p(y|x')p(x') \, dx'} \]

- Normalization

  \[ p(x|y) = \int_z p(x, z, |y) \, dz \]

- Marginalization

- Expectation

  \[ \mathbb{E}_{p(x|y)}[f(x)] = \int_x f(x)p(x|y) \, dx \]

- Optimization

- Statistical mechanics: Compute the partition function

- Penalized likelihood model selection
MCMC: Important Theoretical Results

- Irreducibility
- Aperiodicity
- Detailed Balance Condition

\[ \pi_i P_{ij} = \pi_j P_{ji} \]
MCMC: Applications to Sampling
Algorithms: Metropolis-Hastings Alg. (MHA)

• Proposal Distribution
  \[ q(i, j) = q(x^{(j)} | x^{(i)}) \]

• Acceptance Prob.
  \[ \alpha(i, j) = A(x^{(i)}, x^{(j)}) \]

• Target Distribution
  \[ \pi(i) = p(x^{(i)}) \]

• Detailed Balance
  \[ \pi(i)q(i, j)\alpha(i, j) = \pi(j)q(j, i)\alpha(j, i) \quad \alpha(i, j) = \min\{1, \frac{\pi(j)q(j, i)}{\pi(i)q(i, j)}\} \]
Special Cases of MHA

- Independent Sampler: Assuming that the proposal is independent of the current state
  \[
  q(x^{(j)} | x^{(i)}) = q(x^{(j)})
  \]
  \[
  \alpha(i, j) = \min \left\{ 1, \frac{p(x^{(j)})q(x^{(i)})}{p(x^{(i)})q(x^{(j)})} \right\} = \min \left\{ 1, \frac{w(x^{(j)})}{w(x^{(i)})} \right\}
  \]

- Metropolis Sampler: Assuming that the proposal is symmetric
  \[
  q(x^{(j)} | x^{(i)}) = q(x^{(i)} | x^{(j)})
  \]
  \[
  \alpha(i, j) = \min \left\{ 1, \frac{p(x^{(j)})}{p(x^{(i)})} \right\}
  \]

- Gibbs Sampler: Good property: no rejections
MHA Running Examples
MHA Running Examples
Simulated Annealing (SA)

- **Annealing** is the process of heating a solid until thermal stresses are released. Then, in cooling it very slowly to the ambient temperature until perfect crystals emerge. The quality of the results strongly depends on the cooling temperature. The final state can be interpreted as an energy state (crystalline potential energy) which is **lowest** if a perfectly crystal emerged.

\[
Pr(\mathbf{x}) = \frac{1}{Z} \exp\left(-\frac{E(\mathbf{x})}{k_B T}\right)
\]

\[
Pr(\mathbf{x} \rightarrow \mathbf{x}_p) = \begin{cases} 
1, & \Delta E < 0; \\
\exp\left(-\frac{\Delta E}{k_B T}\right), & \Delta E \geq 0.
\end{cases}
\]
Simulated Annealing Algorithm

1. Initialize $x^{(0)}$ and set $T_0, k = 0$.

2. For $i = 0$ to $N - 1$
   - Loop until converge
     - Sample $u \sim U_{[0,1]}$.
     - Generate candidate state $x^*$ according to current state $x^{(k)}$
     - $\Delta E = E(x^*) - E(x^{(k)})$
     - Accept state $x^*$ with probability:
       \[
       \Pr(x^{(k)} \rightarrow x^*) = \begin{cases} 
       1, & \Delta E < 0; \\
       \exp(-\Delta E/k_B T_i), & \Delta E \geq 0. 
       \end{cases}
       \]
     - if $u < \Pr(x \rightarrow x^*)$, set $x^{(k+1)} = x^*$, else set $x^{(k+1)} = x^{(k)}$
   - Set $T_{i+1}$ according to cooling schedule.
Simulated Annealing Algorithm

1. Initialise \( x^{(0)} \) and set \( T_0 = 1 \).
2. For \( i = 0 \) to \( N - 1 \)
   - Sample \( u \sim \mathcal{U}_{[0,1]} \).
   - Sample \( x^* \sim q(x^* | x^{(i)}) \).
   - If \( u < A(x^{(i)}, x^*) = \min \left\{ 1, \frac{p_T(x^*) q(x^{(i)} | x^*)}{p_T(x^{(i)}) q(x^* | x^{(i)})} \right\} \)
     \[ x^{(i+1)} = x^* \]
     else
     \[ x^{(i+1)} = x^{(i)} \]
   - Set \( T_{i+1} \) according to a chosen cooling schedule.
Toy Example: Maximization
Solve Traveling Salesman Problem with SA

20 cities

Initial configuration
Restricted Boltzmann Machine (RBM)
Restricted Boltzmann Machine (RBM)
Restricted Boltzmann Machine (RBM)
Restricted Boltzmann Machine (RBM)
Basics to Learn RBM

• Fitting $p(v)$ to data

$$p(v) = \frac{1}{Z} \sum_h \exp(-E(v, h))$$

• Notations

$$E(v, h) = -v^T W h - b^T v - c^T h$$

$$= -\sum_{i=1}^{m} \sum_{j=1}^{n} w_{ij} h_i v_j - \sum_{j=1}^{n} b_j v_j - \sum_{i=1}^{m} c_i h_i$$

$$= -W \circ A - b^T v - c^T h$$

$$= -\theta^T \phi(x)$$

$$\theta = [W(:); b; c]$$

$$\phi(x) = [A(:); v; h]$$

$$W = \begin{pmatrix} w_{11} & w_{12} & \cdots & w_{1n} \\ w_{21} & w_{22} & \cdots & w_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ w_{m1} & w_{m2} & \cdots & w_{mn} \end{pmatrix}$$

$$A = \begin{pmatrix} h_{1v_1} & h_{1v_2} & \cdots & h_{1v_n} \\ h_{2v_1} & h_{2v_2} & \cdots & h_{2v_n} \\ \vdots & \vdots & \ddots & \vdots \\ h_{mv_1} & h_{mv_2} & \cdots & h_{mv_n} \end{pmatrix}$$
Learning RBM: Maximum Likelihood Estimation

- Given Observed Samples

\[ S = \{v_1, v_2, \ldots, v_\ell\} \]

- We can get

\[
\frac{1}{\ell} \sum_{v \in S} \frac{\partial \log p(v; \theta)}{\partial w_{ij}} = \mathbb{E}_{p(h|v;\theta)} q(v) [h_i v_j] - \mathbb{E}_{p(x;\theta)} [h_i v_j]
\]

for

\[
\frac{\partial \log p(v_0; \theta)}{\partial \theta} = \mathbb{E}_{p(h|v_0;\theta)} \phi(v_0, h) - \mathbb{E}_{p(x;\theta)} [\phi(x)]
\]

\[
\frac{\partial \log p(v_0; \theta)}{\partial w_{ij}} = \mathbb{E}_{p(h|v_0;\theta)} [h_i v_j] - \mathbb{E}_{p(x;\theta)} [h_i v_j]
\]
Contrastive Divergence

\[
\frac{\partial \log p(v_0; \theta)}{\partial \theta} = \mathbb{E}_p(h|v_0; \theta) \phi(v_0, h) - \mathbb{E}_p(x; \theta) [\phi(x)]
\]

\[
\frac{\partial \log p(v^{(0)}; \theta)}{\partial \theta} \approx \mathbb{E}_p(h|v^{(0)}; \theta) [\phi(v^{(0)}, h)] - \frac{1}{M} \sum_{i=1}^{M} \mathbb{E}_p(h|v^{(k+i)}; \theta) [\phi(v^{(k+i)}, h)]
\]

\[
\text{CD}_k(\theta, v^{(0)}) = \mathbb{E}_p(h|v^{(0)}; \theta) [\phi(v^{(0)}, h)] - \mathbb{E}_p(h|v^{(k)}; \theta) [\phi(v^{(k)}, h)]
\]
**Algorithm 1: k-step contrastive divergence**

**Input:** RBM \( (V_1, \ldots, V_m, H_1, \ldots, H_n) \), training batch \( S \)

**Output:** gradient approximation \( \Delta w_{ij}, \Delta b_j \) and \( \Delta c_i \) for \( i = 1, \ldots, n \), \( j = 1, \ldots, m \)

1. init \( \Delta w_{ij} = \Delta b_j = \Delta c_i = 0 \) for \( i = 1, \ldots, n \), \( j = 1, \ldots, m \)

2. forall the \( v \in S \) do
   
   3. \( v^{(0)} \leftarrow v \)
   
   4. for \( t = 0, \ldots, k - 1 \) do
      
      5. for \( i = 1, \ldots, n \) do sample \( h_i^{(t)} \sim p(h_i | v^{(t)}) \)
      
      6. ;
      
      7. for \( j = 1, \ldots, m \) do sample \( v_j^{(t+1)} \sim p(v_j | h^{(t)}) \)
      
      8. ;

   9. for \( i = 1, \ldots, n \), \( j = 1, \ldots, m \) do
      
      10. \( \Delta w_{ij} \leftarrow \Delta w_{ij} + p(H_i = 1 | v^{(0)}).v_j^{(0)} - p(H_i = 1 | v^{(k)}).v_j^{(k)} \)

   11. for \( j = 1, \ldots, m \) do
      
      12. \( \Delta b_j \leftarrow \Delta b_j + v_j^{(0)} - v_j^{(k)} \)

   13. for \( i = 1, \ldots, n \) do
      
      14. \( \Delta c_i \leftarrow \Delta c_i + p(H_i = 1 | v^{(0)}) - p(H_i = 1 | v^{(k)}) \)
CD-k Running Examples

Note: from bottom to top $k = 1, 2, 5, 10, 20, 100$
Other Algorithms for Training RBM

- Persistent Contrastive Divergence (PCD)
- Fast Persistent Contrastive Divergence (FPCD)
- Stochastic Approximation Procedure (SAP)
- Stochastic Approximation Procedure with Tempered Transitions (Trans-SAP)
- Parallel Tempering (PT)
Stochastic Approximation Procedure with Tempered Transitions

Algorithm 1 Stochastic Approximation Procedure.

1: Given an observation $x_0$. Randomly initialize $\theta^1$ and $M$ sample particles $\{x^{1,1}, \ldots, x^{1,M}\}$.
2: for $t = 1 : T$ (number of iterations) do
3:      for $m = 1 : M$ (number of parallel Markov chains) do
4:         Sample $x^{t+1,m}$ given $x^{t,m}$ using transition operator $T_{\theta^t}(x^{t+1,m} \leftarrow x^{t,m})$.
5:      end for
6:      Update: $\theta^{t+1} = \theta^t + \alpha_t \left[ \phi(x_0) - \frac{1}{M} \sum_{m=1}^{M} \phi(x^{t+1,m}) \right]$.
7:      Decrease $\alpha_t$.
8: end for

Algorithm 2 Tempered Transitions Run.

1: Initialize $\beta_0 < \beta_1 < \ldots < \beta_S = 1$. Given a current state $x^S$.
2: for $s = S - 1 : 0$ (Forward pass) do
3:      Sample $x^s$ given $x^{s+1}$ using $T_s(x^s \leftarrow x^{s+1})$.
4: end for
5: Set $\tilde{x}^0 = x^0$.
6: for $s = 0 : S - 1$ (Backward pass) do
7:      Sample $\tilde{x}^{s+1}$ given $\tilde{x}^s$ using $T_s(\tilde{x}^{s+1} \leftarrow \tilde{x}^s)$.
8: end for
9: Accept a new state $\tilde{x}^S$ with probability: $\min \left[ 1, \prod_{s=1}^{S} p^*(x_s)^{\beta_{s-1} - \beta_s} p^*(\tilde{x}_s)^{\beta_s - \beta_{s-1}} \right]$. 

Stochastic Approximation Procedure with Tempered Transitions
Parallel Tempering
Parallel Tempering

1) Create a sequence of RBMs \( (R_0, R_1, \cdots, R_K) \) such that parameters of \( R_k \) are \( \Theta_k = (T_k W, b, c) \), where \( 0 \leq T_0 < T_1 < \cdots < T_K = 1 \).
2) Create an empty set of samples \( X = \{ \} \).
3) Set \( x_0 = (x_{0,0}, \cdots, x_{K,0}) \) such that every \( x_{k,0} \) is a uniformly distributed random vector (or use old ones from the previous epoch).
4) For \( m = 1 \) to \( M \), repeat
   a) Sample \( x_m = (x_{0,m}, \cdots, x_{K,m}) \) from the sequence of RBMs such that \( x_{k,m} \) is sampled by one-step Gibbs sampling starting from \( x_{k,m-1} \).
   b) For \( j = K \) to 1, repeat
      - Swap \( x_{j,m} \) and \( x_{j-1,m} \) according to \( P_{\text{swap}}(x_{j,m}, x_{j-1,m}) \) computed using \( P_{\text{swap}} \).
   c) Add \( x_{K,m} \) to \( X \).
5) \( X \) is the set of samples collected by parallel tempering sampling.

\[ P_{\text{swap}}(x_{T_1}, x_{T_2}) = \min \left( 1, \frac{P_{T_1}(x_{T_2}) P_{T_2}(x_{T_1})}{P_{T_1}(x_{T_1}) P_{T_2}(x_{T_2})} \right), \]
Parallel Tempering v.s. CD-k:
Running Examples (1/2)

Note: from bottom to top $k = 1, 2, 5, 10, 20, 100$

Note: bottom to top $M = 4, 5, 10, 50$
Parallel Tempering v.s. CD-k: Running Examples (2/2)

Positive Samples

Negative Samples
References


- Salakhutdinov, Ruslan. ‘Learning in Markov Random Fields using Tempered Transitions.’, NIPS, 2009


- Desjardins, Guillaume, Aaron C Courville, Yoshua Bengio, Pascal Vincent, and Olivier Delalleau. ‘Parallel Tempering for Training of Restricted Boltzmann Machines ’, *AISTATS*, 2010


- Cho, Kyunghyun, Tapani Raiko, and Alexander Ilin. ‘Parallel tempering is efficient for learning restricted Boltzmann machines’, International Joint Conference on Neural Networks (IJCNN), 2010
Thanks