CS598JHM: Advanced NLP (Spring 2013)

http://courses.engr.illinois.edu/cs598jhm/

Lecture 5: Sampling (Monte Carlo Methods)

Julia Hockenmaier

juliahmr@illinois.edu

3324 Siebel Center

Office hours: by appointment

The goal of Monte Carlo methods

Given a target distribution $P(\mathbf{x})$ that we cannot evaluate exactly, we use Monte Carlo (= sampling) methods to:

1. Generate samples $\{ \mathbf{x}^{(1)} \ldots \mathbf{x}^{(r)} \ldots \mathbf{x}^{(R)} \}$ from $P(\mathbf{x})$

2. Estimate the expectation of functions $\phi(\mathbf{x})$ under $P(\mathbf{x})$

In Bayesian approaches, model parameters θ are also random variables. So, the target distribution $P(\mathbf{x})$ may in fact be $P(\theta \mid \mathbf{D})$, the posterior of θ given the data

The expectation Φ of $\phi(\mathbf{x})$:

$$\Phi = \langle \phi(\mathbf{x}) \rangle_{P(\mathbf{x})} = \sum_{\mathbf{x}} P(\mathbf{x}) \phi(\mathbf{x})$$

We can estimate Φ by **Monte Carlo integration:**
Draw a finite number of samples $\{\mathbf{x}^{(1)} ... \mathbf{x}^{(R)}\}$ from $P(\mathbf{x})$ and estimate Φ as

$$\hat{\Phi} = \frac{1}{R} \sum_{r} \phi(\mathbf{x}^{(r)})$$

The variance of Φ is a function of σ^2 / R
(σ^2 is the variance of $\phi(\mathbf{x})$; R is the number of samples).

$$\sigma^2 = \mathrm{E}[(\phi - \Phi)^2 \mid \mathbf{x}] = \sum_{\mathbf{x}} P(\mathbf{x}) \, (\phi(\mathbf{x}) - \Phi)^2$$

The accuracy of Φ is independent of the dimensionality of $\mathbf{x}$.
A dozen *independent* samples from $P(\mathbf{x})$ may be enough

Why is sampling hard?

We need to be able to draw **independent** samples from $P(\mathbf{x})$

This is difficult, because:

- Our models are often of the form $P(\mathbf{x}) \propto f(\mathbf{x})$
  i.e $P(\mathbf{x}) = f(\mathbf{x}) / Z$

- We often cannot compute the partition function $Z = \sum_{\mathbf{x}'} f(\mathbf{x}')$
  because we usually operate in very high dimensions
  (or very large state spaces).

Sampling methods

Sampling from a fixed proposal distribution $Q(\mathbf{x}) \neq P(\mathbf{x})$:

- Uniform sampling
- Importance sampling
- Rejection sampling

Markov Chain Monte Carlo (MCMC)

Sampling from a Markov chain: the probability of the next sample (= proposal distribution) depends on the current state

- Metropolis-Hastings
- Gibbs sampling

Uniform sampling

Assume $\mathrm{Q}(\mathbf{x})$ is uniform. Draw samples $\mathbf{x}^{(\mathrm{r})} \sim \mathrm{Q}(\mathbf{x})$ (uniformly) from the state space, evaluate $\mathrm{P}(\mathbf{x}^{(\mathrm{r})})$ at $\mathbf{x}^{(\mathrm{r})}$

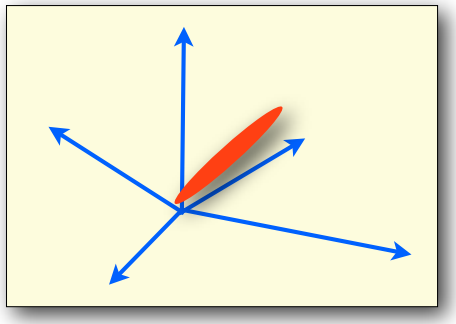
This gives a new distribution

$$P'(\mathbf{x}^{(r)}) = \frac{f(\mathbf{x}^{(r)})}{\sum_{i=1}^{R} f(\mathbf{x}^{(i)})}$$

Estimate $\langle \varphi(\mathbf{x}) \rangle_{\mathrm{P}(\mathbf{x})}$ as

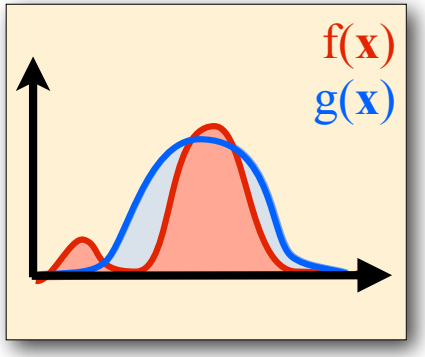
$$\hat{\mathbf{\Phi}} = \sum_{r} \phi(\mathbf{x}^{(r)}) P'(\mathbf{x}^{(r)})$$

Problem: Unless $\mathrm{P}(\mathbf{x})$ itself is close to uniform, this strategy will be very inefficient. In high-dimensional spaces, most regions of the state space have typically very low probability



Importance sampling

Importance sampling can be used to compute expectations Φ



Assumptions:

- We can evaluate $P(\mathbf{x}) \propto f(\mathbf{x})$ at any point
- We have a simple **sample density** $Q(\mathbf{x}) \propto g(\mathbf{x})$, which we can evaluate and draw samples from,
- For any $\mathbf{x}$: if $P(\mathbf{x}) > 0$, then also $Q(\mathbf{x}) > 0$

Algorithm

- Draw samples from $Q(\mathbf{x}) \propto g(\mathbf{x})$
- Re-weight samples by $w_r = f(\mathbf{x}^{(r)})/g(\mathbf{x}^{(r)})$
- Estimate Φ as $\hat{\Phi} = \frac{\sum_r w_r \phi(\mathbf{x}^{(r)})}{\sum_r w_r}$

This converges to Φ. But: We can't estimate the variance of Φ. The *empirical* variances of w_r and $w_r\phi(\mathbf{x}^{(r)})$ may not be a good indicator of their *true* variances

Rejection sampling

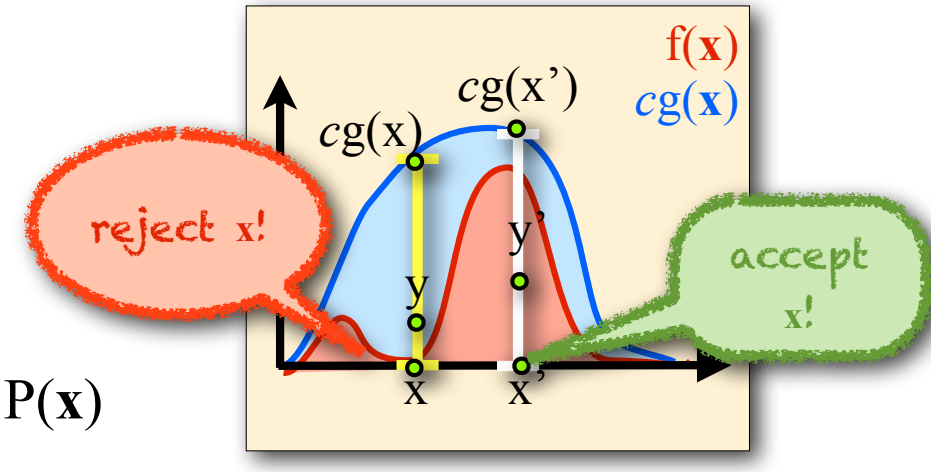
Assumptions:

- We can evaluate $P(\mathbf{x}) \propto f(\mathbf{x})$ at any point
- We have a simple **proposal density** $Q(\mathbf{x}) \propto g(\mathbf{x})$, which we can evaluate and draw samples from
- We know the value of a constant c such that $cg(\mathbf{x}) > f(\mathbf{x})$

Algorithm:

- Sample $\mathbf{x} \sim Q(\mathbf{x})$ and evaluate $cg(\mathbf{x})$
- Sample $y \sim \text{Uniform}([0, cg(\mathbf{x})])$
- If $f(\mathbf{x}) \geq y$, accept $\mathbf{x}$, else reject $\mathbf{x}$

This returns independent samples from $P(\mathbf{x})$



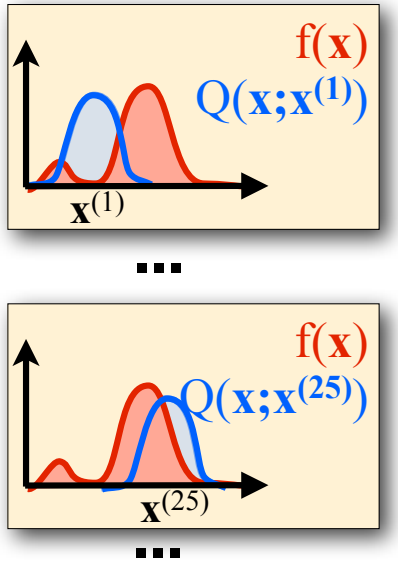
Problems:

Acceptance rate: $\int P(\mathbf{x})d\mathbf{x} / \int cQ(\mathbf{x})\, d\mathbf{x} = 1/c$

But c grows exponentially with the dimensionality of $\mathbf{x}$

Markov Chain Monte Carlo

Rejection sampling and importance sampling only work well when $Q(\mathbf{x})$ is similar to $P(\mathbf{x})$. This is difficult to achieve in high dimensions.



Markov Chain Monte Carlo methods generate a sequence of samples $\mathbf{x}^{(1)}_{\ldots}\ \mathbf{x}^{(t)}_{\ldots}\ \mathbf{x}^{(T)}$

$\mathbf{x}^{(t+1)}$ is drawn from a proposal distribution $Q(\mathbf{x};\ \mathbf{x}^{(t)})$ which depends on the last sample $\mathbf{x}^{(t)}$

Advantage: $Q(\mathbf{x};\ \mathbf{x}^{(t)})$ does not have to be similar to $P(\mathbf{x})$

Disadvantage: the samples $\mathbf{x}^{(t)}$ are correlated, and not independent. We may have to generate a lot of samples $(T \gg R)$ to get a sequence of R independent samples $\mathbf{x}^{(1)}_{\ldots}\ \mathbf{x}^{(R)}$

Markov chains

A (discrete-time, discrete-state) Markov chain over N states $\{x_1,...,x_N\}$ is defined by

- an N-dimensional multinomial $P^{(0)}$, the **initial distribution** over the states $\{x_1,...,x_N\}$

- an $N \times N$ **transition matrix** A that defines the transition probability of moving from state x_i to state x_j :
  $A_{ij} = P(X^{(t+1)} = x_j \mid X^{(t)} = x_i)$

The Markov chain defines a sequence of distributions $P^{(0)}...P^{(t)}...$ over the states $\{x_1,...,x_N\}$:

$$P^{(t)} = P^{(t-1)} \times A = P^{(0)} \times A^{(t-1)}$$

More Markov chain terminology

A Markov chain is **irreducible** if any state can be reached from any other state with nonzero probability.

An irreducible Markov chain is **recurrent** if the probability of reaching any state x_j from any state x_i in finite time is 1.

An irreducible, recurrent Markov chain is **positive recurrent** if it has a *stationary (= equilibrium) distribution* $\boldsymbol{\pi} = \lim_{t\to\infty} P^{(t)}$

An **ergodic** Markov chain will converge to $\boldsymbol{\pi}$ regardless of its start state.

A **reversible** Markov chain obeys **detailed balance:**

$$\boldsymbol{\pi}(x_i)\, P(X^{(t+1)} = x_j \mid X^{(t)} = x_i) = \boldsymbol{\pi}(x_j)\, P(X^{(t+1)} = x_i \mid X^{(t)} = x_j)$$

MCMC: Metropolis-Hastings

Algorithm:

1. Given the last sample $\mathbf{x}^{(t)}$, generate $\mathbf{x}' \sim Q(\mathbf{x}; \mathbf{x}^{(t)})$

2. Compute $$a = \frac{f(\mathbf{x}')}{f(\mathbf{x}^{(t)})} \frac{Q(\mathbf{x}^{(t)}; \mathbf{x}')}{Q(\mathbf{x}'; \mathbf{x}^{(t)})} = \frac{P(\mathbf{x}')}{P(\mathbf{x}^{(t)})} \frac{Q(\mathbf{x}^{(t)}; \mathbf{x}')}{Q(\mathbf{x}'; \mathbf{x}^{(t)})}$$

3. If $a > 1$: accept $\mathbf{x}'$
   Otherwise, accept $\mathbf{x}'$ with probability a

4. If $\mathbf{x}'$ is accepted: $\mathbf{x}^{(t+1)} = \mathbf{x}'$
   Otherwise, $\mathbf{x}^{(t+1)} = \mathbf{x}^{(t)}$

Regardless of $Q(\mathbf{x}; \mathbf{x}^{(t)})$, this algorithm samples from an ergodic Markov chain with states $\mathbf{x}$ and stationary distribution $P(\mathbf{X})$.

Why Q(x; x$^{(t)}$) can be any distribution

$$a_{ij} = \frac{f(\mathbf{x}_j)}{f(\mathbf{x}_i)}\frac{Q(\mathbf{x}_i;\mathbf{x}_j)}{Q(\mathbf{x}_i;\mathbf{x}_j)} = \frac{P(\mathbf{x}_j)}{P(\mathbf{x}_i)}\frac{Q(\mathbf{x}_i;\mathbf{x}_j)}{Q(\mathbf{x}_i;\mathbf{x}_j)} = \frac{1}{a_{ji}}$$

$$\text{accept}(\mathbf{x}_i,\mathbf{x}_j) = \min(1, a_{ij})$$

$$\text{accept}(\mathbf{x}_i,\mathbf{x}_j) < 1 \Rightarrow \text{accept}(\mathbf{x}_j,\mathbf{x}_i) = 1$$

The transition matrix of the Markov chain is defined as

$$P(\mathbf{x}_j|\mathbf{x}_i) = Q(\mathbf{x}_j;\mathbf{x}_i)\text{accept}(\mathbf{x}_i,\mathbf{x}_j)$$

$$P(\mathbf{x}_j|\mathbf{x}_i) = Q(\mathbf{x}_j;\mathbf{x}_i) + \left[1 - \int Q(\mathbf{x}_k;\mathbf{x}_i)\text{accept}(\mathbf{x}_i,\mathbf{x}_k)\right] d\mathbf{x}_k$$

Assume $\text{accept}(\mathbf{x}_i, \mathbf{x}_j) = a_{ij} < 1$. Thus $\text{accept}(\mathbf{x}_j, \mathbf{x}_i) = 1$

$$\text{accept}(\mathbf{x}_i, \mathbf{x}_j)\, P(\mathbf{x}_i)\, Q(\mathbf{x}_j; \mathbf{x}_i) = P(\mathbf{x}_j)\, Q(\mathbf{x}_i; \mathbf{x}_j)\, \text{accept}(\mathbf{x}_j, \mathbf{x}_i)$$

$$P(\mathbf{x}_i)\, P(\mathbf{x}_j \mid \mathbf{x}_i) = P(\mathbf{x}_j)\, P(\mathbf{x}_i \mid \mathbf{x}_j)$$

This obeys detailed balance.

The equilibrium distribution is $P(\mathbf{x}_i)$ regardless of $Q(\mathbf{x}'; \mathbf{x})$

Why $Q(x; x^{(t)})$ still matters

Convergence:
How many steps does it take to reach the equilibrium distribution?

- If Q is positive ($Q(\mathbf{x}'; \mathbf{x}) > 0$ for all $\mathbf{x}'; \mathbf{x}$), the distribution of $\mathbf{x}^{(t)}$ is guaranteed to converge to $P(\mathbf{x})$ in the limit.
- But: convergence is difficult to assess.
- The steps before equilibrium is reached should be ignored (*burn-in)*

Mixing: How fast does the chain move around the state space?

Rejection rate:

- If the step size (distance btw. $\mathbf{x}'$ and $\mathbf{x}^{(t)}$) is large, the rejection probability can be high
- If the step size is too small, only a small region of the sample space may be explored (= slow mixing)

MCMC variants

Metropolis algorithm

$Q(\mathbf{x}'; \mathbf{x})$ is symmetric: $Q(\mathbf{x}'; \mathbf{x}) = Q(\mathbf{x}; \mathbf{x}')$

$$a = \frac{f(\mathbf{x}')}{f(\mathbf{x}^{(t)})} = \frac{P(\mathbf{x}')}{P(\mathbf{x}^{(t)})}$$

Single-component Metropolis-Hastings

- $\mathbf{x}$ is divided into **components** $\mathbf{x}_1 \ldots \mathbf{x}_n$
  Notation: $\mathbf{x}_{-i} := \{\mathbf{x}_1, \ldots, \mathbf{x}_{i-1}, \mathbf{x}_{i+1}, \ldots, \mathbf{x}_n\}$ (all components other than $\mathbf{x}_i$)
- At each iteration t, sample each $\mathbf{x}_i$ in turn, using
  the **full conditional distribution** $P(\mathbf{x}_i \mid \mathbf{x}_{-i}) = P(\mathbf{x}) / \int P(\mathbf{x}) d\mathbf{x}_i$
  and proposal distributions $Q_i(\mathbf{x}_i \mid \mathbf{x}_i^{(t)}, \mathbf{x}_{-i}^{(t)})$ and $Q_i(\mathbf{x}_i^{(t)} \mid \mathbf{x}_i, \mathbf{x}_{-i}^{(t)})$

MCMC: Gibbs sampling

Assumptions:

- $\mathbf{x}$ is a multivariate random variable: $\mathbf{x} = (x_1,...,x_n)$
- The full conditionals $P(x_i \mid x_1,...,x_{i-1}, x_{i+1},...,x_n)$ (=the conditional probabilities of each component given the rest), are easy to evaluate (also true if we split the components of $\mathbf{x}$ into blocks)

Algorithm:

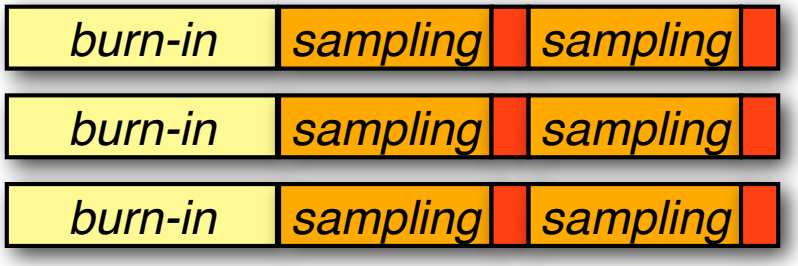
for t = 1...T:
  for i = 1...N:
    sample $x_i^{(t)} \sim P(\mathbf{x}_i \mid \mathbf{x}_1^{(t)},...,\mathbf{x}_{i-1}^{(t)}, \mathbf{x}_{i+1}^{(t-1)},...,\mathbf{x}_n^{(t-1)})$

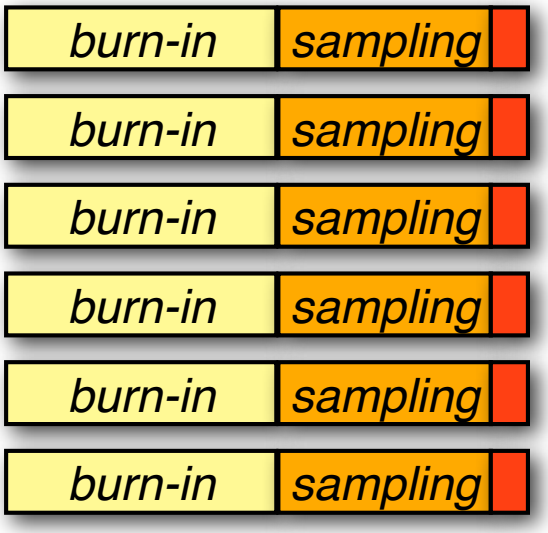
Gibbs sampling is single-component Metropolis-Hastings without rejection. Think of it as using the full conditionals as proposal distributions (so the two fractions cancel, and hence $a = 1$)

$$Q_i(\mathbf{x}_i \mid \mathbf{x}_i^{(t)}, \mathbf{x}_{-i}^{(t)}) = P(\mathbf{x}_i \mid \mathbf{x}_1^{(t)},...,\mathbf{x}_{i-1}^{(t)}, \mathbf{x}_{i+1}^{(t-1)},...,\mathbf{x}_n^{(t-1)})$$

$$Q_i(\mathbf{x}_i^{(t)} \mid \mathbf{x}_i, \mathbf{x}_{-i}^{(t)}) = P(\mathbf{x}_i^{(t)} \mid \mathbf{x}_1^{(t)},...,\mathbf{x}_{i-1}^{(t)}, \mathbf{x}_{i+1}^{(t-1)},...,\mathbf{x}_n^{(t-1)})$$

How should we generate samples?





The longer a chain runs, the more likely it is to have converged.

But it's difficult to know from a *single* chain whether it has converged (or is just slow to mix).

Multiple parallel chains (with independent starting points) can help identify convergence/ mixing problems: do they all generate the same (=indistinguishable) sequences of samples?

Compare within-sequence variance and across-sequence variance.

N.B.: Starting points may come from simpler models.