Stochastic optimization

Markov Chain Monte Carlo

Ethan Fetaya

Weizmann Institute of Science

Assume we have a discrete/non-convex function $f(x)$ we wish to optimize.

Assume we have a discrete/non-convex function $f(x)$ we wish to optimize.

Example: knapsack problem.

Assume we have a discrete/non-convex function $f(x)$ we wish to optimize.

Example: knapsack problem. Given m items with weights $w = (w_1, ..., w_m)$ and values $v = (v_1, ..., v_m)$ Find the subset with maximal value under a weight constraint.

Assume we have a discrete/non-convex function $f(x)$ we wish to optimize.

Example: knapsack problem. Given m items with weights $w = (w_1, ..., w_m)$ and values $v = (v_1, ..., v_m)$ Find the subset with maximal value under a weight constraint.

$$\begin{aligned} &\max v^T z \\ &s.t. \;\; w^T z \leq C \\ &z_i \in \{0, 1\} \end{aligned}$$

Assume we have a discrete/non-convex function $f(x)$ we wish to optimize.

Example: knapsack problem. Given m items with weights $w = (w_1, ..., w_m)$ and values $v = (v_1, ..., v_m)$ Find the subset with maximal value under a weight constraint.

$$\begin{aligned} &\max v^T z \\ &s.t.\ \ w^T z \leq C \\ &z_i \in \{0, 1\} \end{aligned}$$

These problems are in general NP-hard.

Assume we have a discrete/non-convex function $f(x)$ we wish to optimize.

Example: knapsack problem. Given m items with weights $w = (w_1, ..., w_m)$ and values $v = (v_1, ..., v_m)$ Find the subset with maximal value under a weight constraint.

$$\begin{aligned} &\max v^T z \\ &s.t.\ \ w^T z \leq C \\ &z_i \in \{0, 1\} \end{aligned}$$

These problems are in general NP-hard.

For simplicity we will assume the search space X is finite, but our results can be generalized easily.

Stochastic approach - pick items randomly $x_1, ..., x_N$ from your search space X, and return $\arg\max_{i \in [N]} f(x_i)$.

Stochastic approach - pick items randomly $x_1, ..., x_N$ from your search space X, and return $\arg\max_{i \in [N]} f(x_i)$.

What probability distribution should we use?

Stochastic approach - pick items randomly $x_1, ..., x_N$ from your search space X, and return $\arg\max_{i \in [N]} f(x_i)$.

What probability distribution should we use?

Simple (bad) distribution: pick x uniformly from X.

Stochastic approach - pick items randomly $x_1, ..., x_N$ from your search space X, and return $\arg\max_{i \in [N]} f(x_i)$.

What probability distribution should we use?

Simple (bad) distribution: pick x uniformly from X. Problem - we might spend most of the time sampling junk.

Stochastic approach - pick items randomly $x_1, ..., x_N$ from your search space X, and return $\arg\max_{i \in [N]} f(x_i)$.

What probability distribution should we use?

Simple (bad) distribution: pick x uniformly from X. Problem - we might spend most of the time sampling junk.

Great distribution: Softmax $p(x) = e^{f(x)/T}/Z$, where T is a parameter and $Z = \sum_{x \in X} e^{f(x)/T}$ is the partition function.

Stochastic approach - pick items randomly $x_1, ..., x_N$ from your search space X, and return $\arg\max_{i \in [N]} f(x_i)$.

What probability distribution should we use?

Simple (bad) distribution: pick x uniformly from X. Problem - we might spend most of the time sampling junk.

Great distribution: Softmax $p(x) = e^{f(x)/T}/Z$, where T is a parameter and $Z = \sum_{x \in X} e^{f(x)/T}$ is the partition function. Problem - how can you sample from $p(x)$ when you cannot compute Z?

Stochastic approach - pick items randomly $x_1, ..., x_N$ from your search space X, and return $\arg\max_{i \in [N]} f(x_i)$.

What probability distribution should we use?

Simple (bad) distribution: pick x uniformly from X. Problem - we might spend most of the time sampling junk.

Great distribution: Softmax $p(x) = e^{f(x)/T}/Z$, where T is a parameter and $Z = \sum_{x \in X} e^{f(x)/T}$ is the partition function. Problem - how can you sample from $p(x)$ when you cannot compute Z?

To solve this problem we use MCMC (Markov chain Monte carlo) sampling.

Definition 1.1 (Markov chain)

A series of random variables $X_1, ..., X_t, ...,$ is a Markov chain if $P(X_{i+1} = y | X_i, ..., X_1) = P(X_{i+1} = y | X_i)$.

Definition 1.1 (Markov chain)

A series of random variables $X_1, ..., X_t, ...,$ is a Markov chain if $P(X_{i+1} = y | X_i, ..., X_1) = P(X_{i+1} = y | X_i)$.

Example: random walk $X_{i+1} = X_i + \Delta x_i$ where Δx_i are i.i.d is a Markov chain.

Definition 1.1 (Markov chain)

A series of random variables $X_1, ..., X_t, ...,$ is a Markov chain if $P(X_{i+1} = y | X_i, ..., X_1) = P(X_{i+1} = y | X_i)$.

Example: random walk $X_{i+1} = X_i + \Delta x_i$ where Δx_i are i.i.d is a Markov chain.

Example: X_{i+1} is an element of $[N]$ not seem before. This is not a Markov chain.

Definition 1.1 (Markov chain)

A series of random variables $X_1, ..., X_t, ...,$ is a Markov chain if $P(X_{i+1} = y | X_i, ..., X_1) = P(X_{i+1} = y | X_i)$.

Example: random walk $X_{i+1} = X_i + \Delta x_i$ where Δx_i are i.i.d is a Markov chain.

Example: X_{i+1} is an element of $[N]$ not seem before. This is not a Markov chain.

We will consider *homogeneous* Markov chains where $P(X_{i+1} = y | X_i)$ does not depend on i.

We will use matrix notation:

We will use matrix notation: Define distributions as as a row vector π such that $\pi(x)$ is the probability of x.

We will use matrix notation: Define distributions as as a row vector π such that $\pi(x)$ is the probability of x. We can think of a Markov chain as a series $\pi_0, \pi_1, ..., \pi_n, ...$

We will use matrix notation: Define distributions as as a row vector π such that $\pi(x)$ is the probability of x. We can think of a Markov chain as a series $\pi_0, \pi_1, ..., \pi_n, ...$.

Define the transition matrix P such that
$P_{ij} = P_{i \to j} = P(X_{n+1} = j | X_n = i)$.

We will use matrix notation: Define distributions as as a row vector π such that $\pi(x)$ is the probability of x. We can think of a Markov chain as a series $\pi_0, \pi_1, ..., \pi_n, ...$.

Define the transition matrix P such that
$P_{ij} = P_{i \to j} = P(X_{n+1} = j | X_n = i)$.

We then have $\pi_{n+1} = \pi_n P$, and therefore $\pi_n = \pi_0 P^n$.

We will use matrix notation: Define distributions as as a row vector π such that $\pi(x)$ is the probability of x. We can think of a Markov chain as a series $\pi_0, \pi_1, ..., \pi_n, ...$.

Define the transition matrix P such that
$P_{ij} = P_{i \to j} = P(X_{n+1} = j | X_n = i)$.

We then have $\pi_{n+1} = \pi_n P$, and therefore $\pi_n = \pi_0 P^n$.

$$\pi_{n+1}(j) = P(X_{n+1} = j) = \sum_i P(X_{n+1} = j | X_n = i) P(X_n = i)$$

We will use matrix notation: Define distributions as as a row vector π such that $\pi(x)$ is the probability of x. We can think of a Markov chain as a series $\pi_0, \pi_1, ..., \pi_n, ...$.

Define the transition matrix P such that
$P_{ij} = P_{i \to j} = P(X_{n+1} = j | X_n = i)$.

We then have $\pi_{n+1} = \pi_n P$, and therefore $\pi_n = \pi_0 P^n$.

$$\begin{aligned}
\pi_{n+1}(j) =& P(X_{n+1} = j) = \sum_i P(X_{n+1} = j | X_n = i) P(X_n = i) \\
=& \sum_i P_{ij} \pi_n(i) = (\pi_n P)(j).
\end{aligned}$$

For well-behaved Markov chains the nice property holds -

For well-behaved Markov chains the nice property holds - $\pi_n = \pi_0 P^n \to \pi^*$ independent of π_0.

For well-behaved Markov chains the nice property holds - $\pi_n = \pi_0 P^n \to \pi^*$ independent of π_0.

Definition 1.2 (Irreducibility)

A Markov chain is called irreducible if for all i, j there is a k such that $P_{ij}^k > 0$, i.e. you can get to any state from any state.

For well-behaved Markov chains the nice property holds - $\pi_n = \pi_0 P^n \to \pi^*$ independent of π_0.

Definition 1.2 (Irreducibility)

A Markov chain is called irreducible if for all i, j there is a k such that $P_{ij}^k > 0$, i.e. you can get to any state from any state.

Definition 1.3 (Aperiodicity)

A Markov chain is called aperiodical if there exist a k such that $P_{ij}^k > 0$ for all i, j.

For well-behaved Markov chains the nice property holds - $\pi_n = \pi_0 P^n \to \pi^*$ independent of π_0.

Definition 1.2 (Irreducibility)

A Markov chain is called irreducible if for all i, j there is a k such that $P_{ij}^k > 0$, i.e. you can get to any state from any state.

Definition 1.3 (Aperiodicity)

A Markov chain is called aperiodical if there exist a k such that $P_{ij}^k > 0$ for all i, j.

A simple trick to turn a Markov chain aperiodical is to have $P_{ii} > 0$.

Theorem 1.4 (Stationary distribution)

If a Markov chain P is homogeneous, irreducible and aperiodical then for any distribution π_0 we have $\pi_n \to \pi^*$ where π^* is the unique solution to $\pi = \pi P$.

Theorem 1.4 (Stationary distribution)

If a Markov chain P is homogeneous, irreducible and aperiodical then for any distribution π_0 we have $\pi_n \to \pi^*$ where π^* is the unique solution to $\pi = \pi P$.

Proof sketch.

Since P is row-stochastic, $P\mathbb{1} = \mathbb{1}$.

Theorem 1.4 (Stationary distribution)

If a Markov chain P is homogeneous, irreducible and aperiodical then for any distribution π_0 we have $\pi_n \to \pi^*$ where π^* is the unique solution to $\pi = \pi P$.

Proof sketch.

Since P is row-stochastic, $P\mathbb{1} = \mathbb{1}$. Therefore there exits π^* such that $\pi^* P = \pi^*$.

Theorem 1.4 (Stationary distribution)

If a Markov chain P is homogeneous, irreducible and aperiodical then for any distribution π_0 we have $\pi_n \to \pi^*$ where π^* is the unique solution to $\pi = \pi P$.

Proof sketch.

Since P is row-stochastic, $P\mathbb{1} = \mathbb{1}$. Therefore there exits π^* such that $\pi^* P = \pi^*$. From Perron-Frobenius the vector is positive, unique (up to scalar) and each other eigenvalue λ, we have $|\lambda| < 1$.

Theorem 1.4 (Stationary distribution)

If a Markov chain P is homogeneous, irreducible and aperiodical then for any distribution π_0 we have $\pi_n \to \pi^*$ where π^* is the unique solution to $\pi = \pi P$.

Proof sketch.

Since P is row-stochastic, $P\mathbb{1} = \mathbb{1}$. Therefore there exits π^* such that $\pi^* P = \pi^*$. From Perron-Frobenius the vector is positive, unique (up to scalar) and each other eigenvalue λ, we have $|\lambda| < 1$. P may not have a eigen-decomposition but this is enough (with some work) to prove convergence. $\square$

Theorem 1.4 (Stationary distribution)

If a Markov chain P is homogeneous, irreducible and aperiodical then for any distribution π_0 we have $\pi_n \to \pi^*$ where π^* is the unique solution to $\pi = \pi P$.

Proof sketch.

Since P is row-stochastic, $P\mathbb{1} = \mathbb{1}$. Therefore there exits π^* such that $\pi^* P = \pi^*$. From Perron-Frobenius the vector is positive, unique (up to scalar) and each other eigenvalue λ, we have $|\lambda| < 1$. P may not have a eigen-decomposition but this is enough (with some work) to prove convergence. $\square$

How is this helpful?

Theorem 1.4 (Stationary distribution)

If a Markov chain P is homogeneous, irreducible and aperiodical then for any distribution π_0 we have $\pi_n \to \pi^*$ where π^* is the unique solution to $\pi = \pi P$.

Proof sketch.

Since P is row-stochastic, $P\mathbb{1} = \mathbb{1}$. Therefore there exits π^* such that $\pi^* P = \pi^*$. From Perron-Frobenius the vector is positive, unique (up to scalar) and each other eigenvalue λ, we have $|\lambda| < 1$. P may not have a eigen-decomposition but this is enough (with some work) to prove convergence. $\square$

How is this helpful? We will show how to build a Markov chain with any π^*, then sampling from π^* is easy, just go over the chain to convergence (hopefully fast...).

Our interest is in *reversible* Markov chain where detailed balance holds.

Our interest is in *reversible* Markov chain where detailed balance holds.

Lemma 1.5 (detailed balance)

If the detailed balance equation $\pi_i P_{ij} = \pi_j P_{ji}$ *holds then* $\pi = \pi^*$.

Proof - $(\pi P)(j) = \sum_i \pi(i) P_{ij} = \sum_i \pi(j) P_{ji} = \pi(j)$.

Our interest is in *reversible* Markov chain where detailed balance holds.

Lemma 1.5 (detailed balance)

If the detailed balance equation $\pi_i P_{ij} = \pi_j P_{ji}$ *holds then* $\pi = \pi^*$.

Proof - $(\pi P)(j) = \sum_i \pi(i) P_{ij} = \sum_i \pi(j) P_{ji} = \pi(j)$.

So in order to have π steady state we need $\frac{P_{ij}}{P_{ji}} = \frac{\pi_j}{\pi_i}$

Our interest is in *reversible* Markov chain where detailed balance holds.

Lemma 1.5 (detailed balance)

If the detailed balance equation $\pi_i P_{ij} = \pi_j P_{ji}$ *holds then* $\pi = \pi^*$.

Proof - $(\pi P)(j) = \sum_i \pi(i) P_{ij} = \sum_i \pi(j) P_{ji} = \pi(j)$.

So in order to have π steady state we need $\frac{P_{ij}}{P_{ji}} = \frac{\pi_j}{\pi_i}$

One can show that there exists a symmetric postive matrix A, such that P is A after row-normalization.

How fast does a Markov chain converge? There is a huge literature on mixing time, we will state one simple result.

How fast does a Markov chain converge? There is a huge literature on mixing time, we will state one simple result.

The mixing time $t_{mix}(\epsilon)$ is the minimal time such that no mater where we started, for $n \geq t_{mix}(\epsilon)$ we have $||\pi_n - \pi^*||_{TV} = ||\pi_n - \pi^*||_1 \leq \epsilon$

How fast does a Markov chain converge? There is a huge literature on mixing time, we will state one simple result.

The mixing time $t_{mix}(\epsilon)$ is the minimal time such that no mater where we started, for $n \geq t_{mix}(\epsilon)$ we have $||\pi_n - \pi^*||_{TV} = ||\pi_n - \pi^*||_1 \leq \epsilon$

If P is reversibel it has an eigen-decomposition with $1 = \lambda_1 > \lambda_2 \geq ... \geq \lambda_{|X|} > -1$. Define $\lambda_* = \max\{\lambda_2, |\lambda_{|X|}|\}$.

How fast does a Markov chain converge? There is a huge literature on mixing time, we will state one simple result.

The mixing time $t_{mix}(\epsilon)$ is the minimal time such that no mater where we started, for $n \geq t_{mix}(\epsilon)$ we have $||\pi_n - \pi^*||_{TV} = ||\pi_n - \pi^*||_1 \leq \epsilon$

If P is reversibel it has an eigen-decomposition with $1 = \lambda_1 > \lambda_2 \geq ... \geq \lambda_{|X|} > -1$. Define $\lambda_* = \max\{\lambda_2, |\lambda_{|X|}|\}$.

Theorem 1.6 (Mixing time)

If a Markov chain P has all previous requirements and is reversible then

$$t_{mix}(\epsilon) \leq \log\left(\frac{1}{\epsilon \min_i \pi^*(i)}\right) \frac{1}{1-\lambda_*}$$

$$t_{mix}(\epsilon) \geq \log\left(\frac{1}{2\epsilon}\right) \frac{\lambda_*}{1-\lambda_*}$$

Theorem 1.6 (Mixing time)

If a Markov chain P has all previous requirements and is reversible then

$$t_{mix}(\epsilon) \leq \log\left(\frac{1}{\epsilon \min_i \pi^*(i)}\right)\frac{1}{1-\lambda_*}$$

$$t_{mix}(\epsilon) \geq \log\left(\frac{1}{2\epsilon}\right)\frac{\lambda_*}{1-\lambda_*}$$

This shows that the *spectral gap* controls the rate of convergence.


EIN FILM VON
FRITZ LANG
METROPOLIS
IN DEN HAUPTROLLEN
BRIGITTE HELM UND GUSTAV FRÖHLICH
ALFRED ABEL
THEODOR LOOS
HEINRICH GEORGE
FRITZ RASP
RUDOLF KLEIN-ROGGE
MANUSKRIPT THEA VONHARBOU MUSIK GOTTFRIED HUPPERTZ
AN DER KAMERA KARL FREUND UND GÜNTHER RITTAU

The Metropolis-Hastings algorithms allows us to build a Markov chain with a desired stationary distribution. The algorithm requires:

The Metropolis-Hastings algorithms allows us to build a Markov chain with a desired stationary distribution. The algorithm requires:

1) A desired distribution known up to a constant, e.g.
$\pi(x) = \exp(f(x)/T)/Z$.

The Metropolis-Hastings algorithms allows us to build a Markov chain with a desired stationary distribution. The algorithm requires:

1) A desired distribution known up to a constant, e.g. $\pi(x) = \exp(f(x)/T)/Z$.

2) A Markov chain $Q(i \to j)$ called the *proposal distribution*. This is where we should look around state i.

The Metropolis-Hastings algorithms allows us to build a Markov chain with a desired stationary distribution. The algorithm requires:

1) A desired distribution known up to a constant, e.g. $\pi(x) = \exp(f(x)/T)/Z$.

2) A Markov chain $Q(i \to j)$ called the *proposal distribution*. This is where we should look around state i. For example in the knapsack problem it could we uniform over all possibilities of switching a single element Z_k.

The Metropolis-Hastings algorithms allows us to build a Markov chain with a desired stationary distribution. The algorithm requires:

1) A desired distribution known up to a constant, e.g. $\pi(x) = \exp(f(x)/T)/Z$.

2) A Markov chain $Q(i \to j)$ called the *proposal distribution*. This is where we should look around state i. For example in the knapsack problem it could we uniform over all possibilities of switching a single element Z_k. For continuous state spaces $Q(x_0 \to x) = \mathcal{N}(x_0, \sigma I)$ is a common choice.

Algorithm `Metropolis-Hastings`

Input: x_0, π and Q.
for $i = 0 : N$ **do**
  Pick proposition x_* from distribution $Q(x_i \to \cdot)$
  $\alpha = \min\{1, \frac{\pi(x_*)Q(x_* \to x_i)}{\pi(x_i)Q(x_i \to x_*)}\}$
  With probability α set $x_{i+1} = x_*$, else $x_{i+1} = x_i$
end for

Algorithm `Metropolis-Hastings`

Input: x_0, π and Q.
for $i = 0 : N$ **do**
  Pick proposition x_* from distribution $Q(x_i \to \cdot)$
  $\alpha = \min\{1, \frac{\pi(x_*)Q(x_* \to x_i)}{\pi(x_i)Q(x_i \to x_*)}\}$
  With probability α set $x_{i+1} = x_*$, else $x_{i+1} = x_i$
end for

Notice we only ratio of π so the unknown constant is eliminated.

Algorithm `Metropolis-Hastings`

Input: x_0, π and Q.
for $i = 0 : N$ **do**
  Pick proposition x_* from distribution $Q(x_i \to \cdot)$
  $\alpha = \min\{1, \frac{\pi(x_*)Q(x_* \to x_i)}{\pi(x_i)Q(x_i \to x_*)}\}$
  With probability α set $x_{i+1} = x_*$, else $x_{i+1} = x_i$
end for

Notice we only ratio of π so the unknown constant is eliminated.

For example if Q is symmetric and $\pi \propto \exp(f(x)/T)$ then if $f(x_*) \geq f(x_i)$ we always move to x_*,

Algorithm `Metropolis-Hastings`

Input: x_0, π and Q.
for $i = 0 : N$ **do**
  Pick proposition x_* from distribution $Q(x_i \to \cdot)$
  $\alpha = \min\{1, \frac{\pi(x_*)Q(x_* \to x_i)}{\pi(x_i)Q(x_i \to x_*)}\}$
  With probability α set $x_{i+1} = x_*$, else $x_{i+1} = x_i$
end for

Notice we only ratio of π so the unknown constant is eliminated.

For example if Q is symmetric and $\pi \propto \exp(f(x)/T)$ then if $f(x_*) \geq f(x_i)$ we always move to x_*, else we move with probability $exp(-|\Delta f|/T)$

Theorem 2.1

Theorem 2.1

The MH algorithm defines a Markov chain P with stationary distribution $\pi^* = \pi$.

Theorem 2.1

The MH algorithm defines a Markov chain P with stationary distribution $\pi^* = \pi$.

Proof.

We will show P has detailed balance:

Theorem 2.1

The MH algorithm defines a Markov chain P with stationary distribution $\pi^* = \pi$.

Proof.

We will show P has detailed balance: Assume w.l.o.g $\pi_j Q_{j\to i} \leq \pi_i Q_{i\to j}$.

$$\pi_i P_{i\to j} = \pi_i Q_{i\to j} \min\{1, \frac{\pi_j Q_{j\to i}}{\pi_i Q_{i\to j}}\}$$

□

Theorem 2.1

The MH algorithm defines a Markov chain P with stationary distribution $\pi^* = \pi$.

Proof.

We will show P has detailed balance: Assume w.l.o.g $\pi_j Q_{j\to i} \leq \pi_i Q_{i\to j}$.

$$\pi_i P_{i\to j} = \pi_i Q_{i\to j} \min\{1, \frac{\pi_j Q_{j\to i}}{\pi_i Q_{i\to j}}\} = \pi_i Q_{i\to j} \frac{\pi_j Q_{j\to i}}{\pi_i Q_{i\to j}}$$

□

Theorem 2.1

The MH algorithm defines a Markov chain P with stationary distribution $\pi^* = \pi$.

Proof.

We will show P has detailed balance: Assume w.l.o.g $\pi_j Q_{j\to i} \leq \pi_i Q_{i\to j}$.

$$\begin{aligned}\pi_i P_{i\to j} =& \pi_i Q_{i\to j} \min\{1, \frac{\pi_j Q_{j\to i}}{\pi_i Q_{i\to j}}\} = \pi_i Q_{i\to j} \frac{\pi_j Q_{j\to i}}{\pi_i Q_{i\to j}} \\ =& \pi_j Q_{j\to i} = \pi_j Q_{j\to i} \min\{1, \frac{\pi_i Q_{i\to j}}{\pi_j Q_{j\to i}}\} = \pi_j P_{j\to i}\end{aligned}$$

□

Remarks:

Remarks:

- Q must be irreducible!

Remarks:

- Q must be irreducible!
- The convergence rate depends heavily on the auxiliary distribution Q.

Remarks:

- Q must be irreducible!
- The convergence rate depends heavily on the auxiliary distribution Q.
- The algorithm is derivative-free.

Remarks:

- Q must be irreducible!
- The convergence rate depends heavily on the auxiliary distribution Q.
- The algorithm is derivative-free.
- Convergence can be exponentially slow.

Remarks:

- Q must be irreducible!
- The convergence rate depends heavily on the auxiliary distribution Q.
- The algorithm is derivative-free.
- Convergence can be exponentially slow.
- Can have low complexity per iteration, depends on Q.

Remarks:

- Q must be irreducible!
- The convergence rate depends heavily on the auxiliary distribution Q.
- The algorithm is derivative-free.
- Convergence can be exponentially slow.
- Can have low complexity per iteration, depends on Q.
- π can be known up to a constant.

Remarks:

- Q must be irreducible!
- The convergence rate depends heavily on the auxiliary distribution Q.
- The algorithm is derivative-free.
- Convergence can be exponentially slow.
- Can have low complexity per iteration, depends on Q.
- π can be known up to a constant.
- Optimization is just one application of the MH algorithm.

Consider running MH with $\pi \propto \exp(f(x)/T)$. What value of T to use?

Consider running MH with $\pi \propto \exp(f(x)/T)$. What value of T to use?

For large T - rapid mixing time, but π^* is almost uniform.

Consider running MH with $\pi \propto \exp(f(x)/T)$. What value of T to use?

For large T - rapid mixing time, but π^* is almost uniform.

For small T - π^* is highly concentrated on the maximum, but there can be (exponentially) long mixing time.

Consider running MH with $\pi \propto \exp(f(x)/T)$. What value of T to use?

For large T - rapid mixing time, but π^* is almost uniform.

For small T - π^* is highly concentrated on the maximum, but there can be (exponentially) long mixing time.

The idea behind *simulated annealing* - start with high T, then decrease it slowly over time.

Consider running MH with $\pi \propto \exp(f(x)/T)$. What value of T to use?

For large T - rapid mixing time, but π^* is almost uniform.

For small T - π^* is highly concentrated on the maximum, but there can be (exponentially) long mixing time.

The idea behind *simulated annealing* - start with high T, then decrease it slowly over time.

While simulated annealing is not a homogeneous process, if T changes slow enough it is a close approximation.

While simulated annealing is not a homogeneous process, if T changes slow enough it is a close approximation.

One can show that for finite/compact spaces simulated annealing with $T_i = \frac{1}{C \ln(T_0+i)}$ converges to the global optimum.

Online demo - `http://www.youtube.com/watch?v=iaq_Fpr4KZc`

Counter-example: On the blackboard.

If we are at a local maxima or a high probability state, we might reject any proposal with high probability. This is very wasteful.

If we are at a local maxima or a high probability state, we might reject any proposal with high probability. This is very wasteful.

The idea - sample directly the next *accepted* state.

If we are at a local maxima or a high probability state, we might reject any proposal with high probability. This is very wasteful.

The idea - sample directly the next *accepted* state.

This only works for discrete problems such that $Q(x_0 \to x)$ has a reasonable size support.

Define $w(x) = Q(x_0 \to x) \cdot \min\{1, \frac{\pi(x)Q(x \to x_{i-1})}{\pi(x_{i-1})Q(x_{i-1} \to x)}\}$ the probability to chose and accept x.

Define $w(x) = Q(x_0 \to x) \cdot \min\{1, \frac{\pi(x)Q(x \to x_{i-1})}{\pi(x_{i-1})Q(x_{i-1} \to x)}\}$ the probability to chose and accept x.

Define $W = \sum_x w(x)$. This is computable if the support of $Q(x_0 \to \cdot)$ is small and simple.

Define $w(x) = Q(x_0 \to x) \cdot \min\{1, \frac{\pi(x)Q(x \to x_{i-1})}{\pi(x_{i-1})Q(x_{i-1} \to x)}\}$ the probability to chose and accept x.

Define $W = \sum_x w(x)$. This is computable if the support of $Q(x_0 \to \cdot)$ is small and simple.

The probability that x is the next accepted state in the MH run is $w(x)/W$. Use this to pick the next state instead of the regular iteration.

Algorithm `Rejectionless-MH`

Input: x_0, π and Q.
for $i = 0 : N$ **do**
  For $x \in supp(Q(x_i \to \cdot))$ compute $w(x)$,
  $w(x) = Q(x_0 \to x) \cdot \min\{1, \frac{\pi(x)Q(x \to x_{i-1})}{\pi(x_{i-1})Q(x_{i-1} \to x)}\}$
  $W = \sum_{x \in supp(Q(x_i,:))} w(x)$
  Select x_{i+1} with probability $w(x)/W$
end for

Algorithm `Rejectionless-MH`

Input: x_0, π and Q.
for $i = 0 : N$ **do**
  For $x \in supp(Q(x_i \to \cdot))$ compute $w(x)$,
  $w(x) = Q(x_0 \to x) \cdot \min\{1, \frac{\pi(x)Q(x \to x_{i-1})}{\pi(x_{i-1})Q(x_{i-1} \to x)}\}$
  $W = \sum_{x \in supp(Q(x_i,:))} w(x)$
  Select x_{i+1} with probability $w(x)/W$
end for

This can be much slower per iteration, but worth it if W is low enough.