LECTURE 3: PROBABILISTIC LEARNING AND SAMPLING METHODS

COMPUTATIONAL INTELLIGENCE

VRIJE UNIVERSITEIT AMSTERDAM

PROBABILISTIC LEARNING

TOSS A COIN… 🎶

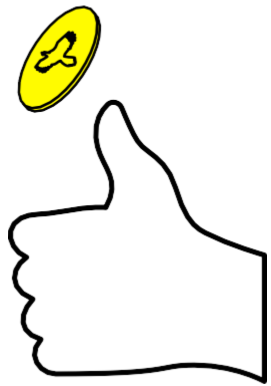
- $c \in \{0,1\}$ - a **random variable** (a result of tossing a coin)

TOSS A COIN… 🎶

- $c \in \{0,1\}$ - a **random variable** (a result of tossing a coin)
- $x = p(c = 1)$ - **probability** of observing *head*
- $1 - x = p(c = 0)$ - probability of observing *tail*
- $p(c|x) = x^c \, (1 - x)^{1-c}$ - **Bernoulli distribution**

TOSS A COIN… 🎶

- $c \in \{0,1\}$ - a **random variable** (a result of tossing a coin)
- $x = p(c = 1)$ - **probability** of observing *head*
- $1 - x = p(c = 0)$ - probability of observing *tail*
- $p(c|x) = x^c \,(1 - x)^{1-c}$ - **Bernoulli distribution**



Quick check:

$p(c = 0|x) = x^0 \,(1 - x)^{1-0} = 1 - x$

$p(c = 1|x) = x^1 \,(1 - x)^{1-1} = x$

TOSS A COIN... 🎶

- $c \in \{0,1\}$ - a **random variable** (a result of tossing a coin)
- $x = p(c = 1)$ - **probability** of observing *head*
- $1 - x = p(c = 0)$ - probability of observing *tail*
- $p(c|x) = x^{c} \, (1 - x)^{1-c}$ - **Bernoulli distribution**
- $\mathcal{D} = \{c_1, c_2, \dots, c_N\}$ - *iid* observations (**data**)

TOSS A COIN… 🎶

- $c \in \{0,1\}$ - a **random variable** (a result of tossing a coin)
- $x = p(c = 1)$ - **probability** of observing *head*
- $1 - x = p(c = 0)$ - probability of observing *tail*
- $p(c|x) = x^c \ (1 - x)^{1-c}$ - **Bernoulli distribution**
- $\mathcal{D} = \{c_1, c_2, \ldots, c_N\}$ - *iid* observations (**data**)

EXAMPLE:

$\mathcal{D} = \{0,0,1,1,0,1,1\}$

TOSS A COIN... 🎶

- $c \in \{0,1\}$ - a **random variable** (a result of tossing a coin)
- $x = p(c = 1)$ - **probability** of observing *head*
- $1 - x = p(c = 0)$ - probability of observing *tail*
- $p(c|x) = x^c \ (1 - x)^{1-c}$ - **Bernoulli distribution**
- $\mathcal{D} = \{c_1, c_2, \dots, c_N\}$ - *iid* observations (**data**)
- The **likelihood function**:

$$p(\mathcal{D}|x) = \prod_{n=1}^{N} p(c_n|x)$$

TOSS A COIN… 🎶

The optimization problem:

TOSS A COIN… 🎶

The optimization problem:

Find such $x \in [0,1]$ that minimizes the **negative log-likelihood function**:

$$\min_{x \in [0,1]} -\log p(\mathcal{D} | x)$$

TOSS A COIN... 🎶

The optimization problem:

Find such $x \in [0,1]$ that minimizes the **negative log-likelihood function**:

$$\min_{x \in [0,1]} -\log p(\mathcal{D} | x)$$

Remarks:

TOSS A COIN... 🎶

The optimization problem:

Find such $x \in [0,1]$ that minimizes the **negative log-likelihood function**:

$$\min_{x\in[0,1]} \; - \; \log p(\mathcal{D} | x)$$

Remarks:

1) Why **negative**? Because: $\max f(x) = \min \{-f(x)\}$.

TOSS A COIN… 🎶

The optimization problem:

Find such $x \in [0,1]$ that minimizes the **negative log-likelihood function**:

$$\min_{x \in [0,1]} - \log p(\mathcal{D} | x)$$

Remarks:

1) Why **negative**? Because: $\max f(x) = \min \{-f(x)\}$.

2) Why **logarithm**? Because: $\log \prod = \sum \log$ and optimum is the same.

TOSS A COIN… 🎶

$$\log p(\mathcal{D}|x) = \log \prod_{n=1}^{N} p(c_n|x)$$

the log-likelihood

TOSS A COIN... 🎶

$$\log p(\mathcal{D} \mid x) = \log \prod_{n=1}^{N} p(c_n \mid x)$$

the log-likelihood

$$= \sum_{n=1}^{N} \log p(c_n \mid x)$$

$$\log \prod = \sum \log$$

TOSS A COIN… 🎶

$$\log p(\mathcal{D} \mid x) = \log \prod_{n=1}^{N} p(c_n \mid x)$$

the log-likelihood

$$= \sum_{n=1}^{N} \log p(c_n \mid x)$$

$\log \prod = \sum \log$

$$= \sum_{n=1}^{N} \log x^{c_n} (1 - x)^{1-c_n}$$

Bernoulli distribution

TOSS A COIN... 🎶

$$\log p(\mathcal{D} \mid x) = \log \prod_{n=1}^{N} p(c_n \mid x)$$

the log-likelihood

$$= \sum_{n=1}^{N} \log p(c_n \mid x)$$

$\log \prod = \sum \log$

$$= \sum_{n=1}^{N} \log x^{c_n} (1-x)^{1-c_n}$$

Bernoulli distribution

$$= \sum_{n=1}^{N} (c_n \log x + (1-c_n) \log(1-x))$$

$\log a^b = b \log a$

$\log ab = \log a + \log b$

TOSS A COIN... 🎶

(Finding $x^\star$) Calculate derivative wrt and set to 0: $\frac{\mathrm{d}}{\mathrm{d}x} f(x) = 0$ gives optimum

TOSS A COIN... 🎶

(Finding $x^\star$) Calculate derivative wrt and set to 0:

$$\frac{\mathrm{d}}{\mathrm{d}x} f(x) = 0 \text{ gives optimum}$$

$$\frac{\mathrm{d}}{\mathrm{d}x} \sum_{n=1}^{N} (c_n \log x + (1 - c_n)\log(1 - x)) = 0$$

TOSS A COIN... 🎶

(Finding x^*) Calculate derivative wrt   and set to 0:

$$\frac{\mathrm{d}}{\mathrm{d}x} f(x) = 0 \text{ gives optimum}$$

$$\frac{\mathrm{d}}{\mathrm{d}x} \sum_{n=1}^{N} (c_n \log x + (1 - c_n)\log(1 - x)) = 0$$

$$\sum_{n=1}^{N} \left(\frac{c_n}{x} - \frac{(1 - c_n)}{(1 - x)} \right) = 0$$

$$\frac{\mathrm{d}}{\mathrm{d}x} \log x = \frac{1}{x}$$

TOSS A COIN… 🎶

(Finding x^{*}) Calculate derivative wrt and set to 0:

$$\frac{\mathrm{d}}{\mathrm{d}x} f(x) = 0 \text{ gives optimum}$$

$$\frac{\mathrm{d}}{\mathrm{d}x} \sum_{n=1}^{N} \left(c_n \log x + (1 - c_n) \log(1 - x) \right) = 0$$

$$\sum_{n=1}^{N} \left(\frac{c_n}{x} - \frac{(1 - c_n)}{(1 - x)} \right) = 0$$

$$\frac{\mathrm{d}}{\mathrm{d}x} \log x = \frac{1}{x}$$

$$\sum_{n=1}^{N} \left(c_n (1 - x) - (1 - c_n) x \right) = 0$$

TOSS A COIN... 🎶

(Finding x^*) Calculate derivative wrt   and set to 0:

$$\frac{\mathrm{d}}{\mathrm{d}x} f(x) = 0 \text{ gives optimum}$$

$$\frac{\mathrm{d}}{\mathrm{d}x} \sum_{n=1}^{N} \left(c_n \log x + (1 - c_n)\log(1 - x)\right) = 0$$

$$\sum_{n=1}^{N} \left(\frac{c_n}{x} - \frac{(1 - c_n)}{(1 - x)} \right) = 0$$

$$\frac{\mathrm{d}}{\mathrm{d}x} \log x = \frac{1}{x}$$

$$\sum_{n=1}^{N} \left(c_n(1 - x) - (1 - c_n)x\right) = 0$$

$$\sum_{n=1}^{N} c_n - x \sum_{n=1}^{N} c_n - Nx + x \sum_{n=1}^{N} c_n = 0 \quad \Rightarrow \quad \boxed{x = \frac{1}{N} \sum_{n=1}^{N} c_n}$$

TOSS A COIN... 🎶

(Finding $x^\star$) Calculate derivative wrt and set to 0:

$$\frac{\mathrm{d}}{\mathrm{d}x} \sum_{n=1}^{N} \left(c_n \log x + (1 - c_n)\log(1 - x)\right) = 0$$

$$\sum_{n=1}^{N} \left(\frac{c_n}{x} - \frac{(1 - c_n)}{(1 - x)} \right) = 0$$

$$\sum_{n=1}^{N} \left(c_n(1 - x) - (1 - c_n)x\right) = 0$$

$$\sum_{n=1}^{N} c_n - x \sum_{n=1}^{N} c_n - Nx + x \sum_{n=1}^{N} c_n = 0 \quad \Rightarrow \quad \boxed{x = \frac{1}{N} \sum_{n=1}^{N} c_n}$$

$\frac{\mathrm{d}}{\mathrm{d}x} f(x) = 0$ gives optimum

$$\frac{\mathrm{d}}{\mathrm{d}x} \log x = \frac{1}{x}$$

EXAMPLE:

$\mathcal{D} = \{0,0,1,1,0,1,1\}$

$x^\star = 4/7$

PROBABILISTIC LEARNING (LIKELIHOOD-BASED)

PROBABILISTIC LEARNING (LIKELIHOOD-BASED)

1) Determine $p(y|x)$.

PROBABILISTIC LEARNING (LIKELIHOOD-BASED)

1) Determine $p(y|x)$.

2) Determine $p(\mathcal{D}|x)$.

PROBABILISTIC LEARNING (LIKELIHOOD-BASED)

1) Determine $p(y|x)$.

2) Determine $p(\mathcal{D}|x)$.

3) Check constraints.

PROBABILISTIC LEARNING (LIKELIHOOD-BASED)

1) Determine $p(y|x)$.

2) Determine $p(\mathcal{D}|x)$.

3) Check constraints.

4) Find the best solution by minimizing $-\log p(\mathcal{D}|x)$.

PROBABILISTIC LEARNING (LIKELIHOOD-BASED)

1) Determine $p(y|x)$. — e.g., Bernoulli, Gaussian…

2) Determine $p(\mathcal{D}|x)$. — e.g., *iid* or sequential

3) Check constraints. — e.g., only values between [0, 1]

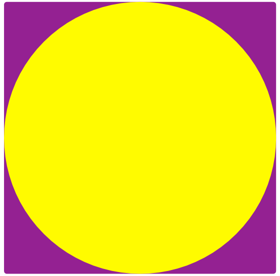
4) Find the best solution by minimizing $-\log p(\mathcal{D}|x)$. — e.g., using gradient-descent or analytically or DFM

SAMPLING METHODS

HOW TO ESTIMATE π?

HOW TO ESTIMATE π?

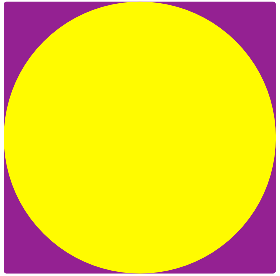
- We know that the circle area is $A_{\bigcirc} = \pi r^2$.
- We know that the square area is $A_{\square} = (2r)^2$.



HOW TO ESTIMATE π?

- We know that the circle area is $A_{\bigcirc} = \pi r^2$.
- We know that the square area is $A_{\square} = (2r)^2$.
- The ratio:

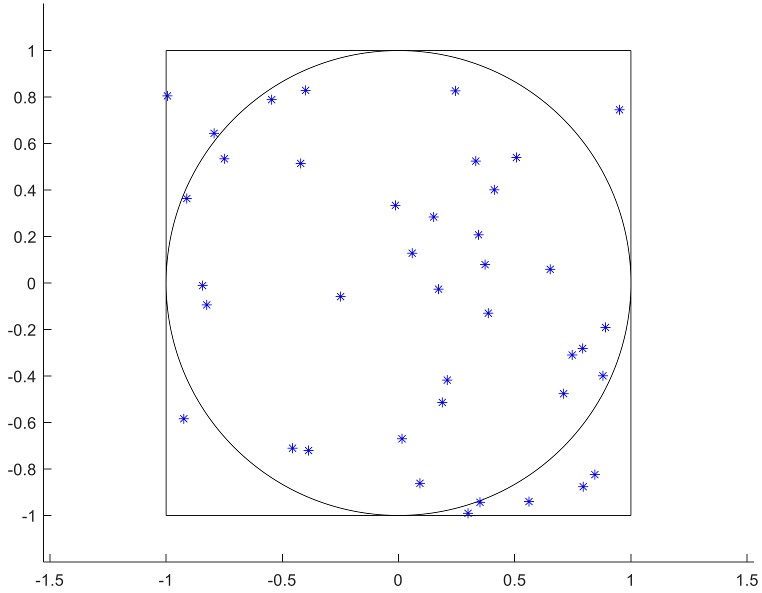
$$\frac{A_{\bigcirc}}{A_{\square}} = \frac{\pi}{4} \Rightarrow \qquad \pi = 4\,\frac{A_{\bigcirc}}{A_{\square}}$$



HOW TO ESTIMATE π?

- We know that the circle area is $A_{\bigcirc} = \pi r^2$.
- We know that the square area is $A_{\square} = (2r)^2$.
- The ratio:

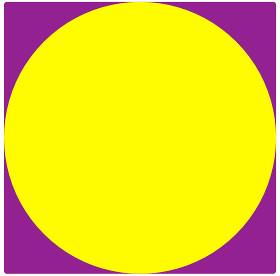
$$\frac{A_{\bigcirc}}{A_{\square}} = \frac{\pi}{4} \Rightarrow \qquad \pi = 4\,\frac{A_{\bigcirc}}{A_{\square}}$$



HOW TO ESTIMATE π?

- We know that the circle area is $A_{\bigcirc} = \pi r^2$.
- We know that the square area is $A_{\square} = (2r)^2$.
- The ratio:

$$\frac{A_{\bigcirc}}{A_{\square}} = \frac{\pi}{4} \Rightarrow \quad \pi = 4\frac{A_{\bigcirc}}{A_{\square}}$$



- We can express it as follows:

$$\pi = 4\mathbb{E}_{(x,y)\sim\mathrm{Unif}[-r,r]}\left[\mathrm{I}[x^2 + y^2 \leq r^2]\right]$$

HOW TO ESTIMATE π?

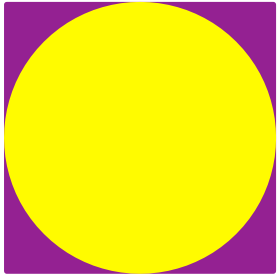
- We know that the circle area is $A_{\bigcirc} = \pi r^2$.
- We know that the square area is $A_{\square} = (2r)^2$.
- The ratio:

$$\frac{A_{\bigcirc}}{A_{\square}} = \frac{\pi}{4} \Rightarrow \quad \pi = 4\,\frac{A_{\bigcirc}}{A_{\square}}$$

- We can express it as follows:

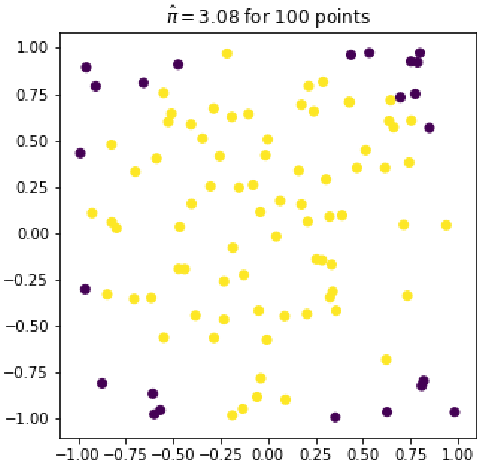
$$\pi = 4\mathbb{E}_{(x,y)\sim\mathrm{Unif}[-r,r]}\Big[\mathrm{I}[x^2 + y^2 \le r^2]\Big]$$

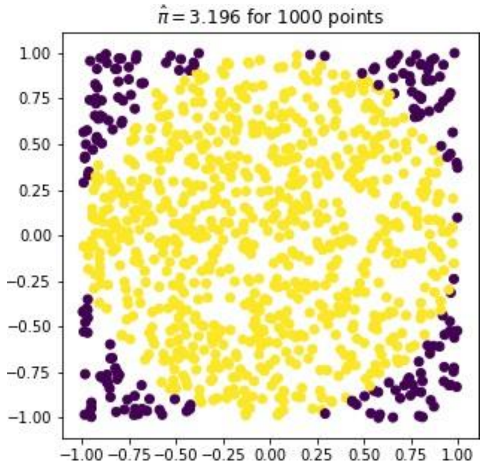
$$\approx 4\frac{1}{N}\sum_{n=1}^{N}\mathrm{I}[x_n^2 + y_n^2 \le r^2]$$

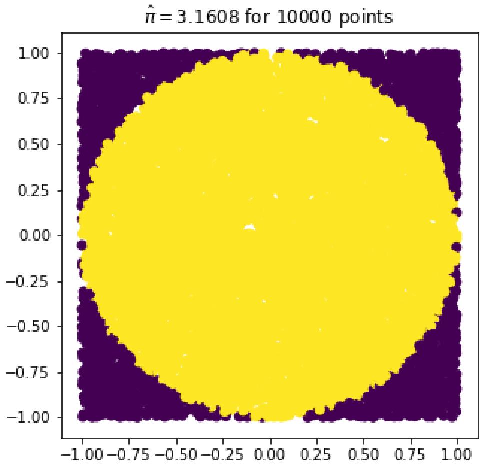


Approximate it by sampling!

HOW TO ESTIMATE π?







MONTE CARLO METHODS

- History: **Stanisław Ulam** played solitaire and asked himself what are the chances that a particular laid out with 52 cards is successful.

Stanisław Ulam (1909-1984) was a Polish mathematician, a participant of the Manhattan project, known for applications of mathematics to physics, biology and computer science, and first formulations of Monte Carlo methods and cellular automata (with John von Neumann).

MONTE CARLO METHODS

- History: **Stanisław Ulam** played solitaire and asked himself what are the chances that a particular laid out with 52 cards is successful.
- The underlying idea: **Approximate by sampling**!

$$\frac{1}{N}\sum_{n} f(x_n) \overset{n\to\infty}{\longrightarrow} \int f(x)\, p(x)\, \mathrm{d}x$$

MONTE CARLO METHODS

- History: **Stanisław Ulam** played solitaire and asked himself what are the chances that a particular laid out with 52 cards is successful.
- The underlying idea: **Approximate by sampling**!

$$\frac{1}{N}\sum_{n} f(x_n) \overset{n\to\infty}{\longrightarrow} \int f(x)\, p(x)\, \mathrm{d}x$$

- General applications:

  A. **Analytically infeasible quantities** (e.g., normalization constants, statistical inference, expectations of some function of interest).

  B. **Optimization**.

MONTE CARLO METHODS

- Problem with sampling (***curse of dimensionality***):

MONTE CARLO METHODS

- Problem with sampling (***curse of dimensionality***):

A grid on a unit interval (dist. adjacent points = 0.01) ➞ 10^2 points.

MONTE CARLO METHODS

- Problem with sampling (***curse of dimensionality***):

A grid on a unit interval (dist. adjacent points = 0.01) ➞ 10^2 points.

Now, we do the same but in 2D ➞ 10^4 points.

MONTE CARLO METHODS

- Problem with sampling (***curse of dimensionality***):

A grid on a unit interval (dist. adjacent points = 0.01) ➞ 10^2 points.

Now, we do the same but in 2D ➞ 10^4 points.

And in 10D ➞ 10^{20} points.

- Even for 10 dimensions the number of possible states is infeasible…

MONTE CARLO METHODS

- Problem with sampling (***curse of dimensionality***):

A grid on a unit interval (dist. adjacent points = 0.01) ➞ 10^2 points.

Now, we do the same but in 2D ➞ 10^4 points.

And in 10D ➞ 10^{20} points.

- Even for 10 dimensions the number of possible states is infeasible…
- A natural question is:

Can we do better than independent sampling?

MARKOV CHAIN MONTE CARLO

- **Idea**: Make a new sample dependent on the past (**Markov chain**).

MARKOV CHAIN MONTE CARLO

- **Idea**: Make a new sample dependent on the past (**Markov chain**).
- In other words, we introduce a **proposal distribution** $q(x_t | x_{t-1})$ to obtain a **chain** that corresponds to a sample from the original distribution $p(x)$.
- Could any distribution be used as a proposal distribution?

MARKOV CHAIN MONTE CARLO

- **Idea**: Make a new sample dependent on the past (**Markov chain**).
- In other words, we introduce a **proposal distribution** $q(x_t | x_{t-1})$ to obtain a **chain** that corresponds to a sample from the original distribution $p(x)$.
- Could any distribution be used as a proposal distribution? **NO**.

MARKOV CHAIN MONTE CARLO

- **Idea**: Make a new sample dependent on the past (**Markov chain**).
- In other words, we introduce a **proposal distribution** $q(x_t | x_{t-1})$ to obtain a **chain** that corresponds to a sample from the original distribution $p(x)$.
- Could any distribution be used as a proposal distribution? **NO**.
- Conditions on the proposal distribution:

  A. (**Irreducibility**) There is a positive probability of visiting all states.

  B. (**Aperiodicity**) The chain should not get trapped in cycles.

METROPOLIS-HASTINGS ALGORITHM

1. Initialize $x_t := x_0$.
2. For $t \in \{0,1,\ldots,T-1\}$:

   (i) (**Generate**) Sample $x' \sim q(x|x_t)$.

   (ii) (**Evaluate**) Calculate acceptance probability:

$$A(x', x_t) = \min\left\{1, \frac{p(x')\, q(x_t|x')}{p(x_t)\, q(x'|x_t)}\right\}$$

   (iii) (**Select**) Sample $u \sim \mathrm{Unif}[0,1]$.

   If $A(x', x_t) > u$, then $x_{t+1} := x'$.

   Else $x_{t+1} := x_t$.

METROPOLIS-HASTINGS ALGORITHM

1. Initialize $x_t := x_0$.
2. For $t \in \{0,1,\ldots,T-1\}$:
   (i) (**Generate**) Sample $x' \sim q(x|x_t)$. #e.g. $q(x|x_t) = \mathcal{N}(x|x_t, \sigma^2)$
   (ii) (**Evaluate**) Calculate acceptance probability:
   $$A(x', x_t) = \min\left\{1, \frac{p(x')\, q(x_t|x')}{p(x_t)\, q(x'|x_t)}\right\}$$
   (iii) (**Select**) Sample $u \sim \mathrm{Unif}[0,1]$.
   If $A(x', x_t) > u$, then $x_{t+1} := x'$.
   Else $x_{t+1} := x_t$.

METROPOLIS-HASTINGS ALGORITHM

1. Initialize $x_t := x_0$.
2. For $t \in \{0,1,\ldots,T-1\}$:
   - (i) (**Generate**) Sample $x' \sim q(x|x_t)$. #e.g. $q(x|x_t) = \mathcal{N}(x|x_t, \sigma^2)$
   - (ii) (**Evaluate**) Calculate acceptance probability:

$$A(x', x_t) = \min\left\{1, \frac{p(x')\ q(x_t|x')}{p(x_t)\ q(x'|x_t)}\right\}$$

   - (iii) (**Select**) Sample $u \sim \mathrm{Unif}[0,1]$.

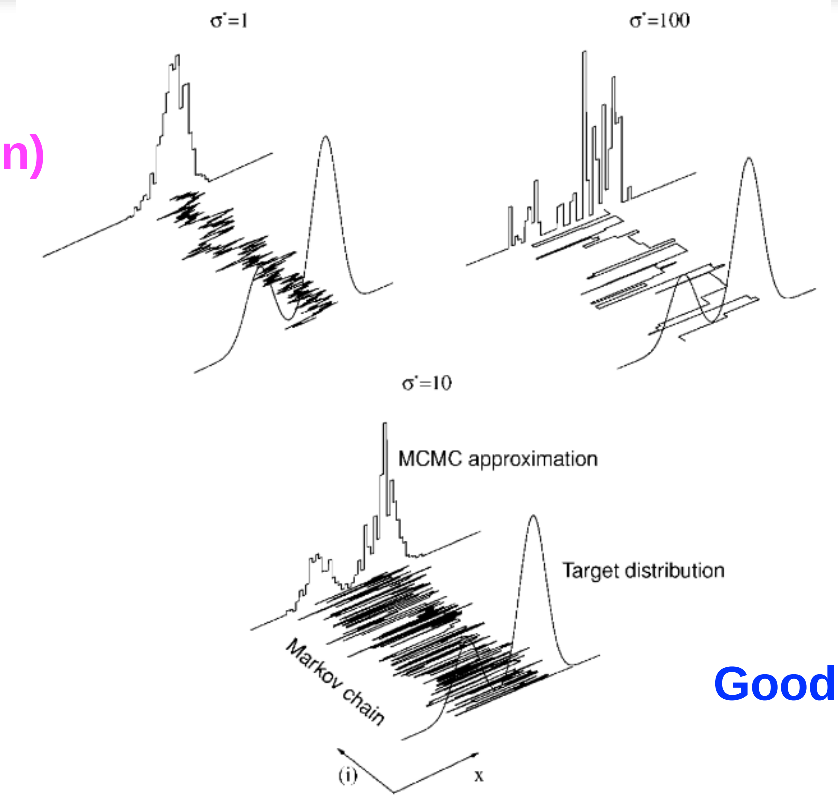
     If $A(x', x_t) > u$, then $x_{t+1} := x'$.

     Else $x_{t+1} := x_t$.

$$\boxed{p(x) \propto \exp(-f(x))}$$

METROPOLIS-HASTINGS ALGORITHM

Too small
(mode omission)



Too large
(low acceptance)

Good!

INDEPENDENT SAMPLER

1. Initialize $x_t := x_0$.
2. For $t \in \{0,1,\dots,T-1\}$:
   (i) (**Generate**) Sample $x' \sim q(x)$.
   (ii) (**Evaluate**) Calculate acceptance probability:

$$A(x', x_t) = \min\left\{1, \frac{p(x')\, q(x_t)}{p(x_t)\, q(x')}\right\}$$

   (iii) (**Select**) Sample $u \sim \mathrm{Unif}[0,1]$.
   If $A(x', x_t) > u$, then $x_{t+1} := x'$.
   Else $x_{t+1} := x_t$.

METROPOLIS ALGORITHM

1. Initialize $x_t := x_0$.
2. For $t \in \{0, 1, \ldots, T-1\}$:
   (i) (**Generate**) Sample $x' \sim q(x|x_t)$, where $q(x'|x_t) = q(x_t|x')$.
   (ii) (**Evaluate**) Calculate acceptance probability:
   $$A(x', x_t) = \min\left\{1, \frac{p(x')}{p(x_t)}\right\}$$
   (iii) (**Select**) Sample $u \sim \mathrm{Unif}[0,1]$.
   If $A(x', x_t) > u$, then $x_{t+1} := x'$.
   Else $x_{t+1} := x_t$.

SIMULATED ANNEALING

1. Initialize $x_t := x_0$ and $T_t := T_0$.
2. For $t \in \{0, 1, \ldots, T - 1\}$:
   (i) (**Generate**) Sample $x' \sim q(x|x_t)$.
   (ii) (**Evaluate**) Calculate acceptance probability:

$$A(x', x_t) = \min\left\{1, \frac{p^{\frac{1}{T_t}}(x')\, q(x_t|x')}{p^{\frac{1}{T_t}}(x_t)\, q(x'|x_t)}\right\}$$

   (iii) (**Select**) Sample $u \sim \mathrm{Unif}[0,1]$.
   If $A(x', x_t) > u$, then $x_{t+1} := x'$.
   Else $x_{t+1} := x_t$.
   (iv) Set T_{t+1} according to chosen cooling schedule.

SIMULATED ANNEALING

1. Initialize $x_t := x_0$ and $T_t := T_0$.
2. For $t \in \{0,1,\ldots,T-1\}$:
   (i) (**Generate**) Sample $x' \sim q(x|x_t)$.
   (ii) (**Evaluate**) Calculate acceptance probability:

$$A(x', x_t) = \min\left\{1, \frac{p^{\frac{1}{T_t}}(x')\, q(x_t|x')}{p^{\frac{1}{T_t}}(x_t)\, q(x'|x_t)}\right\}$$

   (iii) (**Select**) Sample $u \sim \mathrm{Unif}[0,1]$.
   If $A(x', x_t) > u$, then $x_{t+1} := x'$.
   Else $x_{t+1} := x_t$.
   (iv) Set T_{t+1} according to chosen cooling schedule.

If $p(x) \propto \exp(-f(x))$,
then $p^{\frac{1}{T}}(x) \propto \exp\left(-\frac{1}{T} f(x)\right)$.

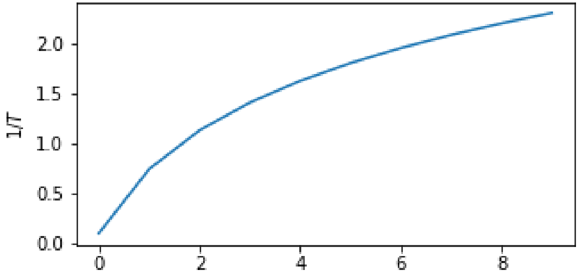
SIMULATED ANNEALING

1. Initialize $x_t := x_0$ and $T_t := T_0$.
2. For $t \in \{0, 1, \dots, T-1\}$:
   (i) (**Generate**) Sample $x' \sim q(x \mid x_t)$.
   (ii) (**Evaluate**) Calculate acceptance probability:

$$A(x', x_t) = \min \left\{ 1, \frac{p^{\frac{1}{T_t}}(x')\; q(x_t \mid x')}{p^{\frac{1}{T_t}}(x_t)} \right\}$$

$$T = C \ln(t + T_0))^{-1}$$

   (iii) (**Select**) Sample $u \sim \mathrm{Unif}[0,1]$.
   If $A(x', x_t) > u$, then $x_{t+1} := x'$.
   Else $x_{t+1} := x_t$.
   (iv) Set T_{t+1} according to chosen cooling schedule.



OPTIMIZATION THROUGH SAMPLING

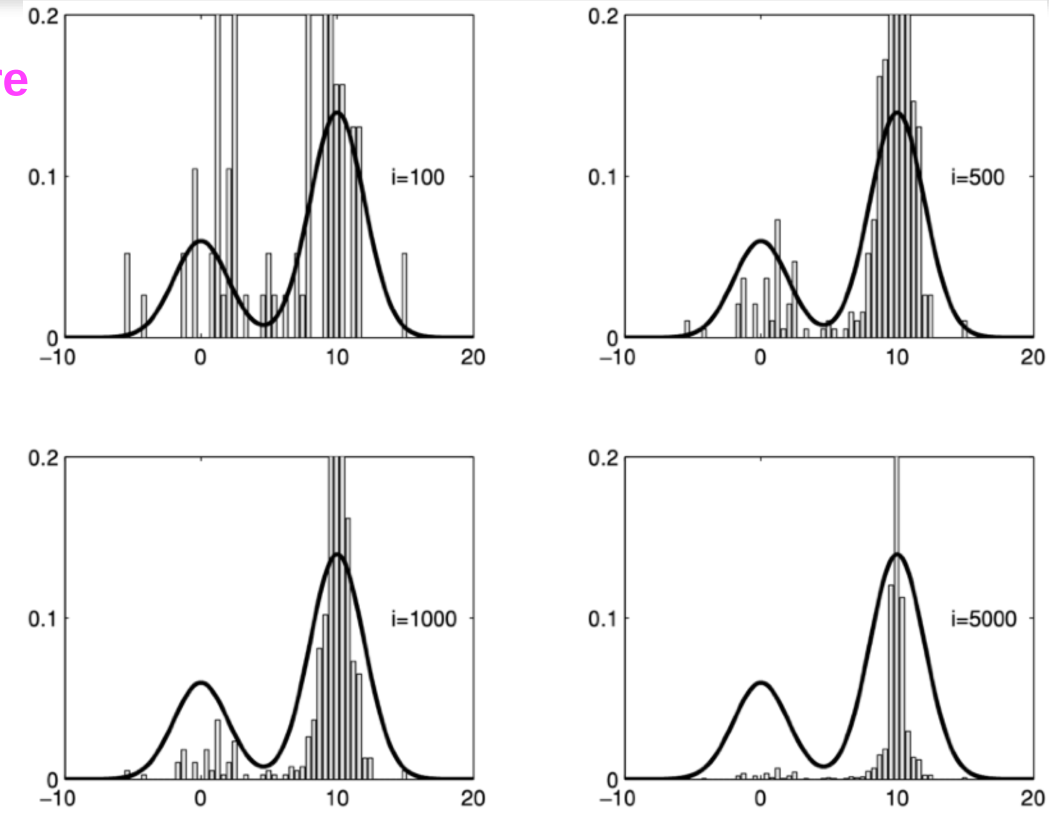
- We are interested in minimizing $f(x)$.
- Alternatively, we can consider a distribution of the form

$$p(x) \propto \exp(-f(x)/T)$$

- By changing the temperature, we can consider almost uniform distribution (high temperature) or peaky distribution (low temperature).
- For uniform distribution we have **exploration**, while for more peaky distribution we have **exploitation**.

EXAMPLE (SIMULATED ANNEALING)

High temperature



Low temperature

METROPOLIS-HASTINGS ALG. VS LOCAL SEARCH

Stochastic search

1. Initialize $x_t := x_0$.
2. For $t \in \{0, 1, \dots, T-1\}$:
   (i) (**Generate**) Sample $x' \sim q(x|x_t)$.
   (ii)(**Evaluate**) Calculate acceptance probability:

$$A(x', x_t) = \min\left\{1, \frac{p(x')\, q(x_t|x')}{p(x_t)\, q(x'|x_t)}\right\}$$

   (iii) (**Select**) Sample $u \sim \mathrm{Unif}[0,1]$.
   If $A(x', x_t) > u$, then $x_{t+1} := x'$.
   Else $x_{t+1} := x_t$.

Deterministic search

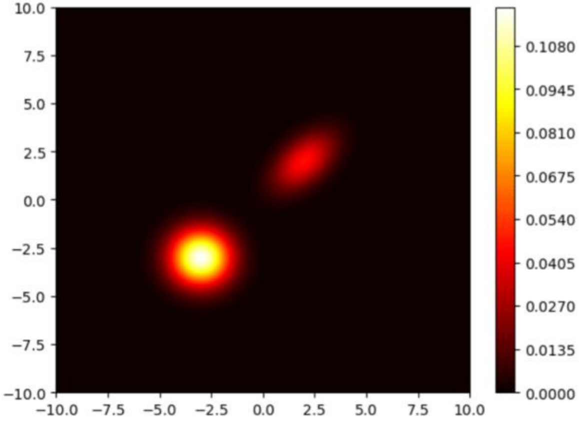
1. Initialize a solution.
2. For $t \in \{0, 1, \dots, T-1\}$:
   (i)(**Generate**) Generate solutions in the neighborhood of the current best solution.
   (ii)(**Evaluate**) Evaluate the potential solutions.
   (iii)(**Select**) Pick the solution with best objective as the best current solution.

ASSIGNMENT 2: Sampling

The goal is to implement Metropolis-Hastings (MH) and Simulated Annealing (SA) algorithms and analyze their behavior.
Here, we are interested in sampling from a mixture of two Gaussians, namely:

Target distribution: $p(\mathbf{x}) = 0.25 \cdot \mathcal{N}\left(\mu = \begin{bmatrix} 2 \\ 2 \end{bmatrix}, \Sigma = \begin{bmatrix} 1 & 0.5 \\ 0.5 & 1 \end{bmatrix}\right) + 0.75 \cdot \mathcal{N}\left(\mu = \begin{bmatrix} -3 \\ -3 \end{bmatrix}, \Sigma = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}\right)$

Visualization:



Thank you!

VU VRIJE UNIVERSITEIT AMSTERDAM

EXTRA READING

Andrieu et al., "An Introduction to MCMC for Machine Learning", Machine Learning, Vol. 50, pp. 5-43, 2003

Ch. Bishop, "Pattern Recognition and Machine Learning", Springer

K. Murphy, "Machine Learning: A Probabilistic Perspective", The MIT Press