Data-Driven Sampling (Lecture 4)
Monte Carlo algorithm, Maximum likelihood estimator, Stochastic search

Hongwei YUAN

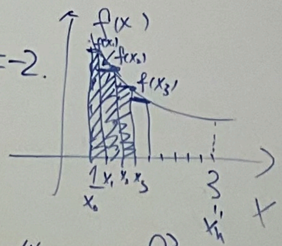
University of Macau

Prob. $f(x) = \mathbb{1}_Q = \begin{cases} 1, & x \text{ is } Q \cap [1,3]. \\ 0, & x \text{ in } [1,3] \setminus Q. \end{cases}$

Q: how to evaluate $\int_a^b f(x)\,dx$ approximately by computer?

$\int_{-\infty}^{\infty} f(x-1)\,dx = \int_{-\infty}^{\infty} g(x)\,dx = \lim_{a \to +\infty} \int_{-a}^{a} g(x)\,dx = -2.$

$\int_{-\infty}^{\infty} f(x)\,dx \overset{?}{=} \lim_{a \to +\infty} \int_{-a}^{a} f(x)\,dx = 0$



$f(x) = \begin{cases} 1, & x \geq 0 \\ -1, & x < 0 \end{cases}$

$g(x) = f(x-1) = \begin{cases} 1, & x \geq 1 \\ -1, & x < 1. \end{cases}$

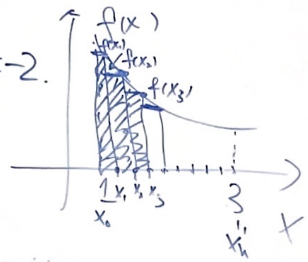
$\sum_{i=1}^{n} f(x_i) \cdot (x_i - x_{i-1}) \leq \int_1^3 f(x)\,dx$

$\lim_{\delta = |x_i - x_{i-1}| = \frac{2}{n} \to 0} \sum_{i=1}^{n} f(x_i) \cdot (x_i - x_{i-1}) = \int_1^3 f(x)\,dx.$

$f(x)\,dx$ approximately by computer?

$dx = \lim_{a \to +\infty} \int_{-a}^{a} g(x)\,dx = -2.$

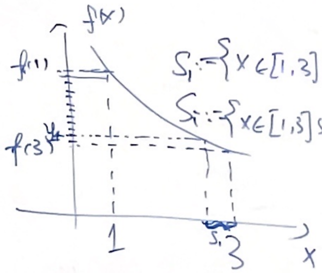
$x)\,dx = 0$



Rieman Sum: $\sum_{i=1}^{n} f(x_i) \cdot (x_i - x_{i-1})$

$\lim_{\delta = |x_i - x_{i-1}| = \frac{2}{n} \to 0} \sum_{i=1}^{n}$

$x \geq 1$



$y_n = f(1)$

$y_0 = f(3)$

$S_1 = \{x \in [1,3] \text{ s.t. } y_0 \leq f(x) \leq y_1\}.$

$S_i = \{x \in [1,3] \text{ s.t. } y_{i-1} \leq f(x) \leq y_i\}.$

Lebesgue: $\sum_{i=0}^{n} \frac{y_i + y_{i-1}}{2} |S_i| \approx \int_1^3 f(x)\,d\mu(x)$

$= \int_1^3 f(x)\,g(x)\,dx$

uniform dist.

$\int_a^b f(x)\,dx = (b-a) \int_a^b f(x) \cdot \boxed{\frac{1}{b-a} \chi_{[a,b]}(x)}\,dx$

Prob. $f(x) = \mathbb{1}_Q = \begin{cases} 1, & x \text{ is } Q \cap [1,3]. \\ 0, & x \text{ in } [1,3] \setminus Q. \end{cases}$

generate 10000 samples

$\sim \mu \sim g(x).$

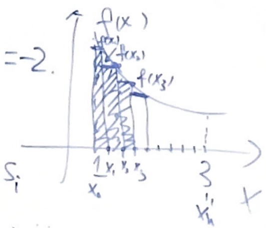
$g(x) = e^{-x}, \; x \geq 0$

$S = [10^4, +\infty), \; \mu(S) = \int_{10^4}^{\infty} e^{-x}\,dx = e^{-10^4}$

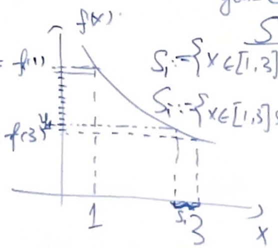
$f(x)\,dx$ approximately by computer?

$dx = \lim_{a \to +\infty} \int_{-a}^{a} g(x)\,dx = -2.$

$x)\,dx = 0$



$x \in S_i$



$y_n = f(1)$

$y_0 = f(3)$

$S_1 = \{x \in [1,3] \text{ s.t. } y_0 \leq f(x) \leq y_1\}.$

$S_i = \{x \in [1,3] \text{ s.t. } y_{i-1} \leq f(x) \leq y_i\}.$

Rieman Sum: $\sum_{i=1}^{n} f(x_i) \cdot (x_i - x_{i-1})$

$\lim_{\delta = |x_i - x_{i-1}| = \frac{2}{n} \to 0} \sum_{i=1}^{n} \qquad \sum_{i=1}^{10000} \frac{f(x_i)}{10000}$

$\sum_{i=1}^{10000} f(x_i) \frac{1}{10000}$

$\approx$

Lebesgue. $\sum_{i=0}^{n} \frac{y_i + y_{i-1}}{2} |S_i| \approx \int_1^3 f(x)\,d\mu(x)$

$\mu(S_i) = \int_1^3 f(x)\,g(x)\,dx$

$\approx \frac{\# x_i \in S_i}{10000} \qquad x_1, x_2, x_3 \in S_i$

Law of large number (LLN)

Theorem

Let $X, X_1, ..., X_n$ *are independent identical distributed random variables and* g *be a function.* $\mathbb{E}[||g(X)||] < \infty$, *then we have*

$$\lim_{n\to\infty} \frac{g(X_1) + ... + g(X_n)}{n} = \mathbb{E}[g(X)], \quad \textit{with probability } 1.$$

If X *has a pdf* $f(x)$, *then*

$$\mathbb{E}[g(X)] = \int_{\mathbb{R}^d} g(x) f(x) dx.$$

Computation of Gamma function

$\Gamma(1) = \int_0^\infty e^{-x} dx = -e^{-x} \Big|_0^\infty$ $= -\lim_{x \to \infty} e^{-x} + 1$ $= 1$

$\lim_{x \to \infty} x^{\lambda-1} e^{-x} = 0$

$\Rightarrow \Gamma(n) = (n-1)!, \ \forall n \geq 1.$

$d(e^{-x}) = -e^{-x} dx$

$\Gamma(\lambda) = \int_0^\infty x^{\lambda-1} e^{-x} dx = \int_0^\infty x^{\lambda-1} - d(e^{-x})$

$= -x^{\lambda-1} \cdot e^{-x} \Big|_0^\infty + \int_0^\infty e^{-x} (\lambda-1) x^{\lambda-2} dx$

$\overset{\lambda>1}{=} 0 + (\lambda-1)\Gamma(\lambda-1)$

$f(x) = \{ 1,$

Lebesgue

Let us consider the following Gamma function:

$$\Gamma(\lambda) = \int_0^\infty x^{\lambda-1} e^{-x} dx$$

where $\lambda \geq 0$.

- When $\lambda = 0, 1, 2, ...,$ we can compute $\Gamma(\lambda)$ by hand (Assignment 1).
- When λ is not an integer, we can not get an exact value.
- Now we use LLN to obtain an approximate number as the following:
  - Sample i.i.d. random variables $X_1, ..., X_n \sim Exp(1)$, and take $g(x) = x^{\lambda-1}$;

    $-\ln(U) \sim Exp(1)$ where $U \in U([0,1])$

    $f(x; \lambda=1) = \begin{cases} e^{-x}, & x \geq 0 \\ 0, & x < 0 \end{cases}$
  - Compute $\frac{g(X_1)+...+g(X_n)}{n}$ for large n.

Python code for the computation of Gamma function

```
import numpy as np
lam,size=2.5,1000
X=np.random.exponential(1,size)
Y=sum(X**(lam-1))/size
print(Y)
```

lam = 2.5 size = 1000

$X \sim Exp(1, size = 1000)$

$X^{lam-1} = g(X), \frac{\sum_{i=1}^{size} g(X_i)}{size}$

Monte Carlo method

Monte Carlo is a casino city in Monaco. The Monte Carlo method is to sample n i.i.d. random variables with probability density f

$$X_1, X_2, ..., X_n$$

and use the following average:

$$\frac{g(X_1) + ... + g(X_n)}{n}$$

to compute the integral

$$\int_{\mathbb{R}^d} g(x) f(x) dx.$$

Example

楊宜樹

Compute

$\Phi(x)$ s.t. $X \sim N(0,1)$

$$\Phi(x) = \int_{-\infty}^{x} \frac{1}{\sqrt{2\pi}} e^{-\frac{t^2}{2}} dt.$$

- One can't get an explicit formula for $\Phi(x)$.
- We can use Monte Carlo method to compute its value: define $h(t) = 1_{(-\infty,x]}(t)$ and $f(t) = \frac{1}{\sqrt{2\pi}} e^{-\frac{t^2}{2}}$, we have

$$\frac{h(X_1) + ... + h(X_n)}{n}$$

is very close to $\Phi(x)$ when n is large, where $X_1, ..., X_n \sim N(0,1)$ are i.i.d.

Python code for the example

```
import numpy as np
x,size=0.78,10000
Y=0
X=np.random.normal(0,1,size)
for i in range(size):
  if X[i] <= x:
    Y=Y+1
Phi=Y/size
print(Phi)
```

Assignment

Assignment 2: Create a python program to compute the following integral:

$$\int_0^{2\pi} [\sin(100x) + \cos(50x)]^2 dx$$

Importance sampling

Let us consider the following integral problem

$$\int_{\mathbb{R}^d} h(x)f(x)dx$$

where f is the pdf of a probability distribution.

- It often happens that the distribution with the pdf f is hard to be sampled.
- One way is to choose another distribution with the pdf g which can be easily sampled and consider

$$\int_{\mathbb{R}^d} \left[h(x)\frac{f(x)}{g(x)}\right] g(x)dx$$

- We sample $X_1, ..., X_n$ with pdf g and compute $\frac{1}{n}\sum_{i=1}^{n} h(X_i)\frac{f(X_i)}{g(X_i)}$.

Idea of maximum likelihood estimator

- Aim: There exists a probability distribution $p(x, \theta)$ with θ being some important parameter not known. We need to learn this θ.
- Method: Take n samples $X_1, X_2, ..., X_n$ (which are usually observed data in practice) and consider the **likelihood** of these n samples as the following

$$L_n(\theta) := p(X_1, \theta)...p(X_n, \theta).$$

- Maximization: Consider the following maximization problem:

$$\hat{\theta}_n = argmax_{\theta} L_n(\theta),$$

where $\hat{\theta}_n$ is a function of $X_1, ..., X_n$. $\hat{\theta}_n$ is called maximum likelihood estimator based on the samples of $X_1, ..., X_n$.

Maximum likelihood estimator (MLE)

Definition

Let $p(x, \theta)$ be a probability distribution with an unknown parameter θ. Let $X_1, ..., X_n$ be n independent random variables with distribution $p(x, \theta)$. The **likelihood function** based on $X_1, ..., X_n$ is defined as

$$L_n(\theta) = \prod_{i=1}^{n} p(X_i, \theta).$$

The **maximum likelihood estimator (MLE)** of θ based on $X_1, ..., X_n$ is defined as

$$\hat{\theta}_n = argmax_{\theta \in \Theta} L_n(\theta),$$

where Θ is a set in which the parameter θ can take its values. We know $\hat{\theta}_n$ is a function of $X_1, ..., X_n$.

Maximum log likelihood estimator (log-MLE)

Definition

Let $p(x, \theta)$ be a probability distribution with an unknown parameter θ. Let $X_1, ..., X_n$ be n independent random variables with distribution $p(x, \theta)$. The **log likelihood function** based on $X_1, ..., X_n$ is defined as

$$l_n(\theta) = \log L_n(\theta) = \sum_{i=1}^{n} \log p(X_i, \theta).$$

The **maximum likelihood estimator (MLE)** of θ based on $X_1, ..., X_n$ can also be defined as

$$\hat{\theta}_n = argmax_{\theta \in \Theta} l_n(\theta).$$

An example

Suppose that we survey 20 individuals working for a large company and ask each whether they favor implementation of a new policy regarding retirement funding. If, in our sample, 6 favored the new policy, find an estimate for θ, the true but unknown proportion of employees that favor the new policy.

Solution

- Randomly choose an individual, he/she has a probability $\theta \in [0,1]$ in favor of the new policy and a probability $1-\theta$ not in favor. We define the random variable X such that

$$X = \begin{cases} 1, & in\ favor, \\ 0, & not\ in\ favor. \end{cases}$$

The probability distribution of X is Bernoulli with a parameter θ, i.e. $p(1,\theta) = \theta$ and $p(0,\theta) = 1-\theta$. We aim to estimate this θ.

- We have 20 samples $X_1, ..., X_{20}$, the likelihood function is

$$L_{20}(\theta) = \prod_{i=1}^{20} p(X_i, \theta) = \theta^6 (1-\theta)^{14}.$$

Solution (continued)

- Maximize the $L_{20}(\theta)$. Differentiating $\log[\theta^6(1-\theta)^{14}]$ about θ, we obtain
$$\frac{6}{\theta} - \frac{14}{1-\theta} = 0.$$
Solving, we obtain
$$\theta = 6/20.$$

Cauchy distribution

A Cauchy distribution with parameter $\theta \in \mathbb{R}$ and $\gamma > 0$, denoted by Cauchy(θ, γ), has the following pdf:

$$f(x, \theta, \gamma) = \frac{1}{\pi} \frac{\gamma}{\gamma^2 + (x - \theta)^2}.$$

Assignment 4: Verify that $f(x, \theta, \gamma)$ is a pdf.

1. $f \geq 0$

2. $\int_{-\infty}^{+\infty} f(x, \theta, \gamma)\, dx = 1$

$(\arctan x)' = \frac{1}{1+x}$

Example: MLE for Cauchy distribution

We consider maximize the likelihood of a Cauchy distribution Cauchy$(\theta, 1)$ based on sample $X_1, ..., X_n$:

joint pdf of $(x_1, \cdots, x_n)$ since $x_1, \cdots, x_n$ are i.i.d

$$L_n(\theta) = \prod_{i=1}^{n} \frac{1}{\pi} \frac{1}{1 + (X_i - \theta)^2}. \tag{1}$$

In order to obtain the MLE, we consider

$$\ell_n(\theta) = n \log \frac{\gamma}{\pi} - \sum_{i=1}^{n} \log(\gamma^2 + (X_i - \theta)^2)$$

$$\frac{dL_n(\theta)}{d\theta} = 0.$$

It is equivalent to compute

$$\frac{d\ell_n'(\theta)}{d\theta} = 0$$

$$\frac{d[\log L_n(\theta)]}{d\theta} = 0. \iff \sum_{i=1}^{n} \frac{1}{\gamma^2 + (X_i - \theta)^2} \cdot 2(X_i - \theta) = 0$$

Unfortunately, there does not exist a closed solution for the previous two equations.

MLE for Cauchy distribution

Theorem

Let $X_1, ..., X_n$ be n independent random variables with Cauchy distribution Cauchy($\theta, 1$). Then the MLE $\hat{\theta}_n$ of θ based on $X_1, ..., X_n$ with the form:

$$\hat{\theta}_n = argmin_{\theta \in \mathbb{R}} L_n(\theta),$$

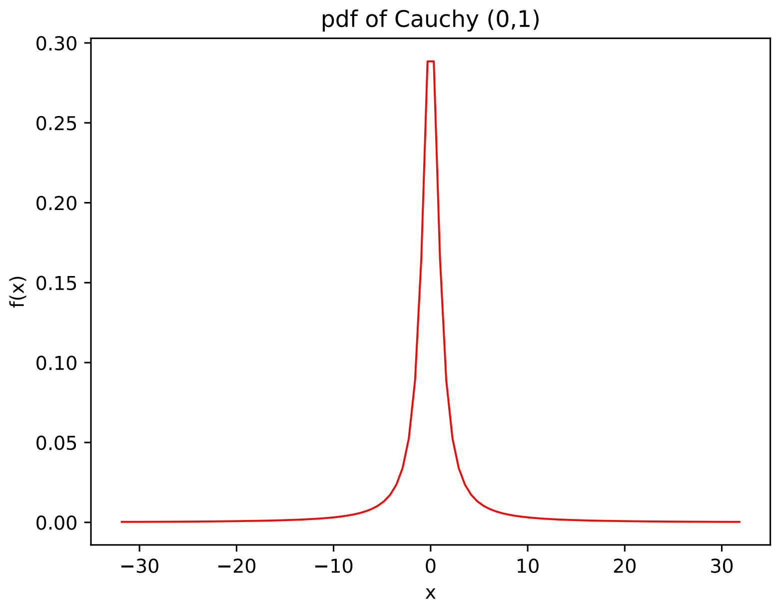
($\hat{\theta}_n$ depends on $X_1, ..., X_n$), satisfies

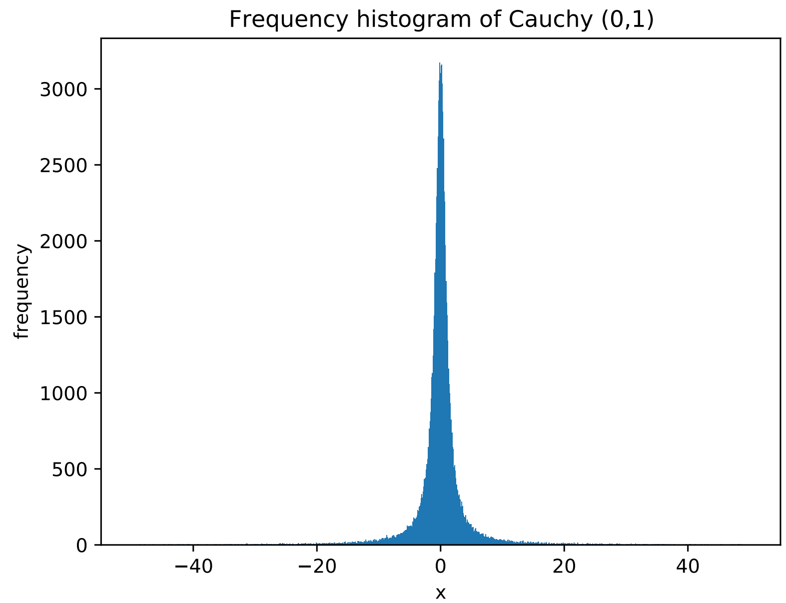
$$\lim_{n \to \infty} \hat{\theta}_n = \theta \qquad \text{with probability 1.}$$

The proof of the theorem is out of the scope of this course, but we will show it is approximately true when $\theta = 0$ by a Python program.

Standard Cauchy distribution Cauchy(0,1)

$f(x;\theta=0;\gamma=1) = \frac{1}{\pi}\frac{1}{1+x^2}$ symmetric around $x=0$





Python of MLE for Cauchy distribution

```python
import numpy as np
from scipy.stats import cauchy
from scipy.optimize import minimize_scalar
import matplotlib.pyplot as plt
fig, ax = plt.subplots(1, 1)
x = np.linspace(cauchy.ppf(0.01),cauchy.ppf(0.99), 100)
ax.plot(x, cauchy.pdf(x),'r-', lw=1, alpha=1, label='cauchy pdf')
plt.title('pdf of Cauchy (0,1)')
plt.xlabel('x')
plt.ylabel('f(x)')
plt.savefig('CPDF.pdf')
plt.show()
X = np.random.standard_cauchy(100000)
X = X[(X>-50) & (X<50)]  # truncate distribution so it plots well
plt.hist(X, bins=1000)
plt.title('Frequency histogram of Cauchy (0,1)')
plt.xlabel('x')
plt.ylabel('frequency')
plt.savefig('CH.pdf')
plt.show()
def L(x):        # Define -log MLE function
    return sum(-np.log(1/(1+(X-x)**2)))
res=minimize_scalar(L)      # find minimizer
res.x
print(res.x)
# Stochastic Search
```

Stochastic search for optimization problems

- Problem: In many problems in Statistics and machine learning, one needs to search the maximimum of a given function h in a domain $\mathcal{D} \subset \mathbb{R}^d$, i.e., $h^* = \max_{x \in \mathcal{D}} h(x)$. h^* = max of $h(x)$ on domain $x \in D$

  grid search by 枚举
- Classical numerical method: One makes a grid $G = \{x_1, ..., x_N\}$ on $\mathcal{D}$, and compute the value of h on the grid and compute $\max\{h(x_1), ..., h(x_N$ as an approximation of h^*. $D \in \mathbb{R}^d$ G has $n^d = e^{\log n \cdot d}$ points complexity increases exponentially
- Stochastic search: We sample n random variables $X_1, ..., X_n$ which satisfy the uniform distribution on $\mathcal{D}$, i.e. each X_i has a distribution $\mathcal{U}_{\mathcal{D}}$ and compute will not grow exponentially

  e.g.. compare $n=1000$ or $n=2000$ to see if max changes

$$\max\{h(X_1), ..., h(X_n)\}$$

  biggerly or not. to scale to 0.001 to see changes.

  as an approximation of h^*.

Stochastic search for optimization problems

- The computation complexity increasing exponentially with the dimension d, and computation resources will usually be huge when $d \geq 5$.
- The advantage of stochastic search is that their computation complexity does not depend on the dimension d.

Stochastic search for MLE for Cauchy distribution

We consider maximizing the likelihood of a Cauchy distribution Cauchy$(\theta, 1)$ based on samples $X_1, ..., X_n$:

$$L_n(\theta) = \prod_{i=1}^{n} \frac{1}{\pi} \frac{1}{1 + (X_i - \theta)^2}. \tag{2}$$

- If the samples $X_1, ..., X_n$ is from Cauchy$(0, 1)$, we generate m (e.g. m=1000) random variables $\theta_1, ..., \theta_m$ from the distribution $\mathcal{U}(-5, 5)$ and find

$$\max\{L_n(\theta_1), ..., L_n(\theta_m)\}.$$

Python of stochastic search for MLE for Cauchy distribution with m=100

```
import numpy as np
from scipy.stats import cauchy
from scipy.optimize import minimize_scalar
import matplotlib.pyplot as plt
fig, ax = plt.subplots(1, 1)
x = np.linspace(cauchy.ppf(0.01),cauchy.ppf(0.99), 100)
ax.plot(x, cauchy.pdf(x),'r-', lw=1, alpha=1, label='cauchy pdf')
plt.title('pdf of Cauchy (0,1)')
plt.xlabel('x')
plt.ylabel('f(x)')
plt.savefig('CPDF.pdf')
plt.show()
X = np.random.standard_cauchy(100000)
X = X[(X>-50) & (X<50)]  # truncate distribution so it plots well
plt.hist(X, bins=1000)
plt.title('Frequency histogram of Cauchy (0,1)')
plt.xlabel('x')
plt.ylabel('frequency')
plt.savefig('CH.pdf')
plt.show()
def L(x):        # Define -log MLE function
    return sum(-np.log(1/(1+(X-x)**2)))
res=minimize_scalar(L)      # find minimizer
res.x
print(res.x)
# Stochastic Search
size=100
x=np.random.uniform(-5,5,size)
mp=x[0]
for i in range(size):
    if L(x[i])<L(mp):
       mp=x[i]
print(mp)
```