Introduction and Examples
(Chapter 1)

Michael Tsiang

Stats 102C: Introduction to Monte Carlo Methods

UCLA

Acknowledgements: Qing Zhou

Outline

What Are Monte Carlo Methods?

Monte Carlo methods are named after the Monte Carlo Casino in Monaco, a world renowned icon for gambling.

Monte Carlo methods use repeated random sampling through computer simulation for:

- Optimization
- Numerical integration
- Generating random variables (samples) from well known or new probability distributions

Using Monte Carlo methods is particularly useful when theoretical (or closed-form) solutions are difficult or impossible.

Pseudorandom Numbers

- Computers cannot generate truly random numbers.
- Computers follow an algorithm (hidden from the user) that generates **pseudorandom** numbers ("pseudo" means fake).
- If we knew the algorithm, the numbers would not be random at all: The numbers are deterministic.
- Pseudorandom numbers are **statistically random**, in that they are "random enough" for statistical analysis and inference.
- We will use and refer to computer generated random numbers as if they are random, but it is implicitly understood that they are pseudorandom.

Pseudorandom Numbers

- A pseudorandom number generator uses a hidden deterministic algorithm that starts from an initial number (called the **seed**) and generates pseudorandom numbers from it.
- The user can often specify (or **set**) the seed so that the "random" numbers that are generated from a given function are the same every time the function is run.
- Being able to set the seed allows researchers to reproduce simulation results.
- In R: `set.seed()`
- More on pseudorandomness: `https://en.wikipedia.org/wiki/Pseudorandomness`

Uniform Assumption

We will rely on the basic assumption that we can generate samples from $\mathrm{Unif}(0,1)$, the uniform distribution on the interval $(0,1)$.

- Probability density function: $f(x) = \begin{cases} 1 & \text{for } x \in (0,1) \\ 0 & \text{otherwise} \end{cases}$
- We will not be concerned with the details of how to generate from $\mathrm{Unif}(0,1)$.
- In R: `runif()` can be used to generate from $\mathrm{Unif}(0,1)$.
- R uses the Mersenne Twister pseudorandom number generator: `https://en.wikipedia.org/wiki/Mersenne_Twister`

Outline

Example 1: Calculating Area

Suppose we want to compute the area of a region D in $\mathbb{R}^2$.

- The region may be irregularly shaped and not easily computed in closed form.
- How can we approximate the area of D?

Example 1: Calculating Area

Suppose we want to compute the area of a region D in $\mathbb{R}^2$.

1. Consider a rectangle A which contains the region D:

$$A : [x_1, x_2] \times [y_1, y_2] \supset D.$$

The area of A is $S(A) = (x_2 - x_1)(y_2 - y_1)$.

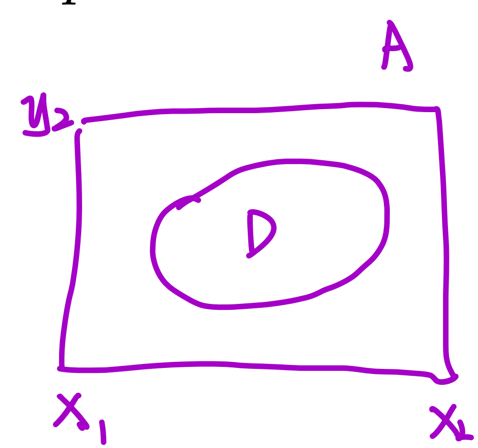
2. Generate n (say 1000) points uniformly in A. Then

$$P(\text{a point in D}) = \frac{S(D)}{S(A)} := p.$$

Then the area of D is $S(D) = S(A) \cdot p$.

known ?

We want a way to estimate p.



Example 1: Calculating Area

- Let M denote the number of points in D out of the n points uniformly generated in A: M is a binomial random variable.

- Specifically, $M \sim \text{Bin}(n, p)$, and

$$E(M) = np, \quad \text{Var}(M) = np(1-p).$$

- Let $\hat{p} = \dfrac{M}{n}$. Then we can estimate $S(D)$ by

unbiased

$$\widehat{S(D)} = S(A) \cdot \overset{p}{\hat{p}} = S(A) \cdot \frac{M}{n}.$$

- Is $\widehat{S(D)}$ a good estimator for $S(D)$? Is it consistent?

$E(\widehat{S(D)}) = S(D)$?

Example 1: Calculating Area

$M \sim (n, p)$ $E(M) = np$

$$
\begin{aligned}
E\left[\widehat{S(D)}\right] = E\left[S(A) \cdot \frac{M}{n}\right] &= \frac{S(A)}{n} \cdot E(M) \\
&= \frac{S(A)}{n} \cdot np \\
&= S(A) \cdot \frac{S(D)}{S(A)} \\
&= S(D)
\end{aligned}
$$

unbiased

$$
\begin{aligned}
\mathrm{Var}\left[\widehat{S(D)}\right] = \mathrm{Var}\left[\underline{S(A)} \cdot \frac{M}{\underline{n}}\right] &= \left[\frac{S(A)}{n}\right]^2 \cdot \mathrm{Var}(M) \\
&= \left[\frac{S(A)}{n}\right]^2 \cdot np(1-p) \\
&= \frac{S(A)^2 p(1-p)}{n}
\end{aligned}
$$

- $E\left[\widehat{S(D)}\right] = S(D)$, so $\widehat{S(D)}$ is an unbiased estimator of $S(D)$.
- $\mathrm{Var}\left[\widehat{S(D)}\right] = \dfrac{S(A)^2 p(1-p)}{n} \xrightarrow{n\to\infty} 0$.

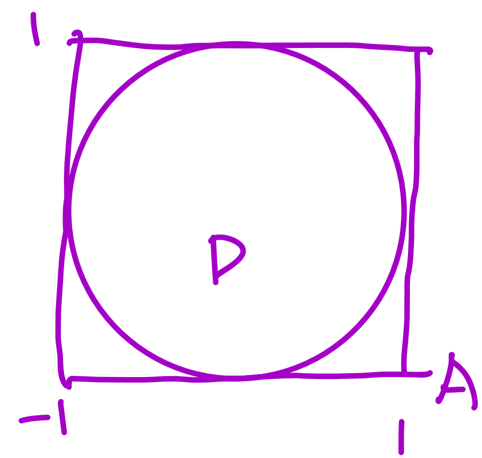
  $n \to \infty$
- So $\widehat{S(D)}$ is a consistent estimator of $S(D)$: $\widehat{S(D)} \xrightarrow{P} S(D)$.
- We actually have something stronger: $\hat{p} = \dfrac{M}{n}$ converges to p almost surely (by the Strong Law of Large Numbers), so $\widehat{S(D)} \xrightarrow{\text{a.s.}} S(D)$.

Example 1a: Estimating π

$x^2+y^2=1$

We can use the previous example to estimate $\pi = 3.14159\ldots$

- Let D denote the (open) unit disc $D = \{(x, y) : x^2 + y^2 < 1\}$.
- $S(D) = \pi r^2 = \pi$.



1. Define $A : [-1, 1] \times [-1, 1] \supset D$.

   $S(A) = 2 \cdot 2 = 4.$

2. Generate n points from $\mathrm{Unif}(A)$. Compute

$$S(D) = \hat{\pi} = \frac{M}{n} \cdot S(A) = \frac{M}{n} \cdot 4,$$ where $M = \#$ of points in D.

Example 1a: Estimating π

R Code to estimate π:

```
> # Set the seed for reproduceability
> set.seed(9999)

> n <- 1000 # Specify the number of points to generate

> # Generate n points from A: [-1,1]x[-1,1]
> X <- runif(n, -1, 1) # Generate the x-coordinates
> Y <- runif(n, -1, 1) # Generate the y-coordinates

> # Compute the number of points inside D
> R2 <- X^2 + Y^2
> M <- sum(R2 < 1)

> # Compute hat(pi)
> pihat <- (M / n) * 4
> pihat
[1] 3.164
```

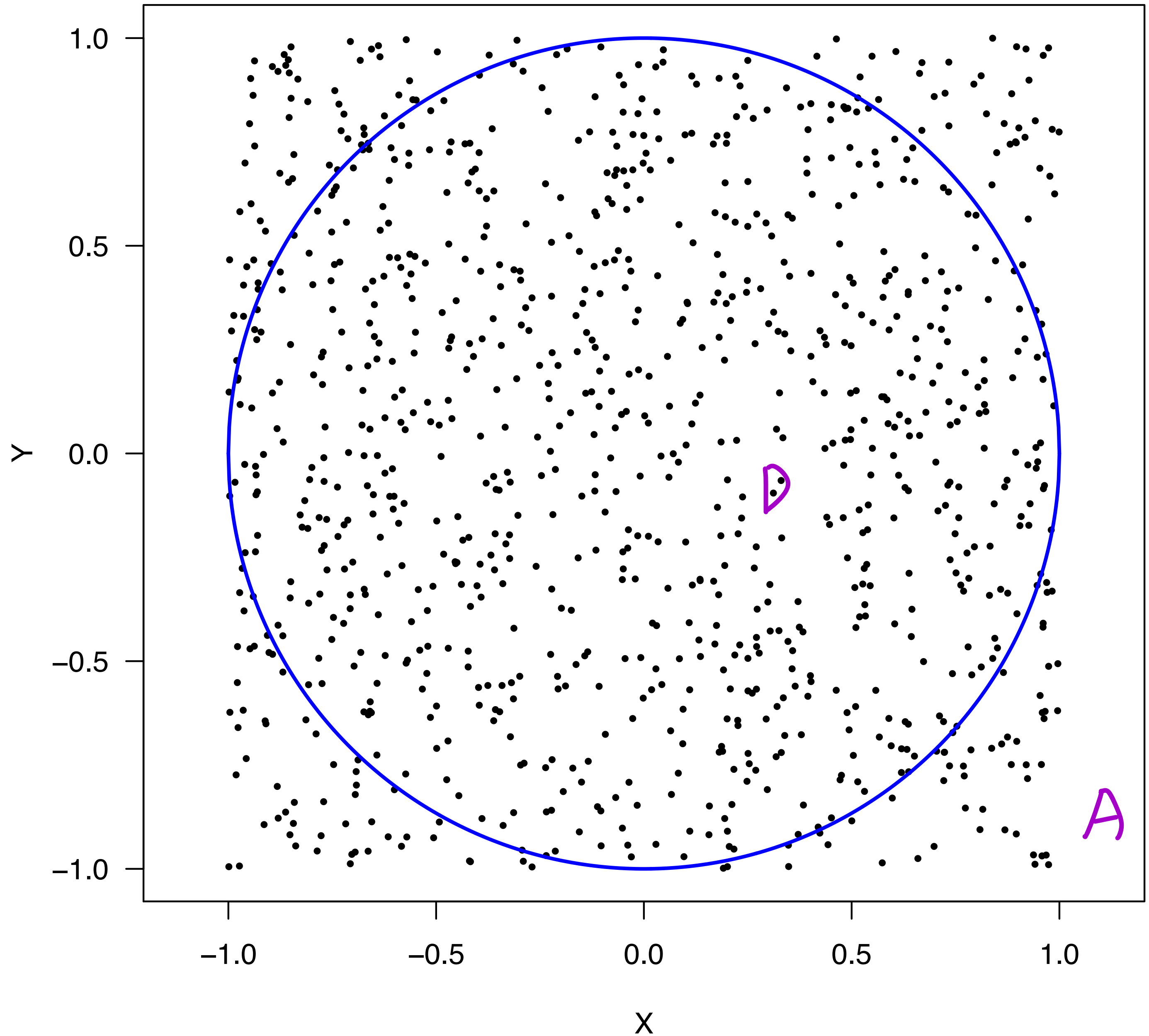
1000

1000

number of point (x,y) satisfy $x^2+y^2<1$

$S(A)$

Example 1a: Estimating π



$S(D) = SA \cdot \frac{m}{N}$

Example 1a: Estimating π

$x^2+y^2=1$

$y=\sqrt{1-x^2}$ → half

$y=-\sqrt{1-x^2}$ → half

R Code for the plot:

```
> # Plot the n points
> plot(X, Y, pch = 19, cex = 0.4, asp = 1, las = 1)

> # Add the unit circle (the boundary of the unit disc D)
> curve(sqrt(1 - x^2), add = TRUE, xlim = c(-1, 1),
+ col = "blue", lwd = 2, n = 1000)
> curve(-sqrt(1 - x^2), add = TRUE, xlim = c(-1, 1),
+ col="blue", lwd = 2, n = 1000)
```

Outline

Example 2: Monte Carlo Integration

Let $g(x)$ be a function, and suppose we want to compute

$$I = \int_a^b g(x)\,\mathrm{d}x.$$

- The function $g(x)$ may be complicated or difficult to integrate in closed form.
- How can we approximate I (assuming it exists)?
- We want to leverage our ability to generate samples from the uniform distribution to approximate $I = \displaystyle\int_a^b g(x)\,\mathrm{d}x$.

Example 2: Monte Carlo Integration

- Let $f(x)$ denote the probability density function of $\mathrm{Unif}(a,b)$:

$$f(x) = \begin{cases} \dfrac{1}{b-a} & \text{for } x \in (a,b) \\ 0 & \text{otherwise} \end{cases}$$

$\int f(x)dx = 1$

- We can rewrite I as

$\int_a^b g(x) \cdot 1 \, dx$

$$\begin{aligned} \int_a^b g(x)\,\mathrm{d}x &= \int_a^b g(x) f(x) \frac{1}{f(x)}\,\mathrm{d}x \\ &= (b-a) \underline{\int_a^b g(x) f(x)\,\mathrm{d}x} \\ &= (b-a)E[g(X)], \end{aligned}$$

where $X \sim \mathrm{Unif}(a,b)$.

fixed

- We want a way to estimate $E[g(X)]$.

Example 2: Monte Carlo Integration

1. Generate $X^{(1)}, X^{(2)}, \ldots, X^{(n)} \overset{\text{iid}}{\sim} \text{Unif}(a, b)$.

2. Compute $g(X^{(1)}), g(X^{(2)}), \ldots, g(X^{(n)})$.

3. We can estimate $E[g(X)]$ by

$$\widehat{E[g(X)]} = \frac{1}{n}\sum_{i=1}^{n} g(X^{(i)}).$$

4. We can then estimate I by

$\frac{1}{n}\Sigma g(x_i)$

$$\hat{I}_n = (b-a)\widehat{E[g(X)]}. = \widehat{\int_a^b g(x)\,dx}$$

When n is large, $\hat{I}_n$ will be a very good approximation to I.
In fact, $\hat{I}_n \xrightarrow{\text{a.s.}} I$.

a integral of any function is write by its expeation value

Example 2: Monte Carlo Integration

This all works because of the Law of Large Numbers!

The (Strong) Law of Large Numbers

Suppose $X^{(1)}, X^{(2)}, \ldots, X^{(n)} \overset{\text{iid}}{\sim} f(x)$, with $E(X^{(1)}) = \mu$ and $\mathrm{Var}(X^{(1)}) = \sigma^2$. Then

$$\frac{1}{n}\sum_{i=1}^{n} X^{(i)} \xrightarrow{\text{a.s.}} \mu.$$

That is,

$$P\left(\lim_{n\to\infty}\left|\frac{1}{n}\sum_{i=1}^{n} X^{(i)} - \mu\right| > \varepsilon\right) = 0.$$

By the Continuous Mapping Theorem, a corollary to this is

$$\frac{1}{n}\sum_{i=1}^{n} g(X^{(i)}) \xrightarrow{\text{a.s.}} E[g(X^{(1)})].$$

Example 2a: Approximating an Integral

We can use the previous example to approximate

$$I = \int_0^1 e^x \, \mathrm{d}x = e - 1 = 1.718\ldots$$ $= (1-0)\, E(e^x)$

$= E(e^x) = \frac{1}{n} \Sigma e^{x_i}$

- We can use $\mathrm{Unif}(0,1)$, with PDF $f(x) = 1$, $x \in (0,1)$.

- Rewrite I as

$$I = \int_0^1 e^x \, \mathrm{d}x = \int_0^1 e^x f(x) \, \mathrm{d}x = E\left[e^X\right],$$

where $X \sim \mathrm{Unif}(0,1)$.

$\int_0^1 e^x f(x) = 1 \cdot \int_0^1 ex\, f(x)\, dx$

$= (1-0)\, E(e^x)$

$= E(e^x) = \frac{e^{x_1} + e^{x_2} + \cdots e^{x_n}}{n}$

Example 2a: Approximating an Integral

A Monte Carlo approach to approximate $I = \int_0^1 e^x \, \mathrm{d}x = e - 1$:

1. Generate $X^{(1)}, X^{(2)}, \ldots, X^{(n)} \overset{\text{iid}}{\sim} \text{Unif}(0, 1)$.
2. Compute $e^{X^{(1)}}, e^{X^{(2)}}, \ldots, e^{X^{(n)}}$.
3. Compute $\hat{I} = \frac{1}{n} \sum_{i=1}^{n} e^{X^{(i)}}$. $= \int_0^1 e^x dx = 1.718$

Example 2a: Approximating an Integral

R Code to approximate I:

```
> # Set the seed for reproduceability
> set.seed(123)

> n <- 1000 # Specify the number of points to generate

> # Generate n points from Unif(0,1)
> X <- runif(n, 0, 1)

> # Compute e^X
> g_X <- exp(X)

> # Compute hat(I)
> mean(g_X)
[1] 1.713043
```

$\frac{\text{sum(g\_x)}}{n}$

Outline

Example 3: Computing Expectations

$x \sim f(x)$ $E(x)$? $var(x)$?

Let $f(x)$ denote the probability density function for a random variable X. Suppose we want to obtain the mean and variance:

$$E(X) = \mu = \int x f(x)\, \mathrm{d}x$$ Continuous

$$\mathrm{Var}(X) = \int (x - \overset{E(x)}{\mu})^2 f(x)\, \mathrm{d}x = E[(X - \mu)^2]$$

- The function $f(x)$ may be known, but the mean and variance may be difficult to compute.
- How can we approximate $E(X)$ and $\mathrm{Var}(X)$?

Example 3: Computing Expectations

1. Generate $X^{(1)}, X^{(2)}, \ldots, X^{(n)} \sim f(x)$.
2. Compute

$$
\begin{aligned}
\hat{\mu} &= \overline{X} = \frac{1}{n}\sum_{i=1}^{n} X^{(i)} \\
\hat{V} &= s_X^2 = \frac{1}{n}\sum_{i=1}^{n} (X^{(i)} - \hat{\mu})^2
\end{aligned}
$$

More generally: $\displaystyle\frac{1}{n}\sum_{i=1}^{n} g(X^{(i)}) \approx E[g(X)]$.

Hard Part: How do we sample from $f(x)$?

We will cover various methods to do this!

Outline

hypothesis : θ

initial : π_0

observation: $x_1 \; x_2 \cdots x_n$ iid from $f_X(\cdot|\theta)$

$$\pi_1(\theta) = \pi_0(\theta | x_1 \cdots x_n) = \frac{f(x_1 \cdots x_n|\theta)\cdot \pi_0(\theta)}{f(X_1=x_1, X_2=x_2 \cdots X_n=x_n)}$$

joint pdf

joint pdf $f(x|\theta)$

$$= f(x_1 \cdots x_n|\theta) = \prod_{i=1}^{n} f_X(x_i|\theta)$$

example:

$p(\text{disease}) = 1\%$ $p(\text{non-disease}) = 99\%$ (D, N)

$p(\text{if disease}, +) = 99\%$ $p(\text{if non-disease}, -) = 99\%$

$p(+|D)$ $p(-|N)$

find $p(D|+) = ?$

$$p(D|+) = \frac{p(D\cap +)}{p(+)}$$ by condition probabilty.

$$p(D|+) = \frac{p(+|D)\cdot p(D)}{p(+)}$$ by Bayes. inference.

$$= \frac{p(+|\theta)\cdot p(\theta)}{p(+)}$$ $p(D) = \pi_0 = p(\text{disease}) = 99\%$

$p(+) = p(+|D)\cdot p(D) + p(+|N)\cdot p(N) \longrightarrow$ law of total

prior prob dis: $\pi_{old} = \pi_0 = \pi_0(\theta)$

posterior prob dist : $\pi_1(\theta) = p(\theta|\text{data})$

$$\rightarrow \underbrace{\pi_1(\theta)}_{\text{posterior}} = \frac{p(\text{data}|\theta)\cdot \pi_0(\theta)}{p(\text{data})}$$ $\longrightarrow$ prior

Example 4: Bayesian Inference

Let $y_1, y_2, \ldots, y_n \sim f(y|\theta)$ denote observed iid data. Suppose we want to estimate θ.

The frequentist perspective uses the maximum likelihood estimator:

$$\hat{\theta}_{\text{MLE}} = \operatorname{argmax} L(\theta|y_1, \ldots, y_n) = \operatorname{argmax} \prod_{i=1}^{n} f(y_i|\theta).$$

The Bayesian perspective:

- θ is a random variable, with prior distribution $\pi(\theta)$ (i.e., the marginal distribution of θ).
- The posterior distribution of θ given $y_1, y_2, \ldots, y_n$ is written as

posterior

porior

$$p(\theta|y_1, \ldots, y_n) = \frac{p(\theta, y_1, \ldots, y_n)}{p(y_1, \ldots, y_n)} = \frac{\pi(\theta) \prod_{i=1}^{n} f(y_i|\theta)}{Z(\boldsymbol{y}) \text{ (fixed)}},$$

so $p(\theta|\boldsymbol{y}) \propto \pi(\theta) \prod_{i=1}^{n} f(y_i|\theta)$.

$$= \frac{p(\text{data}|\theta)}{p(\text{data})} \pi(\theta)$$

Example 4: Bayesian Inference

The posterior distribution of θ is $\boxed{p(\theta|\boldsymbol{y}) \propto \pi(\theta) \prod_{i=1}^{n} f(y_i|\theta).}$

The Bayesian estimator for θ is the posterior mean

$$E(\theta|\boldsymbol{y}) = \int \underline{\theta}\,\underline{p(\theta|\boldsymbol{y})}\,\mathrm{d}\theta.$$

A Monte Carlo approach to approximate $E(\theta|\boldsymbol{y})$:

1. Generate $\theta^{(1)}, \theta^{(2)}, \ldots, \theta^{(n)} \overset{\text{iid}}{\sim} p(\theta|\boldsymbol{y})$. find θ to max $p(\theta|y)$

   $p(\theta_1|y)$ $p(\theta_2|y)$ . . . . .
2. Compute $\hat{\theta}_B = \frac{1}{n}\sum_{i=1}^{n}\theta^{(i)}$. (mean) $\frac{\theta_1 + \cdots + \theta_n}{n}$

Hard Part: How to sample from the posterior distribution $p(\theta|\boldsymbol{y})$?

Spoilers: Markov Chain Monte Carlo (MCMC)!