ВЫЧИСЛИТЕЛЬНЫЕ МЕТОДЫ · ЛЕКЦИЯ 5

Основы метода Монте-Карло

Случайная выборка, интегрированиеи случайное блуждание

Лю Шисян

Кафедра теплофизики (Э6)
МГТУ им. Н.Э. Баумана
Москва / 2026

Место этой лекции в курсе

Лекция 5 — Основы метода Монте-Карло

Метод Монте-Карло использует случайную выборку и статистическое усреднение для вычисления детерминированных величин.

После лекции вы сможете:

- генерировать и анализировать случайные выборки;
- вычислять интегралы методом Монте-Карло;
- объяснять, как точность Monte Carlo зависит от числа выборок;
- объяснять связь случайного блуждания и диффузии.

План

1 Основы метода Монте-Карло

2 Интегрирование методом Монте-Карло

3 Случайное блуждание и диффузия

4 Практика и итоги

TOC

1

Основы метода Монте-Карло

Краткая история метода Монте-Карло

Игла Бюффона
случайный эксперимент для оценки математической величины

Los Alamos
S. Ulam, J. von Neumann
случайное моделирование физических процессов

Metropolis & Ulam
статья
“The Monte Carlo Method”

Почему “Monte Carlo”?

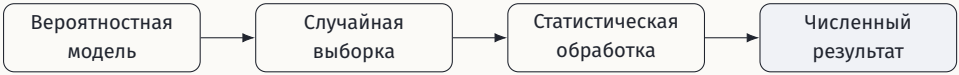
Название связано с Монте-Карло в Монако, известным своими казино: **случайность стала частью вычислительного алгоритма**.

С появлением электронных компьютеров случайные эксперименты превратились в практический численный метод.

Что такое метод Монте-Карло?

Определение

Методы Монте-Карло — это класс численных методов, в которых искомая величина оценивается по результатам **случайной выборки**.



Случайность используется как вычислительный инструмент.

Почему случайный метод может дать точный результат?

Одна случайная реализация

Результат может сильно отличаться от среднего значения.

$$X_1 \neq \mathbb{E}[X].$$

Большая выборка

Среднее по многим независимым реализациям приближается к математическому ожиданию:

$$\bar{X}_N = \frac{1}{N} \sum_{i=1}^{N} X_i \to \mathbb{E}[X].$$

Главная идея: заменить сложное вычисление статистическим усреднением.

Классический пример: оценка числа π

Рассмотрим единичный квадрат и четверть единичного круга.

Площадь квадрата:

$$S_{\square} = 1.$$

Площадь четверти круга:

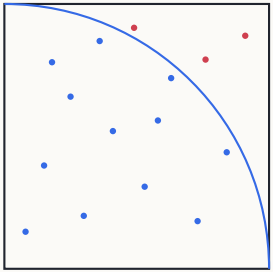
$$S_{\circ} = \frac{\pi}{4}.$$

Если точки равномерно распределены в квадрате, то

$$\frac{N_{\text{in}}}{N} \approx \frac{\pi}{4}.$$

Отсюда

$$\boxed{\pi \approx 4\frac{N_{\text{in}}}{N}}.$$

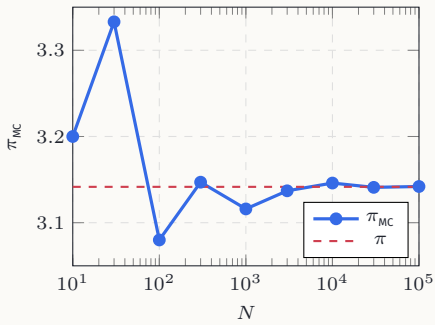


$x^2 + y^2 \leq 1$

Оценка числа π методом Монте-Карло

```python
import numpy as np

rng = np.random.default_rng(42)
N = 100_000

x = rng.random(N)
y = rng.random(N)

inside = x**2 + y**2 <= 1
pi_mc = 4 * inside.mean()

print(pi_mc)
```

- `rng.random(N)` генерирует N случайных чисел, равномерно распределённых на $[0, 1)$;
- логическая маска определяет попадание точки внутрь четверти круга;
- среднее булевой маски равно доле попаданий;



С ростом числа случайных точек оценка стабилизируется и приближается к точному значению π.

Случайная величина и распределение

Случайная величина

Случайная величина X — числовая величина, значение которой определяется исходом случайного эксперимента.

Примеры:

- результат броска кубика;
- координата случайной точки;
- время между случайными событиями;
- длина свободного пробега частицы.

Распределение

Распределение описывает, **какие значения может принимать** X и насколько они вероятны.
Для непрерывной величины используется плотность вероятности $p_X(x)$.
Вероятность попасть в интервал:

$$P(a < X < b) = \int_a^b p_X(x)\,dx.$$

Примеры записи:

$$X \sim U(a, b), \qquad X \sim \mathcal{N}(\mu, \sigma^2).$$

В Monte Carlo сначала задаётся распределение, а затем из него генерируется случайная выборка.

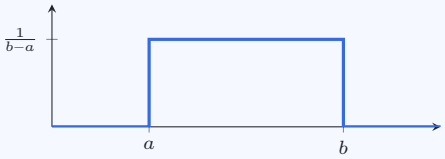
Основные распределения

Равномерное распределение

$$X \sim U(a, b)$$

Плотность вероятности:

$$p_X(x) = \frac{1}{b-a}, \qquad a \le x \le b.$$



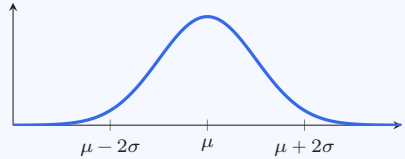
Все интервалы одинаковой длины внутри $[a, b]$ имеют одинаковую вероятность.

Нормальное распределение

$$X \sim \mathcal{N}(\mu, \sigma^2)$$

Плотность вероятности:

$$p_X(x) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left[-\frac{(x-\mu)^2}{2\sigma^2}\right].$$



μ задаёт центр распределения, а σ — характерный разброс.

Математическое ожидание и дисперсия

Математическое ожидание

Среднее значение случайной величины:

$$\mu = \mathbb{E}[X]$$

Для непрерывной величины:

$$\mathbb{E}[X] = \int_{-\infty}^{\infty} x\, p_X(x)\, dx.$$

В Monte Carlo оно оценивается по выборочному среднему:

$$\bar{X} = \frac{1}{N} \sum_{i=1}^{N} X_i$$

Дисперсия

Разброс относительно среднего:

$$\sigma^2 = \mathrm{Var}(X) = \mathbb{E}[(X - \mu)^2]$$

Стандартное отклонение:

$$\sigma = \sqrt{\mathrm{Var}(X)}$$

Большое σ:

сильные флуктуации.

Малое σ:

значения сосредоточены около μ.

Псевдослучайные числа в NumPy

Почему “псевдо”?

Компьютер использует **детерминированный алгоритм**:

$$\boxed{\text{seed} \longrightarrow u_1, u_2, u_3, \ldots}$$

Последовательность выглядит случайной, но полностью определяется начальным состоянием генератора.

```python
import numpy as np

rng = np.random.default_rng(42)

# Uniform distribution
u = rng.uniform(0, 1, 1000)

# Normal distribution
z = rng.normal(0, 1, 1000)

# Random integers
i = rng.integers(0, 10, 1000)
```

- одинаковый seed $\Rightarrow$ одинаковая последовательность;
- это позволяет воспроизводить вычисления;
- генератор должен иметь хорошие статистические свойства.

2

Интегрирование методом Монте-Карло

От интеграла к оценке Монте-Карло

Рассмотрим интеграл

$$I = \int_a^b f(x)\,dx.$$

Пусть

$$X \sim U(a,b), \qquad p_X(x) = \frac{1}{b-a}.$$

Тогда

$$\mathbb{E}[f(X)] = \int_a^b f(x) p_X(x)\,dx$$

и, следовательно,

$$I = (b-a)\mathbb{E}[f(X)]$$

В Monte Carlo математическое ожидание заменяем выборочным средним:

$$\mathbb{E}[f(X)] \approx \frac{1}{N}\sum_{i=1}^{N} f(x_i),$$

где

$$x_i \sim U(a,b).$$

Получаем:

$$I_N = (b-a)\frac{1}{N}\sum_{i=1}^{N} f(x_i)$$

Пример

Рассмотрим интеграл

$$I = \int_0^1 e^{-x^2}\, dx.$$

Шаг 1. Случайно выбираем точки

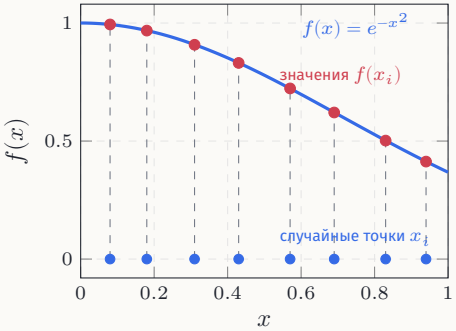
$$x_1, x_2, \dots, x_N \sim U(0,1).$$

Шаг 2. Для каждой точки вычисляем значение

$$f(x_i) = e^{-x_i^2}.$$

Шаг 3. Усредняем эти значения:

$$I_N = \frac{1}{N} \sum_{i=1}^{N} e^{-x_i^2}.$$



Так как длина интервала равна 1, дополнительный множитель $(b - a)$ здесь не появляется.

Интегрирование методом Монте-Карло на Python

```python
import numpy as np

rng = np.random.default_rng(42)

N = 100_000
a, b = 0.0, 1.0

x = rng.uniform(a, b, N)

f = np.exp(-x**2)

I_mc = (b - a) * f.mean()

print(I_mc)
```

Для данного seed:

$$I_{\text{MC}} \approx 0.7464.$$

Алгоритм

1. Генерируем

$$x_i \sim U(a, b)$$

2. Вычисляем

$$f(x_i)$$

3. Усредняем

$$\frac{1}{N} \sum_i f(x_i)$$

4. Умножаем на

$$b - a$$

Почему Monte Carlo полезен в высокой размерности?

Для d-мерной области Ω объёма V_Ω:

$$I = \int_\Omega f(\mathbf{x})\, d\mathbf{x}.$$

При равномерной выборке, получаем:

$$I_N = V_\Omega \frac{1}{N} \sum_{i=1}^{N} f(\mathbf{x}_i), \quad \mathbf{x}_i \sim U(\Omega)$$

Регулярная сетка

Если по каждой координате используется m точек, то $N_{\text{grid}} = m^d$ Например:

$$m = 100, \quad d = 6 \quad \Rightarrow \quad N_{\text{grid}} = 10^{12}.$$

Это проявление**проклятия размерности**.

Monte Carlo

В базовом Monte Carlo характерная статистическая ошибка:

$$\varepsilon_{\text{MC}} \propto N^{-1/2}$$

Показатель сходимости не содержит d явно. Поэтому Monte Carlo часто становится особенно привлекательным в многомерных задачах.

Многомерный интеграл на Python

Рассмотрим шестимерный интеграл

$$I = \int_{[0,1]^6} \exp\left(-\sum_{j=1}^{6} x_j^2\right) d\mathbf{x}.$$

```python
import numpy as np

rng = np.random.default_rng(42)
N = 200_000
d = 6

x = rng.random((N, d))  # N random points in 6D

f = np.exp(-np.sum(x**2, axis=1))

I_mc = f.mean()
SE = f.std(ddof=1) / np.sqrt(N)

print(I_mc, SE)
```

Оценка Monte Carlo

Так как $V_{[0,1]^6} = 1$, то

$$I_N = \frac{1}{N} \sum_{i=1}^{N} f(\mathbf{x}_i)$$

Для проверки:

$$I_{\text{exact}} \approx 0.17350.$$

Переход от 1D к 6D в коде в основном означает добавить новые координаты случайной точки.

3

Случайное блуждание и диффузия

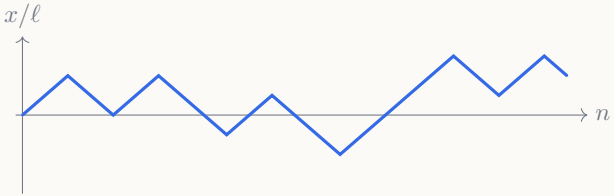
Одномерное случайное блуждание

Через каждый интервал времени Δt частица делает случайный шаг длины ℓ:

$$x_{n+1} = x_n + \Delta x_n,$$

где

$$\Delta x_n = \begin{cases} +\ell, & P = 1/2, \\ -\ell, & P = 1/2. \end{cases}$$



После N шагов:

$$x_N = \Delta x_1 + \Delta x_2 + \cdots + \Delta x_N = \sum_{i=1}^{N} \Delta x_i$$

Среднее и среднеквадратичное смещение

После N шагов

$$x_N = \sum_{i=1}^{N} \Delta x_i.$$

Среднее положение

Из-за симметрии шагов

$$\langle \Delta x_i \rangle = \frac{1}{2}\ell + \frac{1}{2}(-\ell) = 0.$$

Поэтому

$$\langle x_N \rangle = 0$$

Среднее положение ансамбля остаётся в начале координат.

Среднеквадратичное смещение

Для независимых шагов

$$\langle (\Delta x_i)^2 \rangle = \ell^2,$$

Поэтому

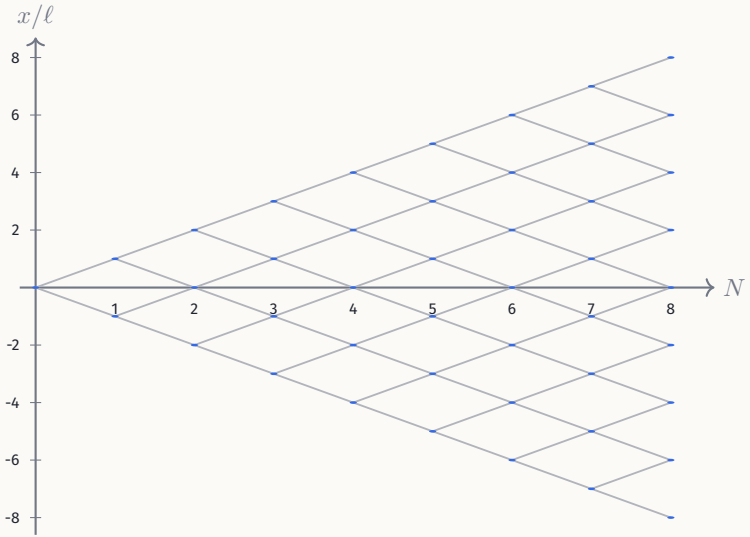
$$\langle x_N^2 \rangle = N\ell^2$$

и характерное расстояние

$$x_{\text{rms}} = \sqrt{\langle x_N^2 \rangle} = \ell\sqrt{N}$$

Среднее положение равно нулю, но ширина распределения растёт как $\sqrt{N}$.

Все возможные траектории случайного блуждания



Вычислим MSD для первых шагов

Пусть длина шага равна ℓ. Рассмотрим распределение положения после нескольких шагов.

$N = 1$

$$x = \begin{cases} -\ell, & P = 1/2, \\ +\ell, & P = 1/2. \end{cases}$$

Поэтому

$$\langle x^2 \rangle = \frac{1}{2}\ell^2 + \frac{1}{2}\ell^2 = \ell^2.$$

$N = 2$

x	-2ℓ	0	$+2\ell$
P	$1/4$	$1/2$	$1/4$

Отсюда

$$\langle x^2 \rangle = \frac{1}{4}(2\ell)^2 + \frac{1}{2}(0)^2 + \frac{1}{4}(2\ell)^2 = 2\ell^2.$$

$N = 3$

x	-3ℓ	$-\ell$	$+\ell$	$+3\ell$
P	$1/8$	$3/8$	$3/8$	$1/8$

Поэтому

$$\begin{aligned} \langle x^2 \rangle &= \frac{1}{8}(3\ell)^2 + \frac{3}{8}\ell^2 \\ &\quad + \frac{3}{8}\ell^2 + \frac{1}{8}(3\ell)^2 \\ &= 3\ell^2. \end{aligned}$$

MSD растёт линейно с числом шагов $\langle x_N^2 \rangle = N\ell^2$, а характерное расстояние растёт как $x_{rms} = \ell\sqrt{N}$.

От случайного блуждания к диффузии

Для одномерного случайного блуждания $\langle x^2 \rangle = N\ell^2$. Каждый шаг занимает время Δt, поэтому

$$t = N\Delta t.$$

Следовательно,

$$\langle x^2 \rangle = \frac{\ell^2}{\Delta t}\, t.$$

Для одномерной диффузии

$$\langle x^2 \rangle = 2Dt$$

Отсюда

$$D = \frac{\ell^2}{2\Delta t}$$

Отдельная траектория случайна, но распределение большого числа частиц эволюционирует закономерно.

Почему возникает уравнение диффузии?

Пусть $P(x,t)$ — вероятность найти частицу в положении x в момент времени t.
Чтобы после следующего шага оказаться в точке x, частица должна прийти либо слева, либо справа:

$$\boxed{P(x,t+\Delta t) = \frac{1}{2}P(x-\ell,t) + \frac{1}{2}P(x+\ell,t)}$$

Для малых ℓ и Δt это дискретное уравнение переходит в

$$\boxed{\frac{\partial P}{\partial t} = D\frac{\partial^2 P}{\partial x^2}}, \qquad \boxed{D = \frac{\ell^2}{2\Delta t}}$$

Уравнение диффузии — непрерывный предел случайного блуждания большого ансамбля частиц.

Случайное блуждание и диффузия на Python

```python
import numpy as np
import matplotlib.pyplot as plt

rng = np.random.default_rng(42)
M = 20_000      # particles
N = 500         # steps
ell = 1.0
dt = 1.0

steps = rng.choice([-ell, ell], size=(M, N))
x = np.cumsum(steps, axis=1)
msd = np.mean(x**2, axis=0)

t = np.arange(1, N + 1) * dt
D = ell**2 / (2 * dt)

plt.plot(t, msd, label="Random walk")
plt.plot(t, 2*D*t, "--", label="2Dt")
plt.xlabel("t")
plt.ylabel("MSD")
plt.legend()
```

Что проверяем?

Теория предсказывает:

$$\boxed{\mathrm{MSD} = 2Dt}$$

Следовательно, график MSD должен быть линейным по времени.

Физический смысл

Одна траектория выглядит случайной.
Но среднее по

$$M \gg 1$$

траекториям даёт детерминированный закон.

4

Практика и итоги

Типовая структура алгоритма Монте-Карло

Общая идея всегда одна: случайная выборка → вычисление → усреднение.

Практическое задание

Задача 1: число π

Вычислить

$$\pi_N = 4\frac{N_{\text{in}}}{N}$$

для $N = 10^2,\ 10^3,\ 10^4,\ 10^5,\ 10^6$.

Построить зависимость

$$|\pi_N - \pi| \quad \text{от} \quad N.$$

Задача 2: случайное блуждание

Смоделировать большое число одномерных траекторий и проверить

$$\boxed{\langle x^2 \rangle \propto N.}$$

Дополнительно: построить распределение положения частиц после заданного числа шагов.

Итоги

1. Интегрирование

Интеграл можно представить как математическое ожидание и оценить по случайной выборке:

$$I \approx V_\Omega \frac{1}{N} \sum_{i=1}^{N} f(\mathbf{x}_i).$$

2. Случайное блуждание

Случайные траектории большого числа частиц приводят к диффузионному поведению:

$$\langle x^2 \rangle = N\ell^2 = 2Dt.$$

Следующая лекция: перейдём от случайного блуждания к моделированию переноса частиц и уравнению Больцмана.