БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ФАКУЛЬТЕТ ПРИКЛАДНОЙ МАТЕМАТИКИ И ИНФОРМАТИКИ

ОТЧЕТ

по индивидуальному заданию
учебной дисциплины «Математическое моделирование»

Лабораторная работа № 5

Метод Монте-Карло

Выполнил:
Босько Антон Павлович, 9 группа

Преподаватель:
Горшунова Екатерина Сергеевна

Минск, 2025

Вариант:

$$I = \iint_{x^2+y^2<1} \ln\left(\frac{1}{\sqrt{x^2+y^2}}\right) dx\, dy$$

---

1. Теоретические сведения и упрощение интеграла

Подставим полярные координаты $x = r\cos\theta$, $y = r\sin\theta$. Тогда $x^2 + y^2 = r^2$, подинтегральная функция

$$\ln\left(\frac{1}{\sqrt{x^2+y^2}}\right) = \ln\left(\frac{1}{r}\right) = -\ln r,$$

а элемент площади $dx\, dy = r\, dr\, d\theta$. Пределы: $\theta \in [0,2\pi]$, $r \in [0,1]$. Получаем

$$I = \int_0^{2\pi} \int_0^1 (-\ln r)\, r\, dr\, d\theta = 2\pi \int_0^1 (-r\ln r)\, dr.$$

Вычислим однократный интеграл. Через интегрирование по частям или известную формулу:

$$\int_0^1 r\ln r\, dr = -\frac{1}{4},$$

откуда

$$\int_0^1 -r\ln r\, dr = \frac{1}{4}.$$

Следовательно

$$\boxed{I = 2\pi \cdot \frac{1}{4} = \frac{\pi}{2}}.$$

Точное значение: $I = \frac{\pi}{2} \approx 1.57079632679$.

---

2. Метод Монте–Карло

2.1 Равномерная выборка по диску

Чтобы сгенерировать nточек равномерно в единичном диске:

- генерируем $\theta_i \sim U(0,2\pi)$,
- генерируем $u_i \sim U(0,1)$,
- задаём $r_i = \sqrt{u_i}$(чтобы плотность радиуса была пропорциональна r),
- точки: $x_i = r_i \cos\ \theta_i,\ y_i = r_i \sin\ \theta_i$.

Поскольку плотность равномерной точки в диске равна $1/\pi$на области D, интеграл можно записать через математическое ожидание:

$$I = \iint_D f(x,y)\,dx\,dy = \pi\,\mathbb{E}[f(X,Y)], f(x,y) = -\ln\ r.$$

Оценка методом Монте–Карло:

$$\hat{I}_n = \pi \cdot \frac{1}{n}\sum_{i=1}^{n} f(\,x_i, y_i) = \pi \cdot \frac{1}{n}\sum_{i=1}^{n}(-\ln\ r_i).$$

Выборочная оценка дисперсии и стандартная ошибка:

$$\widehat{\mathrm{Var}}(f) = \frac{1}{n-1}\sum_{i=1}^{n}(\,f_i - \bar{f})^2, \widehat{\mathrm{SE}}(\hat{I}_n) = \pi\sqrt{\frac{\widehat{\mathrm{Var}}(f)}{n}}.$$

2.2 Повторения и оценка сходимости

Для надёжной оценки зависимости точности от nделаем Rнезависимых повторений для каждого n. Для фиксированного nвычисляем:

- среднюю оценку $\bar{\hat{I}}(n)$по повторениям,
- среднюю SE по повторениям,
- среднюю абсолютную ошибку $\frac{1}{R}\sum_{r=1}^{R} |\,\hat{I}^{(r)} - I_{\text{точн}}\,|$.

Ожидаемая скорость сходимости при доминирующей случайной ошибке: error $\propto 1/\sqrt{n}$.

---

3. Код

```python
#!/usr/bin/env python3

import math, time
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

N_LIST = [100, 500, 1000, 5000, 10000, 50000]
REPEATS = 50
SEED = 123456

# exact value
I_exact = math.pi / 2.0

def sample_uniform_disk(n, rng):
    u = rng.random(n)
    r = np.sqrt(u)
    theta = rng.random(n) * 2 * math.pi
    return r

def estimate_once_uniform(n, rng):
    r = sample_uniform_disk(n, rng)
    f = -np.log(r)
    mean_f = f.mean()
    var_f = f.var(ddof=1)
    se = math.sqrt(var_f / n)
    I_hat = math.pi * mean_f
    se_I = math.pi * se
    return I_hat, se_I

def run_experiment(n_list, repeats, seed):
    rng_master = np.random.default_rng(seed)
    rows = []
    for n in n_list:
        ests = np.zeros(repeats)
        ses = np.zeros(repeats)
        for r in range(repeats):
            sub_seed = rng_master.integers(0, 2**31-1)
            rng = np.random.default_rng(sub_seed)
            Ihat, seI = estimate_once_uniform(n, rng)
            ests[r] = Ihat
            ses[r] = seI
        mean_est = ests.mean()
        mean_se = ses.mean()
        mean_abs_err = np.mean(np.abs(ests - I_exact))
        rel_err = mean_abs_err / abs(I_exact)
        rows.append({
            "method": "uniform_disk",
            "n": n,
            "mean_est": mean_est,
            "mean_se": mean_se,
            "mean_abs_err": mean_abs_err,
            "rel_err": rel_err
        })
        print(f"n={n}: mean_est={mean_est:.6f},
mean_abs_err={mean_abs_err:.6f}, mean_se={mean_se:.6f}")
    df = pd.DataFrame(rows)
    return df

def plot_results(df, png_out="error_vs_n.png"):
    pivot = df.pivot(index="n", columns="method", values="mean_abs_err")
```

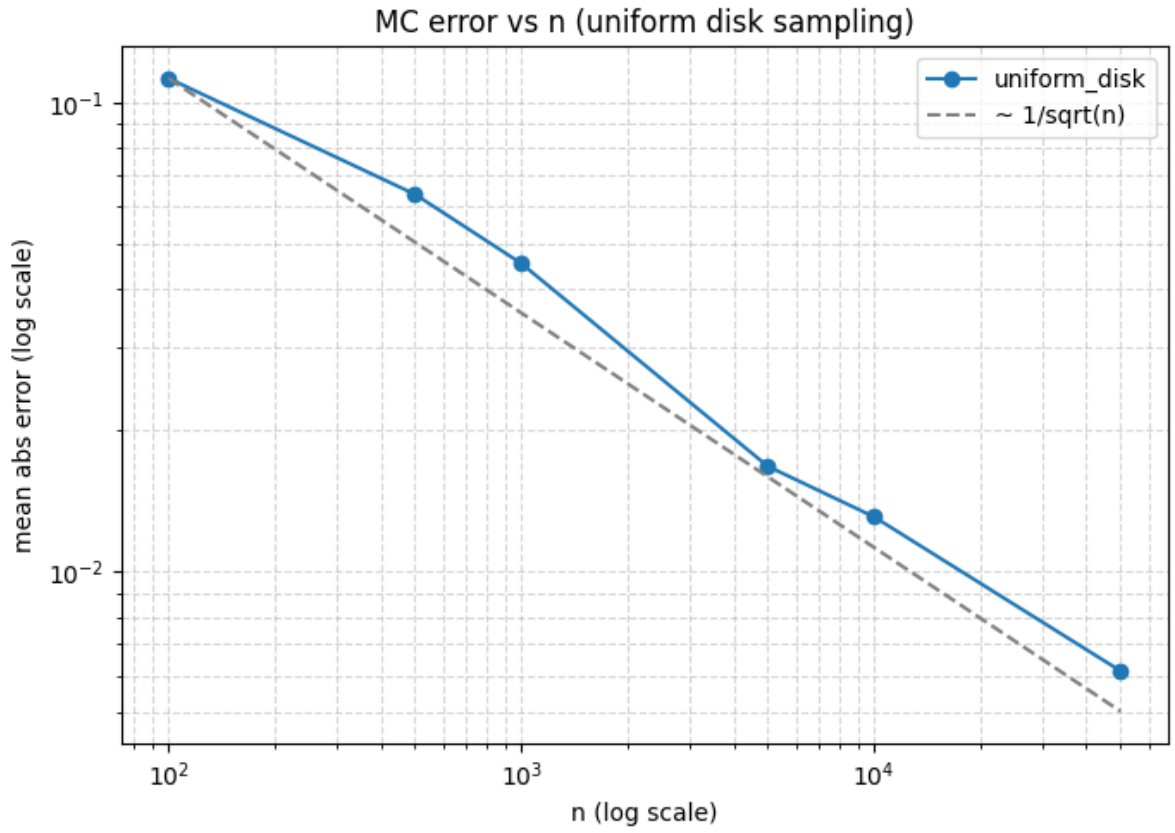
```
    plt.figure(figsize=(7,5))
    for col in pivot.columns:
        plt.plot(pivot.index, pivot[col], marker='o', label=col)
    ns = np.array(sorted(pivot.index))
    ref = pivot.max(axis=1).iloc[0] * (math.sqrt(ns[0]) / np.sqrt(ns))
    plt.plot(ns, ref, linestyle='--', color='gray', label='~ 1/sqrt(n)')
    plt.xscale('log'); plt.yscale('log')
    plt.xlabel('n (log scale)'); plt.ylabel('mean abs error (log scale)')
    plt.title('MC error vs n (uniform disk sampling)')
    plt.legend(); plt.grid(True, which='both', ls='--', alpha=0.5)
    plt.tight_layout(); plt.savefig(png_out)
    print(f"Saved plot {png_out}")

def main():
    t0 = time.time()
    print("Exact I =", I_exact)
    df = run_experiment(N_LIST, REPEATS, SEED)
    df.to_csv("results.csv", index=False)
    print("Saved results.csv")
    plot_results(df, "error_vs_n.png")
    print("Done in {:.1f}s".format(time.time()-t0))

if __name__ == "__main__":
    main()
```

4. Результаты

График `error_vs_n.png` (логарифмический) демонстрирует поведение ошибки ~ $1/\sqrt{n}$.



5. Выводы

1. Аналитически вычисленный интеграл равен $I = \frac{\pi}{2}$.
2. Метод Монте–Карло с равномерной выборкой по диску даёт оценки, сходящиеся к $\frac{\pi}{2}$.
3. Случайная погрешность уменьшается порядка $O(1/\sqrt{n})$, что подтверждается эмпирически.
4. Для снижения дисперсии возможны техники variance reduction (importance sampling по радиусу, стратификация по кольцам, antithetic variates) — при необходимости их можно добавить.