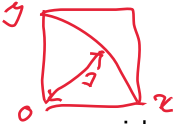
Monte Carlo method

モデル化とシミュレーション特論
2023 年度前期
佐賀大学理工学研究科　只木進一

Monte Carlo method

- General framework for simulations using random numbers
- Numerical integrals
- Modeling stochastic processes
- Approximated solutions for difficult combinatorial optimizations
- sample code in the following URL
  `https://github.com/modeling-and-simulation-mc-saga/MonteCarlo`

Estimation of π



- Generate 2-dimensional random numbers in a square with unit length sides.

$$(x, y) \in [0, 1) \times [0, 1) \tag{2.1}$$

- Count events entering an arc with unit length radius.

$$0 \leq \left(x^2 + y^2\right)^{1/2} < 1 \tag{2.2}$$

- The probability of events entering the arc is $\pi/4$

- $P_N(m)$: Probability of m out of N points falling into the arc.

$$P_N(m) = \binom{N}{m} p^m (1-p)^{N-m}, \quad p = \frac{\pi}{4} \tag{2.3}$$

- Probability generating function for $P_N(m)$

$$G(z) = \sum_{m=0}^{N} P_N(m) z^m = (pz + 1 - p)^N \tag{2.4}$$

$$G'(z) = \sum_{m=0}^{N} m P_N(m) z^{m-1} = Np\left(pz + 1 - p\right)^{N-1} \tag{2.5}$$

$$\langle m \rangle = G'(1) = Np \tag{2.6}$$

$$\begin{aligned} G''(z) &= \sum_{m=0}^{N} m(m-1) P_N(m) z^{m-2} \\ &= N(N-1)p^2\left(pz + 1 - p\right)^{N-2} \end{aligned} \tag{2.7}$$

$$G''(1) = \left\langle m^2 \right\rangle - \langle m \rangle = N(N-1)p^2 \tag{2.8}$$

$$\sigma^2 = \left\langle m^2 \right\rangle - \langle m \rangle^2 = Np(1-p) \tag{2.9}$$

Simulation

$$\frac{\langle m \rangle_{\mathrm{exp}}}{N} \sim \frac{\pi}{4} \tag{2.10}$$

$$\frac{\sigma}{\langle m \rangle} = \frac{1}{N^{1/2}} \left(\frac{1-p}{p} \right)^{1/2} \tag{2.11}$$

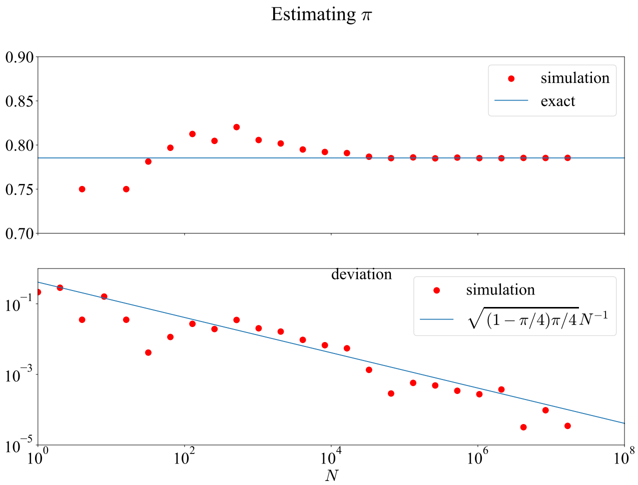
Deviation

$$\left| \frac{\langle m \rangle_{\mathrm{exp}}}{N} - \frac{\pi}{4} \right| \tag{2.12}$$

will reduce with $N^{-1/2}$

```java
public double addOne() {
    all++;
    double x = myRandom.nextDouble();
    double y = myRandom.nextDouble();
    double r = Math.sqrt(x * x + y * y);
    if (r <= 1.) {
        in++;
    }
    return (double) in / all;
}
```

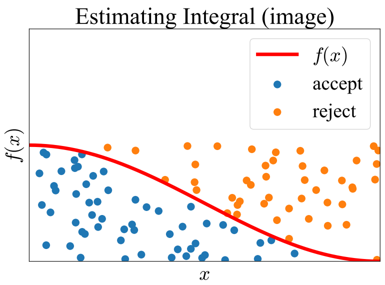
estimatingPi/Pi.java



Estimating π
0.90
0.85
0.80
0.75
0.70
simulation
exact
deviation
simulation
$\sqrt{(1-\pi/4)\pi/4}\,N^{-1}$
10^{-1}
10^{-3}
10^{-5}
10^{0}
10^{2}
10^{4}
10^{6}
10^{8}
N

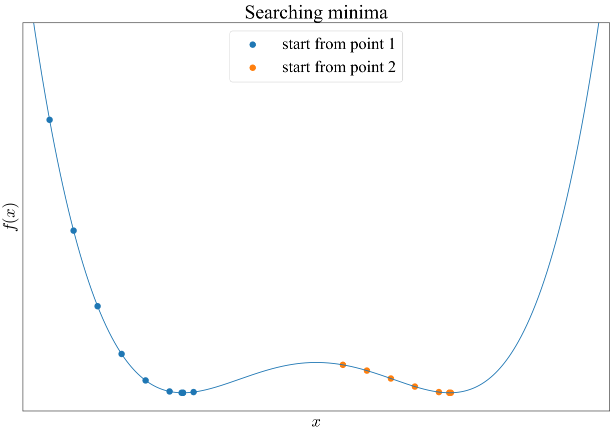
Estimating Integrals

- Function $f(x) \geq 0$ defined in $x \in [a, b)$
- $m > \max f(x)$
- Generate n 2-dimensional random numbers in $[a, b) \times [0, m)$
- k random numbers fallen inside the area under $f(x)$
- The integral is $(k/n) \times m \times (b - a)$



Find minima of $f(x)$

- There is only one minimum of $f(x)$.
  - Starting from an arbitrary x.
  - Change x slightly for reducing $f(x)$
- There are some minima of $f(x)$.
  - Trials with randomly selected starting points.
- For cases of huge number of minima of $f(x)$?


Searching minima
start from point 1
start from point 2
$f(x)$
x

Random Spins

- n variables $s_i = \pm 1$
- Interactions J_{ij} ($J_{ij} = J_{ji}, J_{ii} = 0$)
  - Positive and negative random values
- Minimize

$$E = -\sum_{ij} J_{ij} s_i s_j \tag{5.1}$$

  - Split s_i into two groups
  - Example of bi-partitioning problems

Monte Carlo method for spin systems

- Select randomly one s_i of spins, and try to flip it.

$$s_i \to s_i + \Delta x_i \tag{5.2}$$

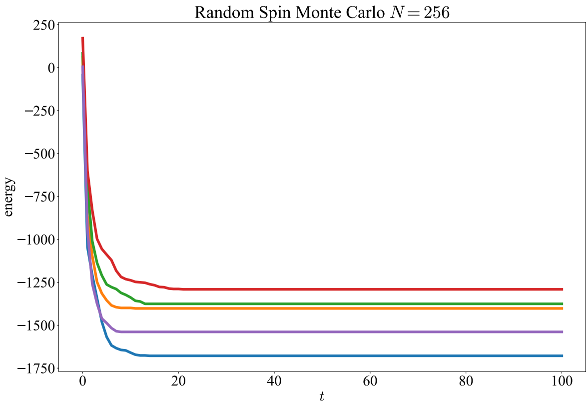
- Evaluate energy change by flipping the spin s_i

$$\Delta E = -2 \sum_j J_{ij} s_j \Delta x_i \tag{5.3}$$

- If $\Delta E < 0$, then employ the new value for s_i
- 1 Monte Carlo step : n trials for changing spins

```java
public double oneStep() {
    int k = random.nextInt(n); //Random selection of a spin
    int ds = -2 * s[k]; //spin flip
    //energy
    double de = 0.;
    for (int j = 0; j < n; j++) {
        de += -2 * J[k][j] * s[j] * ds;
    }
    //the spin flips if energy decreases
    if (de < 0) {
        energy += de;
        s[k] += ds;
    }
    return de;
}
```

`simpleMonteCarlo/SpinSystem.java`



- 5 different initial states lead to different states.
- This shows the existence of many energy minima
- How to find the minimum?

DLA: Diffusion Limited Aggregation

- At the beginning, there is a seed of a cluster at the center.
- particles are diffused and adhered to the cluster if the particle contacts to the cluster.

Simplest Simulation

- 2-dimensional lattice
- At the beginning, there is a seed of a cluster at the center.
- A particle enter the system, and do simple random walk.
  - one of four direction is randomly selected
  - if the position is next to the cluster, the particle is adhered.
  - A particle is adhered to the cluster, another particle enters the system.

https://github.com/modeling-and-simulation-mc-saga/DLA

```java
public Point inject() {
    Point p = new Point(0, 0);
    setNewPosition(p);//Place a particle on the boundary

    while (true) {//Repeat until absorption
        if (isAdjacent(p)) {//The particle is absorbed
            cells[p.x][p.y] = 1;
            return p;
        }
        //Random displacement
        int k = random.nextInt(4);
        switch (k) {
            case 0:
                p.translate(1, 0);
                break;
            case 1:
                p.translate(0, 1);
                break;
            case 2:
                p.translate(-1, 0);
                break;
            default:
                p.translate(0, -1);
                break;
        }
        adjustPosition(p);
    }
}
```

30,000 particles