

方法一：蒲丰投针

```
1. function pi_estimate = buffons_needle_simulation(n)
2. % n: 模拟次数
3. a = 1; % 平行线之间的距离
4. l = 0.5; % 针的长度
5. m = 0; % 与平行线相交的次数
6.
7. for i = 1:n
8.     % 随机位置和角度
9.     x = rand() * (a / 2); %  $\theta \sim a/2$ 
10.    theta = rand() * (pi / 2); %  $\theta \sim \pi/2$ 
11.
12.    % 判断是否与平行线相交
13.    if x <= l/2 * sin(theta)
14.        m = m + 1;
15.    end
16. end
17.
18. % 估计  $\pi$ 
19. pi_estimate = (2 * l * n) / (a * m);
20. end
21.
22. % 测试
23. pi_estimate = buffons_needle_simulation(100000000);
24. disp(pi_estimate);
25.
26. % 输出:
27. >> hw1
28.     3.1415
```

方法二：蒙特卡罗方法

```
1. function pi_estimate = montecarlo_simulation(n)
2. % n: 模拟次数
3. inside = 0;
4.
5. for i = 1:n
6.     % 在  $[-1, 1] \times [-1, 1]$  的正方形内随机生成点
7.     x = 2 * rand() - 1;
8.     y = 2 * rand() - 1;
9.
10.    % 判断是否在单位圆内
```

```
11.     if x^2 + y^2 <= 1
12.         inside = inside + 1;
13.     end
14. end
15.
16. % 估计  $\pi$ 
17. pi_estimate = 4 * inside / n;
18. end
19.
20. % 测试
21. pi_estimate = montecarlo_simulation(100000000);
22. disp(pi_estimate);
23.
24. % 输出
25. > hw1
26.     3.1418
```