Nama: Muhammad Hanif

NIM: 202210370311265

Kelas: Pemodelan dan Simulasi Data B

Laporan Tugas 2 Simulasi Monte Carlo

I. Pendahuluan

Simulasi Monte Carlo adalah model probabilistik yang dapat mencakup elemen ketidakpastian atau acak dalam prediksinya. Simulasi Monte Carlo memberikan beberapa kemungkinan hasil dan probabilitas dari masing-masing dari kumpulan besar sampel data acak. Teknik ini sangat berguna dalam situasi di mana perhitungan analitis sulit dilakukan atau terlalu kompleks untuk diselesaikan secara langsung. Dengan menggunakan pendekatan ini, hasil yang diperoleh tidak hanya memberikan satu nilai estimasi tunggal, tetapi juga menunjukkan bagaimana hasil tersebut dapat bervariasi tergantung pada faktor acak yang terlibat dalam simulasi. Oleh karena itu, Monte Carlo sering digunakan dalam berbagai bidang seperti keuangan, fisika, teknik, dan kecerdasan buatan untuk memodelkan ketidakpastian dan mengevaluasi berbagai kemungkinan skenario.

Pada studi kasus tugas kedua, diperintahkan untuk membuat program Python dalam mencari nilai π menggunakan metode Monte Carlo. Metode ini bekerja dengan menghasilkan titik-titik acak dalam suatu area tertentu dan menghitung proporsi titik yang berada di dalam lingkaran dibandingkan dengan total titik yang dihasilkan. Dengan menggunakan pendekatan probabilistik ini, kita dapat memperkirakan nilai π secara empiris.

II. Pembahasan

Simulasi Monte Carlo dibuat dengan bahasa pemrograman Python dengan library **NumPy** dan **Matplotlib**, yang bertujuan mencari nilai **π (phi)** secara numerik menggunakan metode probabilistik. Metode ini bekerja dengan cara menempatkan titik-titik acak dalam sebuah persegi dan menentukan berapa banyak titik yang jatuh di dalam lingkaran. Dari rasio antara titik yang berada dalam lingkaran terhadap jumlah total titik, kita dapat mengestimasi nilai π menggunakan rumus dasar dari luas lingkaran dan luas persegi yang mengelilinginya.

- **Konsep dan Rumus yang digunakan**:
  Simulasi Monte Carlo bekerja dengan menempatkan titik-titik secara acak dalam suatu persegi yang mengandung lingkaran. Dengan mengamati berapa banyak titik yang jatuh dalam lingkaran dibandingkan jumlah total titik, kita bisa mendapatkan perkiraan nilai π. Pada simulasi ini, digunakan hubungan antara luas lingkaran dan luas persegi sebagai dasar estimasi nilai π
  - Lingkaran dengan jari-jari r = 1:
  - Persegi dengan panjang sisi 2r = 2, sehingga luasnya adalah:

    $$A_S = (2r)^2 = 4$$

    Sedangkan luas lingkaran dihitung dengan rumus:

$$Ac = \pi r^2 = \pi \times 1^2 = \pi$$

Karena titik-titik dihasilkan secara acak di dalam persegi, maka probabilitas titik yang jatuh ke dalam lingkaran akan sebanding dengan luasnya:

$$P = \frac{A_C}{As} = \frac{\pi}{4}$$

- A_C: Luas Lingkaran
- As: Luas Persegi

Sehingga jumlah titik masuk ke dalam lingkaran dapat dihitung dengan rumus:

$$N_c \approx \frac{\pi}{4} \times N_{total}$$

Dari hubungan tersebut, nilai π dapat diperoleh dengan rumus:

$$\pi \approx 4x \frac{N_c}{N_{total}}$$

- N_c adalah jumlah titik yang jatuh di dalam lingkaran
- N_{total} adalah total titik yang dihasilkan dalam simulasi

Seiring bertambahnya jumlah titik yang digunakan dalam simulasi, nilai estimasi $\boldsymbol{\pi}$ diharapkan semakin mendekati nilai sebenarnya.

Pada simulasi ini, nilai N atau jumlah total titik yang digunakan dalam simulasi, diperoleh dari jumlah titik yang dihasilkan secara acak dalam persegi yang mengelilingi lingkaran.
Secara spesifik:

- N_c adalah jumlah titik yang telah dibuat pada iterasi simulasi.
- N_{total} adalah jumlah titik yang jatuh ke dalam lingkaran, dihitung berdasarkan persamaan:

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

(x, y) merupakan koordinat titik yang dihasilkan secara acak dalam rentang [-1, 1] untuk sumbu x dan y.

Nilai N bertambah setiap kali program menambahkan titik baru dalam simulasi. Pada iterasi pertama, N=1, kemudian terus bertambah hingga mencapai jumlah total titik yang diinginkan, misalnya 100.000 titik.

❖ Struktur dan Penjelasan Code

source code: https://github.com/muhhanif27/PSD_TEORI/tree/tugas2

1. **Fungsi plot_construction()**
   Fungsi ini berperan untuk membuat dua subplot yang akan digunakan untuk memvisualisasikan hasil simulasi:
   - Subplot pertama (kiri):
     - Menampilkan titik-titik acak yang dihasilkan dalam ruang persegi dengan rentang [-1,1] pada sumbu x dan y.
     - Titik merah menandakan titik yang berada di area lingkaran, sedangkan titik biru merupakan titik yang berada di luar lingkaran
     - Penggunaan set_aspect('equal') memastikan bahwa skala tampilan antara sumbu **x** dan **y** tetap proporsional.

   - Subplot kedua (kanan):
     - Menampilkan perubahan estimasi nilai π seiring bertambahnya jumlah titik yang dihasilkan.
     - Sumbu x merepresentasikan jumlah titik yang telah diuji, dengan rentang 0 hingga 100.000.
     - Sumbu y merepresentasikan estimasi nilai π, dengan rentang 3.10 hingga 3.20 untuk mempermudah analisis perubahan nilai.
     - Garis horizontal merah putus-putus ditambahkan sebagai referensi nilai π yang sebenarnya.

2. **Inisialisasi Variabel**
   Sejumlah variabel diinisialisasi untuk menyimpan data yang akan digunakan dalam iterasi Monte Carlo:
   - N_samples = np.arange(1, 100_001)
     - Ini adalah array yang berisi angka dari 1 hingga 100.000, menunjukkan jumlah total titik yang akan diuji dalam simulasi.
   - pi_counter = 0
     - Variabel ini digunakan untuk menghitung jumlah titik yang jatuh di dalam lingkaran.
   - dots_x, dots_y, dots_color
     - List ini digunakan untuk menyimpan koordinat x dan y titik-titik yang dihasilkan serta warnanya (merah atau biru).
   - pi_array
     - List ini menyimpan nilai estimasi π untuk setiap iterasi, sehingga bisa diplot sebagai grafik perubahan estimasi π seiring bertambahnya jumlah titik.

3. **Simulasi Monte Carlo**

Proses utama dari simulasi dilakukan dalam loop for yang berjalan sebanyak 100.000 kali. Pada setiap iterasi, langkah-langkah berikut dilakukan:

- Menghasilkan titik acak
  - Titik acak **(x, y)** dihasilkan dalam rentang **[-1, 1]** untuk memastikan bahwa titik tersebut berada dalam persegi yang mengelilingi lingkaran.
  - np.random.uniform(-1, 1) digunakan untuk mengambil nilai acak yang terdistribusi secara uniform.

- Menentukan apakah titik berada dalam lingkaran
  - Sebuah titik dianggap berada dalam lingkaran jika memenuhi persamaan:

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

  - Jika kondisi ini terpenuhi, titik tersebut berada di dalam lingkaran, sehingga pi_counter bertambah satu dan titik diberi warna merah.
  - Jika tidak, titik tersebut dianggap berada di luar lingkaran dan diberi warna biru.

- Menghitung estimasi π
  - Dengan menggunakan prinsip probabilitas, estimasi nilai $\boldsymbol{\pi}$ dihitung dengan rumus:

$$\pi \approx 4x \frac{N_c}{N_{total}}$$

  - N_c adalah jumlah titik yang jatuh di dalam lingkaran (ditandai dengan warna merah).
  - N_{total} adalah jumlah titik yang telah dihasilkan sejauh ini dalam simulasi.

- Update Grafik
  - Setiap 5000 iterasi, tampilan grafik diperbarui agar simulasi lebih interaktif dan pengguna dapat melihat perkembangan estimasi π secara real-time.
  - Grafik pertama diperbarui untuk menampilkan titik-titik yang sudah dihasilkan.
  - Grafik kedua diperbarui untuk menunjukkan tren estimasi nilai π dari waktu ke waktu.
  - Informasi tambahan juga ditampilkan di terminal, seperti jumlah titik merah, jumlah titik biru, nilai estimasi π, selisih dari π sebenarnya, dan waktu eksekusi sejauh ini.

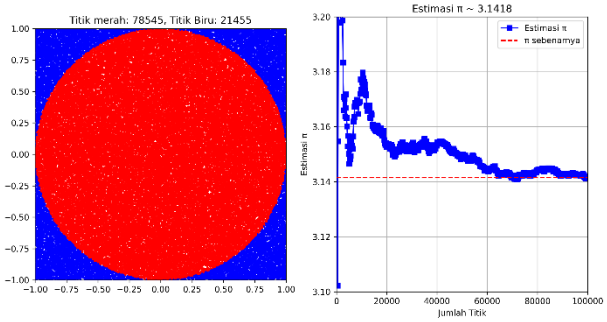
4. **Visualisasi dan Penyimpanan Hasil**

Setelah semua titik telah dihasilkan, kode akan menampilkan hasil akhir dari simulasi Monte Carlo:

- Grafik titik-titik dalam persegi akan menunjukkan distribusi titik merah dan biru.
- Grafik estimasi π akan menunjukkan bagaimana estimasi nilai π berubah seiring bertambahnya jumlah titik.
- Informasi di terminal akan menampilkan hasil akhir dari simulasi, termasuk:
  - Jumlah titik yang digunakan
  - Estimasi akhir nilai π
  - Selisih atau error dari nilai π sebenarnya
  - Waktu eksekusi total yang dibutuhkan untuk menjalankan simulasi

Terakhir, grafik akan disimpan dalam format PNG menggunakan plt.savefig("Tugas2/estimasi_pi.png", dpi=300) agar hasilnya bisa digunakan untuk laporan atau analisis lebih lanjut. Setelah itu, grafik akan ditampilkan di layar menggunakan plt.show().

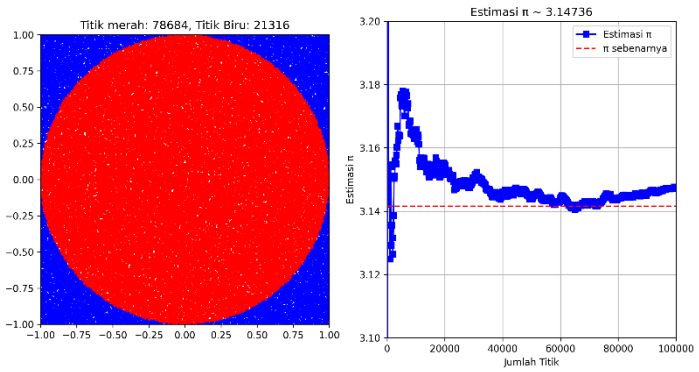
III. Hasil dan Kesimpulan




```
Jumlah titik merah: 3933, Jumlah titik biru: 1067, Jumlah titik: 5000, Estimasi π: 3.1464, Error: 0.004807, Waktu: 0.4367 detik
Jumlah titik merah: 7942, Jumlah titik biru: 2058, Jumlah titik: 10000, Estimasi π: 3.1768, Error: 0.035207, Waktu: 1.2116 detik
Jumlah titik merah: 11847, Jumlah titik biru: 3153, Jumlah titik: 15000, Estimasi π: 3.1592, Error: 0.017607, Waktu: 1.5026 detik
Jumlah titik merah: 15763, Jumlah titik biru: 4237, Jumlah titik: 20000, Estimasi π: 3.1526, Error: 0.011007, Waktu: 1.6103 detik
Jumlah titik merah: 19693, Jumlah titik biru: 5307, Jumlah titik: 25000, Estimasi π: 3.15088, Error: 0.009287, Waktu: 1.9887 detik
Jumlah titik merah: 23652, Jumlah titik biru: 6348, Jumlah titik: 30000, Estimasi π: 3.1536, Error: 0.012007, Waktu: 2.4363 detik
Jumlah titik merah: 27612, Jumlah titik biru: 7388, Jumlah titik: 35000, Estimasi π: 3.1556571428571427, Error: 0.014064, Waktu: 2.9326 detik
Jumlah titik merah: 31539, Jumlah titik biru: 8461, Jumlah titik: 40000, Estimasi π: 3.1539, Error: 0.012307, Waktu: 3.4716 detik
Jumlah titik merah: 35454, Jumlah titik biru: 9546, Jumlah titik: 45000, Estimasi π: 3.1514666666666667, Error: 0.009874, Waktu: 4.0443 detik
Jumlah titik merah: 39356, Jumlah titik biru: 10644, Jumlah titik: 50000, Estimasi π: 3.14848, Error: 0.006887, Waktu: 4.6687 detik
Jumlah titik merah: 43291, Jumlah titik biru: 11709, Jumlah titik: 55000, Estimasi π: 3.1484363636363635, Error: 0.006844, Waktu: 5.3574 detik
Jumlah titik merah: 47160, Jumlah titik biru: 12840, Jumlah titik: 60000, Estimasi π: 3.144, Error: 0.002407, Waktu: 6.0928 detik
Jumlah titik merah: 51074, Jumlah titik biru: 13926, Jumlah titik: 65000, Estimasi π: 3.1430153846153845, Error: 0.001423, Waktu: 6.8730 detik
Jumlah titik merah: 54985, Jumlah titik biru: 15015, Jumlah titik: 70000, Estimasi π: 3.142, Error: 0.000407, Waktu: 7.6913 detik
Jumlah titik merah: 58922, Jumlah titik biru: 16078, Jumlah titik: 75000, Estimasi π: 3.142506666666667, Error: 0.000914, Waktu: 8.5749 detik
Jumlah titik merah: 62884, Jumlah titik biru: 17116, Jumlah titik: 80000, Estimasi π: 3.1442, Error: 0.002607, Waktu: 9.5484 detik
Jumlah titik merah: 66823, Jumlah titik biru: 18177, Jumlah titik: 85000, Estimasi π: 3.1446117647058824, Error: 0.003019, Waktu: 10.6044 detik
Jumlah titik merah: 70708, Jumlah titik biru: 19292, Jumlah titik: 90000, Estimasi π: 3.1425777777777778, Error: 0.000985, Waktu: 11.7116 detik
Jumlah titik merah: 74649, Jumlah titik biru: 20351, Jumlah titik: 95000, Estimasi π: 3.143115789473684, Error: 0.001523, Waktu: 12.8426 detik
Jumlah titik merah: 78545, Jumlah titik biru: 21455, Jumlah titik: 100000, Estimasi π: 3.1418, Error: 0.000207, Waktu: 14.0172 detik

Simulasi Monte Carlo untuk π:
Jumlah titik: 100000
Estimasi π: 3.1418
π asli: 3.141592653589793
Error: 0.00020734641020681
Waktu eksekusi: 15.013710260391235 detik
```

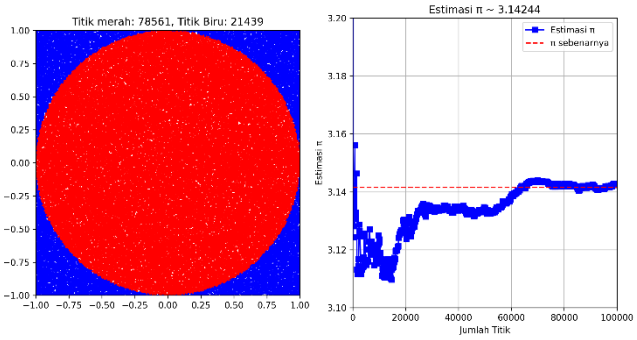
Gambar 1. Percobaan ke-1 Simulasi Program Monte Carlo




```
Jumlah titik merah: 3967, Jumlah titik biru: 1033, Jumlah titik: 5000, Estimasi π: 3.1736, Error: 0.032007, Waktu: 0.0795 detik
Jumlah titik merah: 7909, Jumlah titik biru: 2091, Jumlah titik: 10000, Estimasi π: 3.1636, Error: 0.022007, Waktu: 0.7383 detik
Jumlah titik merah: 11820, Jumlah titik biru: 3180, Jumlah titik: 15000, Estimasi π: 3.152, Error: 0.010407, Waktu: 1.0172 detik
Jumlah titik merah: 15774, Jumlah titik biru: 4226, Jumlah titik: 20000, Estimasi π: 3.1548, Error: 0.013207, Waktu: 1.3471 detik
Jumlah titik merah: 19677, Jumlah titik biru: 5323, Jumlah titik: 25000, Estimasi π: 3.14832, Error: 0.006727, Waktu: 1.7279 detik
Jumlah titik merah: 23623, Jumlah titik biru: 6377, Jumlah titik: 30000, Estimasi π: 3.1497333333333333, Error: 0.008141, Waktu: 2.1610 detik
Jumlah titik merah: 27536, Jumlah titik biru: 7464, Jumlah titik: 35000, Estimasi π: 3.1469714285714288, Error: 0.005379, Waktu: 2.6633 detik
Jumlah titik merah: 31460, Jumlah titik biru: 8540, Jumlah titik: 40000, Estimasi π: 3.146, Error: 0.004407, Waktu: 3.1934 detik
Jumlah titik merah: 35405, Jumlah titik biru: 9595, Jumlah titik: 45000, Estimasi π: 3.1471111111111111, Error: 0.005518, Waktu: 3.7687 detik
Jumlah titik merah: 39305, Jumlah titik biru: 10695, Jumlah titik: 50000, Estimasi π: 3.1444, Error: 0.002807, Waktu: 4.3941 detik
Jumlah titik merah: 43230, Jumlah titik biru: 11770, Jumlah titik: 55000, Estimasi π: 3.144, Error: 0.002407, Waktu: 5.0636 detik
Jumlah titik merah: 47162, Jumlah titik biru: 12838, Jumlah titik: 60000, Estimasi π: 3.1441333333333334, Error: 0.002541, Waktu: 5.7835 detik
Jumlah titik merah: 51034, Jumlah titik biru: 13966, Jumlah titik: 65000, Estimasi π: 3.1405538461538463, Error: 0.001039, Waktu: 6.5633 detik
Jumlah titik merah: 54990, Jumlah titik biru: 15010, Jumlah titik: 70000, Estimasi π: 3.1422857142857143, Error: 0.000693, Waktu: 7.3938 detik
Jumlah titik merah: 58953, Jumlah titik biru: 16047, Jumlah titik: 75000, Estimasi π: 3.14416, Error: 0.002567, Waktu: 8.2746 detik
Jumlah titik merah: 62884, Jumlah titik biru: 17116, Jumlah titik: 80000, Estimasi π: 3.1442, Error: 0.002607, Waktu: 9.2082 detik
Jumlah titik merah: 66829, Jumlah titik biru: 18171, Jumlah titik: 85000, Estimasi π: 3.1448941176470586, Error: 0.003301, Waktu: 10.1903 detik
Jumlah titik merah: 70779, Jumlah titik biru: 19221, Jumlah titik: 90000, Estimasi π: 3.1457333333333333, Error: 0.004141, Waktu: 11.2185 detik
Jumlah titik merah: 74729, Jumlah titik biru: 20271, Jumlah titik: 95000, Estimasi π: 3.146484210526316, Error: 0.004892, Waktu: 12.2993 detik
Jumlah titik merah: 78684, Jumlah titik biru: 21316, Jumlah titik: 100000, Estimasi π: 3.14736, Error: 0.005767, Waktu: 13.4334 detik

Simulasi Monte Carlo untuk π:
Jumlah titik: 100000
Estimasi π: 3.14736
π asli: 3.141592653589793
Error: 0.005767346410206195
Waktu eksekusi: 14.398905515670776 detik
```

Gambar 2. Percobaan ke-2 Simulasi Program Monte Carlo




```
Jumlah titik merah: 3894, Jumlah titik biru: 1106, Jumlah titik: 5000, Estimasi π: 3.1152, Error: 0.026393, Waktu: 0.0804 detik
Jumlah titik merah: 7806, Jumlah titik biru: 2194, Jumlah titik: 10000, Estimasi π: 3.1224, Error: 0.019193, Waktu: 0.7326 detik
Jumlah titik merah: 11676, Jumlah titik biru: 3324, Jumlah titik: 15000, Estimasi π: 3.1136, Error: 0.027993, Waktu: 1.0072 detik
Jumlah titik merah: 15634, Jumlah titik biru: 4366, Jumlah titik: 20000, Estimasi π: 3.1268, Error: 0.014793, Waktu: 1.3364 detik
Jumlah titik merah: 19580, Jumlah titik biru: 5420, Jumlah titik: 25000, Estimasi π: 3.1328, Error: 0.008793, Waktu: 1.7360 detik
Jumlah titik merah: 23511, Jumlah titik biru: 6489, Jumlah titik: 30000, Estimasi π: 3.1348, Error: 0.006793, Waktu: 2.1728 detik
Jumlah titik merah: 27435, Jumlah titik biru: 7565, Jumlah titik: 35000, Estimasi π: 3.1354285714285712, Error: 0.006164, Waktu: 2.6516 detik
Jumlah titik merah: 31335, Jumlah titik biru: 8665, Jumlah titik: 40000, Estimasi π: 3.1335, Error: 0.008093, Waktu: 3.1891 detik
Jumlah titik merah: 35253, Jumlah titik biru: 9747, Jumlah titik: 45000, Estimasi π: 3.1336, Error: 0.007993, Waktu: 3.7699 detik
Jumlah titik merah: 39181, Jumlah titik biru: 10819, Jumlah titik: 50000, Estimasi π: 3.13448, Error: 0.007113, Waktu: 4.3926 detik
Jumlah titik merah: 43091, Jumlah titik biru: 11909, Jumlah titik: 55000, Estimasi π: 3.1338909090909093, Error: 0.007702, Waktu: 5.0639 detik
Jumlah titik merah: 47079, Jumlah titik biru: 12921, Jumlah titik: 60000, Estimasi π: 3.1386, Error: 0.002993, Waktu: 5.7856 detik
Jumlah titik merah: 51044, Jumlah titik biru: 13956, Jumlah titik: 65000, Estimasi π: 3.141169230769231, Error: 0.000423, Waktu: 6.5600 detik
Jumlah titik merah: 55020, Jumlah titik biru: 14980, Jumlah titik: 70000, Estimasi π: 3.144, Error: 0.002407, Waktu: 7.3803 detik
Jumlah titik merah: 58922, Jumlah titik biru: 16078, Jumlah titik: 75000, Estimasi π: 3.142506666666667, Error: 0.000914, Waktu: 8.2710 detik
Jumlah titik merah: 62852, Jumlah titik biru: 17148, Jumlah titik: 80000, Estimasi π: 3.1426, Error: 0.001007, Waktu: 9.2152 detik
Jumlah titik merah: 66753, Jumlah titik biru: 18247, Jumlah titik: 85000, Estimasi π: 3.1413176470588233, Error: 0.000275, Waktu: 10.1833 detik
Jumlah titik merah: 70696, Jumlah titik biru: 19304, Jumlah titik: 90000, Estimasi π: 3.1420444444444446, Error: 0.000452, Waktu: 11.2042 detik
Jumlah titik merah: 74599, Jumlah titik biru: 20401, Jumlah titik: 95000, Estimasi π: 3.1410105263157893, Error: 0.000582, Waktu: 12.2707 detik
Jumlah titik merah: 78561, Jumlah titik biru: 21439, Jumlah titik: 100000, Estimasi π: 3.14244, Error: 0.000847, Waktu: 13.3938 detik

Simulasi Monte Carlo untuk π:
Jumlah titik: 100000
Estimasi π: 3.14244
π asli: 3.141592653589793
Error: 0.000847346410207061
Waktu eksekusi: 14.346374034881592 detik
```

Gambar 3. Percobaan ke-3 Simulasi Program Monte Carlo

Setiap kali program simulasi Monte Carlo dijalankan, hasil estimasi nilai π yang diperoleh selalu sedikit berbeda. Dalam beberapa percobaan, nilai estimasi π berkisar antara 3.1418, 3.1476, hingga 3.1424, dengan error yang bervariasi dalam kisaran 0.0002 hingga 0.0080. Waktu eksekusi pun tidak selalu sama, berkisar antara 13 hingga 15 detik.

Perbedaan ini terjadi karena metode Monte Carlo bergantung pada angka acak. Setiap iterasi menghasilkan titik acak dalam rentang [-1,1] x [-1,1], sehingga jumlah titik yang jatuh ke dalam lingkaran tidak selalu sama di setiap percobaan. Akibatnya, nilai estimasi π mengalami fluktuasi kecil meskipun jumlah sampel yang digunakan tetap sama. Namun, semakin banyak titik yang digunakan dalam simulasi, semakin kecil deviasi dari nilai π sebenarnya, karena distribusi titik semakin mendekati proporsi ideal.

Dari hasil percobaan, bisa disimpulkan bahwa metode Monte Carlo cukup efektif untuk memperkirakan nilai π. Setiap kali program dijalankan, hasilnya sedikit berbeda karena metode ini menggunakan angka acak. Namun, semakin banyak titik yang digunakan dalam simulasi, semakin mendekati nilai π yang sebenarnya, dan selisihnya semakin kecil.. Secara keseluruhan, metode ini adalah cara yang sederhana namun cukup akurat untuk menghitung π, terutama jika jumlah titik yang digunakan cukup banyak.