CALCULATING π USING RANDOM NUMBERS

220136194

Contents

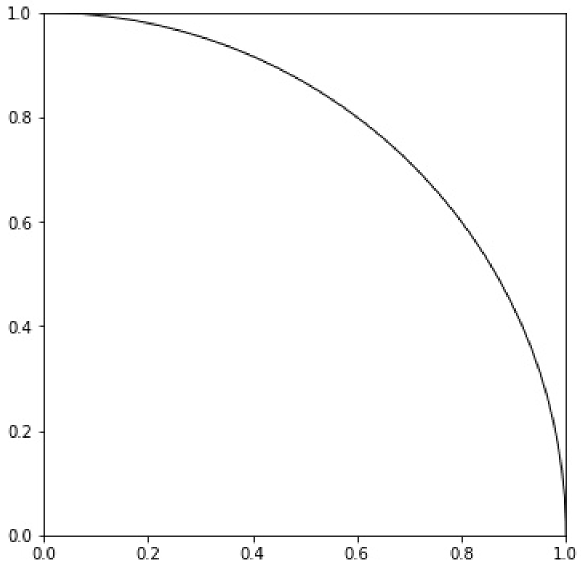
1. Mission

Using programming techniques, I can only use the random function which returns a float between 0.0 and 1.0 to estimate a value for pi.

2. Vizualizing the problem

2.1. **Identifying Areas and Ratios.** Since we are on the hunt for π, lets draw a quarter circle inside a unit square. This is the clue that's given to us.

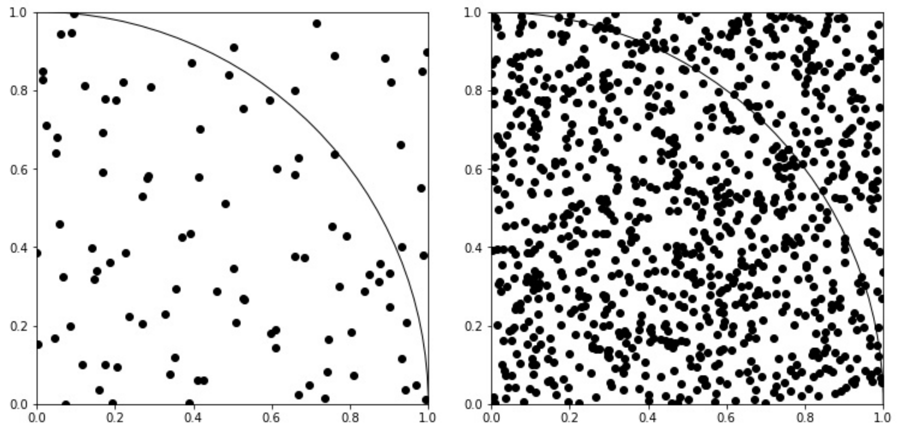
Date: Due November 16, 2022.



We know that the area of a circle with radius 1 equals π. Therefore, the area of the quarter circle $= \pi/4$, which is approximately 0.79. We also know that the area of the unit square holding the semi-circle is $1 \times 1 = 1$. We can then conclude that the ratio of the area of the quater-circle : area of the square is $\pi/4 : 1$.

2.2. **Using a scatter plot to find areas.** We can generate random co-ordinate plots using the random function and plot them on the same xy plane.

Generate 2 numbers between 0.0 and 1.0 (incl).
These 2 numbers become the coordinate pair (x, y) which can be plotted on the plane.



We can logically deduce that as more points are plotted,

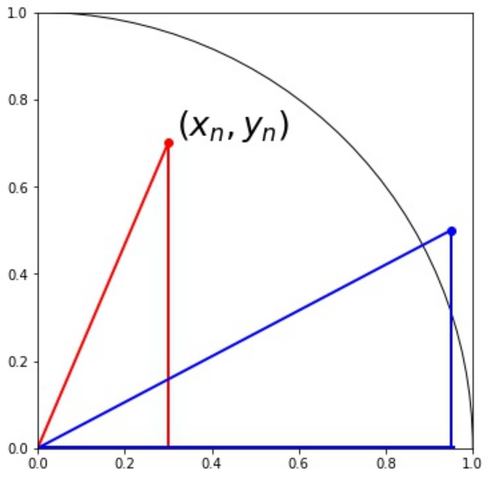
- The total number of points = the area of the square
- The number of points inside the quarter-circle = area of quater-circle

Remembering the ratios we established in section 2.1, we can conclude that

$$lim_{\text{number of points}} \rightarrow \infty \frac{\text{number of points inside quarter-circle}}{\text{total number of points}} = \pi/4$$

3. Checking if a point is inside the circle

In order to work out whether a point is inside the quarter-circle, we need to work out its distance from the origin.



To calculate the distance of any point (x_n, y_n), we work out the euclidean distance r, from the origin using the formula:

$$r = \sqrt{x_n^2 + y_n^2} \tag{1}$$

If $r \leqslant 1$, then the point is within the quarter-circle and hence contributes its area.

4. The Python code: Counting Points

```python
import random

TOTAL = 1_000_000
points_in_circle = 0

for i in range(TOTAL):
    x_coord = random.random()
    y_coord = random.random()

    square_distance = x_coord**2 + y_coord**2

    if square_distance <= 1:
        points_in_circle += 1

print(points_in_circle / TOTAL * 4)
```

4.1. **Explaining the code:**

- The code will generate 1 million scatter points using a loop.
- random.random() returns a float in the range [0.0, 1.0]
- Each time, a random x and y coordinates are assigned in lines 7 and 8
- The squared distance of the (x, y) coordinate is calculated in line 10
- We don't actually need the distance, we only need to know if the distance r is less or equal to 1 to decide whether the point lies in the quarter circle. We can use the following property of squaring floats: If $r \geq 1$, then $r^2 \geq 1$. Also if $r < 1$, then $r^2 < 1$.
- If the square distance is less than or equal to 1, then the point lies in the circle and the number of points in the circle increase in line 13.
- In every loop iteration, new random x-y coordinates are assigned.
- On line 15, we divide the number of points in the circle counted during during the loop by the number of total points generated. As the iterations increase the results gets closer to $\pi/4$. We then multiply the result by 4 to get π.

And just like that, we've found an estimate for π.

5. Try out the code for yourself (Extension Task)

The code can be tried out by entering this link:
`https://colab.research.google.com/drive/18FFB6l_L15iHUwzBZMJtlP2ECBALdKpd?usp=sharing`
You need to sign in with a valid gmail account and that's it. The environment is in the cloud so you don't have to worry about any configurations. Ignore the warnings by clicking 'Run Anyway' as the code is safe.
As an extension task, the user can input the number of coordinates and see the results. The code will calculate the % error of your estimate.

6. Notes

The diagrams are created entirely by me using a python library called matplotlib.