11.16.3.3.4

EE24BTECH11063 - Y. Harsha Vardhan Reddy

Question:
A die is rolled, Find the probability that a number greater than 6 will appear
Solution:
Textual solution:
Probability of a given event 'A'(A: Outcome is greater than 6),

$$P(A) = \frac{0}{6} = 0 \tag{0.1}$$

Textual solution:

Introduction

The task involves simulating the roll of a single die using a C program, compiling it into a shared object (.so) file, and then using Python to call this function. The Python code processes the results and generates a probability distribution plot for outcomes $1, 2, 3, 4, 5, 6$, and >6.

C Code Description

The C program generates random outcomes for rolling a single die, where the possible outcomes range from 1 to 10. Outcomes are categorized as follows:

- $1, 2, \ldots, 6$: Corresponding counts are stored in an array index $0 - 5$.
- >6: Counts for numbers $7 - 10$ are aggregated and stored in index 6.

The program uses the `rand()` function to generate random numbers and increments the respective indices of the results array based on the generated outcome.

Python Code Description

The Python code is responsible for:

1) Loading the shared object file generated from the C program using the `ctypes` library.
2) Calling the C function to roll the die for a specified number of trials (e.g., $10,000$).
3) Retrieving the results from the C function as an array.
4) Calculating the probabilities for each outcome using the formula:

$$P(\text{outcome}) = \frac{\text{frequency of outcome}}{\text{total trials}}$$

5) Plotting the probability distribution using `matplotlib`.

Graphical Output

The Python code generates a bar chart where:

- The x-axis represents the outcomes: $1, 2, 3, 4, 5, 6, >6$.
- The y-axis represents the probabilities of each outcome, ranging from 0 to 1.
- Each bar height corresponds to the probability of the respective outcome.

Key Points

- Using the C program ensures efficient computation of outcomes for large numbers of trials.
- The shared object file facilitates seamless integration with Python, leveraging Python's powerful visualization capabilities.
- The probabilities provide a normalized representation of the frequency distribution, making it easier to interpret the results.

Conclusion

This task demonstrates the integration of C and Python for simulating and visualizing a probabilistic experiment. By combining the computational efficiency of C with the graphical capabilities of Python, we achieve an effective solution for analyzing and representing data. The code clearly shows that the probability of the given event is equal to **zero**

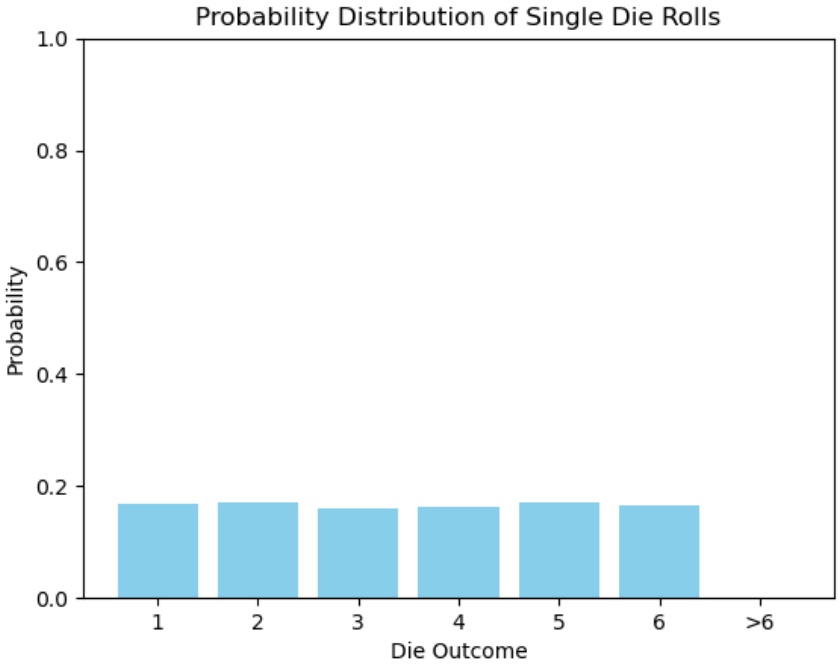

Fig. 5.1: Solution of the system of linear equations