Prime Number Generation

A PROJECT REPORT
for
AI Project (AI101B)
Session (2024-25)

Submitted by

Tanishka Gupta (202410116100218)
Shipra Upadhyay (202410116100196)
Vanshika Garg (202410116100235)
Tripti Rajpoot (202410116100225)

Submitted in partial fulfilment of the
Requirements for the Degree of

MASTER OF COMPUTER APPLICATION

Under the Supervision of
Ms. KOMAL SALGOTRA

Submitted to

DEPARTMENT OF COMPUTER APPLICATIONS
KIET Group of Institutions, Ghaziabad
Uttar Pradesh-201206

INDEX

1. INTRODUCTION

Prime numbers as integers larger than 1 which are only divisible by 1 and themselves are a basic premise in number theory. Prime numbers are the cornerstone of all natural numbers, the same way that atoms are to chemistry. Prime numbers play extremely important roles across many disciplines such as cryptography, computer science, and algorithmics. The intent of this project is to obtain prime numbers of a specified interval and study how they are distributed.

We will utilize the Sieve of Eratosthenes algorithm, the most effective algorithm to determine all primes up to a specified bound, in this project. Furthermore, we will graphically depict the distribution of prime numbers and the gaps among consecutive primes with graphical plots. This project yields information about the prime number structure and statistical features.

In this analysis, we concentrate on two main features of prime numbers:

- Prime number generation: Employing a fast algorithm, the Sieve of Eratosthenes, to generate all prime numbers up to a given limit.
- Visual analysis: Looking at the distribution of prime numbers and the gaps between consecutive primes. We represent these features in histograms and Kernel Density Estimation (KDE) plots.

This research will give insight into the prime numbers' behaviour and how the gaps between them change as numbers increase.

2. METHODOLOGY

In order to accomplish the task of prime number generation and analysis, the following methodology was utilized:

Step 1: Generation of Prime Numbers

The Sieve of Eratosthenes algorithm is utilized for generating prime numbers. It is an old algorithm that effectively identifies prime numbers by iteratively marking the multiples of each prime beginning from 2.

The algorithm proceeds as:

- Creating a Boolean array where the indices represent numbers with True meaning potentially prime.
- Making the index for 0 and 1 False since 0 and 1 are not prime numbers.
- Looping through every number up to the square root of the limit and marking its multiples as non-prime.
- The rest of the True entries represent prime numbers.

Step 2: Data Visualization

After the prime numbers have been created, the gaps and distribution of the prime numbers are examined with the help of visualizations:

- Prime Number Distribution: A histogram is drawn to indicate the frequency of prime numbers within the set limit. This gives us insight into how prime numbers are distributed along the number line.
- Prime Gaps: The gap between adjacent prime numbers (i.e., prime gaps) is calculated and plotted. This enables us to analyse how prime gaps change as numbers get larger.

Step 3: User Input and Execution

The program accepts the upper bound of the number range as user input, computes prime via the sieve algorithm, and prints out both the list of primes and the plots for distribution and prime gaps.

3. CODE

```
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

# Function to generate prime numbers using the Sieve of Eratosthenes
def sieve_of_eratosthenes(limit):
    primes = np.ones(limit + 1, dtype=bool)
    primes[:2] = False  # 0 and 1 are not prime

    for i in range(2, int(limit ** 0.5) + 1):
        if primes[i]:
            primes[i * i: limit + 1: i] = False

    return np.where(primes)[0]  # Return indices of True values (prime numbers)

# Function to plot prime number distribution
def plot_prime_distribution(primes, limit):
    plt.figure(figsize=(12, 6))
    sns.histplot(primes, bins=50, kde=True, color='blue')
    plt.xlabel("Prime Numbers")
    plt.ylabel("Frequency")
    plt.title(f"Distribution of Prime Numbers up to {limit}")
    plt.show()
```

```
# Function to plot prime gaps
def plot_prime_gaps(primes, limit):
    prime_gaps = np.diff(primes)  # Calculate gaps between consecutive primes

    plt.figure(figsize=(12, 6))
    sns.histplot(prime_gaps, bins=30, kde=True, color='green')
    plt.xlabel("Prime Gaps")
    plt.ylabel("Frequency")
    plt.title(f"Distribution of Prime Gaps up to {limit}")
    plt.show()

# Main function to run the analysis
def main():
    limit = int(input("Enter the limit: "))  # User input for limit
    primes = sieve_of_eratosthenes(limit)    # Generate primes

    print(f"Prime numbers up to {limit}:\n", primes)

    # Plot the distributions
    plot_prime_distribution(primes, limit)
    plot_prime_gaps(primes, limit)

# Run the program
if _name_ == "_main_":
    main()
```

4. SCREENSHOTS

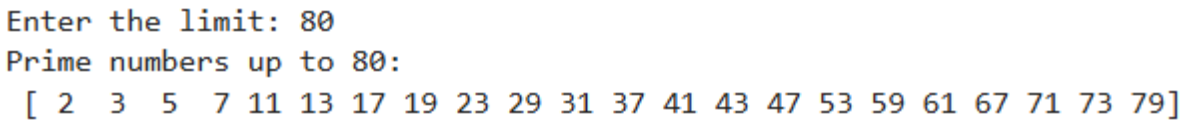

Fig. (1)

This image displays the **output of a Python program** that finds and prints all prime numbers up to a given limit. The user inputs **80**, and the program generates the list of prime numbers up to that value.

Prime numbers are natural numbers greater than 1 that have only two factors: 1 and themselves. The output [2, 3, 5, 7, 11, 13, ..., 79] consists of numbers that meet this condition.

The program likely uses a **loop with a prime-checking condition** (such as trial division or the Sieve of Eratosthenes) to filter out non-prime numbers. The result confirms that primes are more frequent among smaller numbers and become sparser as values increase.

This simple yet effective implementation is useful for mathematical analysis, cryptography, and number theory studies.

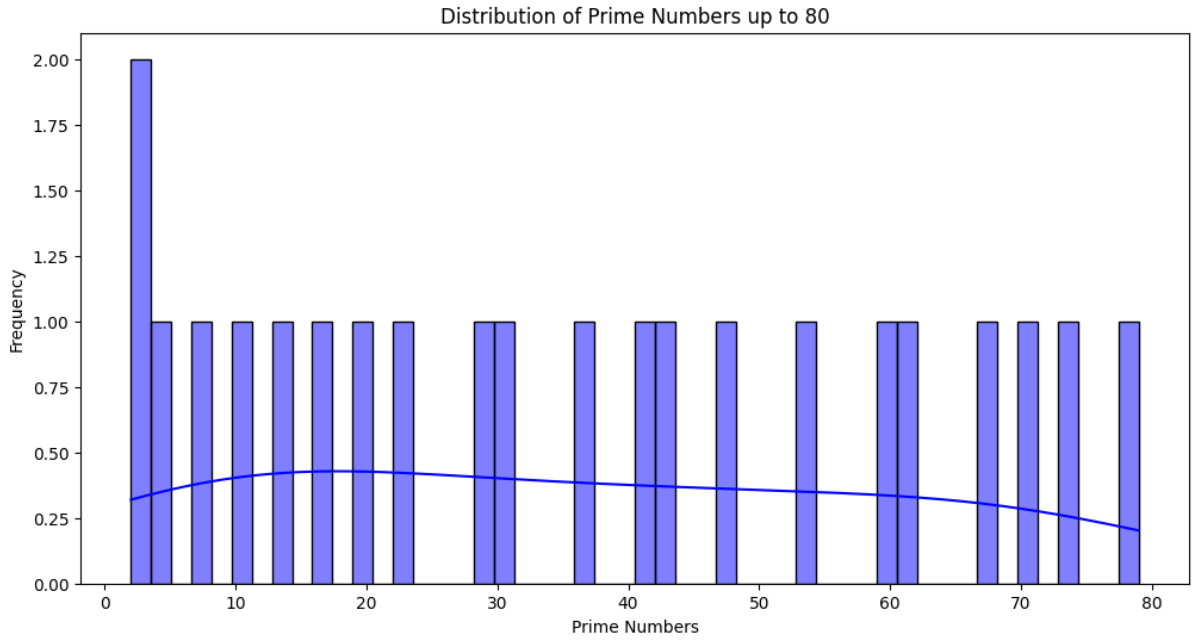

Fig. (2)

This graph illustrates the distribution of primes up to 80 in a histogram with a density curve. The **X-axis** is prime numbers from 2 to 79, and the **Y-axis** is their frequency in various ranges. The histogram bars illustrate how primes are distributed across these ranges, with the highest bars at lower values, reflecting a greater density of primes in that range.

The **blue density curve** also emphasizes this trend, increasing slightly initially and then slowly dropping off, indicating that prime numbers are less common as numbers get larger. This trend is consistent with the mathematical fact that primes become more isolated as numbers get bigger.

In general, the graph well depicts the shifting distribution of primes, supporting the observation that although primes are plentiful in lower ranges, at larger values, they are scarcer.

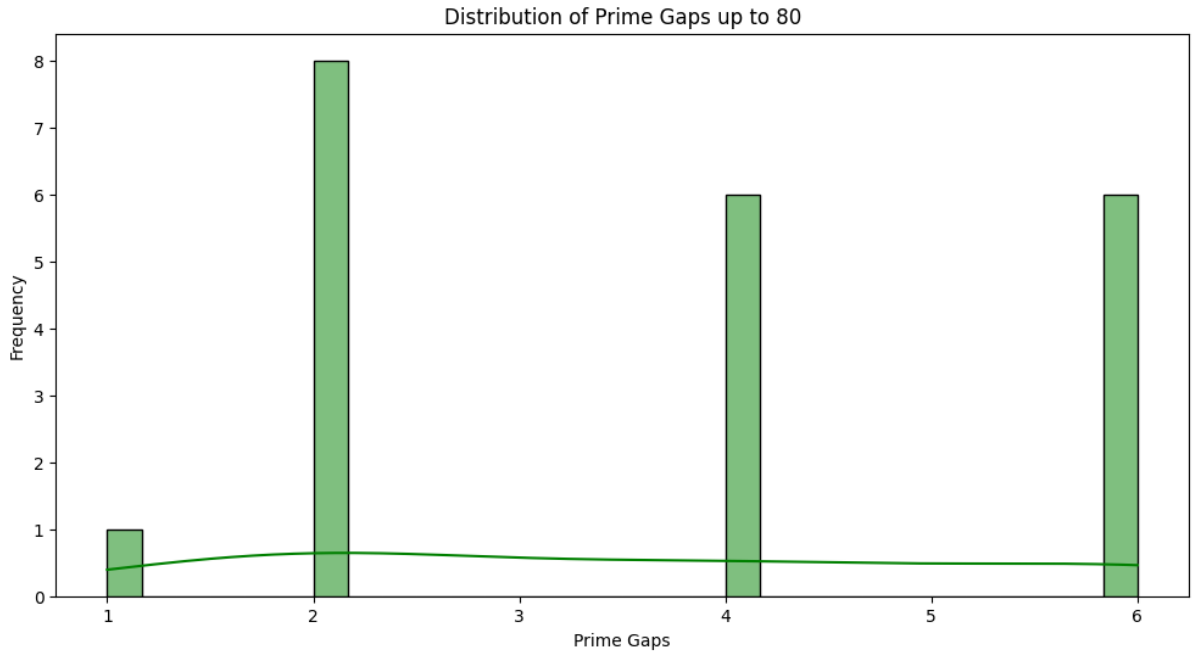

Fig. (3)

This is a graph of the prime gap distribution till 80, where the frequency of the difference between consecutive primes is plotted. The prime gaps (e.g., 2, 4, 6) are shown on the **X-axis** and frequency is shown on the **Y-axis**.

Histogram bars summarize that the 2 is the most common gap, i.e., most consecutive primes (e.g., 3 & 5, 5 & 7) are 2 apart. 4 and 6 gaps also occur with moderate frequency, suggesting primes are more spread out as numbers increase. The **density curve** indicates the general pattern, reinforcing larger than 6 gaps are uncommon in this range.

This is a trend in the behaviour of prime numbers: they come in regular frequency in close but decreasing frequency and widening distance as numbers rise. The graph effectively serves to demonstrate how gaps between primes vary in the given range.