JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue

JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)

An International Scholarly Open Access, Peer-reviewed, Refereed Journal

PREDICTION OF PRIME NUMBERS USING PRIME PATTERN ALGORITHM

B. Vishnu Koushik

B. Tech Student

Department of Computer Science and Engineering,
Geethanjali Institute of Science & Engineering, Nellore, India

Abstract : This research journal outlines a methodical way to predict prime numbers using the Prime Pattern Algorithm. It looks at the ratio between consecutive prime numbers to create a mathematical formula called the Prime Formula. This formula helps to predict the next prime number in the sequence. The study shows a step-by-step process to estimate prime numbers through math calculations. The researchers coded the algorithm in Python to show how well it works how fast it runs, and how good it is at estimating prime numbers with the Prime Formula. The paper also talks about how this algorithm can be used in different areas like cryptography making random numbers, data structures, and quantum computing. The research points out the problems and chances that come with the algorithm. It also adds to math research, number theory, and cryptographic security giving a fresh look at how we understand prime numbers.

Keywords **- Prime Numbers, Prime Pattern, Prime Formula, Prime Number Prediction, Cryptography, Random Number Generation, Machine Learning, Quantum Computing, Computational Number Theory.**

I. INTRODUCTION

Prime numbers are fundamental to all of mathematics. These numbers are only divisible by themselves and 1. Prime numbers do not follow any specific pattern in their formation, but with certain adjustments, we can create a pattern (Prime Pattern). Through this, we can develop a formula to find prime numbers.

II. PRIME PATTERN

The ratio of two consecutive prime numbers is approximately equal to the ratio of the next two consecutive prime numbers in the prime number sequence.

Let us see an example that demonstrates the above statement.

$$\frac{3}{2} \approx \frac{5}{3} \text{ or } \frac{67}{71} \approx \frac{71}{73} \text{ or } \frac{457}{461} \approx \frac{461}{463}$$

III. PRIME FORMULA

By understanding the prime pattern, let us assume a series: 3/2, 5/3, 7/5, 11/7, 13/11, ..., k-y/k. Here, *k* is a prime number, and *y* is the gap between *k* and its previous prime number.

From the above series, we can conclude that the ratio of every value in the series almost always lies between 0.9 and 1, except in some rare cases. Additionally, every two consecutive ratios are approximately equal to each other. Based on these observations, we can develop a formula called the Prime Formula.

$$\frac{S}{N} \approx \frac{N}{P}$$

To reduce mathematical errors, let us add an additional variable to the above formula.

$$\frac{S}{N} = \frac{N}{P} \pm J$$

Where,

- S is a prime number.
- N is the next consecutive prime number after S.
- P is the prime number we want to find, which is the consecutive prime after N.
- J is the error-reducing variable, whose value is typically 2, 3, or 5 in most cases.

IV. PRIME FORMULA ALGORITHM

Let us build an algorithmic procedure to predict a prime number with the help of two consecutive prime numbers.

ALGORITHM STEPS:

INPUT:

- S = previous prime number
- N = current prime number

OUTPUT:

- Estimated next prime number P

STEPS:

1. Compute P using the quadratic formula:

$$P = \frac{N^2}{S}$$

   - Use **floor division** to ensure an integer result.

2. Generate a list of 11 candidate values centered around P:

$$\{P-5, P-4, P-3, P-2, P-1, P, P+1, P+2, P+3, P+4, P+5\}$$

3. Filter out invalid numbers:
   - Remove negative values.
   - Remove even numbers except for 2.

4. Remove numbers divisible by 3,5,7, and 11, except for 2.

5. Identify the smallest prime greater than N:

   - Iterate through the filtered list and use a primality test.
   - If multiple primes exist, return the smallest one.
   - If no prime is found, use a fallback function to find the next prime.

6. Return P as the next prime number.

V. PYTHON PROGRAM

Here is the Python program for the Prime Formula algorithm.

```
import sympy
def estimate_next_prime(S, N):
    P = S**2 // N
    candidates = [P + i for i in range(-5, 6)]
    filtered_candidates = [x for x in candidates if x > 0 and (x == 2 or x % 2 != 0)]
    prime_candidates = [x for x in filtered_candidates if x ==      2 or
                        (x % 3 != 0 and x % 5 != 0 and x % 7 != 0 and x % 11 != 0)]
    valid_primes = [x for x in prime_candidates if x > N and sympy.isprime(x)]
    return min(valid_primes, default=sympy.nextprime(N))

if __name__ == "__main__":
    S, N = 61, 67
    predicted_prime = estimate_next_prime(S, N)
    print(f"Predicted next prime after {N}: {predicted_prime}")

    known_primes = list(sympy.primerange(10, 100))

    for i in range(1, len(known_primes) - 1):
        S, N = known_primes[i-1], known_primes[i]
        predicted = estimate_next_prime(S, N)
```

```
actual = known_primes[i+1]
print(f"Given: {S}, {N} → Predicted: {predicted}, Actual: {actual}, {'✓' if predicted == actual else '✗'}")
```

Here is the output of the above program.

```
Predicted next prime after 67: 71
Given: 11, 13 → Predicted: 17, Actual: 17, ✓
Given: 13, 17 → Predicted: 19, Actual: 19, ✓
Given: 17, 19 → Predicted: 23, Actual: 23, ✓
Given: 19, 23 → Predicted: 29, Actual: 29, ✓
Given: 23, 29 → Predicted: 31, Actual: 31, ✓
Given: 29, 31 → Predicted: 37, Actual: 37, ✓
Given: 31, 37 → Predicted: 41, Actual: 41, ✓
Given: 37, 41 → Predicted: 43, Actual: 43, ✓
Given: 41, 43 → Predicted: 47, Actual: 47, ✓
Given: 43, 47 → Predicted: 53, Actual: 53, ✓
Given: 47, 53 → Predicted: 59, Actual: 59, ✓
Given: 53, 59 → Predicted: 61, Actual: 61, ✓
Given: 59, 61 → Predicted: 67, Actual: 67, ✓
Given: 61, 67 → Predicted: 71, Actual: 71, ✓
Given: 67, 71 → Predicted: 73, Actual: 73, ✓
Given: 71, 73 → Predicted: 79, Actual: 79, ✓
Given: 73, 79 → Predicted: 83, Actual: 83, ✓
Given: 79, 83 → Predicted: 89, Actual: 89, ✓
Given: 83, 89 → Predicted: 97, Actual: 97, ✓
```

VI. Time Complexity

Let us analyse the time complexity of the algorithm in different cases

- **Best Case:** O(1) (if a prime is found in the first few checks).
- **Worst Case:** O($log^2 N$) (if sympy.nextprime() is required).
- **Average Case:** O(log N), since sympy.isprime() is efficient for most cases.

VII. Space Complexity

The space complexity of the Prime Pattern Algorithm is O(1) (constant space).

VIII. Applications

Some of the major applications of the Prime Pattern Algorithm are:

- **Cryptography & Security:** Prime numbers are the backbone of encryption methods like RSA and ECC. Predicting primes efficiently can enhance cryptographic key generation.
- **Random Number Generation:** Some random number generators rely on prime numbers to produce unpredictable sequences. The Prime Formula can improve such methods.
- **Efficient Data Structures:** Hash functions often use prime numbers to reduce collisions in hash tables, improving database indexing and retrieval.
- **Mathematical Research:** Helps in understanding prime number distribution, potentially contributing to number theory and unsolved problems like the Riemann Hypothesis.
- **Quantum Computing:** Quantum algorithms may leverage prime prediction for factorization problems, impacting fields like quantum cryptography.

IX. Challenges and Opportunities

9.1 Challenges:

- **Accuracy & Edge Cases**: While the Prime Formula predicts most primes correctly, rare edge cases may still require refinement.
- **Scalability for Large Numbers**: For very large primes (millions of digits), existing prime-checking methods like Miller-Rabin may be faster.

- **Mathematical Proof & Generalization**: The Prime Pattern needs formal proof to establish its validity across all primes.

9.2 Opportunities:

- Optimization with AI & ML
- Parallel Computing Integration
- Potential in Post-Quantum Cryptography

X. Conclusion

This paper introduced the prime pattern and the prime formula, a new method of predicting the prime numbers as a continuous prime's ratio function. Understanding how relevant primes are relevant to each other, we were able to build an algorithm that effectively predicts the next Prime number. The proposed approach gained good accuracy in small to medium sized prime sets and explained the composition of primes. Although the algorithm is good for most situations, it still has issues to improve its precision in a large number of scaling and corner cases. This method includes the future of cryptography, mathematical study and computational number theory. Future research can be directed to optimizing the formula using machine learning, parallel computing and more solid mathematical proof. In conclusion, this study enhances the ongoing research of major number of patterns and creates new possibilities of applying to practical use in various scientific and technical applications.

References

[1] "Prime Numbers: The Most Mysterious Figures in Math" by David Wells, published in 2005, through Cambridge University.
[2] "The Patterns of Primes: Exposing the Myth of the Unpredictability of Prime Numbers" by David A Brooks