

**THE DISTRIBUTION OF PRIMES AMONG**

**THE NATURAL NUMBERS.**

**Peter G.Bass.**

## **ABSTRACT.**

The distribution of primes among the Natural numbers is, in this short paper, determined via a new method based upon the Multiples attribute of such numbers.

## **CONTENTS.**

**1.0 Introduction.**

**2.0 Prime Number Distribution Via the Multiples Method.**

**3.0 Implementation.**

**4.0 Conclusions.**

**References.**

## **1.0 Introduction.**

The distribution of primes,  $\pi(N)$ , within the Natural numbers, has been the subject of investigation for many centuries past. The most definitive result was developed by Bernhard Riemann in his famous 1859 paper. This solution is an analytical one, replacing the Natural numbers by the Real, thus calculating  $\pi(x)$ . It contains the Logarithmic Integral of the independent variable as its main term, augmented by an auxiliary term derived from the roots of Riemann's Zeta Function, and two other very small terms, one of which is a constant, while the other, marginally variable.

The main difficulty with Riemann's analytical solution is the determination of the second, oscillatory term. This involves obtaining a number of the roots of Riemann's Zeta Function, a process which, because this function is one of an infinite sum in which all terms possess a complex number exponent, can only be effected by a protracted iterative method. In addition, once these roots have been determined, they are then entered into Riemann's formula, which subsequently involves a considerable amount of further computation.

The method employed here works with the Natural numbers directly, and, being based upon the Multiples attribute of these numbers, as introduced in [1], involves the minimum amount of computation, all of which is elementary.

To determine  $\pi(N)$ , the process simply compares the Multiples of each pair of consecutive numbers from 1 to  $N$ , and counts the number of times the comparison produces a zero result. The method is therefore not analytic, but a straightforward counting process.

## **2.0 Prime Number Distribution Via the Multiples Method.**

In [1], the Multiples attribute of the Natural numbers was introduced as

$$M(N) = \sum_{p=2}^{\sqrt{N}} INT\left(\frac{N}{p}\right) \quad (2.1)$$

where  $p$  ranges through all the primes between the limits of the sum. Subsequently,  $N$  was shown to be prime when

$$M(N) - M(N-1) = 0 \quad (2.2)$$

From that it is clear that if

$$M(N) - M(N-1) = D(N) \quad (2.3)$$

$\pi(N)$  will be given by

$$\pi(N) = \sum_{N=2}^{\sqrt{N}} [D(N) = 0] + 1 \quad (2.4)$$

It should be noted that (2.4) includes unity as a prime number.

### 3.0 Implementation.

While (2.4) could be calculated manually for small values of  $N$ , large values are best determined via a computer implementation. This has been done in support of this paper via a demonstration macro driven EXCEL spreadsheet. The display page is shown below.

The screenshot shows an Excel spreadsheet interface with the following components:

- Version 1.0.0** and **© P.G. Bass** in the top corners.
- Input Data:** Fields for  $N1$  and  $N2$ , and a **Start** button.
- Output Result:** A field for the calculation result.
- Computation Time (Secs):** A field for the time taken.
- Error Messages:** A large empty box for displaying errors.
- Processor Speed, MHz:** 3600
- Estimated Time, (Secs):** < 5.0
- Maximum Database Prime Number:** 4,999,999
- Maximum Input for Primality Test:** 24,999,990,000,001
- Instructions:**
  1. To Test a full range for  $\pi(N)$ , insert the range top number into  $N1$ , and unity into  $N2$ . Press **START**. (For a **very** large range, a long computation time will be required).
  2. To test a partial range for  $\pi(N)$ , insert the range top number into  $N1$ , and the range bottom number into  $N2$ . Press **START**. (For **very** large ranges, a long computation time will be required).
  3. To effect a Database Update,  $N2$  **MUST** be < than, or = to, the **Maximum Database Prime Number**, and  $N1$  greater than it. The Database will then be updated to the smallest prime closest to  $N1$ .
  4. To test a number for primeness, insert that number into  $N1$  and one less into  $N2$ . Press **START**.
- Important Note:** This programme assumes unity is Prime.
- Primes Database:** A table of sequential prime numbers.
 

Primes Database																						
All primes in this database MUST be sequential, otherwise errors will result.																						
2	3	5	7	11	13	17	19	23	29	31	37	41	43	47	53	59	61	67	71	73	79	83
1621	1627	1637	1657	1663	1667	1669	1693	1697	1699	1709	1721	1723	1733	1741	1747	1753	1759	1777	1783	1787	1789	1801
3673	3677	3691	3697	3701	3709	3719	3727	3733	3739	3761	3767	3769	3779	3793	3797	3803	3821	3823	3833	3847	3851	3853
5851	5857	5861	5867	5869	5879	5881	5897	5903	5923	5927	5939	5953	5981	5987	6007	6011	6029	6037	6043	6047	6053	6067
8167	8171	8179	8191	8209	8219	8221	8231	8233	8237	8243	8263	8269	8273	8287	8291	8293	8297	8311	8317	8329	8353	8363

**Fig. 3.1 - Demonstration EXCEL Display Page.**

Use of the spreadsheet is very simple and clear from the instructions shown in the above figure. The method does require an extensive prime number database, (up to  $\sqrt{N}$ ), to accurately compute  $\pi(N)$ , but this is not considered a problem because the method can generate its own database as it progresses, (Note that only a very small fraction of the database is shown in the above Figure). Thus the only adverse factor regarding this aspect is the storage space required on the spreadsheet and the resulting large file size.

Once the database has been updated, "The Maximum Database Prime Number" and "Maximum Input for Primality Test" cells are automatically updated. In addition, the database can then also be transferred to the demonstration spreadsheets associated with [1] to extend their capabilities, (Note that extra coding in those spreadsheets will be required to effect same).

Calculation times for  $\pi(N)$  are of course dependent upon processor speed, and for a range,  $N1 - N2$ , of  $\sim 10^6$ , a 3.6GHz P.C. will produce a result in approximately 40 minutes. However, if a database update is effected, this time will be lengthened by  $\sim 20\%$ . A feature to estimate processing time, based loosely upon processor speed, has been included in the spreadsheet.

As shown, and mentioned earlier, this implementation also enables primality testing of a single number, via (2.2), but computation time will be somewhat longer than that in [1], because all calculation here is consecutive, (in [1] it is simultaneous).

#### **4.0 Conclusions.**

As stated earlier, this method is not analytic, it is purely a prime number counting mechanism. Hence it cannot be used to investigate the dynamic characteristics of the prime number distribution, such as density etc, as with Riemann's and other analytic equations for prime number distribution. However, such a feature was not intended for this method, the primary requirement being simplicity and speed of execution.

#### **REFERENCES.**

- [1] P.G.Bass, *Primality Testing - Two New Methods*, [www.relativitydomains.com](http://www.relativitydomains.com).