Journal of Natural Sciences
June 2014, Vol. 2, No. 1, pp. 107-116
ISSN: 2334-2943 (Print) 2334-2951 (Online)
Copyright © The Author(s). 2014. All Rights Reserved.
Published by American Research Institute for Policy Development

Prime Generating Algorithms through Nonprimality of Even Numbers (Except 2) and by Skipping Even Divisors (Except 2)

Neeraj Anant Pande[1]

Abstract

There have been consistent systematic efforts of refining the prime generating sieves. Number 2 enjoys the unique status of being the first and the only even prime. The fact that no even number other than 2 is prime number along with the one that no prime number except 2 has even divisors leads to further improvement of prime generating algorithms of each respective sieve. Three new sieves are presented and exhaustively compared with earlier respective nine.

Keywords: Algorithm, Sieve, Prime Number

Mathematics Subject Classification 2010: 11Y11, 11Y16, 65Y04, 65Y20, 68Q25

1. Introduction

In the hunt for better (consecutive) prime generating algorithms, popularly known as sieves, various techniques have evolved gradually as has been reported in [4], [5] and [6]. Each of these three papers analyzes three algorithms with improving approach. A comparative statement of run-time requirements and number of steps taken for each of these is quite self-explanatory about the efficiency achieved. Total nine sieves appearing in these three works show a continuous graph of superior performance. Continuing on the same path further, in the present work, the approaches of [5] and [6] have been combined illustrating even better results.

[1] Department of Mathematics & Statistics, Yeshwant Mahavidyalaya (College), Nanded-431602, Maharashtra, INDIA.

Every number has some specialty of its own. Number 2, in addition to other aspects, is special in prime number study due to a main reason that it is the only even prime - so all even numbers greater than 2 are non-prime, i.e., composite. Consequently, since any prime except 2 is not even, it cannot have even divisors. These two properties of 2 and even numbers have been harnessed to develop better prime generating algorithms.

2. Sieves Subtype 1 Improved

It would be better to renumber the Sieves 1, 2 and 3 of [4] as Sieves 1.1.1, 1.1.2 and 1.1.3, respectively, in coherence with the numbering later adopted in [5] and [6].

The common approach of rightly named dumb simplest Sieve 1 [4], Sieve 1.2.1 [5] and Sieve 2.1.1 [6], is the property of integers that no positive integer can have a divisor greater than itself. In fact, in primality tests for a positive integer k, as one is interested in only the divisors of k other than 1 and k itself, this common approach searches for possible divisors of k in the range 2 to $k - 1$. On failure, k is declared as prime; while success leads to conclusion that k is non-prime or composite.

Sieve 1.1.1 [4] adopts plain way of testing all numbers from 2 to $k - 1$ for divisibility and hence is most non-efficient. It being so fundamentally simple, we just mention it instead of supplying it explicitly.

Sieve 1.2.1 in [5] reads :

```
Take an integer $n$ larger than 1

For all values of $k$ from 2 to $n$ and only odd values after 2

    For values of integer $d$ from 2 to $k-1$
        If $d$ divides $k$ perfectly,
            Stop checking as $k$ is not prime
        Else
            Continue checking

If checks do not stop for any value of $d$ till $k-1$, $k$ is prime
```

While Sieve 2.1.1 in [6] takes the approach :

```
Take an integer $n$ larger than 1
2 is prime

For all values of $k$ from 3 to $n$
    For values of integer $d$ from 2 to $k-1$ and only odd values after 2
            If $d$ divides $k$ perfectly,
                Stop checking as $k$ is not prime
            Else
                Continue checking
    If checks do not stop for any value of $d$ till $k-1$, $k$ is prime
```

The combination of both of these approaches with a little modification leads to following outcome which we name as Sieve 2.2.1 :

```
Take an integer $n$ larger than 1
2 is prime
For all values of $k$ from 3 to $n$ and only odd values after 2
    For values of integer $d$ from 3 to $k-1$ and only odd values after 2
            If $d$ divides $k$ perfectly,
                Stop checking as $k$ is not prime
            Else
                Continue checking
    If checks do not stop for any value of $d$ till $k-1$, $k$ is prime
```

While devising this improved version through combination, the values of the numbers to be tested as divisors (designated as *d* in the sieve) have started with 3 instead of 2. As 2 is already declared to be prime and only the odd numbers after 2 are being taken on primality test, test of divisibility by 2 becomes altogether redundant and hence 2 is skipped as candidate for possible divisor.

Sieve 2.2.1 takes less number of steps and consequently lesser time in determining the primality.

To have a comparative idea of quantum of improvement, the three algorithms were implemented and run on an electronic computer for the ranges each of 1 – 10, 1 – 100, 1 – 1000, 1 – 10000 and 1 – 100000 and reduction in the number of steps was as in Table 1. All these computations have been rigorously performed by using electronic computer and Java programming Language is employed for implementing the algorithms. In this and further data tables, number range is restricted to 100000 as Sieves 1.1.1, 1.1.2 and 1.1.3 take excess time for higher values and it is impracticable to go for higher ranges even by using fast electronic computers.

Table 1 : Number of Steps Taken by Sieves of Subtype 1

Sr. No.	Range	Steps Taken by Sieve 1.1.1	Steps Taken by Sieve 1.2.1	Steps Taken by Sieve 2.1.1	Steps Taken by Sieve 2.2.1
1	1 – 10	15	11	12	4
2	1 – 100	1133	1084	628	530
3	1 – 1000	78022	77523	39676	38678
4	1 – 10000	5775223	5770224	2894496	2884498
5	1 – 100000	455189150	455139151	227664778	227564780

Figure 1: Depicts This Data Graphically

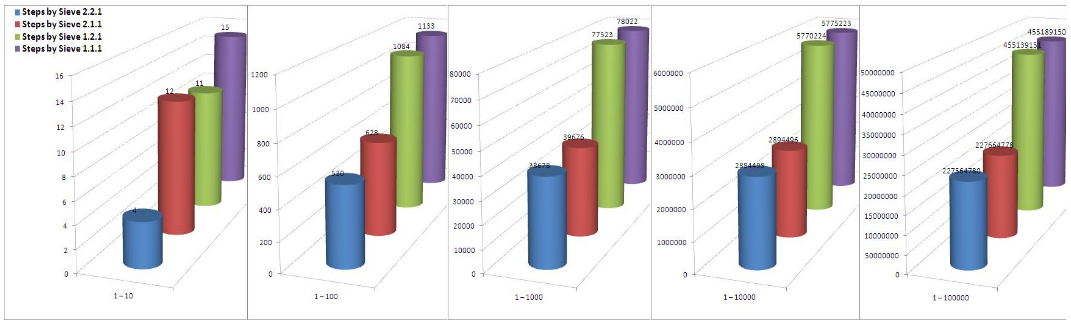

Figure 1 : Number of Steps Taken by Sieves of Subtype 1

The improvement achieved over earlier sieves is as under :

Table 2 : Step Difference of Sieve 2.2.1 with earlier Sieves of Subtype 1

Sr. No.	Range	Steps Lesser than Sieve 1.1.1	Steps Lesser than Sieve 1.2.1	Steps Lesser than Sieve 2.1.1
1	1 – 10	11	7	8
2	1 – 100	603	554	98
3	1 – 1000	39344	38845	998
4	1 – 10000	2890725	2885726	9998
5	1 – 100000	227624370	227574371	99998

The graphical comparison of reduced step of Sieve 2.2.1 is as follows.

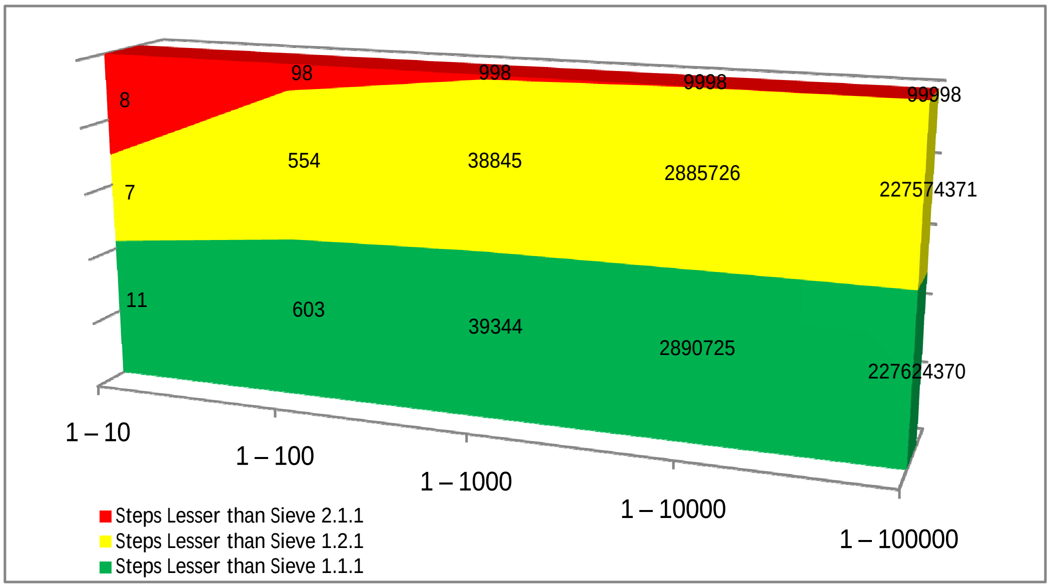

Figure 2 : Step Difference of Sieve 2.2.1 with earlier Sieves of Subtype 1

3. Sieves Subtype 2 Improved

All the sieves subtype 1's occurring in [4], [5] and [6] have been improved in the respective works by corresponding subtypes 2's, labeled as Sieves 1.1.2, 1.2.2, 2.1.2. The refining step in determining the primality of a positive integer k is to test only the positive integers from 2 to $k/2$ for divisibility to reduce the efforts almost to half. The principle behind it being very simple that k cannot have an integral divisor greater than $k/2$.

All those sieve subtype 2's when refined with the method of present work yield Sieve 2.2.2 :

```
Take an integer $n$ larger than 1
2 is prime
For all values of $k$ from 3 to $n$ and only odd values after 2
    For values of integer $d$ from 3 to $k/2$ and only odd values after 2
        If $d$ divides $k$ perfectly,
            Stop checking as $k$ is not prime
        Else
            Continue checking
    If checks do not stop for any value of $d$ till $k/2$, $k$ is prime
```

Steps requirement in determining primality in first few ranges is again seen to be reduced significantly.

Table 3 : Number of Steps Taken by Sieves of Subtype 2

Sr. No.	Range	Steps Taken by Sieve 1.1.2	Steps Taken by Sieve 1.2.2	Steps Taken by Sieve 2.1.2	Steps Taken by Sieve 2.2.2
1	1 – 10	9	5	10	2
2	1 – 100	616	567	376	278
3	1 – 1000	40043	39544	20730	19732
4	1 – 10000	2907640	2902641	1461014	1451016
5	1 – 100000	227995678	227945679	114070446	113970448

These figures compare graphically as follows.

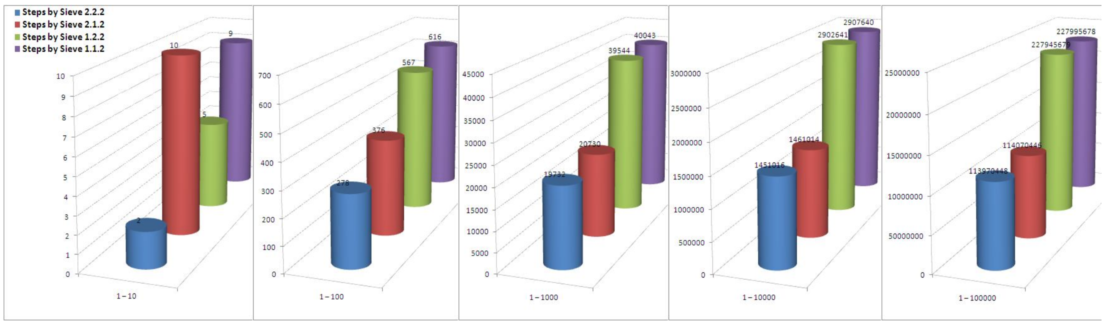

Figure 3 : Number of Steps Taken by Sieves of Subtype 2

The count of gain in reduction is as in following table.

Table 4 : Step Difference of Sieve 2.2.2 with earlier Sieves of Subtype 2

Sr. No.	Range	Steps Lesser than Sieve 1.1.2	Steps Lesser than Sieve 1.2.2	Steps Lesser than Sieve 2.1.2
1	1 – 10	7	3	8
2	1 – 100	338	289	98
3	1 – 1000	20311	19812	998
4	1 – 10000	1456624	1451625	9998
5	1 – 100000	114025230	113975231	99998

Figure 4 illustrates the steps improvement of Sieve 2.2.2.

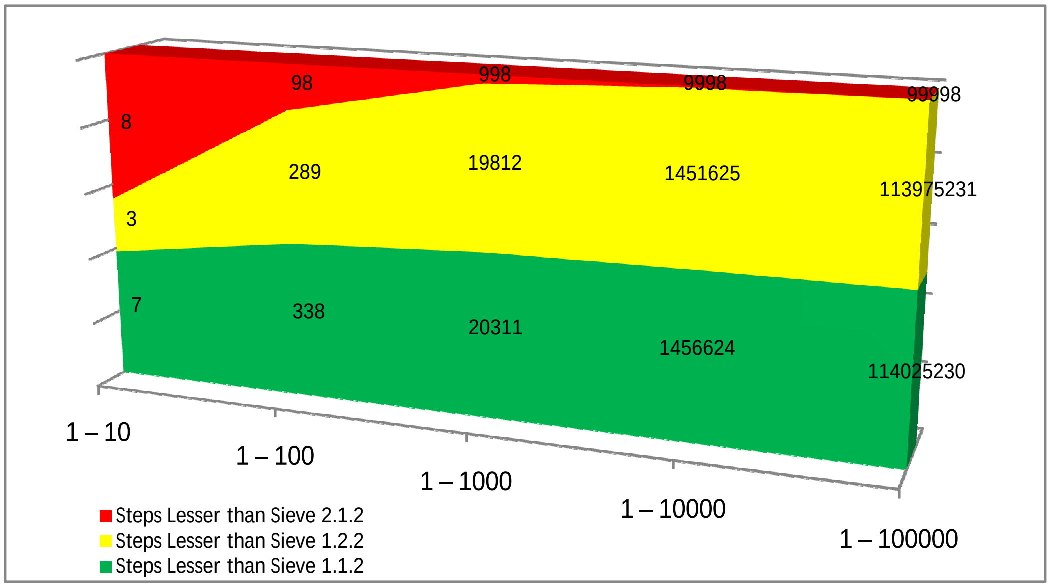

Figure 4 : Step Difference of Sieve 2.2.2 with earlier Sieves of Subtype 2

4. Sieves Subtype 3 Improved

In each of the referred earlier works, the ultimate improvement of succession of sieve subtypes has been subtype 3. That makes the way of formulating the superior Sieve 2.2.3 of present work clear.

As in other subtypes 3 designated as 1.1.3 in [4], 1.2.3 in [5] and 2.1.3 in [6], here also, the advantage can be taken of the fact that for every positive integer *k*; for every divisor of *k* greater than $\sqrt{k}$, there is a divisor of *k* less than $\sqrt{k}$. This pairing effect allows to look for positive divisors of *k* in the further reduced range of 2 to $\sqrt{k}$. If there is a divisor of *k* in this range, obviously *k* is not prime; and if there is no divisor of *k* in this range, then *k* is prime, for there cannot be any divisor in the remaining higher range also as it would then lack its partner in our range. The result is Sieve 2.2.3 :

```
Take an integer $n$ larger than 1
2 is prime
For all values of $k$ from 3 to $n$ and only odd values after 2
    For values of integer $d$ from 3 to $\sqrt{k}$ and only odd values after 2
        If $d$ divides $k$ perfectly,
            Stop checking as $k$ is not prime
        Else
            Continue checking
    If checks do not stop for any value of $d$ till $\sqrt{k}$, $k$ is prime
```

With comparatively less efforts, the statistics of steps requirement of improved subtypes 3's is available.

Table 5 : Number of Steps Taken by Sieves of Subtype 3

Sr. No.	Range	Steps Taken by Sieve 1.1.3	Steps Taken by Sieve 1.2.3	Steps Taken by Sieve 2.1.3	Steps Taken by Sieve 2.2.3
1	1 – 10	8	4	9	1
2	1 – 100	236	187	185	87
3	1 – 1000	5288	4789	3349	2351
4	1 – 10000	117527	112528	65956	55958
5	1 – 100000	2745694	2695695	1445440	1345442

This data is illustrated in the following figure.

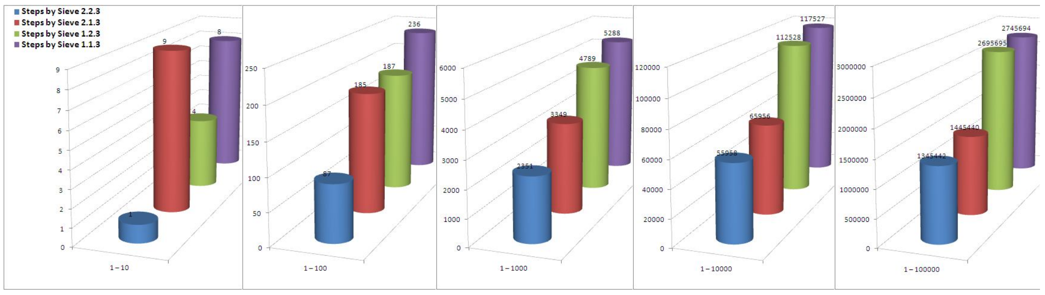

Figure 5 : Number of Steps Taken by Sieves of Subtype 3

The step reduction obtained from Sieve 2.2.3 gives maximum efficiency.

Table 6 : Step Difference of Sieve 2.2.3 with earlier Sieves of Subtype 3

Sr. No.	Range	Steps Lesser than Sieve 1.1.3	Steps Lesser than Sieve 1.2.3	Steps Lesser than Sieve 2.1.3
1	1 – 10	7	3	8
2	1 – 100	149	100	98
3	1 – 1000	2937	2438	998
4	1 – 10000	61569	56570	9998
5	1 – 100000	1400252	1350253	99998

The visual representation in the graph makes the trend clearer.

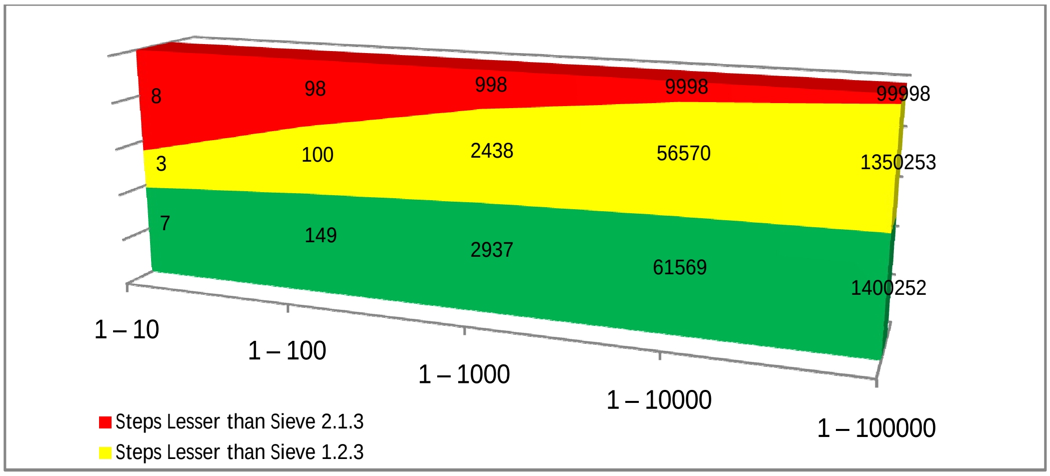

Figure 6 : Step Difference of Sieve 2.2.3 with earlier Sieves of Subtype 3

The three new versions of the sieves presented here, viz., Sieves 2.2.1, 2.2.2 and 2.2.3, are quite promising. No doubt, they are inspired by the earlier work, in particular, a just combination of different better approaches in [5] and [6], which stand on the foundation of [4]. It is equally interesting to note that few best known traditional algorithms also find their roots in these.

Acknowledgements

The author expresses his grateful to the Java (7 Update 25) Programming Language Development Team and the NetBeans IDE 7.3.1 Development Team, whose software have been freely used in implementing the algorithms developed and tested during this work.

The author is also thankful to the University Grants Commission (U.G.C.), New Delhi of the Government of India for funding this research work under a Research Project (F.No. 47-748/13(WRO)).

References

Burton David M. (2007). Elementary Number Theory. Tata McGraw-Hill Education.

Knuth Donald E. (1968). The Art of Computer Programming, Volume 1: Fundamental Algorithms. Addison-Wesley, Reading, MA.

Niven Evan, Zuckerman Herbert S., Montgomery Huge L. (2008). An Introduction to the Theory of Numbers. John Wiley & Sons Inc., U.K.

Pande Neeraj Anant. (2013a). Evolution of Algorithms: A Case Study of Three Prime Generating Sieves, Journal of Science and Arts. 13-3(24): 267-276.

Pande Neeraj Anant. (2013b). Algorithms of Three Prime Generating Sieves Improvised Through Nonprimality of Even Numbers (Except 2). International Journal of Emerging Technologies in Computational and Applied Sciences. 6(4): 274 – 279.

Pande Neeraj Anant. (2013c). Algorithms of Three Prime Generating Sieves Improvised by Skipping Even Divisors (Except 2). American International Journal of Research in Formal, Applied and Natural Sciences. 4(1): 22 – 27.

Schildt Herbert (2006). Java : The Complete Reference (7th Edition). Tata McGraw - Hill Education.