ORIGINAL PAPER

IMPROVED PRIME GENERATING ALGORITHMS BY SKIPPING COMPOSITE DIVISORS AND EVEN NUMBERS (OTHER THAN 2)

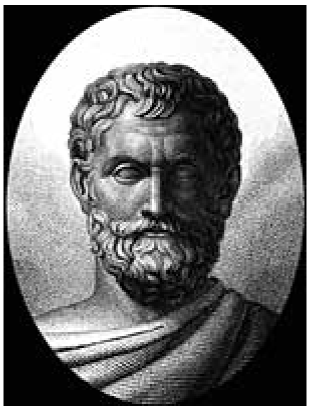
NEERAJ ANANT PANDE[1]

Manuscript received: 19.05.2015; Accepted paper: 03.06.2015;
Published online: 30.06.2015.

Abstract. *In the process of devising better and better sieves for generation of prime numbers, the efforts gradually lead to the classical Sieve of Eratosthenes. The main theme of Sieve of Eratosthenes is that while deciding primality of a positive integer, it is altogether unnecessary to test composite numbers for divisibility. This along with additional property that even numbers other than 2 are never prime, generates refined versions of all; standing on the top of as many as 15 earlier sieves.*

Keywords: *Algorithm, Sieve, Prime Number*
Mathematics Subject Classification (MSC) 2010: *11Y11, 11Y16, 65Y04, 65Y20, 68Q25*

1. INTRODUCTION



In the study of prime numbers, particularly their consecutive generation, the name which inevitably appears is that of Eratosthenes of Cyrene. Chief librarian of the well-known Library of Alexandria, he was a versatile scholar whose talent touched many disciplines like astronomy, geography, geometry, literature, mathematics and music. Being an intellectual denying specializing in only one learning area, he could finish only second in many of his endeavors, and hence was nicknamed by his critics as *Beta* [1]. In spite of this, he has to his credit becoming the first to calculate the circumference of the Earth, to calculate the tilt of the Earth's axis, to predict the distance of the Earth from the Sun, the invention of the leap day and creating the first map of the Earth with parallels and meridians.

Even in the era of advent of many complex domains and theories, prime numbers continue to remain the center of attraction for number theorists. Worthy enough they are for that position as many properties, beginning with their very occurrence amongst natural numbers, are even today mysterious owing to our hitherto inability to fit them in simplistic and precise formulation. In the realm of prime numbers, a lot of work has been undoubtedly done; but much of it is with approximations supported by big oh (O) notation and similar resorts.

Much studied and discussed area is of prime number generation. Sieves are systematic procedures for accomplishing this task. One of the most well-known sieves is that of Eratosthenes. It is worthwhile to note that during the systematic study and gradual evolution of prime generating algorithms in [5], [7] and [9], this sieve; which is very natural in its

[1] Department of Mathematics & Statistics, Yeshwant Mahavidyalaya (College), Nanded-431602, Maharashtra, India. E-mail: napande@gmail.com.

formulation; has evolved as an improvement of as many as 8 sieves, which Eratosthenes could figure out quite straight away.

2. DEVELOPMENT BACKGROUND

Prime number, as is well-known, is a positive integer greater than 1 which has no divisors other than 1 and itself [2].

In the exhaustive efforts to construct gradually improving versions of prime generating sieves, as many as 15 different versions of sieves have been formulated, tested, compared and contrasted in earlier works [5], [6], [7], [8] and [9]. On the top of all these works, combining the advantages of each of them, here 3 new versions of subtypes are presented and the last of it dominates all 17 of them in superiority.

3. SIEVE 3.2.1 – SUPERIOR SIEVE OF SUBTYPE 1

The first property of divisibility is that any positive integer *k* cannot have a positive integral divisor greater than *k*. Sieve subtypes 1 in all earlier referred five works [5], [6], [7], [8] and [9], designated respectively as 1.1.1, 1.2.1, 2.1.1, 2.2.1, and 3.1.1, have a commonality that while determining the primality of numbers *k*, they test all numbers for divisibility (as possible divisors) from 2 to $k-1$. Inter-se relationship and their course of development are as follows:

Sieve subtype 1 in [6] refines the one in [5] by considering only odd numbers after 2.

Sieve subtype 1 in [7] refines the one in [5] by considering only odd divisors after 2.

Sieve subtype 1 in [8] refines both those in [6] and [7] by considering only odd numbers after 2 and only odd divisors after 2.

Sieve subtype 1 in [9] refines both those in [5] and [7] by considering only prime divisors.

Following the approach of work in [6] in improving that in [5] and approach of work in [8] in improving that in [7] and consequently [5], here is the new version of sieve subtype 1, viz., Sieve 3.2.1 :

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

Here, the improvement is achieved through combination of all approaches :

- Only odd numbers are considered for primality (of course after declaring 2 as prime).
- Only prime divisors (which happen to be odd only) are considered for possible divisibility checking.

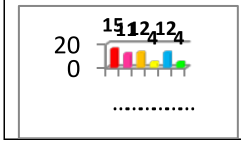
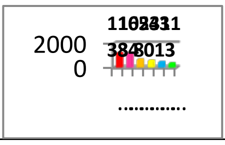
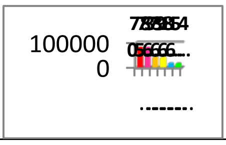
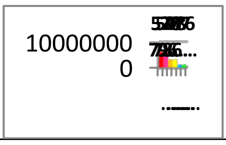
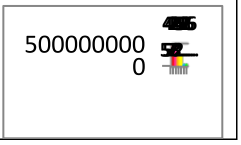
Earlier best sieve subtype 1 (3.1.1) in the previous development line is further improved by considering only half numbers in primality tests, viz., odd numbers (and as exception 2), as all primes (except 2) are necessarily odd.

An exhaustive comparison of all sieves of subtype 1 becomes due. It is presented in terms of the checks required to determine all primes in the specified ranges.

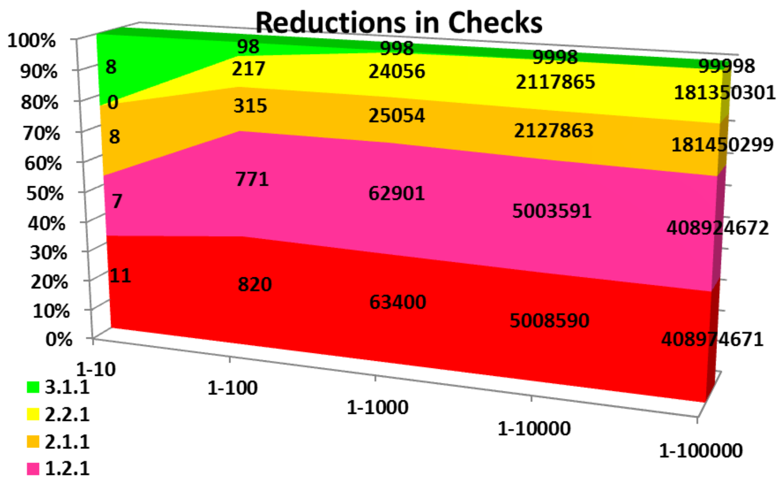
Sr. No.	Range	Number of Primes Found	Checks by Sieve 1.1.1	Checks by Sieve 1.2.1	Checks by Sieve 2.1.1	Checks by Sieve 2.2.1	Checks by Sieve 3.1.1	Checks by Sieve 3.2.1
1	1-10	4	15	11	12	4	12	4
2	1-100	25	1133	1084	628	530	411	313
3	1-1000	168	78022	77523	39676	38678	15620	14622
4	1-10000	1229	5775223	5770224	2894496	2884498	776631	766633
5	1-100000	9592	455189150	455139151	227664778	227564780	46314477	46214479

The data has been generated by implementing the algorithms in Java Programming Language [10] using NetBeans IDE on an electronic computer.

Range-wise graphical comparison of checks done goes this way.

1-10	1-100	1-1000	1-10000	1-100000
				

The reduction in the number of checks over earlier sieves of subtype 1 is as follows :



4. SIEVE 3.2.2 – SUPERIOR SIEVE OF SUBTYPE 2

In deciding primality of k, the method of testing all numbers from 2 to $k - 1$ for divisibility with k is very elementary and in reality those many tests are just not necessary. Since integral divisors of any positive integer k cannot exceed its half, $k/2$, sieve subtypes 1 in [5], [6], [7], [8] and [9] have been refined there only to corresponding sieve subtypes 2, named respectively as 1.1.2, 1.2.2, 2.1.2, 2.2.2, and 3.1.2, by restricting the range of possible divisors to $k/2$. In each sieve subtype 2, the number of checks has reduced to about half as the numbers in the range $k/2$ to $k - 1$ are not tested for divisibility. While this is intra-relationship of sieve subtypes 1 and 2 in each of the corresponding works, the inter-relationships sieve subtypes 2 amongst themselves in different works is as follows.

Sieve subtype 2 in [6] refines that in [5] by neglecting even numbers after 2.

Sieve subtype 2 in [7] refines that in [5] by neglecting even divisors after 2.

Sieve subtype 2 in [8] refines both those in [6] and [7] by neglecting even numbers after 2 and neglecting even divisors after 2.

Sieve subtype 2 in [9] refines both those in [5] and [7] by neglecting composite divisors.

The approach of work in [6] in improving that in [5] and approach of work in [8] in improving that in [7] and consequently [5] opens up a scope here also to go ahead for Sieve 3.2.2. :

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

Here, too the improvement is achieved through combination of all approaches :

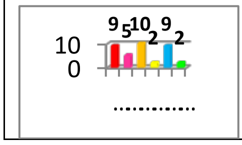
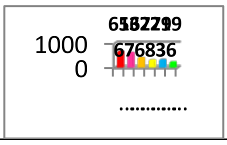
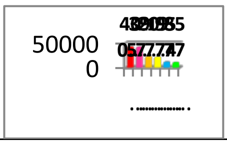
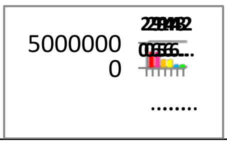
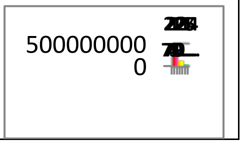
- Even numbers are neglected in determining primality (of course after declaring 2 as prime).
- Composite divisors are omitted in divisibility tests.

Earlier best sieve subtype 2 (3.1.2) in the previous line of development is further improved by neglecting half numbers in primality tests, viz., even numbers (except 2), as no prime (except 2) is even.

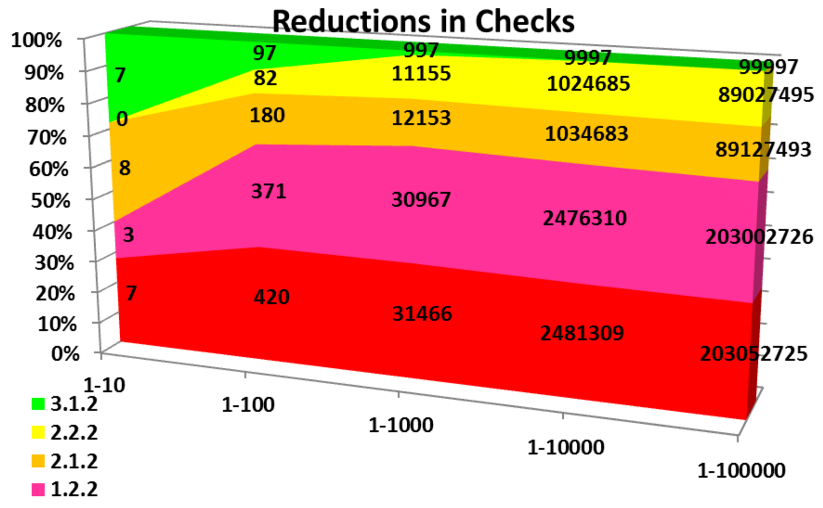
Comparison of number of checks required by all sieves of subtype 2 becomes due.

Sr. No.	Range	Number of Primes Found	Checks by Sieve 1.1.2	Checks by Sieve 1.2.2	Checks by Sieve 2.1.2	Checks by Sieve 2.2.2	Checks by Sieve 3.1.2	Checks by Sieve 3.2.2
1	1-10	4	9	5	10	2	9	2
2	1-100	25	616	567	376	278	293	196
3	1-1000	168	40043	39544	20730	19732	9574	8577
4	1-10000	1229	2907640	2902641	1461014	1451016	436328	426331
5	1-100000	9592	227995678	227945679	114070446	113970448	25042950	24942953

Graphical comparison of checks shows the trend.

1-10	1-100	1-1000	1-10000	1-100000
				

The improvement obtained over earlier subtypes is depicted in following graph.



5. SIEVE 3.2.3 – SUPERIOR SIEVE OF SUBTYPE 3

All sieves of subtype 3 resort to the most promising property of divisibility in this context that positive integral divisors of any positive integer k always occur in pair – whenever d_1 is a divisor, then some d_2 is also a divisor with $k = d_1 \times d_2$, where one of d_1 and d_2 is less than or equal to $\sqrt{k}$ and the other is greater than or equal to $\sqrt{k}$. This hints that it is enough to restrict the search for non-trivial divisors of k in the range of 2 to $\sqrt{k}$ and consequently refines sieve subtypes 1 and 2 in [5], [6], [7], [8] and [9] there only to corresponding sieve subtypes 3, labeled respectively as 1.1.3, 1.2.3, 2.1.3, 2.2.3, and 3.1.3. Doing the same work here we get Sieve 3.2.3. This version being proposed now is even superior to classical Sieve of Eratosthenes. The path track of its evolution is analogues to that of previous subtypes.

Sieve subtype 3 in [6] refines that in [5] by considering odd numbers and neglecting even numbers after 2.

Sieve subtype 3 in [7] refines that in [5] by considering odd divisors and neglecting even divisors after 2.

Sieve subtype 3 in [8] refines both those in [6] and [7] by considering odd numbers and neglecting even numbers after 2 as well as considering odd divisors and neglecting even divisors after 2.

Sieve subtype 3 in [9] refines both those in [5] and [7] by considering prime divisors and neglecting composite divisors.

The approach of work in [6] in improving that in [5] and approach of work in [8] in improving that in [7] and consequently [5] results in Sieve 3.2.3 :

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

```
If checks do not stop for any value of $d$ till $\sqrt{k}$, $k$ is prime
Take next value of $k$
```

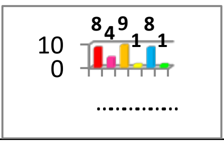
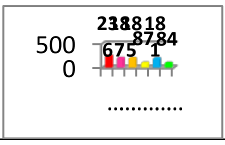
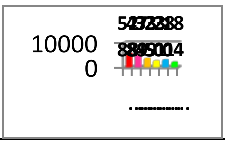
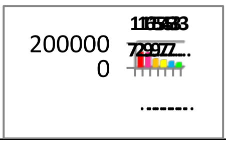
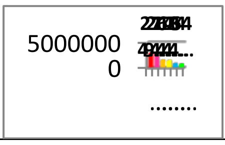
Here, also the betterment comes in through amalgamation of all approaches :

- Only odd numbers are considered and even numbers are neglected for primality (not before declaring 2 as prime).
- Only prime divisors are considered and composite divisors are omitted for possible divisibility checks.

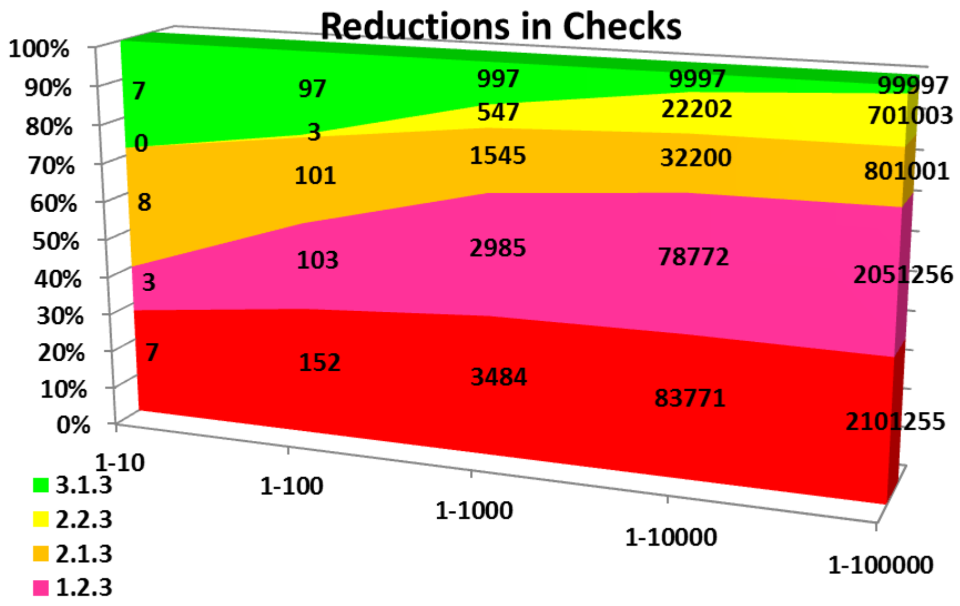
A significant achievement in the performance is reflected in the number of checks taken by this version.

Sr. No.	Range	Number of Primes Found	Checks by Sieve 1.1.3	Checks by Sieve 1.2.3	Checks by Sieve 2.1.3	Checks by Sieve 2.2.3	Checks by Sieve 3.1.3	Checks by Sieve 3.2.3
1	1-10	4	8	4	9	1	8	1
2	1-100	25	236	187	185	87	181	84
3	1-1000	168	5288	4789	3349	2351	2801	1804
4	1-10000	1229	117527	112528	65956	55958	43753	33756
5	1-100000	9592	2745694	2695695	1445440	1345442	744436	644439

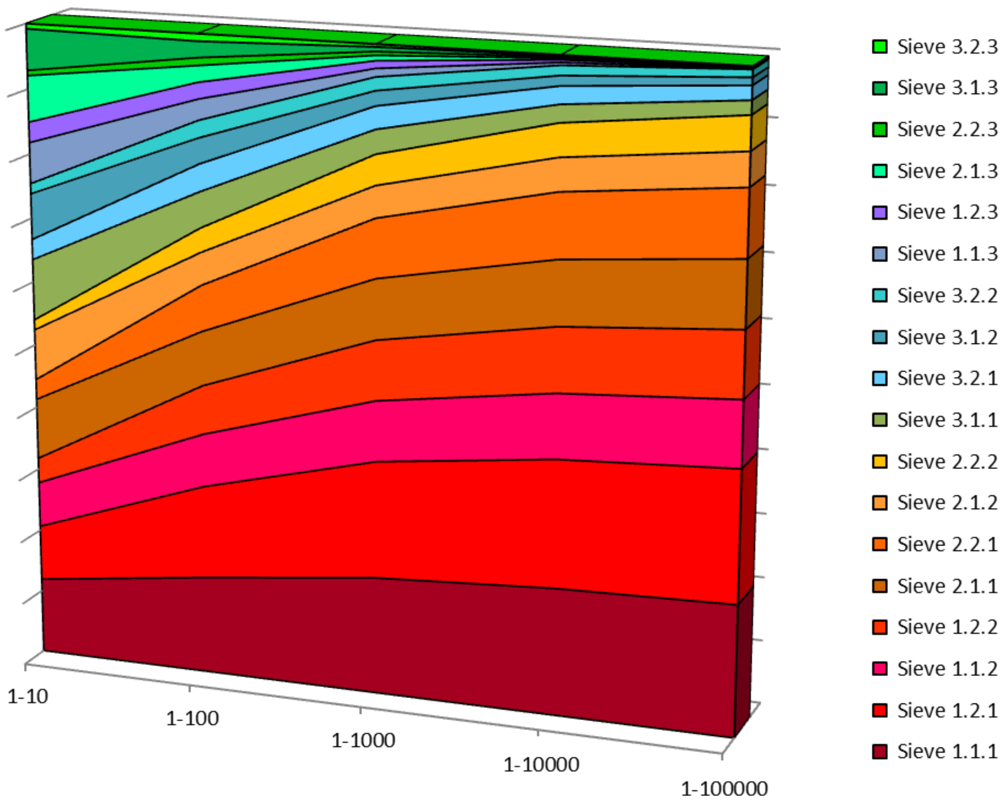
This has following graphical representation.

1-10	1-100	1-1000	1-10000	1-100000
				

Lesser number of checks for finest sieve 3.2.3 than earlier sieves of subtype 3 is as under:



Out of the 18 sieves that have been gradually developed, the combined performance analysis shows that, for higher ranges of numbers, their ranking is as follows, starting with most efficient (taking minimum number of checks) going to lesser and lesser ones (requiring more checks) :



Acknowledgements

The author is thankful to the Java (7 Update 25) Programming Language Development Team and the NetBeans IDE 7.3.1 Development Team, whose software are freely used in implementing the algorithms.The author expresses gratitude 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

[1] Asimov, I., *Asimov's Biographical Encyclopedia of Science and Technology*, New Revised Edition, Pan Books Ltd, London, 1975.
[2] Burton, D.M., *Elementary Number Theory*, Tata McGraw-Hill Education, 2007.
[3] Knuth, D.E., *The Art of Computer Programming, Volume 1: Fundamental Algorithms*, Addison-Wesley, Reading, MA, 1968.
[4] Niven, E., Zuckerman, H.S., Montgomery, H.L., *An Introduction to the Theory of Numbers*, John Wiley & Sons Inc., U.K., 2008.
[5] Pande, N.A., *Journal of Science and Arts*, **3**(24), 267, 2013.

[6] Pande, N.A., *International Journal of Emerging Technologies in Computational and Applied Sciences*, **4**(6), 274, 2013.
[7] Pande, N.A., *American International Journal of Research in Formal, Applied and Natural Sciences*, **1**(4), 22, 2013.

[8] Pande, N.A., *Journal of Natural Sciences*, **2**(1), 107, 2014.
[9] Pande, N.A., *International Journal of Computer Science & Engineering Technology*, **5**(9), 935, 2014.
[10] Schildt, H., *Java : The Complete Reference, 7th Edition*, Tata McGraw - Hill Education, 2006.