Prime Numbers and Prime Factorization

Josephine Kraft

Ladislaus von Bortkiewicz Chair of Statistics
Humboldt–Universität zu Berlin
http://lvb.wiwi.hu-berlin.de

Motivation: Cryptography

Motivation: Cryptography

- **Secret key**: two prime numbers p and q
- **Public key**: $p \cdot q$ is used to encrypt messages
- Decrypt messages with secret key
- Algorithms that find the prime numbers take very long, when only the product is known
- Algorithms that find large prime numbers are needed

Outline

1. Introduction
2. Prime Factorization
3. Finding Prime Numbers
   - 3.1 Sieve of Eratosthenes
   - 3.2 Sieve of Atkin and Bernstein

What are Prime Numbers?

- A **prime number** is an integer $p > 1$ which has exactly two positive divisors, namely 1 and p
- An integer n is **composite** or **non prime** if and only if it admits a nontrivial factorization $n = ab$, where a and b are integers, $1 < a, b < n$
- The resulting sequence of prime numbers is: 2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, ...

Prime Factorization

Theorem (Fundamental Theorem of Arithmetic)

For each natural number n there is a unique factorization

$$n = \prod_{i=1}^{k} p_i^{a_i}$$

where all a_i are positive integers and $p_1, \ldots, p_k$ are primes.

$\rightarrow$ **Fundamental Problem of Arithmetic**

Sieve of Eratosthenes

~~1~~	2	3	~~4~~	5	~~6~~	7	~~8~~	~~9~~	~~10~~
11	~~12~~	13	~~14~~	~~15~~	~~16~~	17	~~18~~	19	~~20~~
~~21~~	~~22~~	23	~~24~~	~~25~~	~~26~~	~~27~~	~~28~~	29	~~30~~
31	~~32~~	~~33~~	~~34~~	~~35~~	~~36~~	37	~~38~~	~~39~~	~~40~~
41	~~42~~	43	~~44~~	~~45~~	~~46~~	47	~~48~~	~~49~~	~~50~~
~~51~~	~~52~~	53	~~54~~	~~55~~	~~56~~	~~57~~	~~58~~	59	~~60~~
61	~~62~~	~~63~~	~~64~~	~~65~~	~~66~~	67	~~68~~	~~69~~	~~70~~
71	~~72~~	73	~~74~~	~~75~~	~~76~~	~~77~~	~~78~~	79	~~80~~
~~81~~	~~82~~	83	~~84~~	~~85~~	~~86~~	~~87~~	~~88~~	89	~~90~~
~~91~~	~~92~~	~~93~~	~~94~~	~~95~~	~~96~~	97	~~98~~	~~99~~	~~100~~

Algorithm

- Aim: Find all prime numbers up to $N = 30$
- Cross from list all multiples of each prime number, starting with the multiples of 2

~~1~~	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30

~~1~~	2	3	~~4~~	5	~~6~~	7	~~8~~	9	~~10~~
11	~~12~~	13	~~14~~	15	~~16~~	17	~~18~~	19	~~20~~
21	~~22~~	23	~~24~~	25	~~26~~	27	~~28~~	29	~~30~~

Algorithm

~~1~~	2	3	~~4~~	5	~~6~~	7	~~8~~	~~9~~	~~10~~
11	~~12~~	13	~~14~~	~~15~~	~~16~~	17	~~18~~	19	~~20~~
~~21~~	~~22~~	23	~~24~~	25	~~26~~	~~27~~	~~28~~	29	~~30~~

~~1~~	2	3	~~4~~	5	~~6~~	7	~~8~~	~~9~~	~~10~~
11	~~12~~	13	~~14~~	~~15~~	~~16~~	17	~~18~~	19	~~20~~
~~21~~	~~22~~	23	~~24~~	~~25~~	~~26~~	~~27~~	~~28~~	29	~~30~~

~~1~~	2	3	~~4~~	5	~~6~~	7	~~8~~	~~9~~	~~10~~
11	~~12~~	13	~~14~~	~~15~~	~~16~~	17	~~18~~	19	~~20~~
~~21~~	~~22~~	23	~~24~~	~~25~~	~~26~~	~~27~~	~~28~~	29	~~30~~

R Code

```
eratosthenes = function(n) {

  x = c(2:n) # list of numbers from 2 to n
  p = 2
  r = c()    # results vector

  while (p*p < n) {
    r = c(r,x[1])                   # first element of n is always prime
    x = x[-(which(x %% p ==0))]     # elements with remainder 0 are multiples of p
    p = x[1]                        # first element of n will always be prime
  }
   return(c(r,x))   # all remaining elements in n are prime
}

```

R Code: Results

```
>
> eratosthenes(100)
 [1]  2  3  5  7 11 13 17 19 23 29 31 37 41 43 47 53 59 61 67 71 73 79 83 89 97
> length(eratosthenes(100))
[1] 25
> eratosthenes(1000)
  [1]   2   3   5   7  11  13  17  19  23  29  31  37  41  43  47  53  59  61  67  71
 [21]  73  79  83  89  97 101 103 107 109 113 127 131 137 139 149 151 157 163 167 173
 [41] 179 181 191 193 197 199 211 223 227 229 233 239 241 251 257 263 269 271 277 281
 [61] 283 293 307 311 313 317 331 337 347 349 353 359 367 373 379 383 389 397 401 409
 [81] 419 421 431 433 439 443 449 457 461 463 467 479 487 491 499 503 509 521 523 541
[101] 547 557 563 569 571 577 587 593 599 601 607 613 617 619 631 641 643 647 653 659
[121] 661 673 677 683 691 701 709 719 727 733 739 743 751 757 761 769 773 787 797 809
[141] 811 821 823 827 829 839 853 857 859 863 877 881 883 887 907 911 919 929 937 941
[161] 947 953 967 971 977 983 991 997
> length(eratosthenes(1000))
[1] 168
>
```

Sieve of Atkin and Bernstein

- Introduced in 2003 by Arthur O.L. Atkin and Daniel J. Bernstein
- All primes (except 2 and 3) are of one of the following forms:
  - **Group 1:** $n \equiv 1 \pmod{4} \iff n = 4k + 1$
  - **Group 2:** $n \equiv 1 \pmod{6} \iff n = 6k + 1$
  - **Group 3:** $n \equiv 11 \pmod{12} \iff n = 12k + 11$

  with $k \in \mathbb{N}_0$
- The algorithm is based on three theorems, one theorem for each group of primes

Theory

Theorem (1)

Let n be a squarefree positive integer with $n \equiv 1 \pmod 4$. *Then n is prime if and only if the cardinality of the following set is odd:*

$$\{(x, y) \mid 4x^2 + y^2 = n,\ x, y \in \mathbb{N}\}$$

Remark:

A **squarefree number** is an integer, which is divisible by no other square number than 1.

Equivalently, an integer is squarefree if and only if in its prime factorization no prime factor occurs more than once.

Theory

Theorem (2)

Let n be a squarefree positive integer with $n \equiv 1 \pmod{6}$. Then n is prime if and only if the cardinality of the following set is odd:

$$\{(x, y) \mid 3x^2 + y^2 = n,\ x, y \in \mathbb{N}\}$$

Theorem (3)

Let n be a squarefree positive integer with $n \equiv 11 \pmod{12}$. Then n is prime if and only if the cardinality of the following set is odd:

$$\{(x, y) \mid 3x^2 - y^2 = n,\ x, y \in \mathbb{N}, x > y\}$$

Theory

- All numbers of the 3 groups can be represented in the form $60k + r$, with varying values for r

Theory

- All numbers of the 3 groups can be represented in the form $60k + r$, with varying values for r
- Modified groups to consider in algorithm:

Theory

- All numbers of the 3 groups can be represented in the form $60k + r$, with varying values for r
- Modified groups to consider in algorithm:
  - **1. Group:** $n = 60k + r$, where $k \in \mathbb{N}_0$ and $r \in \{1, 13, 17, 29, 37, 41, 49, 53\}$

Theory

- All numbers of the 3 groups can be represented in the form $60k + r$, with varying values for r
- Modified groups to consider in algorithm:
  - **1. Group:** $n = 60k + r$, where $k \in \mathbb{N}_0$ and $r \in \{1, 13, 17, 29, 37, 41, 49, 53\}$
  - **2. Group:** $n = 60k + r$, where $k \in \mathbb{N}_0$ and $r \in \{7, 19, 31, 43\}$

Theory

- All numbers of the 3 groups can be represented in the form $60k + r$, with varying values for r
- Modified groups to consider in algorithm:
  - **1. Group:** $n = 60k + r$, where $k \in \mathbb{N}_0$ and $r \in \{1, 13, 17, 29, 37, 41, 49, 53\}$
  - **2. Group:** $n = 60k + r$, where $k \in \mathbb{N}_0$ and $r \in \{7, 19, 31, 43\}$
  - **3. Group:** $n = 60k + r$, where $k \in \mathbb{N}_0$ and $r \in \{11, 23, 47, 59\}$

Theory

Theorem (1)

Let n be a squarefree positive integer with $n \equiv 1 \pmod 4$. Then n is prime if and only if $\#\{(x,y) \mid 4x^2 + y^2 = n,\ x,y \in \mathbb{N}\}$ is odd.

Theorem (2)

Let n be a squarefree positive integer with $n \equiv 1 \pmod 6$. Then n is prime if and only if $\#\{(x,y) \mid 3x^2 + y^2 = n,\ x,y \in \mathbb{N}\}$ is odd.

Theorem (3)

Let n be a squarefree positive integer with $n \equiv 11 \pmod{12}$. Then n is prime if and only if $\#\{(x,y) \mid 3x^2 - y^2 = n,\ x,y \in \mathbb{N},\ x > y\}$ is odd.

Algorithm

1. Create sieve list with N entries, where each number is initially marked as non prime.

Algorithm

1. Create sieve list with N entries, where each number is initially marked as non prime.
2. For each $n \leq N$: Calculate modulo-60 remainder of n.

Algorithm

1. Create sieve list with N entries, where each number is initially marked as non prime.
2. For each $n \leq N$: Calculate modulo-60 remainder of n.
   - **If $r \in \{1, 13, 17, 29, 37, 41, 49, 53\}$**:
     Change n^{th} entry in sieve list from prime to non prime (or vice versa) $\forall$ (x,y) with $4x^2 + y^2 = n$, $x, y \in \mathbb{N}$.

Algorithm

1. Create sieve list with N entries, where each number is initially marked as non prime.
2. For each $n \leq N$: Calculate modulo-60 remainder of n.
   - **If $r \in \{1, 13, 17, 29, 37, 41, 49, 53\}$**:
     Change n^{th} entry in sieve list from prime to non prime (or vice versa) $\forall$ (x,y) with $4x^2 + y^2 = n$, $x, y \in \mathbb{N}$.
   - **If $r \in \{7, 19, 31, 43\}$**:
     Change n^{th} entry in sieve list from prime to non prime (or vice versa) $\forall$ (x,y) with $3x^2 + y^2 = n$, $x, y \in \mathbb{N}$.

Algorithm

1. Create sieve list with N entries, where each number is initially marked as non prime.
2. For each $n \leq N$: Calculate modulo-60 remainder of n.
   - **If $r \in \{1, 13, 17, 29, 37, 41, 49, 53\}$**:
     Change n^{th} entry in sieve list from prime to non prime (or vice versa) $\forall$ (x,y) with $4x^2 + y^2 = n$, $x, y \in \mathbb{N}$.
   - **If $r \in \{7, 19, 31, 43\}$**:
     Change n^{th} entry in sieve list from prime to non prime (or vice versa) $\forall$ (x,y) with $3x^2 + y^2 = n$, $x, y \in \mathbb{N}$.
   - **If $r \in \{11, 23, 47, 59\}$**:
     Change n^{th} entry in sieve list from prime to non prime (or vice versa) $\forall$ (x,y) with $3x^2 - y^2 = n$, $x, y \in \mathbb{N}$.

Algorithm

3. There are still non-squarefree integers that could be marked as prime → Start with lowest number p marked as prime. Change entry for all multiples of the square of p $(p^2, 2p^2, 3p^2, 4p^2, ...)$ to non prime. Repeat until $p^2 > N$.

R Code

```r
atkin = function(n){

  s = c(1,7,11,13,17,19,23,29,31,37,41,43,47,49,53,59)

  L = 16*ceiling(n/60)
  to_test = matrix(numeric(L), ncol=ceiling(n/60))

  for (i in 1:ceiling(n/60))
  {
    for (j in 1:16)
    {
      to_test[j,i] = 60*(i-1)+s[j]
    }
  }
  to_test = as.vector(to_test)
  to_test = to_test[to_test<=n]    # numbers to check for primality
  is_prime = as.logical(numeric(length(to_test)))   # initially marked as FALSE for all numbers

  rest = to_test %% 60              # vector of modulo-60 remainders
  rest1 = c(1,13,17,29,37,41,49,53)  # remainders group 1
  rest2 = c(7,19,31,43)              # remainders group 2
  rest3 = c(11,23,47,59)             # remainders group 3

  for (i in 1:length(to_test))
  {
    if (is.element(rest[i],rest1))    # check: remainder in group 1
    {
      fun = function(y) sqrt((to_test[i]-y^2)/4)  # quadratic form as function of y
      x = fun(1:floor(sqrt(to_test[i])))          # calculate corresponding x values
      k = x[x > 0] == round(x[x > 0])
      t = sum(k)                                  # numbers of integer x-values > 0

      if(t > 0 && !(round(t/2) == t/2)) is_prime[i] = TRUE
    }
```

R Code

```
    if (is.element(rest[i],rest2))    # check: remainder in group 2
    {
      fun = function(y) sqrt((to_test[i]-y^2)/3)
      x = fun(1:floor(sqrt(to_test[i])))
      k = x[x > 0] == round(x[x > 0])
      t = sum(k)

      if(t > 0 && !(round(t/2) == t/2)) is_prime[i] = TRUE
    }

    if (is.element(rest[i],rest3))    # check: remainder in group 3
    {
      fun = function(y) sqrt((to_test[i]+y^2)/3)
      x = fun(1:floor(sqrt(to_test[i])))
      k = x[x > 0 & x > 1:floor(sqrt(to_test[i]))] == round(x[x > 0 & x > 1:floor(sqrt(to_test[i]))])
      t = sum(k)

      if(t > 0 && !(round(t/2) == t/2)) is_prime[i] = TRUE
    }
  }
  primes = to_test[is_prime]
  primes_loop = to_test[is_prime]

  i=1
  while (primes_loop[i]^2 <= n)
  {
    primes2 = primes/primes[i]^2
    primes = primes[primes2 != round(primes2)]    # remove all non-squarefree numbers
    i=i+1
  }
  primes = c(2,3,5,primes)
  return(primes)
}
```

R Code: Results

```
>
> atkin(100)
 [1]  2  3  5  7 11 13 17 19 23 29 31 37 41 43 47 53 59 61 67 71 73 79 83 89 97
> length(atkin(100))
[1] 25
> atkin(1000)
  [1]   2   3   5   7  11  13  17  19  23  29  31  37  41  43  47  53  59  61  67  71
 [21]  73  79  83  89  97 101 103 107 109 113 127 131 137 139 149 151 157 163 167 173
 [41] 179 181 191 193 197 199 211 223 227 229 233 239 241 251 257 263 269 271 277 281
 [61] 283 293 307 311 313 317 331 337 347 349 353 359 367 373 379 383 389 397 401 409
 [81] 419 421 431 433 439 443 449 457 461 463 467 479 487 491 499 503 509 521 523 541
[101] 547 557 563 569 571 577 587 593 599 601 607 613 617 619 631 641 643 647 653 659
[121] 661 673 677 683 691 701 709 719 727 733 739 743 751 757 761 769 773 787 797 809
[141] 811 821 823 827 829 839 853 857 859 863 877 881 883 887 907 911 919 929 937 941
[161] 947 953 967 971 977 983 991 997
> length(atkin(1000))
[1] 168
>
```

Finding Large Prime Numbers

Test for $N = 10^8$	**Eratosthenes**	**Atkin & Bernstein**
Primes found	5,761,455	5,761,455
Largest Prime	99,999,989	99,999,989
Running Time	4.5 min	6.2 hours

Test for $N = 10^9$	**Eratosthenes**
Primes found	50,847,534
Largest Prime	999,999,937
Running Time	1.7 hours

Thank you for your Attention!

Enter any 11-digit prime number to continue ...

Bibliography

A. O. L. Atkin, D. J. Bernstein
Prime Sieves using binary quadratic forms
available on http://www.ams.org/journals/mcom, 2003.

R. Crandall, C. Pomerance
Prime Numbers. A Computational Perspective
Springer 2005.

A. Joux
Algorithmic Cryptanalysis
Chapman Hall/CRC 2009.