Prime Group Percentages

2018-01-08 BY GENE

As usual, I am curious about the distribution of the primes. This morning I wondered what percent of numbers with N digits were prime. Not an earth shattering curiosity… Naturally, I wrote a little perl program to answer it for me. Here is the preamble:

```
#!/usr/bin/env perl
use strict;
use warnings;

use Math::Prime::XS qw( is_prime );
```

This says that we are a Perl program and that we will test primality of numbers.

Next the program sets up some parameters and buckets for our numbers that have been seen. Max is the number of integers we will look at. N is the current integer we are inspecting (from 1 to max).

```
my $max = shift || 1000;

my %all;
my %primes;
my $n = 0;
```

And now for the first bite of the meat of the program:

```
while( $n < $max ) {
    $n++;

    my $key = length $n;

    $all{$key}++;

    next unless is_prime($n);

    $primes{$key}++;

    print "$n ($key)\n";
}
```

This loops over each number, “n”, deciding if the number is prime or not.

The “all” bucket is incremented for each length of “n” (the digit size). If “n” is prime, the program increments the digit size of the “primes” bucket as well. (That is, 3 is of length (digit size) 1, 11 is length 2, etc.)

Additionally, the current number and its digit size are printed out. This is just for convenience, to show that “Yes, we are still running!” But this actually slows down the algorithm… So comment-out if desired!

With these buckets filled, the final section of the program is to compute the percentages seen, and print out the results:

```
for my $key ( sort { $a <=> $b } keys %all ) {
    if ( exists $primes{$key} ) {
        my $perc = sprintf '%.2f', $primes{$key} / $all{$key} * 100;

        print "n = $key, primes = $primes{$key}, all = $all{$key} ==> $perc
    }
}
```

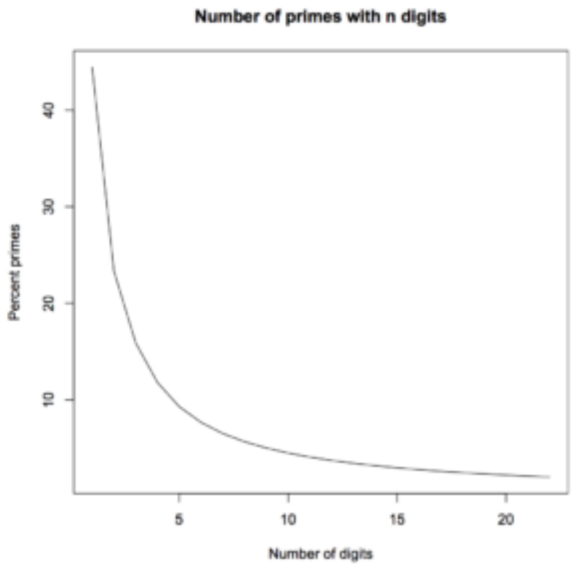
This looks at each digit size of all numbers, sorted from 1 ascending. If the size is in the “primes” bucket, then compute the percentage of times prime numbers were seen for that digit size.

And what are the results when run for all numbers below 1,000,000,000? Well, this algorithm in Perl is slow, for one!

```
n = 1, primes = 4, all = 9 ==> 44.44%
n = 2, primes = 21, all = 90 ==> 23.33%
n = 3, primes = 143, all = 900 ==> 15.89%
n = 4, primes = 1061, all = 9000 ==> 11.79%
n = 5, primes = 8363, all = 90000 ==> 9.29%
n = 6, primes = 68906, all = 900000 ==> 7.66%
n = 7, primes = 586081, all = 9000000 ==> 6.51%
n = 8, primes = 5096876, all = 90000000 ==> 5.66%
n = 9, primes = 45086079, all = 900000000 ==> 5.01%
```

This means that there are 4 single digit primes out of 9 total numbers, which equals 44.44%. There are 21 two digit primes out of 90 total, etc.

Descending, asymptotically:



This is the “Number of primes with n digits” sequence, which goes up to 22 digits, at the OEIS.

FILED UNDER: MATHEMATICS, SOFTWARE
TAGGED WITH: PERL, PRIME NUMBERS

Epistemologist-at-large

^ Top