Prime Sum

To do a sum with only prime terms, start by defining the prime counting function.

In number theory the prime counting function $\pi(n)$ gives the number of primes less than or equal to n. The factorization key (fctr) from screen 6 can be used to compute $\pi(n)$. When we do

k, fctr

the function returns k in the y-register on the stack and a factor, usually the smallest, is displayed in the x-register. Then doing a division removes the factor just found and the fctr function can be used again to continue the factorization.

For computing $\pi(n)$, we can exchange x and y, divide, and then apply the floor function. This will give 0 if n is composite and 1 if n is prime, since only when n is prime will x and y be equal.
Define function f2 to sum this function from 2 to n, and the result will be $\pi(n)$.

f1: fctr, x↔y, /, 2, func, floor

f2: 7, func, 2, x↔y, 1, enter, 1, sum

To test this, try

1e5, enter, 2, $\mathrm{f_n}$

giving 9592, the number of primes up to 100,000.

Now approximate this sum where p runs over all the primes.

$$S = \sum_p \frac{1}{p^2} = \frac{1}{2^2} + \frac{1}{3^2} + \frac{1}{5^2} + \frac{1}{7^2} + \frac{1}{11^2} + \cdots$$

Define f3 like f1 in the example above, but to return $1/j^2$ if j is prime and zero if j is composite. f4 will sum this function from 2 to n

f3: 1, func, 1, sto, $\mathrm{x^2}$, 1/x, 1, rcl, 6, func, fctr, x↔y, /, 2, func, floor, $*$

f4: 7, func, 2, x↔y, 1, enter, 3, sum

Sum this series out to 10^2, 10^3, 10^4, and 10^5 terms.

30, fix, e2, enter, 4, $\mathrm{f_n}$	0.450428788263752450238135415562
e3, enter, 4, $\mathrm{f_n}$	0.452120430249304581998945097122
e4, enter, 4, $\mathrm{f_n}$	0.452237604339950325970565053705
e5, enter, 4, $\mathrm{f_n}$	0.452246617792053878716149984159

It looks like 10^2 terms gave 2 digits correct, 10^3 gave 3 digits, 10^4 gave 4 digits (almost 5). We might guess that 10^5 terms has 5 (maybe 6) digits correct.

On the "Infinite sums" page we had no trouble doing sums with 10^8, 10^{10}, or more terms. That won't work here, because f3(j) is not a smooth function. It jumps from zero to being positive when j is a prime. That means the sum function is not able to analyze f3 and avoid adding all the terms.

Maybe extrapolation can do better. See the "Extrapolation" page for more examples and explanation of how the extr function works. There we extrapolated using the first 25, 50, 100, 200, and 400 terms in the sum and monitored the estimated relative error in each case.

That doesn't work here, because f3(k) is zero most of the time and that makes many consecutive partial sums exactly the same. When lots of consecutive elements in a sequence are identical, the extrapolation methods can be fooled into thinking the sequence has converged to full precision and that is the limit.

Here we have to use many more terms, to force the extrapolation methods to take a sample of the partial sums that is more spread out. Try starting with 10,000 terms and using 10e3, 25e3, 50e3, and 100e3 terms. This is the last result:

7, func, 1, enter, 2, enter, 100e3, enter, 3, extr　　0.452246609004423079705710754022

The relative error estimates for those 4 cases are 3e−5, 1e−6, 5e−6, and 2e−6.

It looks like extrapolation has not helped very much in this case. The primes are irregular in their spacing, and this may mean the convergence of the series above is not smooth enough for extrapolation to work well.

Compare a related but simpler series.

$$S = \sum_{n=1}^{\infty} \left(\frac{1}{n^2} \right) = \frac{1}{1^2} + \frac{1}{2^2} + \frac{1}{3^2} + \frac{1}{4^2} + \frac{1}{5^2} + \cdots$$

This is the series from example 2 of the "Infinite sums" page, where we used the sum function to get over 50 digits correct for S. See if extrapolation does better for this series than the one with only primes terms.

f5 is this new function.

f5: 1, func, x^2, 1/x

Use 200 function values.

40, fix, 7, func, 1, enter, 1, enter, 200, enter, 5, extr　　1.6449340668482264364724151666460251892189

10, sci, x↔y　　(estimated error)　　2.568740097e−49

x↔y, π, 1, func, x^2, 6, /, −　　(true error)　　−1.000000000e−56

Extrapolation was able to get over 50 digits correct. The terms in this sum are getting small in a much more regular way than those of the prime series, so extrapolation is able to get a much more accurate estimate of the sum of the series.