Reliable multiprecision arithmetic for number theory

Fredrik Johansson

LFANT seminar, IMB / INRIA

2014-09-16

Where I come from

Malung (population $\approx$ 5000) in Dalarna, Sweden

Things I've done previously

Software

- Since 2007: **mpmath**, a Python library for arbitrary-precision floating-point arithmetic
- Since 2010: **FLINT**, a C library for number theory (coauthor)
- Since 2012: **Arb**, a C library for arbitrary-precision ball arithmetic

Education

- 2004-2010: MSc in **engineering physics**, Chalmers University of Technology, Gothenburg, Sweden
- 2010-2014: PhD in **symbolic computation**, RISC, Linz, Austria

My current interest

Fast arithmetic mainly in

$$\mathbb{R}, \quad \mathbb{C}, \quad \mathbb{R}[x], \quad \mathbb{C}[x], \quad \mathbb{R}[[x]], \quad \mathbb{C}[[x]]$$

with **rigorous error bounds**.

Fast in precision: ideally $O(p^{1+\varepsilon})$

Fast in polynomial degree: ideally $O(n^{1+\varepsilon})$

Fast in both: ideally $O((np)^{1+\varepsilon})$

Fast in **practice**: low implementation overhead, different algorithms depending on size

Representing real numbers

Floating-point: 3.141592653589793

MPFR, MPC, many others...

Interval (inf-sup): $[3.141592653589793, 3.141592653589794]$

MPFI, ...

Ball (mid-rad): $3.141592653589793 \pm 10^{-15}$

iRRAM, Mathemagix, Arb

A benchmark: power series arithmetic

Computing the Taylor expansion of $\exp(\exp(1+x))$ to order x^n at a precision of n decimal digits.

n	Pari/GP	Arb	Faster
10	0.000014	0.0000113	1.2×
30	0.000066	0.0000583	1.1×
100	0.00105	0.000859	1.2×
300	0.0273	0.0116	2.3×
1000	1.53	0.180	8.5×
3000	69	2.313	29×
10000		29.81	

Time in seconds

Open problem: semantics for ball arithmetic

What should the output of $\sin(10 \pm 3)$ be?

$-0.544021110889370 \pm 3.001$ ("best midpoint, generic radius")

$-0.544021110889370 \pm 1.545$ ("best midpoint, best radius")

-0.544 ± 3.002 ("trimmed best midpoint, generic radius")

-0.544 ± 1.545 ("trimmed best midpoint, best radius")

0 ± 1 ("best ball")

These results are quite different, and one can think of applications where either one is superior to the others.

Open problem: transcendental functions

Typical function:

$$f(z) = \sum_{k=0}^{\infty} c_k(z) = \sum_{k=0}^{N} c_k(z) + \varepsilon(N, z)$$

- Bounding the error in evaluating $\sum_{k=0}^{N} c_k(z)$
  - Trivial in principle with ball arithmetic
  - If z is inexact, may need to be more clever for good output
- Bounding the remainder $\varepsilon(N, z)$
  - Bounds in the literature are often not general enough, lack explicit constants, are computationally ineffective, or simply not available...
- Efficiency
  - Choosing the right algorithm on each subdomain
  - Asymptotic complexity, practical efficiency

The Hurwitz zeta function

$$\zeta(s, a) = \sum_{k=0}^{\infty} \frac{1}{(k+a)^s} \qquad s, a \in \mathbb{C}$$

Special cases: Riemann zeta ($a = 1$), Dirichlet L-functions, polylogarithms, ...

Goal: compute $\zeta(s, a)$ and derivatives with respect to s, to arbitrary precision, with rigorous error bounds

The Euler-Maclaurin formula

$$\sum_{k=N}^{U} f(k) = I + T + R$$

$$I = \int_N^U f(t)\,dt$$

$$T = \frac{1}{2}\left(f(N) + f(U)\right) + \sum_{k=1}^{M} \frac{B_{2k}}{(2k)!}\left(f^{(2k-1)}(U) - f^{(2k-1)}(N)\right)$$

$$R = -\int_N^U \frac{\tilde{B}_{2M}(t)}{(2M)!} f^{(2M)}(t)\,dt$$

Computing $\zeta(s, a)$ using Euler-Maclaurin

$$\zeta(s,a) = \underbrace{\sum_{k=0}^{N-1} f(k)}_{S} + \underbrace{\sum_{k=N}^{\infty} f(k)}_{I+T+R}, \quad f(k) = \frac{1}{(a+k)^s}$$

For derivatives, substitute $s \to s + x \in \mathbb{C}[[x]]$:

$$f(k) = \frac{1}{(a+k)^{s+x}} = \sum_{i=0}^{\infty} \frac{(-1)^i \log^i(a+k)}{i!(a+k)^s} x^i \in \mathbb{C}[[x]]$$

Parts to evaluate

$$S = \sum_{k=0}^{N-1} \frac{1}{(a+k)^{s+x}}$$

$$I = \int_N^\infty \frac{1}{(a+t)^{s+x}}\,dt = \frac{(a+N)^{1-(s+x)}}{(s+x)-1}$$

$$T = \frac{1}{(a+N)^{s+x}}\left(\frac{1}{2} + \sum_{k=1}^{M} \frac{B_{2k}}{(2k)!}\frac{(s+x)_{2k-1}}{(a+N)^{2k-1}}\right)$$

$$R = -\int_N^\infty \frac{\tilde{B}_{2M}(t)}{(2M)!}\frac{(s+x)_{2M}}{(a+t)^{(s+x)+2M}}\,dt \quad \text{(bound)}$$

Bounding the remainder

$$
\begin{aligned}
|R| &= \left| \int_N^\infty \frac{\tilde{B}_{2M}(t)}{(2M)!} \frac{(s+x)_{2M}}{(a+t)^{s+x+2M}} dt \right| \\
&\leq \int_N^\infty \left| \frac{\tilde{B}_{2M}(t)}{(2M)!} \frac{(s+x)_{2M}}{(a+t)^{s+x+2M}} \right| dt \\
&\leq \frac{4\,|(s+x)_{2M}|}{(2\pi)^{2M}} \int_N^\infty \left| \frac{dt}{(a+t)^{s+x+2M}} \right| \in \mathbb{R}[[x]]
\end{aligned}
$$

$$
\int_N^\infty \left| \frac{dt}{(a+t)^{s+x+2M}} \right| = \sum_{k=0}^\infty \left(\int_N^\infty \frac{dt}{k!} \left| \frac{\log(a+t)^k}{(a+t)^{s+2M}} \right| \right) x^k
$$

A sequence of integrals

For $k \in \mathbb{N}$, $A > 0, B > 1, C \geq 0$,

$$J_k(A, B, C) \equiv \int_A^\infty t^{-B}(C + \log t)^k dt$$
$$= \frac{L_k}{(B-1)^{k+1} A^{B-1}}$$

where

$$L_0 = 1, \; L_k = kL_{k-1} + D^k$$
$$D = (B-1)(C + \log A)$$

Error bound

Theorem: for complex numbers $s = \sigma + \tau i$, $a = \alpha + \beta i$ and positive integers N, M such that $\alpha + N > 1$ and $\sigma + 2M > 1$,

$$|R| \leq \frac{4\,|(s+x)_{2M}|}{(2\pi)^{2M}} \left| \sum_{k=0}^{\infty} R_k x^k \right| \in \mathbb{R}[[x]]$$

where $R_k \leq (K/k!)\, J_k(N+\alpha, \sigma+2M, C)$ and K and C are certain numbers given explicitly in terms of s, a, N, M.

Evaluation steps

To evaluate $\zeta(s, a)$ with an error of 2^{-p}:

1. Choose $N, M = O(p)$, bound the error term R
2. Compute the power sum S
3. Compute the integral I
4. Compute the Bernoulli numbers
5. Compute the tail T

Observation: the first n derivatives of $\zeta(s, a)$ can be simultaneously computed to n digits of precision in $O(n^{2+\varepsilon})$ time (**quasi-optimal**).

Fast power series power sum

$$V = \begin{bmatrix} 1 & \log(a) & \cdots & \log^n(a) \\ 1 & \log(a+1) & \cdots & \log^n(a+1) \\ \vdots & \vdots & \ddots & \vdots \\ 1 & \log(a+n) & \cdots & \log^n(a+n) \end{bmatrix}$$

$$Y = \begin{bmatrix} a^{-s} & (a+1)^{-s} & \dots & (a+n)^{-s} \end{bmatrix}^T$$

VY is **multipoint evaluation**. We want $V^T Y$. An algorithm with $O(n^{2+\varepsilon})$ time complexity exists by the **transposition principle** (not yet implemented).

My implementation: $O(n^{3+\varepsilon})$, but supports **parallelization** ($\approx 16\times$ speedup on 16 cores).

Fast power series tail

$T = \sum_{k=1}^{M} B_{2k} t(k) \in \mathbb{C}[[x]]$

$\dfrac{t(k+1)}{t(k)}$ is a rational function in k and x

$$t(k+1) = r(k)t(k)$$
$$s(k+1) = s(k) + b(k)t(k)$$

$$\begin{pmatrix} t(M) \\ s(M) \end{pmatrix} = \underbrace{\begin{pmatrix} r(M-1) & 0 \\ b(M-1) & 1 \end{pmatrix} \cdots \begin{pmatrix} r(0) & 0 \\ b(0) & 1 \end{pmatrix}} \begin{pmatrix} t(0) \\ s(0) \end{pmatrix}$$

Matrix product is computed using **binary splitting** + fast polynomial multiplication

Some computational results

The Keiper-Li coefficients

Define $\{\lambda_n\}_{n=1}^{\infty}$ by

$$\log \xi\left(\frac{1}{1-x}\right) = \log \xi\left(\frac{x}{x-1}\right) = -\log 2 + \sum_{n=1}^{\infty} \lambda_n x^n$$

where $\xi(s) = \frac{1}{2}s(s-1)\pi^{-s/2}\Gamma(s/2)\zeta(s)$.

Keiper (1992):	Riemann hypothesis $\Rightarrow \forall n : \lambda_n > 0$
Li (1997):	Riemann hypothesis $\Leftarrow \forall n : \lambda_n > 0$

Keiper conjectured $2\lambda_n \approx (\log n - \log(2\pi) + \gamma - 1)$

Evaluating the Keiper-Li coefficients

Ingredients:

1. The series expansion of $\zeta(s)$ at $s = 0$
2. A series logarithm: $\log f(x) = \int f'(x)/f(x)dx$
3. Series expansion of $\log \Gamma(s)$ at $s = 1$, essentially $\gamma, \zeta(2), \zeta(3), \zeta(4), \ldots$
4. Right-composing by $x/(x-1)$

The need for high precision: a working precision of $\approx n$ bits is needed to get an accurate value for λ_n.

Complexity: $O(n^{3+\varepsilon})$ (implemented), $O(n^{2+\varepsilon})$ (in theory)

Fast composition

The *binomial transform* of $f = \sum_{k=0}^{\infty} a_k x^k$ is

$$T[f] = \frac{1}{1-x} f\left(\frac{x}{x-1}\right) = \sum_{n=0}^{\infty} \left(\sum_{k=0}^{n} (-1)^k \binom{n}{k} a_k \right) x^n$$

and the *Borel transform* is

$$B[f] = \sum_{k=0}^{\infty} \frac{a_k}{k!} x^k.$$

$T[f(x)] = B^{-1}[e^x B[f(-x)]]$, so we get $f\left(\frac{x}{x-1}\right)$ by a single power series multiplication!

Values of Keiper-Li coefficients

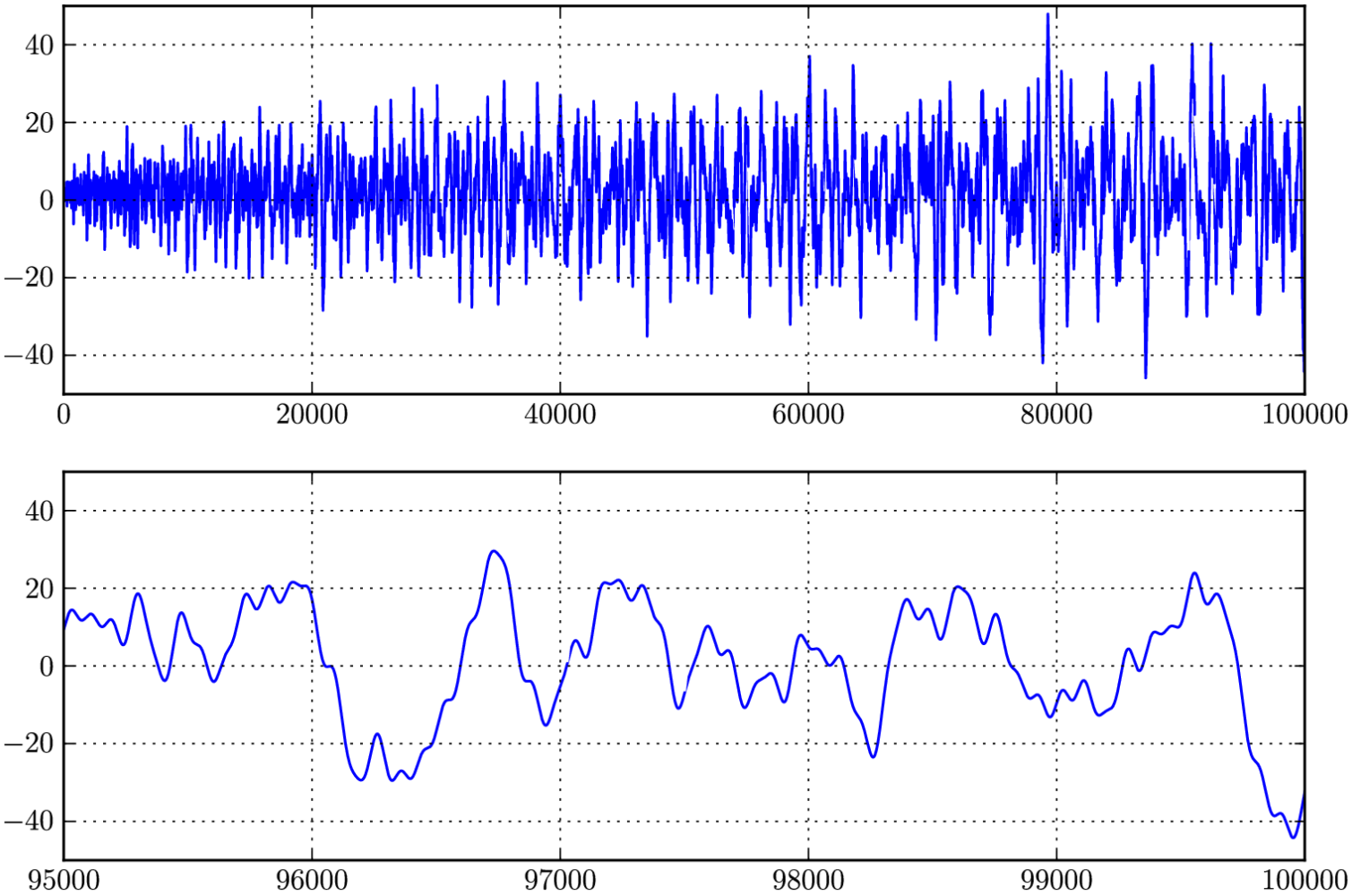
I have computed all λ_n up to $n = 100000$ using 110000 bits of precision. In particular,

$$\lambda_{100000} = 4.625807824069022314094160 38\ldots$$

plus about 2900 more accurate digits.

Keiper's approximation suggests $\lambda_{100000} \approx 4.626132$.

Comparison with approximation formula



Plot of $n\,(\lambda_n - (\log n - \log(2\pi) + \gamma - 1)/2)$.

Computation time (seconds) for Keiper-Li coefficients

	$n = 1000$	$n = 10000$	$n = 100000$
Error bound	0.017	1.0	97
Power sum, 16 threads	0.048	47	65402
(Power sum, CPU time)	(0.65)	(693)	(1042210)
Bernoulli numbers	0.0020	0.19	59
Tail (binary splitting)	0.058	11	1972
Logarithm of power series	0.047	8.5	1126
$\log \Gamma(1 + x)$ power series	0.019	3.0	1610
Power series composition	0.022	4.1	593
Total wall time	0.23	84	71051
Memory	8 MB	730 MB	48700 MB

Stieltjes constants

The **Stieltjes constants** are the coefficients in the Laurent series

$$\zeta(s,a) = \frac{1}{s-1} + \sum_{n=0}^{\infty} \frac{(-1)^n}{n!}\gamma_n(a)\,(s-1)^n.$$

Special case: $\gamma_n(1) \equiv \gamma_n$

$\gamma_0 \approx +0.577216$ $\qquad$ $\gamma_{10} \approx +0.000205$

$\gamma_1 \approx -0.072816$ $\qquad$ $\gamma_{100} \approx -4.25340 \times 10^{17}$

$\gamma_2 \approx -0.009690$ $\qquad$ $\gamma_{1000} \approx -1.57095 \times 10^{486}$

Asymptotics of Stieltjes constants

Open problem: **precise asymptotic bounds/series** for γ_n

Matsuoka (1985): $|\gamma_n| < 0.0001 e^{n \log\log n}$, $n \geq 10$

Knessl and Coffey (2011): asymptotic approximation formula (without explicit bound)

- Predicts sign oscillations
- Appears accurate even for small n
- Correct sign except for $n = 137$?

Knessl-Coffey approximation

$$\gamma_n \sim \frac{B}{\sqrt{n}} e^{nA} \cos(an + b)$$

$$A = \frac{1}{2}\log(u^2 + v^2) - \frac{u}{u^2 + v^2}, \quad B = \frac{2\sqrt{2\pi}\sqrt{u^2 + v^2}}{[(u+1)^2 + v^2]^{1/4}}$$

$$a = \tan^{-1}\left(\frac{v}{u}\right) + \frac{v}{u^2 + v^2}, \quad b = \tan^{-1}\left(\frac{v}{u}\right) - \frac{1}{2}\left(\frac{v}{u+1}\right)$$

where $u = v \tan v$, and v is the unique solution of
$2\pi \exp(v \tan v) = (n/v)\cos(v), \quad 0 < v < \pi/2.$

Similar formula for $\gamma_n(a)$, $a \neq 1$.

Numerical values

Computed value of γ_{100000} (>10860 digits):
$1.99192730631254109565\ldots \times 10^{83432}$

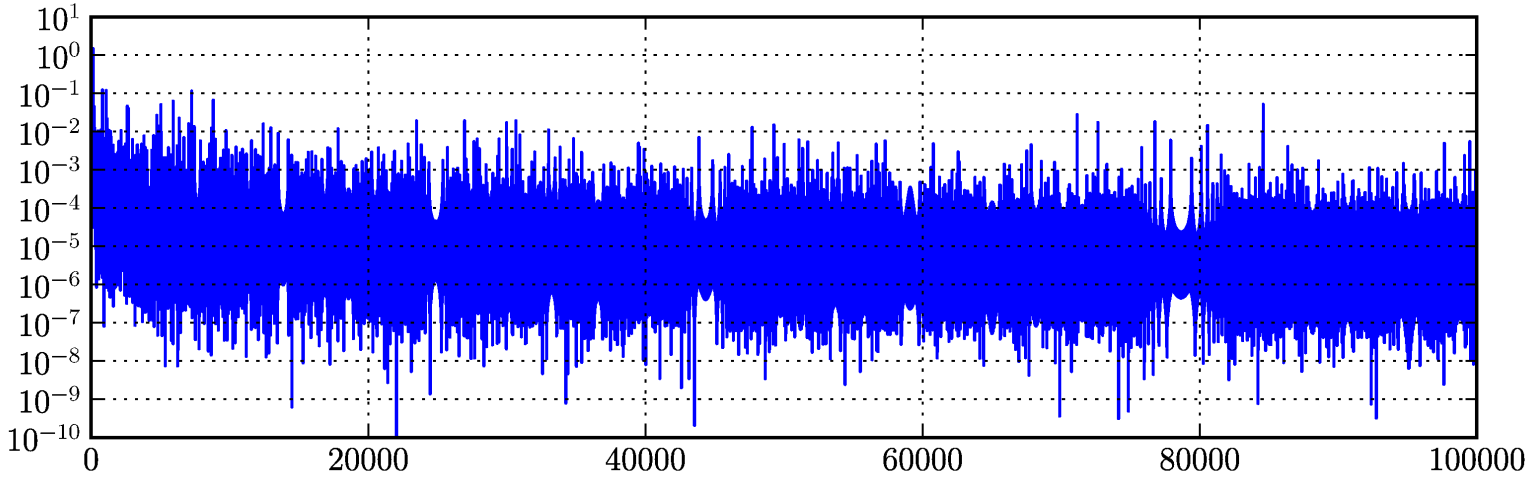
Knessl-Coffey approximation:
$1.9919333 \times 10^{83432}$

Matsuoka bound:
3.71×10^{106114}

Computed value of $\lambda_{50000}(1+i)$:
$(1.032502087431\ldots - 1.441962552840\ldots i) \times 10^{39732}$

Knessl-Coffey approximation:
$(1.0324943 - 1.4419586i) \times 10^{39732}$

Relative error of Knessl-Coffey formula



Nontrivial zeta zeros

I have computed the first nontrivial zero

$$0.5 + 14.13472514173\ldots i$$

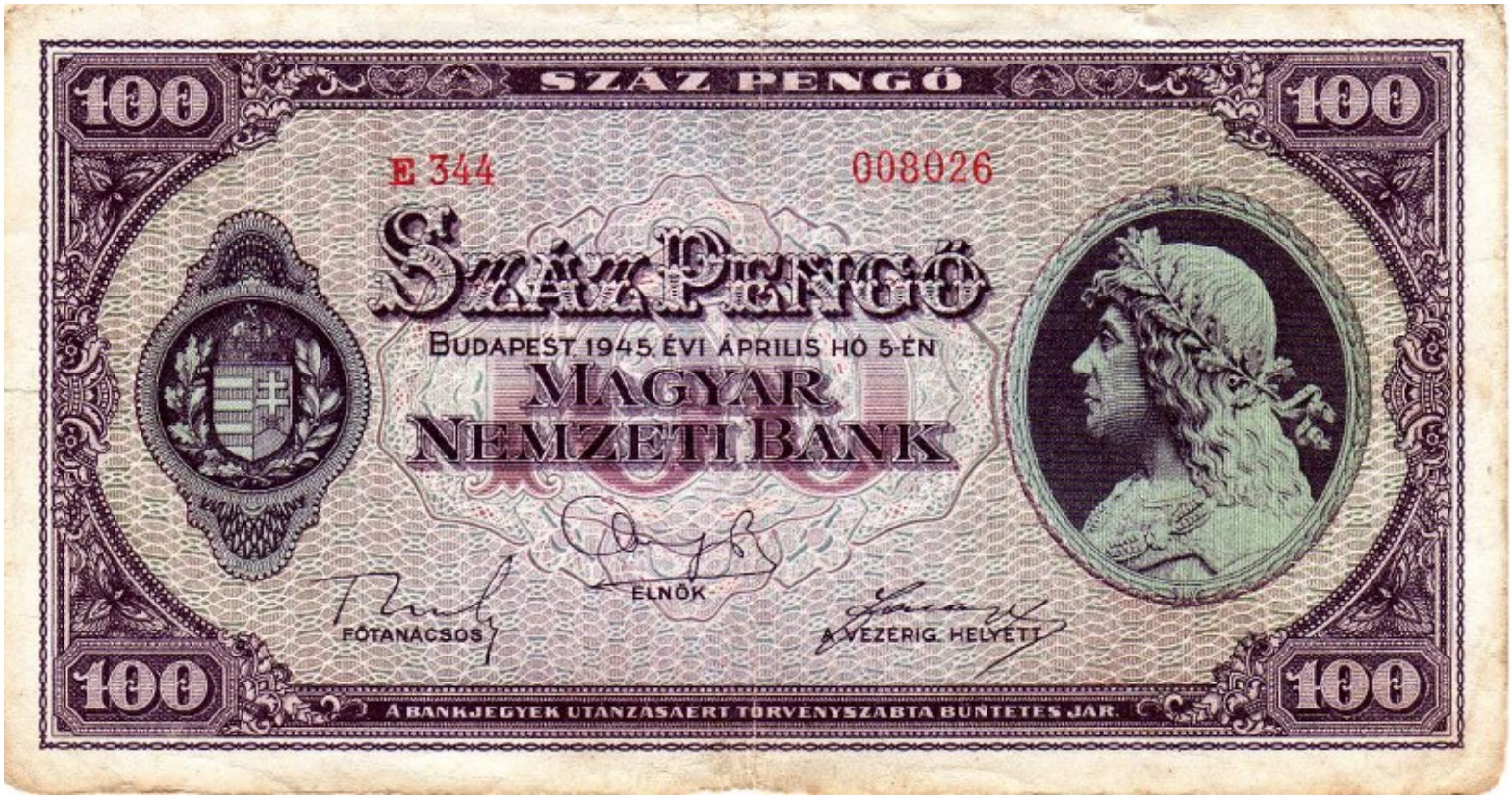
of $\zeta(s)$ to over 300,000 digits (current world record).

Yuri Matiyasevich and Gleb Beliakov have computed the first 40,000 zeros of $\zeta(s)$ to 40,000 digits using Arb. They are currently computing data for Dirichlet L-functions.

The partition function

Hungarian pengő (1 P, 1926)

10^2 pengő (April 1945)



100 pengő

10^3 pengő (July 1945)

1000 pengő

10^4 pengő (July 1945)

10,000 pengő

10^5 pengő (October 1945)

100,000 pengő

10^6 pengő (November 1945)

1,000,000 pengő

10^7 pengő (November 1945)

10,000,000 pengő

10^8 pengő (March 1946)

100,000,000 pengő

10^9 pengő (March 1946)

1,000,000,000 pengő

10^{10} pengő (April 1946)

10,000 milpengő = 10,000,000,000 pengő

10^{11} pengő (April 1946)

100,000 milpengő = 100,000,000,000 pengő

10^{12} pengő (May 1946)

1,000,000 milpengő = 1,000,000,000,000 pengő

10^{13} pengő (May 1946)

10,000,000 milpengő = 10,000,000,000,000 pengő

10^{14} pengő (June 1946)

100,000,000 milpengő = 100,000,000,000,000 pengő

10^{15} pengő (June 1946)

1,000,000,000 milpengő = 1,000,000,000,000,000 pengő

10^{16} pengő (June 1946)

10,000 b.pengő = 10,000,000,000,000,000 pengő

10^{17} pengő (June 1946)

100,000 b.pengő = 100,000,000,000,000,000 pengő

10^{18} pengő (June 1946)

1 million b.pengő = 1,000,000,000,000,000,000 (1 quintillion) pengő

10^{19} pengő (June 1946)

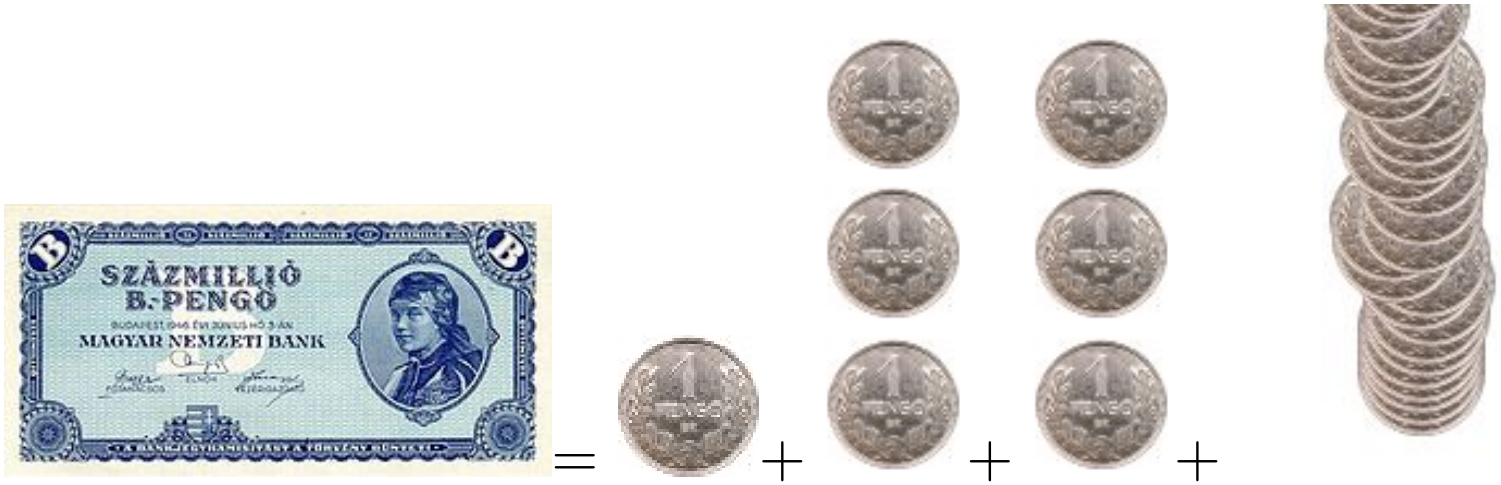
10 million b.pengő = 10,000,000,000,000,000,000 (10 quintillion) pengő

10^{20} pengő (June 1946)

100 million b.pengő = 100,000,000,000,000,000,000 (100 quintillion) pengő

August 1946

Making change (partitions)



$$10^{20} = 1 + 3 + 3 + 99999999999999999993$$

Number of ways to make change: $p(10^{20})$

$$\sum_{n=0}^{\infty} p(n)x^n = \prod_{k=1}^{\infty} \frac{1}{1-x^k}$$

Growth of $p(n)$

$$p(n) \sim \frac{1}{4n\sqrt{3}}\, e^{\pi\sqrt{2n/3}}, \qquad p(n) \text{ has } \sim n^{1/2} \text{ digits}$$

$p(10) = 42$
$p(100) = 190569292$
$p(1000) \approx 2.4 \times 10^{31}$

$p(10000) \approx 3.6 \times 10^{106}$
$p(100000) \approx 2.7 \times 10^{346}$
$p(1000000) \approx 1.5 \times 10^{1107}$

Theorem (FJ, 2014). There are exactly

$$1838176508344882 \ldots 231756788091448$$

(11,140,086,260 digits) different ways to make change for a 10^{20}-pengő banknote using stacks of 1-pengő coins.

Euler's method (1748) to compute $p(n)$

$$\sum_{n=0}^{\infty} p(n)x^n = \prod_{k=1}^{\infty} \frac{1}{1-x^k} = \left(\sum_{k=-\infty}^{\infty} (-1)^k x^{k(3k-1)/2} \right)^{-1}$$

$$p(n) = \sum_{k=1}^{n} (-1)^{k+1} \left(p(n - \tfrac{k(3k-1)}{2}) + p(n - \tfrac{k(3k+1)}{2}) \right)$$

Using recurrence: $O(n^2)$ time

Using fast power series arithmetic: $O(n^{1.5+\varepsilon})$ time (quasi-optimal for computing **all values simultaneously**)

The Hardy-Ramanujan-Rademacher formula

$$p(n) = \sum_{k=1}^{\infty} \frac{\sqrt{k}\, A_k(n)}{\pi\sqrt{2}} \frac{d}{dn} \left(\frac{\sinh\left[\frac{\pi}{k}\sqrt{\frac{2}{3}\left(n - \frac{1}{24}\right)}\right]}{\sqrt{n - \frac{1}{24}}} \right)$$

$$A_k(n) = \sum_{\substack{0 \le h < k \\ \gcd(h,k)=1}} e^{\pi i \left[s(h,k) - \frac{1}{k} 2nh \right]}$$

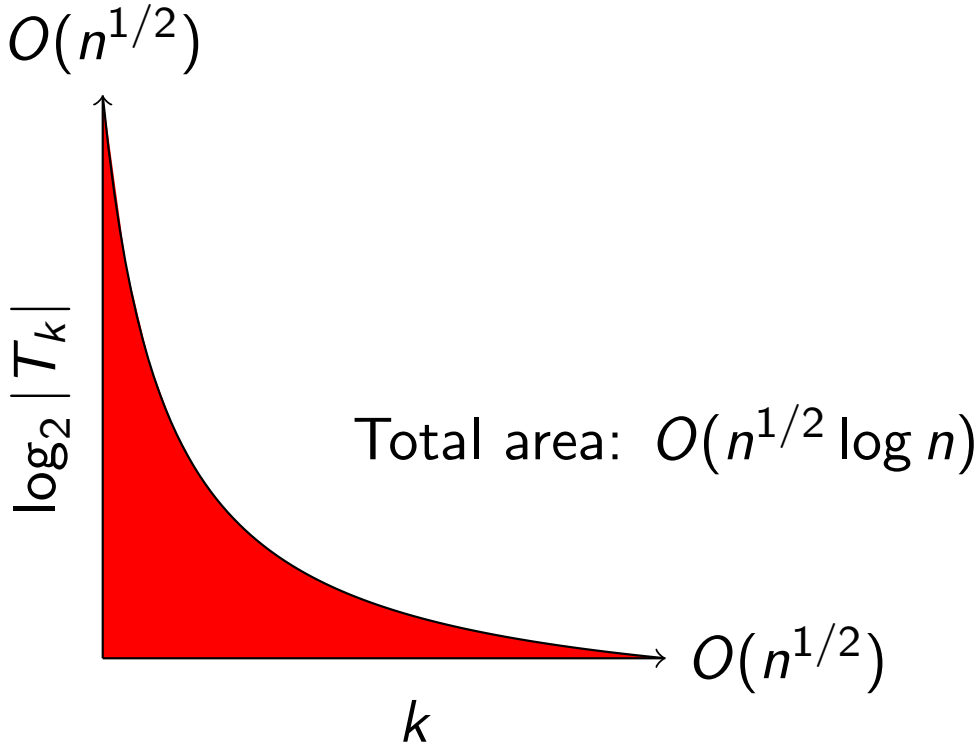
$$s(h,k) = \sum_{i=1}^{k-1} \frac{i}{k} \left(\frac{hi}{k} - \left\lfloor \frac{hi}{k} \right\rfloor - \frac{1}{2} \right)$$

Hardy and Ramanujan (1917) and Rademacher (1936). Explicit error bound by Rademacher.

Quasi-optimal isolated computation of $p(n)$

Theorem (FJ, 2011): $p(n)$ can be computed in time $n^{1/2} \log^{4+o(1)} n$.

The idea: $p(n) \approx \sum_{k=0}^{n^{1/2}} T_k, \quad \log_2 |T_k| = O(n^{1/2}/k)$



The tricky part: need to approximate T_k in quasi-optimal time.

Evaluating exponential sums

$$A_k(n) = \sum_{\substack{0 \le h < k \\ \gcd(h,k)=1}} e^{\pi i \left[s(h,k) - \frac{1}{k} 2nh \right]}$$

$$s(h,k) = \sum_{i=1}^{k-1} \frac{i}{k} \left(\frac{hi}{k} - \left\lfloor \frac{hi}{k} \right\rfloor - \frac{1}{2} \right)$$

Naively:

- $O(k^2)$ arithmetic operations for $A_k(n)$
- $O(n^{3/2})$ arithmetic operations for $p(n)$

Fast computation of Dedekind sums

Let $0 < h < k$ and let $k = r_0, r_1, \ldots, r_{m+1} = 1$ be the sequence of remainders in the Euclidean algorithm for $\gcd(h, k)$. Then

$$s(h,k) = \frac{(-1)^{m+1} - 1}{8} + \frac{1}{12} \sum_{j=1}^{m+1} (-1)^{j+1} \frac{r_j^2 + r_{j-1}^2 + 1}{r_j r_{j-1}}.$$

- $O(\log k)$ integer ops to evaluate $s(h, k)$
- $O(k \log k)$ integer ops to evaluate $A_k(n)$
- $O(n \log n)$ integer ops to evaluate $p(n)$

Still not good enough!

$A_k(n)$ using prime factorization of k

Whiteman (1956):

$$A_{p_1^{e_1}\cdots p_i^{e_i}}(n) = \sqrt{\frac{s}{t}} \cos\left(\frac{\pi r_1}{24k_1}\right) \cdots \cos\left(\frac{\pi r_i}{24k_i}\right)$$

Requires solving quadratic equations modulo divisors of k (careful bit complexity analysis).

Only $O(\log k)$ cosines, so the numerical evaluation becomes fast enough!

My implementation of $p(n)$

2011 version:

- **500 times faster** than the runners-up (Mathematica, Sage)
- $p(10^k)$ computed up to $k = 19$
- Tabulated **22 billion new Ramanujan-type congruences** using Weaver's algorithm. Example:

$$p(289952442924860052459470 69k + 289952213369764311353210 47) \equiv 0 \bmod 29$$

2014 version:

- **Rigorous numerical evaluation** (ball arithmetic)
- 3 times faster, 3 times more memory-efficient
- $p(10^k)$ computed up to $k = 20$

Getting it right

Search Hints

(Greetings from The On-Line Encyclopedia of Integer Sequences!)

A110375 Numbers n such that Maple 9.5, Maple 10, Maple 11 and Maple 12 give the wrong answers for the number of partitions of n.

11269, 11566, 12376, 12430, 12700, 12754, 15013, 17589, 17797, 18181, 18421, 18453, 18549, 18597, 18885, 18949, 18997, 20865, 21531, 21721, 21963, 22683, 23421, 23457, 23547, 23691, 23729, 23853, 24015, 24087, 24231, 24339, 24519, 24591, 24627, 24681, 24825, 24933, 25005, 25023, 25059, 25185, 25293, 27020 (list; graph; refs; listen; history; text; internal format)

OFFSET 1,1

COMMENTS Based on various postings on the Web, sent to N. J. A. Sloane by R. J. Mathar. Thanks to several correspondents who sent information about other versions of Maple.
Mathematica 6.0, DrScheme and pari-2.3.3 all give the correct answers.
Ramanujan's congruence says that numbpart(5*k+4)==0 mod 5, so numbpart(11269)=...851==1 mod 5 can't be correct. [Robert Gerbicz, May 13 2008]

LINKS Table of n, a(n) for n=1..44.
Author?, Concerning this sequence

EXAMPLE From PARI, the correct answer:
numbpart(11269)
231139177231303975514411787649455628959060199360109972557851519105155176 1\
80318215891795874905318274163248033071850
From Maple 11, incorrect:
combinat[numbpart](11269);
231139177231303975514411787649455628959060199360109972557851519105155176 1\
80318215891795874905318274163248033071851
On the other hand, the old Maple 6 gives the correct answer.

CROSSREFS Cf. A000041.
Sequence in context: A217125 A195654 A195649 * A177216 A112441 A104017
Adjacent sequences: A110372 A110373 A110374 * A110376 A110377 A110378

KEYWORD nonn

AUTHOR N. J. A. Sloane, May 13 2008

EXTENSIONS More terms from R. J. Mathar, May 14 2008, based on a comparison of results from Maple 9 and PARI-2.3.3.

STATUS approved

Time breakdown for $p(n)$ (hours)

n	Memory	Pi	Exp	T1	T2	Wall	CPU
10^{17}	4.5 GB	0.2	1.2	1.7	2.1	2.1	3.8
10^{18}	13 GB	0.8	4.5	6.5	5.8	6.5	12
10^{19}	38 GB	3	17	24	23	24	47
10^{20}	128 GB	12	66	94	111	111	205