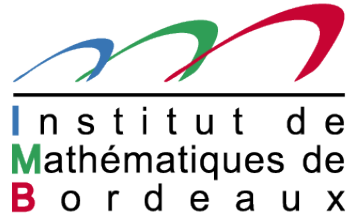
Numerical and symbolic tools for reliable computing with real numbers

Fredrik Johansson

Inria Bordeaux

CAPA seminar
Uppsala University
December 16, 2019



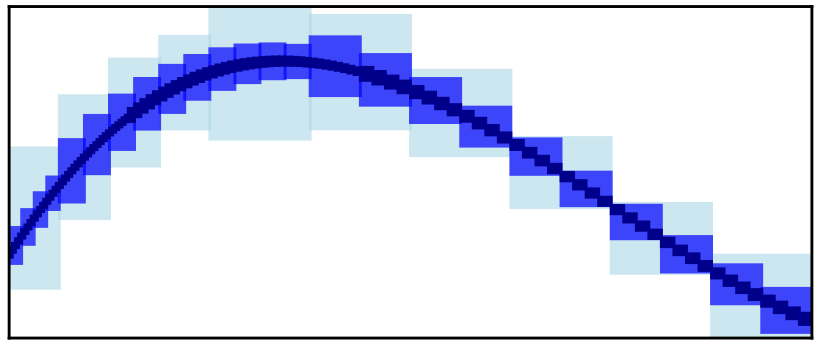
Computing with real numbers

- Symbolic representations: $x = \sqrt{2} \cdot \pi$
- Approximations: $x \approx m \cdot 2^e$ or $x \approx \frac{p}{q}$
- Intervals: $x \in [a, b]$ or balls $x \in [m \pm r] = [m - r, m + r]$

“Interval arithmetic” vs “ball arithmetic”

Intervals $[a, b]$: *better for subdivision of space*

$$[2.0, 3.0] \to [2.0, 2.5] \cup [2.5, 3.0]$$



Balls $[m \pm r]$: *better for approximation of numbers*

$$\pi \in [3.14159265358979323846264338328 \pm 1.07 \cdot 10^{-30}]$$

[J. van der Hoeven, *Ball arithmetic*, 2009]

The Arb library

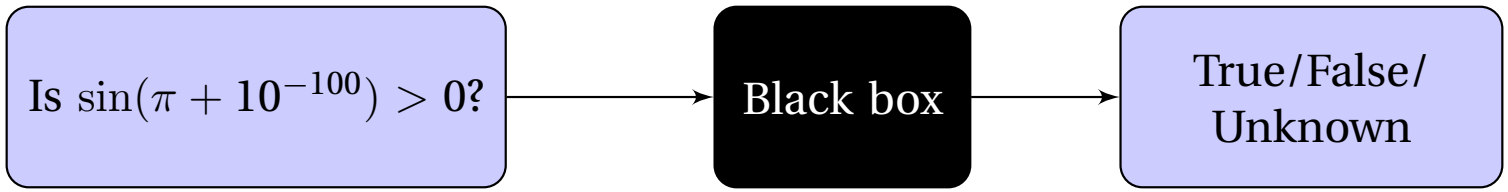
`http://arblib.org/`

- C library, open source (LGPL)
- Arbitrary-precision ball arithmetic, `arb_t` =

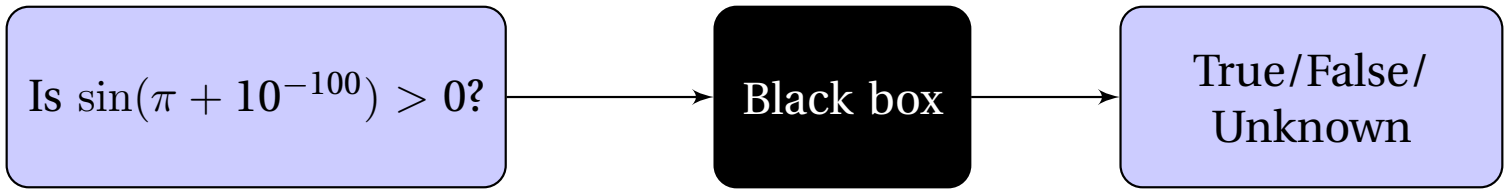
$$[m \cdot 2^e \pm r \cdot 2^f] \qquad \begin{matrix} m, e, f \\ r \end{matrix} \quad \begin{matrix} \text{arbitrary-size integers} \\ \text{30-bit unsigned integer} \end{matrix}$$

- Complex numbers, matrices, polynomials, power series, special functions, some calculus (integration, root-finding)
- Asymptotically fast algorithms
- Developed since 2012, $O(10)$ contributors
- 180 000 lines of code, 3000 functions
- External wrappers: Python, SageMath, Julia

Example: a decision problem



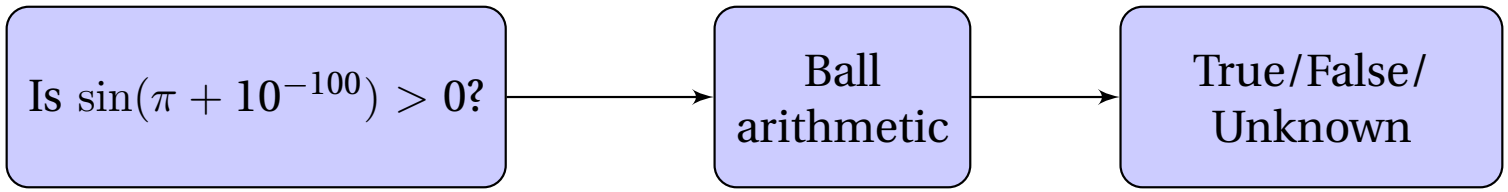
Example: a decision problem



IEEE 754 floating-point?

```
>>> sin(pi + 1e-100)
1.2246467991473532e-16
```

Example: a decision problem

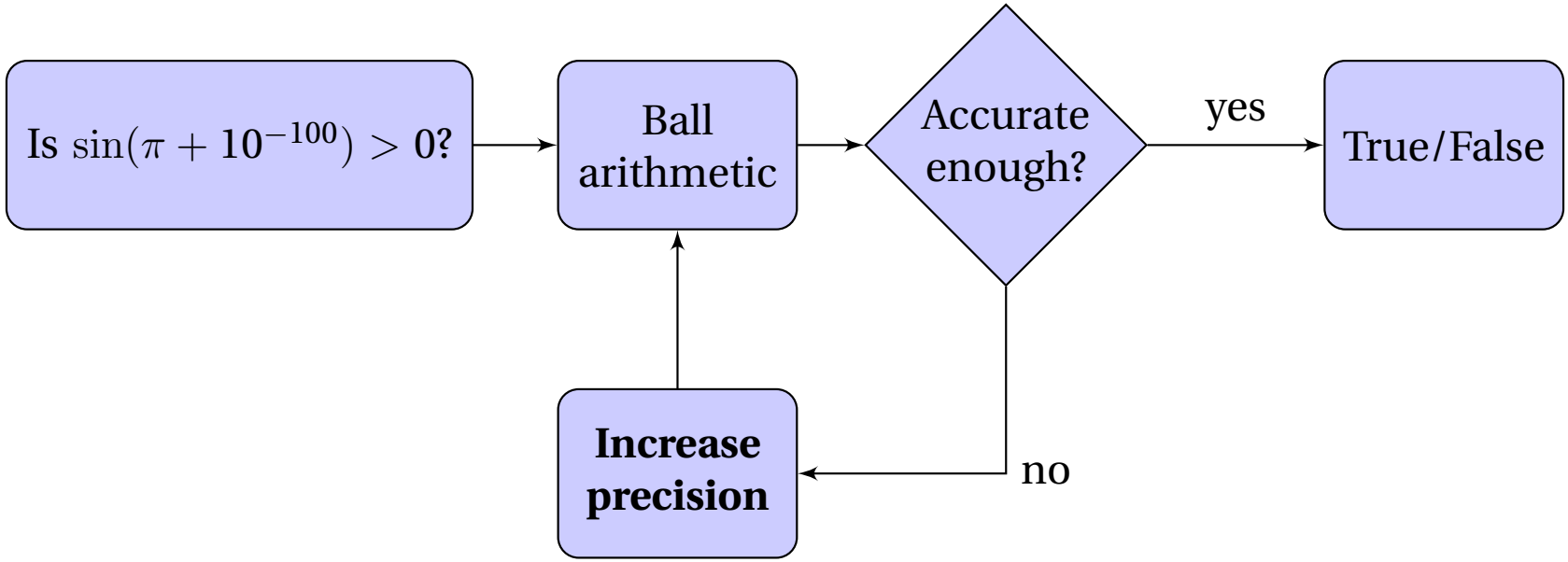


Arb,[1] 53-bit precision:

```
>>> (arb.pi() + arb(10)**-100).sin()
[+/- 7.90e-16]
```

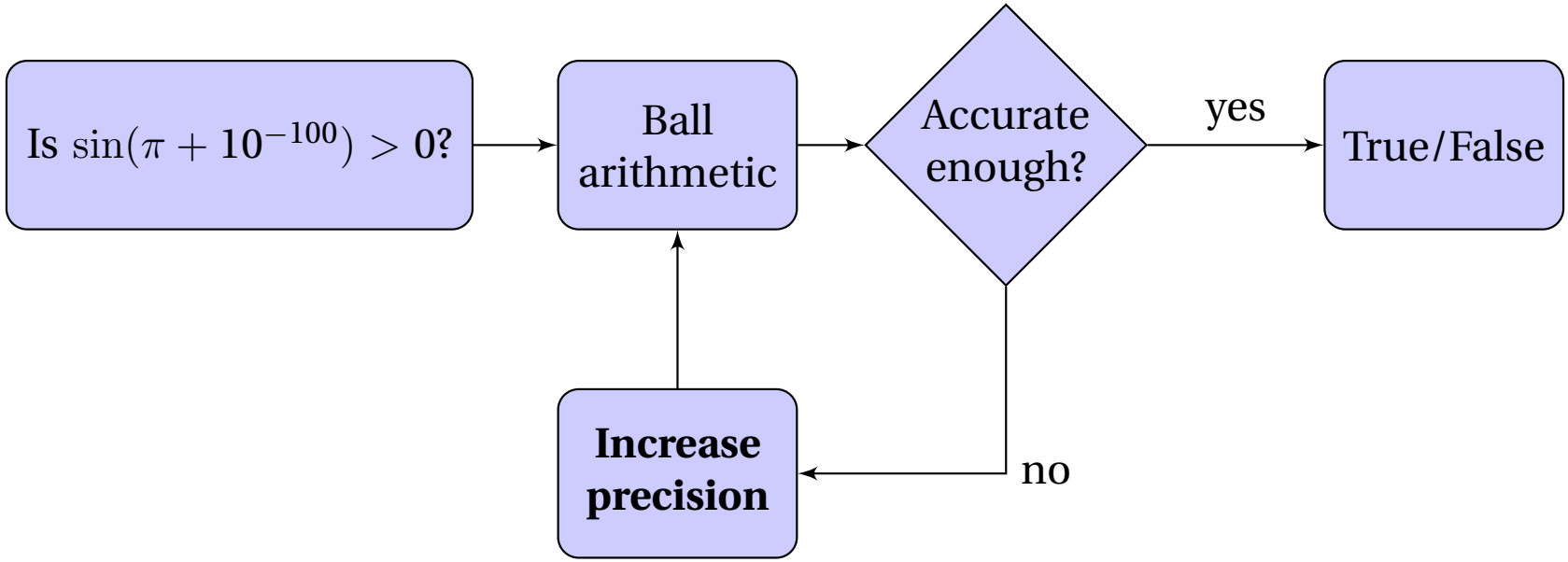
[1] Actually, the Python interface (`python-flint`)

Example: a decision problem



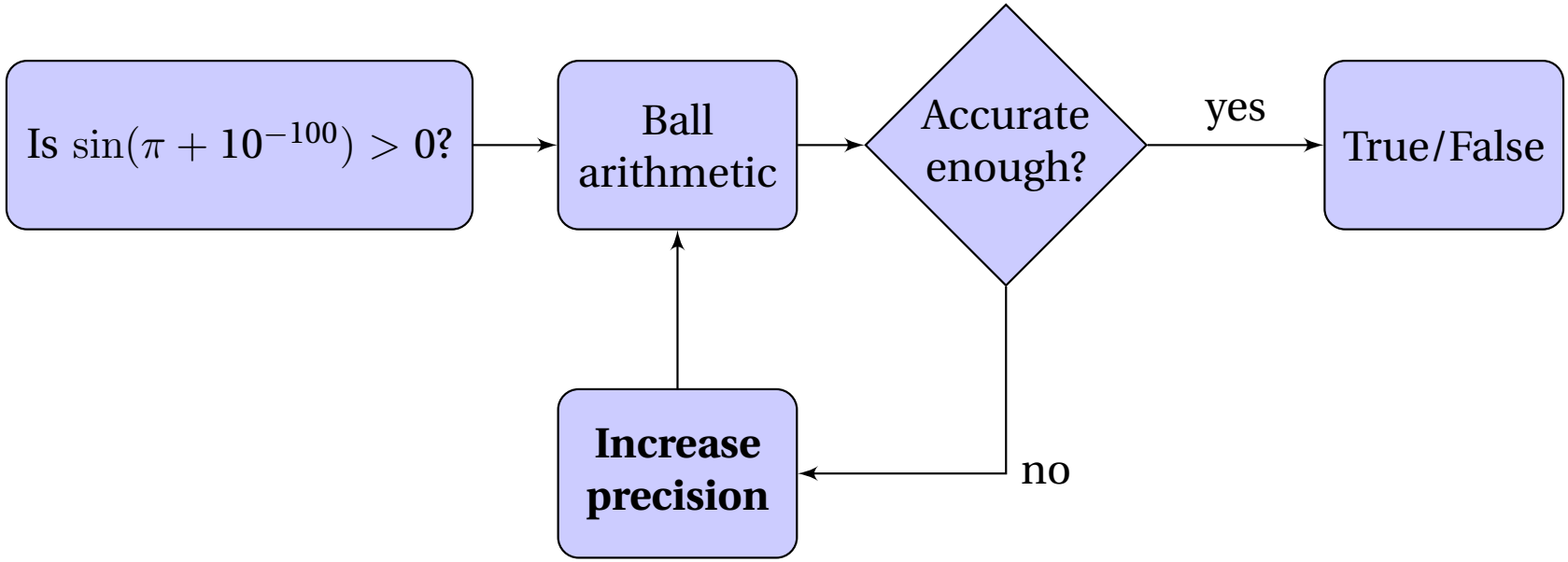

```
>>> ctx.dps = 15; (arb.pi() + arb(10)**-100).sin()
[+/- 7.90e-16]
```

Example: a decision problem



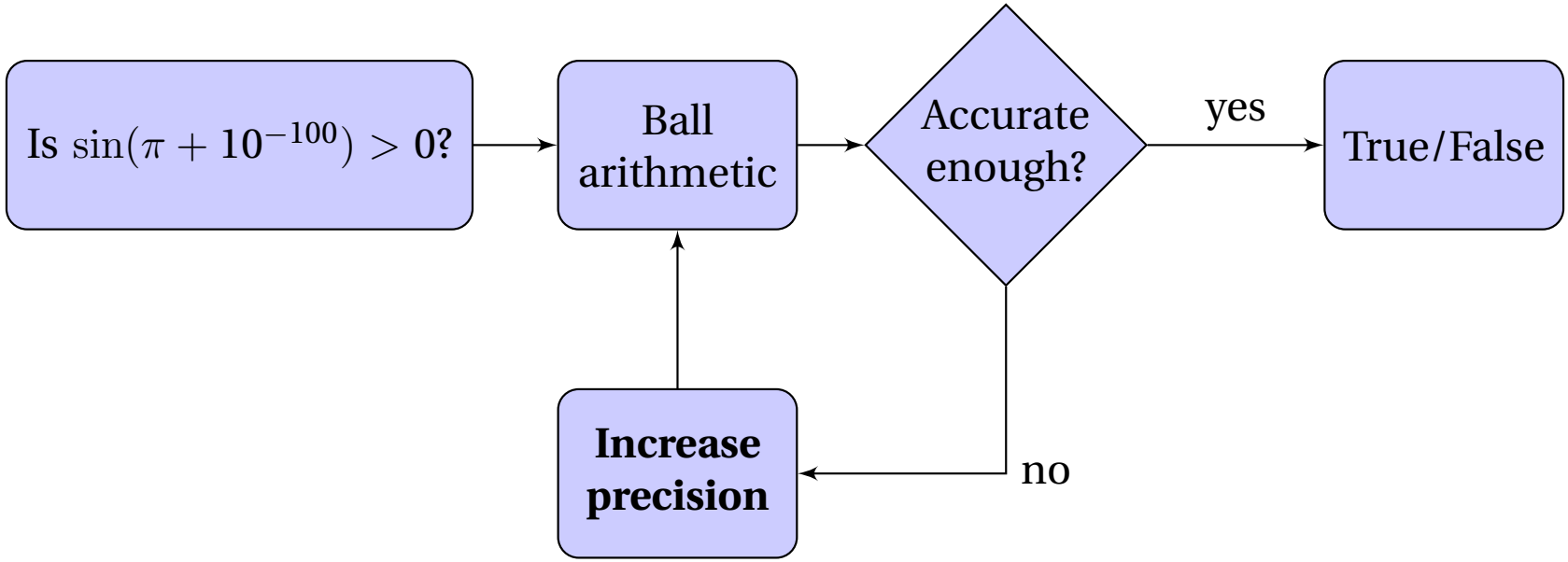

```
>>> ctx.dps = 15; (arb.pi() + arb(10)**-100).sin()
[+/- 7.90e-16]
>>> ctx.dps = 30; (arb.pi() + arb(10)**-100).sin()
[+/- 7.62e-31]
```

Example: a decision problem



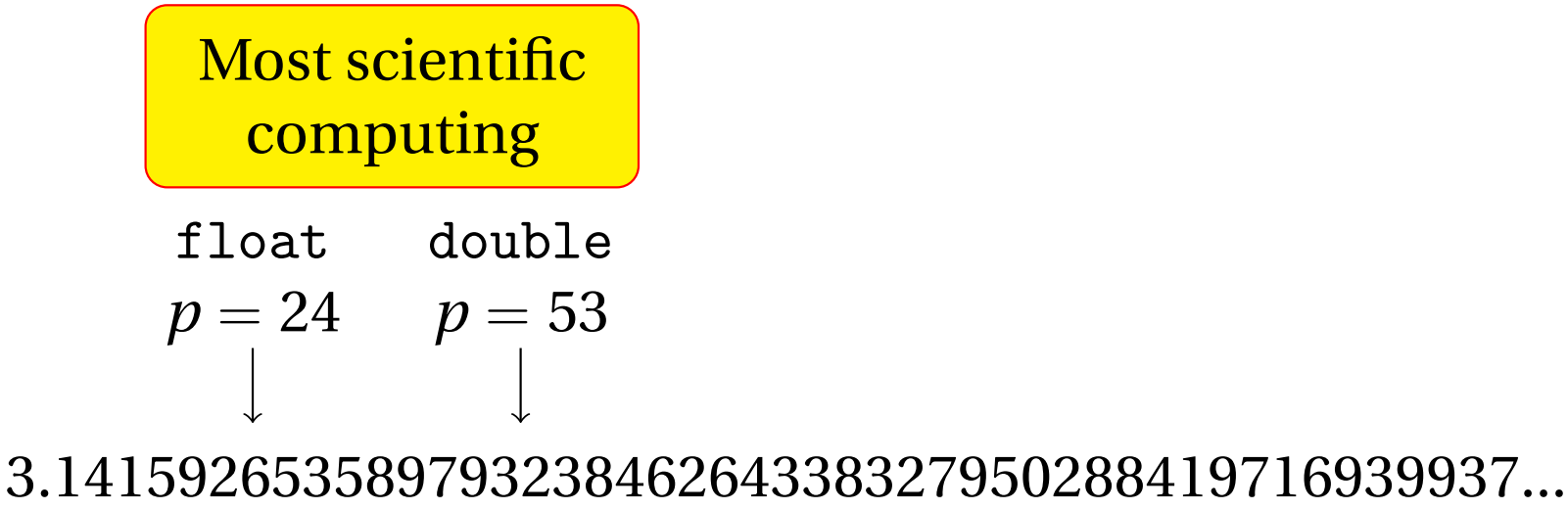

```
>>> ctx.dps = 15; (arb.pi() + arb(10)**-100).sin()
[+/- 7.90e-16]
>>> ctx.dps = 30; (arb.pi() + arb(10)**-100).sin()
[+/- 7.62e-31]
>>> ctx.dps = 60; (arb.pi() + arb(10)**-100).sin()
[+/- 5.82e-61]
```

Example: a decision problem

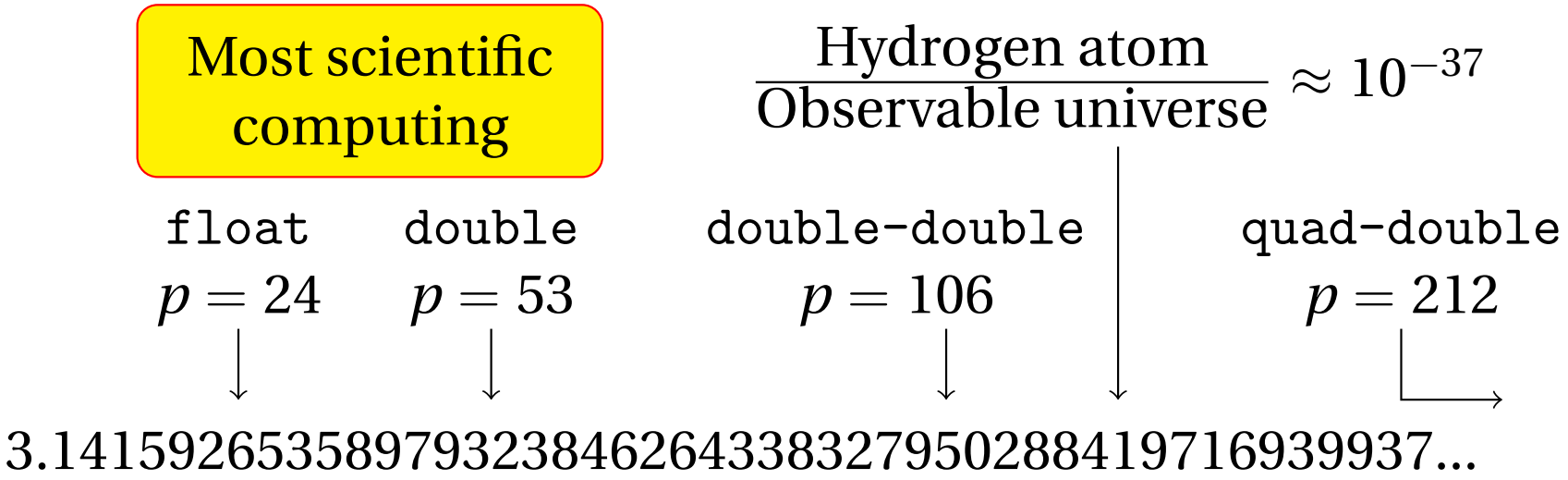



```
>>> ctx.dps = 15; (arb.pi() + arb(10)**-100).sin()
[+/- 7.90e-16]
>>> ctx.dps = 30; (arb.pi() + arb(10)**-100).sin()
[+/- 7.62e-31]
>>> ctx.dps = 60; (arb.pi() + arb(10)**-100).sin()
[+/- 5.82e-61]
>>> ctx.dps = 120; (arb.pi() + arb(10)**-100).sin()
[-1.0000000000000000000e-100 +/- 8.74e-121]
```

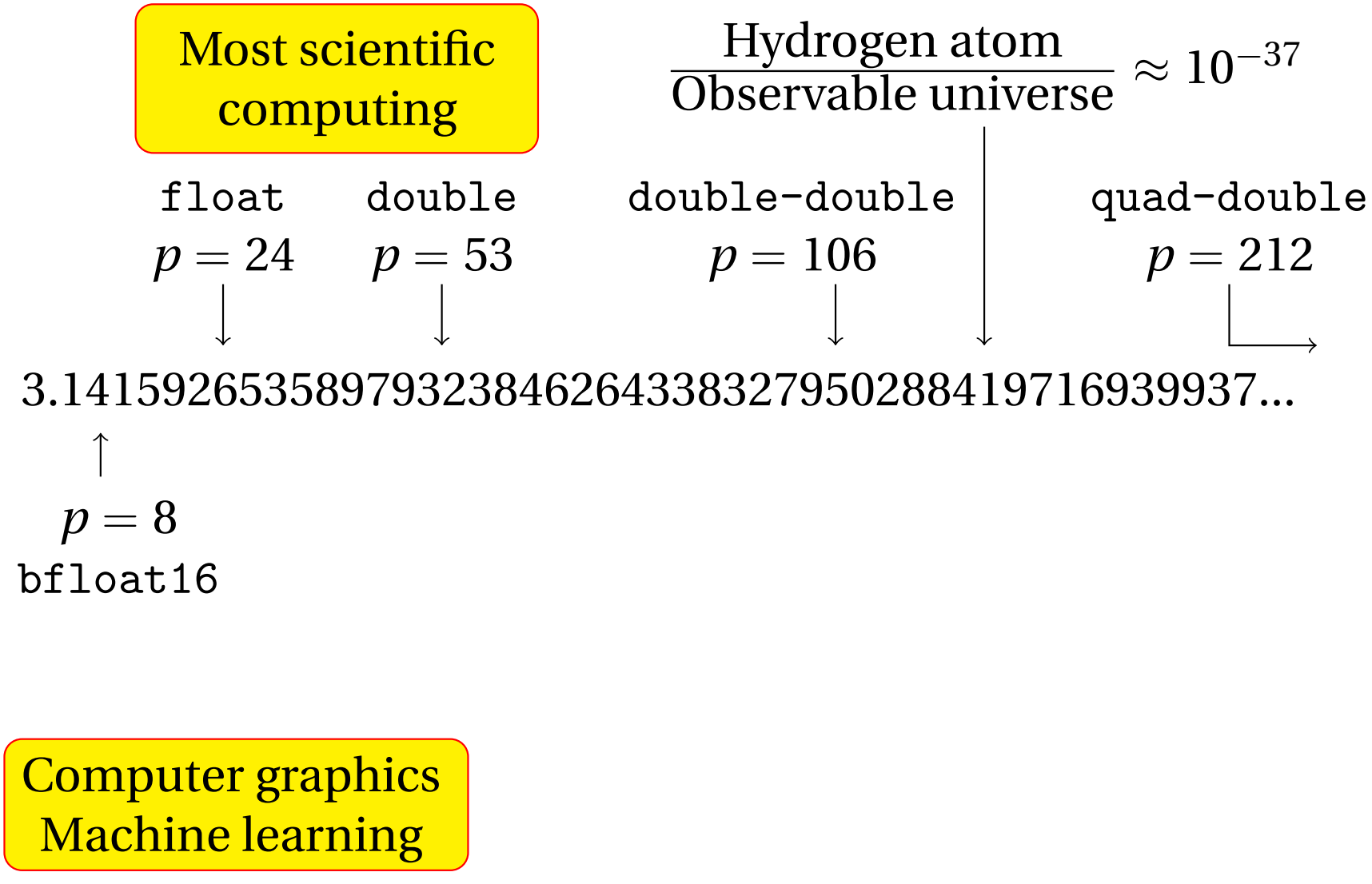
Precision in practice



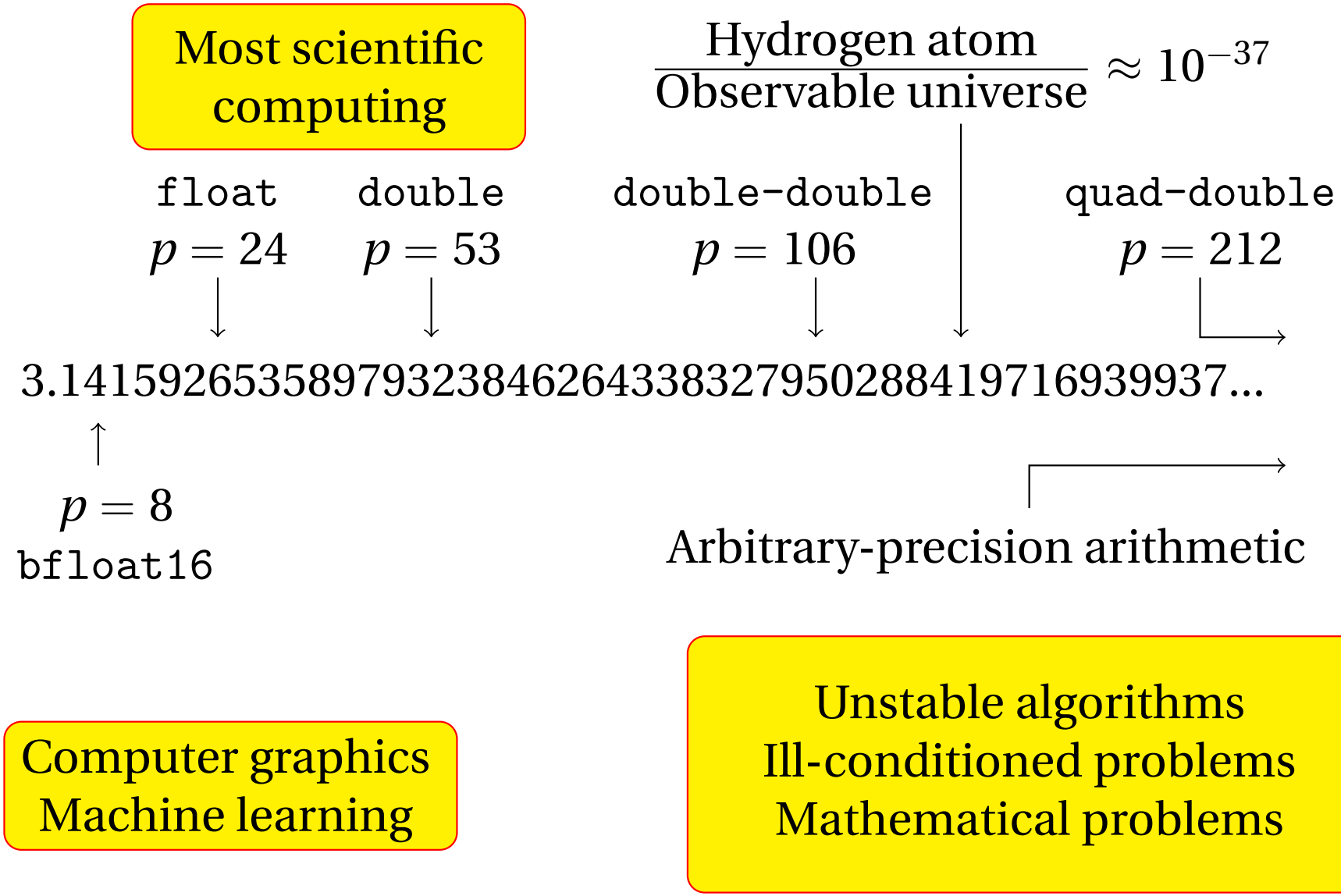
Precision in practice



Precision in practice

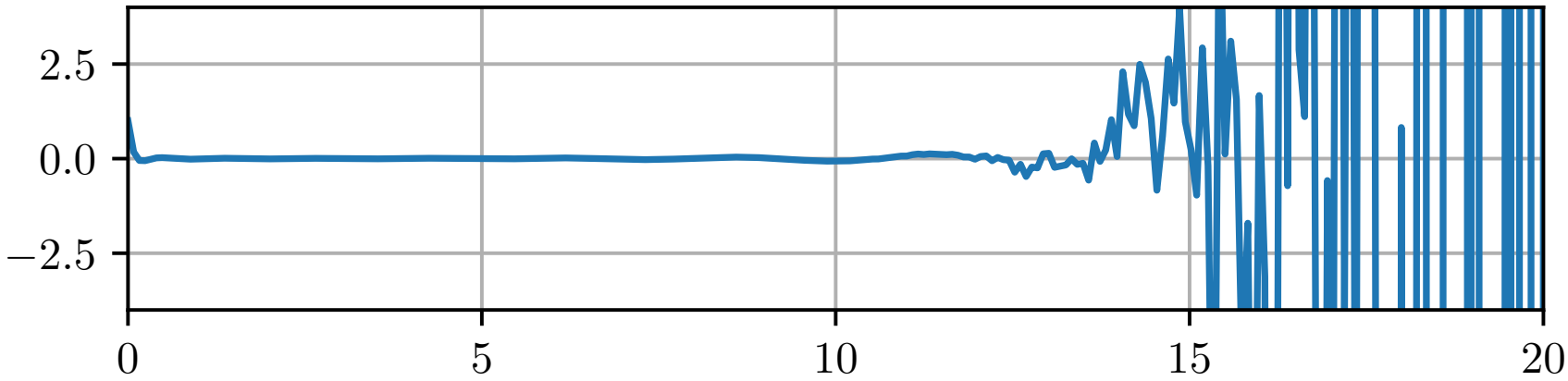


Precision in practice

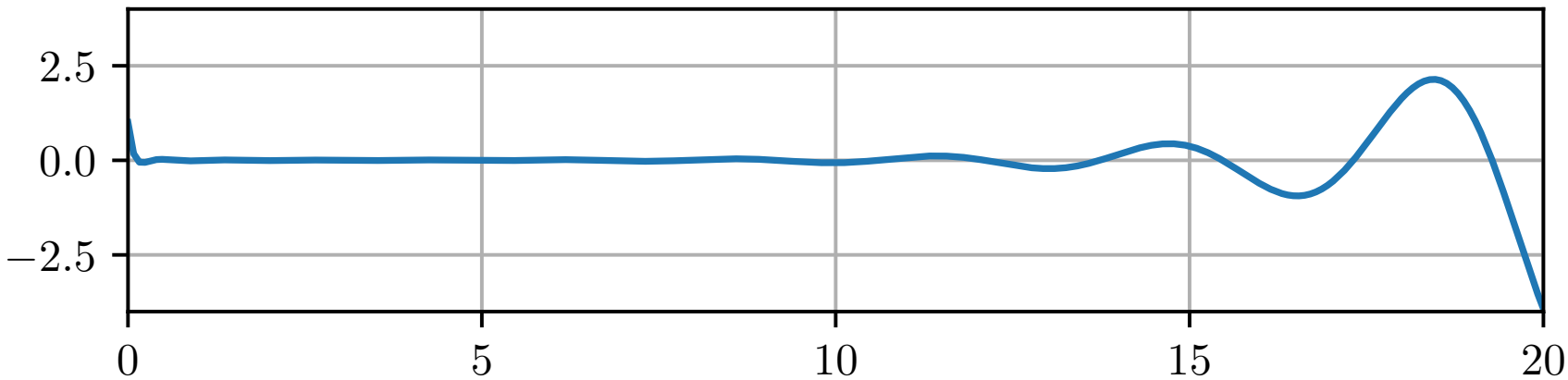


Example: special functions

`scipy.special.hyp1f1(-50,3,x)`



Actual value of ${}_1F_1(-50, 3, x)$ (computed by Arb)



Example: special functions

Rendering of the modular j-invariant $j(\tau)$ on $\tau \in [0,2] + [0,1]\,i$

Example: special functions

Rendering of $j(\tau)$ on $\tau \in \sqrt{\pi} + [0, 2 \cdot 10^{-100}] + [0, 10^{-100}]\,i$

Example: the partition function $p(n)$

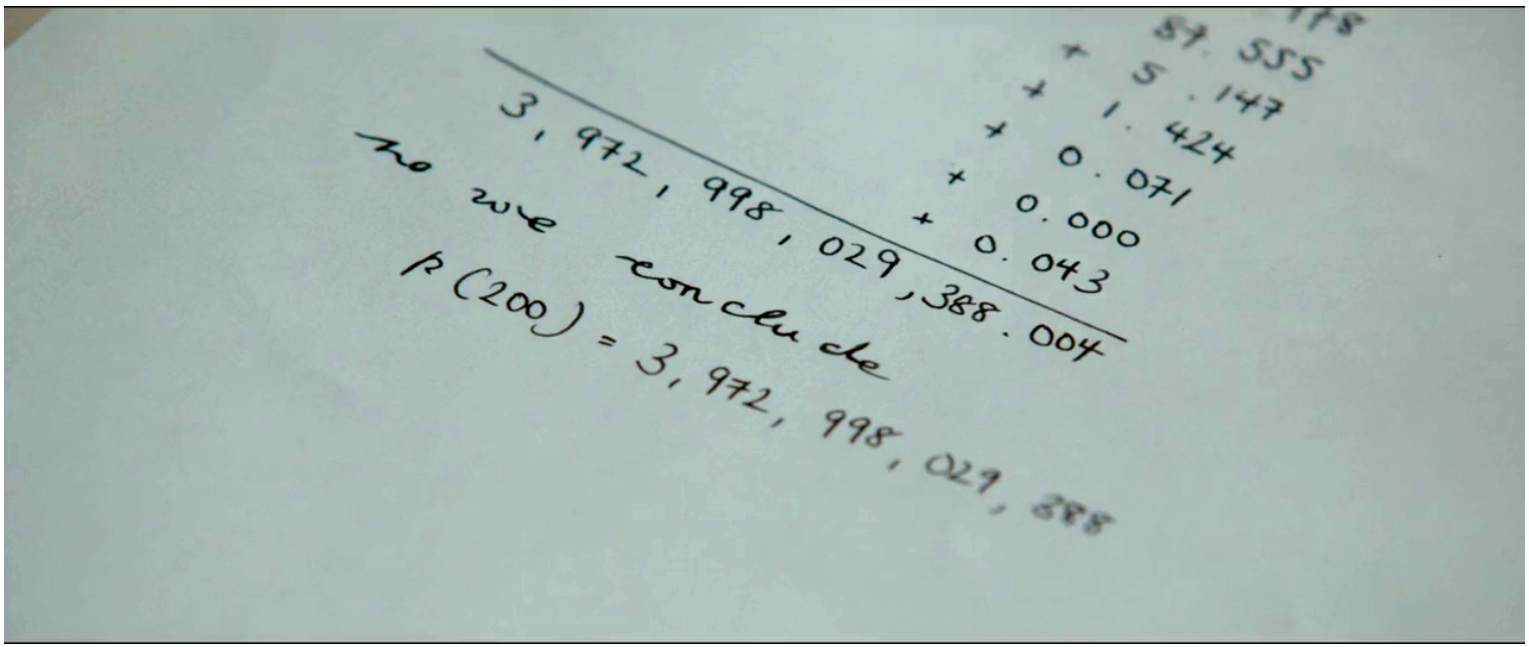
$$p(4) = 5 \quad \text{since} \quad (4) = (3+1) = (2+2) = (2+1+1) = (1+1+1+1)$$

Example: the partition function $p(n)$

$p(4) = 5$ since $(4) = (3+1) = (2+2) = (2+1+1) = (1+1+1+1)$

The Hardy-Ramanujan-Rademacher formula

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



Scene from *The Man Who Knew Infinity*, 2015

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. [5]

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
803182158917958749053182741632480330718 50
From Maple 11, incorrect:
combinat[numbpart](11269);
231139177231303975514411787649455628959060199360109972557851519105155176\1
803182158917958749053182741632480330718 51
On the other hand, the old Maple 6 gives the correct answer.

Implementation of $p(n)$ in Arb

Computing $p(n) = \sum_{k=1}^{\infty} T(k)$ correctly

- Truncation error: $\left|\sum_{k=N+1}^{\infty} T(k)\right| \leq \varepsilon_N$
- Arithmetic error in $\sum_{k=1}^{N} T(k)$
- Unique integer: $[627.000 \pm 0.001] \ \rightarrow \ 627$

Implementation of $p(n)$ in Arb

Computing $p(n) = \sum_{k=1}^{\infty} T(k)$ correctly

- Truncation error: $\left|\sum_{k=N+1}^{\infty} T(k)\right| \leq \varepsilon_N$
- Arithmetic error in $\sum_{k=1}^{N} T(k)$
- Unique integer: $[627.000 \pm 0.001] \ \rightarrow \ 627$

Performance

- Optimal complexity: $\tilde{O}(n^{1/2})$
- $p(10^{10})$: 10^5 digits, 0.2 seconds
- $p(10^{20})$: 10^{10} digits, 200 hours

Linear algebra

Three paradigms:

- Floating-point computation (approx / no error bounds)
- Direct computation in ball arithmetic
- Certification: $3.14159 \rightarrow [3.14159 \pm 3 \cdot 10^{-6}]$

Linear algebra

Three paradigms:

- Floating-point computation (approx / no error bounds)
- Direct computation in ball arithmetic
- Certification: $3.14159 \rightarrow [3.14159 \pm 3 \cdot 10^{-6}]$

Example: solving $Ax = b$

- `arb_mat_approx_solve` (float Gaussian elimination)
- `arb_mat_solve_lu` (ball Gaussian elimination)
- `arb_mat_solve_precond` (Hansen-Smith: compute $R \approx A^{-1}$, then solve $(RA)x = Rb$ in ball arithmetic)
- `arb_mat_solve` (automatic choice)

Well-conditioned matrix, floating-point arithmetic

Solve $Ax = b$ for n by n DCT matrix A, ones in b

```
n       x_1
-----------------------------------------
1       1.00000000000000
2       1.41421356237310
3       1.69270534084004
4       1.92387953251129
5       2.12756936615424
10      2.93381472087850
20      4.08975996439745
30      4.98358363678481
40      5.73967906112813
50      6.40709547252447
100     9.03226736256870
1000    28.4797579795013
```

Float Gaussian elimination, $prec = 53$

Well-conditioned matrix, ball arithmetic

Solve $Ax = b$ for n by n DCT matrix A, ones in b

```
n       x_1
-----------------------------------------
1       1.00000000000000
2       [1.41421356237310 +/- 5.53e-15]
3       [1.69270534084004 +/- 6.07e-15]
4       [1.9238795325113 +/- 2.18e-14]
5       [2.1275693661542 +/- 5.01e-14]
10      [2.93381472088 +/- 9.20e-12]
20      [4.089760 +/- 5.09e-7]
30      [5.0 +/- 0.0468]
40      [+/- 6.36e+3]
50      -
100     -
1000    -
```

Ball Gaussian elimination, $prec = 53$

Well-conditioned matrix, better ball arithmetic

Solve $Ax = b$ for n by n DCT matrix A, ones in b

```
n       x_1
-----------------------------------------
1       1.00000000000000
2       [1.41421356237310 +/- 5.53e-15]
3       [1.69270534084004 +/- 6.07e-15]
4       [1.9238795325113 +/- 2.18e-14]
5       [2.1275693661542 +/- 5.01e-14]
10      [2.93381472087850 +/- 4.74e-15]   HS
20      [4.08975996439745 +/- 8.09e-15]   HS
30      [4.9835836367848 +/- 1.69e-14]    HS
40      [5.7396790611281 +/- 3.38e-14]    HS
50      [6.4070954725245 +/- 4.13e-14]    HS
100     [9.0322673625687 +/- 2.09e-14]    HS
1000    [28.479757979501 +/- 3.23e-13]    HS
```

Ball arithmetic, $prec = 53$, `HS` = Hansen-Smith is chosen

Ill-conditioned matrix, floating-point arithmetic

Solve $Ax = b$ for n by n Hilbert matrix A, ones in b

```
n       x_1
-----------------------------------------
1       1.00000000000000
2       -2.00000000000000
3       3.00000000000005
4       -4.00000000000134
5       5.00000000002543
10      -9.99995126767237
20      -35.6777214905094
30      6.34155467845874
40      44.5343844893017
50      33.9009376509307
100     52.5690415087492
1000    64.6972177711318
```

Float Gaussian elimination, $prec = 53$

Ill-conditioned matrix, ball arithmetic

Solve $Ax = b$ for n by n Hilbert matrix A, ones in b

```
n       x_1                                      prec
-----------------------------------------------------
1       1.00000000000000                         64
2       [-2.00000... +/- 1.85e-18]               64
3       [3.00000... +/- 3.14e-35]                128
4       [-4.00000... +/- 1.64e-32]               128
5       [5.00000... +/- 3.56e-29]                128
10      [-10.00000... +/- 7.41e-50]              256
20      [-20.00000... +/- 8.85e-85]              512
30      [-30.00000... +/- 2.82e-33]              256
40      [-40.00000... +/- 1.81e-142]             1024
50      [-50.00000... +/- 1.61e-90]              1024    # 0.1 s
100     [-100.00000... +/- 1.74e-118]            2048    # 0.8 s
1000    [-1000.00000... +/- 4.62e-937]           8192    # 4 hours
```

Try $prec = 64, 128, 256, \ldots$ until relative error $< 2^{-53}$

Performance: matrix arithmetic

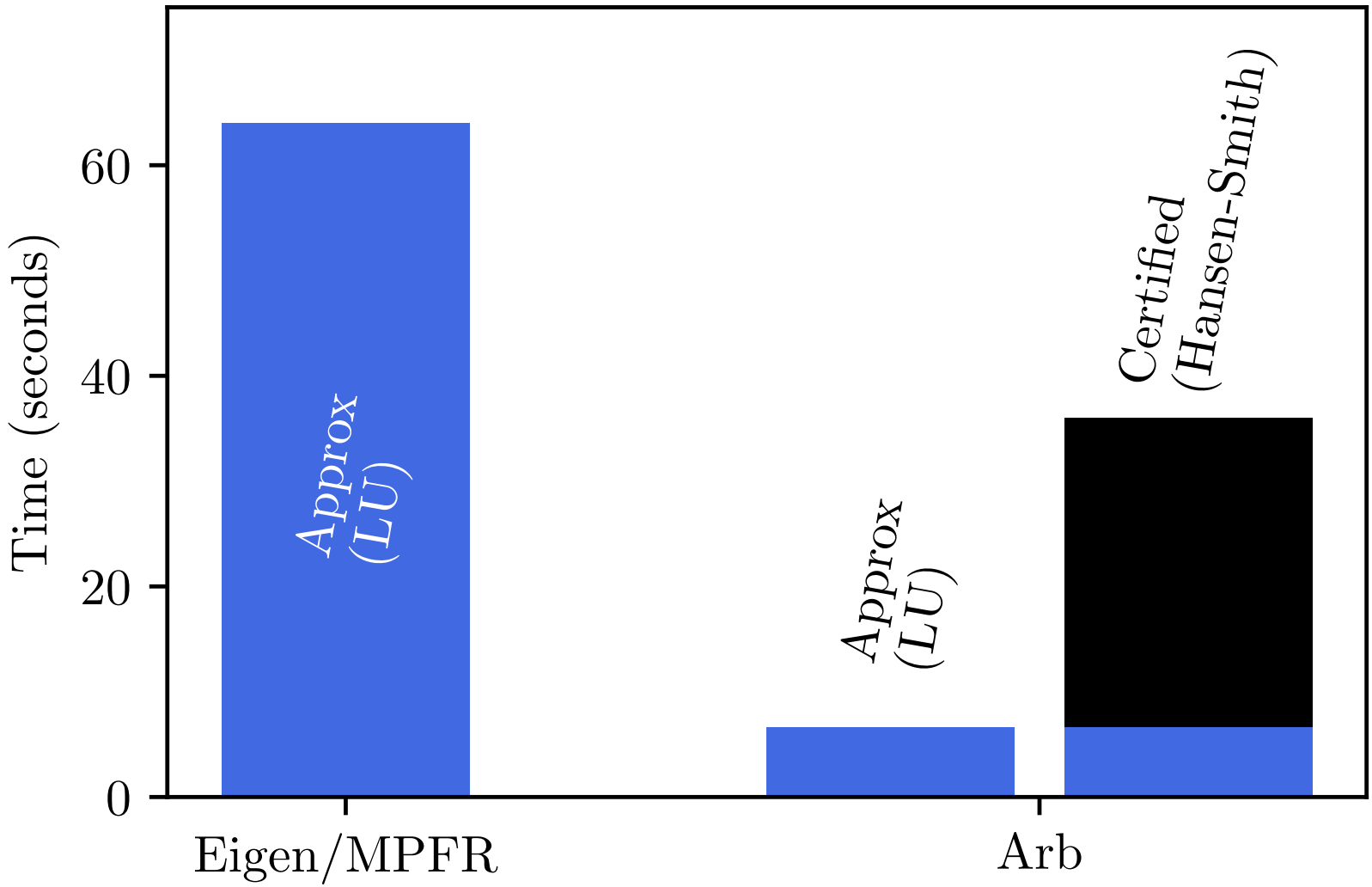
CPU time (s) to multiply two real 1000×1000 matrices

	$p = 53$	$p = 106$	$p = 212$	$p = 848$
BLAS	0.08			
QD		11	111	
MPFR	36	44	110	293
Arb (approx)	2.4	4.1	6.7	26
Arb (ball)	3.6	5.6	8.2	27

[F. J., *Faster arbitrary-precision dot product and matrix multiplication*, 2019]

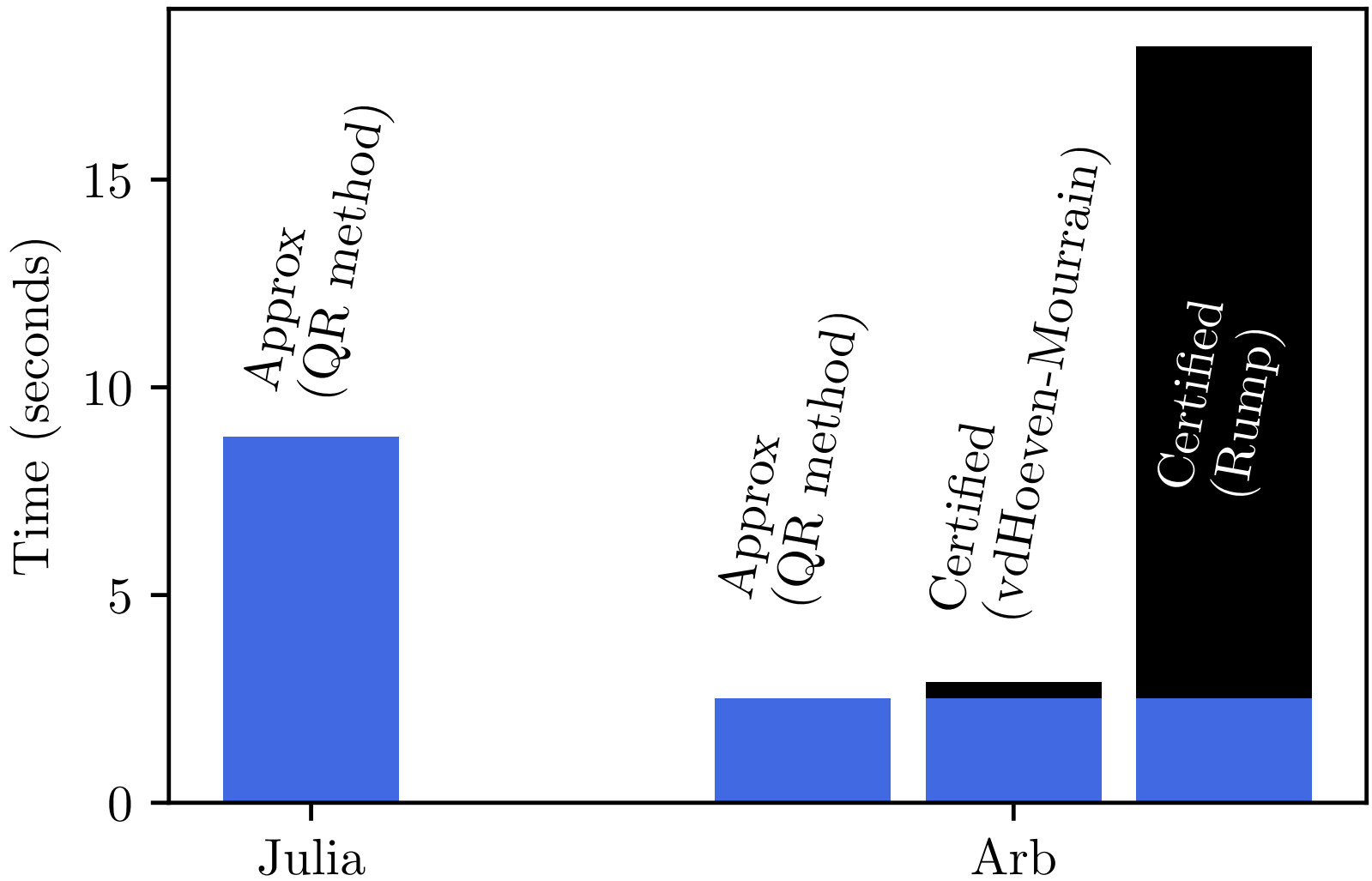
Performance: linear solving

Solving a dense real linear system $Ax = b$ ($N = 1000$, $p = 212$)



Performance: eigenvalues

Computing all eigenvalues and eigenvectors of a nonsymmetric complex matrix ($N = 100$, $p = 128$)



Numerical integration

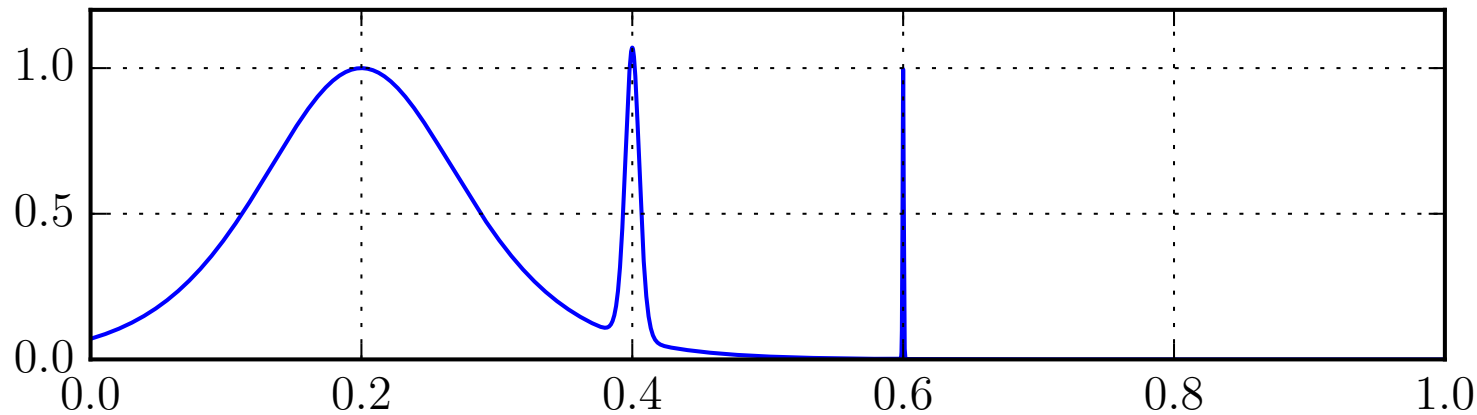
$$\int_a^b f(x)dx \; = \; ?$$

[F. J., *Numerical integration in arbitrary-precision ball arithmetic*, 2018]

[F. J. and M. Mezzarobba, *Fast and rigorous arbitrary-precision computation of Gauss-Legendre quadrature nodes and weights*, 2018]

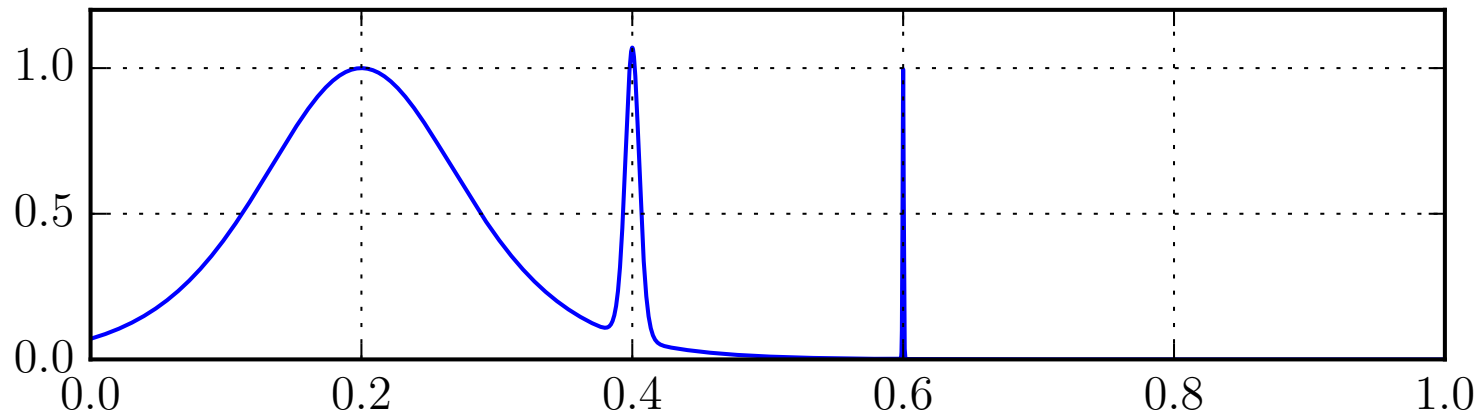
Example: spikes [Cranley and Patterson, 1971]

$$\int_0^1 \operatorname{sech}^2(10(x-0.2)) + \operatorname{sech}^4(100(x-0.4)) + \operatorname{sech}^6(1000(x-0.6))\ dx$$



Example: spikes [Cranley and Patterson, 1971]

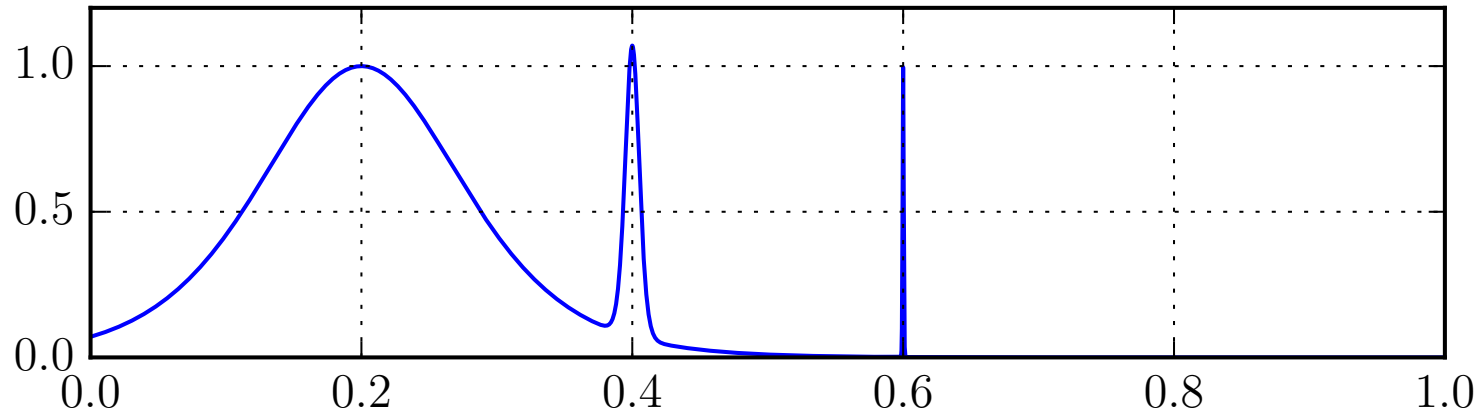
$$\int_0^1 \operatorname{sech}^2(10(x-0.2)) + \operatorname{sech}^4(100(x-0.4)) + \operatorname{sech}^6(1000(x-0.6))\ dx$$



Mathematica `NIntegrate`:	0.209736
Octave `quad`:	0.209736, error estimate 10^{-9}
Sage `numerical_integral`:	0.209736, error estimate 10^{-14}
SciPy `quad`:	0.209736, error estimate 10^{-9}
mpmath `quad`:	0.209819
Pari/GP `intnum`:	0.211316
Actual value:	0.210803

Example: spikes [Cranley and Patterson, 1971]

$$\int_0^1 \operatorname{sech}^2(10(x-0.2)) + \operatorname{sech}^4(100(x-0.4)) + \operatorname{sech}^6(1000(x-0.6))\ dx$$



Arb, 64-bit precision:

```
[0.21080273550054928 +/- 4.55e-18]       # time 0.003 s
```

333-bit precision:

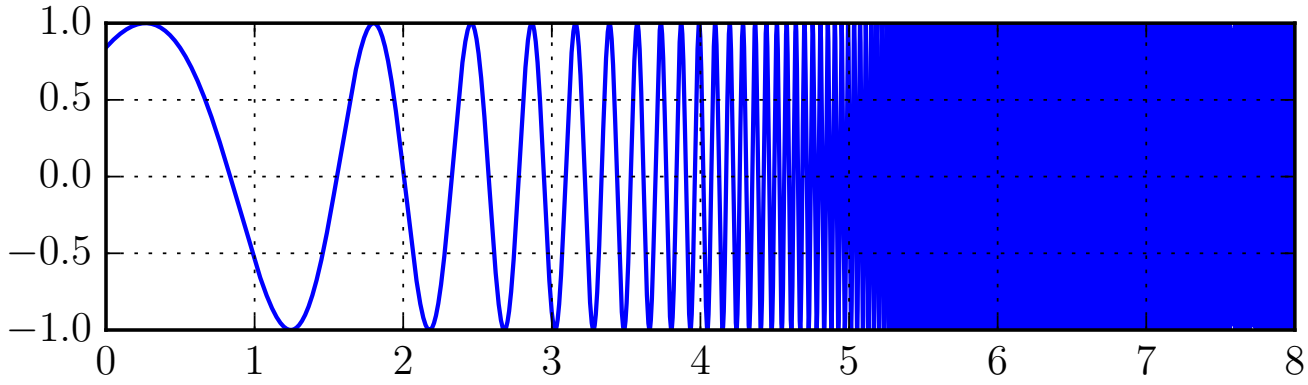
```
[0.21080273550054927737 56... +/- 3.67e-99]     # 0.02 s
```

3333-bit precision:

```
[0.21080273550054927737 56... +/- 1.39e-1001]  # 5.3 s
```

Example: violent oscillation [S. Rump, 2010]

$$\int_0^8 \sin(x+e^x)\,dx$$

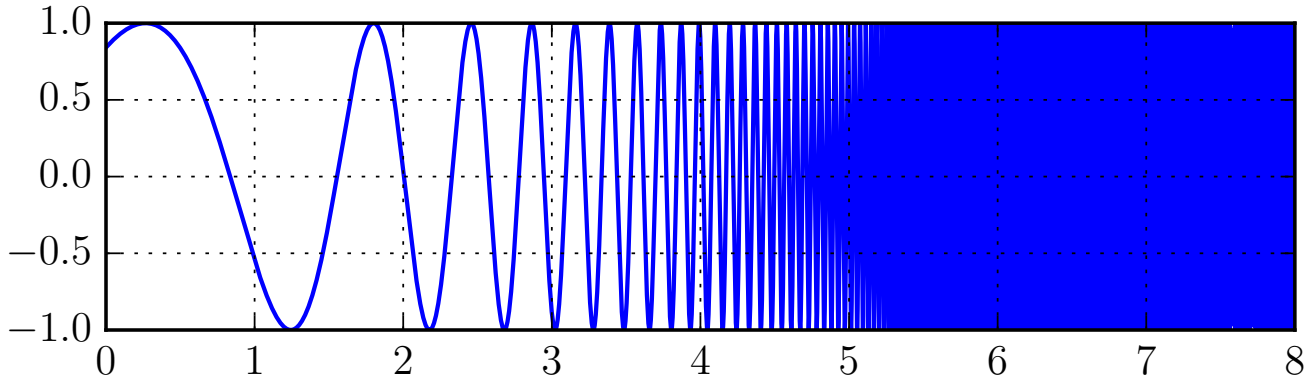


MATLAB's `quad` returned the incorrect `0.2511` in about 1 s

Rump's INTLAB gives `[0.34740016, 0.34740018]` in about 1 s

Example: violent oscillation [S. Rump, 2010]

$$\int_0^8 \sin(x+e^x)\,dx$$



MATLAB's `quad` returned the incorrect `0.2511` in about 1 s

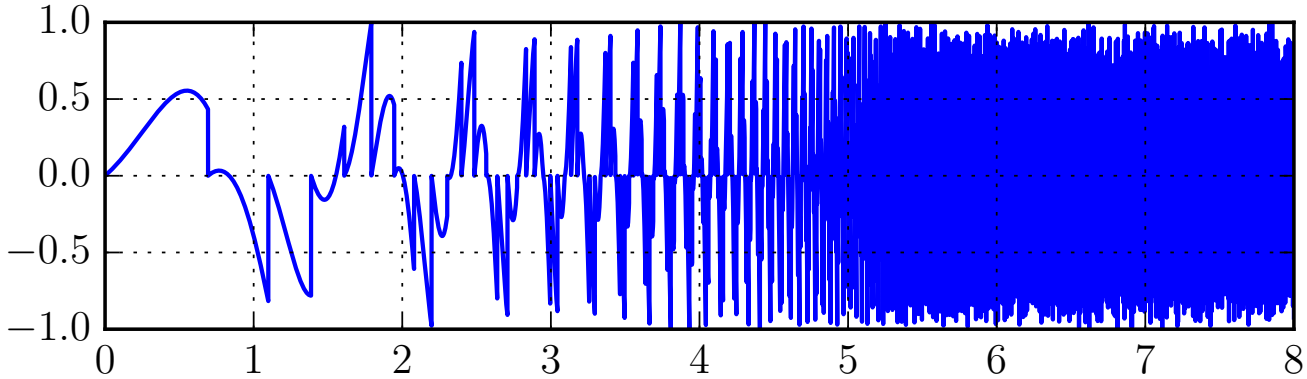
Rump's INTLAB gives `[0.34740016, 0.34740018]` in about 1 s

Arb at 64, 333, and 3333 bits:

```
[0.34740017265725 +/- 3.34e-15]    # 0.004 s
[0.34740017265... +/- 5.31e-96]    # 0.01 s
[0.34740017265... +/- 2.41e-999]   # 1 s
```

Example: a monster

$$\int_0^8 (e^x - \lfloor e^x \rfloor) \sin(x+e^x)\,dx$$ – now with 2979 discontinuities!

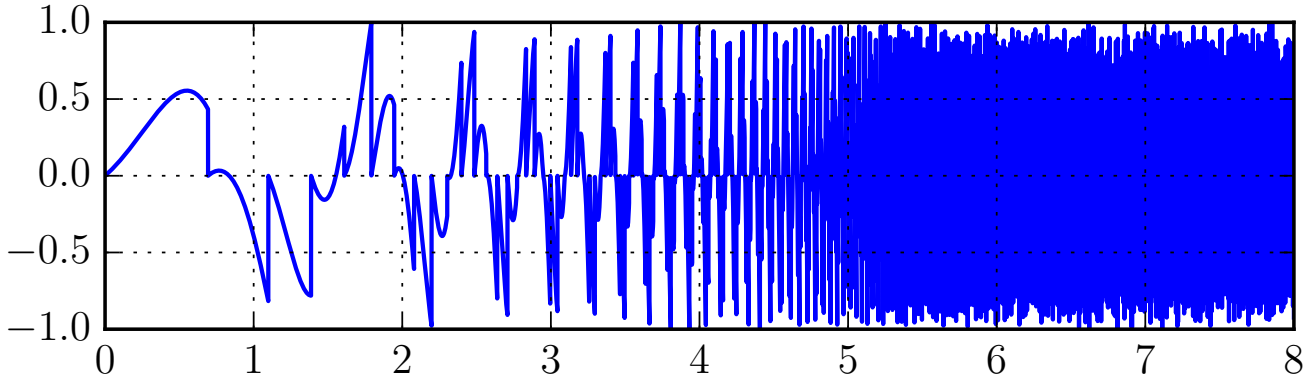


64-bit precision:

```
[+/- 5.45e+3]                                  # time 0.14 s
```

Example: a monster

$$\int_0^8 (e^x - \lfloor e^x \rfloor) \sin(x+e^x)\,dx$$ – now with 2979 discontinuities!



64-bit precision:

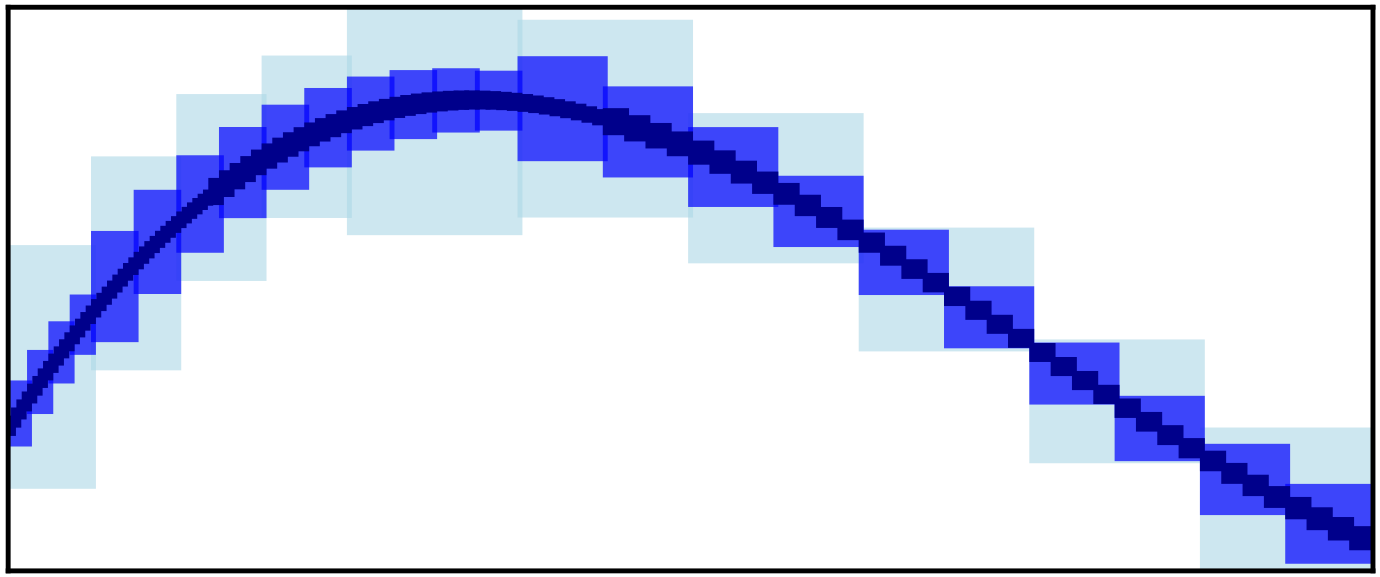
```
[+/- 5.45e+3]                                  # time 0.14 s
[0.0986517044784 +/- 4.46e-14]                 # time 5 s
```

333-bit precision:

```
[0.098651704478365206119658249764859856504169620792384491451091906830826680482290609839624064582
4 +/- 6.28e-95]  # 268 s
```

Brute force interval integration

$$\int_a^b f(x)dx \in (b-a)f([a,b]) \quad + \quad \text{adaptive subdivision of } [a,b]$$

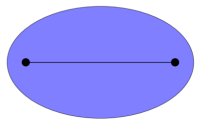


This is simple and general, but we need $2^{O(p)}$ evaluations to achieve p-bit accuracy!

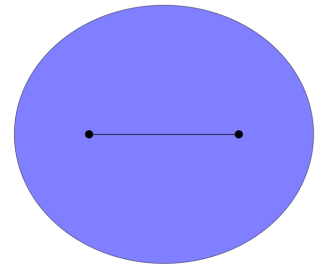
Efficient integration of piecewise analytic functions

- Gauss-Legendre quadrature with error bounds

$$\left| \int_a^b f(x)dx - \sum_{k=1}^n w_k f(x_k) \right| \leq \frac{M}{\rho^{2n}} \cdot |b-a| C_\rho, \quad |f(z)| \leq M$$



$\rho = 2.00$



$\rho = 3.73$

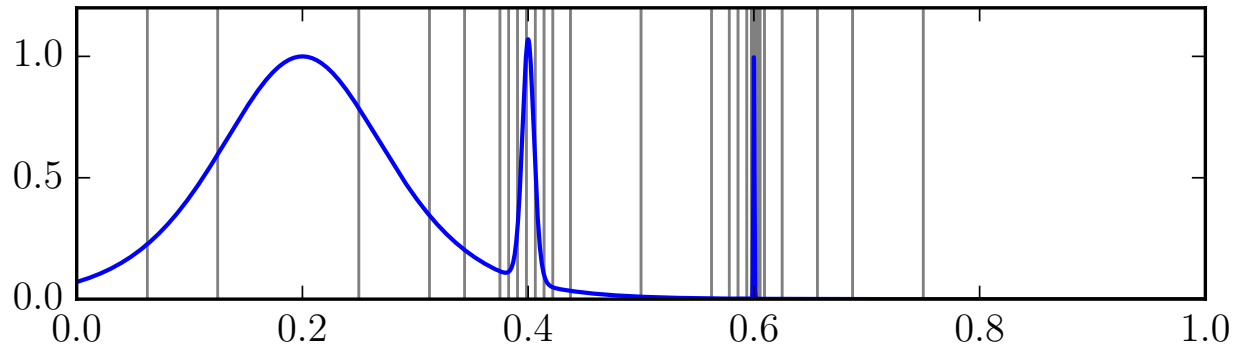
- If there are singularities too close to $[a, b]$, bisect (possibly falling back to naive enclosure [K. Petras, 1998 and 2002])

- Fast computation of high-degree, high-precision Gauss-Legendre nodes and weights

Adaptive subdivision

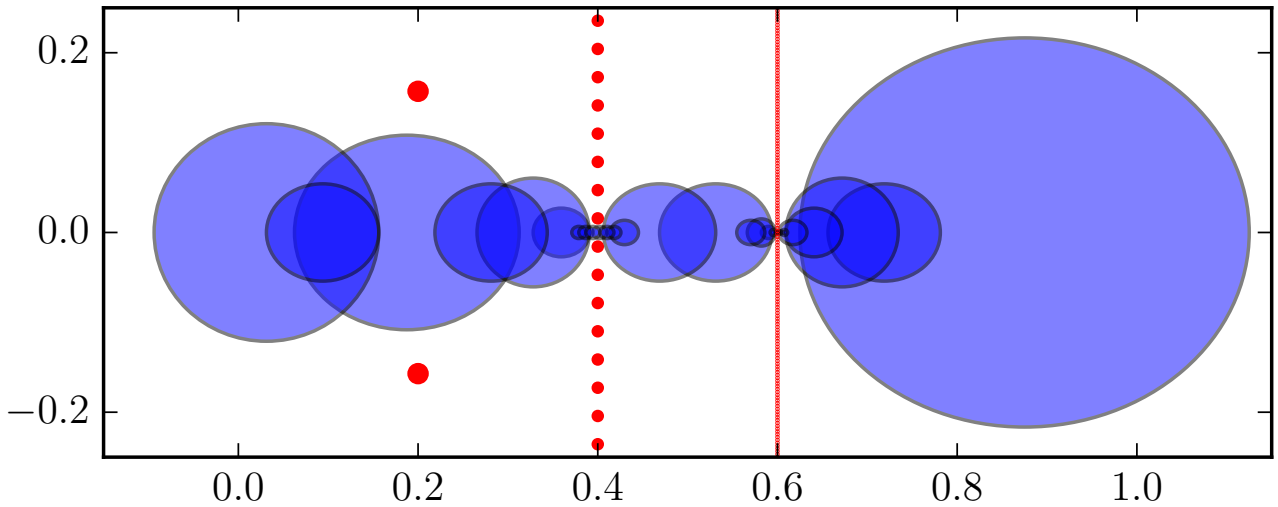
$$\int_0^1 \operatorname{sech}^2(10(x-0.2)) + \operatorname{sech}^4(100(x-0.4)) + \operatorname{sech}^6(1000(x-0.6))\ dx$$

Arb chooses
31 subintervals,
narrowest is 2^{-11}

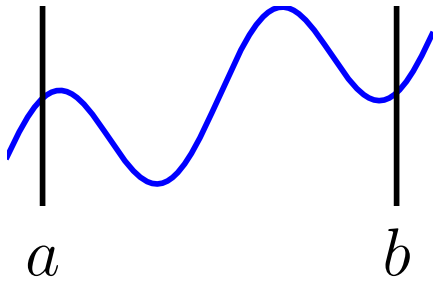


Complex ellipses
used for bounds

Red dots = poles

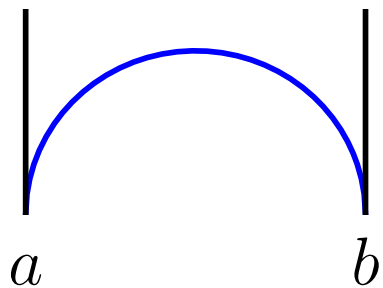


Typical proper integrals



Analytic around $[a, b]$

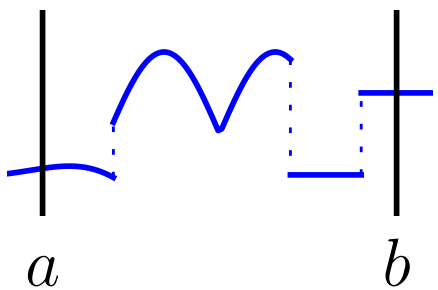
Complexity: $O(p)$ evaluations



Bounded algebraic-type singularities

Example: $\sqrt{1-x^2}$

Complexity: $O(p^2)$



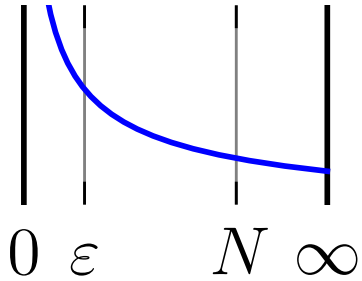
Piecewise analytic functions

Examples: $\lfloor x \rfloor, \operatorname{sgn}(x), |x|, \max(f(x), g(x))$

Complexity: $O(p^2)$

Typical improper integrals ($|a|$, $|b|$ or $|f| \to \infty$)

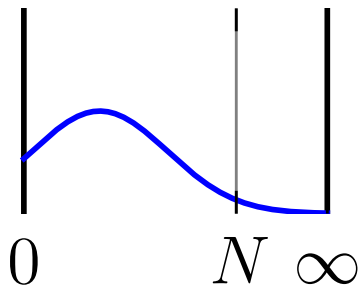
Manual truncation required, e.g. $\int_0^\infty f(x)dx \approx \int_\varepsilon^N f(x)dx$



Algebraic blow-up or decay

Examples: $\int_0^1 \frac{dx}{\sqrt{x}}$, $\int_0^1 \log(x)dx$, $\int_0^\infty \frac{dx}{1+x^2}$

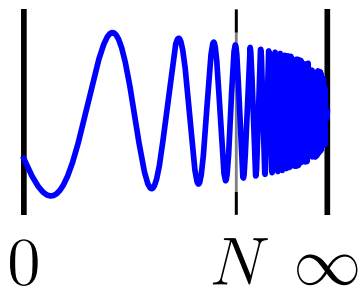
Complexity: $O(p^2)$



Exponential decay

Example: $\int_0^\infty e^{-x}\sin(x)dx$

Complexity: $O(p \log p)$



Essential singularity with slow decay

Example: $\int_1^\infty \frac{\sin(x)}{x}dx$

Complexity: $2^{O(p)}$

Applications of integration in ball arithmetic?

- Computing areas and volumes of bizarre things (duh)
- Special functions

$$\Gamma(s,z) = \int_z^\infty t^{s-1} e^{-t}\, dt$$

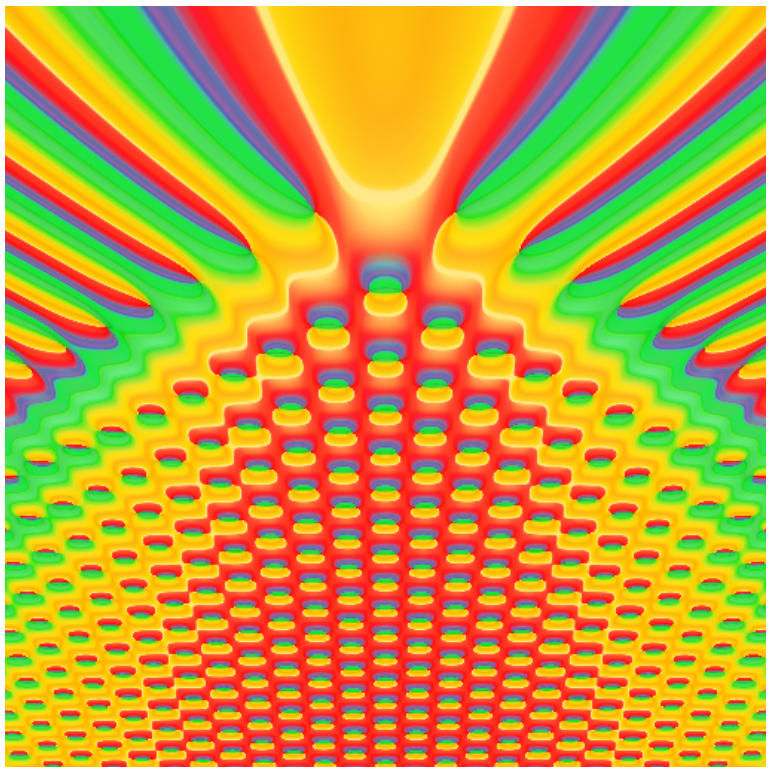
- (Inverse) Laplace/Fourier/Mellin transforms
- Taylor/Laurent/Fourier coefficients
- Counting zeros and poles

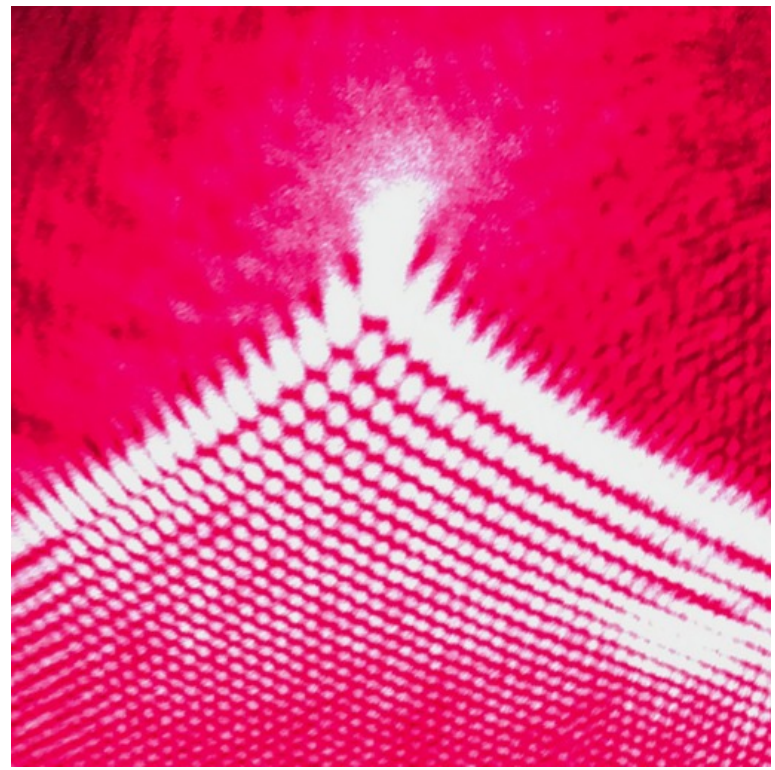
$$N - P = \frac{1}{2\pi i} \oint_C \frac{f'(z)}{f(z)}\, dz$$

- Acceleration of series (Euler-Maclaurin summation. . .)

Example: diffraction catastrophe integrals

$$P(x,y) = \int_{-\infty}^{\infty} e^{i(t^4+yt^2+xt)}\,dt = 2\int_{0}^{\infty} e^{-t^4+at^2+b}\cosh(ct)\,dt$$

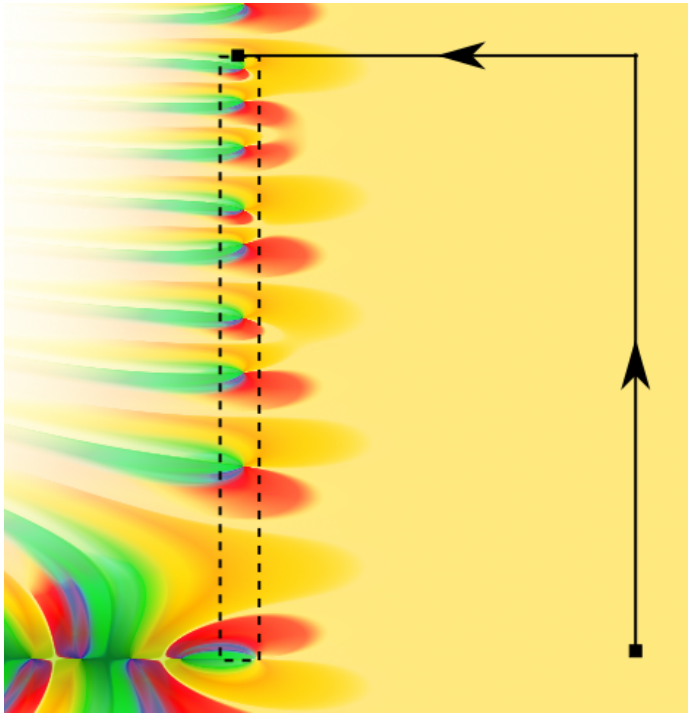




Left: 512×512 image rendered in 15 minutes with Arb ($|x| \leq 12.5$, $-20 \leq y \leq 5$). Using doubling precision (30, 60, ... bits). Near the bottom, $p = 120$ is required.

Right: photo of a cusp caustic produced by illuminating a flat surface with a laser beam through a droplet of water (image credit: Dan Piponi, CC-BY-SA)

Example: zeros of the Riemann zeta function



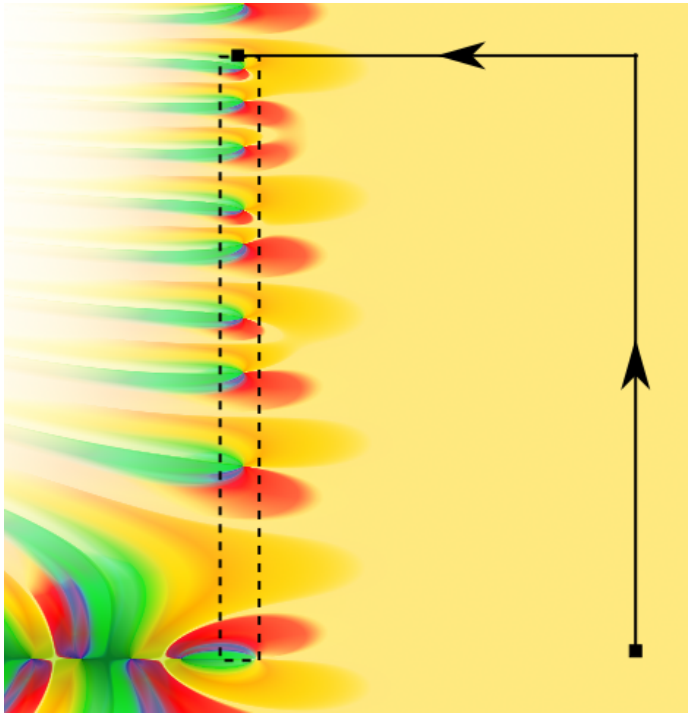
Number of zeros of $\zeta(s)$ on $R = [0,1] + [0,T]i$:

$$N(T) - 1 = \frac{1}{2\pi i}\int_\gamma \frac{\zeta'(s)}{\zeta(s)}\,ds = \frac{\theta(T)}{\pi} +$$

$$\frac{1}{\pi}\operatorname{Im}\left[\int_{1+\varepsilon}^{1+\varepsilon+Ti} \frac{\zeta'(s)}{\zeta(s)}\,ds + \int_{1+\varepsilon+Ti}^{\frac{1}{2}+Ti} \frac{\zeta'(s)}{\zeta(s)}\,ds\right]$$

T	p	Time (s)	Eval	Sub	$N(T)$
10^3	32	0.51	1219	109	`[649.00000 +/- 7.78e-6]`
10^6	32	16	5326	440	`[1747146.00 +/- 4.06e-3]`
10^9	48	1590	8070	677	`[2846548032.000 +/- 1.95e-4]`

Example: zeros of the Riemann zeta function



Number of zeros of $\zeta(s)$ on $R = [0,1] + [0,T]i$:

$$N(T) - 1 = \frac{1}{2\pi i}\int_\gamma \frac{\zeta'(s)}{\zeta(s)}\,ds = \frac{\theta(T)}{\pi} +$$

$$\frac{1}{\pi}\operatorname{Im}\left[\int_{1+\varepsilon}^{1+\varepsilon+Ti} \frac{\zeta'(s)}{\zeta(s)}\,ds + \int_{1+\varepsilon+Ti}^{\frac{1}{2}+Ti} \frac{\zeta'(s)}{\zeta(s)}\,ds\right]$$

T	p	Time (s)	Eval	Sub	$N(T)$
10^3	32	0.51	1219	109	`[649.00000 +/- 7.78e-6]`
10^6	32	16	5326	440	`[1747146.00 +/- 4.06e-3]`
10^9	48	1590	8070	677	`[2846548032.000 +/- 1.95e-4]`

Turing's method (impl. in Arb by D.H.J. Polymath): $T = 10^9$ in 0.1 s

Example: an integral with a large parameter

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

$$\gamma_n(v) = -\frac{\pi}{2(n+1)} \int_{-\infty}^{\infty} \frac{\left(\log\left(v - \frac{1}{2} + ix\right)\right)^{n+1}}{\cosh^2(\pi x)} dx$$

$$\gamma_{10^{100}}(1) \in [3.18743141870239927999741646993 \pm 2.89 \cdot 10^{-30}] \cdot 10^{e}$$

$$e = 23463942922772540809493678383990911609034476898698373852057791115792156640521582344171254175433483694$$

Some pen-and-paper analysis (steepest descent contour, tight enclosures near saddle point) needed for large n

[F. J., I. Blagouchine, *Computing Stieltjes constants using complex integration*, 2019]

Symbolics + numerics: improper integrals

$$\int_0^\infty \sqrt{x}e^{-x}dx = \int_0^N \sqrt{x}e^{-x}dx + \underbrace{\int_N^\infty \sqrt{x}e^{-x}dx}_{\text{Bound tail}}$$

- Computer algebra systems (Mathematica, Maple) use heuristics
- Simple cases in `CoqInterval` [Mahboubi, Melquiond and Sibut-Pinote, *Formally verified approximations of definite integrals*, 2019]

Symbolics + numerics: D-finite functions

Numerical evaluation and analytic continuation of D-finite functions in `ore_algebra` (Marc Mezzarobba)

```
sage: dop = z*Dz**2 + Dz   # ODE for log(z)

sage: dop.generalized_series_solutions(n=5)
[1 + O(z^5), (1 + O(z^5))*log(z)]

sage: dop.numerical_solution(ini=[0,1], path=[1,2])
[0.69314718055994531 +/- 1.23e-18]

sage: dop.numerical_solution(ini=[0,1], path=[1,I,-1])
[+/- 6.62e-28] + [3.141592653589793 +/- 3.88e-17]*I

sage: dop.numerical_solution(ini=[0,1], path=[1,-I,-1])
[+/- 6.62e-28] + [-3.141592653589793 +/- 3.88e-17]*I
```

The Mathematical Functions Grimoire

`http://fungrim.org`

An attempt to make a better (at least for me)

1. reference work
2. software library for symbolic computation

for special functions

---

grimoire = book of magic formulas

Principles

- Both a website and a software library
- All content is symbolic, semantic data
- Explicit, rigorous conditions/assumptions

Example: `http://fungrim.org/entry/15dd69/`

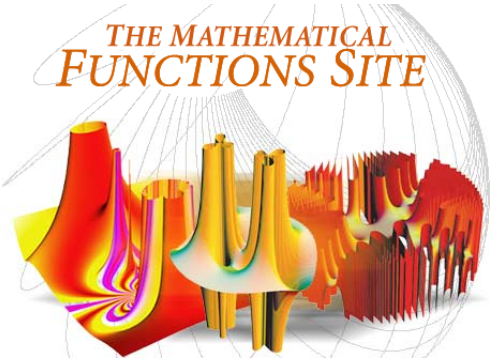
$$|T_n(x)| \le 1$$

Assumptions: $n \in \mathbb{Z}$ and $x \in [-1, 1]$

```
Entry(ID("15dd69"),
    Formula(LessEqual(Abs(ChebyshevT(n, x)), 1)),
    Variables(n, x),
    Assumptions(And(Element(n, ZZ),
        Element(x, ClosedInterval(-1, 1)))))
```

Why yet another reference work?

Digital Library of Mathematical Functions



WIKIPEDIA
The Free Encyclopedia

Dynamic Dictionary of Mathematical Functions

Some combination of:

- Not open source
- Not semantic
- Not general
- Not comprehensive
- Not consistent
- Not explicit about assumptions
- Not useful for complex variables

Inspiration: Rubi by Albert D. Rich

$$\int x^m dx, m \neq -1 \quad \rightarrow \quad \frac{x^{m+1}}{m+1}$$
$$\int x^{-1} dx \quad \rightarrow \quad \log(x)$$

`https://rulebasedintegration.org`

> *[Rubi] uses pattern matching to uniquely determine which of its over 6600 integration rules to apply*

> *Rubi dramatically out-performs other symbolic integrators, including Maple and Mathematica*

> *Certainly much of analysis including equation solving, expression simplification, differentiation, summation, limits, etc. can be automated using this paradigm*

Inspiration: Rubi by Albert D. Rich

$$\int x^m dx, m \neq -1 \quad \rightarrow \quad \frac{x^{m+1}}{m+1}$$
$$\int x^{-1} dx \quad \rightarrow \quad \log(x)$$

`https://rulebasedintegration.org`

> *[Rubi] uses pattern matching to uniquely determine which of its over 6600 integration rules to apply*

> *Rubi dramatically out-performs other symbolic integrators, including Maple and Mathematica*

> *Certainly much of analysis including equation solving, expression simplification, differentiation, summation, limits, etc. can be automated using this paradigm*

Idea: a semantic formula database like Fungrim can be used for symbolic rewriting (semi-automatically?)

Symbolic computation: is it reliable?

> ***The answer might not be valid for certain exceptional values of the parameters.***

[Wolfram Language (*Mathematica*) Documentation, 2019]

> ***One should never take [symbolic] results at face value*** *and should always run some kind of check, e.g., by substituting the claimed solution into the equation or comparing a symbolic solution with a numerically computed one.*

[N. Higham, in *The Princeton Companion to Applied Mathematics*, 2015]

Example: what is ${}_1F_1(-1,-1,1)$?

Mathematica has a split personality

```
In[•]:= Hypergeometric1F1[n, m, x] /. {m → -1, n → -1, x → 1}
Out[•]= 2

In[•]:= (Hypergeometric1F1[n, m, x] /. {m → n}) /. {n → -1, x → 1}
Out[•]= e
```

Example: what is ${}_1F_1(-1,-1,1)$?

`http://fungrim.org/entry/dec042/`

$$ {}_1F_1(-n,b,z) = \sum_{k=0}^{n} \frac{(-n)_k}{(b)_k} \frac{z^k}{k!} $$

Assumptions:
$n \in \mathbb{Z}_{\geq 0}$ and $b \in \mathbb{C}$ and not $(b \in \{0,-1,\ldots\}$ and $b > -n)$ and $z \in \mathbb{C}$

`http://fungrim.org/entry/be533c/`

$$ {}_1F_1(a,b,z) = e^z \, {}_1F_1(b-a,b,-z) $$

Assumptions: $a \in \mathbb{C}$ and $b \in \mathbb{C} \setminus \{0,-1,\ldots\}$ and $z \in \mathbb{C}$

Rigorous conditions

> *The automatic exploration of conditions or alternative results requires considerable computational resources, and for the sake of speed there is an attraction to picking one 'obvious' answer. [...]* ***The difficulty is to balance efficiency against correctness.***

[R. Corless and D. Jeffrey, *Well... It Isn't Quite That Simple*, 1992]

27 years later, what is the right balance?

Rigorous conditions

> *The automatic exploration of conditions or alternative results requires considerable computational resources, and for the sake of speed there is an attraction to picking one 'obvious' answer. [...]* ***The difficulty is to balance efficiency against correctness.***

[R. Corless and D. Jeffrey, *Well... It Isn't Quite That Simple*, 1992]

27 years later, what is the right balance?

A thought:
Rigorous propagation of conditions in symbolic computation
$\cong$ Interval arithmetic in numerical computation

Thank you!