Numerical integration in arbitrary-precision ball arithmetic

Fredrik Johansson
Inria Bordeaux

ICMS 2018
University of Notre Dame, IN, USA
2018-07-27

Introduction

Arb (`http://arblib.org`) – C library for arbitrary-precision ball arithmetic (+polynomials, matrices, special functions...)

```
[3.14159265358979323846264 3 +/- 4.03e-25]
[-0.200293 +/- 8.48e-7] + [0.979736 +/- 3.44e-7]*I
```

Introduction

Arb (`http://arblib.org`) – C library for arbitrary-precision ball arithmetic (+polynomials, matrices, special functions...)

```
[3.141592653589793238462643 +/- 4.03e-25]
[-0.200293 +/- 8.48e-7] + [0.979736 +/- 3.44e-7]*I
```

New feature (2018): **fast rigorous numerical integration**

- Documentation: `http://arblib.org/acb_calc.html`
- Demo: `examples/integrals.c`

Introduction

Arb (`http://arblib.org`) – C library for arbitrary-precision ball arithmetic (+polynomials, matrices, special functions...)

```
[3.141592653589793238462643 +/- 4.03e-25]
[-0.200293 +/- 8.48e-7] + [0.979736 +/- 3.44e-7]*I
```

New feature (2018): **fast rigorous numerical integration**

- Documentation: `http://arblib.org/acb_calc.html`
- Demo: `examples/integrals.c`

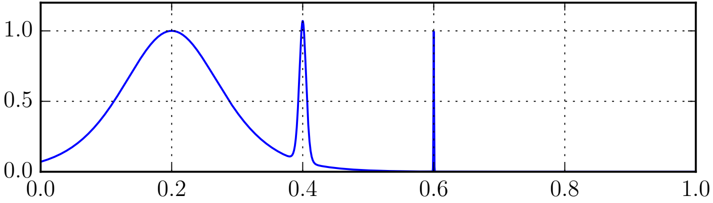
High-level interfaces in SageMath* and Nemo.jl.
Example: $\int_0^1 \cos(x)\sin(x)dx$

```
sage: C = ComplexBallField(100)
sage: C.integral(lambda x, _: cos(x) * sin(x), 0, 1)
[0.354036709136785596749392057 +/- 8.89e-30]
```

*Thanks to Marc Mezzarobba and Vincent Delecroix

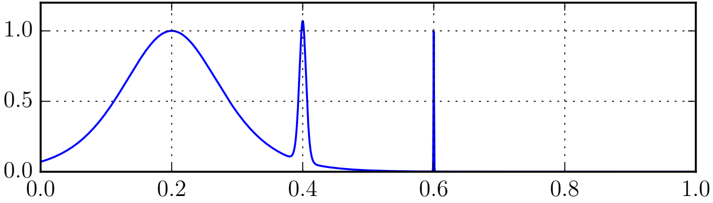
A nice and smooth function (Cranley and Patterson, 1971)

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



A nice and smooth function (Cranley and Patterson, 1971)

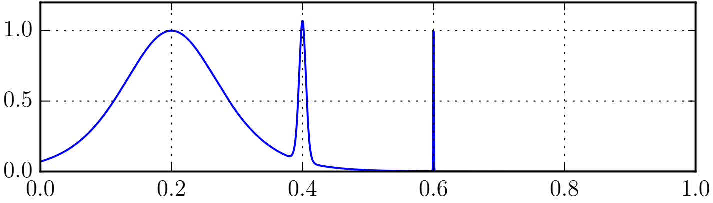
$$\int_0^1 \operatorname{sech}^2(10(x-0.2)) + \operatorname{sech}^4(100(x-0.4)) + \operatorname{sech}^6(1000(x-0.6))\ dx$$



Mathematica `NIntegrate`: 0.209736

A nice and smooth function (Cranley and Patterson, 1971)

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

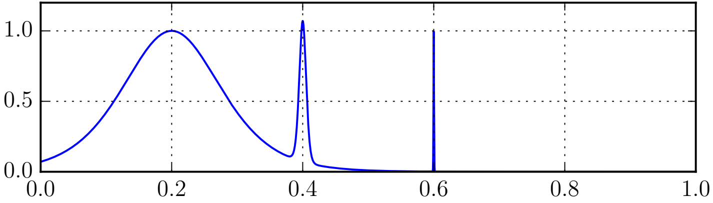


Mathematica `NIntegrate`: 0.209736

Octave `quad`: 0.209736, error estimate 10^{-9}

A nice and smooth function (Cranley and Patterson, 1971)

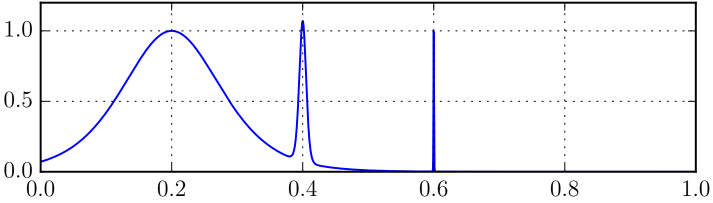
$$\int_0^1 \operatorname{sech}^2(10(x-0.2)) + \operatorname{sech}^4(100(x-0.4)) + \operatorname{sech}^6(1000(x-0.6))\ dx$$



Mathematica `NIntegrate`: 0.209736
Octave `quad`: 0.209736, error estimate 10^{-9}
Sage `numerical_integral`: 0.209736, error estimate 10^{-14}

A nice and smooth function (Cranley and Patterson, 1971)

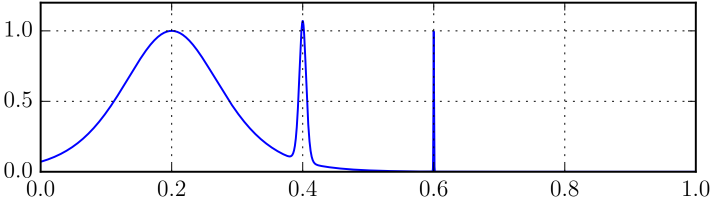
$$\int_0^1 \operatorname{sech}^2(10(x-0.2)) + \operatorname{sech}^4(100(x-0.4)) + \operatorname{sech}^6(1000(x-0.6))\ dx$$



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}

A nice and smooth function (Cranley and Patterson, 1971)

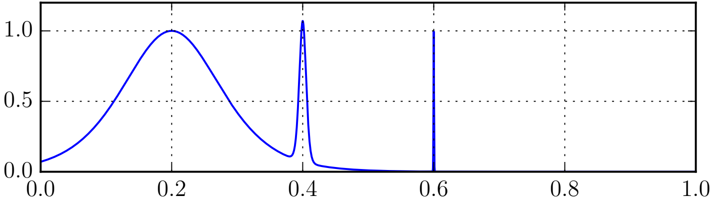
$$\int_0^1 \operatorname{sech}^2(10(x-0.2)) + \operatorname{sech}^4(100(x-0.4)) + \operatorname{sech}^6(1000(x-0.6))\ dx$$



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

A nice and smooth function (Cranley and Patterson, 1971)

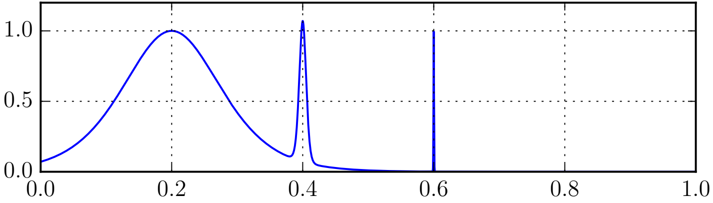
$$\int_0^1 \operatorname{sech}^2(10(x-0.2)) + \operatorname{sech}^4(100(x-0.4)) + \operatorname{sech}^6(1000(x-0.6))\ dx$$



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

A nice and smooth function (Cranley and Patterson, 1971)

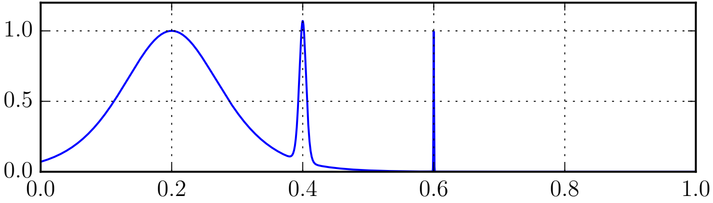
$$\int_0^1 \operatorname{sech}^2(10(x-0.2)) + \operatorname{sech}^4(100(x-0.4)) + \operatorname{sech}^6(1000(x-0.6)) \; dx$$



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

A nice and smooth function (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:

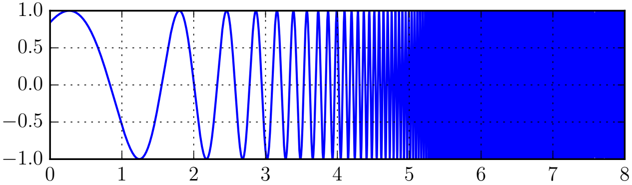
```
[0.21080273550054927737... +/- 3.67e-99]    # 0.02 s
```

3333-bit precision:

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

Another example: violent oscillation

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

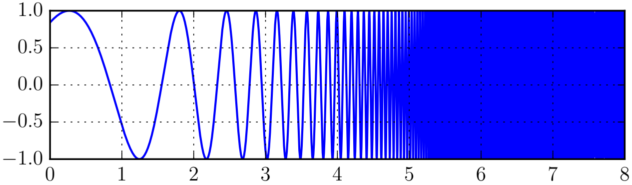


S. Rump (2010) noticed that MATLAB's `quad` returned the incorrect `0.2511` after 1 second of computation.

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

Another example: violent oscillation

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



S. Rump (2010) noticed that MATLAB's `quad` returned the incorrect `0.2511` after 1 second of computation.

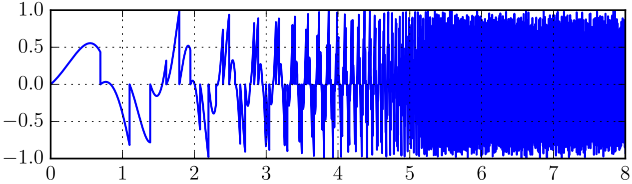
Rump's INTLAB gives `[0.34740016, 0.34740018]` in about 1 s

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
```

Yet another example: a monster

$$\int_0^8 (e^x - \lfloor e^x \rfloor) \sin(x+e^x)\, dx$$ – now with 2980 discontinuities!

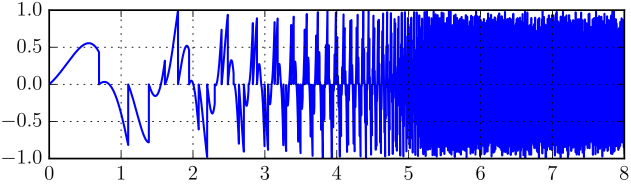


64-bit precision:

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

Yet another example: a monster

$$\int_0^8 (e^x - \lfloor e^x \rfloor) \sin(x+e^x)\, dx$$ – now with 2980 discontinuities!



64-bit precision:

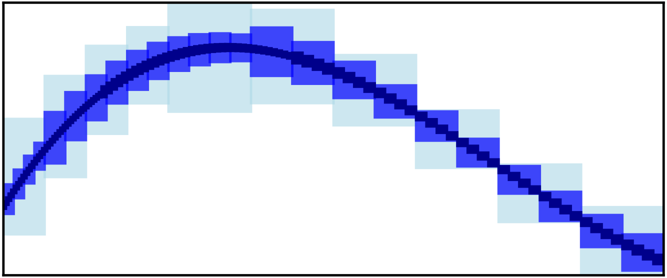
```
[+/- 5.45e+3]                      # time 0.14 s
[0.0986517044784 +/- 4.46e-14]     # time 5 s
```

333-bit precision:

```
[0.09865170447836520611965824976485985650416962079238449145
10919068308266804822906098396240645824 +/- 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 analytic functions

We can achieve p-bit accuracy with $n = O(p)$ work.

$2 \times 2 = 4$ potential algorithms:

Approximation:

$O(x^n)$ Taylor series	n-point quadrature
$\int \sum_{k=0}^{n-1} a_k x^k = \sum_{k=0}^{n-1} a_k \frac{x^{k+1}}{k+1}$	$\int f(x) dx \approx \sum_{k=1}^{n} w_k f(x_k)$

Error bounds:

Using derivatives $f^{(n)}$ on $[a, b]$	Using $\lvert f \rvert$ on a complex domain around $[a, b]$

Efficient integration of analytic functions

We can achieve p-bit accuracy with $n = O(p)$ work.

$2 \times 2 = 4$ potential algorithms:

Approximation:

$O(x^n)$ Taylor series	n-point quadrature
$\int \sum_{k=0}^{n-1} a_k x^k = \sum_{k=0}^{n-1} a_k \frac{x^{k+1}}{k+1}$	$\int f(x) dx \approx \sum_{k=1}^{n} w_k f(x_k)$

Error bounds:

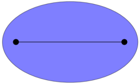
Using derivatives $f^{(n)}$ on $[a, b]$	Using $\lvert f \rvert$ on a complex domain around $[a, b]$

Fast generation of Gauss-Legendre quadrature nodes (x_k, w_k) due to F.J. and M. Mezzarobba (`arxiv.org/abs/1802.03948`).

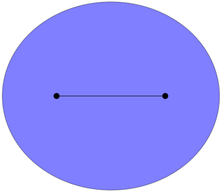
Error bounds for Gauss-Legendre quadrature

If f is analytic with $|f(z)| \leq M$ on an ellipse E with foci $-1, 1$ and semi-axes X, Y with $\rho = X + Y > 1$, then

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



$X = 1.25, Y = 0.75, \rho = 2.00$



$X = 2.00, Y = 1.73, \rho = 3.73$

Fast convergence when no singularities are close to $[a, b]$, but should be combined with subdivision otherwise!

Adaptive integration algorithm

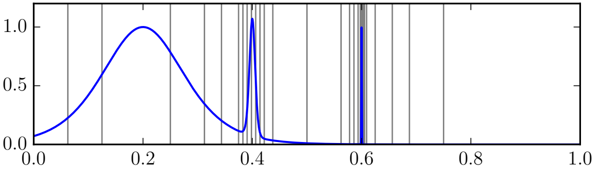
1. Compute $(b-a)f([a,b])$. If the error is $\leq \varepsilon$, done!
2. On an ellipse E around $[a,b]$, bound $|f|$ and check that f is analytic. If the error of Gauss-Legendre quadrature is $\leq \varepsilon$, compute it – done!
3. Split at $m = (a+b)/2$ and integrate on $[a,m]$, $[m,b]$ recursively.

Knut Petras (2002) pointed out that this guarantees rapid convergence for a large class of piecewise analytic functions.

Adaptive subdivision

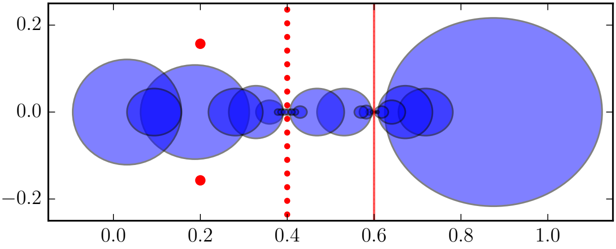
$$\int_0^1 \operatorname{sech}^2(10(x-0.2)) + \operatorname{sech}^4(100(x-0.4)) + \operatorname{sech}^6(1000(x-0.6))\ dx$$

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

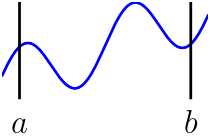


Complex ellipses used for bounds

Red dots = poles

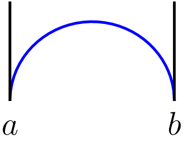


Typical proper integrals



Analytic around $[a, b]$

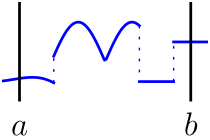
Complexity: $O(p)$



Bounded algebraic-type singularities

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

Complexity: $O(p^2)$



Piecewise analytic functions*

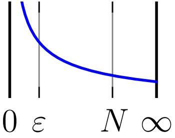
Examples: $\lfloor x \rfloor$, $\mathrm{sgn}(x)$, $|x|$, $\max(f(x), g(x))$

Complexity: $O(p^2)$

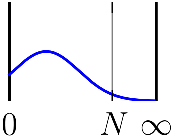
* Trick: extend piecewise real functions to the complex plane.
Discontinuities $\to$ branch cuts.

Typical improper integrals

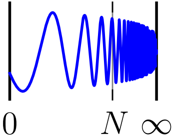
Manual truncation required, e.g. $\int_0^\infty f(x)dx \approx \int_\varepsilon^N f(x)dx$
if $|a|$, $|b|$ or $|f| \to \infty$



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: $e^{-x}\sin(x)$
Complexity: $O(p \log p)$



Essential singularity with slow decay
Example: $\int_1^\infty \frac{\sin(x)}{x} dx$
Complexity: $2^{O(p)}$

Timings: f analytic around $[a, b]$

p	Pari/GP	mpmath	Arb	Sub	Eval	Pari/GP	mpmath	Arb	Sub	Eval
	$I_0 = \int_0^1 1/(1+x^2)\,dx$					$I_1 = \int_0^1 \sum_{k=1}^{3} \operatorname{sech}^{2k}(10^k(x-0.2k))\,dx$				
64	0.00039	0.0011	0.000036	2	52	0.54	5.0	0.0031	31	768
333	0.0043	0.0058	0.00017	2	188	12	38	0.023	31	3086
3333	1.0	0.13	0.012	2	2056	3385	-	5.3	31	30092
	$I_2 = \int_0^{\pi} x\sin(x)/(1+\cos^2(x))\,dx$					$I_3 = \int_0^{1000} W_0(x)\,dx$				
64	0.00077	0.0046	0.00022	6	159	0.0037	0.032	0.00093	12	273
333	0.0088	0.037	0.0018	6	643	0.052	0.25	0.0095	12	1109
3333	2.2	4.4	0.43	6	6171	11	25	1.3	12	12043
	$I_4 = \int_0^{100} \sin(x)\,dx$					$I_5 = \int_0^{8} \sin(x+e^x)\,dx$				
64	0.0012	0.0014	0.000070	1	72	0.063	0.25	0.0035	25	2239
333	0.015	0.018	0.00029	1	139	0.22	0.58	0.013	21	3940
3333	2.0	0.71	0.031	1	526	14	12	0.9	6	8341
	$I_6 = \int_{-1}^{1} e^{-x} \operatorname{erf}\left(\sqrt{1250}\,x + \frac{3}{2}\right)dx$					$I_7 = \int_1^{1+1000i} \Gamma(x)\,dx$				
64	0.024	0.057	0.0054	6	438	0.054	0.093	0.0046	12	324
333	0.50	0.22	0.047	4	791	0.65	1.1	0.091	14	1456
3333	173	466	5.7	2	2923	561	847	48	14	16535

Timings: endpoint singularities and infinite intervals

p	Pari/GP	mpmath	Arb	Sub	Eval	Pari/GP	mpmath	Arb	Sub	Eval
			$E_0 = \int_0^1 \sqrt{1-x^2}\,dx$					$E_1 = \int_0^\infty 1/(1+x^2)\,dx$ *		
64	0.00041	0.00067	0.00057	44	674	0.00060	0.0012	0.0022	190	2887
333	0.0044	0.0060	0.015	223	12687	0.0068	0.011	0.048	997	51900
3333	0.94	0.18	6.6	2223	1.2 M	1.7	0.24	27	9997	4.7 M
			$E_2 = \int_0^1 \log(x)/(1+x)\,dx$ *					$E_3 = \int_0^\infty \operatorname{sech}(x)\,dx$ *		
64	0.00081	0.00094	0.0012	67	1026	0.0011	0.0043	0.00022	7	181
333	0.011	0.011	0.038	336	19254	0.013	0.098	0.0019	9	853
3333	1.7	1.08	106	3336	1.8 M	3.5	3.3	0.68	12	12046
			$E_4 = \int_0^\infty e^{-x^2+ix}\,dx$ *					$E_5 = \int_0^\infty e^{-x}\operatorname{Ai}(-x)\,dx$ *		
64	0.0014	0.016	0.00017	1	98	-	0.91	0.012	9	842
333	0.017	0.13	0.0016	2	397	-	26	0.94	124	24548
3333	4.7	7.1	0.47	4	3894	-	10167	502	1205	0.7 M

* For Arb, the path was truncated manually (with error $\leq 2^{-p}$)

Timings: mid-interval jumps/kinks

p	Arb	Sub	Eval	Arb	Sub	Eval
	$\int_0^1 \lvert x^4+10x^3+19x^2-6x-6\rvert\, e^x\, dx$			$\int_0^{100} \lceil x \rceil\, dx$		
64	0.0016	70	1093	0.014	5536	16606
333	0.049	339	18137	0.12	33512	100534
3333	101	3339	1624951	1.6	345512	1036534
	$\int_0^{10} (x - \lfloor x \rfloor - \frac{1}{2}) \max(\sin(x), \cos(x))\, dx$			$\int_{-1-i}^{-1+i} \sqrt{x}\, dx$		
64	0.026	1257	16168	0.0021	132	1462
333	1.2	7076	394881	0.067	670	28304
3333	2588	71984	39128525	35	6670	2669940

High accuracy with mpmath or Pari/GP is not possible without manually splitting at all the singular points.

Defining functions

The user provides the integrand $f(z)$ as a black-box function that takes two parameters:

- A complex ball (rectangle) representing z
- A boolean flag *analytic*
  - *False* - the function returns an enclosure of $f(z)$. There are no assumptions about analyticity.
  - *True* - the function returns an enclosure of $f(z)$. It must return non-finite (NaN, $[\pm\infty]$) if the ball z contains any non-analytic point of f.

The user can always ignore the *analytic* flag when f is a composition of meromorphic functions.

Defining functions

The *analytic* flag **must** be handled
when the integrand has branch cuts.

$$\int_1^4 \sqrt{x}\, dx = \frac{14}{3}$$

```
sage: F1 = lambda x, _: x.sqrt()                  # WRONG!
sage: CBF.integral(F1, 1, 4)
[4.66941489478101 +/- 7.48e-15]

sage: F2 = lambda x, a: x.sqrt(analytic=a)        # correct
sage: CBF.integral(F2, 1, 4)
[4.6666666666667 +/- 4.62e-14]
```

Versions of common functions ($\sqrt{x}$, $\log(x)$, x^y, $|x|$, $\lfloor x \rfloor$, $\max(x,y)$, $\ldots$) with builtin branch cut detection are provided in Arb.

Optional settings for the integration algorithm

The user specifies:

- Working precision p
- Absolute and relative tolerances ε_{abs} and ε_{rel}

Configurable work limits:

- Maximum quadrature degree (default: $O(p)$)
- Number of calls to the integrand (default: $O(p^2)$)
- Number of queued subintervals (default: $O(p)$)
- Use stack (default) or global priority queue for the list of subintervals generated by bisection

Applications

- Special functions:

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

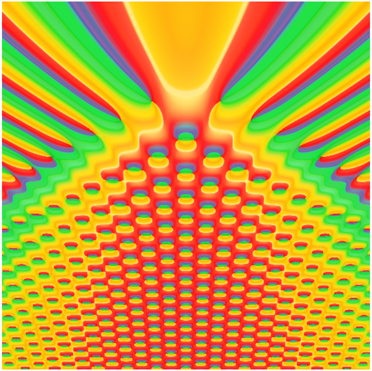
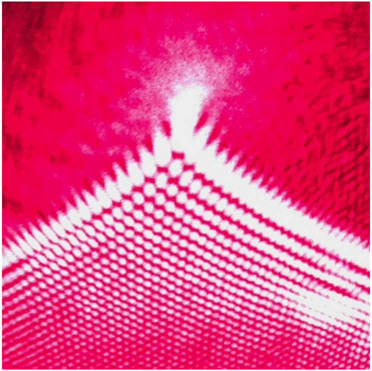
- (Inverse) Laplace/Fourier/Mellin transforms
- Taylor/Laurent/Fourier coefficients
- Counting zeros and poles:

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

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

Example: diffraction catastrophe integrals

$$P(x,y) = \int_{-\infty}^{\infty} e^{i(t^4+yt^2+xt)}\,dt = 2\int_0^{\infty} e^{-t^4+at^2+b}\cosh(ct)dt$$

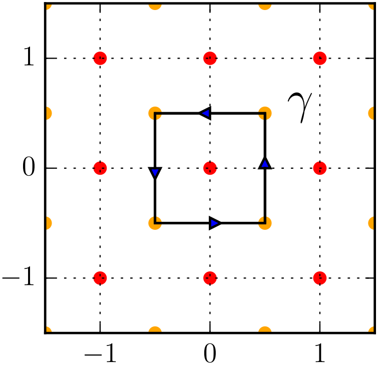
Left: 512×512 image rendered in 15 minutes with Arb ($|x| \leq 12.5$, $-20 \leq y \leq 5$). Using doubling precision (30, 60, ... bits). Near the bottom, $p = 120$ is required.

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)

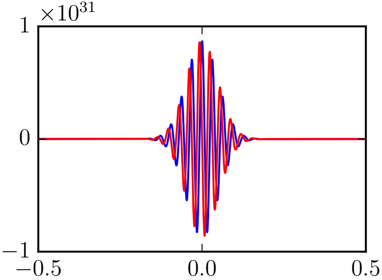
Example: Laurent series of elliptic functions

$$\wp(z;\tau) = \sum_{n=-2}^{\infty} a_n(\tau) z^n, \quad a_n = \frac{1}{2\pi i}\int_\gamma \frac{\wp(z)}{z^{n+1}} dz$$

$\wp(z)$ with $\tau = i$ has poles at $z = M + Ni \quad (M, N \in \mathbb{Z})$.



One segment ($n = 100$):



Example: Laurent series of elliptic functions

$$\wp(z;\tau) = \sum_{n=-2}^{\infty} a_n(\tau) z^n, \quad a_n = \frac{1}{2\pi i} \int_\gamma \frac{\wp(z)}{z^{n+1}} dz$$

$a_{-2}, \ldots, a_{100}$ with 333-bit precision (0.8 seconds for each a_n):

```
a[-2]  = [1.0000000000000000 ... 00000 +/- 3.57e-98] + [+/- 1.89e-98]*I
a[-1]  =                              [+/- 4.11e-98] + [+/- 2.57e-98]*I
a[0]   =                              [+/- 1.02e-97] + [+/- 5.39e-98]*I
a[1]   =                              [+/- 1.41e-97] + [+/- 1.35e-97]*I
a[2]   = [9.453636006461692  ... 52235 +/- 4.44e-97] + [+/- 2.48e-97]*I
a[3]   =                              [+/- 4.47e-97] + [+/- 4.60e-97]*I
...
a[94]  = [380.00000000000135 ... 63746 +/- 9.24e-70] + [+/- 8.27e-70]*I
a[95]  =                              [+/- 1.37e-69] + [+/- 1.37e-69]*I
a[96]  =                              [+/- 2.93e-69] + [+/- 2.91e-69]*I
a[97]  =                              [+/- 5.81e-69] + [+/- 5.82e-69]*I
a[98]  = [395.9999999999996482...46383 +/- 2.90e-68] + [+/- 1.17e-68]*I
a[99]  =                              [+/- 2.32e-68] + [+/- 2.32e-68]*I
a[100] =                              [+/- 4.95e-68] + [+/- 4.95e-68]*I
```

Example: Laurent series of the Hurwitz zeta function

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

Joint work with I. Blagouchine (`arxiv.org/abs/1804.01679`)

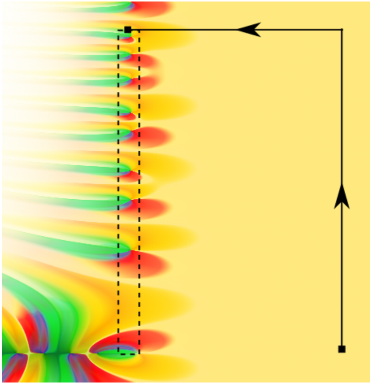
$$\gamma_n(v) = -\frac{\pi}{2(n+1)} \int_{-\infty}^{\infty} \frac{\left(\log\left(v - \frac{1}{2} + ix\right)\right)^{n+1}}{\cosh^2(\pi x)} dx$$

$\gamma_{10^{100}}(1) \in [3.187431418702399279997416469933 \pm 2.89 \cdot 10^{-30}] \cdot 10^{e}$

$e = 2346394292277254080949367838399091160903447689869$
$8373852057791115792156640521582344171254175433483694$

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

Example: zeros of the Riemann zeta function



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

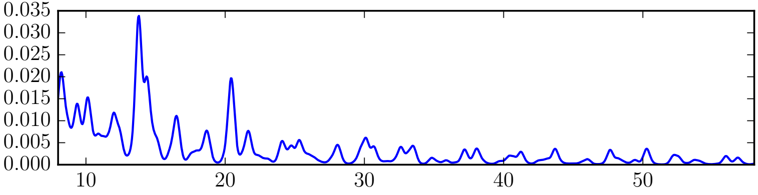
$$N(T) - 1 = \frac{1}{2\pi i}\int_{\gamma} \frac{\zeta'(s)}{\zeta(s)} ds = \frac{\theta(T)}{\pi} +$$

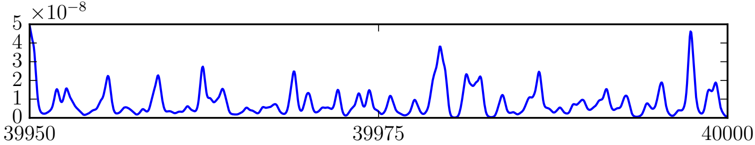
$$\frac{1}{\pi}\operatorname{Im}\left[\int_{1+\varepsilon}^{1+\varepsilon+Ti} \frac{\zeta'(s)}{\zeta(s)} ds + \int_{1+\varepsilon+Ti}^{\frac{1}{2}+Ti} \frac{\zeta'(s)}{\zeta(s)} ds\right]$$

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: $|\zeta(s)|$-integrals (from Harald Helfgott)

$$\int_{-\frac{1}{4}+8i}^{-\frac{1}{4}+40000i} \left| \frac{F_{19}(s+\frac{1}{2})F_{19}(s+1)}{s^2} \right| |ds|, \quad F_N(s) = \zeta(s) \prod_{p \le N} (1-p^{-s})$$





We compute Taylor models $f(s) = g(s) + h(s)i + \varepsilon$ on subintervals $[a, a+0.5]$, and integrate $\sqrt{g^2(s)+h^2(s)}$. Total time: a few hours.

Todo

- Efficient and semi-automatic support for singularities, infinite intervals
  - User may specify scale, e.g. $|f(x)| \leq x^{\alpha} e^{\beta x^{\gamma}}$
  - Dedicated algorithms: Gauss-Jacobi, double exponential...
  - Algorithms for oscillatory integrals
- Symbolic interface
- Let user choose Taylor/Gauss-Legendre quadrature and bounds based on derivatives / complex magnitudes
- Better global adaptivity
- Many applications

Todo

- Efficient and semi-automatic support for singularities, infinite intervals
  - User may specify scale, e.g. $|f(x)| \leq x^{\alpha} e^{\beta x^{\gamma}}$
  - Dedicated algorithms: Gauss-Jacobi, double exponential...
  - Algorithms for oscillatory integrals
- Symbolic interface
- Let user choose Taylor/Gauss-Legendre quadrature and bounds based on derivatives / complex magnitudes
- Better global adaptivity
- Many applications

Thank you!