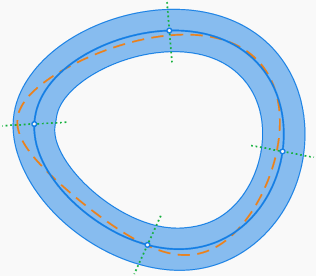
Computing Abelian Integrals in Hilbert's 16th Problem: A Challenge for Validated Numerics

Florent Bréhard, Nicolas Brisebarre, Mioara Joldes, Warwick Tucker

ANR NUSCAP First Meeting

April 27, 2021



Computer-Assisted Proofs and Hilbert's 16th Problem

Computer-Assisted Proofs in Mathematics

Some famous computer-assisted proofs in analysis

- Universality of the Feigenbaum constants
  (O. Lanford, 1982)
- Proof of the Kepler conjecture
  (T. Hales, 1998)
- The Lorenz strange attractor
  (W. Tucker, 2002)
- Chaos in the Kuramoto–Sivashinsky equations
  (D. Wilczak, 2003)
- Equilibria in the 5-body problem
  (A. Albouy and V. Kaloshin, 2021)

Contributions of Computer Science

- Validated Numerics
- Floating-point Arithmetics and Numerical Analysis
- Computer Algebra
- Approximation Theory
- Convex Optimization
- Formal Methods and Formal Proof

Computational Challenge: Abelian Integrals

polynomial or rational functions

$$\int_{\Gamma(h)} P(x,y)dy - Q(x,y)dx$$

oval $H(x,y) = h$ *with* H *polynomial or rational*

Computational Challenge: Abelian Integrals

polynomial or rational functions

$$\int_{\Gamma(h)} P(x,y)\,dy - Q(x,y)\,dx$$

oval $H(x,y) = h$ *with* H *polynomial or rational*

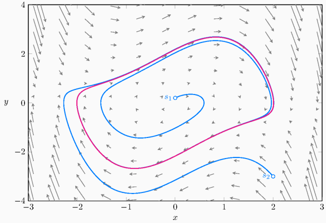
GOALS:

- Very high accuracy
- Efficient algorithm with quasi-linear complexity w.r.t. accuracy digits
- Rigorous and tight error bounds
- Certified calculator checked by a proof assistant

Hilbert's 16th problem (second part)

For a given integer n, what is the maximum number $\mathcal{H}(n)$ of limit cycles a polynomial vector field of degree at most n in the plane can have?

D. Hilbert, International Congress of Mathematicians, Paris, 1900

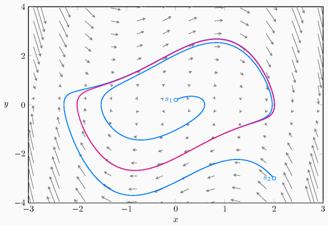


Hilbert's 16th problem (second part)

For a given integer n, what is the maximum number $\mathcal{H}(n)$ of limit cycles a polynomial vector field of degree at most n in the plane can have?

D. Hilbert, International Congress of Mathematicians, Paris, 1900

- 1923: H. Dulac (incorrectly) proved that a *single* polynomial vector field has a finite number of limit cycles
- 1981: Y. S. Il'Yashenko found a major gap in Dulac's proof
- 1991: New proofs of Dulac's result by Y. S. Il'Yashenko and J. Écalle
- But even $\mathcal{H}(2) < \infty$ is open!



Hilbert's 16th Problem

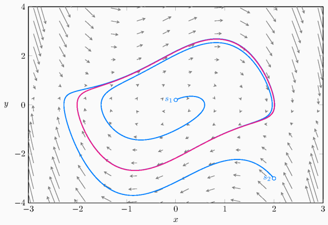
Hilbert's 16th problem (second part)

For a given integer n, what is the maximum number $\mathcal{H}(n)$ of limit cycles a polynomial vector field of degree at most n in the plane can have?

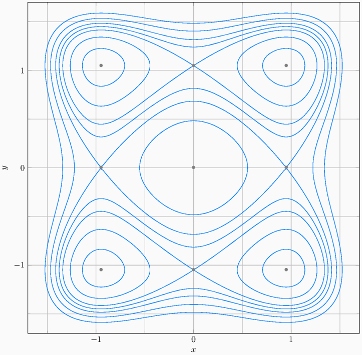
D. Hilbert, International Congress of Mathematicians, Paris, 1900

- 1923: H. Dulac (incorrectly) proved that a *single* polynomial vector field has a finite number of limit cycles
- 1981: Y. S. Il'Yashenko found a major gap in Dulac's proof
- 1991: New proofs of Dulac's result by Y. S. Il'Yashenko and J. Écalle
- But even $\mathcal{H}(2) < \infty$ is open!
- Some lower bounds: $\mathcal{H}(2) \geqslant 4$, $\mathcal{H}(3) \geqslant 13$, $\mathcal{H}(4) \geqslant 28$

$\Rightarrow$ major role of computer-assisted proofs



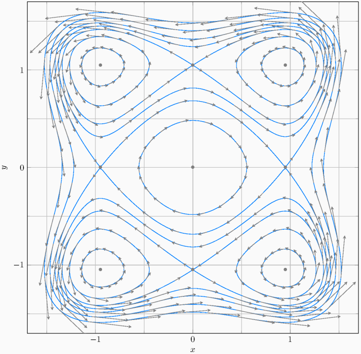
Infinitesimal Hilbert's 16th Problem and Abelian Integrals



$$H(x, y) = \left(x^2 - \frac{9}{10}\right)^2 + \left(y^2 - \frac{11}{10}\right)^2$$

T. Johnson, A quartic system with twenty-six limit cycles, *Experimental Mathematics*, 2011

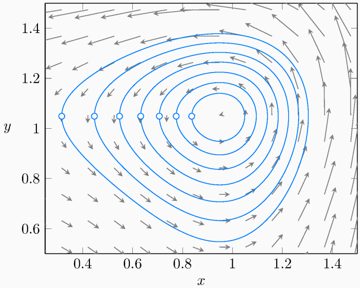
Infinitesimal Hilbert's 16th Problem and Abelian Integrals



$$H(x, y) = \left(x^2 - \frac{9}{10}\right)^2 + \left(y^2 - \frac{11}{10}\right)^2$$

$$\begin{cases} \dot{x} = -H'_y(x, y) = 4y\left(y^2 - \frac{11}{10}\right) \\ \dot{y} = \quad H'_x(x, y) = 4x\left(x^2 - \frac{9}{10}\right) \end{cases}$$

T. Johnson, A quartic system with twenty-six limit cycles, *Experimental Mathematics*, 2011

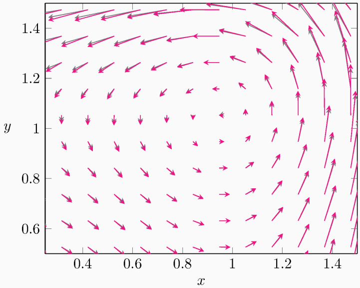


$$H(x, y) = \left(x^2 - \frac{9}{10}\right)^2 + \left(y^2 - \frac{11}{10}\right)^2$$

$$\begin{cases} \dot{x} = 4y\left(y^2 - \frac{11}{10}\right) \\ \dot{y} = 4x\left(x^2 - \frac{9}{10}\right) \end{cases}$$

T. Johnson, A quartic system with twenty-six limit cycles, *Experimental Mathematics*, 2011

Infinitesimal Hilbert's 16th Problem and Abelian Integrals

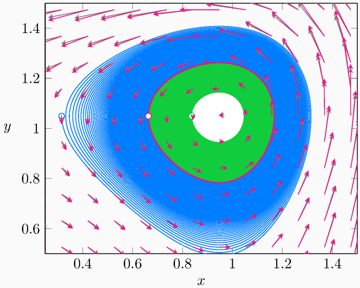


$$H(x, y) = \left(x^2 - \frac{9}{10}\right)^2 + \left(y^2 - \frac{11}{10}\right)^2$$

$$\begin{cases} \dot{x} = 4y\left(y^2 - \frac{11}{10}\right) \\ \dot{y} = 4x\left(x^2 - \frac{9}{10}\right) - 0.4y + 0.46x^2y \end{cases}$$

T. Johnson, A quartic system with twenty-six limit cycles, *Experimental Mathematics*, 2011

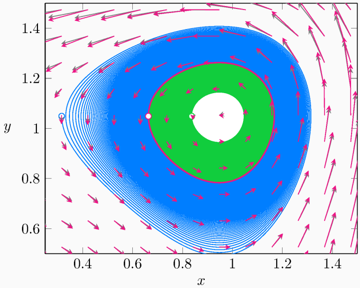
Infinitesimal Hilbert's 16th Problem and Abelian Integrals



$$H(x, y) = \left(x^2 - \frac{9}{10}\right)^2 + \left(y^2 - \frac{11}{10}\right)^2$$

$$\begin{cases} \dot{x} = 4y\left(y^2 - \frac{11}{10}\right) \\ \dot{y} = 4x\left(x^2 - \frac{9}{10}\right) - 0.4y + 0.46x^2y \end{cases}$$

T. Johnson, A quartic system with twenty-six limit cycles, *Experimental Mathematics*, 2011



T. Johnson, A quartic system with twenty-six limit cycles, *Experimental Mathematics*, 2011

Infinitesimal Hilbert's 16th problem

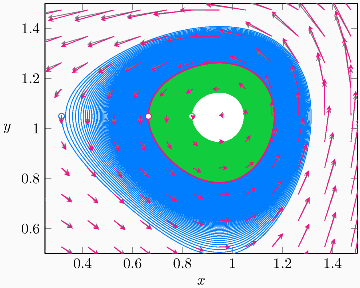
Given n, what is the maximal number $\mathcal{Z}(n)$ of limit cycles a perturbed Hamiltonian vector field of the form:

$$\begin{cases} \dot{x} = -H'_y(x, y) + \varepsilon P(x, y) \\ \dot{y} = \quad H'_x(x, y) + \varepsilon Q(x, y) \end{cases}$$

can have when $\varepsilon \to 0$, with:

- $H(x, y)$ a polynomial potential function of degree $n + 1$
- P, Q polynomial perturbations of degree n

Infinitesimal Hilbert's 16th Problem and Abelian Integrals



T. Johnson, A quartic system with twenty-six limit cycles, *Experimental Mathematics*, 2011

Infinitesimal Hilbert's 16th problem

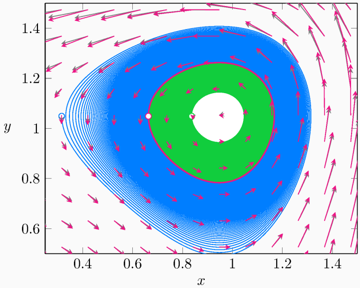
Given n, what is the maximal number $\mathcal{Z}(n)$ of limit cycles a perturbed Hamiltonian vector field of the form:

$$\begin{cases} \dot{x} = -H'_y(x,y) + \varepsilon P(x,y) \\ \dot{y} = \quad H'_x(x,y) + \varepsilon Q(x,y) \end{cases}$$

can have when $\varepsilon \to 0$, with:

- $H(x,y)$ a polynomial potential function of degree $n+1$
- P, Q polynomial perturbations of degree n

- $\mathcal{Z}(n) < \infty$ for all n
- Pessimistic upper bounds



T. Johnson, A quartic system with twenty-six limit cycles, *Experimental Mathematics*, 2011

Infinitesimal Hilbert's 16th problem

Given n, what is the maximal number $\mathcal{Z}(n)$ of limit cycles a perturbed Hamiltonian vector field of the form:

$$\begin{cases} \dot{x} = -H'_y(x,y) + \varepsilon P(x,y) \\ \dot{y} = \quad H'_x(x,y) + \varepsilon Q(x,y) \end{cases}$$

can have when $\varepsilon \to 0$, with:

- $H(x,y)$ a polynomial potential function of degree $n+1$
- P, Q polynomial perturbations of degree n

Poincaré-Pontryagin Theorem

Limit cycles when $\varepsilon \to 0$ are given by the zeros of the Abelian integral:

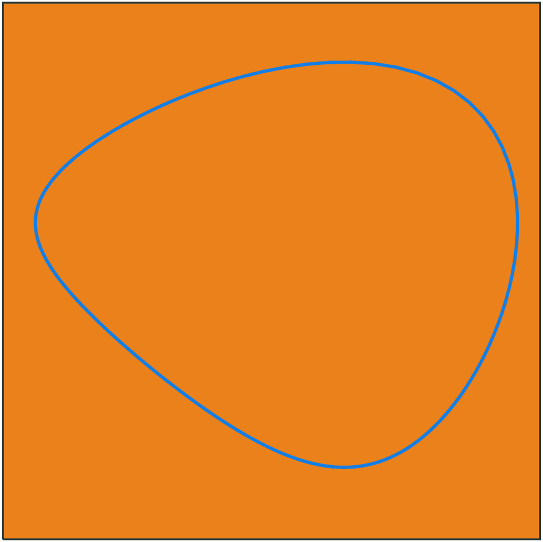
$$\mathcal{I}(h) = \int_{\Gamma(h)} P(x,y)dy - Q(x,y)dx$$

T. Johnson and W. Tucker's Rigorous Subdivision Algorithm

Green's theorem on $\mathcal{I}(h)$:

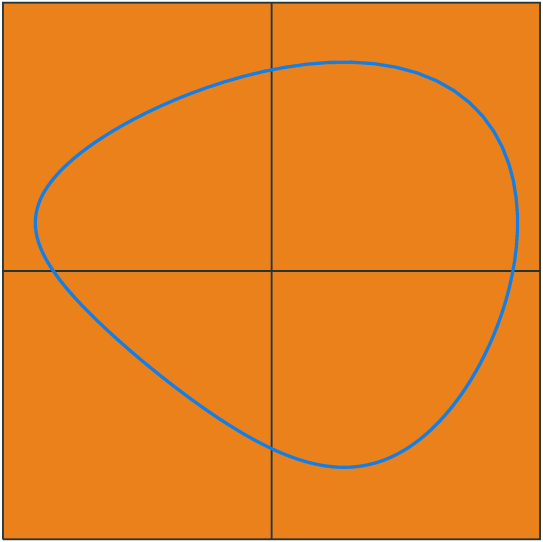
$$\int_{\Gamma(h)} P(x,y)dy - Q(x,y)dx = \int_{D(h)} \left(P'_x(x,y) + Q'_y(x,y)\right) dxdy$$

T. Johnson and W. Tucker’s Rigorous Subdivision Algorithm



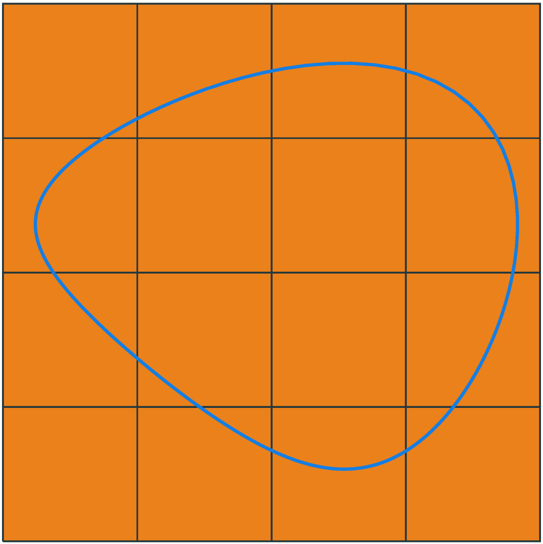
T. Johnson and W. Tucker. *On a computer-aided approach to the computation of Abelian integrals* (2011)

T. Johnson and W. Tucker's Rigorous Subdivision Algorithm



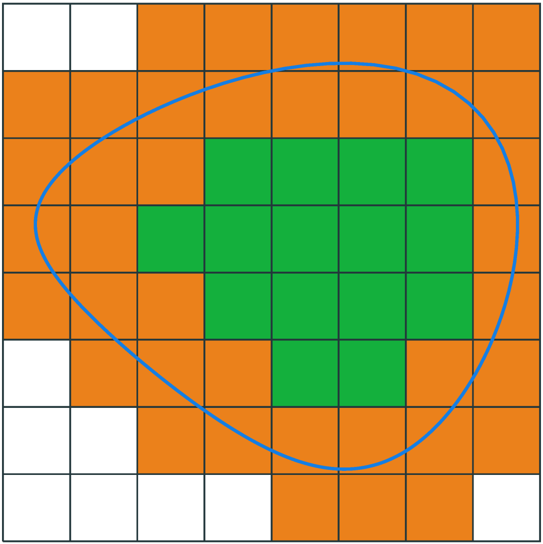
T. Johnson and W. Tucker. *On a computer-aided approach to the computation of Abelian integrals* (2011)

T. Johnson and W. Tucker's Rigorous Subdivision Algorithm



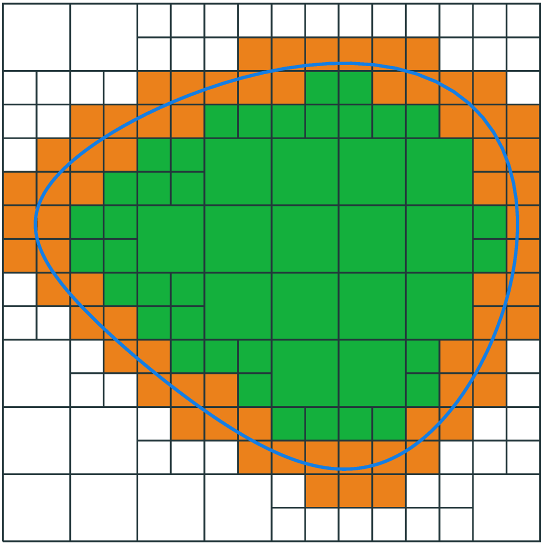
T. Johnson and W. Tucker. *On a computer-aided approach to the computation of Abelian integrals* (2011)

T. Johnson and W. Tucker's Rigorous Subdivision Algorithm



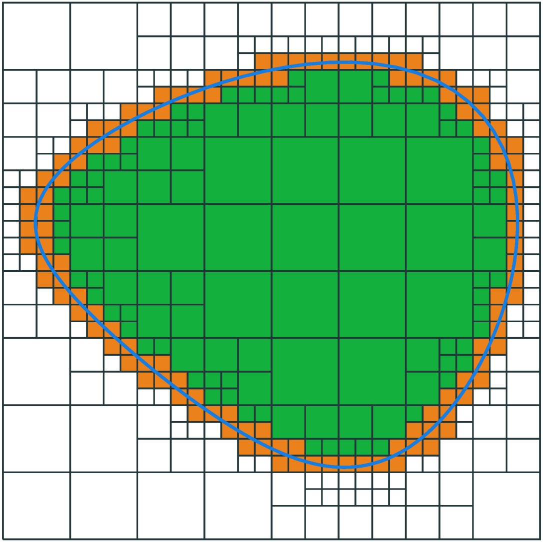
T. Johnson and W. Tucker. *On a computer-aided approach to the computation of Abelian integrals* (2011)

T. Johnson and W. Tucker’s Rigorous Subdivision Algorithm



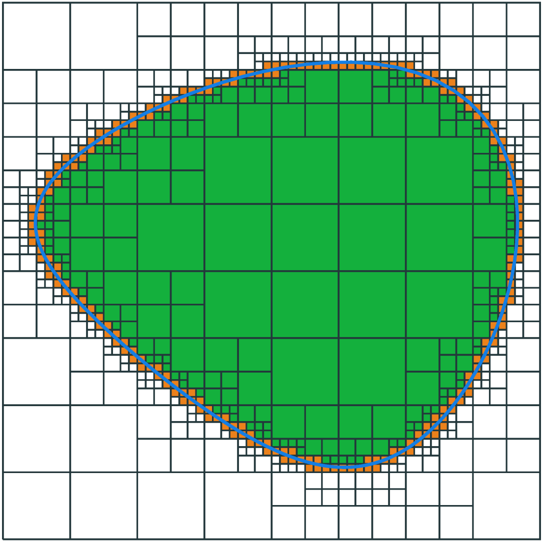
T. Johnson and W. Tucker. *On a computer-aided approach to the computation of Abelian integrals* (2011)

T. Johnson and W. Tucker's Rigorous Subdivision Algorithm



T. Johnson and W. Tucker. *On a computer-aided approach to the computation of Abelian integrals* (2011)

T. Johnson and W. Tucker's Rigorous Subdivision Algorithm



T. Johnson and W. Tucker. *On a computer-aided approach to the computation of Abelian integrals* (2011)

Different Approches to the Computation of Abelian Integrals

- Green theorem + subdivision [*JohnsonTucker2011*]
- Rigorous integration along the vector field [*CAPD, . . .*]
- D-finite (Picard-Fuchs) equation [*LairezMezzarobbaSafey2019*]

Different Approches to the Computation of Abelian Integrals

- Green theorem + subdivision [*JohnsonTucker2011*]
- Rigorous integration along the vector field [*CAPD, ...*]
- D-finite (Picard-Fuchs) equation [*LairezMezzarobbaSafey2019*]

Our approach:

- ∘ Validated algorithm in polynomial complexity w.r.t number of accuracy digits

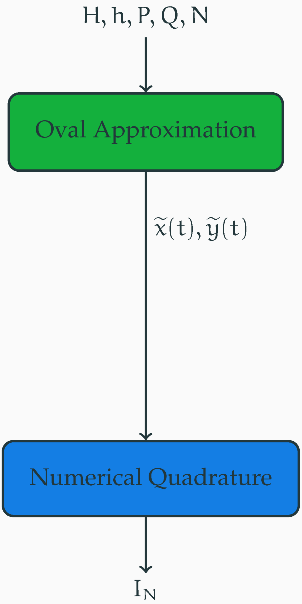
⇒ Higher-order method with Fourier series

- ∘ Formalizable in Coq in the short term

⇒ A posteriori validation approach

⇒ Keep the validation method as minimalist as possible

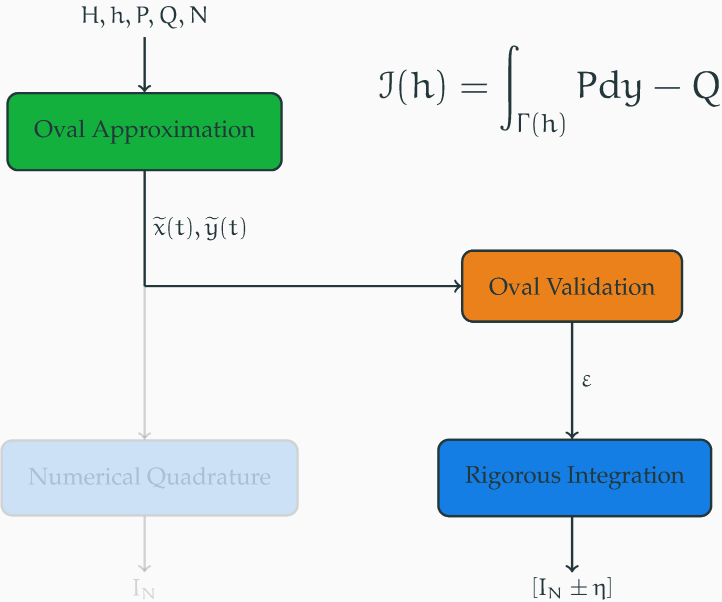
Rigorous Computation of Abelian Integrals



$$\mathcal{I}(h) = \int_{\Gamma(h)} P dy - Q dx$$

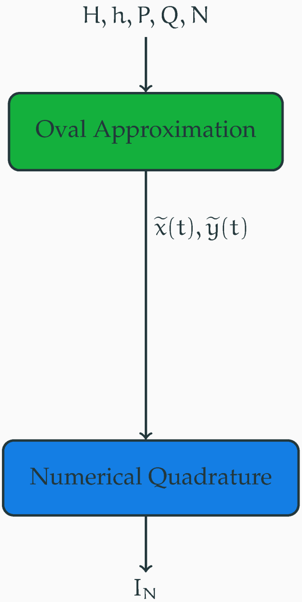
Rigorous Computation of Abelian Integrals

$$\mathcal{I}(h) = \int_{\Gamma(h)} P dy - Q dx$$

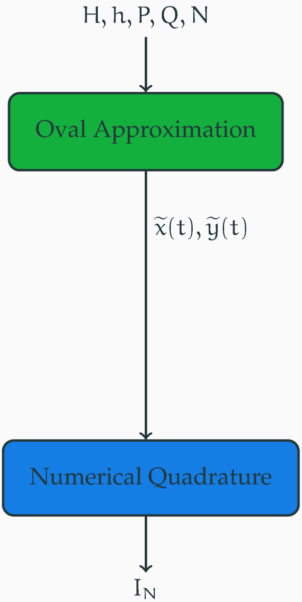


Oval Approximation and Numerical Quadrature

Numerical Computation of Abelian Integrals



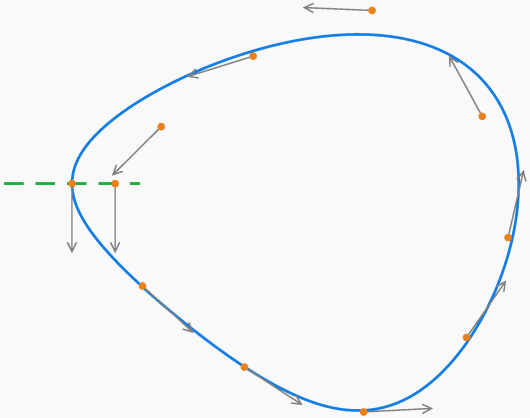
$$\mathcal{I}(h) = \int_{\Gamma(h)} P dy - Q dx$$



$$\mathcal{I}(h) = \int_{\Gamma(h)} P dy - Q dx$$

1. Initial guess
2. Refinement to target accuracy using Newton's method
3. Trapezoidal quadrature rule

Initial Guess for the Oval



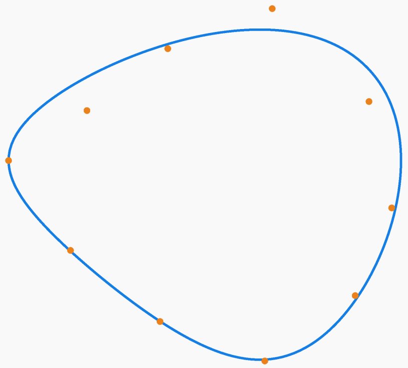
Parameterization of $\Gamma(h)$ following the rotated gradient:

$$\begin{cases} \dot{x} = -H'_y(x,y) \\ \dot{y} = H'_x(x,y) \end{cases}$$

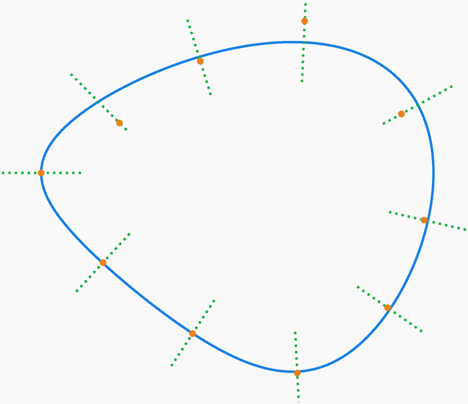
Requierements:

- iterative scheme for moderate accuracy (e.g, Runge-Kutta methods)
- detection of the first return onto the transversal

Refining the Approximation with Newton’s Method

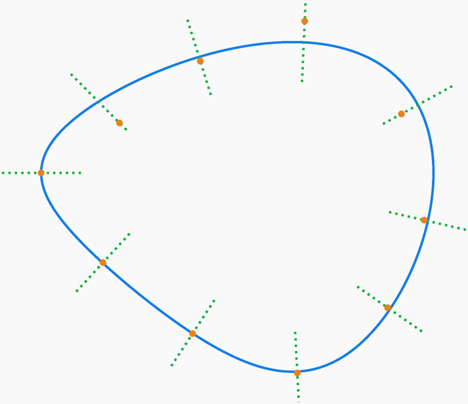


Refining the Approximation with Newton's Method



- Project each point w.r.t. the gradient of H, with Newton's method

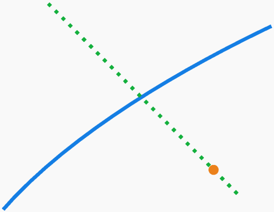
Refining the Approximation with Newton's Method



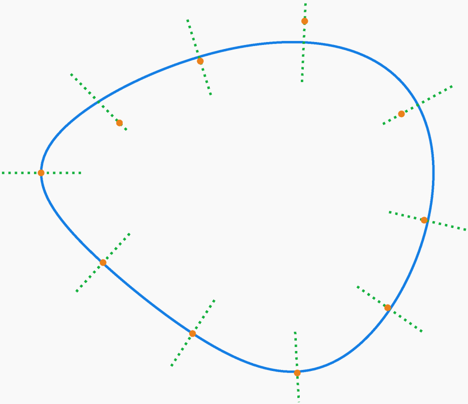
- Project each point w.r.t. the gradient of H, with Newton's method

$$u = H'_x(x, y) \qquad v = H'_y(x, y)$$

SOLVE $\qquad H(x + su, y + sv) - h = 0$



Refining the Approximation with Newton's Method

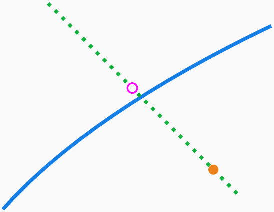


- Project each point w.r.t. the gradient of H, with Newton's method

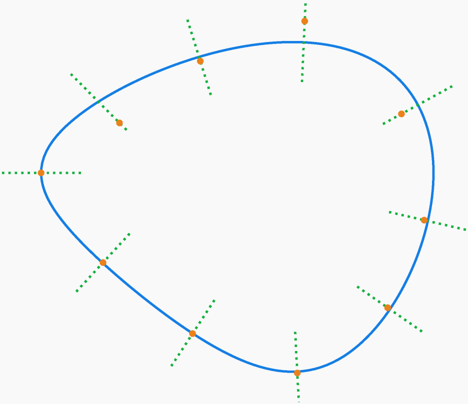
$$u = H'_x(x, y) \qquad v = H'_y(x, y)$$

SOLVE $\qquad H(x + su, y + sv) - h = 0$

$$\mathcal{N}(s) = s - \frac{H(x + su, y + sv) - h}{uH'_x(x + su, y + sv) + vH'_y(x + su, y + sv)}$$



Refining the Approximation with Newton's Method

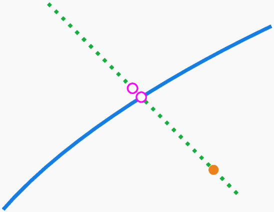


- Project each point w.r.t. the gradient of H, with Newton's method

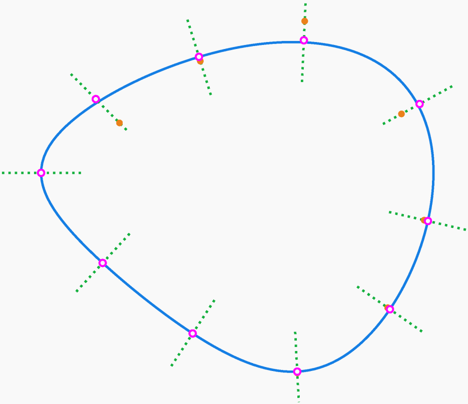
$$u = H'_x(x, y) \qquad v = H'_y(x, y)$$

SOLVE $\quad H(x + su, y + sv) - h = 0$

$$\mathcal{N}(s) = s - \frac{H(x + su, y + sv) - h}{uH'_x(x + su, y + sv) + vH'_y(x + su, y + sv)}$$



Refining the Approximation with Newton's Method

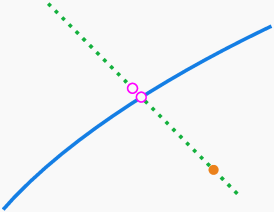


- Project each point w.r.t. the gradient of H, with Newton's method
- Parallelize Newton-Raphson iterations for each point

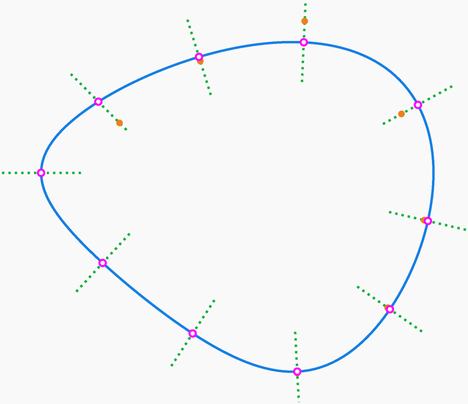
$$u = H'_x(x, y) \qquad v = H'_y(x, y)$$

SOLVE $\quad H(x + su, y + sv) - h = 0$

$$\mathcal{N}(s) = s - \frac{H(x + su, y + sv) - h}{uH'_x(x + su, y + sv) + vH'_y(x + su, y + sv)}$$



Refining the Approximation with Newton's Method

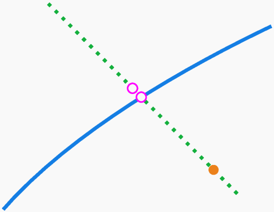


- Project each point w.r.t. the gradient of H, with Newton's method
- Parallelize Newton-Raphson iterations for each point
- Very fast (quadratic) convergence

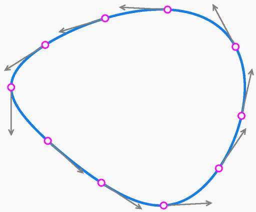
$$u = H'_x(x, y) \qquad v = H'_y(x, y)$$

SOLVE $\quad H(x + su, y + sv) - h = 0$

$$\mathcal{N}(s) = s - \frac{H(x + su, y + sv) - h}{uH'_x(x + su, y + sv) + vH'_y(x + su, y + sv)}$$

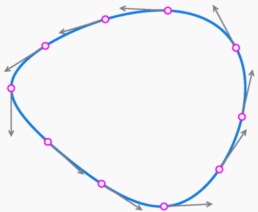


Numerical Integration via Trapezoidal Quadrature Rule



$$I_N = \frac{2\pi}{N}\sum_{j=1}^{N} P(x_j, y_j)y_j' - Q(x_j, y_j)x_j'$$

Numerical Integration via Trapezoidal Quadrature Rule



$$I_N = \frac{2\pi}{N} \sum_{j=1}^{N} P(x_j, y_j) y_j' - Q(x_j, y_j) x_j'$$

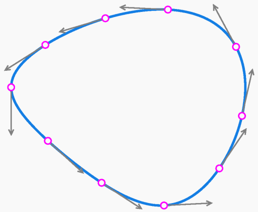
Theorem (Euler-Maclaurin)

The trapezoidal quadrature rule:

$$\int_0^{2\pi} f \approx \frac{2\pi}{N} \left(\frac{f(t_0)}{2} + f(t_1) + \cdots + f(t_j) + \cdots + f(t_{N-1}) + \frac{f(t_N)}{2} \right)$$

with $t_j = \frac{2j\pi}{N}$

Numerical Integration via Trapezoidal Quadrature Rule



$$I_N = \frac{2\pi}{N} \sum_{j=1}^{N} P(x_j, y_j) y_j' - Q(x_j, y_j) x_j'$$

Theorem (Euler-Maclaurin)

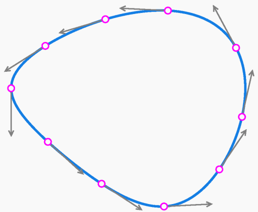
The trapezoidal quadrature rule:

$$\int_0^{2\pi} f \approx \frac{2\pi}{N} \left(\frac{f(t_0)}{2} + f(t_1) + \cdots + f(t_j) + \cdots + f(t_{N-1}) + \frac{f(t_N)}{2} \right)$$

with $t_j = \frac{2j\pi}{N}$

- converges in $O(1/N^2)$ for f analytic over $[0, 2\pi]$

Numerical Integration via Trapezoidal Quadrature Rule



$$I_N = \frac{2\pi}{N} \sum_{j=1}^{N} P(x_j, y_j) y_j' - Q(x_j, y_j) x_j'$$

Theorem (Euler-Maclaurin)

The trapezoidal quadrature rule:

$$\int_0^{2\pi} f \approx \frac{2\pi}{N} \left(\frac{f(t_0)}{2} + f(t_1) + \cdots + f(t_j) + \cdots + f(t_{N-1}) + \frac{f(t_N)}{2} \right)$$

with $t_j = \frac{2j\pi}{N}$

- converges in $O(1/N^2)$ for f analytic over $[0, 2\pi]$
- converges in $O(e^{-\rho N})$ for f analytic and periodic over $[0, 2\pi]$

Numerical Example — What's wrong?

Example (Johnson-Tucker)

$$H(x,y) = \left(x^2 - \frac{9}{10}\right)^2 + \left(y^2 - \frac{11}{10}\right)^2 \qquad h = \frac{16}{25}$$

$$P(x,y) = xy^3 \qquad Q(x,y) = x^2y^2$$

$$\Rightarrow \quad \mathcal{I}(h) = 1.5752210992246893\ldots$$

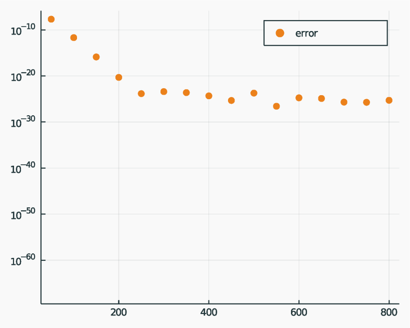
Numerical Example — What's wrong?

Example (Johnson-Tucker)

$$H(x,y) = \left(x^2 - \frac{9}{10}\right)^2 + \left(y^2 - \frac{11}{10}\right)^2 \qquad h = \frac{16}{25}$$

$$P(x,y) = xy^3 \qquad Q(x,y) = x^2y^2$$

$$\Rightarrow \quad \mathcal{I}(h) = 1.5752210992246893\ldots$$



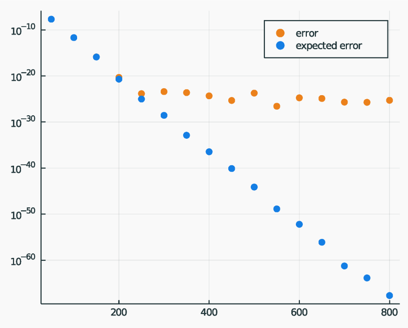
Numerical Example — What's wrong?

Example (Johnson-Tucker)

$$H(x,y) = \left(x^2 - \frac{9}{10}\right)^2 + \left(y^2 - \frac{11}{10}\right)^2 \qquad h = \frac{16}{25}$$

$$P(x,y) = xy^3 \qquad Q(x,y) = x^2y^2$$

$$\Rightarrow \quad \mathcal{I}(h) = 1.5752210992246893\ldots$$



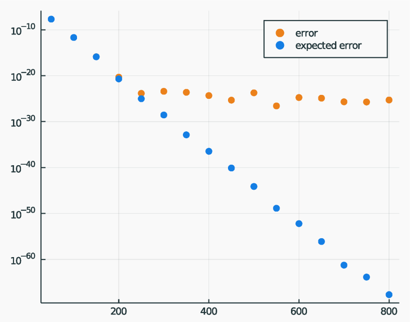
Numerical Example — What's wrong?

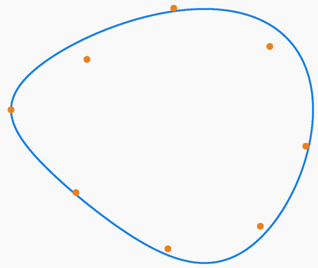
Example (Johnson-Tucker)

$$H(x,y) = \left(x^2 - \frac{9}{10}\right)^2 + \left(y^2 - \frac{11}{10}\right)^2 \qquad h = \frac{16}{25}$$

$$P(x,y) = xy^3 \qquad Q(x,y) = x^2y^2$$

$\Rightarrow \quad \mathcal{I}(h) = 1.5752210992246893\ldots$





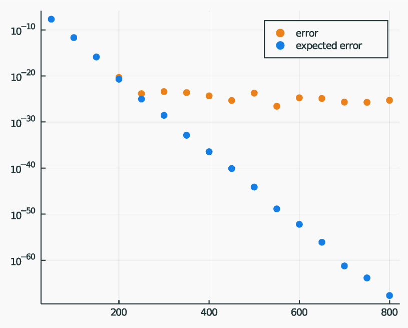
Numerical Example — What's wrong?

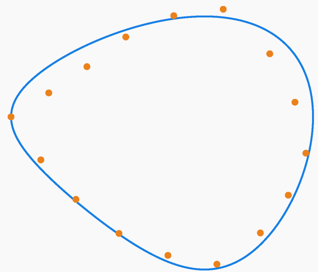
Example (Johnson-Tucker)

$$H(x,y) = \left(x^2 - \frac{9}{10}\right)^2 + \left(y^2 - \frac{11}{10}\right)^2 \qquad h = \frac{16}{25}$$

$$P(x,y) = xy^3 \qquad Q(x,y) = x^2y^2$$

$$\Rightarrow \quad \mathcal{I}(h) = 1.5752210992246893\ldots$$





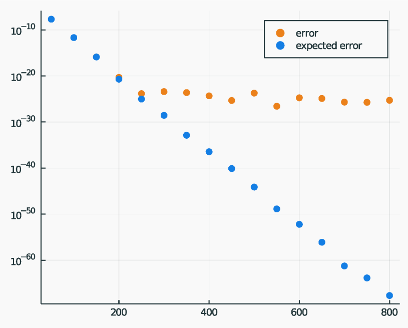
Numerical Example — What's wrong?

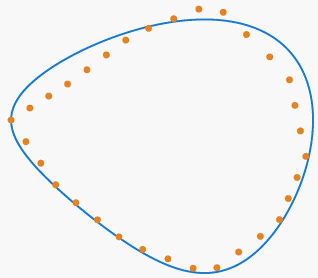
Example (Johnson-Tucker)

$$H(x,y) = \left(x^2 - \frac{9}{10}\right)^2 + \left(y^2 - \frac{11}{10}\right)^2 \qquad h = \frac{16}{25}$$

$$P(x,y) = xy^3 \qquad Q(x,y) = x^2y^2$$

$$\Rightarrow \quad \mathcal{I}(h) = 1.5752210992246893\ldots$$





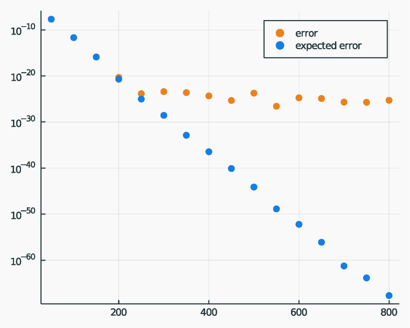
Numerical Example — What's wrong?

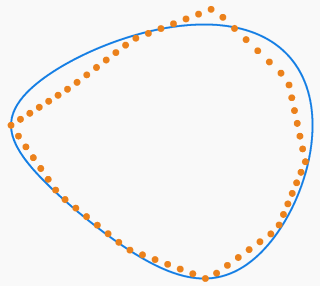
Example (Johnson-Tucker)

$$H(x,y) = \left(x^2 - \frac{9}{10}\right)^2 + \left(y^2 - \frac{11}{10}\right)^2 \qquad h = \frac{16}{25}$$

$$P(x,y) = xy^3 \qquad Q(x,y) = x^2y^2$$

$$\Rightarrow \quad \mathcal{I}(h) = 1.5752210992246893\ldots$$





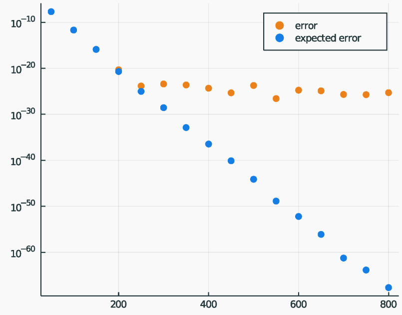
Numerical Example — What's wrong?

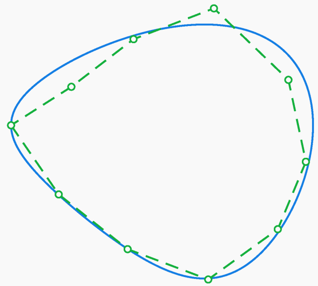
Example (Johnson-Tucker)

$$H(x,y) = \left(x^2 - \frac{9}{10}\right)^2 + \left(y^2 - \frac{11}{10}\right)^2 \qquad h = \frac{16}{25}$$

$$P(x,y) = xy^3 \qquad Q(x,y) = x^2y^2$$

$$\Rightarrow \quad \mathcal{I}(h) = 1.5752210992246893\ldots$$





Fourier Approximation Theory

$$f(t) = a_0 + a_1 \cos t + b_1 \sin t + \cdots + a_n \cos nt + b_n \sin nt + \ldots$$

Fourier Approximation Theory

$$f(t) = \underbrace{a_0 + a_1 \cos t + b_1 \sin t + \cdots + a_n \cos nt + b_n \sin nt}_{f_n(t)} + \dots$$

Fourier Approximation Theory

$$f(t) = \underbrace{a_0 + a_1 \cos t + b_1 \sin t + \cdots + a_n \cos nt + b_n \sin nt}_{f_n(t)} + \ldots$$

$$\langle f, g \rangle = \int_0^{2\pi} f(t)g(t)dt$$

$$\langle \cos nt, \cos mt \rangle = \begin{cases} 2\pi & \text{if } n = m = 0 \\ \pi & \text{if } n = m > 0 \\ 0 & \text{if } n \neq m \end{cases}$$

$$\langle \cos nt, \sin mt \rangle = 0$$

$$\langle \sin nt, \sin mt \rangle = \begin{cases} \pi & \text{if } n = m > 0 \\ 0 & \text{if } n \neq m \end{cases}$$

Fourier Approximation Theory

$$f(t) = \underbrace{a_0 + a_1 \cos t + b_1 \sin t + \cdots + a_n \cos nt + b_n \sin nt}_{f_n(t)} + \ldots$$

$$\langle f, g \rangle = \int_0^{2\pi} f(t)g(t)dt \qquad \langle \cos nt, \cos mt \rangle = \begin{cases} 2\pi & \text{if } n = m = 0 \\ \pi & \text{if } n = m > 0 \\ 0 & \text{if } n \neq m \end{cases}$$

$$\langle \cos nt, \sin mt \rangle = 0 \qquad \langle \sin nt, \sin mt \rangle = \begin{cases} \pi & \text{if } n = m > 0 \\ 0 & \text{if } n \neq m \end{cases}$$

$$a_0 = \frac{1}{2\pi}\int_0^{2\pi} f(t)dt \qquad a_n = \frac{1}{\pi}\int_0^{2\pi} f(t)\cos nt dt \qquad b_n = \frac{1}{\pi}\int_0^{2\pi} f(t)\sin nt dt$$

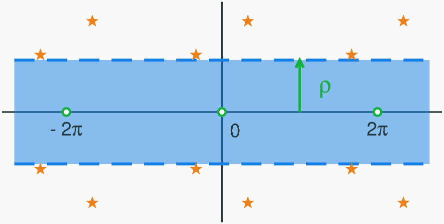
Fourier Approximation Theory

$$f(t) = \underbrace{a_0 + a_1 \cos t + b_1 \sin t + \dots + a_n \cos nt + b_n \sin nt}_{f_n(t)} + \dots$$

$$\langle f, g \rangle = \int_0^{2\pi} f(t) g(t) dt \qquad \langle \cos nt, \cos mt \rangle = \begin{cases} 2\pi & \text{if } n = m = 0 \\ \pi & \text{if } n = m > 0 \\ 0 & \text{if } n \neq m \end{cases}$$

$$\langle \cos nt, \sin mt \rangle = 0 \qquad \langle \sin nt, \sin mt \rangle = \begin{cases} \pi & \text{if } n = m > 0 \\ 0 & \text{if } n \neq m \end{cases}$$

$$a_0 = \frac{1}{2\pi} \int_0^{2\pi} f(t) dt \qquad a_n = \frac{1}{\pi} \int_0^{2\pi} f(t) \cos nt dt \qquad b_n = \frac{1}{\pi} \int_0^{2\pi} f(t) \sin nt dt$$

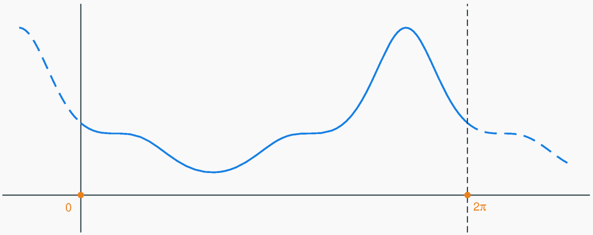


If f periodic analytic without singularities in $\{z \in \mathbb{C},\ |\mathfrak{Im}(z)| \leqslant \rho\}$,

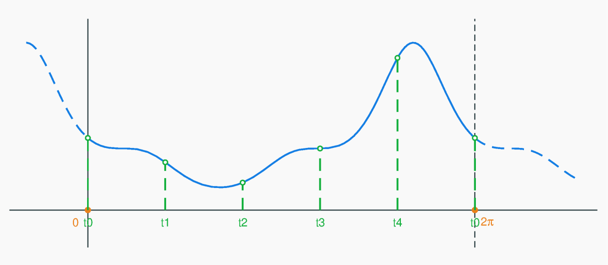
$$|a_n|,\ |b_n| = O(e^{-\rho n})$$

$$\|f_n - f\|_\infty = O(e^{-\rho n})$$

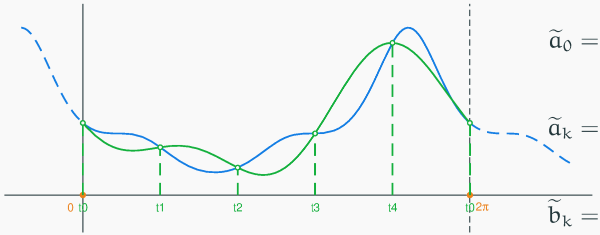
Trigonometric Interpolation



Trigonometric Interpolation



Trigonometric Interpolation



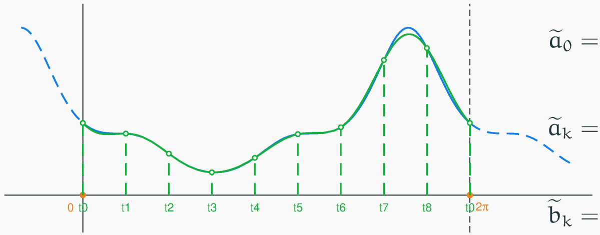
$$\widetilde{a}_0 = \frac{1}{2n+1} \sum_{j=0}^{2n} f(t_j)$$

$$\widetilde{a}_k = \frac{2}{2n+1} \sum_{j=0}^{2n} f(t_j) \cos kt_j$$

$$\widetilde{b}_k = \frac{2}{2n+1} \sum_{j=0}^{2n} f(t_j) \sin kt_j$$

$$p_n(t) = \widetilde{a}_0 + \widetilde{a}_1 \cos t + \widetilde{b}_1 \sin t + \cdots + \widetilde{a}_n \cos nt + \widetilde{b}_n \sin nt \approx f_n(t)$$

Trigonometric Interpolation



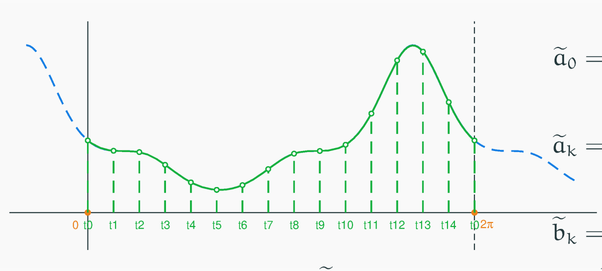
$$\widetilde{a}_0 = \frac{1}{2n+1} \sum_{j=0}^{2n} f(t_j)$$

$$\widetilde{a}_k = \frac{2}{2n+1} \sum_{j=0}^{2n} f(t_j) \cos kt_j$$

$$\widetilde{b}_k = \frac{2}{2n+1} \sum_{j=0}^{2n} f(t_j) \sin kt_j$$

$$p_n(t) = \widetilde{a}_0 + \widetilde{a}_1 \cos t + \widetilde{b}_1 \sin t + \cdots + \widetilde{a}_n \cos nt + \widetilde{b}_n \sin nt \approx f_n(t)$$

Trigonometric Interpolation



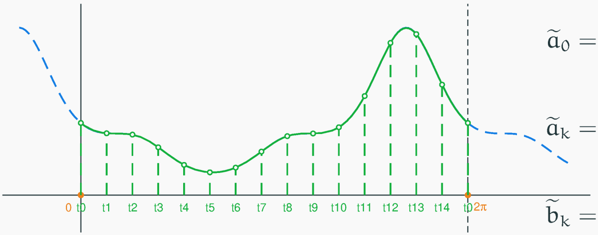
$$\widetilde{a}_0 = \frac{1}{2n+1} \sum_{j=0}^{2n} f(t_j)$$

$$\widetilde{a}_k = \frac{2}{2n+1} \sum_{j=0}^{2n} f(t_j) \cos kt_j$$

$$\widetilde{b}_k = \frac{2}{2n+1} \sum_{j=0}^{2n} f(t_j) \sin kt_j$$

$$p_n(t) = \widetilde{a}_0 + \widetilde{a}_1 \cos t + \widetilde{b}_1 \sin t + \cdots + \widetilde{a}_n \cos nt + \widetilde{b}_n \sin nt \approx f_n(t)$$

Trigonometric Interpolation



$$\tilde{a}_0 = \frac{1}{2n+1} \sum_{j=0}^{2n} f(t_j)$$

$$\tilde{a}_k = \frac{2}{2n+1} \sum_{j=0}^{2n} f(t_j) \cos k t_j$$

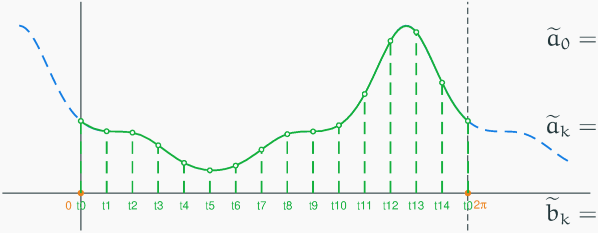
$$\tilde{b}_k = \frac{2}{2n+1} \sum_{j=0}^{2n} f(t_j) \sin k t_j$$

$$p_n(t) = \tilde{a}_0 + \tilde{a}_1 \cos t + \tilde{b}_1 \sin t + \cdots + \tilde{a}_n \cos nt + \tilde{b}_n \sin nt \approx f_n(t)$$

If f periodic analytic without singularities in $\{z \in \mathbb{C},\ |\mathfrak{Im}(z)| \leqslant \rho\}$, then:

$$\|p_n - f\|_\infty = O(e^{-\rho n})$$

Trigonometric Interpolation



$$\widetilde{a}_0 = \frac{1}{2n+1}\sum_{j=0}^{2n} f(t_j)$$

$$\widetilde{a}_k = \frac{2}{2n+1}\sum_{j=0}^{2n} f(t_j)\cos kt_j$$

$$\widetilde{b}_k = \frac{2}{2n+1}\sum_{j=0}^{2n} f(t_j)\sin kt_j$$

$$p_n(t) = \widetilde{a}_0 + \widetilde{a}_1\cos t + \widetilde{b}_1 \sin t + \cdots + \widetilde{a}_n \cos nt + \widetilde{b}_n \sin nt \approx f_n(t)$$

If f periodic analytic without singularities in $\{z \in \mathbb{C},\ |\mathfrak{Im}(z)| \leqslant \rho\}$, then:

$$\|p_n - f\|_\infty = O(e^{-\rho n})$$

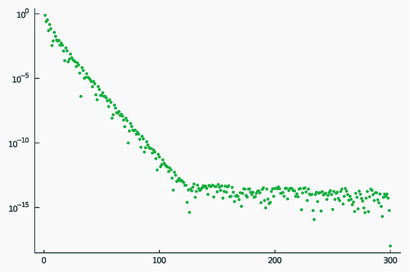
FAST FOURIER TRANSFORM

- fast interpolation
- fast evaluation
- fast multiplication

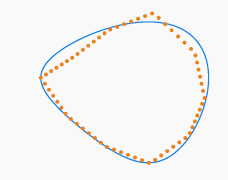
$$\begin{pmatrix} f(t_0) \\ \vdots \\ f(t_{2n}) \end{pmatrix} \underset{\text{IFFT}_n}{\overset{\text{FFT}_n}{\rightleftarrows}} \begin{pmatrix} \widetilde{a}_0 \\ \vdots \\ \widetilde{b}_n \end{pmatrix}$$

$O(n \log n)$ arith. op.

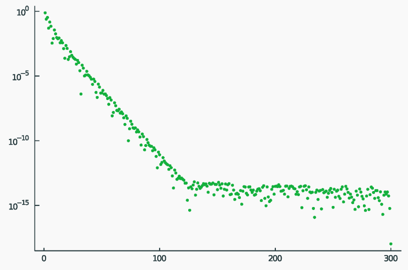
Denoising the Initial Guess



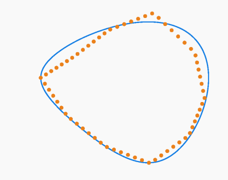
Initial guess

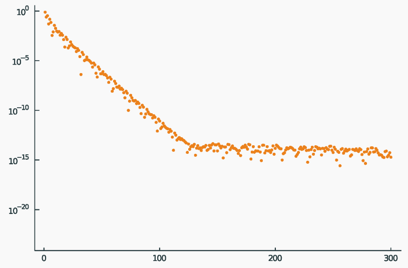


Denoising the Initial Guess

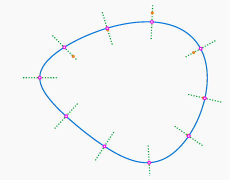


Initial guess



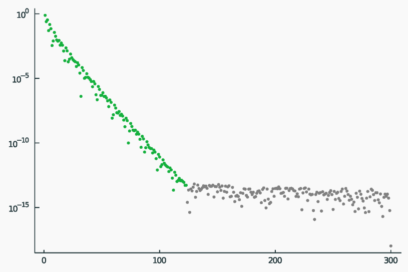


After projection

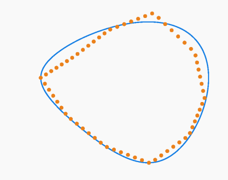


× stagnation to initial guess' accuracy

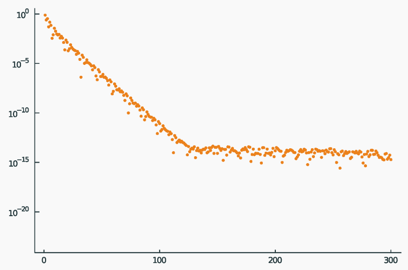
Denoising the Initial Guess



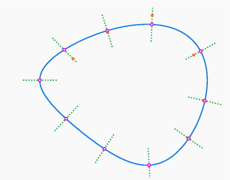
Initial guess



⇒ truncate coefficient list

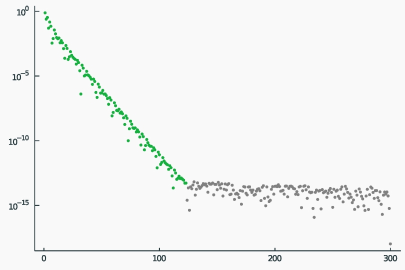


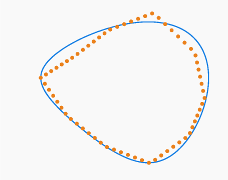
After projection



× stagnation to initial guess' accuracy

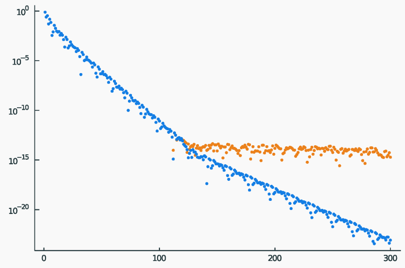
Denoising the Initial Guess

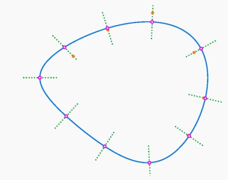




Initial guess

⇒ truncate coefficient list





After projection

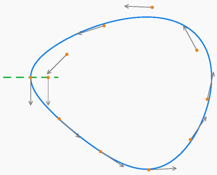
× stagnation to initial guess' accuracy

⇒ exponential convergence

Numerical Evaluation of the Abelian Integral — Algorithm

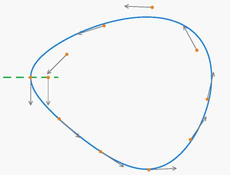
INPUT: H, h, number of points N

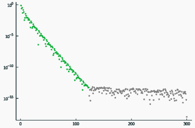
OUTPUT: Numerical approximation I_N for $\mathcal{I}(h)$



INPUT: H, h, number of points N
OUTPUT: Numerical approximation I_N for $\mathcal{I}(h)$

1. Sample N initial points (x_j°, y_j°) using an iterative scheme with moderate precision





INPUT: H, h, number of points N
OUTPUT: Numerical approximation I_N for $\mathcal{I}(h)$

1. Sample N initial points (x_j°, y_j°) using an iterative scheme with moderate precision
2. Denoise via FFT/IFFT + coefficient list truncation

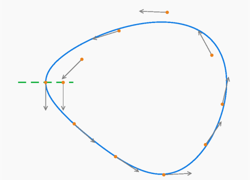
Numerical Evaluation of the Abelian Integral — Algorithm

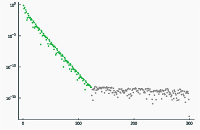
INPUT: H, h, number of points N
OUTPUT: Numerical approximation I_N for $\mathcal{I}(h)$

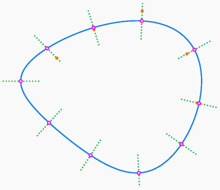
1. Sample N initial points (x_j°, y_j°) using an iterative scheme with moderate precision

2. Denoise via FFT/IFFT + coefficient list truncation

3. For each point, apply $\log N$ Newton-Raphson iterations to project onto $\Gamma(h)$





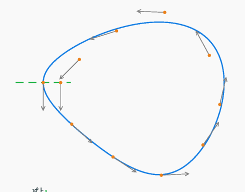


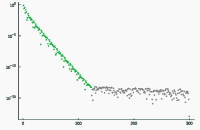
Numerical Evaluation of the Abelian Integral — Algorithm

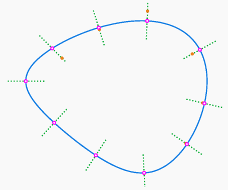
INPUT: H, h, number of points N

OUTPUT: Numerical approximation I_N for $\mathcal{I}(h)$

1. Sample N initial points (x_j°, y_j°) using an iterative scheme with moderate precision
2. Denoise via FFT/IFFT + coefficient list truncation
3. For each point, apply $\log N$ Newton-Raphson iterations to project onto $\Gamma(h)$
4. Evaluate I_N with trapezoidal quadrature rule







Numerical Evaluation of the Abelian Integral — Algorithm

INPUT: H, h, number of points N

OUTPUT: Numerical approximation I_N for $\mathcal{I}(h)$

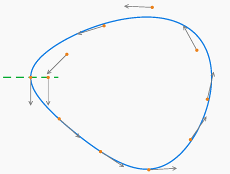
1. Sample N initial points (x_j°, y_j°) using an iterative scheme with moderate precision

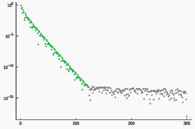
2. Denoise via FFT/IFFT + coefficient list truncation

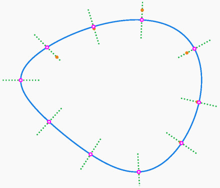
3. For each point, apply $\log N$ Newton-Raphson iterations to project onto $\Gamma(h)$

4. Evaluate I_N with trapezoidal quadrature rule

> Complexity: $O(N \log N)$ floating-point op.







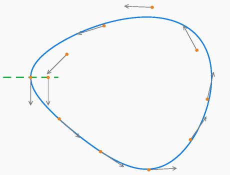
Numerical Evaluation of the Abelian Integral — Algorithm

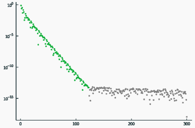
INPUT: H, h, number of points N
OUTPUT: Numerical approximation I_N for $\mathcal{I}(h)$

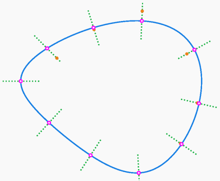
1. Sample N initial points (x_j°, y_j°) using an iterative scheme with moderate precision
2. Denoise via FFT/IFFT + coefficient list truncation
3. For each point, apply $\log N$ Newton-Raphson iterations to project onto $\Gamma(h)$
4. Evaluate I_N with trapezoidal quadrature rule

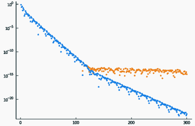
> Complexity: $O(N \log N)$ floating-point op.

> Convergence: $|I_N - \mathcal{I}(h)| = O(\kappa^{-N})$ for a $\kappa > 1$



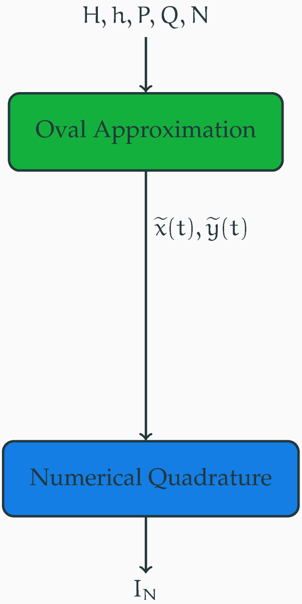






Oval Validation and Rigorous Integration

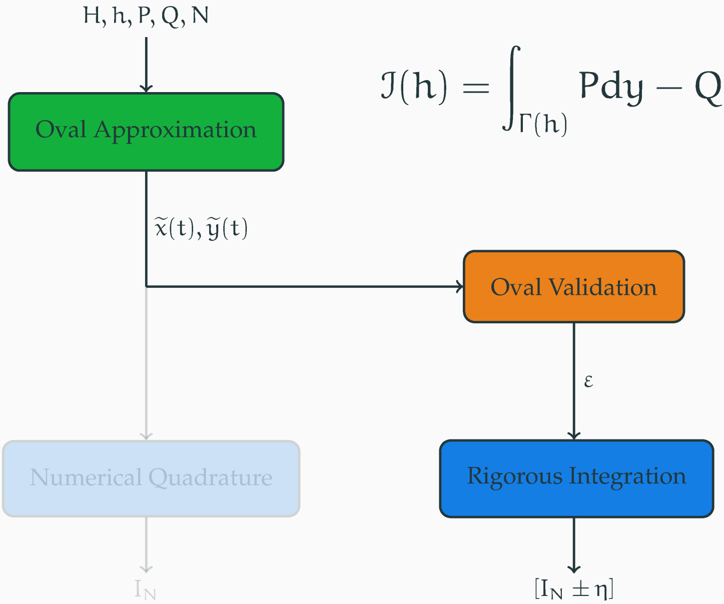
Rigorous Computation of Abelian Integrals



$$\mathcal{I}(h) = \int_{\Gamma(h)} P dy - Q dx$$

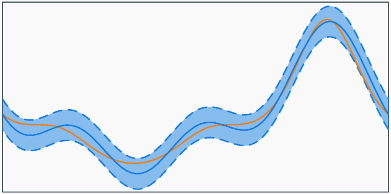
Rigorous Computation of Abelian Integrals

$$\mathcal{I}(h) = \int_{\Gamma(h)} P dy - Q dx$$



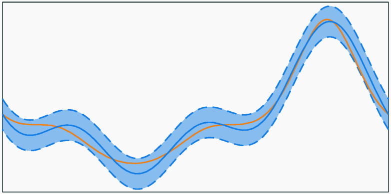
Rigorous Trigonometric Approximations

A pair (p, ε) is a Rigorous Trigonometric Approximation (RTA) for f if $\|p - f\|_\infty \leqslant \varepsilon$.



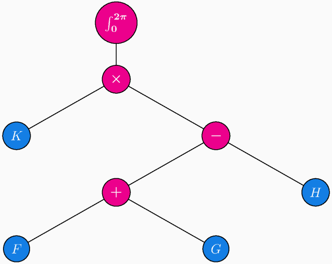
Rigorous Trigonometric Approximations

> A pair (p, ε) is a **Rigorous Trigonometric Approximation (RTA)** for f if $\|p - f\|_\infty \leqslant \varepsilon$.



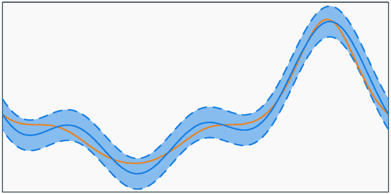
Elementary operations

- $(p, \varepsilon) + (q, \eta) = (p + q, \varepsilon + \eta)$
- $(p, \varepsilon) - (q, \eta) = (p - q, \varepsilon + \eta)$
- $(p, \varepsilon) \cdot (q, \eta) = (pq, \varepsilon\|q\|_\infty + \eta\|p\|_\infty + \varepsilon\eta)$
- $\int_0^{2\pi}(p, \varepsilon) = [a_0 - 2\pi\varepsilon, a_0 + 2\pi\varepsilon]$
- $\ldots$



$$\int_0^{2\pi} k\,(f + g - h)$$

Rigorous Trigonometric Approximations

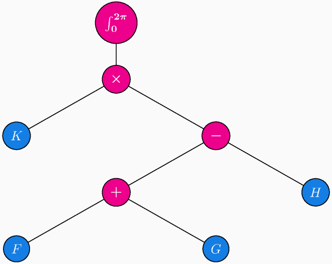


> A pair (p, ε) is a Rigorous Trigonometric Approximation (RTA) for f if $\|p - f\|_\infty \leqslant \varepsilon$.

Elementary operations

- $(p, \varepsilon) + (q, \eta) = (p + q, \varepsilon + \eta)$
- $(p, \varepsilon) - (q, \eta) = (p - q, \varepsilon + \eta)$
- $(p, \varepsilon) \cdot (q, \eta) = (pq, \varepsilon\|q\|_\infty + \eta\|p\|_\infty + \varepsilon\eta)$
- $\int_0^{2\pi}(p, \varepsilon) = [a_0 - 2\pi\varepsilon, a_0 + 2\pi\varepsilon]$
- $\ldots$

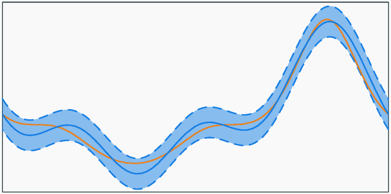
× no rigorous fast multiplication yet!



$$\int_0^{2\pi} k\,(f + g - h)$$

Rigorous Trigonometric Approximations

> A pair (p, ε) is a Rigorous Trigonometric Approximation (RTA) for f if $\|p - f\|_\infty \leqslant \varepsilon$.

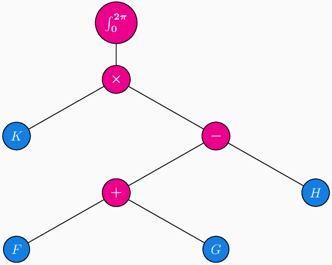


Elementary operations

- $(p, \varepsilon) + (q, \eta) = (p + q, \varepsilon + \eta)$
- $(p, \varepsilon) - (q, \eta) = (p - q, \varepsilon + \eta)$
- $(p, \varepsilon) \cdot (q, \eta) = (pq, \varepsilon\|q\|_\infty + \eta\|p\|_\infty + \varepsilon\eta)$
- $\int_0^{2\pi}(p, \varepsilon) = [a_0 - 2\pi\varepsilon, a_0 + 2\pi\varepsilon]$
- ...

× no rigorous fast multiplication yet!

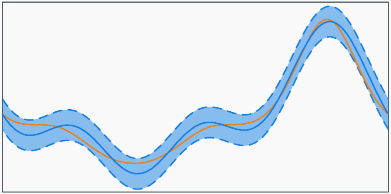
× no direct formulas for more complex operations, e.g. division
⇒ fixed-point (Newton-like) validation methods



$$\int_0^{2\pi} k\,(f + g - h)$$

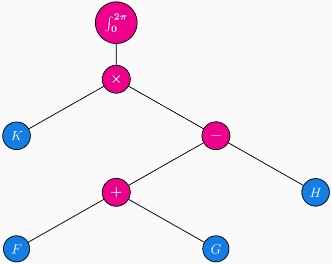
Rigorous Trigonometric Approximations

> A pair (p, ε) is a Rigorous Trigonometric Approximation (RTA) for f if $\|p - f\|_\infty \leqslant \varepsilon$.



Elementary operations

- $(p, \varepsilon) + (q, \eta) = (p + q, \varepsilon + \eta)$
- $(p, \varepsilon) - (q, \eta) = (p - q, \varepsilon + \eta)$
- $(p, \varepsilon) \cdot (q, \eta) = (pq, \varepsilon\|q\|_\infty + \eta\|p\|_\infty + \varepsilon\eta)$
- $\int_0^{2\pi}(p, \varepsilon) = [a_0 - 2\pi\varepsilon, a_0 + 2\pi\varepsilon]$
- $\ldots$



$$\int_0^{2\pi} k\,(f + g - h)$$

× no rigorous fast multiplication yet!

× no direct formulas for more complex operations, e.g. division
⇒ fixed-point (Newton-like) validation methods

× not implemented in Coq yet

Principle of Newton-like Validation Methods

- Candidate approximate root $\tilde{x}$ of $\mathcal{F}$
- Equation $\mathcal{F}\{x\} = 0 \quad \Rightarrow \quad$ Fixed-point equation $\mathcal{N}\{x\} = 0$ with

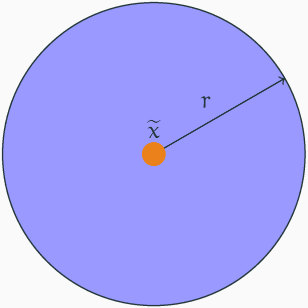
$$\mathcal{N}\{x\} = x - \mathcal{A}\{\mathcal{F}\{x\}\} \qquad \text{where} \qquad \mathcal{A} \approx (D\mathcal{F}_{\tilde{x}})^{-1}$$

- Well suited for a posteriori validation of function space problems

Principle of Newton-like Validation Methods

- Candidate approximate root $\widetilde{x}$ of $\mathcal{F}$
- Equation $\mathcal{F}\{x\} = 0 \quad \Rightarrow \quad$ Fixed-point equation $\mathcal{N}\{x\} = 0$ with

$$\mathcal{N}\{x\} = x - \mathcal{A}\{\mathcal{F}\{x\}\} \qquad \text{where} \qquad \mathcal{A} \approx (D\mathcal{F}_{\widetilde{x}})^{-1}$$

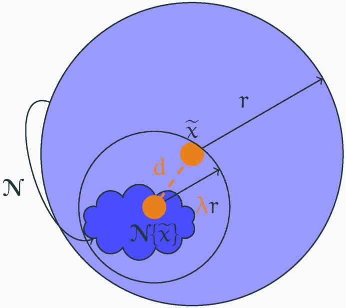


- Well suited for a posteriori validation of function space problems

Principle of Newton-like Validation Methods

- Candidate approximate root $\widetilde{x}$ of $\mathcal{F}$
- Equation $\mathcal{F}\{x\} = 0 \quad \Rightarrow \quad$ Fixed-point equation $\mathcal{N}\{x\} = 0$ with

$$\mathcal{N}\{x\} = x - \mathcal{A}\{\mathcal{F}\{x\}\} \qquad \text{where} \qquad \mathcal{A} \approx (D\mathcal{F}_{\widetilde{x}})^{-1}$$

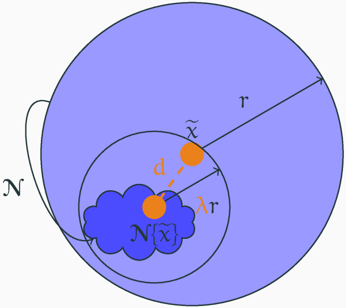


- Well suited for a posteriori validation of function space problems

Principle of Newton-like Validation Methods

- Candidate approximate root $\widetilde{x}$ of $\mathcal{F}$
- Equation $\mathcal{F}\{x\} = 0$ $\Rightarrow$ Fixed-point equation $\mathcal{N}\{x\} = 0$ with

$$\mathcal{N}\{x\} = x - \mathcal{A}\{\mathcal{F}\{x\}\} \qquad \text{where} \qquad \mathcal{A} \approx (D\mathcal{F}_{\widetilde{x}})^{-1}$$



Banach Fixed-Point Theorem. If

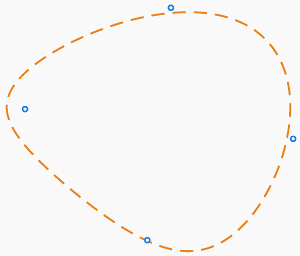
- $\|\mathcal{N}\{\widetilde{x}\} - \widetilde{x}\| \leqslant d$,
- $\mathcal{N}$ λ-contracting over $B(\widetilde{x}, r)$ with $\lambda < 1$,
- $d + \lambda r \leqslant r$,

Then $\mathcal{N}$ has a unique fixed point x^* in $B(\widetilde{x}, r)$, and

$$\frac{d}{1+\lambda} \leqslant \|\widetilde{x} - x^*\| \leqslant \frac{d}{1-\lambda}$$

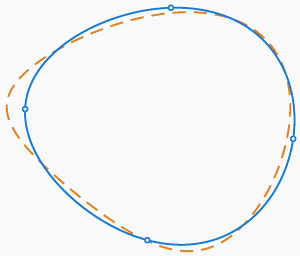
- Well suited for a posteriori validation of function space problems

Validation of an Oval Approximation



Validation of an Oval Approximation

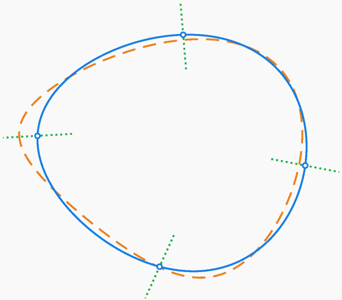
- Oval Approx. $t \mapsto (\widetilde{x}(t), \widetilde{y}(t))$



Validation of an Oval Approximation

- Oval Approx. $t \mapsto (\tilde{x}(t), \tilde{y}(t))$
- Transversal vector field:

$$(u(t), v(t)) \approx \frac{\nabla H(\tilde{x}(t), \tilde{y}(t))}{\|\nabla H(\tilde{x}(t), \tilde{y}(t))\|}$$



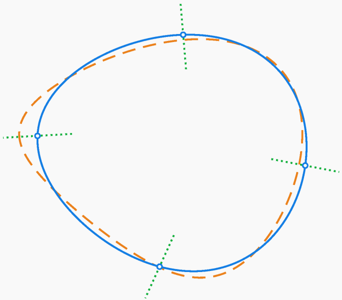
Validation of an Oval Approximation

- Oval Approx. $t \mapsto (\widetilde{x}(t), \widetilde{y}(t))$
- Transversal vector field:

$$(u(t), v(t)) \approx \frac{\nabla H(\widetilde{x}(t), \widetilde{y}(t))}{\|\nabla H(\widetilde{x}(t), \widetilde{y}(t))\|}$$

- Find $s^* : t \mapsto s^*(t)$ s.t.

$$H\left(\widetilde{x}(t) + s^*(t)u(t),\ \ \widetilde{y}(t) + s^*(t)v(t)\right) - h = 0$$



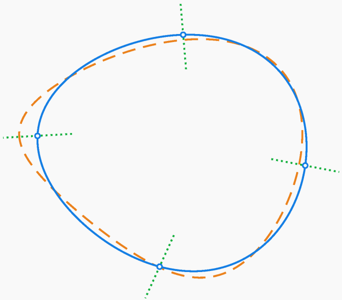
Validation of an Oval Approximation

- Oval Approx. $t \mapsto (\tilde{x}(t), \tilde{y}(t))$
- Transversal vector field:

$$(u(t), v(t)) \approx \frac{\nabla H(\tilde{x}(t), \tilde{y}(t))}{\|\nabla H(\tilde{x}(t), \tilde{y}(t))\|}$$

- Find $s^* : t \mapsto s^*(t)$ s.t.

$$H\left(\tilde{x}(t) + s^*(t)u(t),\ \ \tilde{y}(t) + s^*(t)v(t)\right) - h = 0$$



- Newton-like operator:

$$\mathcal{N}\{s\}(t) = s(t) - \psi(t)\left(H\left(\tilde{x}(t) + s(t)u(t),\ \ \tilde{y}(t) + s(t)v(t)\right) - h\right)$$

$$\approx\ \ 1/\left(u(t)\, H'_x(\tilde{x}(t), \tilde{y}(t)) + v(t)\, H'_y(\tilde{x}(t), \tilde{y}(t))\right)$$

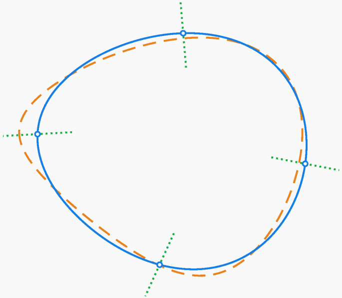
Validation of an Oval Approximation

- Oval Approx. $t \mapsto (\widetilde{x}(t), \widetilde{y}(t))$
- Transversal vector field:

$$(u(t), v(t)) \approx \frac{\nabla H(\widetilde{x}(t), \widetilde{y}(t))}{\|\nabla H(\widetilde{x}(t), \widetilde{y}(t))\|}$$

- Find $s^* : t \mapsto s^*(t)$ s.t.

$$H\left(\widetilde{x}(t) + s^*(t)u(t),\ \ \widetilde{y}(t) + s^*(t)v(t)\right) - h = 0$$



- Newton-like operator:

$$\mathcal{N}\{s\}(t) = s(t) - \psi(t)\left(H\left(\widetilde{x}(t) + s(t)u(t),\ \ \widetilde{y}(t) + s(t)v(t)\right) - h\right)$$

$$\approx \quad 1/\left(u(t)\, H'_x(\widetilde{x}(t), \widetilde{y}(t)) + v(t)\, H'_y(\widetilde{x}(t), \widetilde{y}(t))\right)$$

- Bound $\|D\mathcal{N}\| \leqslant \lambda$ in a neighboorhood of $(\widetilde{x}(t), \widetilde{y}(t))$, i.e. $s(t) = 0$, to get:

$$\frac{\|\mathcal{N}\{0\} - 0\|}{1 + \lambda} \leqslant \|s^*\| \leqslant \frac{\|\mathcal{N}\{0\} - 0\|}{1 - \lambda}$$

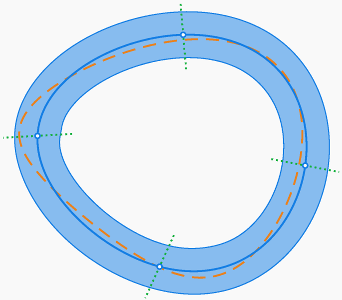
Validation of an Oval Approximation

- Oval Approx. $t \mapsto (\tilde{x}(t), \tilde{y}(t))$
- Transversal vector field:

$$(u(t), v(t)) \approx \frac{\nabla H(\tilde{x}(t), \tilde{y}(t))}{\|\nabla H(\tilde{x}(t), \tilde{y}(t))\|}$$

- Find $s^* : t \mapsto s^*(t)$ s.t.

$$H\Big(\tilde{x}(t) + s^*(t)u(t),\ \ \tilde{y}(t) + s^*(t)v(t)\Big) - h = 0$$



- Newton-like operator:

$$\mathcal{N}\{s\}(t) = s(t) - \psi(t)\left(H\Big(\tilde{x}(t) + s(t)u(t),\ \ \tilde{y}(t) + s(t)v(t)\Big) - h\right)$$

$$\approx\ 1/\Big(u(t)\, H'_x(\tilde{x}(t), \tilde{y}(t)) + v(t)\, H'_y(\tilde{x}(t), \tilde{y}(t))\Big)$$

- Bound $\|D\mathcal{N}\| \leqslant \lambda$ in a neighboorhood of $(\tilde{x}(t), \tilde{y}(t))$, i.e. $s(t) = 0$, to get:

$$\frac{\|\mathcal{N}\{0\} - 0\|}{1 + \lambda} \leqslant \|s^*\| \leqslant \frac{\|\mathcal{N}\{0\} - 0\|}{1 - \lambda}$$

Validation of an Oval Approximation — Algorithm

INPUT: H, h, approximate points (x_j, y_j) for $j = 0..2N$

OUTPUT: Degree N RTAs $\mathbf{x} = (\tilde{x}, \varepsilon)$, $\mathbf{y} = (\tilde{y}, \varepsilon)$

Validation of an Oval Approximation — Algorithm

INPUT: H, h, approximate points (x_j, y_j) for $j = 0..2N$

OUTPUT: Degree N RTAs $\boldsymbol{x} = (\tilde{x}, \varepsilon), \boldsymbol{y} = (\tilde{y}, \varepsilon)$

Oracles (by interpolation):

1. $\tilde{x}(t), \tilde{y}(t)$ interpolating the $(x_j), (y_j)$
2. $u(t), v(t) \approx \nabla H(\tilde{x}(t), \tilde{y}(t)) \,/\, \|\nabla H(\tilde{x}(t), \tilde{y}(t))\|$
3. $\psi(t) \approx 1 \,/\, \big(u(t)\, H'_x(\tilde{x}(t), \tilde{y}(t)) + v(t)\, H'_y(\tilde{x}(t), \tilde{y}(t))\big)$

Validation of an Oval Approximation — Algorithm

INPUT: H, h, approximate points (x_j, y_j) for $j = 0..2N$

OUTPUT: Degree N RTAs $\mathbf{x} = (\tilde{x}, \varepsilon), \mathbf{y} = (\tilde{y}, \varepsilon)$

Oracles (by interpolation):

1. $\tilde{x}(t), \tilde{y}(t)$ interpolating the $(x_j), (y_j)$
2. $u(t), v(t) \approx \nabla H(\tilde{x}(t), \tilde{y}(t)) \;/\; \|\nabla H(\tilde{x}(t), \tilde{y}(t))\|$
3. $\psi(t) \approx 1 \;/\; \big(u(t)\, H'_x(\tilde{x}(t), \tilde{y}(t)) + v(t)\, H'_y(\tilde{x}(t), \tilde{y}(t))\big)$

Rigorous bounds:

4. $d \geqslant \|\mathcal{N}\{0\} - 0\| = \|\psi\, (H(\tilde{x}, \tilde{y}) - h)\|$

INPUT: H, h, approximate points (x_j, y_j) for $j = 0..2N$

OUTPUT: Degree N RTAs $\mathbf{x} = (\tilde{x}, \varepsilon), \mathbf{y} = (\tilde{y}, \varepsilon)$

Oracles (by interpolation):

1. $\tilde{x}(t), \tilde{y}(t)$ interpolating the $(x_j), (y_j)$
2. $u(t), v(t) \approx \nabla H(\tilde{x}(t), \tilde{y}(t)) \,/\, \|\nabla H(\tilde{x}(t), \tilde{y}(t))\|$
3. $\psi(t) \approx 1 \,/\, \big(u(t)\, H'_x(\tilde{x}(t), \tilde{y}(t)) + v(t)\, H'_y(\tilde{x}(t), \tilde{y}(t))\big)$

Rigorous bounds:

4. $d \geqslant \|\mathcal{N}\{0\} - 0\| = \|\psi\,(H(\tilde{x}, \tilde{y}) - h)\|$
5. $\lambda_0 \geqslant \|D_0\mathcal{N}\| = \|1 - \psi\,\big(uH'_x(\tilde{x}, \tilde{y}) + vH'_y(\tilde{x}, \tilde{y})\big)\,\|$

Validation of an Oval Approximation — Algorithm

INPUT: H, h, approximate points (x_j, y_j) for $j = 0..2N$

OUTPUT: Degree N RTAs $\mathbf{x} = (\tilde{x}, \varepsilon)$, $\mathbf{y} = (\tilde{y}, \varepsilon)$

Oracles (by interpolation):

1. $\tilde{x}(t), \tilde{y}(t)$ interpolating the $(x_j), (y_j)$
2. $u(t), v(t) \approx \nabla H(\tilde{x}(t), \tilde{y}(t)) \,/\, \|\nabla H(\tilde{x}(t), \tilde{y}(t))\|$
3. $\psi(t) \approx 1 \,/\, \big(u(t)\, H'_x(\tilde{x}(t), \tilde{y}(t)) + v(t)\, H'_y(\tilde{x}(t), \tilde{y}(t))\big)$

Rigorous bounds:

4. $d \geqslant \|\mathcal{N}\{0\} - 0\| = \|\psi\,(H(\tilde{x}, \tilde{y}) - h)\|$
5. $\lambda_0 \geqslant \|D_0 \mathcal{N}\| = \|1 - \psi\,\big(uH'_x(\tilde{x}, \tilde{y}) + vH'_y(\tilde{x}, \tilde{y})\big)\,\|$
6. $\lambda_1(r) \geqslant \|D^2_{(0,r)} \mathcal{N}\| = \|\psi\,\big(u^2 H''_{xx}(\dots) + 2uvH''_{xy}(\dots) + v^2 H''_{yy}(\dots)\big)\,\|$

Validation of an Oval Approximation — Algorithm

INPUT: H, h, approximate points (x_j, y_j) for $j = 0..2N$

OUTPUT: Degree N RTAs $\mathbf{x} = (\tilde{x}, \varepsilon)$, $\mathbf{y} = (\tilde{y}, \varepsilon)$

Oracles (by interpolation):

1. $\tilde{x}(t), \tilde{y}(t)$ interpolating the $(x_j), (y_j)$
2. $u(t), v(t) \approx \nabla H(\tilde{x}(t), \tilde{y}(t)) \;/\; \|\nabla H(\tilde{x}(t), \tilde{y}(t))\|$
3. $\psi(t) \approx 1 \;/\; \big(u(t)\, H'_x(\tilde{x}(t), \tilde{y}(t)) + v(t)\, H'_y(\tilde{x}(t), \tilde{y}(t))\big)$

Rigorous bounds:

4. $d \geqslant \|\mathcal{N}\{0\} - 0\| = \|\psi\,(H(\tilde{x}, \tilde{y}) - h)\|$
5. $\lambda_0 \geqslant \|D_0 \mathcal{N}\| = \|1 - \psi\,\big(uH'_x(\tilde{x}, \tilde{y}) + vH'_y(\tilde{x}, \tilde{y})\big)\,\|$
6. $\lambda_1(r) \geqslant \|D^2_{(0,r)} \mathcal{N}\| = \|\psi\,\big(u^2 H''_{xx}(\dots) + 2uvH''_{xy}(\dots) + v^2 H''_{yy}(\dots)\big)\,\|$
7. Find $r > 0$ s.t. $\lambda(r) := \lambda_0 + r\lambda_1(r)$ satisfies $d + r\lambda(r) < r$

Validation of an Oval Approximation — Algorithm

INPUT: H, h, approximate points (x_j, y_j) for $j = 0..2N$

OUTPUT: Degree N RTAs $\mathbf{x} = (\tilde{x}, \varepsilon), \mathbf{y} = (\tilde{y}, \varepsilon)$

Oracles (by interpolation):

1. $\tilde{x}(t), \tilde{y}(t)$ interpolating the $(x_j), (y_j)$
2. $u(t), v(t) \approx \nabla H(\tilde{x}(t), \tilde{y}(t)) \;/\; \|\nabla H(\tilde{x}(t), \tilde{y}(t))\|$
3. $\psi(t) \approx 1 \;/\; \big(u(t)\, H'_x(\tilde{x}(t), \tilde{y}(t)) + v(t)\, H'_y(\tilde{x}(t), \tilde{y}(t))\big)$

Rigorous bounds:

4. $d \geqslant \|\mathcal{N}\{0\} - 0\| = \|\psi\,(H(\tilde{x}, \tilde{y}) - h)\|$
5. $\lambda_0 \geqslant \|D_0\mathcal{N}\| = \|1 - \psi\,\big(uH'_x(\tilde{x}, \tilde{y}) + vH'_y(\tilde{x}, \tilde{y})\big)\,\|$
6. $\lambda_1(r) \geqslant \|D^2_{(0,r)}\mathcal{N}\| = \|\psi\,\big(u^2 H''_{xx}(\dots) + 2uvH''_{xy}(\dots) + v^2 H''_{yy}(\dots)\big)\,\|$
7. Find $r > 0$ s.t. $\lambda(r) := \lambda_0 + r\lambda_1(r)$ satisfies $d + r\lambda(r) < r$
8. Compute $\varepsilon = d/(1 - \lambda(r))$ and return RTAs $\mathbf{x} = (\tilde{x}, \varepsilon), \mathbf{y} = (\tilde{y}, \varepsilon)$

Validation of an Oval Approximation — Algorithm

INPUT: H, h, approximate points (x_j, y_j) for $j = 0..2N$

OUTPUT: Degree N RTAs $\mathbf{x} = (\tilde{x}, \varepsilon), \mathbf{y} = (\tilde{y}, \varepsilon)$

Oracles (by interpolation):

1. $\tilde{x}(t), \tilde{y}(t)$ interpolating the $(x_j), (y_j)$
2. $u(t), v(t) \approx \nabla H(\tilde{x}(t), \tilde{y}(t)) \,/\, \|\nabla H(\tilde{x}(t), \tilde{y}(t))\|$
3. $\psi(t) \approx 1 \,/\, \big(u(t)\, H'_x(\tilde{x}(t), \tilde{y}(t)) + v(t)\, H'_y(\tilde{x}(t), \tilde{y}(t))\big)$

Rigorous bounds:

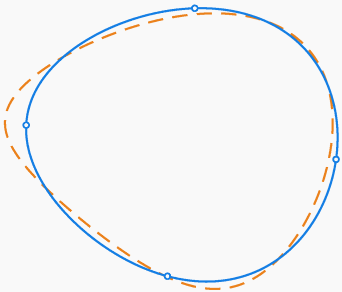
4. $d \geqslant \|\mathcal{N}\{0\} - 0\| = \|\psi\, (H(\tilde{x}, \tilde{y}) - h)\|$
5. $\lambda_0 \geqslant \|D_0 \mathcal{N}\| = \|1 - \psi\, \big(u H'_x(\tilde{x}, \tilde{y}) + v H'_y(\tilde{x}, \tilde{y})\big)\, \|$
6. $\lambda_1(r) \geqslant \|D^2_{(0,r)} \mathcal{N}\| = \|\psi\, \big(u^2 H''_{xx}(\dots) + 2uv H''_{xy}(\dots) + v^2 H''_{yy}(\dots)\big)\, \|$
7. Find $r > 0$ s.t. $\lambda(r) := \lambda_0 + r\lambda_1(r)$ satisfies $d + r\lambda(r) < r$
8. Compute $\varepsilon = d/(1 - \lambda(r))$ and return RTAs $\mathbf{x} = (\tilde{x}, \varepsilon), \mathbf{y} = (\tilde{y}, \varepsilon)$

> Complexity: $O(N^2)$ interval operations

Rigorous Evaluation of Abelian Integral

- Rigorous integration along oval approximation $t \mapsto (\tilde{x}(t), \tilde{y}(t))$

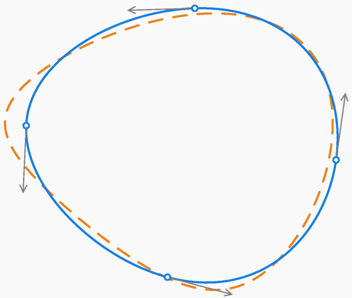
$$I_N = \int_0^{2\pi} \left(P(\tilde{x}(t), \tilde{y}(t))\, \tilde{y}'(t) - Q(\tilde{x}(t), \tilde{y}(t))\, \tilde{x}'(t) \right) dt$$



Rigorous Evaluation of Abelian Integral

- Rigorous integration along oval approximation $t \mapsto (\widetilde{x}(t), \widetilde{y}(t))$

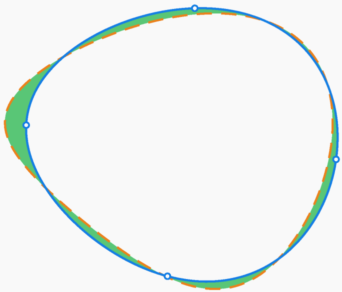
$$I_N = \int_0^{2\pi} \left(P(\widetilde{x}(t), \widetilde{y}(t))\, \widetilde{y}'(t) - Q(\widetilde{x}(t), \widetilde{y}(t))\, \widetilde{x}'(t) \right) dt$$



Rigorous Evaluation of Abelian Integral

- Rigorous integration along oval approximation $t \mapsto (\tilde{x}(t), \tilde{y}(t))$

$$I_N = \int_0^{2\pi} \left(P(\tilde{x}(t), \tilde{y}(t))\, \tilde{y}'(t) - Q(\tilde{x}(t), \tilde{y}(t))\, \tilde{x}'(t) \right) dt$$



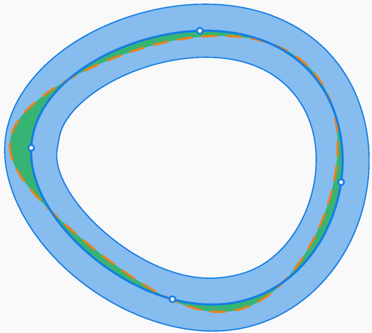
- Use Green's theorem to bound the error

$$|I_N - \mathcal{I}(h)| \leqslant \int_{\bullet} \left| P'_x(x,y) + Q'_y(x,y) \right| dxdy$$

Rigorous Evaluation of Abelian Integral

- Rigorous integration along oval approximation $t \mapsto (\widetilde{x}(t), \widetilde{y}(t))$

$$I_N = \int_0^{2\pi} \Big(P(\widetilde{x}(t), \widetilde{y}(t))\, \widetilde{y}'(t) - Q(\widetilde{x}(t), \widetilde{y}(t))\, \widetilde{x}'(t) \Big)\, dt$$



- Use Green's theorem to bound the error

$$|I_N - \mathcal{I}(h)| \leqslant \int_{\color{green}\bullet} \left| P'_x(x,y) + Q'_y(x,y) \right| dxdy \leqslant \int_{\color{blue}\bullet} \left| P'_x(x,y) + Q'_y(x,y) \right| dxdy$$

Rigorous Evaluation of Abelian Integral — Algorithm

INPUT: Perturbation (P, Q) and $\tilde{x}, \tilde{y}, u, v, \varepsilon$ computed previously

OUTPUT: Interval $\mathbf{I}$ containing $\mathcal{I}(h)$

Rigorous Evaluation of Abelian Integral — Algorithm

INPUT: Perturbation (P, Q) and $\widetilde{x}, \widetilde{y}, u, v, \varepsilon$ computed previously

OUTPUT: Interval $\mathbf{I}$ containing $\mathcal{I}(h)$

1. Rigorously evaluate using RTAs

$$I_N = \int_0^{2\pi} \Big(P(\widetilde{x}(t), \widetilde{y}(t))\, \widetilde{y}'(t) - Q(\widetilde{x}(t), \widetilde{y}(t))\, \widetilde{x}'(t) \Big)\, dt$$

Rigorous Evaluation of Abelian Integral — Algorithm

INPUT: Perturbation (P, Q) and $\tilde{x}, \tilde{y}, u, v, \varepsilon$ computed previously

OUTPUT: Interval $\mathbf{I}$ containing $\mathcal{I}(h)$

1. Rigorously evaluate using RTAs

$$I_N = \int_0^{2\pi} \Big(P(\tilde{x}(t), \tilde{y}(t))\, \tilde{y}'(t) - Q(\tilde{x}(t), \tilde{y}(t))\, \tilde{x}'(t) \Big)\, dt$$

2. Bound the area of the tube

$$\mathcal{A}(\bullet) \leqslant A_\varepsilon = 2\pi\varepsilon \Big(2\|\tilde{x}'v - \tilde{y}'u\| + \varepsilon\|u'v - v'u\| \Big)$$

Rigorous Evaluation of Abelian Integral — Algorithm

INPUT: Perturbation (P, Q) and $\widetilde{x}, \widetilde{y}, u, v, \varepsilon$ computed previously

OUTPUT: Interval $\mathbf{I}$ containing $\mathcal{I}(h)$

1. Rigorously evaluate using RTAs

$$I_N = \int_0^{2\pi} \left(P(\widetilde{x}(t), \widetilde{y}(t))\, \widetilde{y}'(t) - Q(\widetilde{x}(t), \widetilde{y}(t))\, \widetilde{x}'(t) \right) dt$$

2. Bound the area of the tube

$$\mathcal{A}(\bullet) \leqslant A_\varepsilon = 2\pi\varepsilon \left(2\|\widetilde{x}'v - \widetilde{y}'u\| + \varepsilon\|u'v - v'u\| \right)$$

3. Bound the Green term over the tube

$$B \geqslant \|P'_x(\boldsymbol{x}, \boldsymbol{y}) + Q'_y(\boldsymbol{x}, \boldsymbol{y})\| \qquad \text{with} \quad \boldsymbol{x} = (\widetilde{x}, \varepsilon),\ \boldsymbol{y} = (\widetilde{y}, \varepsilon)$$

Rigorous Evaluation of Abelian Integral — Algorithm

INPUT: Perturbation (P, Q) and $\widetilde{x}, \widetilde{y}, u, v, \varepsilon$ computed previously

OUTPUT: Interval $\mathbf{I}$ containing $\mathcal{I}(h)$

1. Rigorously evaluate using RTAs

$$I_N = \int_0^{2\pi} \Big(P(\widetilde{x}(t), \widetilde{y}(t))\, \widetilde{y}'(t) - Q(\widetilde{x}(t), \widetilde{y}(t))\, \widetilde{x}'(t) \Big)\, dt$$

2. Bound the area of the tube

$$\mathcal{A}(\bullet) \leqslant A_\varepsilon = 2\pi\varepsilon \Big(2\|\widetilde{x}'v - \widetilde{y}'u\| + \varepsilon \|u'v - v'u\| \Big)$$

3. Bound the Green term over the tube

$$B \geqslant \|P'_x(\boldsymbol{x}, \boldsymbol{y}) + Q'_y(\boldsymbol{x}, \boldsymbol{y})\| \qquad \text{with} \quad \boldsymbol{x} = (\widetilde{x}, \varepsilon),\ \boldsymbol{y} = (\widetilde{y}, \varepsilon)$$

4. Return $\mathbf{I} = I \pm A_\varepsilon B$

Rigorous Evaluation of Abelian Integral — Algorithm

INPUT: Perturbation (P, Q) and $\widetilde{x}, \widetilde{y}, u, v, \varepsilon$ computed previously

OUTPUT: Interval $\mathbf{I}$ containing $\mathcal{I}(h)$

1. Rigorously evaluate using RTAs

$$I_N = \int_0^{2\pi} \Big(P(\widetilde{x}(t), \widetilde{y}(t))\, \widetilde{y}'(t) - Q(\widetilde{x}(t), \widetilde{y}(t))\, \widetilde{x}'(t) \Big)\, dt$$

2. Bound the area of the tube

$$\mathcal{A}(\bullet) \leqslant A_\varepsilon = 2\pi\varepsilon \Big(2\|\widetilde{x}'v - \widetilde{y}'u\| + \varepsilon\|u'v - v'u\| \Big)$$

3. Bound the Green term over the tube

$$B \geqslant \|P'_x(\boldsymbol{x}, \boldsymbol{y}) + Q'_y(\boldsymbol{x}, \boldsymbol{y})\| \qquad \text{with} \quad \boldsymbol{x} = (\widetilde{x}, \varepsilon),\ \boldsymbol{y} = (\widetilde{y}, \varepsilon)$$

4. Return $\quad \mathbf{I} = I \pm A_\varepsilon B$

> Complexity: $O(N^2)$ interval operations

Toward Efficient and Certified Implementation

Approximation of Functions: Some Available Libraries

- Numerical Chebyshev and Fourier approximations: Chebfun (Matlab), ApproxFun (Julia), Dedalus (Python)

  ⇒ Our current implem. = rigorous wrapper around Julia/ApproxFun

julia

Approximation of Functions: Some Available Libraries

- Numerical Chebyshev and Fourier approximations: Chebfun (Matlab), ApproxFun (Julia), Dedalus (Python)
  - ⇒ Our current implem. = rigorous wrapper around Julia/ApproxFun

- Rigorous approximation of functions: COSY, VNODE-LP, CAPD, Sollya
  - ⇒ Mostly based on Taylor approximations

Approximation of Functions: Some Available Libraries

- Numerical Chebyshev and Fourier approximations: Chebfun (Matlab), ApproxFun (Julia), Dedalus (Python)
  ⇒ Our current implem. = rigorous wrapper around Julia/ApproxFun

- Rigorous approximation of functions: COSY, VNODE-LP, CAPD, Sollya
  ⇒ Mostly based on Taylor approximations

- Certified (formally proved) implementations: Taylor models in Coq (CoqInterval/CoqApprox), HOL-Light, Isabelle, ...
  ⇒ ApproxModels: a recent Coq development for more general approximations *(F. Bréhard, A. Mahboubi, D. Pous)*

A Brief Overview of the ApproxModels Coq Library

- Abstract formalization of rigorous approximations of functions

```
Record Model C := { pol: seq C; rem: C }.
Definition mcontains (F: Model) (f: R → R) :=
  ∃ p: seq R, scontains (pol F) p /\
  ∀ x, lo≤x≤hi → contains (rem F)(f x-eval T p x)
```

A Brief Overview of the ApproxModels Coq Library

- Abstract formalization of rigorous approximations of functions

```
Record Model C := { pol: seq C; rem: C }.
Definition mcontains (F: Model) (f: R → R) :=
  ∃ p: seq R, scontains (pol F) p /\
  ∀ x, lo⩽x⩽hi → contains (rem F)(f x-eval T p x)
```

- Implementation and proof of basic operations

```
Definition mmul (M N: Model): Model :=
  {| pol := pol M * pol N;
     rem := srange (pol M)*rem N +
            srange (pol N)*rem M +
            rem M*rem N |}.
Lemma rmmul: ∀ F f G g,
  mcontains F f → mcontains G g →
  mcontains (F*G) (f*g).
```

A Brief Overview of the ApproxModels Coq Library

- Abstract formalization of rigorous approximations of functions

```
Record Model C := { pol: seq C; rem: C }.
Definition mcontains (F: Model) (f: R → R) :=
  ∃ p: seq R, scontains (pol F) p /\
  ∀ x, lo≤x≤hi → contains (rem F)(f x-eval T p x)
```

- Implementation and proof of basic operations

- A posteriori validation approach for division, square root, ...

⇒ Banach fixed-point theorem proved using the Coquelicot library

- Interval computations provided by CoqInterval

- Full implementation in Chebyshev basis

A Brief Overview of the ApproxModels Coq Library

- Abstract formalization of rigorous approximations of functions

```
Record Model C := { pol: seq C; rem: C }.
Definition mcontains (F: Model) (f: R → R) :=
  ∃ p: seq R, scontains (pol F) p /\
  ∀ x, lo⩽x⩽hi → contains (rem F)(f x-eval T p x)
```

- Implementation and proof of basic operations

- A posteriori validation approach for division, square root, ...

⇒ Banach fixed-point theorem proved using the Coquelicot library

- Interval computations provided by CoqInterval

- Full implementation in Chebyshev basis

GOAL: Extending this work with Rigorous Trigonometric Approximations

Beyond Quadratic Complexity using Rigorous FFT?

- Multiplication in $O(N^2)$ using schoolbook algorithm:

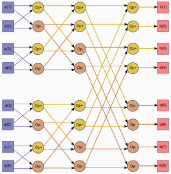
$$\begin{cases} \cos nt \cos mt = \frac{1}{2}(\cos(n-m)t + \cos(n+m)t) \\ \sin nt \sin mt = \frac{1}{2}(\cos(n-m)t - \cos(n+m)t) \\ \sin nt \cos mt = \frac{1}{2}(\sin(n+m)t - \sin(n-m)t) \end{cases}$$

Beyond Quadratic Complexity using Rigorous FFT?

- Multiplication in $O(N^2)$ using schoolbook algorithm:

$$\begin{cases} \cos nt \cos mt = \frac{1}{2}(\cos(n-m)t + \cos(n+m)t) \\ \sin nt \sin mt = \frac{1}{2}(\cos(n-m)t - \cos(n+m)t) \\ \sin nt \cos mt = \frac{1}{2}(\sin(n+m)t - \sin(n-m)t) \end{cases}$$

- Rigorous FFT is not trivial!

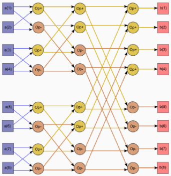


Beyond Quadratic Complexity using Rigorous FFT?

- Multiplication in $O(N^2)$ using schoolbook algorithm:

$$\begin{cases} \cos nt \cos mt = \frac{1}{2}(\cos(n-m)t + \cos(n+m)t) \\ \sin nt \sin mt = \frac{1}{2}(\cos(n-m)t - \cos(n+m)t) \\ \sin nt \cos mt = \frac{1}{2}(\sin(n+m)t - \sin(n-m)t) \end{cases}$$

- Rigorous FFT is not trivial!



→ N. Brisebarre, M. Joldes, J.-M. Muller, A.-M. Naneș, J. Picot. *Error analysis of some operations involved in the Cooley-Tukey Fast Fourier Transform*

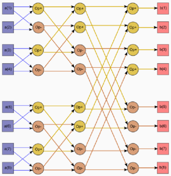
⇒ To be formalized in Coq in the future?

Beyond Quadratic Complexity using Rigorous FFT?

- Multiplication in $O(N^2)$ using schoolbook algorithm:

$$\begin{cases} \cos nt \cos mt = \frac{1}{2}(\cos(n-m)t + \cos(n+m)t) \\ \sin nt \sin mt = \frac{1}{2}(\cos(n-m)t - \cos(n+m)t) \\ \sin nt \cos mt = \frac{1}{2}(\sin(n+m)t - \sin(n-m)t) \end{cases}$$

- Rigorous FFT is not trivial!



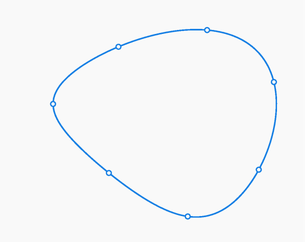
$\longrightarrow$ N. Brisebarre, M. Joldes, J.-M. Muller, A.-M. Nanes, J. Picot. *Error analysis of some operations involved in the Cooley-Tukey Fast Fourier Transform*

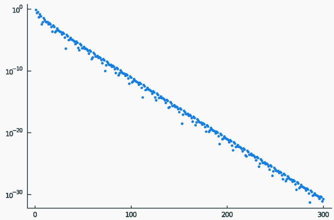
$\Rightarrow$ To be formalized in Coq in the future?

GOAL: Certified fast arithmetic on trigonometric polynomials

Some Other Possible Optimizations

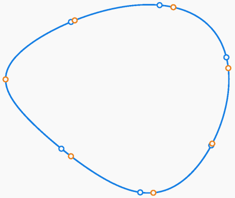
- Oval reparameterization using non uniform FFT

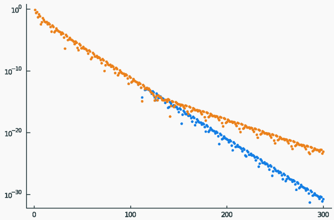




Some Other Possible Optimizations

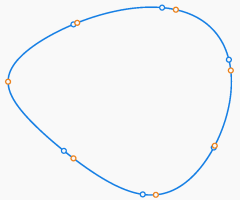
- Oval reparameterization using non uniform FFT

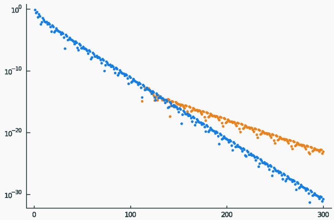




Some Other Possible Optimizations

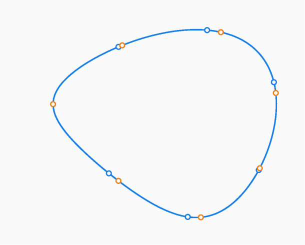
- Oval reparameterization using non uniform FFT

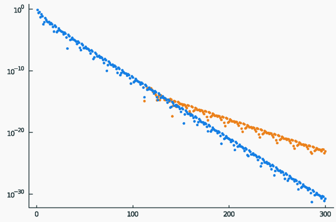




Some Other Possible Optimizations

- Oval reparameterization using non uniform FFT





- Use the Picard-Fuchs equation for $h \mapsto \mathcal{I}(h)$ for multiple evaluations

Example:

$$\left(\frac{10\,097\,980\,101\,h}{78\,125\,000} - \frac{755\,777\,647\,h^2}{1\,562\,500} + \frac{10\,244\,409\,h^3}{15\,625} - \frac{47\,268\,h^4}{125} + \frac{1936\,h^5}{25}\right) D_h^3 +$$
$$\left(\frac{10\,097\,980\,101}{78\,125\,000} - \frac{1\,299\,331\,973\,h}{1\,562\,500} + \frac{1\,064\,001\,h^2}{625} - \frac{870\,636\,h^3}{625} + \frac{1936\,h^4}{5}\right) D_h^2 + \left(-\frac{812\,907\,489}{6\,250\,000} + \frac{32\,163\,753\,h}{62\,500} - \frac{419\,679\,h^2}{625} + \frac{1452\,h^3}{5}\right) D_h$$

Conclusion

Computing Abelian Integrals Efficiently and Rigorously

1. Numerical approximation procedure used as oracle
   - Initial guess + Newton's method on N sampled points
   - Exponentially fast converging numerical quadrature

2. A posteriori validation of the oval
   - Well-posed problem using a transversal vector field
   - Computing with Rigorous Trigonometric Approximations
   - Effective tube obtained with the Banach fixed-point theorem

3. Rigorous Integration using Green's theorem

Present and Future Achievements

- ○ Numerical approximation in quasi-linear complexity
- × Validation in quadratic complexity $\rightarrow$ rigorous FFT for RTAs?

- ○ Proof of the exponential convergence w.r.t. target precision
- × A more in-depth analysis of the complexity w.r.t. all parameters

- ○ Implementation in Julia using ApproxFun.jl and ValidatedNumerics.jl
- × Certified implementation in Coq $\rightarrow$ coming soon!

- ○ Successfully applied to a few examples from the literature
- × Finding new lower bounds on $\mathcal{H}(n)$?