Certified and fast computation of supremum norms of approximation errors

Sylvain Chevillard* Mioara Joldes* Christoph Lauter**

June 2009, ARITH 19

*Arénaire, LIP (CNRS/ÉNS Lyon/INRIA/Université de Lyon)

**Intel Corp. SSG-DPD-PLL-Numerics Group

Outline

- Motivation
  - Correctly rounded elementary functions
  - Supremum norm of error functions
  - Previous approaches and difficulties
- Our approach
  - Automatic differentiation and Taylor models
  - Isolation of roots of polynomials
  - Enclosure of roots of functions
- Results & Conclusion

Motivation - Mathematical Libraries

- What? Compute $\sin$, $\cos$, $\exp$ in finite precision, floating-point environment.
- Why? Software systems: scientific computing, financial, embedded systems.

Motivation - Mathematical Libraries

- What? Compute $\sin$, $\cos$, $\exp$ in finite precision, floating-point environment.
- Why? Software systems: scientific computing, financial, embedded systems.

Motivation - Mathematical Libraries

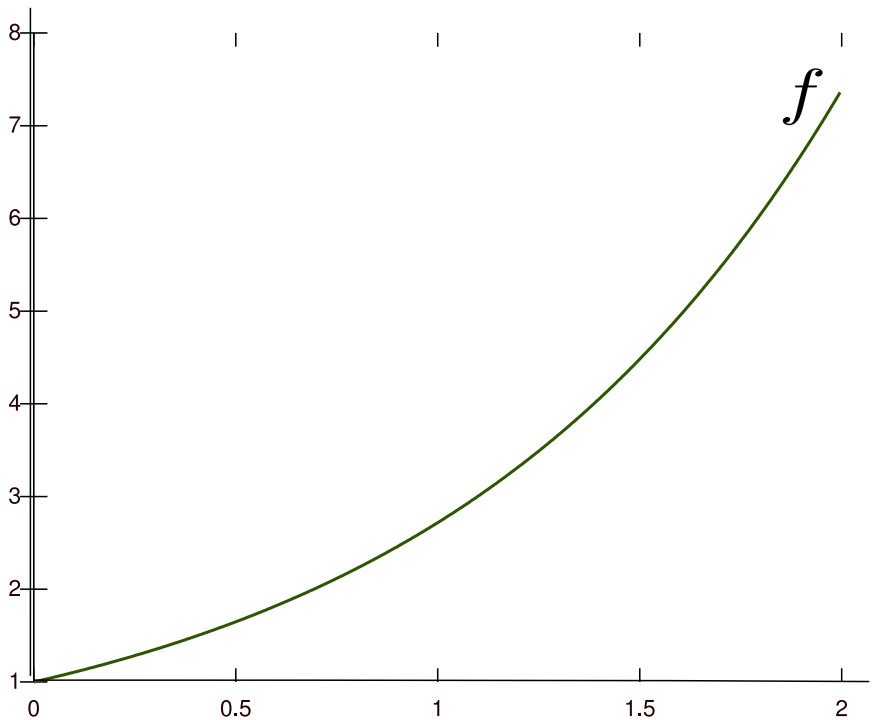
- What? Compute $\sin$, $\cos$, $\exp$ in finite precision, floating-point environment.
- Why? Software systems: scientific computing, financial, embedded systems.
- Most Mathematical Libraries do not provide correctly rounded functions.
- IEEE-754-2008 recommends correct rounding.

Motivation - Mathematical Libraries

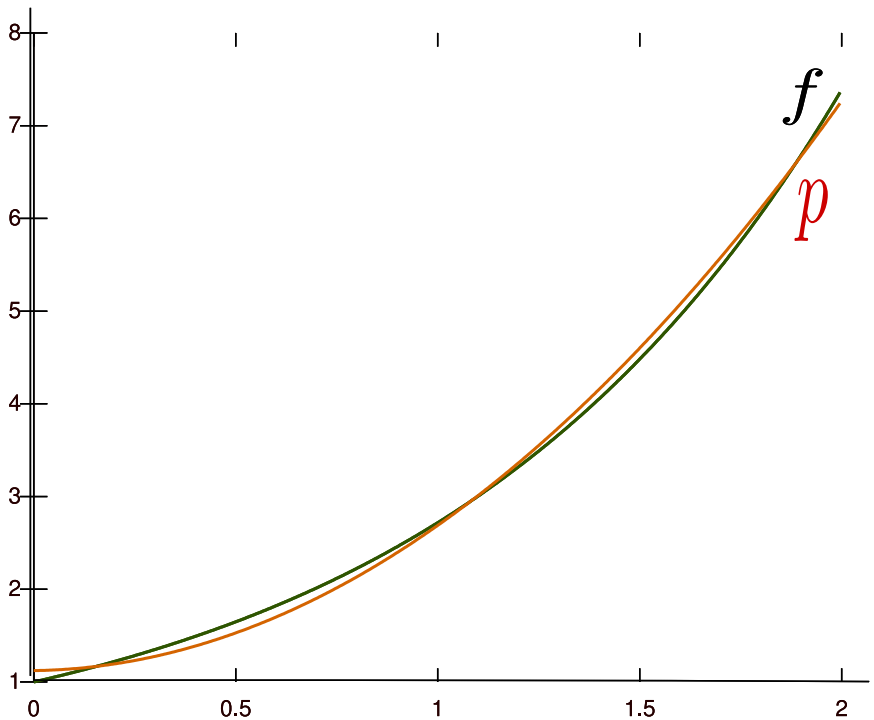
- What? Compute $\sin$, $\cos$, $\exp$ in finite precision, floating-point environment.
- Why? Software systems: scientific computing, financial, embedded systems.
- Most Mathematical Libraries do not provide correctly rounded functions.
- IEEE-754-2008 recommends correct rounding.
- Arenaire team develops the Correctly Rounded Libm (CRLibm)[1].

[1] http://lipforge.ens-lyon.fr/www/crlibm/

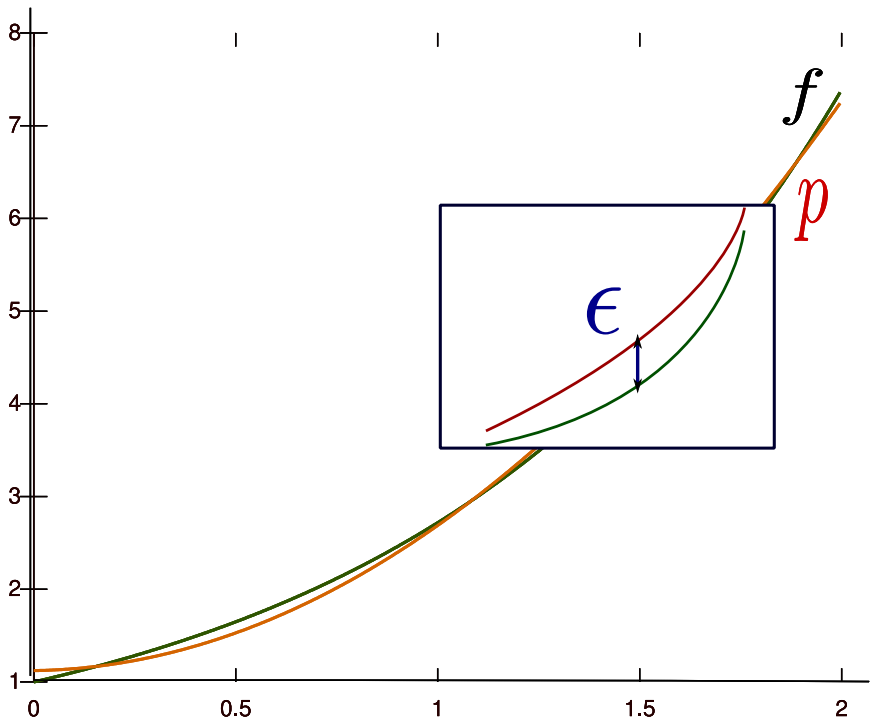
Supremum Norms of Error Functions



Supremum Norms of Error Functions



Supremum Norms of Error Functions

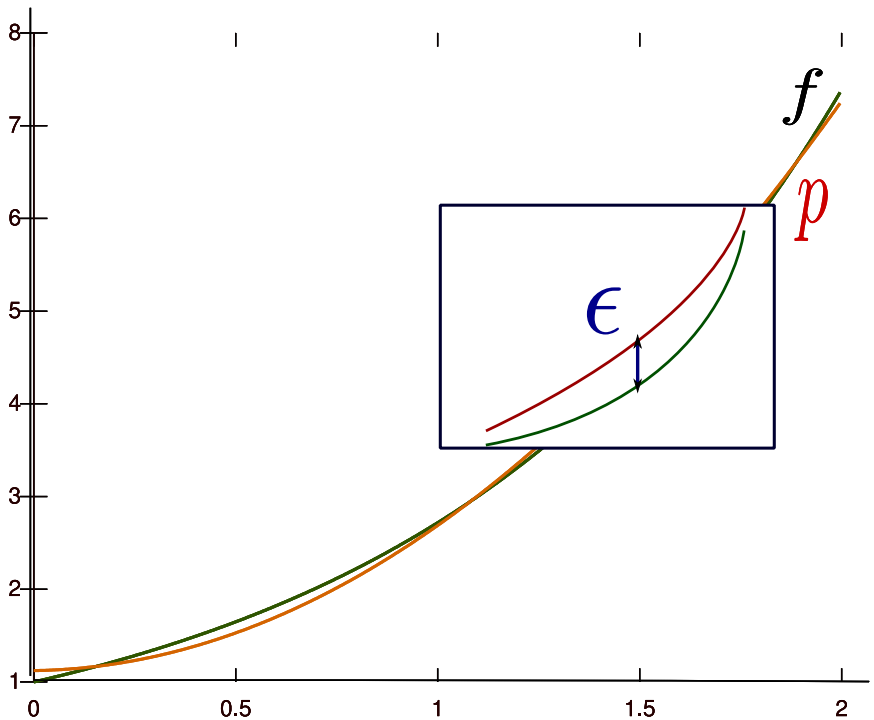


$$\varepsilon(x) = f(x) - p(x),\ x \in [a,b] \text{ or}$$

$$\varepsilon(x) = \frac{p(x)}{f(x)} - 1,\ x \in [a,b]$$

Define $\|\varepsilon\|_\infty = \sup_{x\in[a,b]} |\varepsilon(x)|$

Supremum Norms of Error Functions



$$\varepsilon(x) = f(x) - p(x),\ x \in [a,b] \text{ or}$$

$$\varepsilon(x) = \frac{p(x)}{f(x)} - 1,\ x \in [a,b]$$

Define $\|\varepsilon\|_\infty = \sup_{x \in [a,b]} |\varepsilon(x)|$

Compute certified bound for supremum norm of error function

Supremum Norms of Error Functions

- Error $\varepsilon(x) = f(x) - p(x) \quad \text{or} \quad \varepsilon(x) = \frac{p(x)}{f(x)} - 1, \quad x \in [a, b]$
- Define $\|\varepsilon\|_\infty = \sup_{x \in [a,\, b]} |\varepsilon(x)|$
- Given p and f find a narrow interval $\mathbf{r}$ such that $\|\varepsilon\|_\infty \in \mathbf{r}$.

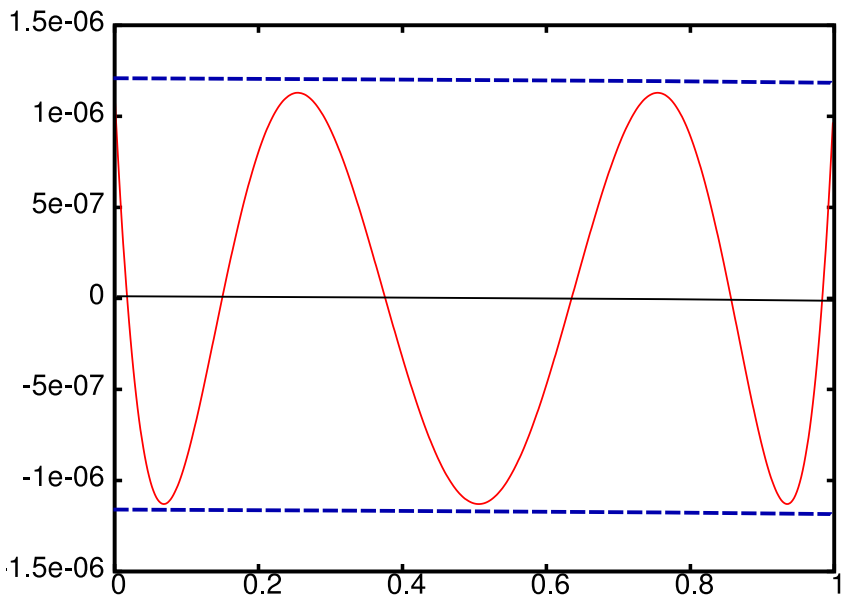
Need for a fast and certified algorithm:

- Correctly rounded elementary functions
- Compute error bounds between a function and thousands of polynomials

Our Problem - Prone to High Dependency Phenomenon

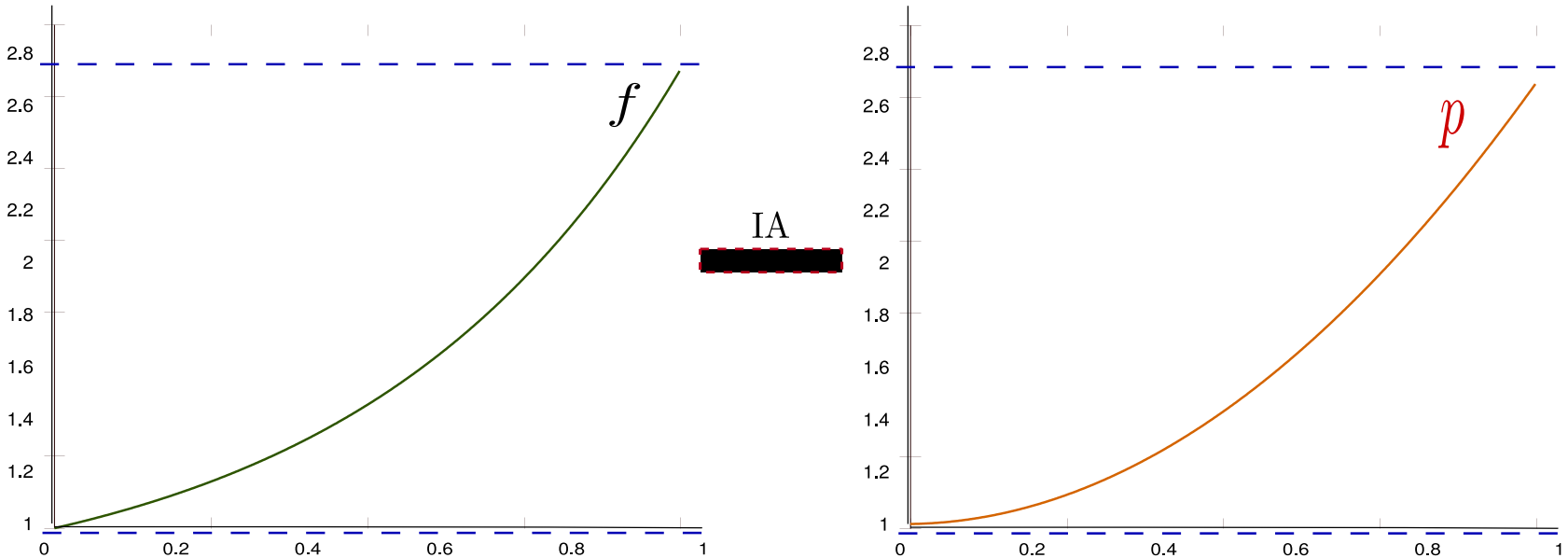
Example:

$f(x) = e^x,\ x \in [0, 1]$
$p(x) = \sum_{i=0}^{5} c_i x^i$ s.t. $\|f - p\|_\infty$ is as small as possible (Remez)
$\varepsilon(x) = f(x) - p(x)$

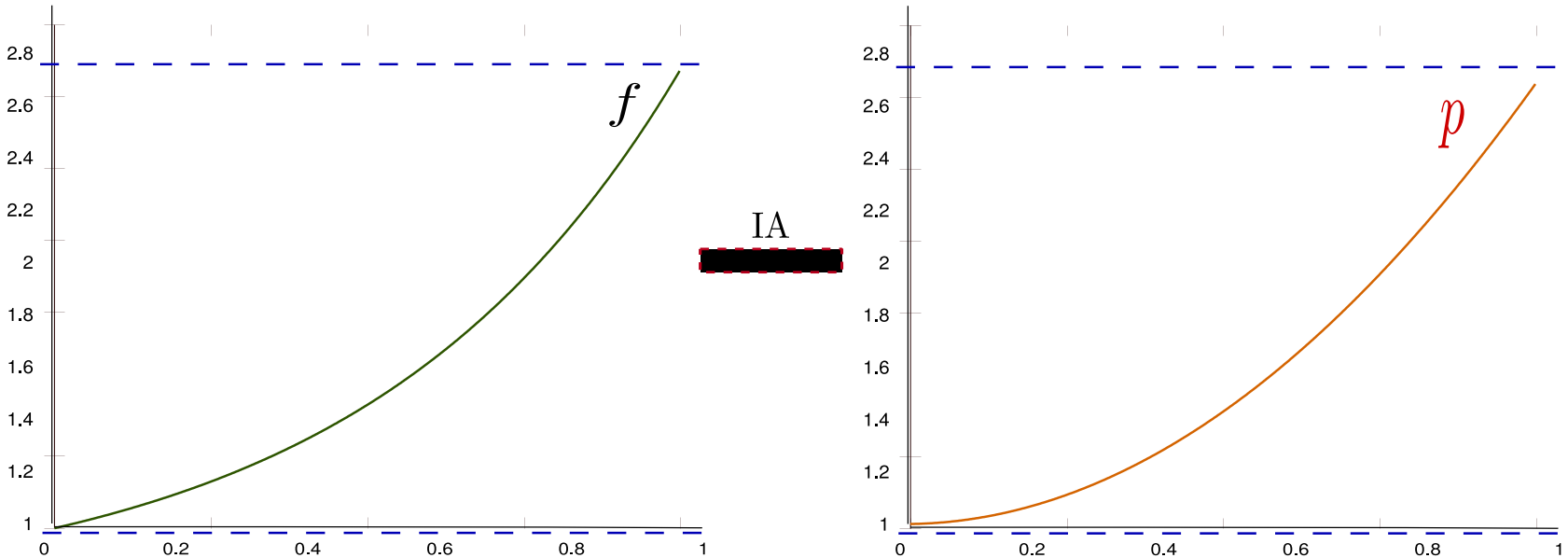


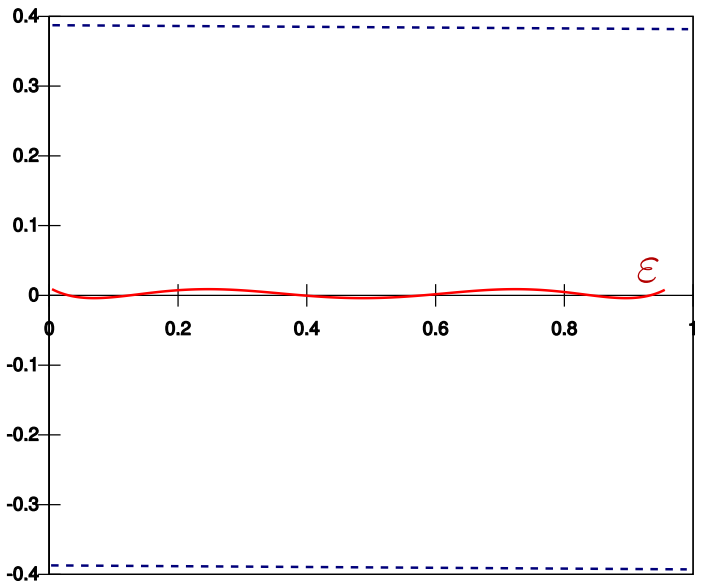
Using IA, $\varepsilon(x) \in [-0.4, 0.4]$, but $\|\varepsilon(x)\|_\infty \simeq 1.1295e-6$:

Interval Arithmetic - Overestimation

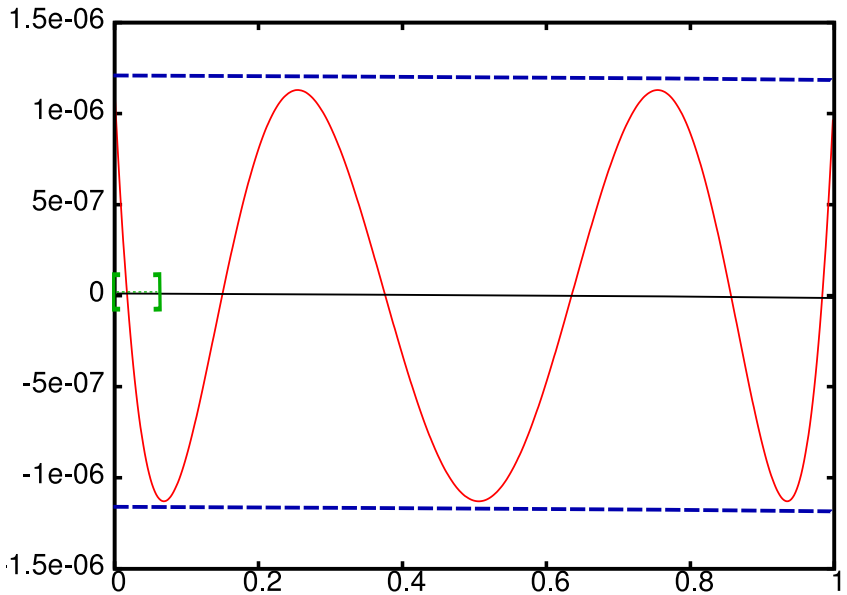


Interval Arithmetic - Overestimation

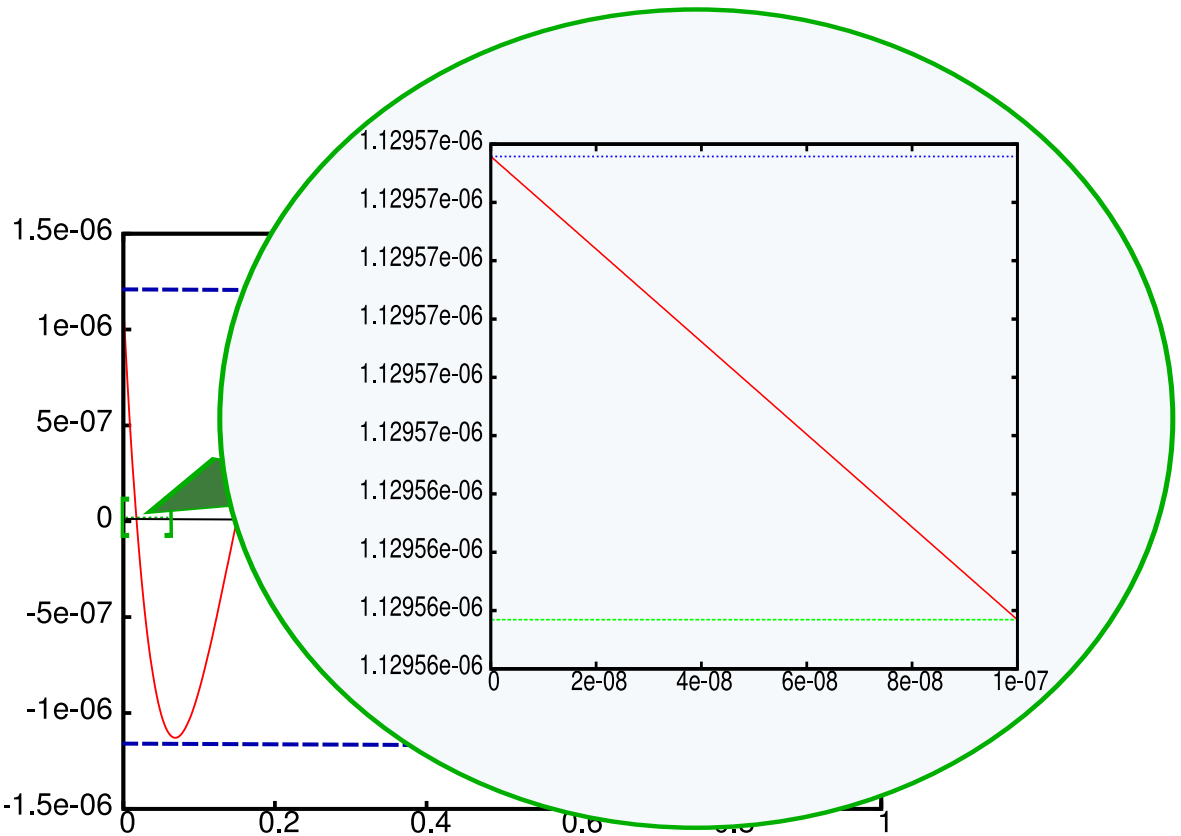




Our Problem - Prone to High Dependency Phenomenon



Our Problem - Prone to High Dependency Phenomenon



In this case, over $[0, 1]$ we need 10^7 intervals!

Previous Approaches

- Floating-point techniques: not “safe“
- Existent Interval Arithmetic methods: not sufficient
- Global optimization software: not tailored for specific problem
- Chevillard and Lauter’s technique:
  - interval arithmetic
  - tight bounding of the zeros of the derivative
  - removes false singularities (like $x = 0$ for $\sin(x)/x$)
  - high computation time for $\deg(p) > 10$.
- Taylor expansion techniques:
  - replace error function with a polynomial
  - bound remainder sufficiently close
- Certified polynomial approximations of analytic functions.

Our Approach

- How to obtain a certified and tight bound for $\|\varepsilon\|_\infty = \|f - p\|_\infty$ AND $\|\varepsilon\|_\infty = \|p/f - 1\|_\infty$, over given interval $[a, b]$?

2

Our Approach

- How to obtain a certified and tight bound for $\|\varepsilon\|_\infty = \|f - p\|_\infty$ AND $\|\varepsilon\|_\infty = \|p/f - 1\|_\infty$, over given interval $[a, b]$?
- Idea: Tightly enclose all the extrema of ε: they are among the zeros of the first derivative.

2

Our Approach

- How to obtain a certified and tight bound for $\|\varepsilon\|_\infty = \|f - p\|_\infty$ AND $\|\varepsilon\|_\infty = \|p/f - 1\|_\infty$, over given interval $[a, b]$?
- Idea: Tightly enclose all the extrema of ε: they are among the zeros of the first derivative.
- $\varepsilon' = (p'f - pf')/f^2$. Consider $\tau = p'f - pf'$.

2

Our Approach

- How to obtain a certified and tight bound for $\|\varepsilon\|_\infty = \|f - p\|_\infty$ AND $\|\varepsilon\|_\infty = \|p/f - 1\|_\infty$, over given interval $[a, b]$?
- Idea: Tightly enclose all the extrema of ε: they are among the zeros of the first derivative.
- $\varepsilon' = (p'f - pf')/f^2$. Consider $\tau = p'f - pf'$.
- Find small intervals that enclose each root of τ.
- Evaluate ε on these intervals.

Our Approach

- How to obtain a certified and tight bound for $\|\varepsilon\|_\infty = \|f - p\|_\infty$ AND $\|\varepsilon\|_\infty = \|p/f - 1\|_\infty$, over given interval $[a, b]$?
- Idea: Tightly enclose all the extrema of ε: they are among the zeros of the first derivative.
- $\varepsilon' = (p'f - pf')/f^2$. Consider $\tau = p'f - pf'$.
- Find small intervals that enclose each root of τ.
- Evaluate ε on these intervals.
- Use multiprecision interval arithmetic: MPFI [2]

[2] http://gforge.inria.fr/projects/mpfi/

Finding enclosures of roots of a function

- How to Solve $\tau(x) = 0$?
- First idea: interval Newton Method BUT Dependency phenomenon present in $\tau = p'f - pf'$ also!

Finding enclosures of roots of a function

- How to Solve $\tau(x) = 0$?
- First idea: interval Newton Method BUT Dependency phenomenon present in $\tau = p'f - pf'$ also!
- Better idea: introduce higher degree approximation polynomial for τ.

Finding enclosures of roots of a function

- How to Solve $\tau(x) = 0$?
- First idea: interval Newton Method BUT Dependency phenomenon present in $\tau = p'f - pf'$ also!
- Better idea: introduce higher degree approximation polynomial for τ.
- Compute a Taylor Model:
  $(T, [-\theta, \theta])$ s.t. $\tau(x) - T(x) \in [-\theta, \theta], \forall x \in I$
- $\tau(x) = 0$ implies $T(x) \in [-\theta, \theta]$

Finding enclosures of roots of a function

- How to Solve $\tau(x) = 0$?
- First idea: interval Newton Method BUT Dependency phenomenon present in $\tau = p'f - pf'$ also!
- Better idea: introduce higher degree approximation polynomial for τ.
- Compute a Taylor Model: $(T, [-\theta, \theta])$ s.t. $\tau(x) - T(x) \in [-\theta, \theta], \forall x \in I$
- $\tau(x) = 0$ implies $T(x) \in [-\theta, \theta]$

How to compute $(T, [-\theta, \theta])$?

Taylor Formula with Lagrange form of remainder

Let $n \in \mathbb{N}$; Consider τ an n times differentiable function over $[a, b]$ around x_0:

$$\tau(x) = \sum_{i=0}^{n-1} \frac{\tau^{(i)}(x_0)(x - x_0)^i}{i!} + \Delta_n(x, \xi)$$

How to compute $(T, [-\theta, \theta])$?

Taylor Formula with Lagrange form of remainder

Let $n \in \mathbb{N}$; Consider τ an n times differentiable function over $[a, b]$ around x_0:

$$\tau(x) = \underbrace{\sum_{i=0}^{n-1} \frac{\tau^{(i)}(x_0)(x - x_0)^i}{i!}}_{T(x)} + \Delta_n(x, \xi)$$

How to compute $(T, [-\theta, \theta])$?

Taylor Formula with Lagrange form of remainder

Let $n \in \mathbb{N}$; Consider τ an n times differentiable function over $[a, b]$ around x_0:

$$\tau(x) = \underbrace{\sum_{i=0}^{n-1} \frac{\tau^{(i)}(x_0)(x - x_0)^i}{i!}}_{T(x)} + \underbrace{\Delta_n(x, \xi)}_{\text{remainder}}$$

$\Delta_n(x, \xi) = \dfrac{\tau^{(n)}(\xi)(x - x_0)^n}{n!}$, $x \in [a, b]$, ξ lies strictly between x and x_0

How to compute $(T, [-\theta, \theta])$?

Taylor Formula with Lagrange form of remainder

Let $n \in \mathbb{N}$; Consider τ an n times differentiable function over $[a, b]$ around x_0:

$$\tau(x) = \underbrace{\sum_{i=0}^{n-1} \frac{\tau^{(i)}(x_0)(x - x_0)^i}{i!}}_{T(x)} + \underbrace{\Delta_n(x, \xi)}_{\text{remainder}}$$

- Compute coefficients of T: $\frac{\tau^{(i)}(x_0)}{i!}, i \in \{0, \ldots, n-1\}$
- Compute $[-\theta, \theta]$ as an enclosure for $\frac{\tau^{(n)}(\xi)}{n!}$, when $\xi \in [a, b]$.

Computing $(T, [-\theta, \theta])$ - Automatic differentiation (AD)

- Purpose: Compute values of high order derivatives of functions
- τ usually represented as an expression tree

Computing $(T, [-\theta, \theta])$ - Automatic differentiation (AD)

- Purpose: Compute values of high order derivatives of functions
- τ usually represented as an expression tree
- Problem: For large values of n, high memory usage for symbolic differentiation.
- Solution: same operations like when evaluating the expression of the derivative, don't write this expression

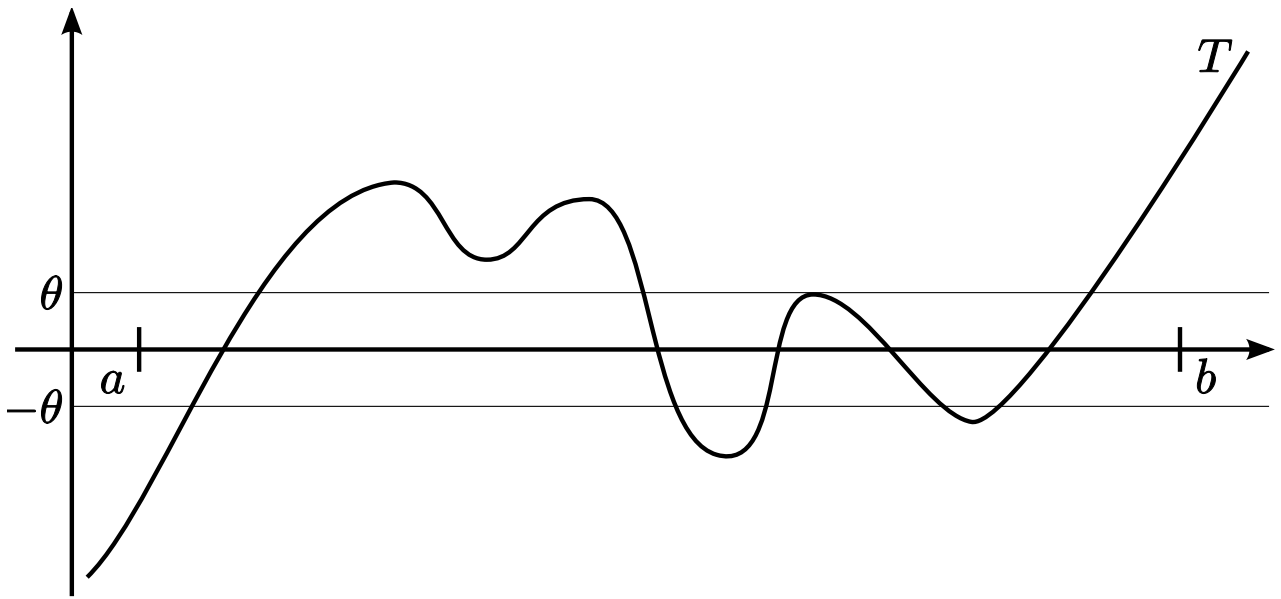
Computing $(T, [-\theta, \theta])$ - Automatic differentiation (AD)

- Purpose: Compute values of high order derivatives of functions
- τ usually represented as an expression tree
- Problem: For large values of n, high memory usage for symbolic differentiation.
- Solution: same operations like when evaluating the expression of the derivative, don't write this expression
- AD: usually developed by program code transformation, operators overloading

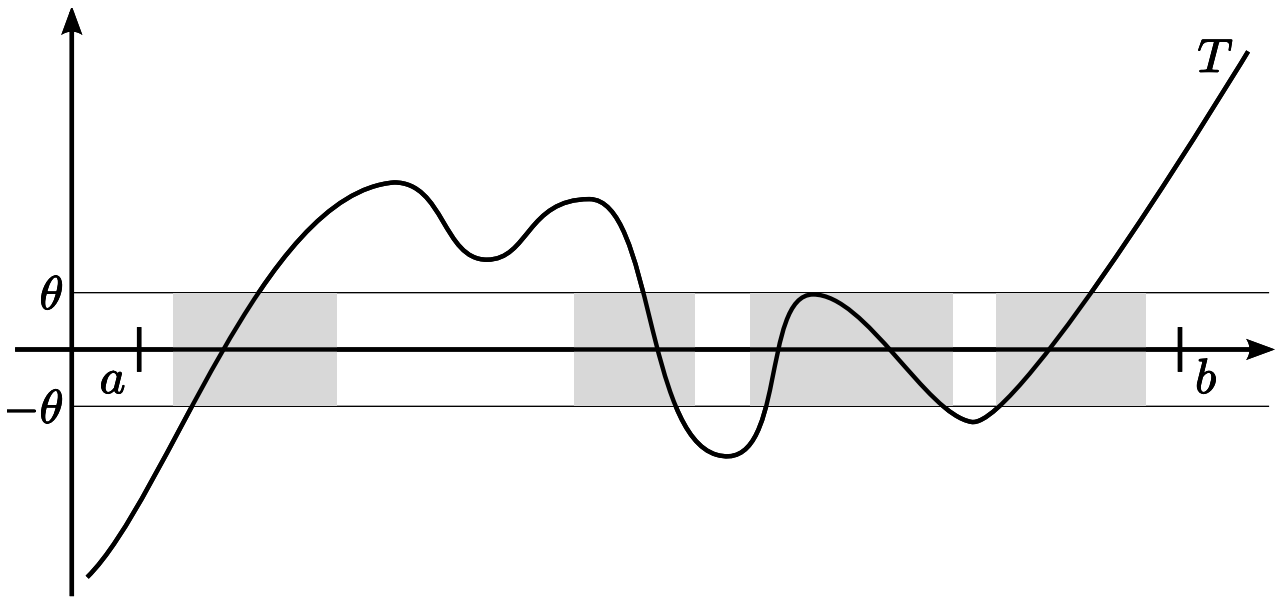
Computing $(T, [-\theta, \theta])$ - Automatic differentiation (AD)

- Purpose: Compute values of high order derivatives of functions
- τ usually represented as an expression tree
- Problem: For large values of n, high memory usage for symbolic differentiation.
- Solution: same operations like when evaluating the expression of the derivative, don't write this expression
- AD: usually developed by program code transformation, operators overloading
- Can be easily extended to work with interval arithmetic

How to solve $T(x) \in [-\theta, \theta]$?

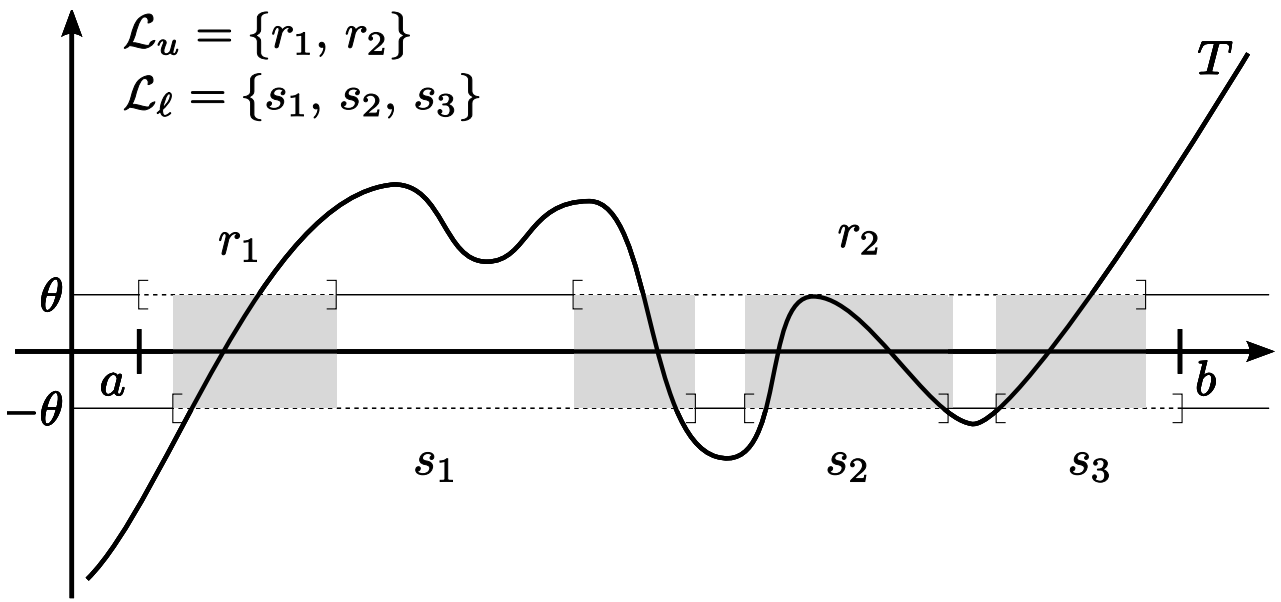


How to solve $T(x) \in [-\theta, \theta]$?



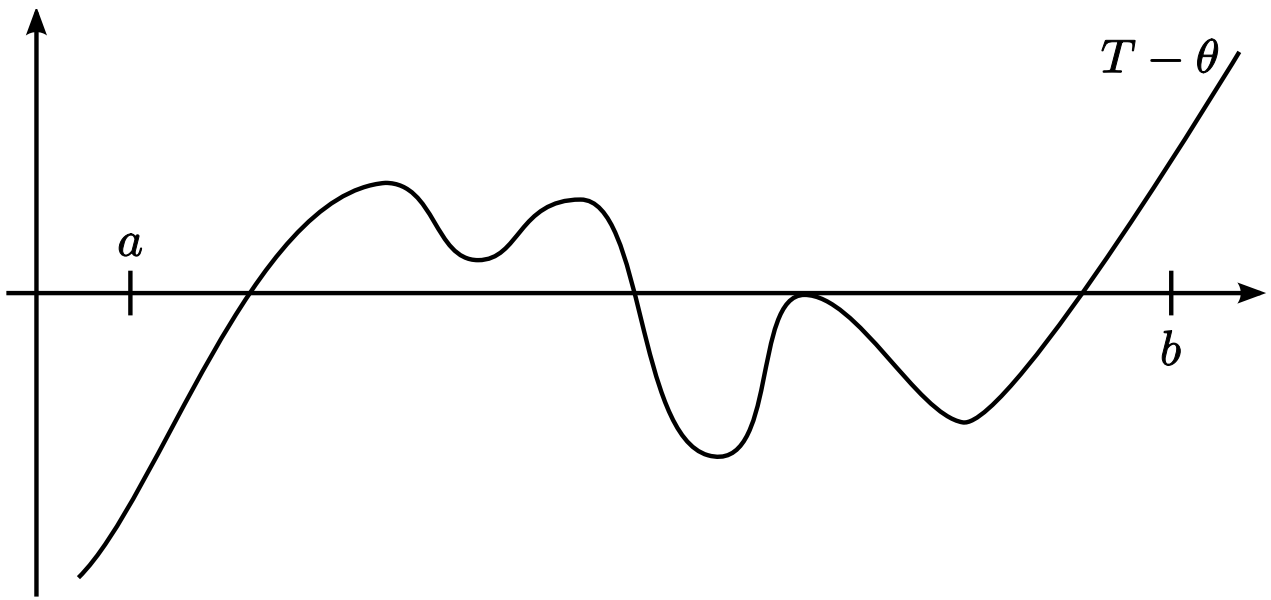
How to solve $T(x) \in [-\theta, \theta]$?

- Compute a list $\mathcal{L}_u$ of intervals where $T - \theta \leq 0$ and $\mathcal{L}_l$ where $T + \theta \geq 0$



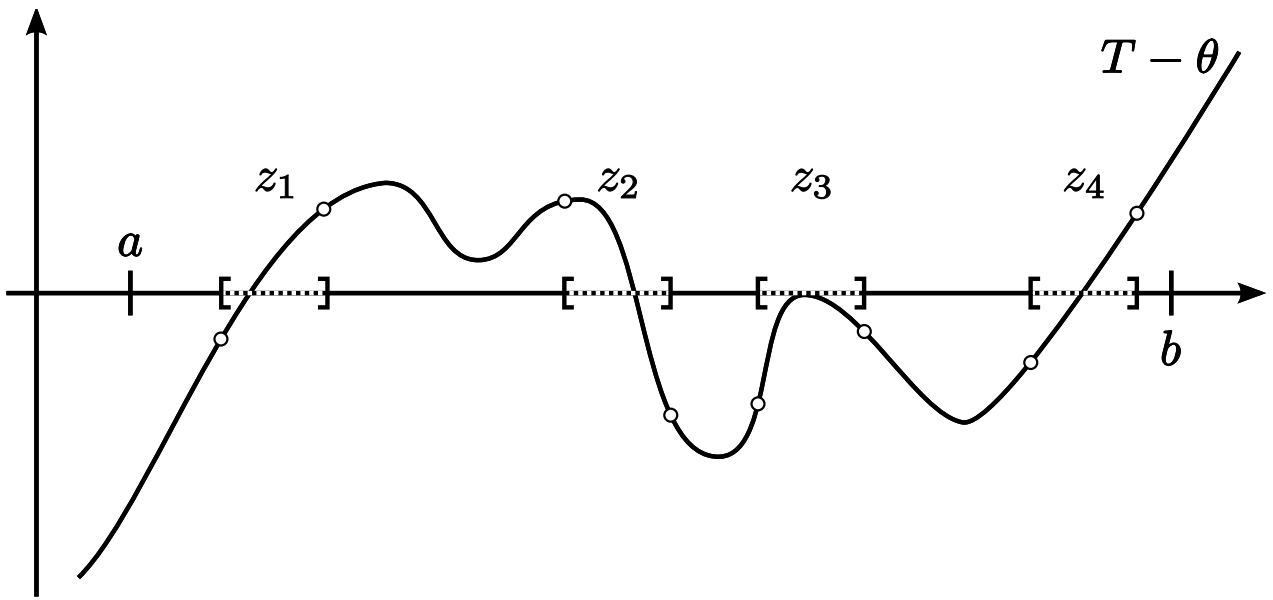
How to solve $T(x) \in [-\theta, \theta]$?

- Compute a list of intervals where $T - \theta \leq 0$, T is a polynomial



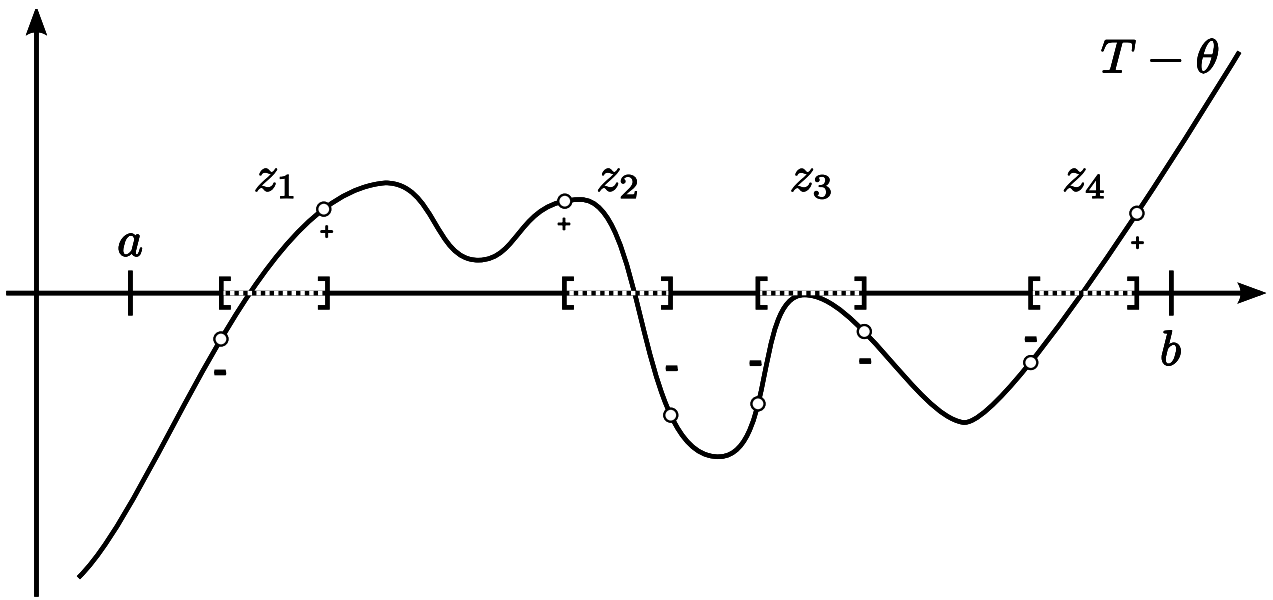
How to solve $T(x) \in [-\theta, \theta]$?

- Compute a list of intervals where $T - \theta \leq 0$, T is a polynomial
- Compute enclosures of the roots of $T - \theta$



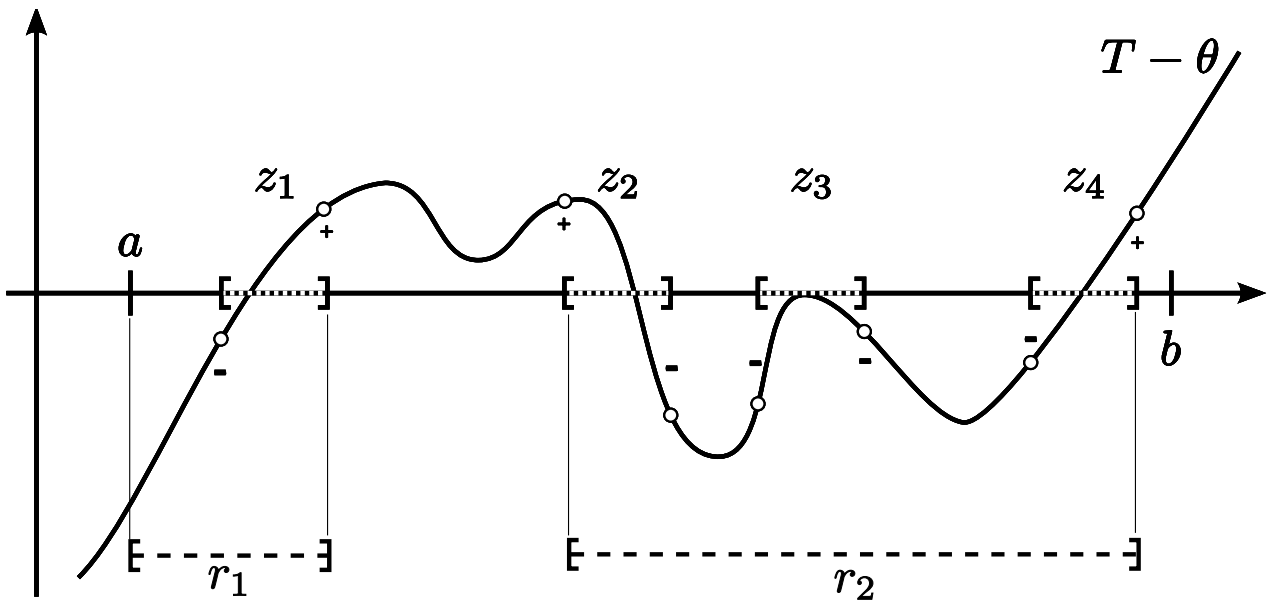
How to solve $T(x) \in [-\theta, \theta]$?

- Compute a list of intervals where $T - \theta \leq 0$, T is a polynomial
- Compute enclosures of the roots of $T - \theta$
- Compute sign changes



How to solve $T(x) \in [-\theta, \theta]$?

- Compute a list of intervals where $T - \theta \leq 0$, T is a polynomial
- Compute enclosures of the roots of $T - \theta$
- Compute sign changes
- Find suitable intervals



Isolation and refinement of roots of polynomials

- Techniques based on counting the number of roots inside an interval considered

Isolation and refinement of roots of polynomials

- Techniques based on counting the number of roots inside an interval considered
  - Sturm Theorem
  - Descartes' Rule of Signs

Isolation and refinement of roots of polynomials

- Techniques based on counting the number of roots inside an interval considered
  - Sturm Theorem
  - Descartes' Rule of Signs
- Use a bisection strategy for isolating the roots

Isolation and refinement of roots of polynomials

- Techniques based on counting the number of roots inside an interval considered
  - Sturm Theorem
  - Descartes' Rule of Signs
- Use a bisection strategy for isolating the roots
- Use dichotomy or Newton iteration process

Our example - Relative error

Example:

$f(x) = \cos(x)$ over $[-1, 1]$, $p(x) = \sum_{i=0}^{5} c_i x^i$,
$\varepsilon(x) = p(x)/f(x) - 1$

Our example - Relative error

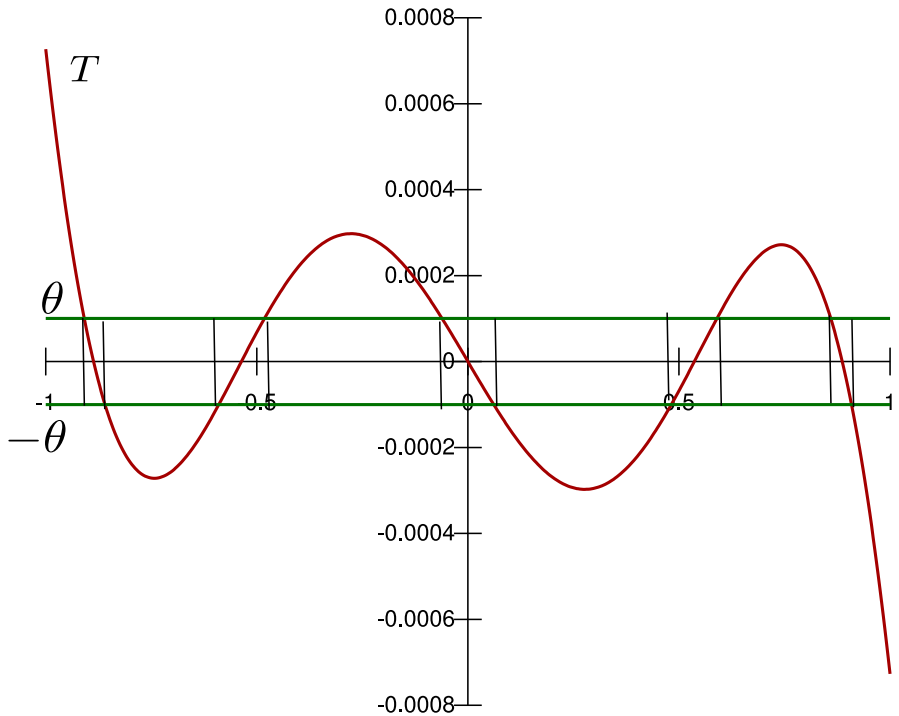
Example:

$f(x) = \cos(x)$ over $[-1, 1]$, $p(x) = \sum_{i=0}^{5} c_i x^i$,
$\varepsilon(x) = p(x)/f(x) - 1$

Compute Taylor Model for $\tau = p'f - pf'$, order 12.

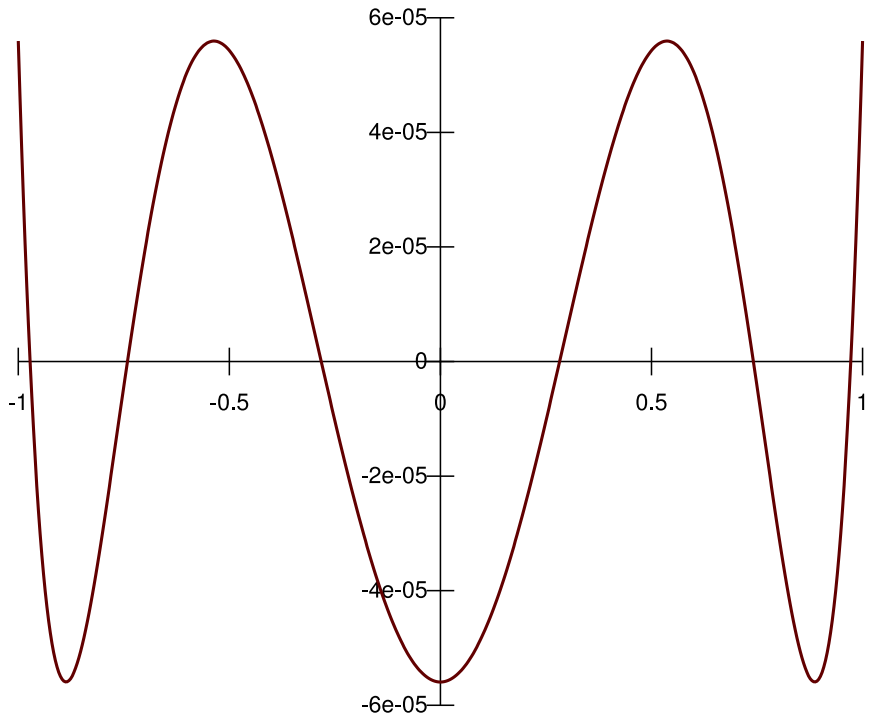
Remainder: $[-1.67352e-7; 1.67352e-7]$

Our example - Relative error



$r_0 \in -0.886[9;8],\ r_1 \in -0.536[6;4],$
$r_2 \in [-0.000001; 0.000001],\ r_3 \in 0.536[4;6],$
$r_4 \in 0.886[8;9]$

Our example - Relative error



$r_0 \in -0.886[9;8],\ r_1 \in -0.536[6;4],$
$r_2 \in [-0.000001; 0.000001],\ r_3 \in 0.536[4;6],$
$r_4 \in 0.886[8;9]$
$\|\varepsilon\|_\infty \in 5.59[3/4] \times e - 5.$

Our Approach - Summary

- Purpose: fast and safely compute the supremum norm $\|f - p\|_\infty$ and $\|p/f - 1\|_\infty$ over an interval $[a, b]$
- Enclose all the zeros of the first derivative of the error
- Evaluate the approximation error on these small intervals only, using multiprecision IA
- Use a Taylor Model based approach to overcome the dependency
- Enclose the zeros of a function using our algorithm

Results

Experiments were made on an Intel Pentium D 3.00GHz with a 2GB RAM.

f	$[a, b]$	d_p[1]	m[2]	acc[3]	time[4]
$\exp(x) - 1$	$[-0.25, 0.25]$	5	r	37.6	412
$\log_2(1 + x)$	$[-2^{-9}, 2^{-9}]$	7	r	83.3	2186
$\cos(x)$	$[-0.5, 0.25]$	15	r	19.5	2235
$\exp(x)$	$[-0.125, 0.125]$	25	r	42.3	7753
$\sin(x)$	$[-0.5, 0.5]$	9	a	21.5	520
$\exp(\cos(x)^2 + 1)$	$[1, 2]$	15	r	25.5	10984
$\tan(x)$	$[0.25, 0.5]$	10	r	26.0	1072
$x^{2.5}$	$[1, 2]$	7	r	15.5	1362

[1] Degree of p

[2] Error mode considered: a=absolute, r=relative

[3] Accuracy

[4] Timings in ms

Conclusion

- Safe and fast algorithm for bounding the supremum norm of the error functions
- Combination and reusal of various techniques (Taylor Models, polynomial roots isolation, interval arith)
- Absolute and Relative errors handled
- Faster and more accurate than other current approaches
- Future works:
  - Formal proof (Taylor Models (AD), multiple precision interval arithmetic are needed in the proof checker)
  - Generalization of the algorithm to multivariate functions.

Questions

Thank you for your attention!

Questions?

Results

Experiments were made on an Intel Pentium D 3.00GHz with a 2GB RAM.

f	$[a, b]$	d_p[1]	m[2]	d_T[3]	acc[4]	time[5]
$\exp(x) - 1$	$[-0.25, 0.25]$	5	r	11	37.6	412
$\log_2(1 + x)$	$[-2^{-9}, 2^{-9}]$	7	r	23	83.3	2186
$\cos(x)$	$[-0.5, 0.25]$	15	r	28	19.5	2235
$\exp(x)$	$[-0.125, 0.125]$	25	r	41	42.3	7753
$\sin(x)$	$[-0.5, 0.5]$	9	a	14	21.5	520
$\exp(\cos(x)^2 + 1)$	$[1, 2]$	15	r	60	25.5	10984
$\tan(x)$	$[0.25, 0.5]$	10	r	21	26.0	1072
$x^{2.5}$	$[1, 2]$	7	r	26	15.5	1362

[1] Degree of p

[2] Error mode considered: a=absolute, r=relative

[3] Degree of T

[4] Accuracy

[5] Timings in ms