Certified and fast computation of supremum norms of approximation errors

Sylvain Chevillard   Mioara Joldes   Christoph Lauter

June 2, 2009

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

"It makes me nervous to fly on airplanes, since I know they are designed using floating-point arithmetic."
A. Householder

Motivation

"It makes me nervous to fly on airplanes, since I know they are designed using floating-point arithmetic."
A. Householder

- Software systems: scientific computing, financial, embedded systems.
- Need to compute $\sin$, $\cos$, $\exp$ in finite precision, floating-point environment.

Motivation

"It makes me nervous to fly on airplanes, since I know they are designed using floating-point arithmetic."
A. Householder

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

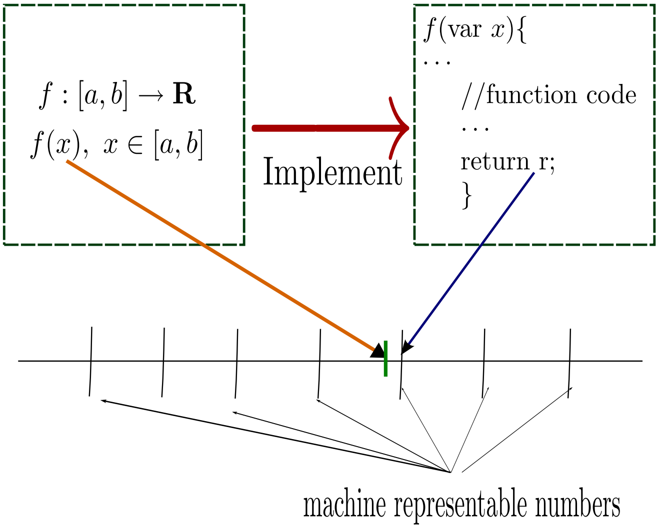
Motivation

"It makes me nervous to fly on airplanes, since I know they are designed using floating-point arithmetic."
A. Householder

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

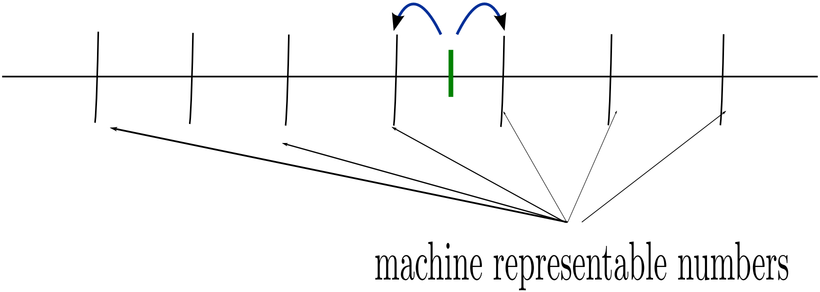
[1] http://lipforge.ens-lyon.fr/www/crlibm/

Correctly rounded functions



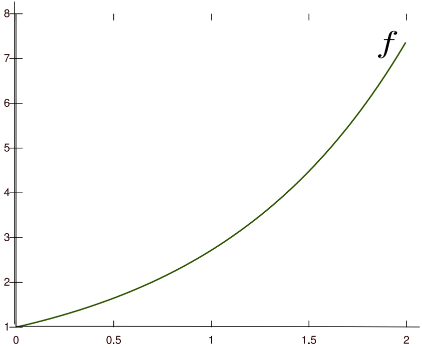
Correctly rounded functions

- Table Maker's Dilemma

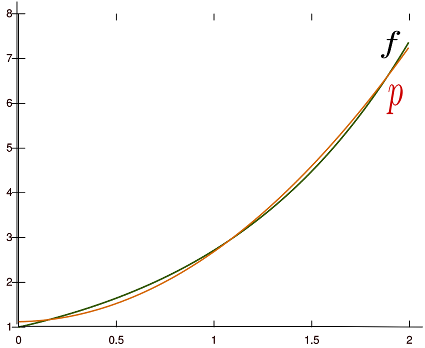


- Increase working precision
- Worst cases search - V. Lefevre and J-M. Muller

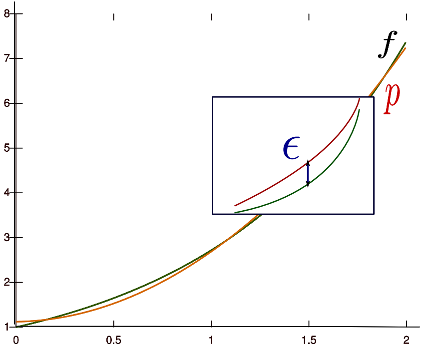
Supremum Norms of Error Functions



Supremum Norms of Error Functions



Supremum Norms of Error Functions

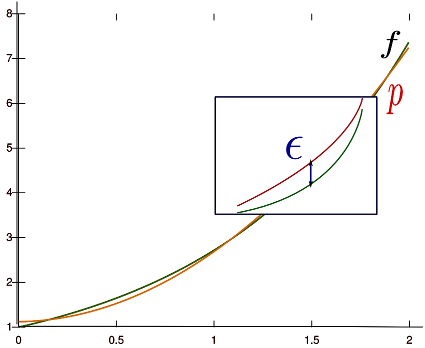


$\varepsilon(x) = f(x) - p(x), \; x \in [a, b]$ or

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

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

Supremum Norms of Error Functions



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

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

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

- Compute a certified bound for the supremum norm of an error function
- “Quick and dirty“ supremum norms - another class of algorithms

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 $\|\varepsilon\|_\infty = \sup_{x\in[a,\,b]} |\varepsilon(x)|$
- Purpose: Compute a certified bound for the supremum norm of an error function
- Given p and f find a narrow interval $\mathbf{r}$ such that $\|\varepsilon\|_\infty \in \mathbf{r}$.

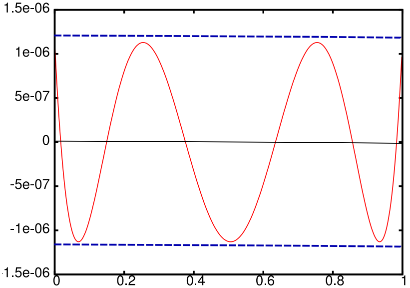
Need for a fast and certified algorithm:

- Correctly rounded elementary functions
- For computing the minimum error between a function and thousands of polynomials with floating-point coefficients

Supremum norm of error functions

Example:

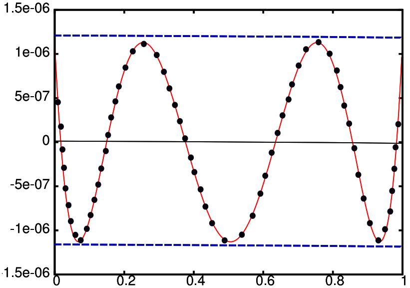
$f(x) = e^x, \; x \in [0, 1]$
$p(x) = \sum_{i=0}^{5} c_i x^i$ s.t. $\|f - p\|_\infty$ is as small as possible (Remez algorithm)
$\varepsilon(x) = f(x) - p(x)$



Supremum norm of error functions

Example:

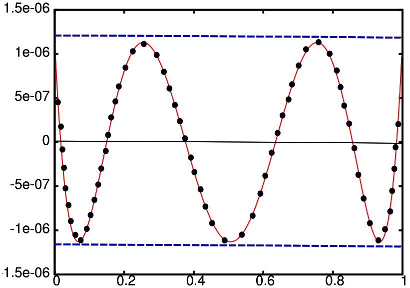
$f(x) = e^x,\ \ x \in [0, 1]$
$p(x) = \sum_{i=0}^{5} c_i x^i$ s.t. $\|f - p\|_\infty$ is as small as possible (Remez algorithm)
$\varepsilon(x) = f(x) - p(x)$



Supremum norm of error functions

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 algorithm)
$\varepsilon(x) = f(x) - p(x)$



How to obtain a certified and tight bound?

Interval Arithmetic

- Each interval = pair of floating-point numbers
- $\pi \in [3.1415, 3.1416]$

Interval Arithmetic

- Each interval = pair of floating-point numbers
- $\pi \in [3.1415, 3.1416]$
- Extend to intervals
  - Arithmetic operations Eg. $[1, 2] + [-3, 2] = [-2, 4]$

Interval Arithmetic

- Each interval = pair of floating-point numbers
- $\pi \in [3.1415, 3.1416]$
- Extend to intervals
  - Arithmetic operations Eg. $[1,2] + [-3,2] = [-2,4]$
  - Functions: Eg. $f(x) = x^2 + x + 1, x \in [-1,2]$

$$
\begin{aligned}
F(X) &= X^2 + X + 1 \\
F([-1,2]) &= [-1,2]^2 + [-1,2] + [1,1] \\
F([-1,2]) &= [0,4] + [-1,2] + [1,1] \\
F([-1,2]) &= [0,7]
\end{aligned}
$$

Interval Arithmetic

- Each interval = pair of floating-point numbers
- $\pi \in [3.1415, 3.1416]$
- Extend to intervals
  - Arithmetic operations Eg. $[1, 2] + [-3, 2] = [-2, 4]$
  - Functions: Eg. $f(x) = x^2 + x + 1, x \in [-1, 2]$

$$\begin{aligned} F(X) &= X^2 + X + 1 \\ F([-1,2]) &= [-1,2]^2 + [-1,2] + [1,1] \\ F([-1,2]) &= [0,4] + [-1,2] + [1,1] \\ F([-1,2]) &= [0,7] \end{aligned}$$

But exact range: $[3/4, 7]$.

Interval Arithmetic

- Each interval = pair of floating-point numbers
- $\pi \in [3.1415, 3.1416]$
- Extend to intervals
  - Arithmetic operations Eg. $[1,2] + [-3,2] = [-2,4]$
  - Functions: Eg. $f(x) = x^2 + x + 1, x \in [-1,2]$

$$
\begin{aligned}
F(X) &= X^2 + X + 1 \\
F([-1,2]) &= [-1,2]^2 + [-1,2] + [1,1] \\
F([-1,2]) &= [0,4] + [-1,2] + [1,1] \\
F([-1,2]) &= [0,7]
\end{aligned}
$$

But exact range: $[3/4, 7]$.
We obtain a certified enclosure: $f(x) \in F(X), \ \forall x \in [-1,2]$.

Interval Arithmetic - Overestimation

$E = x + \varepsilon - x$

1. $\varepsilon \in [-100, 100],\ x \in [-10^{-2}, 10^{-2}]$

$$E \in [-10^{-2}, 10^{-2}] + [-100, 100] - [-10^{-2}, 10^{-2}]$$
$$E \in [-100 - 2 * 10^{-2}, 100 + 2 * 10^{-2}]$$

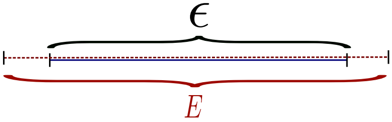
Interval Arithmetic - Overestimation

$E = x + \varepsilon - x$

1. $\varepsilon \in [-100, 100], \; x \in [-10^{-2}, 10^{-2}]$

$$E \in [-10^{-2}, 10^{-2}] + [-100, 100] - [-10^{-2}, 10^{-2}]$$

$$E \in [-100 - 2 * 10^{-2}, 100 + 2 * 10^{-2}]$$

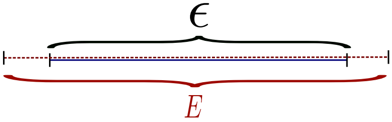


Interval Arithmetic - Overestimation

$E = x + \varepsilon - x$

1. $\varepsilon \in [-100, 100],\ x \in [-10^{-2}, 10^{-2}]$

$$E \in [-10^{-2}, 10^{-2}] + [-100, 100] - [-10^{-2}, 10^{-2}]$$
$$E \in [-100 - 2 * 10^{-2}, 100 + 2 * 10^{-2}]$$



2. $\varepsilon \in [-10^{-2}, 10^{-2}],\ x \in [-100, 100]$

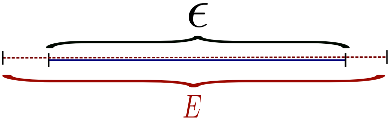
$$E \in [-100, 100] + [-10^{-2}, 10^{-2}] - [-100, 100]$$
$$E \in [-200 - 10^{-2}, 200 + 10^{-2}]$$

Interval Arithmetic - Overestimation

$E = x + \varepsilon - x$

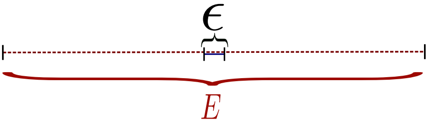
1. $\varepsilon \in [-100, 100],\ x \in [-10^{-2}, 10^{-2}]$

$$E \in [-10^{-2}, 10^{-2}] + [-100, 100] - [-10^{-2}, 10^{-2}]$$
$$E \in [-100 - 2 * 10^{-2}, 100 + 2 * 10^{-2}]$$



2. $\varepsilon \in [-10^{-2}, 10^{-2}],\ x \in [-100, 100]$

$$E \in [-100, 100] + [-10^{-2}, 10^{-2}] - [-100, 100]$$
$$E \in [-200 - 10^{-2}, 200 + 10^{-2}]$$



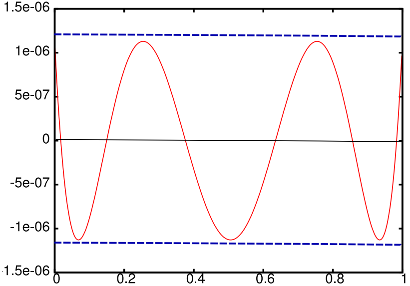
Interval Arithmetic - Overestimation

- When using IA, multiple occurrencies of same variable can not be detected. Dependency Phenomenon
- Overestimation can be reduced using smaller intervals.

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 algorithm)
$\varepsilon(x) = f(x) - p(x)$

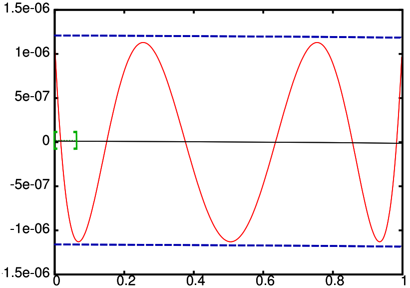


Using IA, $\varepsilon(x) \in [-0.4, 0.4]$, but $\|\varepsilon(x)\|_\infty \simeq 1.1295e - 6$:

Our Problem - Prone to High Dependency Phenomenon

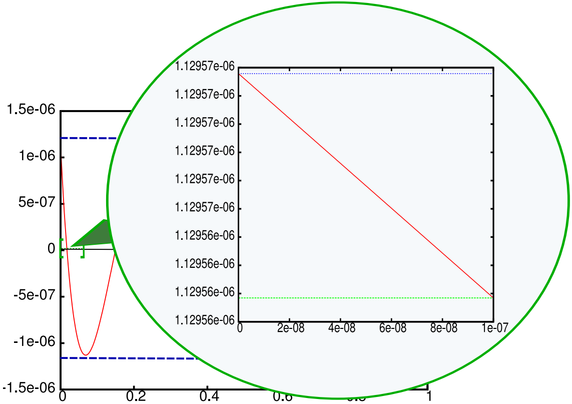
Example:

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



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

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 (eg. Globsol) - not tailored for our specific problem
- Chevillard and Lauter’s technique: interval arithmetic, tight bounding of the zeros of the derivative of the error function, removes false singularities (like $x = 0 \text{ for } \sin(x)/x$). High computation time for $\deg(p) > 10$.
- Techniques based on a high order Taylor expansion of the error function and a sufficiently close bounding of the remainder
- 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 $[a, b]$?
- Use multiprecision IA: MPFI [2]

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

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 $[a, b]$?
- Use multiprecision IA: MPFI [2]
- Use a higher degree approximation polynomial T for the function f
- Compute (T, Δ) s.t $f(x) - T(x) \in \Delta, \forall x \in [a, b]$.

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

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 $[a, b]$?
- Use multiprecision IA: MPFI [2]
- Use a higher degree approximation polynomial T for the function f
- Compute (T, Δ) s.t $f(x) - T(x) \in \Delta, \forall x \in [a, b]$.
- Break the dependency problem in two:

$$\|f - p\|_\infty \leq \underbrace{\|f - T\|_\infty} + \underbrace{\|T - p\|_\infty}$$

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

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 $[a, b]$?
- Use multiprecision IA: MPFI [2]
- Use a higher degree approximation polynomial T for the function f
- Compute (T, Δ) s.t $f(x) - T(x) \in \Delta, \forall x \in [a, b]$.
- Break the dependency problem in two:

$$\|f - p\|_\infty \leq \underbrace{\|f - T\|_\infty}_{\text{bounding a remainder}} + \underbrace{\|T - p\|_\infty}$$

1 Automatic Differentiation/ Taylor Models

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

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 $[a, b]$?
- Use multiprecision IA: MPFI [2]
- Use a higher degree approximation polynomial T for the function f
- Compute (T, Δ) s.t $f(x) - T(x) \in \Delta, \forall x \in [a, b]$.
- Break the dependency problem in two:

$$\|f - p\|_\infty \leq \underbrace{\|f - T\|_\infty}_{\text{bounding a remainder}} + \underbrace{\|T - p\|_\infty}_{\text{tightly bounding a polynomial}}$$

1. Automatic Differentiation/ Taylor Models
2. Tightly bounding the polynomial difference - Roots isolation and refinement techniques

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

How to compute (T, Δ)?

Compute (T, Δ) s.t $f(x) - T(x) \in \Delta, \forall x \in [a, b]$.
How?

- Taylor Formula with Lagrange form of remainder
- By composition of Taylor Models.

How to compute (T, Δ)?

Compute (T, Δ) s.t $f(x) - T(x) \in \Delta, \forall x \in [a, b]$.
How?

- Taylor Formula with Lagrange form of remainder
- By composition of Taylor Models.

How to compute (T, Δ)?

Taylor Formula with Lagrange form of remainder

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

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

How to compute (T, Δ)?

Taylor Formula with Lagrange form of remainder

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

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

How to compute (T, Δ)?

Taylor Formula with Lagrange form of remainder

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

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

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

How to compute (T, Δ)?

Taylor Formula with Lagrange form of remainder

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

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

- Compute coefficients of T: $\dfrac{f^{(i)}(x_0)}{i!}, i = \{0, \dots, n\}$
- Compute Δ as an enclosure for $\dfrac{f^{(n)}(\xi)(x - x_0)^n}{n!}$, when $\xi \in [a, b]$.

Computing (T, Δ) - Automatic differentiation (AD)

- Purpose: Compute values of high order derivatives of f

Computing (T, Δ) - Automatic differentiation (AD)

- Purpose: Compute values of high order derivatives of f
- f usually represented as an expression tree
- Problem: For large values of n, high memory usage for symbolic differentiation.

Computing (T, Δ) - Automatic differentiation (AD)

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

Computing (T, Δ) - Automatic differentiation (AD)

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

Computing (T, Δ) - Automatic differentiation (AD)

- Purpose: Compute values of high order derivatives of f
- f usually represented as an expression tree
- Problem: For large values of n, high memory usage for symbolic differentiation.
- Solution: do the same operations like when evaluating the expression of the derivative, but don't effectively write this expression
- AD: usually developed by program code transformation, operators overloading
- Can be easily extended to work with interval arithmetic
- Compute $\frac{f^{(i)}(x_0)}{i!}, i = \{0, \ldots, n\}$ or an enclosure for $\frac{f^{(i)}(\xi)}{i!}, i = \{0, \ldots, n\}$, when $\xi \in [a, b]$.

Computing (T, Δ) - Automatic differentiation (AD)

Example:

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

Computing (T, Δ) - Automatic differentiation (AD)

Example:

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

Introduce a higher degree polynomial T: Use AD

$$T(x) = \sum_{i=0}^{8} \frac{f^{(i)}(1/2)}{i!}(x - 1/2)^i = \sum_{i=0}^{8} \frac{\exp(1/2)}{i!}(x - 1/2)^i$$

Computing (T, Δ) - Automatic differentiation (AD)

Example:

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

Introduce a higher degree polynomial T: Use AD

$$T(x) = \sum_{i=0}^{8} \frac{f^{(i)}(1/2)}{i!}(x - 1/2)^i = \sum_{i=0}^{8} \frac{\exp(1/2)}{i!}(x - 1/2)^i$$

Compute an enclosure of the remainder: Use AD

$$\Delta_9(x, \xi) = \underbrace{\frac{f^{(9)}(\xi)}{9!}}_{\in \frac{\exp([0,1])}{9!}} \times \underbrace{(x - 1/2)^9}_{\leq 2^{-9}}$$

$|\Delta_9(x, \xi)| \in 1.4630578142[1; 2] \times e - 8$

How to compute (T, Δ)?

Compute (T, Δ) s.t $f(x) - T(x) \in \Delta, \forall x \in [a, b]$.
How?

- Taylor Formula with Lagrange form of remainder
- By composition of Taylor Models.

Computing (T, Δ) - Taylor Models[3] at a Glance

Couple of the form (T, Δ), s.t. $f - T \in \Delta$.

Key idea: Easily define arithmetic operations $+, -, *$

Given: (T_1, Δ_1), (T_2, Δ_2), two TM's of order n.

- Addition: $(T_1 + T_2, \Delta_1 + \Delta_2)$
- Multiplication:
  $((T_1 T_2)_{0..n}, B((T_1 T_2)_{n+1..2n}) + \Delta_1 B(T_2) + \Delta_2 B(T_1))$

[3] Firstly introduced by M. Berz and K.Makino

Computing (T, Δ) - Taylor Models at a Glance

Features:

- Start with trivial TM: $(1, [0, 0])$.
- TM's for elementary functions are defined
- TM's can be recursively defined
- Propagation of error bounds combined with AD

Computing (T, Δ)

- Main idea:

$$\|f-p\|_\infty \leq \underbrace{\|f-T\|_\infty}_{\text{bounding a remainder}} + \underbrace{\|T-p\|_\infty}_{\text{bounding a polynomial}}$$

- Achieved so far:

$$\|\exp - p\|_\infty \leq \underbrace{\|\exp - T\|_\infty}_{\in\ 1.4630578142[1;2]\times e-8} + \underbrace{\|T-p\|_\infty}_{\text{bounding a polynomial}}$$

(2)Bounding the polynomial difference

Purpose: Tightly bound $\|T - p\|_\infty$ over the interval $[a, b]$

Example:

$p(x) = \sum_{i=0}^{5} c_i x^i$

$$T(x) = \sum_{i=0}^{8} \frac{f^{(i)}(1/2)}{i!}(x - 1/2)^i$$

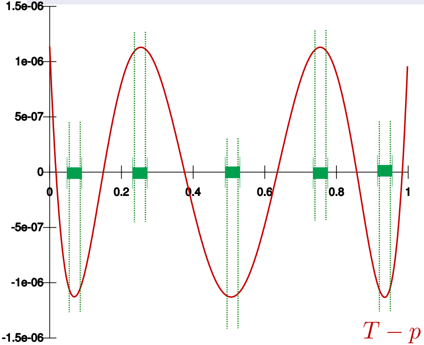
(2)Bounding the polynomial difference

Purpose: Tightly bound $\|T - p\|_\infty$ over the interval $[a, b]$

Example:

$p(x) = \sum_{i=0}^{5} c_i x^i$

$$T(x) = \sum_{i=0}^{8} \frac{f^{(i)}(1/2)}{i!}(x - 1/2)^i$$



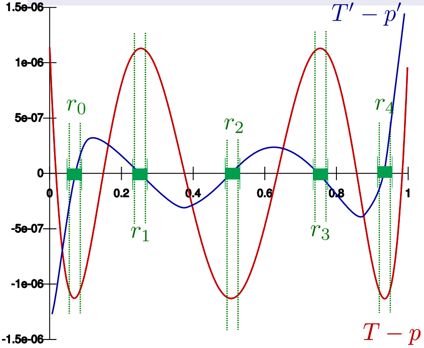
(2)Bounding the polynomial difference

Purpose: Tightly bound $\|T - p\|_\infty$ over the interval $[a, b]$

Example:

$p(x) = \sum_{i=0}^{5} c_i x^i$

$$T(x) = \sum_{i=0}^{8} \frac{f^{(i)}(1/2)}{i!}(x - 1/2)^i$$



- Tightly bound the roots of the derivative

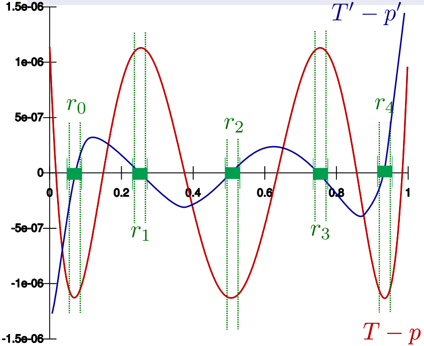
(2)Bounding the polynomial difference

Purpose: Tightly bound $\|T - p\|_\infty$ over the interval $[a, b]$

Example:

$p(x) = \sum_{i=0}^{5} c_i x^i$

$$T(x) = \sum_{i=0}^{8} \frac{f^{(i)}(1/2)}{i!}(x - 1/2)^i$$



- Tightly bound the roots of the derivative
- Evaluate using interval arithmetic

(3)Isolation and refinement of roots of polynomials

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

(3)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

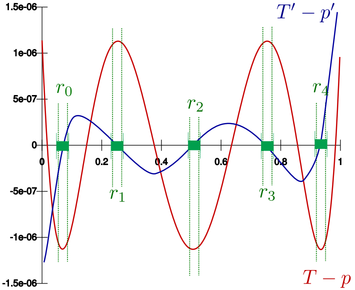
(3)Isolation and refinement of roots of polynomials

- 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

(3)Isolation and refinement of roots of polynomials

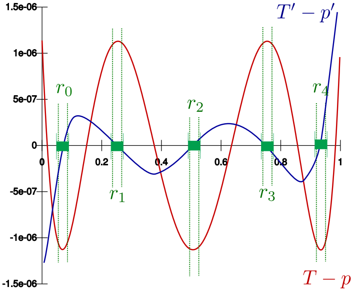
- Techniques based on counting the number of roots inside an interval considered
  - Sturm Theorem
  - Descartes' Rule of Signs
- Use a bisection strategy for isolating the roots
- Use dichotomy or Newton iteration process

(3)Bounding the polynomial difference



$r_0 \in 0.0068[5;6],\ r_1 \in 0.2544[3;4],$
$r_2 \in 0.5059[4;5],\ r_3 \in 0.7544[7;8],$
$r_4 \in 0.9345[8;9]$

(3)Bounding the polynomial difference



$r_0 \in 0.0068[5;6]$, $r_1 \in 0.2544[3;4]$,
$r_2 \in 0.5059[4;5]$, $r_3 \in 0.7544[7;8]$,
$r_4 \in 0.9345[8;9]$

$\|T - p\|_\infty \in \ 1.11486943[1;2] \times e - 6$

Our Approach - Absolute Error - Summary

1 Purpose: fast and safely compute the supremum norm $\|f - p\|_\infty$ over an interval $[a, b]$

Our Approach - Absolute Error - Summary

1. Purpose: fast and safely compute the supremum norm $\|f - p\|_\infty$ over an interval $[a, b]$
2. Introduce a higher degree approximation polynomial: $\|f - p\|_\infty \leq \|f - T\|_\infty + \|T - p\|_\infty$ (AD)

Our Approach - Absolute Error - Summary

1. Purpose: fast and safely compute the supremum norm $\|f - p\|_\infty$ over an interval $[a, b]$
2. Introduce a higher degree approximation polynomial: $\|f - p\|_\infty \leq \|f - T\|_\infty + \|T - p\|_\infty$ (AD)
3. Bound the remainder (AD)

Our Approach - Absolute Error - Summary

1. Purpose: fast and safely compute the supremum norm $\|f - p\|_\infty$ over an interval $[a, b]$
2. Introduce a higher degree approximation polynomial: $\|f - p\|_\infty \leq \|f - T\|_\infty + \|T - p\|_\infty$ (AD)
3. Bound the remainder (AD)
4. Bound the polynomial difference (polynomial roots isolation)

Our Approach - Absolute Error - Summary

1. Purpose: fast and safely compute the supremum norm $\|f - p\|_\infty$ over an interval $[a, b]$
2. Introduce a higher degree approximation polynomial: $\|f - p\|_\infty \leq \|f - T\|_\infty + \|T - p\|_\infty$ (AD)
3. Bound the remainder (AD)
4. Bound the polynomial difference (polynomial roots isolation)
5. Add the two bounds to obtain a tight and safe bound of the approximation error

Our Approach - Absolute Error - Summary

1. Purpose: fast and safely compute the supremum norm $\|f-p\|_\infty$ over an interval $[a,b]$
2. Introduce a higher degree approximation polynomial: $\|f-p\|_\infty \leq \|f-T\|_\infty + \|T-p\|_\infty$ (AD)
3. Bound the remainder (AD)
4. Bound the polynomial difference (polynomial roots isolation)
5. Add the two bounds to obtain a tight and safe bound of the approximation error

Our example:

$$\|f-p\|_\infty \leq \underbrace{\|f-T\|_\infty}_{\in\ 1.4630578142[1;2]\times e-8} + \underbrace{\|T-p\|_\infty}_{\in\ 1.11486943[1;2]\times e-6}$$

$$\|f-p\|_\infty \in\ 1.1295[0;1]\times e-6$$

Our Approach - Absolute Error - Summary

1. Purpose: fast and safely compute the supremum norm $\|f-p\|_\infty$ over an interval $[a,b]$
2. Introduce a higher degree approximation polynomial: $\|f-p\|_\infty \leq \|f-T\|_\infty + \|T-p\|_\infty$ (AD)
3. Bound the remainder (AD)
4. Bound the polynomial difference (polynomial roots isolation)
5. Add the two bounds to obtain a tight and safe bound of the approximation error

Our example:

$$\|f-p\|_\infty \leq \underbrace{\|f-T\|_\infty}_{\in\ 1.4630578142[1;2]\times e-8} + \underbrace{\|T-p\|_\infty}_{\in\ 1.11486943[1;2]\times e-6}$$

$$\|f-p\|_\infty \in\ 1.1295[0;1]\times e-6$$

IA: $\|f-p\|_\infty \leq 0.4$, sampling: $\|f-p\|_\infty \simeq 1.1295e-6$.

Our Approach - Relative Error

- Compute: $\|\varepsilon\|_\infty = \|p/f - 1\|_\infty$ over I

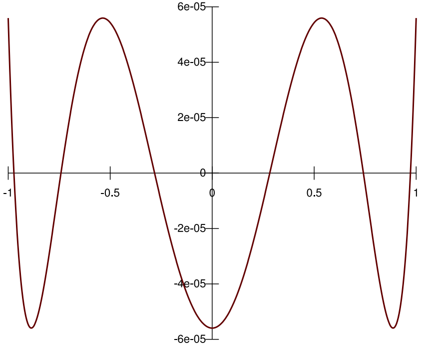
Our Approach - Relative Error

- Compute: $\|\varepsilon\|_\infty = \|p/f - 1\|_\infty$ over I
- Brutal application of the same principle does not work! WHY?

Relative Error Issues

Example:

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



Relative Error Issues

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 the Taylor Model for $\varepsilon(x)$, order 12.

TM for $f(x)$, remainder: $[-1.351322e-10; 1.351322e-10]$
TM for $\frac{1}{f(x)}$, remainder: $[-3.1634304e-2; 0.258216]$

Remainder bounds are unsatisfactory in our case.

Our Approach - Relative Error

- Compute: $\|\varepsilon\|_\infty = \|p/f - 1\|_\infty$ over I
- Brutal application of the same principle does not work:
  - Tight bounds for the polynomials interfering in Taylor Models have to be computed
  - Some compositions (inverse function, logarithm) generate intervals that are too wide in our case

Our Approach - Relative Error

- Compute: $\|\varepsilon\|_\infty = \|p/f - 1\|_\infty$ over I
- Brutal application of the same principle does not work:
  - Tight bounds for the polynomials interfering in Taylor Models have to be computed
  - Some compositions (inverse function, logarithm) generate intervals that are too wide in our case
- Idea: Tightly enclose all the extrema of ε: they are among the zeros of the first derivative.

Our Approach - Relative Error

- Compute: $\|\varepsilon\|_\infty = \|p/f - 1\|_\infty$ over I
- Brutal application of the same principle does not work:
  - Tight bounds for the polynomials interfering in Taylor Models have to be computed
  - Some compositions (inverse function, logarithm) generate intervals that are too wide in our case
- Idea: Tightly enclose all the extrema of ε: they are among the zeros of the first derivative.
- $\varepsilon' = (p'f - pf')/f^2$. Consider $\tau = p'f - pf'$.

Our Approach - Relative Error

- Compute: $\|\varepsilon\|_\infty = \|p/f - 1\|_\infty$ over I
- Brutal application of the same principle does not work:
  - Tight bounds for the polynomials interfering in Taylor Models have to be computed
  - Some compositions (inverse function, logarithm) generate intervals that are too wide in our case
- Idea: Tightly enclose all the extrema of ε: they are among the zeros of the first derivative.
- $\varepsilon' = (p'f - pf')/f^2$. Consider $\tau = p'f - pf'$.
- Solve $\tau(x) = 0$

Our Approach - Relative Error

- Compute: $\|\varepsilon\|_\infty = \|p/f - 1\|_\infty$ over I
- Brutal application of the same principle does not work:
  - Tight bounds for the polynomials interfering in Taylor Models have to be computed
  - Some compositions (inverse function, logarithm) generate intervals that are too wide in our case
- Idea: Tightly enclose all the extrema of ε: they are among the zeros of the first derivative.
- $\varepsilon' = (p'f - pf')/f^2$. Consider $\tau = p'f - pf'$.
- Solve $\tau(x) = 0$
- Find small intervals that enclose each root of τ.

Our Approach - Relative Error

- Compute: $\|\varepsilon\|_\infty = \|p/f - 1\|_\infty$ over I
- Brutal application of the same principle does not work:
    - Tight bounds for the polynomials interfering in Taylor Models have to be computed
    - Some compositions (inverse function, logarithm) generate intervals that are too wide in our case
- Idea: Tightly enclose all the extrema of ε: they are among the zeros of the first derivative.
- $\varepsilon' = (p'f - pf')/f^2$. Consider $\tau = p'f - pf'$.
- Solve $\tau(x) = 0$
- Find small intervals that enclose each root of τ.
- Evaluate ε on these intervals.

Our Approach - Relative Error

- Compute: $\|\varepsilon\|_\infty = \|p/f - 1\|_\infty$ over I
- Brutal application of the same principle does not work:
  - Tight bounds for the polynomials interfering in Taylor Models have to be computed
  - Some compositions (inverse function, logarithm) generate intervals that are too wide in our case
- Idea: Tightly enclose all the extrema of ε: they are among the zeros of the first derivative.
- $\varepsilon' = (p'f - pf')/f^2$. Consider $\tau = p'f - pf'$.
- Solve $\tau(x) = 0$ HOW?
- Find small intervals that enclose each root of τ.
- Evaluate ε on these intervals.

Finding enclosures of roots of a function

- First idea: interval Newton Method BUT Dependency phenomenon present in $\tau = p'f - pf'$ also!

Finding enclosures of roots of a function

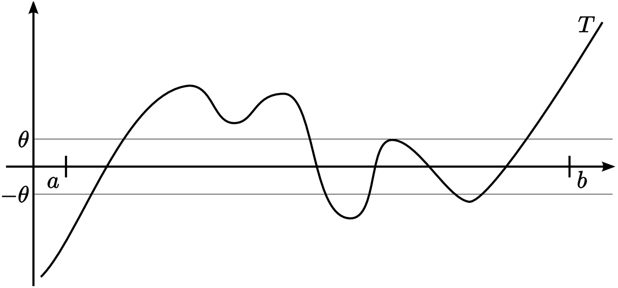
- First idea: interval Newton Method BUT Dependency phenomenon present in $\tau = p'f - pf'$ also!
- Better idea: Compute a Taylor Model $(T, [-\theta, \theta])$ s.t. $\tau(x) - T(x) \in [-\theta, \theta], \forall x \in I$

Finding enclosures of roots of a function

- First idea: interval Newton Method BUT Dependency phenomenon present in $\tau = p'f - pf'$ also!
- Better idea: Compute a Taylor Model $(T, [-\theta, \theta])$ s.t. $\tau(x) - T(x) \in [-\theta, \theta], \forall x \in I$
- $\tau(x) = 0 \implies T(x) \in [-\theta, \theta]$

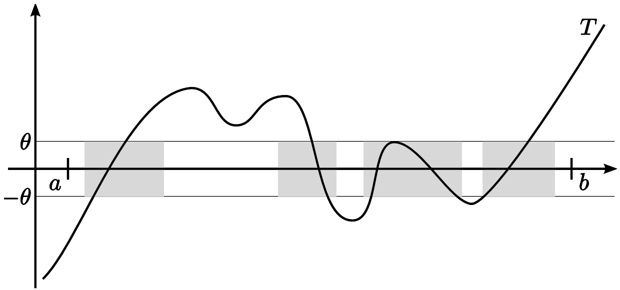
Finding enclosures of roots of a function

- First idea: interval Newton Method BUT Dependency phenomenon present in $\tau = p'f - pf'$ also!
- Better idea: Compute a Taylor Model $(T, [-\theta, \theta])$ s.t. $\tau(x) - T(x) \in [-\theta, \theta], \forall x \in I$
- $\tau(x) = 0 \implies T(x) \in [-\theta, \theta]$



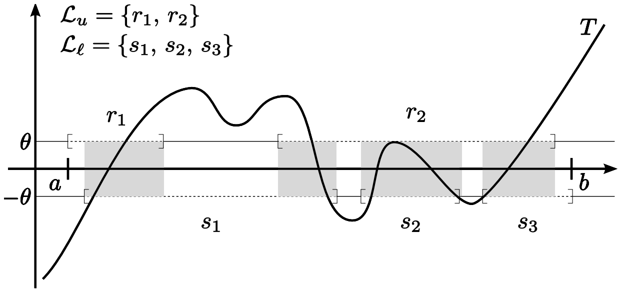
Finding enclosures of roots of a function

- First idea: interval Newton Method BUT Dependency phenomenon present in $\tau = p'f - pf'$ also!
- Better idea: Compute a Taylor Model $(T, [-\theta, \theta])$ s.t. $\tau(x) - T(x) \in [-\theta, \theta], \forall x \in I$
- $\tau(x) = 0 \implies T(x) \in [-\theta, \theta]$



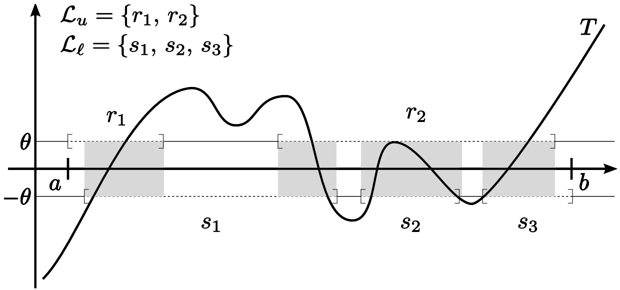
Finding enclosures of roots of a function

- First idea: interval Newton Method BUT Dependency phenomenon present in $\tau = p'f - pf'$ also!
- Better idea: Compute a Taylor Model $(T, [-\theta, \theta])$ s.t. $\tau(x) - T(x) \in [-\theta, \theta], \forall x \in I$
- $\tau(x) = 0 \implies T(x) \in [-\theta, \theta]$
- Compute a list $\mathcal{L}_u$ of intervals where $T - \theta \leq 0$ and $\mathcal{L}_l$ where $T + \theta \geq 0$



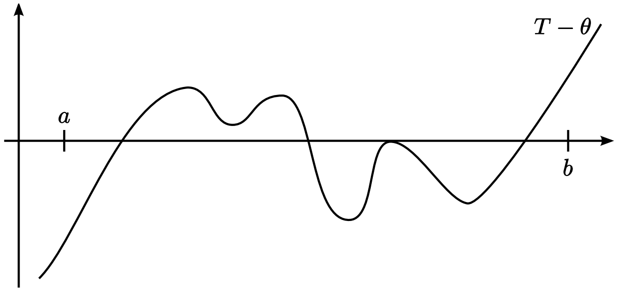
Finding enclosures of roots of a function

- First idea: interval Newton Method BUT Dependency phenomenon present in $\tau = p'f - pf'$ also!
- Better idea: Compute a Taylor Model $(T, [-\theta, \theta])$ s.t. $\tau(x) - T(x) \in [-\theta, \theta], \forall x \in I$
- $\tau(x) = 0 \implies T(x) \in [-\theta, \theta]$
- Compute a list $\mathcal{L}_u$ of intervals where $T - \theta \leq 0$ and $\mathcal{L}_l$ where $T + \theta \geq 0$



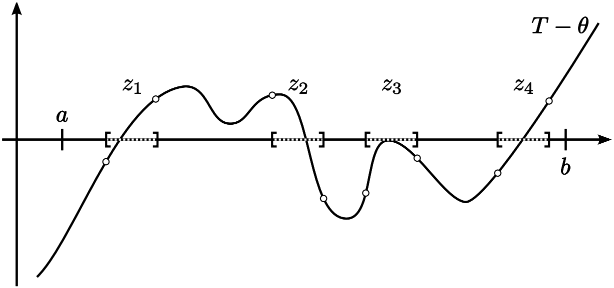
Finding enclosures of roots of a function

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



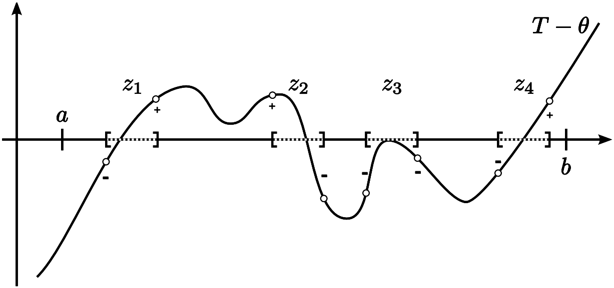
Finding enclosures of roots of a function

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



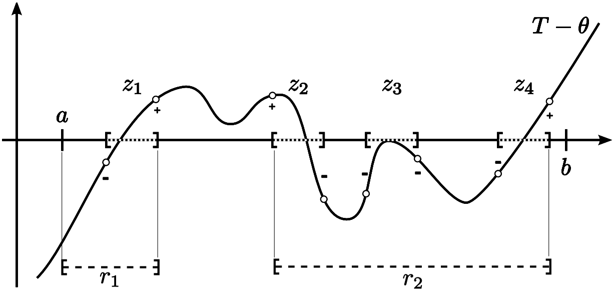
Finding enclosures of roots of a function

- 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



Finding enclosures of roots of a function

- 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



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

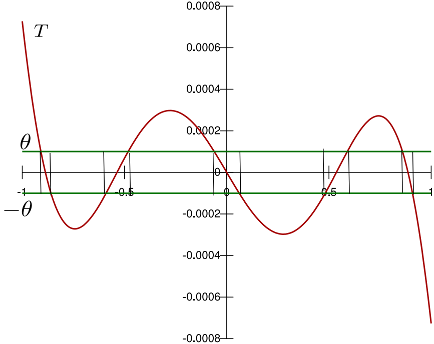
Example:

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

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

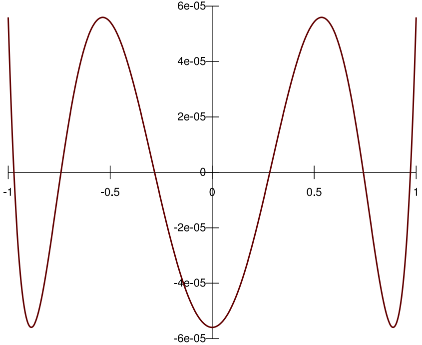
Remainder: $[-1.67352e-7; 1.67352e-7]$

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



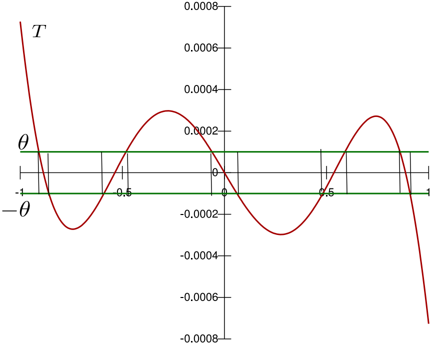
$r_0 \in -0.886[9;8]$, $r_1 \in -0.536[6;4]$,
$r_2 \in [-0.000001; 0.000001]$, $r_3 \in 0.536[4;6]$,
$r_4 \in 0.886[8;9]$
$\|\varepsilon\|_\infty \in 5.59[3/4] \times e - 5$.

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 IA
- Use a Taylor Model based approach to overcome the dependency
- Enclose the zeros of a function using our algorithm

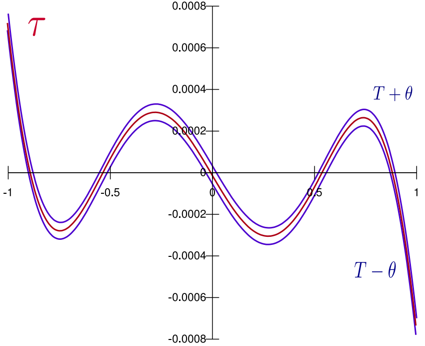
Our example - Relative error - How small should the remainder be?

$\tau(x) - T(x) \in [-\theta, \theta], \forall x \in I$
$\tau(x) = 0 \implies T(x) \in [-\theta, \theta]$



Our example - Relative error - How small should the remainder be?

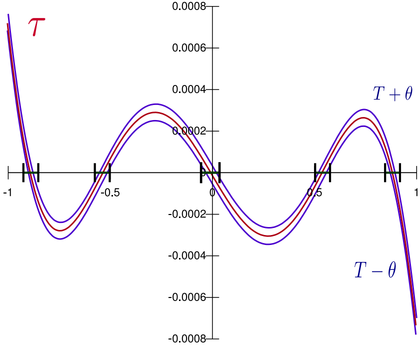
$\tau(x) - T(x) \in [-\theta, \theta], \forall x \in I$
$\tau(x) = 0 \implies T(x) \in [-\theta, \theta]$



Degree: 12

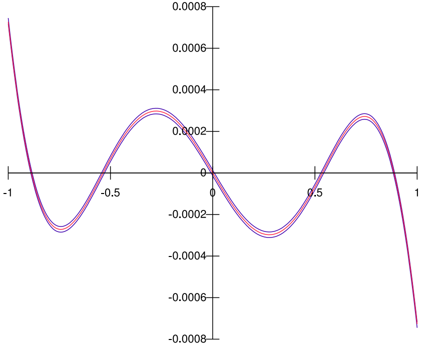
Our example - Relative error - How small should the remainder be?

$\tau(x) - T(x) \in [-\theta, \theta], \forall x \in I$
$\tau(x) = 0 \implies T(x) \in [-\theta, \theta]$



Our example - Relative error - How small should the remainder be?

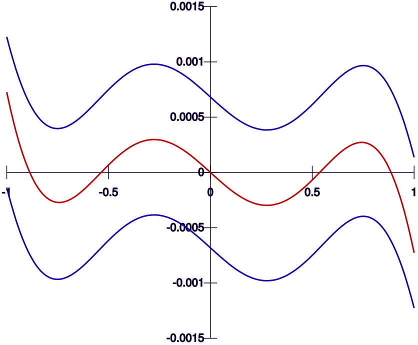
$$\tau(x) - T(x) \in [-\theta, \theta], \forall x \in I$$
$$\tau(x) = 0 \implies T(x) \in [-\theta, \theta]$$



Degree: 14

Our example - Relative error - How small should the remainder be?

$$\tau(x) - T(x) \in [-\theta, \theta], \forall x \in I$$
$$\tau(x) = 0 \implies T(x) \in [-\theta, \theta]$$



Degree: 8

Our example - Relative error - How small should the remainder be?

- Heuristics
- Start with a value close to the approx. max value of τ
- In our example: $\theta_i = 6.44e - 4$

θ	degree T	$\|\varepsilon\|_\infty$
θ_i	8	$[5.59e-5; 1.84]$
$\theta_i/2$	9	$[5.59e-5; 1e-3]$
$\theta_i/10$	10	$[5e-5; 8e-5]$
$\theta_i/10^2$	11	$0.5[5; 7]e-4$
$\theta_i/10^3$	12	$0.559[3; 4]e-4$
$\theta_i/10^4$	13	$0.55935[2; 4]e-4$
$\theta_i/10^5$	14	$0.5593528[2; 4]e-4$

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	2, 186
$\cos(x)$	$[-0.5,\, 0.25]$	15	r	19.5	2, 235
$\exp(x)$	$[-0.125,\, 0.125]$	25	r	42.3	7, 753
$\sin(x)$	$[-0.5,\, 0.5]$	9	a	21.5	520
$\exp(\cos(x)^2 + 1)$	$[1,\, 2]$	15	r	25.5	10, 984
$\tan(x)$	$[0.25,\, 0.5]$	10	r	26.0	1, 072
$x^{2.5}$	$[1,\, 2]$	7	r	15.5	1, 362

[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 (TM, AD, polynomial roots isolation, interval arith)
- Absolute and Relative errors handled
- Faster and more accurate than other current approaches
- Future works:
  - Formal proof (AD, isolation of roots, 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	2, 186
$\cos(x)$	$[-0.5, 0.25]$	15	r	28	19.5	2, 235
$\exp(x)$	$[-0.125, 0.125]$	25	r	41	42.3	7, 753
$\sin(x)$	$[-0.5, 0.5]$	9	a	14	21.5	520
$\exp(\cos(x)^2 + 1)$	$[1, 2]$	15	r	60	25.5	10, 984
$\tan(x)$	$[0.25, 0.5]$	10	r	21	26.0	1, 072
$x^{2.5}$	$[1, 2]$	7	r	26	15.5	1, 362

[1] Degree of p

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

[3] Degree of T

[4] Accuracy

[5] Timings in ms