Certified and fast computation of supremum norms of approximation errors

Sylvain Chevillard   Mioara Joldes   Christoph Lauter

October 24, 2008

Outline

Motivation

- Mathematical Libraries
- Correctly rounded elementary functions
- Supremum norm of error functions
- Previous approaches and difficulties

Outline

Motivation

- Mathematical Libraries
- Correctly rounded elementary functions
- Supremum norm of error functions
- Previous approaches and difficulties

Our approach

- Safe and fast supremum norm of approximation errors
- Automatic differentiation
- Isolation of roots of polynomials

Outline

Motivation

- Mathematical Libraries
- Correctly rounded elementary functions
- Supremum norm of error functions
- Previous approaches and difficulties

Our approach

- Safe and fast supremum norm of approximation errors
- Automatic differentiation
- Isolation of roots of polynomials

Results & Conclusion

Mathematical Libraries (libms)

- What: Compute sin, cos, exp and their inverses

Mathematical Libraries (libms)

- What: Compute sin, cos, exp and their inverses
- Why: Software systems: scientific computing, financial, embedded systems

Mathematical Libraries (libms)

- What: Compute sin, cos, exp and their inverses
- Why: Software systems: scientific computing, financial, embedded systems
- How: Finite precision, floating-point environment

Mathematical Libraries (libms)

- What: Compute sin, cos, exp and their inverses
- Why: Software systems: scientific computing, financial, embedded systems
- How: Finite precision, floating-point environment
- Problem: Most of them do not provide correctly rounded functions. This can ruin portability. IEEE-754 standard revision (June 2008) recommends correct rounding.

Mathematical Libraries (libms)

- What: Compute sin, cos, exp and their inverses
- Why: Software systems: scientific computing, financial, embedded systems
- How: Finite precision, floating-point environment
- Problem: Most of them do not provide correctly rounded functions. This can ruin portability. IEEE-754 standard revision (June 2008) recommends correct rounding.
- Arenaire team develops the Correctly Rounded Libm (CRLibm)[1].

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

Mathematical Libraries (libms)

- Need: Correctly rounded functions
- Given $f : [a, b] \to \mathbb{R}$ and $x \in [a, b]$, the function code should always return the machine number closest to the exact value $f(x)$
- Use an approximation polynomial p of f over $[a, b]$, consider $\varepsilon = f - p \quad \text{or} \quad \varepsilon = \frac{p}{f} - 1$
- Given μ, basic condition to assure: $\forall x \in [a,\ b],\ |\varepsilon(x)| \leq \mu$

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 an interval $\mathbf{r}$ (as thin as desired) such that $\|\varepsilon\|_\infty \in \mathbf{r}$.

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 an interval $\mathbf{r}$ (as thin as desired) 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

Interval Arithmetic

- Each interval = pair of floating-point numbers

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

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: Range Bounding — $f(x) = x^2 + x + 1, x \in [-1, 2]$

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: Range Bounding	$f(x) = x^2 + x + 1, x \in [-1,2]$
All variables are replaced by intervals	$F(X) = X^2 + X + 1$

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: Range Bounding — $f(x) = x^2 + x + 1, x \in [-1,2]$

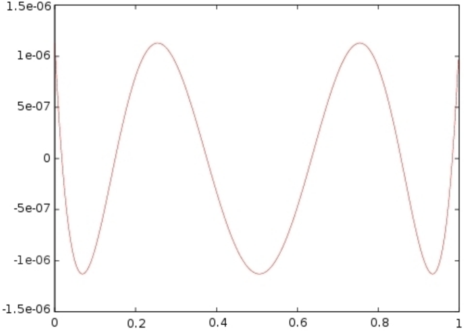
All variables are replaced by intervals — $F(X) = X^2 + X + 1$

Evaluate using interval arithmetic —

$$F([-1,2]) = [-1,2]^2 + [-1,2] + [1,1]$$
$$F([-1,2]) = [0,4] + [-1,2] + [1,1]$$
$$F([-1,2]) = [0,7]$$

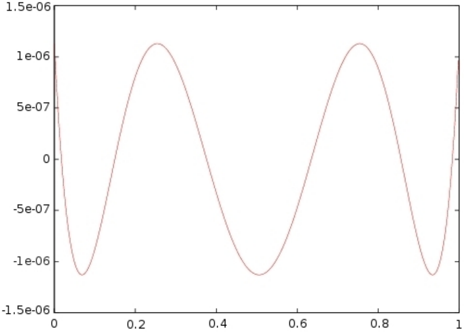
Motivation - Supremum norm of error functions

- Specific problem: supremum norm of error function
- Let $f(x) = e^x, x \in [-1, 1]$
- We are given p with real coefficients, $\deg p \leq 5$ s.t. $\|f - p\|_\infty$ is as small as possible (Remez algorithm)



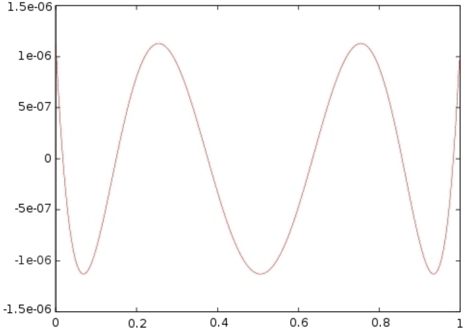
Motivation

- Specific problem: supremum norm of error function



Motivation

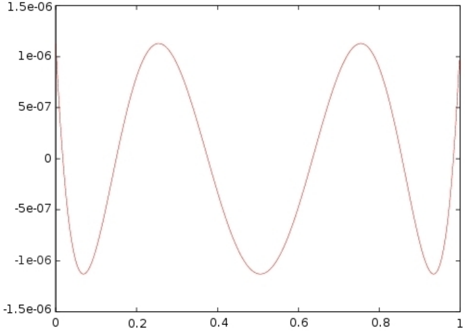
- Specific problem: supremum norm of error function



- Degenerate problem:
  - Cancellation

Motivation

- Specific problem: supremum norm of error function



- Degenerate problem:
  - Cancellation
  - Dependence

Motivation - Difficulties

- Specific problem: supremum norm of error function
- Dependence -

Motivation - Difficulties

- Specific problem: supremum norm of error function
- Dependence - Interval arithmetic is “blind“!
  $x \in [-2, 2], f(x) = x - x$, computing using interval extension,
  $F(X) = [-2, 2] - [-2, 2] = [-4, 4] \neq [0, 0]$

Motivation - Difficulties

- Specific problem: supremum norm of error function
- Dependence - Interval arithmetic is “blind“!
  $x \in [-2, 2], f(x) = x - x,$ computing using interval extension,
  $F(X) = [-2, 2] - [-2, 2] = [-4, 4] \neq [0, 0]$
- Can be reduced by using very small intervals

Previous Approaches

- Floating-point techniques (Brent) - not “safe“

Previous Approaches

- Floating-point techniques (Brent) - not “safe“
- Global optimization software (eg. Globsol) - not tailored for our specific problem

Previous Approaches

- Floating-point techniques (Brent) - not “safe“
- 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 $x = 0$ for $\sin(x)/x$). High computation time for $\deg(p) > 10$.

Previous Approaches

- Floating-point techniques (Brent) - not “safe“
- 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 $x = 0$ 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
  - Sum of squares algorithms (Harrison)
  - The remainder is computed manually

Our Approach

- Compute: $\|\varepsilon\|_\infty = \|f - p\|_\infty$

Our Approach

- Compute: $\|\varepsilon\|_\infty = \|f - p\|_\infty$
- Example:

Consider:

$f(x) = \exp(x)$ over $[0, 1]$
$p(x) = 1.008756 + x * (0.85474264 + x * 0.84602707)$

Our Approach

- Compute: $\|\varepsilon\|_\infty = \|f - p\|_\infty$
- Example:

Consider:

$f(x) = \exp(x)$ over $[0, 1]$
$p(x) = 1.008756 + x * (0.85474264 + x * 0.84602707)$

- Using interval arithmetic we obtain: $\|f - p\|_\infty \leq 0.2836$

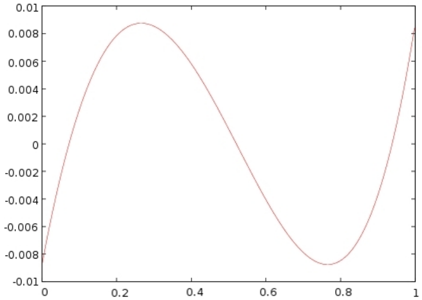
Our Approach

- Compute: $\|\varepsilon\|_\infty = \|f - p\|_\infty$
- Example:

Consider:

$f(x) = \exp(x)$ over $[0, 1]$
$p(x) = 1.008756 + x * (0.85474264 + x * 0.84602707)$

- Using interval arithmetic we obtain: $\|f - p\|_\infty \leq 0.2836$
- By sampling we obtain: $\|f - p\|_\infty \simeq 0.008756064$



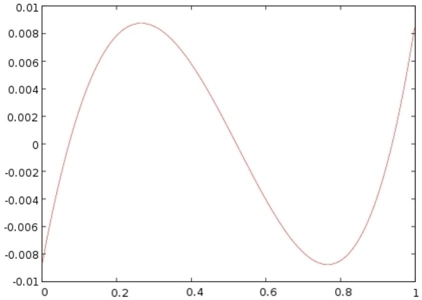
Our Approach

- Compute: $\|\varepsilon\|_\infty = \|f - p\|_\infty$
- Example:

Consider:

$f(x) = \exp(x)$ over $[0, 1]$
$p(x) = 1.008756 + x * (0.85474264 + x * 0.84602707)$

- Using interval arithmetic we obtain: $\|f - p\|_\infty \leq 0.2836$
- By sampling we obtain: $\|f - p\|_\infty \simeq 0.008756064$



Certified bound of $\|f - p\|_\infty$

Our Approach

- Compute: $\|\varepsilon\|_\infty = \|f - p\|_\infty$

Our Approach

- Compute: $\|\varepsilon\|_\infty = \|f - p\|_\infty$
- Idea: Use a higher degree approximation polynomial T.

Our Approach

- Compute: $\|\varepsilon\|_\infty = \|f - p\|_\infty$
- Idea: Use a higher degree approximation polynomial T.

$$\|f - p\|_\infty \leq \|f - T\|_\infty + \|T - p\|_\infty$$

1 Compute T

Our Approach

- Compute: $\|\varepsilon\|_\infty = \|f - p\|_\infty$
- Idea: Use a higher degree approximation polynomial T.

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

1 Compute T

Our Approach

- Compute: $\|\varepsilon\|_\infty = \|f - p\|_\infty$
- Idea: Use a higher degree approximation polynomial T.

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

1 Compute T

Our Approach

- Compute: $\|\varepsilon\|_\infty = \|f - p\|_\infty$
- Idea: Use a higher degree approximation polynomial T.

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

1. Compute T
2. Automatic Differentiation

Our Approach

- Compute: $\|\varepsilon\|_\infty = \|f - p\|_\infty$
- Idea: Use a higher degree approximation polynomial T.

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

1. Compute T
2. Automatic Differentiation
3. Bounding the polynomial difference - Roots isolation and refinement techniques

Our Approach - (1) Introducing a higher degree approximation polynomial

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

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

Our Approach - (1) Introducing a higher degree approximation polynomial

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

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

Our Approach - (1) Introducing a higher degree approximation polynomial

Let $n \in \mathbb{N}$, n times differentiable function f 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

Our Approach - (1) Introducing a higher degree approximation polynomial

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

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

Issues:

Our Approach - (1) Introducing a higher degree approximation polynomial

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

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

Issues:

- Compute the coeffs of $T(x)$: $\dfrac{f^{(i)}(x_0)}{i!}, i = \{0, \cdots, n-1\}$

Our Approach - (1) Introducing a higher degree approximation polynomial

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

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

Issues:

- Compute the coeffs of $T(x)$: $\dfrac{f^{(i)}(x_0)}{i!}, i = \{0, \cdots, n-1\}$
- Bound the remainder $\Delta_n(x, \xi)$: $\dfrac{f^{(n)}(\xi)}{n!}(x - x_0)^n$, ξ between x and x_0

Our Approach - (1) Introducing a higher degree approximation polynomial

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

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

Issues:

- Compute the coeffs of $T(x)$: $\dfrac{f^{(i)}(x_0)}{i!}, i = \{0, \cdots, n-1\}$
- Bound the remainder $\Delta_n(x, \xi)$: $\dfrac{f^{(n)}(\xi)}{n!}(x - x_0)^n$, ξ between x and x_0
- Compute high order derivatives!

Our Approach - (1) Introducing a higher degree approximation polynomial

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

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

Issues:

- Compute the coeffs of $T(x)$: $\dfrac{f^{(i)}(x_0)}{i!}, i = \{0, \cdots, n-1\}$
- Bound the remainder $\Delta_n(x, \xi)$: $\dfrac{f^{(n)}(\xi)}{n!}(x - x_0)^n$, ξ between x and x_0
- Compute high order derivatives!
  - Symbolic differentiation - inefficient

Our Approach - (1) Introducing a higher degree approximation polynomial

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

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

Issues:

- Compute the coeffs of $T(x)$: $\dfrac{f^{(i)}(x_0)}{i!}, i = \{0, \cdots, n-1\}$
- Bound the remainder $\Delta_n(x, \xi)$: $\dfrac{f^{(n)}(\xi)}{n!}(x - x_0)^n$, ξ between x and x_0
- Compute high order derivatives!
    - Symbolic differentiation - inefficient
    - Automatic Differentiation (AD)

Our Approach - (2) Automatic differentiation (AD)

- Purpose: Compute values of high order derivatives of f

Our Approach - (2) Automatic differentiation (AD)

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

Our Approach - (2) 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.

Our Approach - (2) 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

Our Approach - (2) 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

Our Approach - (2) 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

Our Approach - (2) 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 $\frac{f^{(i)}(x_0)}{i!}, i = \{0, \cdots, n\}$ or an enclosure for $\frac{f^{(i)}(\xi)}{i!}, i = \{0, \cdots, n\}$, when $\xi \in [a, b]$.

Our Approach - (2) Automatic differentiation (AD)

- Example:

$f(x) = \exp(x)$ over $[0, 1]$
$p(x) = 1.008756 + x * (0.85474264 + x * 0.84602707)$

Our Approach - (2) Automatic differentiation (AD)

- Example:

$f(x) = \exp(x)$ over $[0, 1]$
$p(x) = 1.008756 + x * (0.85474264 + x * 0.84602707)$

- Introduce a higher degree polynomial T: Use AD

$$T(x) = \sum_{i=0}^{6} \frac{f^{(i)}(1/2)}{i!}(x - 1/2)^i = \sum_{i=0}^{6} \frac{\exp(1/2)}{i!}(x - 1/2)^i$$

Our Approach - (2) Automatic differentiation (AD)

- Example:

$f(x) = \exp(x)$ over $[0, 1]$
$p(x) = 1.008756 + x * (0.85474264 + x * 0.84602707)$

- Introduce a higher degree polynomial T: Use AD

$$T(x) = \sum_{i=0}^{6} \frac{f^{(i)}(1/2)}{i!}(x - 1/2)^i = \sum_{i=0}^{6} \frac{\exp(1/2)}{i!}(x - 1/2)^i$$

- Compute an enclosure of the remainder: Use AD

$$\Delta_7(x, \xi) = \underbrace{\frac{f^{(7)}(\xi)}{7!}}_{\in \frac{exp([0,1])}{7!}} \times \underbrace{(x - 1/2)^7}_{\leq (1/2)^{(-7)}}$$

$|\Delta_7(x, \xi)| \leq 1.46305781422e - 8$

Our Approach - (2) Automatic differentiation (AD)

- Main idea:

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

Our Approach - (2) Automatic differentiation (AD)

- 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:

$$\|\exp - p\|_\infty \leq \underbrace{\|\exp - T\|_\infty}_{\leq 1.46305781422e-8} + \underbrace{\|T - p\|_\infty}_{\text{bounding a polynomial}}$$

Our Approach - (3)Bounding the polynomial difference

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

Our Approach - (3)Bounding the polynomial difference

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

Example:

$p(x) = 1.008756 + x * (0.85474264 + x * 0.84602707)$

$T(x) = \sum_{i=0}^{6} \frac{f^{(i)}(1/2)}{i!}(x - 1/2)^i$

Our Approach - (3)Bounding the polynomial difference

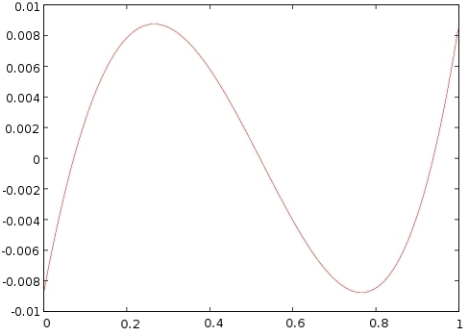
Purpose: Tightly bound $\|T - p\|_\infty$ over the interval $[a, b]$

Example:

$$p(x) = 1.008756 + x * (0.85474264 + x * 0.84602707)$$

$$T(x) = \sum_{i=0}^{6} \frac{f^{(i)}(1/2)}{i!}(x - 1/2)^i$$

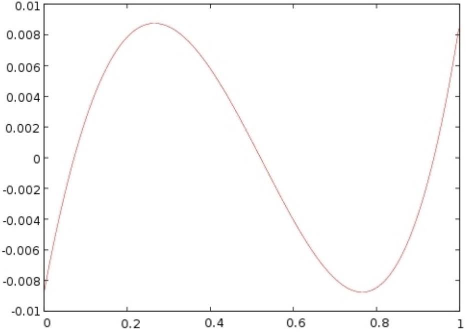
$T(x) - p(x) \Longrightarrow$



Our Approach - (3)Bounding the polynomial difference

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

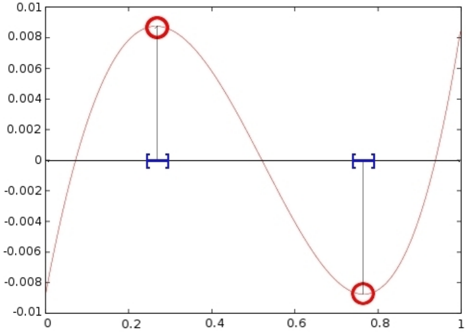
$T(x) - p(x) \quad \Longrightarrow$



Our Approach - (3)Bounding the polynomial difference

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

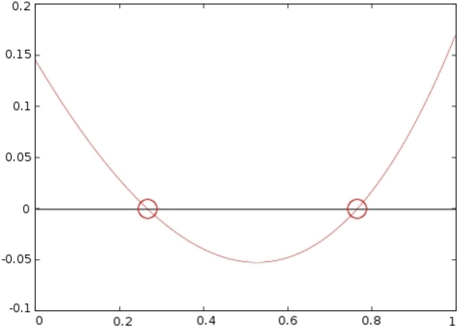
$T(x) - p(x) \quad \Longrightarrow$



Our Approach - (3)Bounding the polynomial difference

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

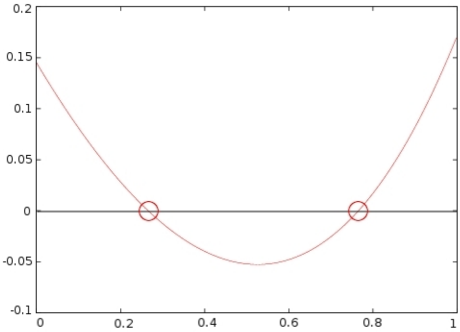
$T'(x) - p'(x)$ $\Longrightarrow$



Our Approach - (3)Bounding the polynomial difference

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

$T'(x) - p'(x) \quad \Longrightarrow$

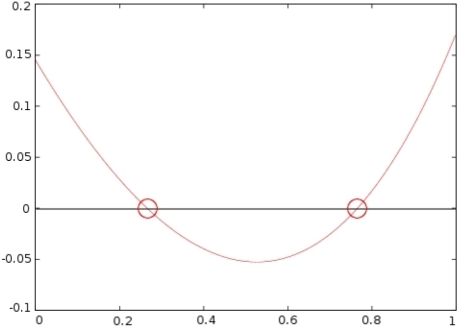


- Tightly bound the roots of the derivative

Our Approach - (3)Bounding the polynomial difference

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

$T'(x) - p'(x)$ $\Longrightarrow$



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

Our Approach - (3)Isolation and refinement of roots of polynomials

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

Our Approach - (3)Isolation and refinement of roots of polynomials

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

Our Approach - (3)Isolation and refinement of roots of polynomials

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

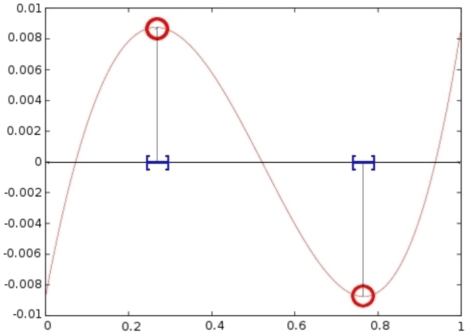
Our Approach - (3)Isolation and refinement of roots of polynomials

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

Our Approach - (3)Bounding the polynomial difference

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

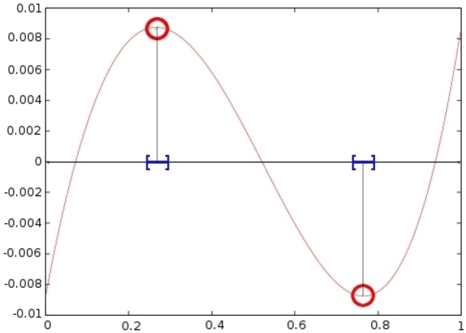
$T(x) - p(x) \quad \Longrightarrow$



Our Approach - (3)Bounding the polynomial difference

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

$T(x) - p(x) \implies$



- Tightly bound the roots of the derivative

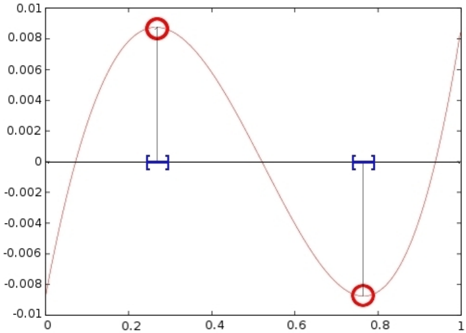
  $r_1 \in [0.2657, 0.2659]$

  $r_2 \in [0.7652, 0.7654]$

Our Approach - (3)Bounding the polynomial difference

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

$T(x) - p(x) \implies$



- Tightly bound the roots of the derivative

  $r_1 \in [0.2657, 0.2659]$

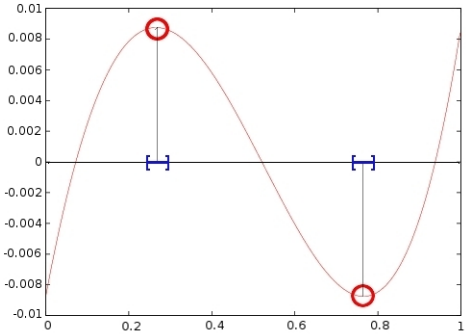
  $r_2 \in [0.7652, 0.7654]$

- Evaluate using interval arithmetic $\|T - p\|_\infty \leq 0.0087566$

Our Approach - (3)Bounding the polynomial difference

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

$T(x) - p(x)$ $\Longrightarrow$



- Tightly bound the roots of the derivative

  $r_1 \in [0.2657, 0.2659]$

  $r_2 \in [0.7652, 0.7654]$

- Evaluate using interval arithmetic

  $\|T - p\|_\infty \leq 0.0087566$

$\|f - p\|_\infty \simeq 0.00875606$

Our Approach - Summary

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

Our Approach - 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 - 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 - 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 - 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 - 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:

$$\|f - p\|_\infty \leq \underbrace{\|f - T\|_\infty}_{\leq 1.46305781422e-8} + \underbrace{\|T - p\|_\infty}_{\leq 0.0087566}$$

Our Approach - 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:

$$\|f-p\|_\infty \leq \underbrace{\|f-T\|_\infty}_{\leq 1.46305781422e-8} + \underbrace{\|T-p\|_\infty}_{\leq 0.0087566}$$

By comparison, using interval arithmetic, we obtain $\|f-p\|_\infty \leq 0.2836$.

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 (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)
  - Replace "Taylor" polynomial with "Chebyshev-like interpolation polynomial"

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