Endpoint and Midpoint Interval Representations Theoretical and Computational Comparison

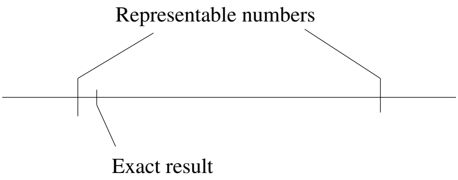
Tomáš Dzetkulič

Institute of Computer Science
Academy of Sciences of the Czech Republic

24th of September 2012

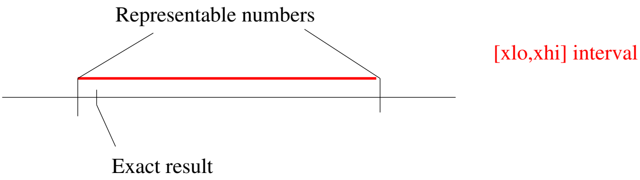
Example

Task: Compute an interval enclosure for $x = 1/15$ based on IEEE standard double precision.



Example

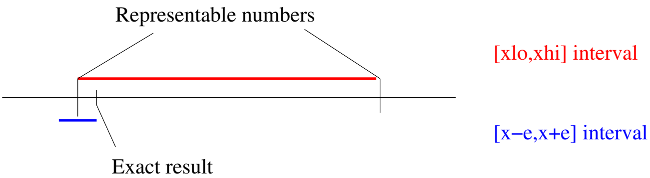
Task: Compute an interval enclosure for $x = 1/15$ based on IEEE standard double precision.



Classical interval analysis:
$x \in [6.6666666666666657415 \times 10^{-2}, 6.6666666666666796193 \times 10^{-2}]$
Width: 1.387779×10^{-17}

Example

Task: Compute an interval enclosure for $x = 1/15$ based on IEEE standard double precision.

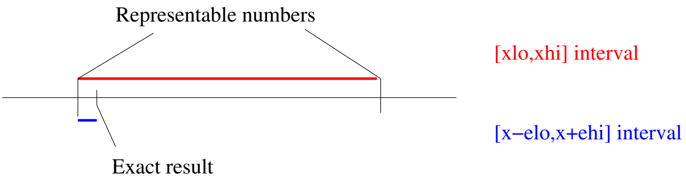


Classical interval analysis:
$x \in [6.66666666666666657415 \times 10^{-2}, 6.66666666666666796193 \times 10^{-2}]$
Width: 1.387779×10^{-17}

Other possible representations:
$6.66666666666666657415 \times 10^{-2} + [-9.252 \times 10^{-19}, 9.252 \times 10^{-19}]$

Example

Task: Compute an interval enclosure for $x = 1/15$ based on IEEE standard double precision.

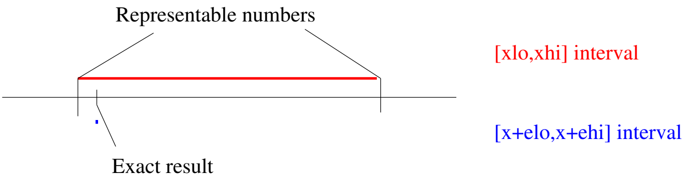


Classical interval analysis:
$x \in [6.666666666666666657415 \times 10^{-2}, 6.666666666666666796193 \times 10^{-2}]$
Width: 1.387779×10^{-17}

Other possible representations:
$6.666666666666666657415 \times 10^{-2} + [-9.252 \times 10^{-19}, 9.252 \times 10^{-19}]$
$6.666666666666666657415 \times 10^{-2} + [0, 9.252 \times 10^{-19}]$

Example

Task: Compute an interval enclosure for $x = 1/15$ based on IEEE standard double precision.



Classical interval analysis:
$x \in [6.666666666666666657415 \times 10^{-2}, 6.666666666666666796193 \times 10^{-2}]$
Width: 1.387779×10^{-17}

Other possible representations:
$6.666666666666666657415 \times 10^{-2} + [-9.252 \times 10^{-19}, 9.252 \times 10^{-19}]$
$6.666666666666666657415 \times 10^{-2} + [0, 9.252 \times 10^{-19}]$

$6.666666666666666657415 \times 10^{-2} + [9.251 \times 10^{-19}, 9.252 \times 10^{-19}]$
Width: $< 10^{-30}$

Interval Types

Let Ω be the set of numbers representable in IEEE double precision format

We consider four kinds of intervals:

1. $[x_{lo}, x_{hi}]$ such that $x_{lo}, x_{hi} \in \Omega$
2. $[x - e, x + e]$ such that $x, e \in \Omega$
3. $[x - e_{lo}, x + e_{hi}]$ such that $x, e_{lo}, e_{hi} \in \Omega$; $e_{lo}, e_{hi} \geq 0$
4. $[x + e_{lo}, x + e_{hi}]$ such that $x, e_{lo}, e_{hi} \in \Omega$

Interval Types

Let Ω be the set of numbers representable in IEEE double precision format

We consider four kinds of intervals:

1. $[x_{lo}, x_{hi}]$ such that $x_{lo}, x_{hi} \in \Omega$
2. $[x - e, x + e]$ such that $x, e \in \Omega$
3. $[x - e_{lo}, x + e_{hi}]$ such that $x, e_{lo}, e_{hi} \in \Omega$; $e_{lo}, e_{hi} \geq 0$
4. $[x + e_{lo}, x + e_{hi}]$ such that $x, e_{lo}, e_{hi} \in \Omega$

Assumptions:

- intervals are narrow
- rounding errors matter

Interval Types

Let Ω be the set of numbers representable in IEEE double precision format

We consider four kinds of intervals:

1. $[x_{lo}, x_{hi}]$ such that $x_{lo}, x_{hi} \in \Omega$
2. $[x - e, x + e]$ such that $x, e \in \Omega$
3. $[x - e_{lo}, x + e_{hi}]$ such that $x, e_{lo}, e_{hi} \in \Omega; e_{lo}, e_{hi} \geq 0$
4. $[x + e_{lo}, x + e_{hi}]$ such that $x, e_{lo}, e_{hi} \in \Omega$

Assumptions:

- intervals are narrow
- rounding errors matter

$\oplus, \otimes$ are *round to nearest, ties to even* addition and multiplication

$\overline{+}, \underline{+}$ denote operations rounded up/down

IEEE Floating Point Format

binary64 number (double precision) stored as:

- 1 bit for sign s
- 11 bit exponent e
- 52 bit mantissa m

It most cases it represents number:

$$(-1)^s \times (1 + 2^{-52} m) \times 2^{(e-1023)}$$

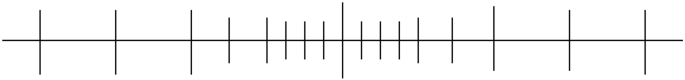
IEEE Floating Point Format

binary64 number (double precision) stored as:

- 1 bit for sign s
- 11 bit exponent e
- 52 bit mantissa m

It most cases it represents number:

$$(-1)^s \times (1 + 2^{-52} m) \times 2^{(e-1023)}$$



0

$\rightarrow$ If mantissa was 4 bits long, then binary numbers 100100.0 and 0.001001 are exactly representable, but the number 100100.001001 is not representable.

Computing With Midpoint Intervals

Dekker(1971) showed that given $a, b \in \Omega$
$(a \oplus b) - (a + b) \in \Omega$ and $(a \otimes b) - (a \times b) \in \Omega$
(if there was not overflow or underflow)

i.e., the exact result of the arithmetic operation can be given as an unevaluated sum of two floating point numbers

Computing With Midpoint Intervals

Dekker(1971) showed that given $a, b \in \Omega$
$(a \oplus b) - (a + b) \in \Omega$ and $(a \otimes b) - (a \times b) \in \Omega$
(if there was not overflow or underflow)

i.e., the exact result of the arithmetic operation can be given as an unevaluated sum of two floating point numbers

Let $\mathtt{add}(a, b) = (\mathtt{res}, \mathtt{err})$ be a function such that $\mathtt{res} = a \oplus b$ and $a + b = \mathtt{res} + \mathtt{err}$

Computing With Midpoint Intervals

Dekker(1971) showed that given $a, b \in \Omega$
$(a \oplus b) - (a + b) \in \Omega$ and $(a \otimes b) - (a \times b) \in \Omega$
(if there was not overflow or underflow)

i.e., the exact result of the arithmetic operation can be given as an unevaluated sum of two floating point numbers

Let $\mathtt{add}(a, b) = (\mathtt{res}, \mathtt{err})$ be a function such that $\mathtt{res} = a \oplus b$ and $a + b = \mathtt{res} + \mathtt{err}$

We can then compute $[x_1 - e_1, x_1 + e_1] + [x_2 - e_2, x_2 + e_2]$:
1. $(x, e_3) := \mathtt{add}(x_1, x_2)$
2. $e := e_1 \overline{+} e_2 \overline{+} |e_3|$
3. **return** $[x - e, x + e]$

Addition Theoretical Analysis Summary

Let $a, b \in \mathbb{R}$. Given an intervals enclosing a and b, compute interval enclosing $a + b$.

Addition Theoretical Analysis Summary

Let $a, b \in \mathbb{R}$. Given an intervals enclosing a and b, compute interval enclosing $a + b$.

Without the loss of generality let $a \in [1, 2]$ and $|b| \leq |a|$.

Addition Theoretical Analysis Summary

Let $a, b \in \mathbb{R}$. Given an intervals enclosing a and b, compute interval enclosing $a + b$.

Without the loss of generality let $a \in [1, 2]$ and $|b| \leq |a|$.

Case	$[a_{lo}, a_{hi}]$	$[a - e, a + e]$	$[a + e_{lo}, a + e_{hi}]$
$\|b\| < \epsilon$	2ϵ	$2\|b\| + O(\epsilon^2)$	$O(\epsilon^2)$
$\epsilon \leq \|b\| < 1/2$	2ϵ	$\epsilon + O(\epsilon^2)$	$O(\epsilon^2)$
$1 \leq \|b\|$ and $b < 0$	0	$O(\epsilon^2)$	$O(\epsilon^2)$
$1 \leq \|b\|$ and $b > 0$	2ϵ	$< 2\epsilon + O(\epsilon^2)$	$O(\epsilon^2)$

$\epsilon = 2^{-53}$

Addition With Medium Magnitude Difference

Example: $(1.3) + (1.4 \times 10^{-10})$
Exact result mantissa is longer than allowed by the standard
$\rightarrow$ rounding occurs

Addition With Medium Magnitude Difference

Example: $(1.3) + (1.4 \times 10^{-10})$
Exact result mantissa is longer than allowed by the standard
$\rightarrow$ rounding occurs

In classical interval analysis the bounds of exact result can lie anywhere in between of two representable numbers
The expected error introduced is ϵ for each bound (2ϵ in total)

Addition With Medium Magnitude Difference

Example: $(1.3) + (1.4 \times 10^{-10})$
Exact result mantissa is longer than allowed by the standard
$\rightarrow$ rounding occurs

In classical interval analysis the bounds of exact result can lie anywhere in between of two representable numbers
The expected error introduced is ϵ for each bound (2ϵ in total)

In intervals of the second kind, we compute $(\texttt{res}, \texttt{err}) := \texttt{add}(a, b)$. The expected magnitude of $\texttt{err}$ is $\epsilon/2$

The expected error introduced is $2\texttt{err} = \epsilon$

Implementation Pitfalls and Wide Intervals

Special care has to be taken for underflowing multiplication

Dekker algorithm does not work in that case

Implementation Pitfalls and Wide Intervals

Special care has to be taken for underflowing multiplication

Dekker algorithm does not work in that case

$\rightarrow$ Underflowing results can be enclosed by 0 ± 10^{-200}

Implementation Pitfalls and Wide Intervals

Special care has to be taken for underflowing multiplication

Dekker algorithm does not work in that case

$\rightarrow$ Underflowing results can be enclosed by 0 ± 10^{-200}

In wide intervals as e, e_{lo} and e_{hi} gain magnitude, additional error is introduced in directed rounding of error

Implementation Pitfalls and Wide Intervals

Special care has to be taken for underflowing multiplication

Dekker algorithm does not work in that case

$\rightarrow$ Underflowing results can be enclosed by 0 ± 10^{-200}

In wide intervals as e, e_{lo} and e_{hi} gain magnitude, additional error is introduced in directed rounding of error

Multiplication of wide intervals $[1-1, 1+1] \times [1-1, 1+1]$ yields suboptimal results ($[1-3, 1+3]$)

$\rightarrow$ shift of the interval center is required in intervals of the second kind

Computational Experiments

1. Add 10000 numbers with high magnitude difference
2. Add 10000 numbers with moderate magnitude difference
3. Add 10000 numbers with alternating sign
4. Add 10000 numbers with similar magnitude
5. Multiply 10000 numbers

	$[a, b]$	$[a-e, a+e]$	$[a-e_{lo}, a+e_{hi}]$
Test	Interval width		
1	2.2×10^{-12}	1.6×10^{-27}	8.6×10^{-40}
2	2.2×10^{-12}	1.1×10^{-12}	0.0
3	0.0	0.0	0.0
4	2.2×10^{-11}	1.8×10^{-11}	0.0
Mul Narrow	1.7×10^{-12}	1.1×10^{-12}	8.2×10^{-27}
Mul Wide	1.3644389579658	1.7683310177080	1.3644389579634

Arithmetic Operations Count and Timings

	$[a, b]$	$[a-e, a+e]$	$[a-e_{lo}, a+e_{hi}]$
Addition			
Add	2	8	10
Time(10^9 operations)	$37s$	$48s$	$49s$
Multiplication			
Add	0	14	29
Mul	8	9	22
Time(10^9 operations)	$57s$	$63s$	$114s$

Application to Rigorous Polynomial

Possible storage formats:

1. $\sum_i [a_i, b_i] x^i$
2. $(\sum_i a_i x^i) + [-e, e]$
3. $(\sum_i a_i x^i) + [e_{lo}, e_{hi}]$

Application to Rigorous Polynomial

Possible storage formats:

1. $\sum_i [a_i, b_i] x^i$
2. $(\sum_i a_i x^i) + [-e, e]$
3. $(\sum_i a_i x^i) + [e_{lo}, e_{hi}]$

In case an operation rounds a polynomial coefficient, the error introduced depends also on the value of the monomial
If $x \in [-1, 1]$ then $x^i \in [-1, 1] \rightarrow$ the sign of the error does not matter

Application to Rigorous Polynomial

Possible storage formats:

1. $\sum_i [a_i, b_i] x^i$
2. $(\sum_i a_i x^i) + [-e, e]$
3. $(\sum_i a_i x^i) + [e_{lo}, e_{hi}]$

In case an operation rounds a polynomial coefficient, the error introduced depends also on the value of the monomial
If $x \in [-1, 1]$ then $x^i \in [-1, 1] \rightarrow$ the sign of the error does not matter

In second and third case we need less memory to store polynomial

Conclusion

We have compared three kinds of intervals

Intervals of second and third kind provide tighter enclosures for narrow intervals

Conclusion

We have compared three kinds of intervals

Intervals of second and third kind provide tighter enclosures for narrow intervals

Computational experiments confirm the advantage of midpoint intervals

Conclusion

We have compared three kinds of intervals

Intervals of second and third kind provide tighter enclosures for narrow intervals

Computational experiments confirm the advantage of midpoint intervals

Thank you for you attention.