On Automatically Proving the Correctness of `math.h` Implementations

<u>Wonyeol Lee</u>[1,2] Rahul Sharma[3] Alex Aiken[1]

[1] Stanford University [2] KAIST [3] Microsoft Research

POPL 2018

Our Goal

$$log\, x$$

mathematical
specification

Our Goal

$$log\ x$$

mathematical specification

```
<log>
...
mulpd   %xmm2, %xmm6
addsd   %xmm0, %xmm5
addsd   %xmm7, %xmm3
addsd   %xmm5, %xmm1
...
```

`math.h` implementation

Our Goal

$$\log x$$

mathematical specification

```
<log>
...
mulpd   %xmm2, %xmm6
addsd   %xmm0, %xmm5
addsd   %xmm7, %xmm3
addsd   %xmm5, %xmm1
...
```

`math.h` implementation

Our Goal

infinite bits

$$\log x$$

mathematical specification

fixed bits

```
<log>
...
mulpd   %xmm2, %xmm6
addsd   %xmm0, %xmm5
addsd   %xmm7, %xmm3
addsd   %xmm5, %xmm1
...
```

`math.h` implementation

Our Goal

infinite bits

$\log x$

mathematical specification

$\neq$

fixed bits

```
<log>
...
mulpd   %xmm2, %xmm6
addsd   %xmm0, %xmm5
addsd   %xmm7, %xmm3
addsd   %xmm5, %xmm1
...
```

`math.h` implementation

Our Goal

infinite bits

$$\log x$$

mathematical specification

∃ precision loss

fixed bits

```
<log>
...
mulpd  %xmm2, %xmm6
addsd  %xmm0, %xmm5
addsd  %xmm7, %xmm3
addsd  %xmm5, %xmm1
...
```

`math.h` implementation

Our Goal

infinite bits

$$\log x$$

mathematical specification

∃ precision loss

fixed bits

```
<log>
...
mulpd  %xmm2, %xmm6
addsd  %xmm0, %xmm5
addsd  %xmm7, %xmm3
addsd  %xmm5, %xmm1
...
```

`math.h` implementation

Industry standard implementations of `math.h` claim:

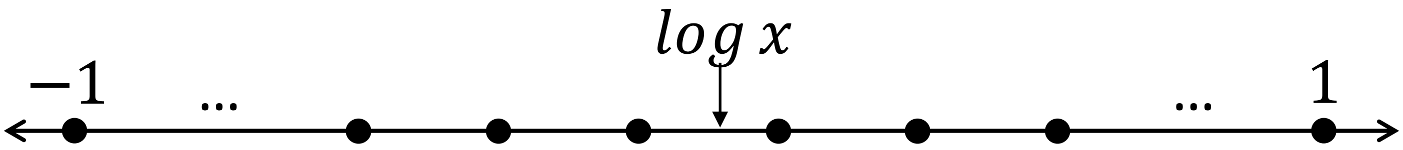
“precision loss is less than 1 ulp”

Our Goal

Industry standard implementations of `math.h` claim:

"precision loss is less than 1 ulp"

Our Goal

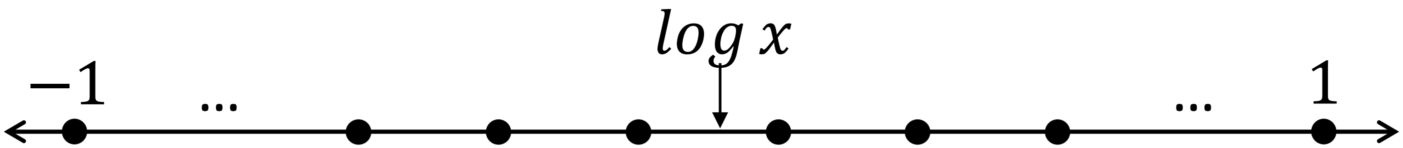


Industry standard implementations of `math.h` claim:

"precision loss is less than 1 ulp"

Our Goal

`log` has precision loss of < 1 ulp:

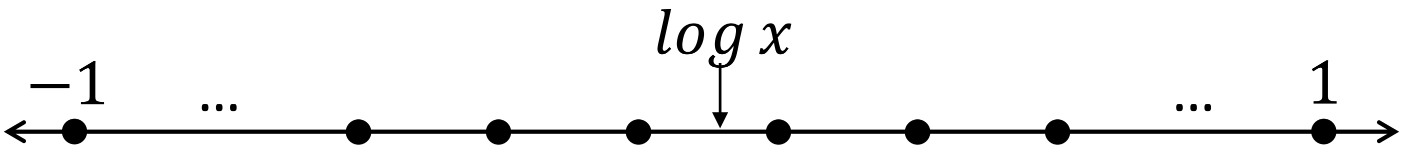


Industry standard implementations of `math.h` claim:

"precision loss is less than 1 ulp"

Our Goal

`log` has precision loss of < 1 ulp:



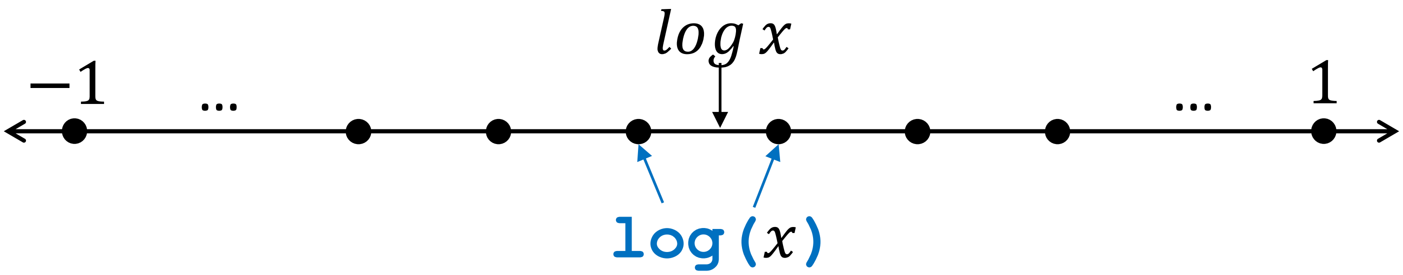
`log(x)`

Industry standard implementations of `math.h` claim:

“precision loss is less than 1 ulp”

Our Goal

`log` has precision loss of < 1 ulp:

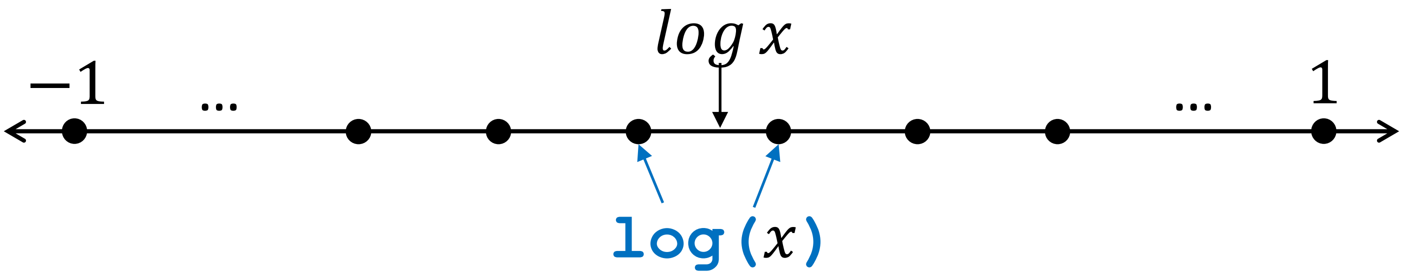


Industry standard implementations of `math.h` claim:

"precision loss is less than 1 ulp"

Our Goal

`log` has precision loss of < 1 ulp:

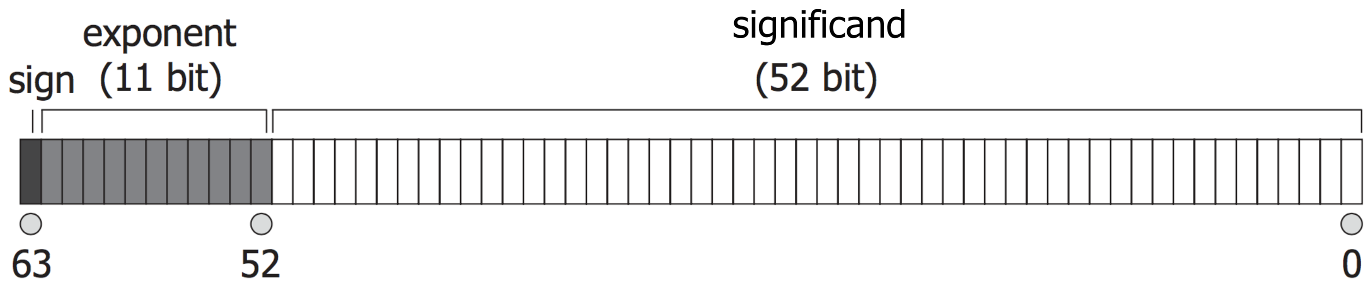


Goal: Prove <u>this claim</u> automatically!

Industry standard implementations of `math.h` claim:

"precision loss is less than 1 ulp"

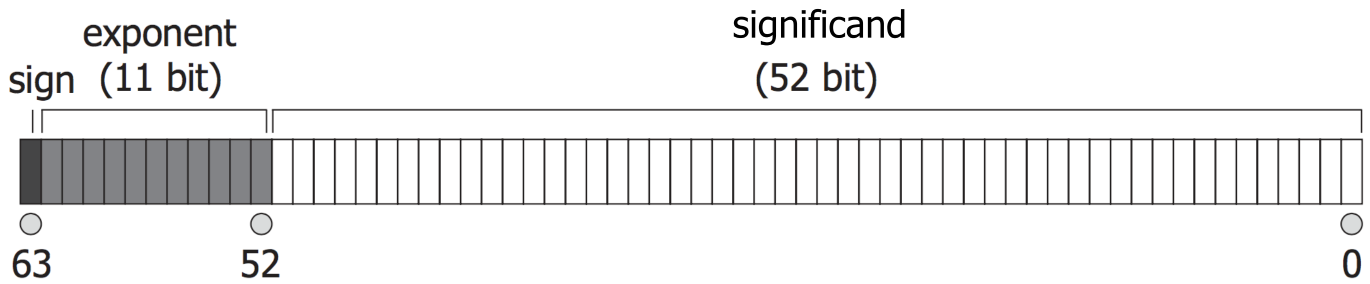
Floating-Point Numbers/Operations



- Example: $\underbrace{1}\ \underbrace{01111111111}\ \underbrace{1100\cdots00}_{(2)}$

$$= (-1)^{1} \cdot 2^{1023-1023} \cdot 1.110\cdots00_{(2)}$$

Floating-Point Numbers/Operations

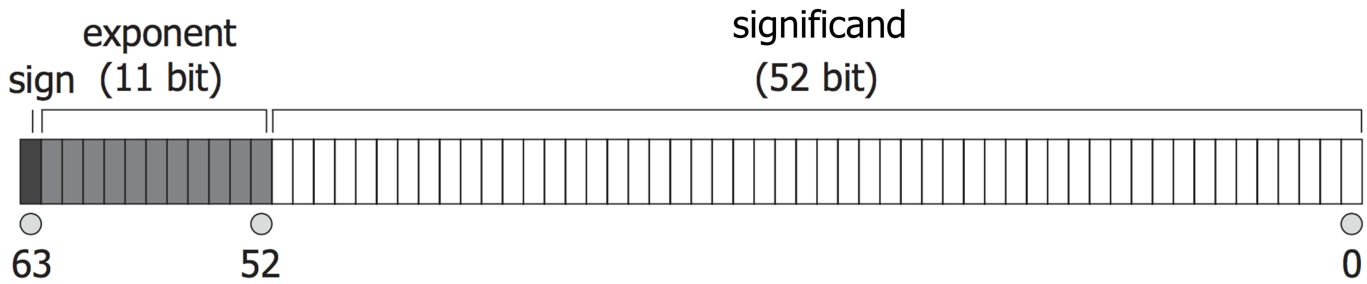


- Example: $\underbrace{1}\ \underbrace{01111111111}\ \underbrace{1100\cdots00}_{(2)}$

$$= (-1)^{1} \cdot 2^{1023-1023} \cdot 1.110\cdots00_{(2)}$$

- Rounding errors

Floating-Point Numbers/Operations



- Example: $1 \; 01111111111 \; 1100\cdots00_{(2)}$

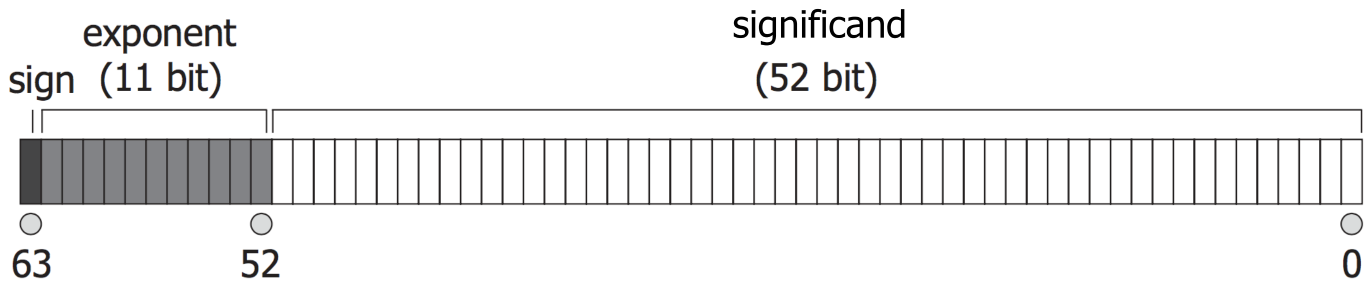
$$= (-1)^{1} \cdot 2^{1023-1023} \cdot 1.110\cdots00_{(2)}$$

- Rounding errors
  - Floating-point arithmetic ≠ Real arithmetic

floating-point operation

$$1 \oplus (2^{100} \ominus 2^{100}) = 1 \neq 0 = (1 \oplus 2^{100}) \ominus 2^{100}$$

Floating-Point Numbers/Operations



- Example: $\underbrace{1}\ \underbrace{01111111111}\ \underbrace{1100\cdots00}_{(2)}$

$$= (-1)^{1} \cdot 2^{1023-1023} \cdot 1.110\cdots00_{(2)}$$

- Rounding errors
  - Floating-point arithmetic ≠ Real arithmetic

floating-point operation

$$1 \oplus (2^{100} \ominus 2^{100}) = 1 \neq 0 = (1 \oplus 2^{100}) \ominus 2^{100}$$

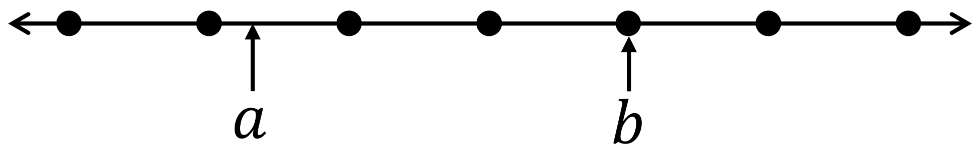
  - Floating-point implementations often have precision loss

Ulp Error

- Typically used to measure accuracy of numeric libraries

Ulp Error

- Typically used to measure accuracy of numeric libraries
- Definition:

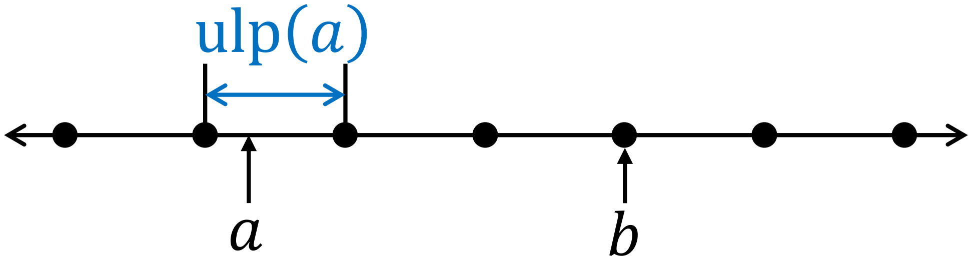


  - $a =$ exact value, $b =$ approximate value

Ulp Error

- Typically used to measure accuracy of numeric libraries

- Definition:

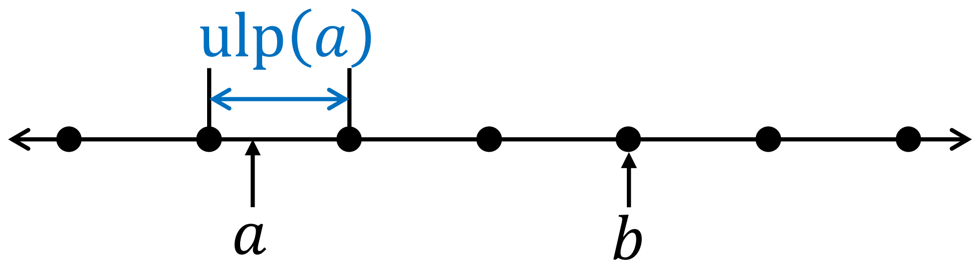


  - a = exact value, b = approximate value
  - $\text{ulp}(a)$ = gap between two doubles that surround a

Ulp Error

- Typically used to measure accuracy of numeric libraries

- Definition:

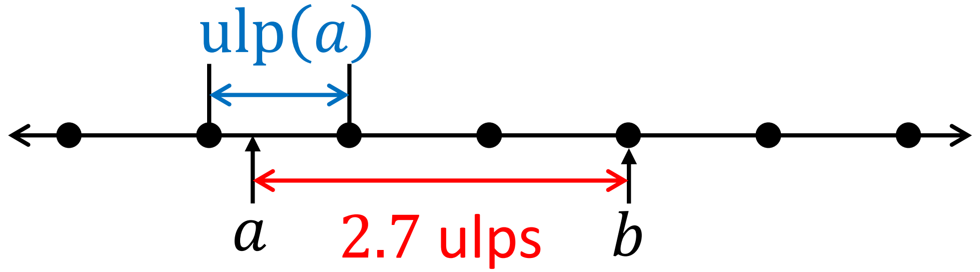


  - a = exact value, b = approximate value
  - ulp(a) = gap between two doubles that surround a

$$\text{ErrUlp}(a, b) = \frac{|a - b|}{\text{ulp}(a)}$$

Ulp Error

- Typically used to measure accuracy of numeric libraries
- Definition:



  - a = exact value, b = approximate value
  - $\mathrm{ulp}(a)$ = gap between two doubles that surround a

$$\mathrm{ErrUlp}(a, b) = \frac{|a - b|}{\mathrm{ulp}(a)}$$

Problem Statement

$$log\ x$$

mathematical specification f

```
<log>
...
mulpd   %xmm2, %xmm6
addsd   %xmm0, %xmm5
addsd   %xmm7, %xmm3
addsd   %xmm5, %xmm1
...
```

`math.h` implementation P

Problem Statement

$log\ x$

mathematical specification f

$(0, 2^{1024})$

input interval X

```
<log>
...
mulpd   %xmm2, %xmm6
addsd   %xmm0, %xmm5
addsd   %xmm7, %xmm3
addsd   %xmm5, %xmm1
...
```

`math.h` implementation P

Problem Statement

$\log x$

mathematical specification f

$(0, 2^{1024})$

input interval X

```
<log>
...
mulpd   %xmm2, %xmm6
addsd   %xmm0, %xmm5
addsd   %xmm7, %xmm3
addsd   %xmm5, %xmm1
...
```

`math.h` implementation P

- Problem: Find a small $\Theta > 0$ automatically such that

$$\mathrm{ErrUlp}(f(x), P(x)) \leq \Theta \quad \text{for all } x \in X$$

Problem Statement

$\log x$

mathematical specification f

$(0, 2^{1024})$

input interval X

```
<log>
...
mulpd   %xmm2, %xmm6
addsd   %xmm0, %xmm5
addsd   %xmm7, %xmm3
addsd   %xmm5, %xmm1
...
```

`math.h` implementation P

- Problem: Find a small $\Theta > 0$ automatically such that

$$\mathrm{ErrUlp}(f(x), P(x)) \leq \Theta \quad \text{for all } x \in X$$

i.e., prove a bound on the maximum precision loss

Problem Statement

$\log x$

mathematical specification f

$(0, 2^{1024})$

input interval X

```
<log>
...
mulpd   %xmm2, %xmm6
addsd   %xmm0, %xmm5
addsd   %xmm7, %xmm3
addsd   %xmm5, %xmm1
...
```

`math.h` implementation P

- Problem: Find a small $\Theta > 0$ automatically such that

$$\mathrm{ErrUlp}(f(x), P(x)) \le \Theta \quad \text{for all } x \in X$$

i.e., prove a bound on the maximum precision loss

Goal: want to find $\Theta < 1$

Previous Work

- Machine-checkable proofs
  - Harrison used HOL Light to prove [FMCAD'00]:

“Intel's `sin` has precision loss of < 0.574 ulps”

Previous Work

- Machine-checkable proofs
  - Harrison used HOL Light to prove [FMCAD'00]:

    "Intel's `sin` has precision loss of < 0.574 ulps"

  - "Construction of these proofs often requires considerable persistence." [FMSD'00]

Previous Work

- Machine-checkable proofs
  - Harrison used HOL Light to prove [FMCAD'00]:

    "Intel's `sin` has precision loss of < 0.574 ulps"

  - "Construction of these proofs often requires considerable persistence." [FMSD'00]

- Automatic techniques
  - Astree [PLDI'03], Fluctuat [FMICS'09], Gappa [TOMS'10], MathSAT [FMCAD'12], Rosa [POPL'14], FPTaylor [FM'15], Lee/Sharma/Aiken [PLDI'16], $\cdots$

Previous Work

- Machine-checkable proofs
  - Harrison used HOL Light to prove [FMCAD'00]:

    "Intel's `sin` has precision loss of < 0.574 ulps"

  - "Construction of these proofs often requires considerable persistence." [FMSD'00]

- Automatic techniques
  - Astree [PLDI'03], Fluctuat [FMICS'09], Gappa [TOMS'10], MathSAT [FMCAD'12], Rosa [POPL'14], FPTaylor [FM'15], Lee/Sharma/Aiken [PLDI'16], $\cdots$
  - None of them can prove < 1 ulp error bound automatically

$(1+\epsilon)$ Property

- A standard way to model rounding errors

$(1+\epsilon)$ Property

- A standard way to model rounding errors
- Theorem For any double a and b, and $* \in \{+, -, \times, /\}$,

$$a \circledast b = (a * b)(1+\delta) \quad \text{for some } |\delta| < \epsilon$$

where $\epsilon = 2^{-53}$

$(1+\epsilon)$ Property

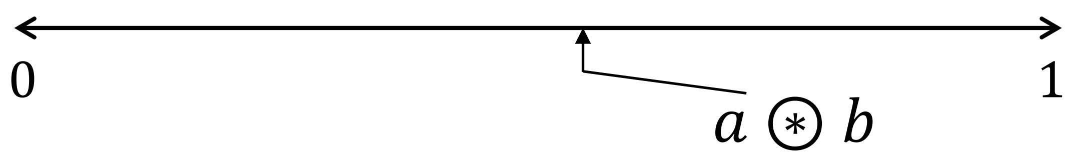
- A standard way to model rounding errors

- Theorem For any double a and b, and $* \in \{+,-,\times,/\}$,

$$a \circledast b = (a * b)(1+\delta) \quad \text{for some } |\delta| < \epsilon$$

where $\epsilon = 2^{-53}$

That is,



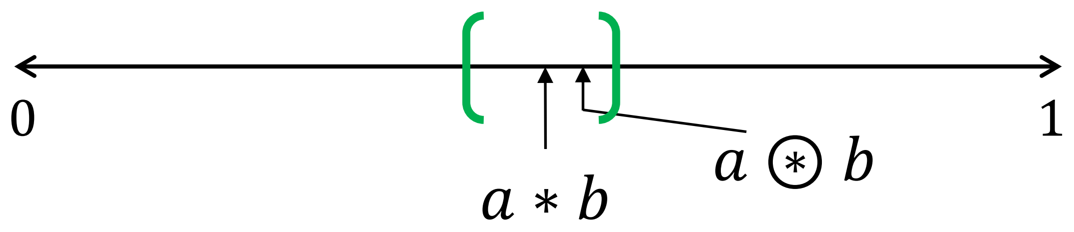
$(1+\epsilon)$ Property

- A standard way to model rounding errors
- <u>Theorem</u> For any double a and b, and $* \in \{+, -, \times, /\}$,

$$a \circledast b = (a * b)(1+\delta) \quad \text{for some } |\delta| < \epsilon$$

where $\epsilon = 2^{-53}$

That is,



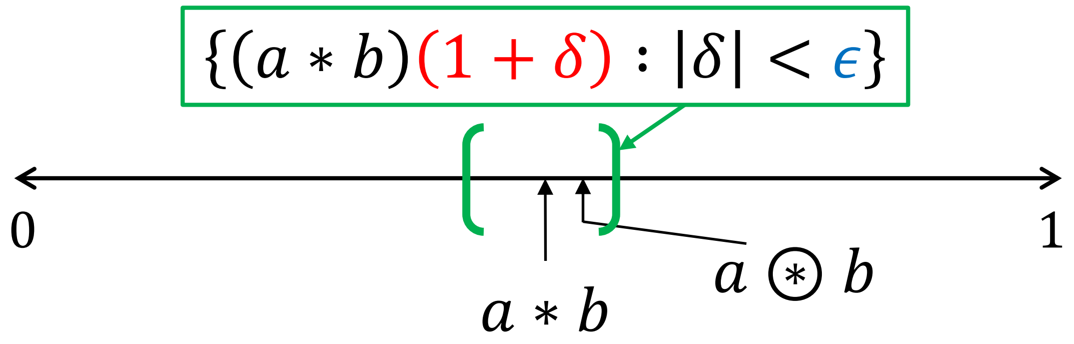
$(1+\epsilon)$ Property

- A standard way to model rounding errors
- Theorem For any double a and b, and $* \in \{+,-,\times,/\}$,

$$a \circledast b = (a * b)(1+\delta) \quad \text{for some } |\delta| < \epsilon$$

where $\epsilon = 2^{-53}$

That is,



$(1+\epsilon)$ Property

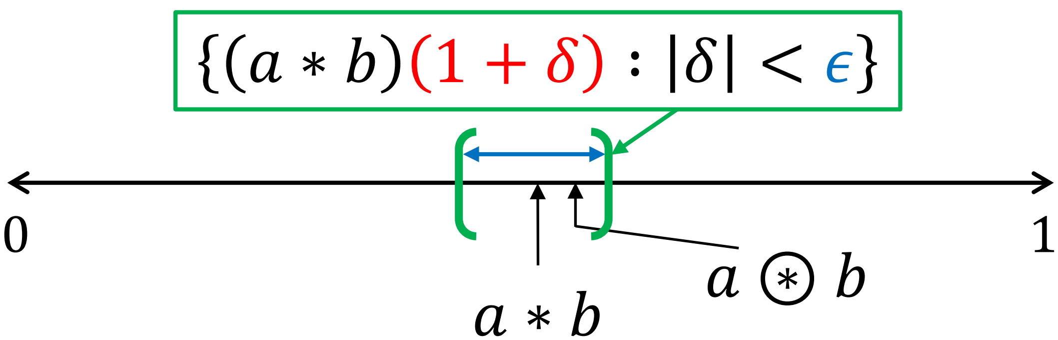
- A standard way to model rounding errors

- <u>Theorem</u> For any double a and b, and $* \in \{+, -, \times, /\}$,

$$a \circledast b = (a * b)(1+\delta) \quad \text{for some } |\delta| < \epsilon$$

where $\epsilon = 2^{-53}$

That is,



Standard Error Analysis

- Example:

$$f(x) = \sqrt{1+x} \quad \text{over} \quad X = [-0.1, 0.1]$$

Standard Error Analysis

- Example:

$$f(x) = \sqrt{1+x} \quad \text{over} \quad X = [-0.1, 0.1]$$

$$e = 1 \oplus (0.5 \otimes x) \quad \longleftarrow \quad \text{implementation of } f(x)$$

Standard Error Analysis

- Example:

$$f(x) = \sqrt{1+x} \quad \text{over} \quad X = [-0.1, 0.1]$$

$$e = 1 \oplus (0.5 \otimes x) \quad \longleftarrow \quad \text{implementation of } f(x)$$

1. Construct abstraction $A_{\vec{\delta}}$ of e:

Standard Error Analysis

- Example:

$$f(x) = \sqrt{1+x} \quad \text{over} \quad X = [-0.1, 0.1]$$

$$e = 1 \oplus (0.5 \otimes x) \quad \longleftarrow \quad \text{implementation of } f(x)$$

1. Construct abstraction $A_{\vec{\delta}}$ of e:

$$A_{\vec{\delta}}(x) = 1 \oplus (0.5 \otimes x)$$

Standard Error Analysis

- Example:

$$f(x) = \sqrt{1+x} \quad \text{over} \quad X = [-0.1, 0.1]$$

$$e = 1 \oplus (0.5 \otimes x) \quad \longleftarrow \quad \text{implementation of } f(x)$$

1. Construct abstraction $A_{\vec{\delta}}$ of e:

$$A_{\vec{\delta}}(x) = 1 + (0.5 \times x)$$

Standard Error Analysis

- Example:

$$f(x) = \sqrt{1+x} \quad \text{over} \quad X = [-0.1, 0.1]$$

$$e = 1 \oplus (0.5 \otimes x) \;\longleftarrow\; \text{implementation of } f(x)$$

1. Construct abstraction $A_{\vec{\delta}}$ of e:

$$A_{\vec{\delta}}(x) = \big(1 + (0.5 \times x)(1+\delta_1)\big)(1+\delta_2)$$

Standard Error Analysis

- Example:

$$f(x) = \sqrt{1+x} \quad \text{over} \quad X = [-0.1, 0.1]$$

$$e = 1 \oplus (0.5 \otimes x) \quad \longleftarrow \quad \text{implementation of } f(x)$$

1. Construct abstraction $A_{\vec{\delta}}$ of e:

$$A_{\vec{\delta}}(x) = \big(1 + (0.5 \times x)(1+\delta_1)\big)(1+\delta_2)$$

- $A_{\vec{\delta}}$ is a sound abstraction of e (or $e \sqsubseteq A_{\vec{\delta}}$):

$$e(x) \in \{A_{\vec{\delta}}(x) : |\delta_1|, |\delta_2| < \epsilon\} \quad \text{for all } x \in X$$

Standard Error Analysis

- Example:

$$f(x) = \sqrt{1+x} \quad \text{over} \quad X = [-0.1, 0.1]$$

$$e = 1 \oplus (0.5 \otimes x) \quad \longleftarrow \quad \text{implementation of } f(x)$$

2. Compute a bound on relative error of e:

$$\Theta_{rel} = \max_{x \in [-0.1, 0.1],\, \delta_1, \delta_2 \in [-\epsilon, \epsilon]} \left| \frac{f(x) - A_{\vec{\delta}}(x)}{f(x)} \right|$$

Standard Error Analysis

- Example:

$$f(x) = \sqrt{1+x} \quad \text{over} \quad X = [-0.1, 0.1]$$

$$e = 1 \oplus (0.5 \otimes x) \quad \longleftarrow \quad \text{implementation of } f(x)$$

2. Compute a bound on relative error of e:

$$\Theta_{rel} = \max_{x \in [-0.1, 0.1],\ \delta_1, \delta_2 \in [-\epsilon, \epsilon]} \left| \frac{f(x) - A_{\vec{\delta}}(x)}{f(x)} \right|$$

$$= \max_{x \in [-0.1, 0.1],\ \delta_1, \delta_2 \in [-\epsilon, \epsilon]} \left| \frac{\sqrt{1+x} - \big(1 + (0.5 \times x)(1+\delta_1)\big)(1+\delta_2)}{\sqrt{1+x}} \right|$$

Standard Error Analysis

- Example:

$$f(x) = \sqrt{1+x} \quad \text{over} \quad X = [-0.1, 0.1]$$

$$e = 1 \oplus (0.5 \otimes x) \quad \longleftarrow \quad \text{implementation of } f(x)$$

2. Compute a bound on relative error of e:

$$\Theta_{rel} = \max_{x \in [-0.1,0.1],\ \delta_1,\delta_2 \in [-\epsilon,\epsilon]} \left| \frac{f(x) - A_{\vec{\delta}}(x)}{f(x)} \right|$$

$$= \max_{x \in [-0.1,0.1],\ \delta_1,\delta_2 \in [-\epsilon,\epsilon]} \left| \frac{\sqrt{1+x} - \big(1 + (0.5 \times x)(1+\delta_1)\big)(1+\delta_2)}{\sqrt{1+x}} \right|$$

<u>Theorem</u> $\mathrm{ErrUlp}(a,b) \le \mathrm{ErrRel}(a,b)/\epsilon$

Standard Error Analysis

- Example:

$$f(x) = \sqrt{1+x} \quad \text{over} \quad X = [-0.1, 0.1]$$

$$e = 1 \oplus (0.5 \otimes x) \quad \longleftarrow \quad \text{implementation of } f(x)$$

2. Compute a bound on relative error of e:

$$\Theta_{rel} = \max_{x \in [-0.1, 0.1],\, \delta_1, \delta_2 \in [-\epsilon, \epsilon]} \left| \frac{f(x) - A_{\vec{\delta}}(x)}{f(x)} \right|$$

$$= \max_{x \in [-0.1, 0.1],\, \delta_1, \delta_2 \in [-\epsilon, \epsilon]} \left| \frac{\sqrt{1+x} - \big(1 + (0.5 \times x)(1+\delta_1)\big)(1+\delta_2)}{\sqrt{1+x}} \right|$$

3. Compute a bound on ulp error of e:

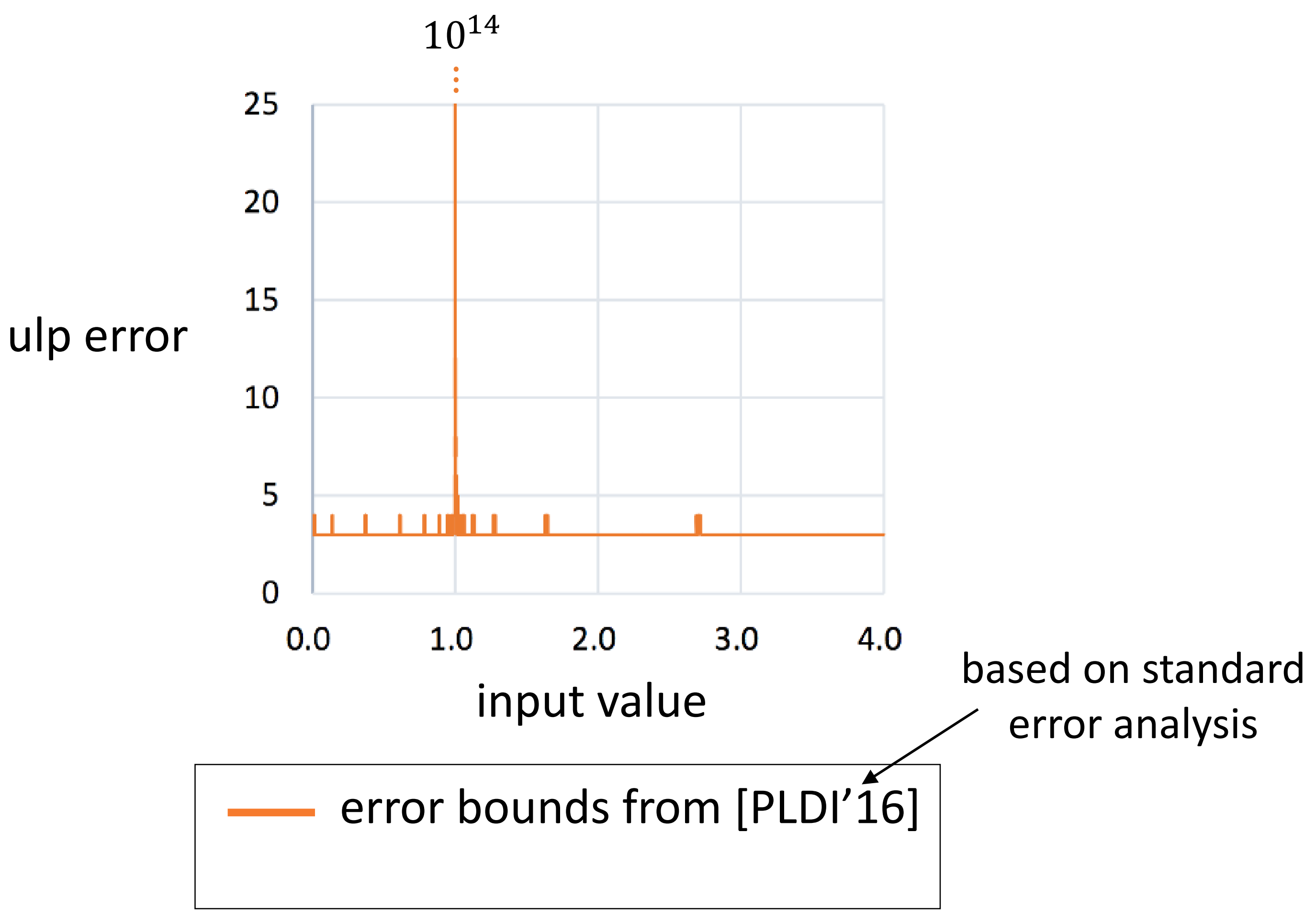
$$\Theta_{ulp} = \Theta_{rel} / \epsilon$$

<u>Theorem</u> $\mathrm{ErrUlp}(a, b) \leq \mathrm{ErrRel}(a, b)/\epsilon$

Standard Error Analysis: Limitation

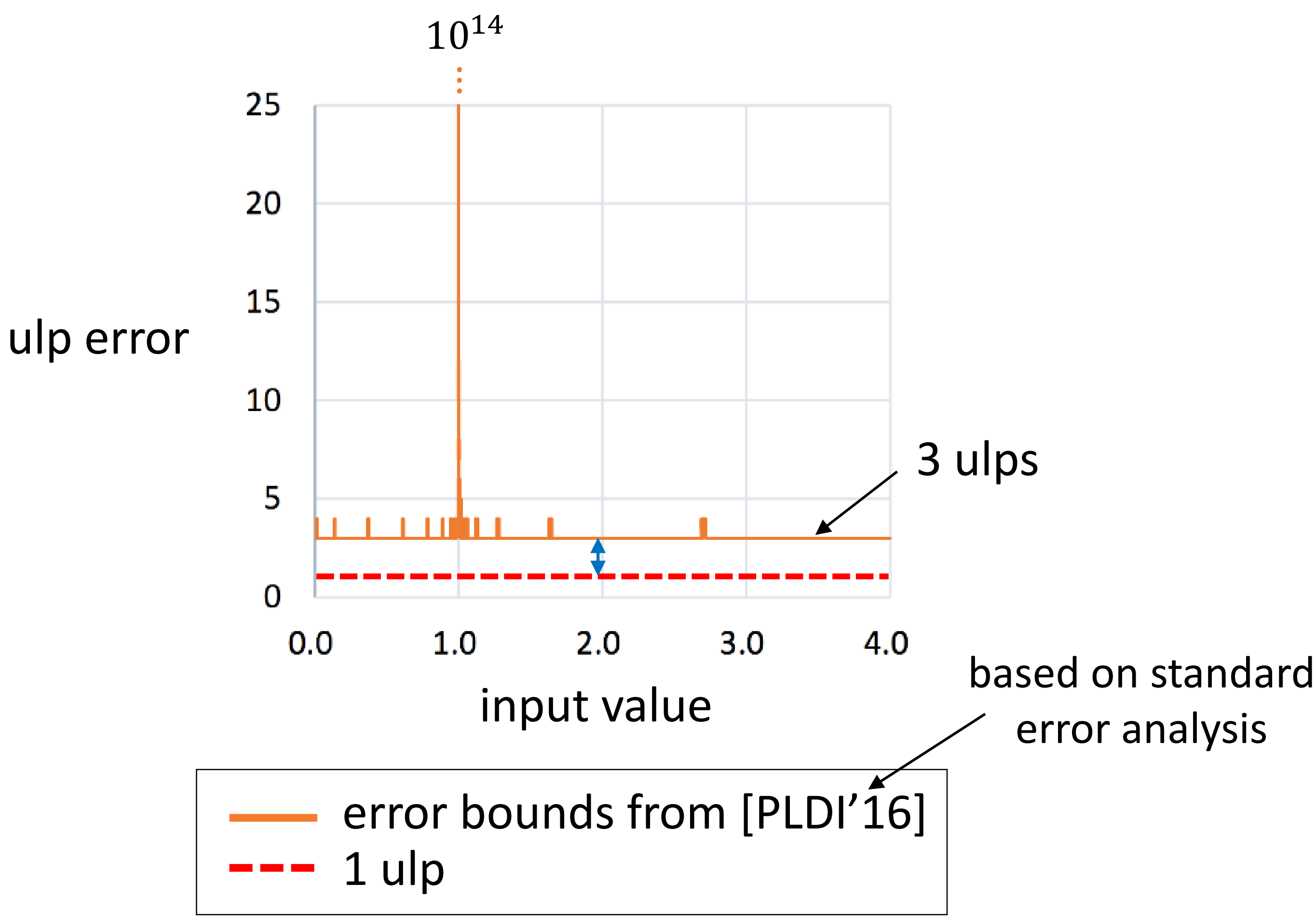
Standard Error Analysis: Limitation

Intel's `log`

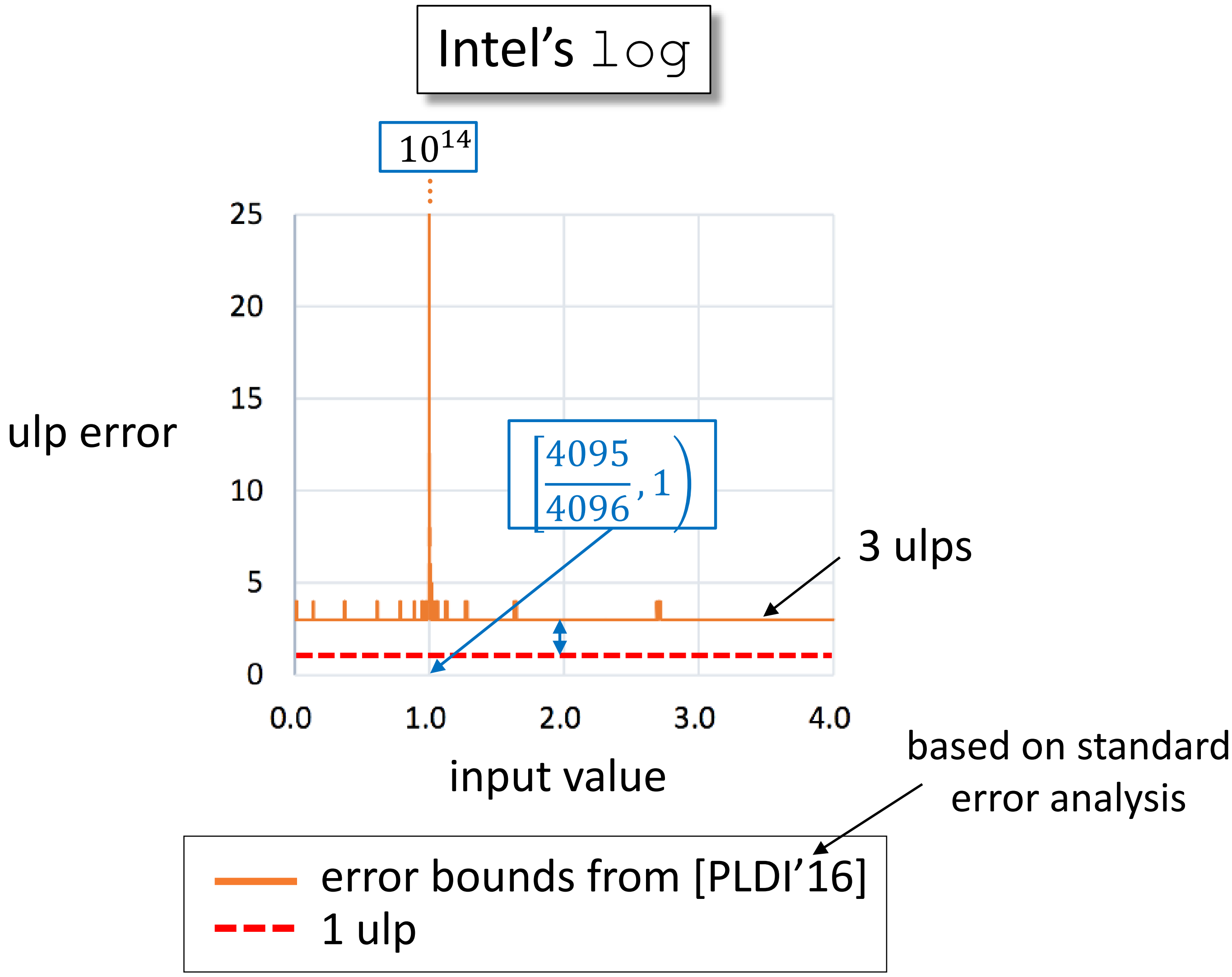


Standard Error Analysis: Limitation

Intel's `log`



Standard Error Analysis: Limitation



Why the Loose Error Bound?

Why the Loose Error Bound?

- Floating-point operations are sometimes **exact**

Why the Loose Error Bound?

- Floating-point operations are sometimes **exact**

$$0.125 \otimes x = 0.125 \times x \qquad \text{if } |0.125x| \geq 2^{-1022}$$
$$x \ominus 1 = x - 1 \qquad \text{if } 0.5 \leq x \leq 2$$

Why the Loose Error Bound?

- Floating-point operations are sometimes **exact**

$$0.125 \otimes x \quad = \quad 0.125 \times x \qquad \text{if } |0.125x| \geq 2^{-1022}$$

$$x \ominus 1 \quad = \quad x - 1 \qquad \text{if } 0.5 \leq x \leq 2$$

- Standard error analysis constructs imprecise abstractions

Why the Loose Error Bound?

- Floating-point operations are sometimes **exact**

$$0.125 \otimes x = 0.125 \times x \quad \text{if } |0.125x| \geq 2^{-1022}$$
$$x \ominus 1 = x - 1 \quad \text{if } 0.5 \leq x \leq 2$$

- Standard error analysis constructs imprecise abstractions

$$0.125 \otimes x \quad \text{if } |0.125x| \geq 2^{-1022}$$

$$x \ominus 1 \quad \text{if } 0.5 \leq x \leq 2$$

Why the Loose Error Bound?

- Floating-point operations are sometimes **exact**

$$0.125 \otimes x \quad = \quad 0.125 \times x \qquad \text{if } |0.125x| \geq 2^{-1022}$$

$$x \ominus 1 \quad = \quad x - 1 \qquad \text{if } 0.5 \leq x \leq 2$$

- Standard error analysis constructs imprecise abstractions

$$0.125 \otimes x \quad \sqsubseteq \quad 0.125 \times x \qquad \text{if } |0.125x| \geq 2^{-1022}$$

$$x \ominus 1 \quad \sqsubseteq \quad x - 1 \qquad \text{if } 0.5 \leq x \leq 2$$

Why the Loose Error Bound?

- Floating-point operations are sometimes **exact**

$$0.125 \otimes x = 0.125 \times x \quad \text{if } |0.125x| \geq 2^{-1022}$$

$$x \ominus 1 = x - 1 \quad \text{if } 0.5 \leq x \leq 2$$

- Standard error analysis constructs imprecise abstractions

$$0.125 \otimes x \sqsubseteq 0.125 \times x \quad \text{if } |0.125x| \geq 2^{-1022}$$

$$0.125 \otimes x \sqsubseteq (0.125 \times x)(1 + \delta)$$

$$x \ominus 1 \sqsubseteq x - 1 \quad \text{if } 0.5 \leq x \leq 2$$

$$x \ominus 1 \sqsubseteq (x - 1)(1 + \delta')$$

Analysis of `log`

- For $x \in \left[\frac{4095}{4096}, 1\right)$, `log` computes

Analysis of `log`

- For $x \in \left[\frac{4095}{4096}, 1\right)$, `log` computes

$$r(x) = \left[\left((2 \otimes x) \ominus \frac{255}{128}\right) \otimes \frac{1}{2}\right] \oplus \left[\left(\frac{255}{128} \otimes \frac{1}{2}\right) \ominus 1\right] \quad (\approx x - 1)$$

and returns

$$r(x) - \frac{1}{2} r(x)^2 + \cdots + \frac{1}{7} r(x)^7$$

Analysis of `log`

- For $x \in \left[\frac{4095}{4096}, 1\right)$, `log` computes

$$r(x) = \left[\left((2 \otimes x) \ominus \frac{255}{128}\right) \otimes \frac{1}{2}\right] \oplus \left[\left(\frac{255}{128} \otimes \frac{1}{2}\right) \ominus 1\right] \quad (\approx x - 1)$$

  and returns

$$\boxed{r(x)} - \frac{1}{2} r(x)^2 + \cdots + \frac{1}{7} r(x)^7$$

Analysis of `log`

- For $x \in \left[\frac{4095}{4096}, 1\right)$, `log` computes

$$r(x) = \left[\left((2 \otimes x) \ominus \frac{255}{128}\right) \otimes \frac{1}{2}\right] \oplus \left[\left(\frac{255}{128} \otimes \frac{1}{2}\right) \ominus 1\right] \quad (\approx x - 1)$$

  and returns

$$\boxed{r(x)} - \frac{1}{2} r(x)^2 + \cdots + \frac{1}{7} r(x)^7$$

- Roughly speaking,

(precision loss of `log`) $\geq$ (precision loss of $r(x)$)

Analysis of `log`

- For $x \in \left[\frac{4095}{4096}, 1\right)$, `log` computes

$$r(x) = \left[\left((2 \otimes x) \ominus \frac{255}{128}\right) \otimes \frac{1}{2}\right] \oplus \left[\left(\frac{255}{128} \otimes \frac{1}{2}\right) \ominus 1\right] \quad (\approx x - 1)$$

  and returns

$$\boxed{r(x)} - \frac{1}{2} r(x)^2 + \cdots + \frac{1}{7} r(x)^7$$

- Roughly speaking,

(precision loss of `log`) $\geq$ (precision loss of $r(x)$)

Standard Analysis of `log`

$$r(x) = \left[\left((2 \otimes x) \ominus \frac{255}{128}\right) \otimes \frac{1}{2}\right] \oplus \left[\left(\frac{255}{128} \otimes \frac{1}{2}\right) \ominus 1\right] \quad \left(\frac{4095}{4096} \le x < 1\right)$$

- Abstraction of $r(x)$:

$$A_{\vec{\delta}}(x) = \left[\left((2\ \times\ x)(1 + \delta_0) - \frac{255}{128}\right)(1 + \delta_1)\ \times \frac{1}{2}\right](1 + \delta_2) + \cdots$$

Standard Analysis of `log`

$$r(x) = \left[\left((2 \otimes x) \ominus \frac{255}{128}\right) \otimes \frac{1}{2}\right] \oplus \left[\left(\frac{255}{128} \otimes \frac{1}{2}\right) \ominus 1\right] \quad \left(\frac{4095}{4096} \le x < 1\right)$$

- Abstraction of $r(x)$:

$$A_{\vec{\delta}}(x) = \left[\left((2 \times x)(1+\delta_0) - \frac{255}{128}\right)(1+\delta_1) \times \frac{1}{2}\right](1+\delta_2) + \cdots$$

Standard Analysis of `log`

$$r(x) = \left[\left((2 \otimes x) \ominus \frac{255}{128}\right) \otimes \frac{1}{2}\right] \oplus \left[\left(\frac{255}{128} \otimes \frac{1}{2}\right) \ominus 1\right] \quad \left(\frac{4095}{4096} \leq x < 1\right)$$

- Abstraction of $r(x)$:

$$A_{\vec{\delta}}(x) = \left[\left((2 \times x)(1 + \delta_0) - \frac{255}{128}\right)(1 + \delta_1) \times \frac{1}{2}\right](1 + \delta_2) + \cdots$$

$$= (x - 1) + \left(x - \frac{255}{256}\right)\delta_1 + \cdots$$

Standard Analysis of `log`

$$r(x) = \left[\left((2 \otimes x) \ominus \frac{255}{128}\right) \otimes \frac{1}{2}\right] \oplus \left[\left(\frac{255}{128} \otimes \frac{1}{2}\right) \ominus 1\right] \quad \left(\frac{4095}{4096} \leq x < 1\right)$$

- Abstraction of $r(x)$:

$$A_{\vec{\delta}}(x) = \left[\left((2 \times x)(1+\delta_0) - \frac{255}{128}\right)(1+\delta_1) \times \frac{1}{2}\right](1+\delta_2) + \cdots$$
$$= (x-1) + \left(x - \frac{255}{256}\right)\delta_1 + \cdots$$

- Bound on relative error of $r(x)$:

$$\max_{\frac{4095}{4096} \leq x < 1,\ |\delta_i| < \epsilon} \left|\frac{A_{\vec{\delta}}(x) - (x-1)}{x-1}\right|$$

Standard Analysis of `log`

$$r(x) = \left[\left((2 \otimes x) \ominus \frac{255}{128}\right) \otimes \frac{1}{2}\right] \oplus \left[\left(\frac{255}{128} \otimes \frac{1}{2}\right) \ominus 1\right] \quad \left(\frac{4095}{4096} \le x < 1\right)$$

- Abstraction of $r(x)$:

$$A_{\vec{\delta}}(x) = \left[\left((2 \times x)(1+\delta_0) - \frac{255}{128}\right)(1+\delta_1) \times \frac{1}{2}\right](1+\delta_2) + \cdots$$

$$= (x-1) + \left(x - \frac{255}{256}\right)\delta_1 + \cdots$$

- Bound on relative error of $r(x)$:

$$\max_{\frac{4095}{4096} \le x < 1,\ |\delta_i| < \epsilon} \left| \frac{A_{\vec{\delta}}(x) - (x-1)}{x-1} \right|$$

Standard Analysis of `log`

$$r(x) = \left[\left((2 \otimes x) \ominus \frac{255}{128}\right) \otimes \frac{1}{2}\right] \oplus \left[\left(\frac{255}{128} \otimes \frac{1}{2}\right) \ominus 1\right] \quad \left(\frac{4095}{4096} \le x < 1\right)$$

- Abstraction of $r(x)$:

$$A_{\vec{\delta}}(x) = \left[\left((2 \times x)(1+\delta_0) - \frac{255}{128}\right)(1+\delta_1) \times \frac{1}{2}\right](1+\delta_2) + \cdots$$

$$= (x-1) + \left(x - \frac{255}{256}\right)\delta_1 + \cdots$$

- Bound on relative error of $r(x)$:

$$\max_{\frac{4095}{4096} \le x < 1,\ |\delta_i| < \epsilon} \left|\frac{A_{\vec{\delta}}(x) - (x-1)}{x-1}\right| \ge \max_{\frac{4095}{4096} \le x < 1} \left|\frac{x - \frac{255}{256}}{x-1}\right| \epsilon$$

Standard Analysis of `log`

$$r(x) = \left[\left((2 \otimes x) \ominus \frac{255}{128}\right) \otimes \frac{1}{2}\right] \oplus \left[\left(\frac{255}{128} \otimes \frac{1}{2}\right) \ominus 1\right] \quad \left(\frac{4095}{4096} \leq x < 1\right)$$

- Abstraction of $r(x)$:

$$A_{\vec{\delta}}(x) = \left[\left((2 \times x)(1+\delta_0) - \frac{255}{128}\right)(1+\delta_1) \times \frac{1}{2}\right](1+\delta_2) + \cdots$$
$$= (x-1) + \left(x - \frac{255}{256}\right)\delta_1 + \cdots$$

- Bound on relative error of $r(x)$:

$$\max_{\frac{4095}{4096} \leq x < 1,\ |\delta_i| < \epsilon} \left|\frac{A_{\vec{\delta}}(x) - (x-1)}{x-1}\right| \geq \max_{\frac{4095}{4096} \leq x < 1} \left|\frac{x - \frac{255}{256}}{x-1}\right| \epsilon$$

unbounded

Standard Analysis of `log`

$$r(x) = \left[\left((2 \otimes x) \ominus \frac{255}{128}\right) \otimes \frac{1}{2}\right] \oplus \left[\left(\frac{255}{128} \otimes \frac{1}{2}\right) \ominus 1\right] \quad \left(\frac{4095}{4096} \le x < 1\right)$$

- Abstraction of $r(x)$:

$$A_{\vec{\delta}}(x) = \left[\left((2 \times x)(1+\delta_0) - \frac{255}{128}\right)(1+\delta_1) \times \frac{1}{2}\right](1+\delta_2) + \cdots$$

$$= (x-1) + \left(x - \frac{255}{256}\right)\delta_1 + \cdots$$

10^{14} ulps for `log` near $x = 1$ [PLDI'16]

- Bound on relative error of $r(x)$:

$$\max_{\frac{4095}{4096} \le x < 1,\ |\delta_i| < \epsilon} \left|\frac{A_{\vec{\delta}}(x) - (x-1)}{x-1}\right| \ge \max_{\frac{4095}{4096} \le x < 1} \left|\frac{x - \frac{255}{256}}{x-1}\right| \epsilon$$

unbounded

Precise Analysis of `log`

$$r(x) = \left[\left((2 \otimes x) \ominus \frac{255}{128}\right) \otimes \frac{1}{2}\right] \oplus \left[\left(\frac{255}{128} \otimes \frac{1}{2}\right) \ominus 1\right] \quad \left(\frac{4095}{4096} \leq x < 1\right)$$

Precise Analysis of `log`

exact

$$r(x) = \left[\left((2 \otimes x) \ominus \frac{255}{128}\right) \otimes \frac{1}{2}\right] \oplus \left[\left(\frac{255}{128} \otimes \frac{1}{2}\right) \ominus 1\right] \quad \left(\frac{4095}{4096} \leq x < 1\right)$$

Precise Analysis of `log`

exact

$$r(x) = \left[\left((2 \otimes x) \ominus \frac{255}{128}\right) \otimes \frac{1}{2}\right] \oplus \left[\left(\frac{255}{128} \otimes \frac{1}{2}\right) \ominus 1\right] \quad \left(\frac{4095}{4096} \le x < 1\right)$$

Precise Analysis of `log`

exact

$$r(x) = \left[\left((2 \otimes x) \ominus \frac{255}{128}\right) \otimes \frac{1}{2}\right] \oplus \left[\left(\frac{255}{128} \otimes \frac{1}{2}\right) \ominus 1\right] \quad \left(\frac{4095}{4096} \leq x < 1\right)$$

Precise Analysis of `log`

exact

$$r(x) = \left[\left((2 \otimes x) \ominus \frac{255}{128}\right) \otimes \frac{1}{2}\right] \oplus \left[\left(\frac{255}{128} \otimes \frac{1}{2}\right) \ominus 1\right] \quad \left(\frac{4095}{4096} \leq x < 1\right)$$

Precise Analysis of `log`

exact

$$r(x) = \left[\left((2 \otimes x) \ominus \frac{255}{128}\right) \otimes \frac{1}{2}\right] \oplus \left[\left(\frac{255}{128} \otimes \frac{1}{2}\right) \ominus 1\right] \quad \left(\frac{4095}{4096} \leq x < 1\right)$$

- More precise abstraction of $r(x)$:

$$A'_{\vec{\delta}}(x) = (x-1) + (x-1)\delta'$$

Precise Analysis of `log`

exact

$$r(x) = \left[\left((2 \otimes x) \ominus \frac{255}{128}\right) \otimes \frac{1}{2}\right] \oplus \left[\left(\frac{255}{128} \otimes \frac{1}{2}\right) \ominus 1\right] \quad \left(\frac{4095}{4096} \leq x < 1\right)$$

- More precise abstraction of $r(x)$:

$$A'_{\vec{\delta}}(x) = (x-1) + (x-1)\delta'$$

- Tighter bound on relative error of $r(x)$:

$$\max_{\frac{4095}{4096} \leq x < 1,\ |\delta'| < \epsilon} \left| \frac{A'_{\vec{\delta}}(x) - (x-1)}{x-1} \right| = \max_{\frac{4095}{4096} \leq x < 1} \left| \frac{x-1}{x-1} \right| \epsilon$$

$$= \epsilon$$

Precise Analysis of `log`

exact

$$r(x) = \left[\left((2 \otimes x) \ominus \frac{255}{128}\right) \otimes \frac{1}{2}\right] \oplus \left[\left(\frac{255}{128} \otimes \frac{1}{2}\right) \ominus 1\right] \quad \left(\frac{4095}{4096} \le x < 1\right)$$

- More precise abstraction of $r(x)$:

$$A'_{\vec{\delta}}(x) = (x-1) + (x-1)\delta'$$

- Tighter bound on relative error of $r(x)$:

$$\max_{\frac{4095}{4096} \le x < 1,\ |\delta'| < \epsilon} \left| \frac{A'_{\vec{\delta}}(x) - (x-1)}{x-1} \right| = \max_{\frac{4095}{4096} \le x < 1} \left| \frac{x-1}{x-1} \right| \epsilon$$

$$= \epsilon$$

We prove a bound of 0.583 ulps for `log`

Precise Analysis of `log`

exact

$$r(x) = \left[\left((2 \otimes x) \ominus \frac{255}{128}\right) \otimes \frac{1}{2}\right] \oplus \left[\left(\frac{255}{128} \otimes \frac{1}{2}\right) \ominus 1\right] \quad \left(\frac{4095}{4096} \le x < 1\right)$$

To prove the 1 ulp error bound,

- Need to construct more precise abstractions

$\frac{4095}{4096} \le x < 1, |\delta'| < \epsilon$ $x - 1$ $\frac{4095}{4096} \le x < 1$ $|x - 1|$

$= \epsilon$

We prove a bound of 0.583 ulps for `log`

Precise Analysis of `log`

exact

$$r(x) = \left[\left((2 \otimes x) \ominus \frac{255}{128}\right) \otimes \frac{1}{2}\right] \oplus \left[\left(\frac{255}{128} \otimes \frac{1}{2}\right) \ominus 1\right] \quad \left(\frac{4095}{4096} \leq x < 1\right)$$

To prove the 1 ulp error bound,

- Need to construct more precise abstractions
- Need to use floating-point exactness results

$\frac{4095}{4096} \leq x < 1, |\delta\prime| < \epsilon$ $\quad x - 1 \quad$ $\frac{4095}{4096} \leq x < 1$ $\quad |x - 1|$

$= \epsilon$

We prove a bound of 0.583 ulps for `log`

Sterbenz's Theorem

- <u>Theorem</u> [Sterbenz, 1973]

$$\frac{1}{2}a \leq b \leq 2a \quad \Longrightarrow \quad a \ominus b = a - b$$

Sterbenz's Theorem

- Theorem [Sterbenz, 1973]

$$\frac{1}{2}a \leq b \leq 2a \quad \Longrightarrow \quad a \ominus b = a - b$$

- Example: `log` for $x \in \left[\frac{4095}{4096}, 1\right)$

Sterbenz's Theorem

- Theorem [Sterbenz, 1973]

$$\frac{1}{2}a \leq b \leq 2a \quad \Longrightarrow \quad a \ominus b = a - b$$

- Example: `log` for $x \in \left[\frac{4095}{4096}, 1\right)$

- Example: `sin` for $x \in \left[2\pi - \frac{\pi}{64}, 2\pi + \frac{\pi}{64}\right]$

  1. Compute $y = x \ominus 2\pi$
  2. Return $y - \frac{1}{3!}y^3 + \cdots + \frac{1}{9!}y^9 \ (\approx sin\, y)$

Sterbenz's Theorem

- Theorem [Sterbenz, 1973]

$$\frac{1}{2}a \leq b \leq 2a \quad \Longrightarrow \quad a \ominus b = a - b$$

- Example: `log` for $x \in \left[\frac{4095}{4096}, 1\right)$

- Example: `sin` for $x \in \left[2\pi - \frac{\pi}{64}, 2\pi + \frac{\pi}{64}\right]$

  1. Compute $y = x \ominus 2\pi$
  2. Return $y - \frac{1}{3!}y^3 + \cdots + \frac{1}{9!}y^9$ $(\approx sin\, y)$

Sterbenz's Theorem

- Theorem [Sterbenz, 1973]

$$\frac{1}{2}a \leq b \leq 2a \quad \Longrightarrow \quad a \ominus b = a - b$$

- Example: `log` for $x \in \left[\frac{4095}{4096}, 1\right)$
- Example: `sin` for $x \in \left[2\pi - \frac{\pi}{64}, 2\pi + \frac{\pi}{64}\right]$
  1. Compute $y = x \ominus 2\pi$ exact
  2. Return $y - \frac{1}{3!}y^3 + \cdots + \frac{1}{9!}y^9$ $(\approx \sin y)$

Sterbenz's Theorem

- Theorem [Sterbenz, 1973]

$$\frac{1}{2}a \le b \le 2a \quad \Longrightarrow \quad a \ominus b = a - b$$

- Example: `log` for $x \in \left[\frac{4095}{4096}, 1\right)$
- Example: `sin` for $x \in \left[2\pi - \frac{\pi}{64}, 2\pi + \frac{\pi}{64}\right]$
  1. Compute $y = x \ominus 2\pi$ exact
  2. Return $y - \frac{1}{3!}y^3 + \cdots + \frac{1}{9!}y^9 \ (\approx sin\, y)$

precision loss of `sin` is small

Sterbenz's Theorem

- How to apply the theorem automatically?

Sterbenz's Theorem

- How to apply the theorem automatically?

$$e \ominus e' \quad \sqsubseteq \quad ?$$

Sterbenz's Theorem

- How to apply the theorem automatically?

$$e \ominus e' \quad \sqsubseteq \quad ?$$

  - Construct: $e \sqsubseteq A_{\vec{\delta}},\ e' \sqsubseteq A'_{\vec{\delta}}$

Sterbenz's Theorem

- How to apply the theorem automatically?

$$e \ominus e' \quad \sqsubseteq \quad ?$$

  - Construct: $e \sqsubseteq A_{\vec{\delta}},\ e' \sqsubseteq A'_{\vec{\delta}}$

  - Precondition of theorem:

$$\frac{1}{2}e(x) \le e'(x) \le 2e(x) \qquad \text{for all } x \in X$$

Sterbenz's Theorem

- How to apply the theorem automatically?

$$e \ominus e' \;\; \sqsubseteq \;\; ?$$

  - Construct: $e \sqsubseteq A_{\vec{\delta}}, \; e' \sqsubseteq A'_{\vec{\delta}}$

$$\min_{x,\vec{\delta}} \left(A'_{\vec{\delta}}(x) - \tfrac{1}{2} A_{\vec{\delta}}(x) \right)$$
$$\max_{x,\vec{\delta}} \left(A'_{\vec{\delta}}(x) - 2A_{\vec{\delta}}(x) \right)$$

  - Precondition of theorem:

$$\frac{1}{2} e(x) \le e'(x) \le 2e(x) \qquad \text{for all } x \in X$$

Sterbenz's Theorem

- How to apply the theorem automatically?

$$e \ominus e' \;\; \sqsubseteq \;\; ?$$

  - Construct: $e \sqsubseteq A_{\vec{\delta}},\ e' \sqsubseteq A'_{\vec{\delta}}$

  - Check: $\min_{x,\vec{\delta}} \left(A'_{\vec{\delta}}(x) - \frac{1}{2} A_{\vec{\delta}}(x) \right) \geq 0$ and
    $\max_{x,\vec{\delta}} \left(A'_{\vec{\delta}}(x) - 2A_{\vec{\delta}}(x) \right) \leq 0$

  - Precondition of theorem:

$$\frac{1}{2} e(x) \leq e'(x) \leq 2e(x) \qquad \text{for all } x \in X$$

Sterbenz's Theorem

- How to apply the theorem automatically?

$$e \ominus e' \;\sqsubseteq\; ?$$

- Construct: $e \sqsubseteq A_{\vec{\delta}},\ e' \sqsubseteq A'_{\vec{\delta}}$
- Check: $\min_{x,\vec{\delta}} \left(A'_{\vec{\delta}}(x) - \frac{1}{2} A_{\vec{\delta}}(x) \right) \geq 0$ and
  $\max_{x,\vec{\delta}} \left(A'_{\vec{\delta}}(x) - 2A_{\vec{\delta}}(x) \right) \leq 0$

  ← true

- Precondition of theorem:

$$\frac{1}{2} e(x) \leq e'(x) \leq 2e(x) \qquad \text{for all } x \in X$$

Sterbenz's Theorem

- How to apply the theorem automatically?

$$e \ominus e' \ \sqsubseteq \ ?$$

  - Construct: $e \sqsubseteq A_{\vec{\delta}},\ e' \sqsubseteq A'_{\vec{\delta}}$
  - Check: $\min_{x,\vec{\delta}} \left(A'_{\vec{\delta}}(x) - \frac{1}{2} A_{\vec{\delta}}(x) \right) \geq 0$ and $\max_{x,\vec{\delta}} \left(A'_{\vec{\delta}}(x) - 2A_{\vec{\delta}}(x) \right) \leq 0$ ← true

$\Longrightarrow \quad \frac{1}{2} A_{\vec{\delta}}(x) \leq A'_{\vec{\delta}}(x) \leq 2A_{\vec{\delta}}(x)$ for all $x \in X$, all $|\delta_i| < \epsilon$

  - Precondition of theorem:

$\frac{1}{2} e(x) \leq e'(x) \leq 2e(x)$ for all $x \in X$

Sterbenz's Theorem

- How to apply the theorem automatically?

$$e \ominus e' \;\sqsubseteq\; ?$$

- Construct: $e \sqsubseteq A_{\vec{\delta}},\; e' \sqsubseteq A'_{\vec{\delta}}$
- Check: $\min_{x,\vec{\delta}} \left(A'_{\vec{\delta}}(x) - \frac{1}{2} A_{\vec{\delta}}(x) \right) \geq 0$ and

$$\max_{x,\vec{\delta}} \left(A'_{\vec{\delta}}(x) - 2A_{\vec{\delta}}(x) \right) \leq 0$$

← true

$\Longrightarrow \quad \frac{1}{2} A_{\vec{\delta}}(x) \leq A'_{\vec{\delta}}(x) \leq 2A_{\vec{\delta}}(x)$ for all $x \in X$, all $|\delta_i| < \epsilon$

- Precondition of theorem:

$\Longrightarrow \quad \frac{1}{2} e(x) \leq e'(x) \leq 2e(x)$ for all $x \in X$

Sterbenz's Theorem

- How to apply the theorem automatically?

can apply the theorem → $e \ominus e' \sqsubseteq ?$

- Construct: $e \sqsubseteq A_{\vec{\delta}},\ e' \sqsubseteq A'_{\vec{\delta}}$

- Check: $\min_{x,\vec{\delta}} \left(A'_{\vec{\delta}}(x) - \frac{1}{2} A_{\vec{\delta}}(x) \right) \geq 0$ and
  $\max_{x,\vec{\delta}} \left(A'_{\vec{\delta}}(x) - 2A_{\vec{\delta}}(x) \right) \leq 0$ ← true

$\Longrightarrow \quad \frac{1}{2} A_{\vec{\delta}}(x) \leq A'_{\vec{\delta}}(x) \leq 2A_{\vec{\delta}}(x)$ for all $x \in X$, all $|\delta_i| < \epsilon$

- Precondition of theorem:

$\Longrightarrow \quad \frac{1}{2} e(x) \leq e'(x) \leq 2e(x)$ for all $x \in X$

Sterbenz's Theorem

- How to apply the theorem automatically?

can apply the theorem → $e \ominus e' \;\sqsubseteq\; A_{\vec{\delta}}(x) - A'_{\vec{\delta}}(x)$

  - Construct: $e \sqsubseteq A_{\vec{\delta}},\; e' \sqsubseteq A'_{\vec{\delta}}$

  - Check: $\min_{x,\vec{\delta}} \left(A'_{\vec{\delta}}(x) - \frac{1}{2} A_{\vec{\delta}}(x) \right) \geq 0$ and
    $\max_{x,\vec{\delta}} \left(A'_{\vec{\delta}}(x) - 2A_{\vec{\delta}}(x) \right) \leq 0$ ← true

$\Longrightarrow \quad \frac{1}{2} A_{\vec{\delta}}(x) \leq A'_{\vec{\delta}}(x) \leq 2A_{\vec{\delta}}(x)$ for all $x \in X$, all $|\delta_i| < \epsilon$

  - Precondition of theorem:

$\Longrightarrow \quad \frac{1}{2} e(x) \leq e'(x) \leq 2e(x)$ for all $x \in X$

Sterbenz's Theorem

- How to apply the theorem automatically?

can apply the theorem → $e \ominus e' \sqsubseteq A_{\vec{\delta}}(x) - A'_{\vec{\delta}}(x)$

  - Construct: $e \sqsubseteq A_{\vec{\delta}}, \; e' \sqsubseteq A'_{\vec{\delta}}$

  - Check: $\min_{x,\vec{\delta}} \left(A'_{\vec{\delta}}(x) - \frac{1}{2} A_{\vec{\delta}}(x) \right) \geq 0$ and

    $\max_{x,\vec{\delta}} \left(A'_{\vec{\delta}}(x) - 2A_{\vec{\delta}}(x) \right) \leq 0$ ← true

$\Longrightarrow \quad \frac{1}{2} A_{\vec{\delta}}(x) \leq A'_{\vec{\delta}}(x) \leq 2A_{\vec{\delta}}(x)$ for all $x \in X$, all $|\delta_i| < \epsilon$

  - Precondition of theorem:

$\Longrightarrow \quad \frac{1}{2} e(x) \leq e'(x) \leq 2e(x)$ for all $x \in X$

Dekker's Theorem

- Theorem [Dekker, 1971]

$$|a| \geq |b| \quad \text{and} \quad r = a \oplus b \ominus a \ominus b$$

Dekker's Theorem

- Theorem [Dekker, 1971]

$$|a| \geq |b| \quad \text{and} \quad r = a \oplus b \ominus a \ominus b$$
$$\Longrightarrow \quad r = (a \oplus b) - (a + b)$$

Dekker's Theorem

- Theorem [Dekker, 1971]

$$|a| \geq |b| \quad \text{and} \quad r = a \oplus b \ominus a \ominus b$$
$$\implies \quad r = (a \oplus b) - (a + b)$$

- Example: $a + b + c,$ where $a \geq b \geq c > 0$

Dekker's Theorem

- Theorem [Dekker, 1971]

$$|a| \geq |b| \quad \text{and} \quad r = a \oplus b \ominus a \ominus b$$
$$\Longrightarrow \quad r = (a \oplus b) - (a + b)$$

- Example: $a + b + c,$ where $a \geq b \geq c > 0$
  - Naïve ways: $(a \oplus b) \oplus c,\ a \oplus (b \oplus c),\ \cdots$

Dekker's Theorem

- Theorem [Dekker, 1971]

$$|a| \geq |b| \;\; \text{and} \;\; r = a \oplus b \ominus a \ominus b$$
$$\Longrightarrow \quad r = (a \oplus b) - (a + b)$$

- Example: $a + b + c,$ where $a \geq b \geq c > 0$
  - Naïve ways: $(a \oplus b) \oplus c,\; a \oplus (b \oplus c),\; \cdots$
  - More accurate way:
    1. Compute $r = a \oplus b \ominus a \ominus b$
    2. Return $(a \oplus b) \oplus (c \ominus r)$

Dekker's Theorem

- Theorem [Dekker, 1971]

$$|a| \geq |b| \quad \text{and} \quad r = a \oplus b \ominus a \ominus b$$
$$\Longrightarrow \quad r = (a \oplus b) - (a + b)$$

- Example: $a + b + c$, where $a \geq b \geq c > 0$
  - Naïve ways: $(a \oplus b) \oplus c,\ a \oplus (b \oplus c),\ \cdots$
  - More accurate way:
    1. Compute $r = a \oplus b \ominus a \ominus b$
    2. Return $(a \oplus b) \oplus (c \ominus r)$

Dekker's Theorem

- Theorem [Dekker, 1971]

$$|a| \geq |b| \text{ and } r = a \oplus b \ominus a \ominus b$$
$$\Longrightarrow \quad r = (a \oplus b) - (a + b)$$

- Example: $a + b + c$, where $a \geq b \geq c > 0$
  - Naïve ways: $(a \oplus b) \oplus c$, $a \oplus (b \oplus c)$, $\cdots$
  - More accurate way:
    1. Compute $r = a \oplus b \ominus a \ominus b$
    2. Return $\underbrace{(a \oplus b)}_{= (a+b)+r} \oplus (c \ominus r)$

Dekker's Theorem

- Theorem [Dekker, 1971]

$$|a| \geq |b| \ \text{ and } \ r = a \oplus b \ominus a \ominus b$$
$$\Longrightarrow \quad r = (a \oplus b) - (a + b)$$

- Example: $a + b + c$, where $\boxed{a \geq b} \geq c > 0$
  - Naïve ways: $(a \oplus b) \oplus c$, $a \oplus (b \oplus c)$, $\cdots$
  - More accurate way:
    1. Compute $r = a \oplus b \ominus a \ominus b$
    2. Return $\underbrace{(a \oplus b)}_{\displaystyle = (a + b)\ \boxed{\cancel{+ r}}} \oplus (c \boxed{\cancel{\ominus r}})$

Dekker’s Theorem

- How to apply the theorem automatically?

Dekker's Theorem

- How to apply the theorem automatically?

$$e \oplus e' \ominus e \ominus e' \quad \sqsubseteq \quad ?$$

Dekker's Theorem

- How to apply the theorem automatically?

$$e \oplus e' \ominus e \ominus e' \quad \sqsubseteq \quad ?$$

  - Construct: $e \sqsubseteq A_{\vec{\delta}},\ e' \sqsubseteq A'_{\vec{\delta}},\ e \oplus e' \sqsubseteq \left(A_{\vec{\delta}}(x) + A'_{\vec{\delta}}(x)\right)(1+\delta)$

Dekker's Theorem

- How to apply the theorem automatically?

$$e \oplus e' \ominus e \ominus e' \quad \sqsubseteq \quad ?$$

  - Construct: $e \sqsubseteq A_{\vec{\delta}},\ e' \sqsubseteq A'_{\vec{\delta}},\ e \oplus e' \sqsubseteq \left(A_{\vec{\delta}}(x) + A'_{\vec{\delta}}(x)\right)(1+\delta)$

$$\min_{x,\vec{\delta}} \left|A_{\vec{\delta}}(x)\right| \qquad \max_{x,\vec{\delta}} \left|A'_{\vec{\delta}}(x)\right|$$

Dekker's Theorem

- How to apply the theorem automatically?

$$e \oplus e' \ominus e \ominus e' \quad \sqsubseteq \quad ?$$

  - Construct: $e \sqsubseteq A_{\vec{\delta}},\ e' \sqsubseteq A'_{\vec{\delta}},\ e \oplus e' \sqsubseteq \left(A_{\vec{\delta}}(x) + A'_{\vec{\delta}}(x)\right)(1+\delta)$
  - Check: $\min_{x,\vec{\delta}} \left|A_{\vec{\delta}}(x)\right| \geq \max_{x,\vec{\delta}} \left|A'_{\vec{\delta}}(x)\right|$

Dekker's Theorem

- How to apply the theorem automatically?

$$e \oplus e' \ominus e \ominus e' \quad \sqsubseteq \quad ?$$

  - Construct: $e \sqsubseteq A_{\vec{\delta}},\ e' \sqsubseteq A'_{\vec{\delta}},\ e \oplus e' \sqsubseteq \left(A_{\vec{\delta}}(x) + A'_{\vec{\delta}}(x)\right)(1+\delta)$
  - Check: $\min_{x,\vec{\delta}} \left|A_{\vec{\delta}}(x)\right| \geq \max_{x,\vec{\delta}} \left|A'_{\vec{\delta}}(x)\right|$ ← true

Dekker's Theorem

- How to apply the theorem automatically?

$$e \oplus e' \ominus e \ominus e' \sqsubseteq ?$$

  - Construct: $e \sqsubseteq A_{\vec{\delta}},\ e' \sqsubseteq A'_{\vec{\delta}},\ e \oplus e' \sqsubseteq \left(A_{\vec{\delta}}(x) + A'_{\vec{\delta}}(x)\right)(1+\delta)$
  - Check: $\min_{x,\vec{\delta}} \left|A_{\vec{\delta}}(x)\right| \geq \max_{x,\vec{\delta}} \left|A'_{\vec{\delta}}(x)\right|$ ← true

$\Longrightarrow \quad \left|A_{\vec{\delta}}(x)\right| \geq \left|A'_{\vec{\delta}}(x)\right|$ for all $x \in X$, all $|\delta_i| < \epsilon$

Dekker's Theorem

- How to apply the theorem automatically?

$$e \oplus e' \ominus e \ominus e' \sqsubseteq ?$$

- Construct: $e \sqsubseteq A_{\vec{\delta}},\ e' \sqsubseteq A'_{\vec{\delta}},\ e \oplus e' \sqsubseteq \left(A_{\vec{\delta}}(x) + A'_{\vec{\delta}}(x)\right)(1+\delta)$
- Check: $\min_{x,\vec{\delta}} \left|A_{\vec{\delta}}(x)\right| \geq \max_{x,\vec{\delta}} \left|A'_{\vec{\delta}}(x)\right|$ ← true

  $\Longrightarrow$ $\left|A_{\vec{\delta}}(x)\right| \geq \left|A'_{\vec{\delta}}(x)\right|$ for all $x \in X$, all $|\delta_i| < \epsilon$

- Precondition of theorem:

  $\Longrightarrow$ $|e(x)| \geq |e'(x)|$ for all $x \in X$

Dekker's Theorem

- How to apply the theorem automatically?

can apply the theorem → $e \oplus e' \ominus e \ominus e' \sqsubseteq ?$

  - Construct: $e \sqsubseteq A_{\vec{\delta}},\ e' \sqsubseteq A'_{\vec{\delta}},\ e \oplus e' \sqsubseteq \left(A_{\vec{\delta}}(x) + A'_{\vec{\delta}}(x)\right)(1+\delta)$
  - Check: $\min_{x,\vec{\delta}} \left|A_{\vec{\delta}}(x)\right| \geq \max_{x,\vec{\delta}} \left|A'_{\vec{\delta}}(x)\right|$ ← true

    $\Longrightarrow$ $\left|A_{\vec{\delta}}(x)\right| \geq \left|A'_{\vec{\delta}}(x)\right|$ for all $x \in X$, all $|\delta_i| < \epsilon$
  - Precondition of theorem:

    $\Longrightarrow$ $|e(x)| \geq |e'(x)|$ for all $x \in X$

Dekker's Theorem

- How to apply the theorem automatically?

can apply the theorem → $e \oplus e' \ominus e \ominus e' \sqsubseteq \left(A_{\vec{\delta}}(x) + A'_{\vec{\delta}}(x)\right)\delta$

  - Construct: $e \sqsubseteq A_{\vec{\delta}},\ e' \sqsubseteq A'_{\vec{\delta}},\ e \oplus e' \sqsubseteq \left(A_{\vec{\delta}}(x) + A'_{\vec{\delta}}(x)\right)(1+\delta)$
  - Check: $\min_{x,\vec{\delta}} \left|A_{\vec{\delta}}(x)\right| \geq \max_{x,\vec{\delta}} \left|A'_{\vec{\delta}}(x)\right|$ ← true

    $\Longrightarrow \quad \left|A_{\vec{\delta}}(x)\right| \geq \left|A'_{\vec{\delta}}(x)\right| \qquad$ for all $x \in X$, all $|\delta_i| < \epsilon$

  - Precondition of theorem:

    $\Longrightarrow \quad |e(x)| \geq |e'(x)| \qquad$ for all $x \in X$

Dekker's Theorem

- How to apply the theorem automatically?

can apply the theorem → $e \oplus e' \ominus e \ominus e' \sqsubseteq \left(A_{\vec{\delta}}(x) + A'_{\vec{\delta}}(x)\right)\delta$

  - Construct: $e \sqsubseteq A_{\vec{\delta}},\ e' \sqsubseteq A'_{\vec{\delta}},\ e \oplus e' \sqsubseteq \left(A_{\vec{\delta}}(x) + A'_{\vec{\delta}}(x)\right)(1+\delta)$
  - Check: $\min_{x,\vec{\delta}} \left|A_{\vec{\delta}}(x)\right| \geq \max_{x,\vec{\delta}} \left|A'_{\vec{\delta}}(x)\right|$ ← true

    $\Longrightarrow$ $\left|A_{\vec{\delta}}(x)\right| \geq \left|A'_{\vec{\delta}}(x)\right|$ for all $x \in X$, all $|\delta_i| < \epsilon$

  - Precondition of theorem:

    $\Longrightarrow$ $|e(x)| \geq |e'(x)|$ for all $x \in X$

More in Our Paper

- More complicated exactness results
  - Their explicit statements
  - How to apply them automatically

More in Our Paper

- More complicated exactness results
  - Their explicit statements
  - How to apply them automatically
  - We check **preconditions** of exactness results

More in Our Paper

- More complicated exactness results
  - Their explicit statements
  - How to apply them automatically
  - We check **preconditions** of exactness results

Solve
optimization problems

More in Our Paper

- More complicated exactness results
  - Their explicit statements
  - How to apply them automatically
  - We check **preconditions** of exactness results

Solve optimization problems

- Abstractions
  - In practice, we cannot use naïve abstractions because solving $\max_{x, \vec{\delta}} \left(\cdots A_{\vec{\delta}}(x) \cdots \right)$ is difficult

More in Our Paper

- More complicated exactness results
  - Their explicit statements
  - How to apply them automatically
  - We check **preconditions** of exactness results

Solve
optimization problems

- Abstractions
  - In practice, we cannot use naïve abstractions because solving $\max_{x, \vec{\delta}} \left(\cdots A_{\vec{\delta}}(x) \cdots \right)$ is difficult
  - We apply our **"compress" operation** to abstractions to reduce # of δ variables without losing much precision

More in Our Paper

- More complicated exactness results
  - Their explicit statements
  - How to apply them automatically
  - We check **preconditions** of exactness results

Solve
optimization problems

- Abstractions
  - In practice, we cannot use naïve abstractions because solving $\max_{x,\vec{\delta}} \left(\cdots A_{\vec{\delta}}(x) \cdots\right)$ is difficult
  - We apply our **"compress" operation** to abstractions to reduce # of δ variables without losing much precision

Case Studies

- Benchmarks
  - `sin`, `log`, `tan`: Intel's implementation of `math.h`
  - These are written directly in x86 assembly

Case Studies

- Benchmarks
  - `sin`, `log`, `tan`: Intel's implementation of `math.h`
  - These are written directly in x86 assembly

- Use bit-level / integer arithmetic operations (bit-shift, int add, ⋯)
  - Mixed in with floating-point operations

Case Studies

- Benchmarks
  - `sin`, `log`, `tan`: Intel's implementation of `math.h`
  - These are written directly in x86 assembly

- Use bit-level / integer arithmetic operations (bit-shift, int add, ⋯)
  - Mixed in with floating-point operations
  - Eliminate them using the technique from [PLDI'16]

  $\longrightarrow$ Results in multiple independent subproblems (on subintervals)

Case Studies

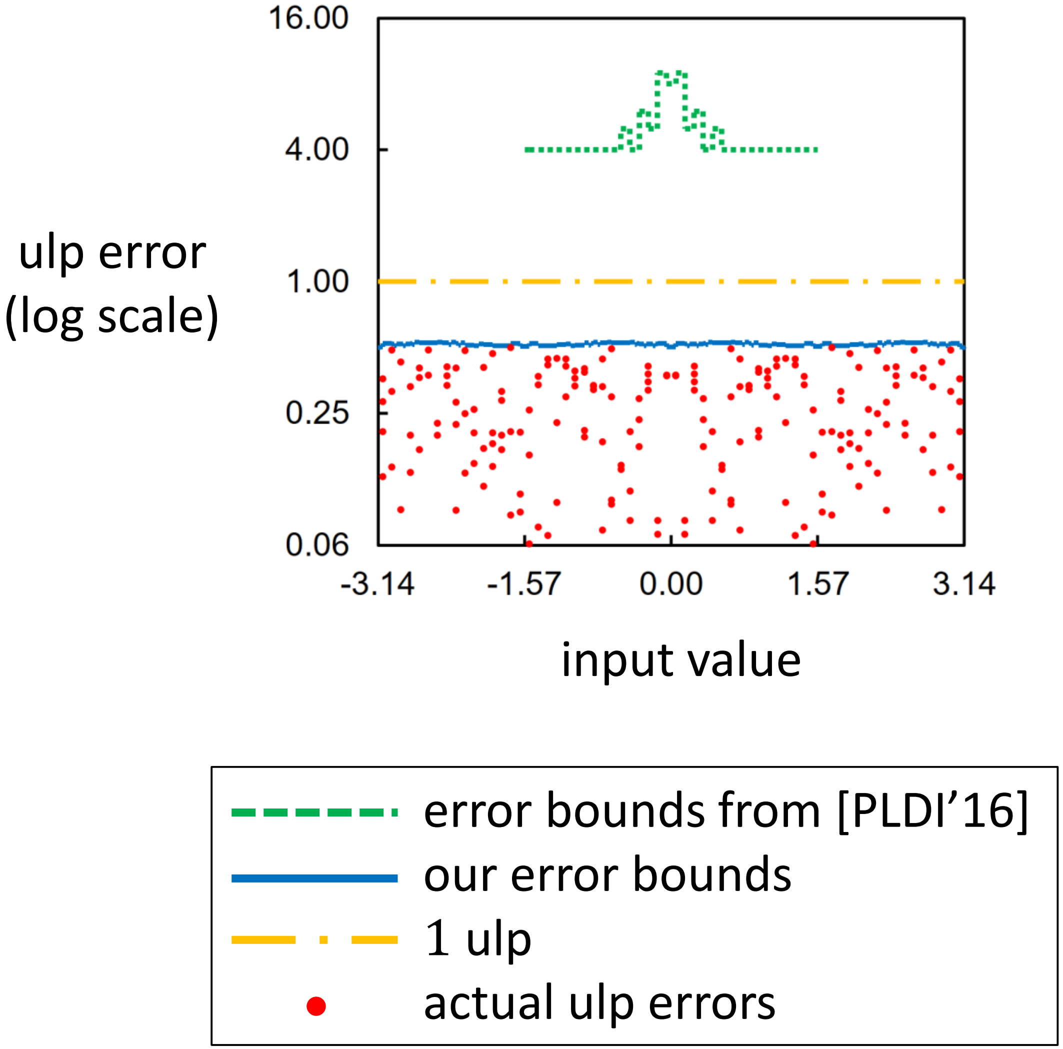
- Benchmarks
  - `sin`, `log`, `tan`: Intel's implementation of `math.h`
  - These are written directly in x86 assembly

- Use bit-level / integer arithmetic operations (bit-shift, int add, ⋯)
  - Mixed in with floating-point operations
  - Eliminate them using the technique from [PLDI'16]

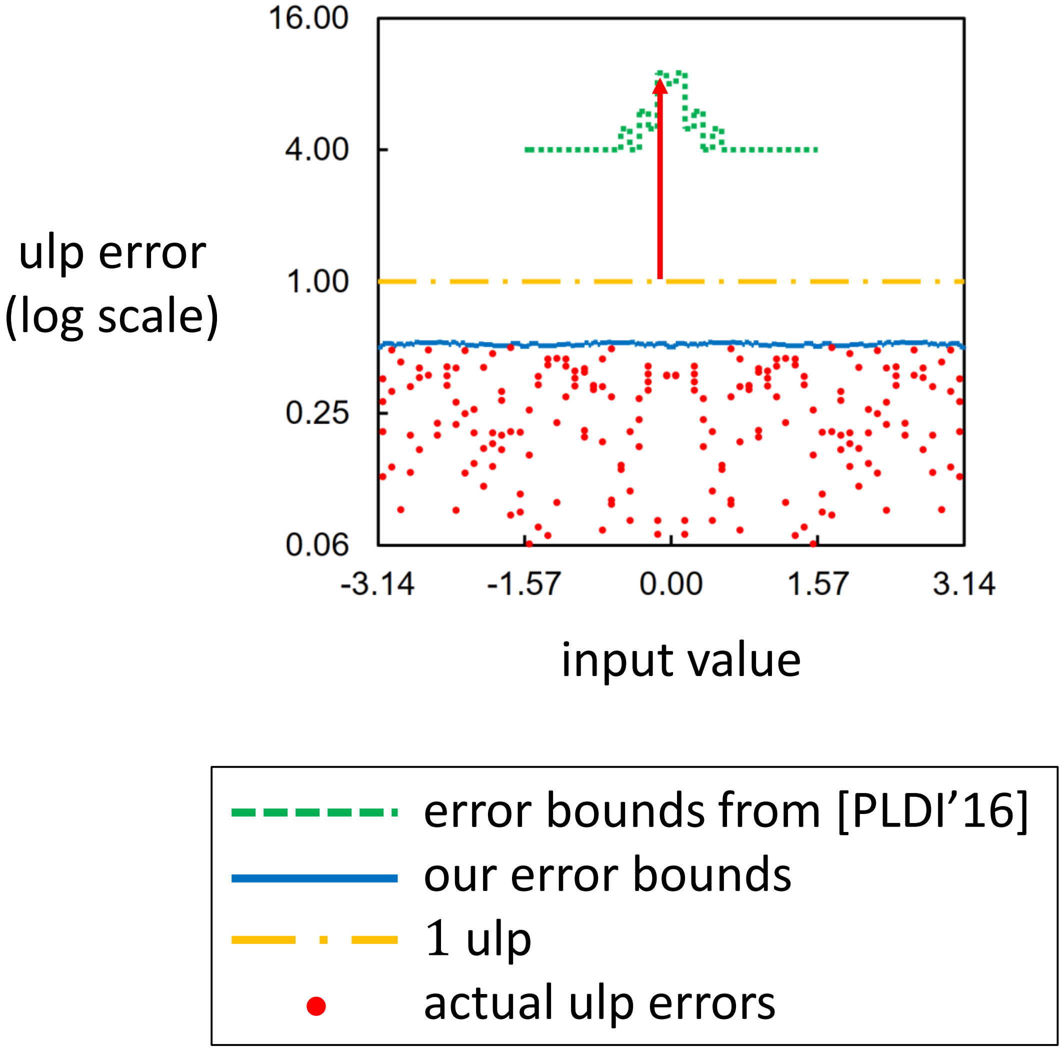
  $\longrightarrow$ Results in multiple independent subproblems (on subintervals)

- Apply our technique to the resulting floating-point expressions
  - Use Mathematica to solve optimization problems analytically

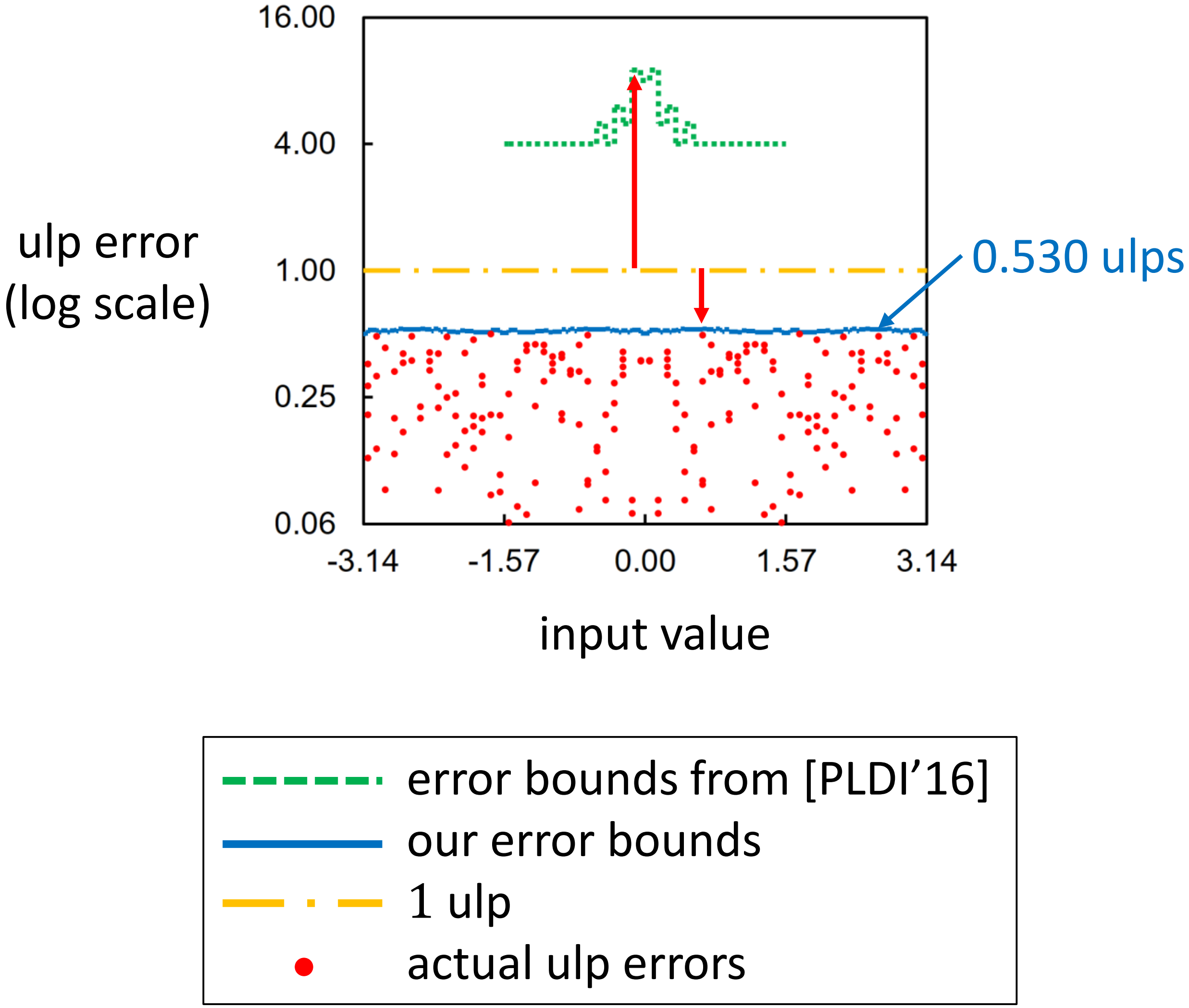
Results: `sin` on $[-\pi, \pi]$



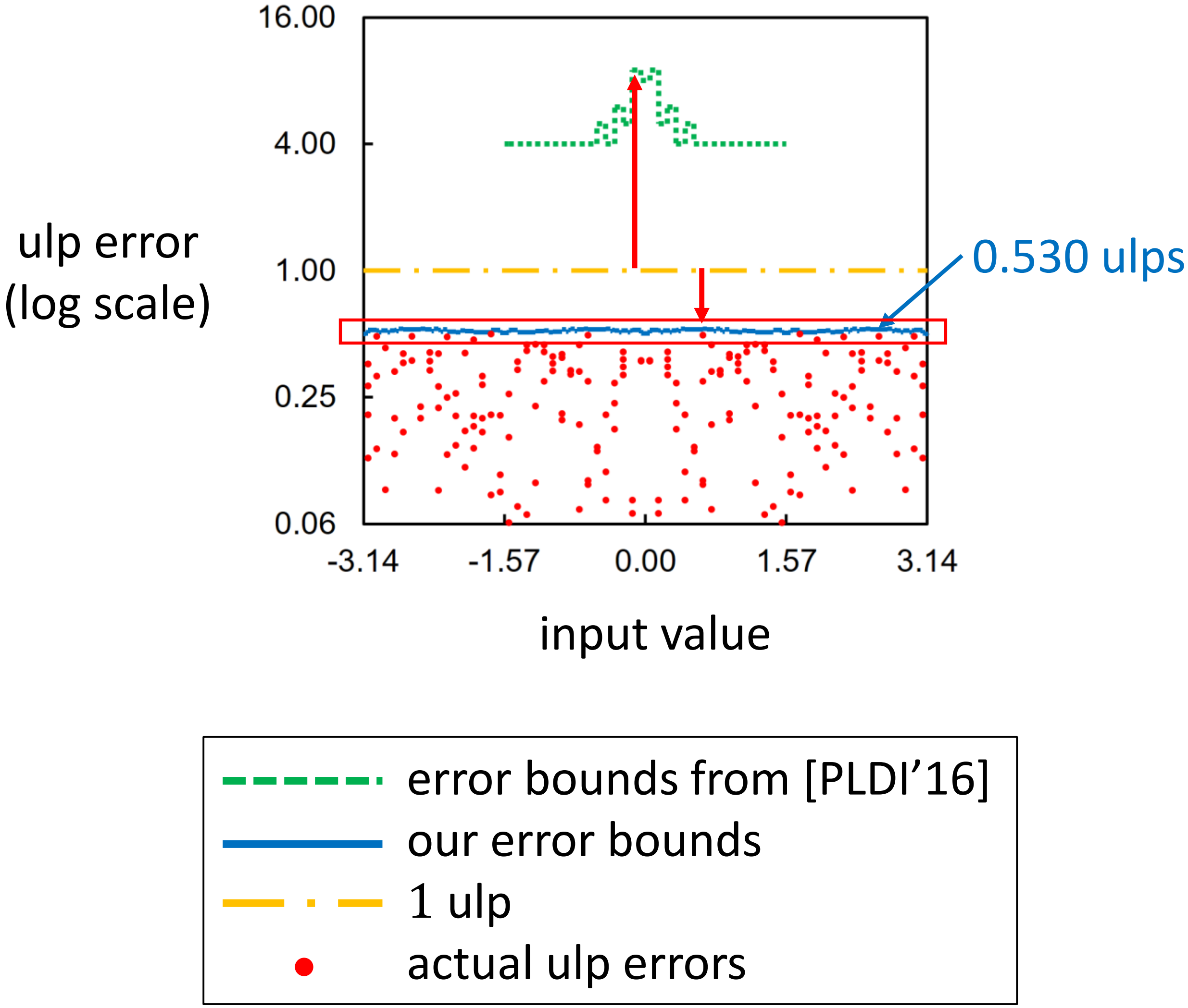
Results: `sin` on $[-\pi, \pi]$



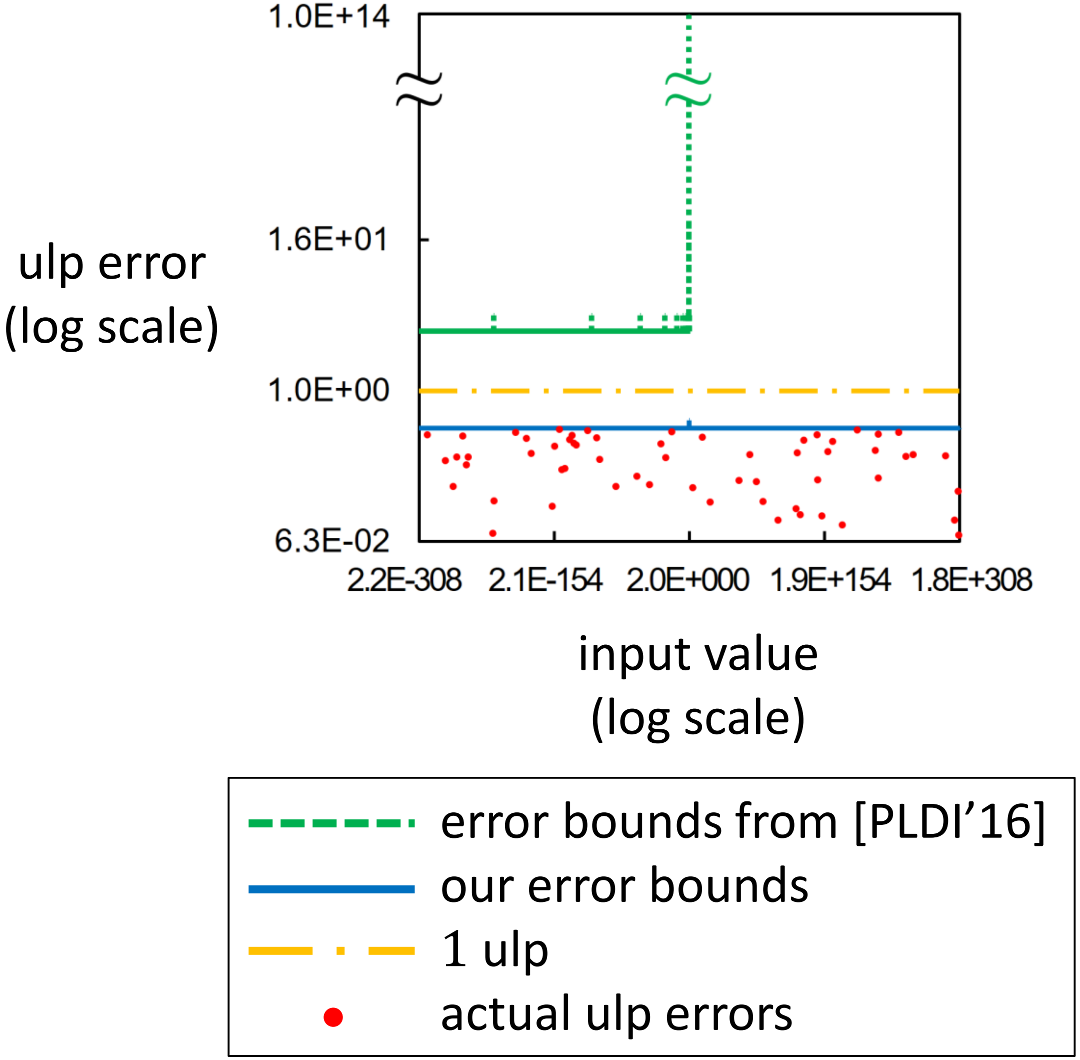
Results: `sin` on $[-\pi, \pi]$



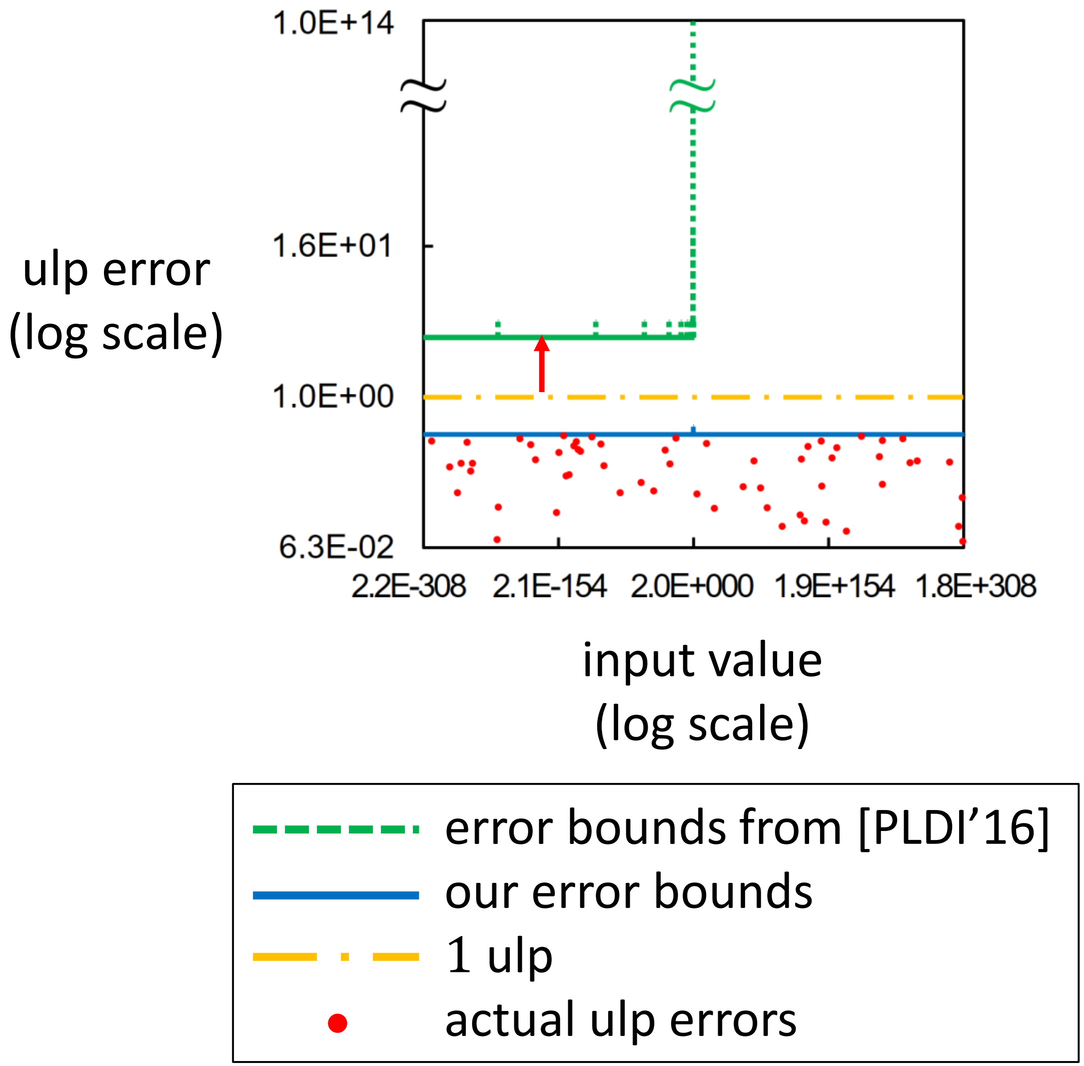
Results: `sin` on $[-\pi, \pi]$



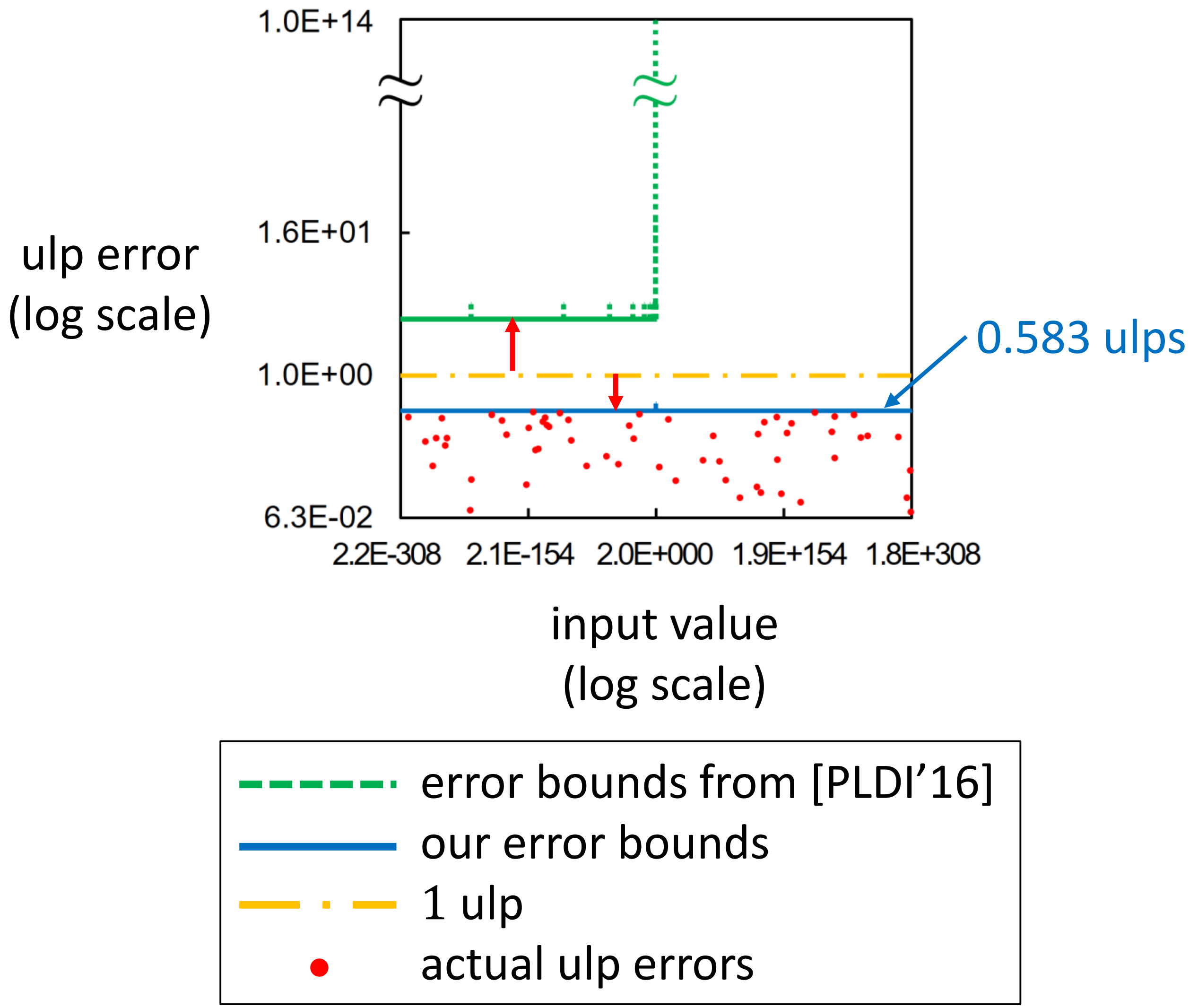
Results: `log` on $(2^{-1022}, 2^{1024})$



Results: `log` on $(2^{-1022}, 2^{1024})$

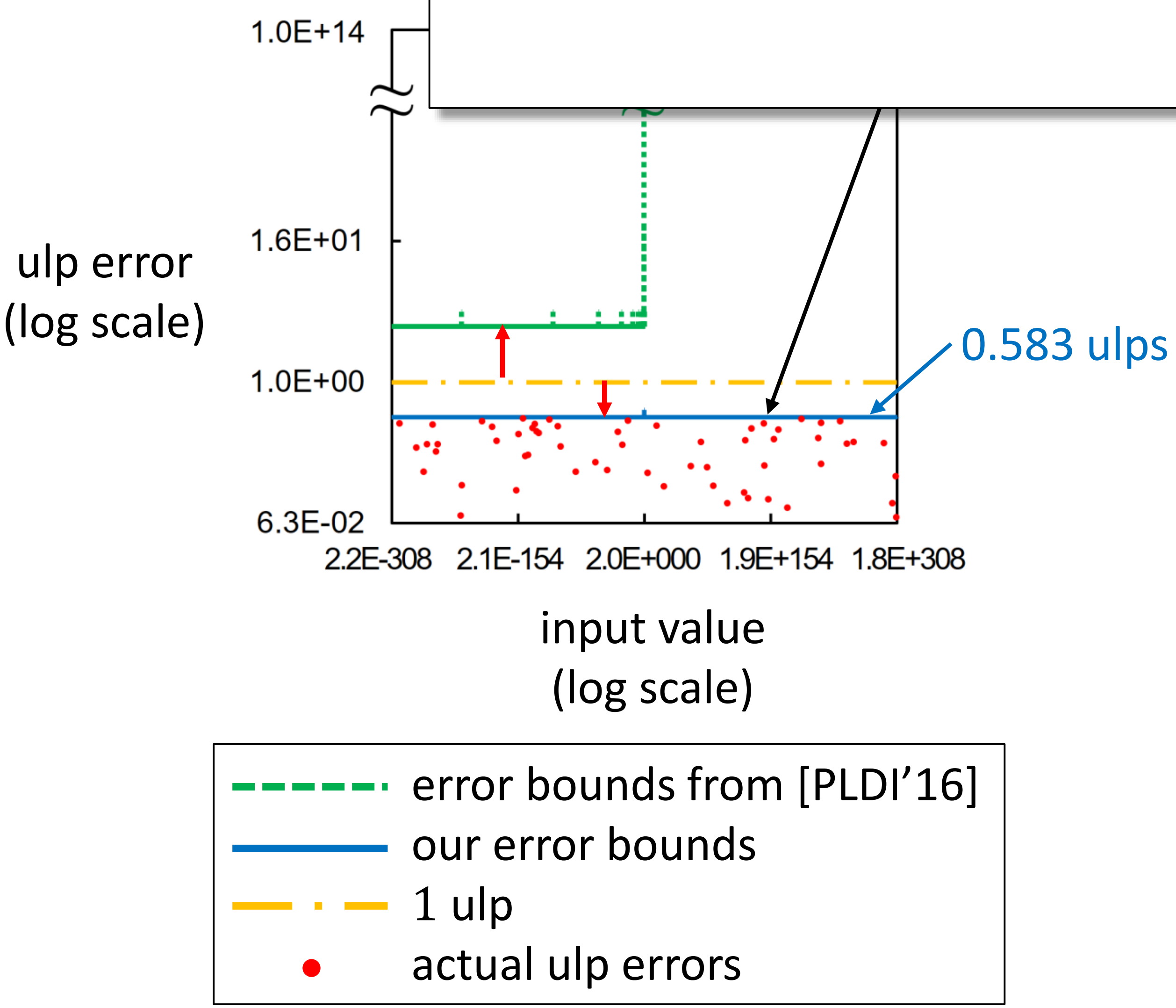


Results: `log` on $(2^{-1022}, 2^{1024})$



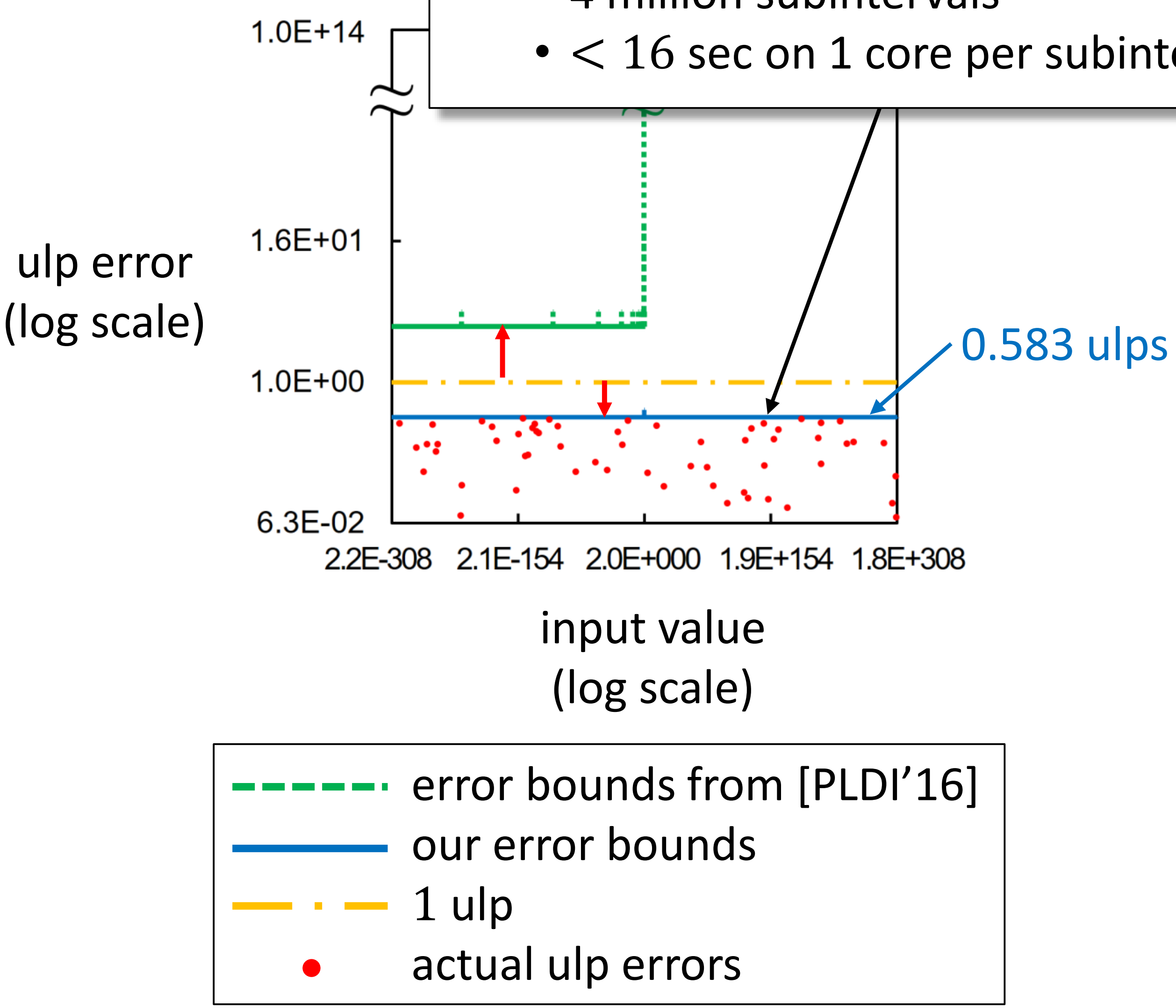
Results: `log` o

- 461 hours on 16 cores

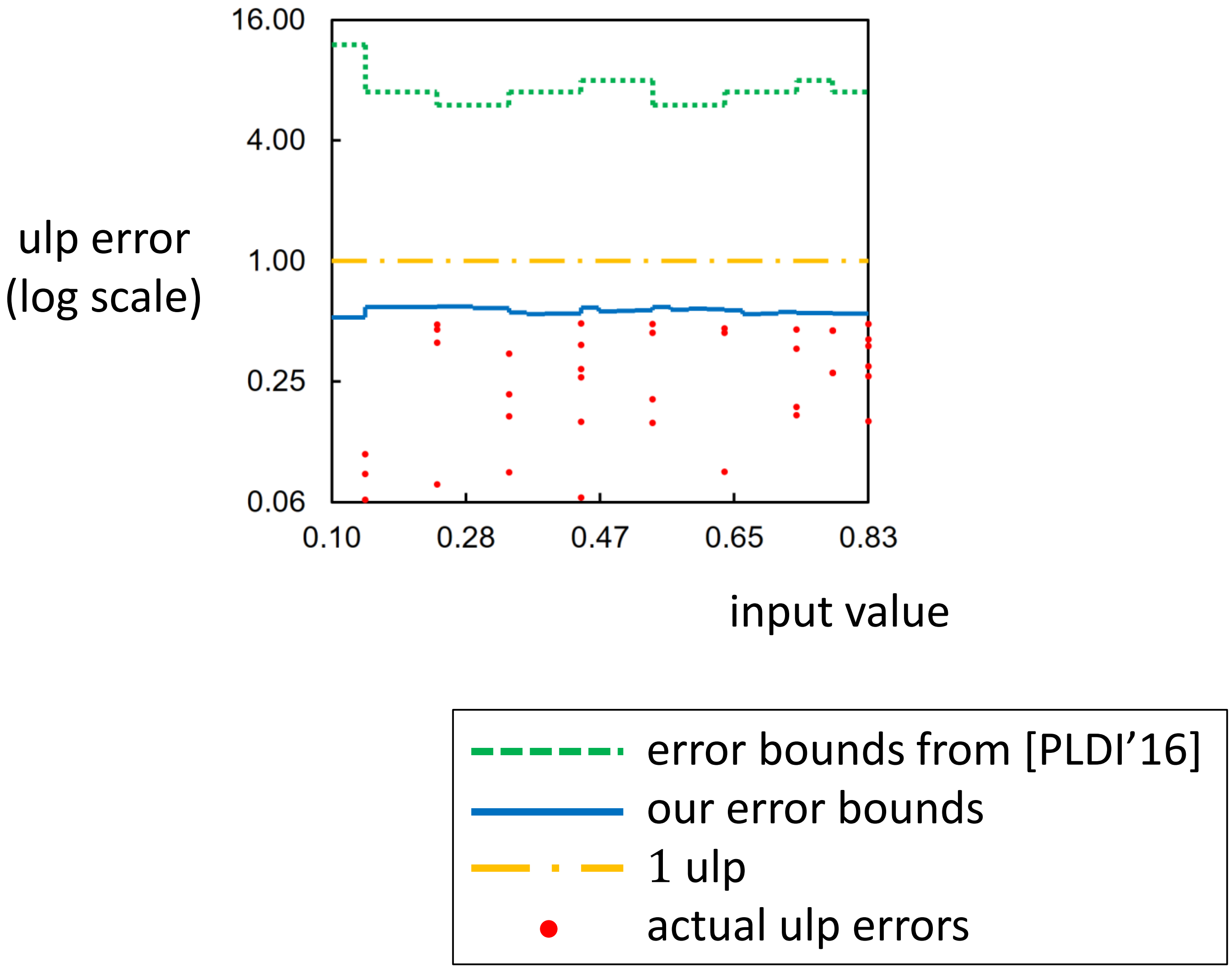


Results: `log` or

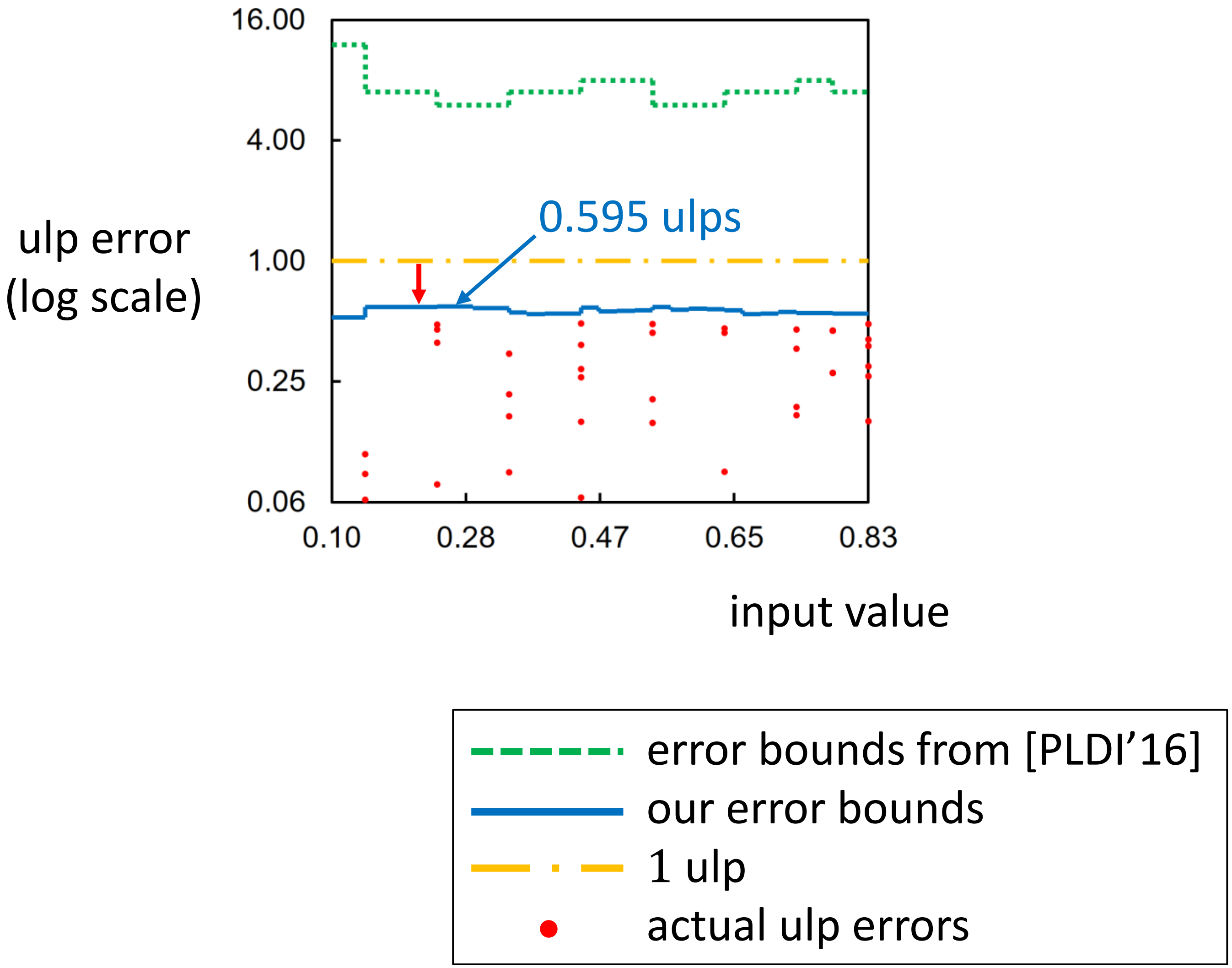
- 461 hours on 16 cores
- Highly parallelizable:
  - 4 million subintervals
  - < 16 sec on 1 core per subinterval



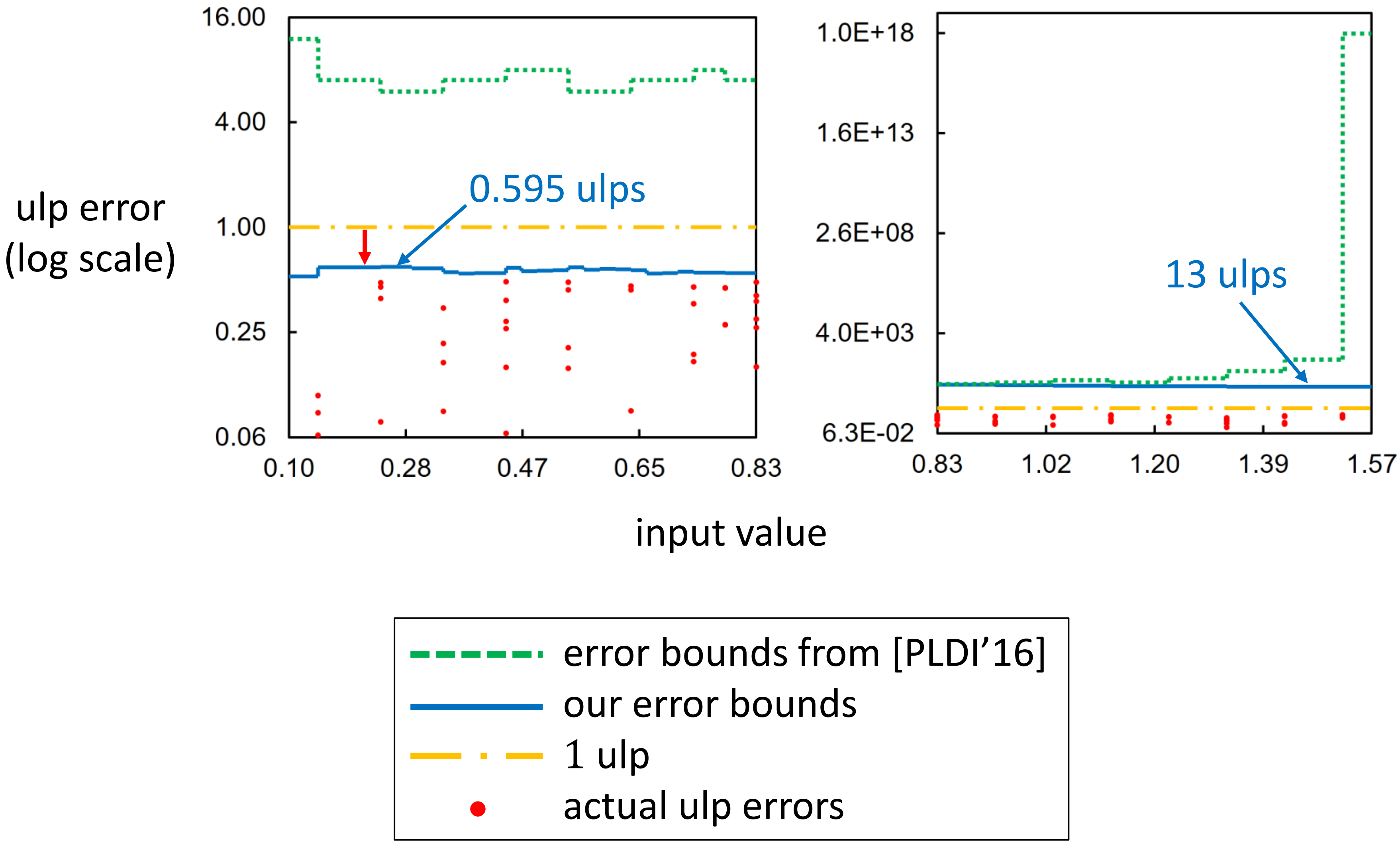
Results: tan on $\left[\frac{13}{128}, \frac{17\pi}{64}\right), \left[\frac{17\pi}{64}, \frac{\pi}{2}\right)$



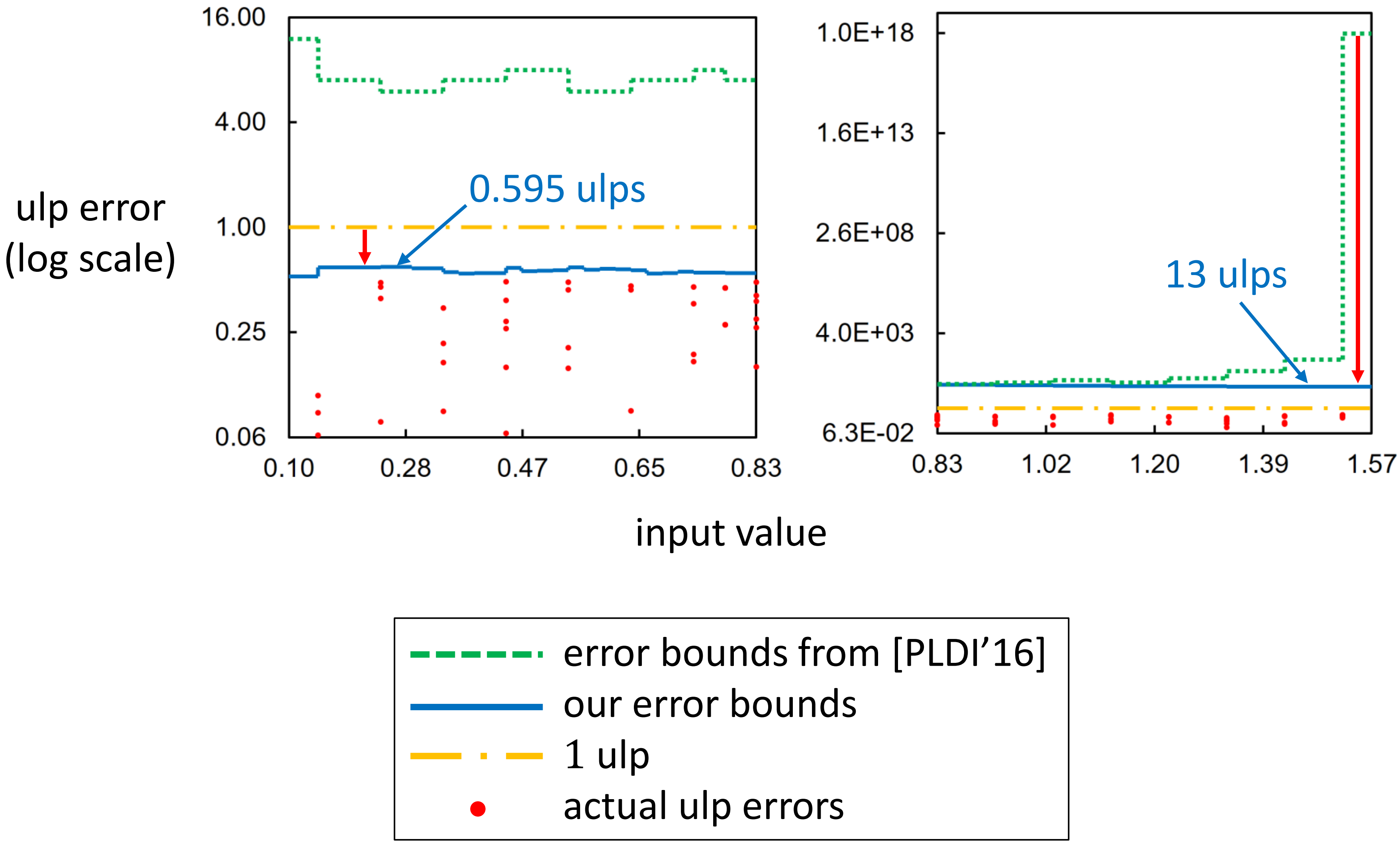
Results: `tan` on $\left[\frac{13}{128}, \frac{17\pi}{64}\right)$, $\left[\frac{17\pi}{64}, \frac{\pi}{2}\right)$



Results: `tan` on $\left[\frac{13}{128}, \frac{17\pi}{64}\right), \left[\frac{17\pi}{64}, \frac{\pi}{2}\right)$

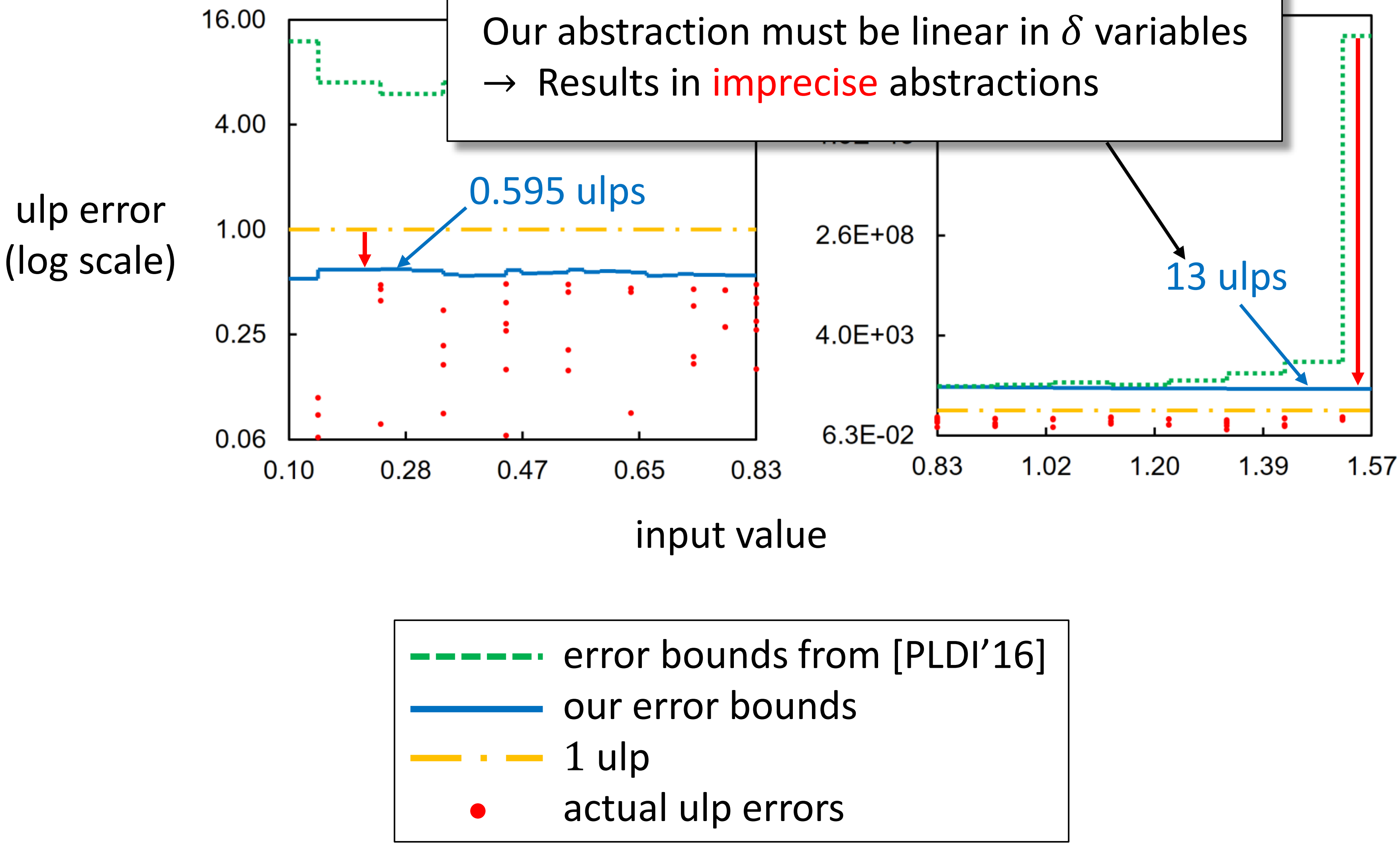


Results: `tan` on $\left[\frac{13}{128}, \frac{17\pi}{64}\right), \left[\frac{17\pi}{64}, \frac{\pi}{2}\right)$



Results: `tan` on $\left[\frac{13}{128}, \frac{17\pi}{64}\right), \left[\frac{17\pi}{64}, \frac{\pi}{2}\right)$

Our abstraction must be linear in δ variables
→ Results in imprecise abstractions



Summary

- A novel static analysis for verifying `math.h` implementations automatically

- A reduction of this verification task to optimization problems

- Our technique verified some of Intel's `math.h` implementations automatically

Summary

- A novel static analysis for verifying `math.h` implementations automatically

- A reduction of this verification task to optimization problems

- Our technique verified some of Intel's `math.h` implementations automatically

Questions?