A Reduced Product of Absolute and Relative Error Bounds of Floating-point Analysis

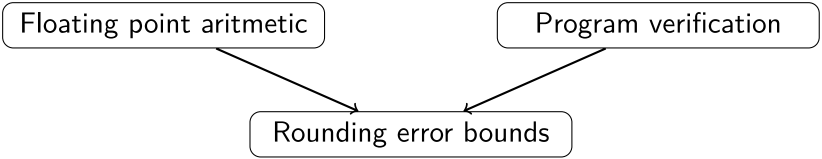
Maxime Jacquemin[1] Sylvie Putot[2] Franck Védrine[1]

[1]CEA, List, Software Reliability and Security Laboratory, PC 174
91191 Gif-Sur-Yvette France

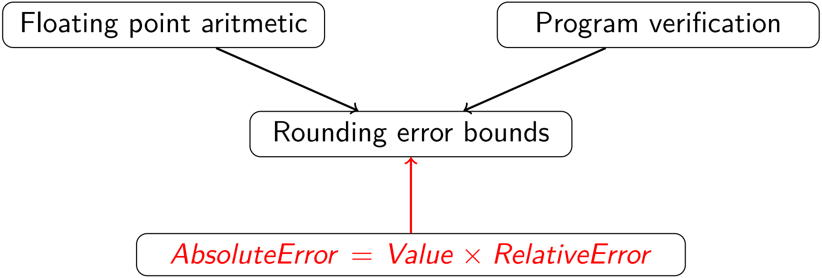
[2]LIX, CNRS and École Polytechnique, Palaiseau, France.

RAIM – November 14, 2018

Goal


Floating point aritmetic
Program verification
Rounding error bounds

Goal


Floating point aritmetic
Program verification
Rounding error bounds
AbsoluteError = Value × RelativeError

Overview

Program Verification

Testing

```
x = random(1,3) ;
y = random(1,3) ;
z = x * y ;

if (z <= 2)
  r = z * z ;

else
  r = 4 - z ;

assume(r <= 4) ;
assume(r <> -1) ;
```

Program Verification

Testing

```
x = random(1,3) ;
y = random(1,3) ;
z = x * y ;

if (z <= 2)
  r = z * z ;

else
  r = 4 - z ;

assume(r <= 4) ;
assume(r <> -1) ;
```

$x = 1$ $y = 2$

$x = 3$ $y = 2$

Program Verification

Testing

```
x = random(1,3) ;
y = random(1,3) ;
z = x * y ;

if (z <= 2)
  r = z * z ;

else
  r = 4 - z ;

assume(r <= 4) ;
assume(r <> -1) ;
```

$x = 1$	$x = 3$
$y = 2$	$y = 2$
$z = 1 \times 2 = 2$	$z = 3 \times 2 = 6$

Program Verification
Testing

`x = random(1,3) ;`	$x = 1$	$x = 3$
`y = random(1,3) ;`	$y = 2$	$y = 2$
`z = x * y ;`	$z = 1 \times 2 = 2$	$z = 3 \times 2 = 6$
`if (z <= 2)`	$z \leq 2 \Leftrightarrow \mathit{True}$	$z \leq 2 \Leftrightarrow \mathit{False}$
`  r = z * z ;`	$r = 2 \times 2 = 4$	
`else`		
`  r = 4 - z ;`		$r = 4 - 6 = -2$
`assume(r <= 4) ;`		
`assume(r <> -1) ;`		

Program Verification

Testing

`x = random(1,3) ;`	$x = 1$	$x = 3$
`y = random(1,3) ;`	$y = 2$	$y = 2$
`z = x * y ;`	$z = 1 \times 2 = 2$	$z = 3 \times 2 = 6$
`if (z <= 2)`	$z \leq 2 \Leftrightarrow \mathit{True}$	$z \leq 2 \Leftrightarrow \mathit{False}$
`  r = z * z ;`	$r = 2 \times 2 = 4$	
`else`		
`  r = 4 - z ;`		$r = 4 - 6 = -2$
`assume(r <= 4) ;`	$r \leq 4 \Leftrightarrow \mathit{True}$	$r \leq 4 \Leftrightarrow \mathit{True}$
`assume(r <> -1) ;`	$r \neq -1 \Leftrightarrow \mathit{True}$	$r \neq -1 \Leftrightarrow \mathit{True}$

Program Verification

Concrete semantic

```
x = random(1,3) ;
y = random(1,3) ;
z = x * y ;

if (z <= 2)
  r = z * z ;

else
  r = 4 - z ;

assume(r <= 4) ;
assume(r <> -1) ;
```

$x = \{1, 2, 3\}$

$y = \{1, 2, 3\}$

$z = \{1, 2, 3, 4, 6, 9\}$

$z \leq 2 \Rightarrow z = \{1, 2\}$

$r = \{1, 4\}$

$z > 2 \Rightarrow z = \{3, 4, 6, 9\}$

$r = \{-5, -2, 0, 1\}$

$r = \{-5, -2, 0, 1, 4\} \leq 4 \Leftrightarrow \mathit{True}$

$-1 \notin r = \{-5, -2, 0, 1, 4\} \Leftrightarrow \mathit{True}$

Program Verification

Some kind of trouble

- Concret semantic is not always decidable
- When it does, it is really costly
- Need of a simplification!

Program Verification

Abstract interpretation

- Abstract the set of possible value by something simpler:
- Do not miss any possible concrete value
- Accept to contain infeasable values
- Redefine the semantics for the abstraction

Program Verification

Abstract semantic: intervals

```
x = random(1,3) ;
y = random(1,3) ;
z = x * y ;

if (z <= 2)
  r = z * z ;

else
  r = 4 - z ;

assume(r <= 4) ;
assume(r <> -1) ;
```

$x = [1..3]$

$y = [1..3]$

Program Verification

Abstract semantic: intervals

```
x = random(1,3) ;
y = random(1,3) ;
z = x * y ;

if (z <= 2)
  r = z * z ;

else
  r = 4 - z ;

assume(r <= 4) ;
assume(r <> -1) ;
```

$x = [1..3]$

$y = [1..3]$

$z = [1..3] \times [1..3] = [1..9]$

Program Verification

Abstract semantic: intervals

```
x = random(1,3) ;
y = random(1,3) ;
z = x * y ;

if (z <= 2)
  r = z * z ;

else
  r = 4 - z ;

assume(r <= 4) ;
assume(r <> -1) ;
```

$x = [1..3]$

$y = [1..3]$

$z = [1..3] \times [1..3] = [1..9]$

$z \leq 2 \Rightarrow z = [1..2]$

$r = [1..2] \times [1..2] = [1..4]$

Program Verification

Abstract semantic: intervals

```
x = random(1,3) ;
y = random(1,3) ;
z = x * y ;

if (z <= 2)
  r = z * z ;

else
  r = 4 - z ;

assume(r <= 4) ;
assume(r <> -1) ;
```

$x = [1..3]$

$y = [1..3]$

$z = [1..3] \times [1..3] = [1..9]$

$z \leq 2 \Rightarrow z = [1..2]$

$r = [1..2] \times [1..2] = [1..4]$

$z > 2 \Rightarrow z = [3..9]$

$r = [4..4] - [3..9] = [-5..1]$

Program Verification

Abstract semantic: intervals

```
x = random(1,3) ;
y = random(1,3) ;
z = x * y ;

if (z <= 2)
  r = z * z ;

else
  r = 4 - z ;

assume(r <= 4) ;
assume(r <> -1) ;
```

$x = [1..3]$

$y = [1..3]$

$z = [1..3] \times [1..3] = [1..9]$

$z \leq 2 \Rightarrow z = [1..2]$

$r = [1..2] \times [1..2] = [1..4]$

$z > 2 \Rightarrow z = [3..9]$

$r = [4..4] - [3..9] = [-5..1]$

$r = [1..4] \sqcup [-5..1] = [-5..4]$

Program Verification

Abstract semantic: intervals

```
x = random(1,3) ;
y = random(1,3) ;
z = x * y ;

if (z <= 2)
  r = z * z ;

else
  r = 4 - z ;

assume(r <= 4) ;
assume(r <> -1) ;
```

$x = [1..3]$

$y = [1..3]$

$z = [1..3] \times [1..3] = [1..9]$

$z \leq 2 \Rightarrow z = [1..2]$

$r = [1..2] \times [1..2] = [1..4]$

$z > 2 \Rightarrow z = [3..9]$

$r = [4..4] - [3..9] = [-5..1]$

$r = [1..4] \sqcup [-5..1] = [-5..4]$

$r = [-5..4] \leq 4 \Leftrightarrow \mathit{True}$

$-1 \notin r = [-5..4] \Leftrightarrow \mathit{Maybe}$

Idea

- Classical methods focused on absolute errors
- Natural errors bounds are on relative errors

> Improving absolute errors bounds using reduced product with relative errors bounds

Motivating Example

$(\mathit{Value}, \mathit{AbsoluteError}, \mathit{RelativeError})$

```
t = [1, 100] ;
x = t * t ;
if (x <= 2.0) r = x ;
```

$t = ([1, 100], 0, 0)$

Motivating Example

$(Value, AbsoluteError, RelativeError)$

```
t = [1, 100] ;
x = t * t ;
if (x <= 2.0) r = x ;
```

$t = ([1, 100], 0, 0)$
$x = ([1, 10^4], \pm 10^{-12}, \pm 10^{-16})$

Motivating Example

$$(Value, AbsoluteError, RelativeError)$$

```
t = [1, 100] ;
x = t * t ;
if (x <= 2.0) r = x ;
```

$$t = ([1, 100], 0, 0)$$
$$x = ([1, 10^4], \pm 10^{-12}, \pm 10^{-16})$$
$$r = ([1, 2], \pm 10^{-12}, \pm 10^{-16})$$

Motivating Example

$$(\mathit{Value}, \mathit{AbsoluteError}, \mathit{RelativeError})$$

```
t = [1, 100] ;
x = t * t ;
if (x <= 2.0) r = x ;
```

$$t = ([1, 100], 0, 0)$$
$$x = ([1, 10^4], \pm 10^{-12}, \pm 10^{-16})$$
$$r = ([1, 2], \pm 10^{-12}, \pm 10^{-16})$$

$$\mathit{AbsoluteError} = \mathit{Value} \times \mathit{RelativeError}$$

Motivating Example

$$(\mathit{Value}, \mathit{AbsoluteError}, \mathit{RelativeError})$$

```
t = [1, 100] ;
x = t * t ;
if (x <= 2.0) r = x ;
```

$$t = ([1, 100], 0, 0)$$
$$x = ([1, 10^4], \pm 10^{-12}, \pm 10^{-16})$$
$$r = ([1, 2], \pm 2 \times 10^{-16}, \pm 10^{-16})$$

$$\mathit{AbsoluteError} = \mathit{Value} \times \mathit{RelativeError}$$

Modeling Rounding Errors

Rounding to nearest model

$$\mathrm{rnd}(x) = x + \mathrm{ufp}(x) e_x + d_x$$

- $e_x =$ relative error when rounding to normalized
- $d_x =$ absolute error when rounding to subnormal
- Bounded by the format
  - Simple precision : $|e_x| \leq 2^{-24}$ and $|d_x| \leq 2^{-149}$
  - Double precision : $|e_x| \leq 2^{-53}$ and $|d_x| \leq 2^{-1075}$
- ufp = unit in the first place

Modeling Rounding Errors

Errors definitions

Let $\widetilde{x}$ be an approximation of $x \in \mathbb{R}$, we define:

$$\mathcal{E}_a(x) = \widetilde{x} - x \text{ the absolute error}$$
$$\mathcal{E}_r(x) = \frac{\widetilde{x} - x}{x} \text{ the relative error}$$

The relative error is **not** defined if $x = 0$

Modeling Rounding Errors

Elementary rounding error

Let the approximation $\widetilde{x}$ be the result of the rounding operator applied to x. We define the *elementary rounding errors* of x as:

$$\Gamma_a(x) = \mathsf{ufp}(x)e_x + d_x \quad \text{and} \quad \Gamma_r(x) = \frac{\mathsf{ufp}(x)e_x + d_x}{x}$$

Modeling Rounding Errors

Arithmetic operations rounding errors

Let $\widetilde{X} \in \mathbb{F}^k$ be the approximation of $X \in \mathbb{R}^k$ and $\widetilde{op}$ be the floating point counterpart of $op : \mathbb{R}^k \to \mathbb{R}$. We define:

$$\mathcal{E}_a(op(X)) = \widetilde{op}(\widetilde{X}) - op(X)$$

$$\mathcal{E}_r(op(X)) = \frac{\widetilde{op}(\widetilde{X}) - op(X)}{op(X)}$$

Modeling Rounding Errors

Correctly rounded operations errors

If $\widetilde{op}$ is a *correctly rounded* operation:

$$\mathcal{E}_a(op(X)) = op(\widetilde{X}) + \Gamma_a(\widetilde{op}(\widetilde{X})) - op(X)$$

$$\mathcal{E}_r(op(X)) = \frac{op(\widetilde{X})}{op(X)}(1 + \Gamma_r(\widetilde{op}(\widetilde{X}))) - 1$$

Abstraction

Propagated values

- Relying on lattice of intervals:
  - Real range $\boldsymbol{x}$
  - Floating point range $\widetilde{\boldsymbol{x}}$
  - Absolute error range $\mathcal{E}_a(\boldsymbol{x})$
  - Relative error range $\mathcal{E}_r(\boldsymbol{x})$
- Complete lattice with a componentwise join
- Stable test assumption

Abstraction

Transfer functions

- Intervals $\Rightarrow$ lost of correlations
- Reorganizing terms to reduce variable repetitions

Abstraction

Absolute error of the division

$$\begin{aligned}
\mathcal{E}_a(\widetilde{x} \div \widetilde{y}) &= (\widetilde{x} \div \widetilde{y}) + \Gamma_a(\widetilde{x} \div \widetilde{y}) - (x \div y) \\
&= \frac{y\widetilde{x} - x\widetilde{y}}{y\widetilde{y}} + \Gamma_a(\widetilde{x} \div \widetilde{y}) \\
&= \frac{y(x + \mathcal{E}_a(x)) - x(y + \mathcal{E}_a(y))}{y\widetilde{y}} + \Gamma_a(\widetilde{x} \div \widetilde{y}) \\
&= \frac{\mathcal{E}_a(x) - x\frac{\mathcal{E}_a(y)}{y}}{\widetilde{y}} + \Gamma_a(\widetilde{x} \div \widetilde{y})
\end{aligned}$$

Abstraction

Absolute error of the division

$$
\begin{aligned}
\mathcal{E}_a(\widetilde{x} \div \widetilde{y}) &= (\widetilde{x} \div \widetilde{y}) + \Gamma_a(\widetilde{x} \div \widetilde{y}) - (x \div y) \\
&= \frac{y\widetilde{x} - x\widetilde{y}}{y\widetilde{y}} + \Gamma_a(\widetilde{x} \div \widetilde{y}) \\
&= \frac{y(x + \mathcal{E}_a(x)) - x(y + \mathcal{E}_a(y))}{y\widetilde{y}} + \Gamma_a(\widetilde{x} \div \widetilde{y}) \\
&= \frac{\mathcal{E}_a(x) - x\frac{\mathcal{E}_a(y)}{y}}{\widetilde{y}} + \Gamma_a(\widetilde{x} \div \widetilde{y})
\end{aligned}
$$

Abstraction

Absolute error of the division

$$\begin{aligned}
\mathcal{E}_a(\widetilde{x} \div \widetilde{y}) &= (\widetilde{x} \div \widetilde{y}) + \Gamma_a(\widetilde{x} \div \widetilde{y}) - (x \div y) \\
&= \frac{y\widetilde{x} - x\widetilde{y}}{y\widetilde{y}} + \Gamma_a(\widetilde{x} \div \widetilde{y}) \\
&= \frac{y(x + \mathcal{E}_a(x)) - x(y + \mathcal{E}_a(y))}{y\widetilde{y}} + \Gamma_a(\widetilde{x} \div \widetilde{y}) \\
&= \frac{\mathcal{E}_a(x) - x\frac{\mathcal{E}_a(y)}{y}}{\widetilde{y}} + \Gamma_a(\widetilde{x} \div \widetilde{y}) \\
&= \frac{\mathcal{E}_a(x) - x{\color{red}\mathcal{E}_r(y)}}{\widetilde{y}} + \Gamma_a(\widetilde{x} \div \widetilde{y})
\end{aligned}$$

Abstraction

Reduced product

Propagating the relative error is useful in two ways :

- After each $\boldsymbol{z} = op(\boldsymbol{x}, \boldsymbol{y})$, we can perform a reduction:

$$\begin{cases} \mathcal{E}_a(\boldsymbol{z}) := \mathcal{E}_a(\boldsymbol{z}) \cap \mathcal{E}_r(\boldsymbol{z})\boldsymbol{z} \\ \mathcal{E}_r(\boldsymbol{z}) := \mathcal{E}_r(\boldsymbol{z}) \cap \dfrac{\mathcal{E}_a(\boldsymbol{z})}{\boldsymbol{z}} \text{ whenever } 0 \notin \boldsymbol{z} \end{cases}$$

- Absolute error of some operations naturally involves the relative errors of their operands:

$$\mathcal{E}_a(\boldsymbol{x} \div \boldsymbol{y}) = \frac{\mathcal{E}_a(\boldsymbol{x}) - \boldsymbol{x}\mathcal{E}_r(\boldsymbol{y})}{\widetilde{\boldsymbol{y}}} + \Gamma_a(\widetilde{\boldsymbol{x}} \div \widetilde{\boldsymbol{y}})$$

Abstraction

Example

Compute error bounds for $\frac{t}{t+1}$ with $t := ([0, 999], 0, 0)$

Abstraction

Example

Compute error bounds for $\dfrac{t}{t+1}$ with $t := ([0, 999], 0, 0)$

1. Compute error bounds for $t + 1$

$$\mathcal{E}_a(\boldsymbol{t}+1) = \mathcal{E}_a(\boldsymbol{t}) + \mathcal{E}_a(1) + \Gamma_a(\widetilde{\boldsymbol{t}}+1) = [-5.68e^{-14}, 5,68e^{-14}]$$

$$\mathcal{E}_r(\boldsymbol{t}+1) = \left(\frac{\mathcal{E}_r(\boldsymbol{t}) - \mathcal{E}_r(1)}{1+1/\boldsymbol{t}} + \mathcal{E}_r(1)\right)(1+\Gamma_r(\widetilde{\boldsymbol{t}}+1)) + \Gamma_r(\widetilde{\boldsymbol{t}}+1)$$
$$= [-1.1e^{-16}, 1.1e^{-16}]$$

Abstraction

Example

Compute error bounds for $\frac{t}{t+1}$ with $t := ([0, 999], 0, 0)$

2. Compute absolute error bounds for $\frac{t}{t+1}$

$$\mathcal{E}_a\left(\frac{\boldsymbol{t}}{\boldsymbol{t}+1}\right) = \frac{\boldsymbol{t}\mathcal{E}_r(\boldsymbol{t}+1)}{\boldsymbol{t}+1} + \Gamma_a\left(\frac{\widetilde{\boldsymbol{t}}}{\widetilde{\boldsymbol{t}}+1}\right) = [-1.67e^{-13}, 1.67e^{-13}]$$

$$\left([-5.68e^{-11}, 5.68e^{-11}] \text{ without reduced product}\right)$$

Abstraction

Example

Compute error bounds for $\dfrac{t}{t+1}$ with $t := ([0, 999], 0, 0)$

3. Compute relative error bounds for $\dfrac{t}{t+1}$

$$\begin{aligned}\mathcal{E}_r\left(\frac{\boldsymbol{t}}{\boldsymbol{t}+1}\right) &= \frac{\mathcal{E}_r(\boldsymbol{t})+1}{\mathcal{E}_r(\boldsymbol{t}+\boldsymbol{1})+1}\left(1+\Gamma_r\left(\frac{\widetilde{\boldsymbol{t}}}{\widetilde{\boldsymbol{t}}+1}\right)\right)-1 \\ &= \frac{1}{[-1.1e^{-16}, 1.1e^{-16}]+1}\left(1+\Gamma_r\left(\frac{\widetilde{\boldsymbol{t}}}{\widetilde{\boldsymbol{t}}+1}\right)\right)-1\end{aligned}$$

Bounded by $[-1, 1]$ because of the subnormals near 0

Abstraction

Handling conditional statements

Let $\gamma(x_1, \cdots, x_n)$ be a conditional expression. The components of the abstract value $\boldsymbol{x}_\gamma$ of variable x after the interpretation of conditional γ are obtained as following:

- Values ranges computed by classical backward propagation
- Relative error bounds unchanged: $\mathcal{E}_r(\boldsymbol{x}_\gamma) = \mathcal{E}_r(\boldsymbol{x})$
- Absolute error bounds reduced: $\mathcal{E}_a(\boldsymbol{x}_\gamma) = \mathcal{E}_a(\boldsymbol{x}) \cap \boldsymbol{x}_\gamma \mathcal{E}_r(\boldsymbol{x})$

Experimental evaluations

- Implemented in the abstract domain **Numerors** of the Eva plugin of Frama-C (available at the next release)
- Evaluation against state of the art tools:
  - Fluctuat
  - FPTaylor
  - Daisy
- Compared using FPBench[1] benchmark and some new benchmarks

[1] http://fpbench.org

Experimental evaluations: Absolute errors

Name	Under-Approx	Numerors	Fluctuat Intervals	Fluctuat Affine	Daisy 1	Daisy 2	FPTaylor
log_approx	–	**6.25e-14**	*3.56e-11*	*3.56e-11*	–	–	–
conditional_ex	–	**2.22e-16**	*9.09e-13*	*9.09e-13*	–	–	*9.09e-13*
conditional_1	–	**8.43e-13**	*6.82e-12*	*6.82e-12*	–	–	2.09e-11
sqrt_1	2.11e-16	5.51e-15	3.72e-14	3.38e-14	3.72e-14	*4.52e-16*	**2.75e-16**
complex_sqrt	5.00e-16	*1.29e-15*	3.93e-15	2.52e-15	3.92e-15	1.89e-15	**5.70e-16**
kepler0	2.42e-13	*3.63e-13*	*3.63e-13*	*3.63e-13*	*3.63e-13*	7.15e-13	**3.18e-13**
intro_example	1.65e-16	1.68e-13	5.68e-11	5.67e-11	5.68e-11	*2.52e-16*	**1.67e-16**
sec4_example	3.25e-15	6.35e-11	1.16e-09	1.16e-09	1.16e-09	**7.00e-14**	*3.73e-13*
test02_sum8	4.00e-15	**6.22e-15**	**6.22e-15**	**6.22e-15**	**6.22e-15**	*9.55e-15*	**6.22e-15**
test05_nonlin1_r4	1.32e-12	2.78e-07	1.67e-06	1.67e-06	1.67e-06	**5.93e-11**	*2.21e-09*
test05_nonlin1_test2	8.29e-17	**8.33e-17**	**8.33e-17**	**8.33e-17**	**8.33e-17**	*1.39e-16*	**8.33e-17**
doppler1	6.13e-14	*1.62e-13*	3.45e-13	3.45e-13	3.91e-13	1.74e-13	**9.91e-14**
doppler2	1.14e-13	3.27e-13	8.78e-13	8.78e-13	9.78e-13	*3.18e-13*	**1.84e-13**
doppler3	4.16e-14	*8.50e-14*	1.36e-13	1.36e-13	1.60e-13	9.13e-14	**5.70e-14**
rigidBody1	1.79e-13	*2.40e-13*	*2.40e-13*	*2.40e-13*	*2.40e-13*	5.08e-13	**2.13e-13**
rigidBody2	1.81e-11	*2.31e-11*	*2.31e-11*	*2.31e-11*	*2.31e-11*	6.32e-11	**2.27e-11**
turbine1	4.30e-15	4.73e-14	6.04e-14	5.76e-14	6.04e-14	*2.80e-14*	**1.24e-14**
turbine2	4.41e-15	8.57e-15	8.57e-15	*8.54e-15*	8.57e-15	1.71e-14	**7.38e-15**
turbine3	3.22e-15	3.85e-14	4.72e-14	4.54e-14	4.72e-14	*1.65e-14*	**7.15e-15**
verhulst	1.70e-16	3.77e-16	3.77e-16	*3.00e-16*	3.80e-16	4.21e-16	**1.79e-16**
predatorPrey	8.79e-17	1.40e-16	1.40e-16	*1.38e-16*	1.41e-16	2.27e-16	**1.01e-16**
carbonGas	3.13e-09	2.00e-08	2.00e-08	1.58e-08	2.06e-08	*1.03e-08*	**4.96e-09**

	Numerors	Fluctuat Intervals	Fluctuat Affine	Daisy 1	Daisy 2	Daisy 3	FPTaylor
Times	0.271	0.059	0.049	4.220	652.062	16056.987	197.774

Experimental evaluations: Relative errors

Name	**Under-Approx**	**Numerors**	**Posteriori**	**Daisy 1**	**Daisy 2**	**Daisy 3**	**FPTaylor**
log_approx	–	∞	∞	–	–	–	–
conditional_ex	–	**1.11e-16**	9.09e-13	–	–	–	*1.13e-16*
conditional_1	–	*8.94e-16*	3.41e-12	–	–	–	**7.22e-16**
sqrt_1	3.42e-16	*6.61e-16*	6.83e-12	1.10e-12	1.02e-15	∞	**4.26e-16**
complex_sqrt	2.04e-16	*4.98e-16*	8.64e-14	4.01e-15	1.94e-15	∞	**2.64e-16**
kepler0	3.65e-16	1.20e-15	1.20e-15	1.20e-15	2.13e-15	*1.06e-15*	**5.71e-16**
intro_example	1.87e-16	**1.00**	∞	∞	∞	∞	∞
sec4_example	6.52e-15	**1.40e-13**	8.71e-06	8.71e-06	3.60e-13	*2.34e-13*	6.65e-12
test02_sum8	3.42e-16	*5.94e-16*	7.77e-16	7.77e-16	1.19e-15	7.73e-16	**4.82e-16**
test05_nonlin1_r4	2.63e-12	**5.55e-12**	5.00e-01	5.00e-01	*2.07e-10*	1.68e-05	3.46e-06
test05_nonlin1_test2	1.66e-16	*2.26e-16*	2.50e-16	2.50e-16	4.17e-16	2.96e-16	**1.69e-16**
doppler1	6.70e-16	*1.10e-15*	1.17e-11	1.33e-11	5.91e-12	1.26e-15	**9.69e-16**
doppler2	7.17e-16	*1.21e-15*	4.62e-11	5.14e-11	1.67e-11	1.37e-15	**9.13e-16**
doppler3	5.63e-16	*9.75e-16*	3.11e-13	3.65e-13	1.82e-13	1.14e-15	**7.36e-16**
rigidBody1	3.44e-16	*7.79e-16*	1.04e-12	1.04e-12	2.21e-12	9.76e-16	**4.39e-16**
rigidBody2	4.84e-16	*9.65e-16*	1.32e-15	1.32e-15	3.50e-15	1.17e-15	**6.27e-16**
turbine1	4.21e-16	3.05e-14	3.90e-14	3.90e-14	1.41e-14	*1.75e-15*	**7.95e-16**
turbine2	2.30e-16	*4.98e-16*	*4.98e-16*	*4.98e-16*	9.23e-16	6.92e-16	**3.97e-16**
turbine3	3.53e-16	7.50e-14	1.01e-13	1.01e-13	2.91e-14	*6.51e-15*	**2.40e-15**
verhulst	2.26e-16	*3.75e-16*	1.20e-15	1.21e-15	1.16e-15	4.59e-16	**2.41e-16**
predatorPrey	3.12e-16	*4.82e-16*	3.76e-15	3.77e-15	6.09e-15	6.87e-16	**3.58e-16**
carbonGas	3.39e-16	**7.16e-16**	9.52e-15	9.80e-15	2.40e-15	8.11e-16	*7.67e-16*

	Numerors	**Fluctuat** Intervals	**Fluctuat** Affine	**Daisy** 1	**Daisy** 2	**Daisy** 3	**FPTaylor**
Times	0.271	0.059	0.049	4.220	652.062	16056.987	197.774

Conclusion & Future Works

- Interval-based reduced product of absolute and relative errors
- Low additional cost – Great enhancement
- Next step: using relational abstraction
- Going further: local subdivisions, tracking cancellations