Odyssey Link

https://herbie-fp.github.io/odyssey/

Odyssey

An Interactive Workbench for Floating Point Analysis

Edward Misback & Ben Wang, University of Washington

In collaboration with the **Herbie project** (Pavel Panchekha, U of Utah, + Zachary Tatlock, UW)

PAUL G. ALLEN SCHOOL OF COMPUTER SCIENCE & ENGINEERING

THE UNIVERSITY OF UTAH®

HERBIE

$$(x+1)-x$$

(x+1)-x

```
Python 3.11.9 (tags/v3.11.9:de54cf5, Apr  2 2024,
 10:12:12) [MSC v.1938 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license"
for more information.
>>>
```

(x+1)-x

```
Python 3.11.9 (tags/v3.11.9:de54cf5, Apr  2 2024,
 10:12:12) [MSC v.1938 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license"
for more information.
>>> x = 1e16
```

(x+1)-x

```
Python 3.11.9 (tags/v3.11.9:de54cf5, Apr  2 2024,
 10:12:12) [MSC v.1938 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license"
for more information.
>>> x = 1e16
>>> (1e16 + 1.0) - 1e16
```

(x+1)-x

```
Python 3.11.9 (tags/v3.11.9:de54cf5, Apr  2 2024,
 10:12:12) [MSC v.1938 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license"
for more information.
>>> x = 1e16
>>> (1e16 + 1.0) - 1e16
0.0
```

(x+1)-x

```
Python 3.11.9 (tags/v3.11.9:de54cf5, Apr  2 2024,
 10:12:12) [MSC v.1938 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license"
for more information.
>>> x = 1e16
>>> (1e16 + 1.0) - 1e16
0.0
>>> (1e16 + 1.0) == 1e16
```

(x+1)-x

```
Python 3.11.9 (tags/v3.11.9:de54cf5, Apr  2 2024,
 10:12:12) [MSC v.1938 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license"
for more information.
>>> x = 1e16
>>> (1e16 + 1.0) - 1e16
0.0
>>> (1e16 + 1.0) == 1e16
True
```

Rounding error

Rounding error

LARGE + small = LARGE

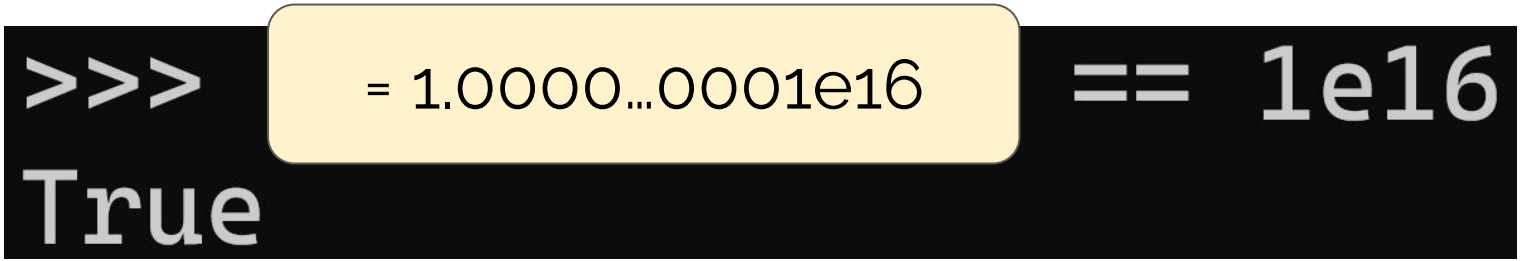
Rounding error

LARGE + small = LARGE

```
>>> (1e16 + 1.0) == 1e16
True
```

Rounding error

LARGE + small = LARGE

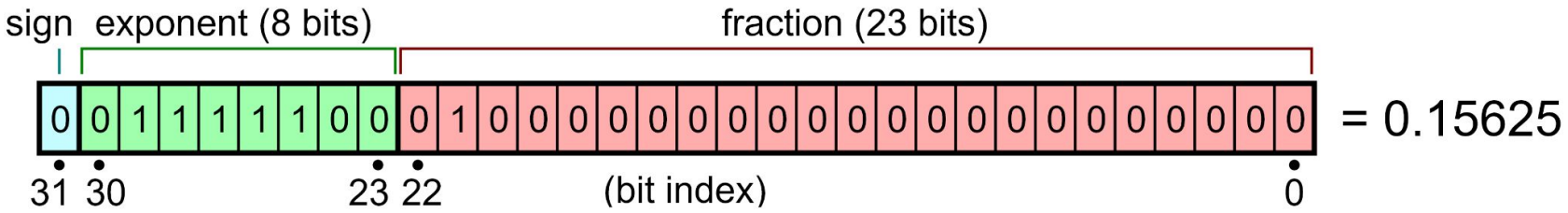


Rounding error

LARGE + small = LARGE

```
>>>                                    == 1e16
True
```

= 1.0000…0001e16



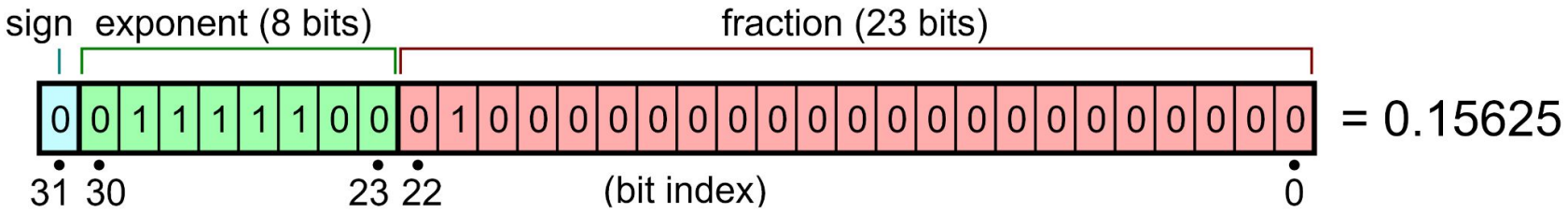
Rounding error

LARGE + small = LARGE

```
>>>                                    == 1e16
True
```

= **1.0000…0001**e16

Too long!



Catastrophic Cancellation

Catastrophic Cancellation

For A - B = C,
if magnitude(A) ≈ magnitude(B),
error in C will be much greater than
in A or B!

Catastrophic Cancellation

For A - B = C,
if magnitude(A) ≈ magnitude(B),
error in C will be much greater than
in A or B!

```
>>> (1e16 + 1.0) - 1e16
0.0
```

Catastrophic Cancellation

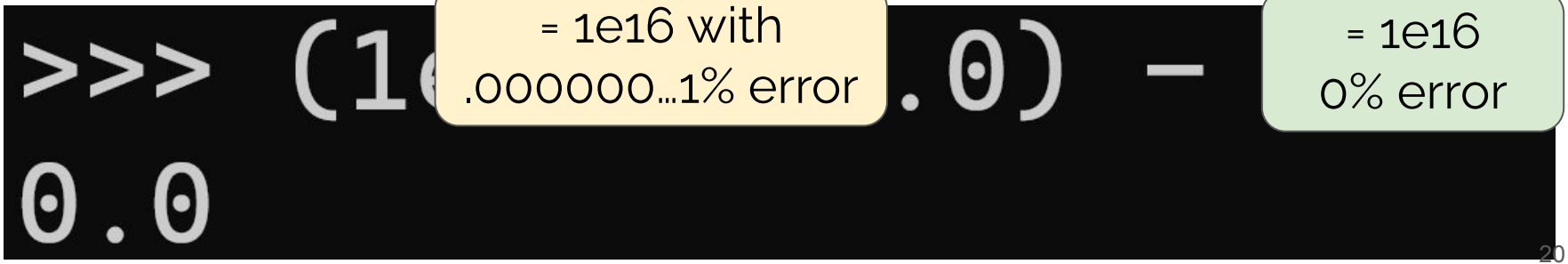
For A - B = C,
if magnitude(A) ≈ magnitude(B),
error in C will be much greater than
in A or B!

= 1e16 with
.000000…1% error

```
>>> (1e      .0) - 1e16
0.0
```

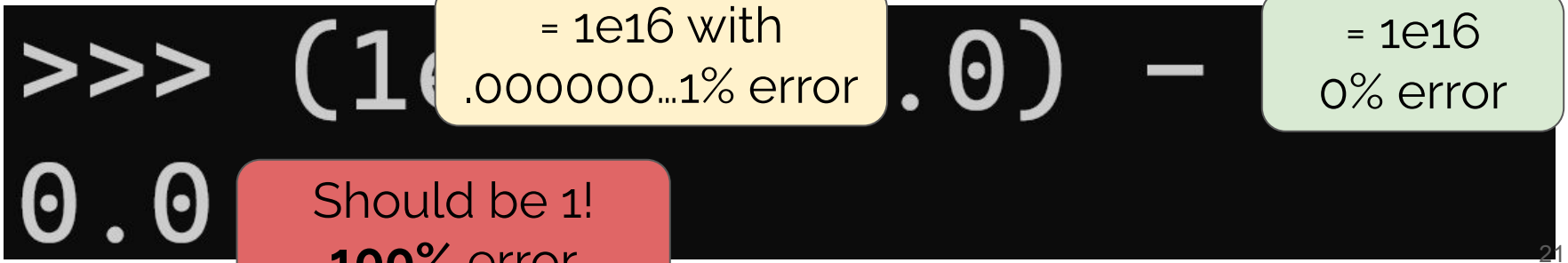
Catastrophic Cancellation

For A - B = C,
if magnitude(A) ≈ magnitude(B),
error in C will be much greater than
in A or B!



Catastrophic Cancellation

For A - B = C,
if magnitude(A) ≈ magnitude(B),
error in C will be much greater than
in A or B!



What is floating-point expression **rewriting**?

What is floating-point expression **rewriting**?

9th c.

The Compendious Book on Calculation by Completion and Balancing aka al-Jabr, by Al-Khwarizmi

What is floating-point expression **rewriting**?

9th c.

The Compendious Book on Calculation by Completion and Balancing aka al-Jabr, by Al-Khwarizmi

```
(x + 1) - x
```

What is floating-point expression **rewriting**?

9th c.

The Compendious Book on Calculation by Completion and Balancing aka al-Jabr, by Al-Khwarizmi

```
  (x + 1) - x
=(x - x) + 1
```

What is floating-point expression **rewriting**?

9th c.

The Compendious Book on Calculation by Completion and Balancing aka al-Jabr, by Al-Khwarizmi

$$(x + 1) - x$$
$$= (x - x) + 1$$

```
>>> (1e16 - 1e16) + 1
1.0
```

What is floating-point expression **rewriting**?

9th c.

The Compendious Book on Calculation by Completion and Balancing aka al-Jabr, by Al-Khwarizmi

$$(x + 1) - x$$
$$= (x - x) + 1$$

```
>>> (1e16 - 1e16) + 1
1.0
```

✔

$$= 1$$

```
>>> 1
1
```

✔

Real Consequences

Since the invention of computers, real number calculations have produced hidden, unreported errors, sometimes catastrophically.

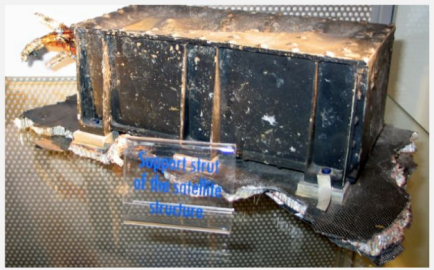
Photo from www.army.mil

PATRIOT MISSILE FAILURE

The most notorious floating point error catastrophe was the Patriot Missile Failure at Dhahran, Saudi Arabia, February 25, 1991, when a Patriot missile failed to destroy a SCUD missile and 128 U.S. military soldiers were killed or wounded as a result. This was the greatest combat loss in an Army unit since Vietnam. The conversion of 100 hours in tenths of a second (3600000) to floating point introduced an undetectable error resulting in the missile guidance software incorrectly locating the SCUD missile.

ARIANE 5 ROCKET, FLIGHT 501

On June 4th, 1996, 40 seconds into flight and at an altitude of 3.7 kilometers, the initial launch of the Ariane 5 rocket, flight 501, ended in RUD (colloquially, Rapid Unplanned Disassembly). Estimates of the loss of the rocket and cargo are as high as $500M. Cause of the failure was an inappropriate floating point conversion. (Photo from Deadpan)



Recovered piece of Ariane 5 after RUD

VANCOUVER STOCK EXCHANGE

In January of 1982 the Vancouver Stock Exchange started a stock index accumulating total stock value for all 1,400 stocks listed on the exchange. but truncating (rounding down) that sum up to 3000 times per day resulting in a loss of index value of about $25 per month for about 23 months indicating an index value of $524.811 when the actual value was $1098.892. (Image by Mafue)

Courtesy of *True North Floating Point*

Floating Point Tools

Herbie

FPTaylor

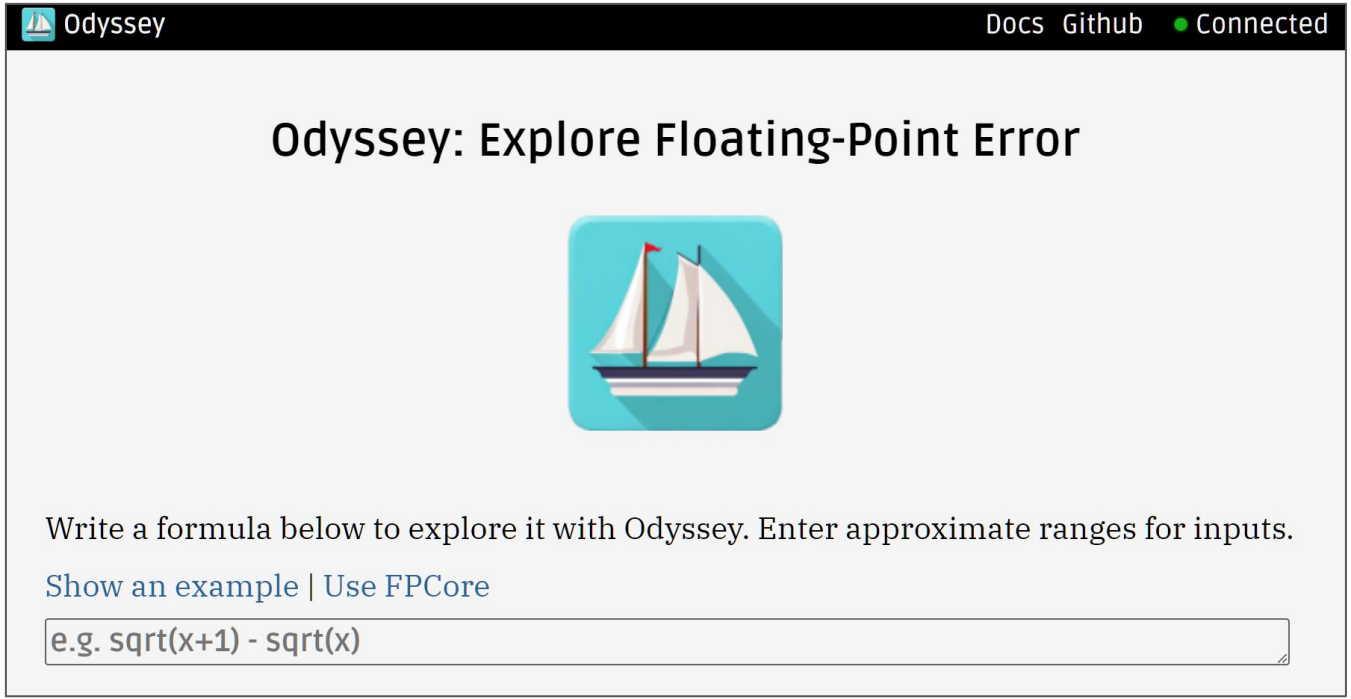
…and many more:
https://fpbench.org/community.html

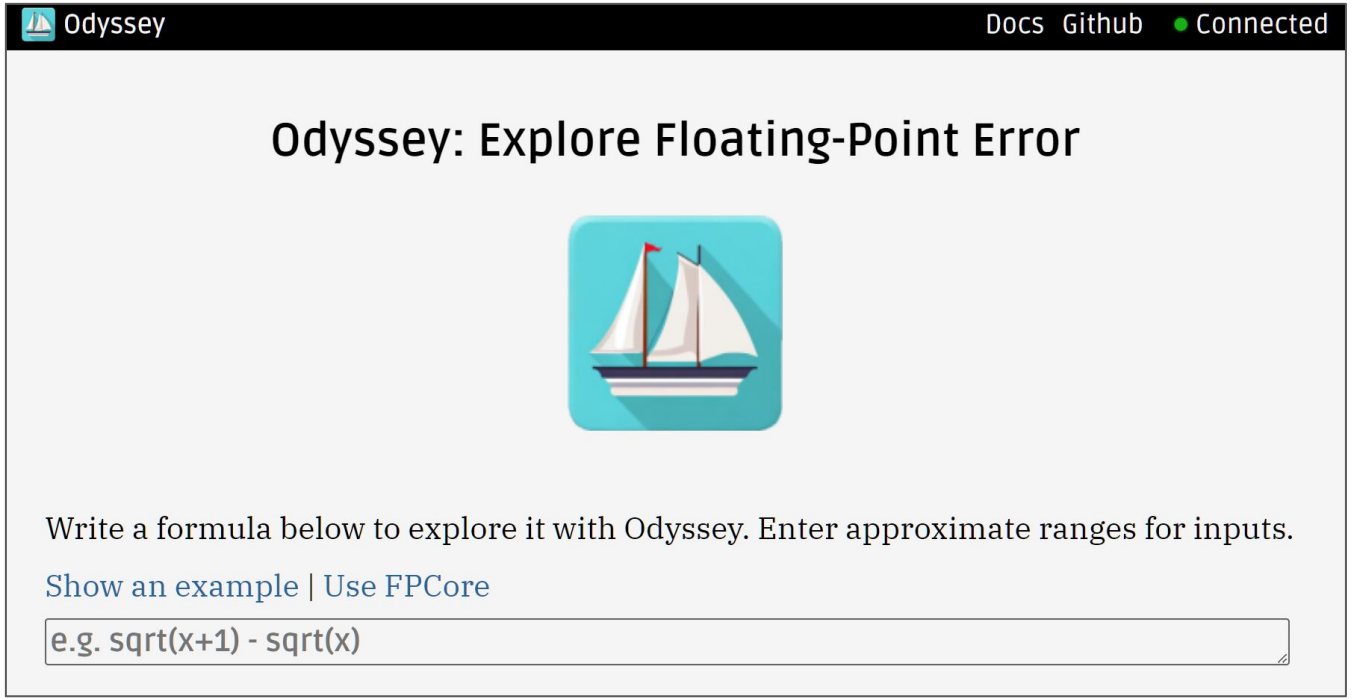
Floating Point Tools

Herbie

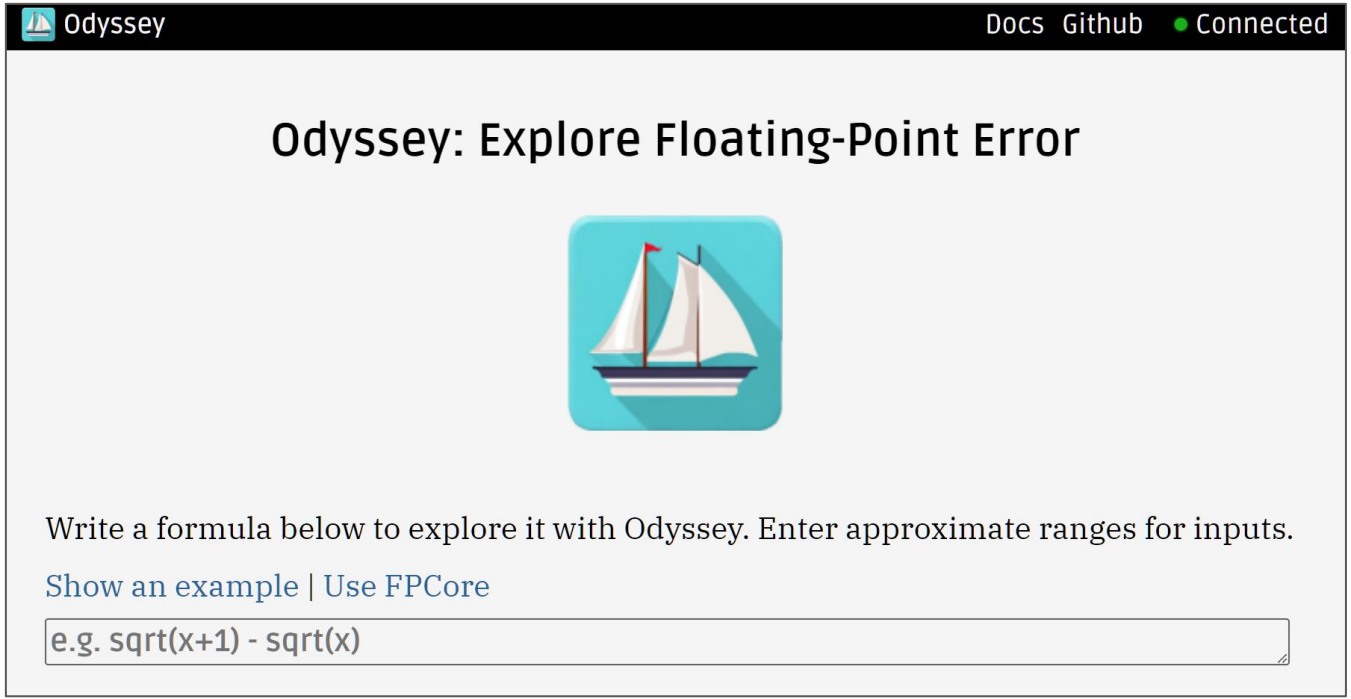
FPTaylor

…and many more:
https://fpbench.org/community.html

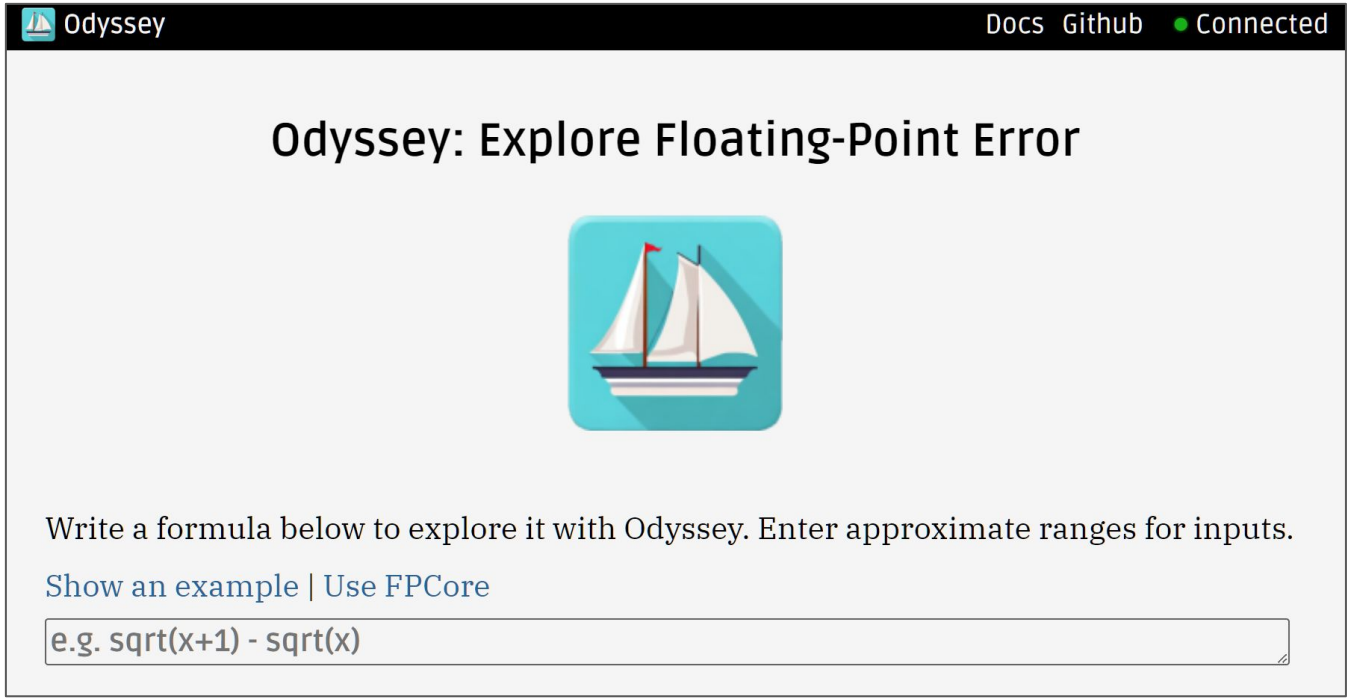

Odyssey
Docs Github Connected
Odyssey: Explore Floating-Point Error
Write a formula below to explore it with Odyssey. Enter approximate ranges for inputs.
Show an example | Use FPCore
e.g. sqrt(x+1) - sqrt(x)



- User-friendly web app



- User-friendly web app
- Tool integration



- User-friendly web app
- Tool integration
- Iterative user experience

DEMO

Compound Interest

Compound Interest

$$P \cdot \frac{\left(1 + \frac{i}{n}\right)^{n} - 1}{\frac{i}{n}}$$

Total value of all payments with interest in a period of time

Payment: $100
Interest rate: 5%
Periods: 12 (Monthly)

$$P \cdot \frac{\left(1 + \frac{i}{n}\right)^{n} - 1}{\frac{i}{n}}$$

Total value of all payments with interest in a period of time

Payment: $100
Interest rate: 5%
Periods: 12 (Monthly)

$$100 \cdot \frac{\left(1 + \frac{0.05}{12}\right)^{12} - 1}{\frac{0.05}{12}}$$

Total value of all payments with interest in a period of time

Payment: $100
Interest rate: 5%
Periods: 12 (Monthly)

$$100 \cdot \frac{\left(1 + \frac{0.05}{12}\right)^{12} - 1}{\frac{0.05}{12}}$$

Total value of all payments with interest in a period of time

$$100 \cdot 12.278855491615914 = \boxed{1227.89}$$

Payment: $100
Interest rate: 5%
Periods: 12 (Monthly)

$$100 \cdot \frac{\left(1 + \frac{0.05}{12}\right)^{12} - 1}{\frac{0.05}{12}}$$

$$100 \cdot 12.278855491615914 = \boxed{1227.89}$$

Total: $1227.89
Interest: $27.89

Areas of Error

$$P \cdot \frac{\left(1 + \frac{i}{n}\right)^n - 1}{\frac{i}{n}}$$

What if small *i*, large *n*?

Areas of Error

$$P \cdot \frac{\left(1 + \frac{i}{n}\right)^n - 1}{\frac{i}{n}}$$

What if small *i*, large *n*?

Areas of Error

$$P \cdot \frac{\left(1 + \frac{i}{n}\right)^n - 1}{\frac{i}{n}}$$

What if small *i*, large *n*?

***i / n* ≈ 0**

Areas of Error

$$P \cdot \frac{(1+0)^n - 1}{0}$$

What if small *i*, large *n*?

***i / n* ≈ 0**

Areas of Error

LARGE + small

$$P \cdot \frac{(1+0)^n - 1}{0}$$

What if small *i*, large *n*?

***i / n* ≈ 0**

Areas of Error

$$P \cdot \frac{(1+0)^n - 1}{0}$$

$$= \boxed{P \cdot \frac{0}{0}}$$

DEMO

Transition to Part-2…

Quadratic

$$\frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

History

- 1960's - Kahan identified cancellation errors in quadratic formula
- 1977 - Franklin: "Fundamental formulas of physics"
- 1986 - Hamming: "Numerical methods for scientists and engineers"

More History

- 2004 - Kahan: “On the cost of floating-point computation without extra-precise arithmetic”
- 2005 - Einarsson: “Accuracy and reliability in scientific computing”
- 2015 - Panchekha et al: “Automatically improving accuracy for floating point expressions”

Areas of Error

$$\frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

Cancellation

$$\sqrt{b^2 - 4ac}$$

Cancellation

$$\sqrt{b^2 - 4ac}$$

$$b^2 \gg 4ac$$

Cancellation

$$\sqrt{b^2 - 4ac}$$

$$\Delta \approx b^2$$

$$b^2 \gg 4ac$$

Cancellation

$$\frac{-b+\sqrt{b^2}}{2a} = \frac{-b+b}{2a}$$

$$b^2 \gg 4ac$$

Cancellation

$$\frac{-b+\sqrt{b^2}}{2a} = \frac{-b+b}{2a}$$

$$b^2 \gg 4ac$$

Cancellation

Similar magnitudes!

$$\frac{-b+\sqrt{b^2}}{2a} = \frac{-b+b}{2a}$$

$$b^2 \gg 4ac$$

Cancellation

$$b^2 - 4ac$$

Cancellation

$$b^2 - 4ac$$

$$b^2 \sim 4ac$$

Cancellation

$$b^2 - 4ac$$

Similar magnitudes!

$$b^2 \sim 4ac$$

Overflow/Underflow

$$\frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

a, b, c extremely small or large

DEMO

Odyssey's Attempt

- ~62% → ~97% for + operation
- ~61% → ~96% for - operation

Odyssey's Attempt

- Uses Taylor Expansion and FMA's

Odyssey's Attempt

- Uses Taylor Expansion and FMA's
- Rewrite is not algebraically equivalent, rather an optimized implementation

Odyssey's Attempt

- Uses Taylor Expansion and FMA's
- Rewrite is not algebraically equivalent, rather an optimized implementation

```
function kahan_discriminant_fma(a,b,c)
    d = b*b - 4*a*c
    if 3*abs(d) >= b*b + 4*a*c # b^2 and 4ac are different enough?
        return d
    end
    p = b*b
    dp = fma(b,b,-p)
    q = 4*a*c
    dq = fma(4*a,c,-q)
    d = (p-q) + (dp-dq)
    return d
end
```

DEMO

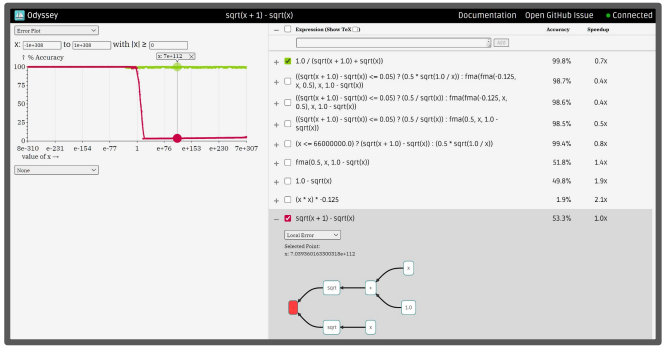
Discussion

- Survey
- Potential use cases
- Features you would like to see

Thank you!

Odyssey: Explore Floating-Point Error

Website:
https://herbie-fp.github.io/odyssey/



Contacts:

Edward Misback
misback@cs.washington.edu

Ben Wang
benwang3@cs.washington.edu

Github:

https://github.com/herbie-fp/odyssey

This work is supported by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research, ComPort: Rigorous Testing Methods to Safeguard Software Porting, under Award Number DE-SC0022081.