Automatically improving floating point code
Scientists Write Code

Every scientist needs to write code

- Analyze data
- Simulate models
- Control experiments
Scientists Write Code

Every scientist needs to write code
  Analyze data
  Simulate models
  Control experiments

They have little computer science training
How scientific code is written

1. Come up with mathematical formula

\[ f(x) = \sqrt{x + 1} - \sqrt{x} \]
How scientific code is written

1. Come up with mathematical formula

\[ f(x) = \sqrt{x + 1} - \sqrt{x} \]

2. Write as floating-point code

\[ f(x) = \text{sqrt}(x + 1) - \text{sqrt}(x) \]
How scientific code is written

\[ f(x) = \sqrt{x + 1} - \sqrt{x} \]

3. Test code

\[ f(1) = 0.41421\ldots \]

\[ f(1) = 0.41421\ldots \]
How scientific code is written

\[ f(x) = \sqrt{x + 1} - \sqrt{x} \]

3. Test code

\[ f(1) = 0.41421... \]

\[ f(1) = 0.41421... \]

4. Profit
How scientific code is written

\[ f(x) = \sqrt{x + 1} - \sqrt{x} \]

3. Test code

\[ f(1) = 0.41421... \]

\[ f(1) = 0.41421... \]

4. Profit Publish
But try a few other values...

\[ f(3141592653589793) = 7.451 \times 10^{-9} \]
\[ f(3141592653589793) \approx 8.921 \cdot 10^{-9} \]

That’s a 16% error!
Outline

Why did this happen?

How can we fix it?

How does Casio help?
Outline

Why did this happen?

How can we fix it?

How does Casio help?
Why did this happen?

“Because floating-point is imprecise!”
Why did this happen?

“Because floating-point is imprecise!”

We want something more constructive.
Why did this happen?

“Because floating-point is imprecise!”

We want something more constructive.

“Rounding or something”
Why did this happen?

“Because floating-point is imprecise!”

We want something more constructive.

“Rounding or something”

Let’s try to be a bit more precise…
Let’s try to be a bit more precise…

177.24559232...
- 177.24277136...

.00282006…
Let’s try to be a bit more precise...

\[
\begin{array}{c}
177.24559 \quad 232 \\
- \quad 177.24277 \quad 136 \\
\hline
.00282 \quad 006
\end{array}
\]

Rounding error
Let’s try to be a bit more precise...

\[
\begin{array}{cc}
177.24559 & 232 \\
- 177.24277 & 136 \\
\hline
0.00282 & 006
\end{array}
\]

\[\rightarrow 0.000001\% \text{ error}\]

\[\rightarrow 0.000001\% \text{ error}\]

\[\rightarrow 0.03\% \text{ error}\]
Let’s try to be a bit more precise...

\[
\begin{array}{c}
177.24559 & 232 \\
- 177.24277 & 136 \\
\hline
.00282 & 006
\end{array}
\rightarrow \text{.000001\% error}

\rightarrow \text{.000001\% error}

\rightarrow \text{.03\% error}

\underline{Rounding error}

Error in \textit{output} related to size of \textit{input}.
For our example

The **output** to the subtraction is **small**
The **input** to the subtraction is **large**

This means the error will be **large**.
For our example

\[ \sqrt{x + 1} - \sqrt{x} \approx \frac{1}{2\sqrt{x}} \]
For our example

\[ \sqrt{x + 1} - \sqrt{x} \approx \frac{1}{2\sqrt{x}} \]

\[ f(x) \approx f(x) + \sqrt{x} \cdot \epsilon \]
For our example

\[
\sqrt{x + 1} - \sqrt{x} \approx \frac{1}{2 \sqrt{x}}
\]

\[
f(x) \approx f(x) + \sqrt{x} \cdot \epsilon
\]

\[
(f(x) - f(x)) \approx \sqrt{x} \cdot \epsilon
\]
For our example

\[ \sqrt{x + 1} - \sqrt{x} \approx \frac{1}{2\sqrt{x}} \]

\[ f(x) \approx f(x) + \sqrt{x} \cdot \epsilon \]

\[ (f(x) - f(x)) \approx \sqrt{x} \cdot \epsilon \approx 2x f'(x) \epsilon \]
For our example

\[ \sqrt{x + 1} - \sqrt{x} \approx \frac{1}{2\sqrt{x}} \]

\[ f(x) \approx f(x) + \sqrt{x} \cdot \epsilon \]

\[ (f(x) - f(x)) \approx \sqrt{x} \cdot \epsilon \approx 2x f(x) \epsilon \]

error \approx 2x \epsilon
So, in summary

Code is imprecise

The subtraction is the culprit

Figuring out why was hard
Outline

Why did this happen?

How can we fix it?

How does Casio help?
How can we fix it?

Run with higher precision?
  Software floating point is slow.

Add correction terms?
  Very hard to do; very error-prone
How can we fix it?

Better idea: rephrase the problem

Compute the same thing in a different way.

Somehow get rid of the subtraction
One way of rephrasing it

$$\sqrt{x + 1} - \sqrt{x}$$
One way of rephrasing it

\[ \sqrt{x + 1} - \sqrt{x} = (\sqrt{x + 1} - \sqrt{x}) \frac{\sqrt{x + 1} + \sqrt{x}}{\sqrt{x + 1} + \sqrt{x}} \]
One way of rephrasing it

\[
\sqrt{x + 1} - \sqrt{x} = (\sqrt{x + 1} - \sqrt{x}) \frac{\sqrt{x + 1} + \sqrt{x}}{\sqrt{x + 1} + \sqrt{x}} = \frac{\sqrt{x + 1}^2 - \sqrt{x}^2}{\sqrt{x + 1} + \sqrt{x}}
\]
One way of rephrasing it

\[ \sqrt{x+1} - \sqrt{x} = (\sqrt{x+1} - \sqrt{x}) \frac{\sqrt{x+1} + \sqrt{x}}{\sqrt{x+1} + \sqrt{x}} \]

\[ = \frac{(\sqrt{x+1})^2 - \sqrt{x}^2}{\sqrt{x+1} + \sqrt{x}} \]

\[ = \frac{x+1 - x}{\sqrt{x+1} + \sqrt{x}} \]

\[ = \frac{1}{\sqrt{x+1} + \sqrt{x}} \]
One way of rephrasing it

\[ \sqrt{x+1} - \sqrt{x} = (\sqrt{x+1} - \sqrt{x}) \frac{\sqrt{x+1} + \sqrt{x}}{\sqrt{x+1} + \sqrt{x}} \]

\[ = \frac{\sqrt{x+1}^2 - \sqrt{x}^2}{\sqrt{x+1} + \sqrt{x}} \]

\[ = \frac{x+1 - x}{\sqrt{x+1} + \sqrt{x}} \]

\[ = \frac{1}{\sqrt{x+1} + \sqrt{x}} \]
One way of rephrasing it

\[
\sqrt{x + 1} - \sqrt{x} = (\sqrt{x + 1} - \sqrt{x}) \frac{\sqrt{x + 1} + \sqrt{x}}{\sqrt{x + 1} + \sqrt{x}} \\
= \frac{\sqrt{x + 1}^2 - \sqrt{x}^2}{\sqrt{x + 1} + \sqrt{x}} \\
= \frac{x + 1 - x}{\sqrt{x + 1} + \sqrt{x}} \\
= \frac{1}{\sqrt{x + 1} + \sqrt{x}}
\]
Implementing this rephrasing

\[ f(x) = \frac{1}{\sqrt{x + 1} + \sqrt{x}} \]

We’ve transformed the minus into a plus.
Implementing this rephrasing

\[ f(x) = \frac{1}{\sqrt{x + 1} + \sqrt{x}} \]

We’ve transformed the minus into a plus. This version has effectively no error.
Implementing this rephrasing

\[ f(x) = \frac{1}{\sqrt{x + 1} + \sqrt{x}} \]

We’ve transformed the minus into a plus. This version has effectively no error.

But it’s a bit confusing:

Is this computing the right function?
So, in summary
Problem solved
Algebra required
Fixing it was hard
Outline

Why did this happen?

How can we fix it?

How does Casio help?
How does Casio help?

Diagnosing and fixing were both hard
How does Casio help?

Diagnosing and fixing were both hard

Casio automatically…
  Computes error
  Finds better code
Automatically computing error

Compute exact answers with arbitrary precision

Use program analysis tools to find the problem
Automatically improving code

Small database of mathematical identities
  e.g. \(a^2 - b^2 = (a + b)(a - b)\)

Apply identities to the problem subexpression

Evaluate resulting code versus exact answer
Current Prototype

Equation

Input Distribution
Current Prototype

Equation → Computed Output
Input Distribution → Exact Output

Arbitrary Precision
Current Prototype

- Equation
- Input Distribution
- Computed Output
- Exact Output

Arbitrary Precision → Numeric Analysis → Bad Subexpressions
Current Prototype

- Equation
- Computed Output
- Exact Output
- Bad Subexpressions
- Rewrite Database
- Candidate Programs

Arbitrary Precision
Numeric Analysis
Rewrite Generation
Current Prototype

Equation → Computed Output → Exact Output → Bad Subexpressions → Rewrite Database → Candidate Programs → Best Candidate

Arbitrary Precision → Numeric Analysis → Rewrite Generation → Ranking

Arbitrary Precision

Numeric Analysis

Rewrite Generation

Ranking

Equation

Input Distribution

Exact Output

Bad Subexpressions

Rewrite Database

Candidate Programs

Best Candidate
> (improve '(λ (x) (- (sqrt (+ x 1)) (sqrt x))) 3)
> (improve '(λ (x) (- (sqrt (+ x 1)) (sqrt x))) 3)
(λ (x) (/ 1 (+ (sqrt x) (sqrt (+ x 1))))))
> (improve '(% (λ (x) (- (sqrt (+ x 1)) (sqrt x))) 3)
(λ (x) (/ 1 (+ (sqrt x) (sqrt (+ x 1))))))
Improvement by an average of 20.3 bits of precision
> (improve '(λ (x) (- (sqrt (+ x 1)) (sqrt x))) 3)
(λ (x) (/ 1 (+ (sqrt x) (sqrt (+ x 1))))))
Improvement by an average of 20.3 bits of precision

> (improve '(λ (x) (- (/ 1 x) (/ 1 (+ x 1)))) 3)
(λ (x) (/ 1 (* x (+ x 1))))
Improvement by an average of 21.2 bits of precision
> (improve '(% (x) (- (sqrt (+ x 1)) (sqrt x))) 3)
(λ (x) (/ 1 (+ (sqrt x) (sqrt (+ x 1))))))
Improvement by an average of 20.3 bits of precision

> (improve '(% (x) (- (/ 1 x) (/ 1 (+ x 1)))) 3)
(λ (x) (/ 1 (* x (+ x 1))))
Improvement by an average of 21.2 bits of precision

> (improve '(% (x) (- x (sqrt (+ (sqr x) 1)))) 3)
(λ (x) (/ -1 (+ x (sqrt (+ (sqr x) 1))))))
Improvement by an average of 17.3 bits of precision
Future work

Subtler precision problems

\[ \frac{e^x - 1}{x} \]
Future work

Subtler precision problems

Extracting floating point computation from code

\[ \frac{e^x - 1}{x} \]
Future work

Subtler precision problems

\[
\frac{e^x - 1}{x}
\]

Extracting floating point computation from code

Provide explanation of what Casio did