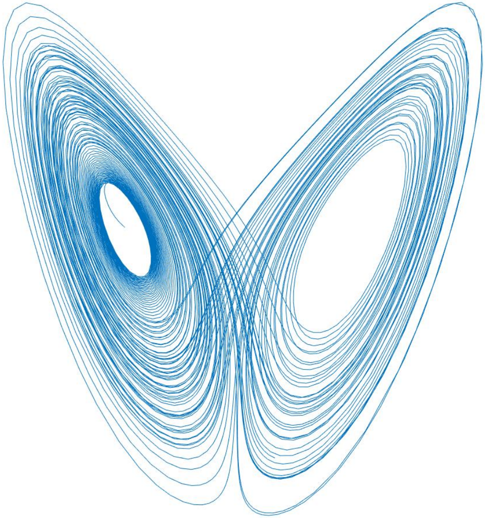

FLOATING-POINT NUMBERS: (some people think)

- Numerical analysis is the study of floating-point arithmetic.
- Floating-point arithmetic is unpredictable and hard to understand.

We intend to convince you that both of these claims are false.

How numbers are represented in a (binary) computer

$$x = \pm(1 + f) \times 2^e$$

$$0 \leq f < 1$$

$$f = \left(\text{integer} < 2^{52}\right) / 2^{52}$$

$$e = \text{integer in} \;\; [-1022, 1023]$$

$$x = \pm(1 + f) \times 2^e$$

$$x = \pm(1 + f) \times 2^e$$

- Finite f implies finite *precision*.

$$x = \pm(1 + f) \times 2^e$$

- Finite f implies finite *precision*.
- Finite e implies finite *range*.

$$x = \pm(1 + f) \times 2^e$$

- Finite f implies finite *precision*.
- Finite e implies finite *range*.
- Floating point numbers have discrete spacing, a maximum and a minimum.

The fractional part f is represented in the computer in binary. So

$$f = \sum_{j=1}^{\infty} \frac{d_j}{2^j}$$

The fractional part f is represented in the computer in binary. So

$$f = \sum_{j=1}^{\infty} \frac{d_j}{2^j}$$

which can also be written as

$$f = .d_1 d_2 d_3 \cdots$$

Example

The number

$$t = \frac{1}{10}$$

(a perfectly fine, rational number) cannot be represented exactly in a floating-point binary computer.

Example

The number

$$t = \frac{1}{10}$$

(a perfectly fine, rational number) cannot be represented exactly in a floating-point binary computer.

$$0.1 = \frac{1}{2^3} + \frac{1}{2^4} + \frac{1}{2^7} + \frac{1}{2^8} + \cdots$$

Example

The number

$$t = \frac{1}{10}$$

(a perfectly fine, rational number) cannot be represented exactly in a floating-point binary computer.

$$0.1 = \frac{1}{2^3} + \frac{1}{2^4} + \frac{1}{2^7} + \frac{1}{2^8} + \cdots$$

Or, in binary

$$\begin{aligned} 0.1 &= .0001100110011\cdots \\ &= 1.100110011\cdots \times 2^{-3} \\ &= 1.1\overline{0011} \times 2^{-3} \end{aligned}$$

Example

The number

$$t = \frac{1}{10}$$

(a perfectly fine, rational number) cannot be represented exactly in a floating-point binary computer.

$$0.1 = \frac{1}{2^3} + \frac{1}{2^4} + \frac{1}{2^7} + \frac{1}{2^8} + \cdots$$

Or, in binary

$$\begin{aligned} 0.1 &= .0001100110011\cdots \\ &= 1.100110011\cdots \times 2^{-3} \\ &= 1.1\overline{0011} \times 2^{-3} \end{aligned}$$

No matter where you chop, t as represented in the computer is never exactly equal to $1/10$!

Example

The number

$$t = \frac{1}{10}$$

(a perfectly fine, rational number) cannot be represented exactly in a floating-point binary computer.

$$0.1 = \frac{1}{2^3} + \frac{1}{2^4} + \frac{1}{2^7} + \frac{1}{2^8} + \cdots$$

Or, in binary

$$\begin{aligned} 0.1 &= .0001100110011\cdots \\ &= 1.100110011\cdots \times 2^{-3} \\ &= 1.1\overline{0011} \times 2^{-3} \end{aligned}$$

No matter where you chop, t as represented in the computer is never exactly equal to $1/10$!

Here $e = -3$ and

$$f = .100110011\cdots$$

eps is the distance from 1 to the next largest floating-point number.

eps is the distance from 1 to the next largest floating-point number.

There are no floating-point numbers between

$$1 \quad \text{and} \quad 1 + \text{eps}$$

eps is the distance from 1 to the next largest floating-point number.

There are no floating-point numbers between

$$1 \quad \text{and} \quad 1 + \text{eps}$$

In ‘double-precision’ (64 total bits – 52 for f, 11 for e and 1 for $\pm$)

$$\text{eps} = 2^{-52}$$

For any real number r (in the range of floats), there is a mantissa and exponent such that

$$(1 + f)2^e \leq r \leq (1 + f + \text{eps})2^e$$

For any real number r (in the range of floats), there is a mantissa and exponent such that

$$(1 + f)2^e \leq r \leq (1 + f + \text{eps})2^e$$

Thus eps is the maximum relative roundoff error when r is rounded to the nearest float.

For any real number r (in the range of floats), there is a mantissa and exponent such that

$$(1 + f)2^e \leq r \leq (1 + f + \text{eps})2^e$$

Thus eps is the maximum relative roundoff error when r is rounded to the nearest float.

eps is not the smallest float – not by a long shot!

For any real number r (in the range of floats), there is a mantissa and exponent such that

$$(1+f)2^e \leq r \leq (1+f+\mathrm{eps})2^e$$

Thus eps is the maximum relative roundoff error when r is rounded to the nearest float.

eps is not the smallest float – not by a long shot!

The smallest float is obtained by taking the smallest f and smallest e. This is called `realmin`.

For any real number r (in the range of floats), there is a mantissa and exponent such that

$$(1+f)2^e \leq r \leq (1+f+\text{eps})2^e$$

Thus eps is the maximum relative roundoff error when r is rounded to the nearest float.

eps is not the smallest float – not by a long shot!

The smallest float is obtained by taking the smallest f and smallest e. This is called `realmin`.

The largest float is obtained by taking the largest f and largest e. This is called `realmax`

	Binary	Decimal
`eps`	2^{-52}	2.2204×10^{-16}
`realmin`	2^{-1022}	2.2251×10^{-308}
`realmax`	$(2 - \texttt{eps})2^{1023}$	1.7977×10^{308}