## Floating Point numbers

Convert $13 = (1101)_2$ into floating point representation.

$1.s_1 s_2 s_3 \cdot 2^{exp}$ $\quad exp \in [-127, 128]$

$1.101 \cdot 2^3$

What pieces do you need to store an FP number?

## FP Addition

$13 + 129$

$1.101 \cdot 2^3 + 1.0000001 \cdot 2^7$

$\rightarrow$ 4 bits

$0.0001101 \cdot 2^7 + 1.0\ldots01 \cdot 2^7$

$= 1.0001110$

$13 \cdot 129 = ?$

$1.101 \cdot 1.0000001 \cdot 2^3 2^7$

$2^{10}$

$A$ $\qquad\qquad$ $B$

$1.2345 \times 10^6 \cdot 7.65432 \cdot 10^2$

$\underbrace{7 \cdot 1.2345}\ \cdot 10^8$

$+ .6 \cdot 1.2345$

$+ .05 \cdot 1.2345$

$\dots$

$C = A \cdot B$

the number of sig. digits
stays the same if we round

$10^{14}$

$A + B = C$

A is $2.34 \cdot 10^{16}$

B is $7.65 \cdot 10^{2}$

machine epsilon $\varepsilon$

$1 + \varepsilon/2 = 1$

$1 + \varepsilon > 1$

$C = 234000 \ldots 0\,\cancel{76}500.$

$A + B = C$

$B \approx -A$

$A = 128.34$

$B = -127.12$

$A = 1.2834 \cdot 10^2$

$B = -1.2712 \cdot 10^2$

$C = .00122$

## Floating Point and Rounding Error

What is the relative error produced by working with floating point numbers?

What is smallest floating point number $> 1$? Assume 4 stored bits in the significand.

$\frac{1 - 1 + \varepsilon}{1} = \frac{\varepsilon}{1} = \varepsilon$ $1.s_1 s_2$ | $1.234 \cdot 10^{-18}$ $1.2\underline{3} \cdot 10^{-18}$

What's the smallest FP number $> 1024$ in that same system?

exponent range $[-127, 128]$

sig. has 4 bits $\varepsilon = 2^{-4}$, $UFL = 2^{-127}$

Can we give that number a name?

abs $= |x - x_0|$ rel $= \frac{|x - x_0|}{x_0}$ $1.11111 \cdot 2^{-127}$ $1.1111 \cdot 2^{-127}$

## Implementing Arithmetic

How is floating point addition implemented?
Consider adding $a = (1.101)_2 \cdot 2^1$ and $b = (1.001)_2 \cdot 2^{-1}$ in a system with three bits in the significand.

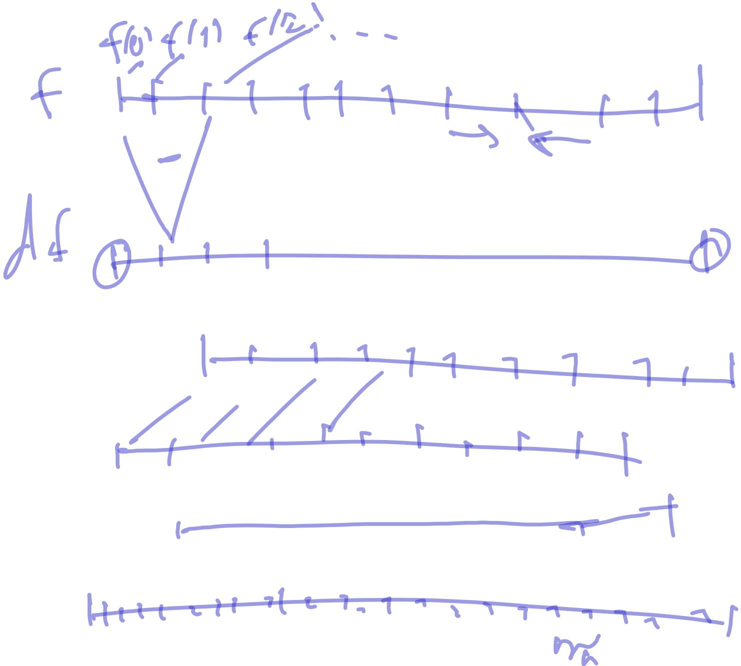
## **Demo:** Floating point and the harmonic series

## Problems with FP Addition

What happens if you subtract two numbers of very similar magnitude?
As an example, consider $a = (1.1011)_2 \cdot 2^0$ and $b = (1.1010)_2 \cdot 2^0$.

## **Demo:** Catastrophic Cancellation

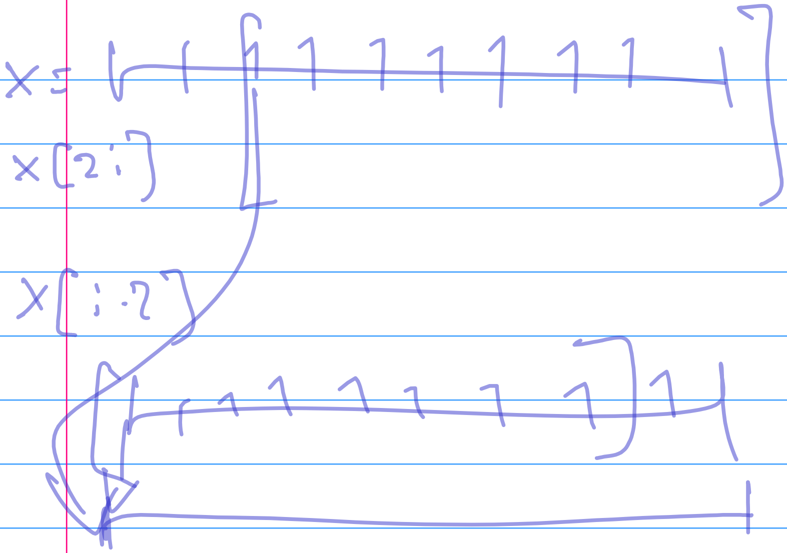
## **In-class activity:** Floating Point

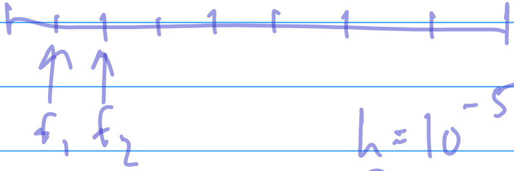


x =

x[2:]

x[:-2]





$h = 10^{-5}$

$f(x) = x \cdot 10^{6}$

$f(x_1) = f_1 = 1.23456\ldots \cdot 10^{6}$

$f(x_1 + h) = f_2 = 1.23457\ldots \cdot 10^{6}$

$f_1 - f_2 = 0.00001$

$(f_1 - f_2)/(2h) = 0.5????$

abs_err$(f(x+h) - f(x-h)) = f(x) \cdot \varepsilon$

abs_err_overall $= \dfrac{f(x) \cdot \varepsilon}{2 \cdot h}$

relative error
with respect to $f'(x) \to \dfrac{|f(x) \cdot \varepsilon / (2h)|}{|f'(x)|}$

$$err = O\left(\frac{1}{h}\right)$$

**Demo:** Picking apart a floating point number

## Unrepresentable numbers?

Can you think of a somewhat central number that we cannot represent as

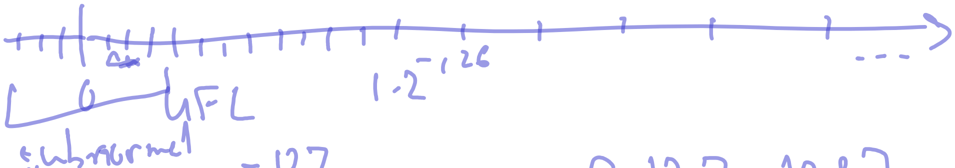
$$x = (1.\_\_\_\_\_\_\_\_\_)_2 \cdot 2^{-p}?$$

$1.s_1 s_2 s_3 \cdot 2^{x} = 0$?

$1.111 \cdot 2^{-127}$

$1 \cdot 2^{-126}$ ~ $.001 \cdot 2^{-126}$

$1.000 \cdot 2^{-126}$



$1 \cdot 2^{-127}$

$\exp = [-127, 128]$

## Subnormal Numbers

What is the smallest representable number in an FP system with 4 stored bits in the significand and an exponent range of $[-7, 7]$?

First attempt:

- Significand as small as possible $\rightarrow$ all zeros after the implicit leading one
- Exponent as small as possible: $-7$

$$(1.0000)_2 \cdot 2^{-7}.$$

Unfortunately: wrong. We can go way smaller by using the special exponent (which turns off the implicit leading one). We'll assume that the special exponent is $-8$. So:

$$(0.0001)_2 \cdot 2^{-8}$$

Numbers with the special epxonent are called subnormal (or denormal) FP numbers. Technically, zero is also a subnormal.
**Note:** It is thus quite natural to 'park' the special exponent at the low end of the exponent range.

## Subnormal Numbers (II)

Why would you want to know about subnormals? Because computing with them is often slow, because it is implemented using 'FP assist', i.e. not in actual hardware. Many C compilers support options to 'flush subnormals to zero'.

- FP systems without subnormals will underflow (return 0) as soon as the exponent range is exhausted.
- This smallest representable *normal* number is called the underflow level, or UFL.
- Beyond the underflow level, subnormals provide for gradual underflow by 'keeping going' as long as there are bits in the significand, but it is important to note that subnormals don't have as many accurate digits as normal numbers.
- Analogously (but much more simply–no 'supernormals'): the overflow level, OFL.