

# Floating Point Precision & Accuracy

Vincent LAFAGE

IJCLab, CNRS/IN2P3 & Université Paris-Saclay, Orsay, France

10 July 2023



# Floating World Computation

Revisiting "What Every Computer Scientist Should Know About Floating-point Arithmetic"



- Numbers: real, decimal, binary, floating point...
- When computations don't turn out as expected...(why, how)
  - ▶ global errors
  - ▶ local errors
  - ▶ composing errors
- Heuristics for accuracy:
  - how a rough estimate can save epsilons
- How to reconcile adimensionalisation and performance
- How to reconcile abstraction and accuracy: functions of a complex variable
- Why are geometrical computations so hard
- The hidden side of functional programming: towards total functions



# Formats

« computing is about representation »

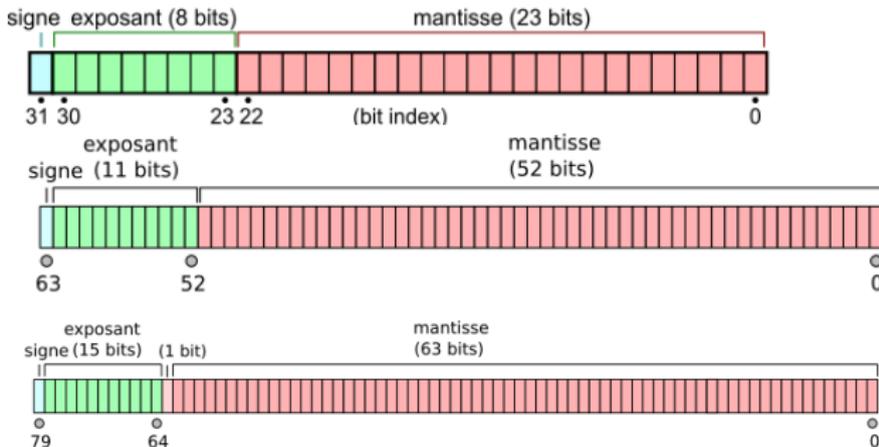
Scientific notation:

significand  $\times$  base<sup>exponent</sup>      significand  $\in \mathbb{Z}$ , exponent  $\in \mathbb{Z}$

Standard form:      mantissa, alias *normalized significand*

mantissa  $\times$  base<sup>exponent</sup>      mantissa  $\in [1; \text{base}[$ , exponent  $\in \mathbb{Z}$

Trick, for base 2: the most significant digit is always 1...



In the registers, we widen mantissa with three bits:

- guard bit
- round bit
- sticky bit

Problems

- apparent: rounding  $\Rightarrow$  catastrophic cancellation
- apparent: conversion. Goes unnoticed or perceived as minor.
- apparent: overflow. Apparent, but not treated.



# Example float32

$$\text{float} = (-1)^S \times 2^{E-127} \times (1 + M), \quad M \in [0, 1[$$

31302928272625242322212019181716151413121110 9 8 7 6 5 4 3 2 1 0

S	E	M
0	0111 1111	000 0000 0000 0000 0000 0000
0	1000 0000	000 0000 0000 0000 0000 0000
0	1000 0000	100 0000 0000 0000 0000 0000
0	1000 0000	110 0000 0000 0000 0000 0000
0	1000 0000	111 0000 0000 0000 0000 0000
0	1000 0000	100 1001 0000 1111 1101 1011
1	0111 1111	000 0000 0000 0000 0000 0000
0	0111 1110	000 0000 0000 0000 0000 0000
0	0111 1100	100 1100 1100 1100 1100 1101
0	0000 0000	000 0000 0000 0000 0000 0000
1	0000 0000	000 0000 0000 0000 0000 0000
0	1111 1111	000 0000 0000 0000 0000 0000
0	1111 1111	1xx xxxx xxxx xxxx xxxx xxxx
0	1111 1111	1xx xxxx xxxx xxxx xxxx xxxx
0	1111 1111	01x xxxx xxxx xxxx xxxx xxxx

} float =  $(-1)^S \times 2^{E-127} \times (1 + M)$   
 }  $1 = 2^0 \times (1 + 0)$   
 }  $2 = 2^1 \times (1 + 0)$   
 }  $3 = 2^1 \times (1 + 1/2)$   
 }  $3.5 = 2^1 \times (1 + 1/2 + 1/4)$   
 }  $3.75 = 2^1 \times (1 + 1/2 + 1/4 + 1/8)$   
 }  $\pi \approx 2^1 \times (1 + 1/2 + 1/16 + 1/128 + \dots)$   
 }  $-1 = -2^0 \times (1 + 0)$   
 }  $1/2 = 2^{-1} \times (1 + 0)$   
 }  $0.2 = 2^{-3} \times (1 + 1/2) \times \sum_n 1/16^n$   
 } 0 special representation  
 } 0\_ special representation  
 }  $+\infty = \text{Inf}$  special representation  
 } NaN special representation  
 } qNaN *quiet* special representation  
 } sNaN *signaling* special representation

⇒ exercise: using the link below, represent number such as  $\sqrt{2}$ ,  $\frac{1+\sqrt{5}}{2}$ ,  $e$ ,  $\ln 2$ ,...

<https://www.h-schmidt.net/FloatConverter/IEEE754.html>



# Get HexaDecimal displayed C

```
#include <stdio.h>
```

```
int main ()  
{  
    float x = 1.0f;  
    printf ("%f_\u00a0%\a\n", x, x);  
    x = 2.0f;  
    printf ("%f_\u00a0%\a\n", x, x);  
    x = 3.0f;  
    printf ("%f_\u00a0%\a\n", x, x);  
    x = 3.141592653589793f;  
    printf ("%f_\u00a0%\a\n", x, x);  
}
```

1.000000 = 0x1p+0

2.000000 = 0x1p+1

3.000000 = 0x1.8p+1

3.141593 = 0x1.921fb6p+1



# Get Hexadecimal displayed C++

```
#include <iostream>
```

```
int main ()
```

```
{
```

```
    float x = 1.0f;
```

```
    std::cout << x << " = " << std::hexfloat << x << std::defaultfloat << '\n';
```

```
    x = 2.0f;
```

```
    std::cout << x << " = " << std::hexfloat << x << std::defaultfloat << '\n';
```

```
    x = 3.0f;
```

```
    std::cout << x << " = " << std::hexfloat << x << std::defaultfloat << '\n';
```

```
    x = 3.141592653589793f;
```

```
    std::cout << x << " = " << std::hexfloat << x << std::defaultfloat << '\n';
```

```
}
```

```
1 = 0x1p+0
```

```
2 = 0x1p+1
```

```
3 = 0x1.8p+1
```

```
3.14159 = 0x1.921fb6p+1
```



# Get Hexadecimal displayed Fortran

```
program hexfloat
  use, intrinsic :: iso_fortran_env, only: real32
  implicit none
  real (real32) :: x

  x = 1
  write (*, '(F10.6,A,Z16)') x, 'uFu', x
  x = 2
  write (*, '(F10.6,A,Z16)') x, 'uFu', x
  x = 3
  write (*, '(F10.6,A,Z16)') x, 'uFu', x
  x = acos (-1.0_real32)
  write (*, '(F10.6,A,Z16)') x, 'uFu', x
end program hexfloat
```

```
1.000000 =      3F800000
2.000000 =      40000000
3.000000 =      40400000
3.141593 =      40490FDB
```



# $0.1 + 0.2 \neq 0.3?$

$$\Sigma = a + b \stackrel{?}{=} c \quad \Delta = a + b - c$$

with

$$a = 0.1 \quad b = 0.2 \quad c = 0.3$$



# 0.1 + 0.2 ≠ 0.3?

$$\Sigma = a + b \stackrel{?}{=} c \quad \Delta = a + b - c$$

with

$$a = 0.1 \quad b = 0.2 \quad c = 0.3$$

$a$	$b$	$c$	$\Sigma$	$\Delta$
0.100000001	0.200000003	0.300000012	0.300000012	0
0.1000000000000000001	0.2000000000000000001	0.2999999999999999999	0.3000000000000000004	5.551... 10 <sup>-17</sup>
0.100000000000000000001	0.200000000000000000003	0.300000000000000000011	0.300000000000000000011	0



# 0.1 + 0.2 ≠ 0.3?

$$\Sigma = a + b \stackrel{?}{=} c \quad \Delta = a + b - c$$

with

$$a = 0.1 \quad b = 0.2 \quad c = 0.3$$

a	b	c	$\Sigma$	$\Delta$
0.100000001	0.200000003	0.300000012	0.300000012	0
0.1000000000000000001	0.2000000000000000001	0.29999999999999999	0.3000000000000000004	5.551... $\cdot 10^{-17}$
0.100000000000000000001	0.200000000000000000003	0.300000000000000000011	0.300000000000000000011	0

$\Rightarrow \mathbb{D} \not\subset \mathbb{B}$ : some decimal are not binary

$\Rightarrow$  binary conversion needs some rounding

$$\frac{1}{5} = 0.2_{10} = 0.00\overline{1100}_2 \dots \ominus 13421773 \times 2^{-26} = 0.2 + 2,98 \times 10^{-9}$$

*God created the integers, all else is the work of man.* L. KRONECKER



# Decimal vs. binary ...and binary vs. floating

$$\mathbb{D} = \left\{ \frac{n}{10^p}, n \in \mathbb{Z}, p \in \mathbb{N} \right\} = \mathbb{Z}[1/10] \text{ (decimal)}$$

$$\mathbb{B} = \left\{ \frac{n}{2^p}, n \in \mathbb{Z}, p \in \mathbb{N} \right\} = \mathbb{Z}[1/2] \text{ (binary)}$$

$\mathbb{B} \subset \mathbb{D}$  mais  $\mathbb{D} \not\subset \mathbb{B}$ :  $\frac{1}{5} \in \mathbb{D}$ ,  $\frac{1}{5} \notin \mathbb{B} \Rightarrow 0.1 + 0.2 \neq 0.3$  ( $\frac{1}{5} = 0.00\overline{1100}_2 \dots$ )  $\Rightarrow$  not good for financial computations...

● closure:

$$\forall (x, y) \in \mathbb{B}^2, \quad x + y \in \mathbb{B},$$

$$\forall (x, y) \in \mathbb{B}^2, \quad x \times y \in \mathbb{B}$$

● commutativity:

$$\forall (x, y) \in \mathbb{B}^2, \quad x + y = y + x,$$

$$\forall (x, y) \in \mathbb{B}^2, \quad x \times y = y \times x$$

● associativity:

$$\forall (x, y, z) \in \mathbb{B}^3, \quad x + (y + z) = (x + y) + z,$$

$$\forall (x, y, z) \in \mathbb{B}^3, \quad x \times (y \times z) = (x \times y) \times z$$

● distributivity:

$$\forall (x, y, z) \in \mathbb{B}^3, \quad x \times (y + z) = x \times y + x \times z$$

● total order:

$$\forall (x, y, z) \in \mathbb{B}^3, \quad x \leq y \text{ and } y \leq z \Rightarrow x \leq z \quad (\text{transitivity});$$

$$\forall (x, y) \in \mathbb{B}^2, \quad x \leq y \text{ and } y \leq x \Rightarrow x = y \quad (\text{antisymmetry});$$

$$\forall x \in \mathbb{B}, \quad x \leq x \quad (\text{reflexivity});$$

$$\forall (x, y) \in \mathbb{B}^2, \quad x \leq y \text{ or } y \leq x \quad (\text{totality}).$$

● topology:

$\mathbb{B} \subset \mathbb{D} \subset \mathbb{Q}$  are dense in  $\mathbb{R} \Rightarrow$  arbitrarily close approximations to the real numbers



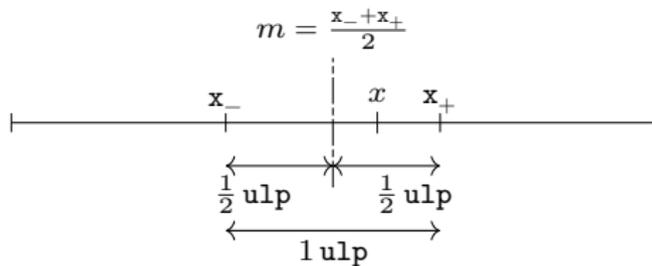
# Decimal vs. binary ...and binary vs. floating

- **closure:**  
 $\exists(x, y) \in \mathbb{F}^2, \quad x + y \notin \mathbb{F},$   
 $\exists(x, y) \in \mathbb{F}^2, \quad x \times y \notin \mathbb{F}$   
 $\Rightarrow$  rounding and extension  $\bar{\mathbb{F}} = \mathbb{F} \cup \{\pm\text{Inf}\} \cup \{\text{NaN}\} \cup \{0_-\}$  overflow, underflow, inexact
- **commutativity:**  
 $\forall(x, y) \in \mathbb{F}^2, \quad x + y = y + x,$   
 $\forall(x, y) \in \mathbb{F}^2, \quad x \times y = y \times x$
- **associativity:**  
 $\exists(x, y, z) \in \mathbb{F}^3, \quad x + (y + z) \neq (x + y) + z,$   
 $\exists(x, y, z) \in \mathbb{F}^3, \quad x \times (y \times z) \neq (x \times y) \times z$
- **distributivity:**  
 $\exists(x, y, z) \in \mathbb{F}^3, \quad x \times (y + z) \neq x \times y + x \times z$
- **total order:**  
 $\forall(x, y, z) \in \mathbb{F}^3, \quad x \leq y \text{ and } y \leq z \Rightarrow x \leq z \quad (\text{transitivity});$   
 $\forall(x, y) \in \mathbb{F}^2, \quad x \leq y \text{ and } y \leq x \Rightarrow x = y \quad (\text{antisymmetry});$   
 $\forall x \in \mathbb{F}, \quad x \leq x \quad (\text{reflexivity});$   
 $\forall(x, y) \in \mathbb{F}^2, \quad x \leq y \text{ or } y \leq x \quad (\text{totality}).$   
 $\exists(x, y) \in \bar{\mathbb{F}}^2, \quad x \leq y \text{ and } y \leq x \quad (\text{NaN}).$
- **topology:**  
 $\mathbb{B} \subset \mathbb{D} \subset \mathbb{Q}$  are dense in  $\mathbb{R} \Rightarrow$  arbitrarily close approximations to the real numbers  
but  
 $\mathbb{F}$ : floating point numbers, finite parts of  $\mathbb{B}$  (or  $\mathbb{D}$ ) are dense nowhere



# Rounding

$\forall x \in \mathbb{R}, \exists (x_-, x_+) \in \mathbb{F}^2 \mid x_- \leq x \leq x_+$  (closest representable neighbours)



$\Rightarrow$  correct rounding requires at least 2 extra bits beyond target accuracy  
or even more (*table maker's dilemma*)

correct rounding, faithful rounding, happy-go-lucky rounding

rounding is non-linear but completely deterministic!



# Conversion

- $\mathbb{D} \not\subset \mathbb{B}$ : every decimal is not a binary

⇒ conversion to binary relies on rounding  $\frac{1}{5} = 0.2_{10} = 0.00\overline{1100}_2 \dots \ominus 13421773 \times 2^{-26} = 0.2 + 2,98 \times 10^{-9}$

4 byte	float	25.4E0 = 25.399999619...
8 byte	double	25.4D0 = 25.39999999999999858...
10 byte	long-double	25.4T0 = 25.3999999999999999653...
16 byte	quadruple	25.4Q0 = 25.3999999999999999999999999999877...

- $\mathbb{B} \subset \mathbb{D}$ : every binary is a decimal

However, converting a binary, usually from a computation, usually for display or storage, is not toward the exactly corresponding decimal: it would require too many meaningless decimal digits.

$$\frac{1}{8} = 0.001_2 = 0.125_{10} \ominus 0.1_{10} \dots$$

⇒ conversion to decimal also relies on rounding

Can division by a constant be replaced by multiplication by this constant reciprocal?

This replacement can induce an extra uncertainty.

**Counter-example:** dividing by 2 (has an exact representation) induces no uncertainty, and the reciprocal of 2 having an exact representation, multiplying by  $\frac{1}{2}$  induces no uncertainty either.

**Example:** dividing by 5 or by 10, or even by 3: one uncertainty coming from division operation, two uncertainties coming from multiplication operation and misrepresentation of operand

**Counter-example:** dividing by  $\pi$ : the inexact representation of  $\pi$  induces one uncertainty, the inexact representation of its reciprocal also induces one uncertainty (almost the same relative uncertainty: 0,37 ulp and 0,43 ulp respectively)



# Addition

$$\sum_{n=1}^N 1/n \sim \ln N + \gamma$$

Table: *Harmonic sum*

fp	N	up sum	down sum	theoretical sum
fp16	250	6.063	6.098	6.098
fp16	500	7.039	6.793	6.793
fp16	1000	7.086	7.477	7.484
fp16	2000	7.086	8.188	8.180
fp16	4000	7.086	8.789	8.875
fp16	8000	7.086	9.797	9.563
fp16	16000	7.086	9.797	10.26
fp16	32000	7.086	9.797	10.95
fp32	32000	10.95073	10.95072	10.95071
fp32	3200000	15.55911	15.55588	15.55588



# Hierarchy of operations

- **arithmetic:**  $+$ ,  $-$ ,  $\times$ ,  $/$ , integer powers
- **algebraic:**  $\sqrt{\quad}$ ,  $\sqrt[n]{\quad}$ , fractional powers and roots of polynomials
- **elementary (transcendental) functions:**  
exp, ln, sin, cos, irrational powers, all circular and hyperbolic trigonometry
- **higher transcendental functions *a.k.a.* special functions:**  
BESSEL, AIRY, Polylogarithm, elliptic integral, EULER  $\Gamma$  function, RIEMANN  $\zeta$  function,...

Correct rounding is guaranteed by the standard for:

- **arithmetic**
- **square roots**



# The Table Maker's Dilemma

## transcendental functions

- ... costly
- ... **correct rounding** is not guaranteed

to get correct rounding with  $n$  digits/bits...<sup>1</sup>

$$\exp(0.5091077534282133) = \underbrace{1.663806007261509}_{16 \text{ digits}} \underbrace{5000000000000000}_{16 \text{ digits}} 49 \dots$$

$$\exp(0.7906867968553504) = \underbrace{2.204910231771509}_{16 \text{ digits}} \underbrace{4999999999999999}_{16 \text{ digits}} 16 \dots$$

**Double rounding** (rounding from high precision to intermediate precision, then to low precision) can also give worse final rounding than expected.



# Catastrophic Cancellation?



By way of exception in base 10 (not in binary)! mantissa: 3 decimal digits

For  $a = 3.34$  and  $b = 3.33$

- $a \ominus b = 0.01 \Rightarrow$  **cancellation** (réduction de la précision relative)  
but a **benign** one (the floating point result is exact:  $a \ominus b = a - b$ )

- $$\begin{cases} a^2 - b^2 & = 0.0667 = 6.67 \times 10^{-2} \\ a \otimes a \ominus b \otimes b & = 0.1 = 1.00 \times 10^{-1} \end{cases}$$

50% of relative error on the result, or 333 ulp, no digit is even correct: **catastrophic cancellation**

- When does this occur?
- How many digits are lost?

Plus, there is an **overflow** risk

$\Rightarrow$  Let's factorize this!

$$(a \oplus b) \otimes (a \ominus b) = 6.67 \otimes 0.01 = 6.67 \times 10^{-2} \quad \text{exact}$$

$\Rightarrow$  The Right Way™



- A difference...
  - ...of squares...
  - ...of sums...
  - ...and sums...
  - ...of squares...
- 
- two passes approach
  - arbitrary data shift towards some expected average value
  - 1-pass online Welford's algorithm (one more division per iteration)



# Typical computation

- dot product
- convolution product (“backwards” dot product)
- Fourier transform
- matrix product is a matrix of dot products

turns out to be a sum of simple products (quadratic in essence).

⇒ we expect to encounter problems similar to difference of squares and variance computation.  
But here we can't use the factorisation trick...

- mixed precision
- `fma` (fused multiply accumulate)
- `fma` used to extract exact product
- combined with Kahan or other compensated sums



# Quadratic

$$\begin{aligned}ax^2 + bx + c &= 0 \quad (a \neq 0) \\ \Delta &= b^2 - 4ac \\ x_{\pm} &= \frac{-b \pm \sqrt{\Delta}}{2a}\end{aligned}$$

2 possible *catastrophic cancelation* (« *compensation calamiteuse* »)

- between  $-b$  and  $\sqrt{\Delta}$

$$\begin{aligned}\Rightarrow q &= -b - \operatorname{sgn}(b)\sqrt{\Delta} = -\operatorname{sgn}(b) (|b| + \sqrt{\Delta}) \\ \begin{cases} x_1 &= \frac{q}{2a} \\ x_2 &= \frac{2c}{q} = \frac{c}{ax_1} \end{cases}\end{aligned}$$

- discriminant  $\Delta = b^2 - 4ac \quad \Rightarrow$  fma

4 possible *overflow*:

- $b^2$  : *spurious overflow* (if  $|b| > 10^{19}$ ,  $\Delta = \text{Inf}$ ,  $|q| = \text{Inf}$  while  $|q| \sim 2 \times 10^{19}$ )
- $ac$
- $b/a$
- $c/b$



# COMPTON Scattering

$$\theta = \arccos \left[ 1 + m_e c^2 \left( \frac{1}{E_1 + E_2} - \frac{1}{E_2} \right) \right]$$

- en fait, 2 soustractions (de positifs)

$$\theta = \arccos \left[ 1 - m_e c^2 \left( \frac{1}{E_2} - \frac{1}{E_1 + E_2} \right) \right]$$

- algèbre de base:

$$\theta = \arccos \left[ 1 - \frac{m_e c^2 E_1}{E_2 (E_1 + E_2)} \right]$$

- ...reste 1 soustraction (de positifs)

- trigonométrie de base:  $\cos 2\alpha = 1 - 2 \sin^2 \alpha \Leftrightarrow \sin^2 \frac{\theta}{2} = \frac{1 - \cos \theta}{2}$

$$\theta = 2 \arcsin \sqrt{\frac{m_e c^2 E_1}{2 E_2 (E_1 + E_2)}}$$

- ...reste 0 soustraction (de positifs)



# COMPTON Scattering

$$\theta = \arccos \left[ 1 + m_e c^2 \left( \frac{1}{E_1 + E_2} - \frac{1}{E_2} \right) \right]$$

- en fait, 2 soustractions (de positifs)

$$\theta = \arccos \left[ 1 - m_e c^2 \left( \frac{1}{E_2} - \frac{1}{E_1 + E_2} \right) \right]$$

- algèbre de base:

$$\theta = \arccos \left[ 1 - \frac{m_e c^2 E_1}{E_2 (E_1 + E_2)} \right]$$

- ...reste 1 soustraction (de positifs)

- trigonométrie de base:  $\cos 2\alpha = 1 - 2 \sin^2 \alpha \Leftrightarrow \sin^2 \frac{\theta}{2} = \frac{1 - \cos \theta}{2}$

$$\theta = 2 \arcsin \sqrt{\frac{m_e c^2 E_1}{2 E_2 (E_1 + E_2)}}$$

- ...reste 0 soustraction (de positifs)



# COMPTON Scattering

$$\theta = \arccos \left[ 1 + m_e c^2 \left( \frac{1}{E_1 + E_2} - \frac{1}{E_2} \right) \right]$$

- en fait, 2 soustractions (de positifs)

$$\theta = \arccos \left[ 1 - m_e c^2 \left( \frac{1}{E_2} - \frac{1}{E_1 + E_2} \right) \right]$$

- algèbre de base:

$$\theta = \arccos \left[ 1 - \frac{m_e c^2 E_1}{E_2 (E_1 + E_2)} \right]$$

- ...reste 1 soustraction (de positifs)

- trigonométrie de base:  $\cos 2\alpha = 1 - 2 \sin^2 \alpha \Leftrightarrow \sin^2 \frac{\theta}{2} = \frac{1 - \cos \theta}{2}$

$$\theta = 2 \arcsin \sqrt{\frac{m_e c^2 E_1}{2 E_2 (E_1 + E_2)}}$$

- ...reste 0 soustraction (de positifs)



# COMPTON Scattering

$$\theta = \arccos \left[ 1 + m_e c^2 \left( \frac{1}{E_1 + E_2} - \frac{1}{E_2} \right) \right]$$

- en fait, 2 soustractions (de positifs)

$$\theta = \arccos \left[ 1 - m_e c^2 \left( \frac{1}{E_2} - \frac{1}{E_1 + E_2} \right) \right]$$

- algèbre de base:

$$\theta = \arccos \left[ 1 - \frac{m_e c^2 E_1}{E_2 (E_1 + E_2)} \right]$$

- ...reste 1 soustraction (de positifs)

- trigonométrie de base:  $\cos 2\alpha = 1 - 2 \sin^2 \alpha \Leftrightarrow \sin^2 \frac{\theta}{2} = \frac{1 - \cos \theta}{2}$

$$\theta = 2 \arcsin \sqrt{\frac{m_e c^2 E_1}{2 E_2 (E_1 + E_2)}}$$

- ...reste 0 soustraction (de positifs)



# COMPTON Scattering

$$\theta = \arccos \left[ 1 + m_e c^2 \left( \frac{1}{E_1 + E_2} - \frac{1}{E_2} \right) \right]$$

- en fait, 2 soustractions (de positifs)

$$\theta = \arccos \left[ 1 - m_e c^2 \left( \frac{1}{E_2} - \frac{1}{E_1 + E_2} \right) \right]$$

- algèbre de base:

$$\theta = \arccos \left[ 1 - \frac{m_e c^2 E_1}{E_2 (E_1 + E_2)} \right]$$

- ...reste 1 soustraction (de positifs)

- trigonométrie de base:  $\cos 2\alpha = 1 - 2 \sin^2 \alpha \Leftrightarrow \sin^2 \frac{\theta}{2} = \frac{1 - \cos \theta}{2}$

$$\theta = 2 \arcsin \sqrt{\frac{m_e c^2 E_1}{2 E_2 (E_1 + E_2)}}$$

- ...reste 0 soustraction (de positifs)



# Compound interest

$$A = P \left(1 + \frac{r}{n}\right)^{nt}$$

$$I = P \left(1 + \frac{r}{n}\right)^{nt} - P$$

$$I = P \left[ \left(1 + \frac{r}{n}\right)^{nt} - 1 \right]$$

$$I = P \left[ \text{pow} \left( \left(1 + \frac{r}{n}\right), nt \right) - 1 \right]$$

$$I = P \left[ \exp \left( nt \ln \left(1 + \frac{r}{n}\right) \right) - 1 \right]$$

$$I = P \left[ \exp \left( nt \log_{1p} \left( \frac{r}{n} \right) \right) - 1 \right]$$

$$I = P \left[ \text{expm1} \left( nt \log_{1p} \left( \frac{r}{n} \right) \right) \right]$$



# Compound interest

$$A = P \left(1 + \frac{r}{n}\right)^{nt}$$

$$I = P \left(1 + \frac{r}{n}\right)^{nt} - P$$

$$I = P \left[ \left(1 + \frac{r}{n}\right)^{nt} - 1 \right]$$

$$I = P \left[ \text{pow} \left( \left(1 + \frac{r}{n}\right), nt \right) - 1 \right]$$

$$I = P \left[ \exp \left( nt \ln \left(1 + \frac{r}{n}\right) \right) - 1 \right]$$

$$I = P \left[ \exp \left( nt \log_{1p} \left( \frac{r}{n} \right) \right) - 1 \right]$$

$$I = P \left[ \text{expm1} \left( nt \log_{1p} \left( \frac{r}{n} \right) \right) \right]$$



# Compound interest

$$A = P \left(1 + \frac{r}{n}\right)^{nt}$$

$$I = P \left(1 + \frac{r}{n}\right)^{nt} - P$$

$$I = P \left[ \left(1 + \frac{r}{n}\right)^{nt} - 1 \right]$$

$$I = P \left[ \text{pow} \left( \left(1 + \frac{r}{n}\right), nt \right) - 1 \right]$$

$$I = P \left[ \text{exp} \left( nt \ln \left(1 + \frac{r}{n}\right) \right) - 1 \right]$$

$$I = P \left[ \text{exp} \left( nt \log_{1p} \left( \frac{r}{n} \right) \right) - 1 \right]$$

$$I = P \left[ \text{expm1} \left( nt \log_{1p} \left( \frac{r}{n} \right) \right) \right]$$



# Compound interest

$$A = P \left(1 + \frac{r}{n}\right)^{nt}$$

$$I = P \left(1 + \frac{r}{n}\right)^{nt} - P$$

$$I = P \left[ \left(1 + \frac{r}{n}\right)^{nt} - 1 \right]$$

$$I = P \left[ \text{pow} \left( \left(1 + \frac{r}{n}\right), nt \right) - 1 \right]$$

$$I = P \left[ \text{exp} \left( nt \ln \left(1 + \frac{r}{n}\right) \right) - 1 \right]$$

$$I = P \left[ \text{exp} \left( nt \log_{1p} \left(\frac{r}{n}\right) \right) - 1 \right]$$

$$I = P \left[ \text{expm1} \left( nt \log_{1p} \left(\frac{r}{n}\right) \right) \right]$$



# Compound interest

$$A = P \left(1 + \frac{r}{n}\right)^{nt}$$

$$I = P \left(1 + \frac{r}{n}\right)^{nt} - P$$

$$I = P \left[ \left(1 + \frac{r}{n}\right)^{nt} - 1 \right]$$

$$I = P \left[ \text{pow} \left( \left(1 + \frac{r}{n}\right), nt \right) - 1 \right]$$

$$I = P \left[ \text{exp} \left( nt \ln \left(1 + \frac{r}{n}\right) \right) - 1 \right]$$

$$I = P \left[ \text{exp} \left( nt \log_{1p} \left(\frac{r}{n}\right) \right) - 1 \right]$$

$$I = P \left[ \text{expm1} \left( nt \log_{1p} \left(\frac{r}{n}\right) \right) \right]$$



# Area of triangle

area  $S$  as a function of lengths  $a$ ,  $b$  and  $c$  of edges

$$S = \sqrt{p(p-a)(p-b)(p-c)} \quad (\text{HERON of ALEXANDRIA})$$

$$p = \frac{a+b+c}{2} \quad \text{half-perimeter}$$

Symmetric, but numerically unstable, for needle-like triangles (when large and small values meet in the same formula)

KAHAN Re-labelling:  $a > b > c$

$$\frac{1}{4} \sqrt{[a + (b + c)] [c - (a - b)] [c + (a - b)] [a + (b - c)]}$$

Apparent Symmetry is lost, but the formula is way more robust

Originating from a determinantal expression

$$S = \frac{1}{4} \sqrt{\begin{vmatrix} 0 & a^2 & b^2 & 1 \\ a^2 & 0 & c^2 & 1 \\ b^2 & c^2 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{vmatrix}}$$

⇒ exercise: code and test data from

<https://people.eecs.berkeley.edu/~wkahan/Triangle.pdf>

could also apply to cyclic quadrilateral area  $S = \sqrt{(p-a)(p-b)(p-c)(p-d)}$  (BRAHMAGUPTA's formula) or more likely to BRETSCHNEIDER's formula for general non crossed quadrilateral



# Volume of the tetrahedron

$$V = \sqrt{\frac{1}{288} \begin{vmatrix} 0 & a^2 & b^2 & c^2 & 1 \\ a^2 & 0 & C^2 & B^2 & 1 \\ b^2 & C^2 & 0 & A^2 & 1 \\ c^2 & B^2 & A^2 & 0 & 1 \\ 1 & 1 & 1 & 1 & 0 \end{vmatrix}}$$

$$X = (c - A + b)(A + b + c)$$

$$x = (A - b + c)(b - c + A)$$

$$Y = (a - B + c)(B + c + a)$$

$$y = (B - c + a)(c - a + B)$$

$$Z = (b - C + a)(C + a + b)$$

$$z = (C - a + b)(a - b + C)$$

$$\xi = \sqrt{xYZ} \quad \eta = \sqrt{yZX} \quad \zeta = \sqrt{zXY} \quad \lambda = \sqrt{xyz}$$

$$V = \frac{1}{192abc} \sqrt{(\xi + \eta + \zeta - \lambda)(\lambda + \xi + \eta - \zeta)(\eta + \zeta + \lambda - \xi)(\zeta + \lambda + \xi - \eta)}$$



## Testing precision with BASIC'85

```
for (unsigned nbTot = NBITERMIN; nbTot < NBITERMAX; nbTot++) {  
    float x = X0;  
    for (unsigned nbIter = 0; nbIter < nbTot; nbIter++) x = sqrt (x);  
    float bottomRadix = x;  
    for (unsigned nbIter = 0; nbIter < nbTot; nbIter++) x = x * x;  
    printf ("%d□%f□%f□(%+e)□%f□(%+e)\n", nbTot, X0, x, x-X0, bottomRadix, bottomRadix-1.0);  
}
```

iter	X0	x	x - X0	btmRdx	btmRdx - 1
10	2.000000	1.999958	(-4.184246e-05)	1.000677	(+6.771088e-04)
11	2.000000	2.000196	(+1.962185e-04)	1.000339	(+3.385544e-04)
12	2.000000	2.000196	(+1.962185e-04)	1.000169	(+1.692772e-04)
13	2.000000	2.000196	(+1.962185e-04)	1.000085	(+8.463860e-05)
14	2.000000	2.000196	(+1.962185e-04)	1.000042	(+4.231930e-05)
15	2.000000	1.996286	(-3.713965e-03)	1.000021	(+2.110004e-05)
16	2.000000	1.988545	(-1.145530e-02)	1.000010	(+1.049042e-05)
17	2.000000	1.988545	(-1.145530e-02)	1.000005	(+5.245209e-06)
18	2.000000	1.988545	(-1.145530e-02)	1.000003	(+2.622604e-06)
19	2.000000	1.988545	(-1.145530e-02)	1.000001	(+1.311302e-06)
20	2.000000	1.868132	(-1.318680e-01)	1.000001	(+5.960464e-07)
21	2.000000	1.648514	(-3.514862e-01)	1.000000	(+2.384186e-07)
22	2.000000	1.648514	(-3.514862e-01)	1.000000	(+1.192093e-07)
23	2.000000	1.000000	(-1.000000e+00)	1.000000	(+0.000000e+00)



What is the relative sensitivity of a function with respect to input argument fluctuation?  
⇒ *condition number* or absolute value of *elasticity*

$$\kappa(x) = \frac{\left| \frac{f(x_a) - f(x)}{f(x)} \right|}{\left| \frac{x_a - x}{x} \right|} = \frac{\left| \frac{f(x_a) - f(x)}{(x_a - x)} \right|}{\left| \frac{f(x)}{x} \right|} \sim \left| \frac{x f'(x)}{f(x)} \right| = \left| \frac{d(\ln |f(x)|)}{d \ln |x|} \right| \quad (1)$$

$\kappa$  is dimensionless, a pure number (*doubly logarithmic derivative*)

Power law  $x \rightarrow C \times x^n$  (with  $C$  and  $n$  real constants) are the functions with uniform condition number:  $\forall x, \kappa(x) = n$ .

$\log_2 \kappa$ : number of accuracy bits lost *in the best case, with correct rounding*

$f: x \rightarrow x^2 \Rightarrow \kappa = \frac{2x \cdot x}{x^2} = 2$ : no singularity, relative error doubles on each iteration

$f: x \rightarrow \sqrt{x} \Rightarrow \kappa = \frac{1}{2}$ : no singularity, relative error is halved on each iteration (but can't really get below  $\frac{1}{2}$  ulp)

Very few uncertainty caused by iterations of  $\sqrt{\cdot}$ , still the last half ulp is responsible for losing 100% of accuracy

then iterations of  $x \rightarrow x^2$  amplify this generally negligible error to a macroscopic one.



$$\kappa_{f \circ g} = \kappa_f \times \kappa_g$$

$$\kappa_{f \times g} = \kappa_f + \kappa_g$$

$$\kappa_{f^n} = n\kappa_f$$

- $f : x \rightarrow x - c \Rightarrow \kappa = \frac{x}{x-c}$ : singularity  $x = c$  (*catastrophic cancellation*)
- $f : x \rightarrow \ln x \Rightarrow \kappa(x) = \frac{1}{\ln x}$ : singularity  $x = 1$ ,  $f(x = 1 + h) = \ln(1 + h)$

$$\kappa(h) = \frac{h}{(1+h)\ln(1+h)} \underset{h \rightarrow 0}{\sim} \frac{1}{(1+h)} \quad \text{hence the importance of log1p}$$

- $f : x \rightarrow \exp x - 1 \Rightarrow \kappa(x) = \frac{x \exp x}{\exp x - 1}$ : indeterminate form  $x = 0$ ,  $\kappa(h) \underset{h \rightarrow 0}{\sim} 1$   
hence the importance of expm1

- $f : x \rightarrow \cos x - 1 \Rightarrow \kappa(x) = \frac{-x \sin x}{\cos x - 1}$ : indeterminate form  $x = 0$ ,  $\kappa(h) = \frac{h \cos \frac{h}{2}}{\sin \frac{h}{2}} \underset{h \rightarrow 0}{\sim} 2$   
hence the importance of trigonometry

To bypass cleanly this «tower of roots» problem (even in single precision), one needs to change the naive approach and use log1p and expm1  $\Rightarrow$  *exercise: do it!*



⇒ higher degree polynomials can degrade high resolutions

$$P = 333.75y^6 + x^2(11x^2y^2 - y^6 - 121y^4 - 2) + 5.5y^8 + x/(2y)$$

with  $x = 77617$  and  $y = 33096$  (coprime integers)

[S.M. RUMP, 1983, *"How reliable are results of computers"*

<https://www.tuhh.de/ti3/paper/rump/Ru83b.pdf>]



⇒ higher degree polynomials can degrade high resolutions

$$P = 333.75y^6 + x^2(11x^2y^2 - y^6 - 121y^4 - 2) + 5.5y^8 + x/(2y)$$

with  $x = 77617$  and  $y = 33096$  (coprime integers)

[S.M. RUMP, 1983, *"How reliable are results of computers"*

<https://www.tuhh.de/ti3/paper/rump/Ru83b.pdf>]

float:             $P = -6.33825300e + 29$



⇒ higher degree polynomials can degrade high resolutions

$$P = 333.75y^6 + x^2(11x^2y^2 - y^6 - 121y^4 - 2) + 5.5y^8 + x/(2y)$$

with  $x = 77617$  and  $y = 33096$  (coprime integers)

[S.M. RUMP, 1983, *"How reliable are results of computers"*

<https://www.tuhh.de/ti3/paper/rump/Ru83b.pdf>]

float:  $P = -6.33825300e + 29$

double:  $P = -1.1805916207174113e + 021$



⇒ higher degree polynomials can degrade high resolutions

$$P = 333.75y^6 + x^2(11x^2y^2 - y^6 - 121y^4 - 2) + 5.5y^8 + x/(2y)$$

with  $x = 77617$  and  $y = 33096$  (coprime integers)

[S.M. RUMP, 1983, *"How reliable are results of computers"*

<https://www.tuhh.de/ti3/paper/rump/Ru83b.pdf>]

float:  $P = -6.33825300e + 29$

double:  $P = -1.1805916207174113e + 021$

long double:  $P = +5.76460752303423489188e + 17$



⇒ higher degree polynomials can degrade high resolutions

$$P = 333.75y^6 + x^2(11x^2y^2 - y^6 - 121y^4 - 2) + 5.5y^8 + x/(2y)$$

with  $x = 77617$  and  $y = 33096$  (coprime integers)

[S.M. RUMP, 1983, *"How reliable are results of computers"*

<https://www.tuhh.de/ti3/paper/rump/Ru83b.pdf>]

float:	$P = -6.33825300e + 29$
double:	$P = -1.1805916207174113e + 021$
long double:	$P = +5.76460752303423489188e + 17$
quad:	$P = +1.17260394005317863185883490452018380$



⇒ higher degree polynomials can degrade high resolutions

$$P = 333.75y^6 + x^2(11x^2y^2 - y^6 - 121y^4 - 2) + 5.5y^8 + x/(2y)$$

with  $x = 77617$  and  $y = 33096$  (coprime integers)

[S.M. RUMP, 1983, *"How reliable are results of computers"*

<https://www.tuhh.de/ti3/paper/rump/Ru83b.pdf>]

float:	$P = -6.33825300e + 29$
double:	$P = -1.1805916207174113e + 021$
long double:	$P = +5.76460752303423489188e + 17$
quad:	$P = +1.17260394005317863185883490452018380$
exact:	$P \approx -0.827396059946821368141165095479816292$
	$P = -\frac{54767}{66192}$

How to control rounding errors?



# The art of nondimensionalization

nondimensionalizing a set of quantities from **dimensional analysis equation**, one can

- link together **characteristic scales** of the different quantities
- to make appear a set of **dimensionless parameters** (pure numbers) of the problem (“shape parameters” as opposed to “scale parameters”)
- for a set of quantities, there are generally *many ways* to scale it
- we need to add other criteria to select one way:
  - ▶ symmetries of the problem,
  - ▶ limit the complexity of the calculation,
  - ▶ limit the exceptions of the calculation.
- this scaling cannot solve all the problems  
if the solution is not representable, the intermediate calculations are allowed not to be representable
- the exceptions on  $\beta'$  do not necessarily propagate correctly to the roots  
the roots thus found are not necessarily relevant ;  $\Rightarrow$  asymptotic development (*cf* condition)
- anticipate the overflows
- qualify the asymptotic regimes of the function ( $x$  large... but in front of what?)
- evaluate the proximity to 0 ( $y$  small... but in front of what?)

A bit expensive...

- 2 more roots and 2 more divisions
- 3 more branches (⚠ **branchless** paradigm: specific to floating point) (catch exception?)
- naive double precision evaluation would solve it

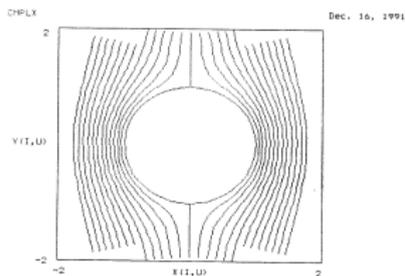


Figure 1 : Eluding Flow Past the Unit Disk

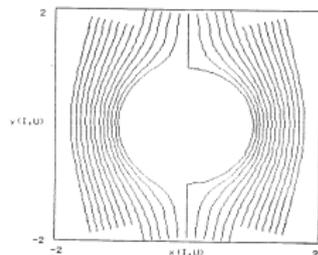


Figure 2 : Eluding Flow Past the Unit Disk, Almost

7

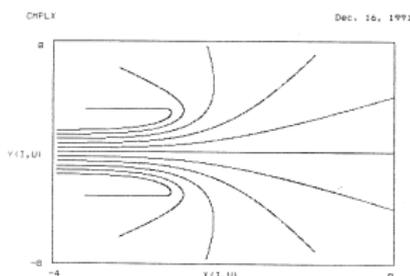


Figure 3 : Borda's Mouthpiece

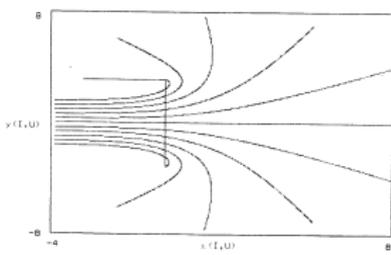


Figure 4 : Borda's Mouthpiece, Almost

8

**Eluding Flow past a Disk:**  $f: Z \mapsto \frac{Z-1/Z}{2}$  and  $g: W \mapsto W - i\sqrt{iW-1}\sqrt{iW+1}$

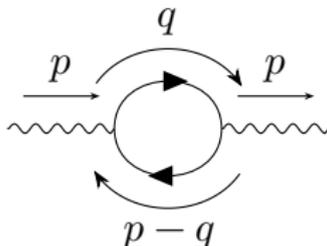
Do not "simplify"  $g(W)$  to  $W - i\sqrt{-W^2-1}$  nor to  $W - \sqrt{W^2+1}$  since they behave differently. Though  $\forall W, f(g(W)) = W, \forall |Z| > 1, g(f(Z)) = Z$  only, and some  $|Z| = 1$ ; otherwise  $g(f(Z)) = -1/Z$ . Deducing where these identities hold is tricky.

**Borda's Mouthpiece:**  $W \mapsto 1 + W^2 + W\sqrt{W^2+1} + \ln(W^2 + W\sqrt{W^2+1})$

as  $W$  runs on radial straight lines through 0 in the right half-plane, including the imaginary axis.



## Function of a complex variable



$$\begin{aligned} B_0(p, m_1, m_2) &= 16\pi^2 Q^{4-n} \int \frac{d^n q}{i(2\pi)^n} \frac{1}{[q^2 - m_1^2 + i\varepsilon][ (q-p)^2 - m_2^2 + i\varepsilon ]} \\ &= \frac{1}{\varepsilon} - \int_0^1 dx \ln \frac{(1-x) m_1^2 + x m_2^2 - x(1-x) p^2 - i\varepsilon}{Q^2} \\ &= \frac{1}{\varepsilon} - \ln\left(\frac{p^2}{Q^2}\right) - f_B(x_+) - f_B(x_-) \end{aligned}$$

$$s = p^2 - m_2^2 + m_1^2, x_{\pm} = \frac{s \pm \sqrt{s^2 - 4p^2(m_1^2 - i\varepsilon)}}{2p^2}, f_B(x) = \ln(1-x) - x \ln(1-x^{-1}) - 1$$

⇒ the (microscopic) difference of  $\varepsilon$  induces a (macroscopic) difference of  $2\pi$  on the imaginary part

⇒ the analytic functions<sup>2</sup> of complex analysis are sharply discontinuous at the crossing of their *branch cut*

# Discrete Stochastic Arithmetic (DSA) [Vignes'04]

Classic arithmetic

$$A \oplus B \rightarrow R$$

$R = 3.14237654356891$

DSA

Random  
rounding

$$A_1 \oplus B_1 \rightarrow R_1$$

$$A_2 \oplus B_2 \rightarrow R_2$$

$$A_3 \oplus B_3 \rightarrow R_3$$

$R_1 = \mathbf{3.141354786390989}$

$R_2 = \mathbf{3.143689456834534}$

$R_3 = \mathbf{3.142579087356598}$

- each operation executed 3 times with a random rounding mode
- number of correct digits in the results estimated using Student's test with the confidence level 95%
- operations executed synchronously
  - ⇒ detection of numerical instabilities  
Ex: `if (A>B)` with A-B numerical noise
  - ⇒ optimization of stopping criteria



- implements stochastic arithmetic for C/C++ or Fortran codes
- few code rewriting
- all operators and mathematical functions overloaded
- support for MPI, OpenMP, GPU, vectorised codes
- supports emulated ou native half precision
- in one CADNA execution: accuracy of any result, complete list of numerical instabilities

## CADNA cost

- memory: 4
- run time  $\approx 10$



## Executing CADNA

Before modifying the precisions used, we want to explore the current accuracy.



## Executing CADNA

Before modifying the precisions used, we want to explore the current accuracy.

To execute CADNA, we essentially **change the types**.



Before modifying the precisions used, we want to explore the current accuracy.

To execute CADNA, we essentially change the types.

This execution exposed multiple numerical instabilities that hide potential massive loss of accuracy.

```
-----  
CADNA_C 3.1.11 software
```

```
CRITICAL WARNING: the self-validation detects major problem(s).  
The results are NOT guaranteed.
```

```
There are 538393974 numerical instabilities
```

```
10409 UNSTABLE DIVISION(S)
```

```
40122229 UNSTABLE MULTIPLICATION(S)
```

```
267297 UNSTABLE BRANCHING(S)
```

```
448561143 UNSTABLE INTRINSIC FUNCTION(S)
```

```
266 UNSTABLE MATHEMATICAL FUNCTION(S)
```

```
49432630 LOSS(ES) OF ACCURACY DUE TO CANCELLATION(S)  
-----
```

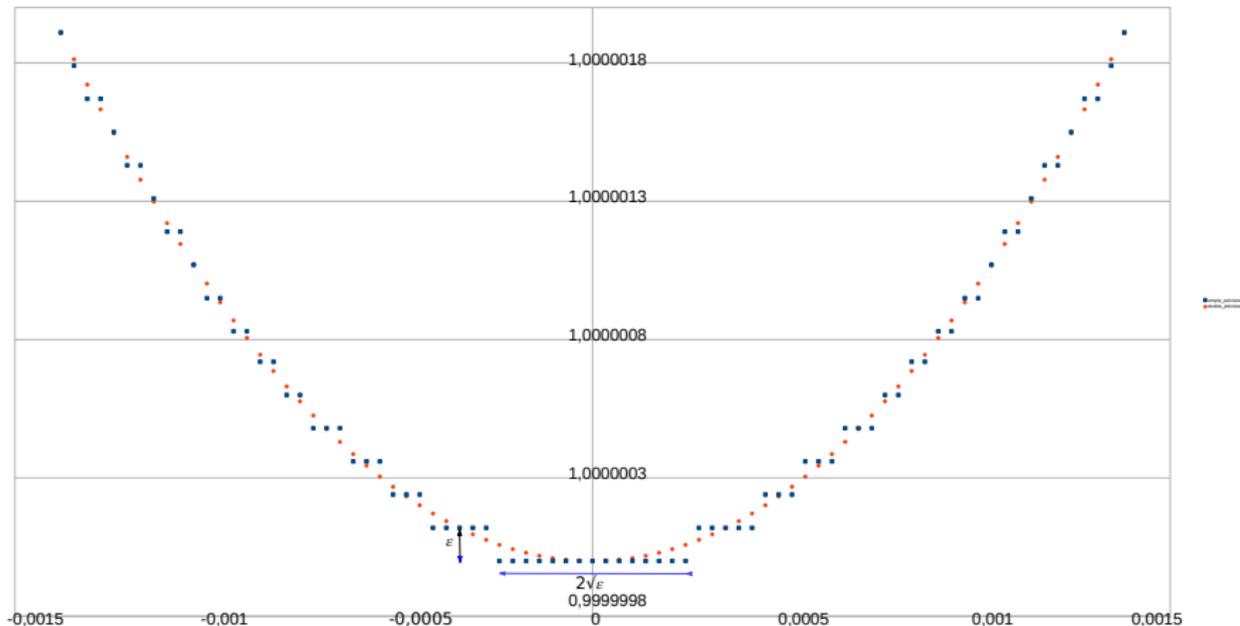


# Minimisation

Numerical evaluation of derivatives / gradients / Jacobian / Hessian

$$x \mapsto 1 + (x - 1)^2 \Rightarrow f(x = x_0 + h) = f(x_0) + \underbrace{h \cdot \frac{\partial f}{\partial x}}_{=0 \text{ at extremum}} + {}^t h \cdot \frac{\partial^2 f}{\partial x^2} \cdot h + o(h^2) \dots \text{TAYLOR}$$

Forme Quadratique





# Neural Network

## Exploration of Machine learning for Polynomial Root Finding

Vitaliy Gyrya, Mikhail Shashkov, Aleval Skurikhin  
(T-5) Applied Mathematics & Plasma Physics, (XCP-4) Methods & Algorithms, (ISR-3) Space Data Science & Systems



Machine Learning for Computational Fluid and Solid Dynamics  
February 19-21, 2019



### Motivation

We are interested in application of Machine Learning (ML) for improving numerical methods for solving partial differential equations (PDEs). One example of such an improvement is the optimization of the parameters of artificial viscosity for Lagrangian and arbitrary-Lagrangian-Eulerian methods. Another example is solving the Riemann problem, which is at the core of many numerical methods for computational gas and solid dynamics. To build confidence in ML methods and understand their strengths and weaknesses we decided to start by applying ML to solve simple quadratic equations of one variable.

### Problem

Consider a quadratic equation,  $ax^2 + bx + c = 0$ , whose roots are  $r_L$  and  $r_R$ . We would like to learn the function

$$(a, b, c) \rightarrow (r_L, r_R)$$

without relying on our knowledge of the underlying processes. Instead we will consider a number of observations observations (training set)

$$(\tilde{a}^i, \tilde{b}^i, \tilde{c}^i) \rightarrow (r_L^i, r_R^i), \quad i = 1, \dots, N.$$

From which we will try to predict

$$(\hat{a}^j, \hat{b}^j, \hat{c}^j) \rightarrow (\hat{r}_L^j, \hat{r}_R^j), \quad j = N + 1, \dots, N + K.$$

The goal is to minimize

$$\text{COST} = \sum_i (r_L^i - \hat{r}_L^i)^2 + \sum_j (r_R^j - \hat{r}_R^j)^2.$$

### Challenges

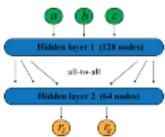
The quadratic equation was selected as a proxy for the following reasons that are relevant to many complex practical problems:

- There are several branches in the solution: if  $a = 0$ , the quadratic equation becomes a linear equation, which has one root – this is a qualitative change from one regime to a different one; depending on the discriminant the number of roots as well as the nature of the roots changes (real vs. complex).
- Finding solution involves different arithmetic operations some of which can be difficult to model by machine learning techniques. For example, division and square root are a challenge for neural networks to represent as activation functions.
- Probably, the most significant challenge is that for a small range of input parameters for which output values are increasingly large.

### Feed-forward Neural Network

#### NN Architecture:

Input Layer: 3 nodes  
 Hidden Layer 1: 128 ReLU  
 Hidden Layer 2: 64 ReLU  
 Output Layer: 2 Linear  
 Connectivity: full.



#### NN Training:

Batch size: 200  
 Training epochs: under 500  
 Optimizer: Adam (<https://arxiv.org/abs/1412.6980v8>)

### Gauss Process Regression (GPR)

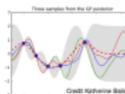
- Probabilistic Bayesian generalization of linear regression approach.

- Built in model of uncertainty estimator.

- Need to specify a covariance kernel.

Our choice of kernel:

ConstantKernel() +  
 Matern(length\_scale = 2, nu = 3/2) +  
 WhiteKernel(noise\_level = 1)



### Test & training sets

We considered a number of distributions for the coefficients  $(a, b, c)$ . In all these cases we assumed that

$$a \in [r, 1], \quad b \in [-1, 1], \quad c \in [-1, 1], \quad \epsilon = 1/20$$

and the roots  $(r_L, r_R)$  are real, i.e.  $D = b^2 - 4ac \geq 0$ .

We considered the following distributions for  $(a, b, c)$

- Uniform random distribution.
  - Regular distribution for  $(a, b, c)$ , i.e. distribution on a grid.
  - Regular distribution for  $(1/a, b, c)$ , i.e. distribution on a grid.
- The sizes of the training and test sets were approximately equal and were on the order of 40K to 50K data points.

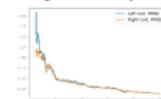
### GPR for large datasets

- GPR performance degrades quickly (scaling  $\sim N^3$ ).
- Depending on the machine the threshold of tractable training sets was between 5K and 50K sample points.
- More advanced techniques are needed for larger data sets.
- Ensembles of smaller GPR could be used.

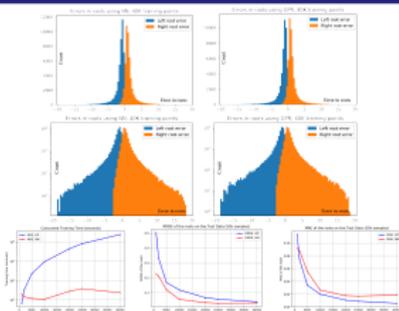
### Adaptive sampling with GPR

#### Adaptation procedure:

- Consider the pool of uniformly distributed parameters  $(\tilde{a}^i, \tilde{b}^i, \tilde{c}^i)$ .
- Select an initial training set of points (50) at random. Generated GPR based on these points.
- For the given GPR consider the "uncertainty"  $\sigma$  at all of the sample points. Find the triples  $(\hat{a}^i, \hat{b}^i, \hat{c}^i)$  with the largest uncertainty and add them to the training set.
- Generate a new GPR for the updated training set.
- Repeat steps 3-4 until stopping criteria is satisfied, e.g. training set reached predefined size.



### Results



### Conclusions

- For small data sets (2K points) GPR is more accurate
- GPR can utilize adaptive sampling
- GPR does not scale well to larger data sets (~2K points).
- NN scales well for large data sets and has better accuracy over GPR (more than 5K points).



# Floating Point Types

`float` and `double` are identified as simple or even primitive types, but they are much richer than it seems.

**Object** point of view: do these types fit into a hierarchy of classes?

⇒ Violation of the LISKOV's substitution principle (LSP)

if S subtypes T, what holds for T-objects holds for S-objects.

If S is a subtype of T, objects of type T in a program can be replaced by objects of type S without changing any of the desirable properties of that program (e.g. correct results)



A poorly encapsulated abstraction (*leaky*): we can measure the smallest positive non-zero float, the largest one, the machine epsilon, the base: we can access the implementation details



# Interface

Table: *leaky abstraction*: standardize the interface

C / C++	Fortran'90	ieee_arithmetic	Ada
<code>copysign (d x, d y)</code>	<code>sign (x, y)</code>	<code>ieee_copy_sign (x, y)</code>	<code>F'Copy_Sign (value, sign)</code>
<code>frexp (d x, i *exp)</code>	<code>exponent (x)</code>	<code>ieee_logb (x)</code>	<code>F'Exponent (x)</code>
<code>ldexp (d x, i exp)</code>	<code>fraction (x)</code>		<code>F'Fraction (x)</code>
<code>scalbn (d x, i exp)</code>	<code>set_exponent (x, i)</code>	<code>ieee_scalb (x, i)</code>	<code>F'Scaling (x, adjustment)</code>
<code>nextafter(d x, d y)</code>	<code>nearest (x, s)</code>	<code>ieee_next_after (x, y)</code>	<code>F'Adjacent (x, towards)</code>
<code>numeric_limits::radix</code>	<code>radix (x)</code>		<code>F'Machine_Radix</code>
<code>numeric_limits::epsilon ()</code>	<code>epsilon (x)</code>		<code>F'Model_Epsilon</code>
<code>numeric_limits::digits</code>	<code>precision (x)</code>		
	<code>digits (x)</code>		
	<code>range (x)</code>		
<code>numeric_limits::min_exponent</code>	<code>minexponent (x)</code>		<code>F'Machine_Mantissa</code>
<code>numeric_limits::max_exponent</code>	<code>maxexponent (x)</code>		<code>F'Machine_Emin</code>
	<code>spacing (x)</code>		<code>F'Machine_Emax</code>
	<code>rrspacing (x)</code>		
<code>nearbyint (d x)</code>			
<code>rint(d x)</code>	<code>nint (x)</code>	<code>ieee_rint (x)</code>	<code>F'Rounding (x)</code>
<code>floor (d x)</code>	<code>floor (x)</code>		<code>F'Floor (x)</code>
<code>ceil (d x)</code>	<code>ceiling (x)</code>		<code>F'Ceiling (x)</code>
		<code>ieee_rem (x, y)</code>	<code>F'Remainder (x, y)</code>

Unfortunately the C/C++ API doesn't vectorise well.



Not metrology: we do not seek “precision for precision’s sake”

The **functional** paradigm invites us to write computer function approaching mathematical functions, and we tend to focus on the aspect of **purity**.

But a mathematical function also seeks **totality** (being defined on the largest domain of definition):

*the function should be calculable for any argument for which it is defined.*

- removing non-jump and non-essential discontinuity:  $\Rightarrow \frac{\sin x}{x} \Big|_{x=0} = 1$  (naively  $\sin(0.0) / 0.0 = \text{NaN}$ )
- analytic continuation: factorial  $\Rightarrow \Gamma$ , or RIEMANN  $\zeta$  function
  - $\Rightarrow$  maximal extension of function domain
  - $\Rightarrow$  piecewise function definition, *casuistry*

Using IEEE-754 exceptional values, we can reach a “weak totality”:

- $\log(0.0) = -\text{Inf}$  (mathematically correct)
- $\log(-1.0) = \text{NaN}$  (mathematically correct? more precisely NaN)

Precision limitations lead to a gray zone in this kind of totality:

- $\expf(88.72284) = +\text{Inf}$  (but mathematically it's  $2^{128} \Rightarrow \text{domainException}$ )
- $\expf(-103.972084) = 0.0f$  (but mathematically it's just below  $2^{-150} \Rightarrow \text{domainException}$ )
- $\text{gammaf}(35.0401001) = +\text{Inf}$  (but mathematically it's  $2^{128}$ )

OK with `double`, but not with `float`.

Not all `Inf` have the same meaning, not all `NaN` have the same meaning, *cf* `null` in SQL



## « Why aiming for precision? »

⇒ Implicit **contract**: the fonction will

- 1 (if the argument is inside the mathematical domain of the mathematical function)
- 2 (if the type representation of the argument is inside the domain of the function that has a representable image in the return type)
- 3 return a result
- 4 this result is relevant(?)
- 5 (ideally the returned value is the representation of the image of the mathematical function applied on the represented argument)



# « Why aiming for precision? »

totality (mathematical) vs. representable totality

A representable solution resulting from representable arguments CAN go through a non-representable intermediate calculation. IEEE-754 exceptional values are not the value of the function, relative error of 100%, as in catastrophic cancelation.

## least surprise principle

- we agree to compute erroneous results, because we know that we cannot compute exact results: exact results are rarely (= almost never) representable:  $\pi$ ,  $e$ ,  $\sqrt{2}$ ,  $1/3$ ,  $1/5$  in base 2...
- On the other hand, we don't want things to be very wrong: mathematical result 2 but the function returns NaN

If the calculation is badly carried out, we can end up with

- infinite roots, where they exist and can be represented
- to an absence of roots, where they exist and are representable
- to a presence of roots, where they do not exist

**a difference of degree generates a difference of nature** (catastrophe theory, bifurcation, chaos)

The relative size of the danger zone in the parameter space will be much larger in low precision.

Annex for a less costly nondimensionalization:

« *You Could Learn a Lot from a Quadratic* » doi:10.1145/609742.609746, shows how to nondimensionalize with binary, much less costly in time and accuracy than divisions (and roots) in physicist nondimensionalization. Easy when knowing IEEE-754

API.



# « precision? » a take-away

$$\text{PRECISE NUMBER} + \text{PRECISE NUMBER} = \text{SLIGHTLY LESS PRECISE NUMBER}$$

$$\text{PRECISE NUMBER} \times \text{PRECISE NUMBER} = \text{SLIGHTLY LESS PRECISE NUMBER}$$

$$\text{PRECISE NUMBER} + \text{GARBAGE} = \text{GARBAGE}$$

$$\text{PRECISE NUMBER} \times \text{GARBAGE} = \text{GARBAGE}$$

$$\sqrt{\text{GARBAGE}} = \text{LESS BAD GARBAGE}$$

$$(\text{GARBAGE})^2 = \text{WORSE GARBAGE}$$

$$\frac{1}{N} \sum (N \text{ PIECES OF STATISTICALLY INDEPENDENT GARBAGE}) = \text{BETTER GARBAGE}$$

$$\left( \frac{\text{PRECISE NUMBER}}{\text{NUMBER}} \right)^{\text{GARBAGE}} = \text{MUCH WORSE GARBAGE}$$

$$\text{GARBAGE} - \text{GARBAGE} = \text{MUCH WORSE GARBAGE}$$

$$\frac{\text{PRECISE NUMBER}}{\text{GARBAGE} - \text{GARBAGE}} = \text{MUCH WORSE GARBAGE, POSSIBLE DIVISION BY ZERO}$$

$$\text{GARBAGE} \times 0 = \text{PRECISE NUMBER}$$

<https://xkcd.com/2295/>



# Variance: ALICE '08

- dimuons spectrometer, wire chamber:  $10^6$  channels
- $\forall$  channel,  $\exists$  noise:
  - ▶ average noise  $\bar{n} = En$
  - ▶ noise fluctuation  $\sigma = \sqrt{Vn}$
- $\Rightarrow$  99,7 % noise cut pedestal  $p = \bar{n} + 3\sigma = En + 3\sqrt{Vn}$   
to be reevaluated  $\forall$  channel, before each run (temperature or hygrometry fluctuations, high voltage, detector getting old)
- 2 ways:

$$\sigma^2 = Vn = \overline{(n - \bar{n})^2} \quad \text{2 passes} \geq 0 \quad (2)$$

$$= \overline{n^2} - (\bar{n})^2 \quad \text{1 pass} \quad (3)$$



BUT

- $n$  is digitized on 12 bits [0..4095]
  - $n^2$  is digitized on 24 bits
  - the sample has  $N = 500$  measures  $\sim 2^9$  (not big data...),  
 $\sum_N n^2$  is digitized on 33 bits
- $\Rightarrow$  doesn't fit in a float (24 bits mantissa),  
inexact representation (9 bits loss)  
 $\Rightarrow$  *catastrophic cancellation*, negative variance<sup>3</sup>,  
 $\Rightarrow$  trivially avoided with double precision!
- $\Rightarrow$  doesn't fit in an unsigned int, (overflow)  
 $\Rightarrow$  unsigned long long allows exact summations<sup>4</sup>

---

<sup>3</sup>seen on production launch



# Variance '91

- TP solid state physic:  
measuring solid state property transition with temperature
- IBM PC data acquisition with BASIC
- display of whisker boxes

⇒ online variance evaluation, 1-pass

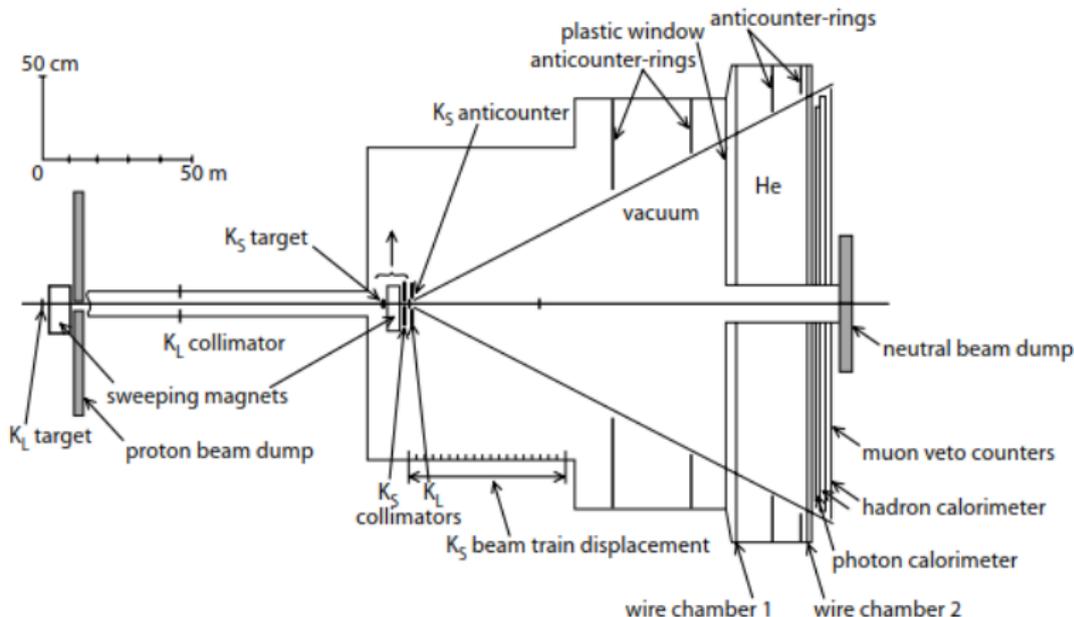
It was a time with no Wikipedia around to check for other algos...

- two passes approach
- arbitrary data shift towards some expected average value
- 1-pass online Welford's algorithm (one more division per iteration)



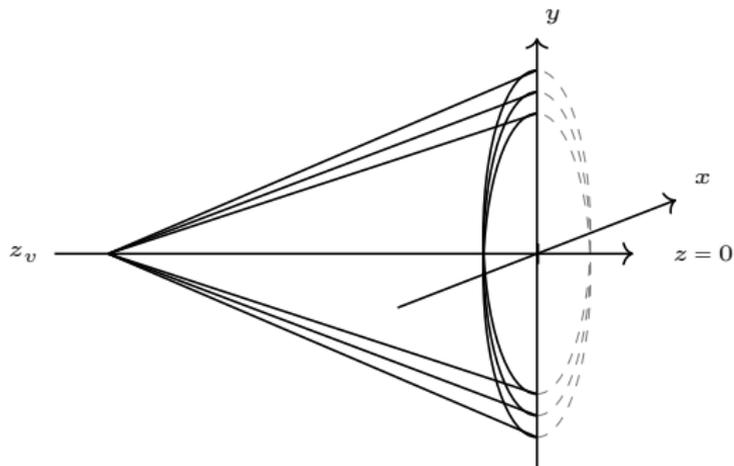
# Quadratic: NA-31 '92

- neutral kaons precise mass measurement
- $m_{K^0} = 497,611 \pm 0,013 = 497,611 \times (1 \pm 2,6 \times 10^{-5}) \text{ MeV}/c^2$   
 $m_{K^0} = 497,671 \pm 0,031 = 497,671 \times (1 \pm 6,2 \times 10^{-5}) \text{ MeV}/c^2$  in PDG 1990 & 1992
- $K^0 \rightarrow \pi^0 \pi^0$  and  $\pi^0 \rightarrow \gamma \gamma$ :  $K^0 \rightarrow \gamma \gamma \gamma \gamma$





# Quadratic: NA-31 '92





# Quadratic: NA-31 '92

- $\tau_{\pi^0} \ll \tau_{K^0}$  ( $\tau$  electromagnetic  $\ll$   $\tau$  weak)
- $E_{K^0} \sim 100 \text{ GeV}$      $\theta_{K^0\pi^0} \sim 4, 18 \text{ mrad}$   
 $E_{\pi^0} \sim 50 \text{ GeV}$      $\theta_{\pi^0\gamma} \sim 2, 70 \text{ mrad}$   
 $\Rightarrow$                      $\theta_{K^0\gamma} \in [1, 48; 6, 88] \text{ mrad}$
- $\Rightarrow$  small angles approximation  $\sin \theta \sim \theta \sim \tan \theta$
- for two photons impacts  $(w_1, E_1), (w_2, E_2)$  in the calorimeter,  
vertex reconstruction on axis  $z \sim \frac{|w_1 - w_2|}{\frac{m_{\pi^0}}{\sqrt{E_1 E_2}}} \underset{\eta \rightarrow 0}{\sim} \frac{|w_1 - w_2|}{\sqrt{2}\eta}$
- $\frac{\delta \sin}{\sin} \in [0, 11; 2, 37] \times 10^{-5}$   
 $\frac{\delta m}{m} \in [2, 61; 6, 23] \times 10^{-5}$
- $\sin \theta = \theta - \frac{1}{6} \theta^3 + o(\theta^3) = \theta \left( 1 - \frac{1}{6} \theta^2 + o(\theta^2) \right)$
- $\sin \arctan \rho = \rho - \frac{1}{2} \rho^3 + o(\rho^3) = \rho \left( 1 - \frac{1}{2} \rho^2 + o(\rho^2) \right)$



# Quadratic: NA-31 '92

biquadratic equation (quadratic in  $\mathcal{Z} = z^2$ )

$$\eta = \frac{m\pi}{\sqrt{2E_1E_2}} \sim 1.9 \times 10^{-3} \sim 2^{-9}$$

$$\xi = 1 - \eta^2$$

$$(1 - \xi^2) \mathcal{Z}^2 + [2\Re w_1 w_2^* - \xi^2(|w_1|^2 + |w_2|^2)] \mathcal{Z} + [(\Re w_1 w_2^*)^2 - \xi^2(|w_1|^2 |w_2|^2)] = 0$$

Expanded form

$$\begin{aligned} & \eta^2 (2 - \eta^2) \mathcal{Z}^2 \\ & - [ |w_1 - w_2|^2 - \eta^2 (2 - \eta^2) (|w_1|^2 + |w_2|^2) ] \mathcal{Z} \\ & - [ (\Im w_1 w_2^*)^2 - \eta^2 (2 - \eta^2) |w_1|^2 |w_2|^2 ] = 0 \end{aligned}$$

$$\Delta = \xi^2 [ |w_1 - w_2|^4 - \eta^2 (2 - \eta^2) (|w_1|^2 - |w_2|^2)^2 ]$$

The problem is not in evaluating roots (we compute the cancellation-free one: we can evaluate it the naive way)  
the problem is in properly evaluating polynomial coefficients  
the correction happens around 15th bit (single precision is enough);  
naive evaluation leads to 9th bit discrepancy  
the correction always occurs in the same direction.

- 1 Approximation wrongly evaluated
- 2 Quadratic solution level wrongly evaluated



# Quadratic: NA-31 '92

$$\eta = \frac{m_\pi}{\sqrt{2E_1 E_2}} \sim 1.9 \times 10^{-3} \sim 2^{-9}$$

$$\xi = 1 - \eta^2$$

$$z = \frac{|w_1 - w_2|^2 - \eta^2(2 - \eta^2)(|w_1|^2 + |w_2|^2)}{2\eta^2(2 - \eta^2)} + \frac{(1 - \eta^2)\sqrt{|w_1 - w_2|^4 - \eta^2(2 - \eta^2)(|w_1|^2 - |w_2|^2)^2}}{2\eta^2(2 - \eta^2)}$$

$$z = \frac{|w_1 - w_2|^2}{2\eta^2(2 - \eta^2)} - \frac{1}{2}(|w_1|^2 + |w_2|^2) + \frac{(1 - \eta^2)|w_1 - w_2|^2 \sqrt{1 - \eta^2(2 - \eta^2) \left(\frac{|w_1|^2 - |w_2|^2}{|w_1 - w_2|^2}\right)^2}}{2\eta^2(2 - \eta^2)}$$

$$z = \frac{(2 - \eta^2)|w_1 - w_2|^2}{2\eta^2(2 - \eta^2)} - \frac{1}{2}(|w_1|^2 + |w_2|^2) + (1 - \eta^2)|w_1 - w_2|^2 \frac{\sqrt{1 - \eta^2(2 - \eta^2) \left(\frac{|w_1|^2 - |w_2|^2}{|w_1 - w_2|^2}\right)^2} - 1}{2\eta^2(2 - \eta^2)}$$

$$z = \left(\frac{|w_1 - w_2|}{\sqrt{2}\eta}\right)^2 - \frac{1}{2}(|w_1|^2 + |w_2|^2) + (1 - \eta^2)|w_1 - w_2|^2 \frac{\sqrt{1 - \eta^2(2 - \eta^2) \left(\frac{|w_1|^2 - |w_2|^2}{|w_1 - w_2|^2}\right)^2} - 1}{2\eta^2(2 - \eta^2)}$$

$$z = \left(\frac{|w_1 - w_2|}{\sqrt{2}\eta}\right)^2 \left[ 1 - \eta^2 \frac{|w_1|^2 + |w_2|^2}{|w_1 - w_2|^2} + (1 - \eta^2) \frac{\sqrt{1 - \eta^2(2 - \eta^2) \left(\frac{|w_1|^2 - |w_2|^2}{|w_1 - w_2|^2}\right)^2} - 1}{2 - \eta^2} \right]$$



Matrix condition number

$$\kappa(A) = \|A^{-1}\| \|A\| \geq \|A^{-1}A\| = 1$$

maximum ratio in relative error on  $x$  to relative error on  $b$  in  $Ax = b$

multi variables functions condition number

$$\kappa = \frac{\|J(x)\|}{\|f(x)\|/\|x\|} = \frac{\|x\| \|J(x)\|}{\|f(x)\|} = \frac{\|x\| \kappa \left\| \frac{\partial f_i}{\partial x_j} \right\|}{\|f(x)\|}$$



# Dimensional analysis

- scale relativity: *laws of nature shouldn't depend on our arbitrary scale choices*  
(in the same way that there is neither center of the Universe, nor privileged direction, nor reference Ether of the speeds, nor **absolute** reference potential... but centers, directions, reference frame, scales **relevant relatively** to a given problem)  
invariance of scale, covariance of scale
- Scale effects: KANT: intensive / extensive quantities  
GALILEO: size of animals (*Square-Cube Law*); KLEIBER: animal metabolism (3/4, not 2/3)
- **dimension**: nature of a quantity (length, surface, volume, mass, duration, speed, force, angle...)
- **homogeneity**: *thou shouldst only add same dimension quantities (cf homogeneous function  $f(tx) = t^\alpha f(x)$ )*
- pendulum period (small oscillations):  $T = 2\pi\sqrt{\frac{l}{g}}$      $g \sim L \cdot T^{-2}$
- constraint on the arguments of analytic functions:  $f(z) = \sum_n a_n z^n \Rightarrow z$  dimensionless argument (and result  $f$ )
- fluid mechanics & scaling (models): MACH  $\frac{v}{c_s}$ , REYNOLDS  $\frac{VL}{\nu}$ , FROUDE, WEBER, PRANDTL, NUSSELT, GRASHOF, HERSEY, SOMMERFELD...
- SOMMERFELD fine structure constant:  $\alpha = \frac{e^2}{4\pi\epsilon_0\hbar c} = 1/137,035\,999\,206\dots$
- SCHWARZSCHILD black hole horizon:  $R_s = \frac{2GM}{c^2}$
- PLANCK relativistic quantum gravity:  $l_P = \sqrt{\frac{\hbar G}{c^3}} = 1,6 \times 10^{-35}$  m,  $t_P = \sqrt{\frac{\hbar G}{c^5}} = 5,4 \times 10^{-44}$  s
- Sir Geoffrey Ingram TAYLOR, 1950: Trinity test atomic bomb energy, starting from photos with radius of action:  
 $R(t, E, \mu) = Kt^{\frac{2}{5}} E^{\frac{1}{5}} \mu^{-\frac{1}{5}}$
- BERTRAND, 1878: cooling time of the homogeneous sphere
- dimensional equation, (BERTRAND-VASCHY-BUCKINGHAM) *II théorème A physically meaningful equation involving a certain number  $n$  of physical variables, can be rewritten in terms of a set of  $p = n - k$  dimensionless parameters  $\pi_1, \pi_2, \dots, \pi_p$  constructed from the original variables, where  $k$  is the number of physical dimensions involved*
- « **zeroth principle of theoretical Physics** » (John A. WHEELER, *Spacetime Physics*, 1966)  
*« Never make a calculation until you know the answer. »*  
qualitative Physics, heuristic guide, and source of inspiration for the rest of us.



# The art of nondimensionalization dimensional analysis equation

Back to our quadratic function equation  $y = ax^2 + bx + c$

Let's use the following notations for dimensions of the argument and of the function:  $x \sim X$ ,

$y \sim Y$

Then the dimensions of coefficients are

- $c \sim Y$
- $b \sim Y \cdot X^{-1}$
- $a \sim Y \cdot X^{-2}$
- $\Rightarrow \Delta = b^2 - 4ac \sim Y^2 \cdot X^{-2}$

We try to evaluate the possible roots which are homogeneous to  $X$ , from the coefficients, of which none is homogeneous to  $X$ .

What quantities can be formed as monomials of the data in the problem?

$$G \sim a^n b^m c^l \sim X^{-2n-m} Y^{n+m+l}$$

- $G \sim X \Leftrightarrow \begin{array}{rcl} n & + & m & + & l & = & 0 \\ -2n & - & m & & & = & 1 \end{array}$  affine equation
- $G \sim Y \Leftrightarrow \begin{array}{rcl} n & + & m & + & l & = & 1 \\ -2n & - & m & & & = & 0 \end{array}$  affine equation
- $G \sim 1 \Leftrightarrow \begin{array}{rcl} n & + & m & + & l & = & 0 \\ -2n & - & m & & & = & 0 \end{array}$  linear equation



# The art of nondimensionalization

We lack an equation...

- normal for the dimensionless  $\alpha$  coefficient:

$$\forall \lambda, \alpha^\lambda \sim 1 \text{ is also dimensionless}$$

- if the linear system was fully specified, it would get a single null solution:

$$G = a^0 b^0 c^0 = 1 \text{ would be the only dimensionless parameter of this problem}$$

- How to plug the holes?  $\Rightarrow$  exploit formal symmetry between  $a$  and  $c$  (associating one of the zeros to its reciprocal):  $l = \pm n$

$\alpha = \left| \frac{ac}{b^2} \right|^{\lambda=1}$  is the simplest to evaluate, but is subject to overflow (and underflow)

$\beta = \sqrt{\alpha} = \left| \frac{ac}{b^2} \right|^{\lambda=\frac{1}{2}} = \frac{\sqrt{|a|}\sqrt{|c|}}{|b|}$  is better protected from overflow (and ...)

$\Rightarrow$  exponents are smaller (limits the exposure to overflow),  
but not the simplest (roots must be extracted)

- the solutions of affine systems are the linear combination of a particular solution of the affine system and the space of solutions of the linear system

a solution for the scale  $X$  is  $\frac{|b|}{|a|}$ ; solutions for the scale  $X$  are  $\forall \mu, \frac{|b|}{|a|} \beta^\mu$

so for example, another solution for  $X$  is  $\frac{|b|}{|a|} \beta = \frac{\sqrt{|c|}}{\sqrt{|a|}}$



# The art of nondimensionalization

$$Y \sim |b| \sqrt{\frac{|c|}{|a|}}$$

What is the interest of  $Y$  in finding roots, homogeneous to  $X$ ?

When we **check that a value**  $x_0$  found as root of the trinomial  $y = P(x) = ax^2 + bx + c$  is **valid**, we won't find exactly  $P(x_0) = 0$  :

the rounding errors mean that we expect to find a value of the order of  $\varepsilon_{\text{machine}}$ .  
But  $\varepsilon_{\text{machine}}$  is dimensionless, and dimensional analysis tells us that  $P(x_0) \sim Y$ :

$$P(x_0) \approx Y \varepsilon_{\text{machine}}$$

So we can find values of  $P(x_0)$  which seem to be large but remain small in front of  $Y$ , and it is this smallness which makes it a good root.

Similarly, to **evaluate the sign of  $\Delta$**  the theoretical threshold of 0 will be obtained for  $\varepsilon_{\text{machine}} Y^2 \cdot X^{-2} = \varepsilon_{\text{machine}} |ac|$ . This reflection applies to all **comparisons between floats**: "*float equality considered harmful*": it is illusory to expect strict equalities, and if the margin associated with rounding can only be proportional to  $\varepsilon_{\text{machine}}$ , it must also have the right dimension.

$$\begin{aligned}
 X &\sim \frac{|b|}{2|a|} & Y &\sim \frac{|b|^2}{4|a|} & \beta' &= 2 \frac{\sqrt{|a|}\sqrt{|c|}}{|b|} & x &= X\xi & y &= Y\nu \\
 \nu &= \frac{P(X\xi)}{Y} & &= \text{sgn } a \left[ \xi^2 + 2 \text{sgn } a \text{sgn } b \xi + \text{sgn } a \text{sgn } c \beta'^2 \right] & &= \text{sgn } a \left[ \xi^2 + 2\sigma\xi + \tau\beta'^2 \right] \\
 \xi_{\pm} &= -\sigma \left[ 1 \pm \sqrt{\delta = 1 - \tau\beta'^2} \right] & \Rightarrow & \xi_1 = -\sigma \left[ 1 + \sqrt{\delta} \right] & ; & \xi_2 = \tau \frac{\beta'^2}{\xi_1} \\
 \delta = 1 - \beta'^2 &= (1 - \beta')(1 + \beta') & \parallel & \delta = 1 + \beta'^2 = \text{fma}(\beta', \beta', 1) & ; & \beta' > \sqrt{2/\varepsilon} \Rightarrow \sqrt{\delta} := \beta' \\
 & & & \beta' > 2/\varepsilon \Rightarrow \sqrt{\delta} := \beta' & ; & \xi_1 = -\sigma\beta' & ; & \xi_2 = -\sigma\tau\beta'
 \end{aligned}$$



# The art of nondimensionalization

$$\begin{aligned} a &= \sigma_a 2^\alpha A & b &= \sigma_b 2^\beta B & c &= \sigma_c 2^\gamma C \\ a &= \sigma_a 2^{2\alpha'} A' & b &= \sigma_b 2^\beta B & c &= \sigma_c 2^{\gamma'} C' \\ A' &\in [1; 4[ & B &\in [1; 2[ & C' &\in [1; 4[ \end{aligned}$$
$$X \sim \frac{|b|}{2|a|} = 2^{\beta-2\alpha'-1} \frac{B}{A'} \quad Y \sim \frac{|b|^2}{4|a|} = 2^{2(\beta-\alpha'-1)} \frac{B^2}{A'}$$
$$\begin{aligned} \Delta &= 1 - 4 \frac{\sigma_a 2^{2\alpha'} A' \times \sigma_c 2^{2\gamma'} C'}{(2^\beta B)^2} \\ &= 1 - 4 \sigma_a \sigma_c 2^{2\alpha'+2\gamma'-2\beta} \frac{A' C'}{B^2} \\ &= 1 - \sigma_a \sigma_c 2^{2(\alpha'+\gamma'-\beta+1)} \frac{A' C'}{B^2} \end{aligned}$$
$$\begin{aligned} A' C' &\in [1; 16[ & B^2 &\in [1; 4[ & B^{-2} &\in [1/4; 1] & A' C' B^{-2} &\in [1/4; 16[ \\ \sqrt{A' C' B^{-2}} &= \sqrt{A' C'} B^{-1} &\in [1/2; 4[ & B A'^{-1} &\in [1/4; 2[ & B^2 A'^{-1} &\in [1/4; 4[ \\ \tau &= 2^{\alpha'+\gamma'-\beta+1} \sqrt{A' C' B^{-2}} = 2^{\alpha'+\gamma'-\beta+1} \sqrt{A' C'} B^{-1} \end{aligned}$$