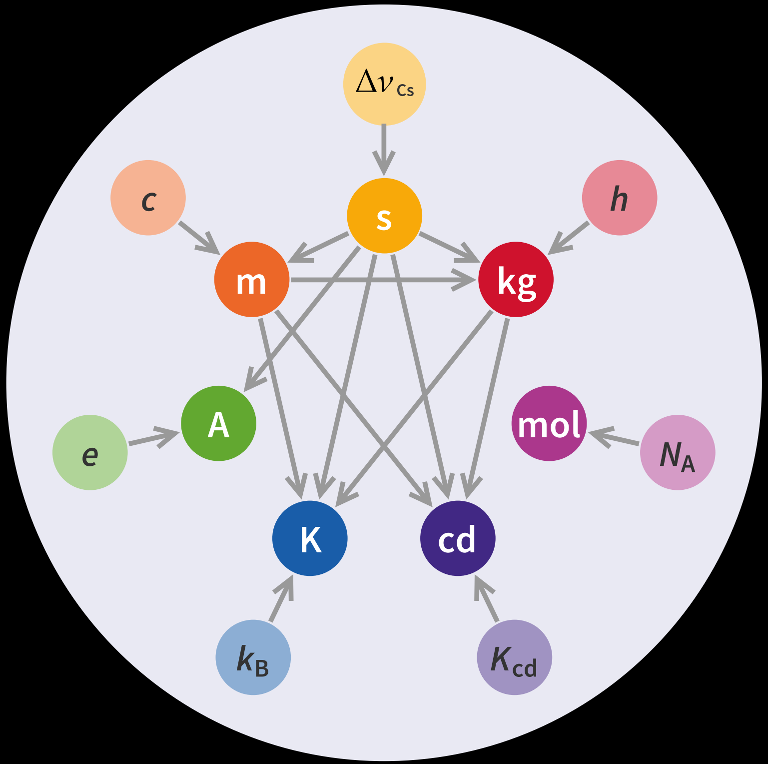
On the Fundamental Nature of Unit Systems and Physical Quantities

Animascience – January 26th, 2024 – LAPP

Vincent Reverdy



What are physical units?

An introductory tale

Imagine,

A LONG TIME AGO IN A GALAXY FAR, FAR AWAY...

or more precisely $5.2 \times 10^9\ years$ ago,
in a spiral galaxy of $150 \times 10^9\ M_\odot$
at a comoving distance of $1900\ Mpc \cong 5.863 \times 10^{22}\ km$ from us,
at redshift of $z = 0.5$
corresponding to a scale factor of $a = 2/3$,
on a planet of average density $6.5\ g \cdot cm^{-3}$
orbiting two stars with surface temperatures of $5200\ K$ and $6100\ K$

...THERE WAS A BASE OF THE GALACTIC EMPIRE...

Imagine,

IN A PARTICLE ACCELERATOR...

with a $27\ km$ circonference,
maintained at a cryogenic temperature of $1.9\ K$,
accelerating protons at $99.9999991\ \%$ of the speed of light
in a cavity with an initial pressure of $10^{-13}\ atm$
using eletromagnets producing a field of $8.3\ T$
thanks to a total current of $11080\ A$
in order to produce collisions at $13.6\ TeV$

...WE DETECTED FOR THE FIRST TIME THE HIGGS BOSON...

m, kg, s, K ... $length, mass, time, temperature$ ...

Units and quantities are at the basis of physics. They are critical to specify, run, and analyze experiments. Dimensional analysis is key to theoretical physics.

Beyond than that units and quantities are an essential part of our language to describe the world and produce accurate mental pictures.

A paradox/anomaly

(even if plenty of units and quantities software libraries exist)

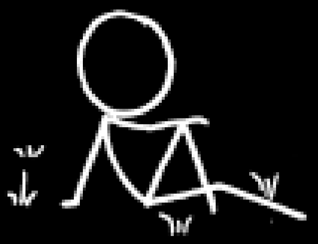
Units and quantities are ubiquitous in physics and descriptive language.

For the most part, nobody uses them in physics code and programming languages.

Work hypotheses

A. « Soft science » reasons: sociology of research, diffusion of practices...

➡ B. « Hard science » reasons: a fundamental maths/info/physics problem ⬅



Framing the problem of numerical units

Starting from real-life code

A C++ navigation code actually used to fly airplanes

```cpp
void xXY_Brg_Rng(double X_1, double Y_1, double X_2, double Y_2, double *Bearing, double *Range);

void DistanceBearing(double lat1, double lon1,
                     double lat2, double lon2,
                     double *Distance, double *Bearing);

double DoubleDistance(double lat1, double lon1,
                      double lat2, double lon2,
                      double lat3, double lon3);

void FindLatitudeLongitude(double Lat, double Lon,
                           double Bearing, double Distance,
                           double *lat_out, double *lon_out);

double CrossTrackError(double lon1, double lat1,
                       double lon2, double lat2,
                       double lon3, double lat3,
                       double *lon4, double *lat4);

double ProjectedDistance(double lon1, double lat1,
                         double lon2, double lat2,
                         double lon3, double lat3,
                         double *xtd, double *crs);

void LatLon2Flat(double lon, double lat, int *scx, int *scy);
```

Solving the problem with a standard library to rule them all

How it all started

How to make units and quantities available in the C++ programming language?

ISO WG21 Standards Committee

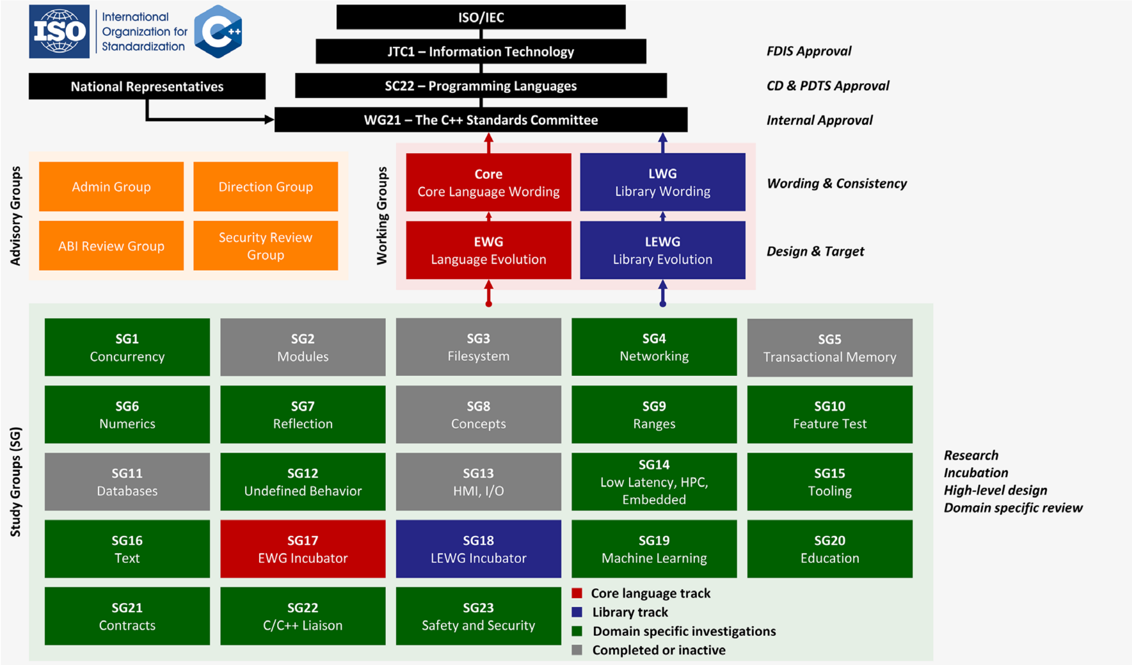
- C++ Programming Language

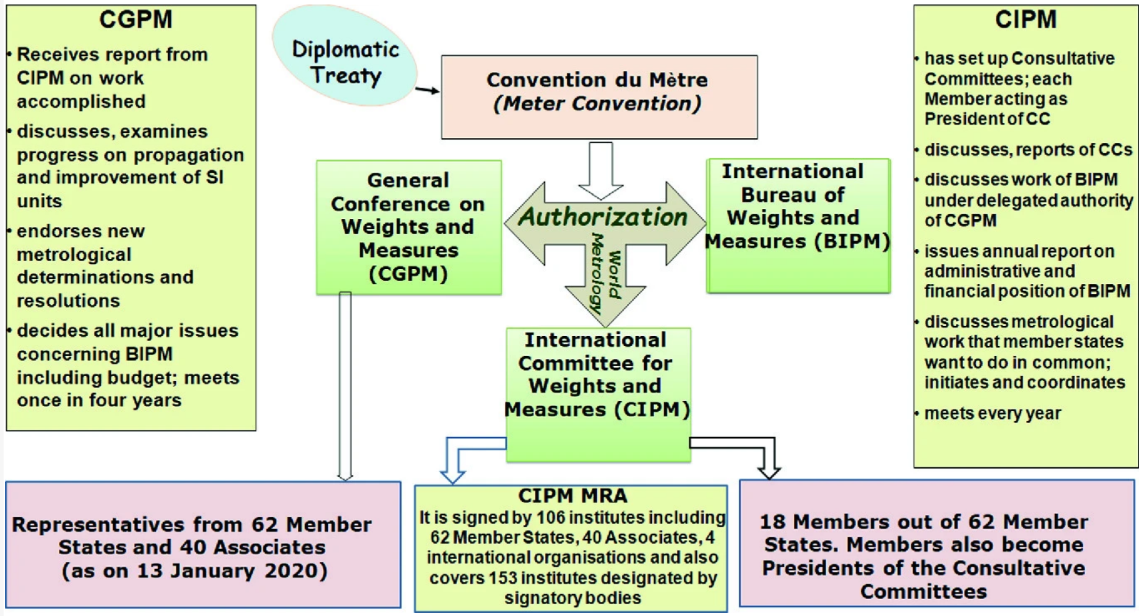
BIPM

- International System of Units (SI)

ISO 80000

- Quantities and Units

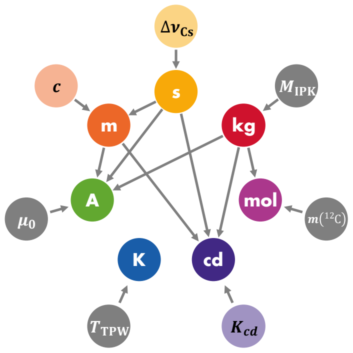




Hint

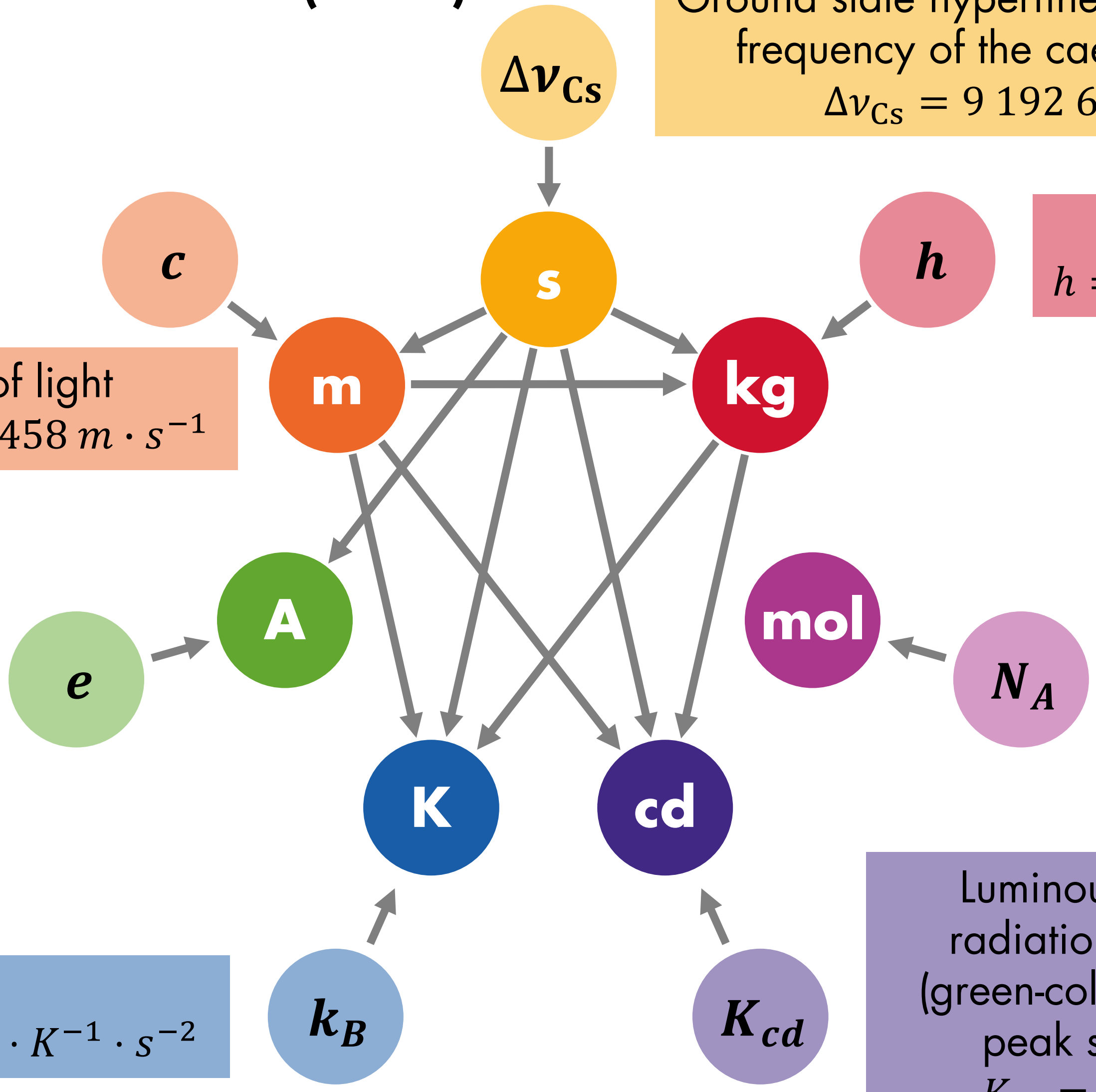
It's insanely complicated...

The 2019 redefinition from 7 exactly defined constants



Old SI (1983)

New SI (2019)



Ground state hyperfine structure transition frequency of the caesium-133 atom
$\Delta\nu_{Cs} = 9\,192\,631\,770\ s^{-1}$

Planck constant
$h = 6.62607015 \times 10^{-34}\ kg \cdot m^2 \cdot s^{-1}$

Speed of light
$c = 299\,792\,458\ m \cdot s^{-1}$

Elementary charge
$e = 1.602176634 \times 10^{-19}\ A \cdot s$

Avogadro constant
$N_A = 6.02214076 \times 10^{23}\ mol^{-1}$

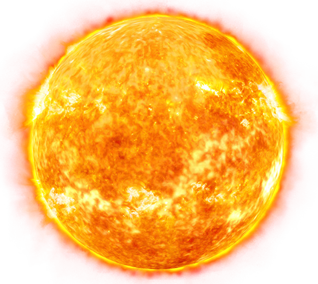
Boltzmann constant
$k_B = 1.380649 \times 10^{-23}\ kg \cdot m^2 \cdot K^{-1} \cdot s^{-2}$

Luminous efficacy of monochromatic radiation of frequency $540 \times 10^{12}\ Hz$ (green-colored light at approximately the peak sensitivity of the human eye)
$K_{cd} = 683\ cd \cdot sr \cdot s^3 \cdot kg^{-1} \cdot m^{-2}$

Quantities with measurement units

$$Q_a = \{Q_a\}\,[Q]$$

- Q_a → Symbol for individual quantity
- $\{Q_a\}$ → Symbol for the numerical value of the quantity Q_a expressed in the unit $[Q]$
- $[Q]$ → Symbol for unit



$$M_\odot = 1.98847 \times 10^{30}\ kg$$

$$\{M_\odot\} = \frac{(1.98847 \times 10^{30}\ kg)}{kg} = 1.98847 \times 10^{30}$$

$$[M_\odot] = kg$$

Quantity values

Quantity (ISO 80000-1:2009)

Property of a phenomenon, body, or substance, where the property has a magnitude that can be expressed by means of a number and a reference.

Scalar | Vector | Tensor

Quantity value = Number + Reference

Measurement unit

- Length of a given rod: $\boxed{5{,}34\ m}$
- Mass of a given body : $\boxed{0{,}152\ kg}$
- Curvature of a given arc: $\boxed{112\ m^{-1}}$
- Electric impedance of a given circuit element at a given frequency: $\boxed{(7+3j)\ \Omega}$
- Refractive index of a given sample of glass (dimensionless, unit 1): $\boxed{1{,}32}$

Measurement procedure

- Rockwell C hardness of a given sample (150 kg load): $\boxed{43.5\ HRC(150\ kg)}$

Reference material

- Amount-of-substance concentration of lutropin in a given sample of plasma (WHO international standard 80/552): $\boxed{5{,}0\ IU/l}$ (WHO International Units per litre)

Standardization for programming

At this point, not that simple but doable… Does it get worse?

Exploring edge cases and conceptual limits

Unit calculus on a computer

$$1200\, m + 2.4\, km = 1200\, m + 2400\, m = 3600m$$ ☑

$$45\, kg + 8300\, g = 45\, kg + 8.3\, kg = 53.3\, kg$$ ☑

$$1.3\, s + 2900\, ms = 1.3\, s + 2.9\, s = 4.2\, s$$ ☑

And what about…

$$4.24\, ly + 3.51\, pc = ?$$

Possible strategies

- smallest unit $\Longrightarrow 4.24\, ly + 11.44809 \ldots\, ly = 15.68809 \ldots ly$
- largest unit $\Longrightarrow 1.2999899 \ldots\, pc + 3.51\, pc = 4.8099899 \ldots pc$
- leftmost unit $\Longrightarrow 4.24\, ly + 11.44809 \ldots\, ly = 15.68809 \ldots ly \Longrightarrow$ not commutative anymore
- rightmost unit $\Longrightarrow 1.2999899 \ldots\, pc + 3.51\, pc = 4.8099899 \ldots pc \Longrightarrow$ not commutative anymore
- base unit $\Longrightarrow 4.011 \ldots \times 10^{16}\, m + 1.083 \ldots \times 10^{17}\, m = 1.484 \ldots \times 10^{17}\, m \Longrightarrow$ precision loss/overflow

Result

All solutions lead to different results because of machine precision

Two subproblems ($2 \times 5 = 10$ different strategies)

- Unit used for internal computation
- Unit used to display the result

Just a technical problem?

$$1200\,m + 2.4\,km = 1200\,m + 2400\,m = 3600m$$ ☑

$$45\,kg + 8300\,g = 45\,kg + 8.3\,kg = 53.3\,kg$$ ☑

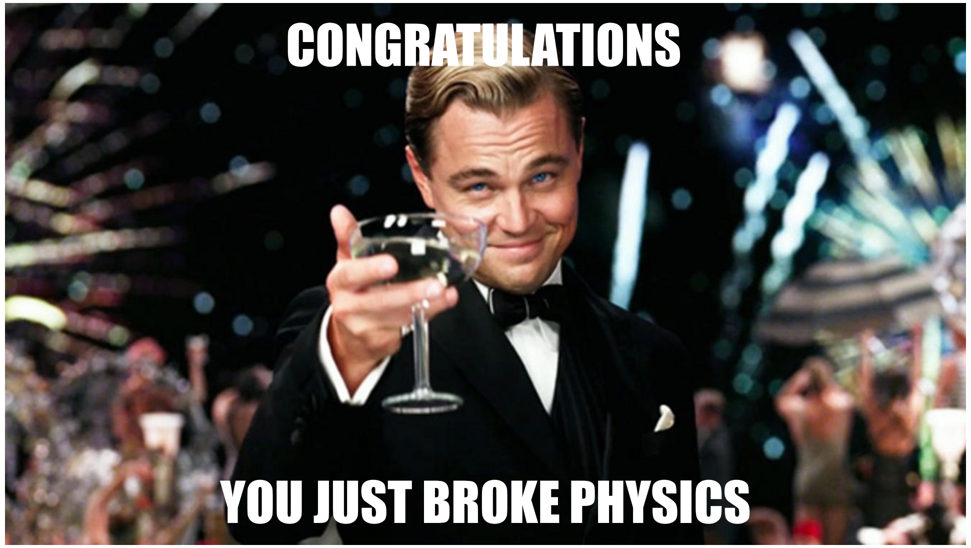
$$1.3\,s + 2900\,ms = 1.3\,s + 2.9\,s = 4.2\,s$$ ☑

And what about…

$$152\,K + 17\,°C = ?$$

Result

$$152\,K + 17\,°C = 152K + 290.15\,K = 442.15\,K$$ ☒



Temperature is more complicated

Two concepts:

- quantity point
- quantity difference

Mathematically speaking

Temperatures live in an affine space

Temperature is a scale

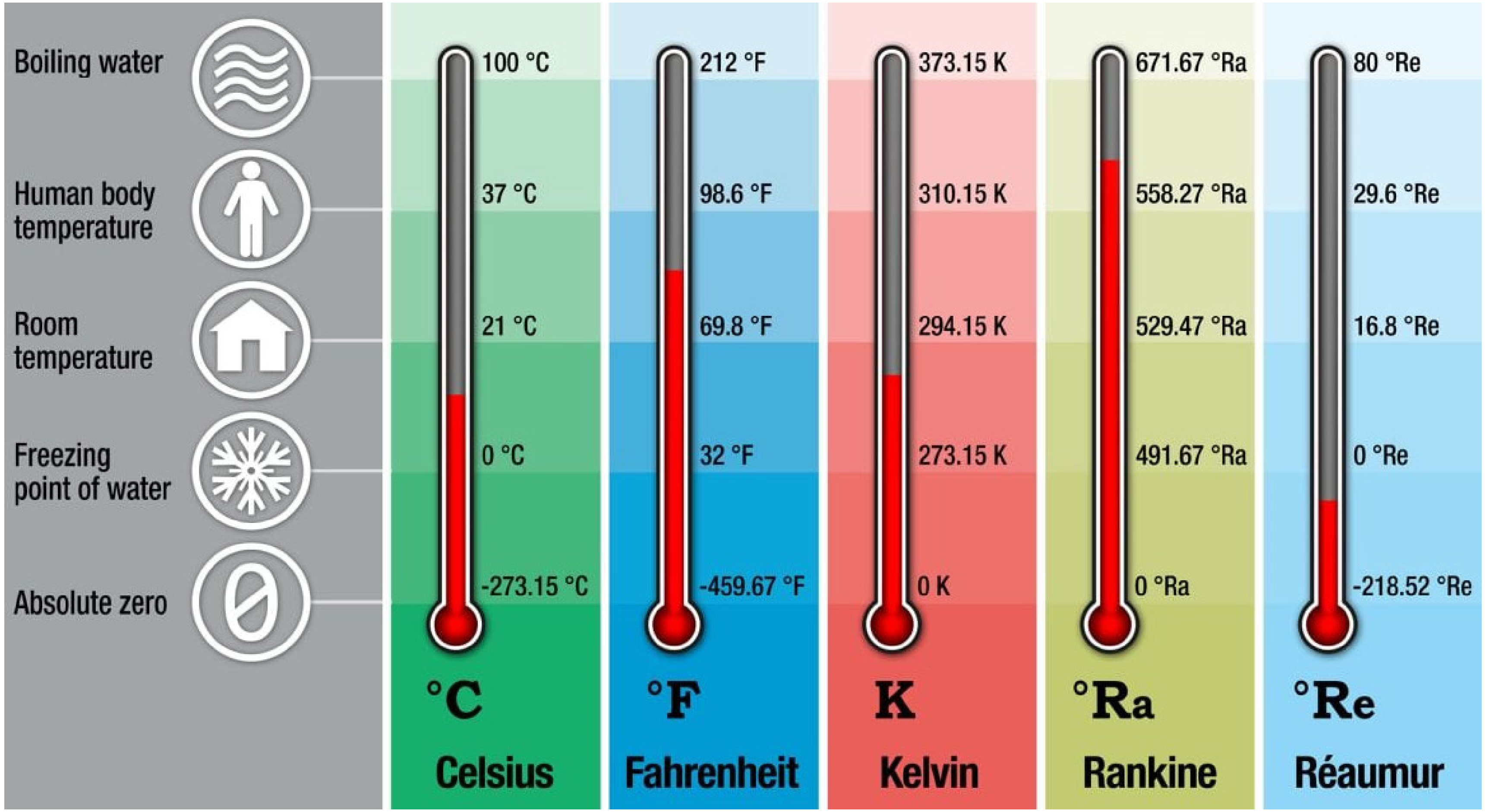
Hotness series: $\mathbb{H}$ (ordered countable set dense-in-itself)

Fixed points: $\mathbb{F}$ (ordered countable set dense-in-itself)

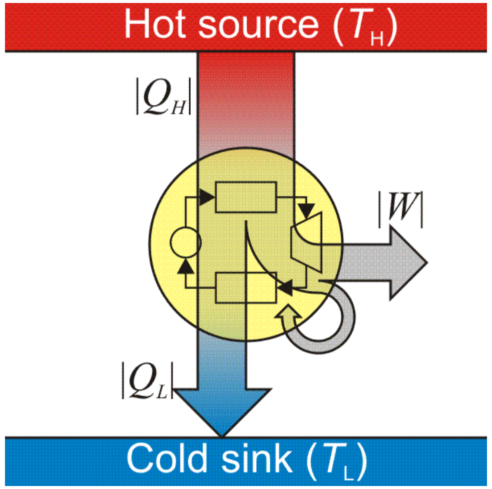
Temperature scale

Ideal heat engine [1]

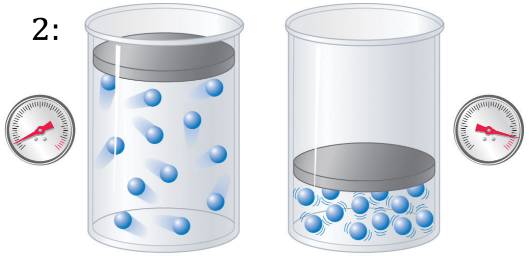
Ideal gas [2]



1:

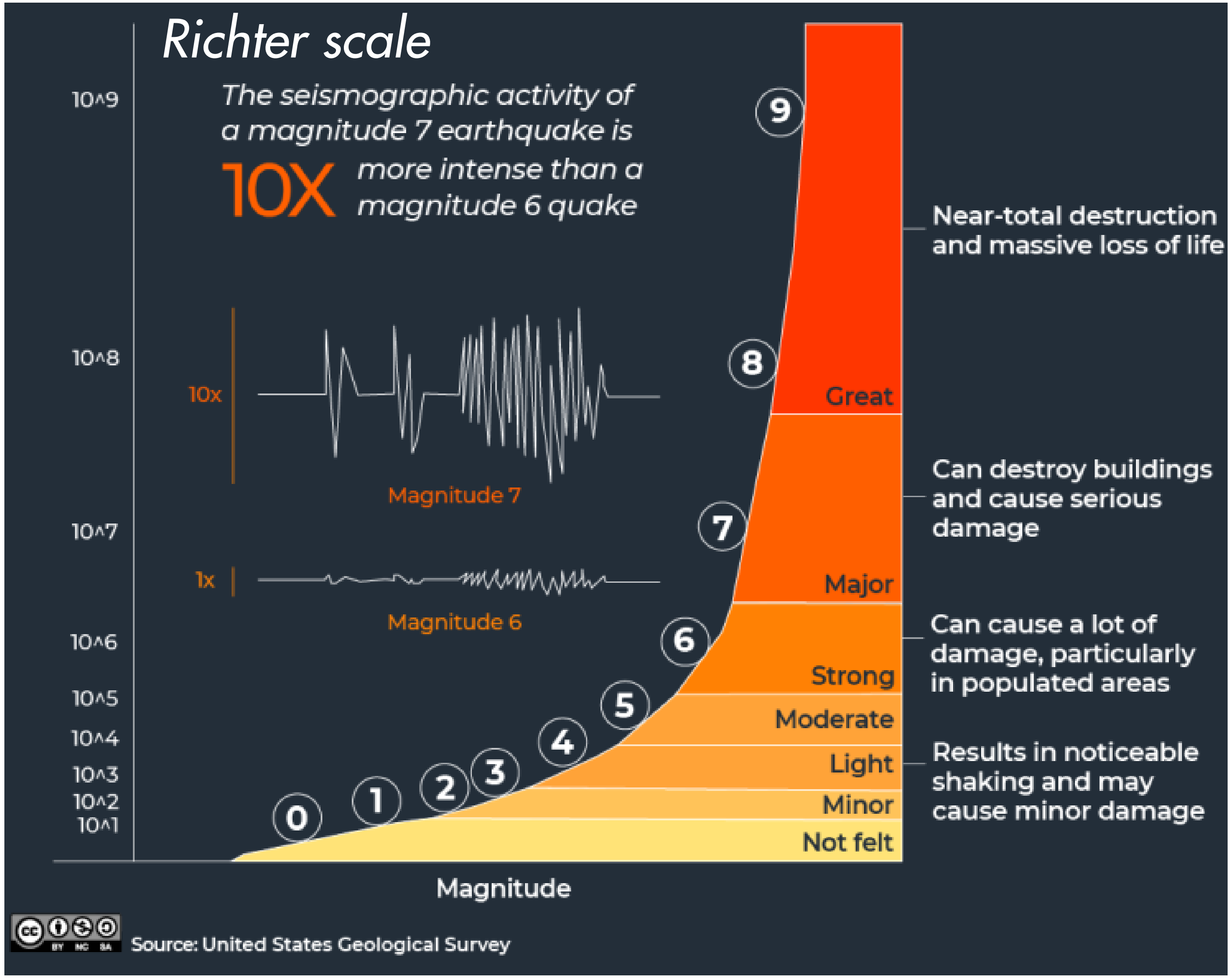


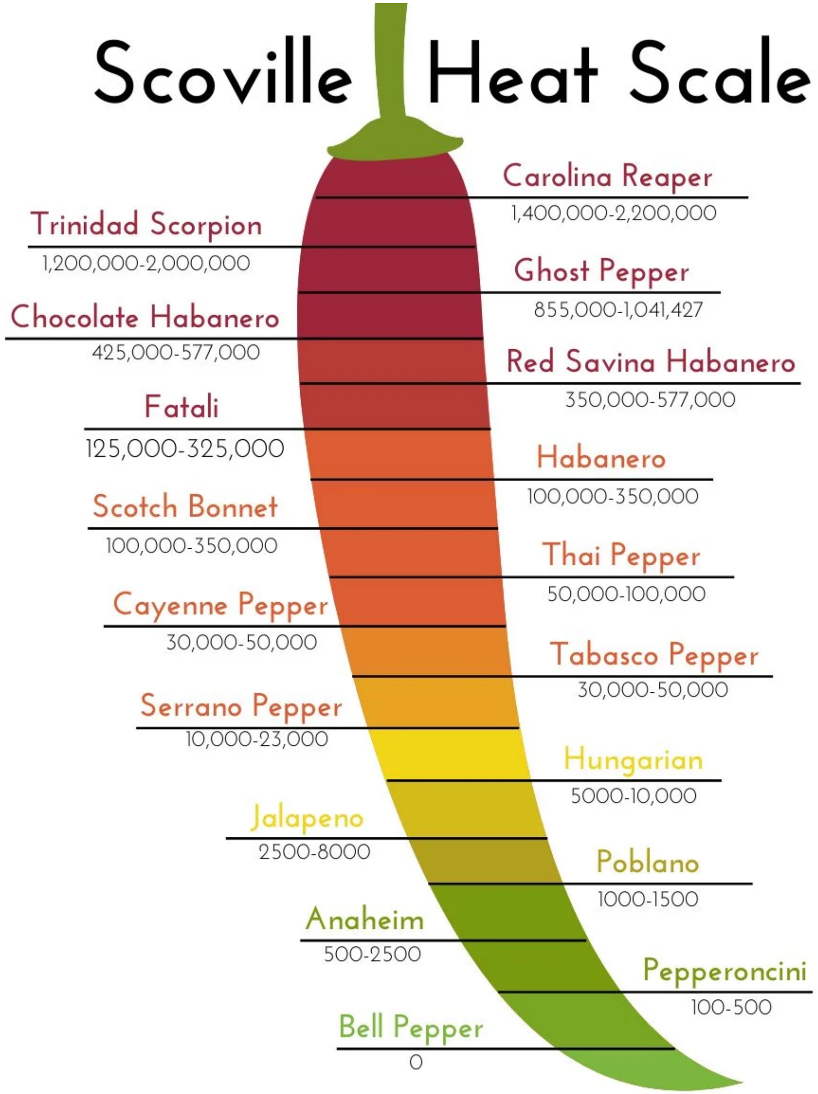
2:



Reference: *Do we know what the temperature is?, Mares, Journal of Thermal Analysis and Calorimetry, 2015*

Ordinal units





Logarithmic quantities

Nepers and decibels

$$L_P = \frac{1}{2}\ln\left(\frac{P}{P_0}\right) Np = 10 \times \log_{10}\left(\frac{P}{P_0}\right) dB$$

$2.878\ Np$
Nepers

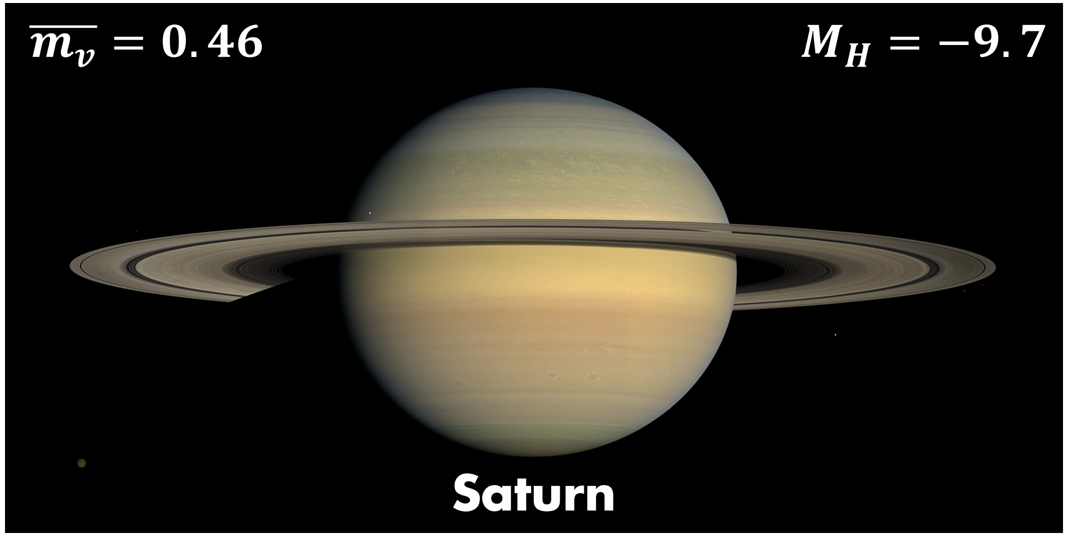
$25\ dB$
Decibels

Apparent magnitude

$$m_x = -5\log_{100}\left(\frac{F_x}{F_x^0}\right)$$

Absolute magnitude

$$M_\star - M_\odot = -2.5\log_{10}\left(\frac{L_\star}{L_\odot}\right)$$



Dimensions and quantity kinds

Same kind $\Rightarrow$ Same dimension

Same dimension $\not\Rightarrow$ Same kind

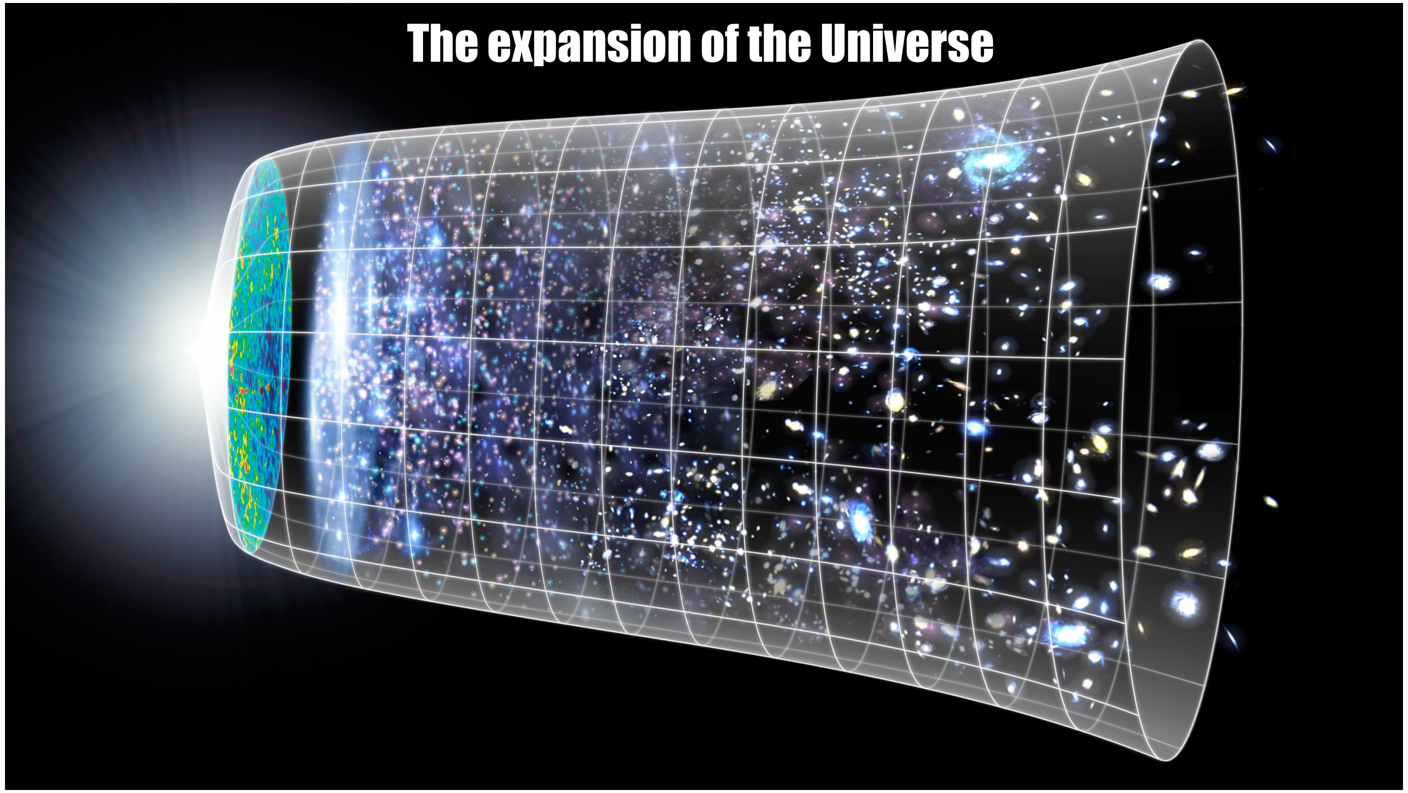
Example 1	Example 2	Example 3
Energy: $M \cdot L^2 \cdot T^{-2}$	Entropy: $M \cdot L^2 \cdot T^{-2} \cdot \Theta^{-1}$	Frequency: T^{-1}
Moment of force: $M \cdot L^2 \cdot T^{-2}$	Heat capacity: $M \cdot L^2 \cdot T^{-2} \cdot \Theta^{-1}$	Radioactive activity: T^{-1}

$$1.2\ Bq + 5.0\ Hz = ?$$ ✗

Explicit cast

$$1.2\ s^{-1} + 5.0\ s^{-1} = 6.2\ s^{-1}$$ ✓

Unit simplification



Hubble's constant

$$H_0 = 73.2\ (km/s)/Mpc$$

$$\dim H_0 = T^{-1} \rightarrow [H_0] = s^{-1}?\quad [H_0] = Hz?\quad [H_0] = Bq?$$

Units collapse

```
auto v = quantity(67.)[km/s];
auto d = quantity(1.)[Mpc];
auto H = quantity(70.)[km/s/Mpc];
std::cout << v / d << std::endl;  // Display in s^-1 or km/s/Mpc?
std::cout << H << std::endl;      // Display in s^-1 or km/s/Mpc?
```

Evolution of unit systems

Working with old data

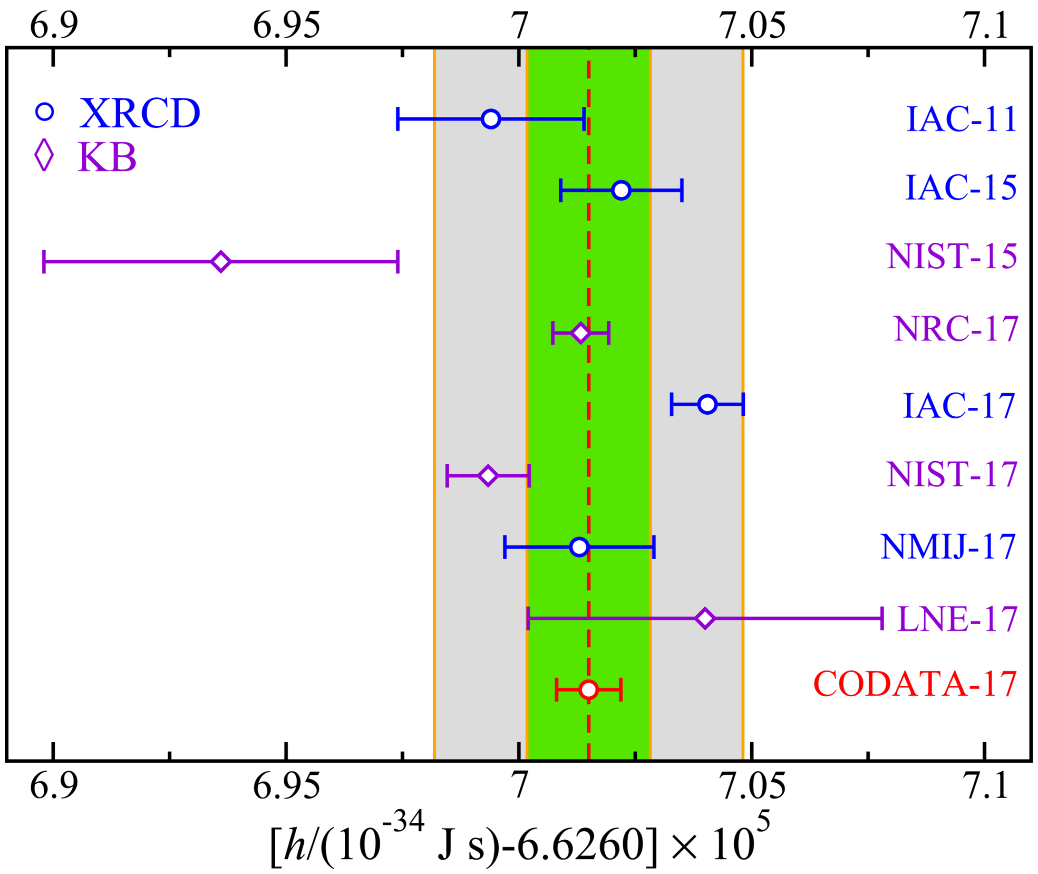
```cpp
#include "oldexperiment.hpp"
using namespace units::systems::si;

auto old_m = oldexperiment.get_mass();// in kg
auto new_m = mass(1.4289)[kg];        // in kg

auto mass_difference = new_d - old_d; // problem?
```

Evolving standards

- SI:1960 vs SI:1983 vs SI:2019
- Physical constants gets refined over time
- In software, what to do when users mix kg and K from SI:1960 with kg and K from SI:2019?

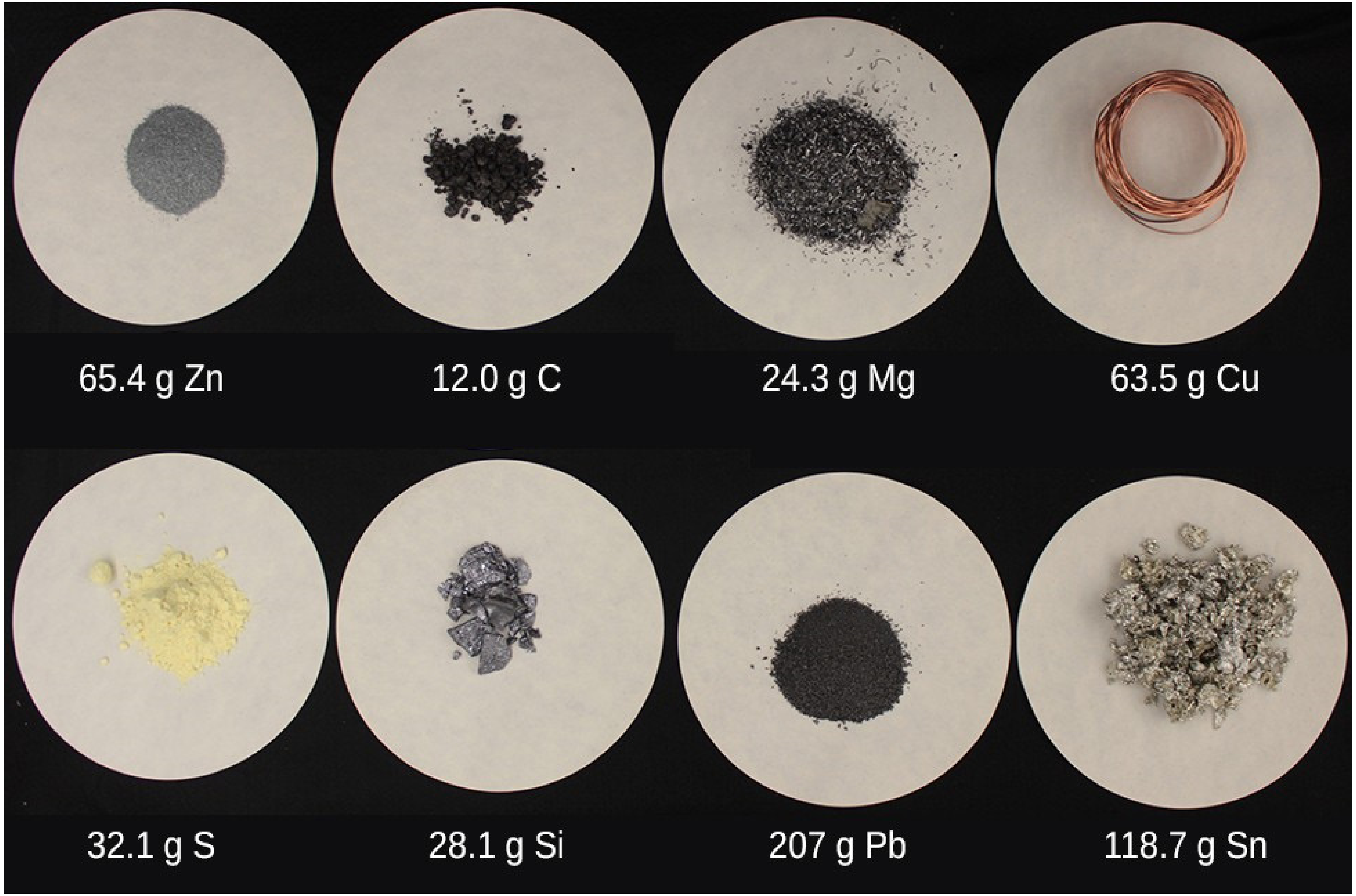


What to do in software...

- ... to handle the mixing of old units and new units?

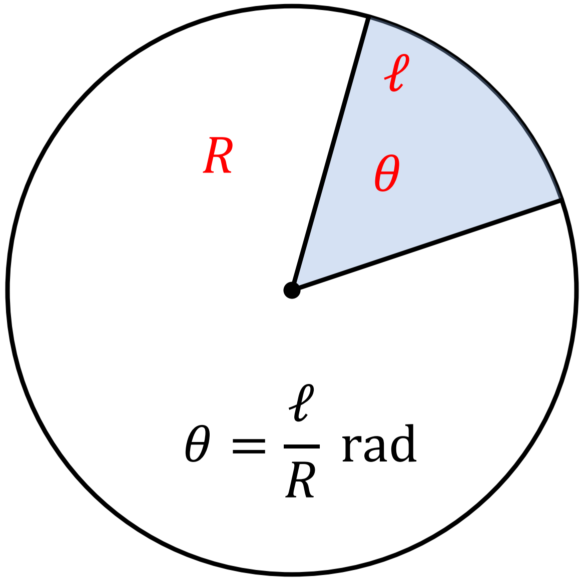
Reference: *The CODATA 2017 values of h, e, k, and N_A for the revision of the SI, Newell et al., Metrologia, 2018*

Amount of substance, moles, and counting quantities

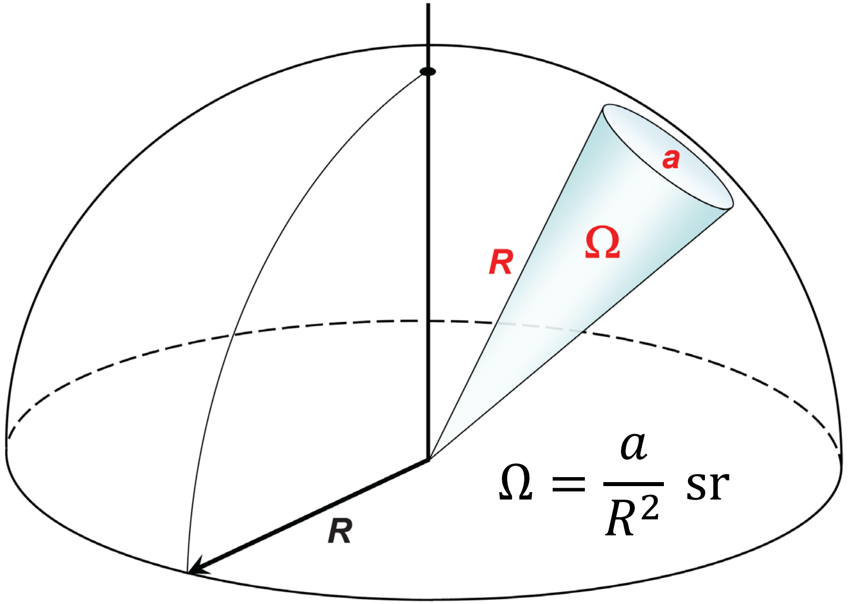


Angles in the International System of Units

Plane angle (radians)



Solid angle (steradians)



Current status of angles in the SI

- SI derived units with special names and symbols
- Radians expressed in SI as rad $= m/m = 1$ (dimensionless)
- Steradians expressed in SI as sr $= m^2/m^2 = 1$ (dimensionless)

Open question

- Should plane angles and solid angles be treated as dimensionless quantities?

Natural units and nondimensionalization

Usage of natural units

- Widely used all over theoretical physics

High-energy / particle physics

$$c = \bar{h} = \epsilon_0 = k_B = 1$$

General relativity (geometrized unit sytem)

$$c = G = 1$$

$$G_{\mu\nu} = \frac{8\pi G}{c^4} T_{\mu\nu} \quad \Rightarrow \quad G_{\mu\nu} = 8\pi T_{\mu\nu}$$

Motivations

- "Make equations easier to handle"
- "Closer to fundamental physics"
- "Easy to find constants back using dimensional analysis"

Reality check

- In some cases, not so straightforward to find constants back in practice
- Loss of information for dimensional analysis
- Some errors cannot be traced anymore by pure dimensional analysis

Conceptual collapse with natural units

length (l) ⇒ light-year

time (t) ⇒ year

speed (v) ⇒ 1 light-year per year

speed (v) = 1

dim(time) = dim(length)

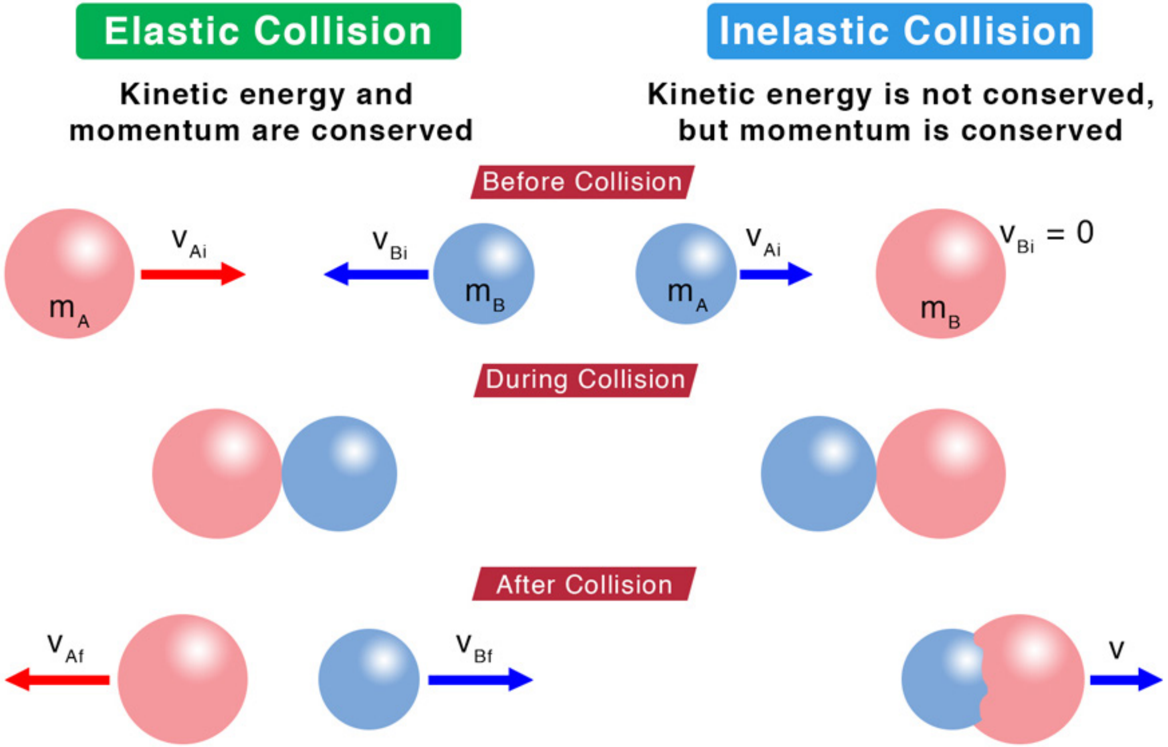
Noether's theorem

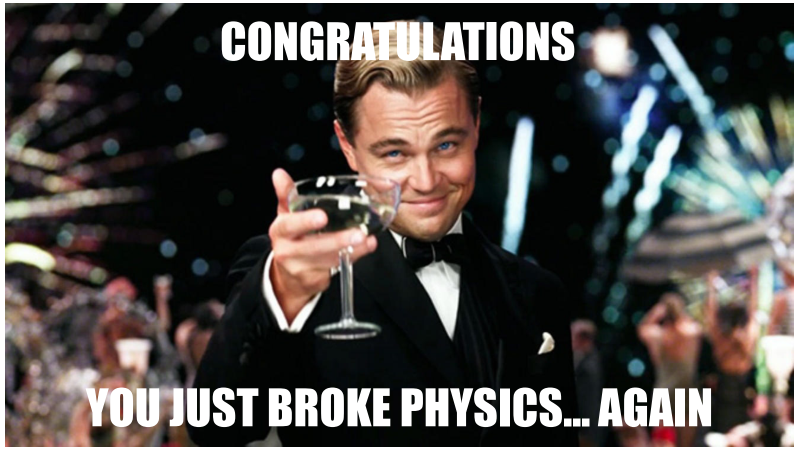
- Conservation of energy
- Conservation of momentum

dim(energy) = dim(momentum)

Collapse of usual physical concepts

- Loss of one conservation law





Reference: *The role of unit systems in expressing and testing the laws of nature, Quincey and Burrows, Metrologia, 2019*

A problem defined by its corner cases

Quantity kinds	Explicit vs implicit casting	Numerical precision	Named units
Unit simplification	Custom systems of units	Mixing systems of units	Evolution of unit systems
Variable conversion rate	Natural units	Nondimension-alization	Complete vs unit-specific equations
Angles	Affine spaces	Amount of substance	Logarithmic quantities
Vector, tensor, and complex quantities	Ordinal quantities	Uncertainties	Compile-time and runtime

Toward the mathematical formalization of units

Hypotheses

The reason why nobody uses units in code
is that all libraries are broken

⬇

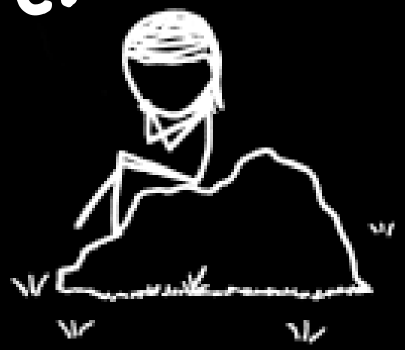
The reason why all libraries are broken
is that no one really knows what units and quantities are

⬇

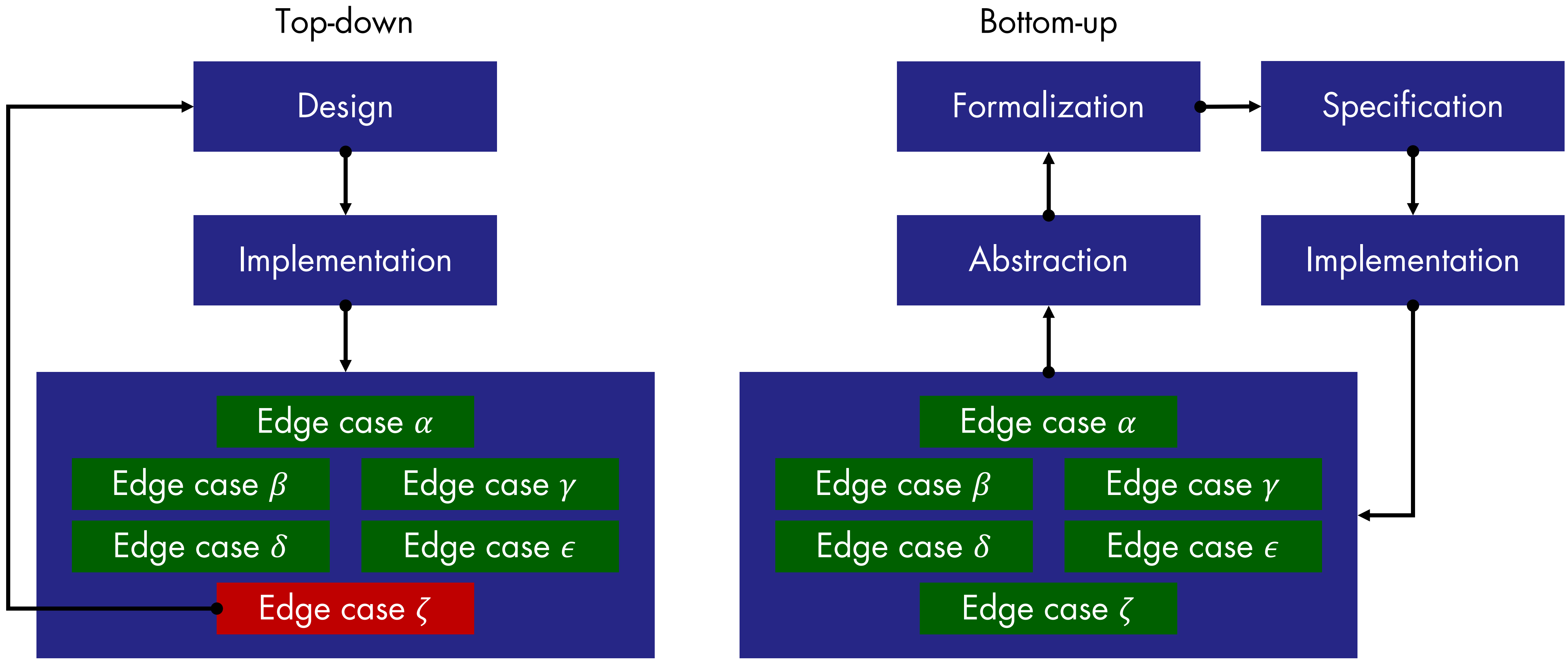
To date, there is no rigorous mathematical formalization
of the notions of units, quantities, and unit systems

In other words, the problem is that:

1. Every physicist have an intuitive notion of what a unit is
2. No mathematician knows what a unit is



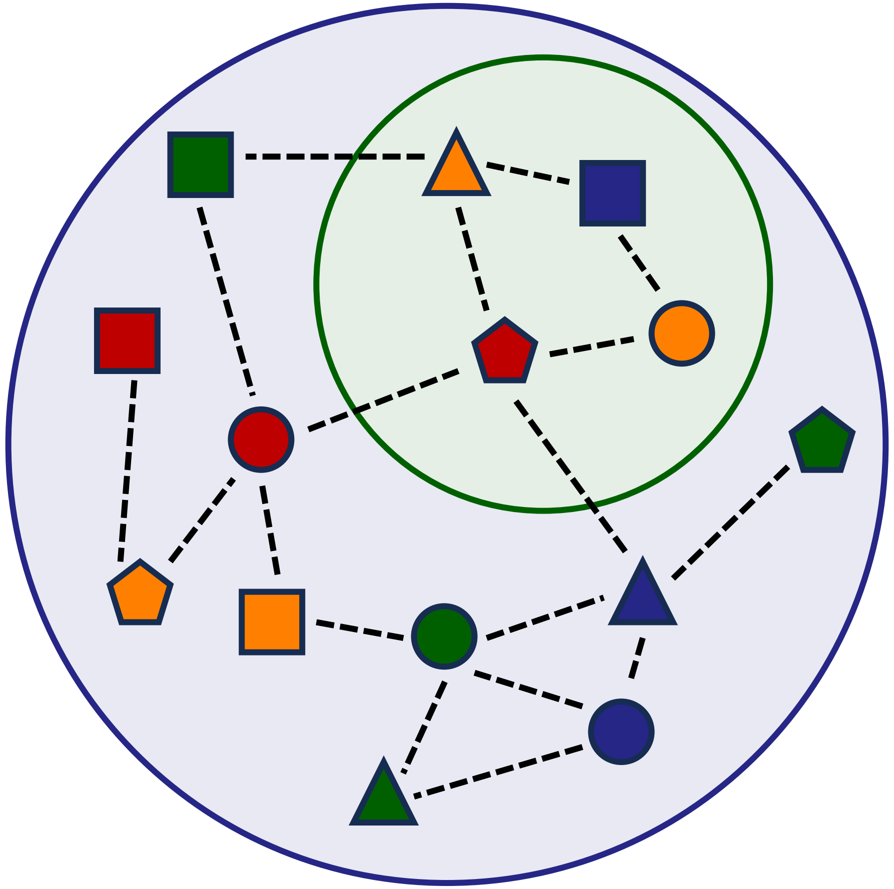
Top-down vs bottom-up approach

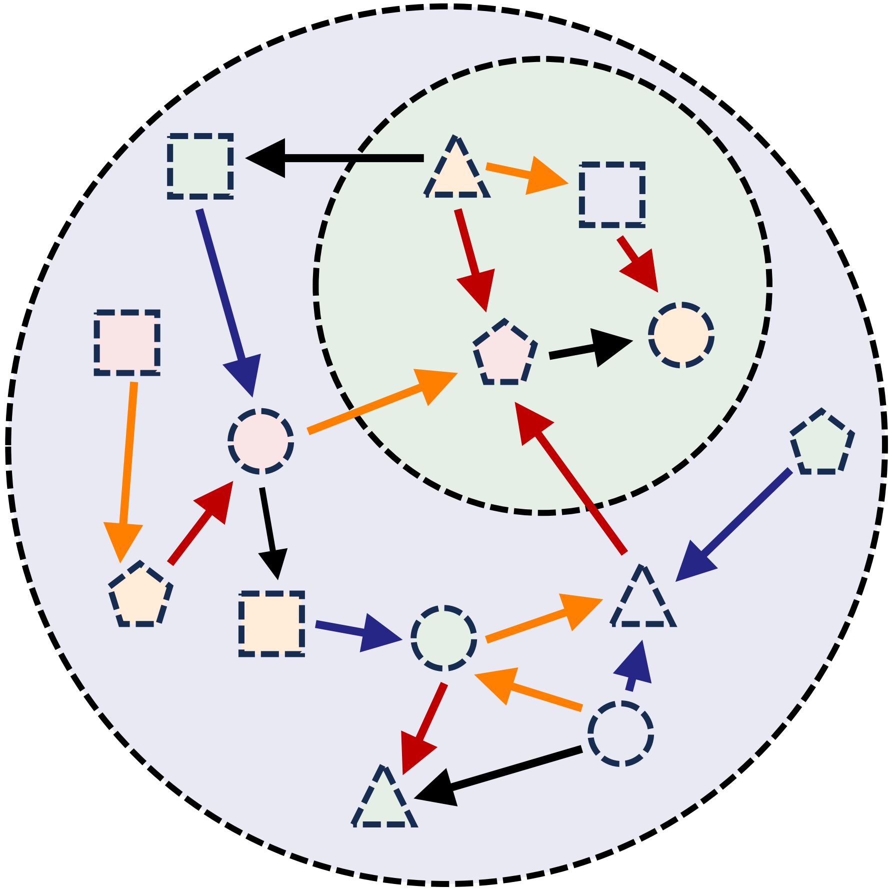


Current status of units library investigations

- Top-down approaches: do not seem to converge
- Strategy: try a more bottom-up approach through Applied Category Theory

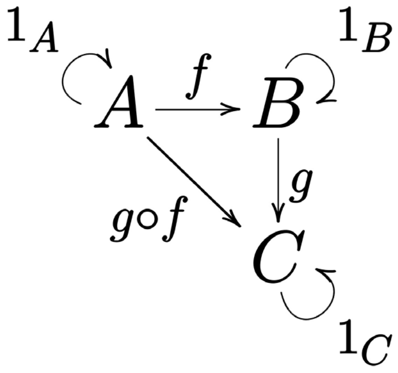
Two ways of building mathematics from scratch

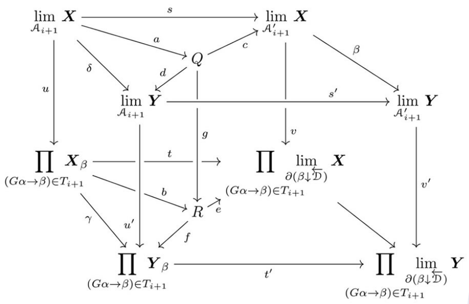


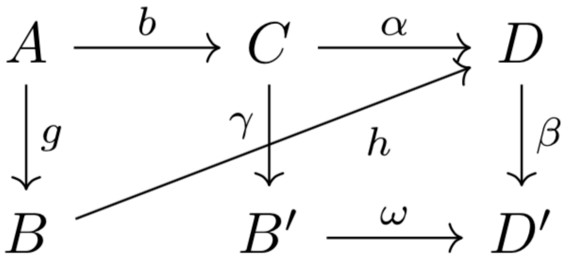


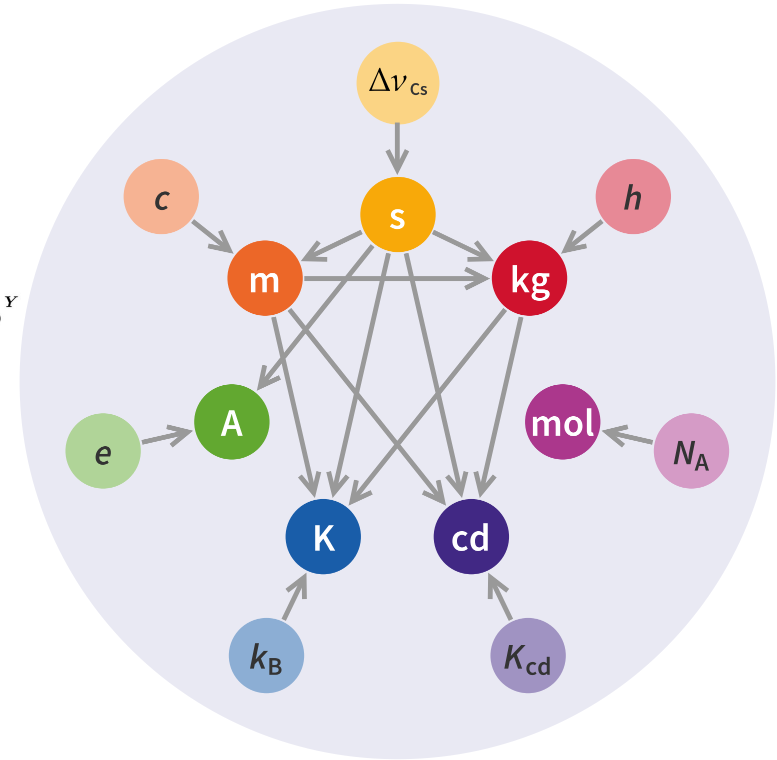
Set theory	Category theory
■ Objects are first-class citizen ■ Relationships come second ■ Objects are defined in themselves	■ Relationships are first-class citizen ■ Objects come second ■ Objects are defined through their relationships

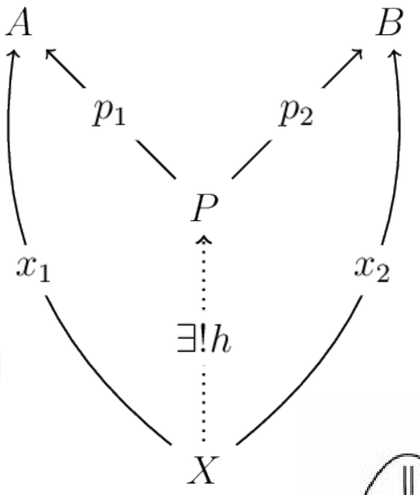
Formalization strategy

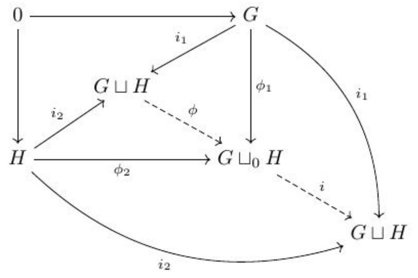


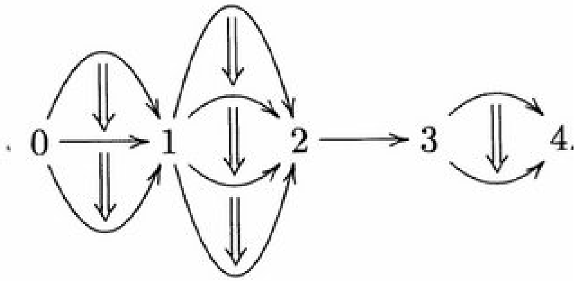








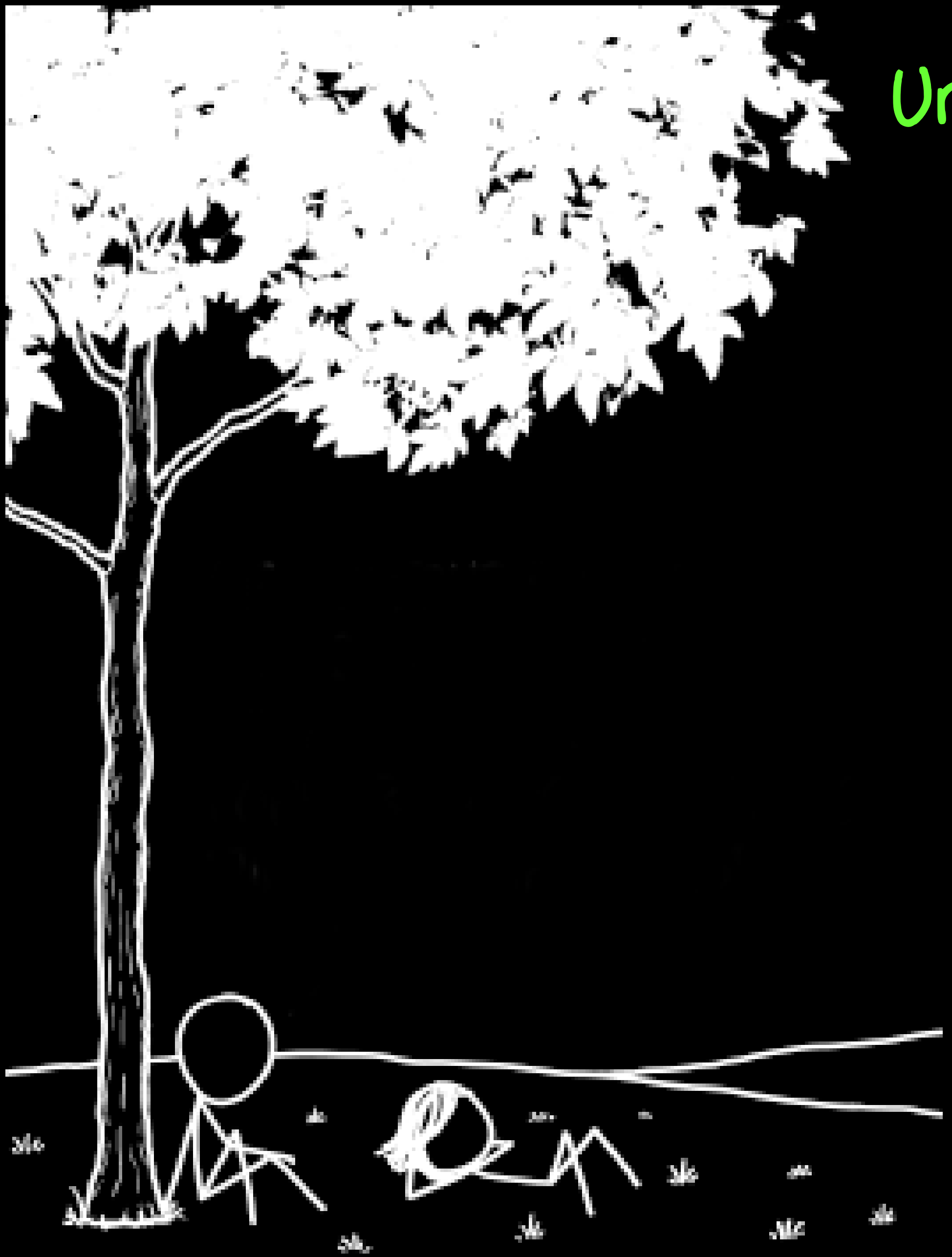




Units, quantities, and systems as categories

- Use category theory to formalize units, quantities, and systems through their relationships

Category theory (relationships) → Type theory (types and semantics) → Implementation (syntax and API)



Unit and quantity systems are worldviews

« I'm 5 min away from the building »

distance in min ?

Assuming:

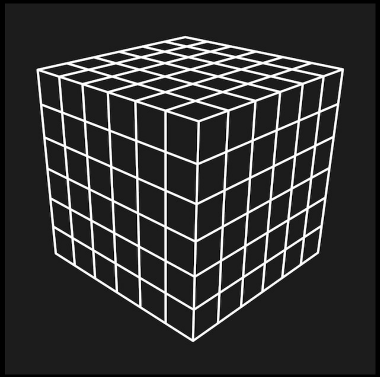
- I am driving there
- An average speed of 50 km/h
- No road closed
- No accident…

Examples in physics:

- Assuming Newton's laws or General relativity?
- Assuming classical physics or quantum mechanics?
- …

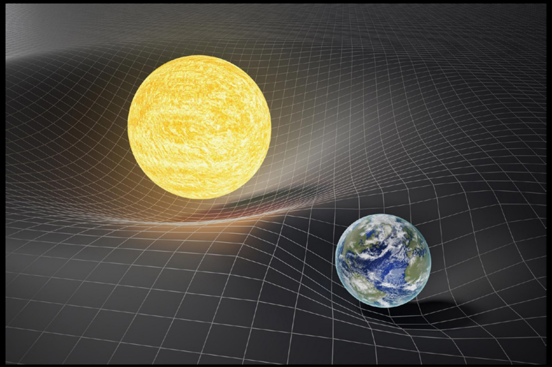
Formalizing transformations between unit systems $\Longleftrightarrow$ Formalizing transformations between worldviews

Newton



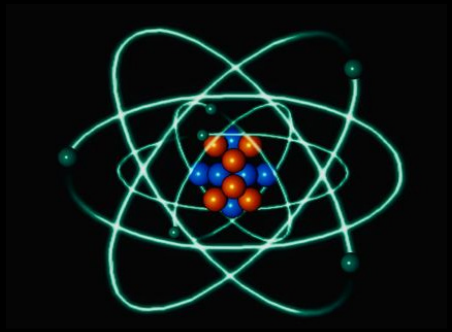
$\Longleftarrow$

$\Longrightarrow$



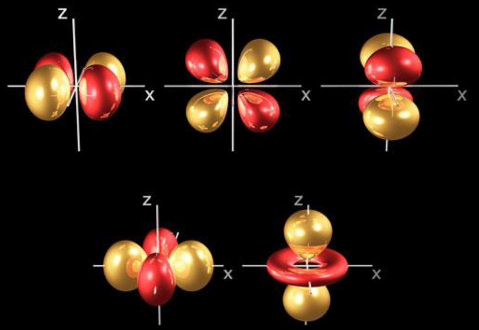
Einstein

Classical physics



$\Longleftarrow$

$\Longrightarrow$

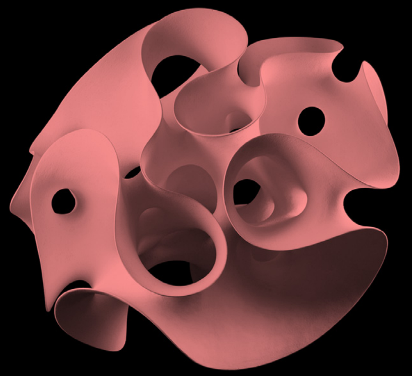


Quantum mechanics

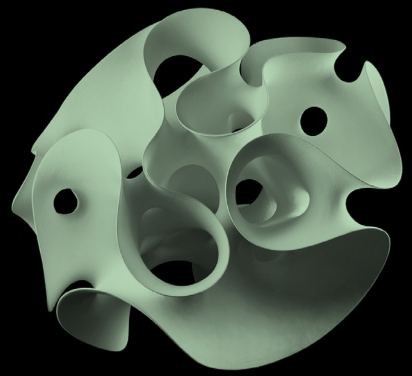
Key to the problem

Formalization of the notion of unit/quantity systems
and transformation between systems

What is the fundamental nature of « ⇒ » ?

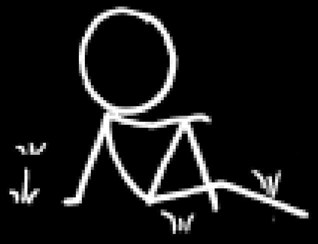


⇛

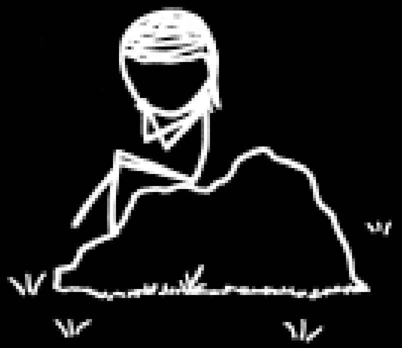


Structure of physics

Structure of (programming) languages



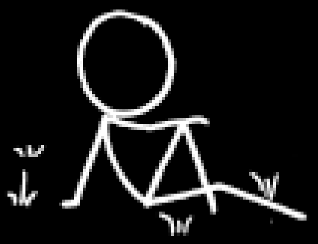
→ morphism between units/quantities
⇒ functors between systems/worldviews
⇛ natural transformations between physics & languages



Summary and concluding remarks

Conclusions

1. Units are complicated
2. Nobody uses units in code because all libraries are broken
3. Mathematicians do not know what units really are
4. Units and quantities systems can be seen as worldviews
5. Project to formalize unit systems using category theory
6. Weird, unusual, limit cases are welcome!



Thanks for listening!
Questions?