

# Introduction to Monte Carlo methods and its applications to Particle Physics

---

Ph.D. school in Physics

23, 25, 28, and 30 September 2020

Roma Tre

Carlo Mancini Terracciano

[carlo.mancini.terracciano@roma1.infn.it](mailto:carlo.mancini.terracciano@roma1.infn.it)

# Overview

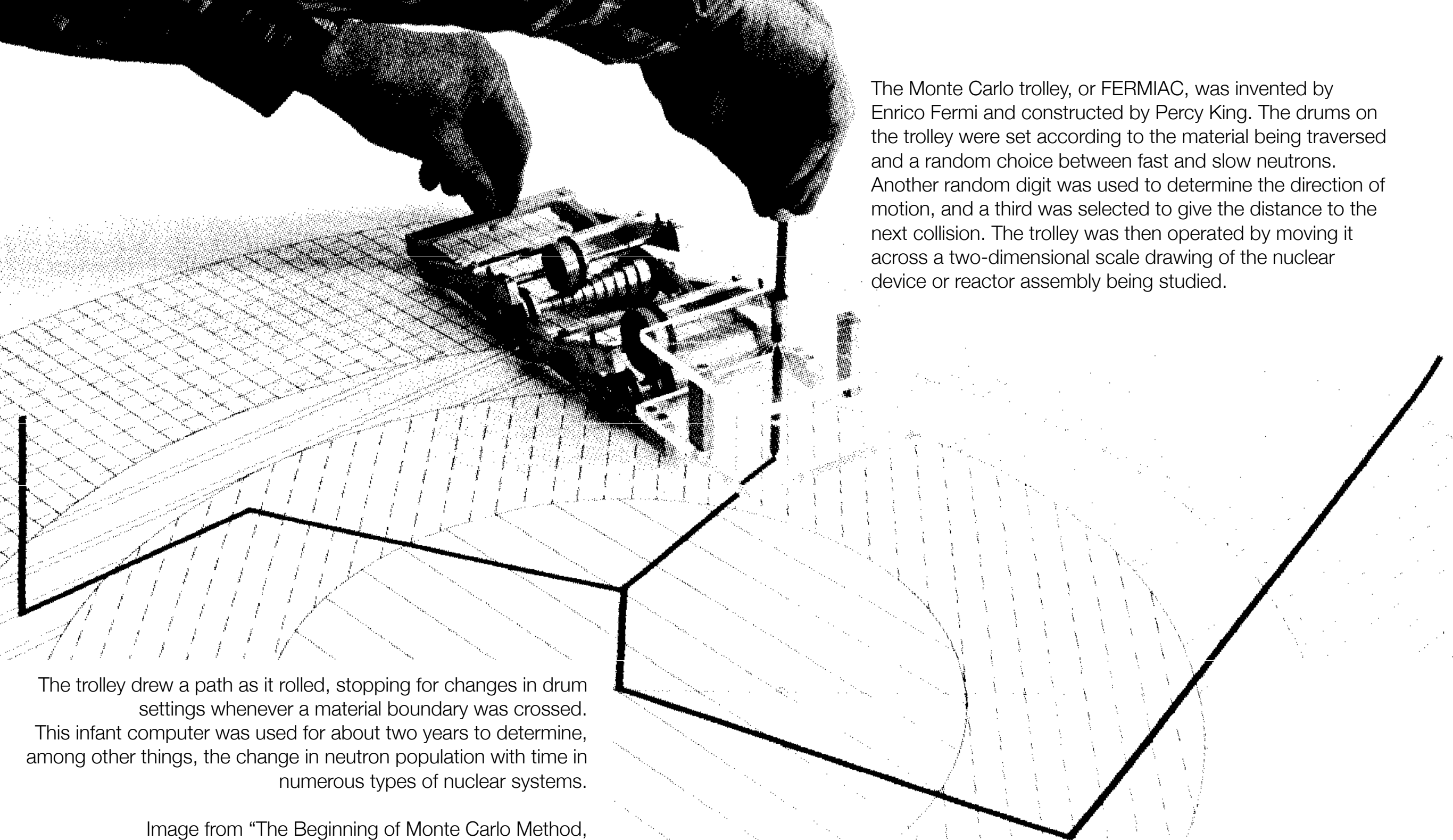
---

- The Monte Carlo method
- How to install Geant4
- Some basic features of C++
- A short introduction to Geant4

For these slides I took inspiration from:

- M. Asai (SLAC, Stanford)
- A. Dotti (SLAC, Stanford)
- S. Incerti (CNRS, Bordeaux)
- L. Pandola (INFN-LNS, Catania)
- C. Pistillo (LHEP, Bern)
- S. Rahatlou (Sapienza, Roma)
- [www.cplusplus.com](http://www.cplusplus.com)

- You can download examples and slides from:  
<http://www.roma1.infn.it/~mancinit/Teaching/PhD-RomaTre/>



The Monte Carlo trolley, or FERMIAC, was invented by Enrico Fermi and constructed by Percy King. The drums on the trolley were set according to the material being traversed and a random choice between fast and slow neutrons. Another random digit was used to determine the direction of motion, and a third was selected to give the distance to the next collision. The trolley was then operated by moving it across a two-dimensional scale drawing of the nuclear device or reactor assembly being studied.

The trolley drew a path as it rolled, stopping for changes in drum settings whenever a material boundary was crossed. This infant computer was used for about two years to determine, among other things, the change in neutron population with time in numerous types of nuclear systems.

Image from "The Beginning of Monte Carlo Method,  
N. Metropolis 1987

# Introduction to the Monte Carlo method

The "FERMIAC"

# Monte Carlo methods

---

- It is a mathematical approach using a sequence of random numbers to solve a problem
- Stochastic quantities (e.g.: average value of *mm* of rain)
- Deterministic problems (definite integral)
- Generate  $N$  random “points”  $\vec{x}_i$  in the problem space
- Calculate  $\langle f \rangle = \frac{1}{N} \sum_{i=1}^N f(\vec{x}_i)$  and  $\langle f^2 \rangle = \frac{1}{N} \sum_{i=1}^N f^2(\vec{x}_i)$

# Monte Carlo methods

---

- Comte de Buffon (1777): needle tossing experiment to calculate the  $\pi$ ;
- Laplace (1886): random points in a rectangle to calculate  $\pi$ ;
- Fermi (1930): random approach to calculate the properties of the newly discovered neutron;
- Manhattan project (40's): simulations during the initial developments of thermonuclear weapons;
- Von Neumann and Ulam coined the term 'Monte Carlo' (1949);
- Exponential growth of the electronic computers (40's-60's);
- Berger (1963): first complete coupled electron-photon transportation code 'ETRAN'.

# How to calculate an integral

---

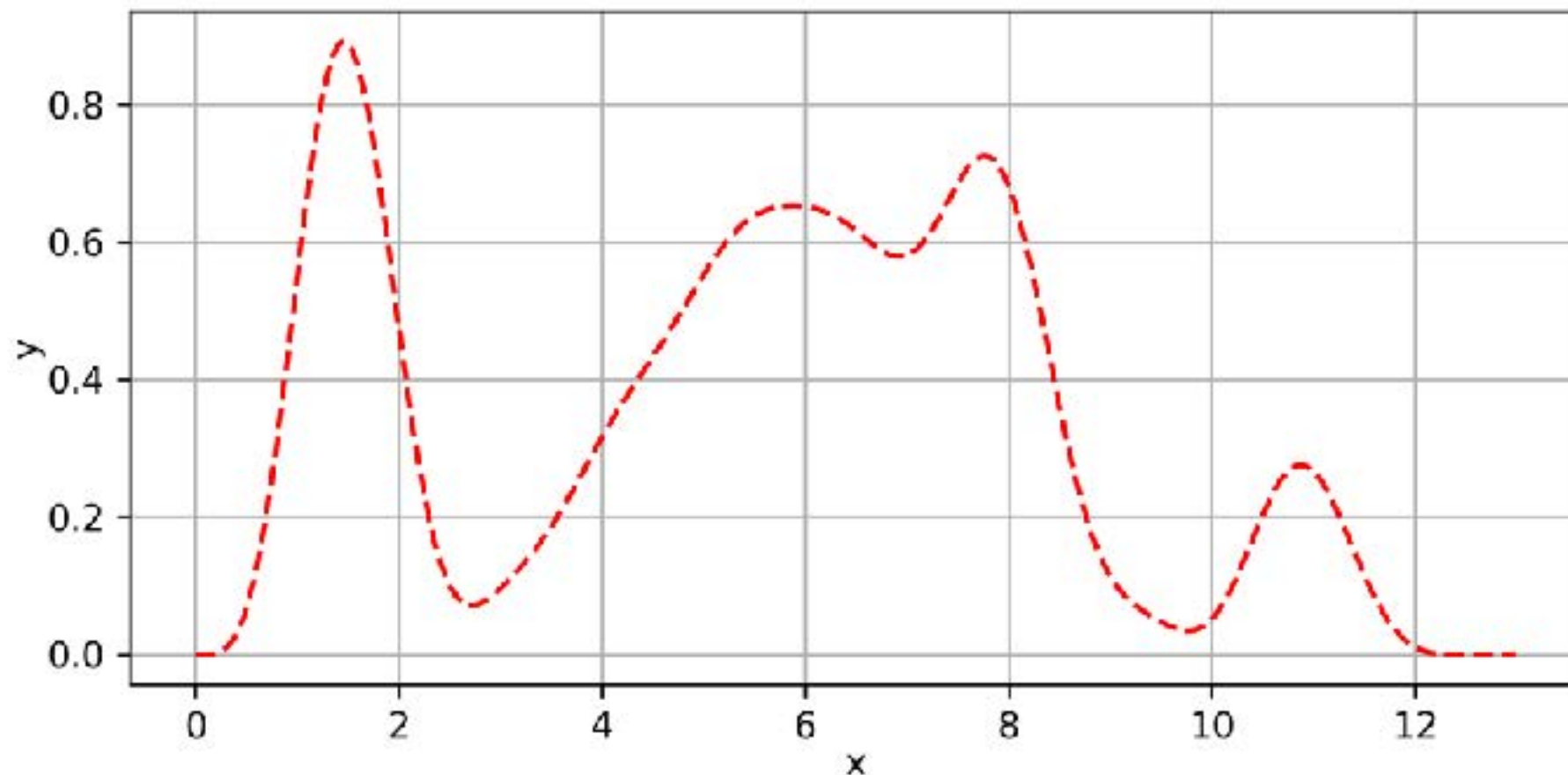
- Randomly choose couple of numbers  $(x_i, y_i)$  from the range and the domain, respectively, of the function  $f$
- The fraction of points where  $y_i \leq f(x_i)$  is equal to the fraction of the area below the function
- Technique proposed by Von Neumann, known as the “acceptance-rejection method”
- It is used to generate random numbers for an arbitrary Probability Density Function (PDF)

# Example of an integral

---

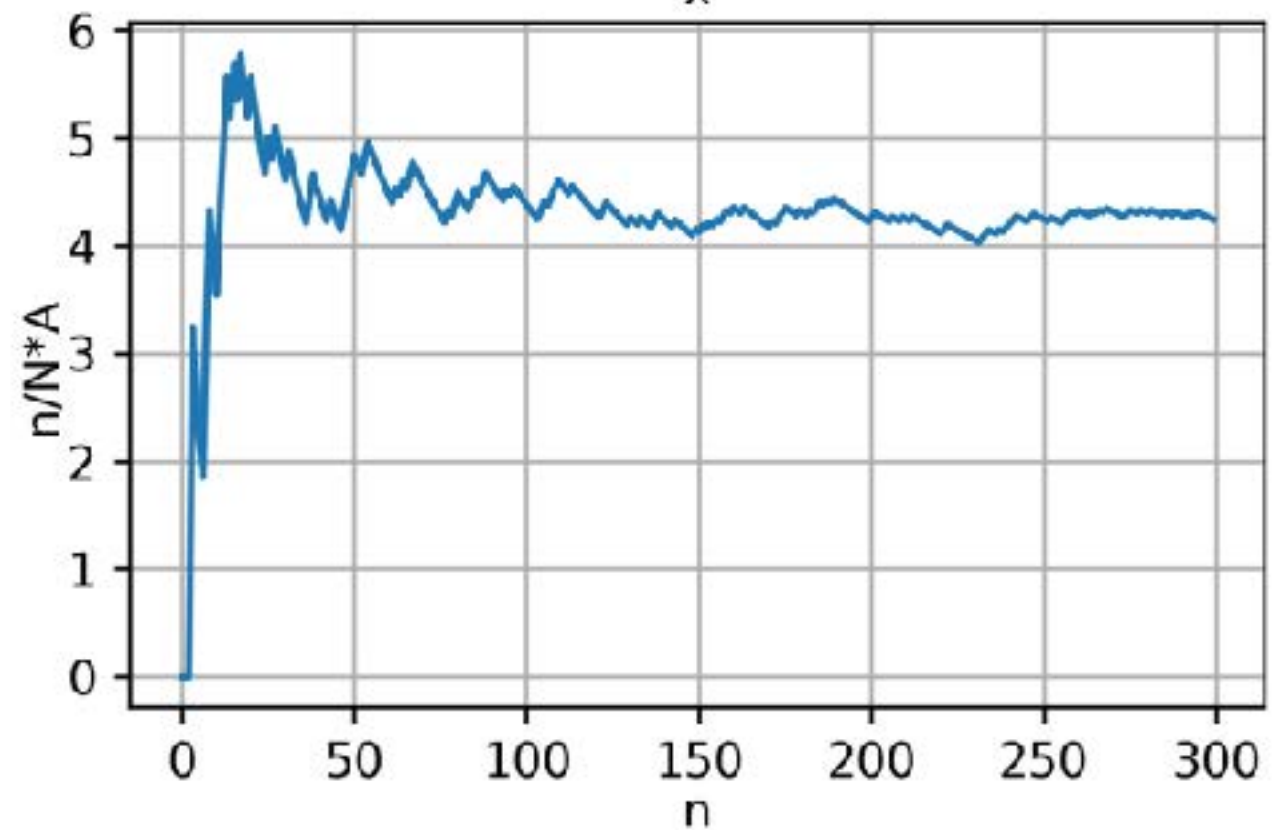
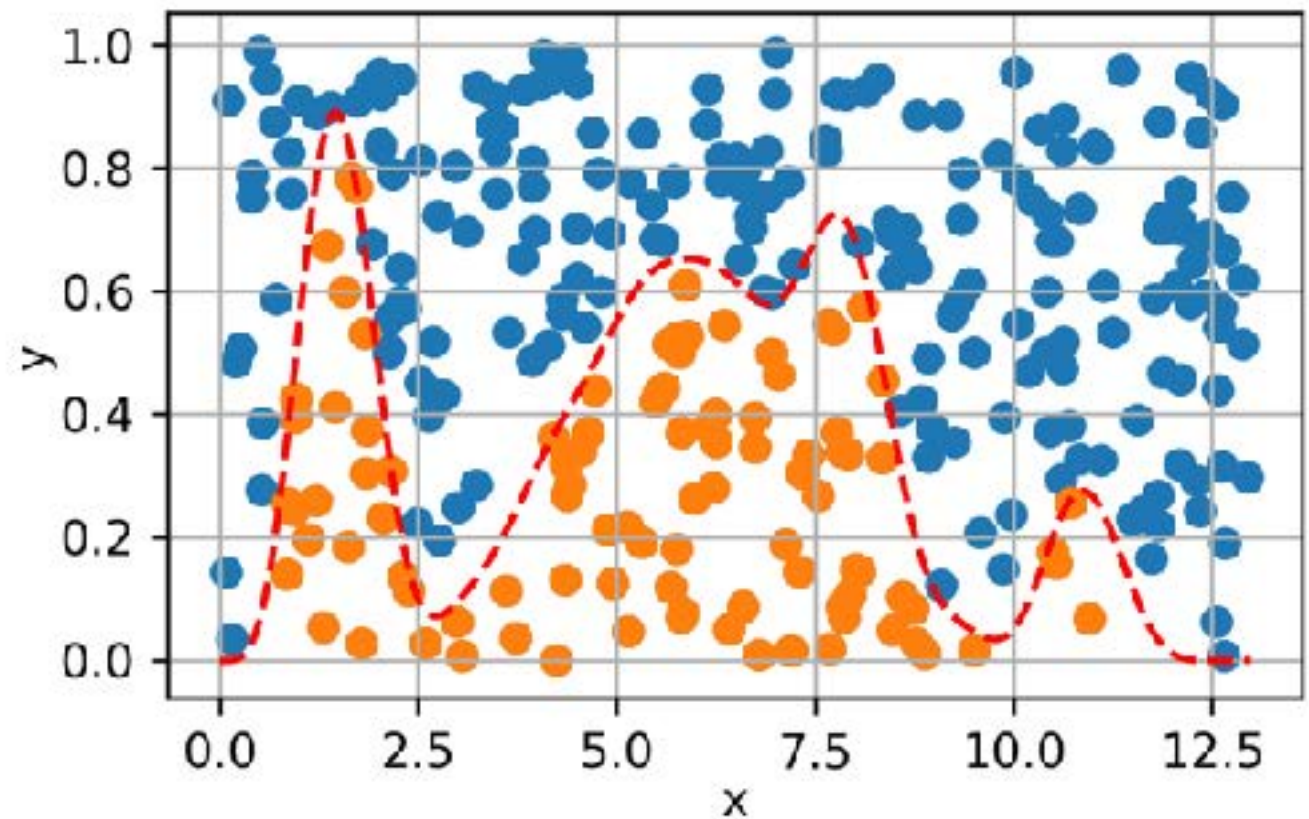
- What if you have to calculate the integral of a function as:

$$f(x) = 1.4 \left[ \sin^4(x) \cos^2\left(\frac{x}{3}\right) + \sin^6\left(\frac{x}{4}\right) \right] \exp\left(-\frac{x}{8}\right)$$



# Example of an integral

- Using the acceptance-rejection method
- The orange points are the accepted ( $n$ )
- The blue are the rejected ( $N = \text{accepted} + \text{rejected}$ )





# Let's calculate $\pi$

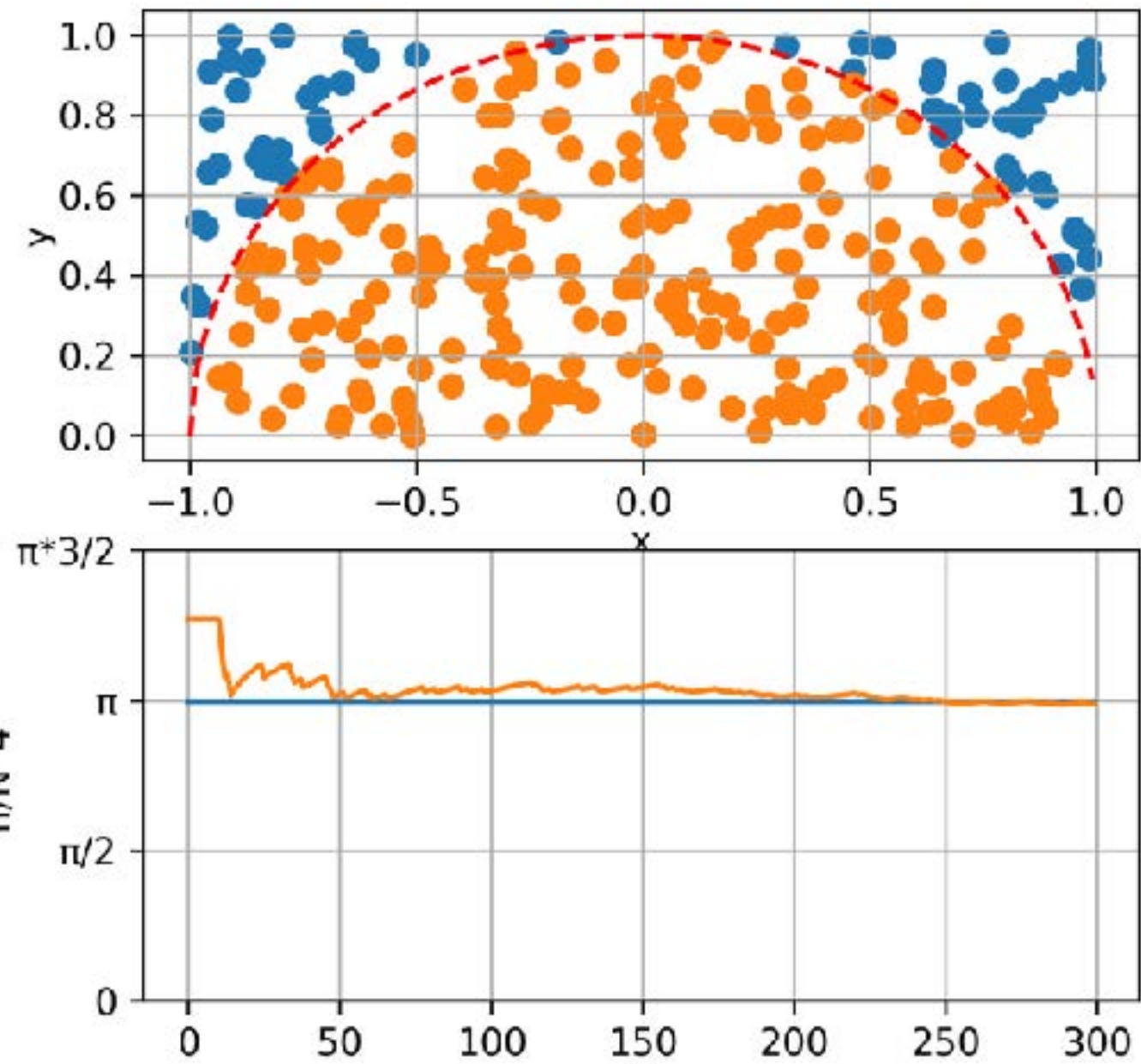
$$f(x) = \sqrt{(1 - x^2)}$$

$$A_{circ} = \int_{-1}^1 f(x) dx = \pi \frac{r^2}{2}$$

$$A_{rect} = \Delta y \cdot \Delta x$$

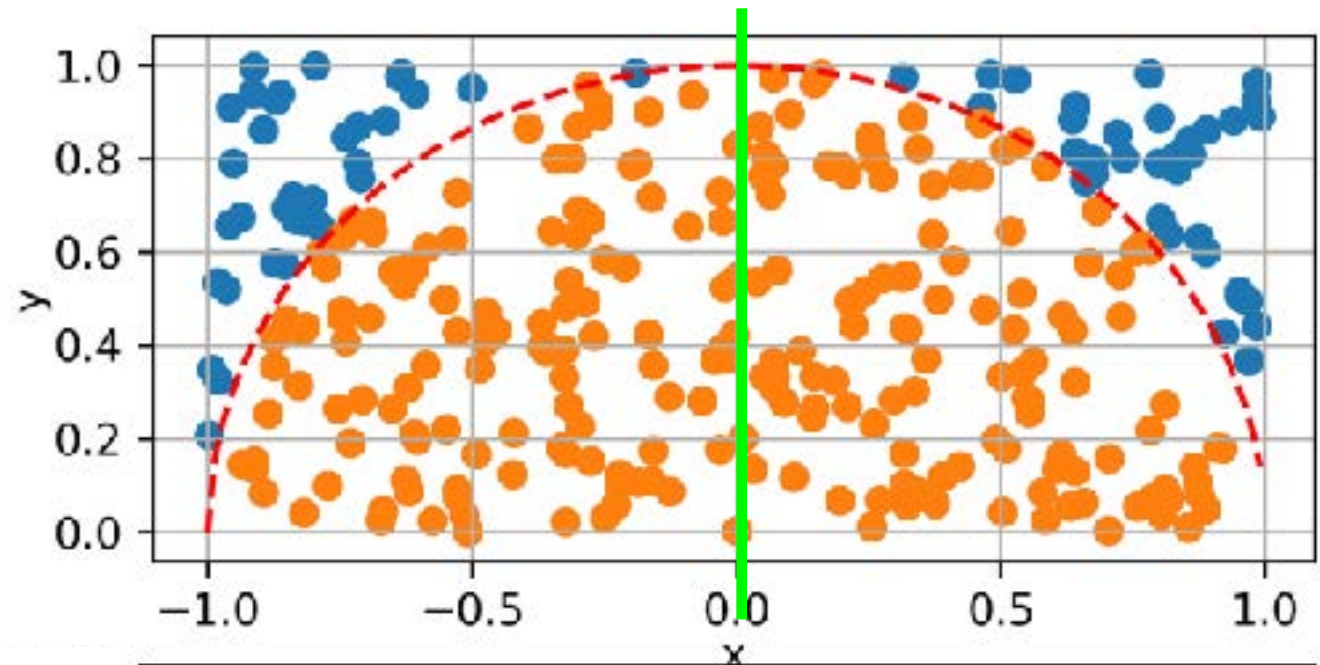
$$\frac{n}{N} \propto \frac{A_{circ}}{A_{rect}} = \frac{\pi r^2 / 2}{\Delta x \cdot \Delta y}$$

$$\pi \approx 2 \frac{n}{N} \frac{\Delta x \cdot \Delta y}{r^2} = 4 \frac{n}{N}$$



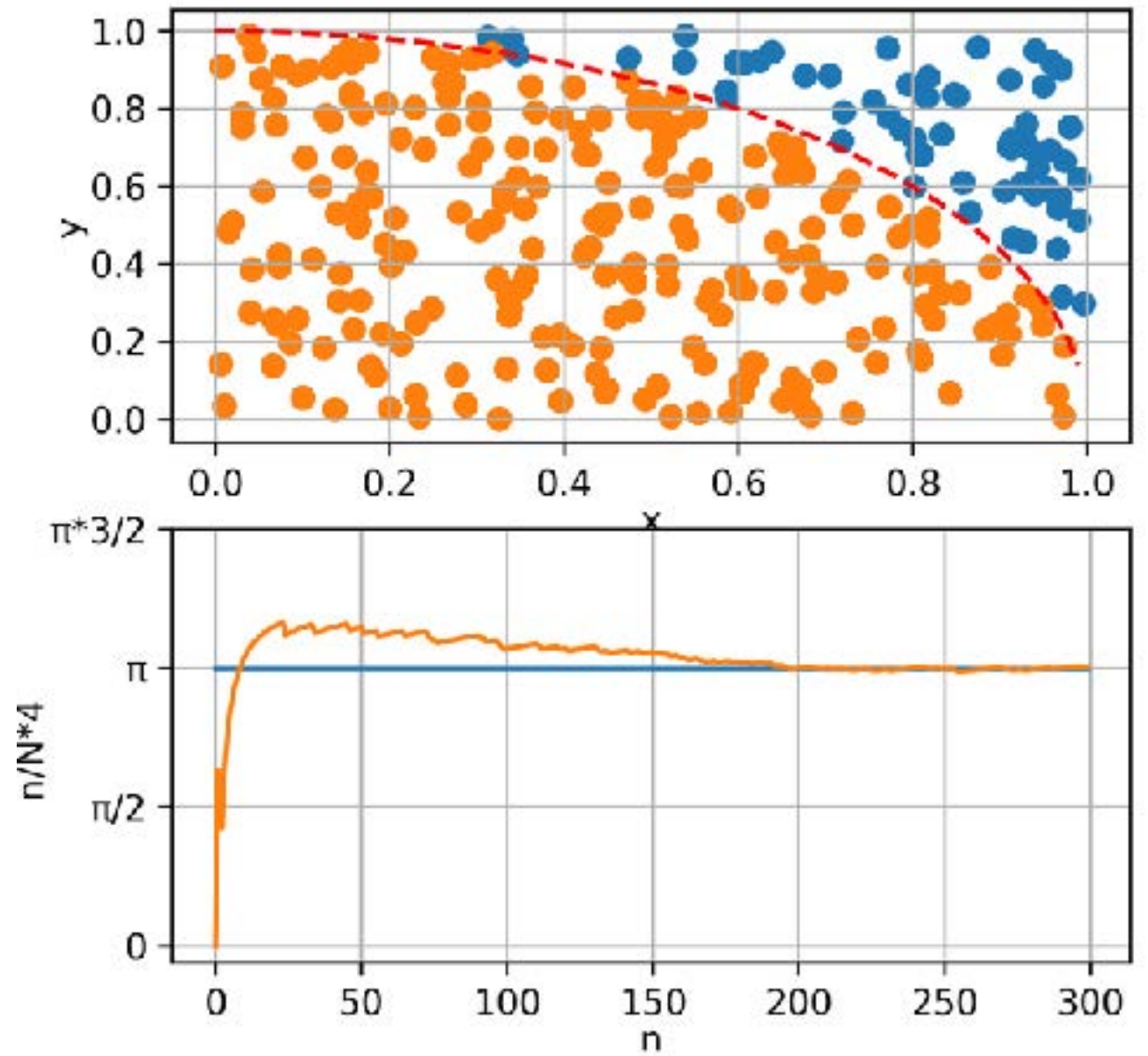
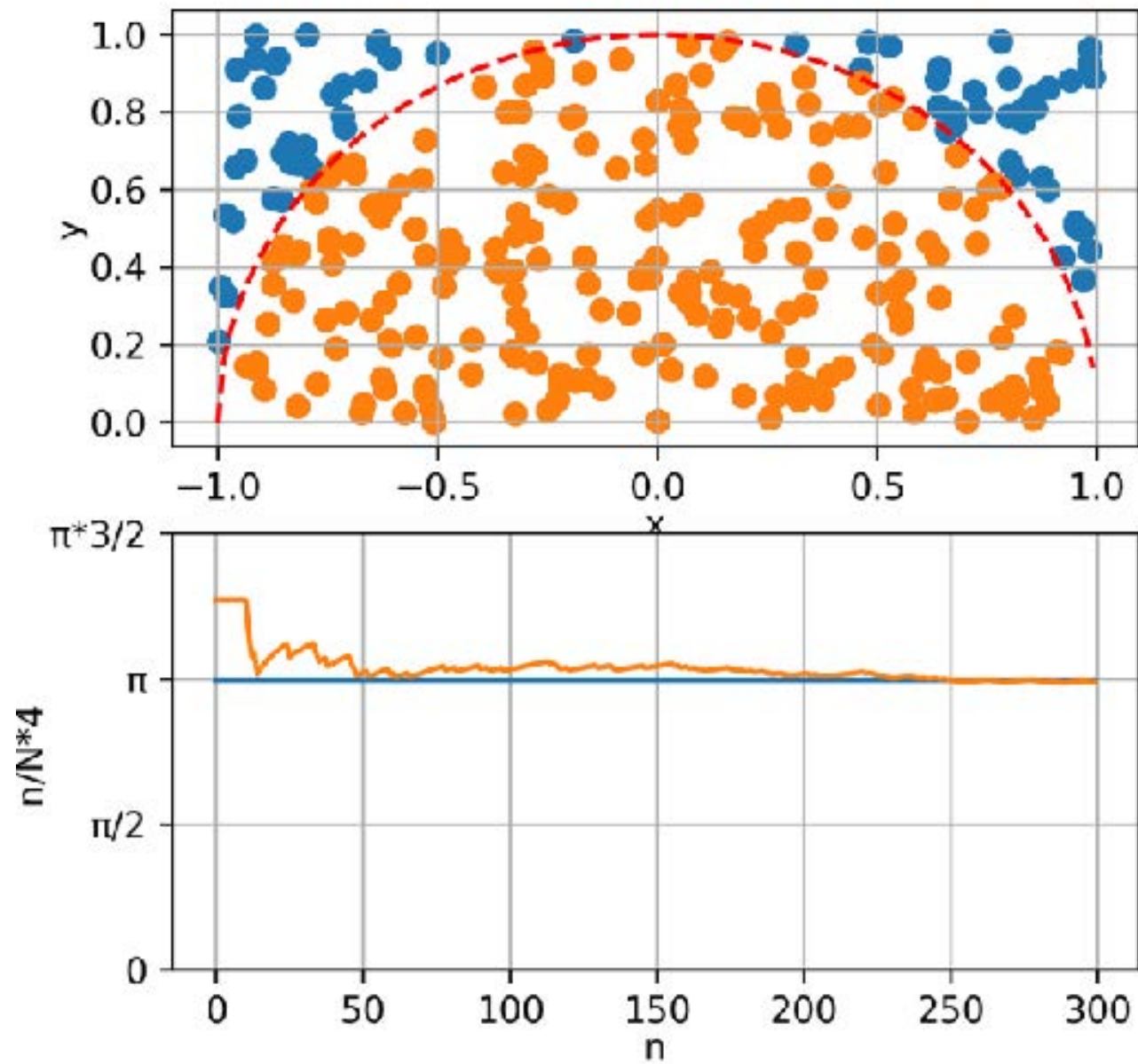
# Let's calculate $\pi$

- Is there a way to speed up the convergence of the computation?
- Use the symmetry!
- This is the method for calculating  $\pi$  was proposed by Laplace in “Théorie Analytique des Probabilités” (1825)!





# Use the symmetry!



# Random Numbers Generators

---

- At the core of all Monte Carlo calculations is some mechanism to produce a long sequence of random numbers  $r_i$  that are uniformly distributed over the open interval  $[0, 1)$
- Digital computers, by design, are incapable of producing random results
- A true random sequence could, in principle, be obtained by coupling to our computer some external device that would produce a truly random signal
- However, use of such a random number generator would not be practical!
- Impossible to debug!



# Pseudo-random Number Generators

---

- Such a generator is a deterministic algorithm that, given the previous numbers (usually just the last number) in the sequence, the next number can be efficiently calculated

$$x_{n+1} = f(x_n, x_{n-1}, \dots, x_0)$$

- $x_0$  is called “**seed**”
- A unique seed returns a unique random number sequence
- It is important to use a **new seed every time** that a random selection is initiated
- A typical error is the use of the same seed for multiple generation, which leads to the generation of the same sample of random numbers

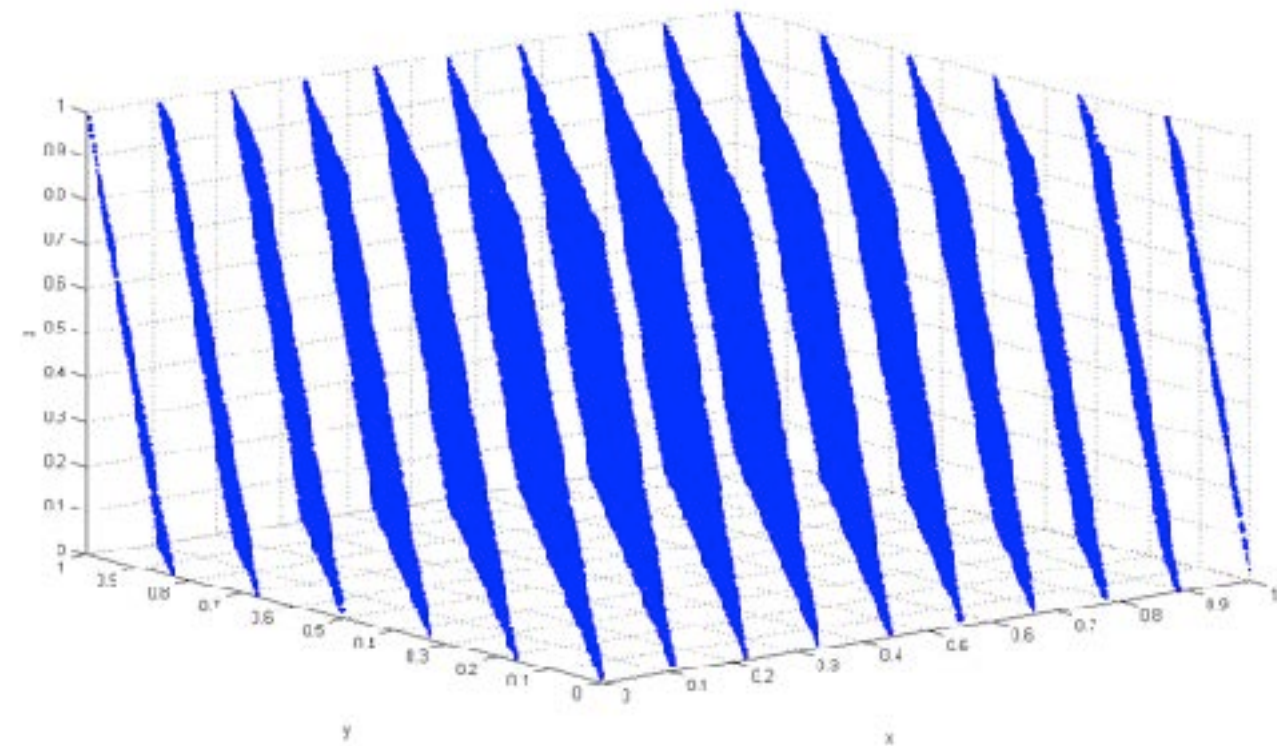
# Pseudo-random Number Generators

---

- Because the set of numbers directly representable in the computer is finite, the sequence will necessarily repeat
- The length of the sequence prior to beginning to repeat is called **period**
- Many pseudo-random number generators have been proposed and used over the years in a wide variety of Monte Carlo work.
- Designing better random number generators (and test them) is still an active area of research

# Bad PRNG: RANDU

- Linear congruential PRNG used since '60
- Was the most widely used random number generator in the world
- Developed by IBM
- Three-dimensional plot of 100,000 values generated with RANDU
- Each point represents 3 consecutive pseudorandom values



[Image from Wikipedia]

$$x_i = 65539 \cdot x_{i-1} \bmod (2^{31})$$

*not only the period is important!*



# Simple case: decay in flight

---

- Suppose a  $\pi^+$  with momentum  $p$
- The life time is a random value with a pdf

$$f(t) = \frac{1}{\tau} \exp\left(-\frac{t}{\tau}\right)$$

- Therefore,  $t$  can be sampled from the inverse of the cumulative:

$$t = F^{-1}(r) = -\tau \ln(1 - r)$$

$$r \in [0, 1)$$



# Simple case: decay in flight

- Select the decay channel:

Mode		Fraction ( $\Gamma_i / \Gamma$ )	
$\Gamma_1$	$\mu^+ \nu_\mu$	[1]	$(99.98770 \pm 0.00004)\%$
$\Gamma_2$	$\mu^+ \nu_\mu \gamma$	[2]	$(2.00 \pm 0.25) \times 10^{-4}$
$\Gamma_3$	$e^+ \nu_e$	[1]	$(1.230 \pm 0.004) \times 10^{-4}$
$\Gamma_4$	$e^+ \nu_e \gamma$	[2]	$(7.39 \pm 0.05) \times 10^{-7}$
$\Gamma_5$	$e^+ \nu_e \pi^0$		$(1.036 \pm 0.006) \times 10^{-8}$
$\Gamma_6$	$e^+ \nu_e e^+ e^-$		$(3.2 \pm 0.5) \times 10^{-9}$
$\Gamma_7$	$e^+ \nu_e \nu \bar{\nu}$		$< 5 \times 10^{-6}$

[table from PDG]

- In the CM frame the decay is isotropic

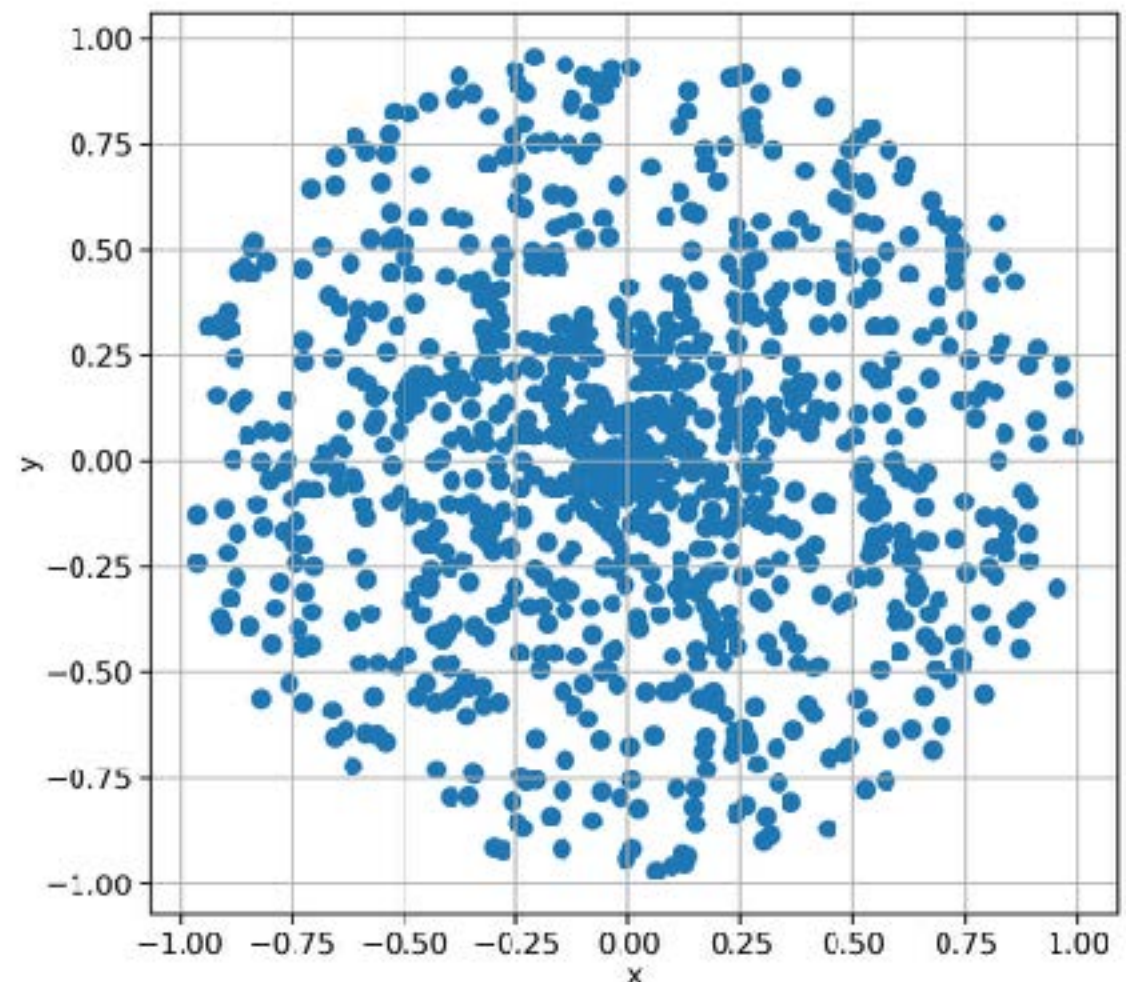
$$\theta \in [0, \pi); \quad \phi \in [0, 2\pi)$$

- Finally, Lorentz-boost in the Lab. frame
- 4 random numbers for one decay!

# Problem

---

- Why did I sampled  $\theta$  and  $\phi$  in the CM frame?
- What if I sample uniformly  $\theta \in [0, 2\pi)$ ;  $r \in [0, 1)$  ?
- The extracted points gather in the centre
- A uniform distribution in polar coordinates is not uniform in orthogonal coordinate system



# Inverse transform sampling

---

- If a PDF  $f$  is integrable (called **cumulative**,  $F$ )
- and the cumulative is invertible  $F^{-1}$
- It is possible to sample  $x$  accordingly to  $f$ :

$$x = F^{-1}(u)$$

where  $u$  is uniformly distributed

# Particle tracking

---

- It is the most common application of MC in Particle Physics
- Assume that all the possible interactions are known
- The distance  $s$  between two subsequent interactions is distributed as  $p(s) = \mu \exp(-\mu s)$
- Being  $\mu$  a property of the medium

# Particle tracking

---

- $\mu$  is proportional to the probability of an interaction per unit length, therefore:
- is proportional to the **total cross section**
$$\mu = N\sigma = N \sum_i \sigma_i = \sum_i \mu_i$$
- $\mu_i$  are the partial cross section of **all the competing processes**
- depends on the **density** of the material  
( $N$  is the number of scattering centres in the medium)

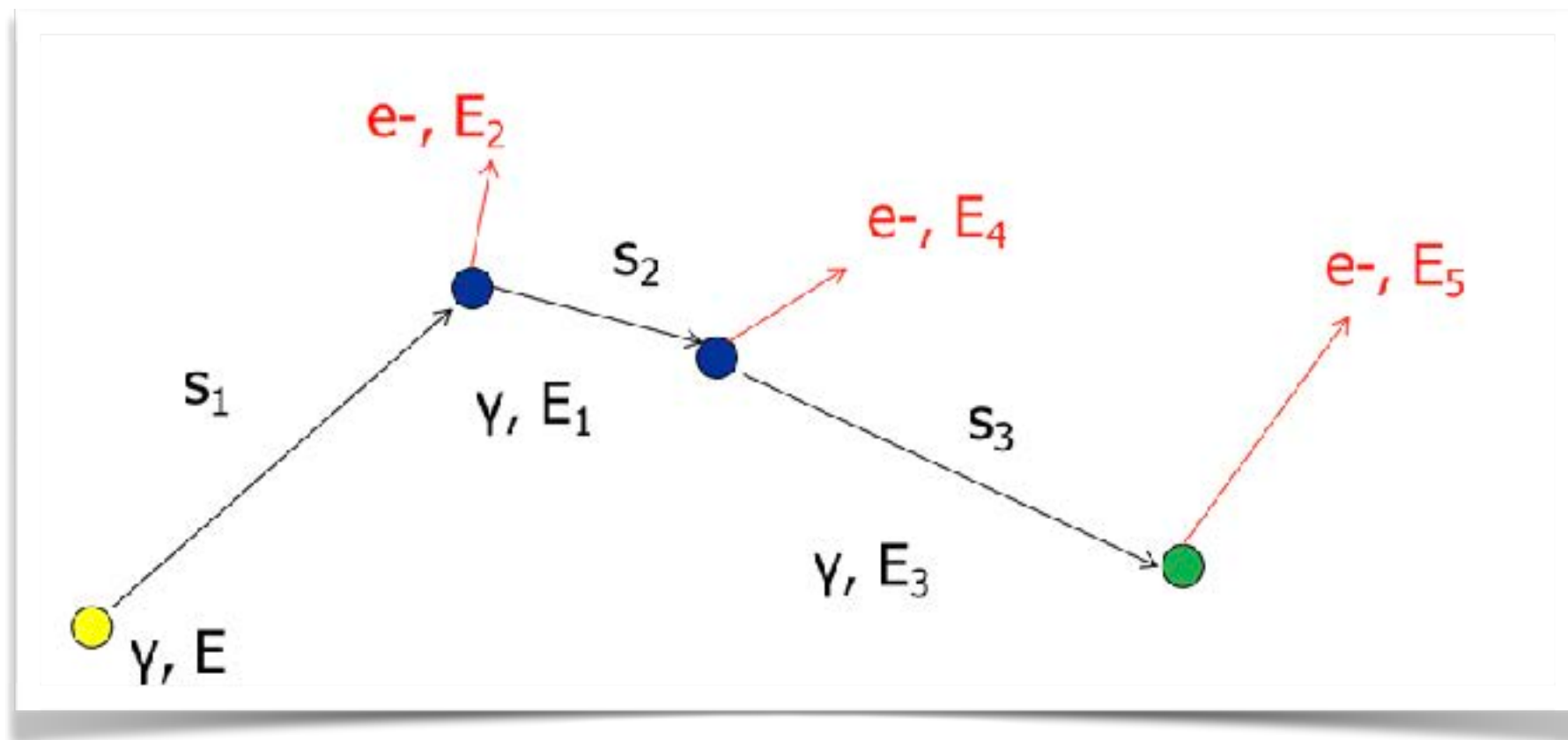
# Particle tracking

---

- Divide the particle trajectory in “**steps**”
  - Straight free-flight tracks along the step
  - Could be limited by geometry boundaries
- Sampling the step length accordingly to  $p(s)$
- Sampling the interaction at the end of the step
- Sampling the interaction accordingly to  $\mu_i/\mu$
- Sampling the final state using the physics model of the interaction  $i$ 
  - Update the properties of the primary particle
  - Add the possible secondaries produced (to be tracked later)

# Particle tracking

- Follow all secondaries, until absorbed (or leave the geometry)
- $\mu$  depends on the energy (cross sections do!)



# Tracking, not so easy...

---

- This basic recipe doesn't work well for charged particles
- The **cross sections** of some processes (ionisation and bremsstrahlung) **is very high**, so the **steps** would be very **small**
- In each interaction **only a small fraction of energy is lost** and the effect on the particle are small
- A lot of CPU time used to simulate many interactions having small effects

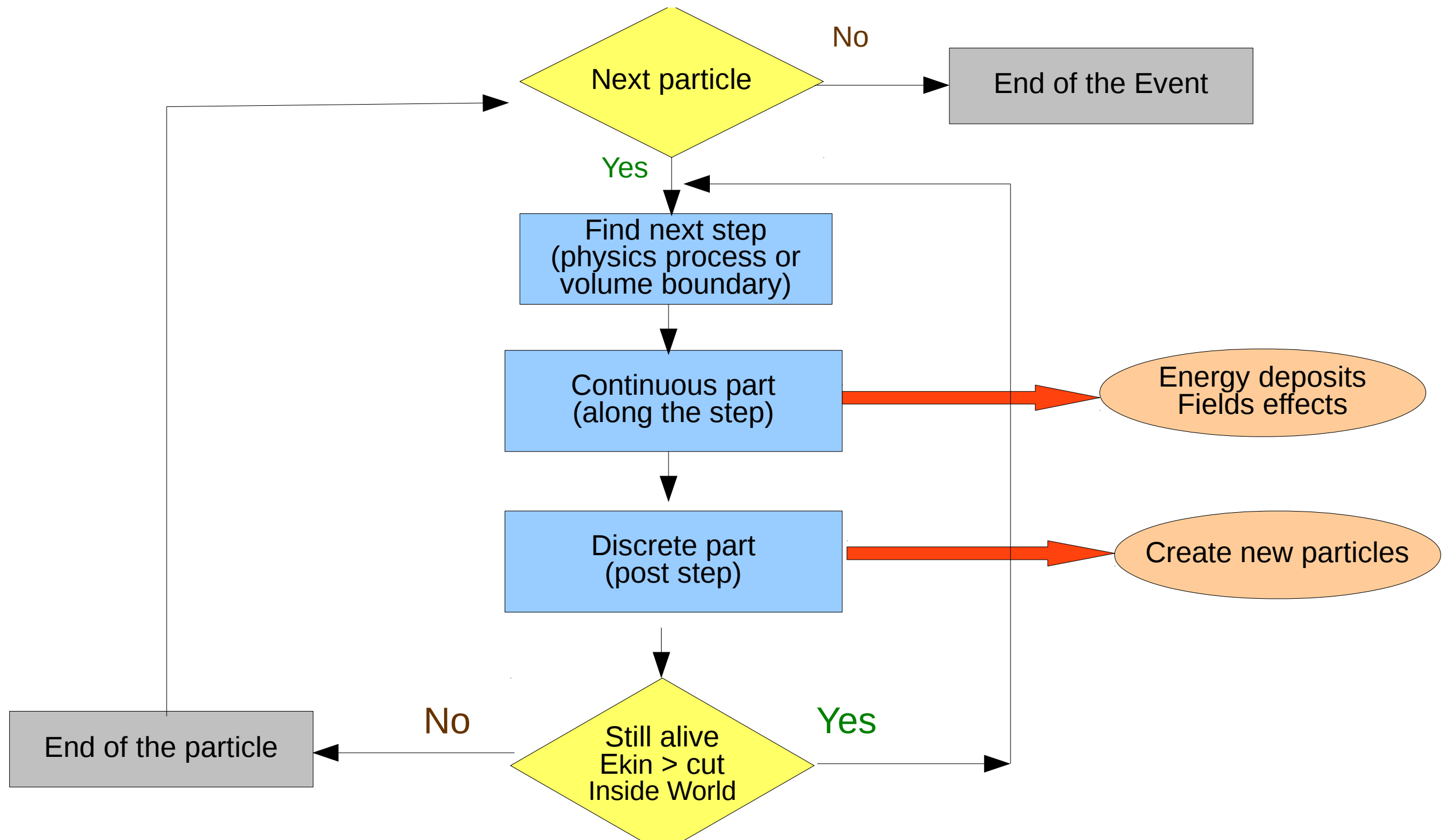


# The solution: approximate

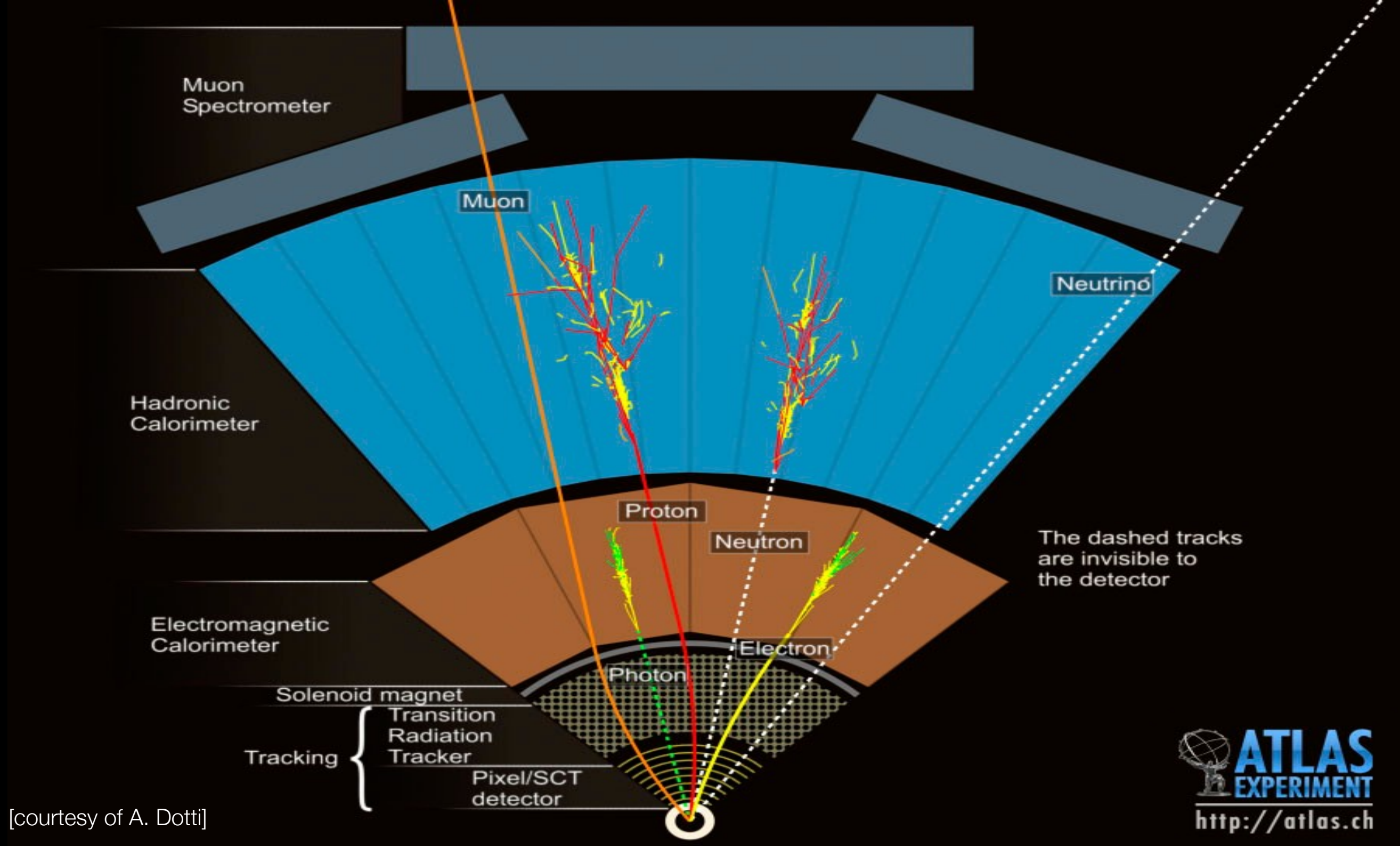
---

- Simulate explicitly interactions only if the energy loss is above a threshold  $E_0$  (**hard** interactions)
  - Detailed simulation
- The effects of all sub-threshold interactions is described cumulatively (**soft** interactions)
- Hard interactions occur much less frequently than soft interactions

# Flowchart of an event



...luckily enough, somebody else already  
implemented the tracking algorithms for us  
(and much more)



# A short introduction to Geant4

A sketch of the ATLAS MC simulation

# Geant4 (GEometry ANd Traking)

---

- Developed by an International Collaboration
  - Established in 1998
  - Approximately 100 members, from Europe, US and Japan
  - <http://geant4.org>
- Open source
- Written in C++ language
  - Takes advantage from the Object Oriented software technology

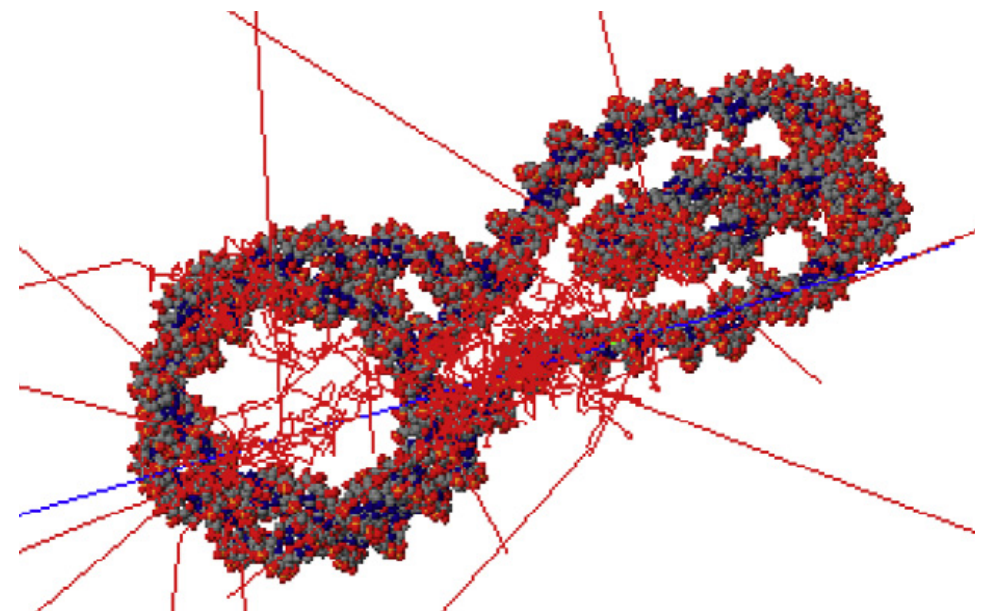
[Geant4, a simulation toolkit Nucl. Inst. and Methods Phys. Res. A, 506 250-303]

Geant4 developments and applications Transaction on Nuclear Science 53, 270-278]



# Geant4 applications

- Physics experiments
- but also:
  - Hadrontherapy
  - Radiobiology
  - and many others...



atomistic view of a dinucleosome  
irradiated by a single 100 keV proton  
Image from M. A. Bernal et al Physica Medica, vol. 31, no. 8, pp.  
861–874, Dec. 2015.



# Geant4, further applications

- Radio-protection in space mission
- Shielding for satellites
- Single event upset and radiation damages to electronics
- Simulations for nuclear spallation sources
- Radioactive waste



First slide of the talk “ESA Geant4 R&D Activities from the Geant4 Space User Workshop Hiroshima, 26 August 2015

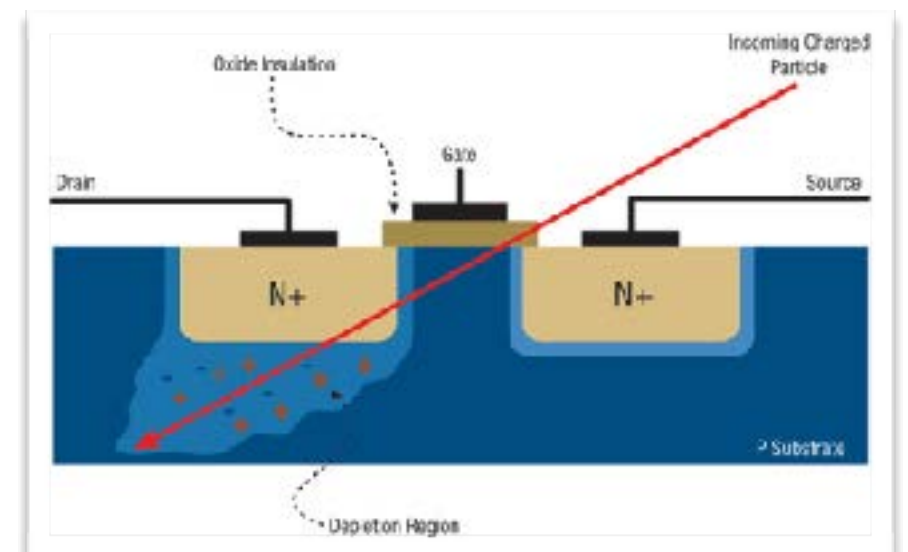
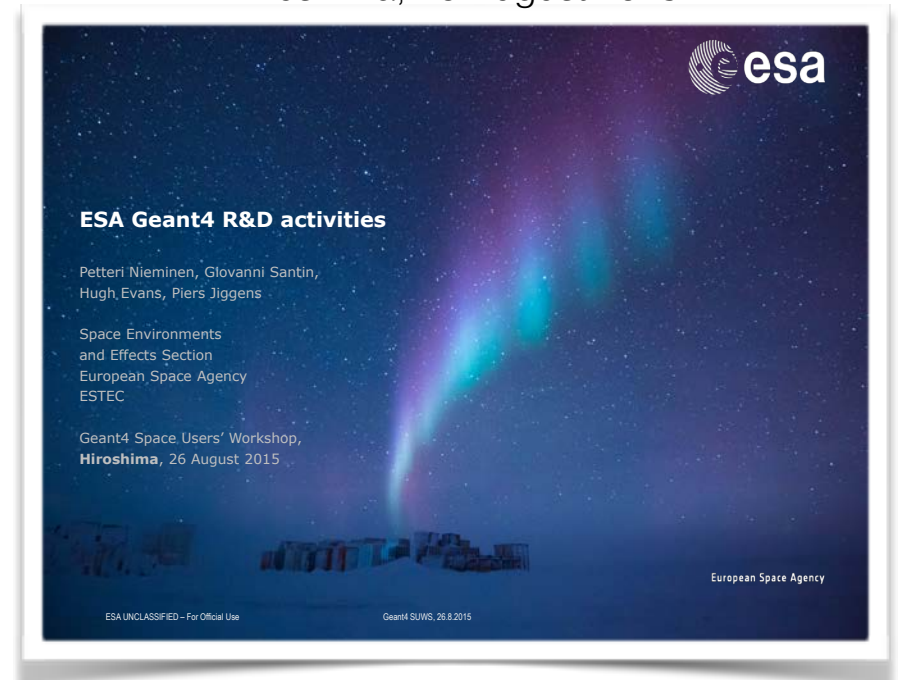
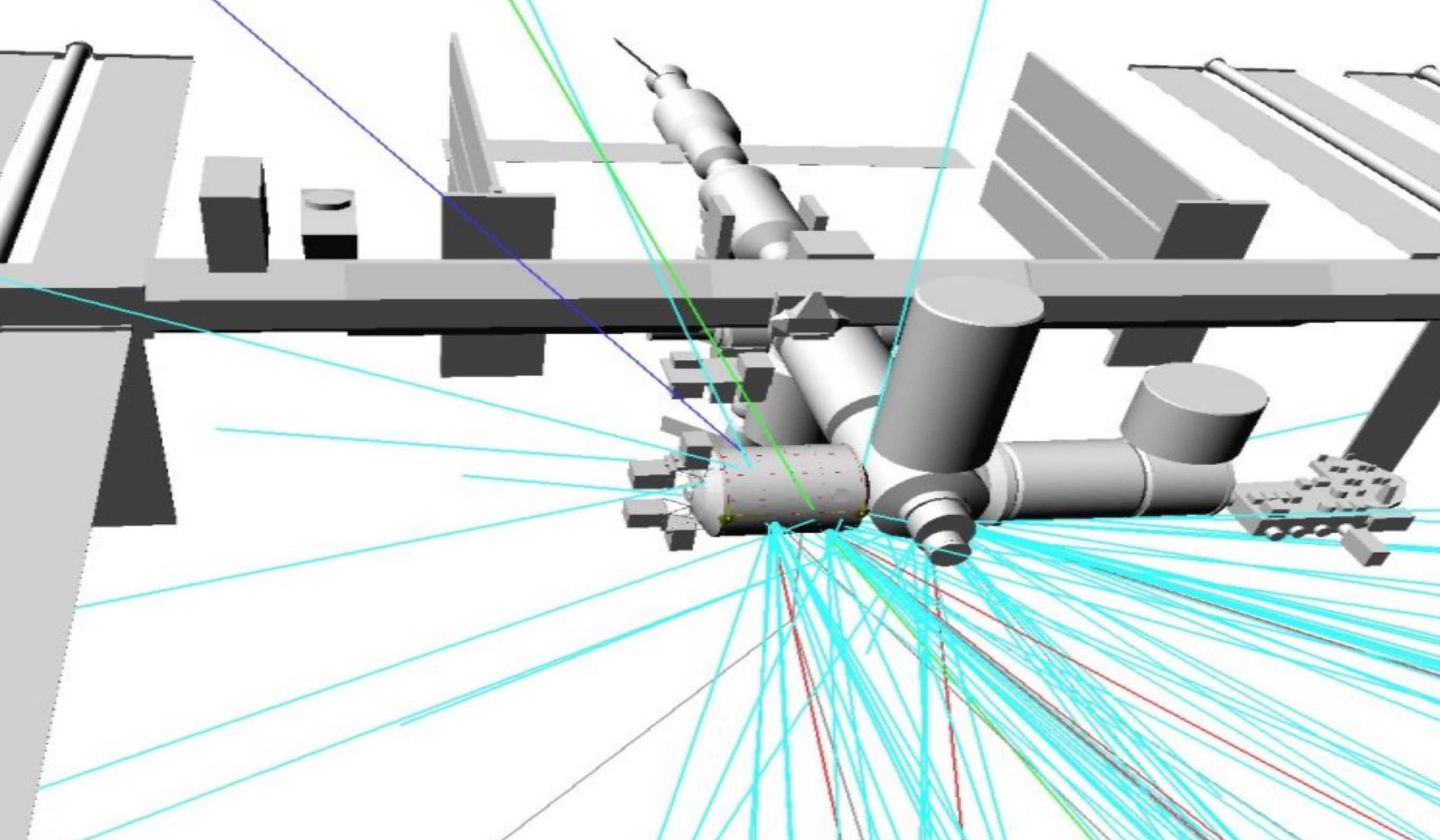


Figure from M. Sawant, COTS Journal Jan. 2012



# How to install Geant4

some tips



# Check on website

---

- You can find the most recent tips on:

[http://www.roma1.infn.it/~mancinit/?action=3Teaching/  
Suggestions on Geant4 installation](http://www.roma1.infn.it/~mancinit/?action=3Teaching/Suggestions%20on%20Geant4%20installation)



# Prerequisites on CentOS7

---



- `sudo yum update`
- `sudo yum groupinstall "Development Tools"`
- `sudo yum install xerces-c xerces-c-devel`
- `sudo yum install centos-release-scl-rh`
- `sudo yum install qt qt-x11 qt-devel`
- optional:
  - `sudo yum install devtoolset-7`  
`scl enable devtoolset-7 bash`
  - `sudo yum install python27 python27-python-pip`  
`python27-python-tools python27-numpy`



# CMake3 for CentOS7

---

- cmake is needed to compile and use Geant4
- the version available on the CentOS7 repository is too old, we have to compile it
- `wget https://github.com/Kitware/CMake/releases/download/v3.13.2/cmake-3.13.2.tar.gz`
- `tar -xvzf cmake-3.13.2.tar.gz`
- `cd cmake-3.13.2`
- `./bootstrap && gmake`
- Add to ~/.bashrc the line:  
`export PATH=/home/soft/cmake-3.13.2/bin:$PATH`



# Prerequisites on RedHat

---

- `sudo yum update`
- `sudo yum groupinstall "Development Tools"`
- `sudo yum install xerces-c xerces-c-devel libXmu libXmu`
- `sudo yum install cmake`
- `sudo yum install qt qt-x11 qt-devel`
- optional:
  - `sudo yum install devtoolset-7`  
`scl enable devtoolset-7 bash`
  - `sudo yum install python27 python27-python-pip`  
`python27-python-tools python27-numpy`

# Prerequisites on Debian/Ubuntu

---



- `sudo apt update`
- `sudo apt install build-essential git`
- `sudo apt install cmake`
- `sudo apt install qt4-default qt4-dev-tools libxerces-c-dev`



# Prerequisites on Mac Os X

---

- install XCode and type in a terminal:  
`xcode-select -install`
- install XQuartz (<https://www.xquartz.org/>)
- install brew:  
`/usr/bin/ruby -e "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/master/install)"`
- `brew doctor; brew update`
- `brew install cmake xerces-c wget`
- ~~`brew install carlomt/repo/qt@4`~~

# Preparing the installation

---



- `wget http://cern.ch/geant4-data/releases/geant4.10.05.tar.gz`
- `tar -xvzf geant4.10.05.tar.gz`  
this will create `geant4.10.05/`
- `mkdir geant4.10.05-build`
- `mkdir geant4.10.05-install`
- `mkdir geant4-data`
- `cd geant4.10.05-build`

# Compile and install



- ```
cmake -DCMAKE_BUILD_TYPE=RelWithDebInfo
-DGEANT4_INSTALL_DATA=ON
-DGEANT4_INSTALL_DATADIR=../geant4-data
-DGEANT4_BUILD_MULTITHREADED=ON
-DGEANT4_USE_GDML=ON -DGEANT4_USE_QT=ON
-DGEANT4_BUILD_CXXSTD=c++11
-DCMAKE_INSTALL_PREFIX=../geant4.10.05-install
../geant4.10.05
```
- ```
make -j `nproc`
```
- ```
make install
```
- Add to your .bashrc (~/.bash\_profile on a Mac):  

```
export G4DIR=~ /geant4.10.05-install
. $G4DIR/bin/geant4.sh
alias g4make='cmake -DGeant4_DIR=$G4DIR'
```



# The cluster at Roma3

---

- CentOS7 machine
- `ssh username@ui7-01.roma3.infn.it`
- All the software (Geant4.10.5, ROOT6, CMake):  
`/storage/DATA-05/gridrm3/Geant4/soft/`
- to add the executables to your path and sett all the environmental variables:  
`source /storage/DATA-05/gridrm3/Geant4/soft/setup.sh`

# Docker / Singularity

---

- You can also use a container system
- <https://github.com/wtakase/docker-geant4>
- The Geant4 dataset is usually expected on the host machine



```

16  $( "#word-list-out" ).e( " " );
17  var b = k();
18  h();
19  var c = l(), a = " ", d = parseInt( $( "#limit_val" ).a() ), f = parseInt( $( "#limit_val" ).a() );
20  function( "LIMIT_total:" + d );
21  function( "rand:" + f );
22  d < f && ( f = d, function( "check rand\u00f3\u00f3rand: " + f + "tops: " + d );
23  var n = [], d = d - f, e;
24  if ( 0 < c.length ) {
25      for ( var g = 0; g < c.length; g++ ) {
26          e = m( b, c[ g ] ), -1 < e && b.splice( e, 1 );
27      }
28      for ( g = 0; g < c.length; g++ ) {
29          b.unshift( { use_wystepuje: "parameter", word: c[ g ] } );

```

## Some basic features of C++

[slides made getting inspiration from  
<http://www.cplusplus.com>]

# Just an introduction

---

- This is not a C++ course
- Just few information useful to understand the Geant4 examples
- For a complete course:  
<http://www.roma1.infn.it/people/rahatlou/index.php?link=Didattica&sublink=ppp>



# Few things about C++

---

- A general-purpose programming language
- Has imperative, **object-oriented** and generic programming features
- Provides facilities for low-level memory manipulation
- In 1983, "C with Classes" was renamed to "C++" (++ being the increment operator in C)
- Initially standardised in 1998  
(current standard is C++17 but the most used is C++11)

# Classes

---

- Classes are an expanded concept of data structures: like data structures, they can contain data members, but they can also contain functions as members



Like Plato's ideas (the idea of apple), classes have generic attributes (e.g. color). Each instance (this Golden Delicious apple) of the class have a specific attribute (e.g. yellow)

```
class Apple {  
public:  
    void setColor(color);  
    color getColor();  
  
private:  
    color fColor;  
    double fWeight;  
};
```

# Example of class usage

```
#include <iostream>
using std::cout;
```

```
class Rectangle {
    int width, height;
public:
    void set_values (int,int);
    int area() {return width*height;}
};
```

```
void Rectangle::set_values (int x, int y)
{
    width = x;
    height = y;
}
```

```
int main () {
    Rectangle rect;
    rect.set_values (3,4);
    cout << "area: " << rect.area();
}
```

Idea of rectangle

An instance  
of rectangle

# Example of class usage

```
#include <iostream>
using std::cout;
```

```
class Rectangle {
    int width, height;
public:
    void set_values (int, int);
    int area() {return width*height;}
};
```

Declaration

Namespace

```
void Rectangle::set_values (int x, int y)
{
    width = x;
    height = y;
}
```

Implementation

```
int main () {
    Rectangle rect;
    rect.set_values (3, 4);
    cout << "area: " << rect.area();
}
```

Usage of the  
methods



# Example of class usage

```
#include <iostream>
using std::cout;

class Rectangle {
    int width, height;
public:
    void set_values (int,int);
    int area() {return width*height;}
};

void Rectangle::set_values (int x, int y)
{
    width = x;
    height = y;
}

int main () {
    Rectangle rect;
    rect.set_values (3,4);
    cout << "area: " << rect.area();
}
```

Hyperuranion  
(ὑπερουράνιος τόπος)  
literally: "place beyond heaven"



"Real" world

What if I want to protect the rectangle properties (the dimensions), once instantiated?



# Constructors

---


```
#include <iostream>
using std::cout;

class Rectangle {
    int width, height;
public:
    Rectangle(int x, int y);
    int area() {return width*height;}
};

Rectangle::Rectangle(int x, int y)
{
    width = x;
    height = y;
}

int main () {
    Rectangle rect(3,4);
    cout << "area: " << rect.area();
    return 0;
}
```

Using the  
constructor and  
removing the  
setting method



# Constructors

---

```
#include <iostream>
using std::cout;

class Rectangle {
    int width, height;
public:
    Rectangle(int x, int y);
    int area() {return width*height;}
};

Rectangle::Rectangle (int x, int y) :
width(x), height(y) { }

int main () {
    Rectangle rect(3,4);
    cout << "area: " << rect.area();
    return 0;
}
```

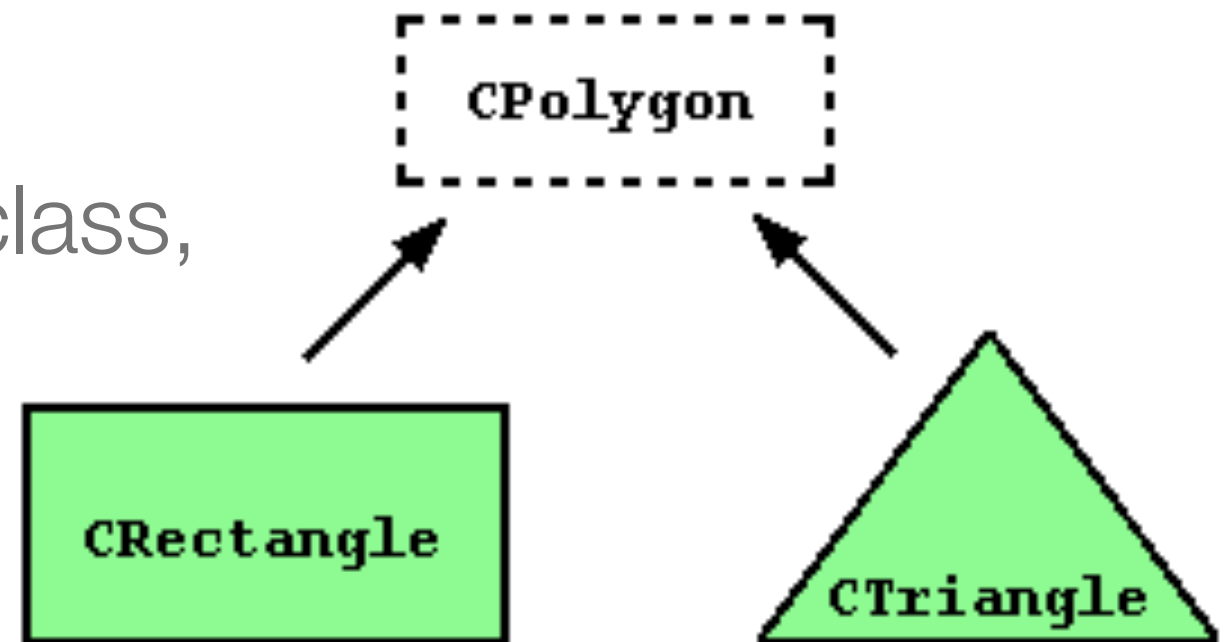


Better  
implementation!

# Inheritance

---

- Classes in C++ can be extended, creating new classes which retain characteristics of the base class
- This process, known as inheritance, involves a base class and a derived class
- The derived class inherits the members of the base class, on top of which it can add its own members



# Inheritance, an example

---

```
class Polygon {  
    protected:  
        int width, height;  
    public:  
        void set_values (int a, int b)  
            { width=a; height=b; }  
};
```

```
class Rectangle: public Polygon  
{  
    public:  
        int area ()  
        {  
            return width*height;  
        }  
};
```

```
class Triangle: public Polygon  
{  
    public:  
        int area()  
        {  
            return width*height/2;  
        }  
};
```



# Protected and not private!

---

- The protected access specifier used in class Polygon is similar to private. Its only difference occurs in fact with inheritance:
- When a class inherits another one, the members of the derived class can access the protected members inherited from the base class, but not its private member
- By declaring width and height as protected instead of private, these members are also accessible from the derived classes Rectangle and Triangle, instead of just from members of Polygon
- If they were public, they could be accessed just from anywhere

# Let's use the classes...

---

```
#include <iostream>
using std::cout;
using std::endl;

int main () {
    Rectangle rect;
    Triangle trgl;
    rect.set_values (4,5);
    trgl.set_values (4,5);
    cout << rect.area() << endl;
    cout << trgl.area() << endl;
    return 0;
}
```

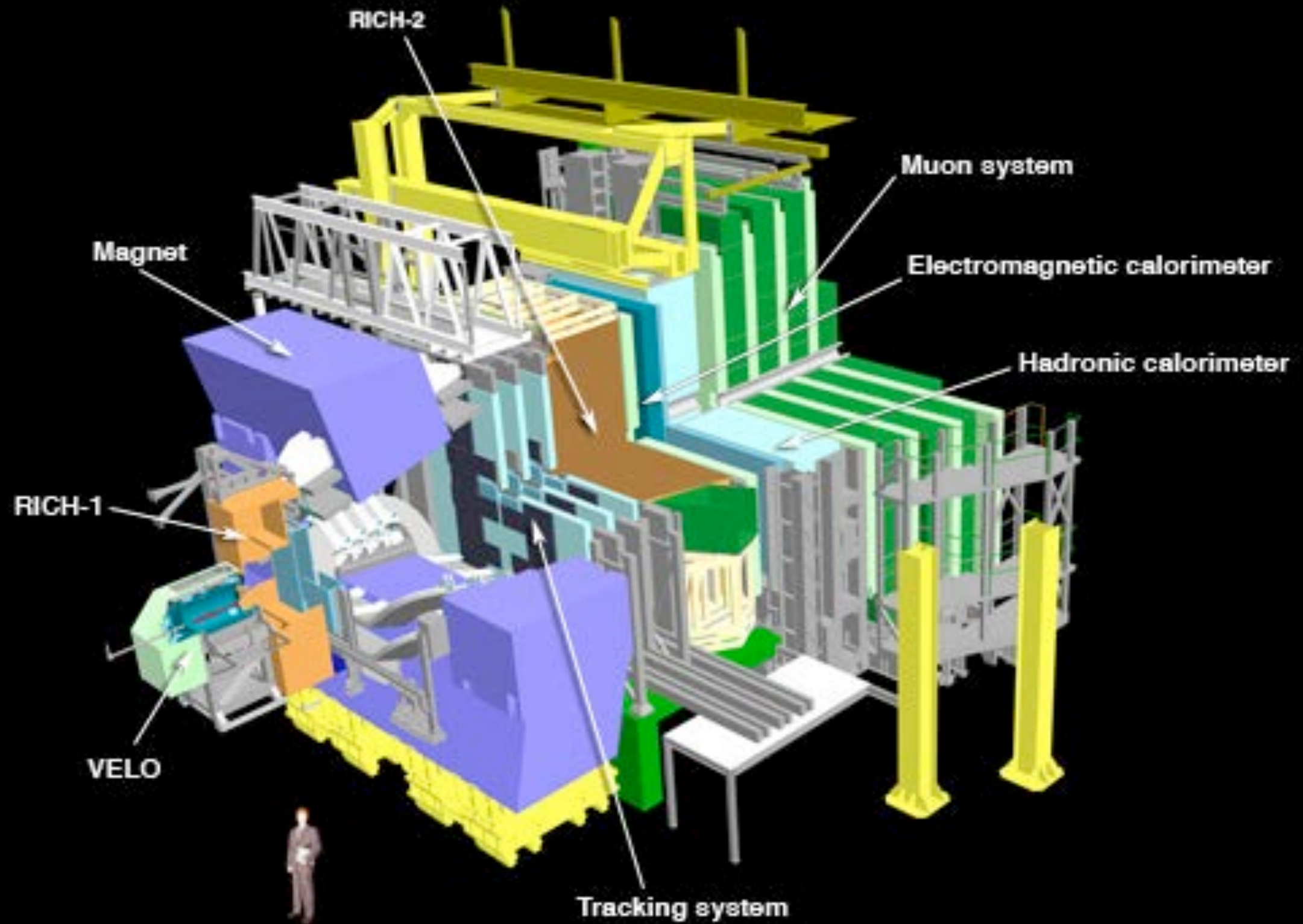
have a look at the example

[https://github.com/carlomt/inheritance\\_example](https://github.com/carlomt/inheritance_example)  
for more details

# CMake



- a cross-platform free and open-source software application for managing the build process of software using a compiler-independent method
- supports directory hierarchies and multiple libraries
- can locate executables, files, and libraries
- <https://cliutils.gitlab.io/modern-cmake/>
- use a version of CMake that came out after your compiler
- since CMake will dumb itself down to the minimum required version in your CMake file, installing a new CMake, even system wide, is pretty safe



Lets get back to Geant4

Reminder...

# Geant4 is a toolkit

---

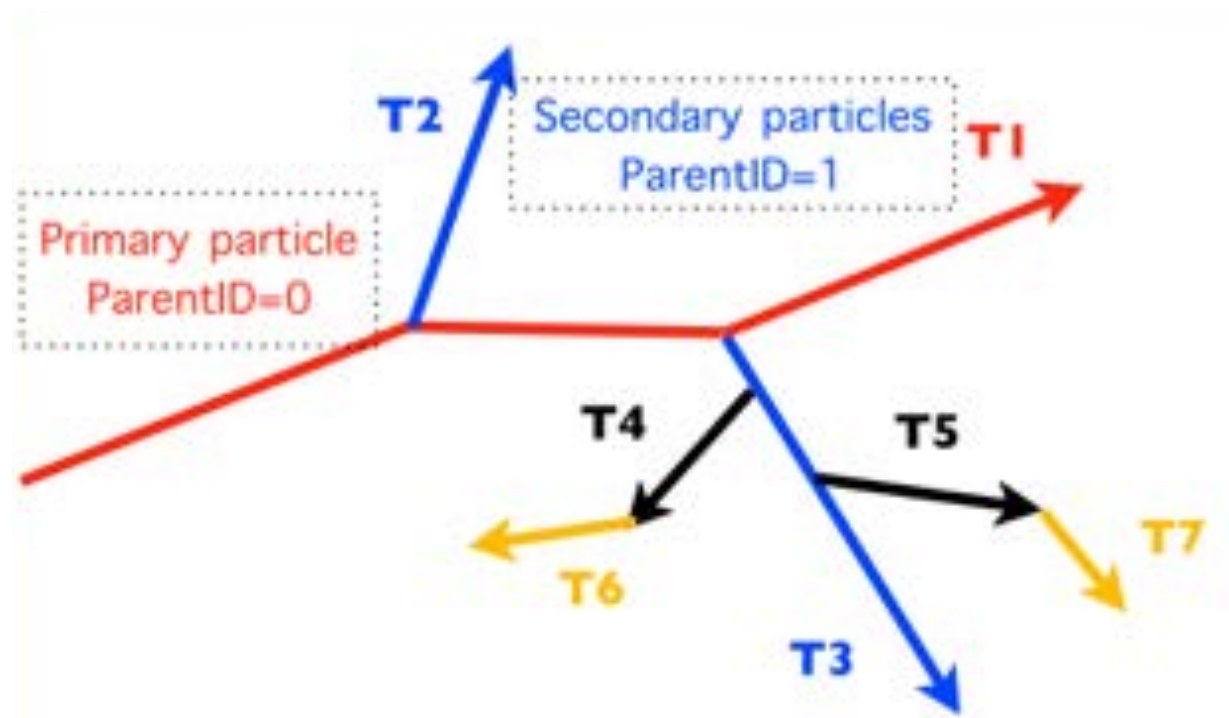
- Geant4 is a **toolkit** (= a collection of tools)
  - i.e. you **cannot** run it out of the box
  - You **must write an application**, which uses Geant4
- Consequences:
  - There are **no** such concepts as “Geant4 **defaults**”
  - You must provide the necessary information to configure your simulation
  - You must deliberately choose which Geant4 tools to use
- Guidance: many examples are provided
  - **Basic Examples**: overview of Geant4 tools
  - Advanced Examples: Geant4 tools in real-life applications



# Geant4 way of tracking

---

- Force step ending at **geometry boundaries**
- All **AlongStep** processes **co-occur**
- The **PostStep** compete, i.e.: **only one** is selected

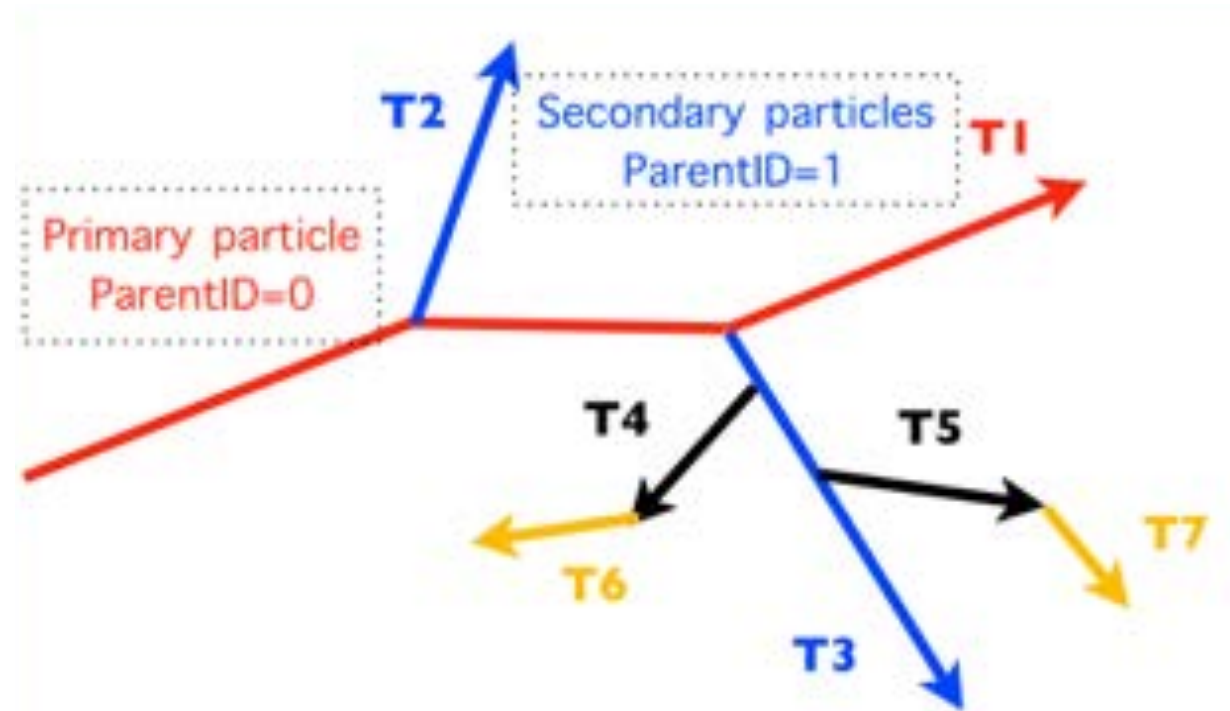




# Geant4 way of tracking

---

- **If particle is at rest** chose one of the **AtRest** processes
- The secondaries are saved in the stack
- To be further tracked with a **last in first out** approach





# You MUST:

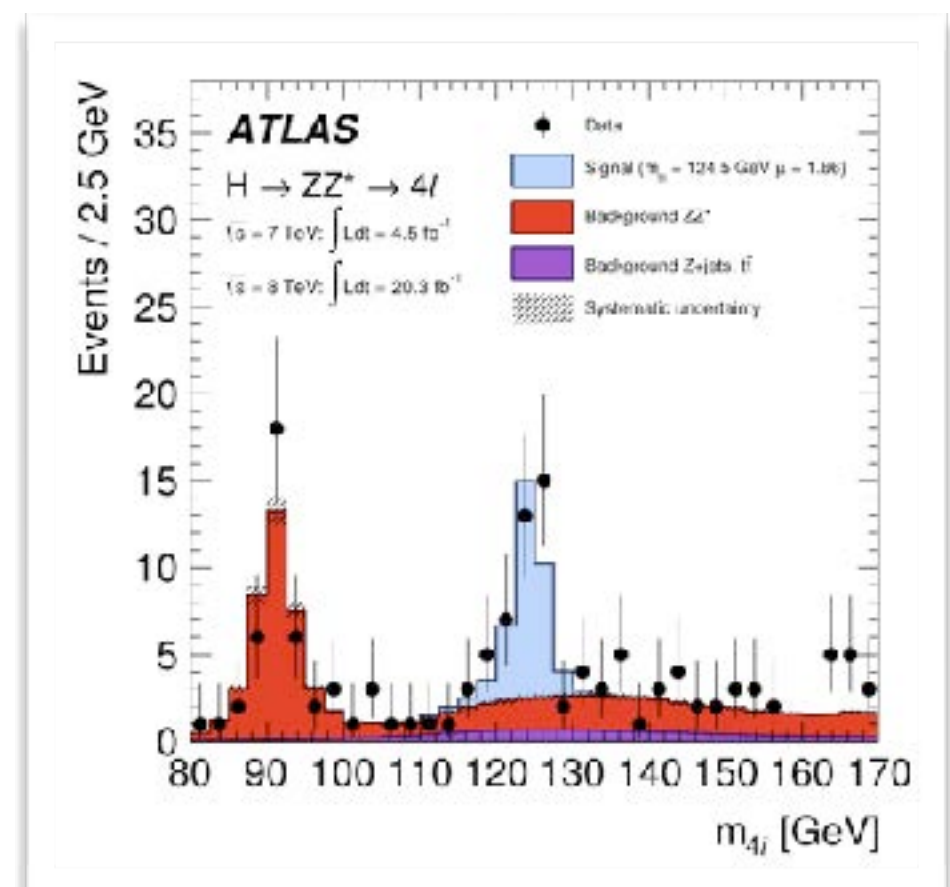
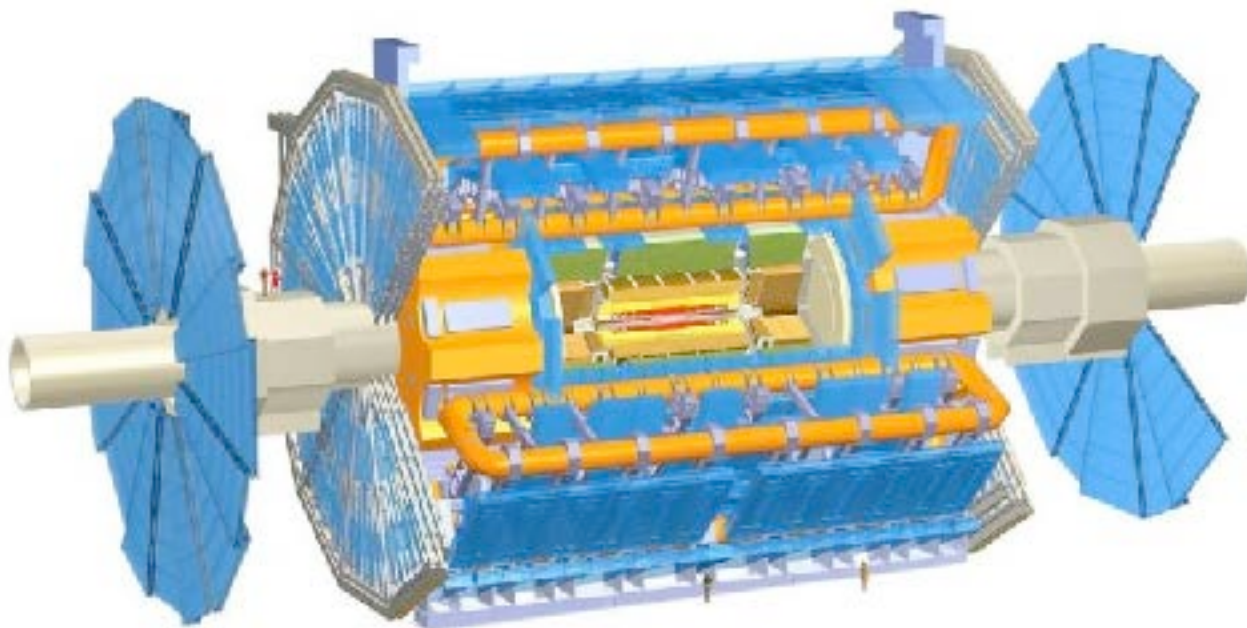
---

- Describe your experimental set-up
- Provide the primary particles input to your simulation
- Decide **which particles** and **physics models** you want to use out of those available in Geant4 and the precision of your simulation (cuts to produce and track secondary particles)



# You may also want to:

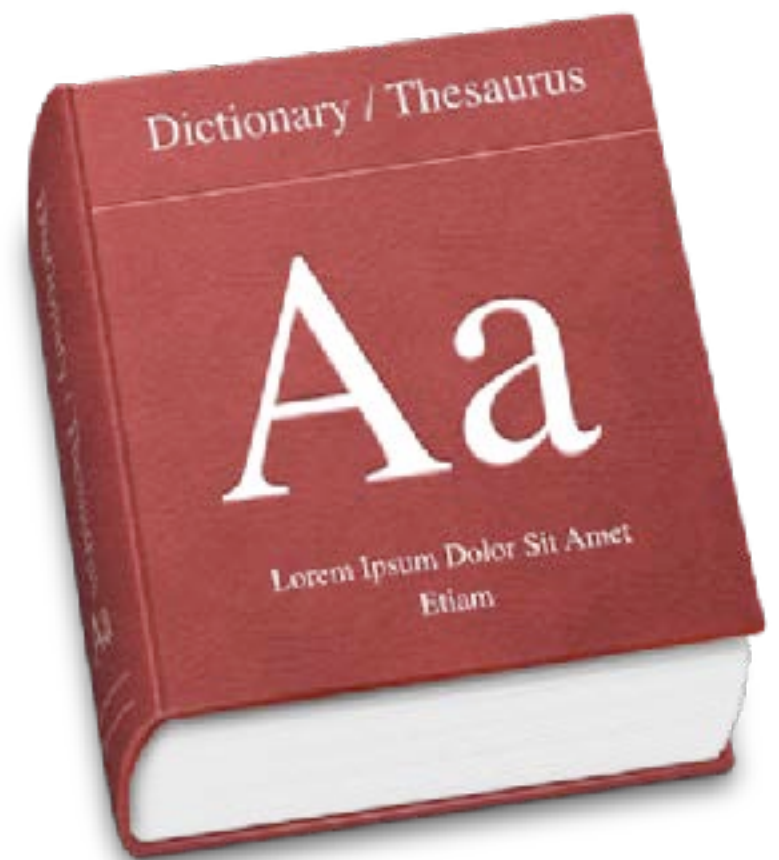
- Interact with Geant4 kernel to control your simulation
- Visualise your simulation configuration or results
- Produce histograms, tuples etc. to be further analysed



# Jargons

---

- Run, event, track, step, step point
- Process
  - At rest, along step, post step
- Cut = production threshold
- Sensitive detector, score, hit, hits collection,

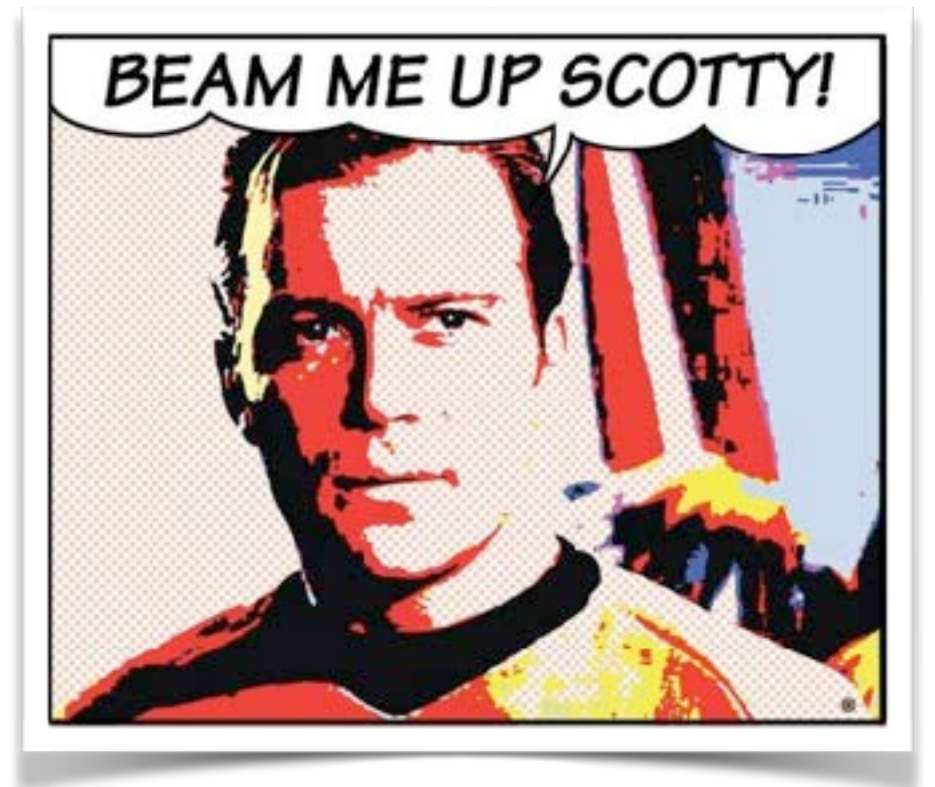




# A run in Geant4

---

- As an analogy of the real experiment, a run of Geant4 starts with “**Beam On**”
- A run is a collection of events which share the same detector and physics conditions
- **G4RunManager** class manages processing a run
- **G4UserRunAction** is the optional user hook



# Just like a HEP experiment...

Run

Hard interaction

Secondaries

Detectors

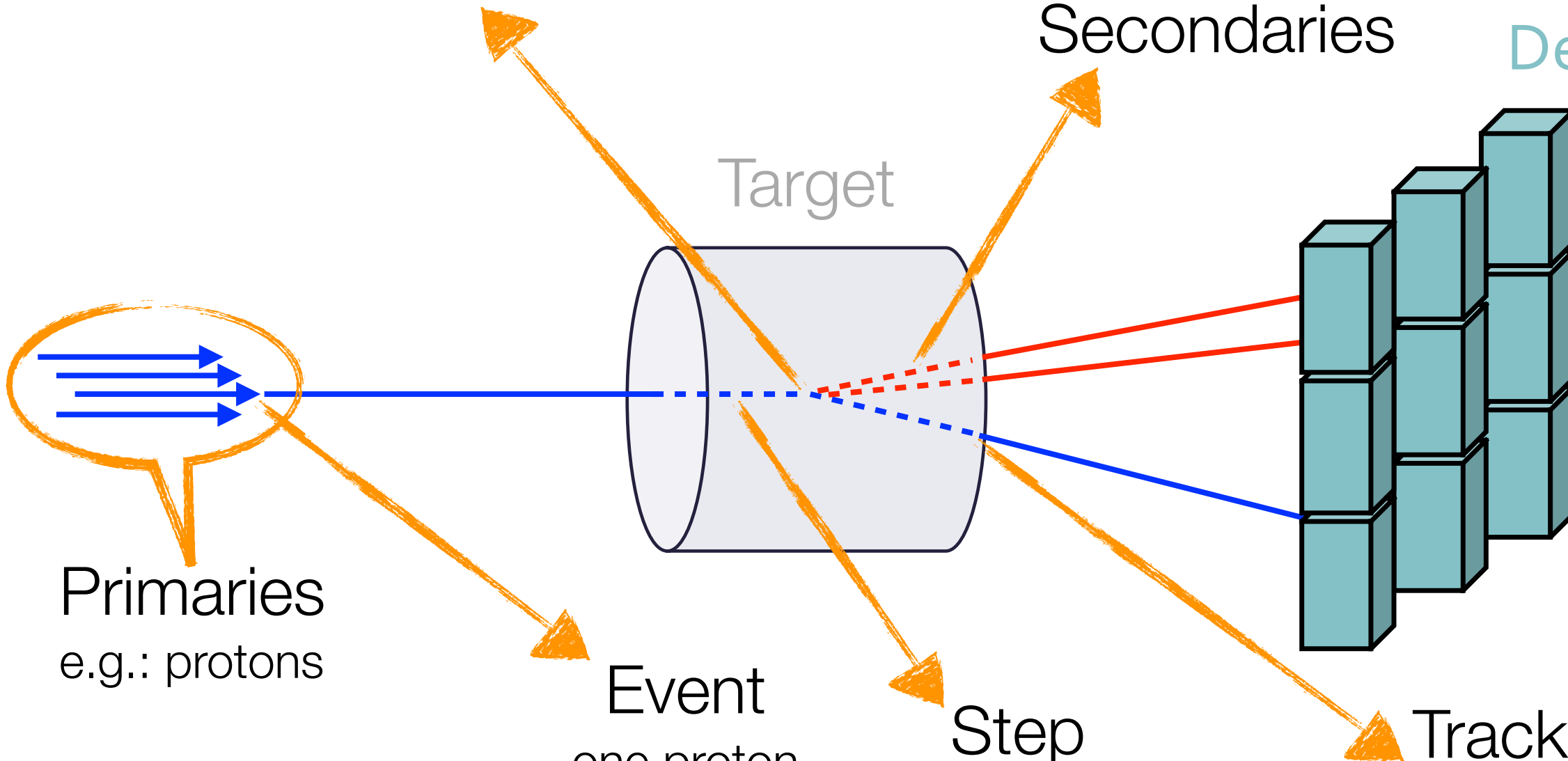
Target

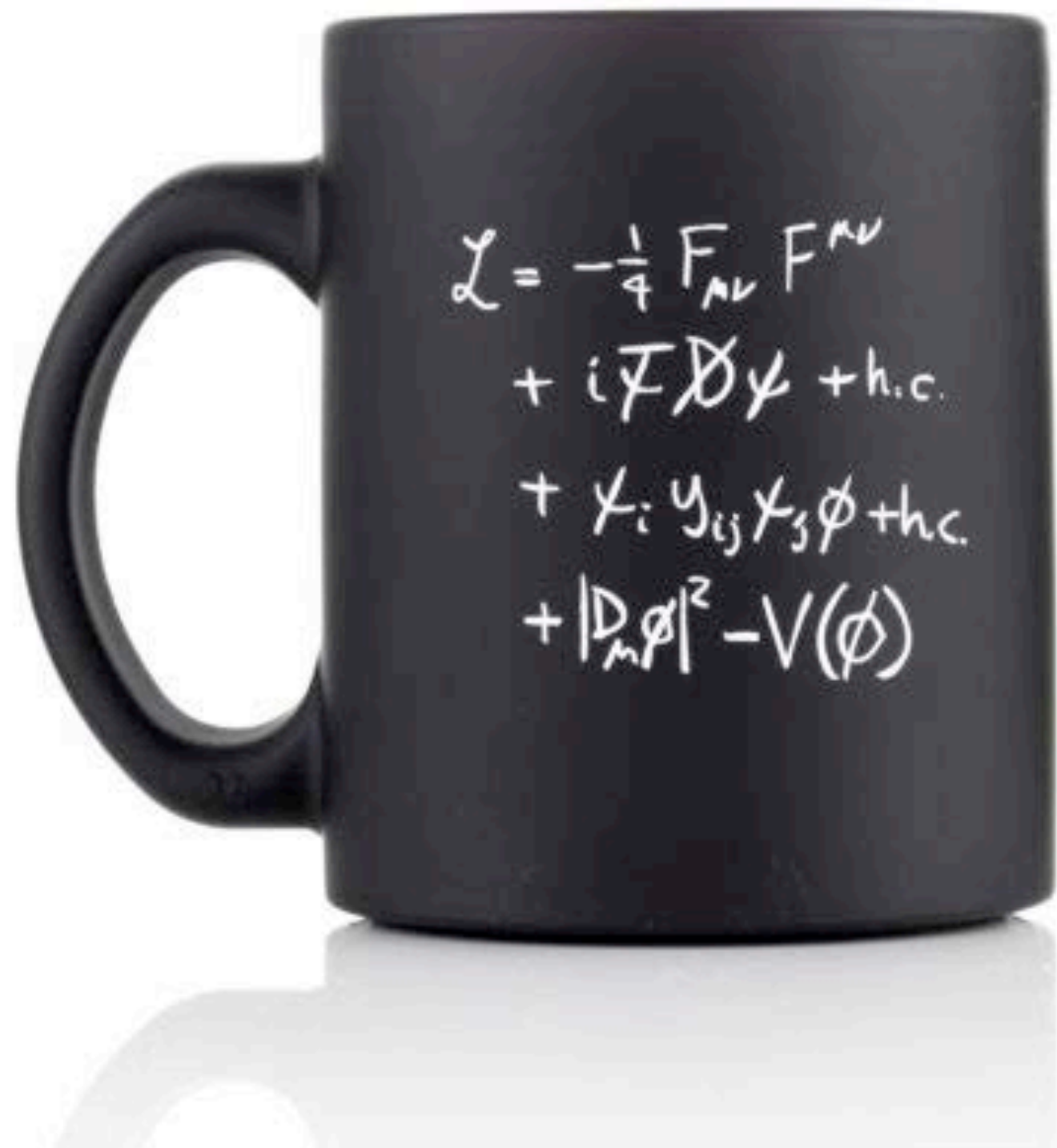
Primaries  
e.g.: protons

Event  
one proton

Step  
elementary  
interaction

Track  
the properties of a particle  
in a specific moment





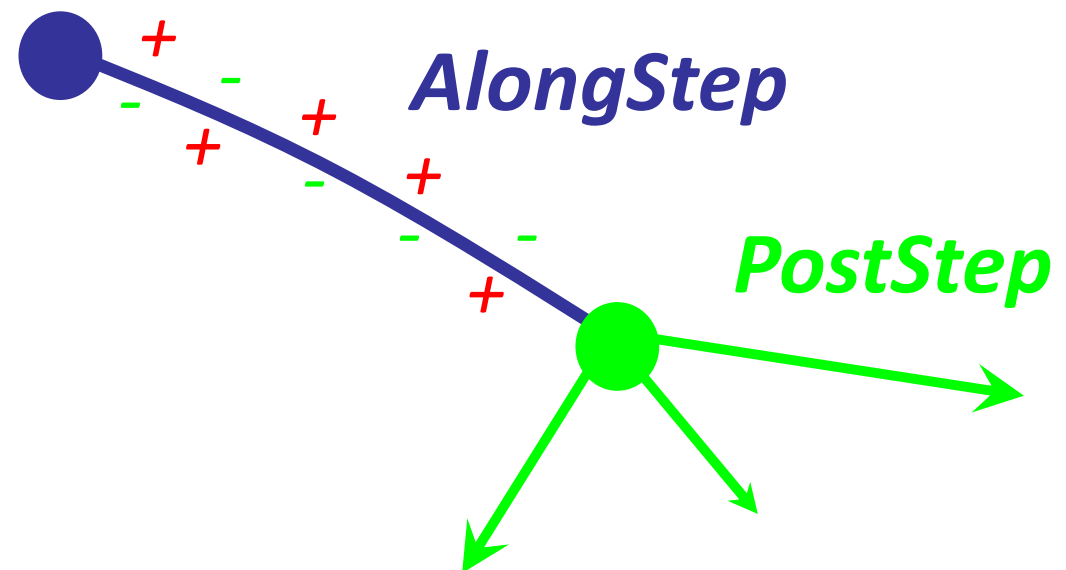
Physics processes

an overview...

# The G4VProcess

---

- All physics processes derive from G4VProcess
- G4VProcess is an abstract class
- It defines the common interface of all processes in Geant4
- Three kind of “actions”:
  - **AlongStep**  
all the soft interactions
  - **PostStep**  
all the hard interactions
  - **AtRest**  
decays, e+ annihilation





# $\gamma$ model inventory

- Many models available for each process
- Differ for energy range, precision and CPU speed
- Final state generators

| Model                           | $E_{\min}$ | $E_{\max}$ |
|---------------------------------|------------|------------|
| G4LivermoreRayleighModel        | 100 eV     | 10 PeV     |
| G4PenelopeRayleighModel         | 100 eV     | 10 GeV     |
| G4KleinNishinaCompton           | 100 eV     | 10 TeV     |
| G4KleinNishinaModel             | 100 eV     | 10 TeV     |
| G4LivermoreComptonModel         | 100 eV     | 10 TeV     |
| G4PenelopeComptonModel          | 10 keV     | 10 GeV     |
| G4LowEPComptonModel             | 100 eV     | 20 MeV     |
| G4BetheHeitlerModel             | 1.02 MeV   | 100 GeV    |
| G4PairProductionRelModel        | 10 MeV     | 10 PeV     |
| G4LivermoreGammaConversionModel | 1.02 MeV   | 100 GeV    |
| G4PenelopeGammaConversionModel  | 1.02 MeV   | 10 GeV     |
| G4PEEFluoModel                  | 1 keV      | 10 PeV     |
| G4LivermorePhotoElectricModel   | 10 eV      | 10 PeV     |
| G4PenelopePhotoElectricModel    | 10 eV      | 10 GeV     |


# ElectroMagnetic models

---

- The same physics processes can be described by different models
- For instance: Compton scattering can be described by
  - `G4KleinNishinaCompton`
  - `G4LivermoreComptonModel` (low-energy, based on the Livermore database)
  - `G4PenelopeComptonModel` (low-energy, based on the Penelope analytical model)
  - `G4LivermorePolarizedComptonModel` (low-energy, Livermore database with polarization)
  - `G4PolarizedComptonModel` (Klein-Nishina with polarization)
  - `G4LowEPComptonModel` (full relativistic 3D simulation)
- Different models can be combined, so that the appropriate one is used in each given energy range (à performance optimization)

# EM Physics constructors

---

|                               |                                                                                       |
|-------------------------------|---------------------------------------------------------------------------------------|
| G4EmStandardPhysics           | – default                                                                             |
| G4EmStandardPhysics_option1   | – HEP fast but not precise                                                            |
| G4EmStandardPhysics_option2   | – Experimental                                                                        |
| G4EmStandardPhysics_option3   | – medical, space                                                                      |
| G4EmStandardPhysics_option4   | – optimal mixture for precision                                                       |
| G4EmLivermorePhysics          |  |
| G4EmLivermorePolarizedPhysics |                                                                                       |
| G4EmPenelopePhysics           |                                                                                       |
| G4EmLowEPPhysics              |                                                                                       |
| G4EmDNAPhysics_option...      |                                                                                       |

Combined Physics  
Standard > 1 GeV  
**LowEnergy < 1 GeV**

...

- Advantage of using of these classes – they are **tested on regular basis** and are used for regular validation

# Hadronic processes

---

- At rest
  - Stopped muon, pion, kaon, anti-proton
  - Radioactive decay
  - Particle decay (decay-in-flight is PostStep)
- Elastic
  - Same process to handle all long-lived hadrons (multiple models available)
- Inelastic
  - Different processes for each hadron (possibly with multiple models vs. energy)
  - Photo-nuclear, electro-nuclear, mu-nuclear
- Capture
  - Pion- and kaon- in flight, neutron
- Fission

# Hadronic physics challenge

---

- Three energy regimes
  - $< 100$  MeV
  - resonance and cascade region (100 MeV - 10 GeV)
  - $> 20$  GeV (QCD strings)
- Within each regime there are several models
- Many of these are phenomenological

# Hadronic models

---

- Two families of builders for the high-energy part
  - **QGS**, or list based on a model that use the Quark Gluon String model for high energy hadronic interactions of protons, neutrons, pions and kaons
  - **FTF**, based on the FTF (FRITIOF like string model) for protons, neutrons, pions and kaons
- Three families for the cascade energy range
  - **BIC**, binary cascade
  - **BERT**, Bertini cascade
  - **INCLXX**, Liege Intranuclear cascade model

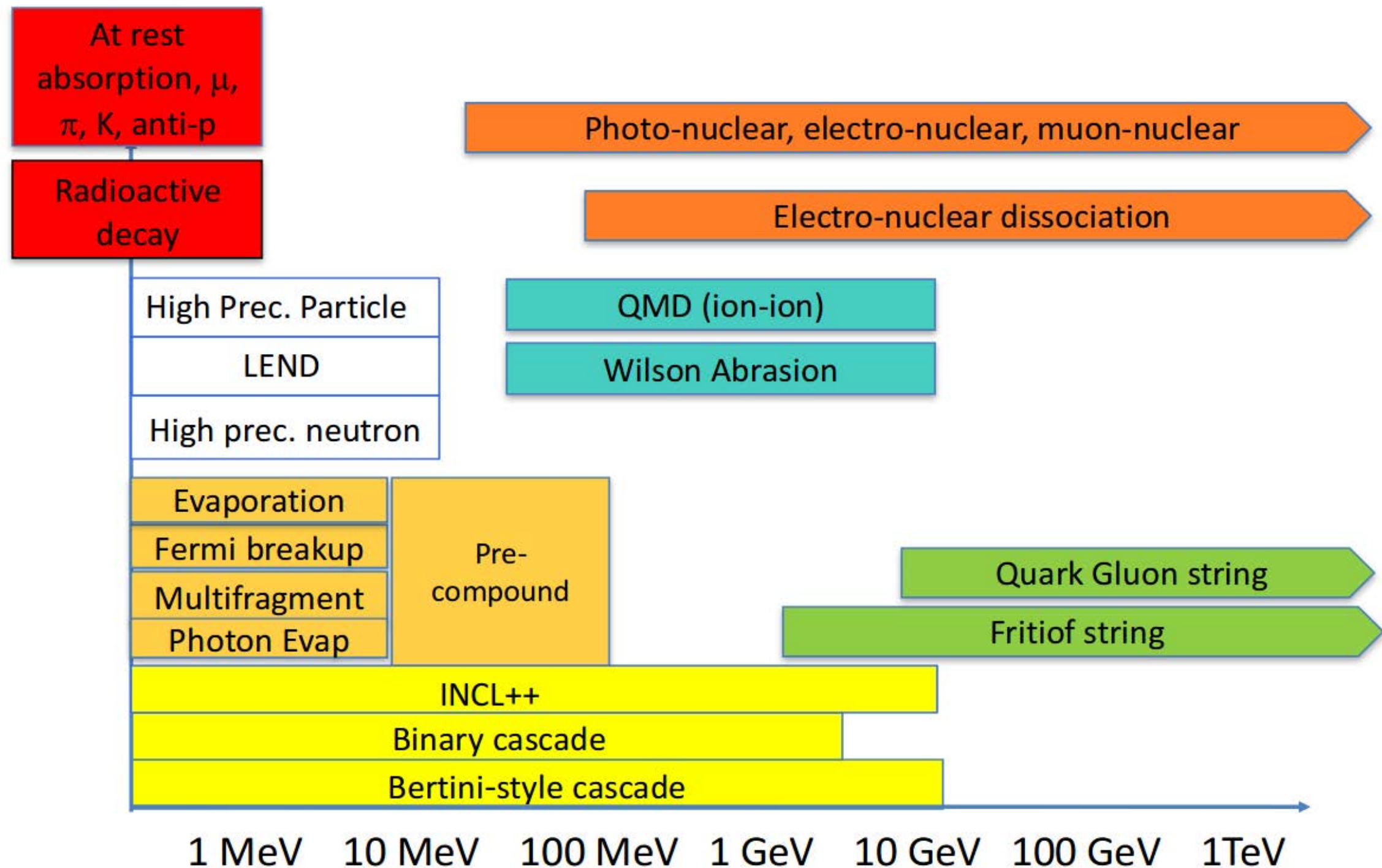
# ParticleHP

---

- Data-driven approach for inelastic reactions for n (in place since many years, named NeutronHP) p, d, t,  $^3\text{He}$  and  $\alpha$
- Data based on TENDL-2014 (charged particles) and ENDFVII.r1 (neutrons).
- For neutrons, includes information for elastic and inelastic scattering, capture, fission and isotope production
- Range of applicability: from thermal energies up to 20 MeV
- Very precise tracking, but also very slow
- Use it with care: thermal neutron tracking is very CPU-demanding



# Hadronic model inventory



# Physics List

---

- A class which collects all the particles, physics processes and production thresholds needed for your application
- It tells the run manager how and when to invoke physics
- It is a very flexible way to build a physics environment
  - user can pick the particles he wants
  - user can pick the physics to assign to each particle
- But, **user must have a good understanding of the physics required**
- Omission of particles or physics could cause errors or **poor simulation**

# There is not default, but...

---

- Geant4 provides several “production physics lists” which are routinely validated and updated with each release these should be considered only as starting points which you may need to validate or modify for your application
- There are currently 19 packaged physics lists available
- 6 reference physics lists:
  - FTFP\_BERT, FTFP\_BERT\_HP QGSP\_BERT, QGSP\_BERT\_HP, QGSP\_BIC QGSP\_FTFP\_BERT

# Naming...

---

- The following acronyms refer to various hadronic options
  - QGS -> Quark Gluon String model ( $>\sim 20$  GeV) FTF -> Fritiof string model ( $>\sim 5$  GeV)  
BIC -> Binary Cascade ( $<\sim 10$  GeV)  
BERT -> Bertini-style cascade ( $<\sim 10$  GeV)
- HP -> High Precision neutron model ( $< 20$  MeV) P -> G4Precompund model used for de-excitation
- EM options designated by
  - no suffix: standard EM physics
  - EMV suffix: older but faster EM processes
    - other suffixes for other EM options

# Production Physics Lists

---

- FTFP\_BERT
  - recommended by Geant4 for HEP
  - contains all standard EM processes
  - uses Bertini-style cascade for hadrons  $< 5$  GeV
  - uses FTF (Fritiof) model for high energies ( $> 4$  GeV)
- QGSP\_BERT
  - all standard EM processes
  - Bertini-style cascade up to 9.9 GeV
  - QGS model for high energies ( $> \sim 18$  GeV) FTF in between

# Production Physics Lists

---

- QGSP\_BIC
  - same as QGSP\_BERT, but replaces Bertini cascade with Binary cascade and G4Precompound model
  - recommended for use at energies below 200 MeV (many medical applications)
- FTFP\_BERT\_HP
  - same as FTFP\_BERT, but with high precision neutron model used for neutrons below 20 MeV
  - significantly slower than FTFP\_BERT when full thermal cross sections used there's an option to turn this off
  - for radiation protection and shielding applications

# Other Physics Lists

---

- If primary particle energy in your application is  $< 5$  GeV (for example, clinical proton beam of 150 MeV)
  - start with a physics list which includes BIC or BERT
  - e.g. QGSP\_BIC, QGSP\_BERT, FTFP\_BERT, etc.
- If neutron transport is important
  - start with physics list containing “HP”
  - e.g. QGSP\_BIC\_HP, FTFP\_BERT\_HP, etc.
- If you’re interested in Bragg curve physics
  - use a physics list ending in “EMV” or “EMX” or “EMY”
  - e.g. QGSP\_BIC\_EMY



# An event in Geant4

---

- An event is the basic unit of simulation in Geant4
- **G4Event** class represents an event. It has following objects at the end of its (successful) processing
  - List of primary vertices and particles (as input)
  - Hits and Trajectory collections (as output)
- **G4EventManager** class manages processing an event
- **G4UserEventAction** is the optional user hook



# A track in Geant4

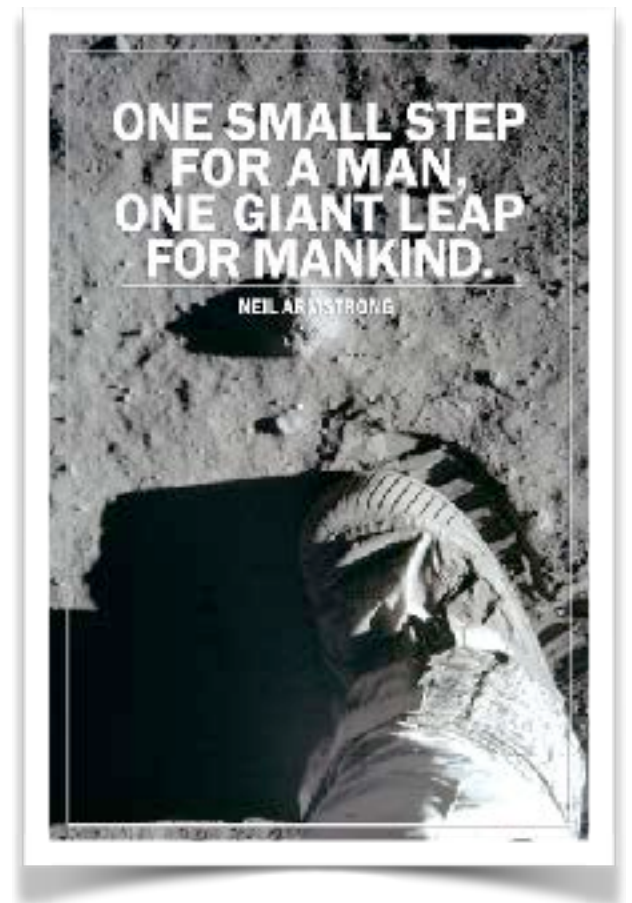
---

- Track is a snapshot of a particle
- It has physical quantities of current instance only. It does not record previous quantities
- It's not a collection of steps. Instead, a track is being updated by steps
- G4TrackingManager manages processing a track, a track is represented by G4Track class
- G4UserTrackingAction is the optional user hook

# A step in Geant4

---

- A step is a variation of a track
- Has two points (pre and post step points)
- In case a step is limited by a boundary, the end point stands on the boundary, and it logically belongs to the next volume
- Boundary processes such as transition radiation or refraction could be simulated
- **G4SteppingManager** class manages processing a step, a step is represented by **G4Step** class
- **G4UserSteppingAction** is the optional user hook



# Particle in Geant4

---

- A particle in Geant4 is represented by three layers of classes:
  - **G4Track**
    - Geometrical information (position)
  - **G4DynamicalParticle**
    - Dynamic physical properties (momentum, energy, spin...)
  - **G4ParticleDefinition**
    - Static properties (charge, mass, life time)



# Sampling the step size

---

- In Geant4, particle transportation is a process as well, by which a particle interacts with geometrical volume boundaries and field of any kind
- Each particle has its own list of applicable processes. At each step, all processes listed are invoked to get proposed physical interaction lengths
- The process which requires the shortest interaction length limits the step

# Let's cut it out... (cuts in MC)

---

- The traditional Monte Carlo solution is to set a tracking cut-off in energy:
  - a particle is stopped when its energy goes below it
  - its residual energy is deposited at that point
- Imprecise stopping and energy deposition location
- Particle and material dependence





# Let's cut it out... (cuts in Geant4)

---

- Geant4 does not have tracking cuts  
i.e.: all tracks are tracked down to 0 energy
- A Cut in Geant4 is a production threshold
- It is applied only for physics processes that have infrared divergence
  - Bremsstrahlung
  - Ionisation  $e^-$  ( $\delta$  rays)
  - Protons from hadronic elastic scattering



# A range cut

---

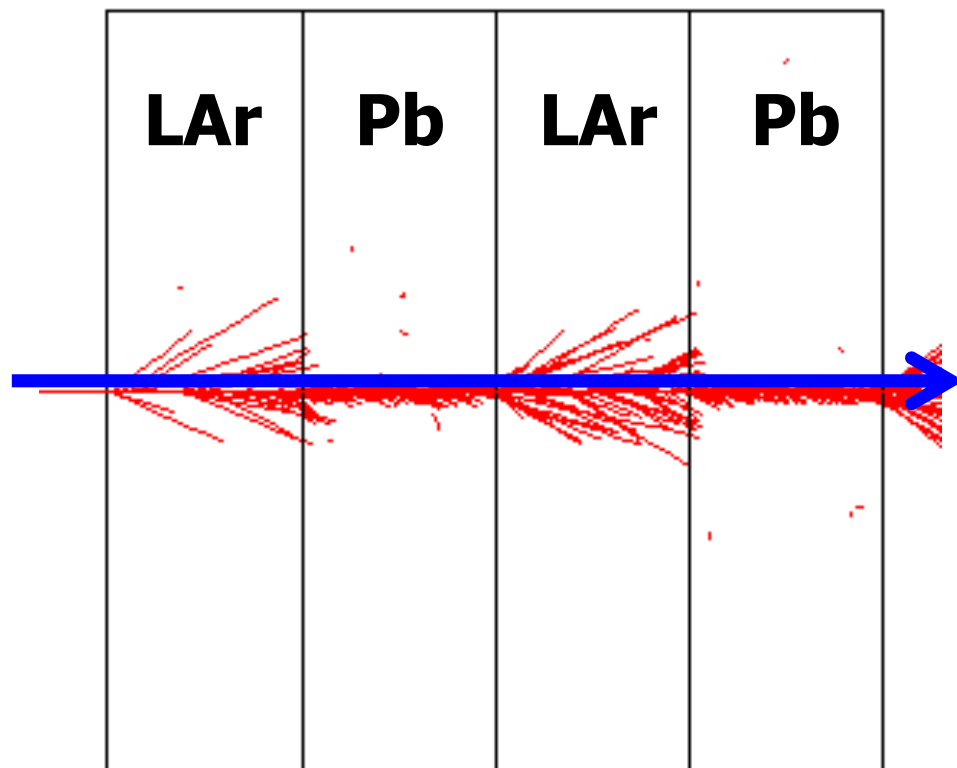
- The threshold is a **distance**!
- Default = 1 mm
- Particles unable to travel at least the range cut value are not produced
- Sets the "spatial accuracy" of the simulation
- Production threshold is internally converted to an energy threshold for each material



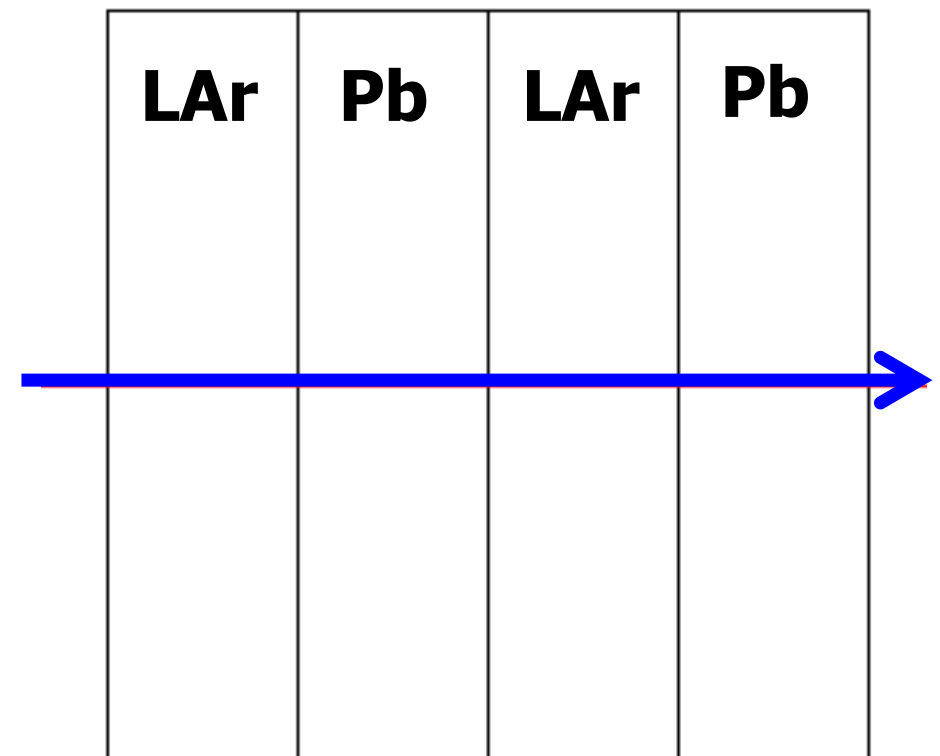
# Cut in energy

---

- 460 keV
- good for LAr
- not for Pb



- 2 MeV
- good for Pb
- not for Lar

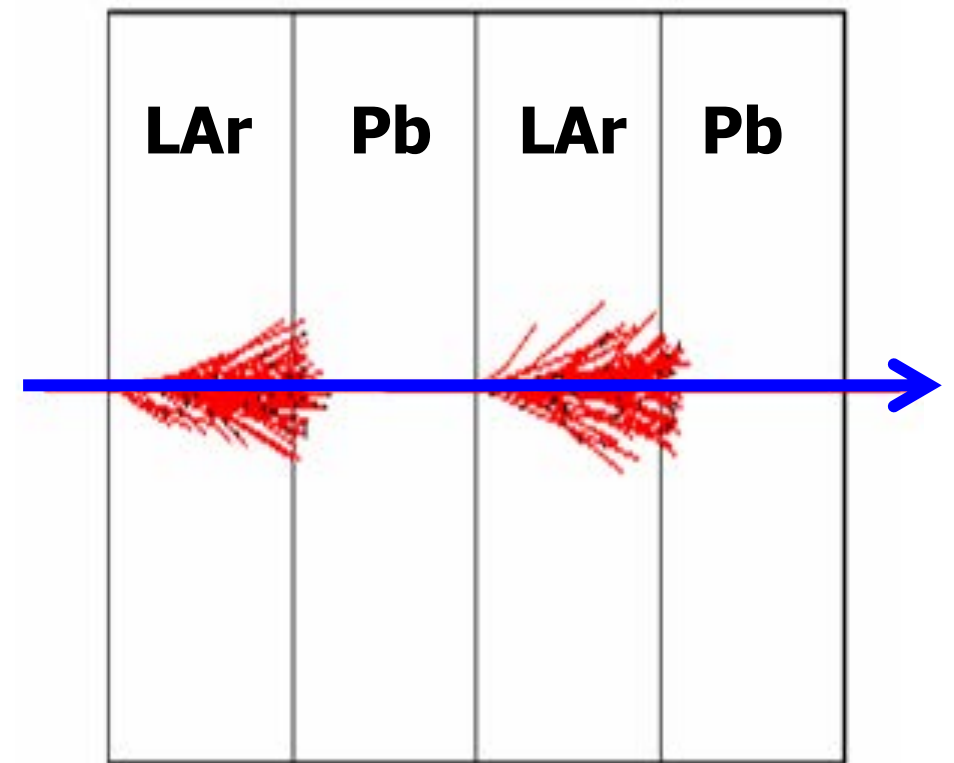


# Cut in range

---

- 1.5 mm
- ~460 KeV in LAr
- ~2 MeV in Pb

*run with the hares and  
hunt with the hounds...  
(good for both!)*



# Setting the cuts

---

- Optional method in G4VPhysicsList

```
void MyPhysicsList::SetCuts()  
{  
    //G4VUserPhysicsList::SetCuts();  
    defaultCutValue = 0.5 * mm;  
    SetCutsWithDefault();  
  
    SetCutValue(0.1 * mm, "gamma");  
    SetCutValue(0.01 * mm, "e+");  
    G4ProductionCutsTable::GetProductionCutsTable()  
        ->SetEnergyRange(100*eV, 100.*GeV);  
}
```



- not all models are able to work with very low production thresholds
- an energy threshold limit is used,
- its default value is set to 990 eV.
- You can change this value

# Cuts UI command

---

```
# Universal cut (whole world, all particles)
/run/setCut 10 mm

# Override low-energy limit
/cuts/setLowEdge 100 eV

# Set cut for a specific particle (whole world)
/run/setCutForAGivenParticle gamma 0.1 mm

# Set cut for a region (all particles)
/run/setCutForARegion myRegion 0.01 mm

# Print a summary of particles/regions/cuts
/run/dumpCouples
```



# Cuts per region

---

- Complex detector may contain many different sub-detectors involving:
  - finely segmented volumes
  - position-sensitive materials (e.g. Si trackers)
  - large, undivided volumes (e.g. calorimeters)
- The same cut may not be appropriate for all of these
- User can define regions (independent of geometry hierarchy tree) and assign different cuts for each region
- A region can contain a subset of the logical volumes

# To limit the step

---

- To have more precise energy deposition
- To increase precision in magnetic field
- Include `G4StepLimiter` in your physics list
  - as a Physics process
  - compete with the others

# System of Units

---

- Internal unit system used in Geant4 is completely hidden not only from user's code but also from Geant4 source code implementation
- Each hard-coded number must be multiplied by its proper unit:
  - `radius = 10.0 * cm; E = 1.*GeV;`
- To get a number, it must be divided by a proper unit:
  - `G4cout<< "E dep: "<< eDep/MeV <<" [MeV]"<<G4endl;`



# User classes

---

- You **have** to write the **main()**
- Initialisation classes:
  - **G4VUserDetectorConstruction**
  - **G4VUserPhysicsList**
  - **G4VUserActionInitialization**
- Action classes
  - **G4VUserPrimaryGeneratorAction**
  - G4UserRunAction
  - G4UserEventAction
  - G4UserStackingAction
  - G4UserTrackinAction
  - G4UserSteppingAction



classes written in red  
are mandatory!

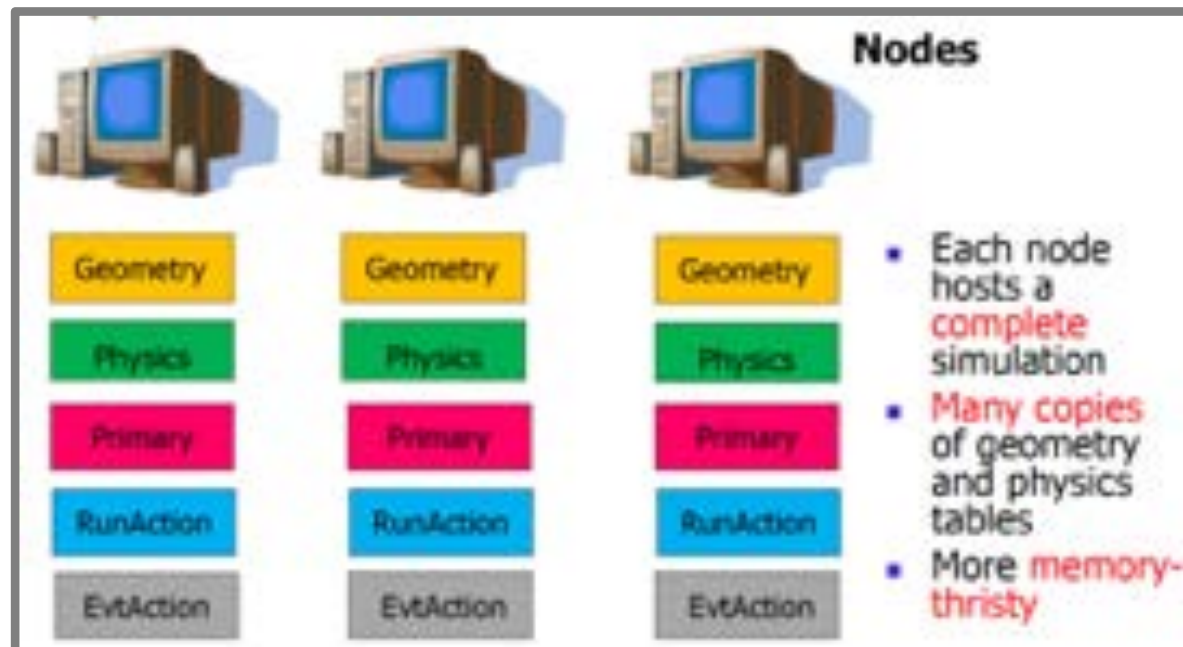
# Your program

---

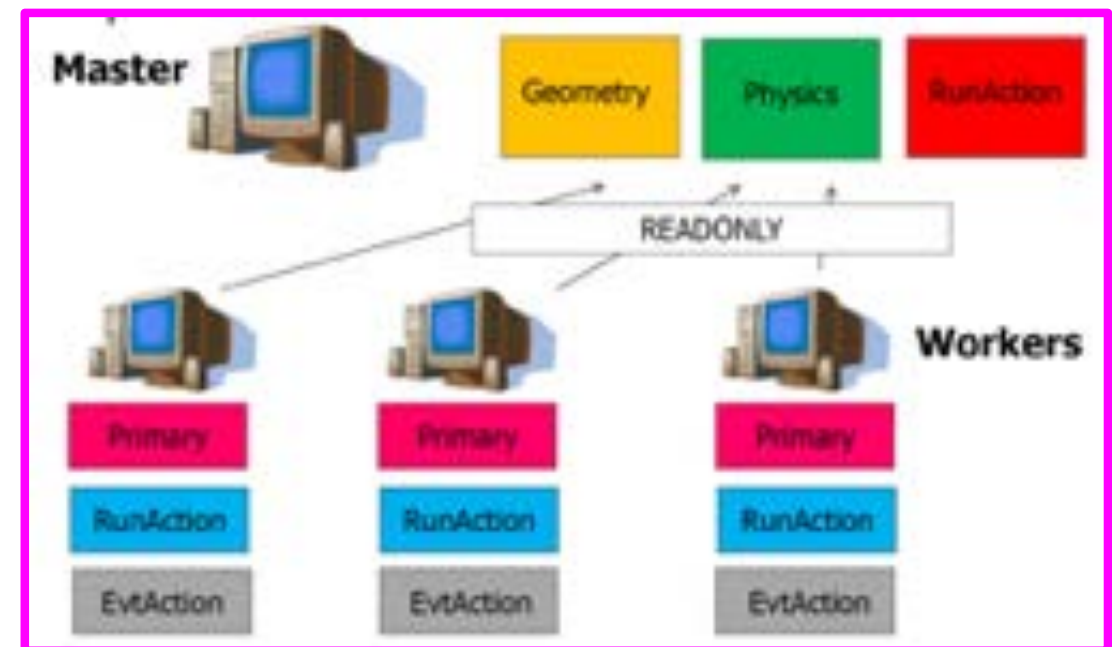
- Geant4 does not provide a main()
- In your main(), you have to
- Construct `G4RunManager` (sequential mode)
- Set user mandatory initialisation classes to `RunManager`
  - `G4VUserDetectorConstruction`
  - `G4VUserPhysicsList`
  - `G4VUserActionInitialization`
- You can define `VisManager`, (G)UI session,
- You can initialise optional user action classes

# Multi-threading in Giant

## Parallel



## Multithread



- Speed-up of simulations even on a laptop
- More efficient usage of memory: all cores only read geometry and physics of the Geant4 simulation (no duplication)



How to survive in the  
real life

Some tips, you will find your way...



# The Geant4 website

- [www.geant4.org](http://www.geant4.org)

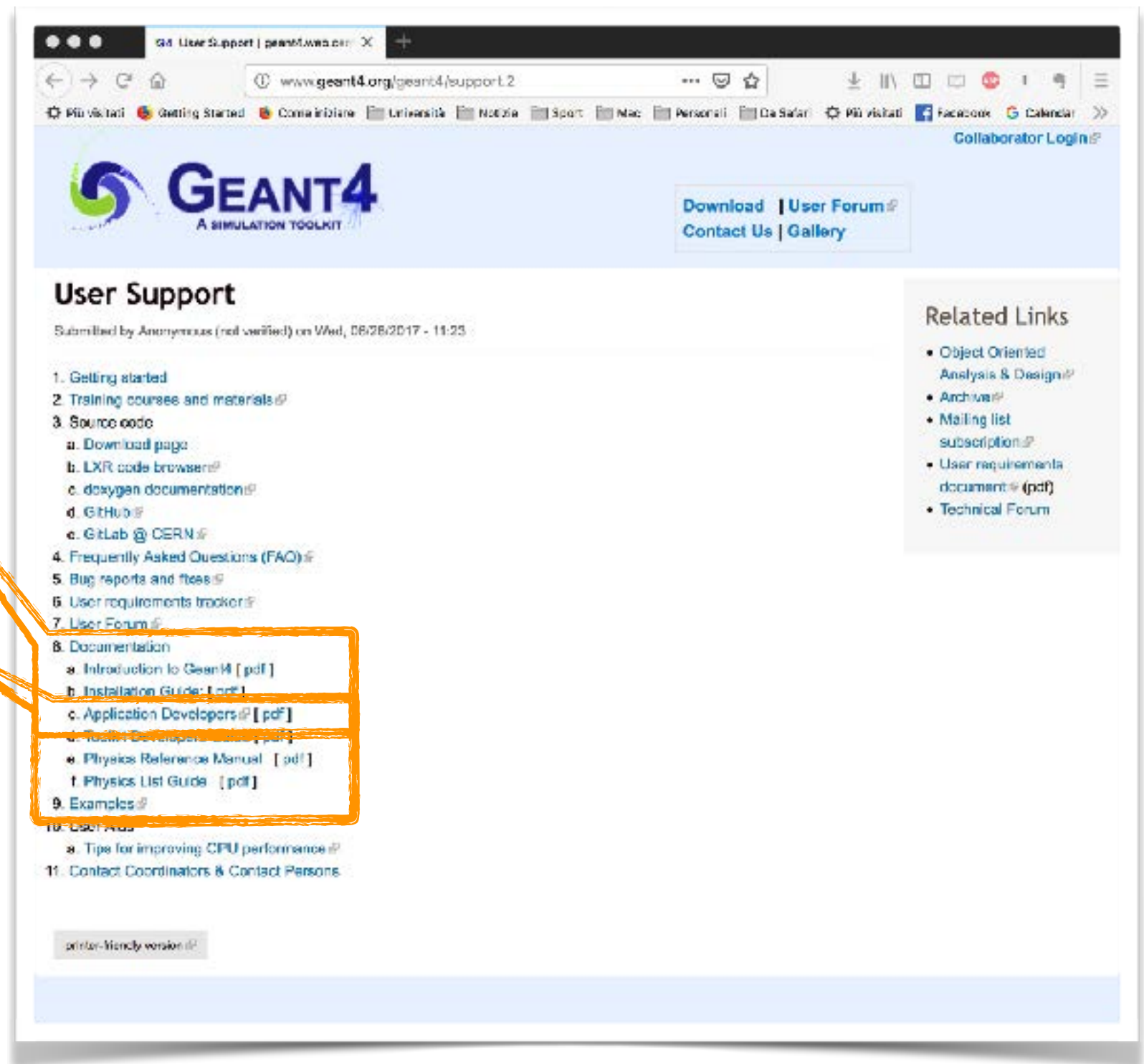
- Future and past events (e.g.: this course)

- User support (one of the best friends for Geant4 users...)



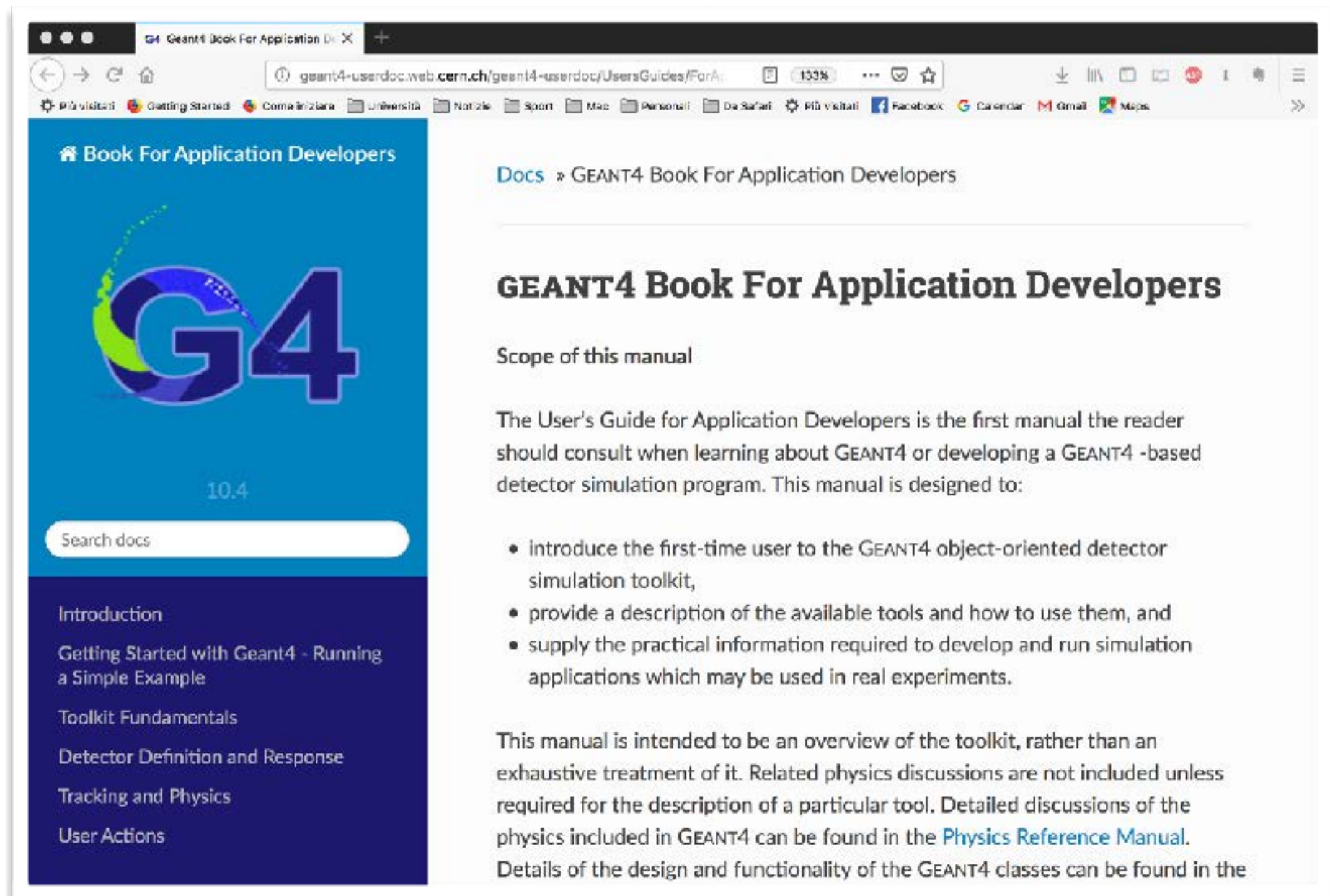
# The Geant4 website

- [www.geant4.org/geant4/support.2](http://www.geant4.org/geant4/support.2)
- Documentation
- Application Developers (you!) guide





# Application Developers user guide



The screenshot shows a web browser window displaying the GEANT4 Book For Application Developers. The browser's address bar shows the URL `geant4-userdoc.web.cern.ch/geant4-userdoc/UsersGuides/ForApp.../10.4`. The page has a blue header with the GEANT4 logo and the text "Book For Application Developers". Below the header, there is a search bar labeled "Search docs". The main content area is titled "GEANT4 Book For Application Developers" and includes a section "Scope of this manual". This section states that the User's Guide for Application Developers is the first manual the reader should consult when learning about GEANT4 or developing a GEANT4-based detector simulation program. It lists three bullet points: introduce the first-time user to the GEANT4 object-oriented detector simulation toolkit, provide a description of the available tools and how to use them, and supply the practical information required to develop and run simulation applications which may be used in real experiments. The page also includes a sidebar with a list of topics: Introduction, Getting Started with Geant4 - Running a Simple Example, Toolkit Fundamentals, Detector Definition and Response, Tracking and Physics, and User Actions.

Book For Application Developers

GEANT4

10.4

Search docs

Introduction

Getting Started with Geant4 - Running a Simple Example

Toolkit Fundamentals

Detector Definition and Response

Tracking and Physics

User Actions

Docs » GEANT4 Book For Application Developers

## GEANT4 Book For Application Developers

### Scope of this manual

The User's Guide for Application Developers is the first manual the reader should consult when learning about GEANT4 or developing a GEANT4 -based detector simulation program. This manual is designed to:

- introduce the first-time user to the GEANT4 object-oriented detector simulation toolkit,
- provide a description of the available tools and how to use them, and
- supply the practical information required to develop and run simulation applications which may be used in real experiments.

This manual is intended to be an overview of the toolkit, rather than an exhaustive treatment of it. Related physics discussions are not included unless required for the description of a particular tool. Detailed discussions of the physics included in GEANT4 can be found in the [Physics Reference Manual](#). Details of the design and functionality of the GEANT4 classes can be found in the

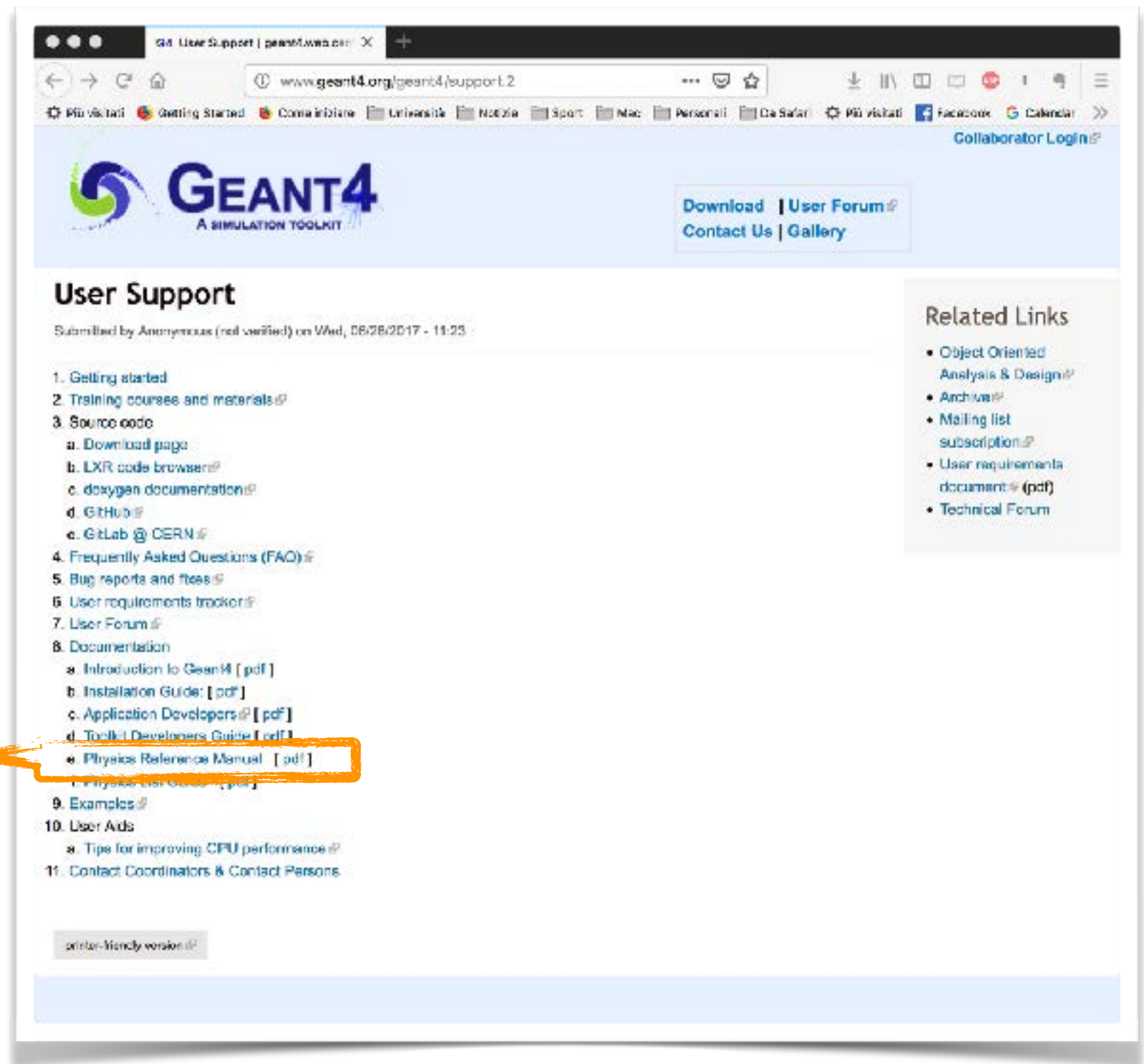
# Application Developers user guide

---

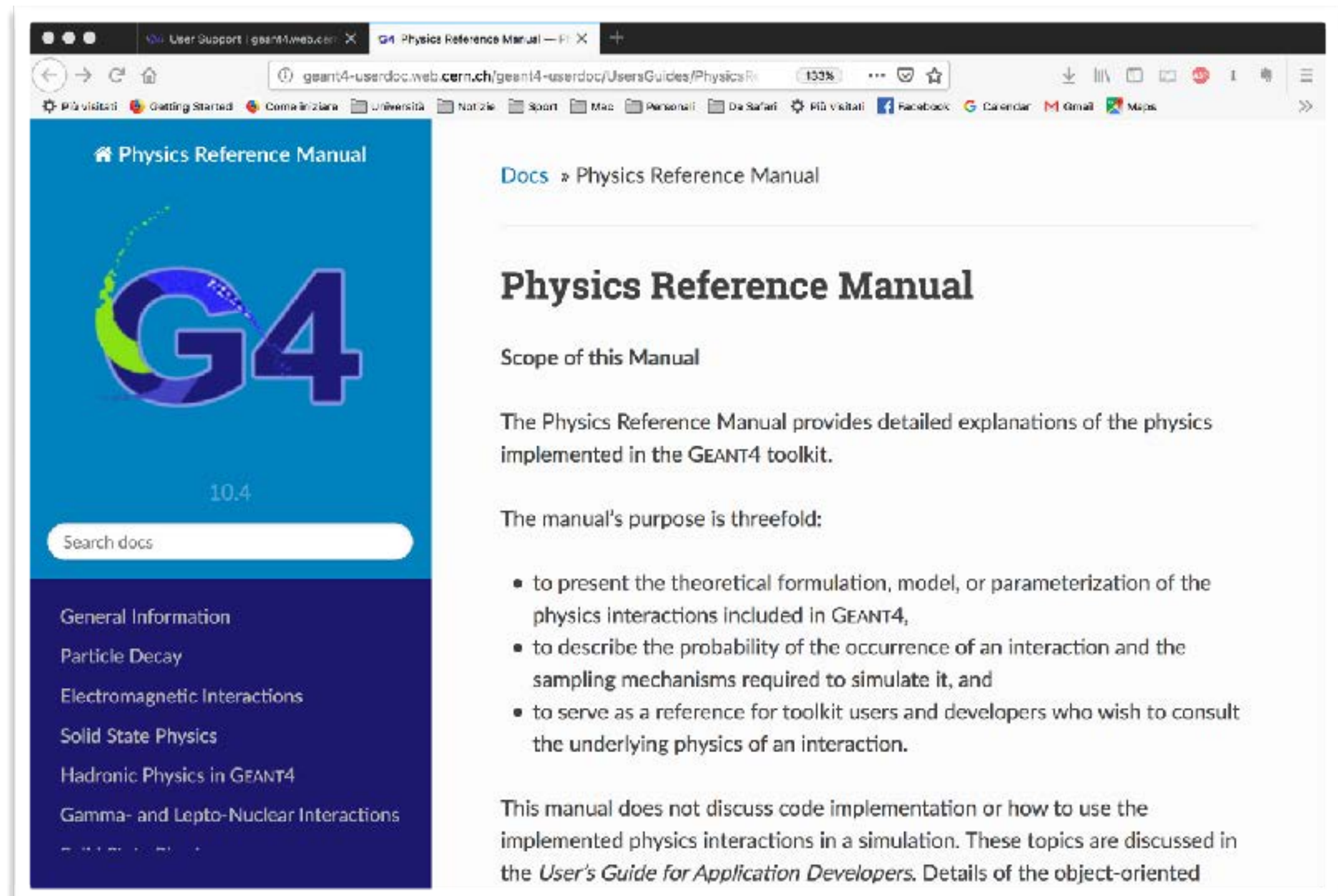
- <http://geant4-userdoc.web.cern.ch/geant4-userdoc/UsersGuides/ForApplicationDeveloper/html/index.html>
- Introduces new Users to the Geant4 toolkit
- Describes the most useful tools
- Describes how to set-up and run a simulation application
- Intended as an overview of the toolkit, not an exhaustive treatment

# The Geant4 website

- [www.geant4.org](http://www.geant4.org)
- Documentation
- Application Developers (you!) guide
- Physics Reference Manual



# Physics Reference Manual





# Physics Reference Manual

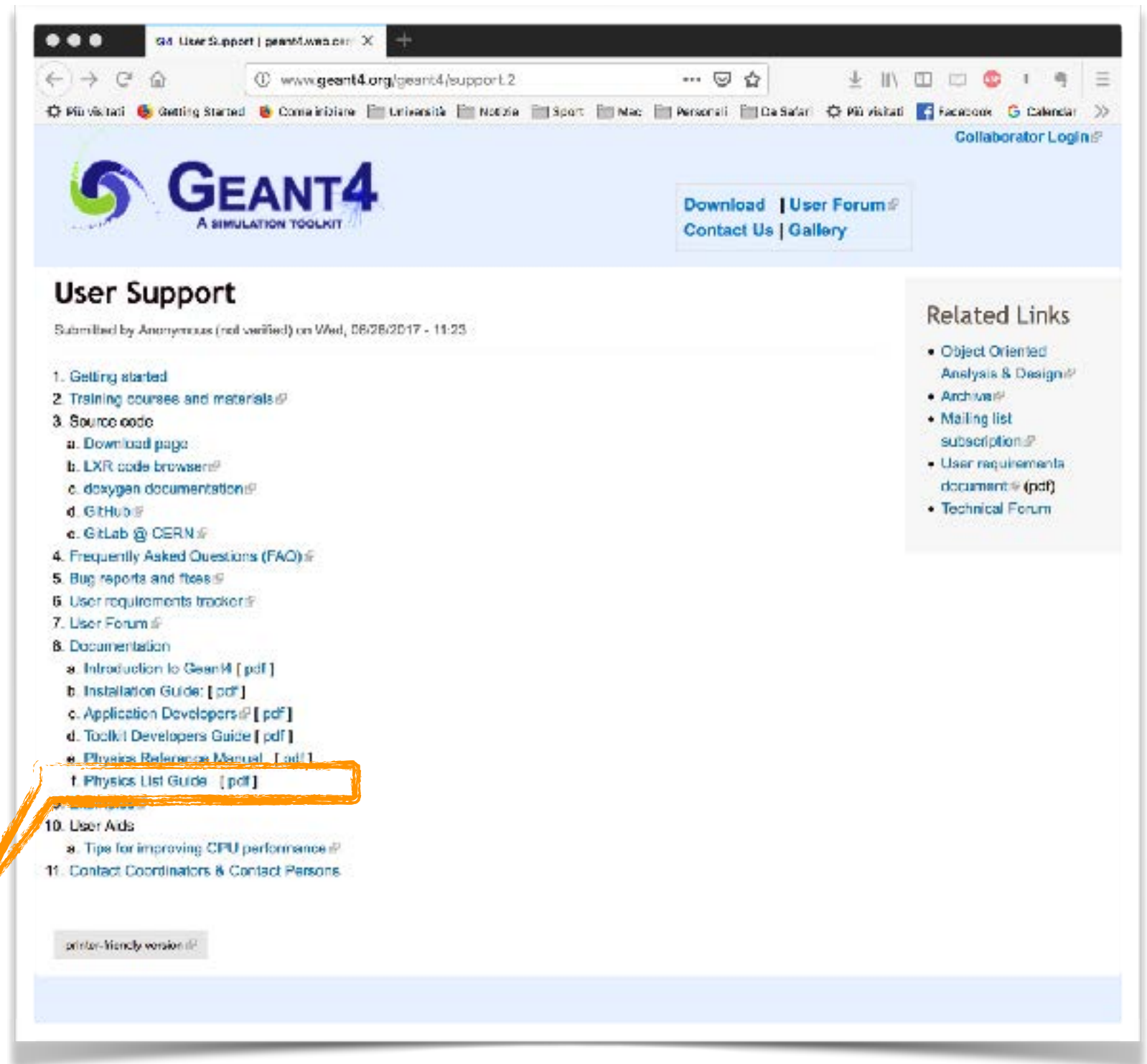
---

- <http://geant4-userdoc.web.cern.ch/geant4-userdoc/UsersGuides/PhysicsReferenceManual/html/index.html>
- A reference for toolkit Users and developers who wish to consult and study the physics of an interaction/model
- Present the theoretical formulation, model or parameterisation of the physics interactions provided by Geant4



# The Geant4 website

- [www.geant4.org](http://www.geant4.org)
- Documentation
- Application Developers (you!) guide
- Physics Reference Manual
- Physics Lists Guide



# Physics List Guide

PhysicsListGuide

10.4

Search docs

CONTENTS:

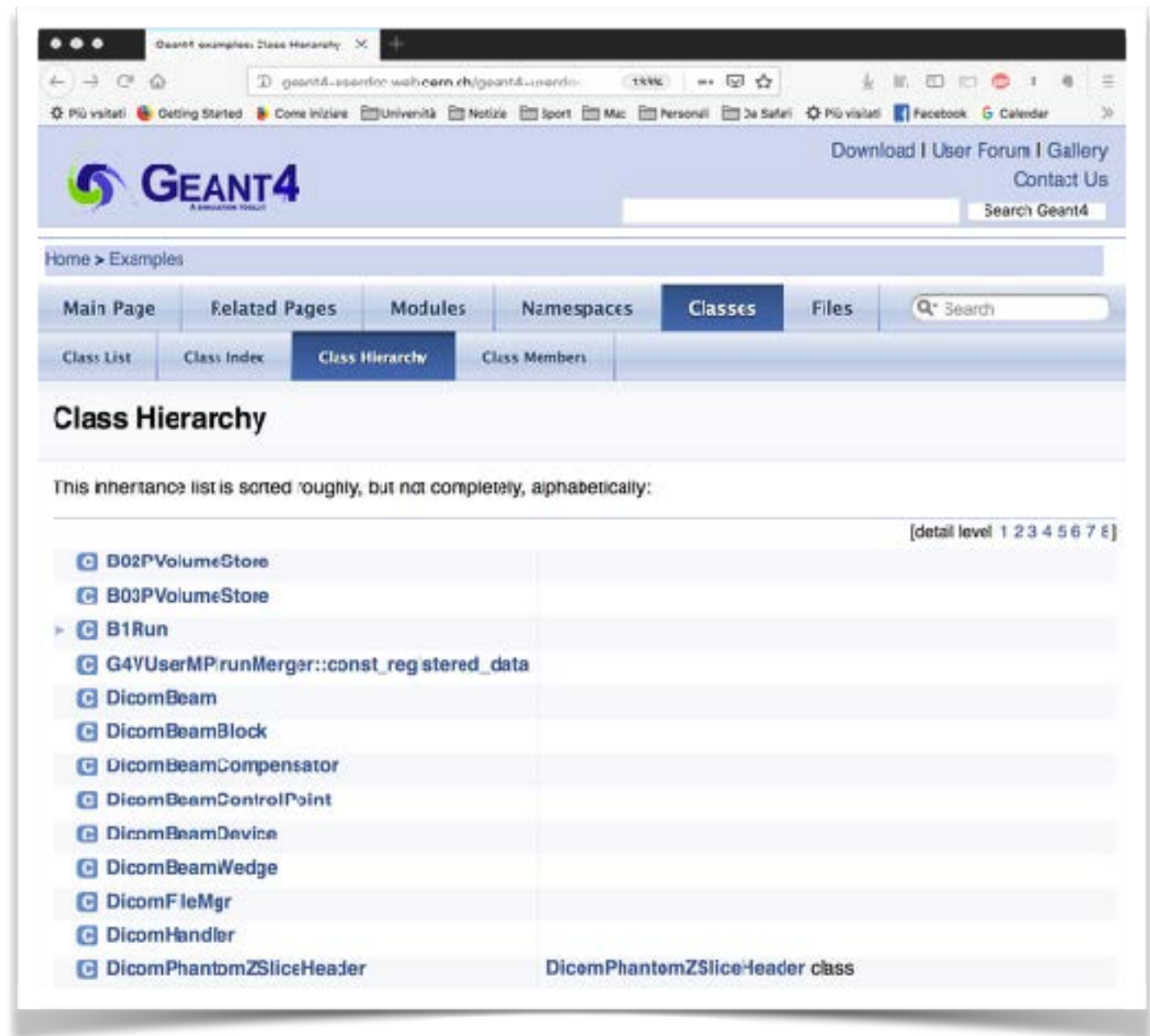
- Physics List Guide
- Reference Physics Lists
- Electromagnetic physics constructors

Contents:

- Physics List Guide
  - Bibliography
- Reference Physics Lists
  - FTFP\_BERT Physics List
  - QBBC Physics List
  - QGSP\_BERT Physics List
  - Shielding Physics List
- Electromagnetic physics constructors
  - EM physics constructors
  - EM Opt0
  - EM Opt1
  - EM Opt2
  - EM Opt3
  - EM Opt4
  - EM Liv
  - EM Pen
  - EM GS
  - EM SS
  - EM DNA
  - Tables by constructor
  - Tables by particle

# Doxygen

- [http://geant4-userdoc.web.cern.ch/geant4-userdoc/Doxygen/examples\\_doc/html/hierarchy.html](http://geant4-userdoc.web.cern.ch/geant4-userdoc/Doxygen/examples_doc/html/hierarchy.html)
- All the class interfaces







Examples

an overview...

# Examples omnia divisa est in partes trees...

---

- **Basic** set of examples is oriented to novice users and covering the most typical use-cases of a Geant4 application with keeping simplicity and ease of use
- **Extended** set of examples may require some additional libraries besides of Geant4. This set covers many specific use cases for actual detector simulation
- **Advanced** set of examples covers the use-cases typical of a "toolkit"- oriented kind of development, where real complete applications for different simulation studies are provided; may require additional third party products to be built

# Where?

---

- Where to find the examples:
  - `$G4DIR/examples/basic`
  - `$G4DIR/examples/extended`
  - `$G4DIR/examples/advanced`



# Basic examples

| Code name  | Few Characteristics                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Example B1 | <ul style="list-style-type: none"> <li>• Simple geometry with a few solids</li> <li>• Geometry with simple placements (<a href="#">G4PVPlacement</a>)</li> <li>• Scoring total dose in a selected volume user action classes</li> <li>• Geant4 physics list (<a href="#">QBBC</a>)</li> </ul>                                                                                                                                                                                                                                                                                                                     |
| Example B2 | <ul style="list-style-type: none"> <li>• Simplified tracker geometry with global constant magnetic field</li> <li>• Geometry with simple placements (<a href="#">G4PVPlacement</a>) and parameterisation (<a href="#">G4PVParameterisation</a>)</li> <li>• Scoring within tracker via G4 sensitive detector and hits</li> <li>• Geant4 physics list (FTFP_BERT) with step limiter</li> <li>• Started from novice/N02 example</li> </ul>                                                                                                                                                                           |
| Example B3 | <ul style="list-style-type: none"> <li>• Schematic Positron Emitted Tomography system</li> <li>• Geometry with simple placements with rotation (<a href="#">G4PVPlacement</a>)</li> <li>• Radioactive source</li> <li>• Scoring within Crystals via G4 scorers</li> <li>• Modular physics list built via builders provided in Geant4</li> </ul>                                                                                                                                                                                                                                                                   |
| Example B4 | <ul style="list-style-type: none"> <li>• Simplified calorimeter with layers of two materials</li> <li>• Geometry with replica (<a href="#">G4PVReplica</a>)</li> <li>• Scoring within layers in four ways: via user actions, via user own objects via G4 sensitive detector and hits and via scorers</li> <li>• Geant4 physics list (FTFP_BERT)</li> <li>• Histograms (1D) and ntuple saved in the output file</li> <li>• Started from novice/N03 example</li> </ul>                                                                                                                                              |
| Example B5 | <ul style="list-style-type: none"> <li>• A double-arm spectrometer with wire chambers, hodoscopes and calorimeters with a local constant magnetic field</li> <li>• Geometry with placements with rotation, replicas and parameterisation</li> <li>• Scoring within wire chambers, hodoscopes and calorimeters via G4 sensitive detector and hits</li> <li>• Geant4 physics list (FTFP_BERT) with step limiter</li> <li>• UI commands defined using <a href="#">G4GenericMessenger</a></li> <li>• Histograms (1D, 2D) and ntuple saved in the output file</li> <li>• Started from extended/analysis/A01</li> </ul> |

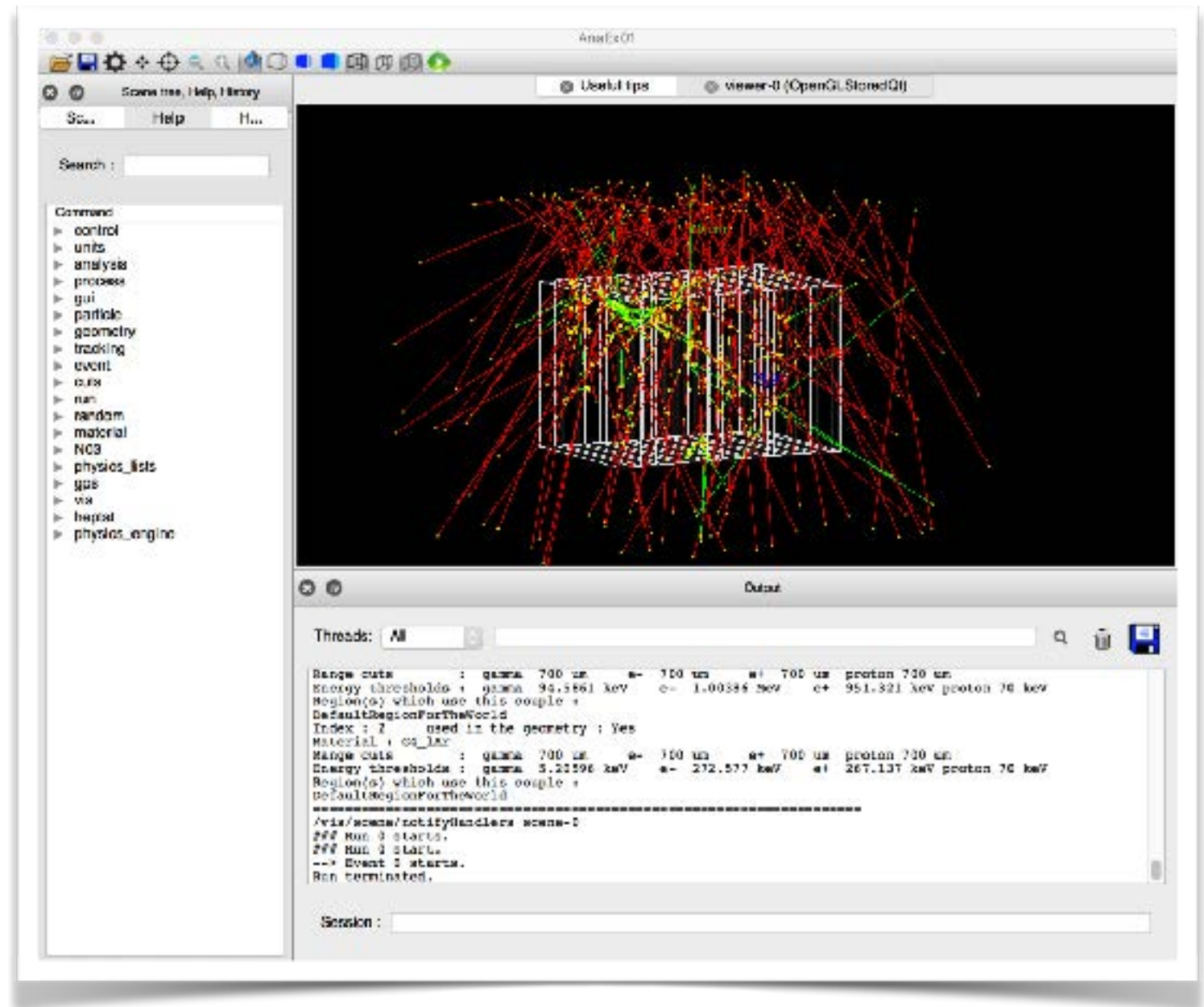


# Basic examples

| Code name                                                                                                 | Few Characteristics                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|-----------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <div data-bbox="57 605 186 925">Basic!</div> <div data-bbox="263 605 598 670">Example B1</div>            | <ul style="list-style-type: none"> <li>• Simple geometry with a few solids</li> <li>• Geometry with simple placements (<a href="#">G4PVPlacement</a>)</li> <li>• Scoring total dose in a selected volume user action classes</li> <li>• Geant4 physics list (<a href="#">QBBC</a>)</li> </ul>                                                                                                                                                                                                                                                                                                                            |
| <div data-bbox="263 829 606 895">Example B2</div>                                                         | <ul style="list-style-type: none"> <li>• Simplified tracker geometry with global constant magnetic field</li> <li>• Geometry with simple placements (<a href="#">G4PVPlacement</a>) and parameterisation (<a href="#">G4PVParameterisation</a>)</li> <li>• Scoring within tracker via G4 sensitive detector and hits</li> <li>• Geant4 physics list (FTFP_BERT) with step limiter</li> <li>• Started from novice/N02 example</li> </ul>                                                                                                                                                                                  |
| <div data-bbox="57 1116 186 1794">A bit complex</div> <div data-bbox="263 1075 606 1140">Example B3</div> | <ul style="list-style-type: none"> <li>• <b>Schematic Positron Emitted Tomography system</b></li> <li>• Geometry with simple placements with rotation (<a href="#">G4PVPlacement</a>)</li> <li>• Radioactive source</li> <li>• Scoring within Crystals via G4 scorers</li> <li>• Modular physics list built via builders provided in Geant4</li> </ul>                                                                                                                                                                                                                                                                   |
| <div data-bbox="263 1351 606 1416">Example B4</div>                                                       | <ul style="list-style-type: none"> <li>• Simplified calorimeter with layers of two materials</li> <li>• Geometry with replica (<a href="#">G4PVReplica</a>)</li> <li>• Scoring within layers in four ways: via user actions, via user own objects via G4 sensitive detector and hits and via scorers</li> <li>• Geant4 physics list (FTFP_BERT)</li> <li>• Histograms (1D) and ntuple saved in the output file</li> <li>• Started from novice/N03 example</li> </ul>                                                                                                                                                     |
| <div data-bbox="263 1688 606 1753">Example B5</div>                                                       | <ul style="list-style-type: none"> <li>• <b>A double-arm spectrometer with wire chambers, hodoscopes and calorimeters with a local constant magnetic field</b></li> <li>• Geometry with placements with rotation, replicas and parameterisation</li> <li>• Scoring within wire chambers, hodoscopes and calorimeters via G4 sensitive detector and hits</li> <li>• Geant4 physics list (FTFP_BERT) with step limiter</li> <li>• UI commands defined using <a href="#">G4GenericMessenger</a></li> <li>• Histograms (1D, 2D) and ntuple saved in the output file</li> <li>• Started from extended/analysis/A01</li> </ul> |

# My example

- <https://github.com/carlomt/AnaEx01>
- A modified version of extended/AnaEx01
- output in root files
- input in GPS
- example of cosmic muons...



thank you for your attention!