esipap
European School of Instrumentation
in Particle & Astroparticle Physics

esi
European Scientific Institute

Detector Simulation Introduction

Witek Pokorski
Alberto Ribon
CERN

10-11.02.2020

Content

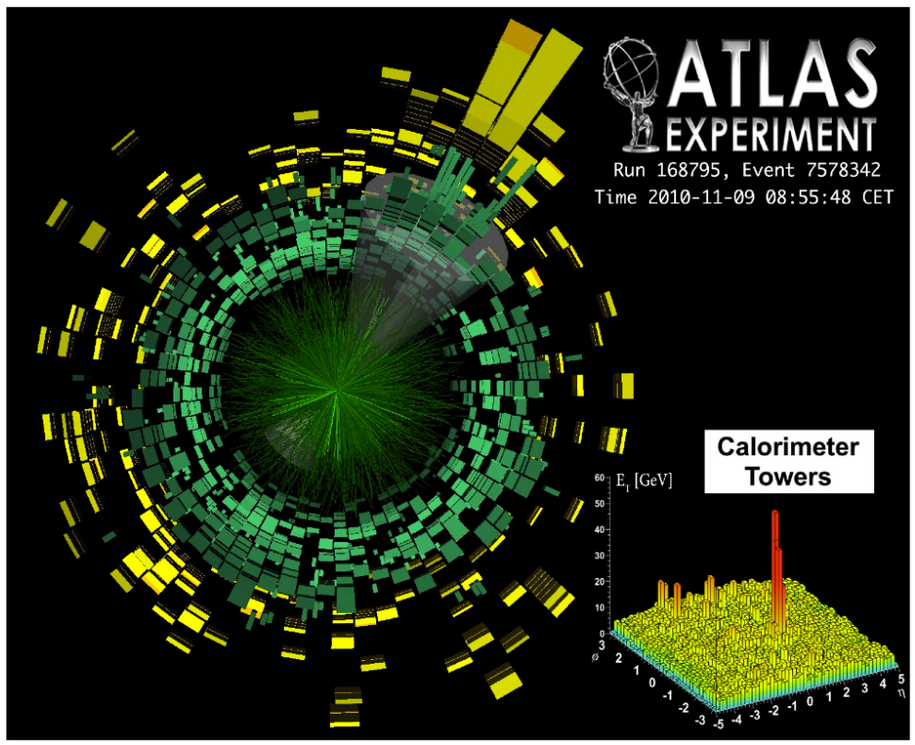
- Introduction to Monte Carlo methods and to Geant4 toolkit
- Geant4 toolkit architecture and physics
- Building a concrete Geant4 application

Introduction to Monte Carlo methods

- What problems do we face?
- What is simulation?
- Why do we need Monte Carlo?

Problems

- to design the apparatus (detector) to fulfil its role
- to run it according to our needs
- to understand the results
- to observe new phenomena



Solution

- SIMULATION
  - we need to simulate the detector before we build it
  - we need to simulate it when we run it
  - we need to simulate it when we analyse results

What is simulation

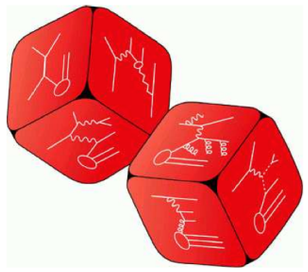
- simulation = doing ‘virtual’ experiment
- take all the known physics
- start from your ‘initial condition’ (two protons colliding)
- calculate the ‘final state’ of your detector to get the ‘experimental’ results
  - solve equations of motion, etc
- can we do it analytically? No…!

Way to do it: Monte Carlo

- analytical solutions impossible due to
  - complexity of the problem
    - number of particles, etc
  - lack of analytical description
  - need of randomness like in nature
    - to pick among possible choices
      - quantum mechanics, fluctuations
        - Quantum mechanics amplitudes = probabilities. Anything that possibly can happen, will. (but more or less often)

Monte Carlo method

- repeated random sampling to find the result
- used for
  - deterministic problems like numerical integration, calculation, solving systems of differential equations, etc



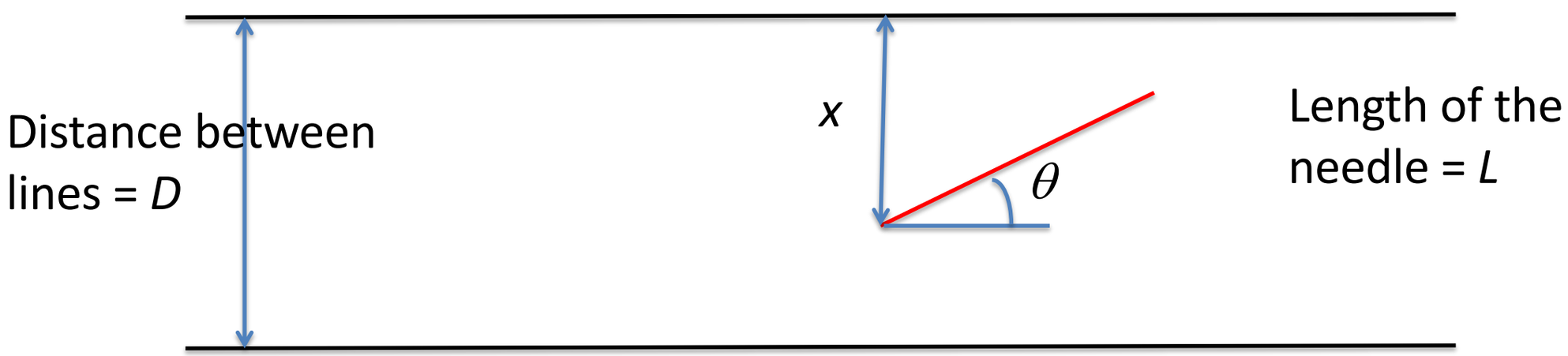
  - stochastic problems involving generation of samples from probability distribution ⇐ our use case
- applications: particle physics, quantum field theory, astrophysics, molecular modelling, semiconductor devices, light transport calculations, traffic flow simulations, environmental sciences, financial market simulations, optimisation problems, etc

Buffon's Needle

- One of the oldest problems in the field of geometrical probability, first stated in 1777.
- Drop a needle on a lined sheet of paper and determine the probability of the needle crossing one of the lines
- Remarkable result: probability is directly related to the value of **π**
- The needle will cross the line if $x \leq L \sin(\theta)$. Assuming $L \leq D$, how often will this occur?

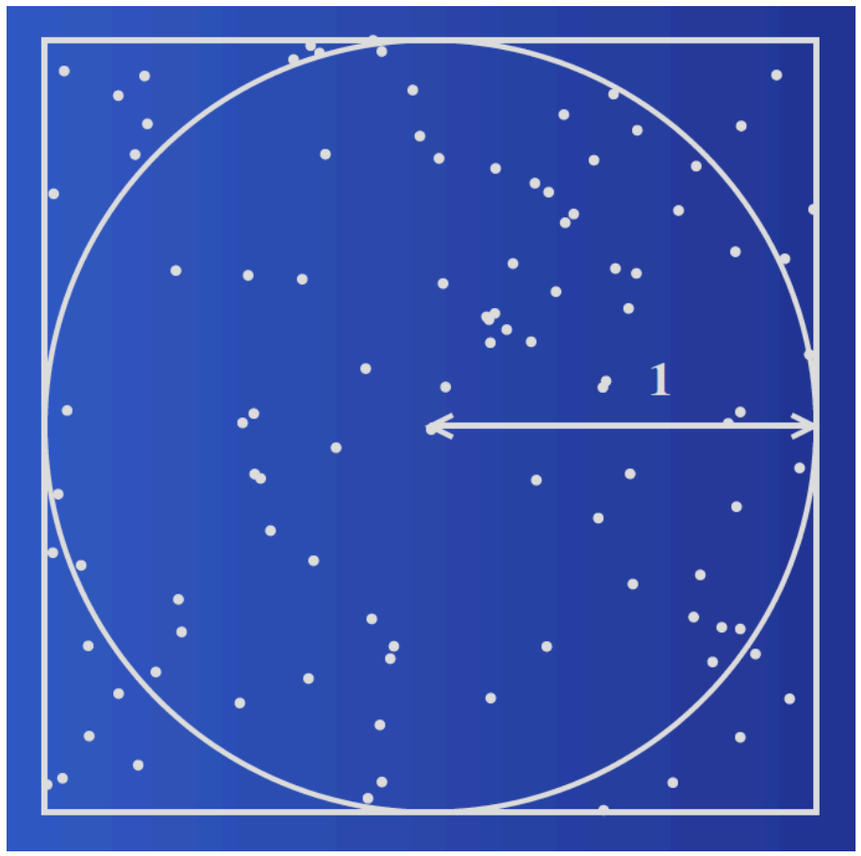
$$P_{cut} = \int_0^{\pi} P_{cut}(\theta)\frac{d\theta}{\pi} = \int_0^{\pi} \frac{L\sin\theta}{D}\frac{d\theta}{\pi} = \frac{L}{\pi D}\int_0^{\pi} \sin\theta\, d\theta = \frac{2L}{\pi D}$$

- By sampling P_{cut} one can estimate **π**.



Laplace's method of calculating **π** (1886)

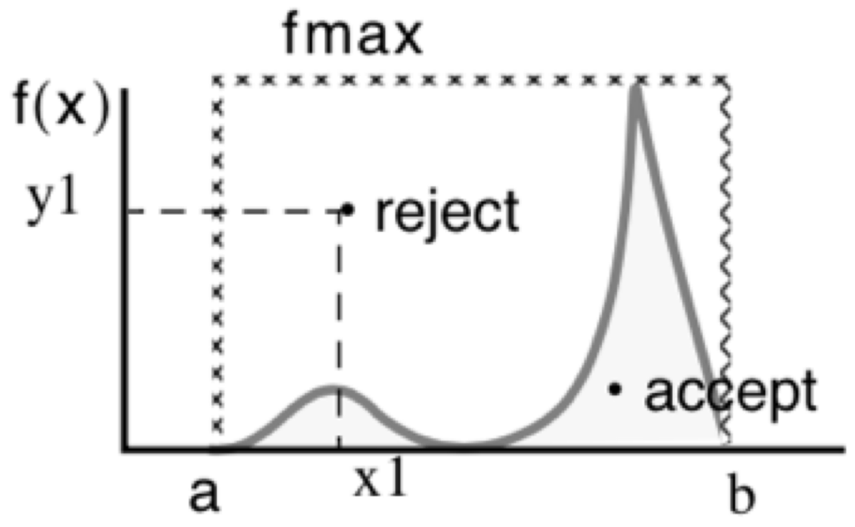
- Area of the square = 4
- Area of the circle = **π**
- Probability of random points inside the circle = **π** / 4



- Random points : N
- Random points inside circle : N_c

$$\pi \sim 4\, N_c / N$$

- can be easily generalized for integral calculation of an arbitrary function



Monte Carlo methods for radiation transport

- Fermi (1930): random method to calculate the properties of the newly discovered neutron

- Manhattan project (40's): simulations during the initial development of thermonuclear weapons. von Neumann and Ulam coined the term "Monte Carlo"

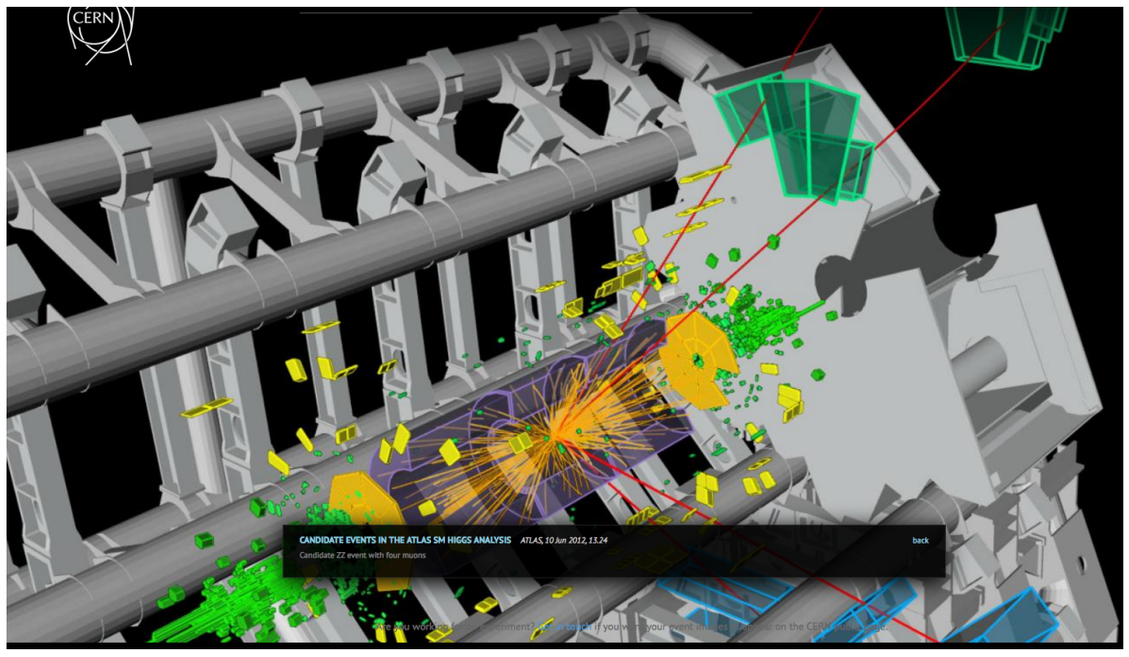
- Metropolis (1948) first actual Monte Carlo calculations using a computer (ENIAC)

- Berger (1963): first complete coupled electron-photon transport code that became known as ETRAN

- Exponential growth since the 1980's with the availability of digital computers

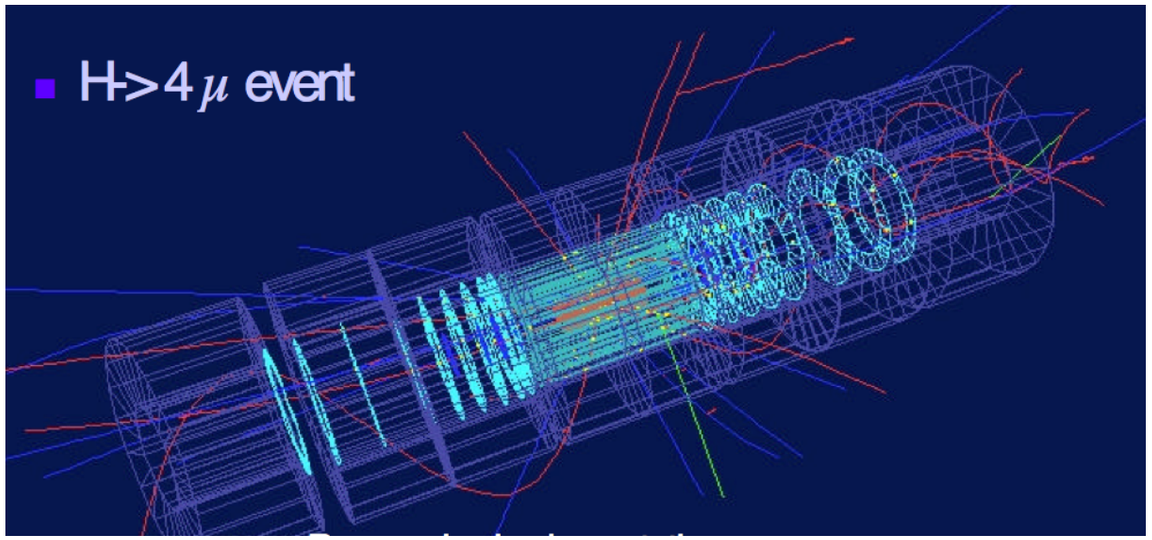
Monte Carlo in HEP

- Simulation is an essential tool in modern particle physics for:
  - prediction event rates and topologies
    - can estimate feasibility
  - evaluation of possible backgrounds
    - can devise analysis strategies
    - study detector requirements
  - optimisation detector/trigger design
  - studying detector imperfections
    - can evaluate acceptance corrections
  - analysing the data

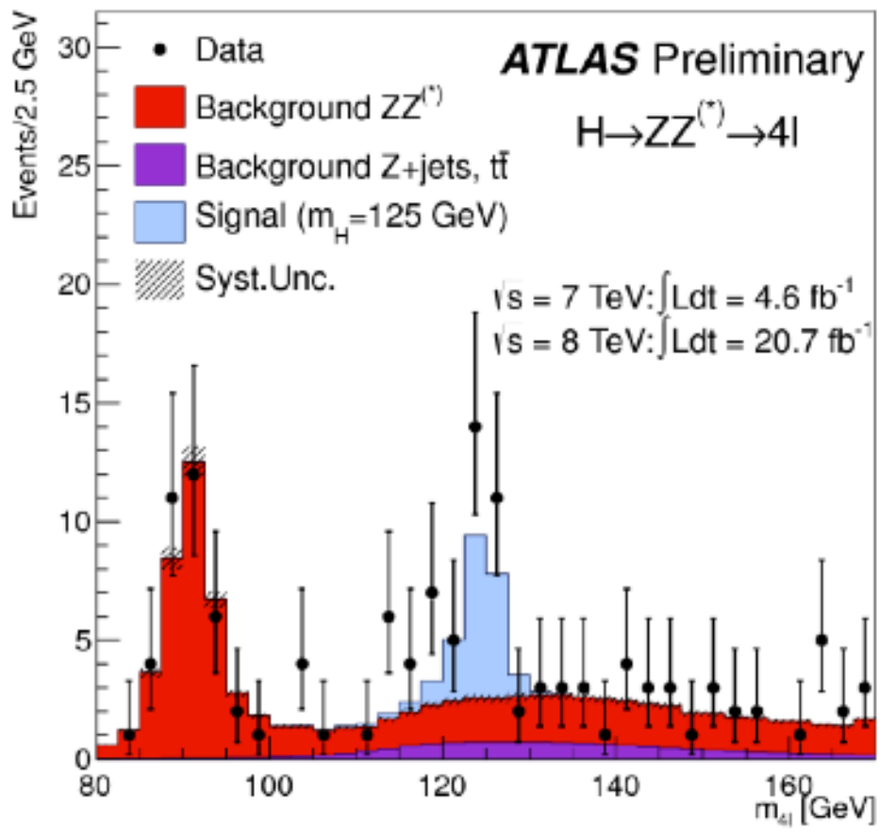


Simulation is needed to make discoveries

- We need to understand the detector to do physics



- We need to know what to expect to
  - verify existing models
  - find new physics



Non-exhaustive list of Monte Carlo codes

- EM physics
  - ETRAN (Berger & Seltzer; NIST)
  - EGS4 (Nelson, Hirayama, Rogers; SLAC)
  - EGS5 (Hirayama et al.; KEK/SLAC)
  - EGSnrc (Kawrakow & Rogers; NRCC)
  - Penelope (Salvat et al.; U. Barcelona)
- Hadronic physics / general purpose
  - Fluka (Ferrari et al., CERN/INFN)
  - Geant4 (Geant4 Collaboration)
  - MARS (James & Mokhov; FNAL)
  - MCNPX / MCNP5 (LANL)
  - PHITS (Niita et al.; JAEA)

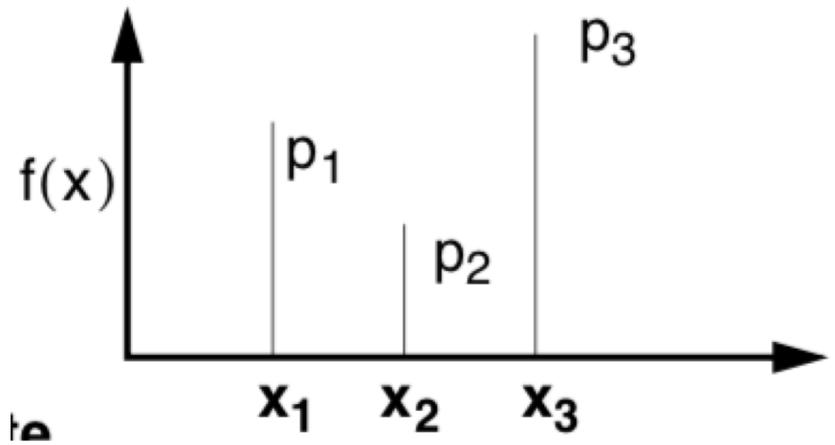
Some theory

Stochastic processes

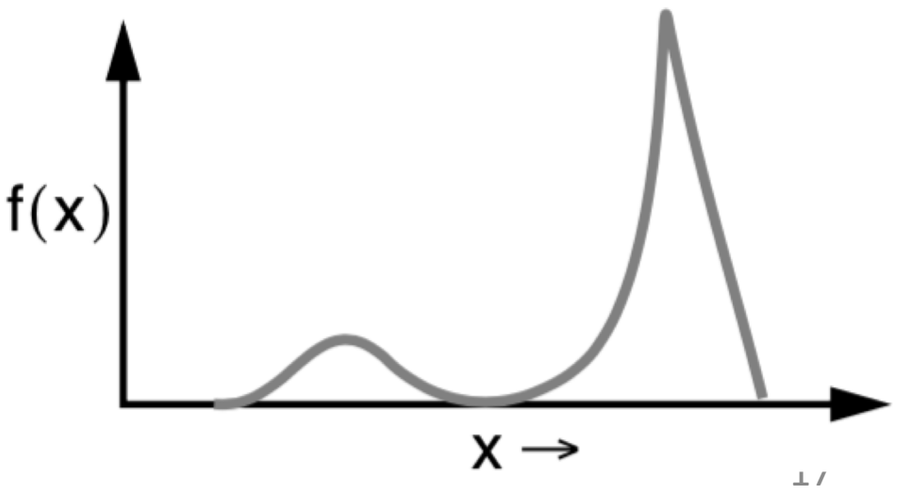
- stochastic (random) process are described by means of probabilities
    - result cannot be specified in advance of observing it
    - discrete processes
        - selecting a decay channel, throwing a die
    - continuous processes
        - decay time of an unstable particle

Probability distribution

- Natural laws give us probability distributions
  - discrete
    - probability for each discrete event to happen



  - continuous (Probability Density Function or PDF)
    - probability ‘density’ as function of the variable
      - integral over an interval gives the probability for that interval of values

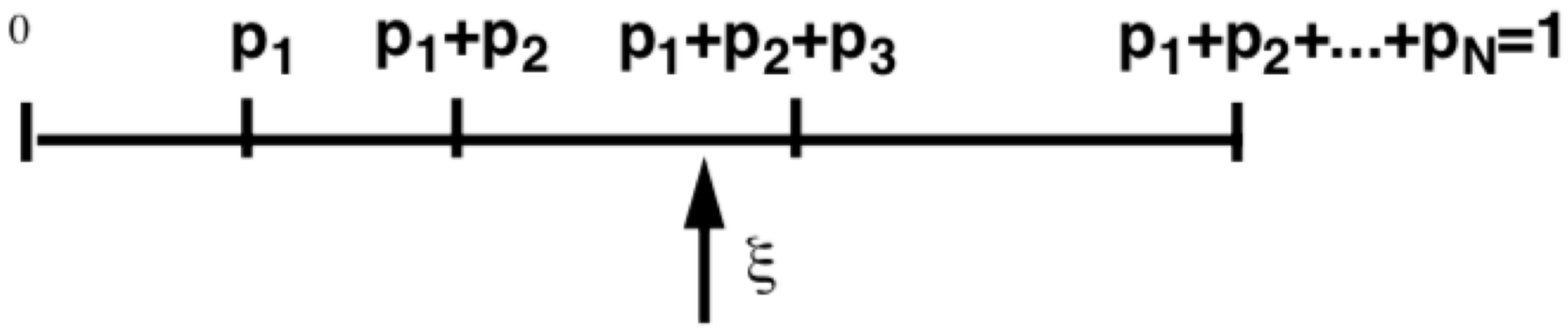


Sampling

- the role of Monte Carlo is to reproduce those natural laws, i.e. to reproduce the same probability distributions
- we want to generate samples that would follow the same distributions
  - this is called sampling of PDFs

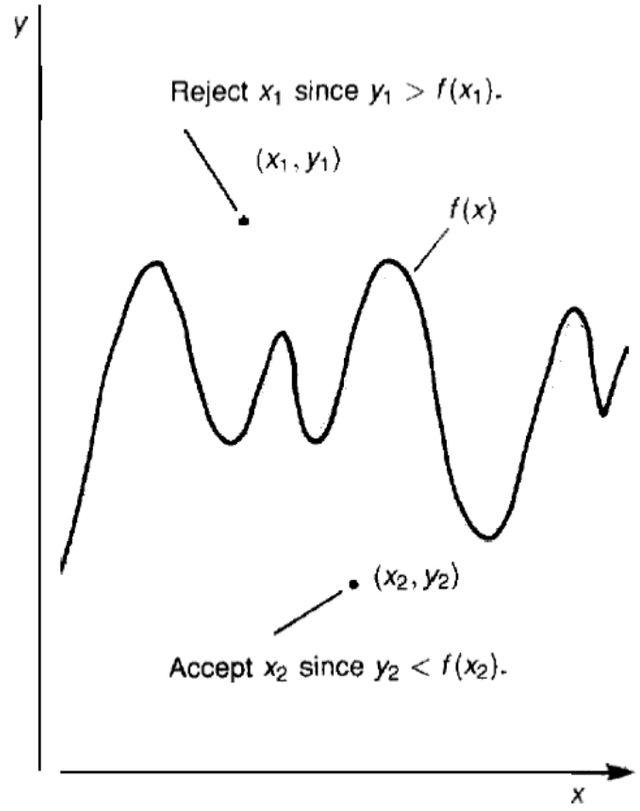
Discrete sampling

- split [0,1] into intervals corresponding to discrete probabilities
- generate random number $\xi \in [0,1]$
- look which interval it falls in
  - different generated ‘events’ will be distributed according to their probabilities



Acceptance-rejection method

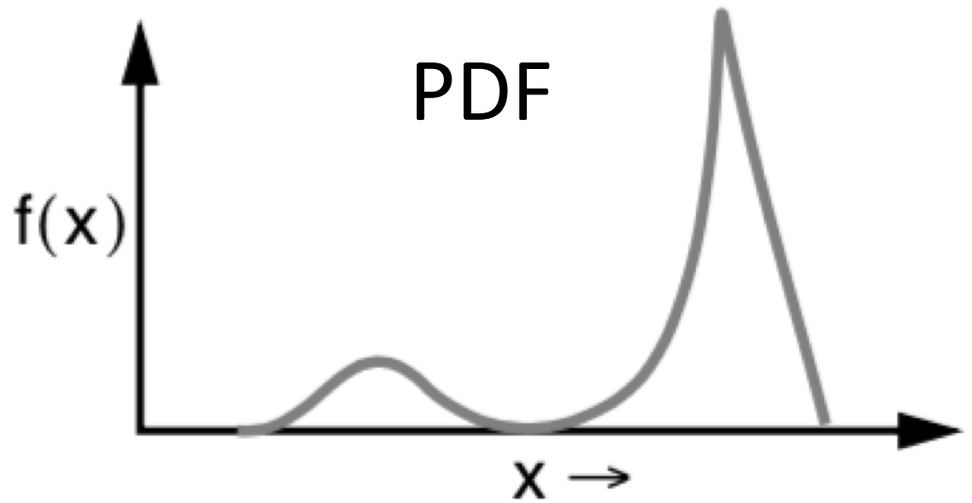
- If x_i and y_i are selected randomly from the range and domain, respectively, of the function f, then each pair of numbers represents a point in the function's coordinate plane (x, y)
- When $y_i > f(x_i)$ the point lies above the curve for $f(x)$, and x_i is rejected; when $y_i \leq f(x_i)$ the points lies on or below the curve, and x_i is accepted
- Thus, fraction of accepted points is equal to fraction of the area below curve
- This technique, first proposed by von Neumann, is also known as the acceptance-rejection method of generating random numbers for arbitrary Probability Density Function (PDF)



Direct Sampling

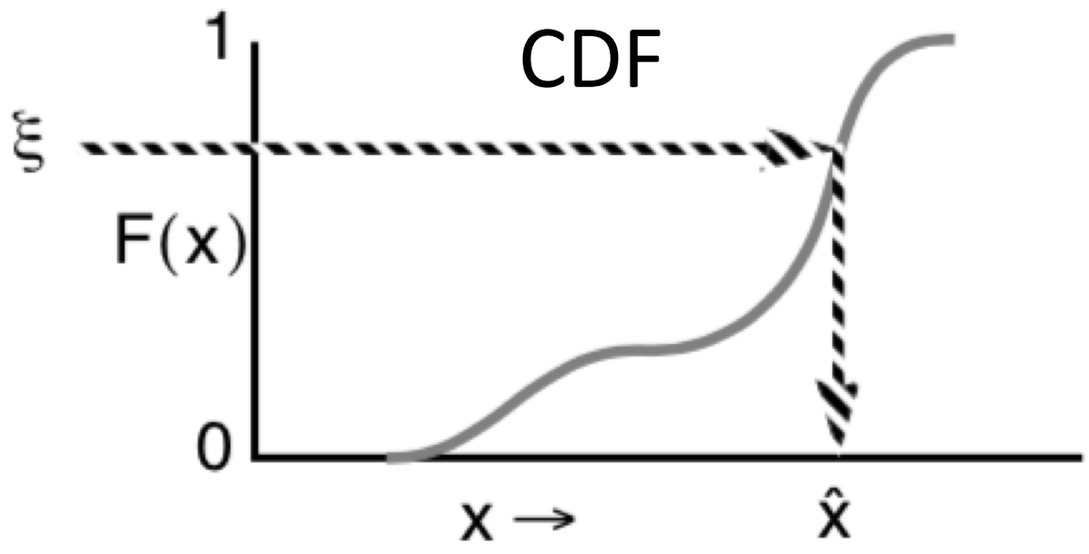
- if $f(x)$ is a PDF over the interval $[a, b]$ we can define a Cumulative Distribution Function (CDF) by

$$F(x) \equiv \int_a^x f(x')\,dx'$$



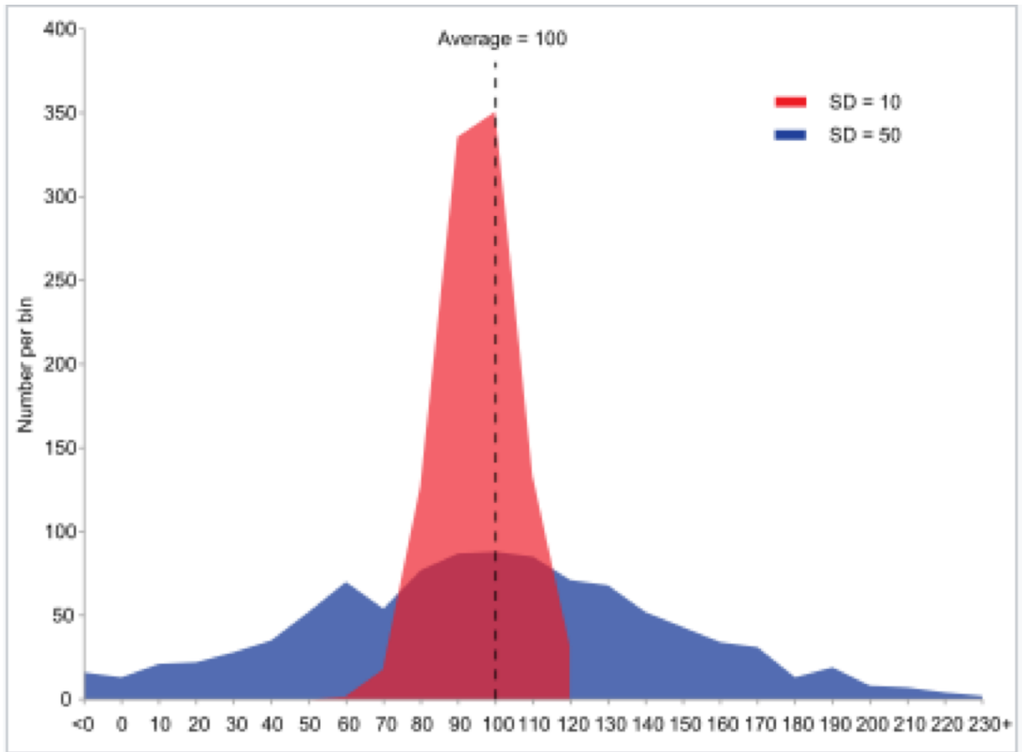
- CDF is a direct measure of probability Prob$\{a \le x \le x_i\} = F(x_i)$
- if we generate random number $\xi \in [0,1]$ we can get

$$\hat{x} = F^{-1}(\xi)$$



Stochastic variables

- stochastic variables don’t have a ‘value’, but a ‘mean value’
- measurements give values around the mean
  - variance measures how big is the spread
    - standard deviation is square root of variance



$$\langle x \rangle \equiv E(x) \equiv \mu(x) \equiv \int_a^b x f(x)\, dx.$$

$$\sigma^2(x) \equiv \langle [x - \langle x \rangle]^2 \rangle = \int_a^b [x - \langle x \rangle]^2 f(x)\, dx.$$

Confidence of results

- Monte Carlo gives the estimate of the expected value AND the uncertainty of the estimate
- Central Limit Theorem (CLT) tells us how our samples means (generated ones) are distributed around the distribution mean (‘real’ one)

$$\lim_{N\to\infty} \text{Prob}\left\{\frac{|\bar{z}-\langle z\rangle|}{\sigma(z)/\sqrt{N}} \le \lambda\right\} = \frac{1}{\sqrt{2\pi}}\int_{-\lambda}^{\lambda} e^{-u^2/2}\,du.$$

- For a given λ in unit of standard deviation, the Monte Carlo estimate of <z> is usually reported as

$$\bar{z} \pm \lambda s(z)/\sqrt{N}.$$

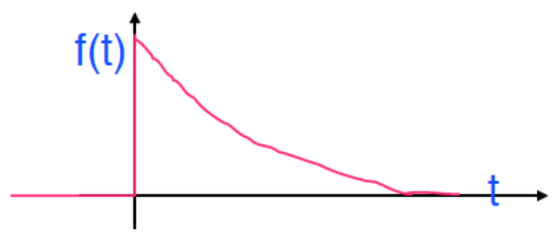
λ	confidence coefficient	confidence level
0.25	0.1974	20%
0.50	0.3829	38%
1.00	0.6827	68%
1.50	0.8664	87%
2.00	0.9545	95%
3.00	0.9973	99%
4.00	0.9999	99.99%

Simplest case – decay in flight (1)

- Suppose an unstable particle of life time t has initial momentum p (→ velocity v).
  - Distance to travel before decay : d = t v
- The decay time t is a random value with probability density function

$$f(t) = \frac{1}{\tau}\exp\left(-\frac{t}{\tau}\right) \quad t \geq 0$$

τ is the mean life of the particle



- the probability that the particle decays before time t is given by the cumulative distribution function F which is itself is a random variable with uniform probability on [0,1]

$$r = F(t) = \int_{-\infty}^{t} f(u)du$$

- Thus, having a uniformly distributed random number r on [0,1], one can sample the value t with the probability density function f(t).

$$t = F^{-1}(r) = -\tau \ln(1 - r) \qquad 0 \leq r < 1$$

Simplest case – decay in flight (2)

- When the particle has traveled the d = t v, it decays.
- Decay of an unstable particle itself is a random process → Branching ratio
  - For example:

$\pi^+ \rightarrow \mu^+ \nu_\mu$ (99.9877 %)

$\pi^+ \rightarrow \mu^+ \nu_\mu \gamma$ (2.00×10^{-4} %)

$\pi^+ \rightarrow e^+ \nu_e$ (1.23×10^{-4} %)

$\pi^+ \rightarrow e^+ \nu_e \gamma$ (7.39×10^{-7} %)

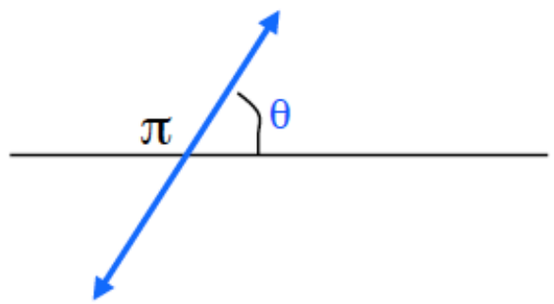
$\pi^+ \rightarrow e^+ \nu_e \pi^0$ (1.036×10^{-8} %)

$\pi^+ \rightarrow e^+ \nu_e e^+ e^-$ (3.2×10^{-9} %)

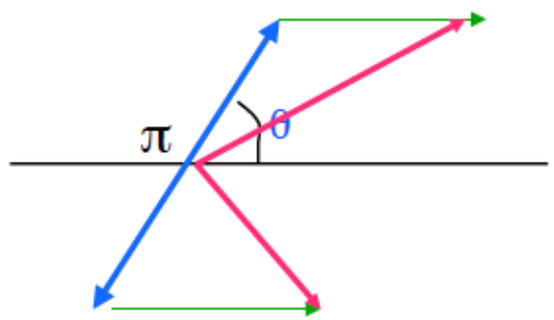
- Select a decay channel by shooting a random number
- In the rest frame of the parent particle, rotate decay produc in θ [0,π) and ϕ *[0,2π)* by shooting a pair of random numbers

$$d\Omega = \sin\theta \, d\theta \, d\phi$$

$$\theta = \cos^{-1}(r_1), \quad \phi = 2\pi \times r_2 \qquad 0 \le r_1, r_2 < 1$$



- Finally, Lorentz-boost the decay products
- You need at least 4 random numbers to simulate one decay in flight



Compton scattering (1)

- Compton scattering

  $e^- \gamma \rightarrow e^- \gamma$

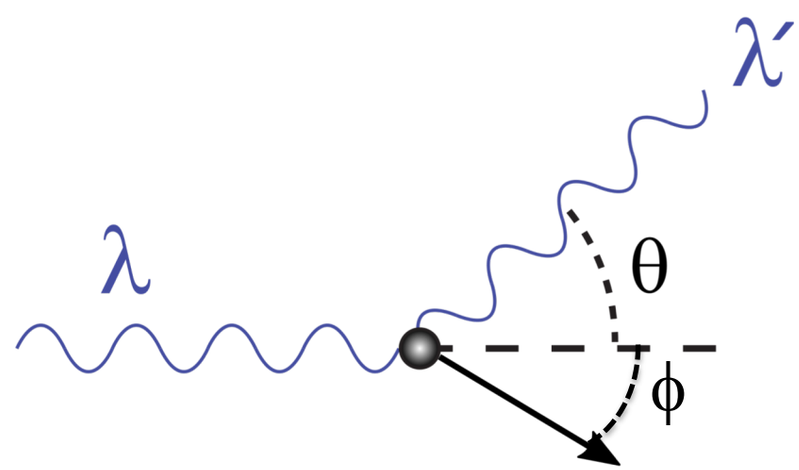
- Distance traveled before Compton scattering, l, is a random value

  Cross section per atom : $\sigma(E,z)$

  Number of atoms per volume : $n = \rho N_A / A$

  ρ : density, N_A : Avogadro number, A : atomic mass

  Cross section per volume : $\eta(E, \rho) = n \sigma$



- η is the probability of Compton interaction per unit length. $\lambda(E, \rho) = \eta^{-1}$ is the mean free path associated with the Compton scattering process.
- The probability density function $f(l)$

$$f(l) = \eta \exp(-\eta l) = \frac{1}{\lambda} \exp\left(-\frac{l}{\lambda}\right)$$

- With a uniformly distributing random number r on [0,1), One can sample the distance l.

$$l = -\lambda \ln(r) \qquad 0 \le r < 1$$

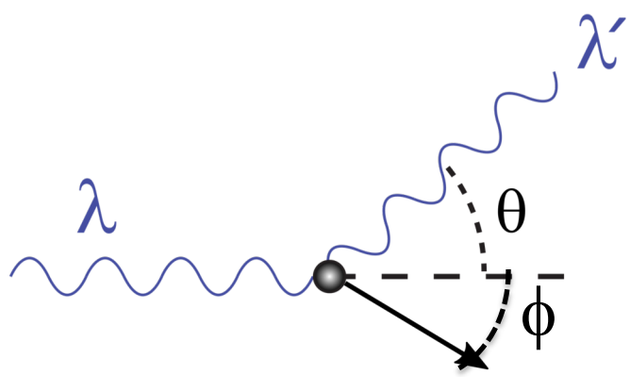
Compton scattering (2)

- The relation between photon deflection (θ) and energy loss for Compton scattering is determined by the conservation of momentum and energy between the photon and recoiled electron.

$$h\nu = \frac{h\nu_0}{1 + \left(\frac{h\nu_0}{m_e c^2}\right)(1 - \cos\theta)},$$

$$E = h\nu_0 - h\nu = m_e c^2 \frac{2(h\nu_0)^2 \cos^2\phi}{(h\nu_0 + m_e c^2)^2 - (h\nu_0)^2 \cos^2\phi},$$

$$\tan\phi = \frac{1}{1 + \left(\frac{h\nu_0}{m_e c^2}\right)} \cot\frac{\theta}{2},$$



$h\nu$: energy of incident photon
$h\nu_0$: energy of scattered photon
E : energy of recoil electron
m_e : rest mass of electron
c : speed of light

- For unpolarized photon, the Klein-Nishina angular distribution function per steradian of solid angle Ω

$$\frac{d\sigma_c^{KN}}{d\Omega}(\theta) = r_0^2 \frac{1 + \cos^2\theta}{2} \frac{1}{[1 + h\nu(1 - \cos\theta)]^2} \left\{1 + \frac{h\nu^2(1 - \cos\theta)^2}{(1 + \cos^2\theta)[1 + h\nu(1 - \cos\theta)]}\right\}$$

$$= \frac{1}{2} r_0^2 \left(\frac{k}{k_0}\right)^2 \left(\frac{k}{k_0} + \frac{k_0}{k} - \sin^2\theta\right) \qquad (cm^2 sr^{-1} electron^{-1}),$$

$$k_0 = \frac{h\nu_0}{m_e c^2}, \qquad k = \frac{h\nu}{m_e c^2}$$

- One can use acceptance-rejection method to sample the distribution.

Monte Carlo in LHC experiments

- For the LHC experiments, the simulation is made two distinct steps:

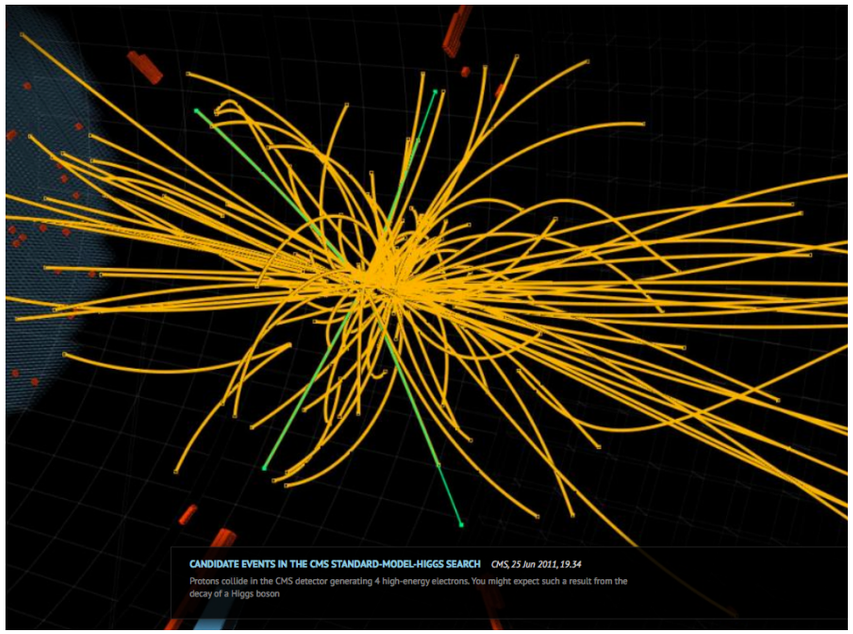
  1. Simulation of the p-p collision

     - Monte Carlo event generators

  2. Simulation of the passage of the produced particles through the experimental apparatus

     - Monte Carlo radiation transportation, or simply “detector simulation”

- The output of 1. is the input of 2.



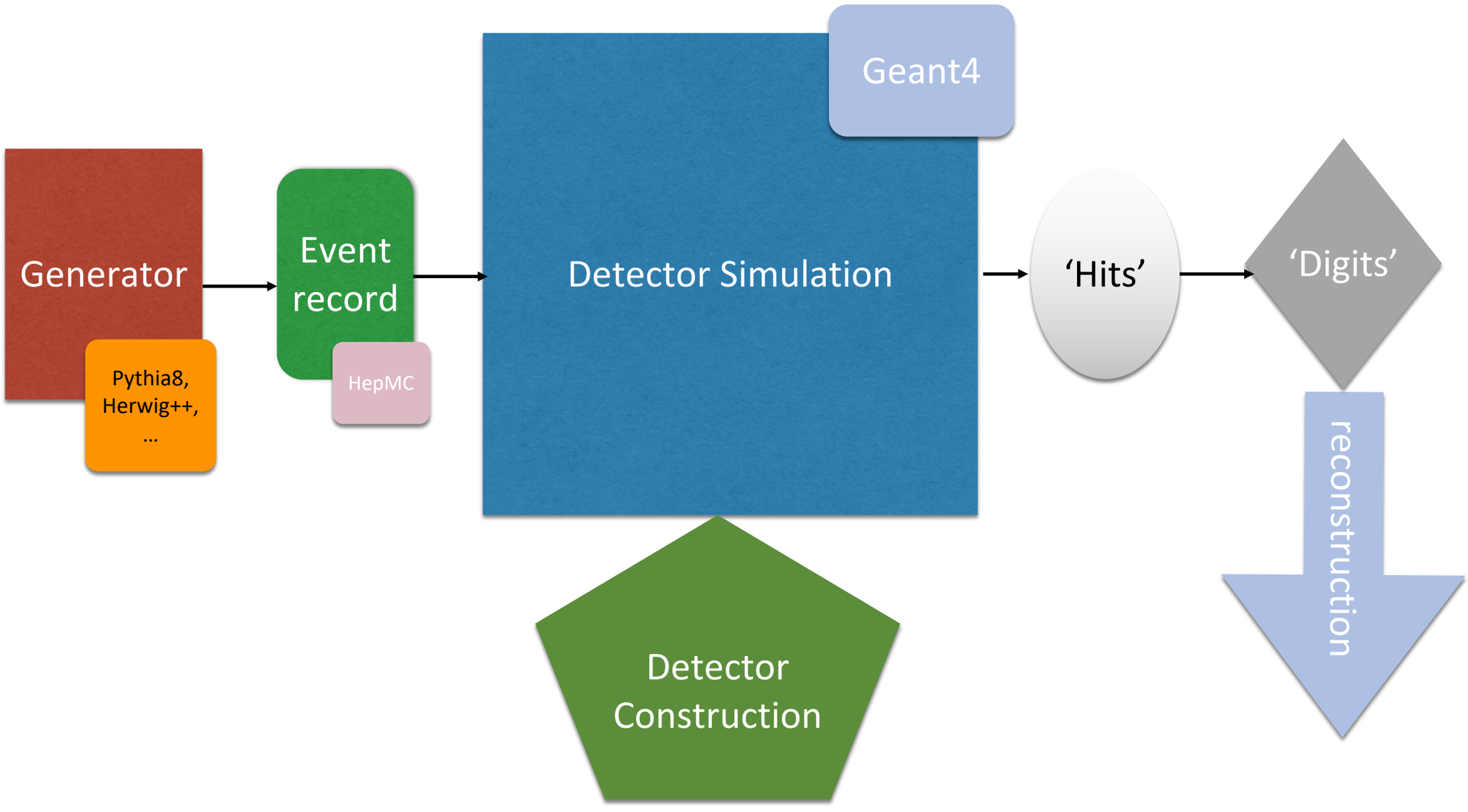
Monte Carlo radiation transportation codes

- The simulation of the p-p collision is the same for different experiments at the same collider, e.g. ATLAS and CMS
- The detector simulation is different for each experiment. However, general codes exist that can be used for simulating any detector
  - An experimental apparatus can be modelled in terms of elementary geometrical objects
  - The physics processes are detector independent
- These general codes, e.g. Geant4, are called "Monte Carlo radiation transportation codes"
  - Non-deterministic (e.g. do not solve equations); use random numbers to reproduce distributions
  - Transport particles through matter

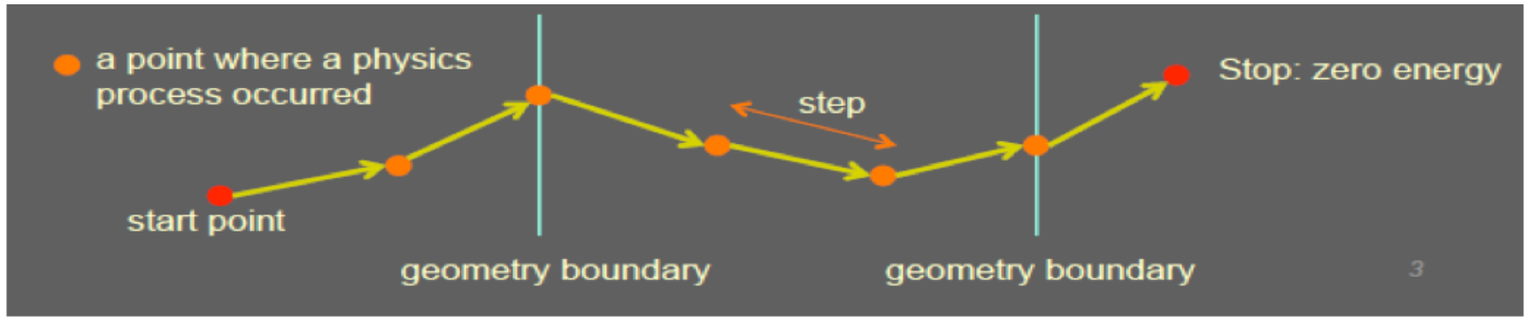
Digitization

- Besides the geometry, another experiment-specific aspect of the detector simulation is the “digitization”
  - It is not part of the general radiation transportation codes
- It consists of producing the detector response in terms of electric current & voltage signals, as it would happen in the real experiment
  - The same reconstruction chain can be applied for both real and simulated data
- The general radiation transportation code provides energy deposits in the whole detector; from these, the “digitization” simulates the electrical signals induced in the sensitive parts of the detector

Simulation chain for HEP experiment



How does detector simulation work?



- Treat one particle at the time
- Treat a particle in steps
- For each step
  - the step length is determined by the cross sections of the physics processes and the geometrical boundaries; if new particles are created, add them to the list of particles to be transported;
  - calculate local energy deposit; effect of magnetic and electric fields;
  - if the particle is destroyed by the interaction, or it reaches the end of the apparatus, or its energy is below a (tracking) threshold, then the simulation of this particle is over; else continue with another step.
- Output
  - new particles created (indirect)
  - local energy deposits throughout the detector (direct)

Accuracy vs speed

- Huge samples (billions) of simulated events are needed by the experiments for their physics analyses
- The number of simulated events is limited by CPU
- The simulation time is dominated by the detector simulation
- Tradeoff between accuracy and speed of the detector simulation
  - More precise physics models are slower and, more importantly, create more secondaries and/or steps
  - Smaller geometrical details slow down the simulation
    - Never model explicitly screws, bolts, cables, etc.
  - Continuous spectrum of types of detector simulations:
    - From full, detailed detector simulations
    - To very fast, fully parametrized detector simulations

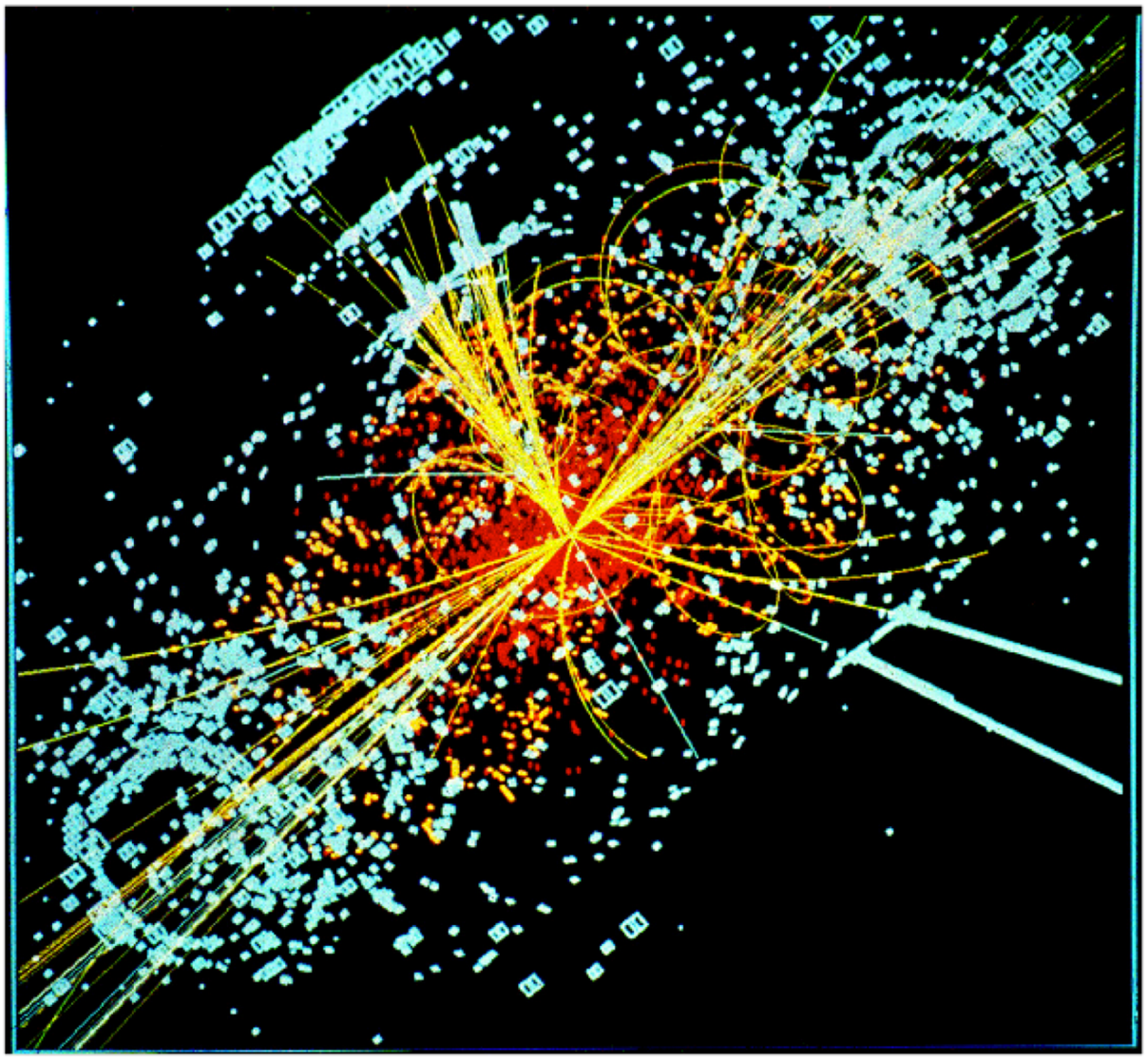
Application domains

- We are considering here mainly high-energy physics, but... There are other domains where the same radiation transportation codes are successfully used:
  - Nuclear physics
  - Accelerator science
  - Astrophysics
  - Space engineering
  - Radiation damage
  - Medical physics
  - Industrial applications
- So, detector simulation is a multi-disciplinary field!

Geant4 toolkit architecture and physics

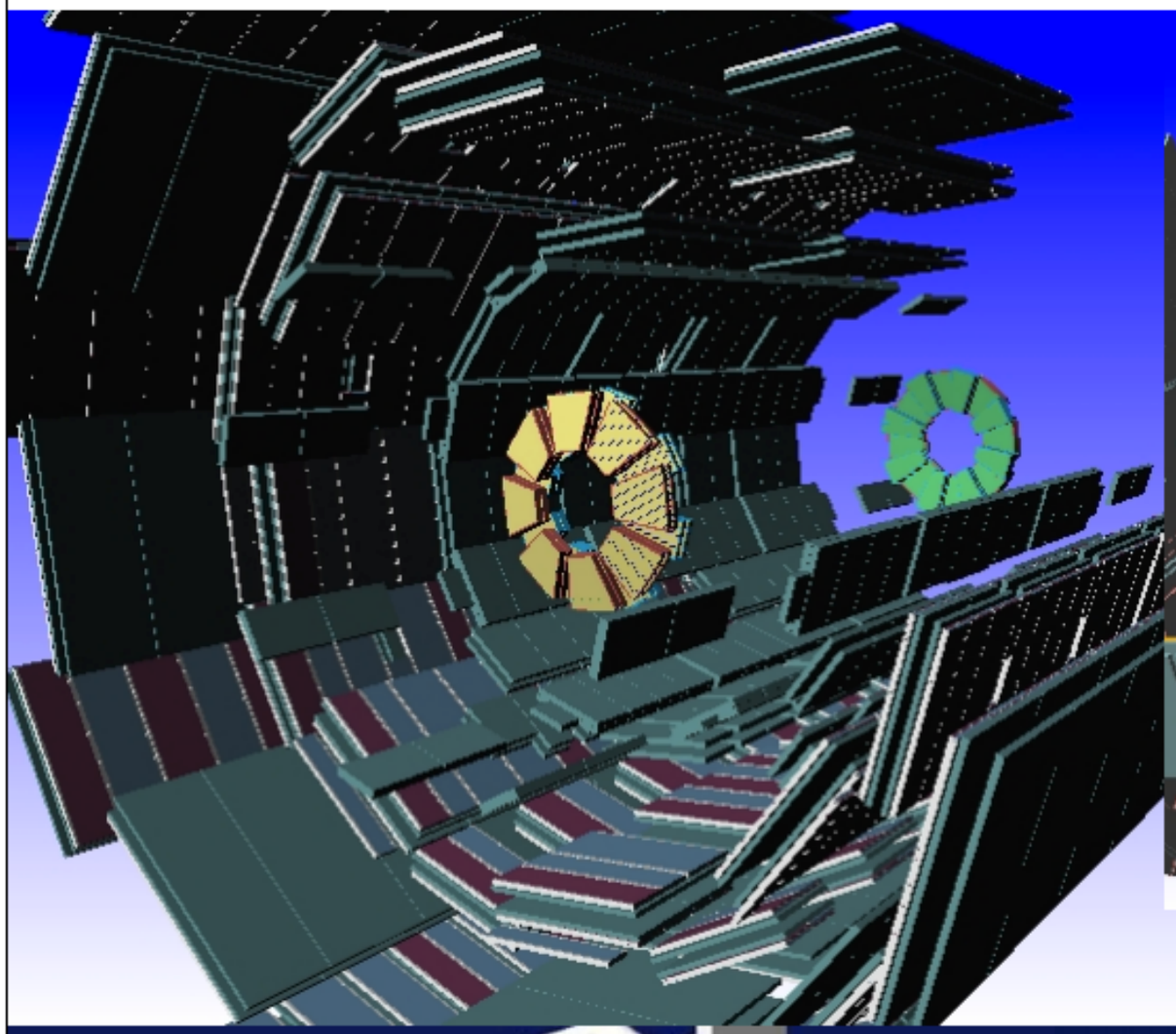
What is Geant4?

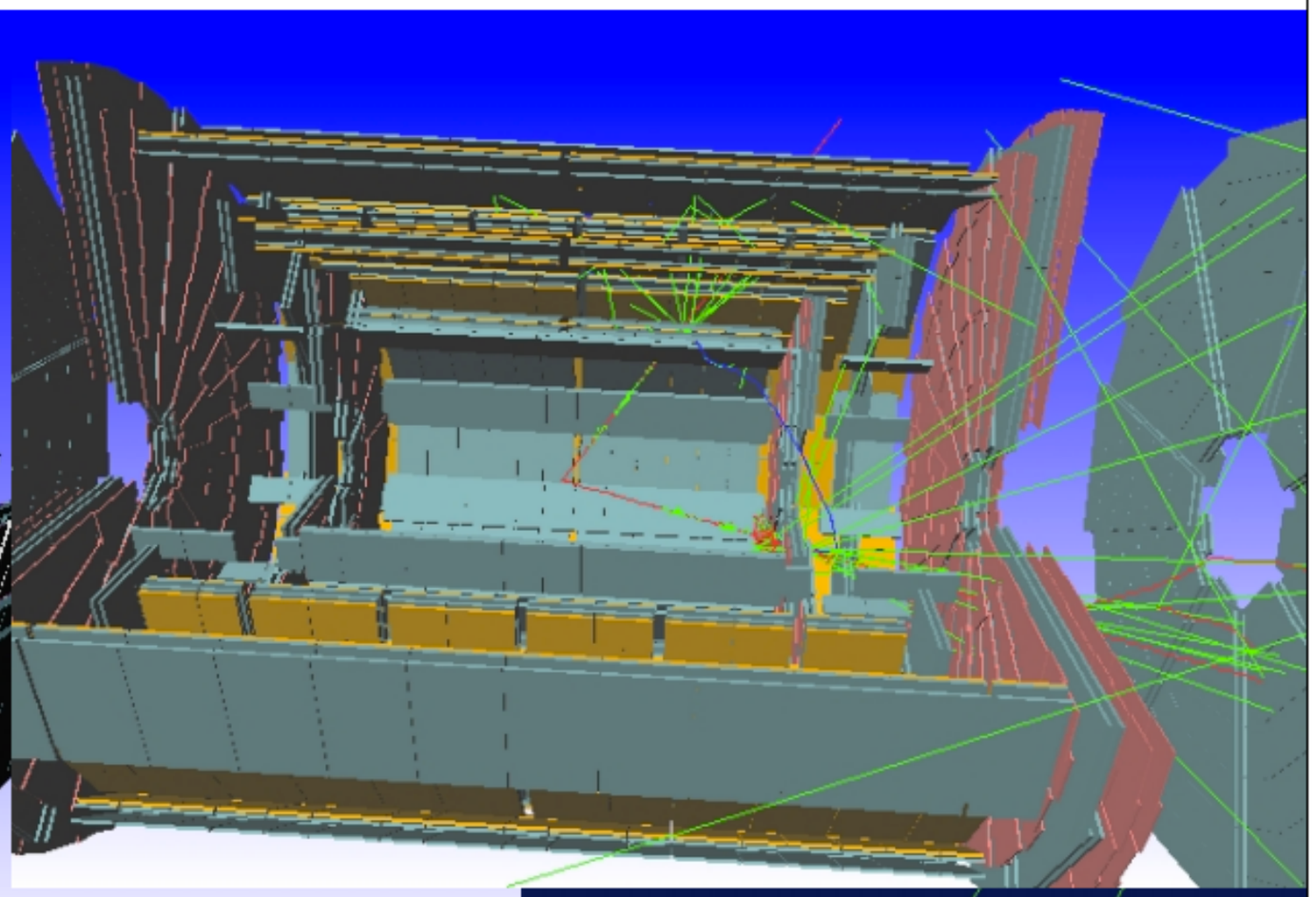
- A software (C++) toolkit for the Monte Carlo simulation of the passage of particles through matter
  - ‘propagates’ particles through geometrical structures of materials, including magnetic field
  - simulates processes the particles undergo
    - creates secondary particles
    - decays particles
  - calculates the deposited energy along the trajectories and allows to store the information for further processing (‘hits’)

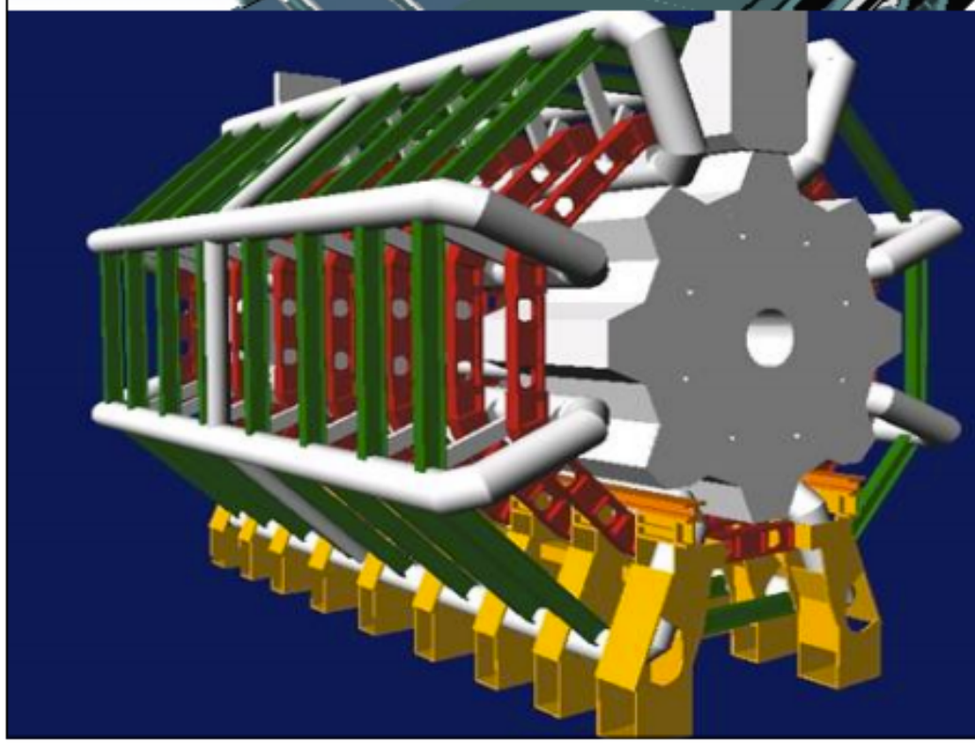


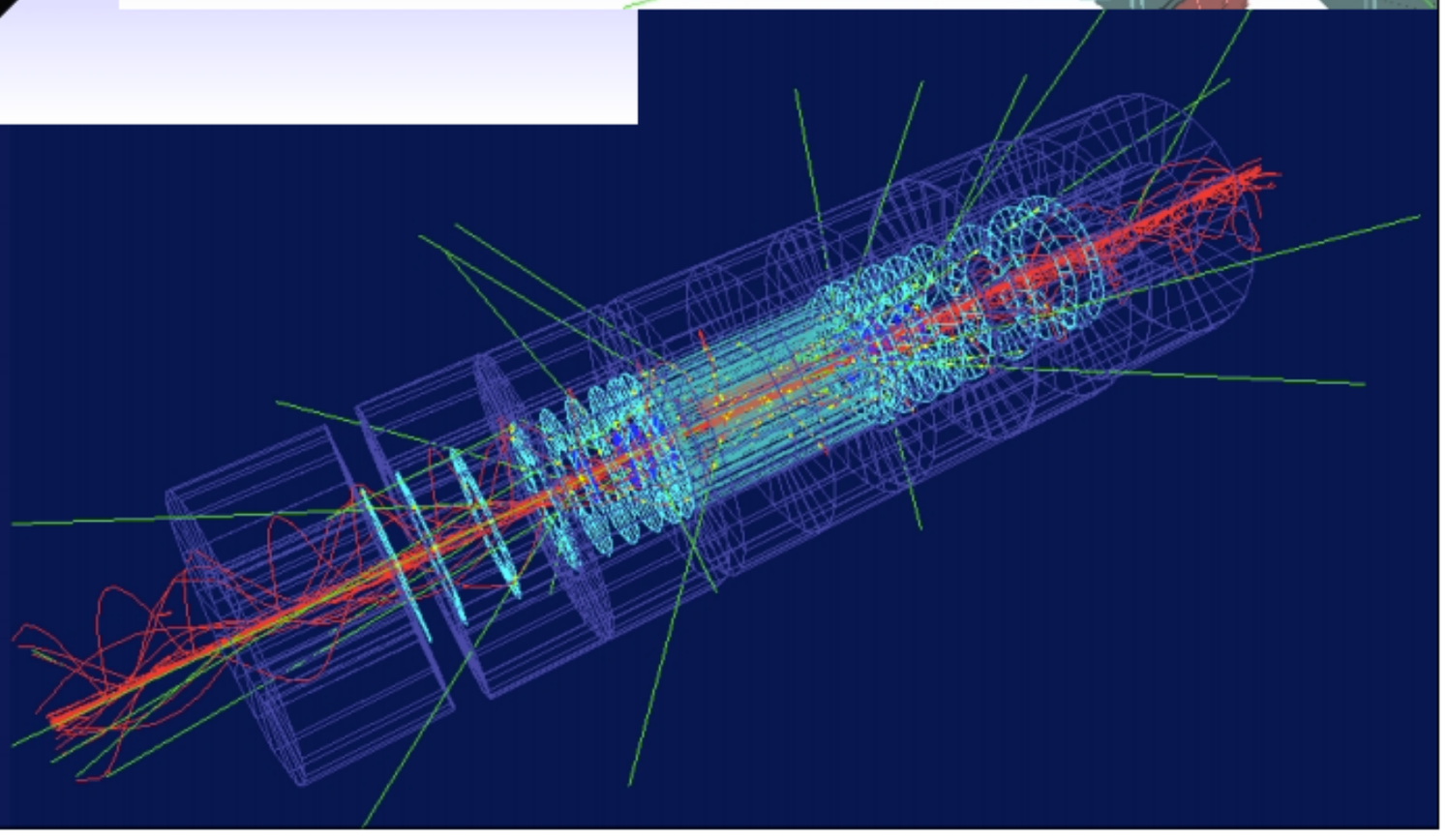
Simulated Higgs event in CMS

Geant4 in High Energy Physics (ATLAS at LHC)









Smart-2

ACE

LISA

INTEGRAL

Cassini

Bepi Colombo

GLAST

Herschel

SWIFT

Astro-E2

XMM-Newton

GAIA

JWST

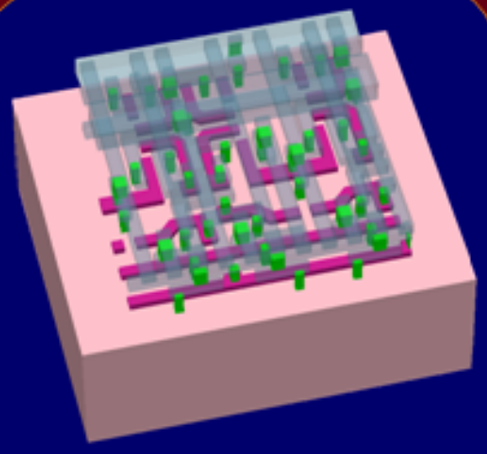
ISS Columbus

EUSO

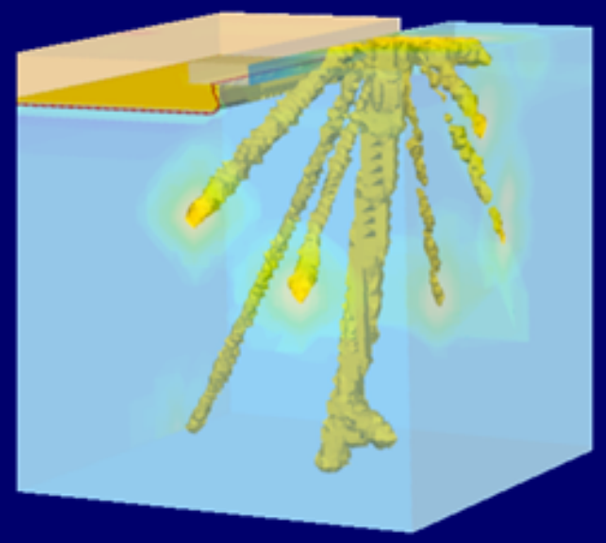
AMS

MAXI

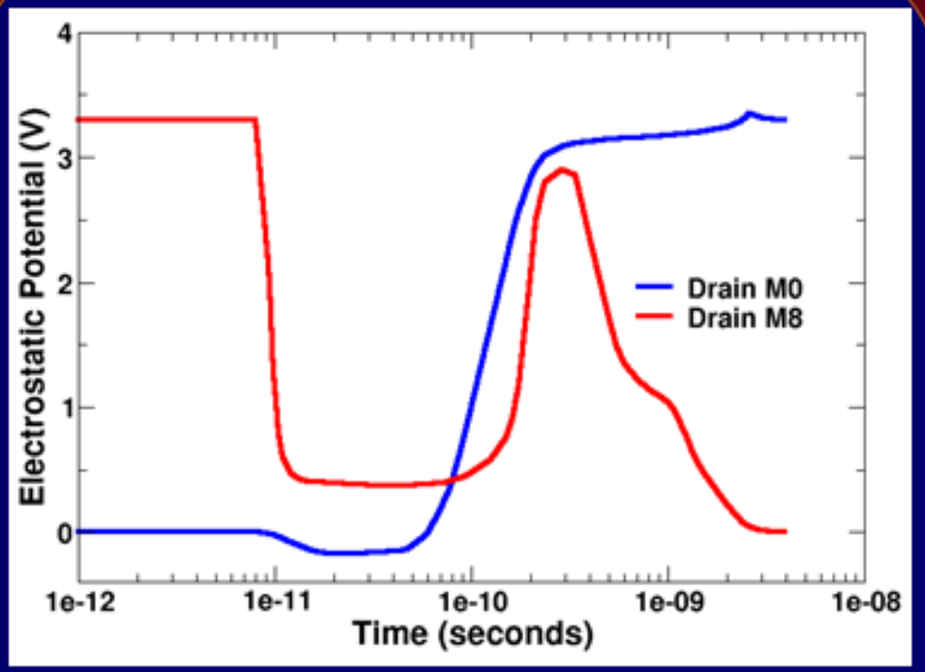
RADSAFE on SEE in SRAMs



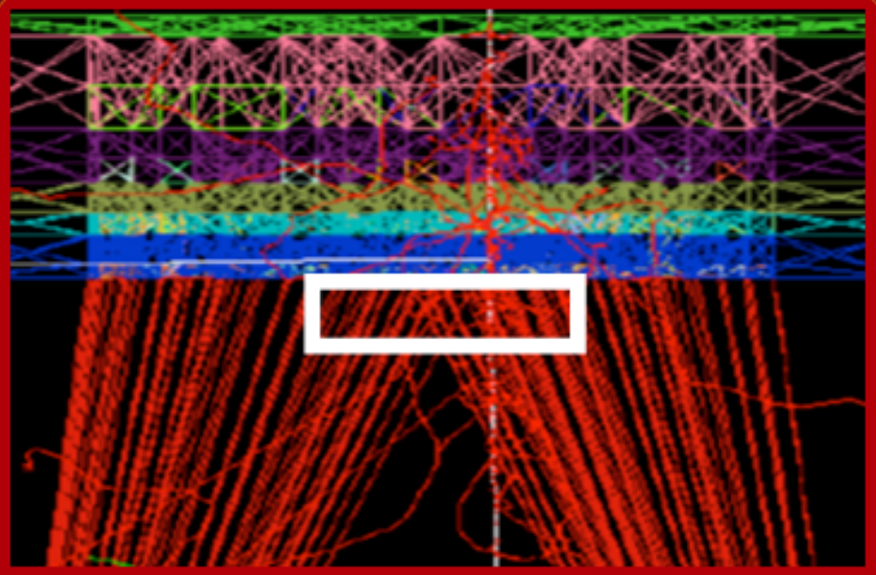
TCAD Cell Structure: SRAM Cell



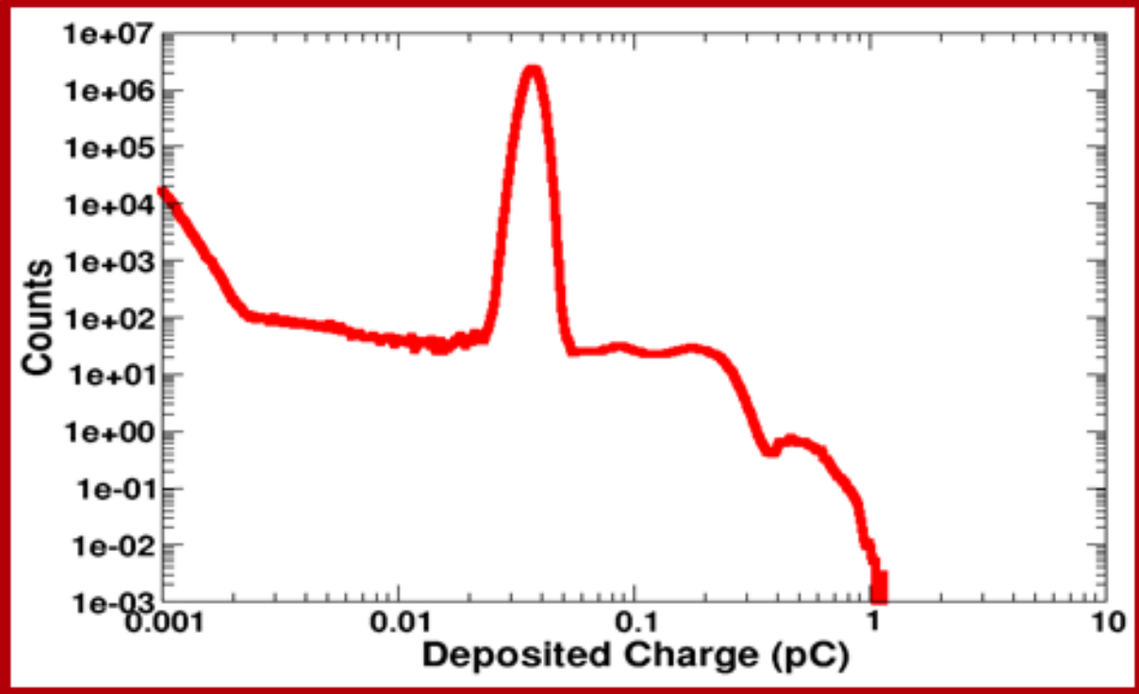
Single Charge Deposition in TCAD: Ne+W Event



SRAM Cell Upset

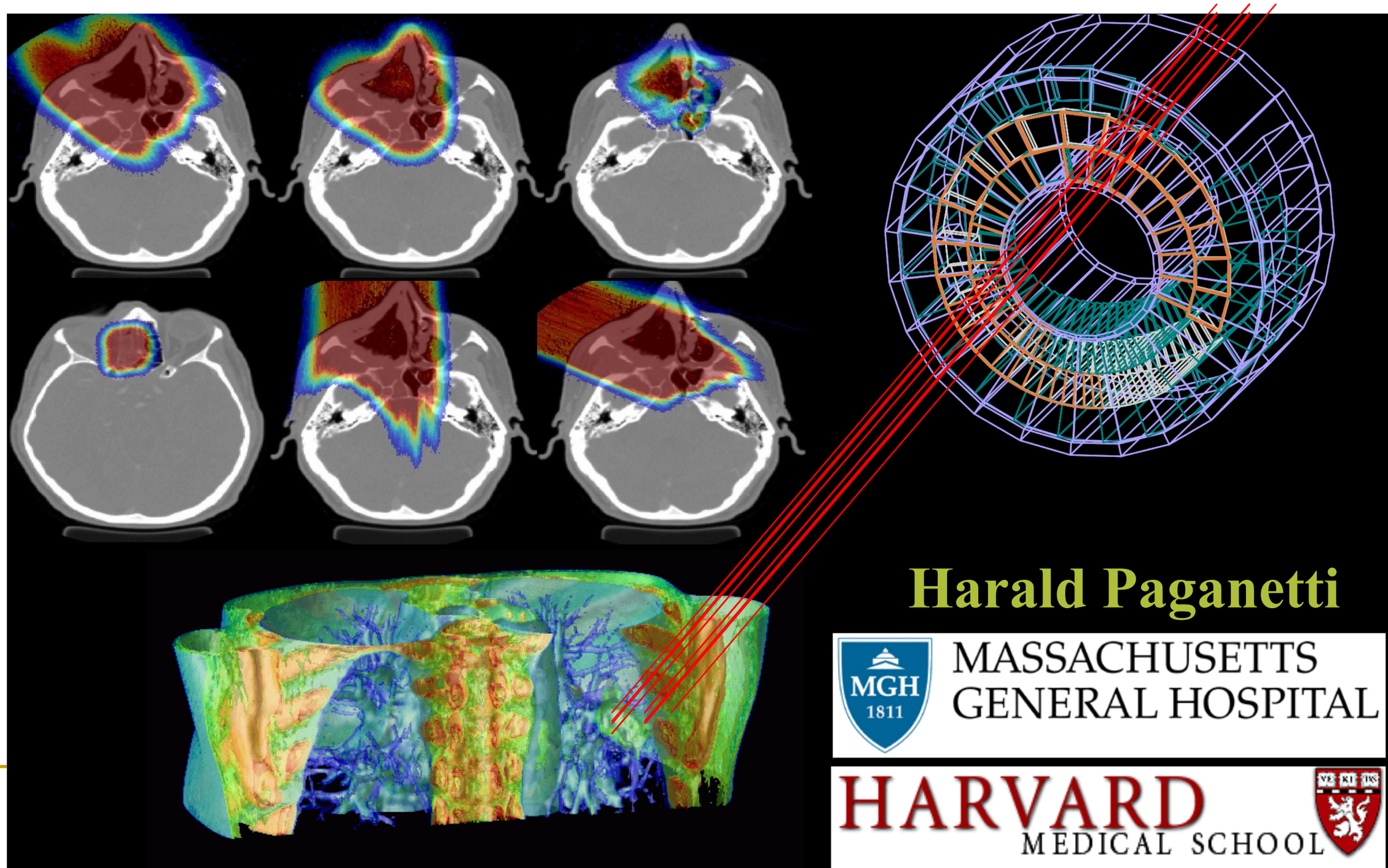


Geant4 Geometry and 523 MeV Neon Event



MRED Energy Deposition for 10^8 Events

GEANT4 based proton dose calculation in a clinical environment: technical aspects, strategies and challenges



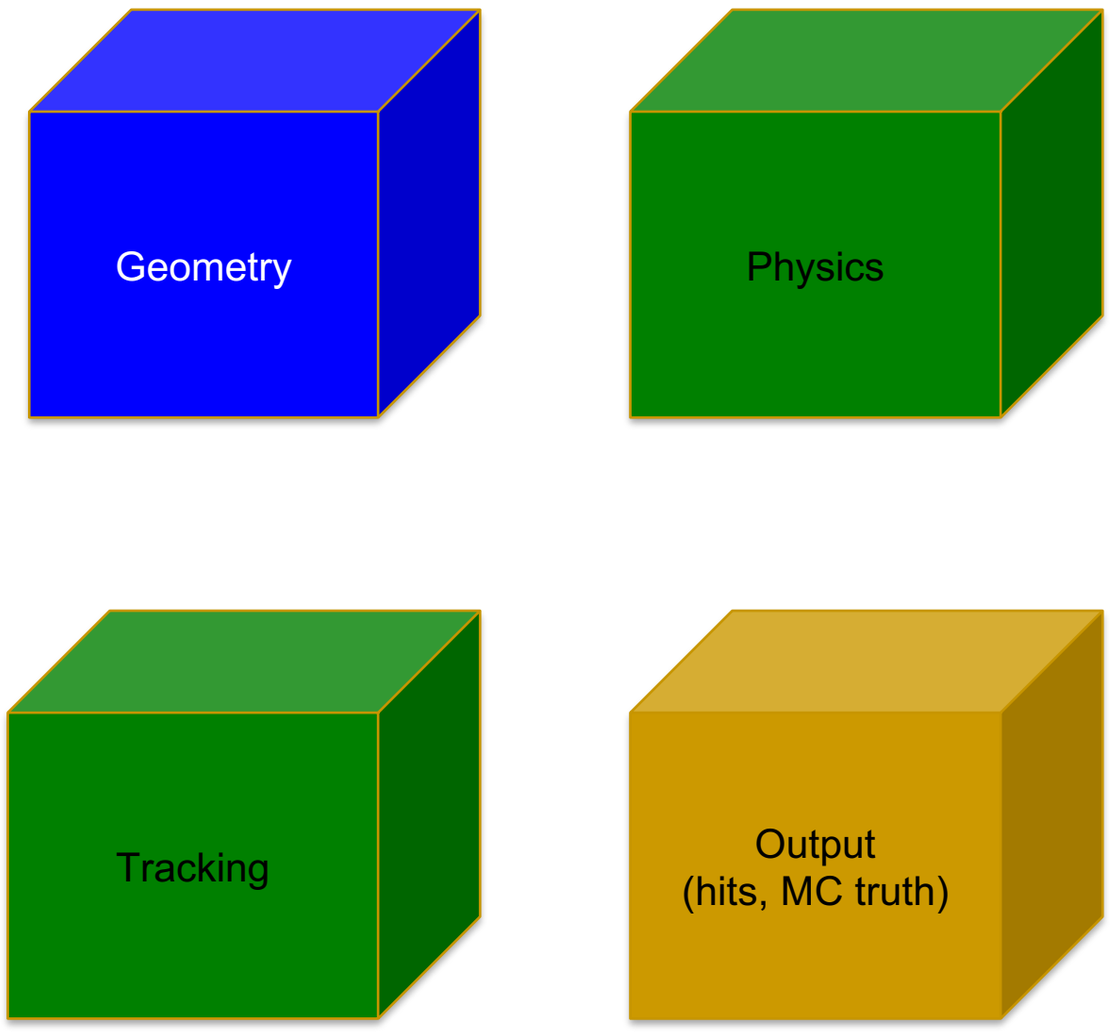
Harald Paganetti

MASSACHUSETTS GENERAL HOSPITAL

HARVARD MEDICAL SCHOOL

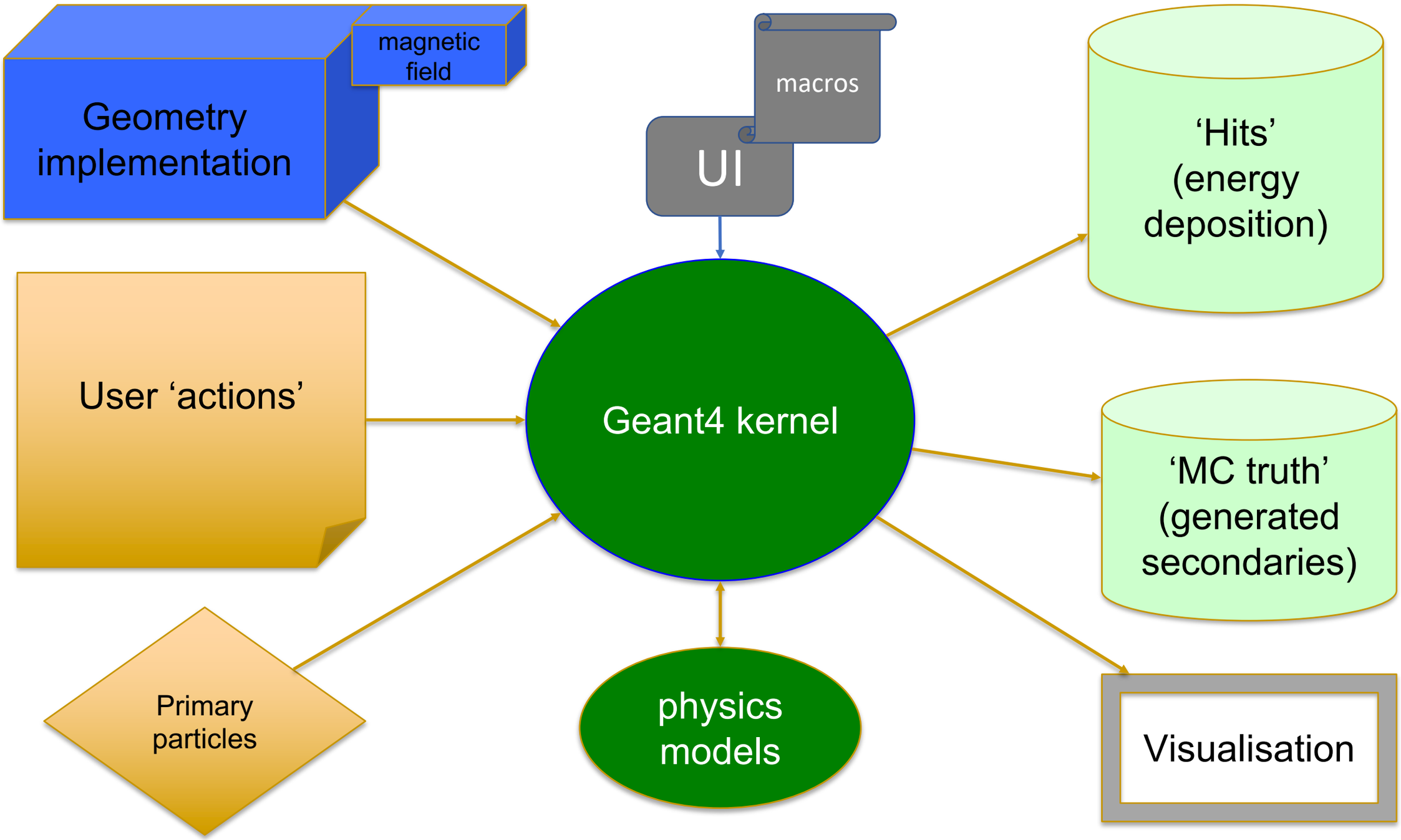
TOOLKIT ARCHITECTURE

Geant4 Components



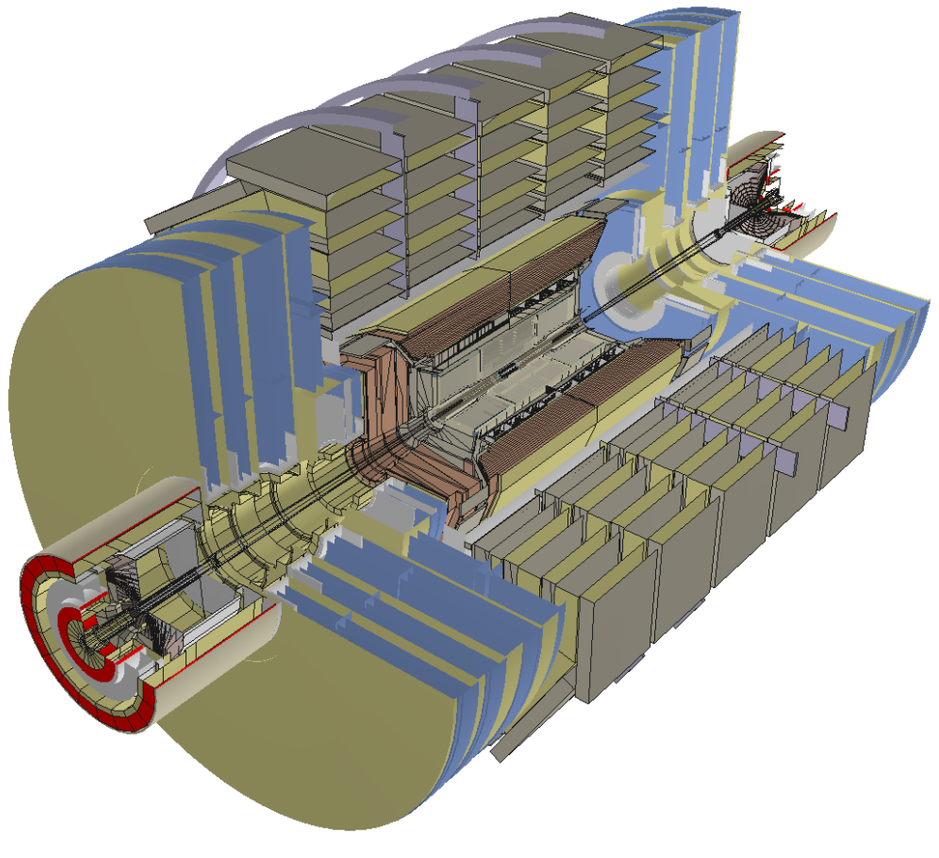
- ‘kernel’, internals of the engine, no direct interaction with the user code
- ‘user interface’
  - classes directly instantiated by the users with specific parameters
    - box of dimension x, y, z
  - base classes for concrete users implementations
    - ‘user actions’, sensitive detectors

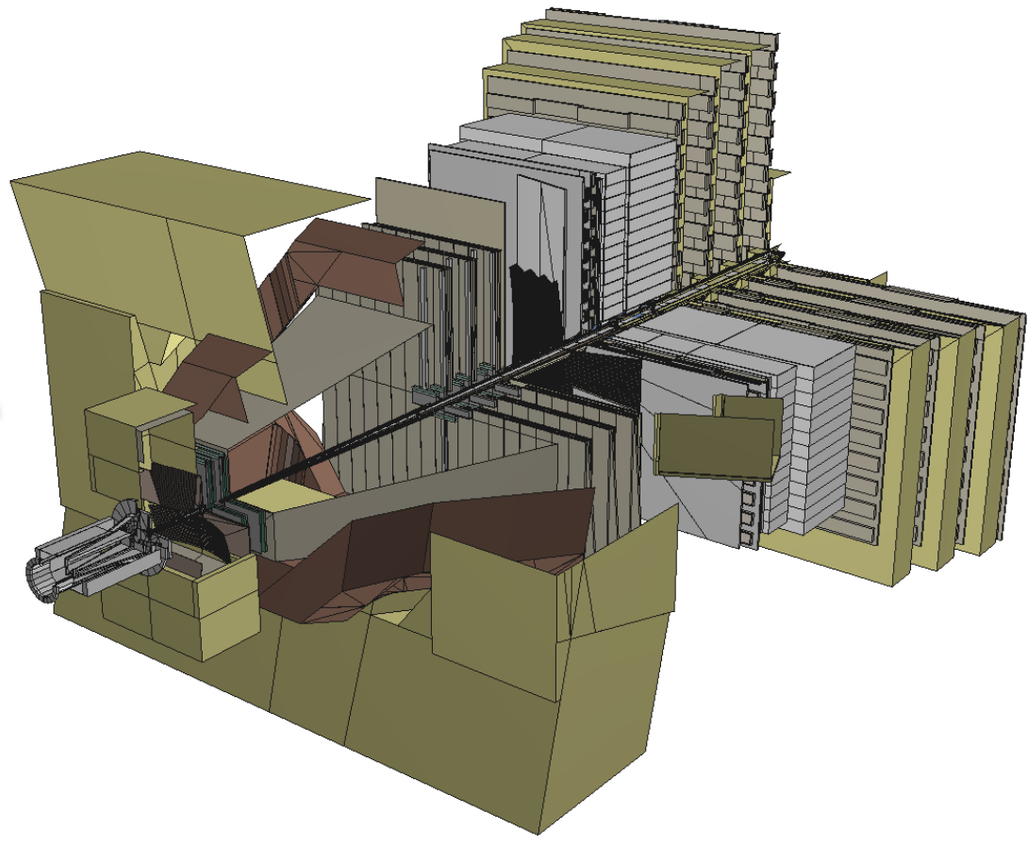
Geant4 application



Geometry

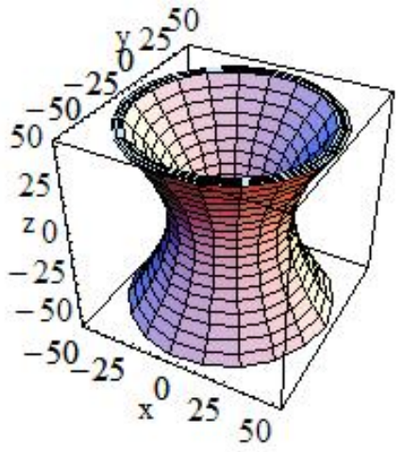
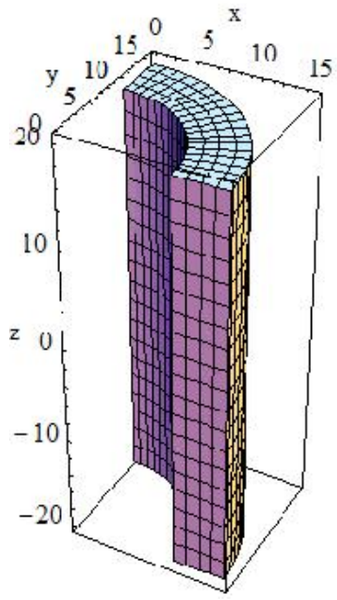
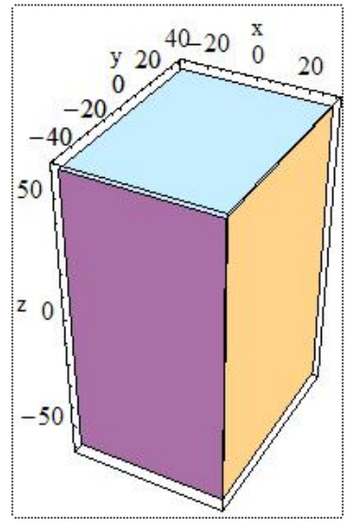
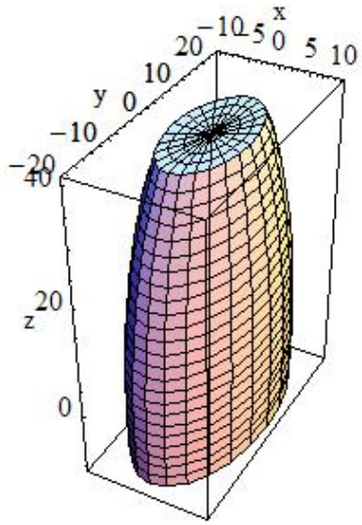
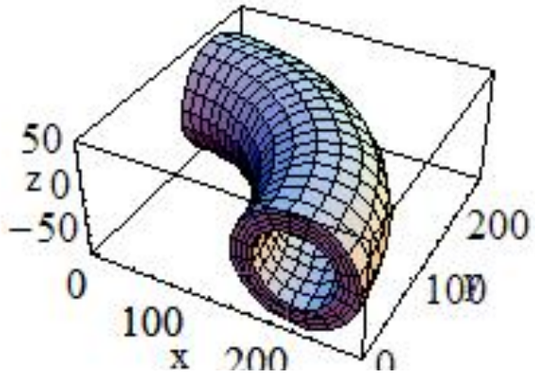
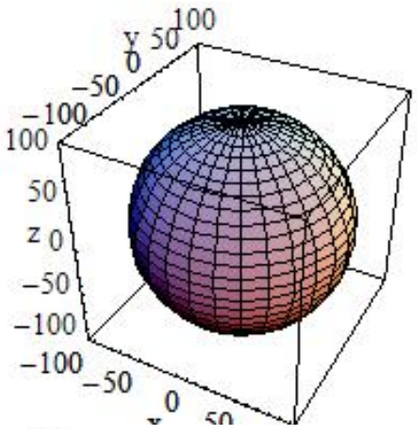
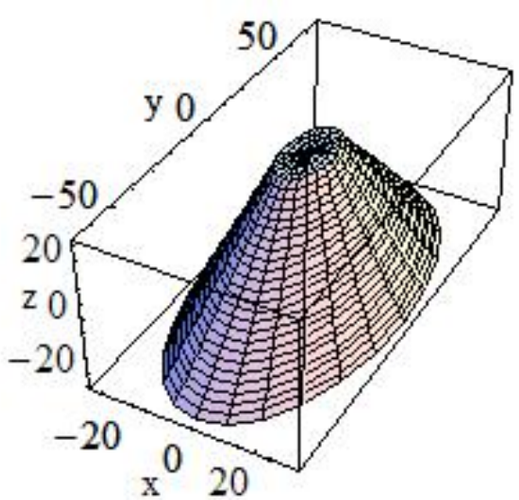
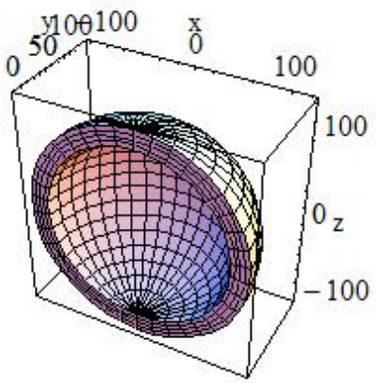
- how to implement (efficiently) this in your computer program?
  - you need ‘bricks’
    - ‘solids’, ‘shapes’
    - you need to position them
    - you want to ‘reuse’ as much as possible the same ‘templates’



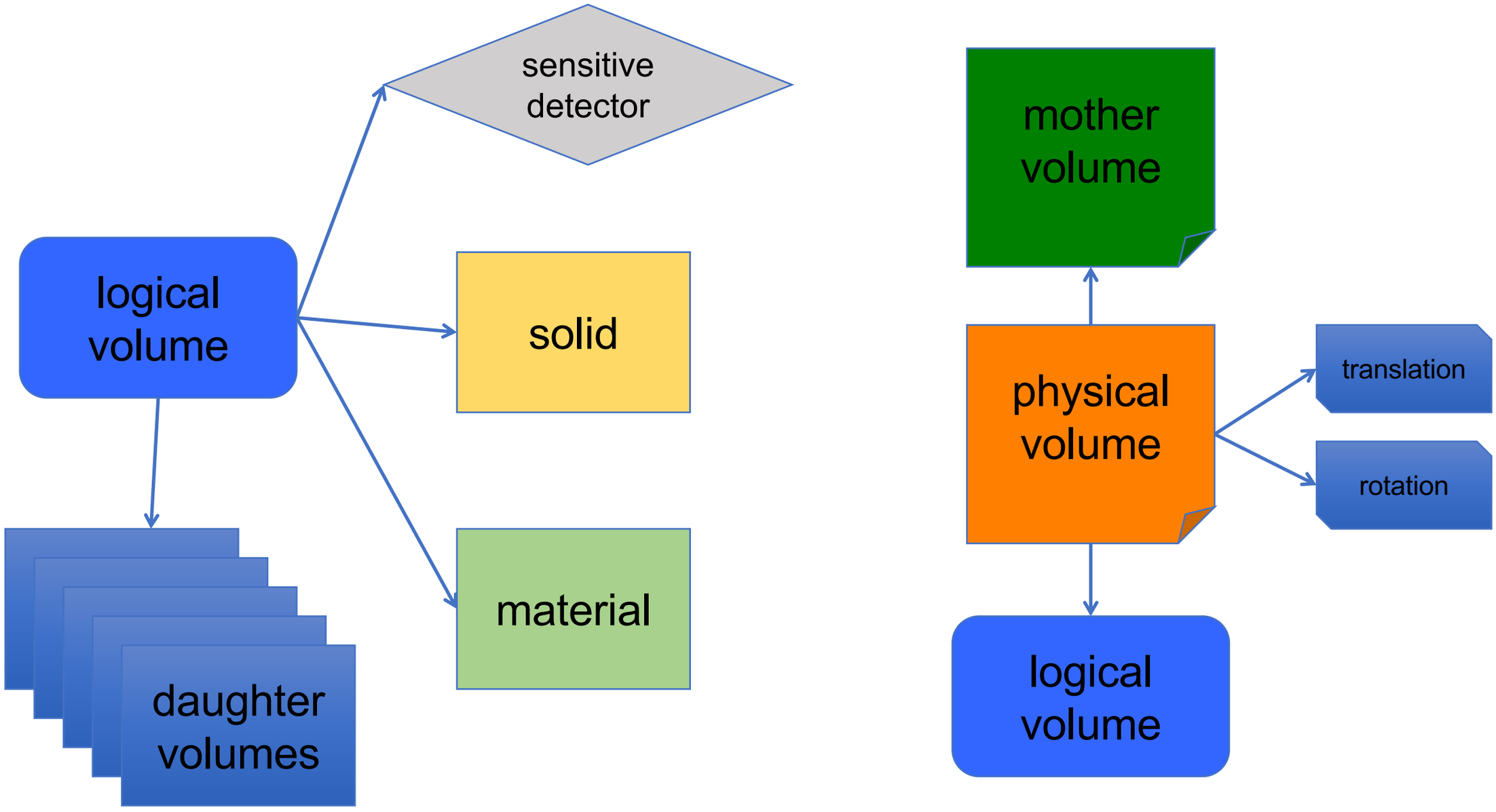


Geometry (1/3)

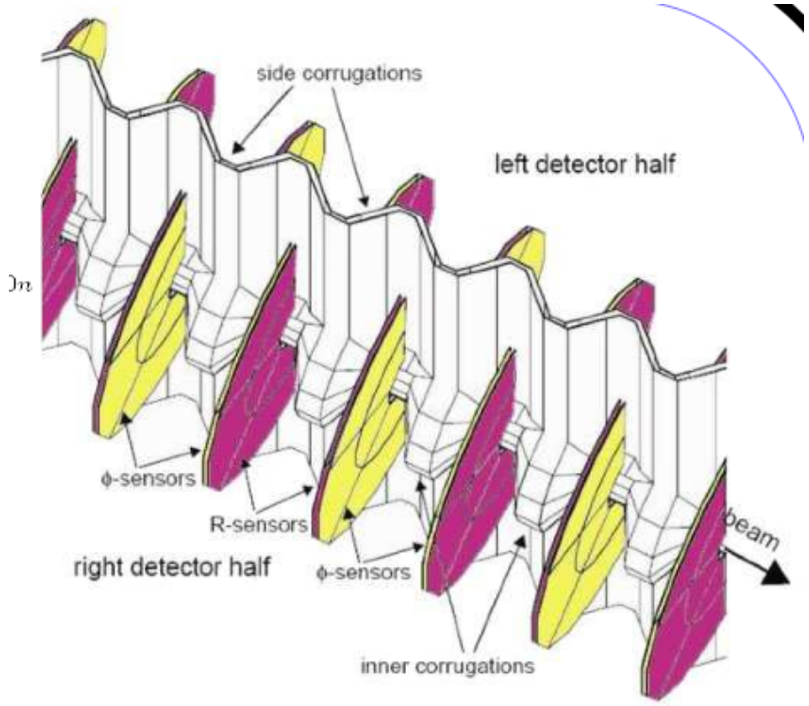
- set of solids (shapes) classes
  - box, sphere, tube, etc, etc…
  - boolean operations on solids
- logical volumes
  - unpositioned volumes with associated materials and possibly with ‘daughter’ volumes
    - unpositioned hierarchies of volumes
- physical volumes
  - concrete ‘placements’ of logical volumes
  - can reuse the same logical volume several times

Geometry (2/3)


sensitive detector
logical volume
solid
material
daughter volumes
mother volume
physical volume
translation
rotation
logical volume

Geometry (3/3)

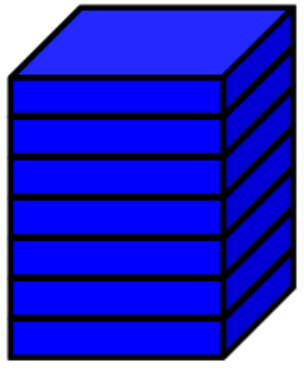

side corrugations
left detector half
ϕ-sensors
R-sensors
right detector half
ϕ-sensors
beam
inner corrugations

world (PV)
experiment cavern (LV)
tracker 1 (PV)
tracker 2 (PV)
tracker LV
module 1(PV)
module 1(PV)
module 1(PV)
module 1(PV)
module LV
calorimeter (PV)
calorimeter (LV)
cell1 (PV)
cell2 (PV)
cell LV

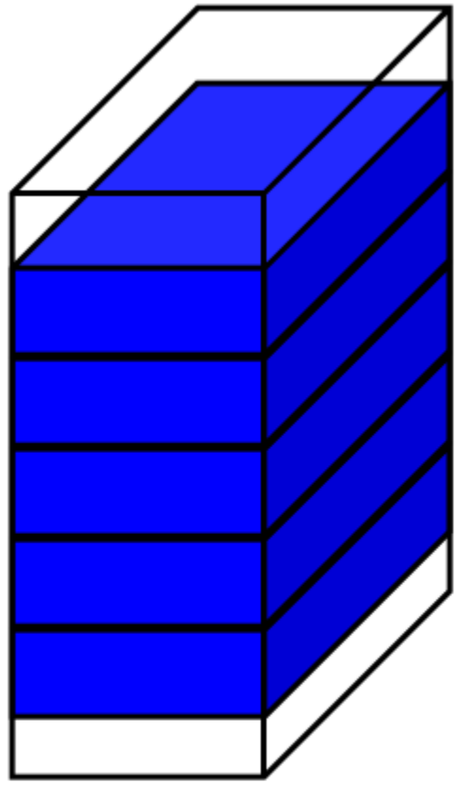
Advance geometry

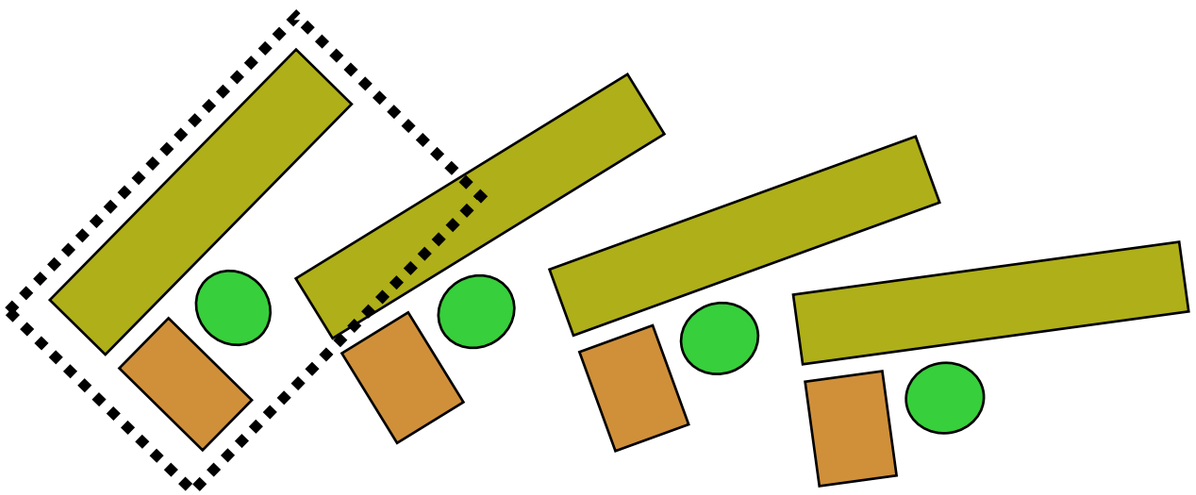
- replicas
- divisions
- reflections
- assemblies
- parameterizations

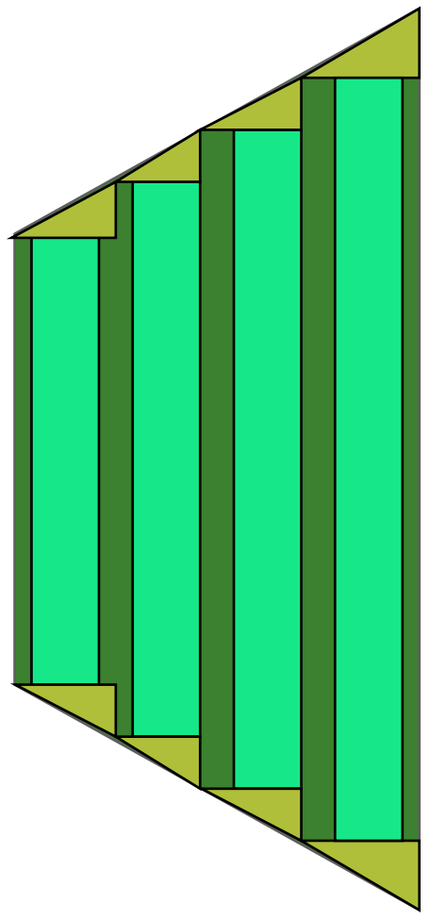
a daughter logical volume to be replicated

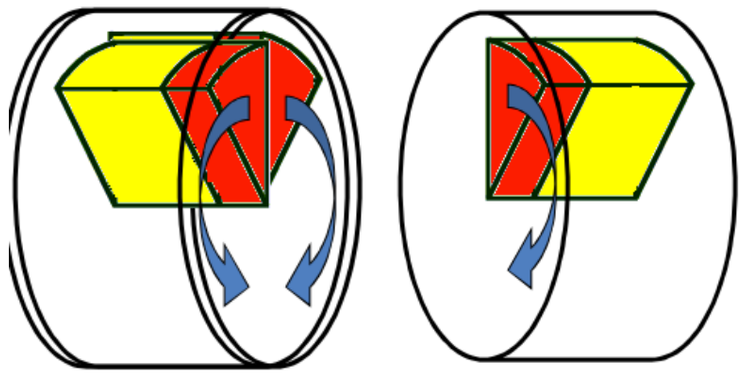


mother volume









Materials

- In nature, materials (chemical compounds, mixtures) are made of elements, and elements are made of isotopes
- In Geant4: G4Isotope, G4Element, G4Material
  - users can ‘build’ their materials, instantiating elements and adding them with the right fractions
- Geant4 contains also National Institute of Standards (NIST) database of materials
  - materials can be instantiated directly from it
  - strongly recommended to be used

```
==================================
###   Elementary Materials from the NIST Data Base
      ==============================
 Z Name  ChFormula       density(g/cm^3)  I(eV)
==================================
1   G4_H    H_2              8.3748e-05       19.2
2   G4_He                    0.000166322      41.8
3   G4_Li                    0.534            40
4   G4_Be                    1.848            63.7
5   G4_B                     2.37             76
6   G4_C                     2                81
7   G4_N    N_2              0.0011652        82
8   G4_O    O_2              0.00133151       95
9   G4_F                     0.00158029       115
10  G4_Ne                    0.000838505      137
11  G4_Na                    0.971            149
12  G4_Mg                    1.74             156
13  G4_Al                    2.6989           166
14  G4_Si                    2.33             173
```

NIST Materials

- Natural isotope compositions
- More than 3000 isotope masses
  - Used for elements definition

- NIST Elementary materials:
  - **H -> Cf (Z = 1 -> 98)**
- NIST compounds:
  - e.g. "**G4_ADIPOSE_TISSUE_ICRP**"
- HEP and Nuclear materials:
  - e.g. **Liquid Ar, PbWO**
- It is possible to build mixtures of NIST and user-defined materials

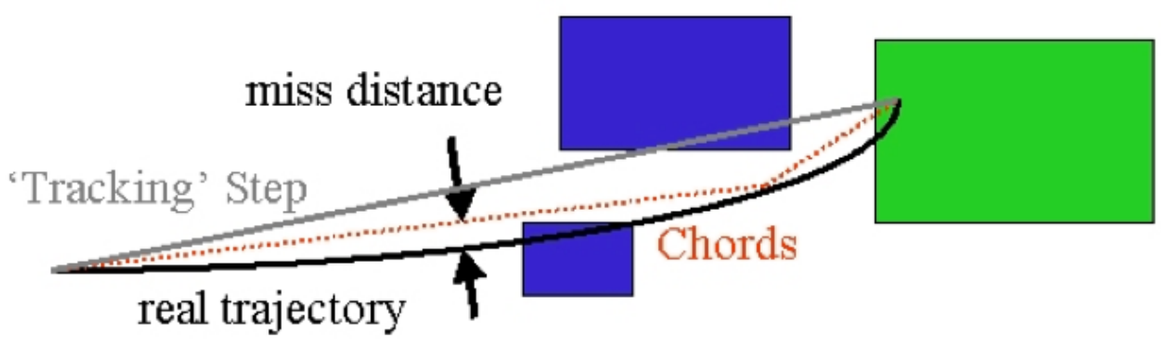
```
==================================
###   Compound Materials from the NIST Data Base
==================================
 N Name    ChFormula      density(g/cm^3)  I(eV)
==================================
13  G4_Adipose_Tissue          0.92          63.2
        1   0.119477
        6   0.63724
        7   0.00797
        8   0.232333
       11   0.0005
       12   2e-05
       15   0.00016
       16   0.00073
       17   0.00119
       19   0.00032
       20   2e-05
       26   2e-05
       30   2e-05
4  G4_Air                     0.00120479    85.7
        6   0.000124
        7   0.755268
        8   0.231781
       18   0.012827
2  G4_CsI                     4.51          553.1
       53   0.47692
       55   0.52308
```

Navigation and tracking

- ‘navigator’ role is to provide geometry information to tracking mechanism
  - locates the point in the geometry structure
    - which volume I am in?
  - calculates the distances to the boundaries (along specified direction)
- non-trivial problem of simulation ‘continuous’ physics (space-time) with discrete steps
  - steps cannot be infinitely small
- steps need to be limited by crossing geometrical boundaries, physics or kinetic energy going to 0
- accuracy of tracking on the surfaces defined by geometrical ‘tolerance’

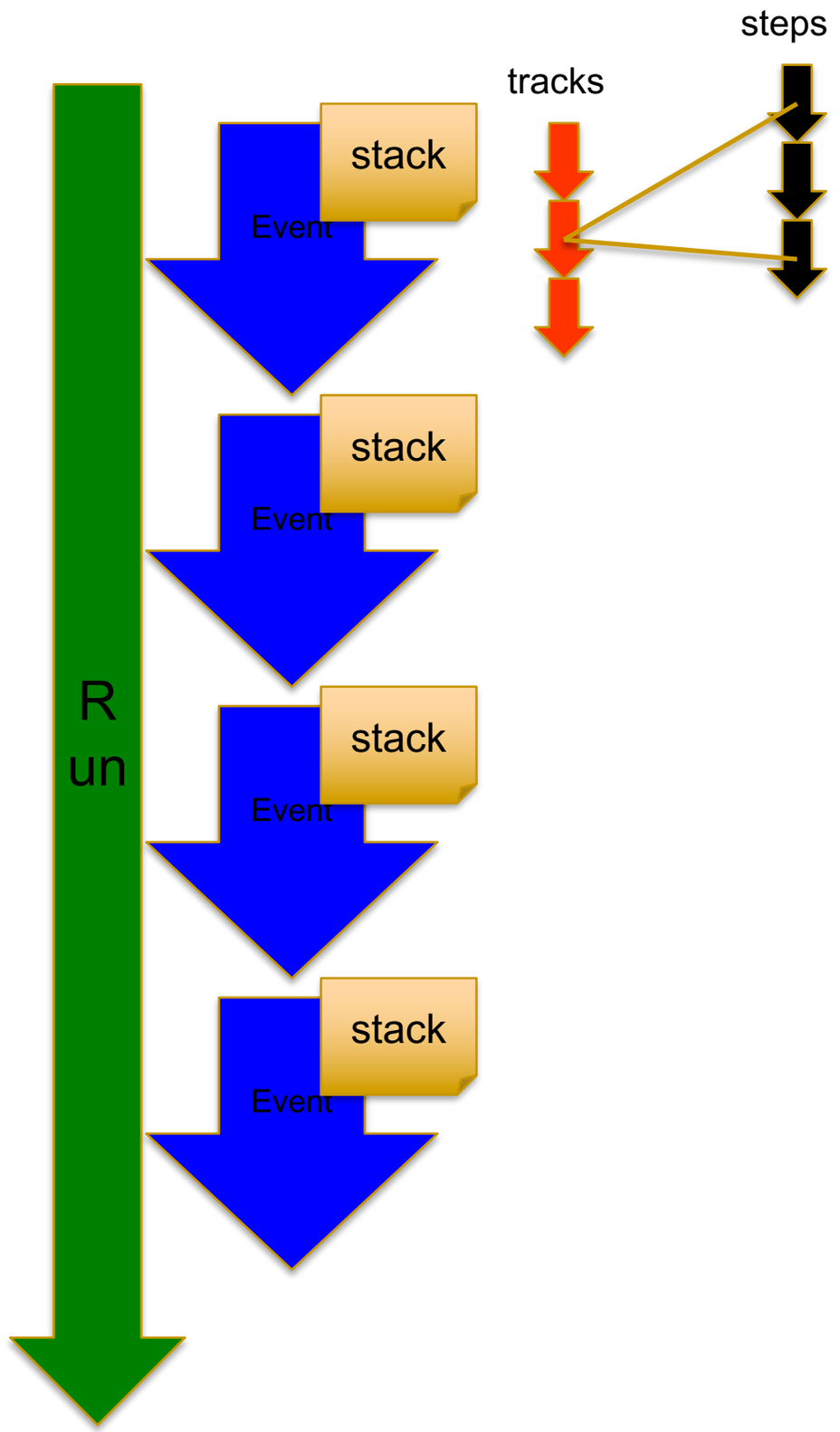
Magnetic field

- Geant4 can propagate in magnetic fields, electric fields, electromagnetic fields, and gravity fields, uniform or non-uniform
- the equation of motion of the particle in the field is integrated using Runge-Kutta method
  - in particular cases analytical solutions can be used
- curved trajectory broken up in linear cord segments
- parameter ‘miss distance’ sets how closely the curved path is approximated
- non trivial problem to avoid qualitatively wrong results



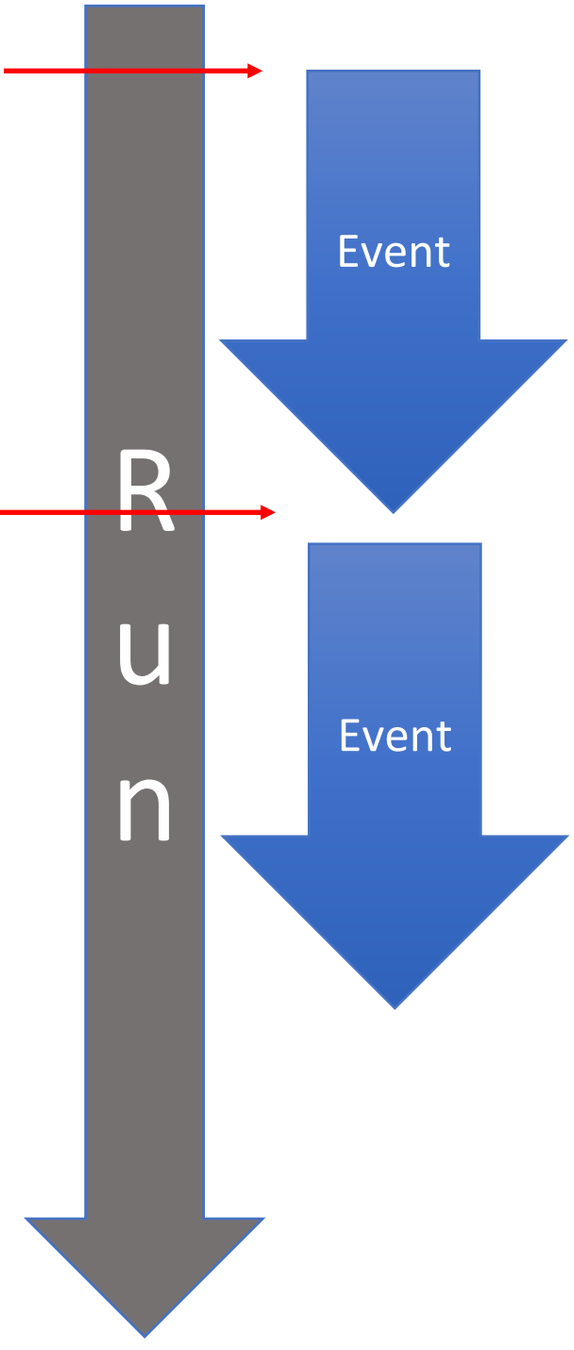
User actions

- how to control your simulation?
  - ‘user actions’
    - G4UserRunAction
    - G4UserEventAction
    - G4UserStackingAction
    - G4UserTrackingAction
    - G4UserSteppingAction
- fully customizable (empty by default)
- used to setup and/or modify the simulation or collect information about the run
- allow user to take actions specific for his simulation
  - simulated only relevant particles
  - save specific information, fill histograms
  - speed-up simulation by applying different limits



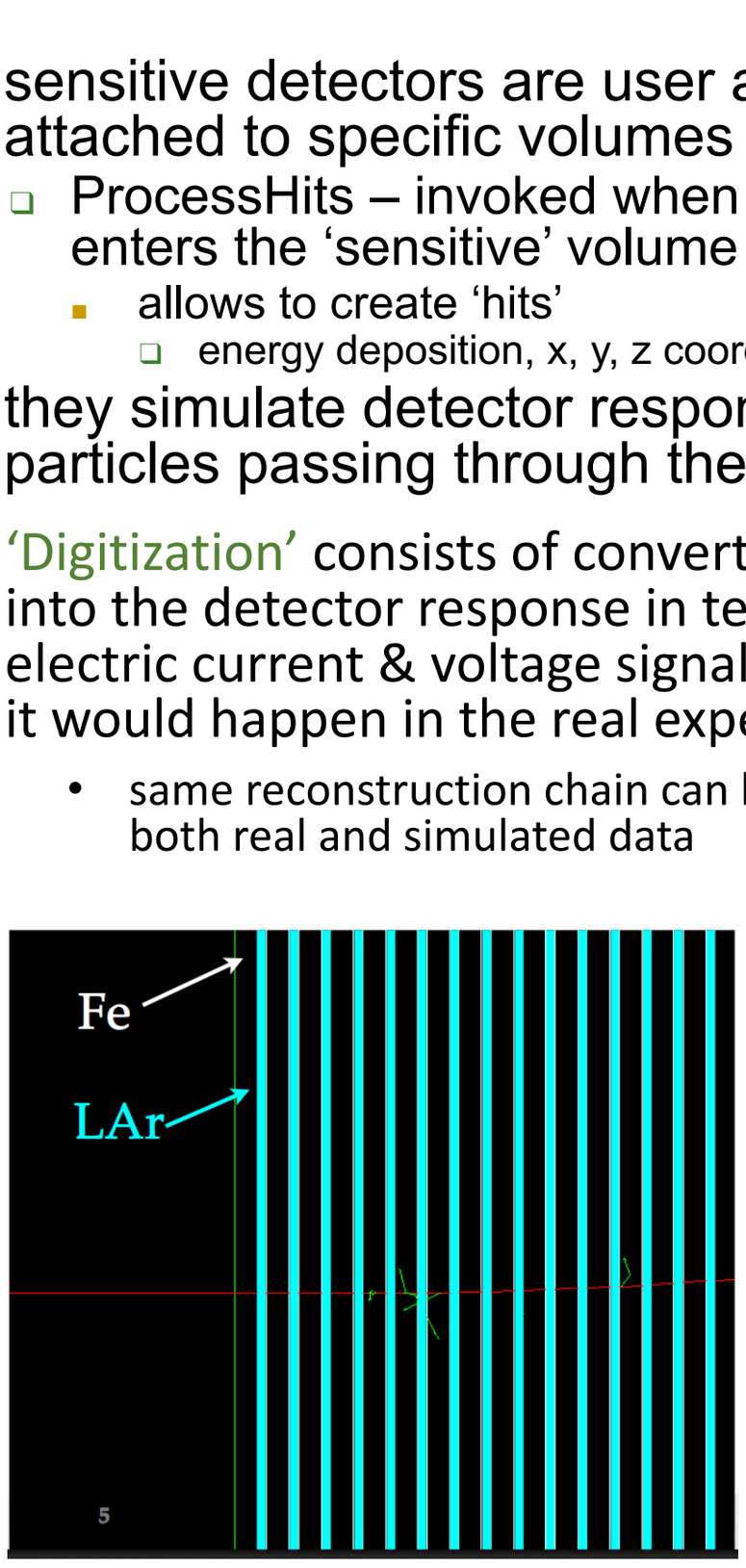
Example: UserEventAction

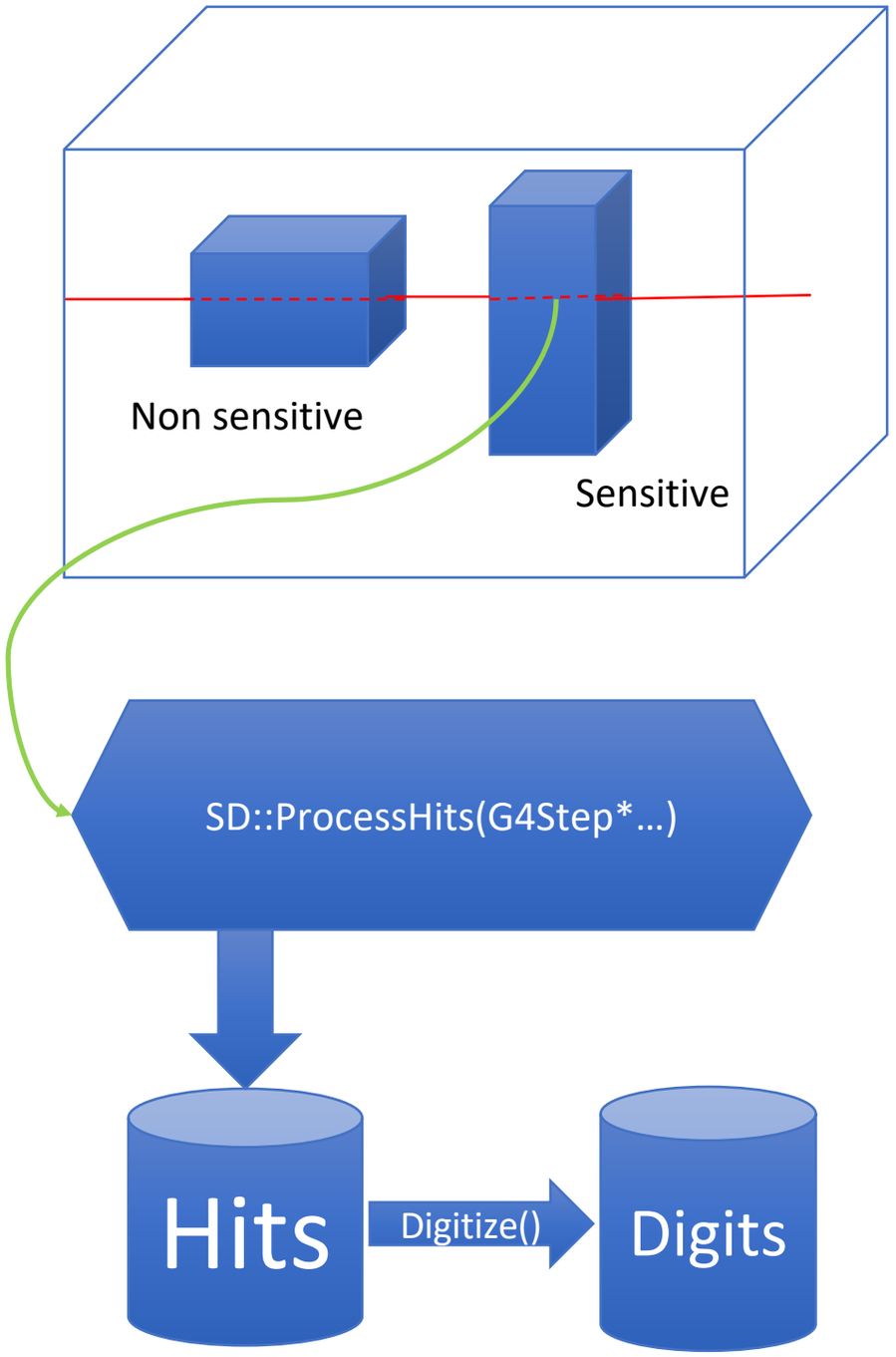
- virtual void BeginOfEventAction(const G4Event*)
  - This method is invoked before converting the primary particles to G4Track objects
  - A typical use of this method would be to initialize and/or book histograms for a particular event
- virtual void EndOfEventAction(const G4Event*)
  - This method is invoked at the very end of event processing
  - Typically used for a simple analysis of the processed event



Users actions – sensitive detectors

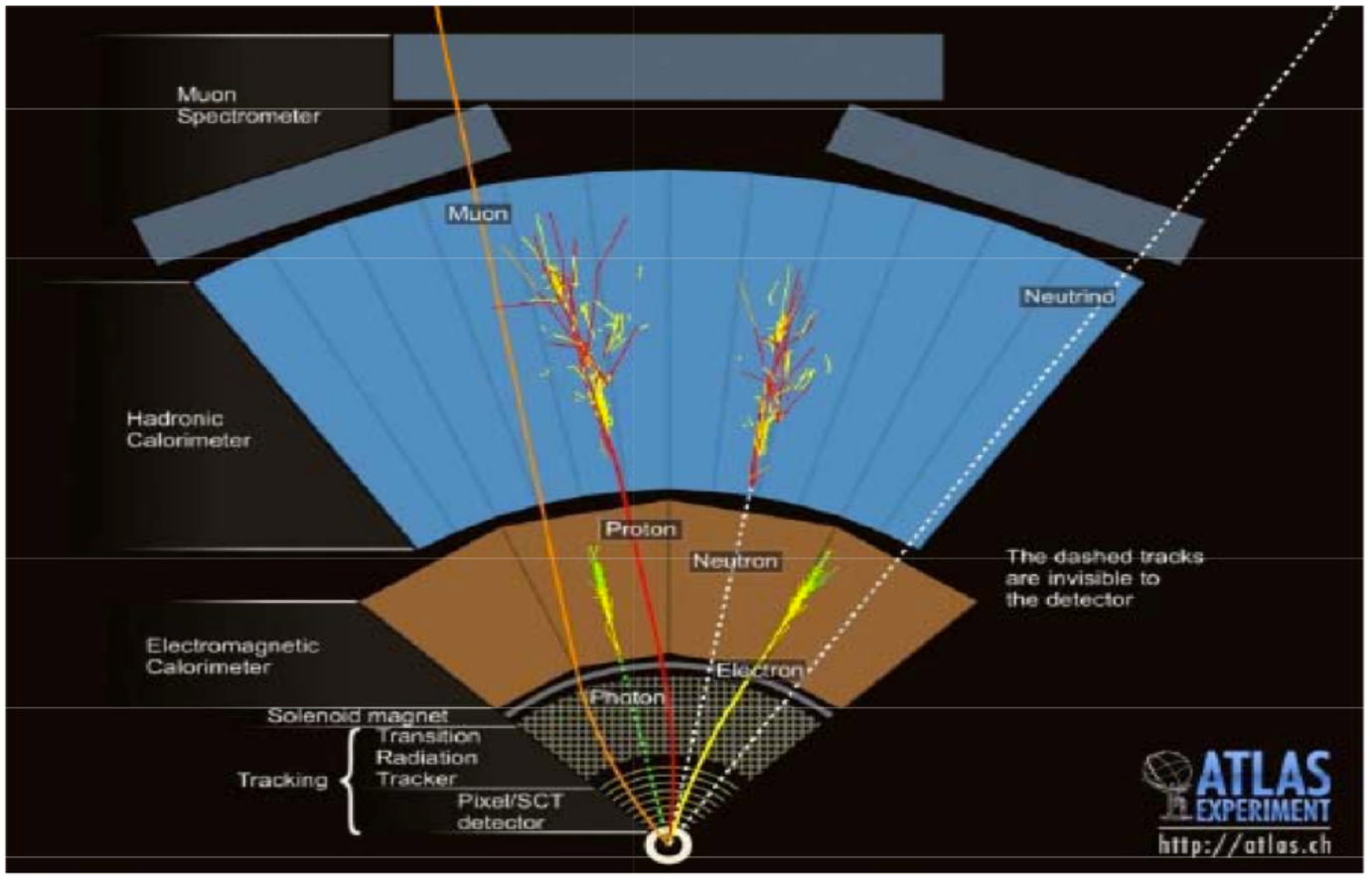
- sensitive detectors are user actions attached to specific volumes
  - ProcessHits – invoked when a particle enters the ‘sensitive’ volume
    - allows to create ‘hits’
      - energy deposition, x, y, z coordinates, etc
- they simulate detector response to the particles passing through the sensors
- ‘Digitization’ consists of converting ‘Hits’ into the detector response in terms of electric current & voltage signals (digits), as it would happen in the real experiment
  - same reconstruction chain can be applied for both real and simulated data



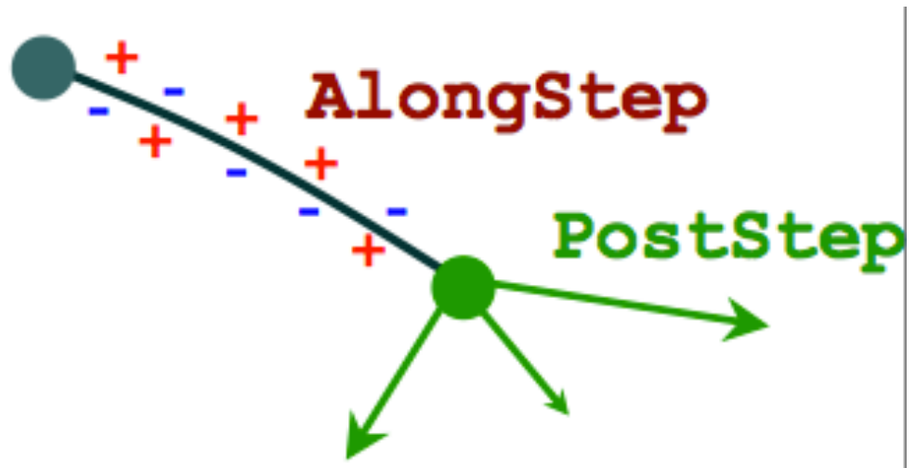


Physics…

- what happens to when a particle traverses some matterial?
- we need to implement the physics we know



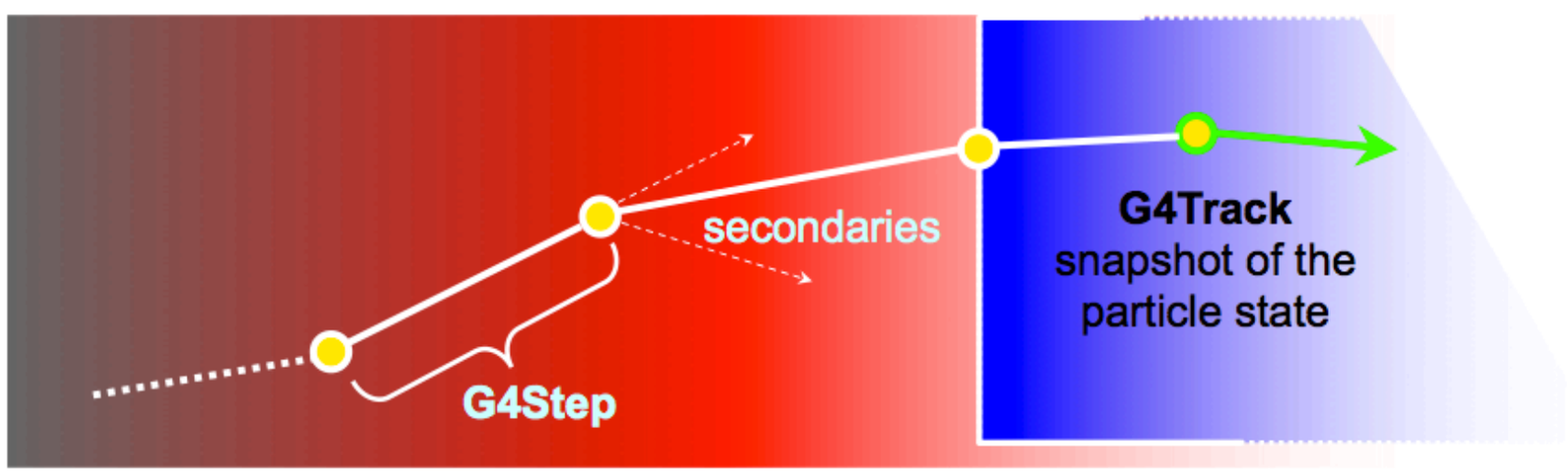
Physics processes (1/2)



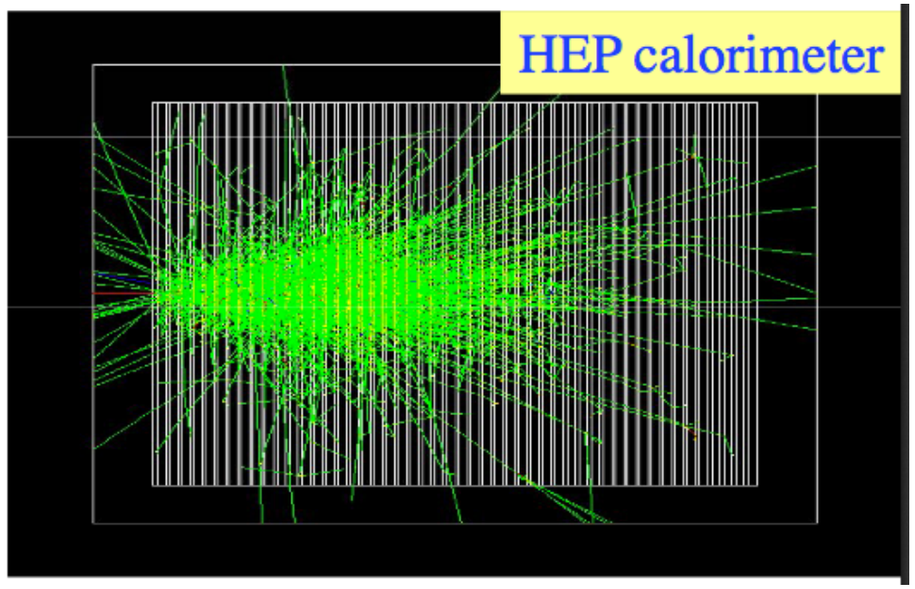
- each process provides
  - GetPhysicalInteractionLength method
    - calculates probability of interactions from the cross-section
  - *DoIt methods
    - AlongStepDoIt – always invoked for all the processes defined for a given particle
    - PostStepDoIt – invoked at the end of the step if the given process provided the smallest Physical Interaction Lenght.
    - AtRestDoIt – like PostStepDoIt but for stopped particles

Physics processes (2/2)

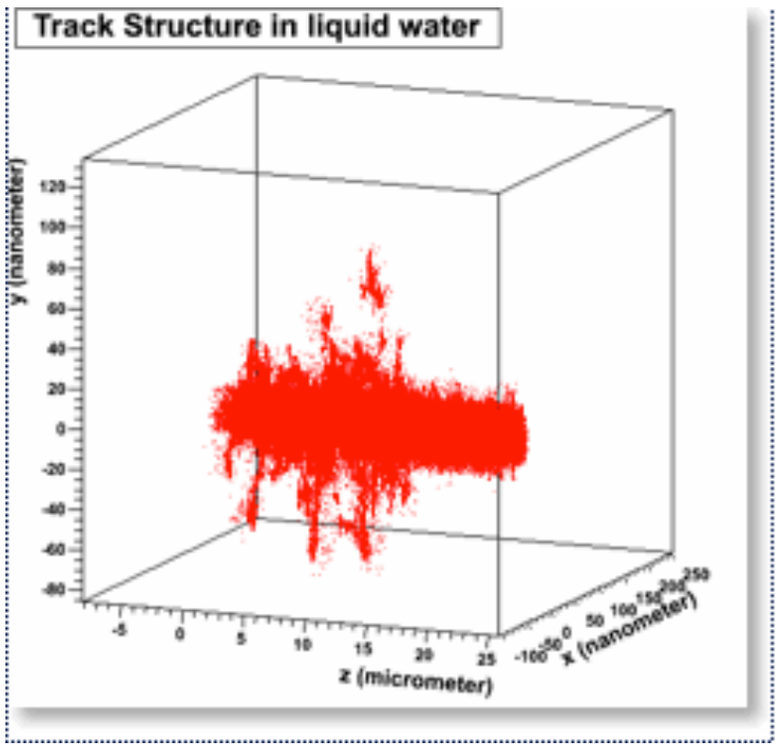
- Physical interaction lengths and distance to the next boundary (eventually recalculated) are compared
  - the smallest value wins
- along step called for all processes and resulting change of the track accumulated in G4Step
- particle change object stores the final state information including secondary tracks generated by DoIt methods



Electromagnetic (1/2)

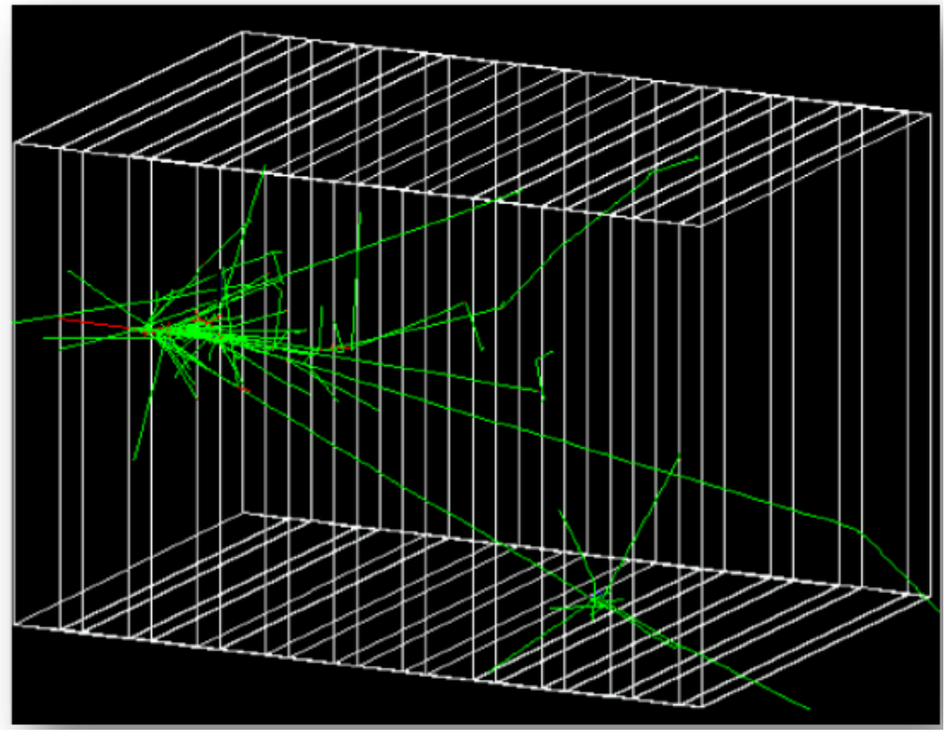


- simulation of electromagnetic interactions of charged particles, gammas and optical photons
  - standard electromagnetic physics
    - optimized for high and medium energy applications
      - energy range from 1keV to 1PeV
  - low energy electromagnetic physics
    - down to eV
      - medical and biological application
        - Geant4-DNA project



Electromagnetic (2/2)

- Gammas:
  - Gamma-conversion, Compton scattering, Photo-electric effect
- Leptons(e, m), charged hadrons, ions
  - Energy loss (Ionisation, Bremstrahlung), Multiple scattering, Transition radiation, Synchrotron radiation, e+ annihilation.
- Photons:
  - Cerenkov, Rayleigh, Reflection, Refraction, Absorption, Scintillation
- High energy muons
- A choice of implementations for most processes
  - “Standard”: performant, where relevant physics above 1 KeV
  - “Low Energy”: Extra accuracy, for application delving below 1 KeV



50 MeV e- entering LAr-Pb calorimeter

Need for production cuts

- some electromagnetic processes have infrared divergences
  - threshold needed below which no secondaries are generated
    - production threshold for gammas, electrons and positrons
- ‘cuts’ can be defined per region of the geometry
  - detailed simulation of the EM shower in the sensitive part, but no details needed in some ‘dead’ materials (support, etc)
- Geant4 uses ‘cut in range’
  - converted to energy for each material
  - assures better coherency of the simulation that a cut in energy would do

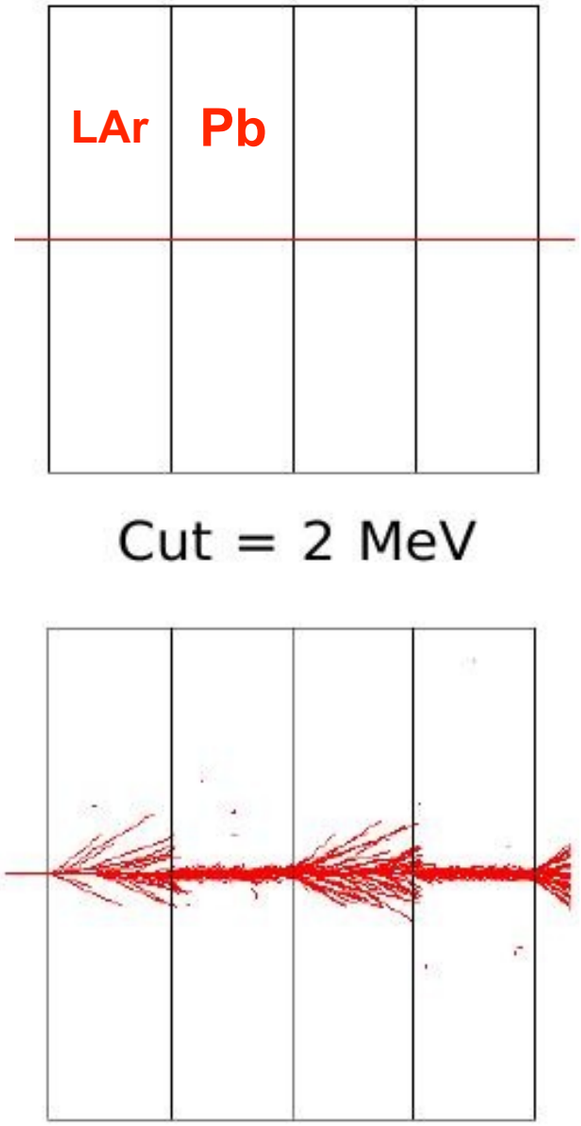
Why cut in range

- traditionally Monte Carlo simulations impose absolute cutoff in energy
  - particle are stopped when this energy is reached
  - remaining energy is dumped at that point
- this can lead to imprecise stopping location and deposition of energy
- there is a particle dependence
  - range of a 10 keV γ in Si is a few cm
  - range of a 10 keV e- in Si is a few microns
- and a material dependence
  - suppose you have a detector made of alternating sheets of Pb and plastic scintillator
  - if the cutoff is OK for Pb it will likely be wrong for the scintillator which does the actual energy measurement

Cut in range (production threshold) vs. energy cut

Example: 500 MeV p in LAr-Pb Sampling Calorimeter

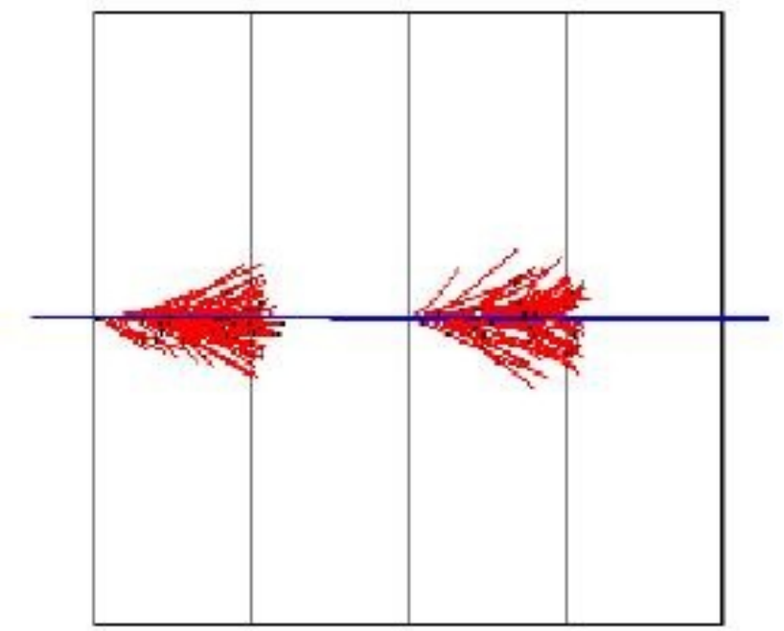
Geant3 (and others)



Cut = 2 MeV

Cut = 450 keV

Geant4

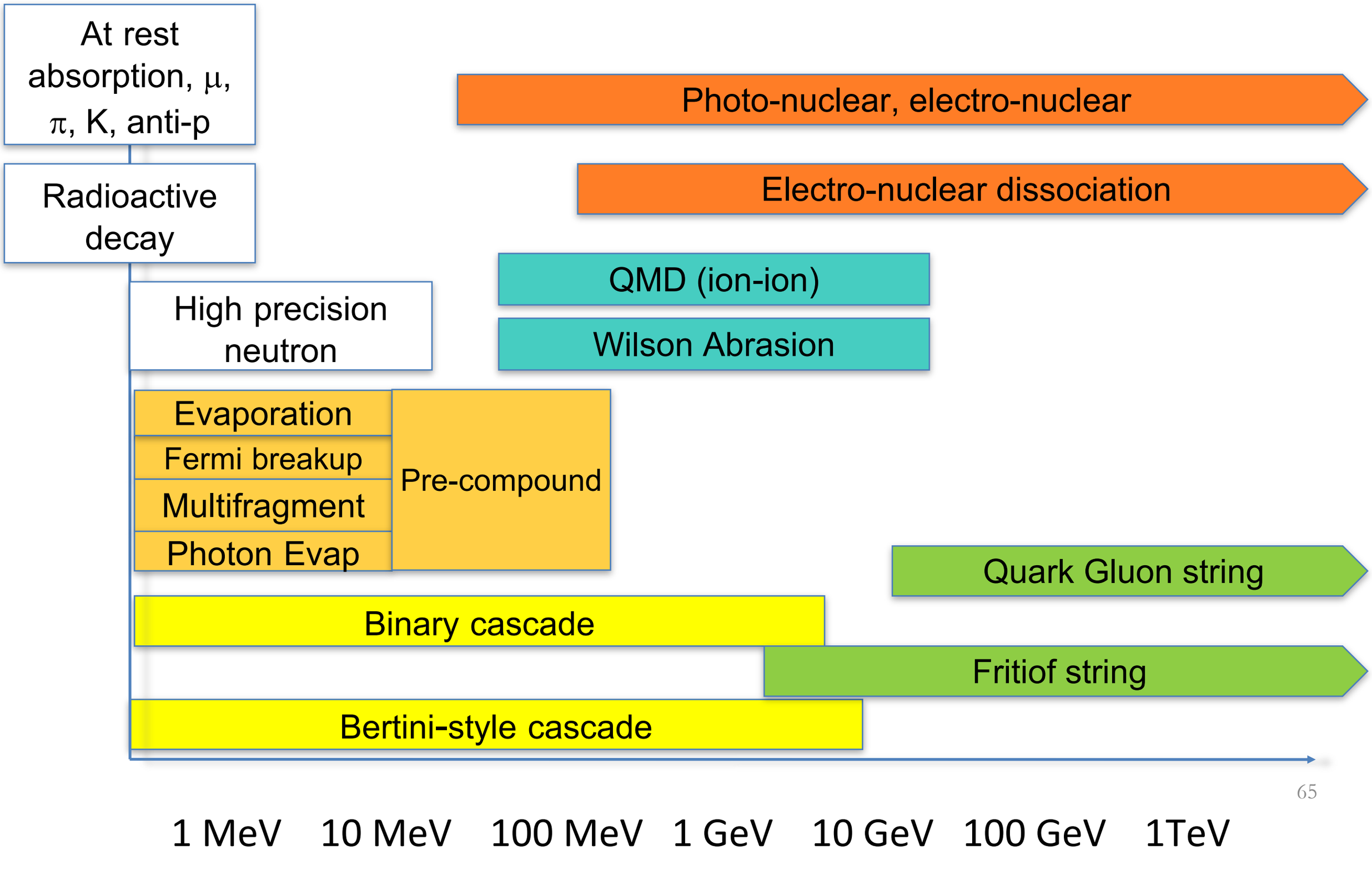


Production range = 1.5 mm

Hadronic

- hadrons at rest
  - π, K absorption
  - neutron capture
  - anti-proton, anti-neutron annihilation
  - μ- capture
- hadrons in flight
  - inelastic scattering
  - elastic scattering
  - fission
  - neutron, anti-neutron capture

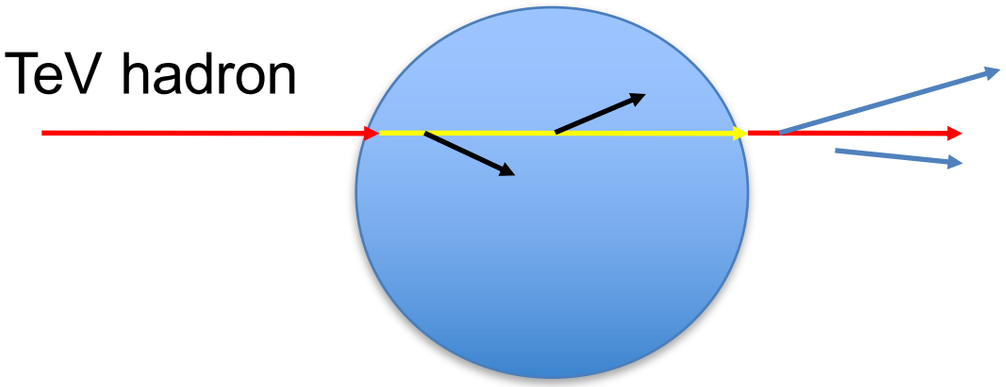
Partial hadronic model inventory



Hadronic Interactions from TeV to meV

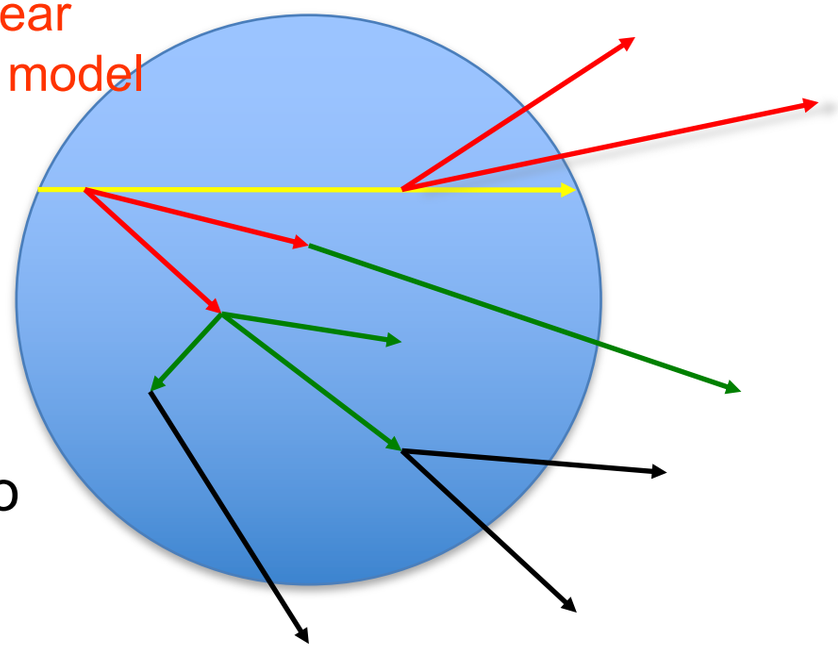
High energy QCD string model

TeV hadron



dE/dx ~ $A^{1/3}$ GeV

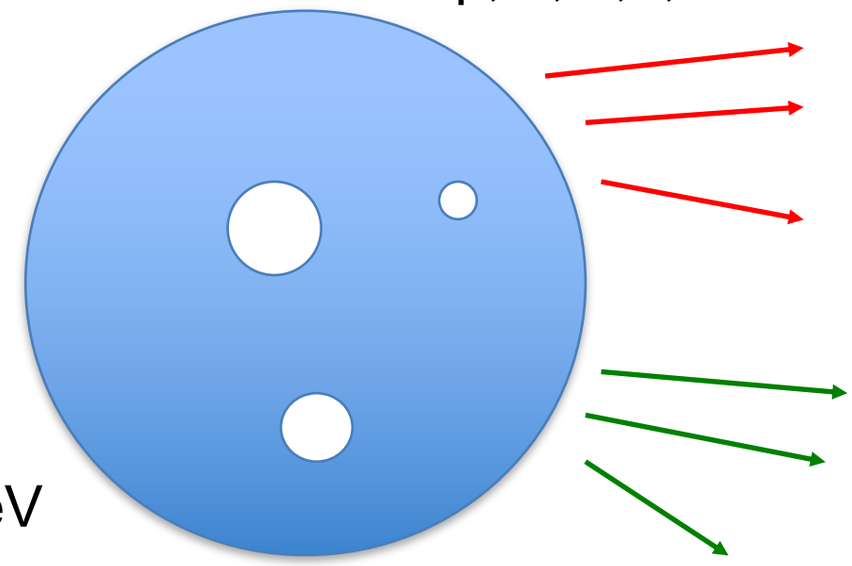
Intranuclear cascade model



~GeV to ~100 MeV

pre-compound/de-excitation

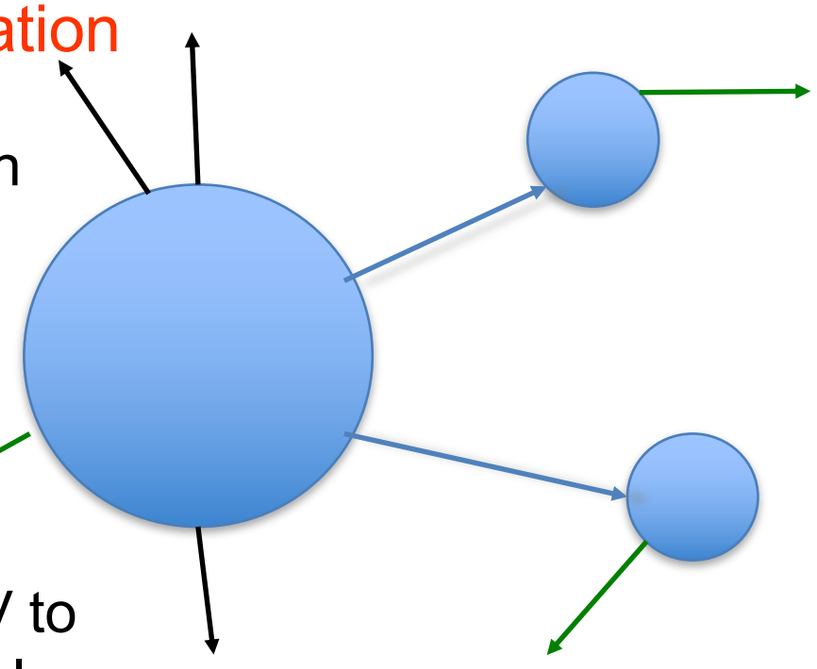
p, n, d, t,



~100 MeV to ~10 MeV

Evaporation

 and n

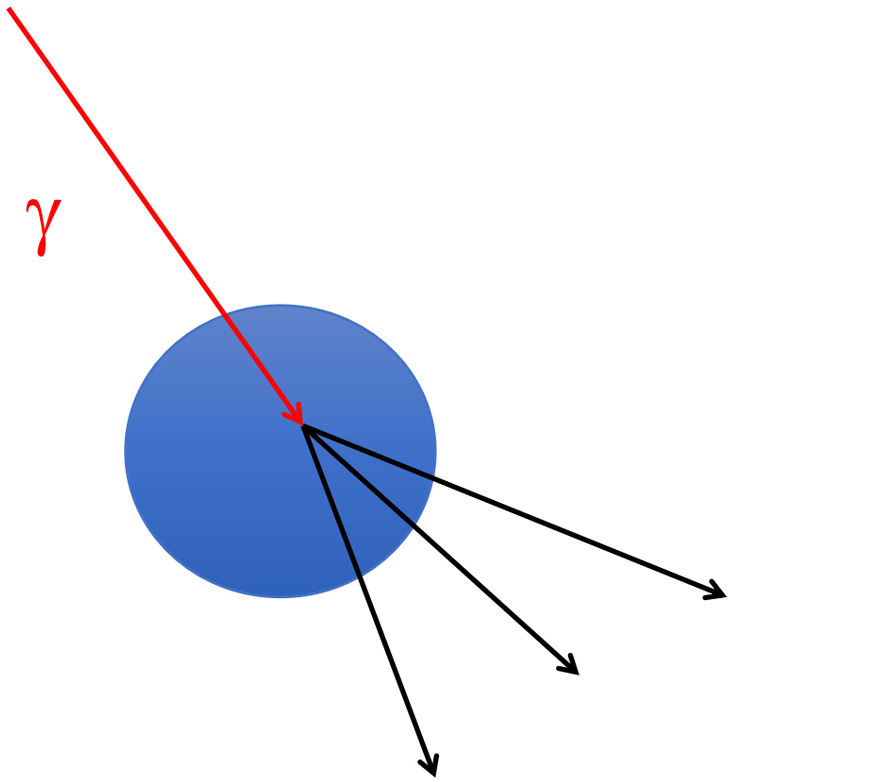


~10 MeV to thermal

Gamma- and Lepto-nuclear Models (1/2)

- Gammas interact directly with the nucleus
  - at low energies they are absorbed and excite the nucleus as a whole
  - at high energies they act like hadrons (pion, rho, etc.) and form resonances with protons and neutrons

Gamma-nuclear

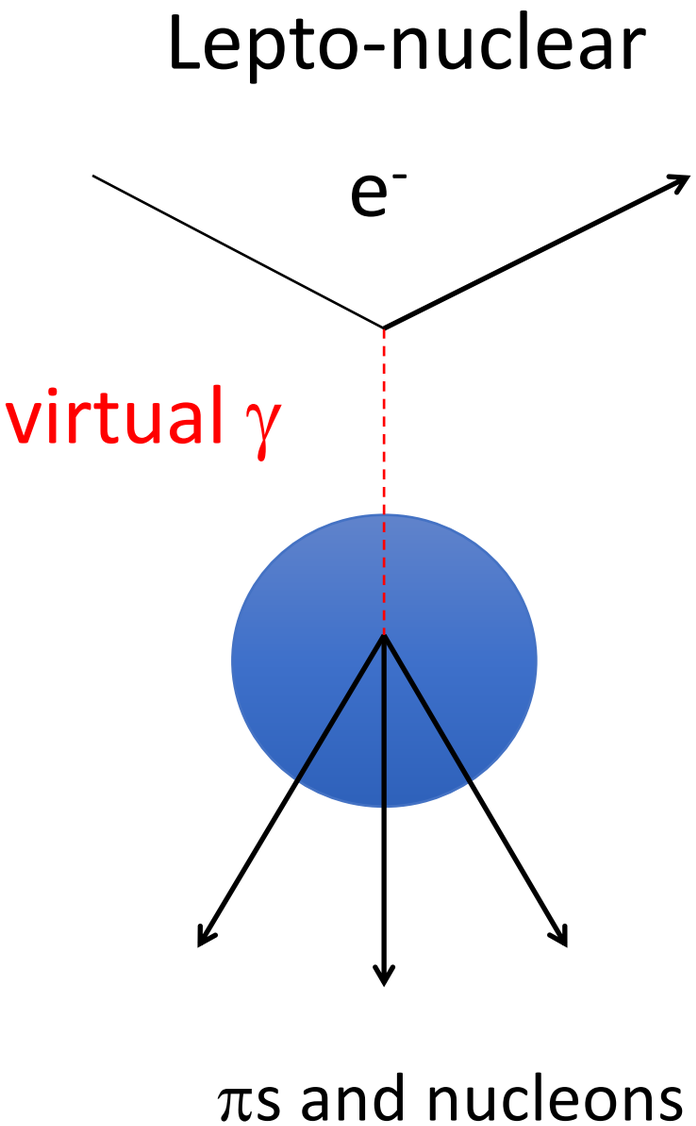


Gamma- and Lepto-nuclear Models (2/2)

- Electrons and muons
cannot interact hadronically, except through virtual photons

- electron or muon passes by a nucleus and exchanges virtual photon

- virtual photon then interacts directly with nucleus (or nucleons within nucleus)

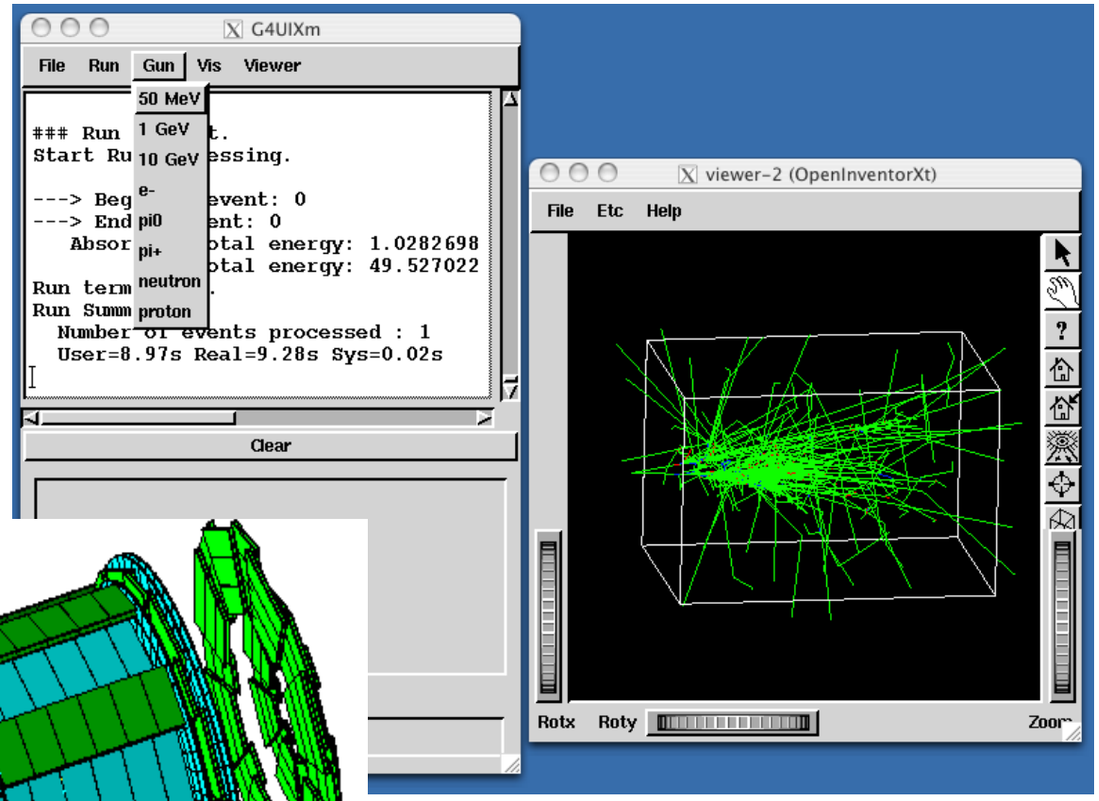


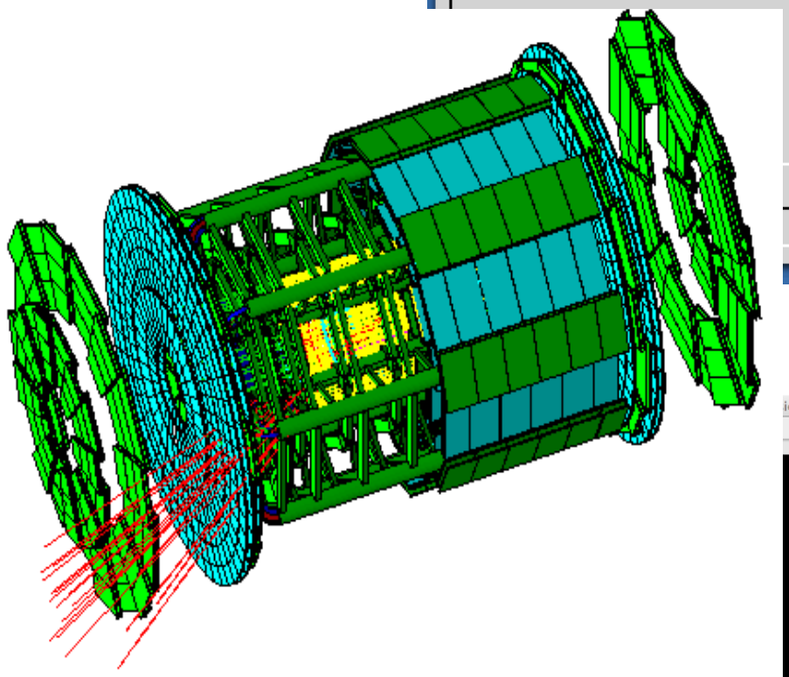
Geant4 ‘Physics Lists’

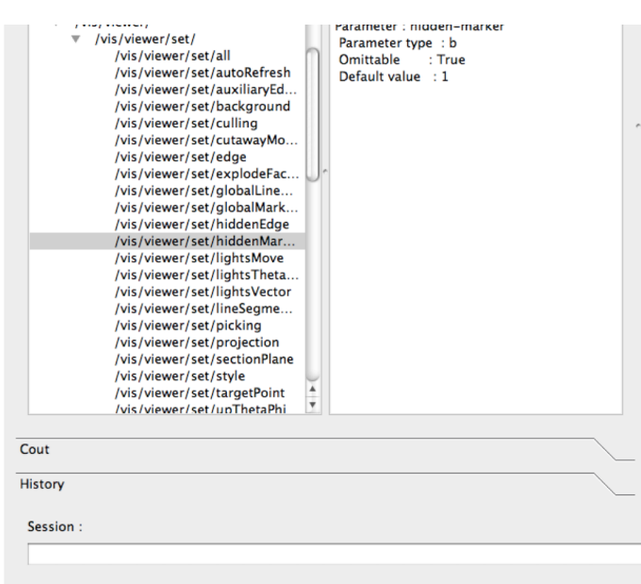
- set of physics models covering all the needed processes for the relevant particles and cross sections for them
- predefined physics lists for specific applications
  - general high energy physics
  - high precision high energy physics (detailed tracking of neutrons
  - low energy
  - etc

Visualisation

- visualize
  - geometry
  - tracks
  - hits
- available visualization drivers
  - OpenGL
  - OpenInventor
  - HepRep
  - DAWN
  - VRML
  - RayTracer
  - gMocren
  - ASCIITree





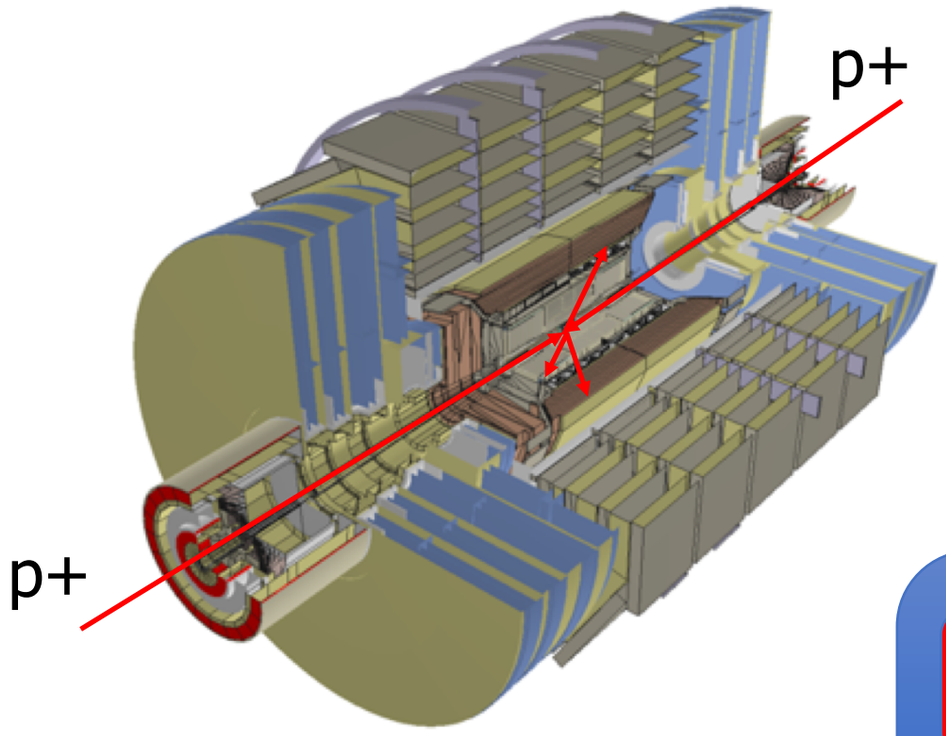


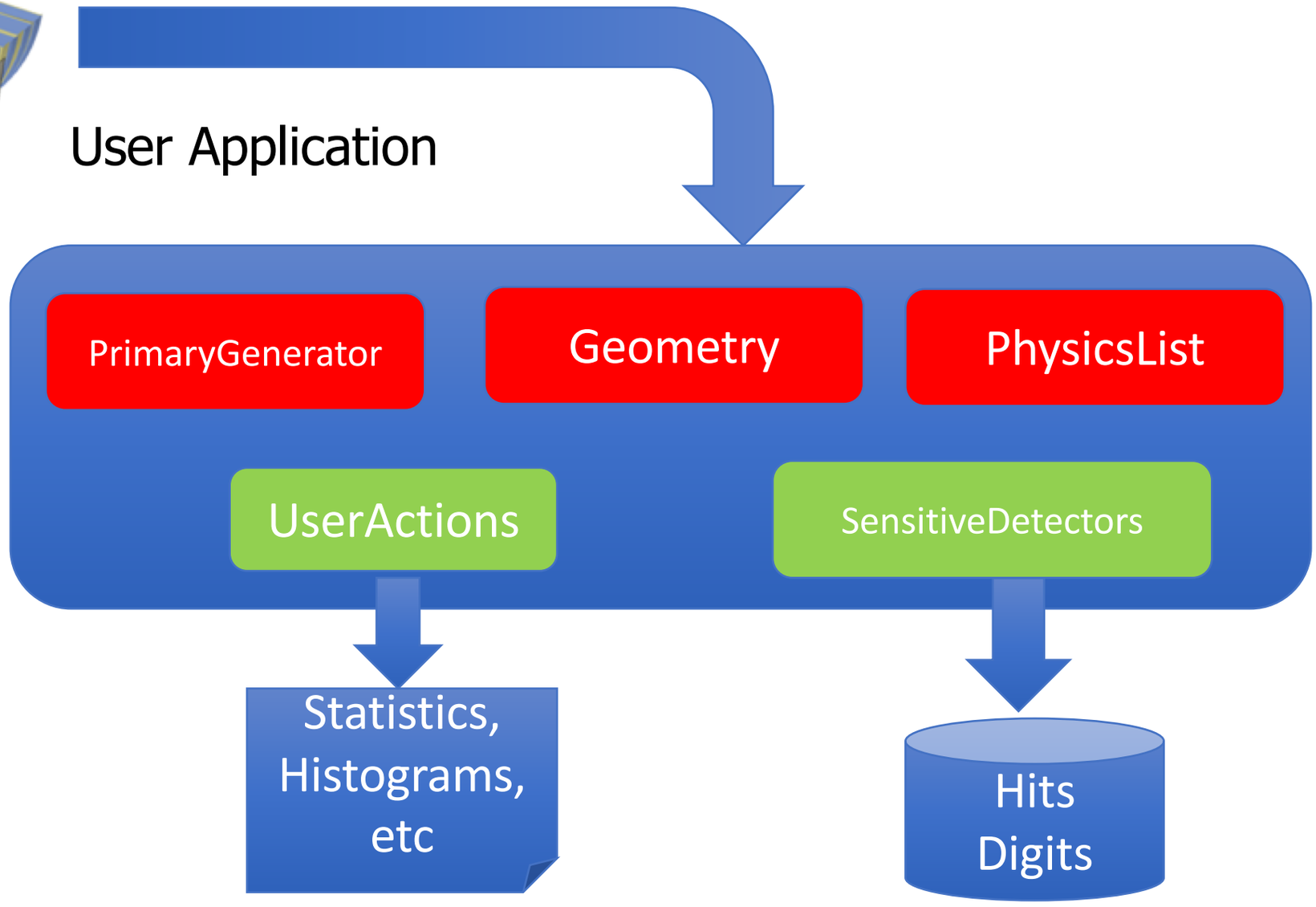
Building a concrete Geant4 application

Let’s summarize

- What does a typical G4 application consist of?
- What is there to reuse?
- What do you need to implement and how?

What do we need to run simulation?





- Given geometry, physics and primary track generation, Geant4 does proper physics simulation "silently".
  - You have to add a bit of code to extract information useful to you.
- The user action classes, if provided, are called by Geant4 kernel during all phases of tracking

Your first Geant4 application recipe

- geometry
- primary generator
- user actions
- give commands to UI manager

```
# e+ 200MeV
/gun/energy 200 MeV
/gun/particle e+
/run/beamOn 1
```

- look into Geant4 basic examples
  - in the examples directory

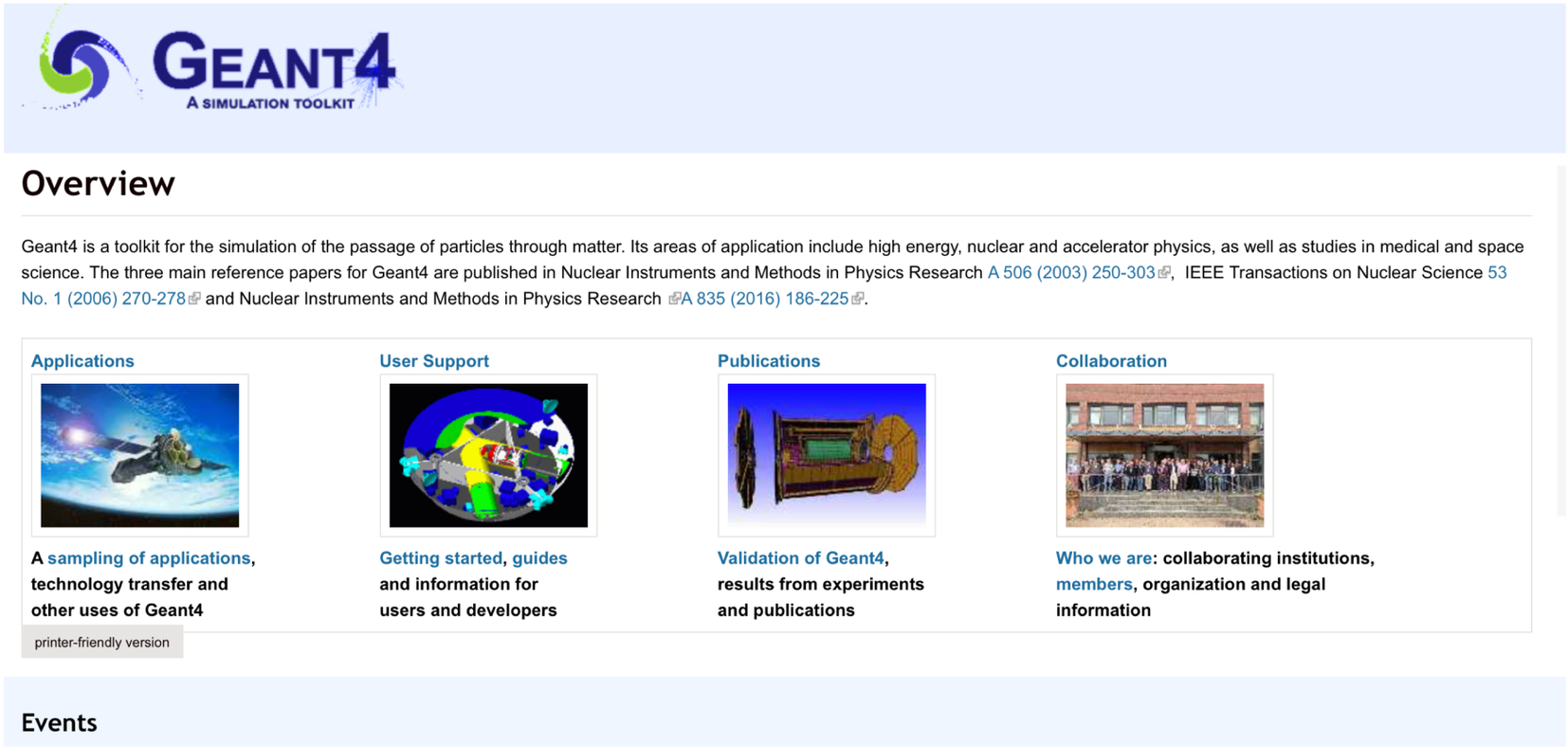
You must implement

```
int main(int argc,char** argv)
{
  // Construct the default run manager

  G4RunManager * runManager = new G4RunManager;

  // Set mandatory initialization classes

  runManager->SetUserInitialization(new B2aDetectorConstruction());

  G4VModularPhysicsList* physicsList = new FTFP_BERT;

  runManager->SetUserInitialization(physicsList);

  // Set user action classes

  runManager->SetUserAction(new B2PrimaryGeneratorAction());
  runManager->SetUserAction(new B2RunAction());
  runManager->SetUserAction(new B2EventAction());

  // Initialize G4 kernel

  runManager->Initialize();

  // Get the pointer to the User Interface manager
  G4UImanager* UImanager = G4UImanager::GetUIpointer();


  G4UIExecutive* ui = new G4UIExecutive(argc, argv);

  return 0;
}

//....oooOOOOOooo.......oooOOOOOooo.......oooOOOOOooo.......oooOOOOOooo.....
```

You can implement

You can reuse existing one

Where to start



- Geant4 web page
  - http://geant4.web.cern.ch
- Geant4 documentation
  - http://geant4.web.cern.ch/support
- Geant4 examples
  - geant4/examples directory in the installation tree

Conclusion

- Geant4 is a Monte Carlo simulation toolkit used in different domains like High Energy Physics, astrophysics, space research, medical physics, biology, etc
  - it allows you to simulate the passage of the particle through the matter, the process their undergo and the energy deposition they make
- the use of such a simulation is essential to build, understand and use the detectors or other radiation-exposed devices