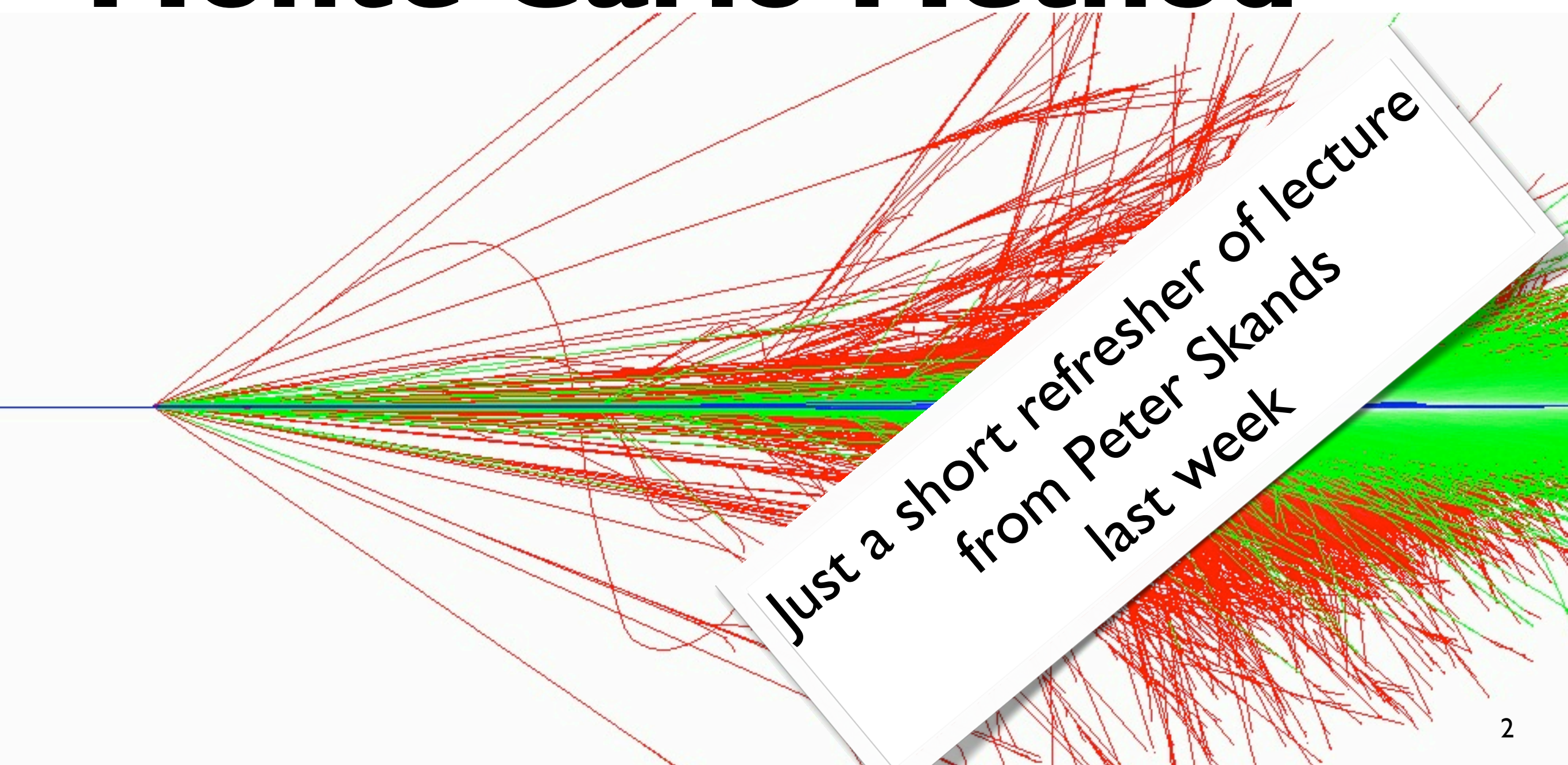


Monte Carlo codes

Second African School of Physics, 2012

Monte Carlo Method



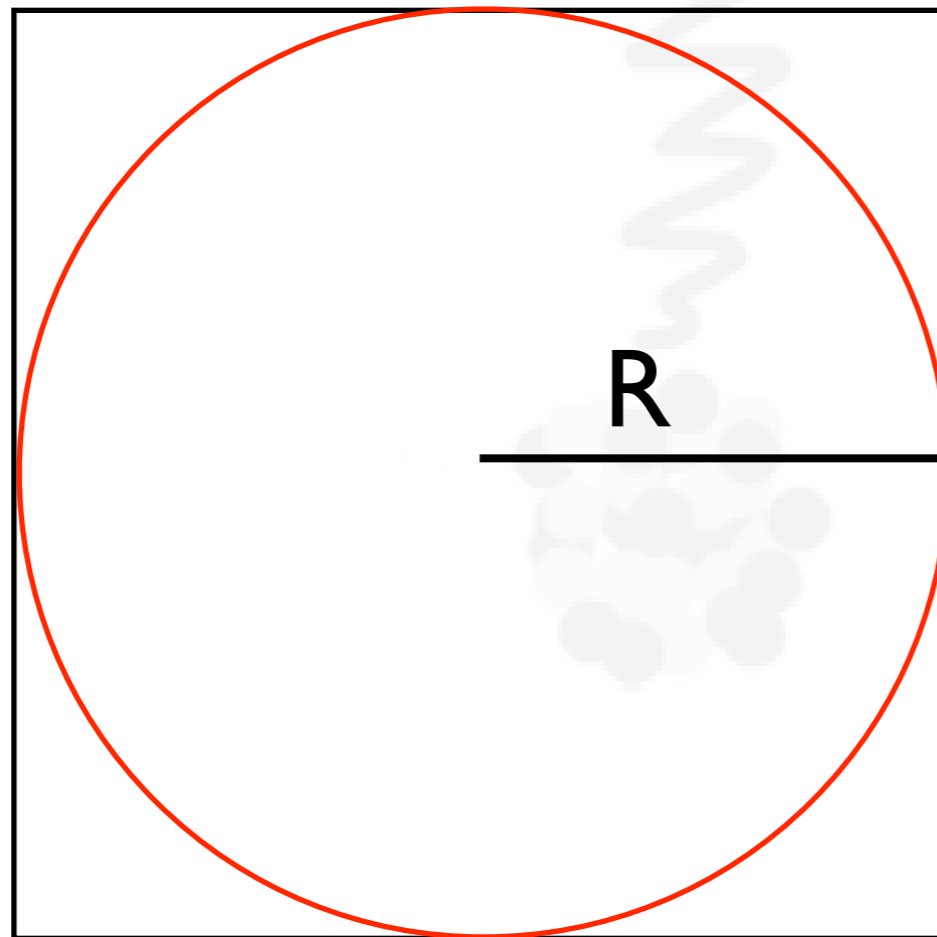
Just a short refresher of lecture
from Peter Skands
last week

What is it?

- The Monte Carlo method is:
 - **A method to perform complex calculations using random numbers**
- For example:
 - **Integrate** a complex function
 - **Approximate** a complex system for which is not possible to completely determine its behavior
- In physics it has a “special” meaning:
 - A **program** or a toolkit to simulate a physical system (e.g. Pythia, Geant4)

An Example

- Let's assume we want to write an **algorithm** to write an approximation of the constant π
- We can see the problem geometrically:



Circle Area:

$$C = \pi R^2$$

Square Area:

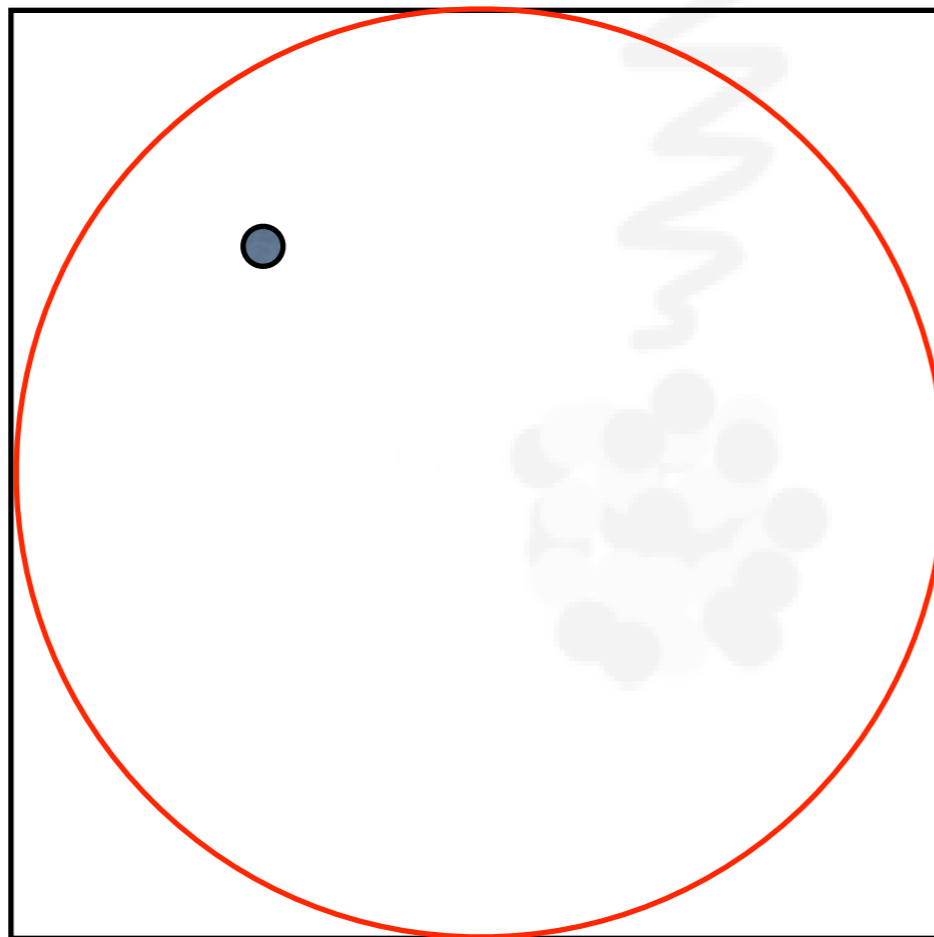
$$S = (2R)^2$$

Ratio:

$$C/S = \pi/4$$

An Example

- Let's draw a point in a random position in the square
- Probability being inside the circle is C/S
- Draw randomly N points, $N_{\text{inside}}/N_{\text{tot}} \approx C/S = \pi/4$



Circle Area:

$$C = \pi R^2$$

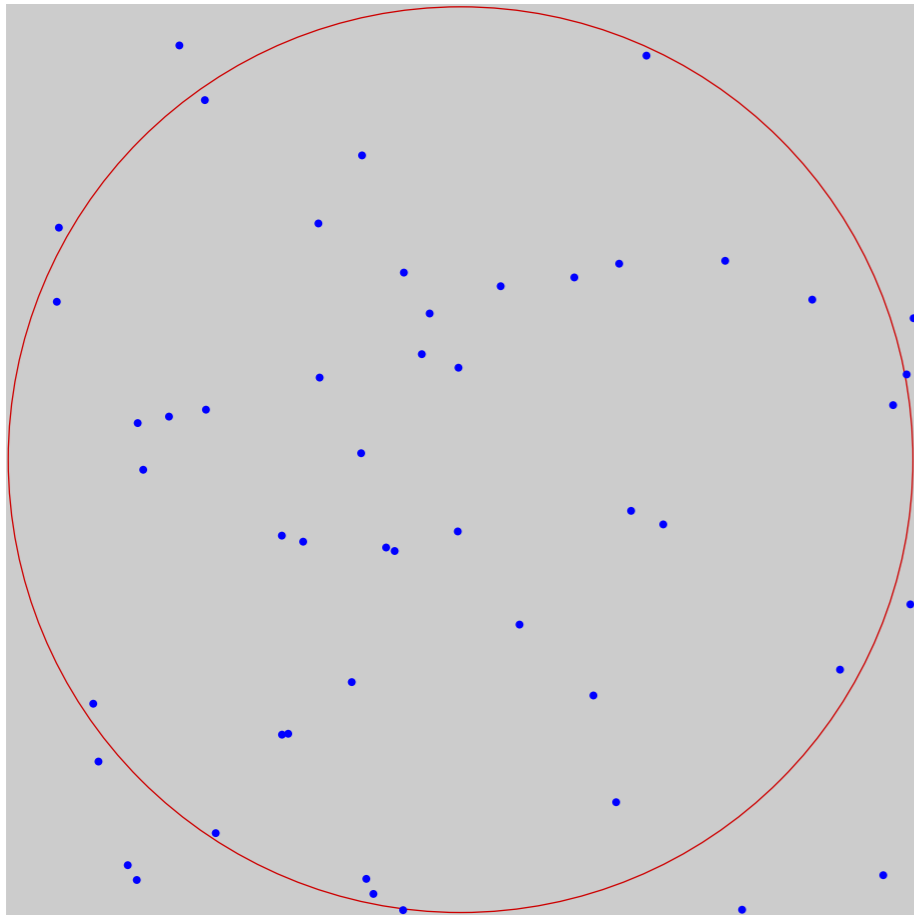
Square Area:

$$S = (2R)^2$$

Ratio:

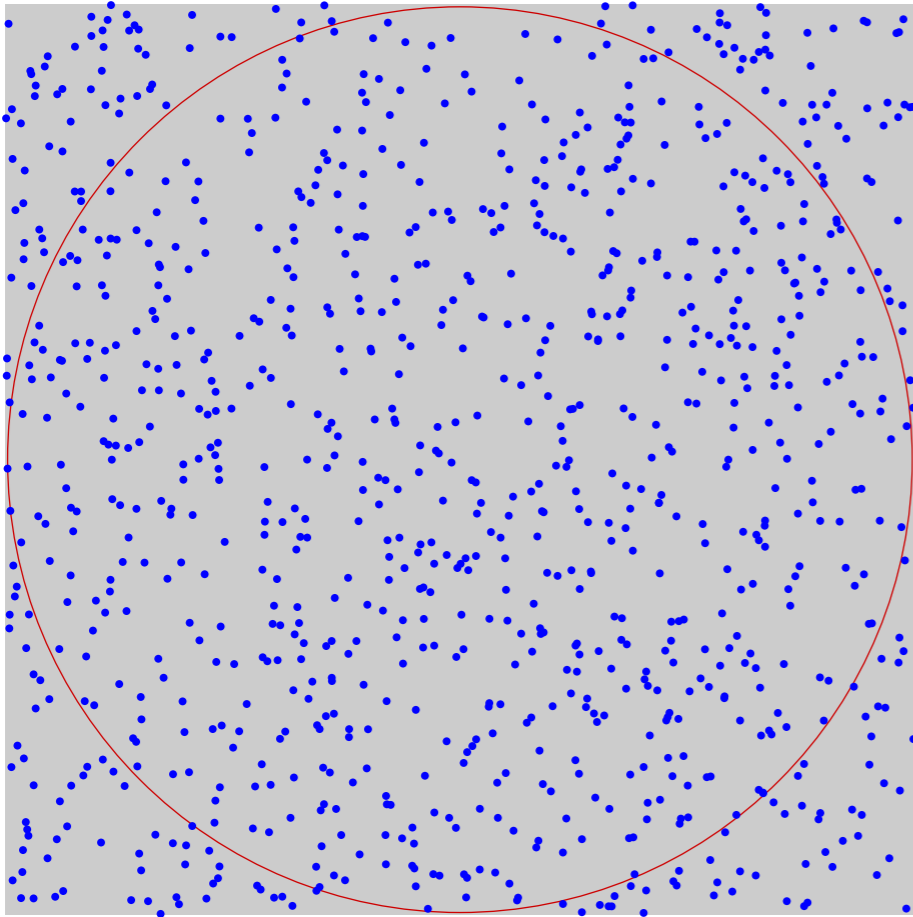
$$C/S = \pi/4$$

N=50



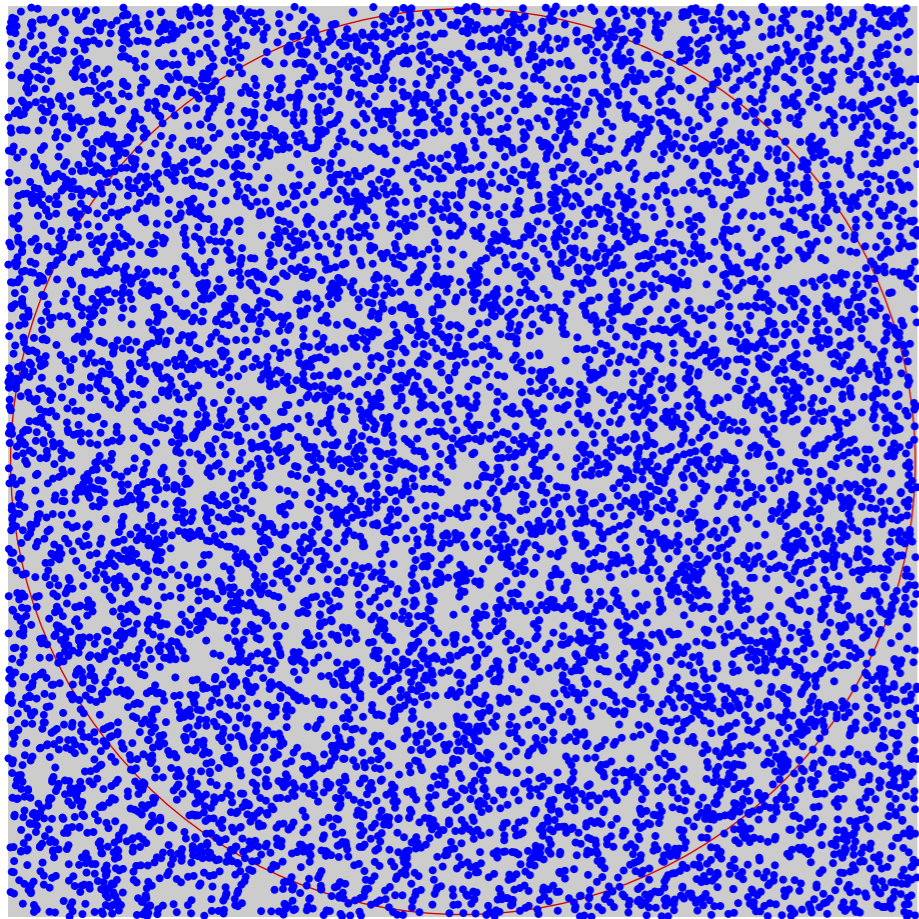
$$\pi_{\text{approx}} = 3.04$$

N=1000



$$\pi_{\text{approx}} = 3.216$$

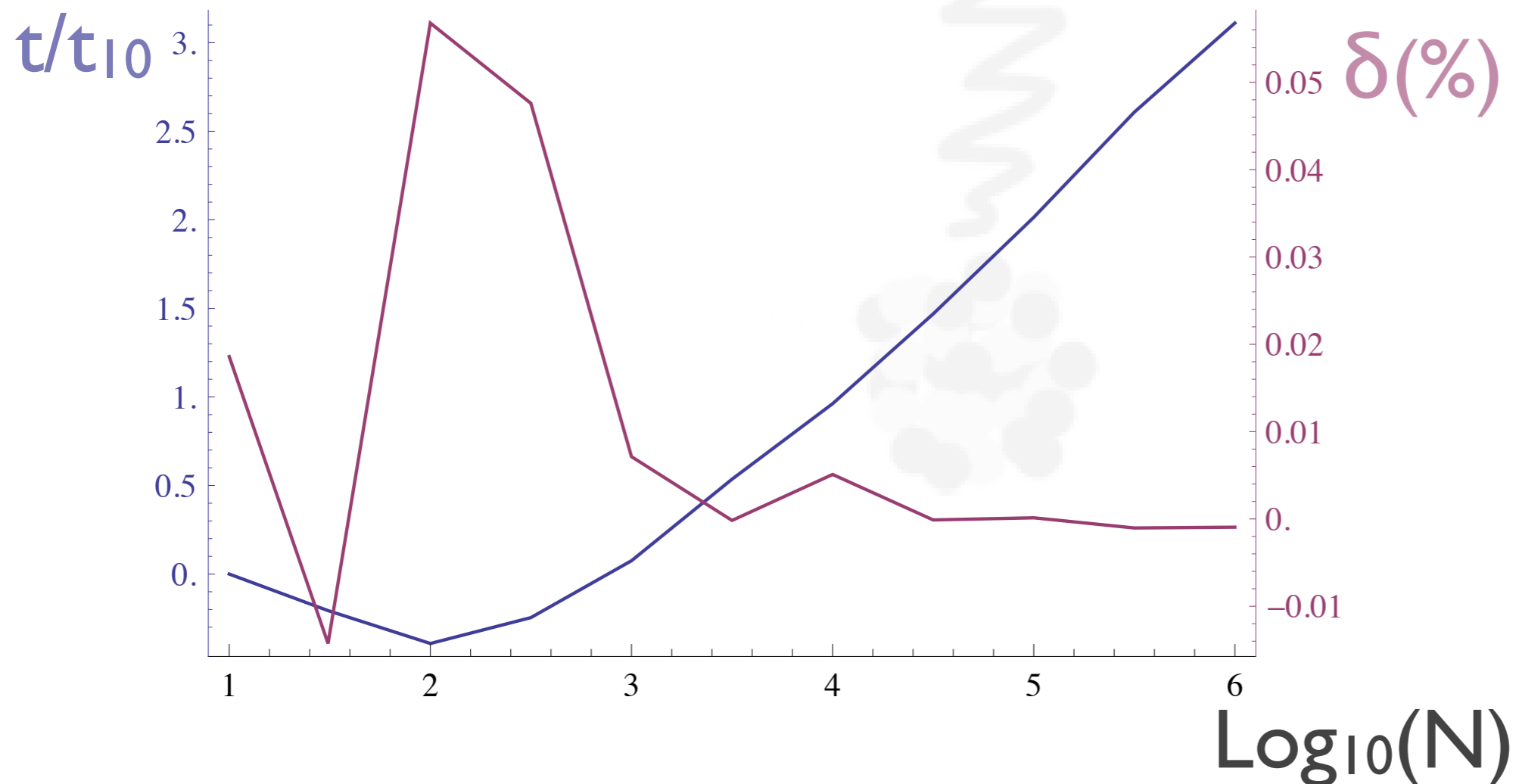
N=10 000



$$\pi_{\text{approx}} = 3.122$$

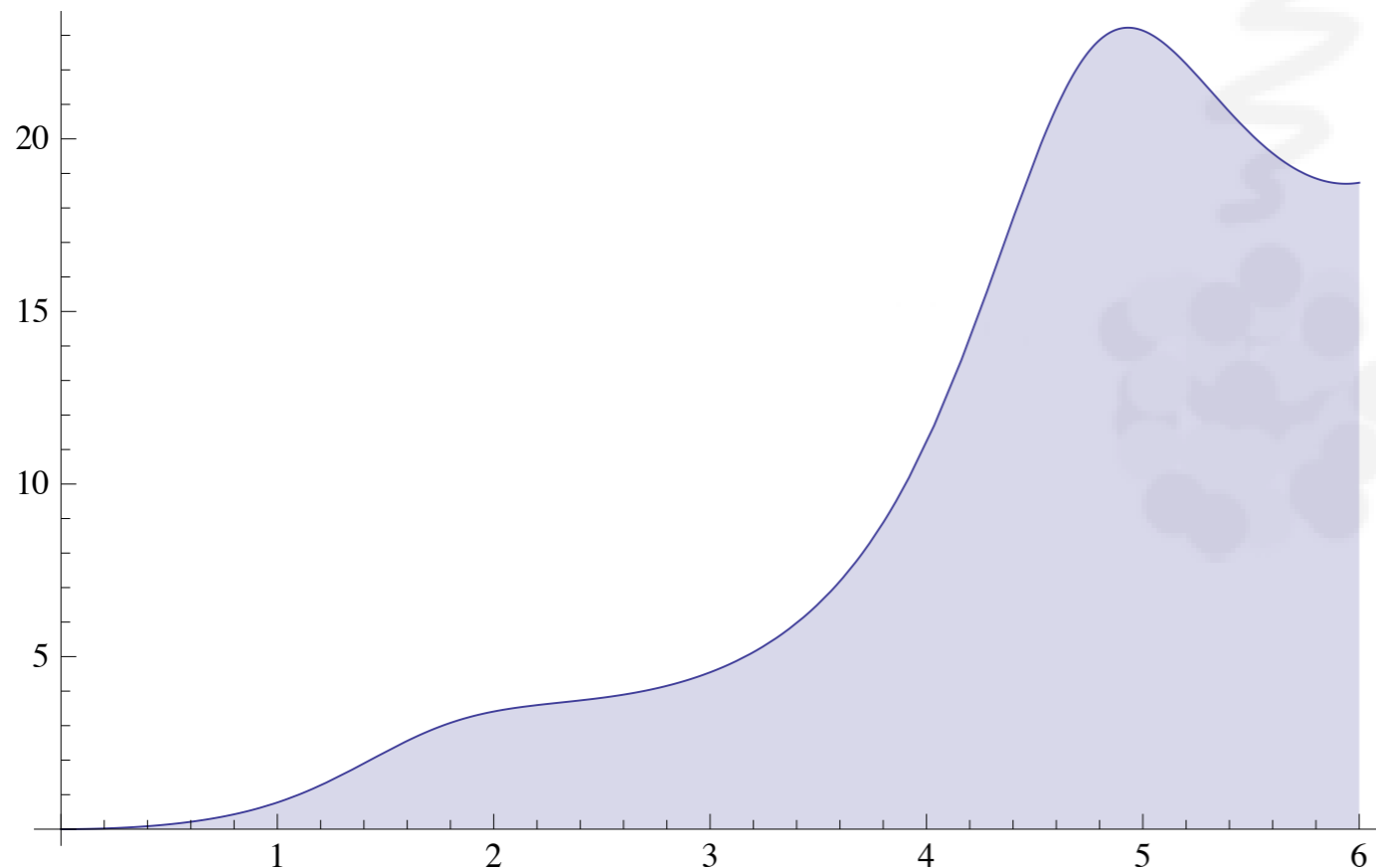
What we have seen

- At the increasing number of N we can get better and better approximations on π
- At the price or longer execution time



The accept-reject method

- We use this method to:
 - Sample a number x distributed according to a general function $f(x)$
 - Throw a pair of random number x, y
 - Calculate $f(x)$
 - If $y < f(x)$ accept x , otherwise repeat



- This is the simplest method, others can be more efficient
- MC methods are “an art” per se and **different algorithms exists:**
 - [1] J.C. Butcher and H. Messel. Nucl. Phys. 20 15 (1960)
 - [2] H. Messel and D. Crawford. Electron-Photon shower distribution, Pergamon Press (1970)
 - [3] R. Ford and W. Nelson. SLAC-265, UC-32 (1985)

The one used in G4

— If the normalized PDF for $x \in [x_1, x_2]$ can be written as:

$$f(x) = \sum_{i=1}^n N_i f_i(x) g_i(x)$$

— With $0 \leq g_i(x) \leq 1; N_i > 0$

— x can be sampled with:

— Select a random integer $i \in \{1, 2, \dots, n\}$ with probability proportional to N_i

— Select x_0 with probability $f_i(x_0)$

— Accept $x = x_0$ with probability $g_i(x_0)$

— With this method you need to throw 5 random numbers and calculate 2 functions, but:

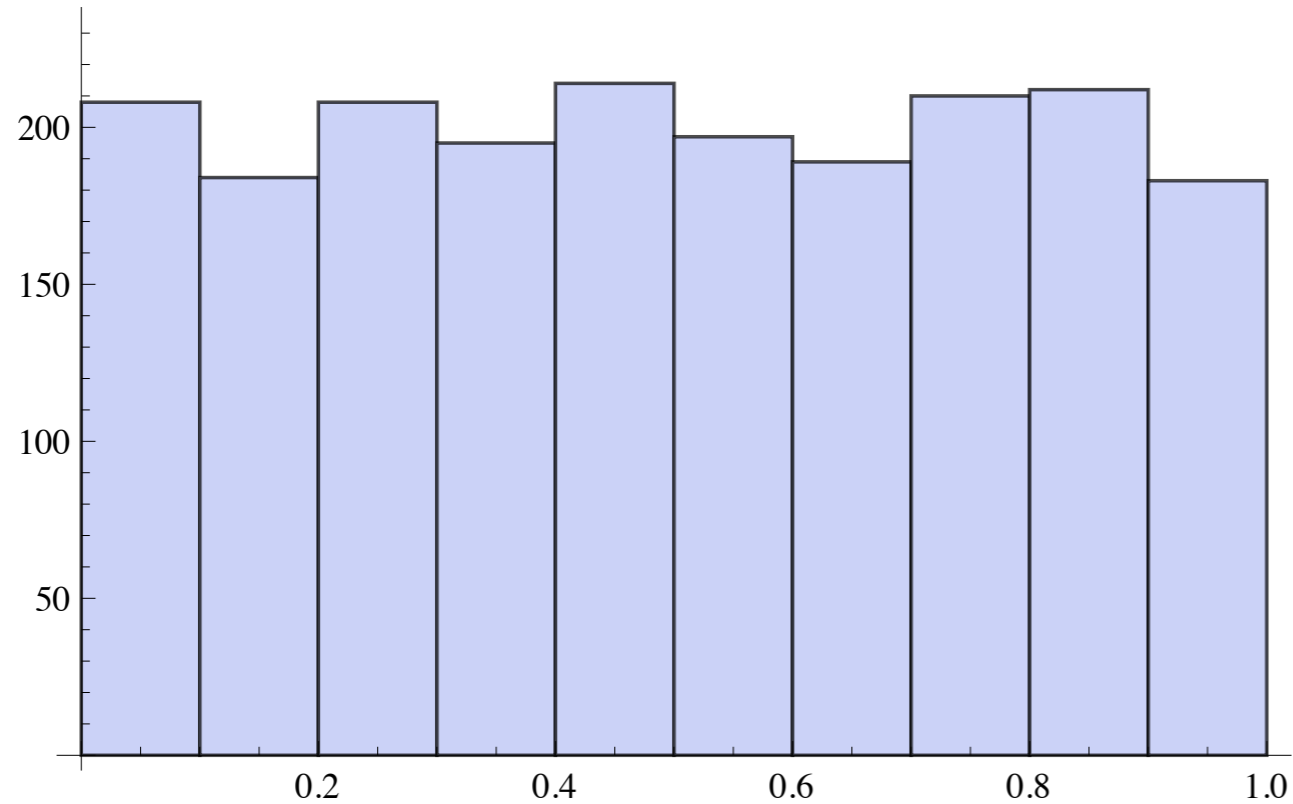
— The method can be fast if f_i, g_i are considerably simpler than f

A word on random numbers

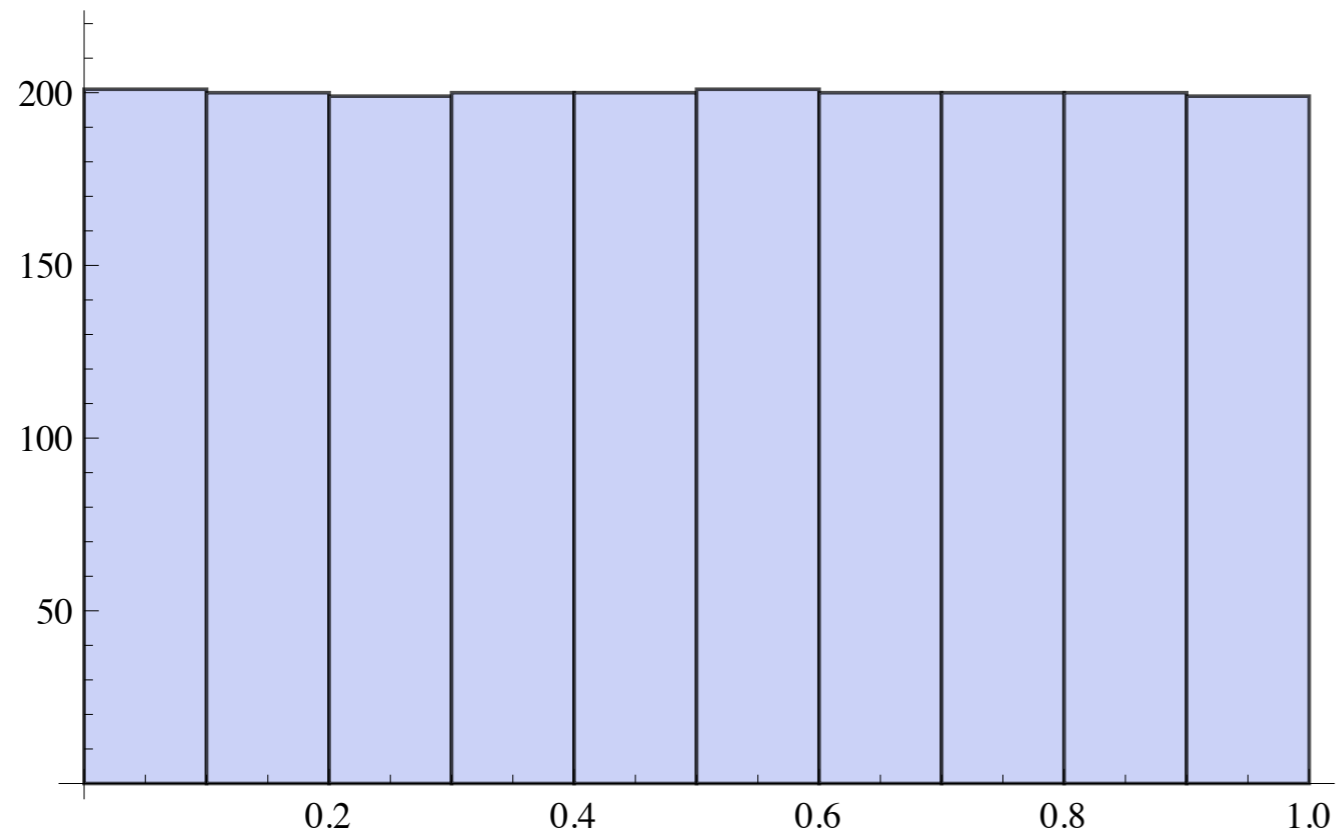
- Computers cannot generate **pure** random numbers
- They are Turing machines (fully deterministic):
 - Each **state is determined** by the previous state and the operation to be performed
- But you can generate a **“pseudo” random number**
 - A number that looks “random” at any practical purpose
 - Simply called “random number”
- Given a function f :
 - Start from a “seed” x_0 : $x_1=f(x_0)$; $x_2=f(x_1)$; ... ; $x_{n+1}=f(x_n)$
 - **f needs to have some properties:**
 - The period has to be large (the series of x repeats only after very large steps)
 - The correlation between two numbers has to be small
 - Simple functions involve mod operation and polynomial
- **Example ex2.cpp in the “Mini C++ Tutorial”**

- For the purpose of our simulation programs:
 - Given a random seed
 - And in the absence of software bugs
- Each time you run it will reproduce exactly the same results
- **This is a good property:**
 - In case of problems you can re-run the simulation of a specific setup and debug your code

2000 random numbers [0, 1]



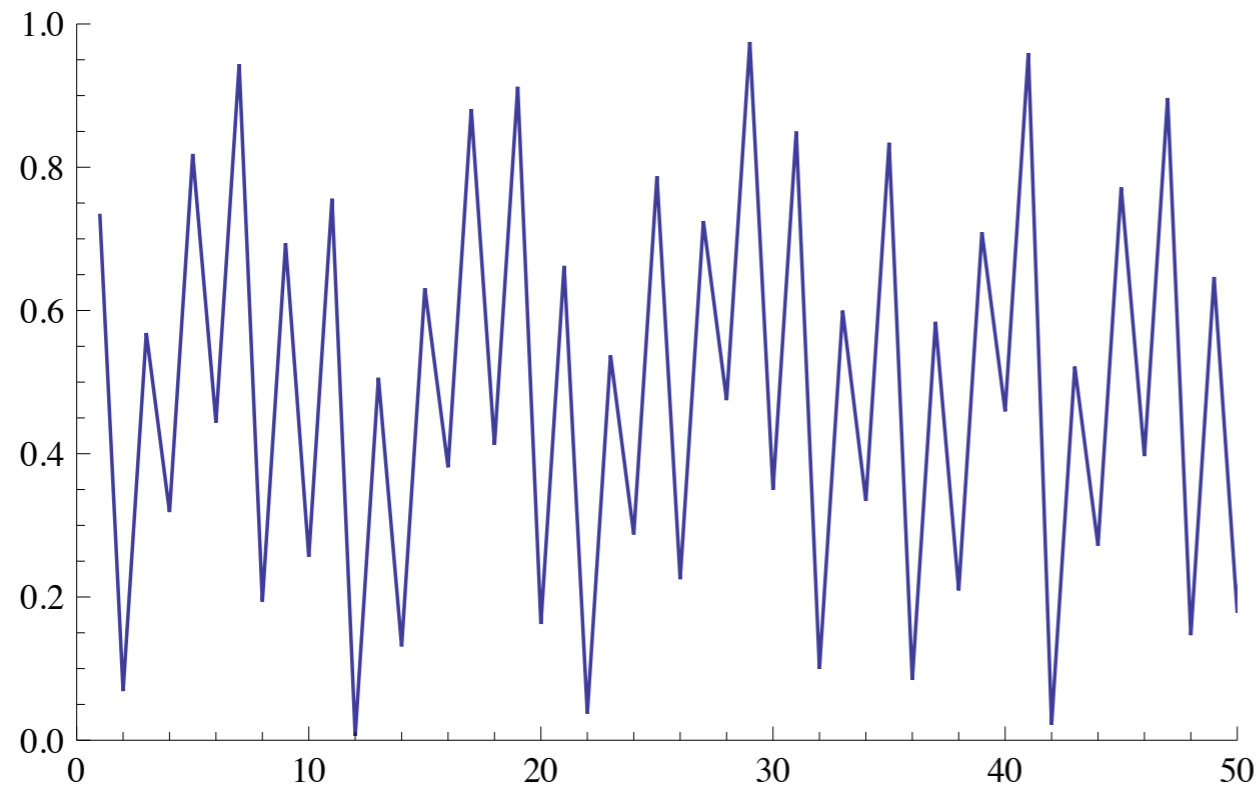
Niederreiter generator



extended cellular
automaton generator

Which is better?

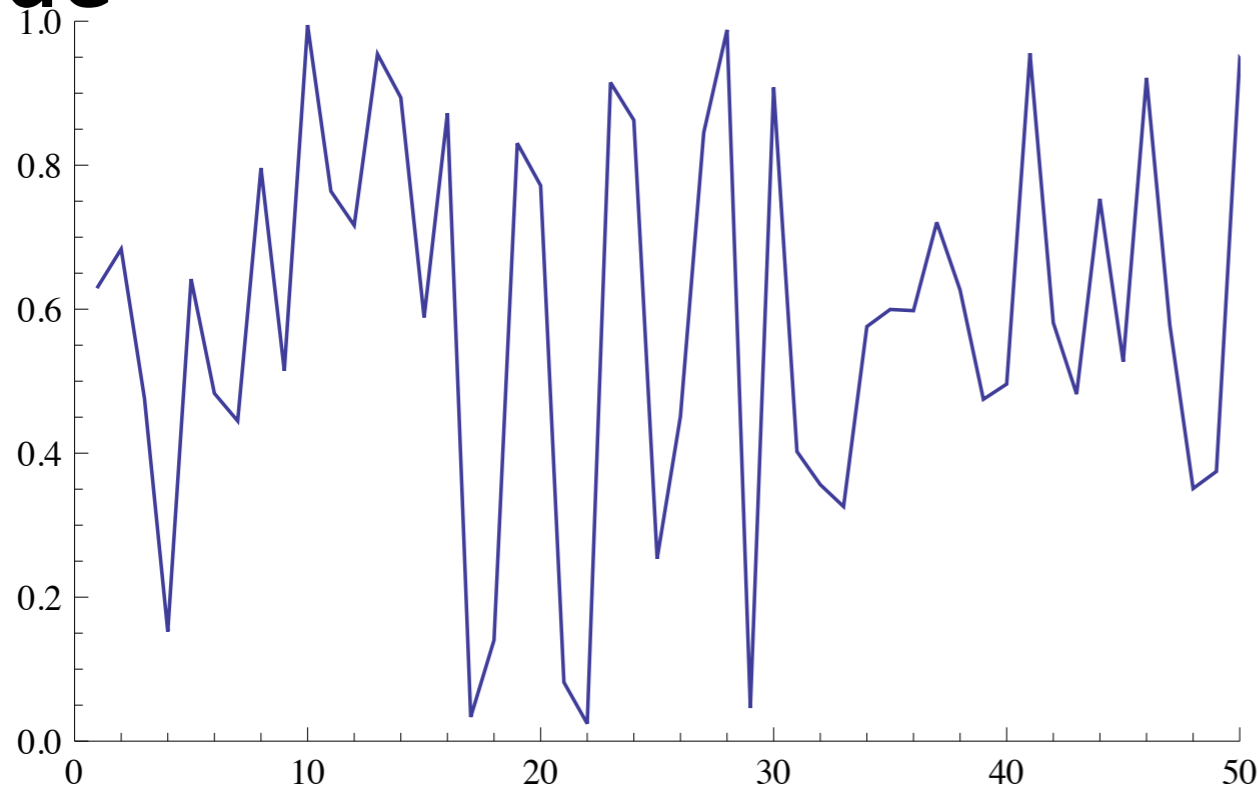
Value



Niederreiter generator

Sequence #

Value



extended cellular
automaton generator

Sequence #

Which is better?

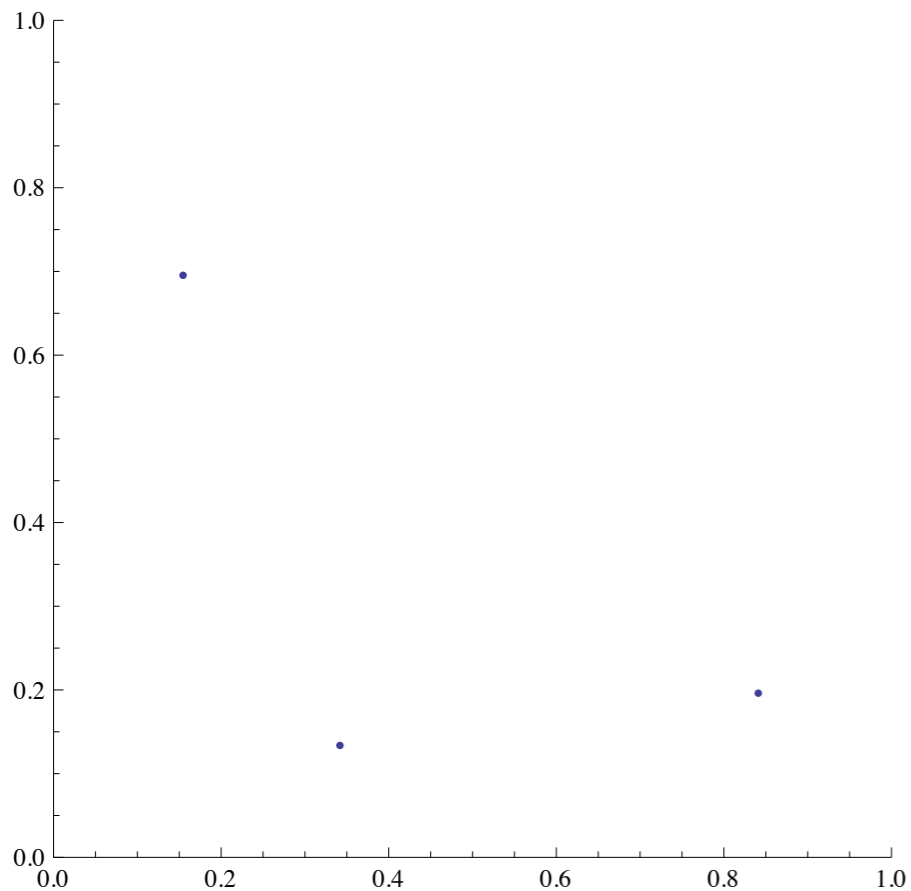
$$\text{point}_n = \{x, y\} = \{r_{2n}, r_{2n+1}\}; n=0, \dots, 999$$

$$\text{point}_0 = \{r_0, r_1\}$$

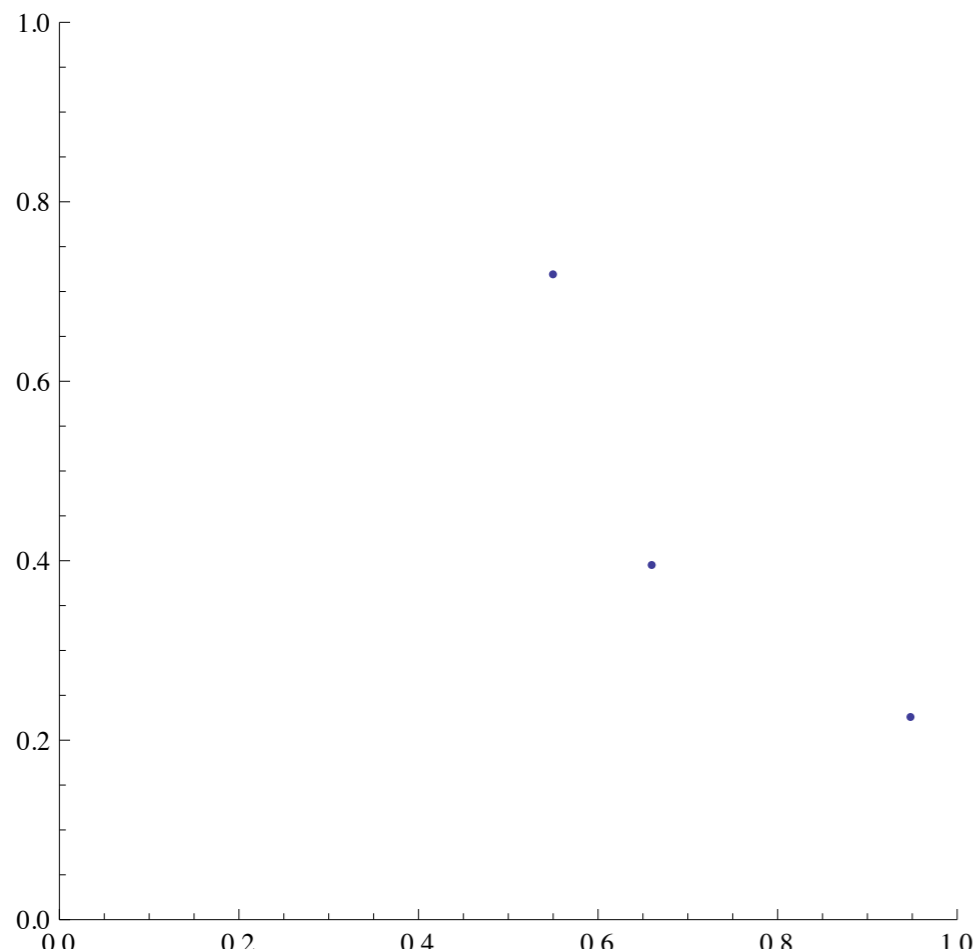
$$\text{point}_1 = \{r_2, r_3\}$$

$$\text{point}_2 = \{r_4, r_5\}$$

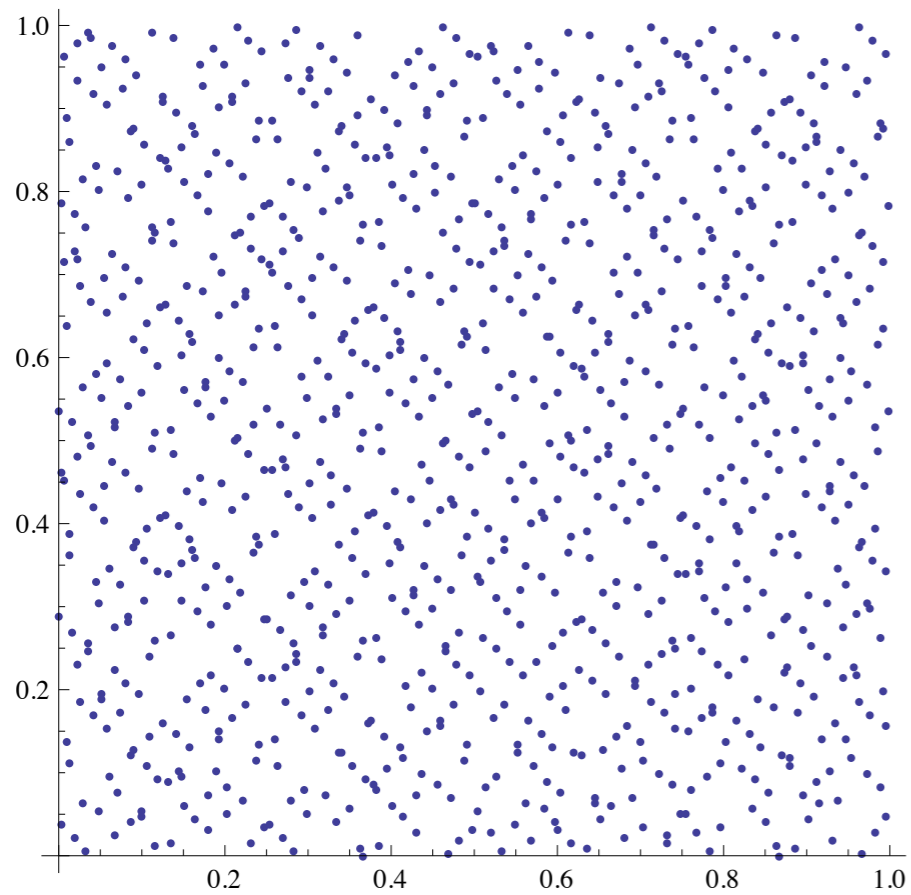
Niederreiter generator



extended cellular
automaton generator



Which is better?



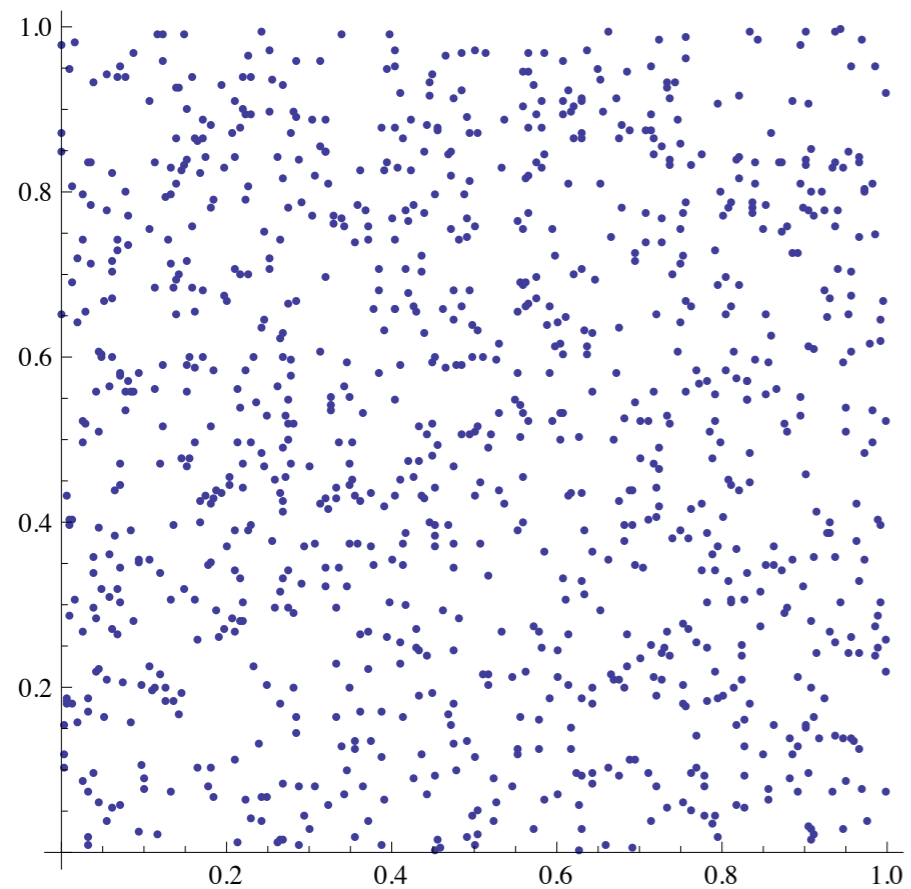
$$\text{point}_n = \{x, y\} = \{r_{2n}, r_{2n+1}\}; n=0, \dots, 999$$

$$\text{point}_0 = \{r_0, r_1\}$$

$$\text{point}_1 = \{r_2, r_3\}$$

$$\text{point}_2 = \{r_4, r_5\}$$

Niederreiter generator

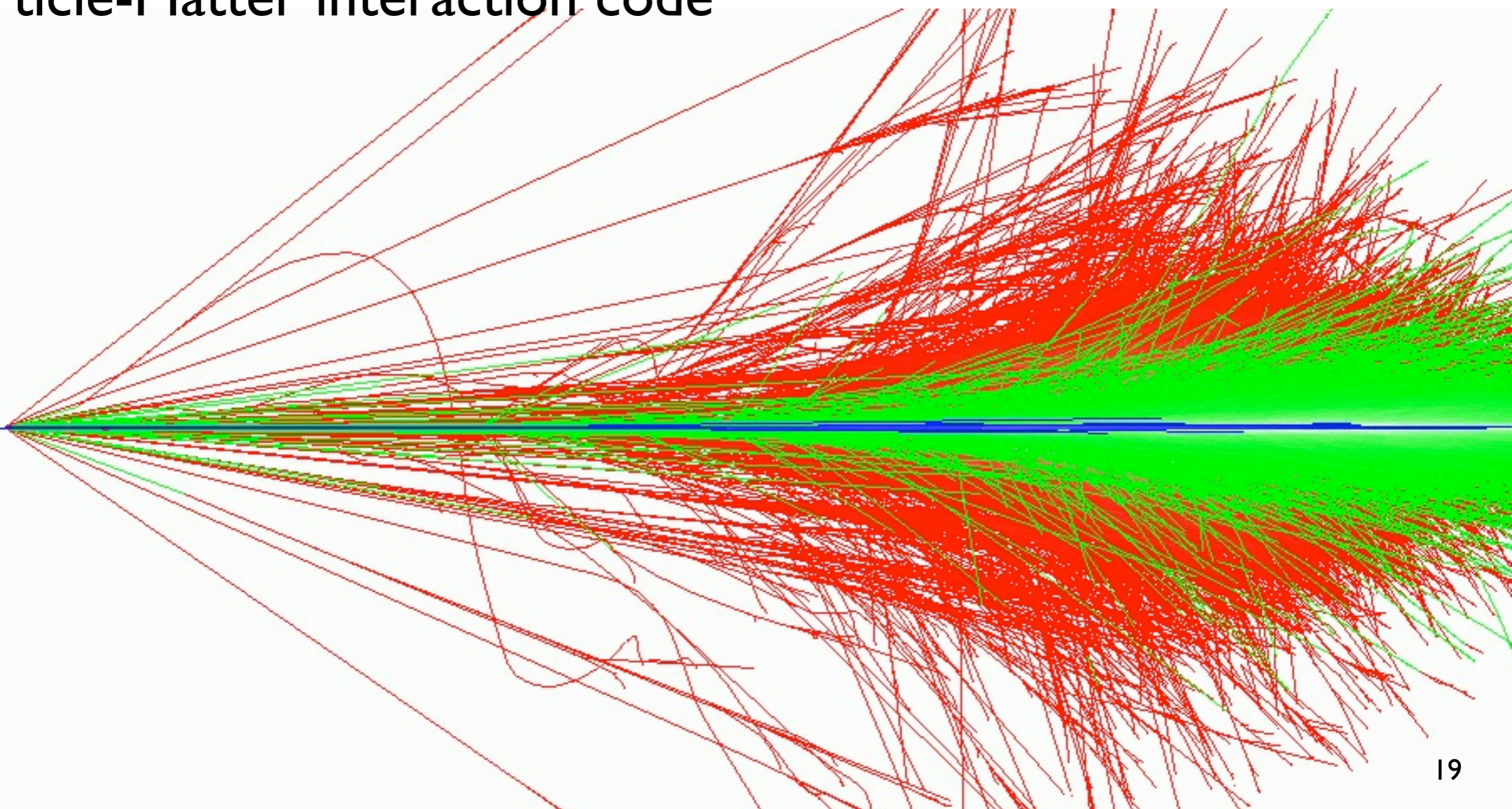


extended cellular
automaton generator

Which is better?

Simulations

Particle-Matter interaction code



- This lecture is aimed to offer a **simple and general** introduction to detector simulation.
- Geant4 will be considered as a concrete example but only to illustrate general aspects of detector simulation.
 - This lecture is not a tutorial on how to use Geant4 !
- **The best way to learn how to use any simulation package is by starting with an example**
 - For Geant4: download the code and go in sub-dir example, read the files README

Outline

1. Introduction

- Why do we need to simulate a detector?
- How does it work?

2. Geometry

- How do we describe an experimental apparatus?

3. Physics

- What is available and what to use?
- What are the challenges?

Introduction

A decorative graphic consisting of a horizontal line on the left that branches into four lines extending to the right. Below the branching point, a vertical wavy line leads to a cluster of approximately 15 small, light gray circles.

- [Simulation is a very useful, essential tool in modern physics for:
 - [**designing** an experiment/detector (e.g. medical irradiation facility, new HEP experiment)
 - [**analyzing** the data (e.g. now ATLAS, CMS, LHCb)

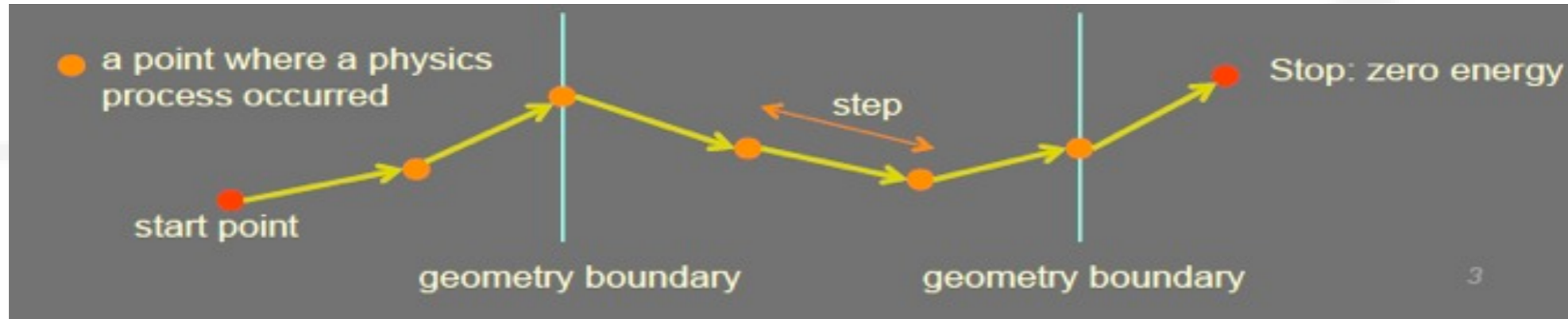
- [Reminder (connection with P. Skands' lecture)
- [For the HEP experiments, the simulation is made of two distinct steps:
 - [1. Simulation of the p-p collision
 - [Monte Carlo event generators (see PYTHIA tutorial)
 - [**2. Simulation of the passage of the produced particles through the experimental apparatus**
 - [Monte Carlo radiation transportation, or simply “detector simulation”
 - [From the beam pipe to the end of the cavern
 - [The output of 1. is the input of 2.

Monte Carlo radiation transportation codes

- The detector simulation is different for each setup. However, **general codes exist** that can be used for simulating any detector
 - An experimental apparatus can be modeled in **terms of elementary geometrical objects**
 - The physics processes are **detector independent**
- These general codes 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

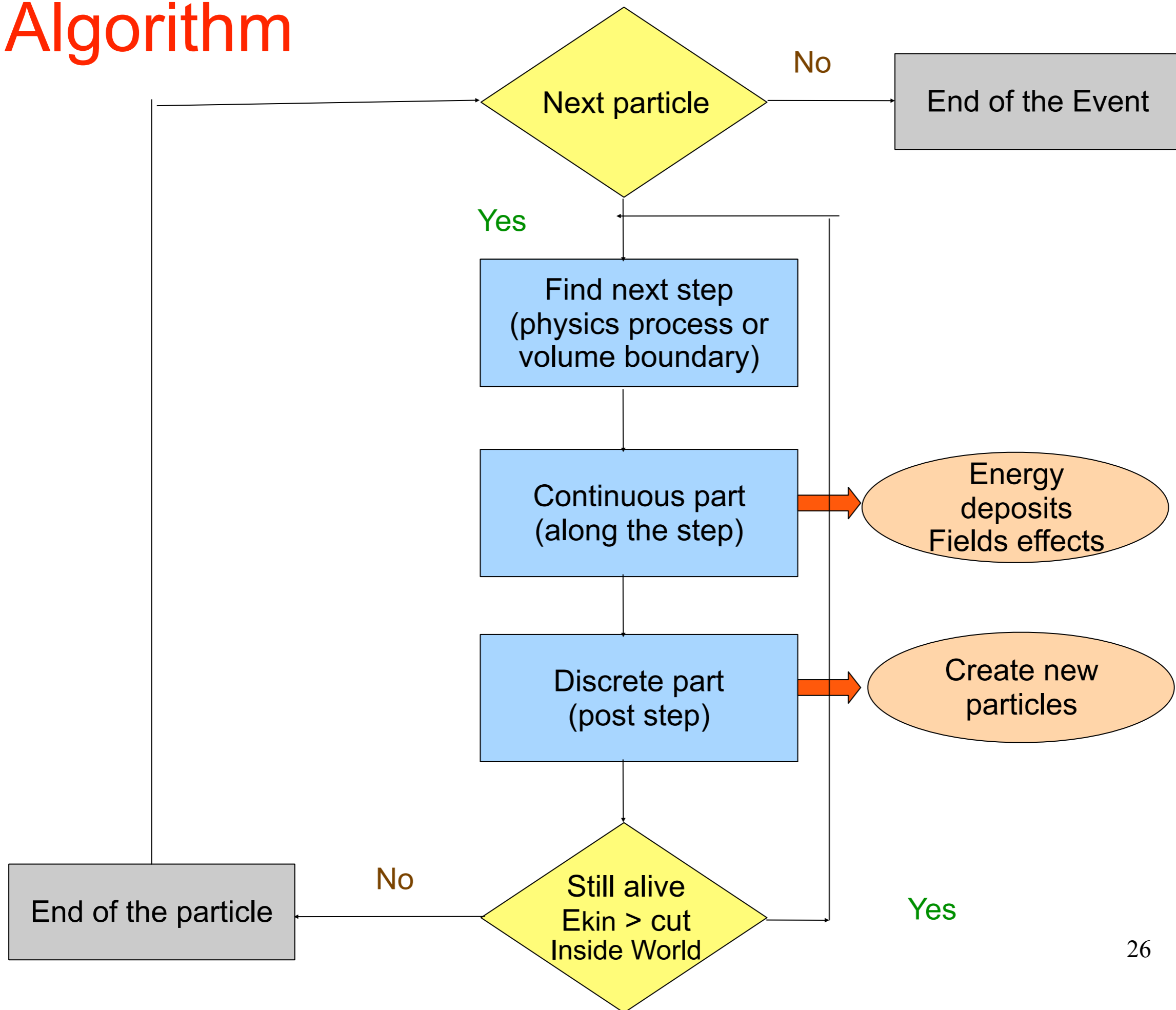
How does it work?

- 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
 - **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)

Algorithm



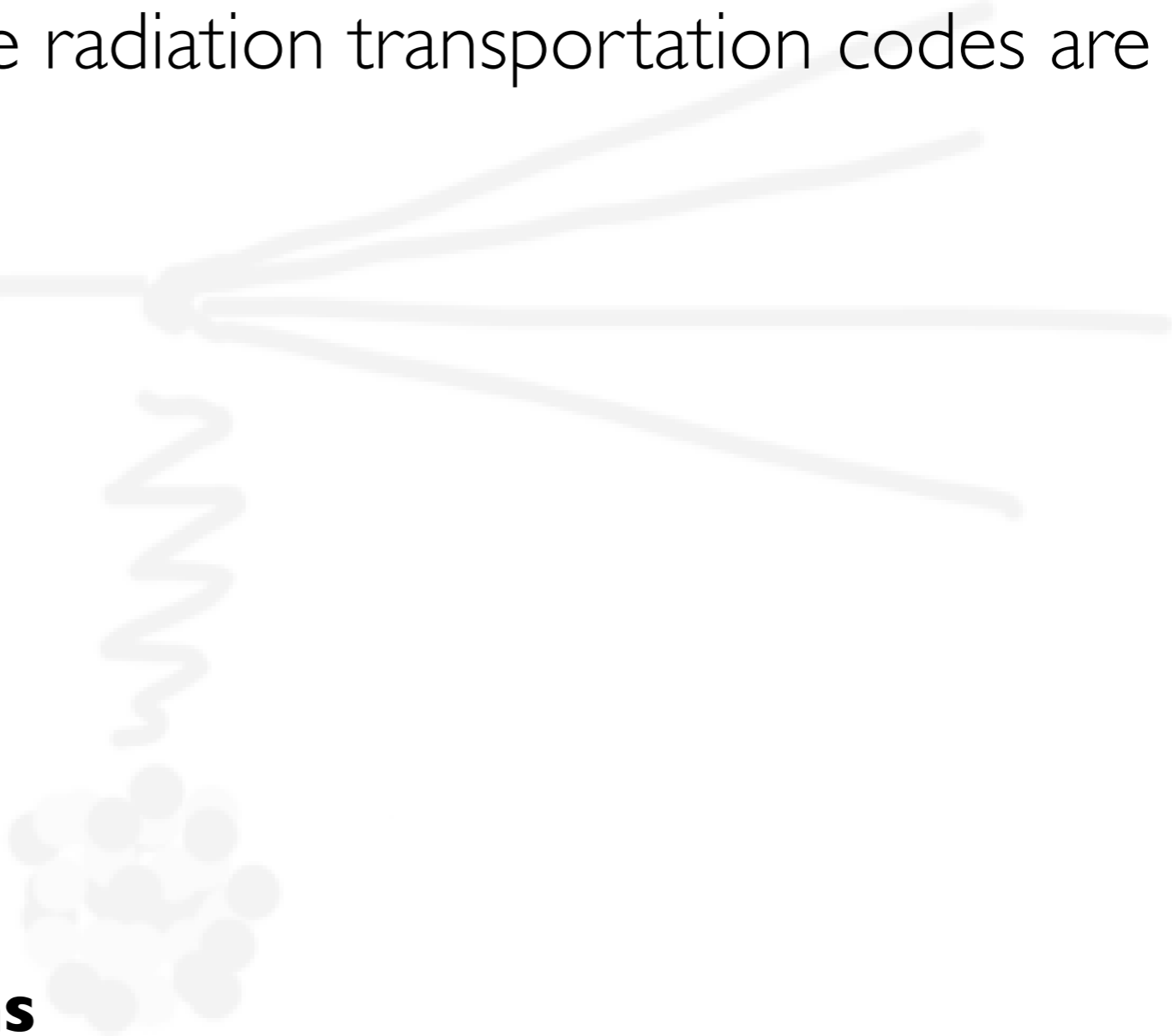
“Digitization”

- 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 world
 - The same reconstruction chain can be applied for both real and simulated data
- The general radiation transportation codes provide energy deposits in the whole detector; from these, the “digitization” simulates the electrical signals induced in the sensitive parts of the detector
 - It is an optional step (i.e. we will not do any “digitization” in Geant4 exercises)

Accuracy vs. Speed

- Huge samples (billions) of simulated events are often needed
- The number of simulated events is **limited by CPU**
- 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.

Application domains

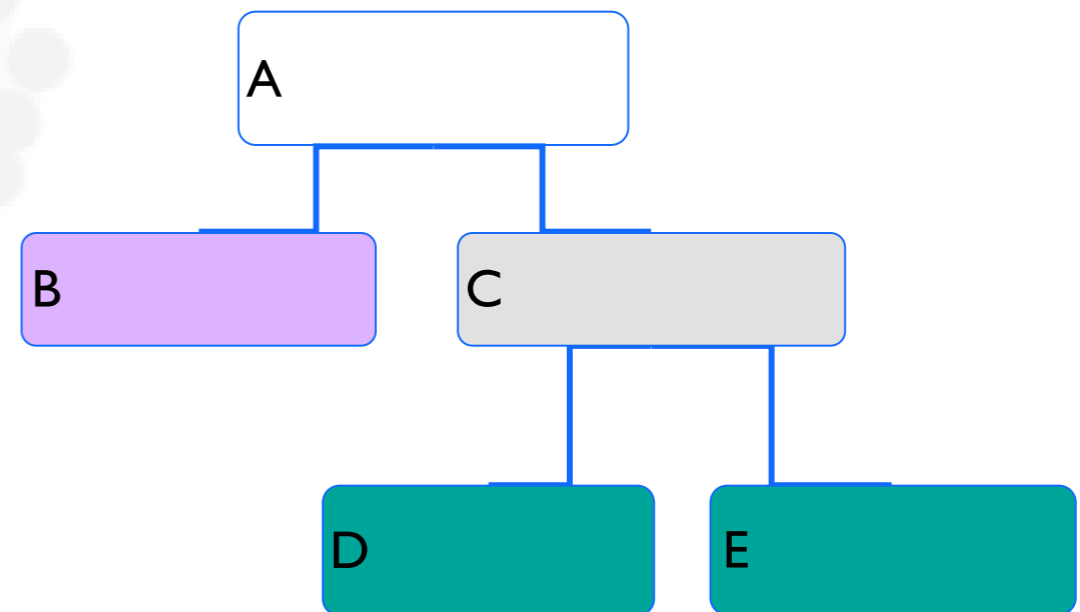
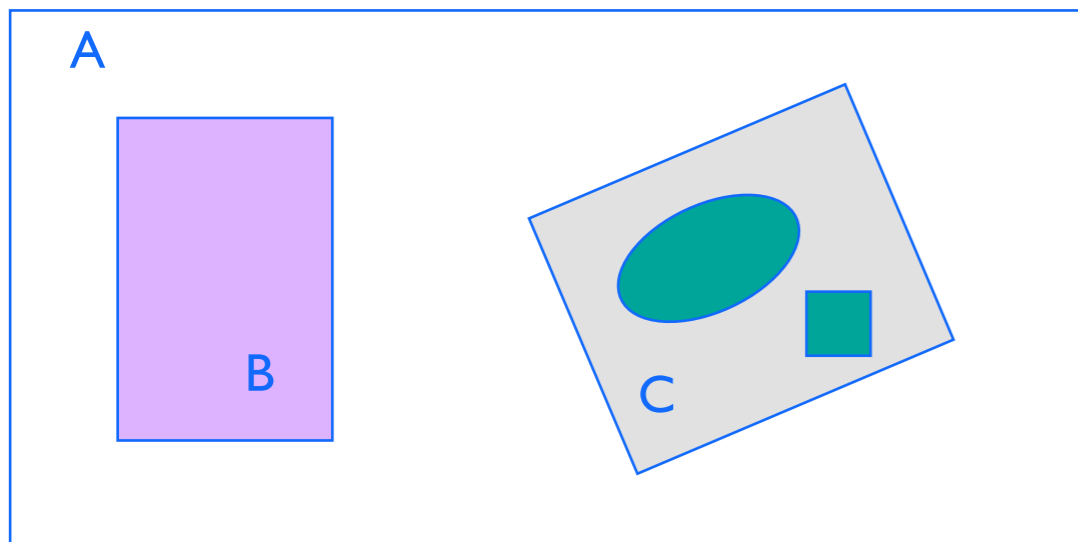
- Domains where the same radiation transportation codes are successfully used:
 - **High Energy Physics**
 - **Nuclear physics**
 - **Accelerator science**
 - **Astrophysics**
 - **Space engineering**
 - **Radiation damage**
 - **Medical physics**
 - **Industrial applications**
 - So, detector simulation is a multi-disciplinary field!
- 

Geometry

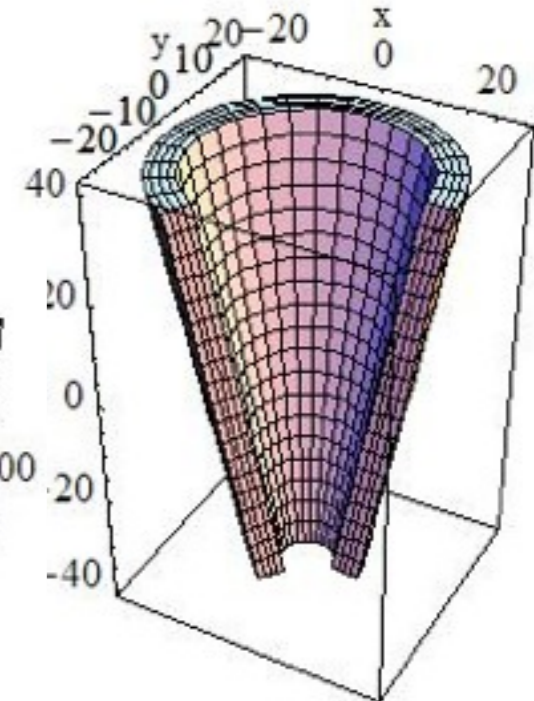
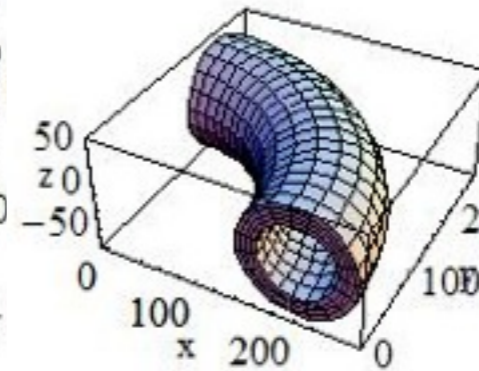
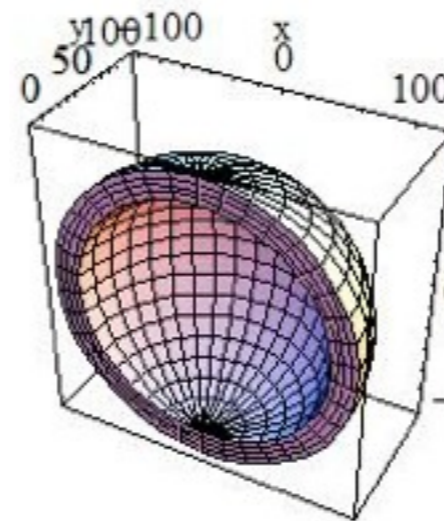
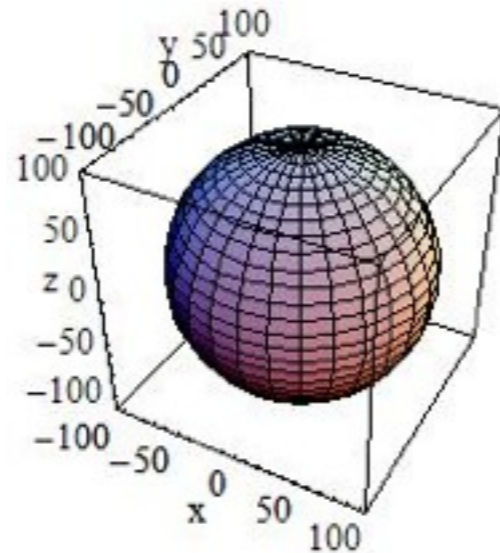
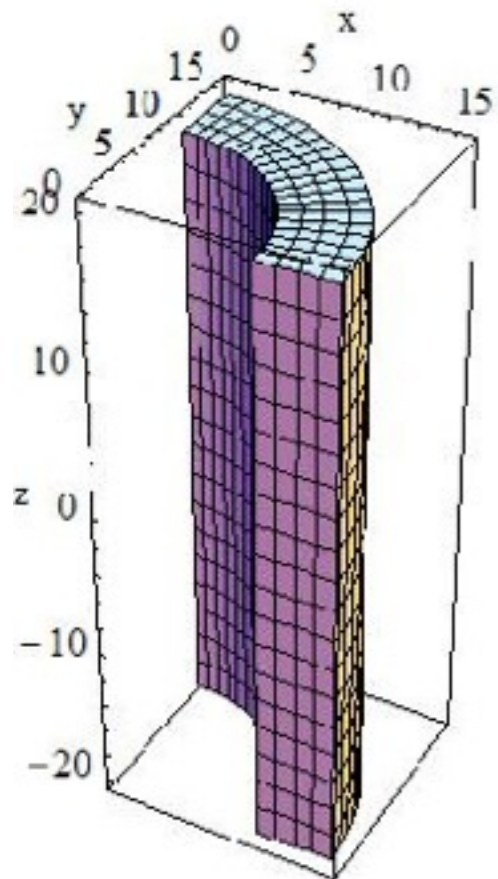
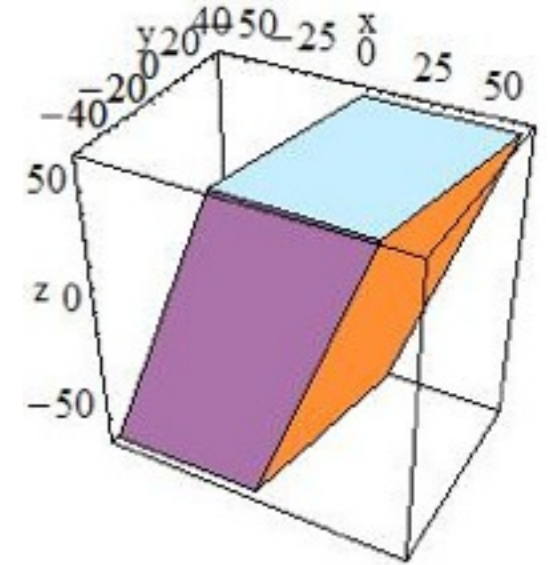
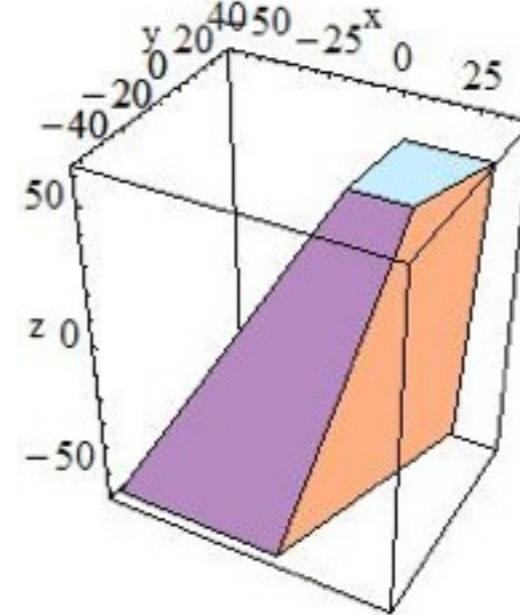
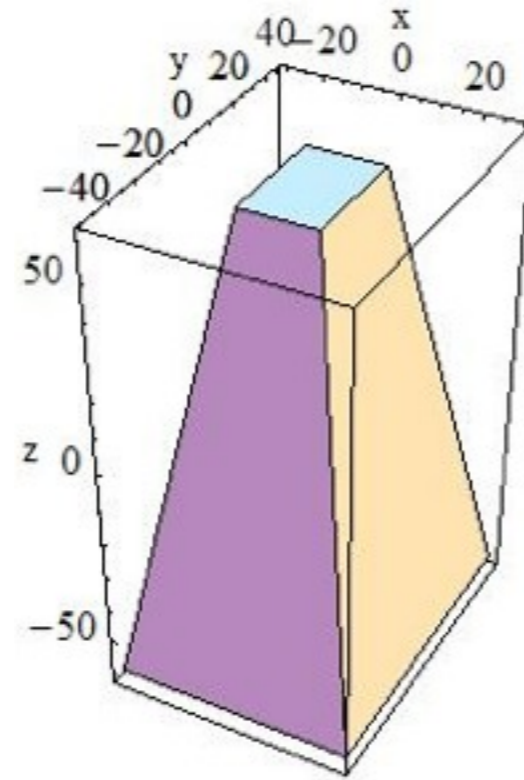
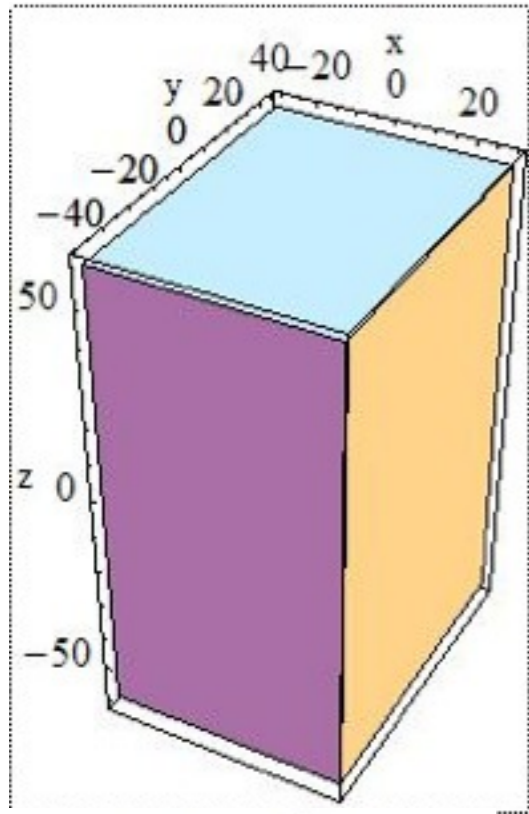


Geometry

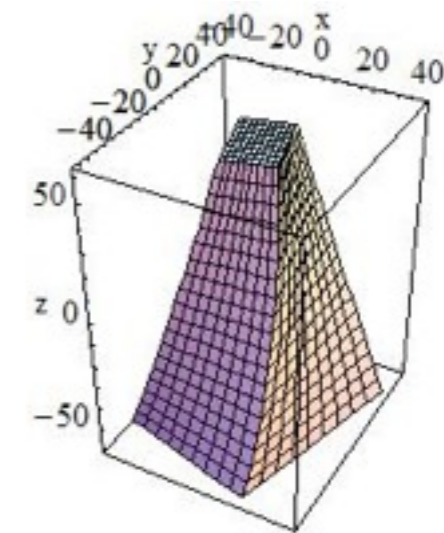
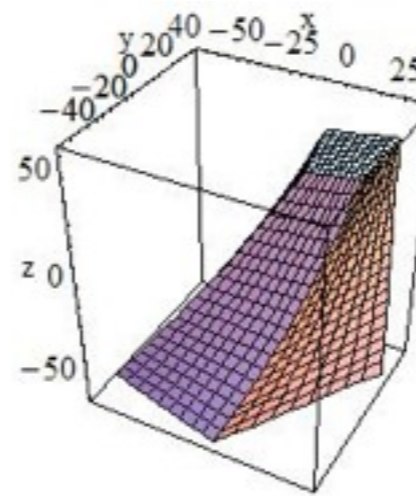
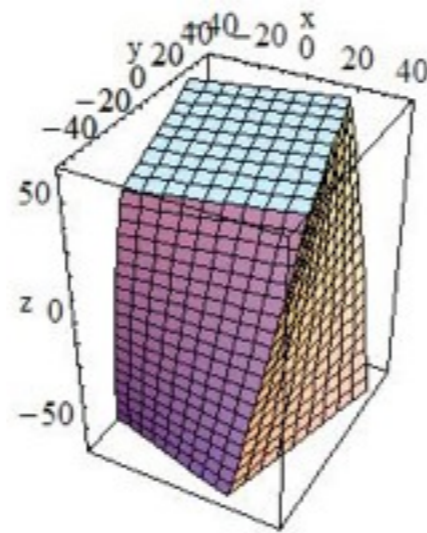
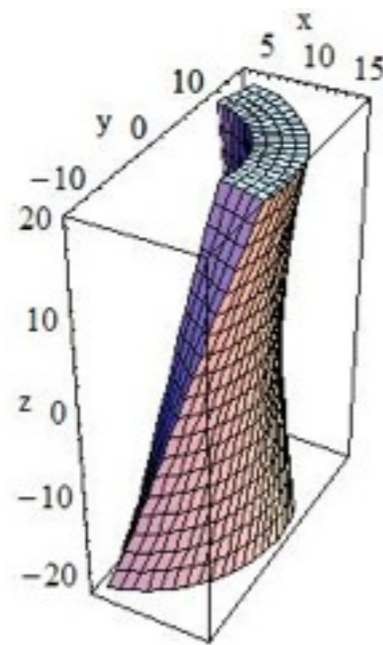
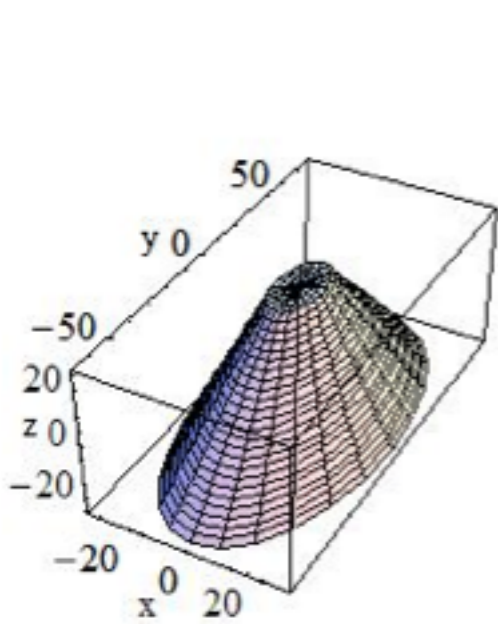
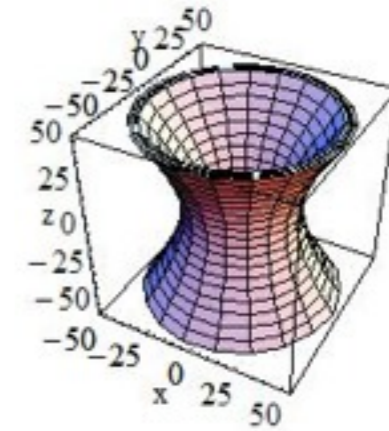
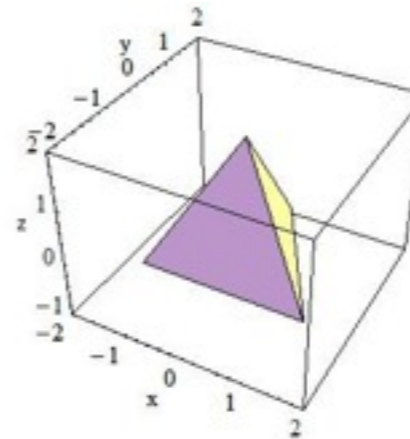
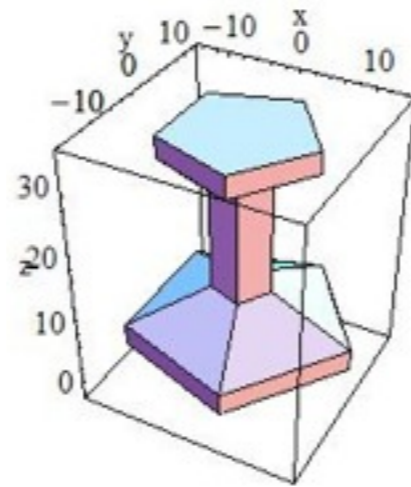
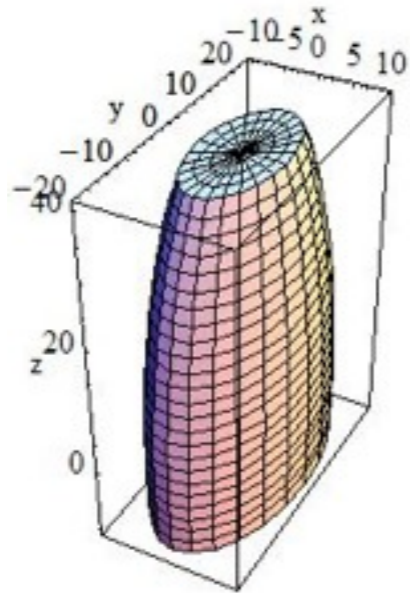
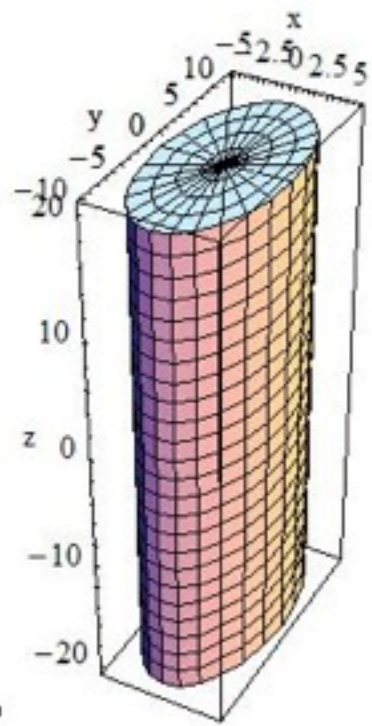
- The way to describe the geometry varies widely between the different simulation engines
 - In Geant4, you need to write some C++ code
 - Geometry objects are instances of classes
 - Geometry parameters (e.g. dimensions) are arguments of the constructors
- The geometry can be “flat” or “hierarchical”
 - In Geant4, it is hierarchical: a volume is placed in its mother volume; there are mother-daughter relationships
- A **material** should be assigned to each volume



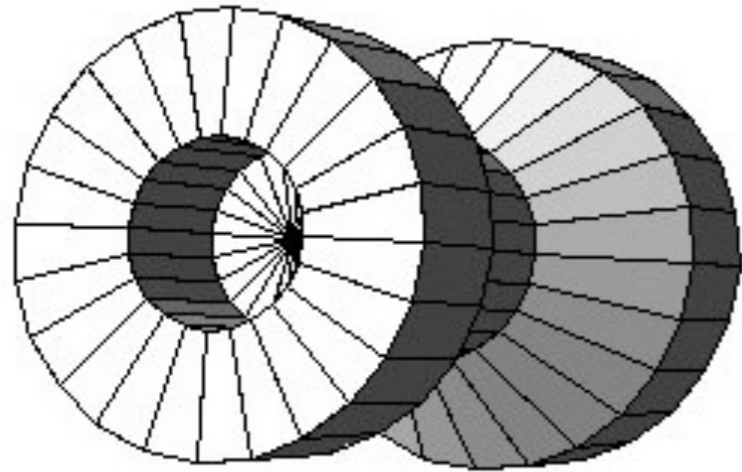
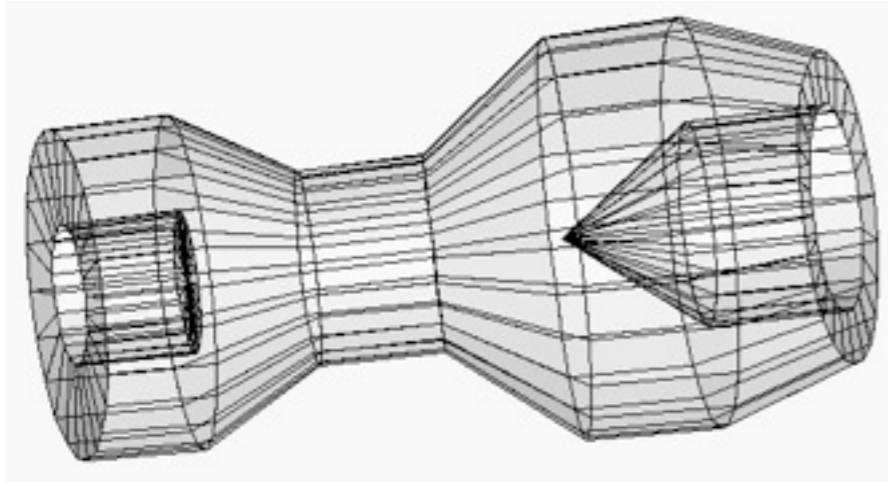
CGS (Constructed Geometry) Solids



Other CGS solids

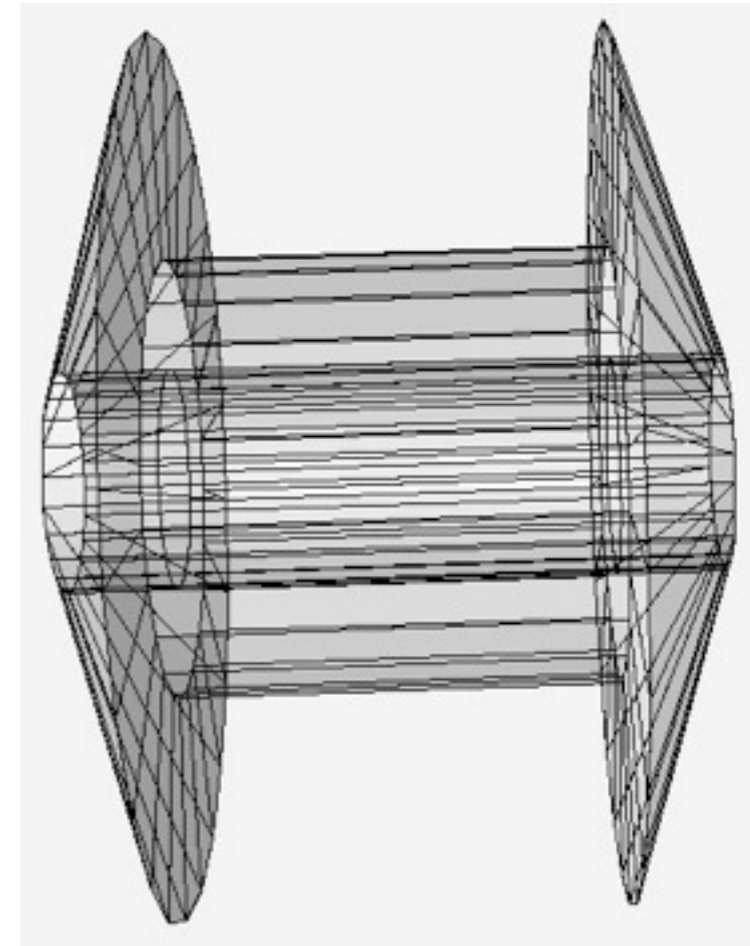
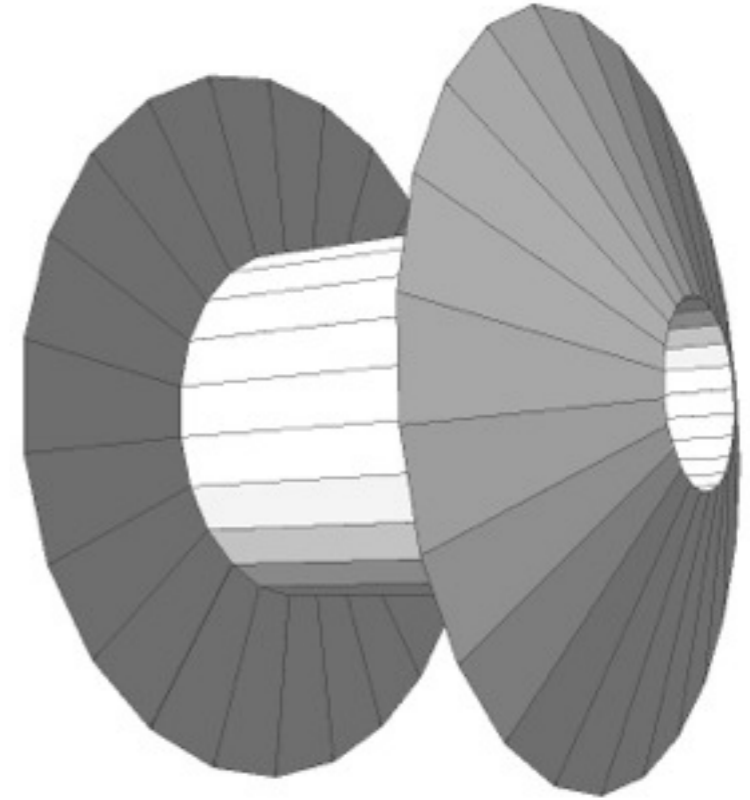


Polycone



BREP

(Boundary REPresented Solid)



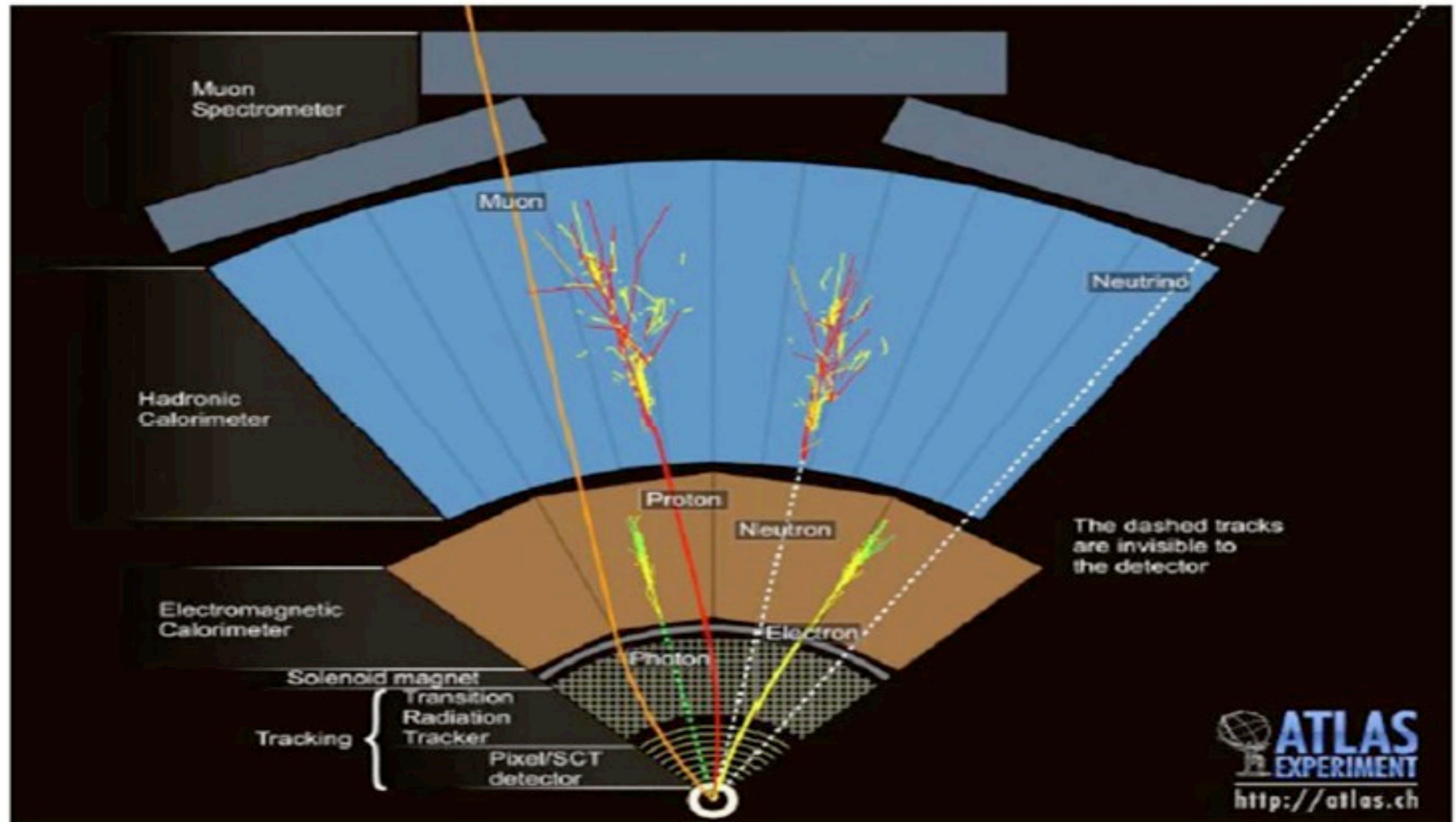
Physics



Particle interactions

Each particle type has its own set of physics processes

Only **electromagnetic** effects are directly measurable



Main electromagnetic processes

Gamma

- Conversion :
 $\gamma \rightarrow e^+ e^- , \mu^+ \mu^-$
- Compton scattering :
 $\gamma \text{ (atomic)} e^- \rightarrow \gamma \text{ (free)} e^-$
- Photo-electric
 $\gamma \text{ material} \rightarrow \text{(free)} e^-$
- Rayleigh scattering
 $\gamma \text{ atom} \rightarrow \gamma \text{ atom}$

Muon

- Pair production
 $\mu^- \text{ atom} \rightarrow \mu^- e^+ e^-$
- Bremsstrahlung
 $\mu^- \text{ (atom)} \rightarrow \mu^- \gamma$
- MSC (Coulomb scattering) :
 $\mu^- \text{ atom} \rightarrow \mu^- \text{ atom}$
- Ionization :
 $\mu^- \text{ atom} \rightarrow \mu^- \text{ ion} + e^-$

Total cross section:
⇒ step length

Differential & partial
cross sections :
⇒ final state
(multiplicity & spectra)

Electron, Positron

- Bremsstrahlung
 $e^- \text{ (atom)} \rightarrow e^- \gamma$
- MSC (Coulomb scattering):
 $e^- \text{ atom} \rightarrow e^- \text{ atom}$
- Ionization :
 $e^- \text{ atom} \rightarrow e^- \text{ ion} + e^-$
- Positron annihilation
 $e^+ e^- \rightarrow \gamma \gamma$

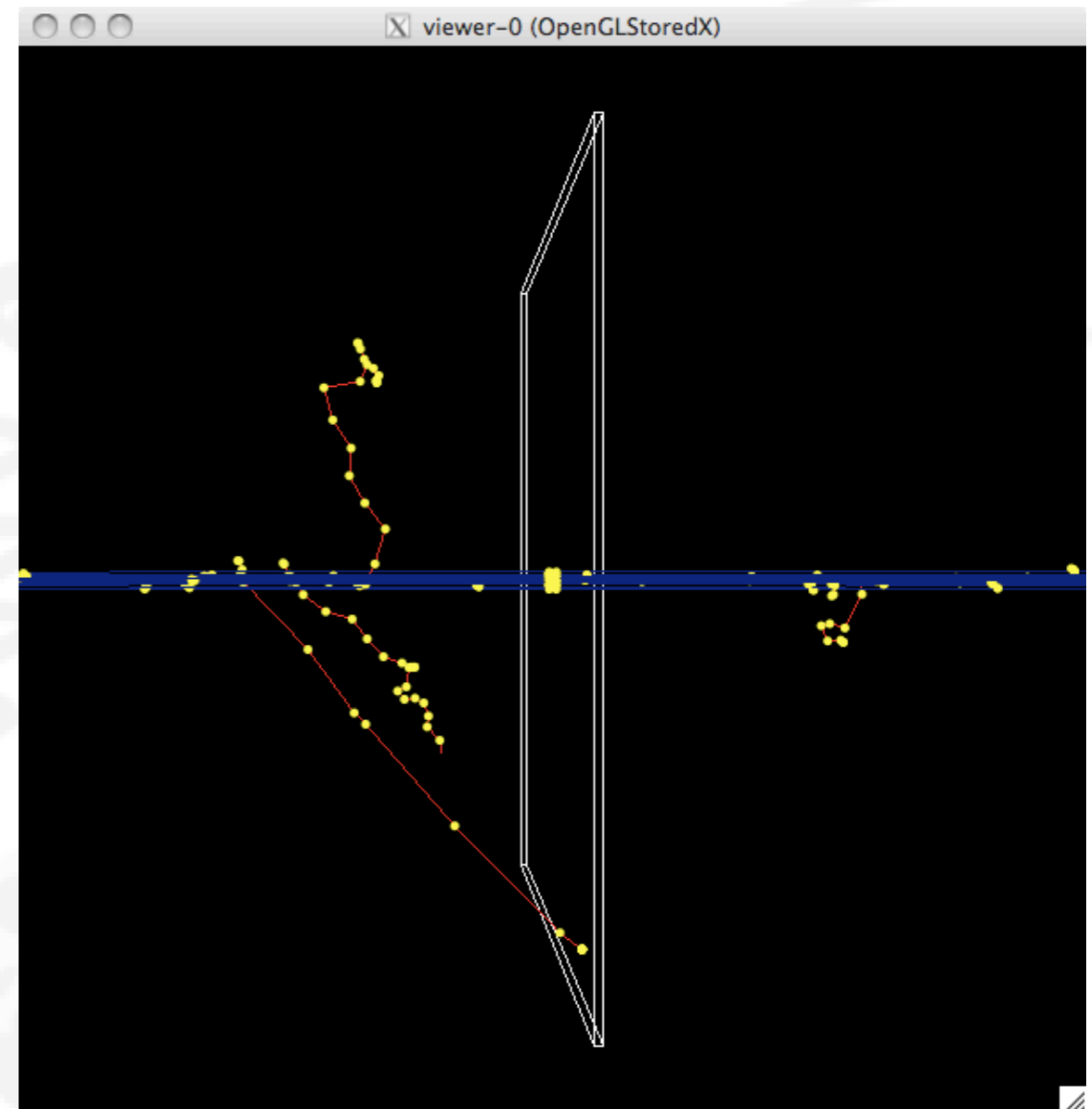
Charged hadron, ion

- (Bremsstrahlung
 $h^- \text{ (atom)} \rightarrow h^- \gamma$)
- MSC (Coulomb scattering):
 $h^- \text{ atom} \rightarrow h^- \text{ atom}$
- Ionization :
 $h^- \text{ atom} \rightarrow h^- \text{ ion} + e^-$

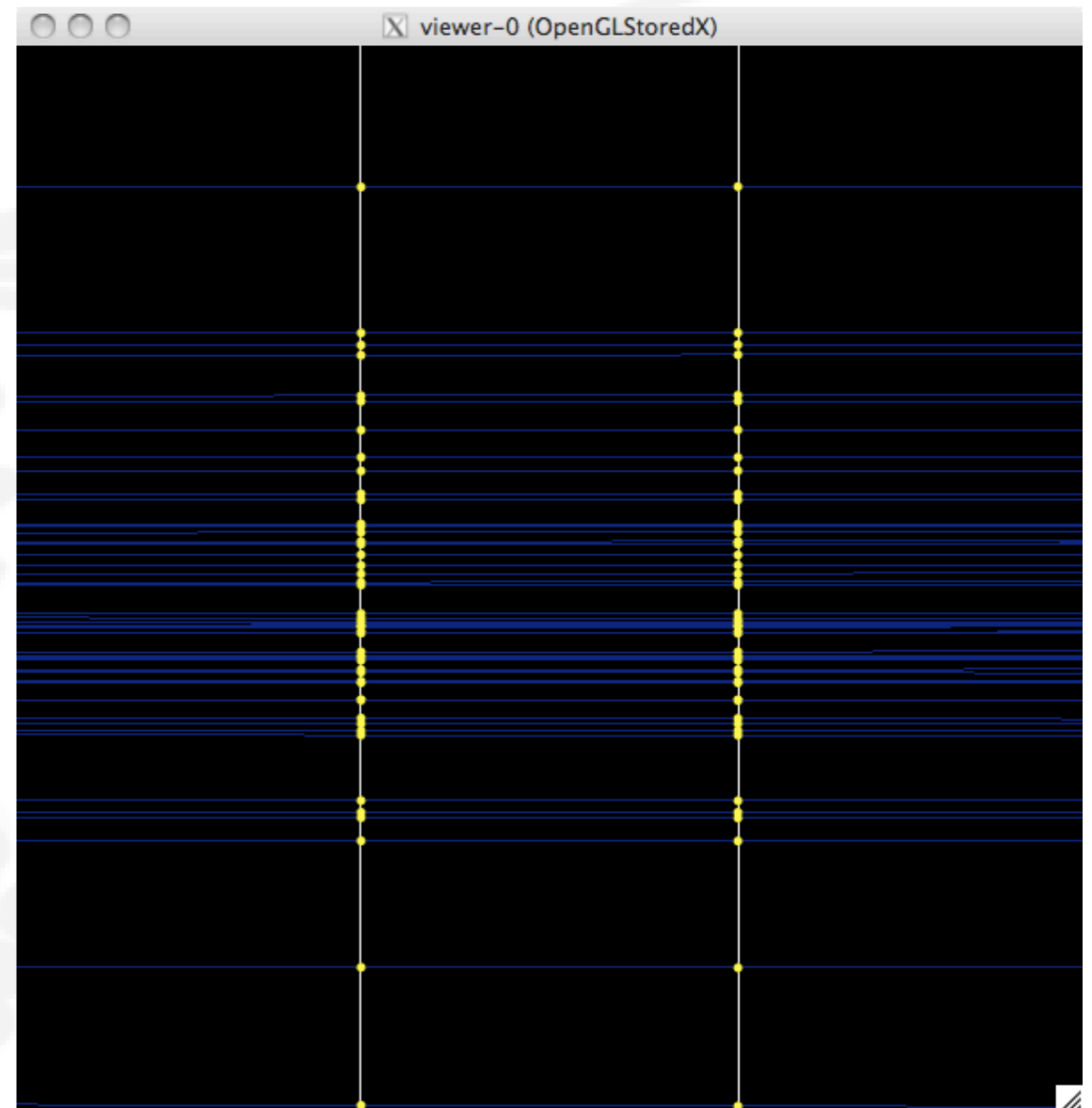
Production and tracking cuts

- Ionization and bremsstrahlung processes produce an increasing number of secondaries as the secondary energy decreases, so we need to set a **production cut**
 - Above the cut, new particles (e^- , γ) are created
 - Below the cut, **“continuous” energy loss** of the primary
- Once a charged particle is created, it can be reliably transported down to **$E_{kin} \sim 1 \text{ keV}$**
 - Either, stop it below a **tracking cut** and deposit its energy locally
 - Or, go down to $E_{kin} \rightarrow 0$ using its approximated range
- Production and tracking cuts can be expressed directly as **kinetic energy thresholds** or indirectly as equivalent **range thresholds**

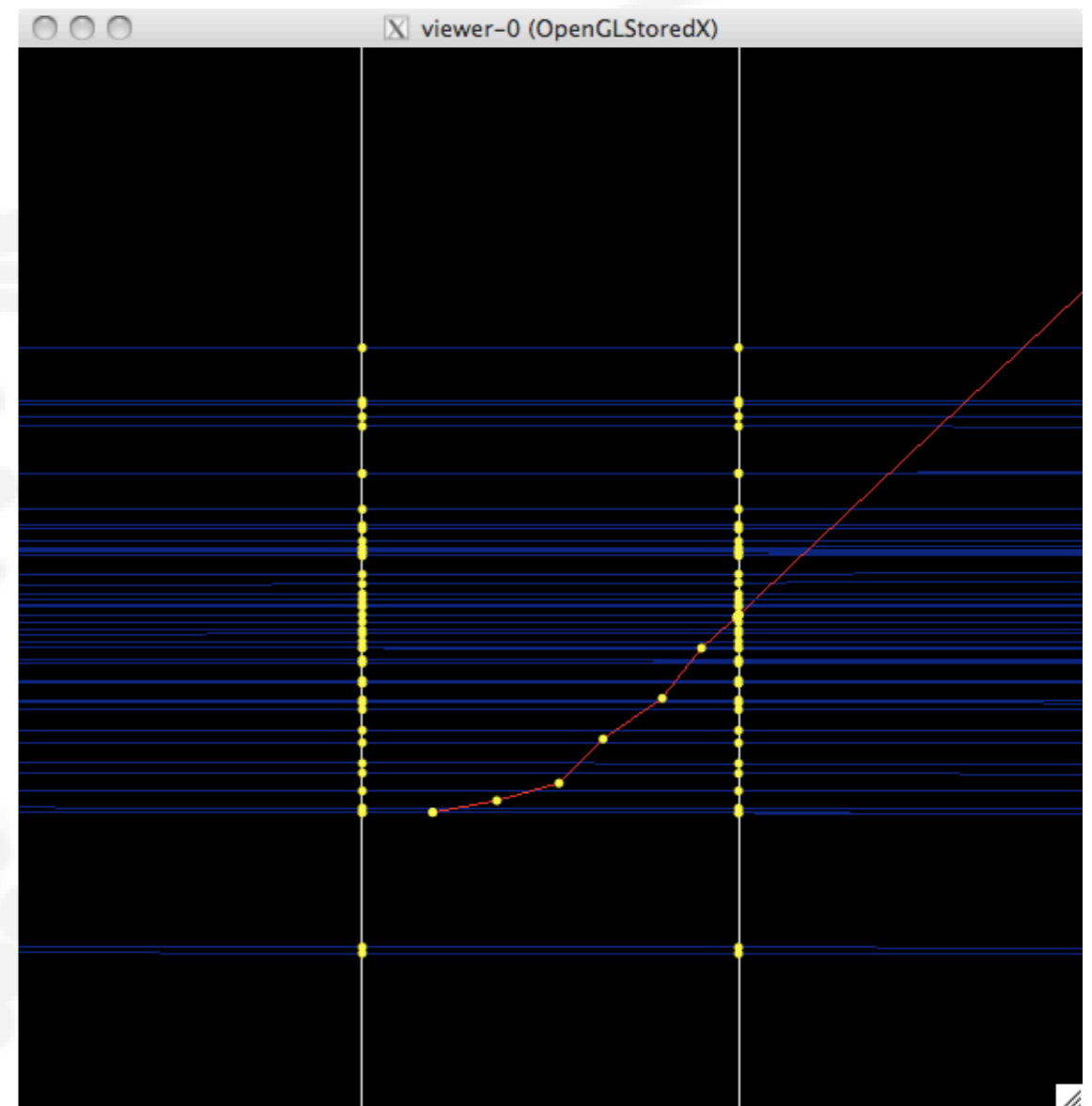
- [150 MeV protons (blue)
- [100 μm Si
- [Default parameters
 - [1mm “cut”
 - [550 keV δ threshold in Si
- [δ (red) and γ (green)



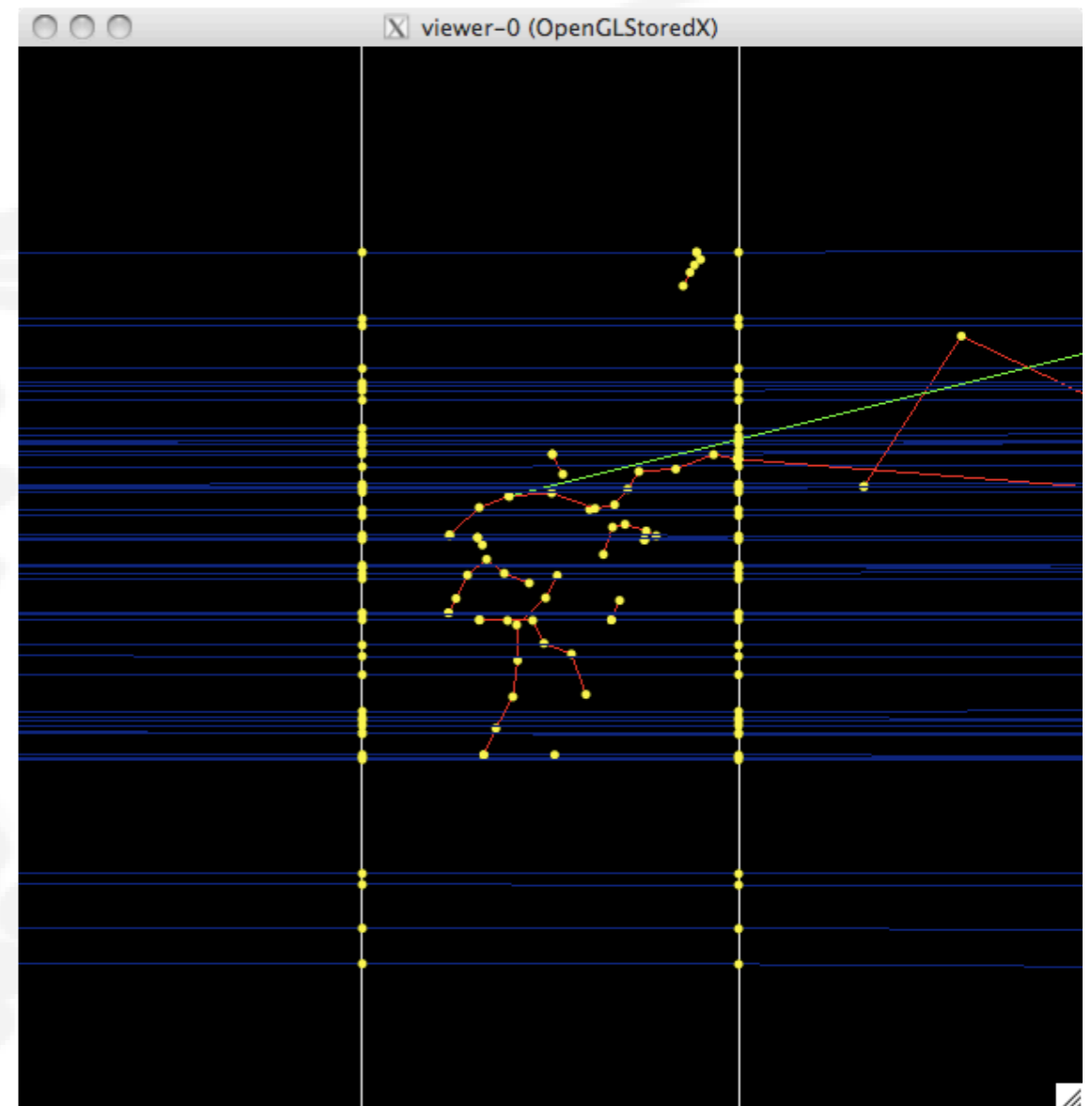
- [150 MeV protons (blue)
- [100 μm Si
- [Default parameters
 - [1mm “cut”
 - [550 keV δ threshold in Si
- [δ (red) and γ (green)



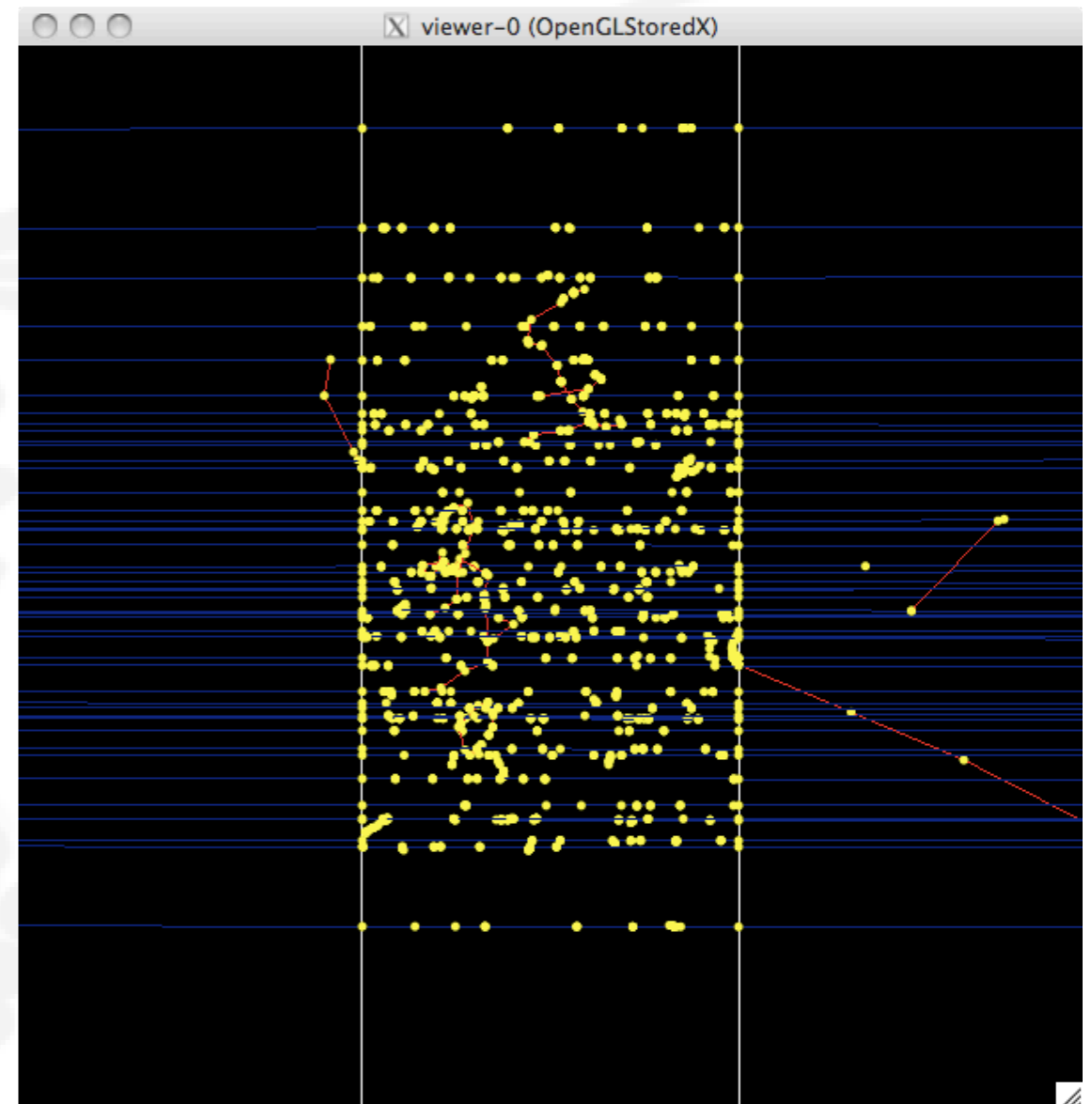
- [100 μm “cut”
- [120 keV δ threshold in silicon



- [10 μm “cut”
- [32 keV δ threshold in silicon

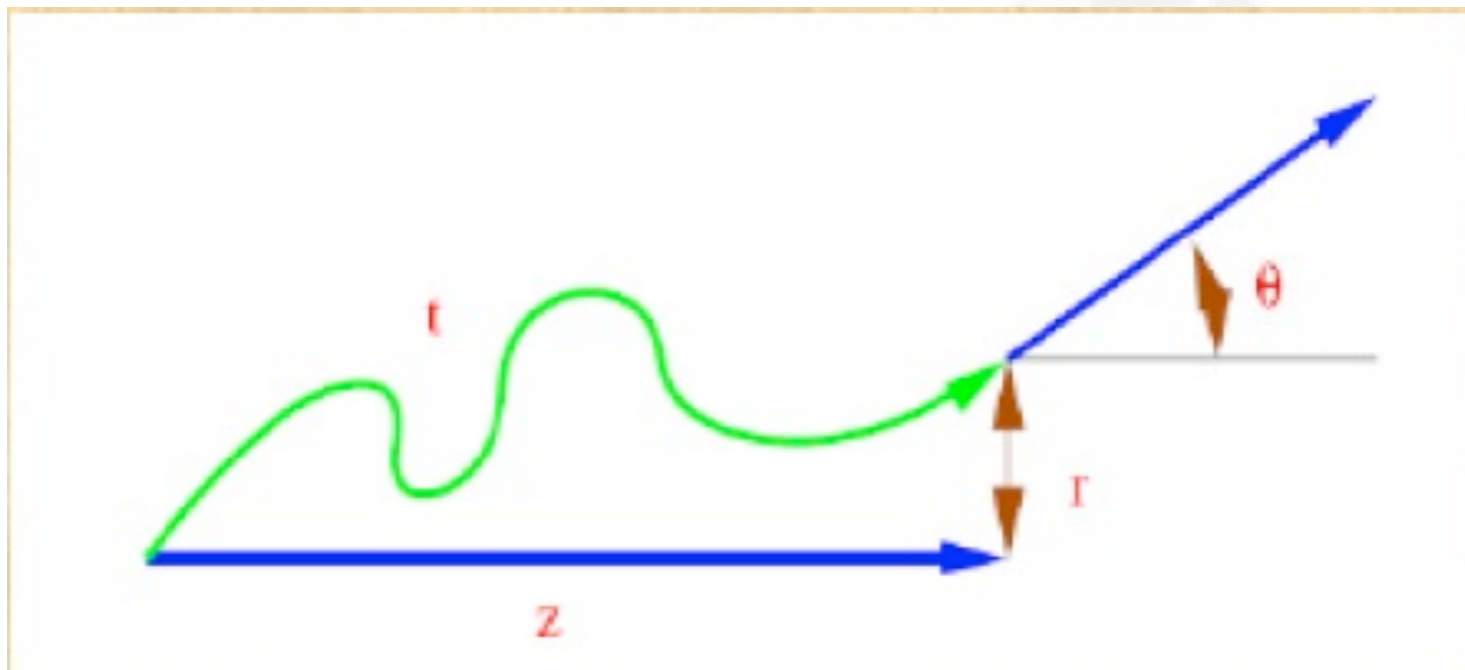


- [1 μm “cut”
- [1 keV δ threshold in silicon



Multiple (Coulomb) scattering (MSC)

- Charged particles traversing a finite thickness of matter suffer a huge number (millions) of elastic Coulomb scatterings
- The cumulative effect of these small angle scatterings is mainly a net deflection from the original particle direction
- In most cases, to save CPU time, these **multiple scatterings are not simulated individually, but in a “condensed” form**
- Various algorithms exist, and new ones under development. One of the main differences between codes



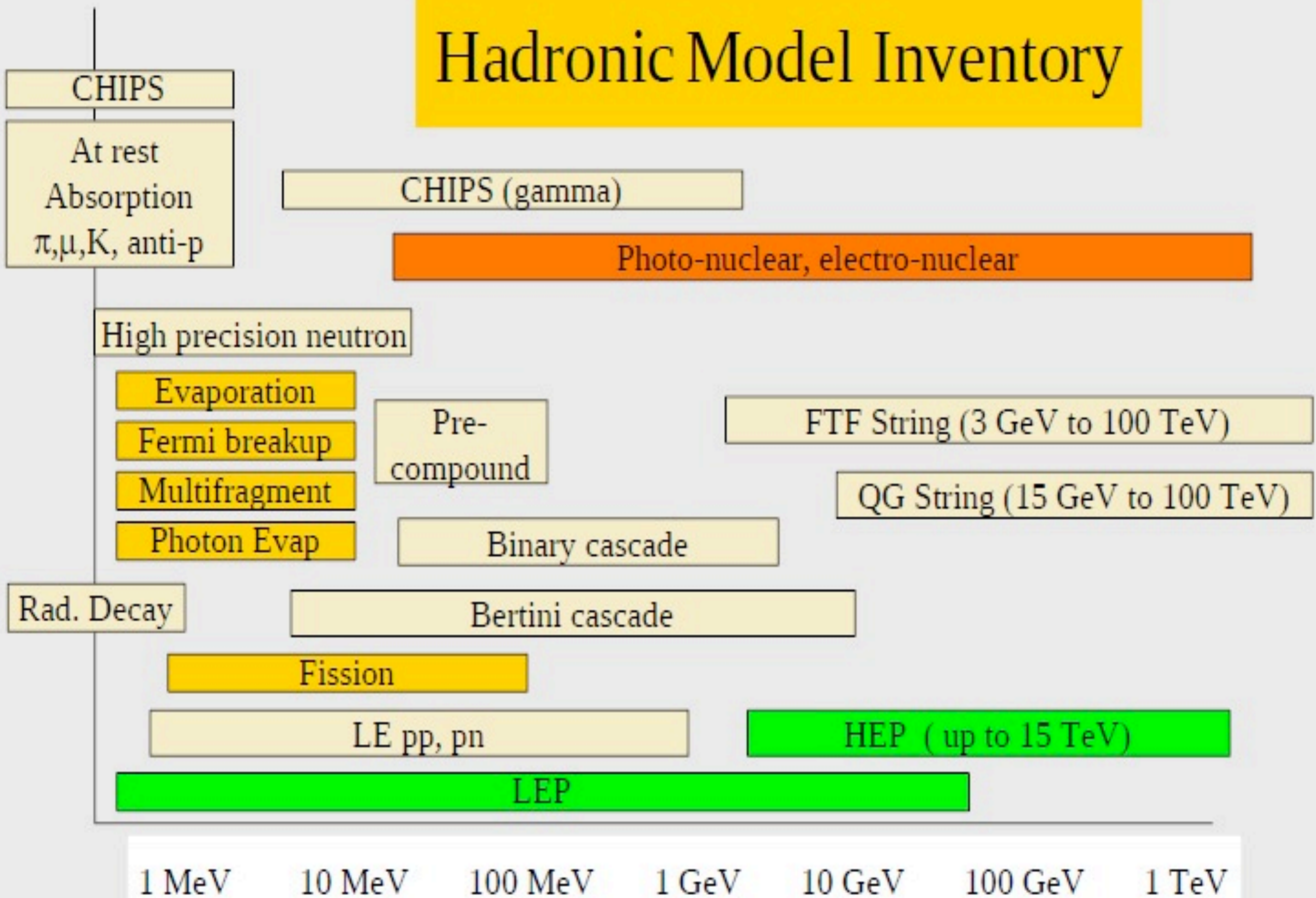
Electromagnetic physics

- Typical validity of electromagnetic physics $\geq 1 \text{ keV}$;
for a few processes, extensions to lower energies
- CPU performance of electromagnetic physics is critical : continuous effort to improve it
- Typical precision in electromagnetic physics is $\sim 1\%$
 - QED is extremely precise for elementary processes,
but atomic and medium effects, important for detector simulations, introduce larger uncertainties...
 - Moreover, the “condensed” description of multiple scattering introduces further approximations...
 - Continuous effort to improve the models

Hadronic interactions

- Hadrons (π^\pm , K^\pm , K^0_L , p , n , α , etc.), traverse the detectors (H,C,Ar,Si,Al,Fe,Cu,W,Pb...)
- Therefore we need to model **hadronic interactions**
 $h+A \rightarrow X$
- In principle, QCD is the theory that describes all hadronic interactions; in practice, perturbative calculations are available only for a tiny phase-space region
 - the hard scattering at high transverse momentum
- whereas for the rest, i.e. **most of the phase space**
 - soft primary scattering, re-scattering, and nucleus de-excitation
- **Only approximate models are available**
- Hadronic models are valid for limited combinations of
 - **particle type - energy - target material**

Hadronic Model Inventory

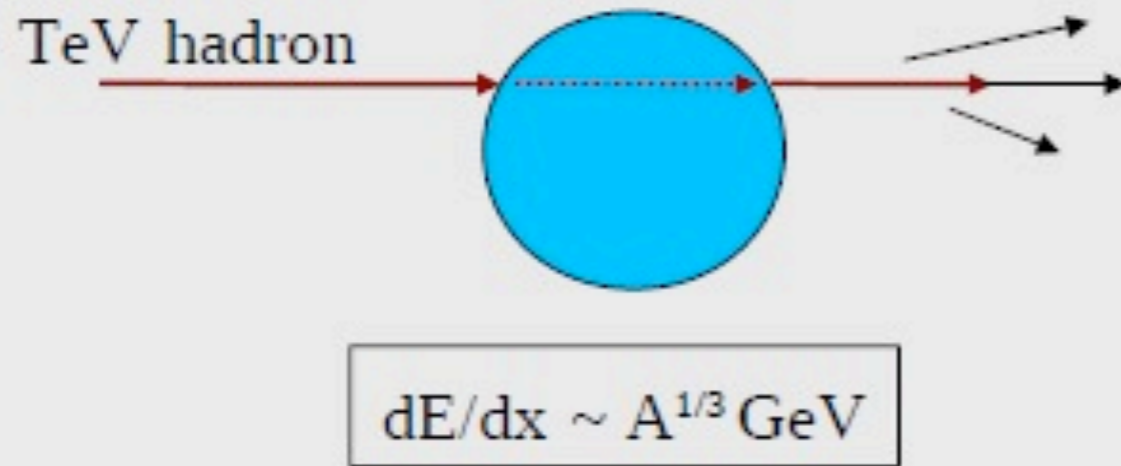


Physics configuration

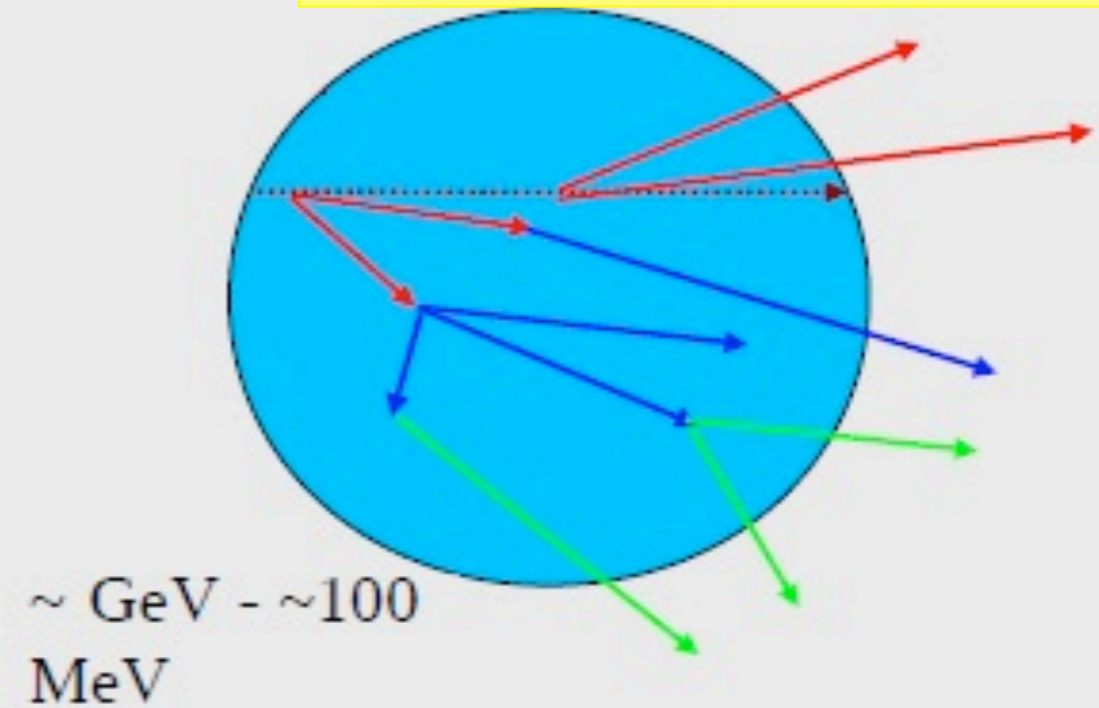
- No “unified” hadronic model: need to choose a set of hadronic models able to cover all possible interactions
 - The **choice depends on the use-case**, because of:
 - The energy scale involved
 - The compromise between accuracy and CPU speed
- In the case of Geant4
 - These physics configurations are called “physics lists”
 - The particles to be considered in the simulation are also specified
 - There is no default
 - **Ready-to-use “physics lists” exist, for different use-cases**
 - Users can also tailor/modify any of these, or write their own

Hadronic Interactions from TeV to meV

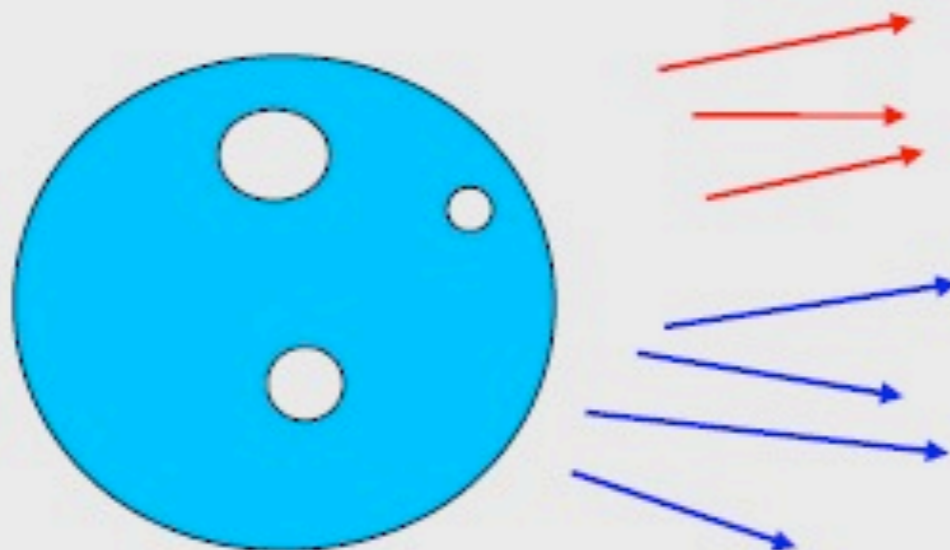
String model



Intra-nuclear cascade model

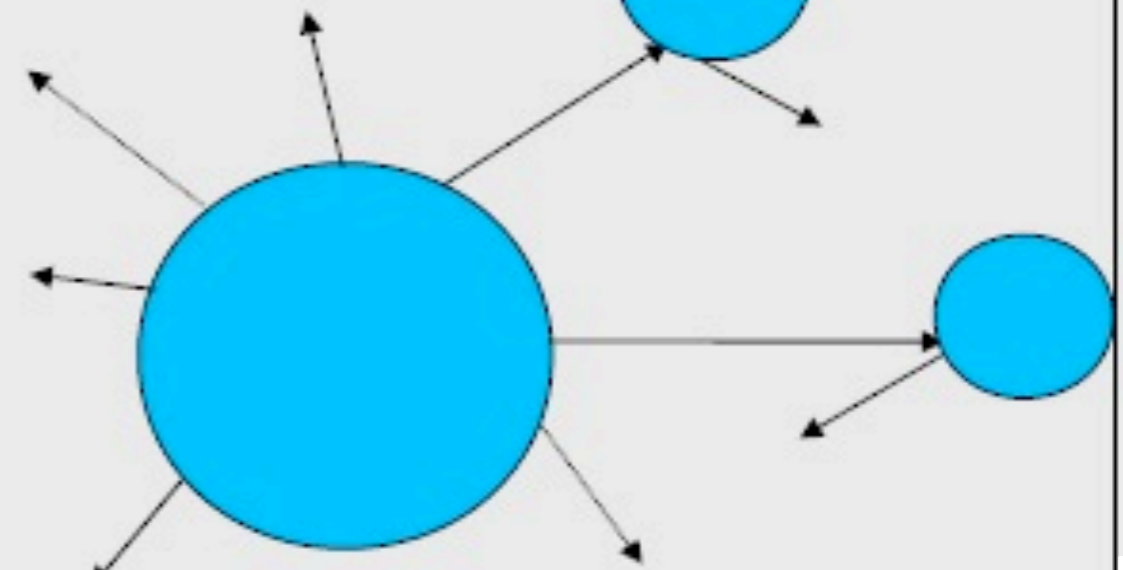


$\sim 100 \text{ MeV} - \sim 10 \text{ MeV}$



Pre-equilibrium (Precompound) model

$\sim 10 \text{ MeV}$ to thermal



Equilibrium (Evaporation) model

An interesting complication: Neutrons

- Neutrons are **abundantly produced**, mostly by hadron-nucleus interactions
- It is typically the 3rd most produced particle (after e-, γ)
- Before a neutron “disappears” via an inelastic interaction, it can have many **elastic scatterings** with nuclei, and eventually it can “thermalize” in the environment
- The CPU time of the detector simulation can vary by orders of magnitudes according to the physical accuracy of the **neutron transportation simulation**
 - For typical high-energy applications, a simple treatment is enough
 - For activation and radiation damage studies, a more precise, **data-driven and isotope-specific** treatment is needed, especially for neutrons of kinetic energy **below ~ MeV**

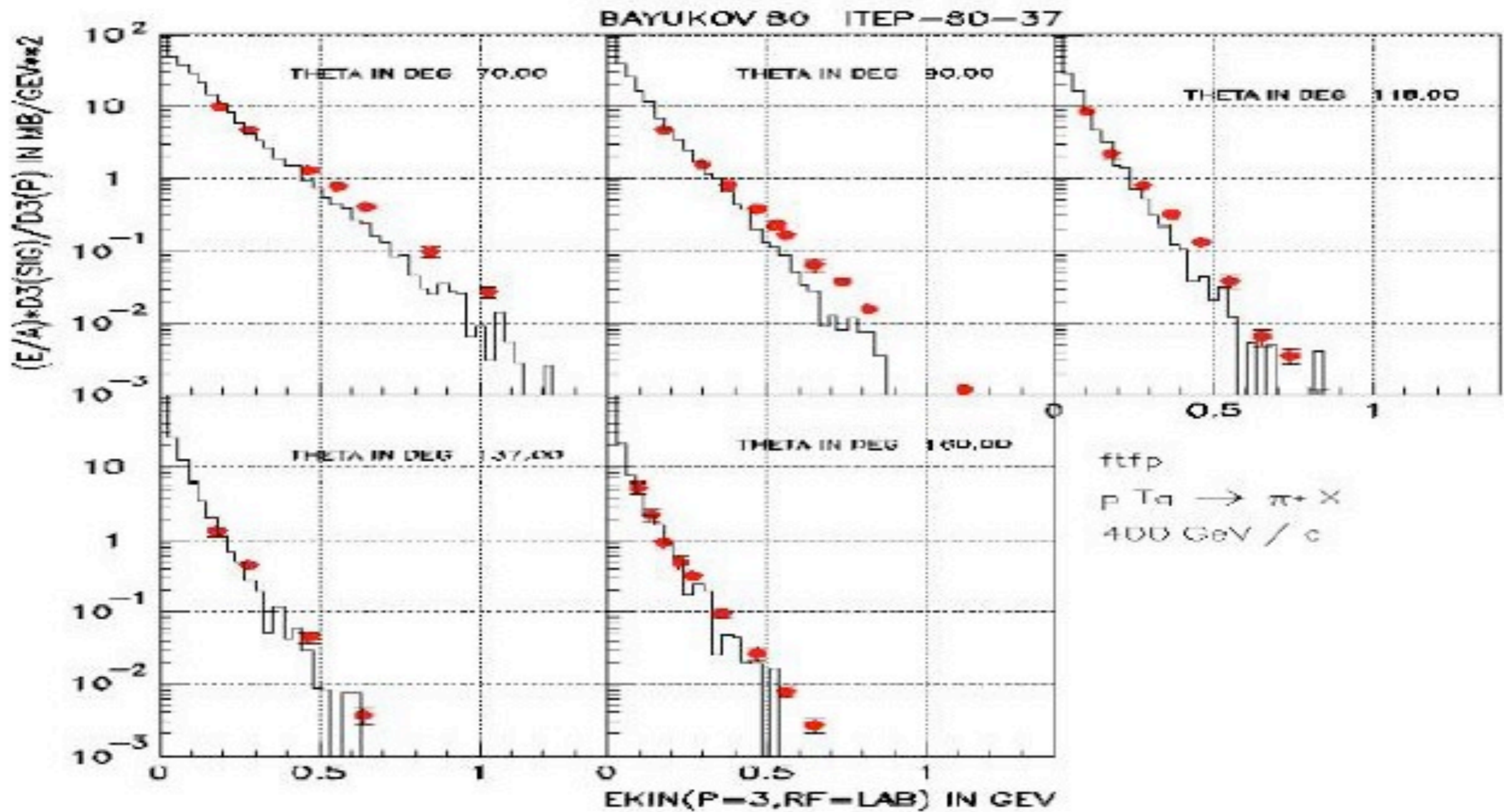
Validation

A decorative graphic on the right side of the slide. It features a horizontal line that branches into three diagonal lines extending upwards and to the right. Below this junction, a vertical wavy line descends to a cluster of approximately 15 small, light gray circles.

Note: the following slides refer to Geant4 simulation code

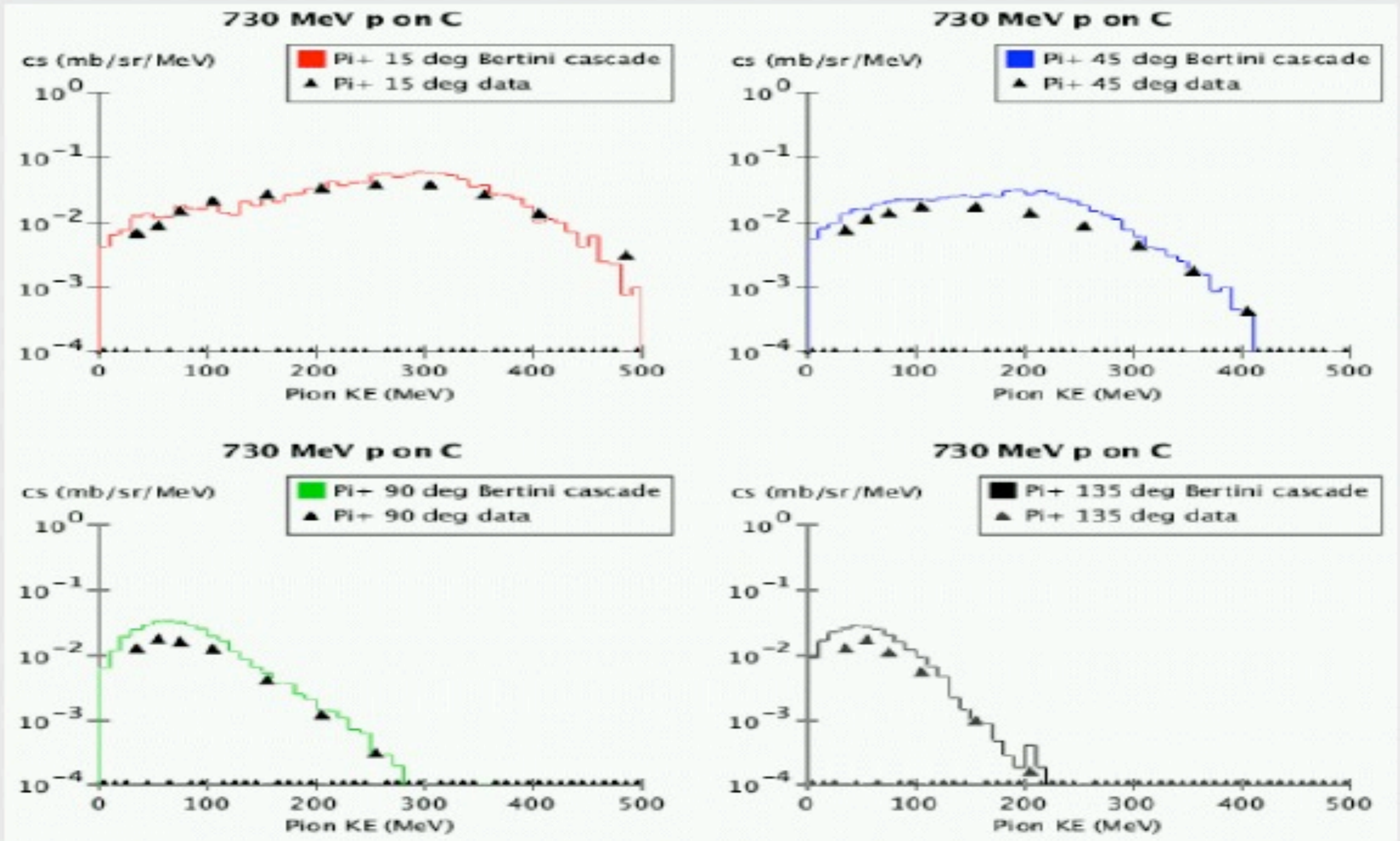
Model-level thin-target test

FTF Results at 400 GeV/c
 $p \text{ Ta} \rightarrow \pi^+ X$



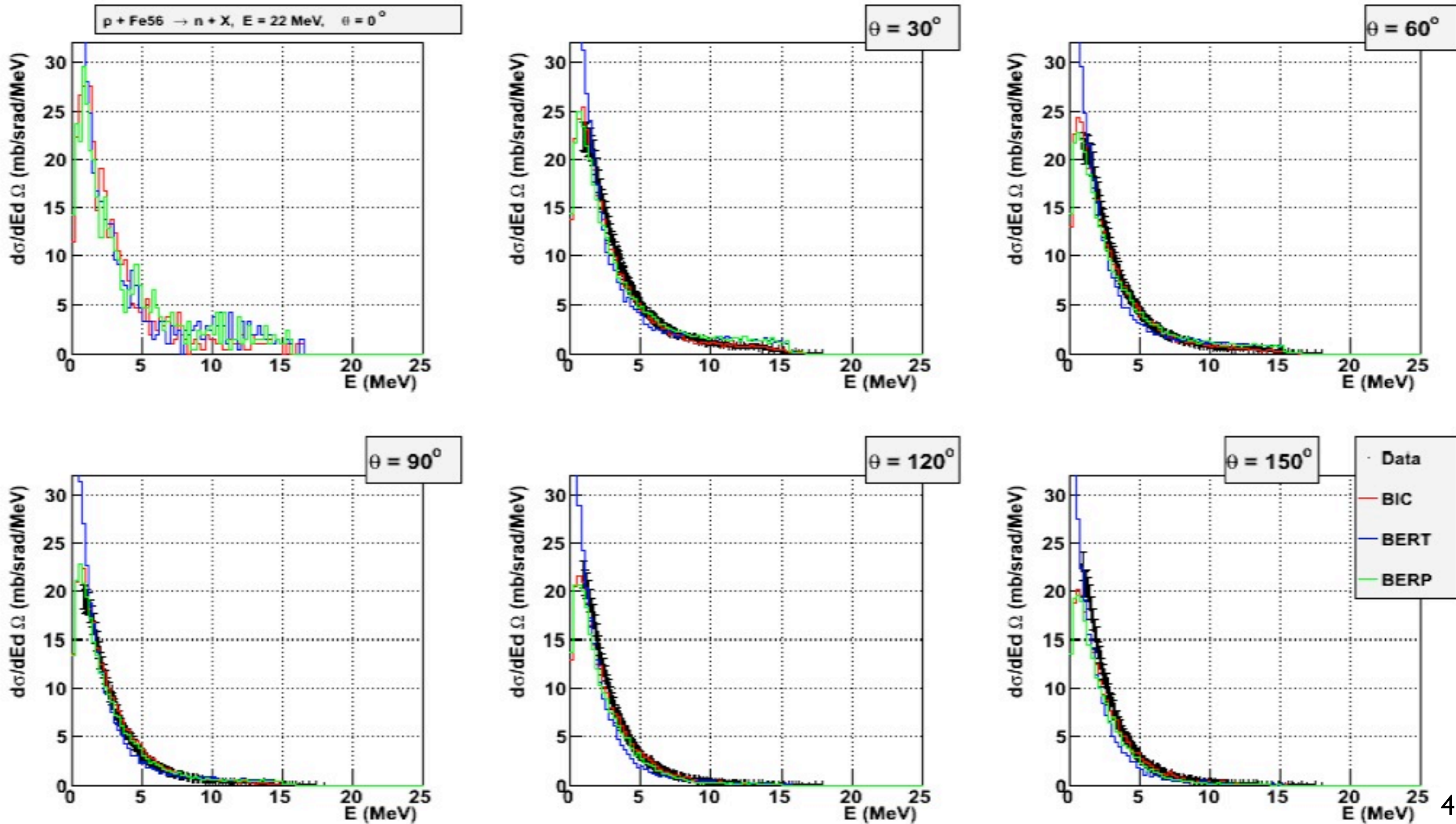
Model-level thin-target test

Validation of the Bertini Cascade



Model-level thin-target test

Preco validation, 22 MeV $p - Fe \rightarrow n$



Summary

- MC simulations are one of the main tools of modern physics (in HEP is fundamental)
- The main challenges of detector simulation are:
 - **Physics accuracy**
 - **CPU performance**
 - **Validation**
- Suggestions for you:
 - Learn by studying and playing with existing examples
 - Be **critical** and pragmatic when using simulations
 - **Contribute** to the validation and provide feedback

Other codes

- { General

- { Fluka

- { Geant3

- { MARS

- { MCNP / MCNPX

- { Dedicated to electromagnetic physics

- { ETRAN

- { EGS4

- { EGS5

- { EGSnrc

- { Penelope

Acknowledgment

- I would like to thank my Geant4 colleague A. Ribon (CERN/PH-SFT) for the material on the second part of this talk
- I would like to thank my Geant4 colleague J. Allison (G4AI) for the material on the tracking cut