Physics in Geant4

XXI Seminar on Software for Nuclear, Subnuclear and Applied Physics
9th June 2024

Carlo Mancini Terracciano
carlo.mancini-terracciano@uniroma1.it

Reminder

- Tonight h 21:00 talk by ICSC grant winners
- Melba D’Alfonzo
- Francesca Fede
- Maria F. H. Moyano

We expect all of you!

Simple case: decay in flight

- Suppose a π^+ with momentum *p*

- The life time is a random value with a pdf

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

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

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

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

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

Simple case: decay in flight

- Select the decay channel:

Mode			Fraction (Γ_i / Γ)
Γ_1	$\mu^+\nu_\mu$	[1]	$(99.98770 \pm 0.00004)\%$
Γ_2	$\mu^+\nu_\mu\gamma$	[2]	$(2.00 \pm 0.25) \times 10^{-4}$
Γ_3	$e^+\nu_e$	[1]	$(1.230 \pm 0.004) \times 10^{-4}$
Γ_4	$e^+\nu_e\gamma$	[2]	$(7.39 \pm 0.05) \times 10^{-7}$
Γ_5	$e^+\nu_e\pi^0$		$(1.036 \pm 0.006) \times 10^{-8}$
Γ_6	$e^+\nu_e e^+e^-$		$(3.2 \pm 0.5) \times 10^{-9}$
Γ_7	$e^+\nu_e\nu\overline{\nu}$		$< 5 \times 10^{-6}$

[table from PDG]

- In the CM frame the decay is isotropic

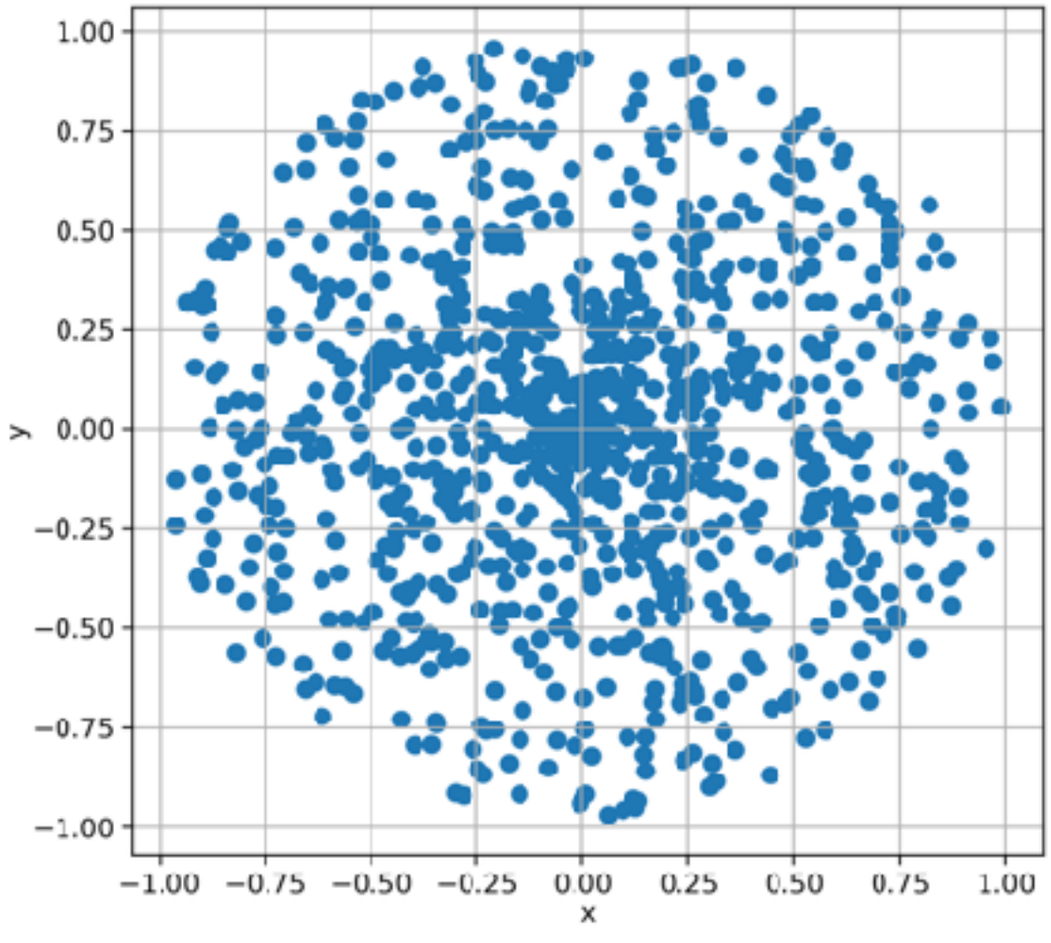
$$\cos(\theta) \in [-1,1); \quad \phi \in [0,2\pi)$$

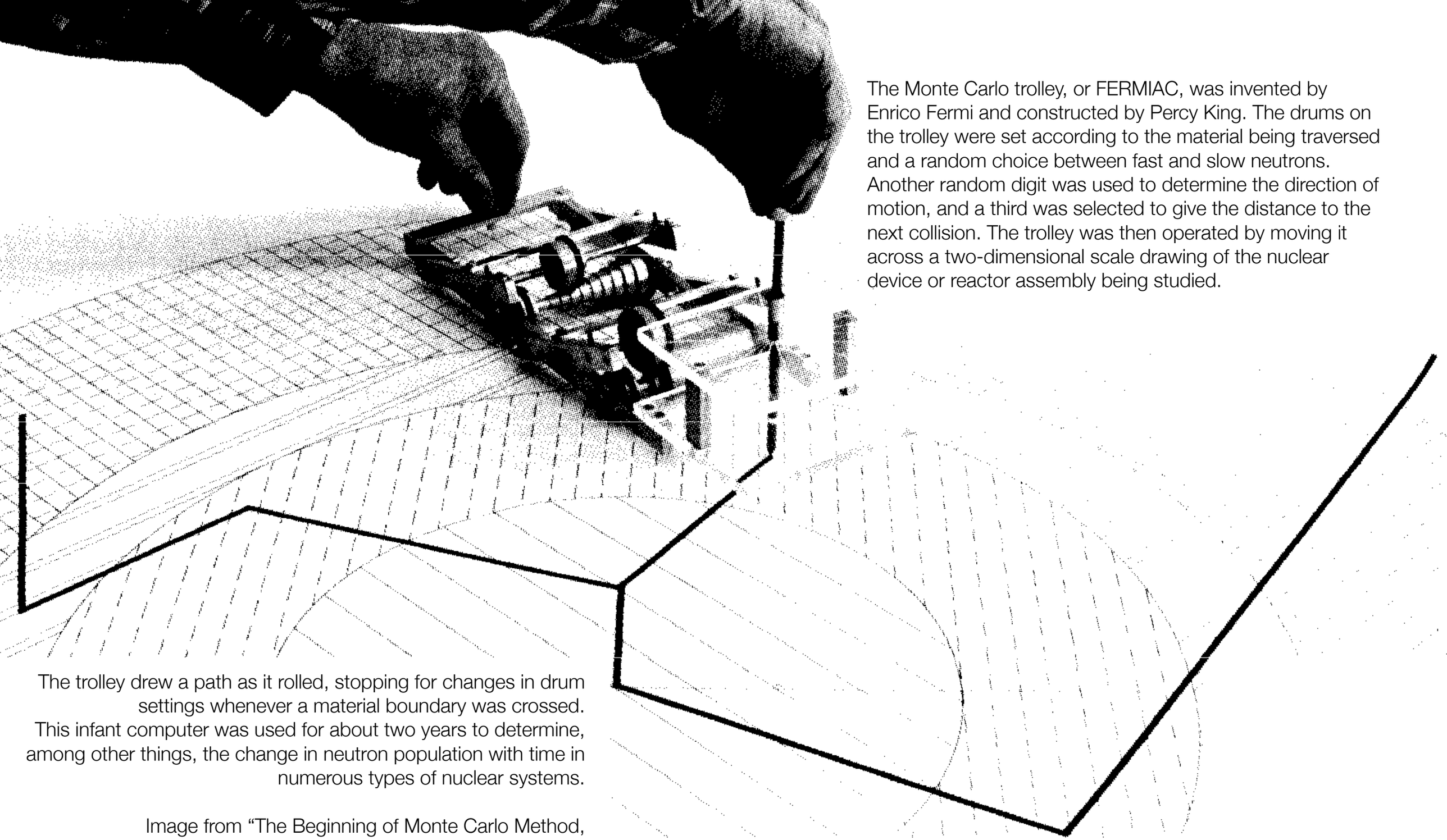
- Finally, Lorentz-boost in the Lab. frame

- 4 random numbers for one decay!

Problem

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





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

Tracking of particles

The “FERMIAC”

Particle tracking

- It is the most common application of MC in Particle Physic

- 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 μ a property of the medium

Particle tracking

- μ is proportional to the probability of an interaction per unit length, therefore:

  - is proportional to the **total cross section**

$$\mu = N\sigma = N\sum_i \sigma_i = \sum_i \mu_i$$

- μ_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)

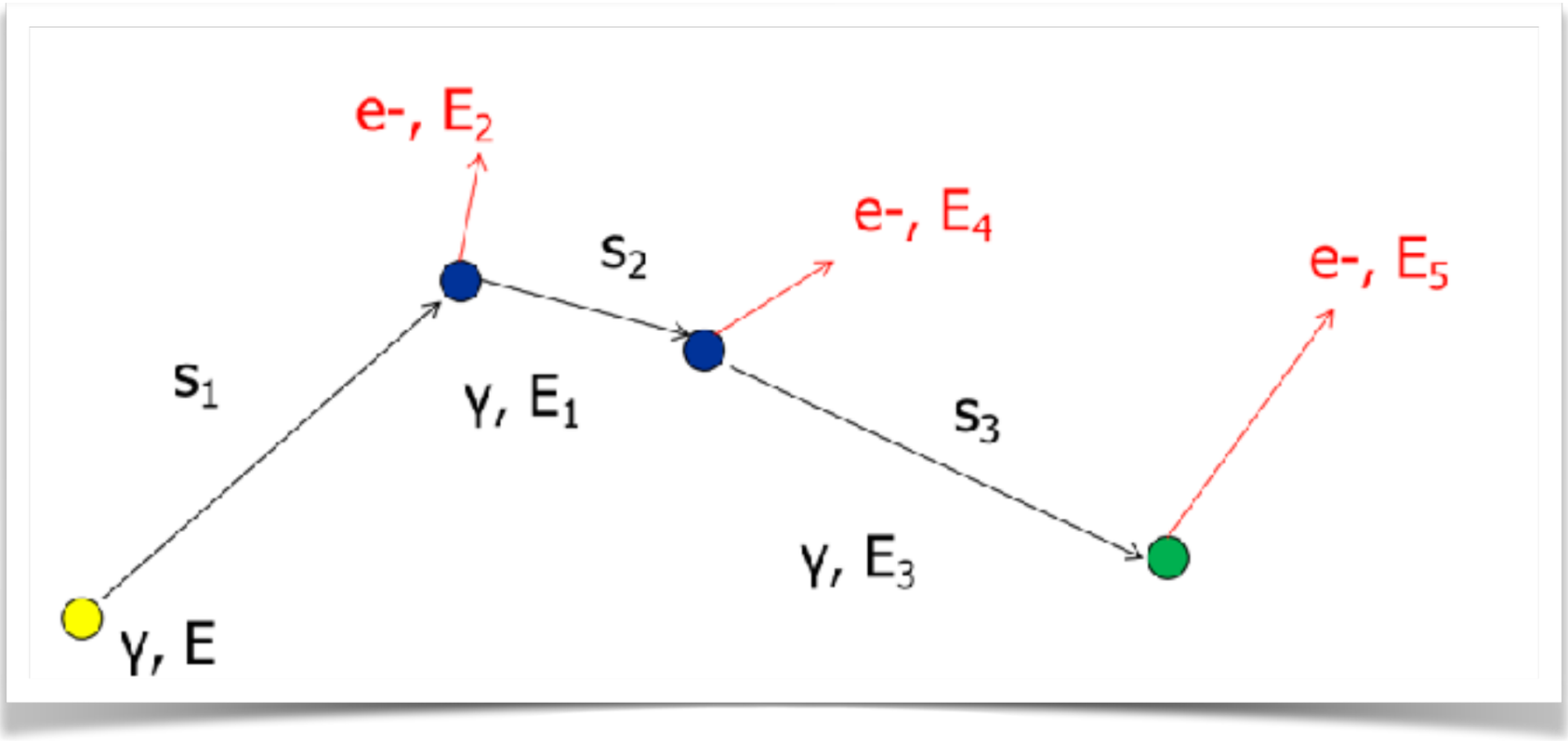
Particle tracking

- 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
  - accordingly to μ_i / μ
- 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)

- μ depends on the energy (cross sections do!)



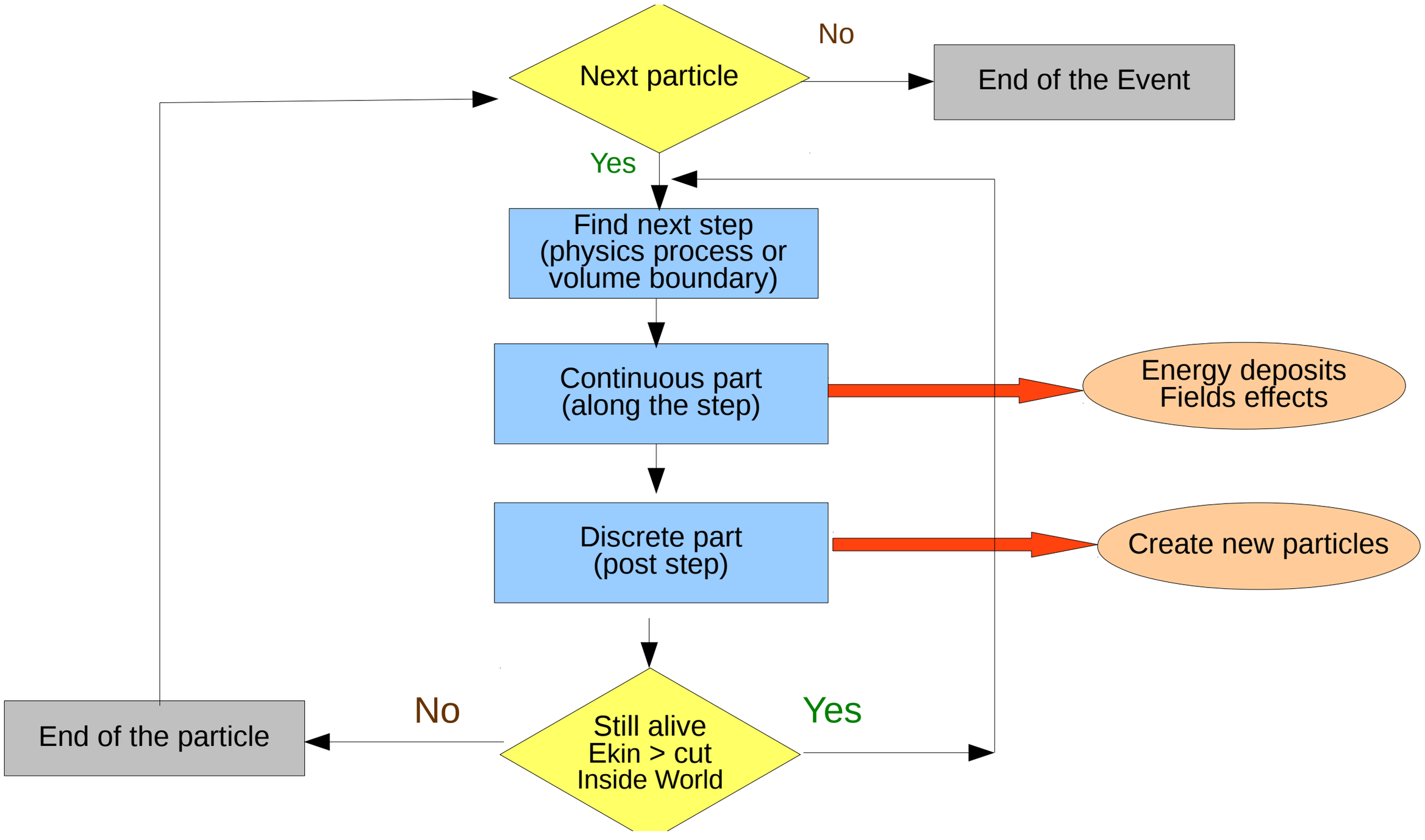
Tracking, not so easy…

- 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

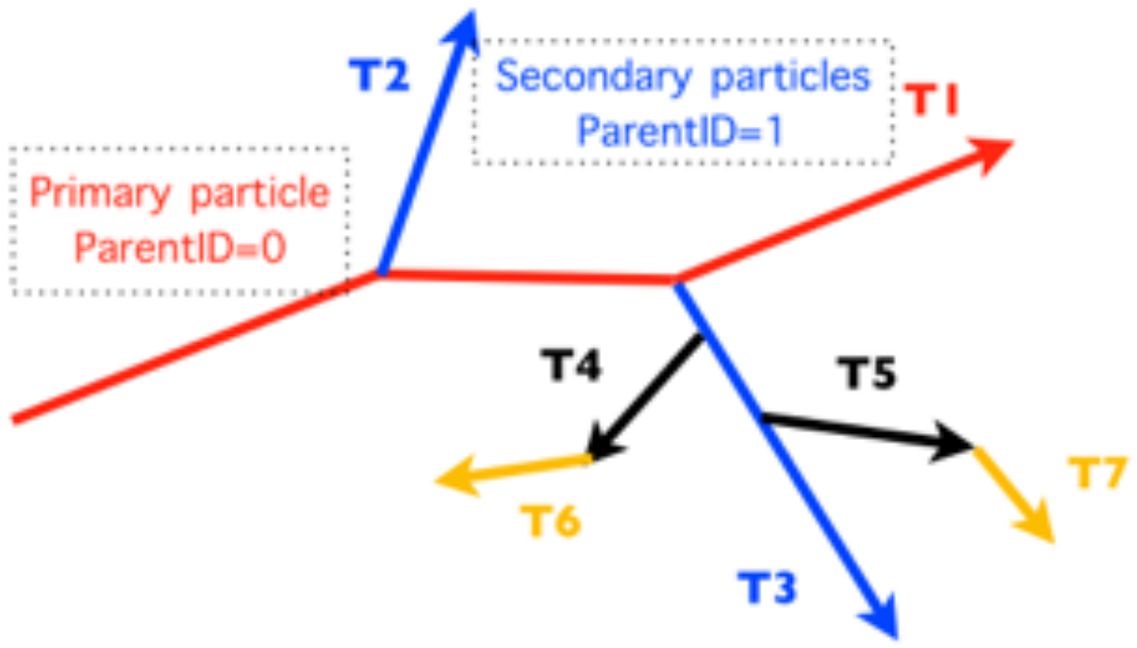
- Simulate explicitly interactions only if the energy loss is above a threshold E_0 (**hard** interactions)
  - Detailed simulation
- The effects of all sub-threshold interactions is described cumulatively (**soft** interactions)
- Hard interactions occur much less frequently than soft interactions

Flowchart of an event



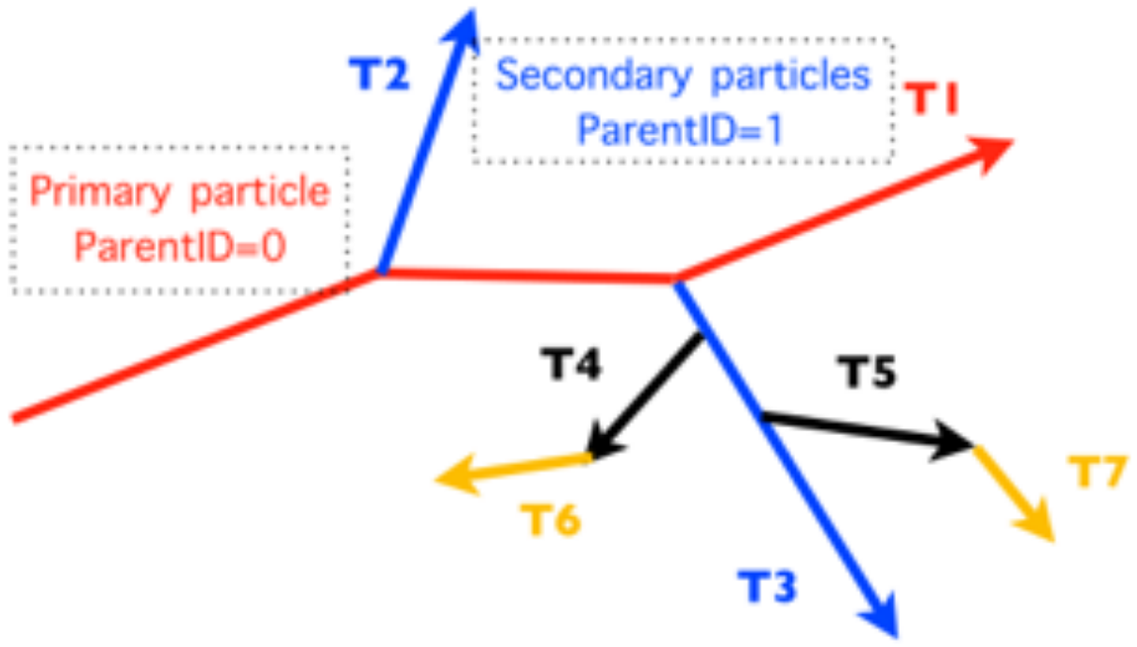
Geant4 way of tracking

- 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



Particles description in Geant4

- Three classes:

  - **G4ParticleDefinition**
    Particle static properties, e.g.:
    name, mass, spin, PDG number

  - **G4DynamicParticle**
    Particle dynamic state, e.g.:
    energy, momentum, polarization

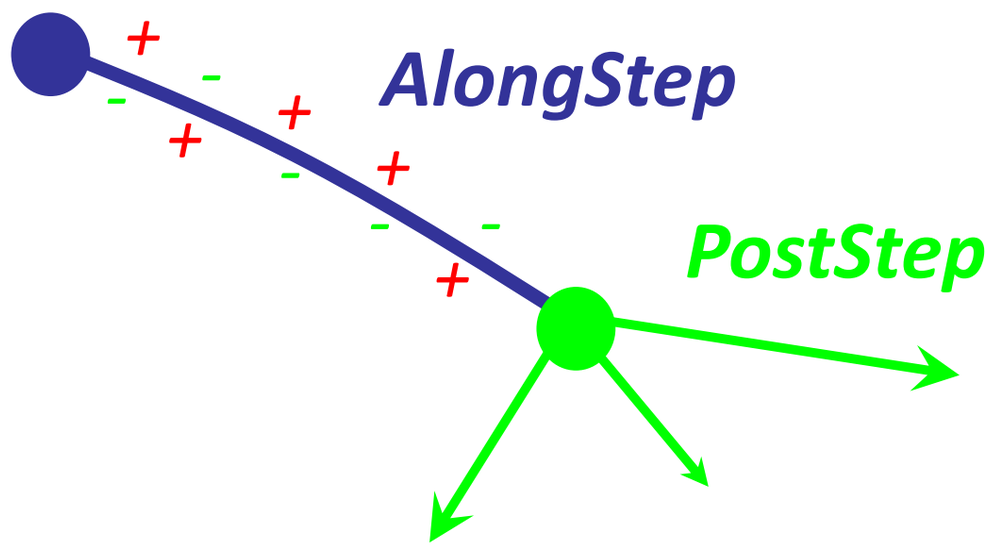
  - **G4Track**
    Information for tracking in a detector simulation, e.g.:
    position, step, current volume, track ID, parent ID

Kind of particles in Geant4

- Particle Data Group (PDG) particles
- Optical photons (different from gammas!)
- Special particles: geantino and charged geantino
  - Only transported in the geometry (no interactions)
  - Charged geantino also feels the EM fields
- Short-lived particles $\left(\tau < 10^{-14}\text{s}\right)$ are not transported (decay immediately)
- Light ions (as deuterons, tritons, alphas)
- Heavier ions represented by a single class: G4Ions

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



Sampling the step size

- In Geant4, particle transportation is a process, 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 are invoked to get a proposed interaction lengths

- The shortest interaction length limits the step and select the process

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^- (δ rays)
  - Protons from hadronic elastic scattering

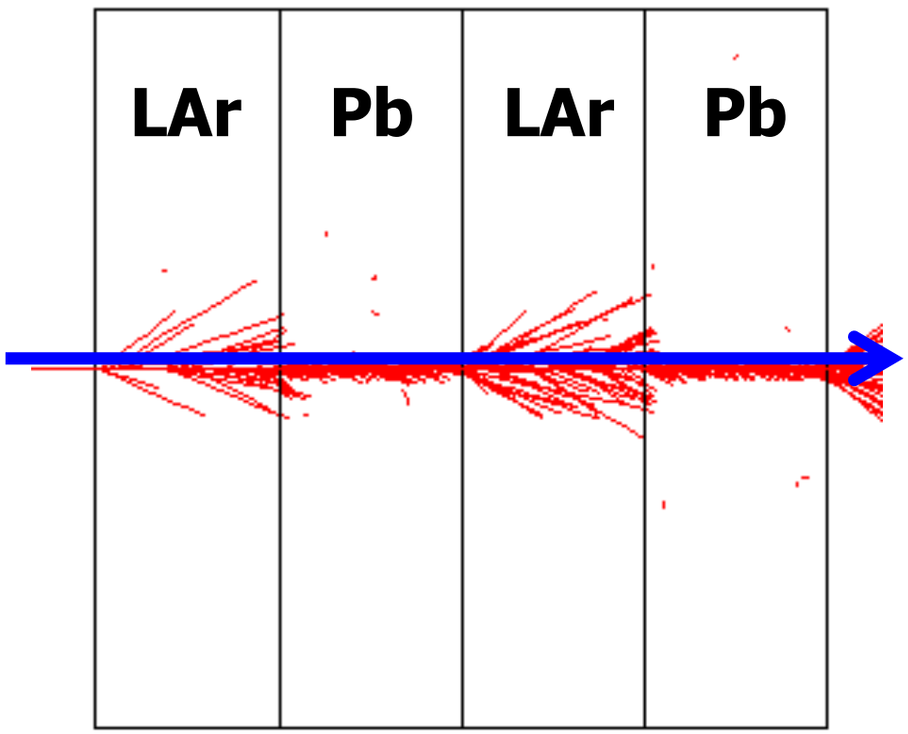
A range cut

- The threshold is a **distance**!
- Default = 1 mm
- Particles unable to travel at least the range cut value are not produced

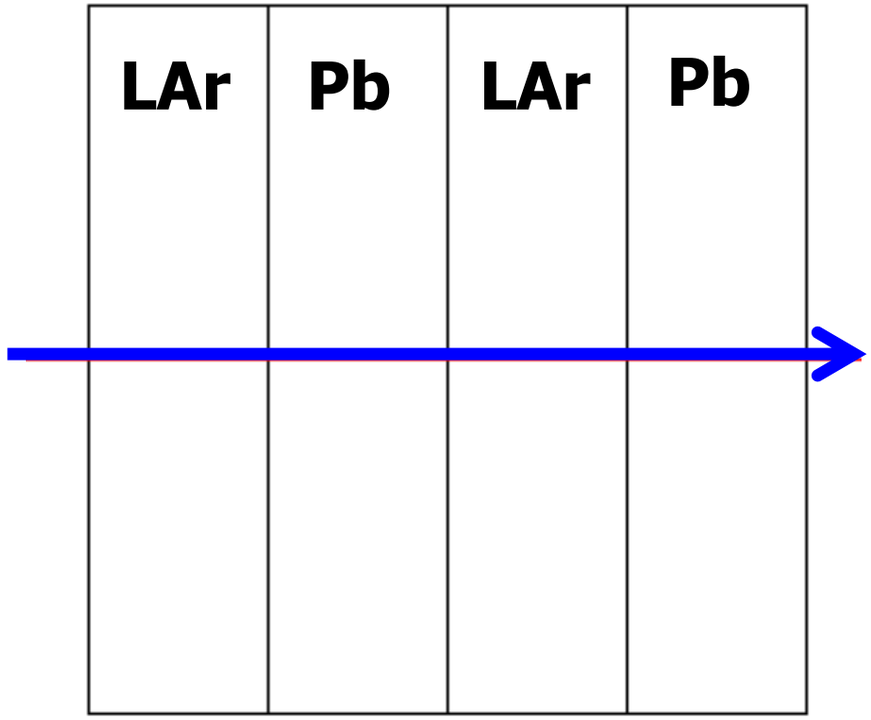
- Sets the "spatial accuracy" of the simulation
- Production threshold is internally converted to an energy threshold for each material

Cut in energy

- 460 keV
- good for LAr
- not for Pb

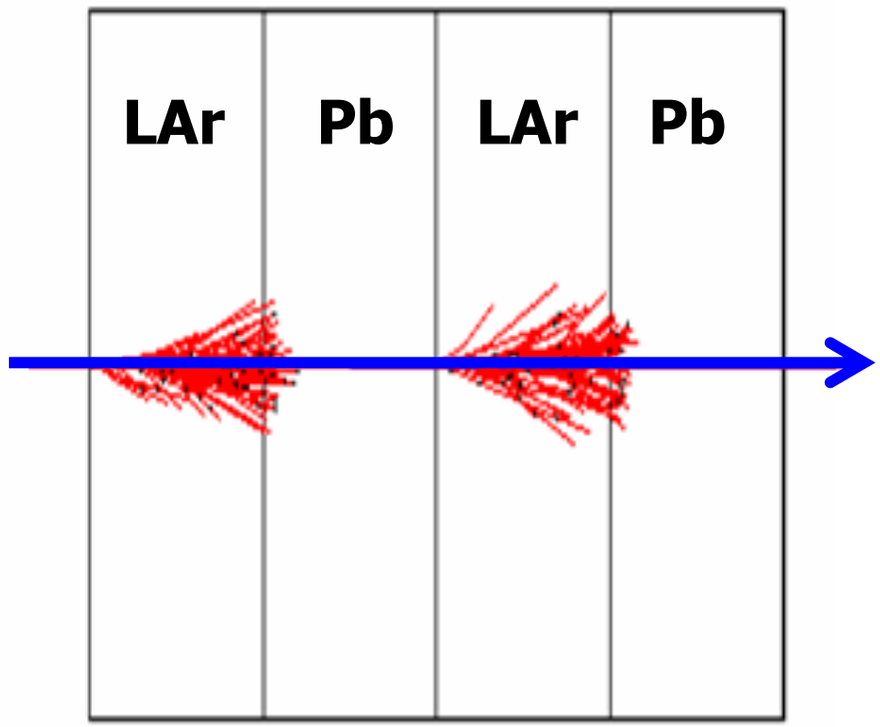


- 2 MeV
- good for Pb
- not for Lar



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

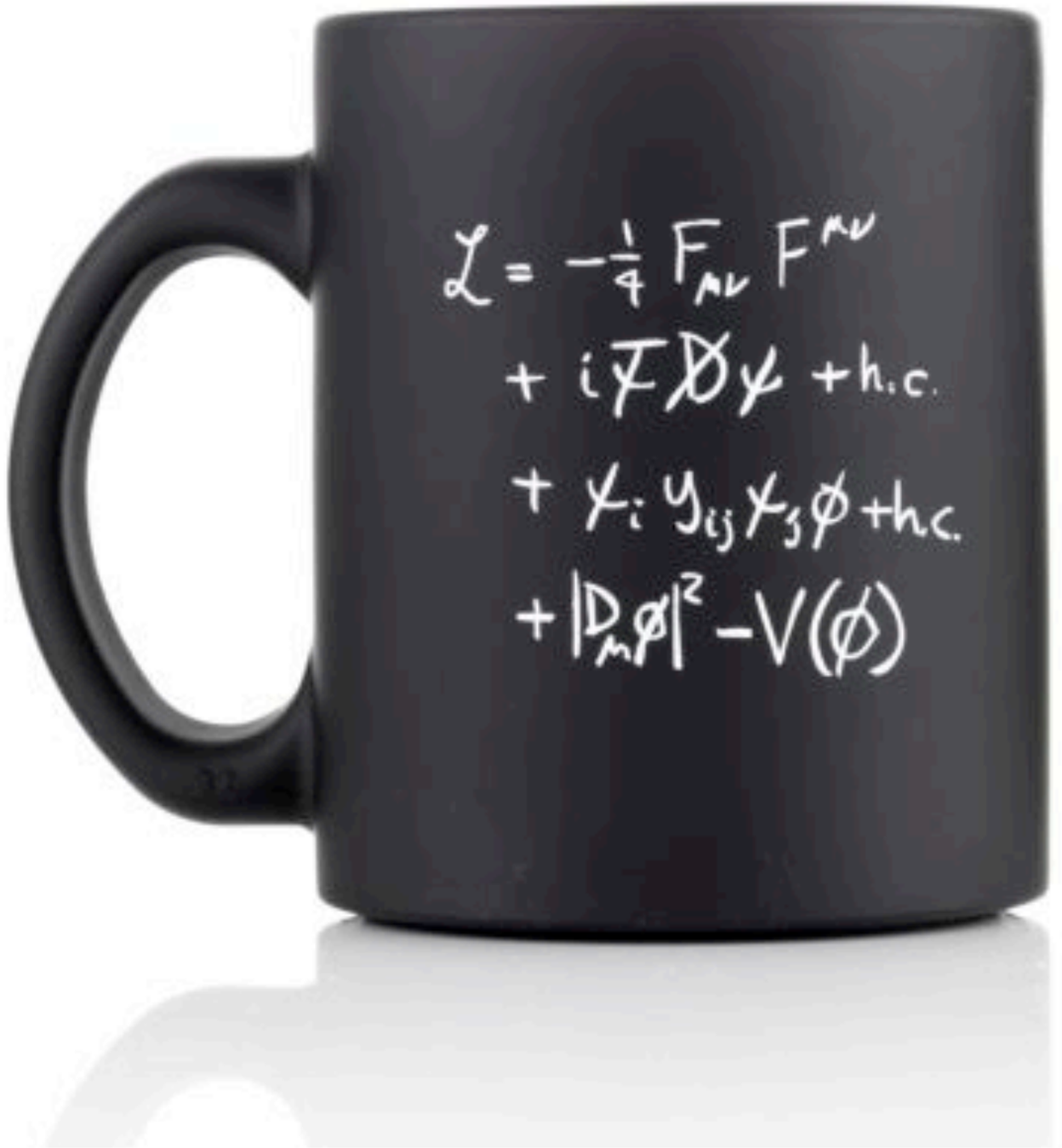
Via UI: /cuts/setLowEdge 250 eV

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
```

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



Physics models

an overview…

Principles

- Provide a general model framework that allows the implementation of complementary/alternative models to describe the same process
(e.g. Compton scattering)

- A given model could work better in a certain energy range

- Decouple models for cross sections and of final state generation

- Provide processes containing
  - Many possible models and cross sections
  - Default cross sections for each model

γ 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

When use Low Energy Models

- Use Low-Energy models (Livermore or Penelope), as an alternative to Standard models, when you:
  - need precise treatment of EM showers and interactions at low-energy (keV scale)
  - are interested in atomic effects, as fluorescence x-rays, Doppler broadening, etc.
  - can afford a more CPU-intensive simulation
  - want to cross-check an other simulation (e.g. with a different model)
- Do not use when you are interested in EM physics > MeV
  - same results as Standard EM models, performance penalty

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

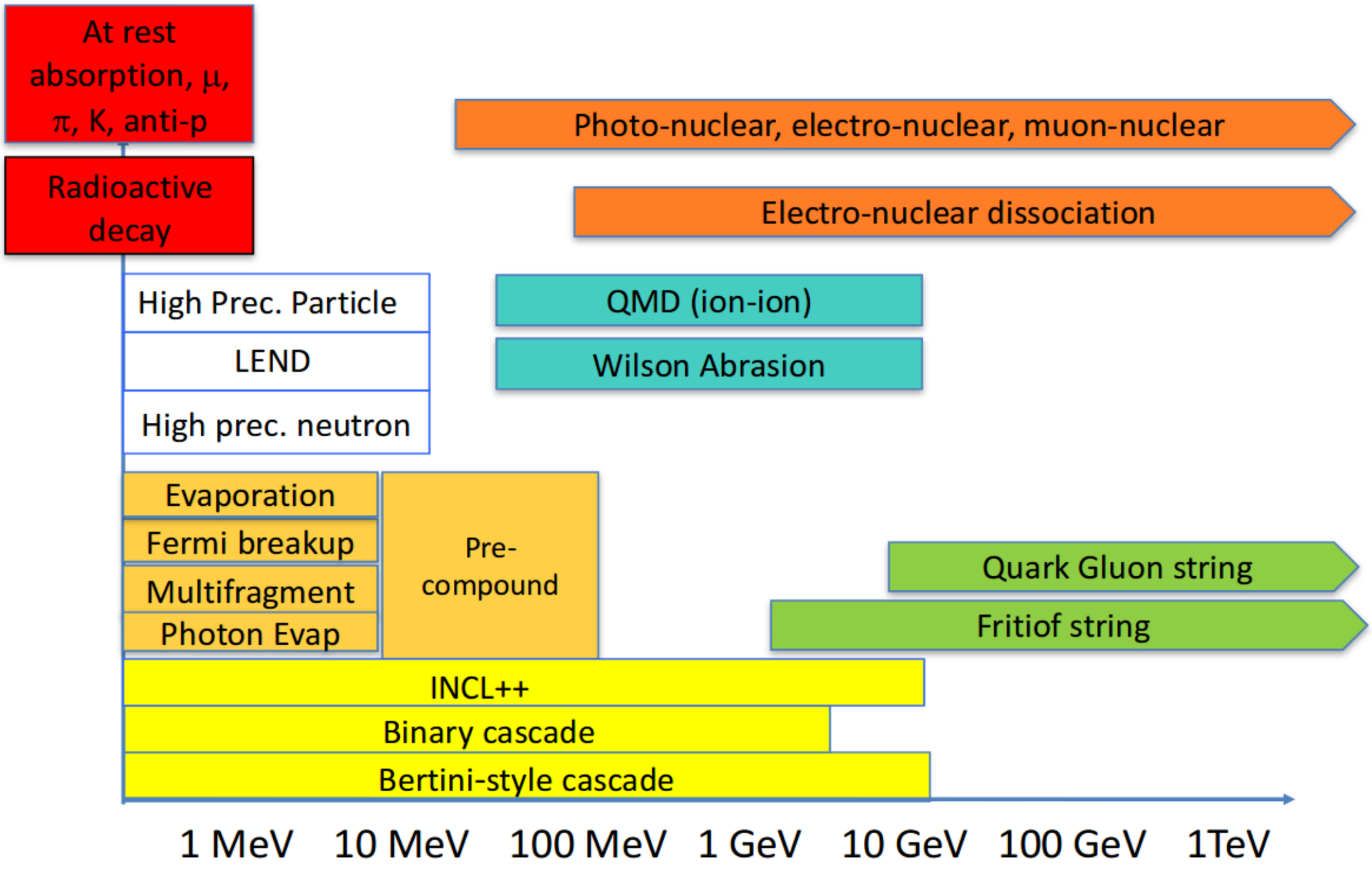
Hadronic models

- 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

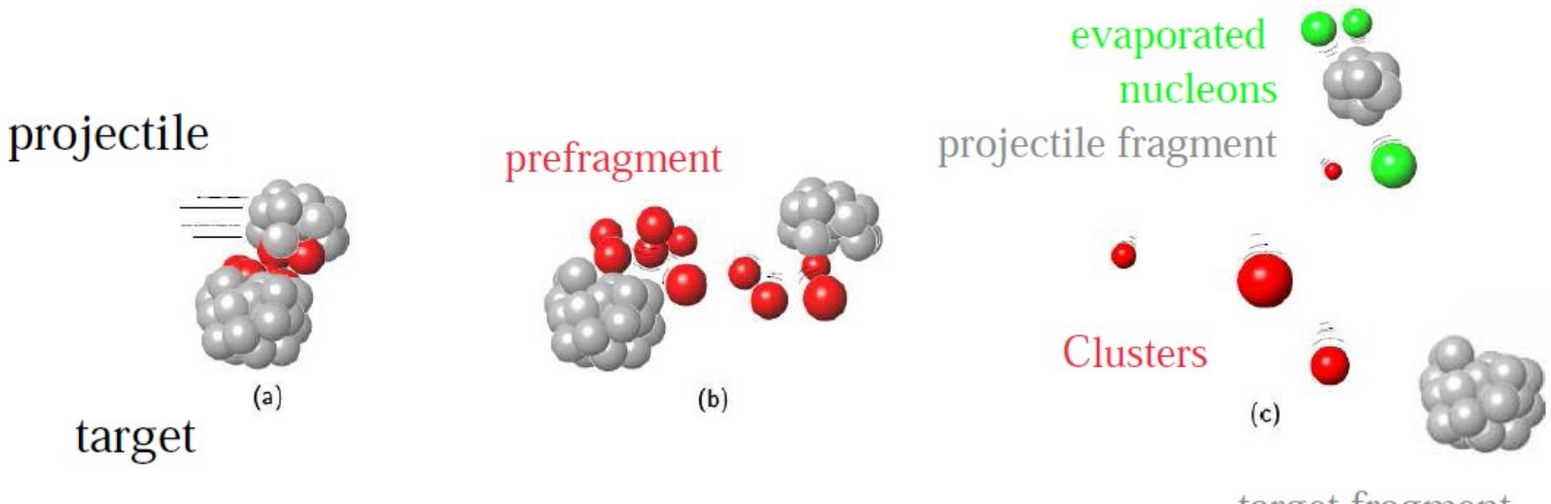
ParticleHP

- Data-driven approach for inelastic reactions for n (in place since many years, named NeutronHP) p, d, t, 3He and α
- 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



Nuclear interactions



- Hadronic interactions are simulated in two different stages:
  - The first one describes the interaction from the collision until the excited nuclear species produced in the collision are in equilibrium
  - The second one, such as the Fermi break-up, models the emission of such excited, but equilibrated, nuclei

Physics lists

How to set the Physics processes in your simulation

What is it?

- A class developed by you inheriting from G4VUserPhysicsList
- One instance per application
- registered to run manager in main()

- Defines:
  - The particle types (electron) you want to use
  - The processes (bremsstrahlung) you want to keep into account
  - The production cuts (e.g. 1 mm for electrons, ...)

User classes

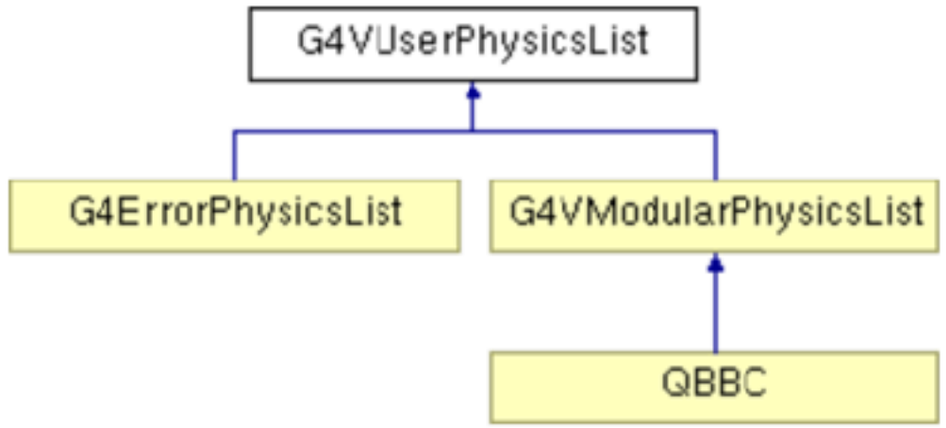
- You **have** to write the **main**()
- Initialisation classes:
  - **`G4VUserDetectorConstruction`**
  - **`G4VUserPhysicsList`**
  - **`G4VUserActionInitialization`**
- Action classes
  - **`G4VUserPrimaryGeneratorAction`**
  - **`G4UserRunAction`**
  - **`G4UserEventAction`**
  - **`G4UserStackingAction`**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

classes written in red are mandatory!

G4VUserPhysicsList Class Reference

`#include <G4VUserPhysicsList.hh>`

Inheritance diagram for G4VUserPhysicsList:



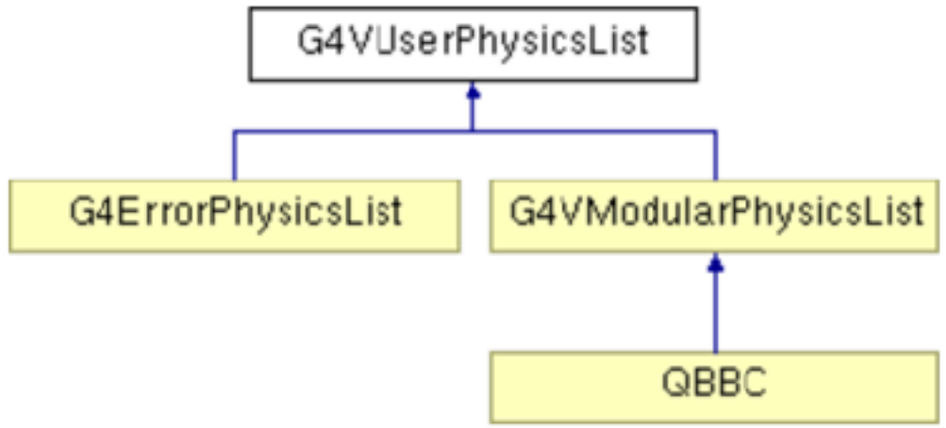
Public Member Functions

	G4VUserPhysicsList ()
virtual	~G4VUserPhysicsList ()
	G4VUserPhysicsList (const G4VUserPhysicsList &)
G4VUserPhysicsList &	operator= (const G4VUserPhysicsList &)
virtual void	ConstructParticle ()=0
void	Construct ()
virtual void	ConstructProcess ()=0
void	UseCoupledTransportation (G4bool vl=true)
virtual void	SetCuts ()
void	SetDefaultCutValue (G4double newCutValue)
G4double	GetDefaultCutValue () const
void	BuildPhysicsTable ()

G4VUserPhysicsList Class Reference

`#include <G4VUserPhysicsList.hh>`

Inheritance diagram for G4VUserPhysicsList:



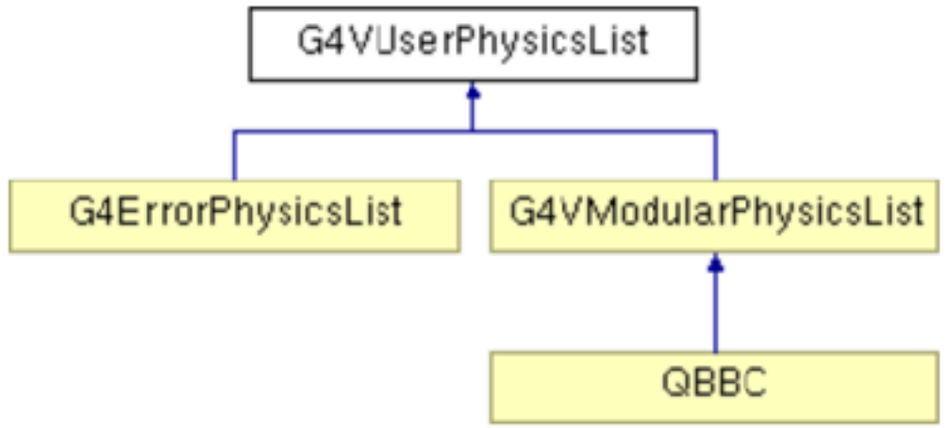
Public Member Functions

	G4VUserPhysicsList ()
virtual	~G4VUserPhysicsList ()
	G4VUserPhysicsList (const G4VUserPhysicsList &)
G4VUserPhysicsList &	operator= (const G4VUserPhysicsList &)
virtual void	ConstructParticle ()=0
void	Construct ()
virtual void	ConstructProcess ()=0
void	UseCoupledTransportation (G4bool vl=true)
virtual void	SetCuts ()
void	SetDefaultCutValue (G4double newCutValue)
G4double	GetDefaultCutValue () const
void	BuildPhysicsTable ()

G4VUserPhysicsList Class Reference

`#include <G4VUserPhysicsList.hh>`

Inheritance diagram for G4VUserPhysicsList:



Public Member Functions

	G4VUserPhysicsList ()
virtual	~G4VUserPhysicsList ()
	G4VUserPhysicsList (const G4VUserPhysicsList &)
G4VUserPhysicsList &	operator= (const G4VUserPhysicsList &)
virtual void	ConstructParticle ()=0
void	Construct ()
virtual void	ConstructProcess ()=0
void	UseCoupledTransportation (G4bool vl=true)
virtual void	SetCuts ()
void	SetDefaultCutValue (G4double newCutValue)
G4double	GetDefaultCutValue () const
void	BuildPhysicsTable ()

3 ways to do a physics list

- **Manual:** Write your own class, to specify all particles & processes that may occur in the simulation (very flexible, but difficult)

- **Physics constructors:** Combine your physics from pre-defined sets of particles and processes.
  - Still you define your own class – modular physics list (easier)

- **Reference physics lists:** Take one of the pre-defined physics lists.
  - You don't create any class (easy)

Manual

```
class MyPhysicsList : public G4VUserPhysicsList {
public:
    // ...
    void ConstructParticle();  // pure virtual
    void ConstructProcess();   // pure virtual
    void SetCuts();
    // ...
}
```

- Advantage: most flexible
- Disadvantages:
  - most verbose
  - most difficult to get right

G4VUserPhysicsList

- ConstructParticle()
  - choose the particles you need in your simulation, define all of them here
- ConstructProcess()
  - for each particle, assign all the physics processes relevant to your simulation
- SetCuts()
  - set the range cuts for secondary production for processes with infrared divergence

Modular

```
class MyPhysicsList : public G4VModularPhysicsList {
public:
    MyPhysicsList();                // define physics constructors
    void ConstructParticle();       // optional
    void ConstructProcess();        // optional
    void SetCuts();                 // optional
}
```

- Similar structure as G4VUserPhysicsList (same methods to override, though not mandatory)
- Differences to the manual way:
  - Particles and processes typically handled by physics constructors (still customizable)
  - Transportation automatically included

How to

```
MyModularList::MyModularList() {
  // Hadronic physics
  RegisterPhysics(new G4HadronElasticPhysics());
  RegisterPhysics(new G4HadronPhysicsFTFP_BERT_TRV());
  // EM physics
  RegisterPhysics(new G4EmStandardPhysics());
}
```

- Add physics constructor in the class constructor
- This already works and no further method overriding is necessary!

Reference physics list

- Pre-defined ("plug-and-play") physics lists
- already containing a complete set of particles & processes (that work together)
- targeted at specific area of interest (HEP, medical physics, ...)
- constructed as modular physics lists, built on top of physics constructors
- customizable (by calling appropriate methods before initialization)

How to

```
#include "QGSP_BERT.hh"
int main()  {
    // Run manager
    G4RunManager * runManager = new G4RunManager();
    // ...
    G4VUserPhysicsList* physics = new QGSP_BERT();
    // Here, you can customize the "physics" object
    runManager->SetUserInitialization(physics);
    // ...
}
```

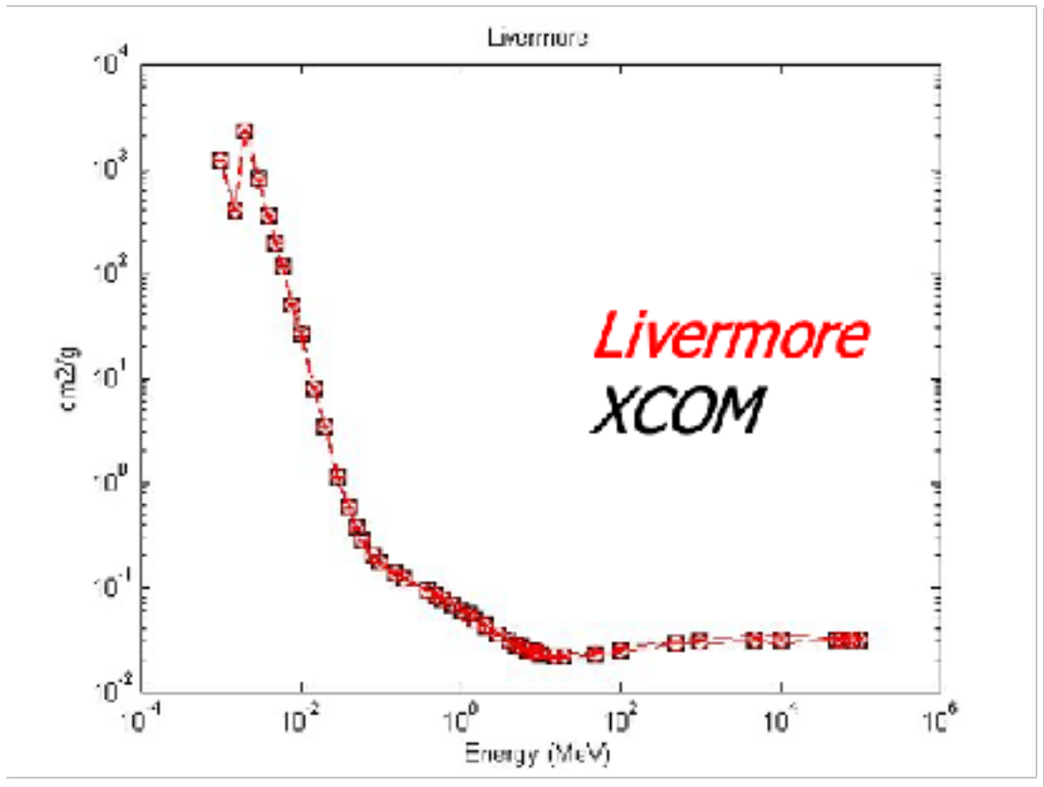
- in the main() function, just register an instance of the physics list to the G4(MT)RunManager

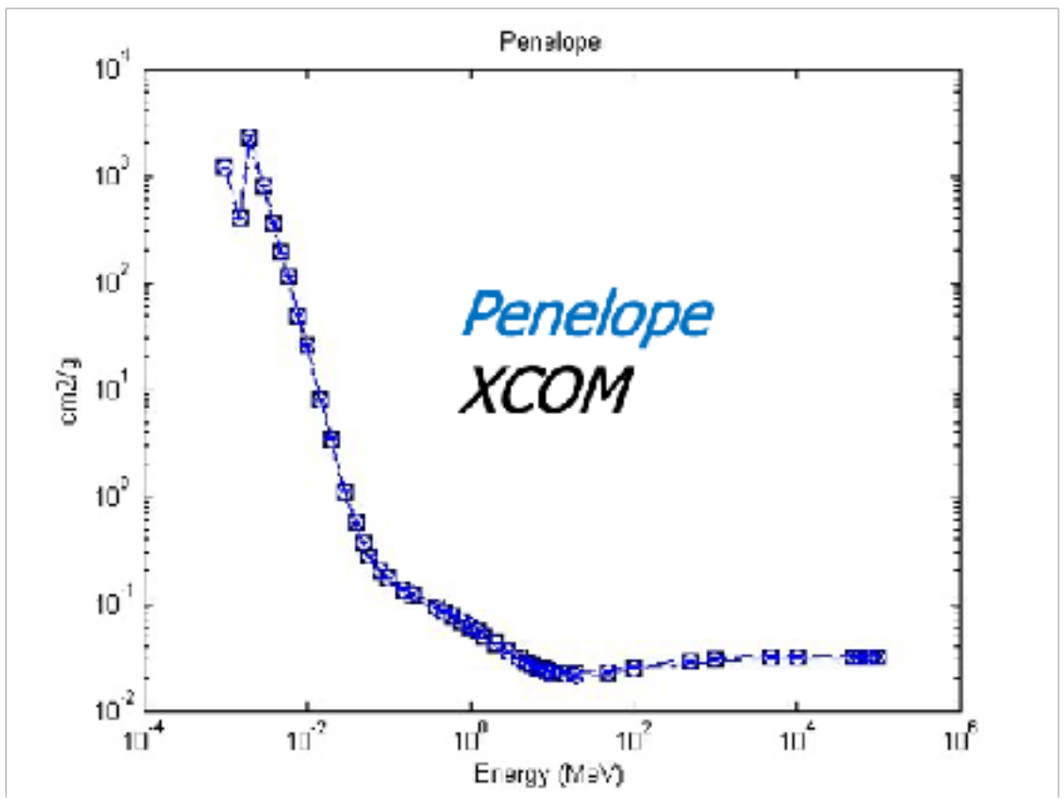
Validation overview

Quick…

EM Validation

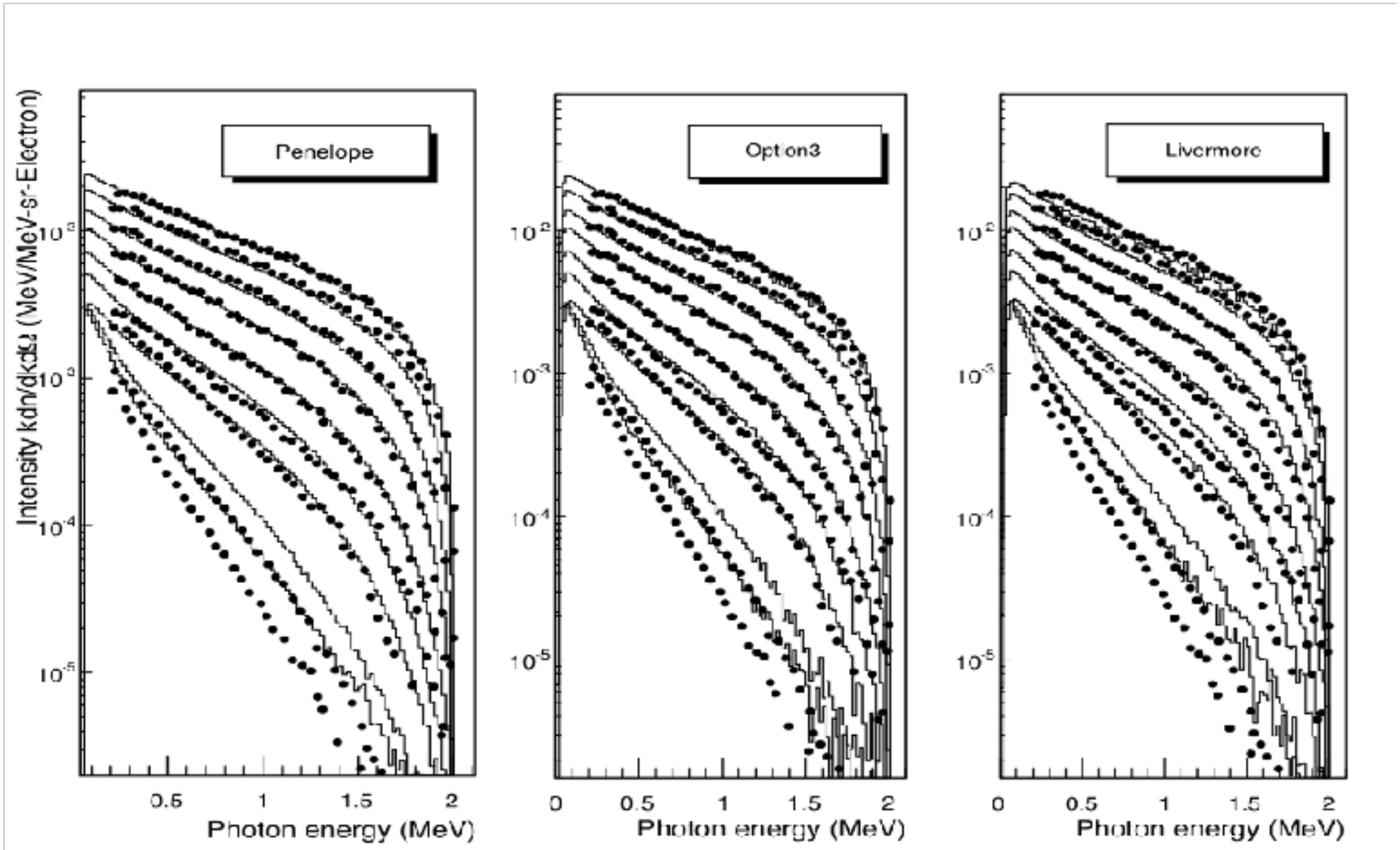
- Tens of papers and studies published
  - Geant4 Collaboration + User Community
- Results can depend on the specific observable/reference
  - Data selection and assessment critical





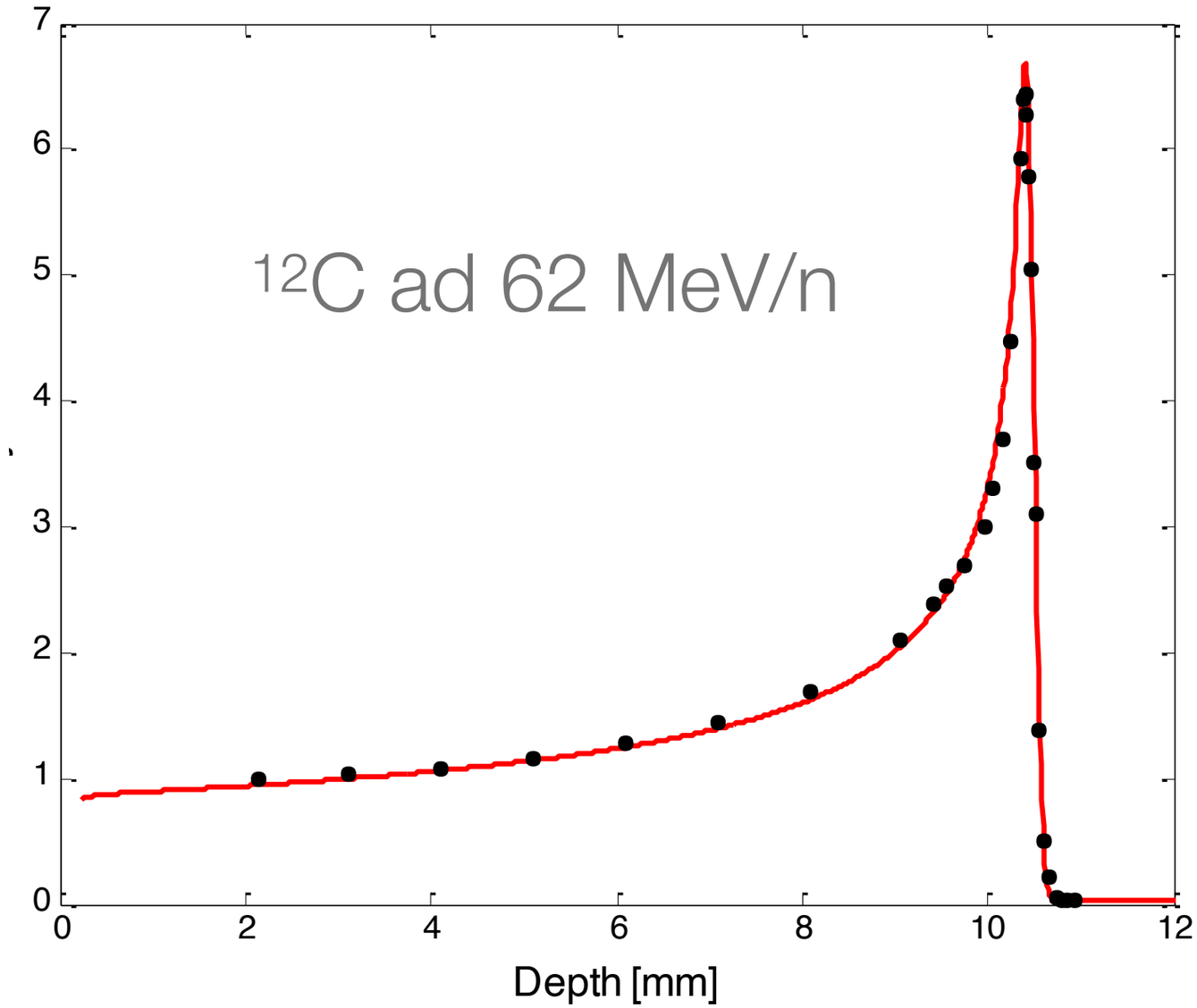
EM Validation

- In general satisfactory agreement



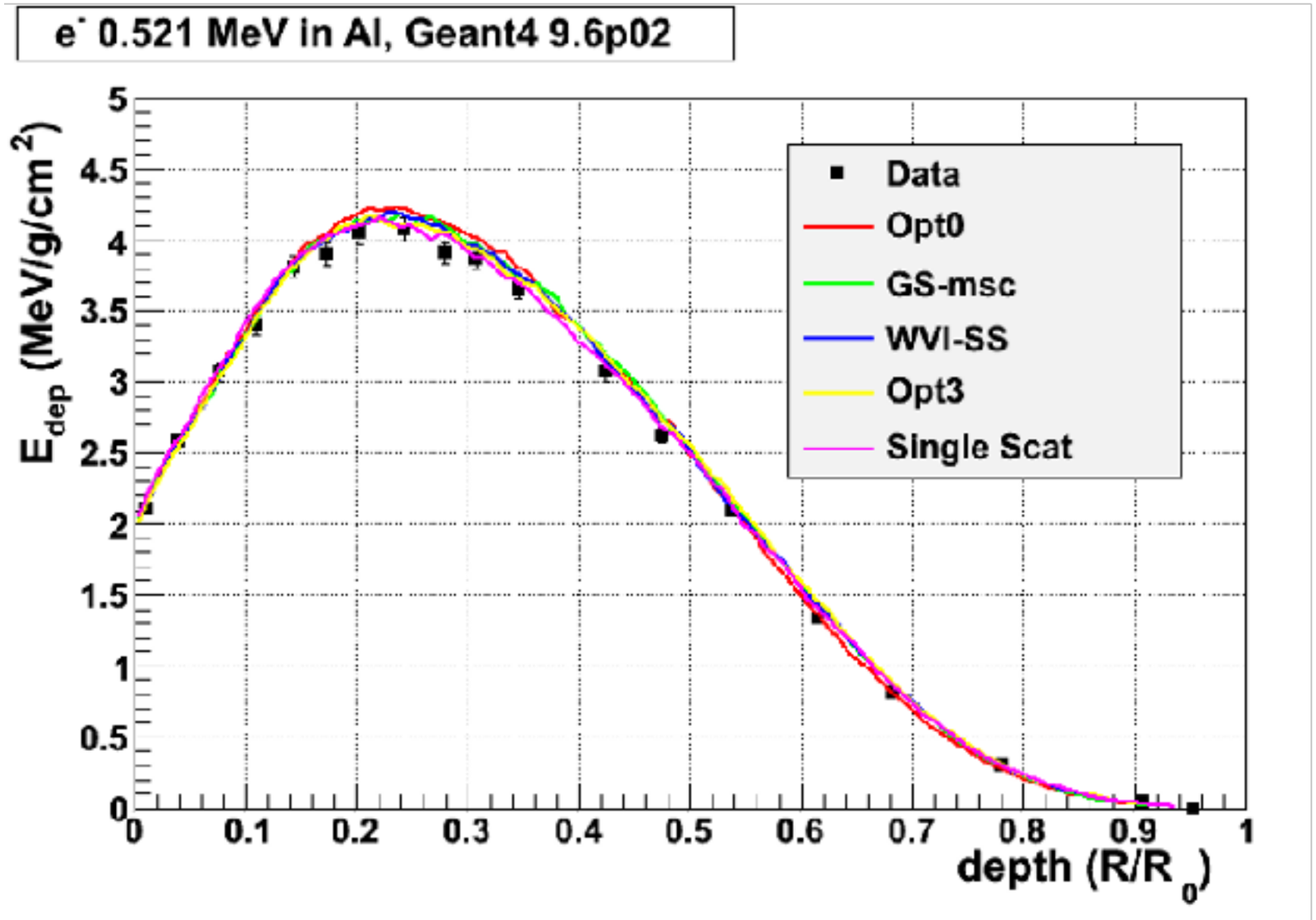
EM Validation

- In general satisfactory agreement



EM Validation

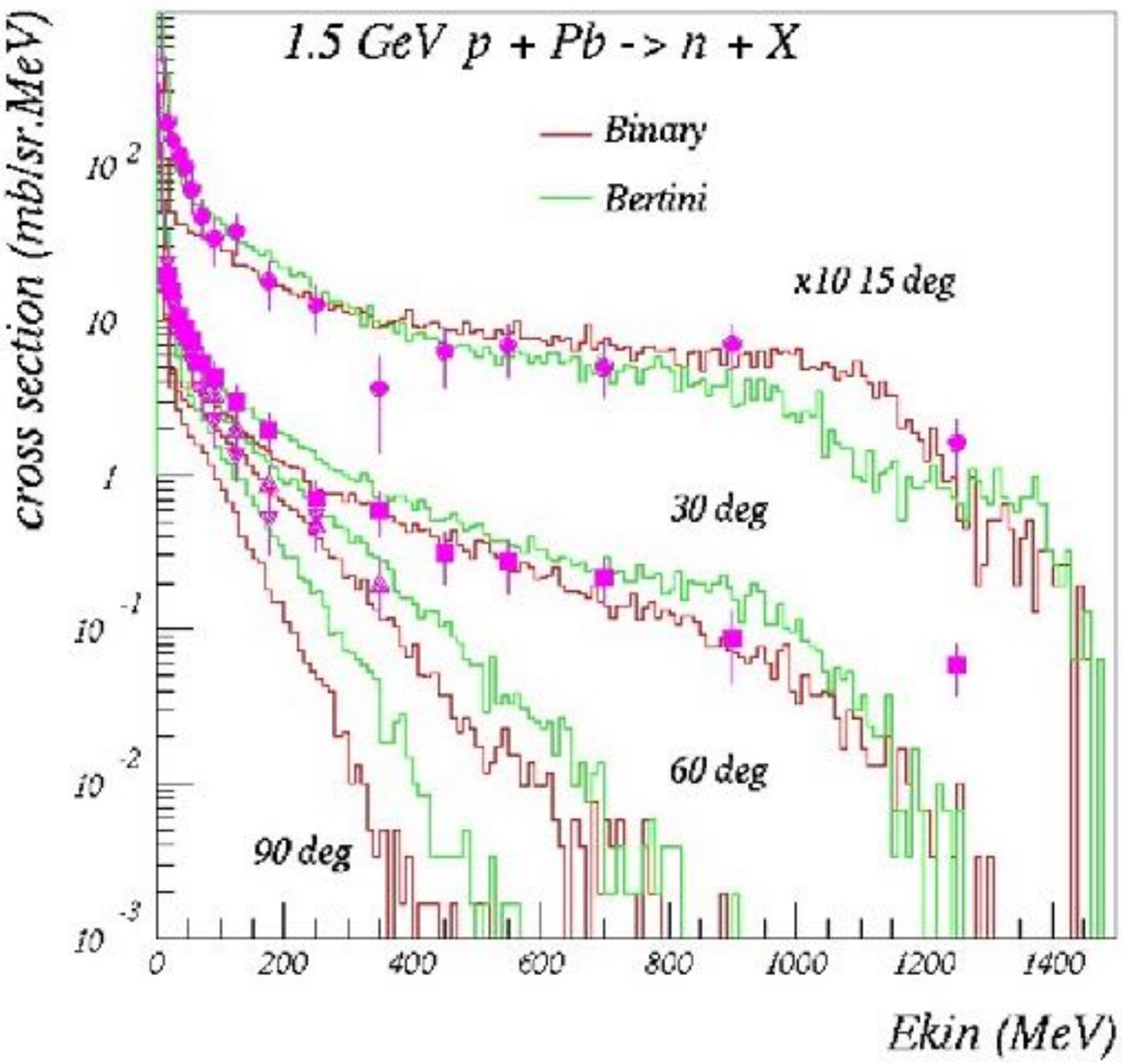
- In general satisfactory agreement



e- showers (longitudinal profile)

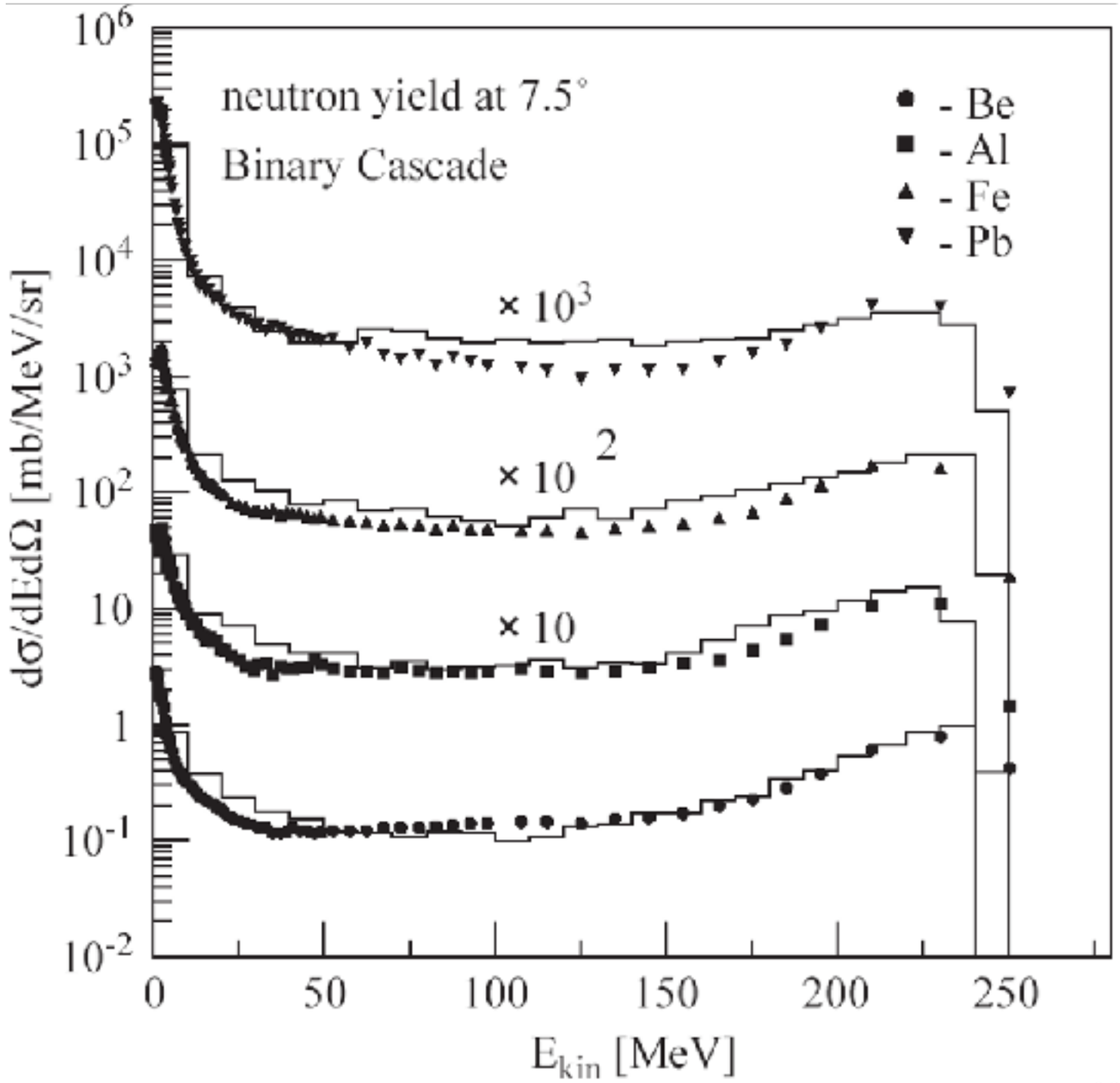
Nuclear fragmentation

- Bertini and Binary cascade models
- neutron production vs. angle
- 1.5 GeV protons
- Lead target



Neutron production

- Binary cascade model
- double differential cross-section for neutrons produced
- 256 MeV protons
- different targets



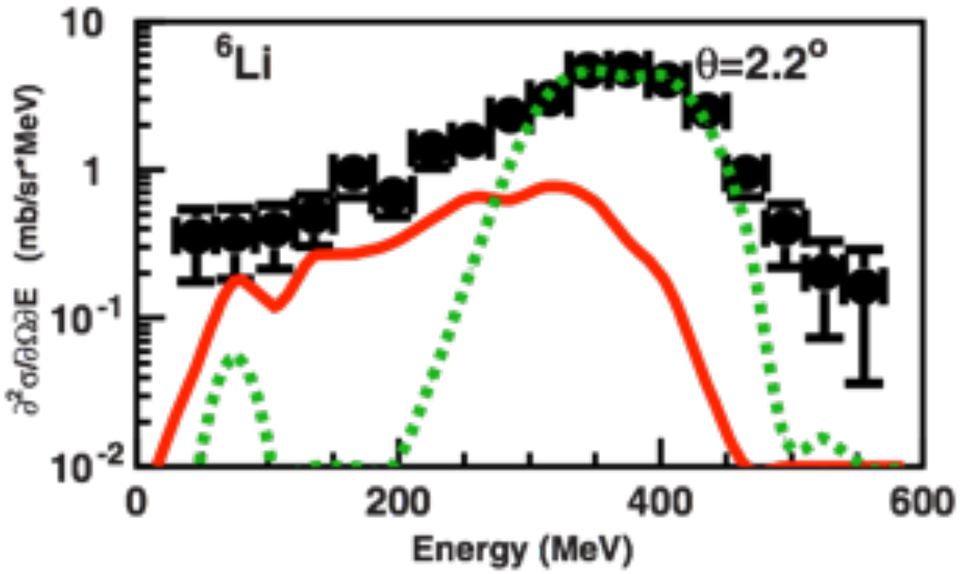
Nuclear interactions below 100 MeV/u

- Despite the numerous and relevant application would use it, there is no dedicated model to nuclear interaction below 100 MeV/u in Geant4

- Many papers showed the difficulties of Geant4 in this energy domain:
  - Braunn et al. have shown discrepancies up to one order of magnitude in ^{12}C fragmentation at 95 MeV/u on thick PMMA target
  - De Napoli et al. showed discrepancy specially on angular distribution of the secondaries emitted in the interaction of 62 MeV/u ^{12}C on thin carbon target
  - Dudouet et al. found similar results with a 95 MeV/u ^{12}C beam on H, C, O, Al and Ti targets

- **Exp. data**
- **G4-BIC**
- **G4-QMD**

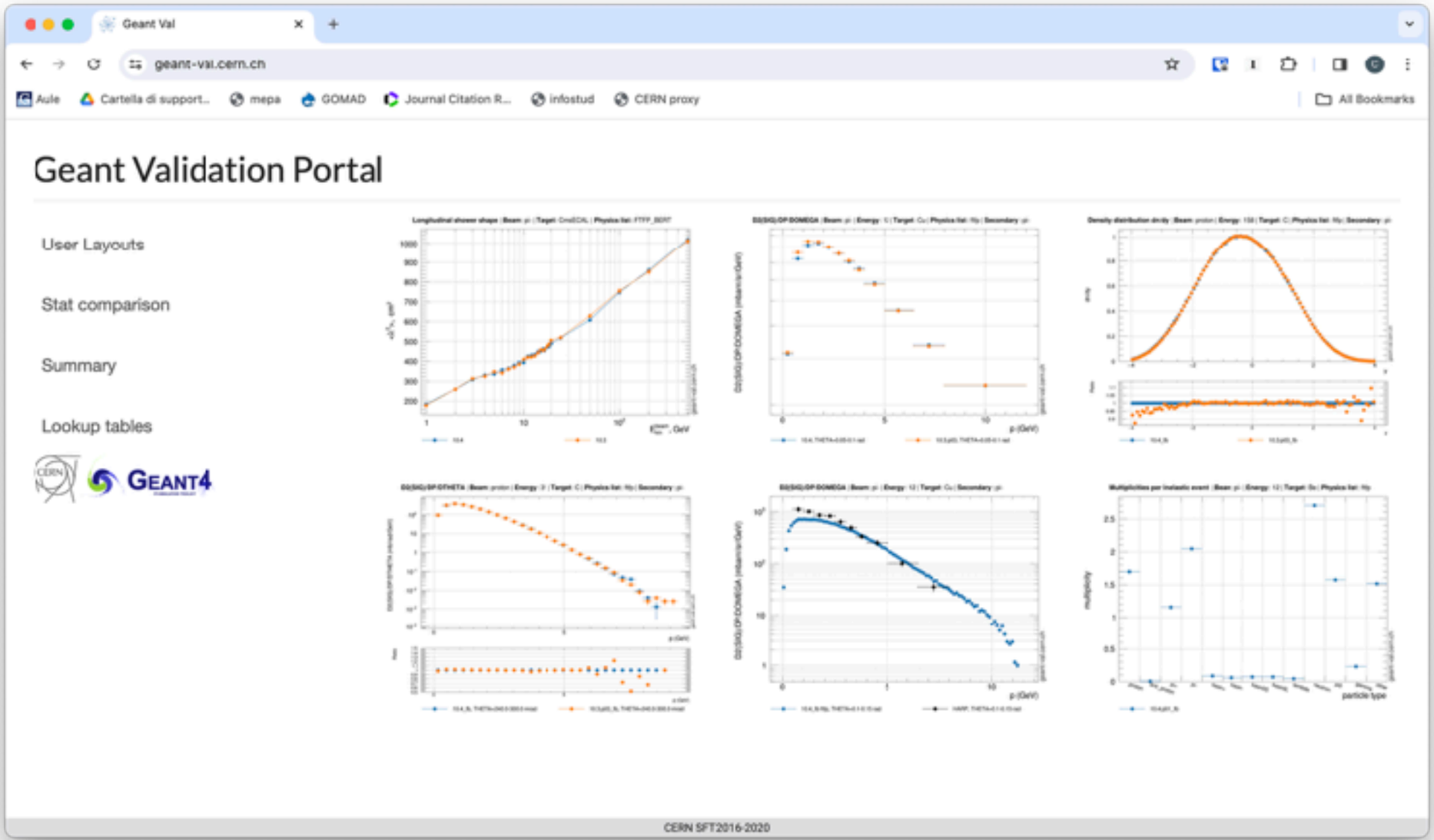
[Plot from De Napoli et al. Phys. Med. Biol., vol. 57, no. 22, pp. 7651–7671, Nov. 2012]



Cross section of the ^{6}Li production at 2.2 degree in a ^{12}C on ^{nat}C reaction at 62 MeV/u.

Geant-val

- https://geant-val.cern.ch/

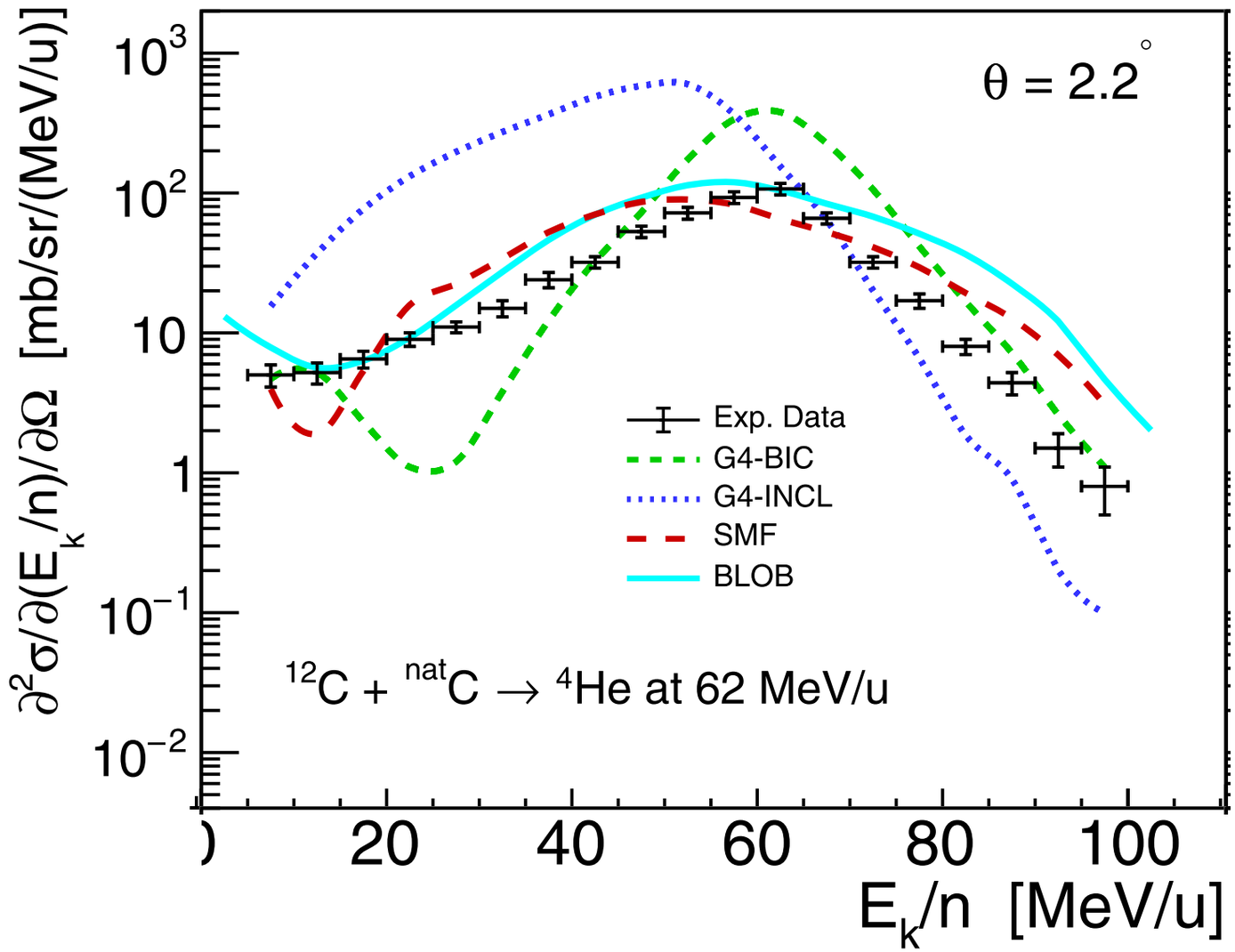


Backup slides

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

Interfacing new low-energy models



- C. Mancini-Terracciano et al. *Preliminary results in using Deep Learning to emulate BLOB, a nuclear interaction model*. Submitted to Phys. Med
- C. Mancini-Terracciano et al. *Preliminary results coupling SMF and BLOB with Geant4* Phys. Med. vol. 67, no. 22, Nov. 2019
- C. Mancini-Terracciano et al. *Validation of Geant4 nuclear reaction models for hadron therapy and preliminary results with BLOB* IFMBE Proceedings Series 68/1 (mar. 2018)
- P. Napolitani, M. Colonna and C. Mancini-Terracciano. *Cluster formation in nuclear reactions from mean-field inhomogeneities*. In: Journal of Physics: Conference Series 1014.1 (mar. 2018)