X SEMINAR ON SOFTWARE FOR NUCLEAR, SUBNUCLEAR AND APPLIED PHISICS

Porto Conte, Alghero, Italy
3 - 7 June 2013

Physics in Geant4: Particles, processes, cuts and models

Geant 4 **tutorial course**

Introduction

Mandatory user classes in a Geant4:

- `G4VUserPrimaryGeneratorAction`
- `G4VUserDetectorConstruction`
- `G4VUserPhysicsList`

***Particles**, **physics processes** and **cut-off parameters** to be used in the simulation must be defined in the **`G4VUserPhysicsList`** class*

Why a physics list?

- "*Physics is physics* – shouldn't Geant4 provide, as a default, a complete set of physics that everyone can use?"
- **NO**:
  - Software can only capture Physics through a modelling
    - No unique Physics modelling
      - Very much the case for hadronic physics
      - But also the electromagnetic physics
      - Existing models still evolve and new models are created
    - Some modellings are more suited to some energy ranges
      - Medical applications not interested in multi-GeV physics in general
      - HEP experiments not interested in effects due to atomic shell structure
  - computation speed is an issue
    - a user may want a less-detailed, but faster approximation

Why a physics list?

- For this reason Geant4 takes an atomistic, rather than an integral approach to physics
  - provide many physics components (processes) which are de-coupled from one another
  - user selects these components in custom-designed physics lists

- This physics environment is built by the user in a flexible way:
  - picking up the particles he wants
  - picking up the physics to assign to each particle

- User must have a good understanding of the physics required
  - omission of particles or physics could cause errors or poor simulation

User may also use some provided "ready-to-use" physics list

`G4VUserPhysicsList`: required methods

`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
  - What's a process ?
    - a class that defines how a particle should interact with matter, or decays
      - » it's where the physics is!

`SetCuts()` :

- set the range cuts for secondary production
  - What's a range cut ?
    - a threshold on particle production
      - » Particle unable to travel at least the range cut value are not produced

Particles: basic concepts

There are three levels of class to describe particles in Geant4:

- **`G4ParticleDefinition`**
  – define a particle
  aggregates information to characterize a particle's properties (name, mass, spin, etc…)

- **`G4VDynamicParticle`**
  – describe a particle interacting with materials
  aggregates information to describe the dynamic of particles (energy, momentum, polarization, etc…)

- **`G4VTrack`**
  – describe a particle travelling in space and time
  includes all the information for tracking in a detector simulation (position, step, current volume, track ID, parent ID, etc…)

Definition of a particle

Geant4 provides the **`G4ParticleDefinition`** definition class to represent a large number of elementary particles and nuclei, organized in six major categories:
lepton, meson, baryon, boson, shortlived and ion

- Each particle is represented by its own class, which is derived from **`G4ParticleDefinition`**
- Proprieties characterizing individual particles are "read only" and can not be changed directly

*User must define **all particles** type which are used in the application: not only **primary particles** but also all other particles which may appear as **secondaries** generated by the used physics processes*

Constructing particles

Due to the large number of particles can be necessary to define, this method sometimes can be not so comfortable

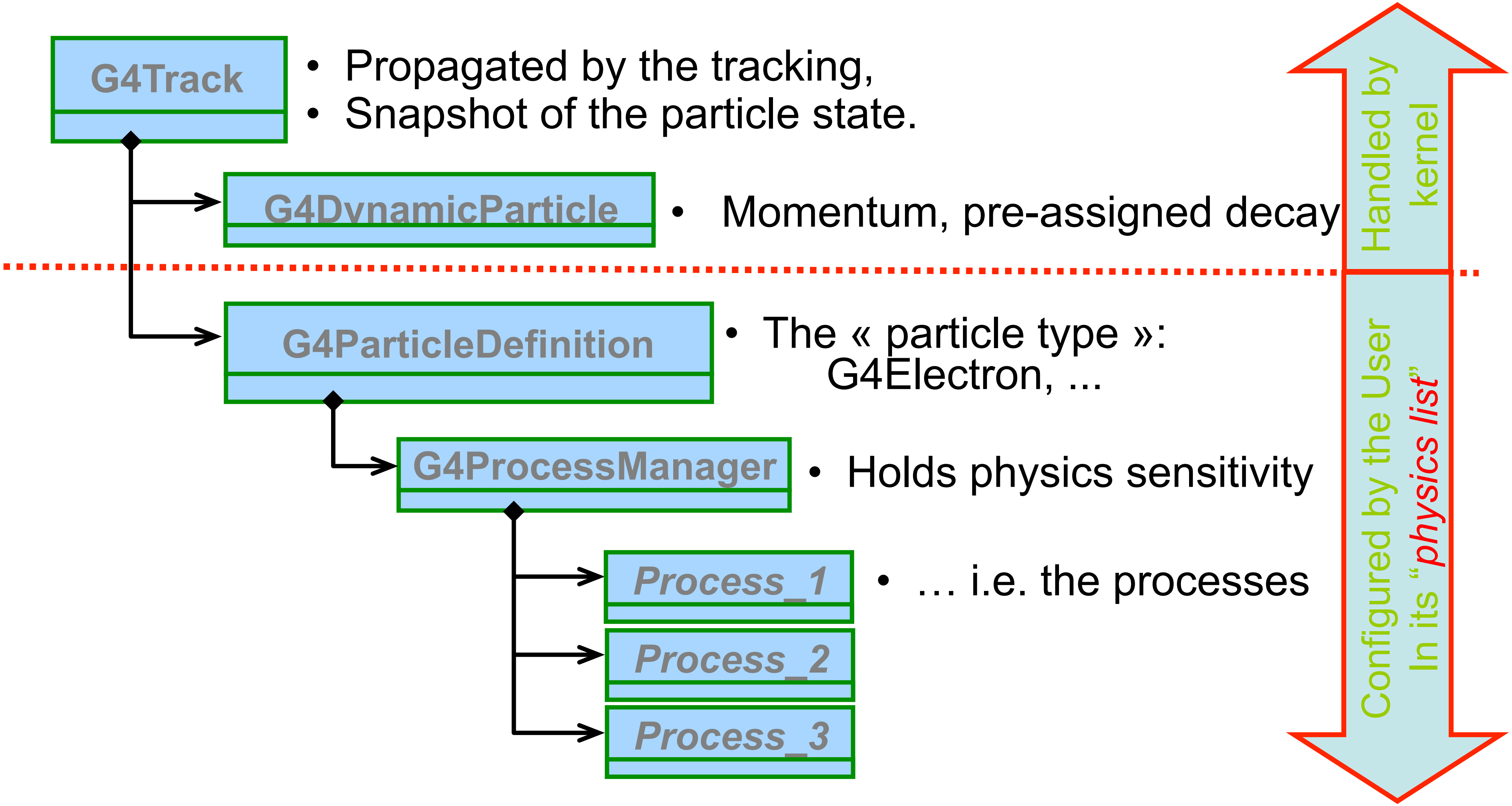
```
void MyPhysicsList::ConstructParticle
 ()
{
    G4Electron::ElectronDefinition();
    G4Proton::ProtonDefinition();
    G4Neutron::NeutronDefinition();
    G4Gamma::GammaDefinition();
    ....
}
```

It is possible to define **all** the particles belonging to a **Geant4 category:**

- G4LeptonConstructor
- G4MesonContructor
- G4BarionConstructor
- G4BosonConstructor
- G4ShortlivedConstructor
- G4IonConstructor

```
void
 MyPhysicsList::ConstructBaryons()
{
    // Construct all baryons
 G4BaryonConstructor pConstructor;
 pConstructor.ConstructParticle();
}
```

From particles to processes



- Propagated by the tracking,
- Snapshot of the particle state.
- Momentum, pre-assigned decay
- The « particle type »: G4Electron, ...
- Holds physics sensitivity
- … i.e. the processes

Processes

Physics processes describe how particles interact with materials

Geant4 provides seven major categories of processes:

- Electromagnetic
- Hadronic
- Decay
- Optical
- Photolepton_hadron
- Parameterization
- Transportation

A process does two things:

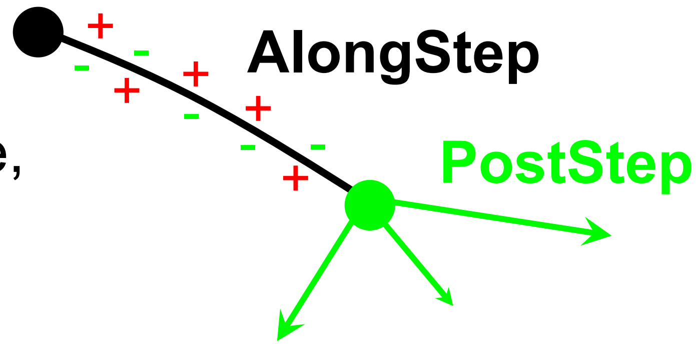
- decides when and where an interaction will occur
  - method: **`GetPhysicalInteractionLength()`** → *limit the step*
  - this requires a cross section
  - for the transportation process, the distance to the nearest object
- generates the final state of the interaction (changes momentum, generates secondaries, etc.)
  - method: **`DoIt()`**
  - this requires a model of the physics

G4Vprocess class

Physics processes are derived from the **G4VProcess** base class

- Abstract class defining the common interface of all processes in Geant4:
  - Used by all physics processes (also by the transportation, etc…
  - Defined in **source/processes/management**
- Define three kinds of actions:
  - **AtRest** actions:

    - Decay, e^+ annihilation …
  - **AlongStep** actions:
    - To describe continuous (inter)actions, occurring along the path of the particle, like ionisation;



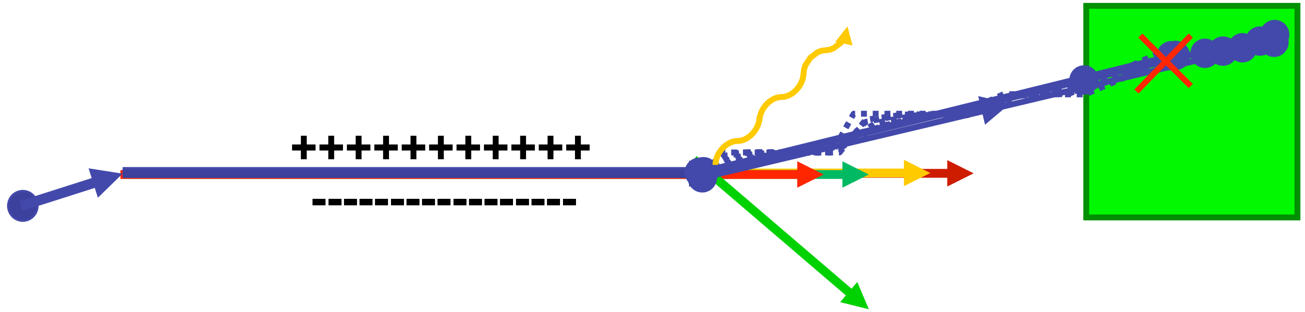
  - **PostStep** actions:
    - For describing point-like (inter)actions, like decay in flight, hadronic interactions …

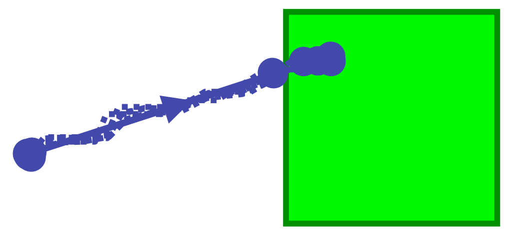
A process can implement a combination of them (decay = AtRest + PostStep)

Handling multiple processes

- STAGE 1: a particle is shot and “transported”
- STAGE 2: all processes associated to the particle propose a geometrical step length (depends on process cross-section)
- STAGE 3: The process proposing the shortest step “wins” and the particle is moved to destination (if shorter than “Safety”)
- STAGE 4: All processes “along the step” are executed (e.g. ionization)
- STAGE 5: “post step” phase of the process that limited the step is executed New tracks are “pushed” to the stack
- STAGE 6: If E_{kin}=0 all “at rest” processes are executed; if particle is stable the track is killed. Else:
- STAGE 7: A new step starts and sequence repeats...

Processes return a “true path length”. The multiple scattering “virtually folds up” this true path length into a shorter ”geometrical” path length. Based on this new length, the transportation can geometrically limits the step.





Example processes

- Discrete process: Compton Scattering, hadronic inelastic, ...
  - step determined by cross section, interaction at end of step
    - PostStepGPIL(), PostStepDoIt()
- Continuous process: Cerenkov effect
  - photons created along step, roughly proportional to step length
    - AlongStepGPIL(), AlongStepDoIt()
- At rest process: mu- capture at rest
  - interaction at rest
    - AtRestGPIL(), AtRestDoIt()

pure

- Rest + discrete: positron annihilation, decay, ...
  - both in flight and at rest
- Continuous + discrete: ionization
  - energy loss is continuous
  - knock-on electrons (δ-ray) are discrete

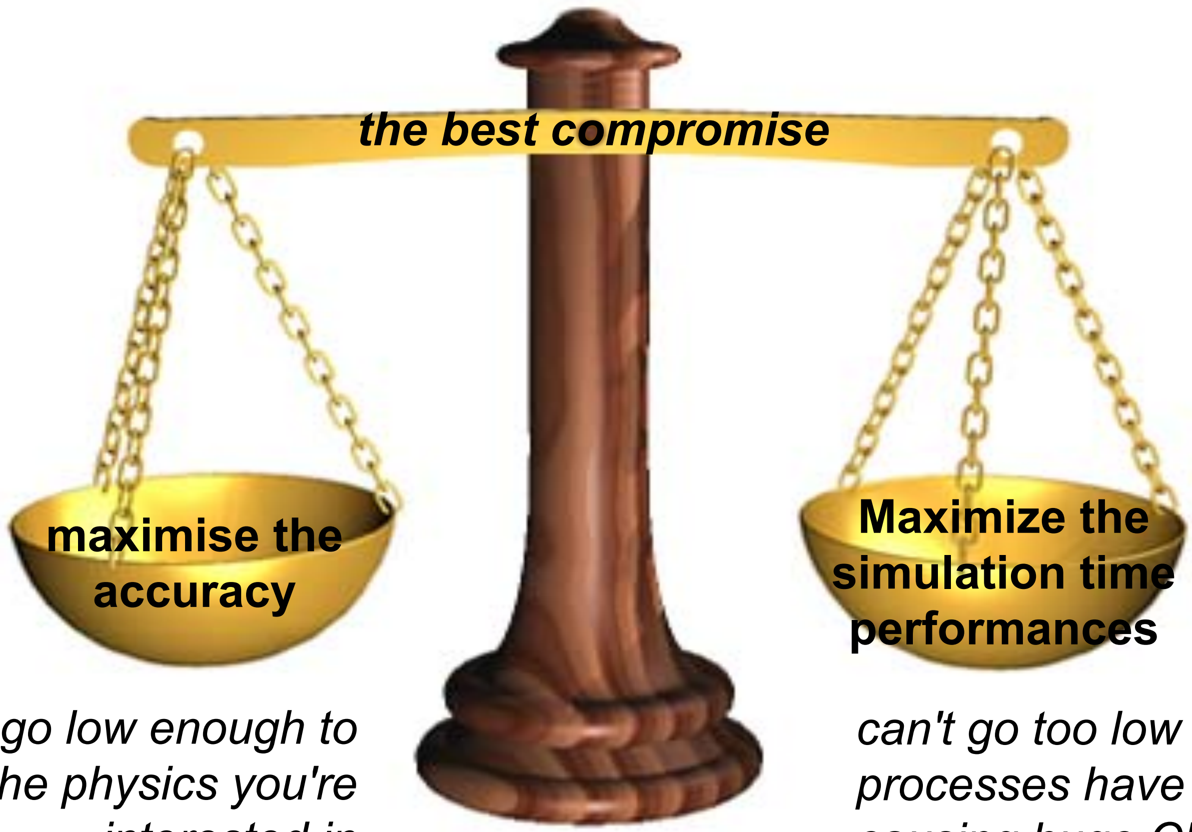
combined

Production thresholds: cut

Each simulation developer must answer the question:
how low can you go?

– should I produce (and track) everything or consider thresholds?

This is a balancing act:



need to go low enough to get the physics you're interested in

can't go too low because some processes have infrared divergence causing huge CPU time

Production thresholds: cut

- The traditional Monte Carlo solution is to impose an absolute cutoff in energy:
  - particles are stopped when this energy is reached
  - remaining energy is dumped at that point
- But, such a cut may cause **imprecise stopping location** and deposition of energy
- There is also a **particle dependence**
  - range of 10 keV γ in Si is different from range of 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 deposition measurement

Production thresholds: cut

- In Geant4 there are <u>no tracking cuts</u>
  - *<u>particles are tracked down to a zero range/kinetic energy</u>*

- Only **<u>production cuts</u>** exist
  - i.e. cuts allowing a particle to be born or not
  - Applied to: ***gamma***, ***electron***, ***positron***, ***proton***

- *Why are production cuts needed ?*

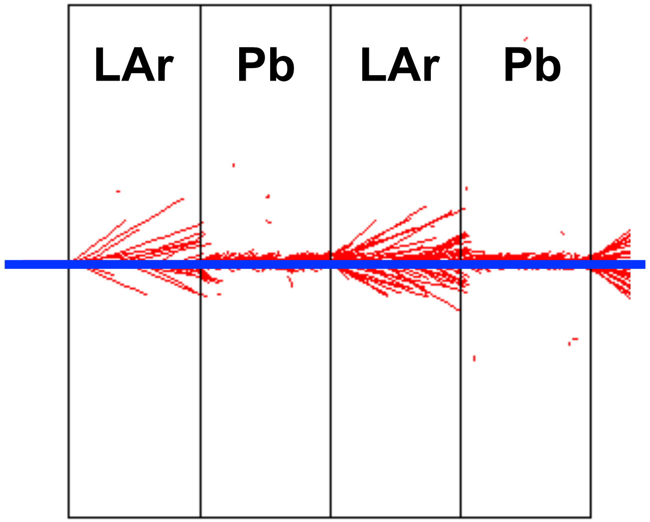
  Some electromagnetic processes involve infrared divergences
  - this leads to a huge number of smaller and smaller energy photons/electrons (such as in Bremsstrahlung, d-ray production)
  - production cuts limit this production to particles above the threshold
  - the remaining, divergent part is treated as a continuous effect (i.e. AlongStep action)

Production thresholds: cut

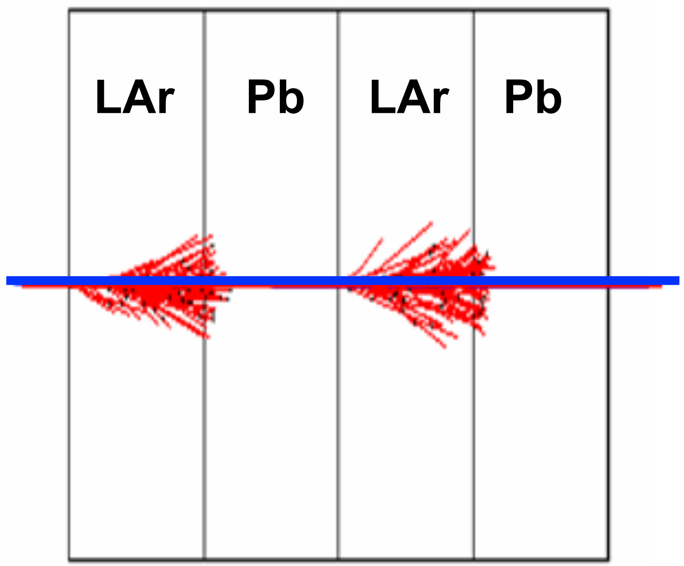
- Geant4 solution: impose a “range” production threshold
  - this threshold is a distance, not an energy
  - default = 1 mm
  - the primary particle loses energy by producing secondary electrons or gammas
  - if primary no longer has enough energy to produce secondaries which travel at least 1mm, two things happen:
    - discrete energy loss ceases (no more secondaries produced)
    - the primary is tracked down to zero energy using continuous energy loss
- Stopping location is therefore correct
- Only one value of production threshold distance is needed for all materials because it corresponds to different energies depending on material.

Production thresholds: cut

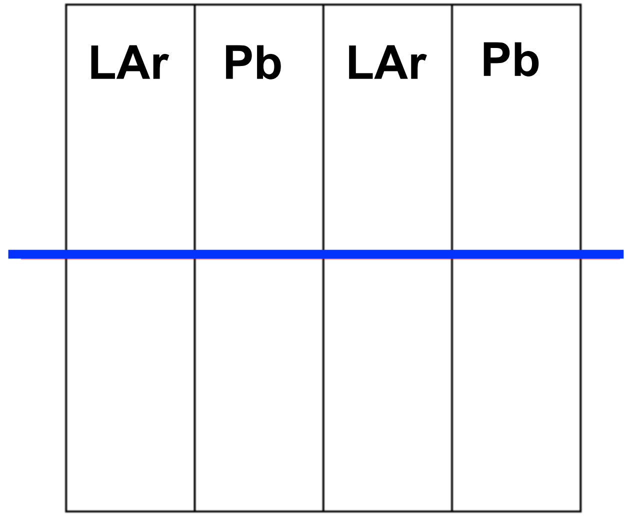
500 MeV p in
LAr-Pb sampling
calorimeter



Cut = 455 keV
(range in LAr = 1.5 mm)



Production range =1.5 mm



Cut = 2 MeV

Threshold in range: 1.5 mm

455 keV electron energy in liquid Ar

2 MeV electron energy in Pb

Cuts per region

- In a complex detector there may be many different types of sub-detectors involving
  - finely segmented volumes
  - very sensitive materials
  - large, undivided volumes
  - inert materials
- The same value of the secondary production threshold may not be appropriate for all of these
  - user must define regions of similar sensitivity and granularity and assign a different set of production thresholds (cuts) for each
- Warning: this feature is for users who are
  - simulating complex detectors
  - experienced at simulating EM showers in matter

Philosophy of physics definition

Philosophy of physics definition

- Provide a **general model framework** that allows the implementation of **complementary/alternative models** to **describe the same process** (e.g. Compton scattering)
  - A certain model could work better in a certain energy range
- **Decouple** modeling of cross sections and of final state generation
- Provide **processes** containing
  - Many possible models and cross sections
  - Default cross sections for each model

Models under continuous development

Physics definition

- Different ways to implement the physics models

1. Explicitly associating a given model to a given particle for a given energy range
   - Error prone
   - Done at code level (requires C++ coding)

2. Use of **BUILDER** and **REFERENCE PHYSICS LISTS**
   - The BUILDERS are process-related (standard, lowenergy, Bertini, etc.)
     - **Building blocks** to be used in a physics list
     - Allows **mix-and-match** done by the user
   - THE REF PHYSICS LISTS are complete physics lists
     - Can be instantiated by UI (macro files)

Builder with the G4VModularPhysicsList

- It is used to build a realistic physics list which would be too long and complicated with the previous approach
- It is derived from **`G4VUserPhysicsList`**
- **`AddTransportation()`** automatically called
- Allows the definition of **"physics modules"** for a given process
  - Electromagnetic, Hadronic, Decay, Optical physics, Ion physics

```
void myList::ConstructProcess()
{
  AddTransportation();
  //Em physics
  G4VPhysicsConstructor* emList = new G4EmStandardPhysics();
  emList->ConstructProcess();
  //Inelastic physics for protons
  G4VPhysicsConstructor* pList = new G4QGSPProtonBuilder();
  pList->ConstructProcess();
```

Reference physics lists

- Provide a complete and realistic physics with ALL models of interest
- Provided according to some use-cases
- Few choices are available for EM physics
- Several possibilities for hadronic
- They are intended as starting point and their builders can be reused
  - They are made up of builders, so easy to change/replace each given block

How to use a Geant4 physics list

- In your main(), just register an instance of the physics list to the **G4RunManager**

```
#include "QGSP_BERT.hh"
int main()
{
  // Run manager
  G4RunManager * runManager = new G4RunManager();

  …
 G4VUserPhysicsList* physics = new QGSP_BERT();
  runManager-> SetUserInitialization(physics);
}
```

The complete lists of Reference Physics List

……./**source/physics_lists/lists**

```
-rw-r--r--  1 cirrone  staff   4102 16 Aug 09:14 QGSP_BERT_EMV.icc
-rw-r--r--  1 cirrone  staff   2564 11 May  2009 QGSP_BERT_EMX.hh
-rw-r--r--  1 cirrone  staff   4232 16 Aug 09:14 QGSP_BERT_EMX.icc
-rw-r--r--  1 cirrone  staff   2542 31 Oct  2006 QGSP_BERT_HP.hh
-rw-r--r--  1 cirrone  staff   4322 16 Aug 09:14 QGSP_BERT_HP.icc
-rw-r--r--  1 cirrone  staff   2586 17 Oct  2008 QGSP_BERT_NOLEP.hh
-rw-r--r--  1 cirrone  staff   4224 16 Aug 09:14 QGSP_BERT_NOLEP.icc
-rw-r--r--  1 cirrone  staff   2580 26 Apr  2007 QGSP_BERT_NQE.hh
-rw-r--r--  1 cirrone  staff   4240 16 Aug 09:14 QGSP_BERT_NQE.icc
-rw-r--r--  1 cirrone  staff   2557  7 May  2007 QGSP_BERT_TRV.hh
-rw-r--r--  1 cirrone  staff   4236 16 Aug 09:14 QGSP_BERT_TRV.icc
-rw-r--r--  1 cirrone  staff   2496 31 Oct  2006 QGSP_BIC.hh
-rw-r--r--  1 cirrone  staff   4578 16 Aug 09:14 QGSP_BIC.icc
-rw-r--r--  1 cirrone  staff   2552 11 May  2009 QGSP_BIC_EMY.hh
-rw-r--r--  1 cirrone  staff   4176 16 Aug 09:14 QGSP_BIC_EMY.icc
-rw-r--r--  1 cirrone  staff   2550 24 Nov  2006 QGSP_BIC_HP.hh
-rw-r--r--  1 cirrone  staff   4140 16 Aug 09:14 QGSP_BIC_HP.icc
-rw-r--r--  1 cirrone  staff   2563 13 Nov  2007 QGSP_DIF.hh
-rw-r--r--  1 cirrone  staff   4317 16 Aug 09:14 QGSP_DIF.icc
-rw-r--r--  1 cirrone  staff   2502 31 Oct  2006 QGSP_EMV.hh
-rw-r--r--  1 cirrone  staff   4822 16 Aug 09:14 QGSP_EMV.icc
-rw-r--r--  1 cirrone  staff   2541 26 Apr  2007 QGSP_EMV_NQE.hh
-rw-r--r--  1 cirrone  staff   4260 16 Aug 09:14 QGSP_EMV_NQE.icc
-rw-r--r--  1 cirrone  staff   2582 23 Apr  2009 QGSP_FTFP_BERT.hh
-rw-r--r--  1 cirrone  staff   4174 16 Aug 09:14 QGSP_FTFP_BERT.icc
-rw-r--r--  1 cirrone  staff   3499 19 Jul  2009 QGSP_INCL_ABLA.hh
-rw-r--r--  1 cirrone  staff   4262 16 Aug 09:14 QGSP_INCL_ABLA.icc
-rw-r--r--  1 cirrone  staff   2528 26 Apr  2007 QGSP_NQE.hh
-rw-r--r--  1 cirrone  staff   4234 16 Aug 09:14 QGSP_NQE.icc
-rw-r--r--  1 cirrone  staff   2523 28 Nov  2006 QGSP_QEL.hh
-rw-r--r--  1 cirrone  staff   4413 16 Aug 09:14 QGSP_QEL.icc
-rw-r--r--  1 cirrone  staff   2507 13 Nov  2007 QGS_BIC.hh
-rw-r--r--  1 cirrone  staff   4188 16 Aug 09:14 QGS_BIC.icc
-rw-r--r--  1 cirrone  staff   2521  8 Jun 18:05 Shielding.hh
-rw-r--r--  1 cirrone  staff   4113 16 Aug 09:14 Shielding.icc
-rw-r--r--  1 cirrone  staff   3710 31 Oct  2006 SpecialCuts.hh
lists Lavora! > _
```

Electromagnetic physics

EM concept - 1

- The same physics processes (e.g. Compton scattering) can be described by different models, that can be **alternative** or **complementary** in a given energy range
- For instance: Compton scattering can be described by
  - **`G4KleinNishinaCompton`**
  - **`G4LivermoreComptonModel`** (specialized low-energy, based on the Livermore database)
  - **`G4PenelopeComptonModel`** (specialized low-energy, based on the Penelope analytical model)
  - **`G4LivermorePolarizedComptonModel`** (specialized low-energy, Livermore database with polarization)
  - **`G4PolarizedComptonModel`** (Klein-Nishina with polarization)
- Different models can be combined, so that the appropriate one is used in each given energy range (→ performance optimization)

EM concept - 2

- A physical interaction or process is described by a process class
  - Naming scheme : « G4ProcessName »
  - Eg. : « G4Compton » for photon Compton scattering

- A physical process can be simulated according to several models, each model being described by a model class
  - The usual naming scheme is: « G4ModelNameProcessNameModel »
  - Eg. : « G4LivermoreComptonModel » for the Livermore Compton model
  - Models can be alternative and/or complementary on certain energy ranges
  - Refer to the Geant4 manual for the full list of available models

Packages overview

- Models and processes for the description of the EM interactions in Geant4 have been grouped in several packages

Package	Description
Standard	γ-rays, $e^{\pm}$ up to 100 TeV, Hadrons, ions up to 100 TeV
Muons	Muons up to 1 PeV
X-rays	X-rays and optical photon production
Optical	Optical photons interactions
High-Energy	Processes at high energy (> 10 GeV). Physics for exotic particles
Low-Energy	Specialized processes for low-energy (down to 250 eV), including atomic effects
Polarization	Simulation of polarized beams

When/why to use Low Energy Models

- **Use** Low-Energy models (Livermore or Penlope), 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

Example: PhysicsList, γ-rays

Only PostStep

```
G4ProcessManager* pmanager =
   G4Gamma::GetProcessManager();
pmanager->AddDiscreteProcess(new G4PhotoElectricEffect);
pmanager->AddDiscreteProcess(new G4ComptonScattering);
pmanager->AddDiscreteProcess(new G4GammaConversion);
pmanager->AddDiscreteProcess(new G4RayleighScattering);
```

- Use **`AddDiscreteProcess`** because γ-rays processes have **only PostStep actions**
- For each process, the **default model** is used among all the available ones (e.g. **`G4KleinNishinaCompton`** for **`G4ComptonScattering`**)

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
G4EmDNAPhysics

Combined Physics
Standard > 1 GeV
LowEnergy < 1 GeV

- **`$G4INSTALL/source/physics_list/builders`**
- Advantage of using of these classes – they are tested on regular basis and are used for regular validation

Hadronic physics

Hadronic physics

- Data-driven models
- Parametrised models
- Theory-driven models

Reference physics lists for Hadronic interactions

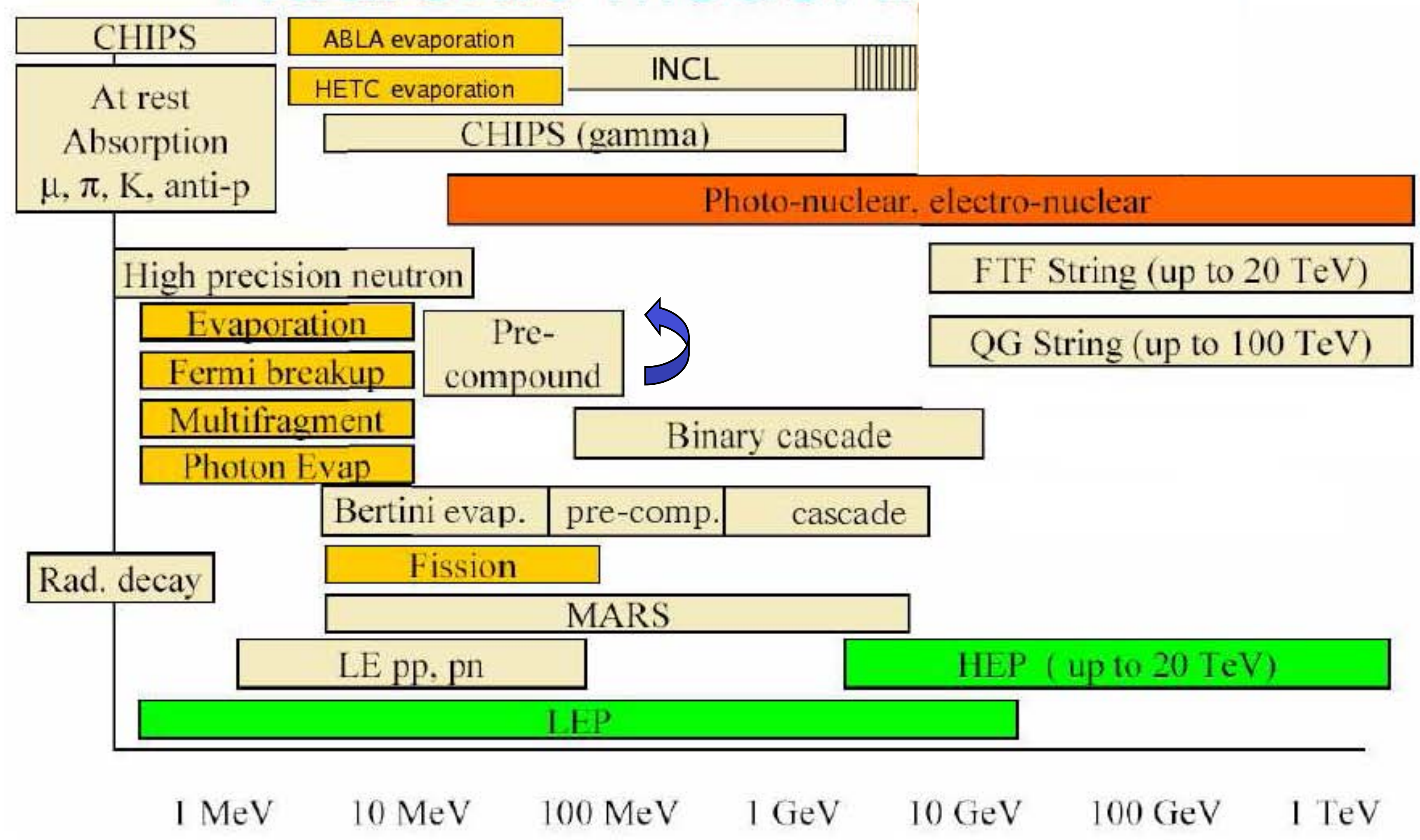
- Are part of the Geant4 code
- Four families of lists
  - **LHEP**, parameterised modelling of hadronic interactions
    - Based on the old GEISHA package
  - **QGS**, or list based on a model that use the Quark Gluon String model for high energy hadronic interactions of protons, neutrons, pions and kaons
  - **FTF**, based on the FTF (FRITIOF like string model) for protons, neutrons, pions and kaons
  - **Other** specialized physics lists

Cross sections

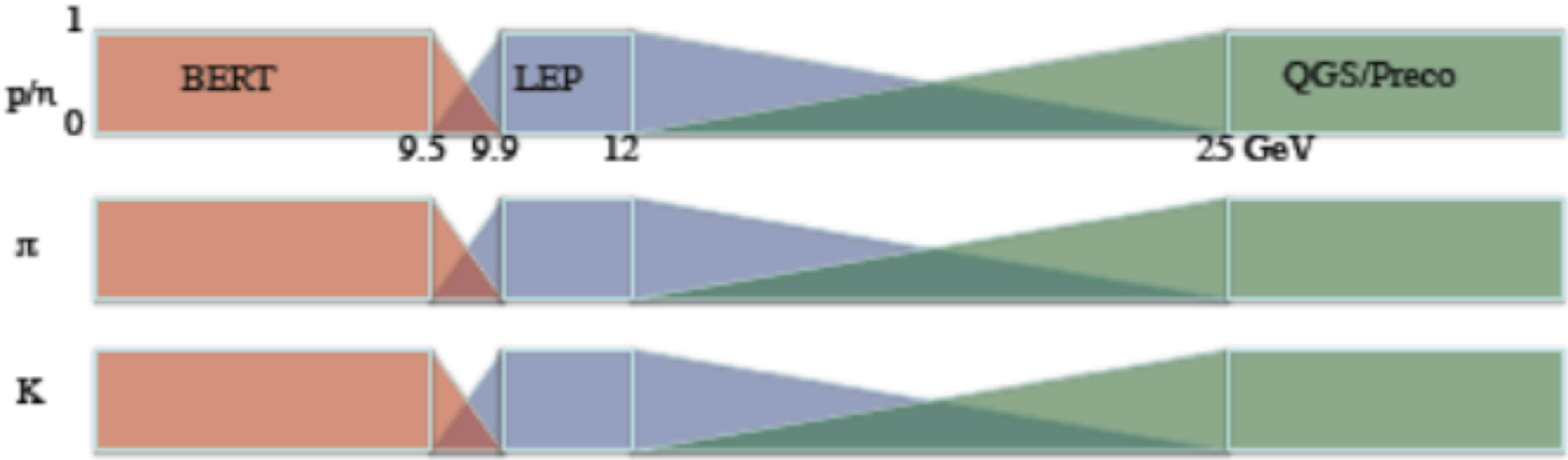
- Default cross section sets are provided for each type of hadronic process:
  - Fission, capture, elastic, inelastic
- Can be **overridden** or **completely replaced**
- Different types of cross section sets:
  - Some contain only a few numbers to parameterize cross section
  - Some represent large databases (data driven models)
- Cross section management
  - `GetCrossSection()` → sees last set loaded for energy range

Hadronic model inventory

http://geant4.cern.ch/support/proc_mod_catalog/models



Hadronic models match – inelastic interactions



Recommended reference physics lists

- A dedicated web page

Info to help users to choose the proper physics list:

`http://geant4.cern.ch/support/proc_mod_catalog/physics_lists/physicsLists.shtml`

- Application fields are identified
  - High energy physics
  - LHC neutron fluxes
  - Shielding
  - Medical
  - …

Where to find information?

User Support

1. Getting started
2. Training courses and materials
3. Source code
    a. Download page
    b. LXR code browser -or- draft doxygen documentation
4. Frequently Asked Questions (FAQ)
5. Bug reports and fixes
6. User requirements tracker
7. User Forum
8. Documentation
    a. Introduction to Geant4
    b. Installation Guide
    c. Application Developers Guide
    d. Toolkit Developers Guide
    e. Physics Reference Manual
    f. Software Reference Manual
9. Physics lists
    a. Electromagnetic
    b. Hadronic

Code Example (1/2)

```
G4ParticleDefinition* proton=
      G4Proton::ProtonDefinition();
G4ProcessManager* protonProcessManager =
      proton->GetProcessManager();
```

retrieve the process manager for proton

```
// Elastic scattering
G4HadronElasticProcess* protonElasticProcess =
 new G4HadronElasticProcess();
```

*create the **process** for elastic scattering*

```
G4LElastic* protonElasticModel =
 new G4LElastic();
```

*get the LE parametrized **model** for elastic scattering*

```
protonElasticProcess->
 RegisterMe(protonElasticModel);
```

***register** the model to the process*

```
protonProcessManager->
 AddDiscreteProcess(protonElasticProcess);
```

attach the process to proton

Code example (2/2)

```
...
// Inelastic scattering
G4ProtonInelasticProcess* protonInelasticProcess
  = new G4ProtonInelasticProcess();
```

*creates the **process** for inelastic scattering*

Model 1

```
G4LEProtonInelastic* protonLEInelasticModel
 = new G4LEProtonInelastic();
protonLEInelasticModel->
   SetMaxEnergy(20.0*GeV);
```

*gets the LEP **model** up to 20 GeV*

```
protonInelasticProcess->
   RegisterMe(protonLEInelasticModel);
```

registers LEP model to the process

Model 2

```
G4HEProtonInelastic* protonHEInelasticModel =
   new G4HEProtonInelastic();
protonHEInelasticModel->SetMinEnergy(20.0*GeV);
protonInelasticProcess
   ->RegisterMe(protonHEInelasticModel);
```

*gets the HEP **model** from 20 GeV*

registers HEP model to the process

Quick overview of validation

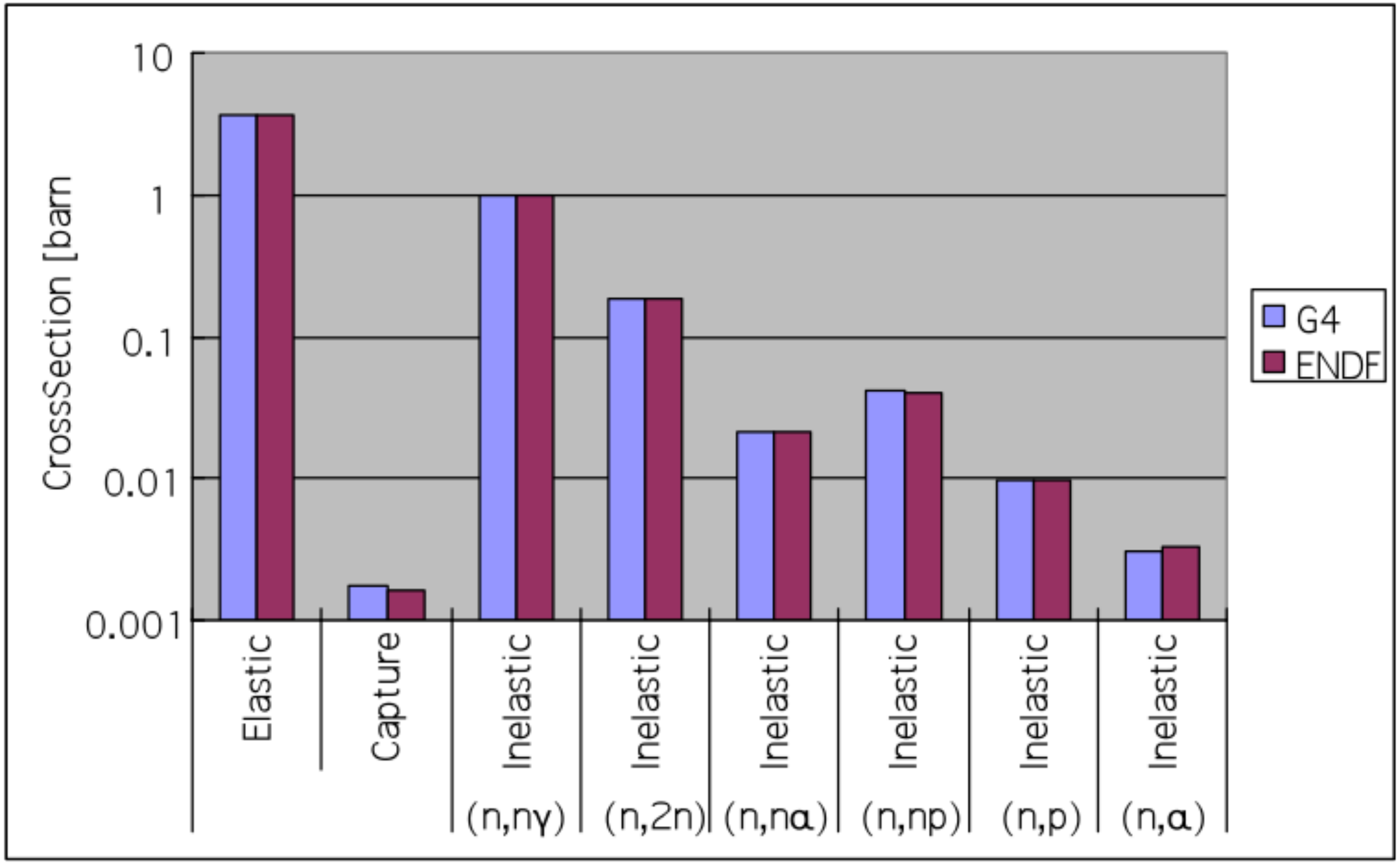
Hadronic validation

- A website is available to collect relevant information for validation of Geant4 hadronic models (plots, tables, references to data and to models, etc.)

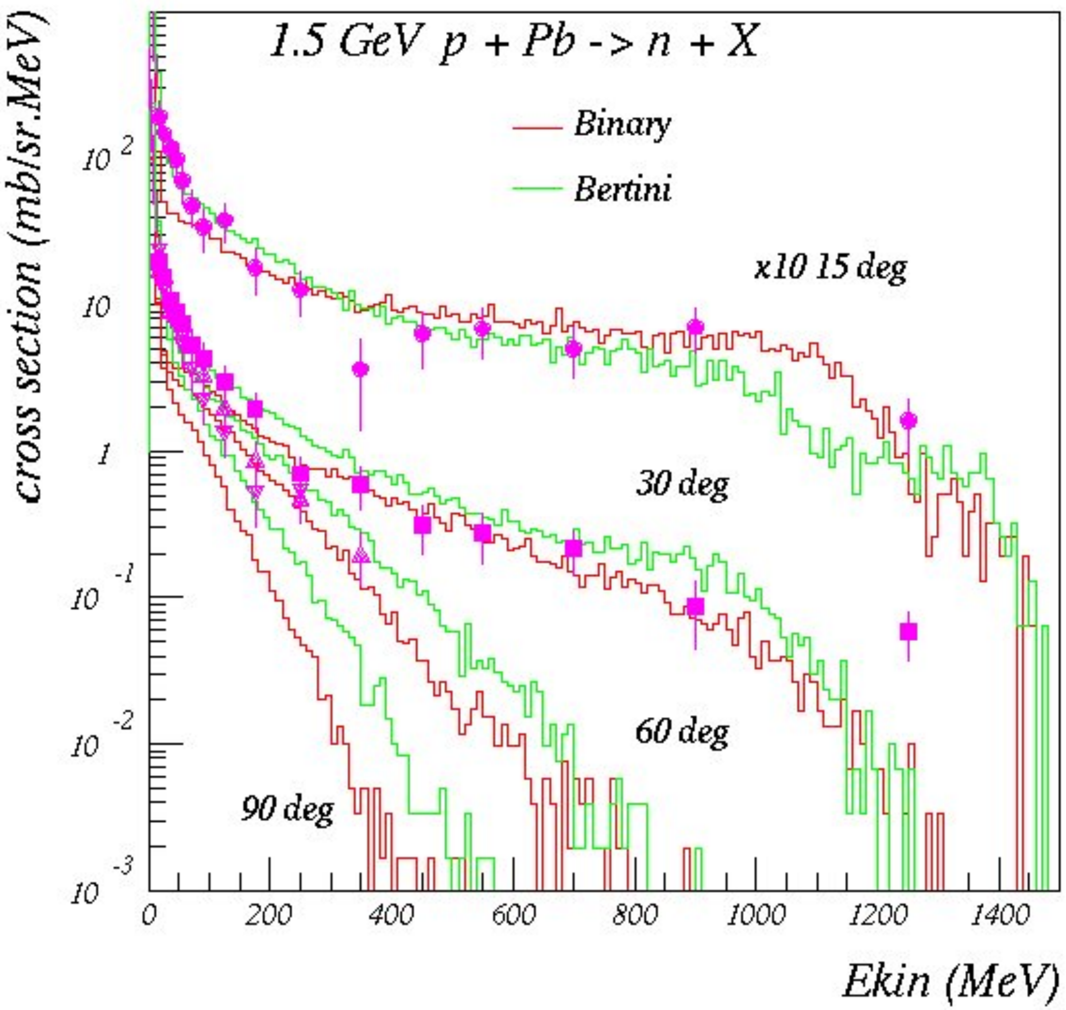
  `http://geant4.fnal.gov/hadronic_validation/validation_plots.htm`

- Several physics lists and several use-cases have been considered (e.g. thick target, stopped particles, low-energy)
- Includes final states and cross sections

Some verification: channel cross section

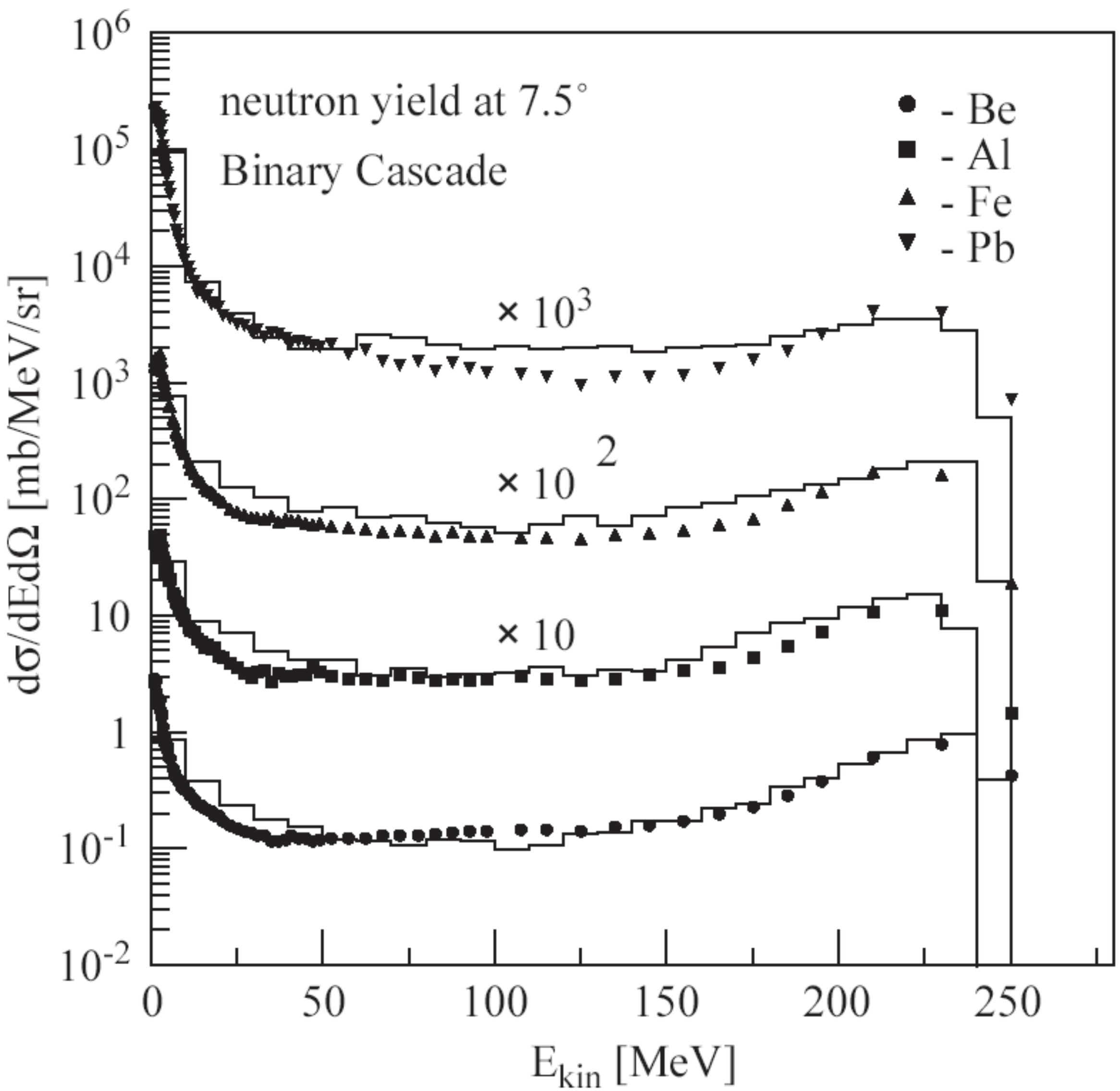


Nuclear fragmentation



Bertini and **Binary cascade** models: neutron production vs. angle from 1.5 GeV protons on Lead

Neutron production by protons



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

Thanks for your attention