



Physics in Geant4



Luciano Pandola

INFN – Laboratori Nazionali del Sud



Physics lists & Co.



User Classes

Initialisation classes

Invoked at the initialization

- G4VUserDetectorConstruction
- G4VUserPhysicsList

Global: **only one instance** of them exists in memory, shared by all threads (**readonly**). Managed only by the **master** thread.

Action classes

Invoked during the execution loop

- G4VUserActionInitialization
 - G4VUserPrimaryGeneratorAction
 - G4UserRunAction (*)
 - G4UserEventAction
 - G4UserTrackingAction
 - G4UserStackingAction
 - G4UserSteppingAction

Local: an **instance** of each action class exists **for each thread**.
(*) Two RunAction's allowed: one for master and one for threads



G4VUserPhysicsList

- All **physics lists** **must** derive from this class
 - And then be **registered** to the G4(MT)RunManager
 - **Mandatory** class in Geant4

```
class MyPhysicsList: public G4VUserPhysicsList {  
public:  
    MyPhysicsList();  
    ~MyPhysicsList();  
    void ConstructParticle();  
    void ConstructProcess();  
    void SetCuts();  
}
```

- User must **implement** the following (purely virtual) **methods**:
 - `ConstructParticle()`, `ConstructProcess()`
- **Optional Virtual method**:
 - `SetCuts()` (used to be purely virtual up to 10.2)



ConstructParticle()

- Choose the **particles** you need in your simulation and **define all** of them here
 - `G4Electron::ElectronDefinition()`
 - `G4Gamma::GammaDefinition()`
 - ...
- It is possible use **Geant4 classes** that **create categories** of particles
 - `G4BosonConstructor()`
 - `G4LeptonConstructor()`
 - ...
- Special particles: **geantino** and **charged geantino**
 - They are only **transported** in the geometry (**no physical interaction**)
 - The charged geantino also feels the **EM fields**



Particles in Geant4

- Geant4 "knows" **all PDG particles**
 - And also **optical** photons
- Particle properties stored as **G4ParticleDefinition** objects (singletons)
 - Mass, lifetime, width, decay tables
 - Spin, isospin, parity
 - Quark content, etc.
- **Short-lived** particles ($\tau < 10^{-14}$ s) are **not transported** by Geant4
 - They **immediately decay**, according to their own decay scheme



SetCuts()

- Define all **production** cuts for **gamma**, **electrons** and **positrons**
 - Recently also for **neutrons**
- Notice: this is a **production cut**, not a tracking cut

MORE ON THIS LATER



The definition of physics - 1

- At the beginning of Geant4 the philosophy was: "the **user is in charge** for deciding and implemented the most suitable models for his/her own application"
 - Completely **transparent** physics (no black box!)
 - **Complicated** to know and assess the validity of many models
- Long "**flat**" physics lists:
 - Explicitly associating a **given model** to a given **particle** for a given **energy range**
 - Done at code level (requires C++ coding)
- Still a **possibility**
 - Provided you know what you are doing



The definition of physics - 2

- **Modular** physics lists: the list is built from **basic "blocks"** (constructors)
 - The constructors are **process-related** (standard, lowenergy, Bertini, etc.)
 - Allows **mix-and-match** done by the user
 - Some constructors **provided by Geant4**, but users can create and register their own **customized**
- Class derives from **G4VModularPhysicsList** which inherits from **G4VUserPhysicsList**
 - **SetCuts ()** is the only **mandatory** virtual method
 - But one can use the default from **G4VUserPhysicsList**
 - **ConstructParticle ()** and **ConstructProcess ()** are optional



Builder with the G4VModularPhysicsList

- **AddTransportation()** automatically called
- Allows the definition of “**physics modules**”
 - Electromagnetic
 - Hadronic
 - Decay, Optical physics, Ion physics
- User **customized constructors** can be created, derived class from **G4VPhysicsConstructor**
- Modules can be **registered** using the method **RegisterPhysics()**
 - Can be done at *run-time* (i.e. select physics via macro)

How to build a modular physics list - 1

- Create a class derived by **G4VModularPhysicsList**
 - `class myList : public G4VModularPhysicsList`
- Implement the **mandatory** method `SetCuts ()`
- Register the **appropriate constructors** (or **create your own**) in the constructor or in `ConstructProcess ()`
 - In the first case, you cannot change at run-time

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

How to build a modular physics list - 2

- **Other option:** instantiate the **constructors** in `ConstructProcess()` and invoke their own `ConstructProcess()`
- Constructors made out from "elementary" builders

```
void myList::ConstructProcess()  
{  
    //Em physics  
    G4VPhysicsConstructor* emList = new G4EmStandardPhysics();  
    emList->ConstructProcess();  
    //Inelastic physics for protons  
    G4VPhysicsConstructor* pList = new G4HadronPhysicsQGS_BIC();  
    pList->ConstructProcess();  
}
```

- `$G4INSTALL/source/physics_lists/constructors`



The definition of physics - 3

- Geant4 provides a **few ready-for-the-use** physics lists
 - **Complete** physics lists
 - Can be **instantiated** by **UI** (macro files)
- Provide a complete and **realistic physics** with **ALL models** of interest
- Provided according to some **use-cases**
 - **Many options** available for EM and hadronic physics
- They are intended as **starting point** and **their builders can be reused**
 - They are **made up of constructors**, so easy to change/replace each given block



Reference physics lists

- These families share **components** to attach certain types of processes to **groups of particles**. These components are:
 - **Electromagnetic** interactions for all particles
 - **Inelastic** interactions
 - **Elastic** scattering
 - **Capture**
 - **Decay** of unstable particles
 - **Specialised** treatment of low energy neutrons (< 20 MeV)
- They are **modular physics lists** by themselves, so you can register **additional** constructors (e.g. optical physics)

How to use a Geant4 physics list

- In your main(), just register an instance of the physics list to the **G4 (MT) RunManager**

```
#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

`$G4INSTALL/source/physics_lists/lists`

| | | |
|--------------------------------------|---|--------------------------------|
| <code>FTF_BIC.hh</code> | <code>G4PhysListRegistry.hh</code> | <code>QGSP_BIC_AllHP.hh</code> |
| <code>FTFP_BERT.hh</code> | <code>G4PhysListStamper.hh</code> | <code>QGSP_BIC.hh</code> |
| <code>FTFP_BERT_HP.hh</code> | <code>INCLXXPhysicsListHelper.hh</code> | <code>QGSP_BIC_HP.hh</code> |
| <code>FTFP_BERT_TRV.hh</code> | <code>LBE.hh</code> | <code>QGSP_FTFP_BERT.hh</code> |
| <code>FTFP_INCLXX.hh</code> | <code>NuBeam.hh</code> | <code>QGSP_INCLXX.hh</code> |
| <code>FTFP_INCLXX_HP.hh</code> | <code>QBBC.hh</code> | <code>QGSP_INCLXX_HP.hh</code> |
| <code>G4GenericPhysicsList.hh</code> | <code>QGS_BIC.hh</code> | <code>Shielding.hh</code> |
| <code>G4PhysListFactoryAlt.hh</code> | <code>QGSP_BERT.hh</code> | |
| <code>G4PhysListFactory.hh</code> | <code>QGSP_BERT_HP.hh</code> | |

Geant 4

[Download](#) | [User Forum](#) | [Gallery](#)
[Contact Us](#)

Search Geant4

[Home](#) > [User Support](#) > [Process/model catalog](#) > [Physics Lists](#) > [Reference Physics Lists](#)

Reference Physics Lists

A web page [recommending physics lists](#) according to the use case is under construction. The previous version of physics list web pages referring to 'are still available'.

String model based physics lists

These Physics lists apply a **string model** for the modeling of interactions of high energy hadrons, i.e. for protons, neutrons, pions and kaons above ~ (5-25) GeV depending on the exact physics list. Interactions at lower energies are handled by one of the intranuclear cascade models or the precompound model. Nuclear capture of negative particles and neutrons at rest is handled using either the Chiral Invariant Phase Space (CHIPS) model or the Bertini intranuclear cascade. Hadronic inelastic interactions use:

- a tabulation of the Barashenkov pion cross sections
- the Axen-Wellisch parameterization of the proton and neutron cross sections

The physics lists are:



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](#)

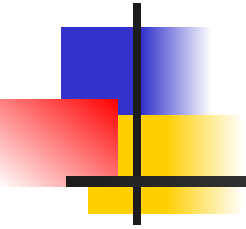




Summary – three kinds of physics lists for Geant4

- Old-style flat physics list
 - You code **what you want**, particle by particle and process by process
 - Very much flexible, but **not really encouraged**
- User-custom modular physics list
 - **Blocks** (constructors) **provided** by Geant4
 - Can register **user-custom** constructors
 - Usually the *optimal compromise* between flexibility and user-friendliness
- Ready-for-the-use Geant4 physics list
 - **Plug and play** (directly registered in the main!)
 - Can still register **extra constructors**

Physics processes and models





Philosophy

- 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 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

Inventory (and specs) of the models for γ -rays

1 MeV γ in Al

- Many models available for each process
 - Plus one full set of polarized models
- Differ for energy range, precision and CPU speed
 - Final state generators
- Different mixtures available the Geant4 EM constructors

| Model | E_{\min} | E_{\max} | CPU |
|---------------------------------|------------|------------|-----|
| G4LivermoreRayleighModel | 100 eV | 10 PeV | 1.2 |
| G4PenelopeRayleighModel | 100 eV | 10 GeV | 0.9 |
| G4KleinNishinaCompton | 100 eV | 10 TeV | 1.4 |
| G4KleinNishinaModel | 100 eV | 10 TeV | 1.9 |
| G4LivermoreComptonModel | 100 eV | 10 TeV | 2.8 |
| G4PenelopeComptonModel | 10 keV | 10 GeV | 3.6 |
| G4LowEPComptonModel | 100 eV | 20 MeV | 3.9 |
| G4BetheHeitlerModel | 1.02 MeV | 100 GeV | 2.0 |
| G4PairProductionRelModel | 10 MeV | 10 PeV | 1.9 |
| G4LivermoreGammaConversionModel | 1.02 MeV | 100 GeV | 2.1 |
| G4PenelopeGammaConversionModel | 1.02 MeV | 10 GeV | 2.2 |
| G4PEEFluoModel | 1 keV | 10 PeV | 1 |
| G4LivermorePhotoElectricModel | 10 eV | 10 PeV | 1.1 |
| G4PenelopePhotoElectricModel | 10 eV | 10 GeV | 2.9 |

Similar situation for e^{\pm}



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)
 - `G4LowEPComptonModel` (full relativistic 3D simulation)
- 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 |



EM processes for γ -rays, e^\pm

| Particle | Process | G4Process |
|----------|-----------------------------|------------------------------------|
| Photons | Gamma Conversion in e^\pm | <code>G4GammaConversion</code> |
| | Compton scattering | <code>G4ComptonScattering</code> |
| | Photoelectric effect | <code>G4PhotoElectricEffect</code> |
| | Rayleigh scattering | <code>G4RayleighScattering</code> |
| e^\pm | Ionisation | <code>G4eIonisation</code> |
| | Bremsstrahlung | <code>G4eBremsstrahlung</code> |
| | Multiple scattering | <code>G4eMultipleScattering</code> |
| e^+ | Annihilation | <code>G4eplusAnnihilation</code> |

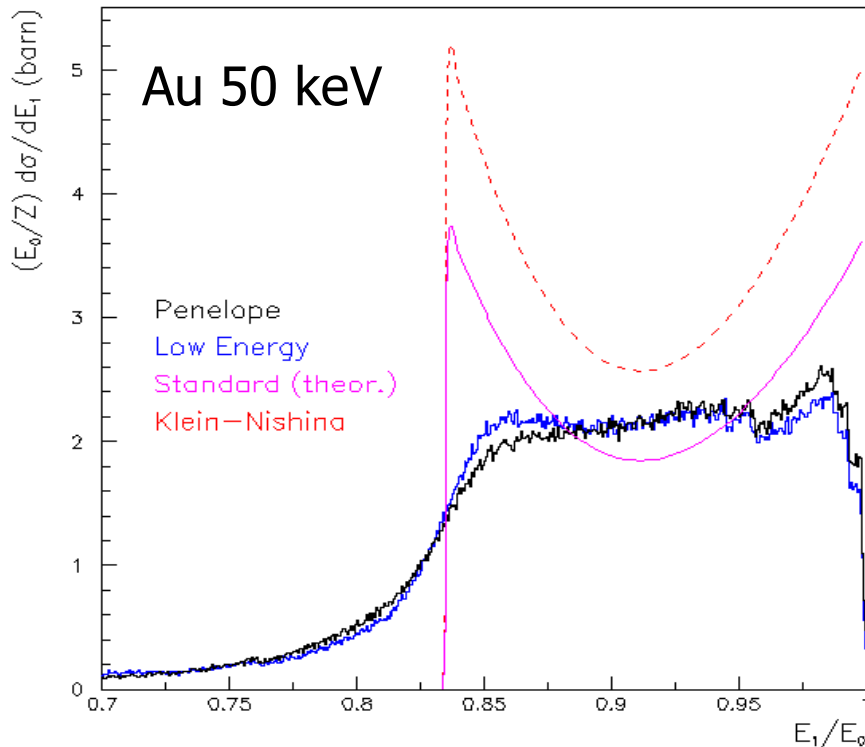


EM processes muons

| Particle | Process | G4Process |
|-----------|-------------------------|-------------------------------------|
| μ^\pm | Ionisation | <code>G4MuIonisation</code> |
| | Bremsstrahlung | <code>G4MuBremsstrahlung</code> |
| | Multiple scattering | <code>G4MuMultipleScattering</code> |
| | e^\pm pair production | <code>G4MuPairProduction</code> |

Only **one model available** for these processes (but in principle users may write *their own* models, if needed)

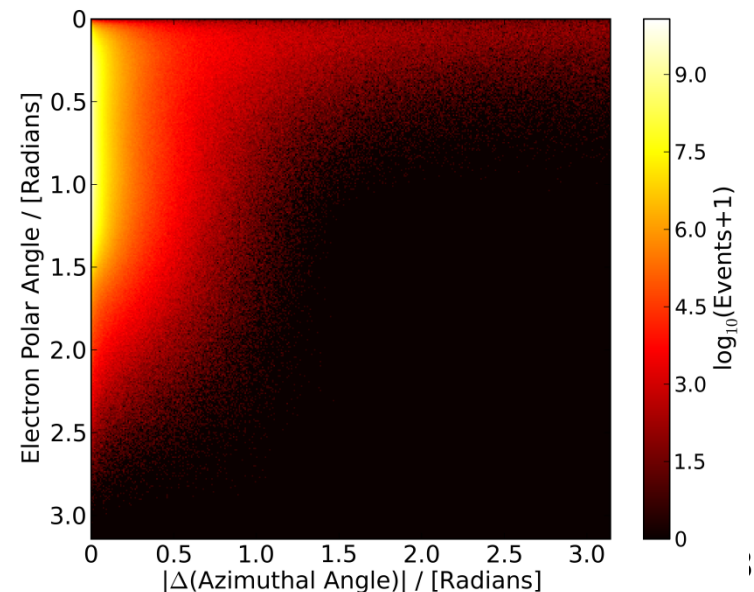
For example: Compton scattering



250 keV γ Pb

CPU time is the **price to pay** for better precision

- **New model: G4LowEPComptonModel** (Monash U.)
 - Two-body relativistic **3-dim framework**
 - Relativistic impulse approximation
 - Bound atomic electrons
 - **Electron distribution** not uniform in φ wrt **photon scattering plane**





Standard models

- Complete set of models for e^\pm , γ , ions, hadrons, μ^\pm
- Tailored to requirements from HEP applications
 - "Cheaper" in terms of CPU
 - Include high-energy corrections (e.g. LPM), assumptions made in the low-energy regime
- Theoretical or phenomenological models
 - Bethe-Bloch, corrected Klein-Nishina, ...
 - Photoabsorption Ionization (PAI)
 - ionization energy loss of a relativistic charged particle in matter
- Specific high-energy extensions available
 - Extra processes, as $\gamma \rightarrow \mu^+\mu^-$, $e^+e^- \rightarrow \mu^+\mu^-$
- Dedicated sub-library for optical photons
 - Produced by scintillation or Cherenkov effect



Livermore (& polarized) models

- Based on publicly available **evaluated data tables** from the **Livermore** data library: e^- , γ
 - EADL : Evaluated Atomic Data Library, EEDL : Evaluated Electrons Data Library, EPDL97 : Evaluated Photons Data Library, Binding energies: Scofield
 - Mixture of **experiments** and **theories**
 - In principle, tables go down to **~ 10 eV**
- Applications: medical, underground and rare events, space
- **Polarized** models
 - Same calculation of the cross section, **different** way to produce the **final state**
 - Describe in detail the kinematics of **polarized photon interactions**
 - Application: space missions for the detection of polarized photons



Penelope models

- Geant4 includes the low-energy models for electrons, positrons and photons from the **Monte Carlo code PENELOPE** (PENetration and Energy LOss of Positrons and Electrons)
 - Nucl. Instr. Meth. B 207 (2003) 107
 - Geant4 implements **v2008 of Penelope**
- Physics models **specifically developed** by the group of F. Salvat et al.
 - Great care dedicated to the **low-energy description**
 - Atomic effects, fluorescence, Doppler broadening...
- **Mixed approach**: analytical, parameterized and database-driven
 - Applicability energy range: **100 eV – 1 GeV**
- Include **positrons**
 - Not described by Livermore models

When/why to 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 for Geant4 10.2 - ready-for-the-use

- 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
- G4EmLowEPPysics
- 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 physics



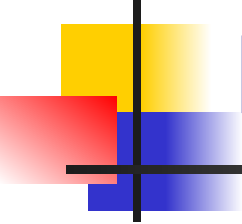
Hadronic Physics

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



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



Reference physics lists for Hadronic interactions

- **Three families** of builders
 - **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
- Up to Geant4 9.6: **LEP** and **HEP**
 - **parameterised** modelling of hadronic interactions
 - Based on the **old GEISHA** package of Geant3
 - Deprecated as **obsolete, dismissed** from version 10.0



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**



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



NeutronHP Models

- Dedicated **NeutronHP** models in Geant4 since many years
 - HP = **high-precision**
- Cross sections and final state information based on **ENDF/BVII.1 tabulated data** (G4LEND)
 - Includes **information** for *elastic* and *inelastic scattering*, *capture*, *fission* and *isotope production*
- Applicable from **thermal energies** to **20 MeV**
 - Very **precise** tracking, but also very **slow**
 - Use it with care: thermal neutron tracking is very CPU-demanding
 - A thermal neutron can have 100's of thermal scatterings before being captures
- Included in all physics lists ending with **_HP** and **Shielding**



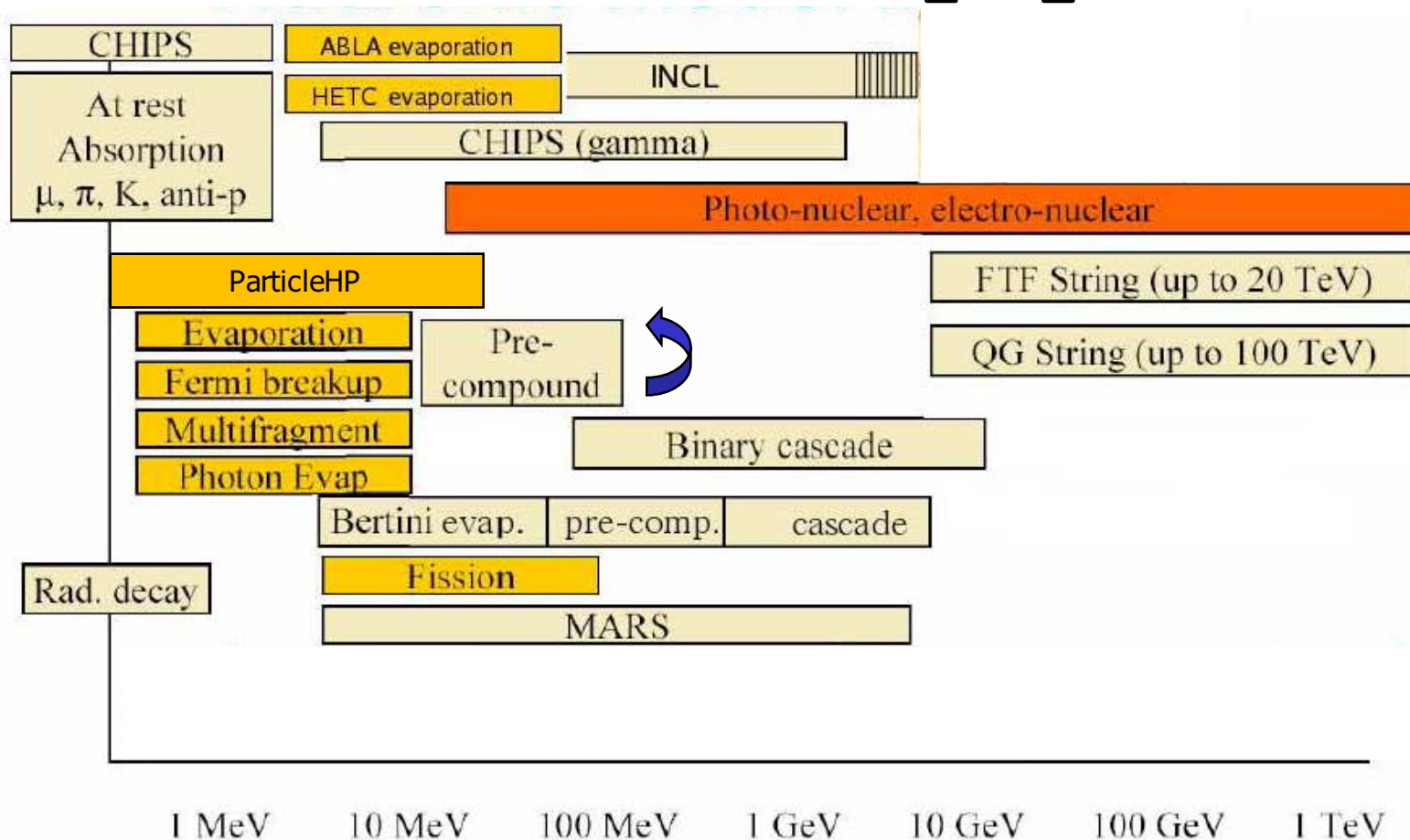
ParticleHP Models



- Since **Geant4 10.2** → **ParticleHP**
 - Data-driven approach extended to *inelastic* reactions for **p, d, t, ³He and α** (G4TENDL)
 - Data based on **TENDL-2014** and **ENDFVII.r1** (zipped format)
 - Same range of **applicability** → up to **20 MeV**
- **NeutronHP** fully **merged** with **ParticleHP** since **10.3**
 - **NeutronHP headers** are still **included the release 10.3** for backwards compatibility, but **they will be removed**
 - **G4NeutronHPXXX** → **G4ParticleHPXXX**
- Neutron models debugged since a long while, but it is a **fresh development** for the other particles
 - Use it with care, as **bugs are possible**

Hadronic model inventory

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

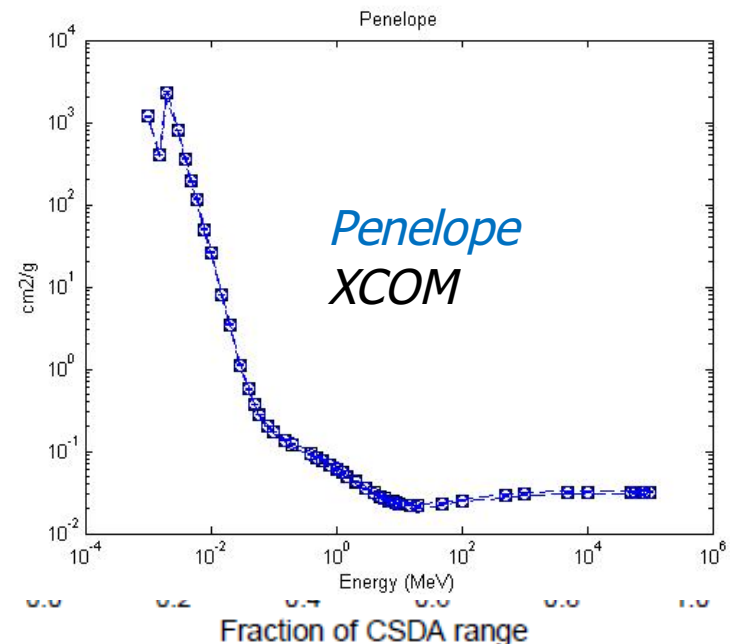
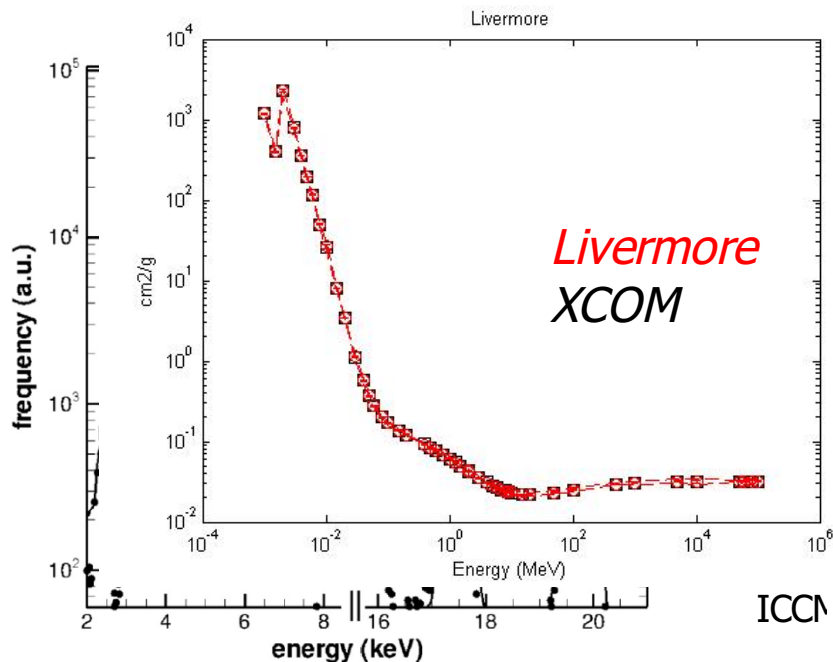




Quick overview of validation

EM validation - 1

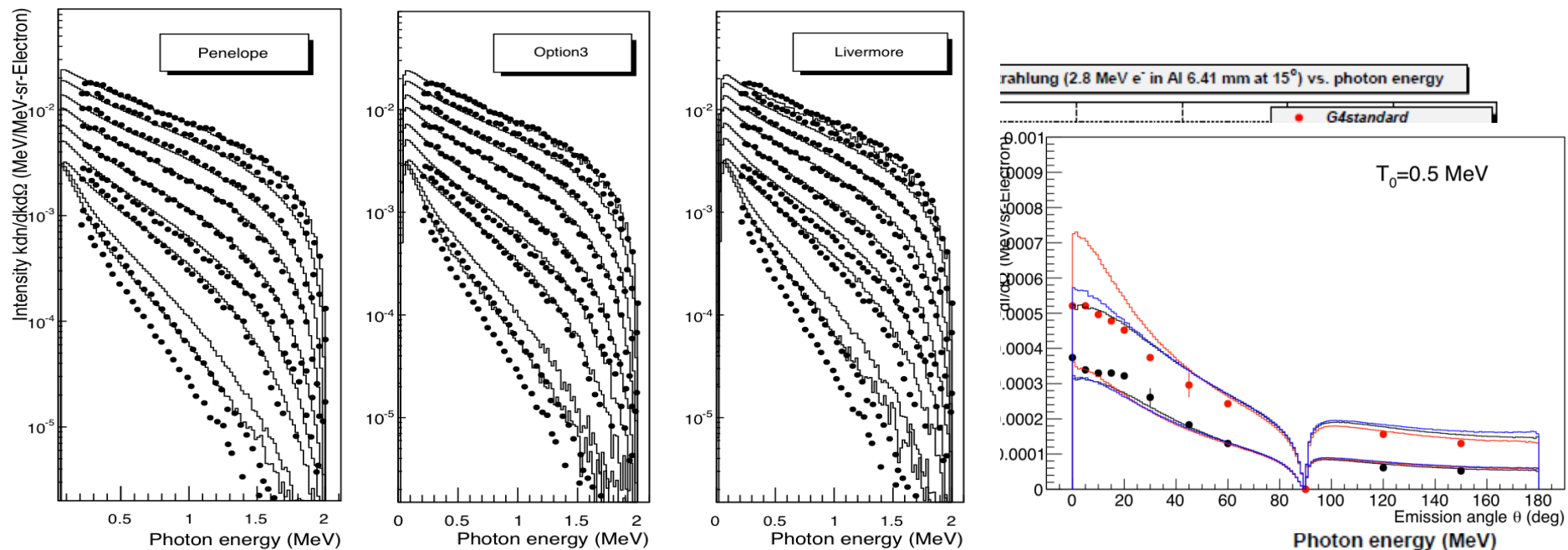
- Tens of papers and studies available
 - Geant4 Collaboration + User Community
- Results can depend on the specific observable/reference
 - Data selection and assessment critical



EM validation – 2

- In general **satisfactory agreement**
- Validation/verification **repository** available on **web**

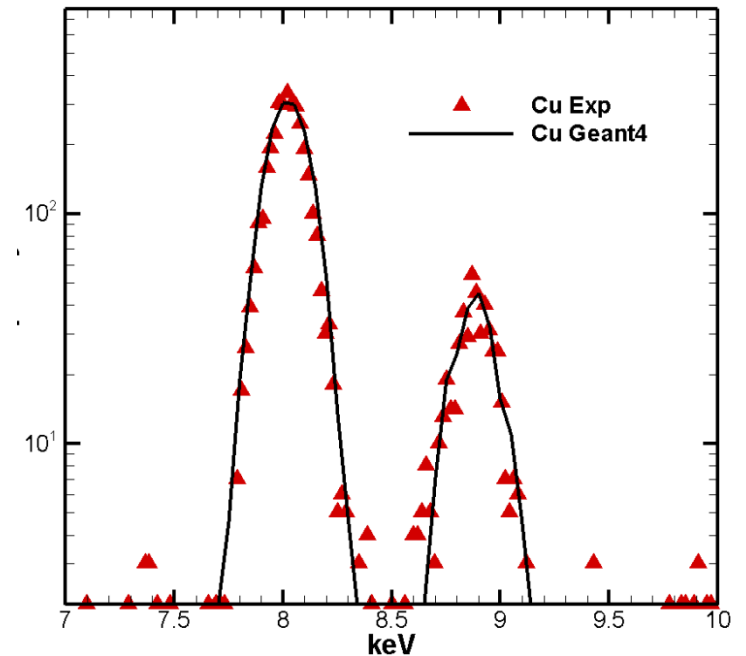
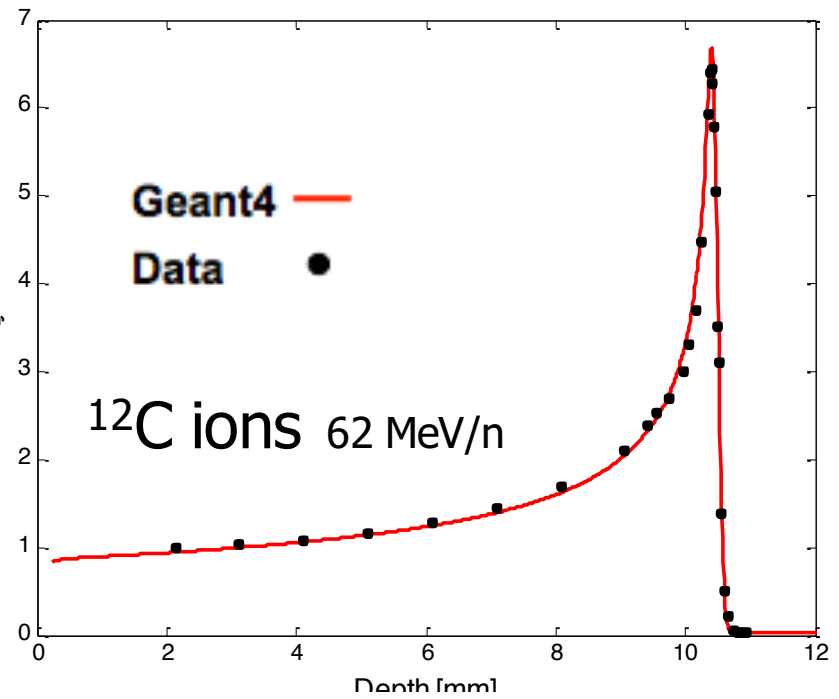
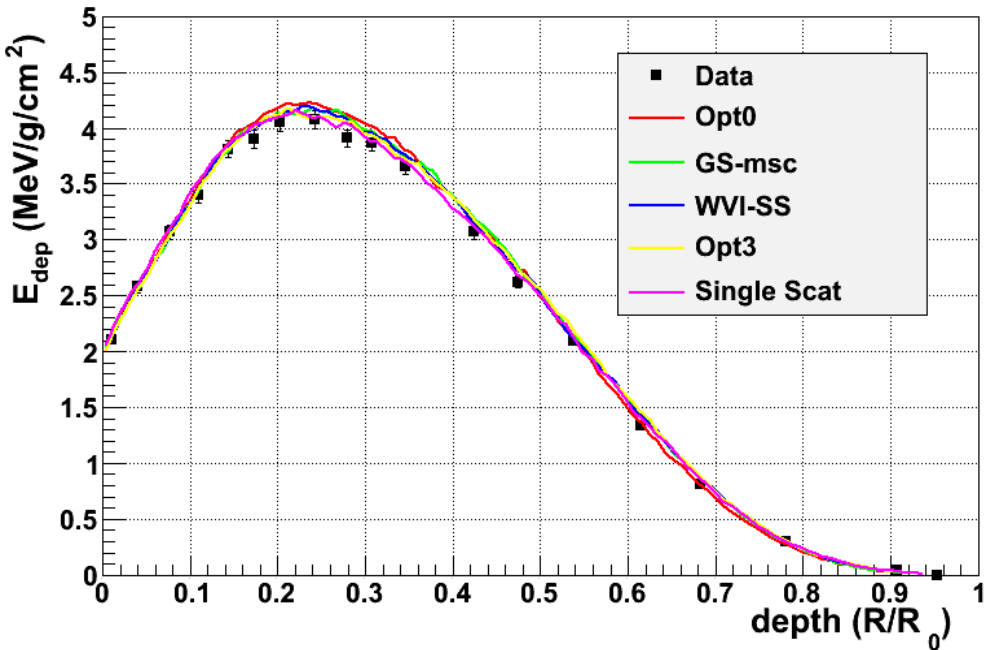
<http://cern.ch/vnivanch/verification/verification/electromagnetic/>



EM validation -

3

e⁻ showers, longitudinal profiles





Hadronic validation

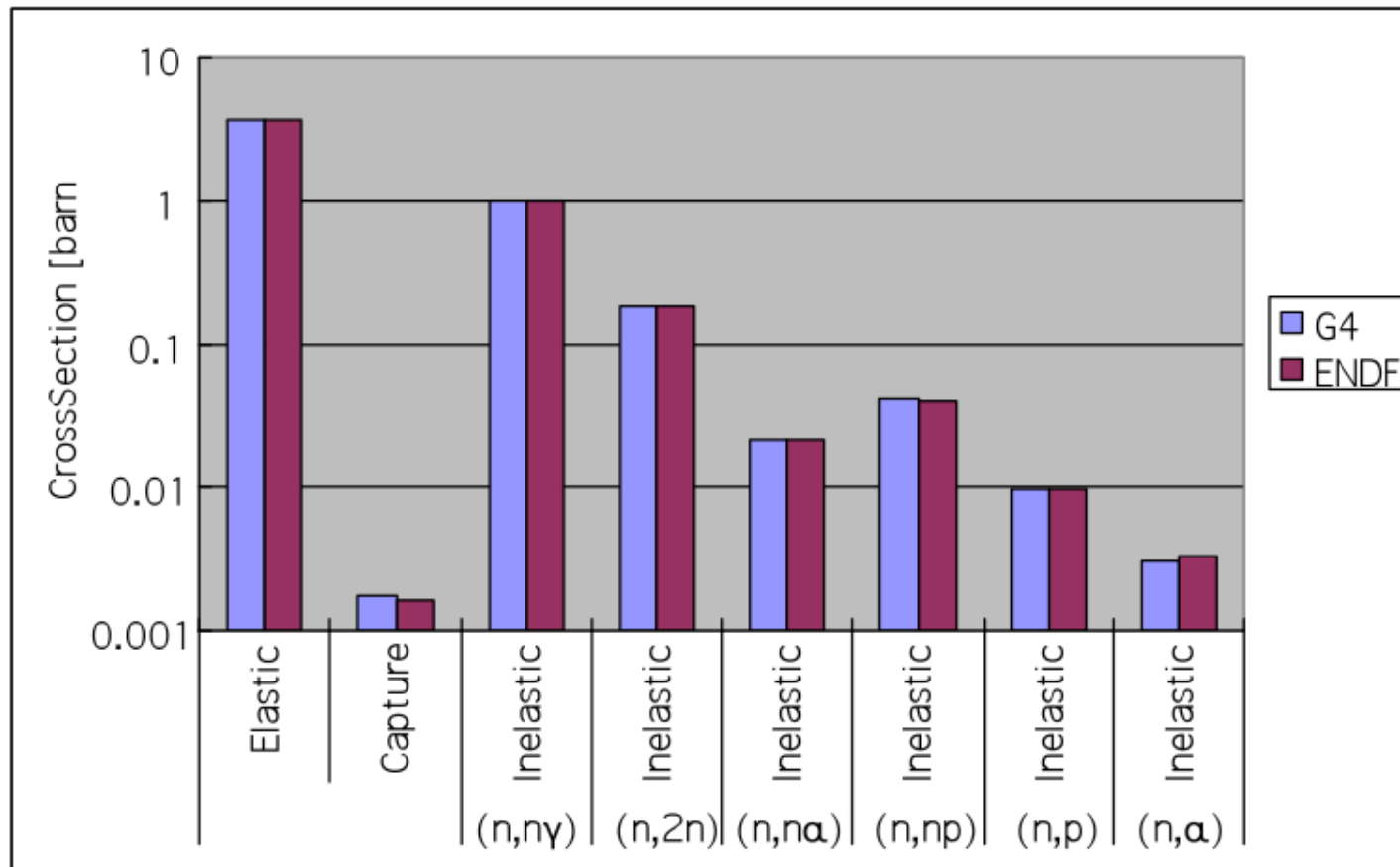
- A **website** is available to collect relevant **information** for validation of **Geant4 hadronic models** (plots, tables, references to data and to models, etc.)

`http://geant4.cern.ch/results/validation_plots.htm`

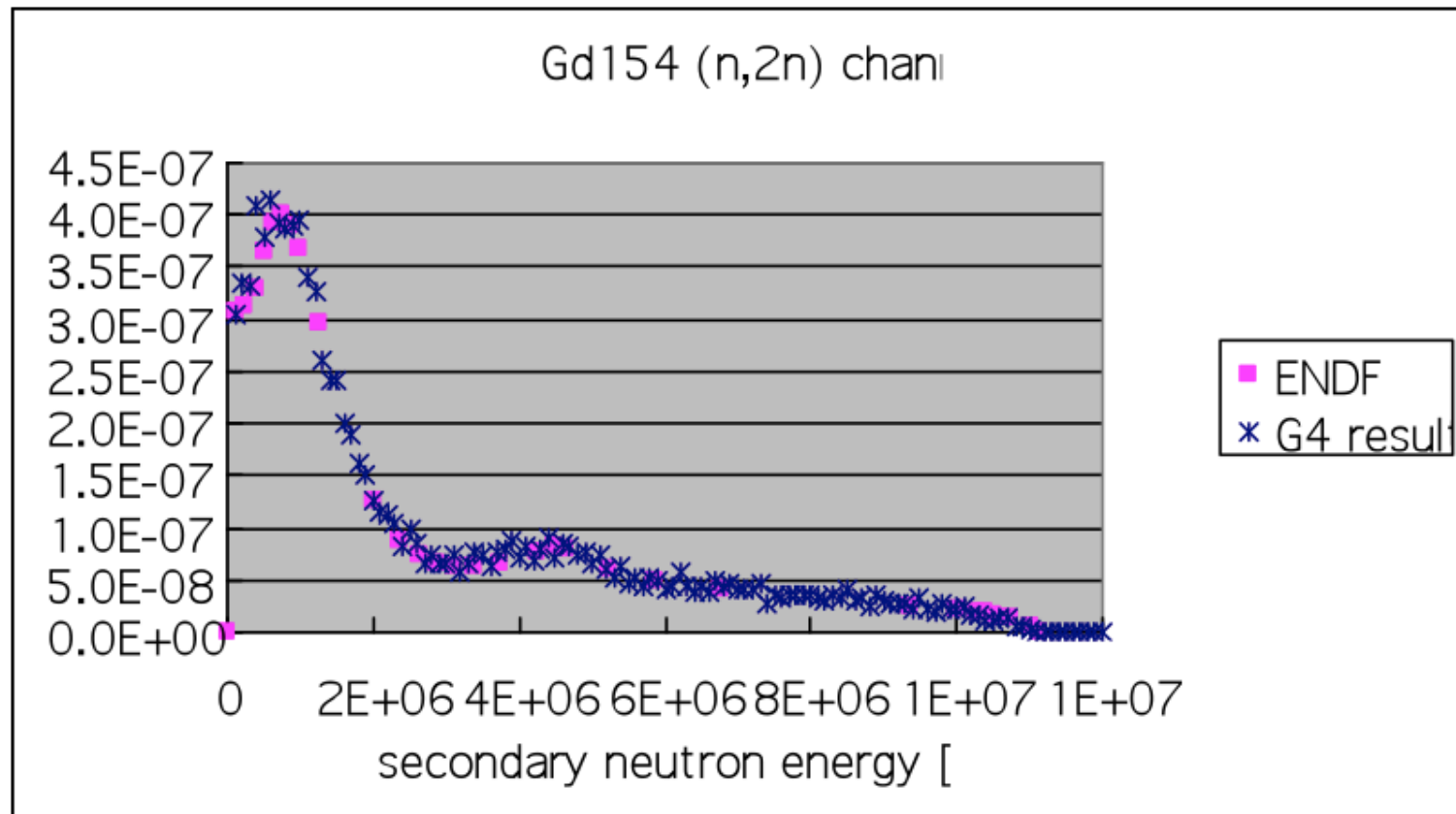
`http://g4validation.fnal.gov:8080/G4ValidationWebApp/`

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

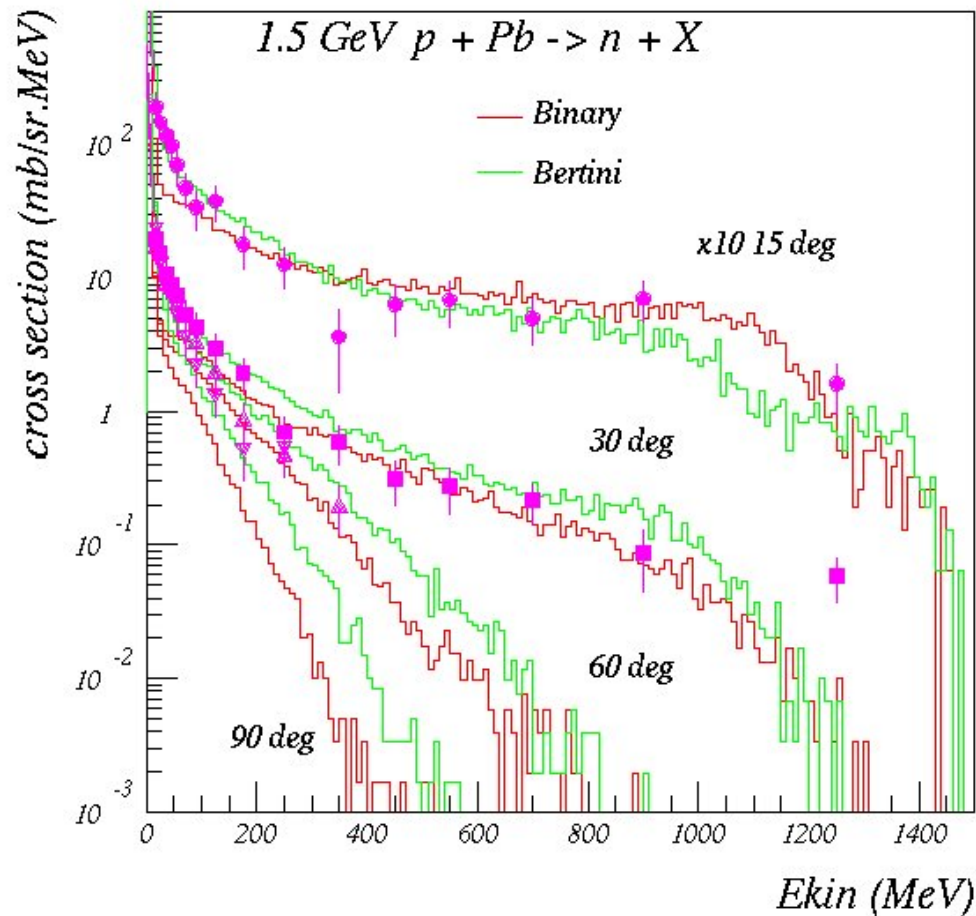
Some verification: channel cross section



Some verification: secondary energy spectrum

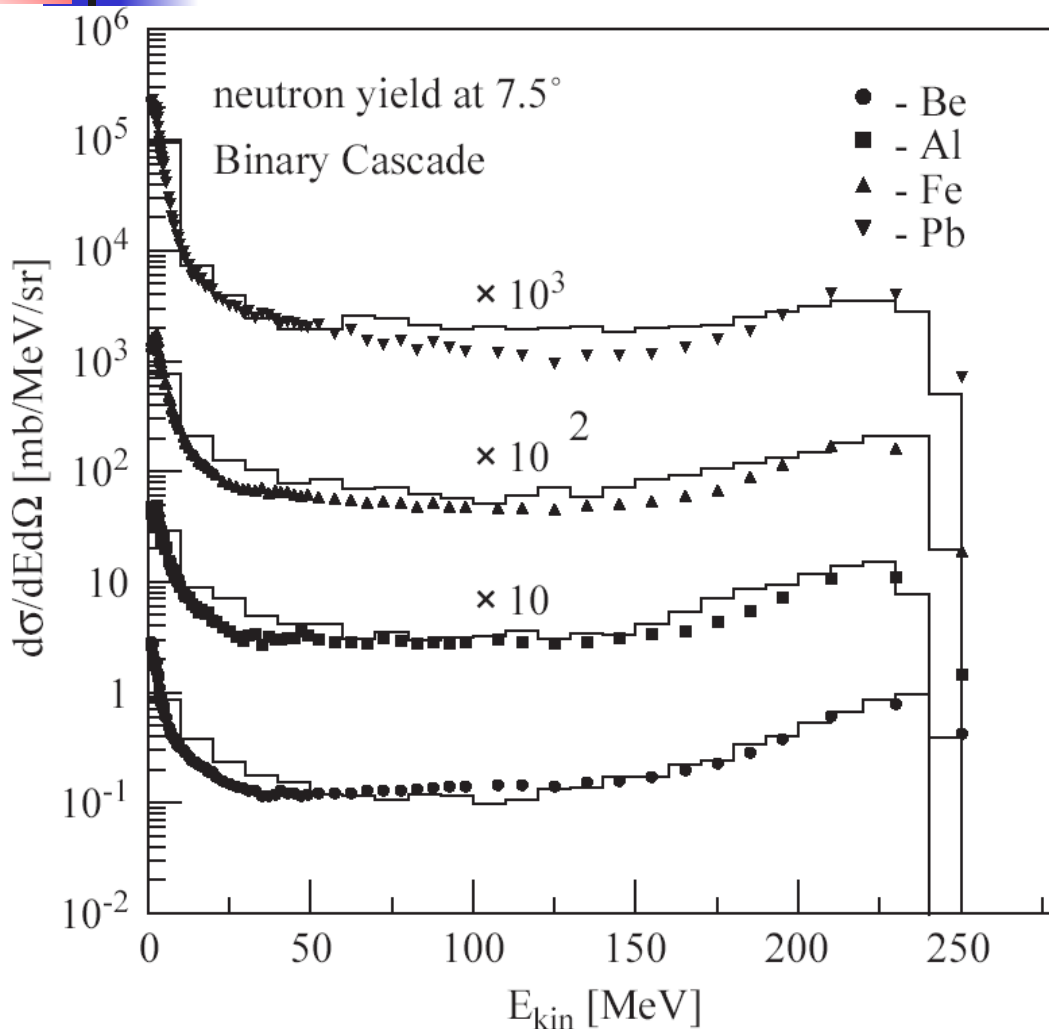


Nuclear fragmentation



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

Neutron production by protons

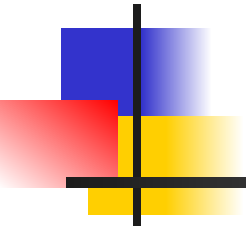


Binary cascade model:
double differential
cross-section for
neutrons produced
by 256 MeV protons
impinging on different
targets



Hands-on session

- Task3
- `http://geant4.lngs.infn.it/belgrade2016/task3`



Backup



Code Example (1/2)

```
G4ParticleDefinition* neutron=  
    G4Neutron::NeutronDefinition();  
G4ProcessManager* protonProcessManager =  
    proton->GetProcessManager();
```

} retrieve the
process
manager for
neutron

```
// Elastic scattering
```

```
G4HadronElasticProcess* neutronElasticProcess =  
    new G4HadronElasticProcess();
```

} create the
process for
elastic scattering

```
G4NeutronHPElastic* neutronElasticModel =  
    new G4NeutronHlastic();  
neutronElasticModel->SetMaxEnergy(20.*MeV);
```

} get the **HP model** for
elastic scattering

```
neutronElasticProcess->  
RegisterMe(neutronElasticModel);
```

} **register** the model to the
process

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

} attach the process to
neutron

Code example (2/2)

```
// Inelastic scattering
```

```
G4ProtonInelasticProcess* protonInelasticProcess  
= new G4ProtonInelasticProcess();
```

creates the **process** for inelastic scattering

```
G4BinaryCascade* protonInelasticModel1  
= new G4BinaryCascade();
```

```
protonInelasticModel1->SetMaxEnergy(4*GeV);
```

```
protonInelasticProcess->
```

```
RegisterMe(protonInelasticModel1);
```

gets the **Binary model** up to 4 GeV

registers model to the process

```
G4TheoFSGenerator* protonInelasticModel2 =  
new G4TheoFSGenerator("FTFB");
```

```
protonInelasticModel2->SetHighEnergyGenerator(  
new G4FTFModel);
```

```
protonInelasticModel2->SetMinEnergy(4.0*GeV);
```

```
protonInelasticProcess
```

```
->RegisterMe(protonInelasticModel2);
```

gets the **FTF model** from 4 GeV

registers model to the process

Model 1

Model 2



Example: PhysicsList, γ -rays

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

Only PostStep



- 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**)



How to extract Physics ?

- Possible to **retrieve physics quantities** via **G4EmCalculator** or directly from the **physics models**
 - Physics List should be initialized
- Example for retrieving the **total cross section** (cm^{-1}) of a process with name *procName*: for particle *partName* and material *matName*

```
G4EmCalculator emCalculator;  
G4Material* material =  
    G4NistManager::Instance()->FindOrBuildMaterial("matName");  
G4double massSigma = emCalculator.ComputeCrossSectionPerVolume  
    (energy,particle,procName,material);  
G4cout << G4BestUnit(massSigma, "Surface/Volume") << G4endl;
```

A good example:

```
$G4INSTALL/examples/extended/electromagnetic/  
TestEm14
```



Alternative cross sections

- To be used for specific applications, or for a **given particle** in a **given energy range**, for instance:
 - Low energy neutrons
 - **elastic, inelastic, fission** and **capture** (< 20 MeV)
 - Neutron and proton inelastic cross sections
 - $20 \text{ MeV} < E < 20 \text{ GeV}$
 - Ion-nucleus reaction cross sections (several models)
 - Good for $E/A < 1 \text{ GeV}$
 - Isotope production data
 - $E < 100 \text{ MeV}$
 - Photo-nuclear cross sections

Information on the available cross sections at

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