

# Physics

Witold Pokorski, Alberto Ribon  
CERN PH/SFT



# G4 Datasets (1)

- Some physics models are data-driven, i.e. they are phenomenological models that need some input data (either experimental or computed from theory)
- If you build G4 with the option **GEANT4\_INSTALL\_DATA** then the data-sets are automatically downloaded & installed
- Else (you want or need to do it manually, e.g. for older versions of G4) you need to install the data-sets yourself and then inform G4 where they are by defining the following environmental variables (e.g. for the latest version G4 10.2):

```
export G4LEDDATA=/dir-path/G4EMLOW6.48
```

```
export G4LEVELGAMMADATA=/dir-path/PhotonEvaporation3.2
```

```
export G4SAIDXSDATA=/dir-path/G4SAIDDATA1.1
```

```
export G4NEUTRONXSDATA=/dir-path/G4NEUTRONXS1.4
```

```
export G4ENSDFSTATEDATA=/dir-path/G4ENSDFSTATE1.2
```

```
export G4NEUTRONHPDATA=/dir-path/G4NDL4.5
```

```
export G4RADIOACTIVEDATA=/dir-path/RadioactiveDecay4.3
```

```
export G4REALSURFACEDATA=/dir-path/RealSurface1.0
```

## G4 Datasets (2)

- **G4LEDDATA** : low-energy electromagnetic data, mostly derived from Livermore data libraries; used in all EM options
- **G4LEVELGAMMADATA** : photon evaporation data, come from the Evaluated Nuclear Structure Data File (ENSDF); used by Preco
- **G4SAIDXSDATA** : data evaluated from the SAID database for nucleon and pion cross sections below 3 GeV; used in all physics lists
- **G4NEUTRONXSDATA** : evaluated neutron cross sections derived from G4NDL by averaging in bin of energies; used in all physics lists
- **G4ENSDFSTATEDATA** : nuclear properties, from Evaluated Nuclear Structure Data File (ENSDF); used in all physics lists

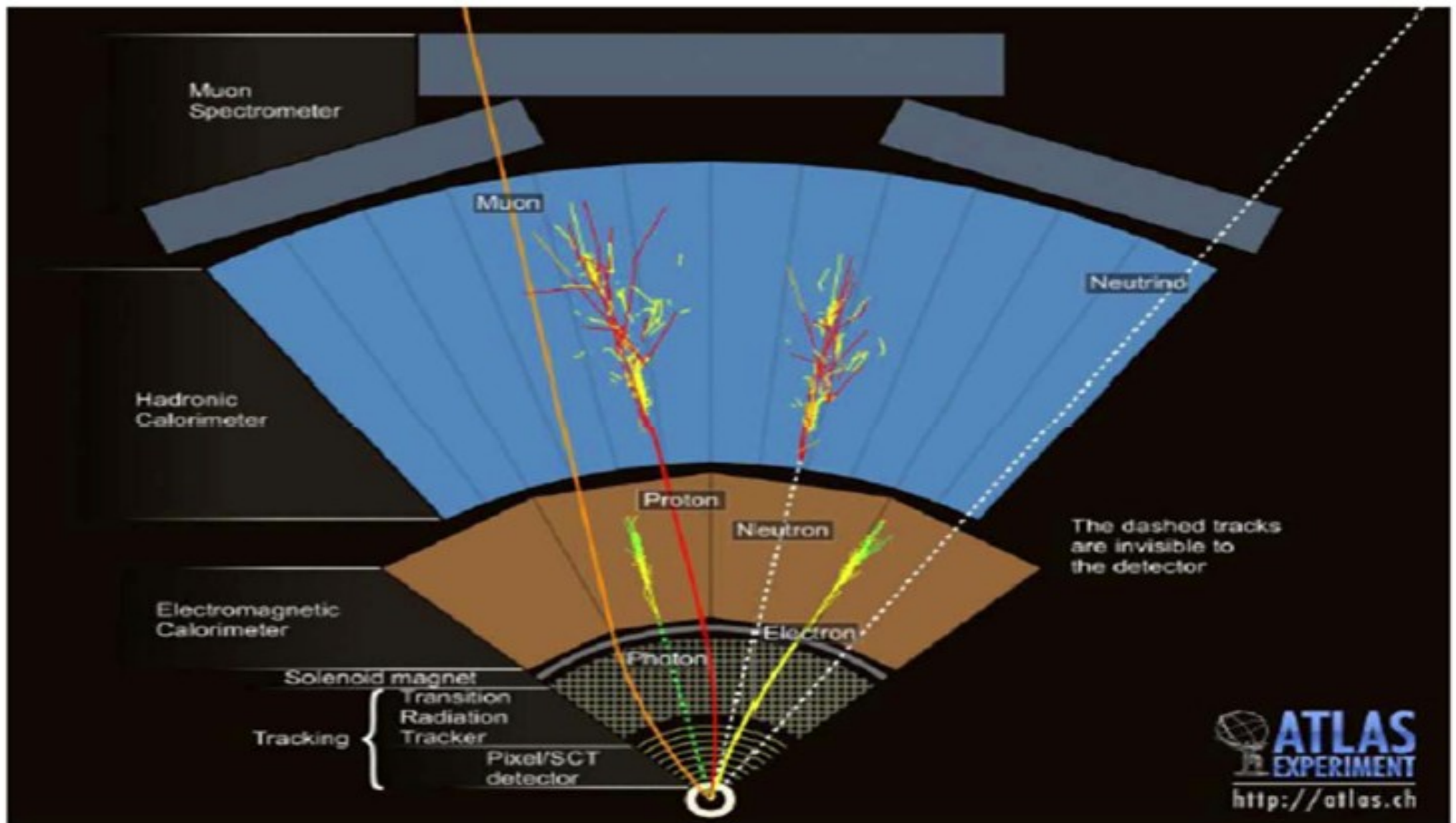
# G4 Datasets (3)

- **G4NEUTRONHPDATA** : evaluated neutron data of cross sections, angular distributions and final-state information; come largely from the ENDF/B-VII library; used only in `_HP` physics lists
- **G4RADIOACTIVEDATA** : radioactive decay data, come from the ENSDF; used only when radioactive decay is activated
- **G4REALSURFACEDATA** : data for measured optical surface reflectance look-up tables; used only when optical physics is activated
- **G4TENDL** : data for ParticleHP (p, d, t, He3,  $\alpha$ )

# Electromagnetic physics (EM)

# 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

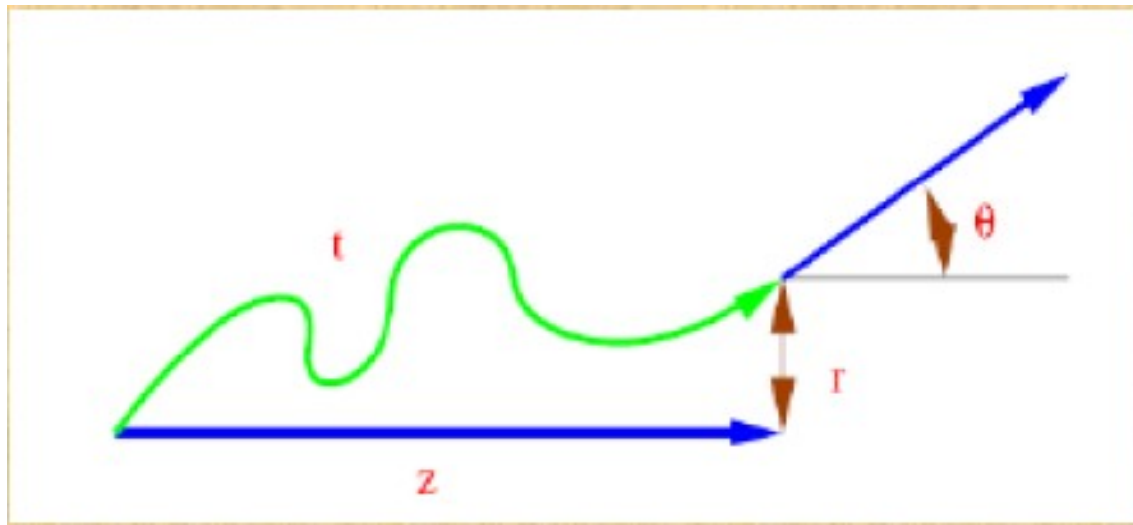
- Bremsstrahlung  
 $e^- (\text{atom}) \rightarrow e^- \gamma$
- MSC (Coulomb scattering):  
 $e^- \text{ atom} \rightarrow e^- \text{ atom}$
- Ionization :  
 $e^- \text{ atom} \rightarrow e^- \text{ ion}^+ e^-$
- Positron annihilation  
 $e^+ e^- \rightarrow \gamma \gamma$

## Charged hadron, ion

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

# 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  $\geq 1$  keV ;  
for a few processes, extensions to lower energies
- CPU performance of electromagnetic physics is critical :  
significant effort to improve it
- Detailed validation of electromagnetic physics is necessary  
before the validation of hadronic physics
- Typical precision in electromagnetic physics is  $\sim 1\%$ 
  - QED is extremely precise for elementary processes,  
but atomic and medium effects, important for detector simulations,  
bring larger uncertainties...
  - Moreover, the “condensed” description of multiple scattering  
introduces further approximations...
  - Major effort to improve the models

# EM options

- **Baseline** (default)
  - Used in production by ATLAS
  - Available in all reference physics lists, e.g. **FTFP\_BERT**
- **Fast** (EMV)
  - Used in production by CMS: good for crystals, not for sampling calo
  - Available in **\_EMV** variants of physics lists
- **Accurate** (EMY)
  - Used in medical and space science applications
  - Available in **\_EMY** variants of physics lists
- Other options are available:
  - **\_EMX** : experimental, used by LHCb
  - **\_EMZ** : the most precise EM available in G4
  - **\_LIV** : models based on the Livermore database
  - **\_PEN** : Penelope models implemented in Geant4

# Optical Photons

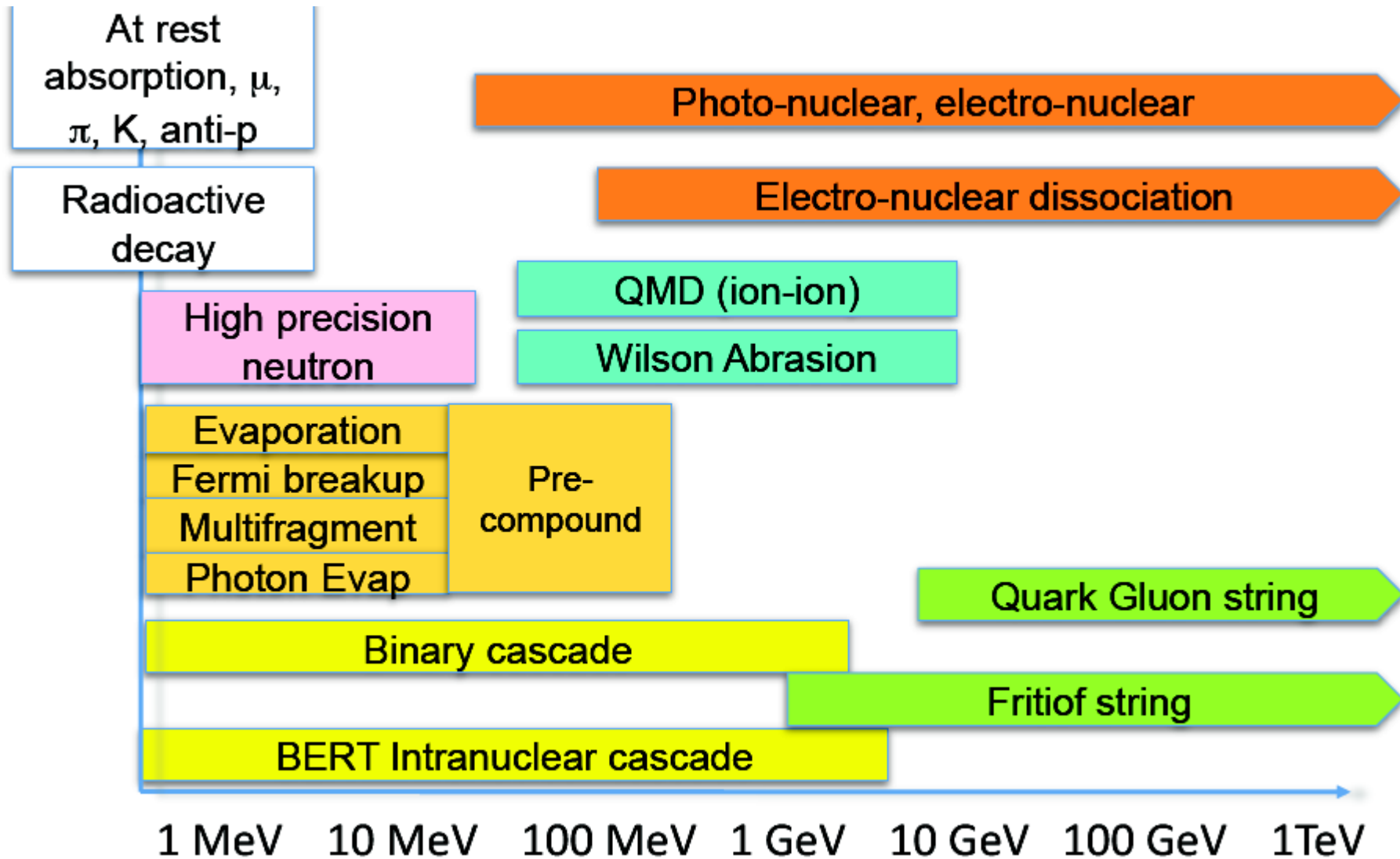
- A **photon** is considered to be **optical** when its wavelength is greater than the typical inter-atomic distance
- In Geant4, for convenience, optical photons are treated as a separated particle class, **G4OpticalPhoton**, distinct from the class of high-energy photons, **G4Gamma**
- Three processes in Geant4 can produce optical photons: **Cerenkov** effect, **scintillation**, and **transition radiation**
- Geant4 processes that can be associated to optical photons: **refraction**, **reflection**, **absorption**, **scattering**, **wavelength shifting**
- Optical properties of media (reflectivity, transmission, etc.) should be specified (in G4MaterialPropertiesTable linked to G4Material)
- For some examples, see: [\*examples/extended/optical/\*](#)

# Hadronic physics (HAD)

# Hadronic interactions

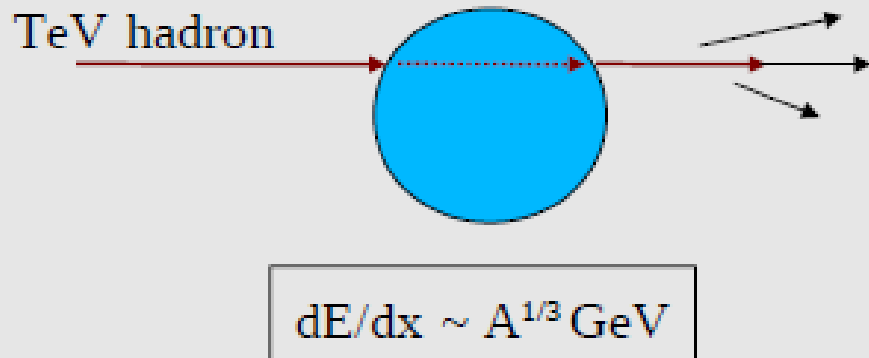
- Hadrons ( $\pi^\pm$ ,  $K^\pm$ ,  $K^0_L$ ,  $p$ ,  $n$ ,  $\alpha$ , etc.), produced in jets and decays, traverse the detectors (H, C, Ar, Si, Al, Fe, Cu, W, Pb...)
- Therefore we need to model **hadronic interactions**  
**hadron – nucleus -> anything**  
in our detector simulations
- In principle, QCD is the theory that describes all hadronic interactions; in practice, perturbative calculations are applicable only in a tiny (but important!) phase-space region
  - the hard scattering at high transverse momentumwhereas for the rest, i.e. most of the phase space
  - soft scattering, re-scattering, hadronization, nucleus de-excitation  
**only approximate models are available**
- **Hadronic models are valid for limited combinations of**
  - **particle type - energy - target material**

# Partial Hadronic Model Inventory

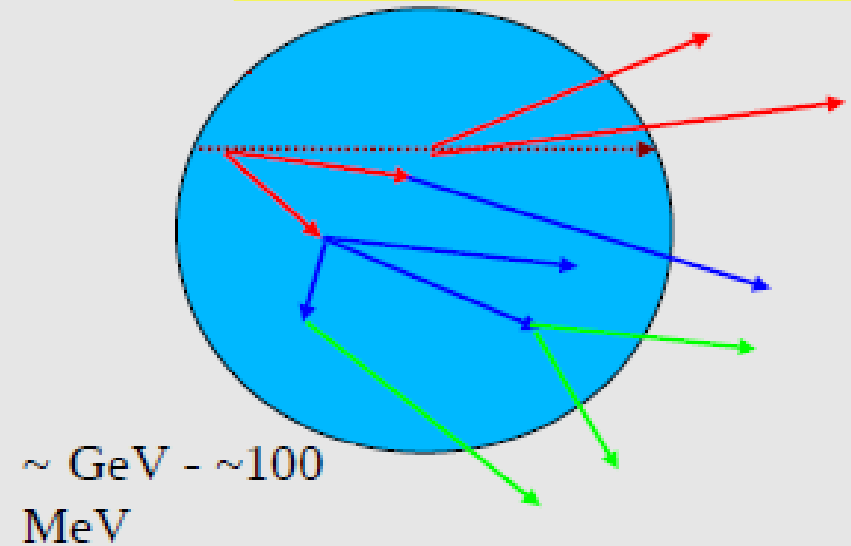


# Hadronic Interactions from TeV to meV

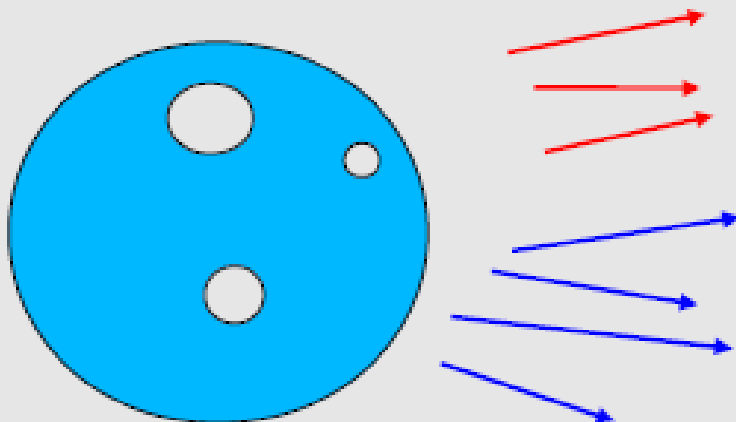
String model



Intra-nuclear cascade model

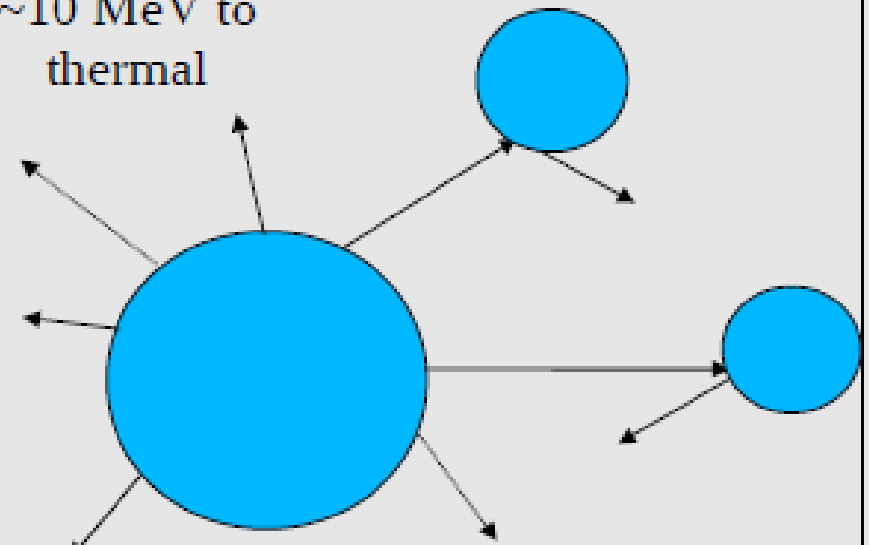


$\sim 100 \text{ MeV} - \sim 10 \text{ MeV}$



Pre-equilibrium (Precompound) model

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



Equilibrium (Evaporation) model

# An interesting complication: Neutrons

- Neutrons are abundantly produced
  - Mostly “soft” neutrons, produced by the de-excitation of nuclei, after hadron-nucleus interactions
  - It is typically the 3<sup>rd</sup> most produced particle (after e<sup>-</sup>,  $\gamma$ )
- 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 an order of magnitude according to the physical accuracy of the **neutron transportation simulation**
  - For typical high-energy applications, a simple treatment is enough (luckily!)
  - 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**

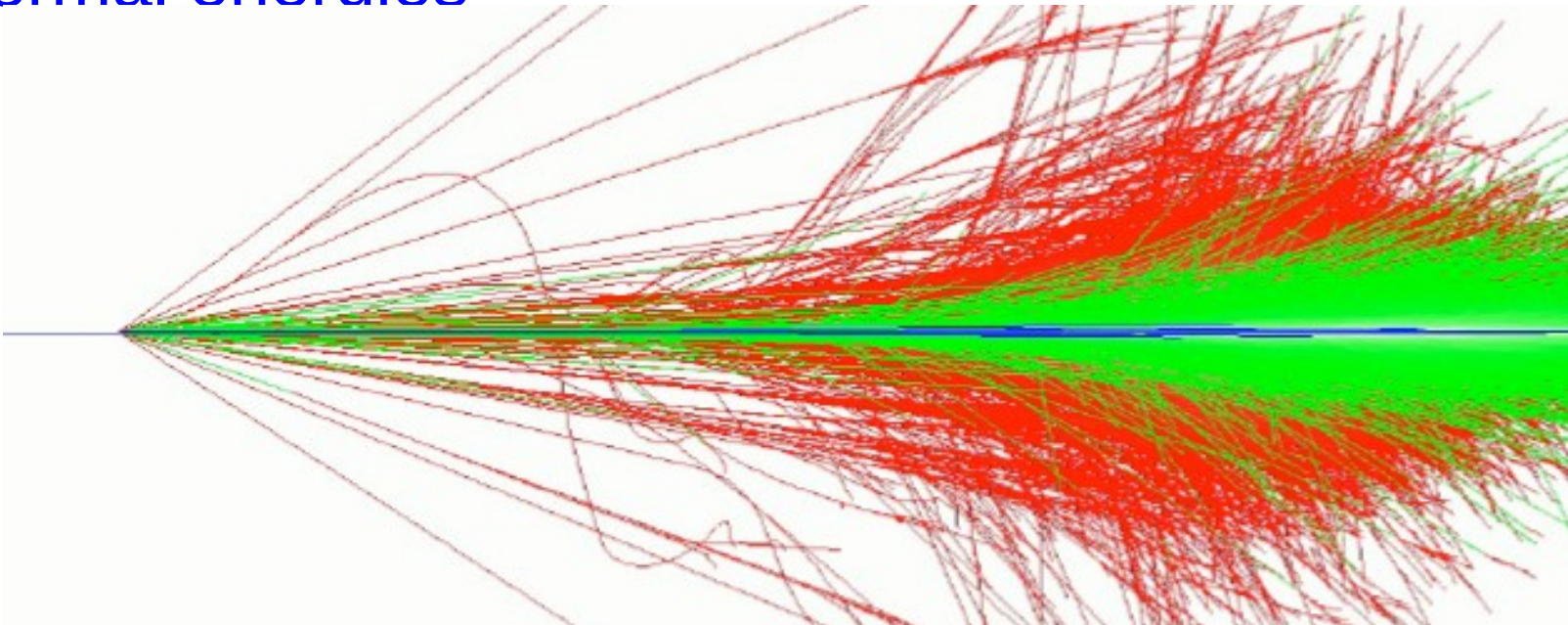


# Neutron-HP

- **High Precision treatment of low-energy neutrons**
  - $E_{\text{kin}} < 20 \text{ MeV}$  , down to thermal energies
  - Includes 4 types of interactions:  
radiative capture, elastic scattering, fission, inelastic scattering
  - Based on evaluated neutron scattering data libraries  
(pointed by the environmental variable **G4NEUTRONHPDATA** )
  - It is precise, but very slow!
- It is not needed for most high-energy applications; useful for:
  - cavern background, shielding, radiation damage, radio-protection
- Not used in most physics lists.  
If you need it, use one of the **\_HP** physics lists:  
**FTFP\_BERT\_HP** , **QGSP\_BERT\_HP** , **QGSP\_BIC\_HP** , **Shielding**

# Hadronic showers

- A single hadron impinging on a large block of matter (e.g. a hadron calorimeter) produces secondary hadrons of lower energies, which in turn can produce other hadrons, and so on: the set of these particles is called a **hadronic shower**
  - $e^-/e^+/\gamma$  (electromagnetic component) are also produced copiously because of  $\pi^0 \rightarrow \gamma\gamma$  and ionization of charged particles
- The development of a hadronic shower involves **many energy scales, from hundreds of GeV down to thermal energies**



# Jets

The simulation of **hadronic showers** is an important ingredient for the simulation of **jets**

- The other ingredients are:
  - the Monte Carlo event generator
  - the experiment-specific aspects: geometry, digitization, pile-up
- Jets (**collimated sprays of hadrons**) are produced by strong (QCD) or electroweak (hadronic decays of  $\tau$  / W / Z / H ) interactions
- Jets can be part of the signal and/or the background
  - multi-jets in the same event are typical in hadron colliders as LHC, but it is also frequent in high-energy e<sup>+</sup>-e<sup>-</sup> linear colliders as ILC/CLIC
- For future accelerators (e.g. LC (ILC/CLIC), FCC), the simulation of jets is essential for the optimal **design** of the detector
- For ATLAS and CMS, the simulation of jets is now important for **physics analysis**

# Physics Lists

# What is a Physics List ?

- A class that collects all the particles, physics processes, and production thresholds needed by your application
- **One and only one** physics list should be present in each Geant4 application
- There is **no default** physics list: it should always be explicitly specified
- It is a very **flexible** way to build a physics environment:
  - Users can pick only the particles they need
  - Users can assign to each selected particle only the processes they are interested in
- But users must have a good understanding of the physics required in their application:
  - Omission of particles or physics processes will cause errors or poor simulation

# Why do we need a Physics List ?

Nature has just one “physics”: so why Geant4 does not provide a complete and unique set of particles and physics processes that everyone can use?

- There are many **different physics models**, corresponding to a variety of approximations of the real phenomena
  - very much the case for hadronic physics,
  - but also for electromagnetic physics.

According to the application, one can be better than another. Comparing them can give an idea of systematic errors.

- **Simulation speed** is important
  - a user may prefer a less detailed but faster approximation
- Often all the physics and particles are not needed:
  - e.g. most high-energy applications do not need a detailed transportation of low-energy neutrons

# Reference Physics Lists

- Writing a complete and realistic physics list for EM physics and even more for hadronic physics is involved, and it depends on the application. To make things easier, pre-packaged **reference physics lists** are provided by Geant4, according to some reference use cases
- Few choices are available for EM physics (different production cuts and/or multiple scattering configurations); several possibilities are available for hadronics physics: e.g. **FTFP\_BERT**, **FTFP\_BERT\_HP**, **Shielding**  
**QGSP\_FTFP\_BERT**, **QGSP\_BIC\_EMY**
- These lists are “best guess” of the physics needed in a given case; they are intended as starting point (and their builders can be re-used); users are responsible of validating the physics lists for their application

# FTFP\_BERT

Recommended physics list for High-Energy Physics.  
Its main components are the following:

- **FTF** (Fritiof string) model, above 4 GeV
- **BERT** (Bertini cascade) model, below 5 GeV
- Nucleus de-excitation: **P**recompound + evaporation
- Neutron capture
- Nuclear capture of negatively charged hadrons at rest
- Gamma- and electro-nuclear
- Standard electromagnetics
  
- NO : neutron-HP, radioactive decay, optical photons



# How to use a reference Physics List

Let's consider the example of `FTFP_BERT` :  
In your main program:

```
#include "FTFP_BERT.hh"  
  
...  
int main( int argc, char** argv ) {  
  
...  
G4VModularPhysicsList* physicsList = new FTFP_BERT;  
runManager->SetUserInitialization( physicsList );  
  
...  
}
```

# How to add extra physics to a reference P.L.

- Adding **radioactive decay** :

In your main program:

```
#include "G4RadioactiveDecayPhysics.hh"
int main( int argc, char** argv ) {
    ...
    G4VModularPhysicsList* physicsList = new FTFP_BERT;
    physicsList->RegisterPhysics( new G4RadioactiveDecayPhysics );
    runManager->SetUserInitialization( physicsList );
    ...
}
```

- Adding **optical photon and its processes** :

In your main program:

```
#include "G4OpticalPhysics.hh"
int main( int argc, char** argv ) {
    ...
    G4VModularPhysicsList* physicsList = new FTFP_BERT;
    physicsList->RegisterPhysics( new G4OpticalPhysics );
    runManager->SetUserInitialization( physicsList );
    ...
}
```

# Recap: Model, Process, Physics List

- **Physics model** = final-state generator
  - Validated and tuned by Geant4 developers with thin-target data
- **Physics process** = cross section + final-state model
  - Different physics models can share the same cross section
- **Physics list** = a list of physics processes associated to each particle present in the simulation
  - Chosen by users: trade-off accuracy vs. speed
  - Geant4 offers some reference physics lists ready to be used
  - Validated by the users with (test-beam and/or collision) data

# Backup

# Optical Photons (1)

- A **photon** is considered to be **optical** when its wavelength is greater than the typical inter-atomic distance
- In Geant4, optical photons are treated as a separated particle class, **G4OpticalPhoton**, distinct from the class of high-energy photons, **G4Gamma**
  - This allows to incorporate some of the wave-like properties of electromagnetic radiation into the optical photon processes
  - But there is no smooth transition as a function of energy between optical photons and gammas
- Optical photons in G4 must always have linear polarization
  - This is guaranteed by the processes that can create them
  - For primary particle, the linear polarization should be set by the user: in the case of an unpolarized source, the linear polarization should be sampled randomly for each new primary photon

# Optical Photons (2)

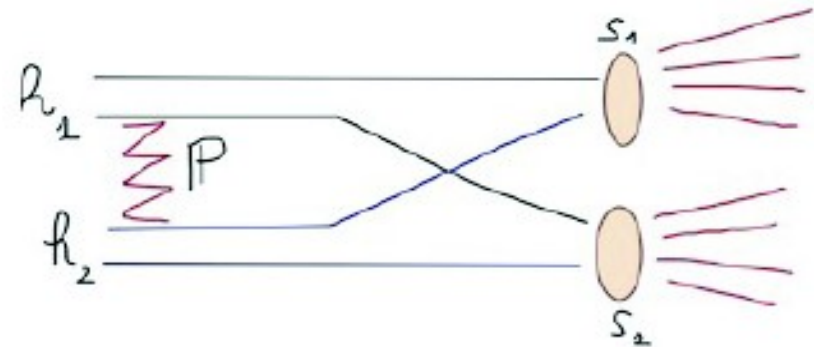
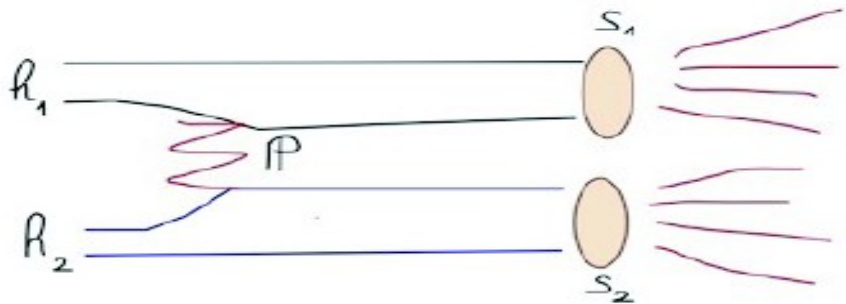
- Three processes in Geant4 can produce optical photons: **Cerenkov** effect, **scintillation**, and **transition radiation**
  - optical photons are generated without energy conservation
- Geant4 processes that can be associated to optical photons:
  - **refraction** and **reflection** at medium boundaries
  - bulk **absorption**
  - Mie and Rayleigh **scattering**
  - **wavelength shifting** (WLS)
- Optical properties of media:
  - the processes associated to optical photons require some optical properties of media: reflectivity, transmission efficiency, dielectric constants, surface properties
  - these properties are stored as entries in `G4MaterialPropertiesTable` which is linked to `G4Material`

# Optical Photons (3)

- For some examples, see: [\*examples/extended/optical/\*](#)
  - [\*OpNovice/\*](#) : simulation of optical photons generation and transport
  - [\*LXe/\*](#) : scintillation inside a bulk scintillator with PMT
  - [\*wls/\*](#) : propagation of photons inside a Wave Length Shifting (WLS) fiber
- In practice, optical photons are not used frequently
  - Substantial slow down of the simulation
  - Very difficult to model accurately all properties (and imperfections!) of real, experimental set-ups
  - Easier to treat the effects of optical photons at the level of “digitization”, i.e. applying a factor (determined from data) to the deposited energy obtained from Geant4
  - Most of the physics lists do not include optical photons, and their related processes, by default

# Fritiof (FTF) model (1)

- High-energy string model valid for any hadron projectile with kinetic energy between  $\sim 3$  GeV and  $\sim 1$  TeV
- Selection of collision partners: projectile, nucleon
- Splitting of nucleons into quarks and diquarks
- Formation and excitation of strings (between constituents)
- String hadronization/fragmentation
  - Insert q-q pair ,  $u : d : s : qq = 1 : 1 : 0.27 : 0.1$
  - At break: new string plus hadron
    - gets  $P_{\parallel}$  from sampling fragmentation functions
    - gets  $P_{\perp}$  from sampling a Gaussian





# FTF (2)

- Build up 3D model of nucleus
- Calculate impact parameter with all nucleons
  - Calculate hadron-nucleon collision probabilities
  - Multiple collisions are allowed
  - Use gaussian density distributions for hadrons and nucleons
- Sample number of strings exchanged in each collision
- String formation and then fragmentation into hadrons
- Remnant nucleus
  - After the HE interaction is performed an excited nucleus remains
  - De-excitation via precompound model
- Recently extended to nucleus-nucleus collision
  - Applicable for kinetic energies above  $\sim 2$  GeV/nucleon
- Under developing/testing: re-scattering
  - Use Binary Cascade or Bertini to propagate string fragments in nuclear media

# Bertini-like intra-nuclear cascade model (BERT)

- The Geant4 implementation started as the original Bertini (1960), but it was then extended and modified significantly
- Valid for  $p$ ,  $n$ ,  $\pi$ ,  $K$ , hyperons with  $E_{\text{kin}} \sim < 10 \text{ GeV}$
- The incident hadron penetrates the nucleus, and propagates in a density-dependent nuclear potential
- All hadron-nucleon interactions are based on free-space cross sections, angular distributions, etc., but the Pauli exclusion principle is taken into account
- Each secondary is propagated in the nuclear potential until re-interacts or leaves the nucleus
- Particle-hole excitons are created during the cascade

# Preco: pre-equilibrium

Native pre-equilibrium de-excitation model in Geant4 is a version of the standard **exciton model**. Key ingredients:

- Internal transition rates:
  - CEM (Cascade Exciton Model): default
  - Blann-Machner's parameterization
- Emission rates:
  - Nucleon emission in standard exciton formulation
  - Complex particle emission ( d , t , 3He ,  $\alpha$  ) from CEM

The pre-equilibrium phase continues until:

*number of excitons  $\leq$  number of excitons in equilibrium*

then transition to equilibrium

# Preco: equilibrium

Five processes are considered:

*Alternates:*

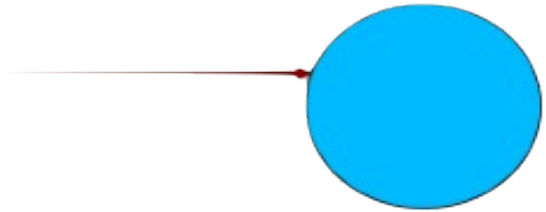
- **Fermi Breakup**, for  $Z < 9$  and  $A < 17$  (Botvina et al)
- **Statistical Multifragmentation**, for  $E^*/A > 3$  MeV (Botvina et al)

*Competitors:*

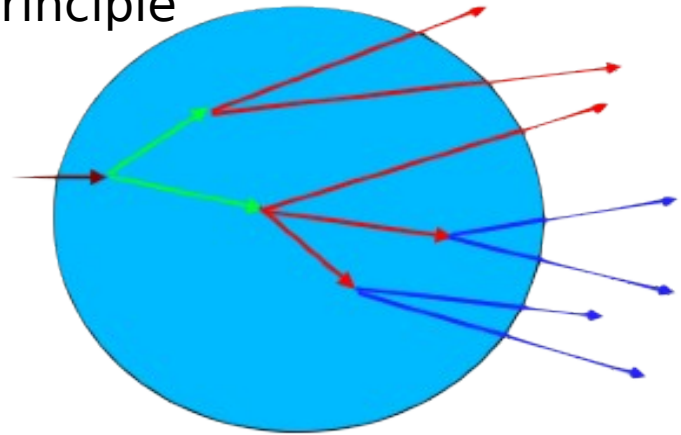
- **Fission** (Bohr-Wheeler model + Amelin prescript.)
- **Particle Evaporation** :
  - Evaporation Model WE (Weisskopf-Ewing)  
(evaporation of:  $n$  ,  $p$  ,  $d$  ,  $t$  ,  $3\text{He}$  ,  $\alpha$  )
  - Generalized Evaporation Model GEM (Furihata)  
(heavier ejected fragments:  $Z < 13$  and  $A < 29$  )
- **Photon Evaporation** :
  - Discrete (tabulated E1, M1, E2)
  - Continuum (GDR strength)

# BERT + Preco (1)

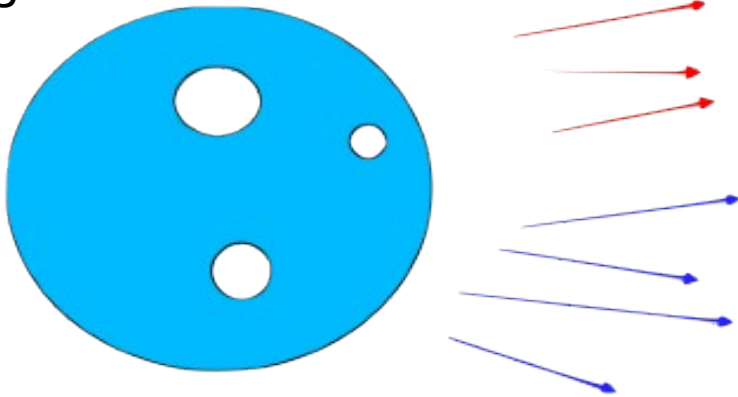
Hadron penetrates nucleus  
Nucleus: density-dependent potential



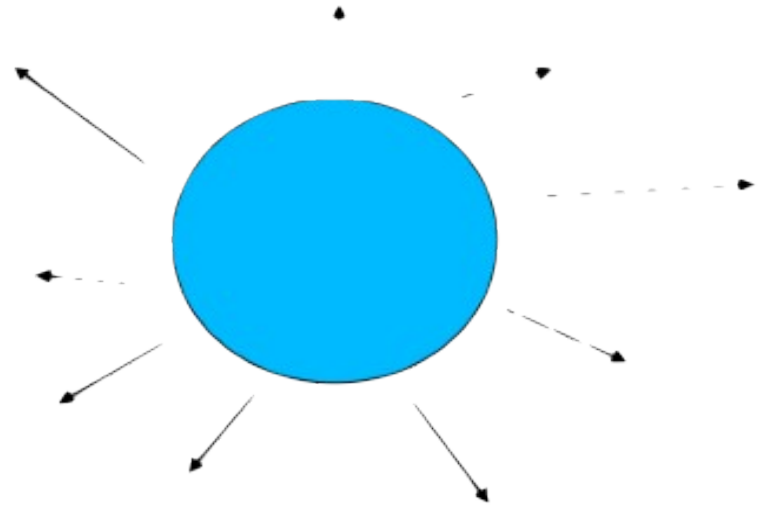
Free-space  $\sigma$  and angular distributions  
Pauli principle



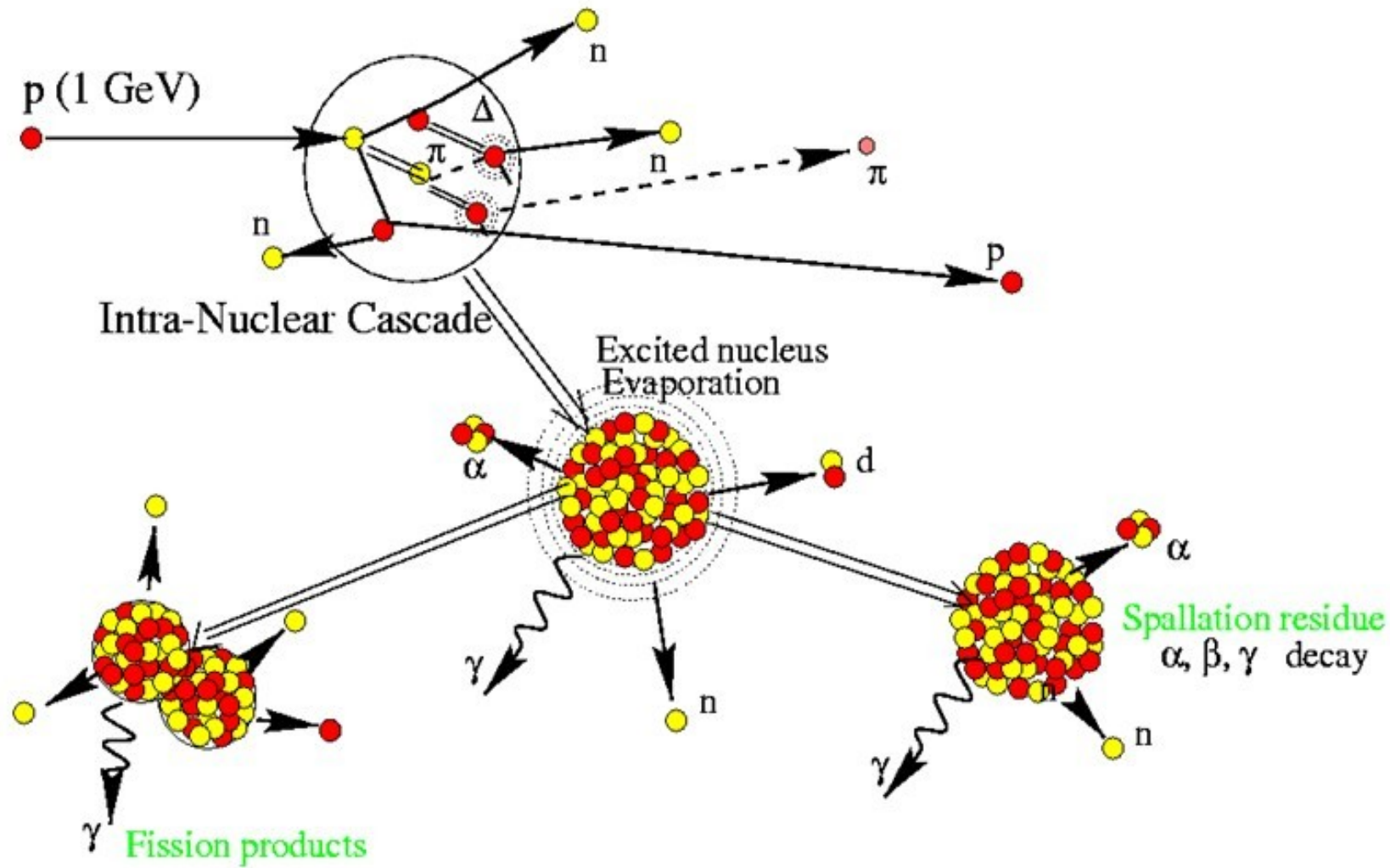
Particle-hole excitations are created during cascade



Nucleus is de-excited



# BERT + Preco (2)



# Component-based approach to physics

Geant4 takes a component-based approach to physics:

- Provide many **physics components (processes)** which are decoupled from one another
- User selects these components in custom-designed physics lists in much the same way as a detector geometry is built
- Exceptions:
  - A few electromagnetic processes must be used together
  - Future processes involving interference of electromagnetic and strong interactions may require coupling as well

# G4VUserPhysicsList

- All physics lists must derive from this class
  - and then be registered with the Run Manager
- Example:

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

- User must implement the methods:  
**ConstructParticle** , **ConstructProcess** , and **SetCuts**



# ConstructParticle()

- Choose the particles you need in your simulation and define all of them here. Example:

```
void MyPhysicsList::ConstructParticle() {  
    G4Electron::ElectronDefinition();  
    G4Positron::PositronDefinition();  
    G4Gamma::GammaDefinition();  
    G4Proton::ProtonDefinition();  
    G4Neutron::NeutronDefinition();  
    ...  
}
```

- It is possible to use Geant4 classes that create group of particles (instead of individual ones):

```
G4BosonConstructor , G4LeptonConstructor ,  
G4MesonConstructor , G4BaryonConstructor , G4IonConstructor
```

# ConstructProcess()

For each particle defined in ConstructParticle() assign all the physics processes that you want to consider in your simulation

```
void MyPhysicsList::ConstructProcess() {
```

```
    AddTransportation();
```

```
// method provided by G4VUserPhysicsList , assign transportation process  
// to all particles defined in ConstructParticle()
```

```
    ConstructEM();
```

```
// convenience method that user may define; put electromagnetic physics here
```

```
    ConstructGeneral();
```

```
// convenience method that user may define
```

```
}
```

# ConstructEM()

```
void MyPhysicsList::ConstructEM() {  
  theParticleIterator->reset();  
  while ( (*theParticleIterator)() ) {  
    G4ParticleDefinition* particle = theParticleIterator->value();  
    G4ProcessManager* pmanager =  
        particle->GetProcessManager();  
    G4String particleName = particle->GetParticleName();  
  
    if ( particleName == "gamma" ) {  
      pmanager->AddDiscreteProcess(  
          new G4GammaConversion() );  
  
      ...  
    }  
  
    ...  
  }  
  
  ...  
}
```

# ConstructGeneral()

```
void MyPhysicsList::ConstructGeneral() {  
    G4Decay* theDecayProcess = new G4Decay();  
    theParticleIterator->reset();  
    while ( (*theParticleIterator)() ) {  
        G4ParticleDefinition* particle = theParticleIterator->value();  
        G4ProcessManager* pmanager =  
            particle->GetProcessManager();  
        if ( theDecayProcess->IsApplicable( *particle ) ) {  
            pmanager->AddProcess( theDecayProcess );  
            pmanager->SetProcessOrdering( theDecayProcess,  
                idxPostStep );  
            pmanager->SetProcessOrdering( theDecayProcess,  
                idxAtRest );  
        }  
    }  
}
```

# SetCuts()

```
void MyPhysicsList::SetCuts() {  
    defaultCutValue = 1.0*mm;  
  
    SetCutValue( defaultCutValue, "gamma" );  
    SetCutValue( defaultCutValue, "e-" );  
    SetCutValue( defaultCutValue, "e+" );  
}
```

# A simple G4VModularPhysicsList

- Constructor

```
MyModPhysList::MyModPhysList() : G4VModularPhysicsList() {  
    defaultCutValue = 1.0*mm;  
    RegisterPhysics( new ProtonPhysics() );  
    // all physics processes having to do with protons  
  
    RegisterPhysics( new ElectronPhysics() );  
    // all physics processes having to do with electrons  
  
    RegisterPhysics( new DecayPhysics() );  
    // decay of unstable particles  
}
```

- SetCuts

```
void MyModPhysList::SetCuts() {  
    SetCutsWithDefault();  
}
```

# G4VModularPhysicsList

Use [G4ModularPhysicsList](#) to build a realistic physics list which would be too long, complicated and hard to maintain with the previous approach.

Its features are:

- Derived from **G4VUserPhysicsList**
- [AddTransportation\(\)](#) automatically called for all registered particles
- Allows to define “[physics modules](#)”:
  - electromagnetic physics
  - hadronic physics
  - decay physics
  - ion physics
  - radioactive physics
  - optical physics
  - etc.

# Physics Constructors

Allow you to group particle and process construction according to physics domains

```
class ProtonPhysics : public G4VPhysicsConstructor {  
public:  
ProtonPhysics( const G4String& name = "proton" );  
virtual ~ProtonPhysics();  
  
virtual void ConstructParticle();  
// easy – only one particle to build in this case  
  
virtual void ConstructProcess();  
// put here all the processes a proton can have  
}
```