Geant4 10.6 beta

Physics II: Processes and Production Thresholds

Geant4 Tutorial at Chalk River

Dennis Wright (SLAC)

27 August 2019

Outline

- Overview of Geant4 physics
  - EM, hadronic, decay, transportation
- Processes
  - What is a process and what does it do ?
  - Examples of Geant4 processes
- Production thresholds
  - How they work, how to set them
  - Cuts per region

Geant4 Physics

- Geant4 provides a wide variety of physics components for use in simulation
- Physics components are coded as processes
  - a process is a well-defined interaction of a particle with matter
  - determines when the interaction happens and what the result is
  - Geant4 provides a large number of these
  - users may write their own, but they must be derived from a Geant4 process
  - all processes are developed by implementing the interface G4VProcess
- Processes are classified as
  - electromagnetic, hadronic, decay, parameterized or transportation

Geant4 Physics: Electromagnetic

- Standard – complete set of processes covering charged particles and gammas
  - energy range 1 keV to ~PeV
- Low energy – specialized routines for e^-, γ, charged hadrons
  - more atomic shell structure details
  - some processes valid down to 250 eV or below
  - others not valid above a few GeV
- Optical photons – only for long wavelength photons (x-rays, UV, visible)
  - processes for reflection/refraction, absorption, wavelength shifting, Rayleigh scattering
  - users select these components in their physics lists

Geant4 Physics: Hadronic

- Pure hadronic (0 to ~TeV)
  - elastic
  - inelastic
  - capture
  - fission
- Radioactive decay
  - at rest and in-flight
- Photo-nuclear (~10 MeV - ~TeV)
  - gamma-induced nuclear reactions
- Lepto-nuclear (~10 MeV - ~TeV)
  - e^+, e^- induced nuclear reactions
  - muon-induced nuclear reactions

Physics Processes: Decay, Parameterized and Transportation

- Decay processes include
  - weak decay (leptonic, semi-leptonic decays, radioactive decay of nuclei)
  - electromagnetic decay (π^0 , Σ^0 , etc.)
  - strong decays not included here (they are part of hadronic models)
- Parameterized process
  - electromagnetic showers propagated according to parameters averaged over many events
  - faster than detailed shower simulation
- Transportation
  - only one process which is responsible for moving the particle through the geometry
  - must be assigned to each particle type

Physics Processes

- All the work of particle decays and interactions is done by processes
- A process does two things:
  - decides when and where an interaction will occur
    - method: GetPhysicalInteractionLength()
    - this requires a cross section or decay lifetime
    - for the transportation process the distance to the nearest object along the track is required
  - generates the final state of the interaction (changes momentum, generates secondaries, etc.)
    - method: DoIt()
    - this requires a model of the physics

Physics Processes

- There are three flavors of processes:
  - well-located in space -> PostStep
  - distributed in space -> AlongStep
  - well-located in time -> AtRest
- A process may be a combination of all three of the above
  - in that case six methods must be implemented , one GetPhysicalInteractionLength() and one DoIt() for each type
- "Shortcut" processes are defined which invoke only one
  - Discrete process (has only PostStep physics)
  - Continuous process (has only AlongStep physics)
  - AtRest process (has only AtRest physics)

Example Processes

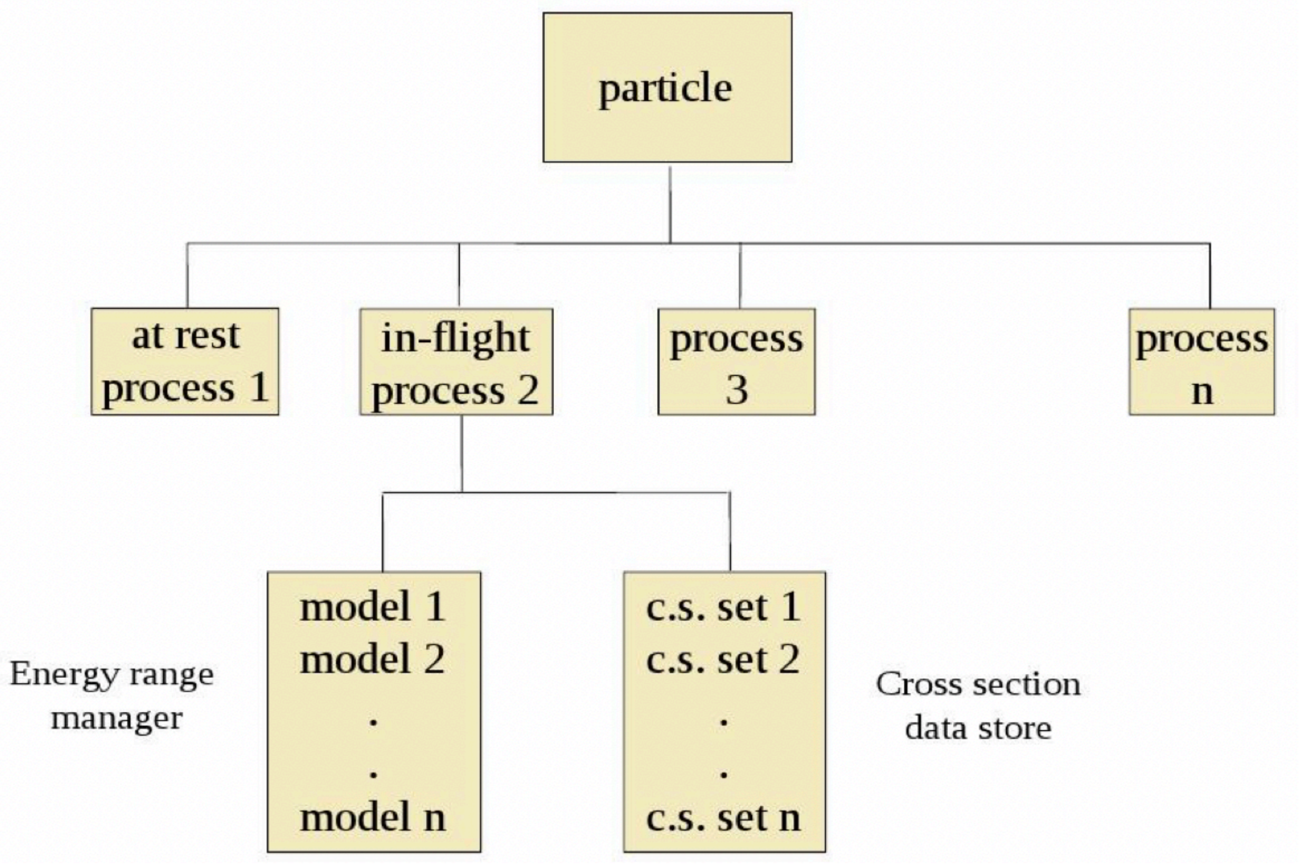
- Discrete process: Compton scattering
  - length of step determined by cross section, interaction at end of step
    - PostStepGetPhysicalInteractionLength()
    - PostStepDoIt()
- Continuous process: Cerenkov effect
  - photons created along step, # proportional to step length
    - AlongStepGetPhysicalInteractionLength()
    - AlongStepDoIt()
- At rest process: μ^- capture at rest
  - muon has already stopped so time is the relevant variable
    - AtRestGetPhysicalInteractionLength()
    - AtRestDoIt()

Example Processes

- Continuous + discrete: ionization
  - energy loss is continuous (low energy electrons not generated)
  - Moller/Bhabha scattering and knock-on electrons are discrete
- Continuous + discrete: bremsstrahlung
  - energy loss is continuous (soft photon emission not performed)
  - hard photon emission is discrete
- Discrete + at-rest: e^+ annihilation
  - in-flight annihilation is discrete
  - at rest annihilation when positron stops
- Multiple scattering is also continuous + discrete

Handling Multiple Processes

- Many processes (and therefore many interactions) may be assigned to the same particle



Handling Multiple Processes

- How does Geant4 decide which interaction happens at any one time?
  - interaction length (or decay length) is sampled for each process
  - sampling of length L is done from the distribution $1 - e^{-\sigma\rho L}$
  - done for each process assigned to the particle to the particle

  - now we have several lengths, including distance to next volume boundary
  - the interaction (or boundary crossing) that happens is the one with the shortest length

  - processes which did not occur are not re-sampled, but have the "expended probability" ($L_n N_n \sigma_n$) from the previous step subtracted from the originally sampled total probability of interaction

Threshold for Secondary Production

- Every simulation developer must answer the question: how low can you go?
  - at what energy do I stop tracking particles?

- 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 CPU to skyrocket

- 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

Threshold for Secondary Production

- But such a cut may cause imprecise stopping location and deposition of energy

- There is also a particle dependence
  - range of a 10 keV γ in Si is a few cm
  - range of a 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 measurement

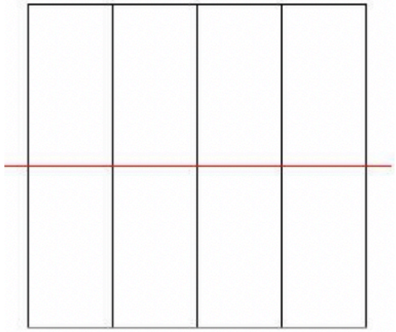
Threshold for Secondary Production

- Geant4 solution: impose a production threshold
    - this threshold is a distance, not an energy
    - default = 1 mm
    - the primary particle loses energy by producing secondary electrons and gammas
    - if primary no longer has enough energy to produce secondaries which can travel at least 1 mm, 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 threshold distance is needed for all materials because it corresponds to different energies depending on the material

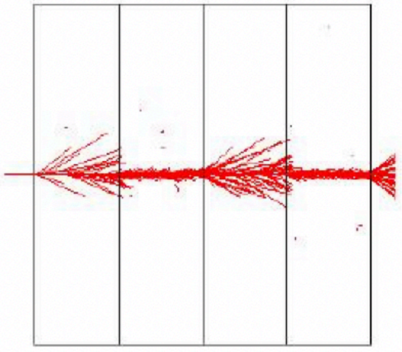
Production Threshold vs. Energy Cut

- Example: 500 MeV p in LAr-Pb Sampling Calorimeter

Geant3 (and others)

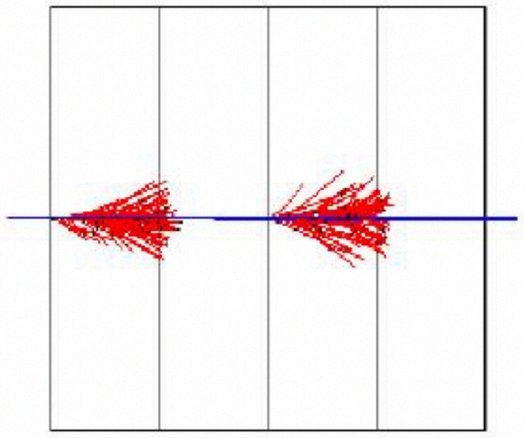


Cut = 2 MeV



Cut = 450 keV

Geant4

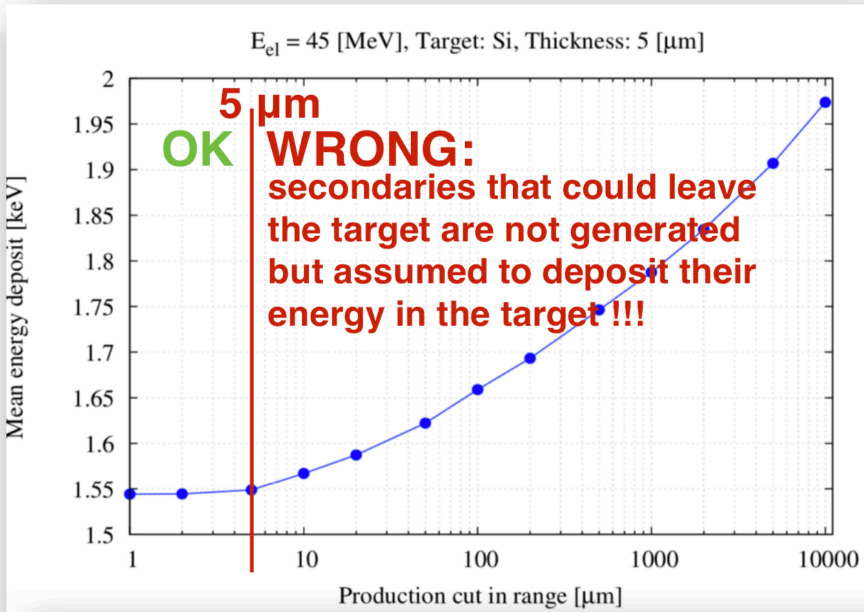


Production range = 1.5 mm

Threshold for Secondary Production

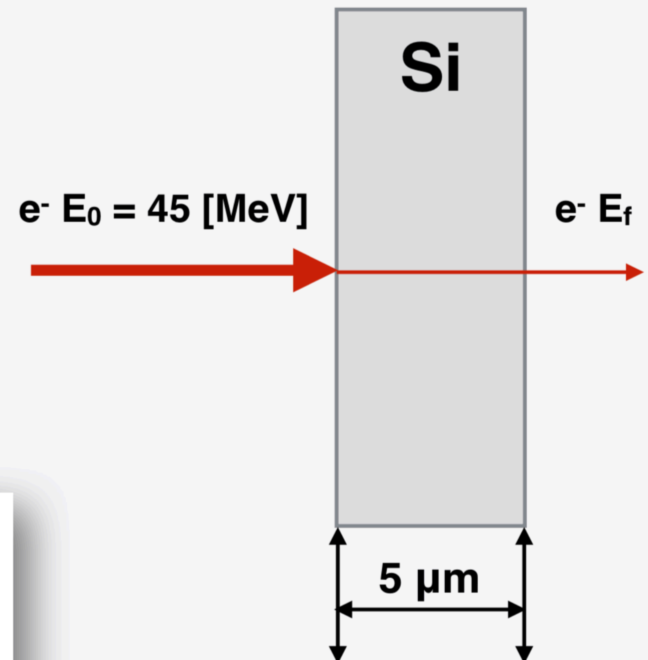
- Geant4 recommends the default value of 1 mm
  - user needs to decide the best value
  - will depend on size and sensitive elements within the simulated detector, and on available CPU
- This value is set in the SetCuts() method of your physics list
  - defined for γ, e^-, e^+, proton secondaries
  - UI commands
    - /run/setCut 0.1 mm
    - /run/setCutForAGivenParticle e- 0.1 mm
- Instead of "secondary production threshold distance" it is more convenient to simply say "cuts"
  - but please remember that this does not mean that any particle is stopped before it runs out of energy

Choosing the Correct Production Threshold



cut [μm]	mean E_{dep}	rms E_{dep}	prod. thres. [keV]		mean num. sec.	
			γ	e^-	γ	e^-
1	1.54423	0.000573911	0.99	0.99	0.0006811	0.1018230
2	1.54443	0.000583879	0.99	2.9547	0.0006843	0.0316897
5	1.54882	0.000605834	0.99	13.1884	0.0006857	0.0068261
10	1.56717	0.000665733	0.99	31.9516	0.0006730	0.0028232
20	1.58734	0.000743473	1.08038	47.8191	0.0006651	0.0018811
50	1.62223	0.000912408	1.67216	80.7687	0.0006557	0.0011304
100	1.65893	0.001108240	2.32425	121.694	0.0006518	0.0007536
200	1.69338	0.001342180	3.2198	187.091	0.0006465	0.000477
500	1.74642	0.001774670	5.00023	337.972	0.0006184	0.0002617
1000	1.78751	0.002219870	6.95018	548.291	0.0006054	0.0001622
2000	1.83440	0.002861020	9.66055	926.09	0.0005786	9.3e-05
5000	1.90700	0.004243030	14.9521	2074.3	0.0005427	4.07e-05
10000	1.97378	0.006036600	20.6438	4007.59	0.000521	2.22e-05

Compute the mean of the energy deposit (E_f-E_0) in the target



Cuts per Region

- In a complex detector there may be many different types of sub-detector 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 the most complex detectors
  - experienced at simulating EM showers in matter

Cuts per Region

- A default region is created automatically for the world volume
  - it uses the cut values which you set in SetCuts() in your physics list
  - these will be used everywhere except for user-defined regions

- In the geometry an instance of G4Region must be created which corresponds to the volume where the cuts are to be changed

- To define different cuts for this special region, user must
  - create a G4ProductionCuts object
  - initialize it with the new cuts
  - assign it to a new region which has already been created

Cuts per Region

- void MyPhysicsList::SetCuts() {

```
SetCutValue(defaultCutValue, "gamma");   // same for e-, e+, p

// Get the region
G4Region* aRegion =
    G4RegionStore::GetInstance()->GetRegion("RegionA");
// Define cuts object for the new region and set values
G4ProductionCuts* cuts = new G4ProductionCuts();
cuts->SetProductionCut(0.01*mm);   // here, same for all

// Assign cuts to region
aRegion->SetProductionCuts(cuts);
}
```

Summary

- Processes handle all the physics of particle interactions

- Geant4 provides processes to cover nearly all particles over energies ranging from 0 to ~TeV
  - users may define their own processes

- Many processes may be assigned to a particle type

- The precision of particle stopping and the production of secondary particles are determined by a secondary production threshold
  - which is a length

- For complex detectors with varying types of sensitive volumes, different production thresholds may be defined for different regions within the detector

Backup

Handling Multiple Processes

- Step 1:
  - all lengths sampled
  - Compton occurs
- Step 2:
  - Compton re-sampled
  - boundary is crossed
- Step 3:
  - Compton occurs again
  - new boundary found
- Step 4:
  - Compton re-sampled
  - pair production occurs