Geant4 Gamma-Electron Angular Correlations Extension Documentation

Rishita Gudapati

December 2017

"Evan's version" refers to the Geant4 Gamma-Gamma Angular Correlations extension written by Evan Rand. It is available on the GRIFFINCollaboration GitHub page under:

`Geant4GammaGammaAngularCorrelations10.01.p01`

"original source files" refers to the `Geant4` version 10.01.p03 source files.

G4DiscreteGammaTransition

This is where the program decides whether to emit a γ or an e^- based on the internal conversion coefficients in the Photon Evaporation file. The actual generation of the particle is done in `G4VGammaDeexcitation`.

This is also where `GetThetaFromWTheta` is located (called in `G4VGammaDeexcitation`). This method returns a θ value consistent with a specific correlation (passed in as *casenumber*). The correlation is not applied – it is applied in `G4VGammaDeexcitation`.

Header File

Additions to Evan's version:

- modify `GetThetaFromWTheta` to accept *casenumber* argument
- declarations for vectors mentioned below

Source File

Additions to Evan's version:

- each case (gg, ge...) has its own set of coefficients for which the vectors are populated at each level
- the above is also true for the *maxWTheta* that belongs to each case
- remove the subtraction of bond energy for electrons: this has been moved to `G4VGammaDeexcitation` to be consistent with the newer `Geant4` versions
- normalization is now performed separately for each case
  - I'm still not entirely sure how this normalization procedure works, and it is the likely culprit behind our inability to simulate the decay of `Tl198[543.6]` – see the emails from Evan, which Mike Bowry has copies of
- the normalization procedure produces a θ value which is then returned to `G4VGammaDeexcitation`

G4NuclearLevel

This is where the values of the angular coefficients are calculated.
This is also where `GenerateWThetaParameters` is located (called in `G4NuclearLevelManager`).

Header File

Additions to Evan's version:

- declarations for vectors mentioned below
- declare methods for filling b, α parameters for a specific level used in `G4NuclearLevelManager` when reading and storing values in multipole file

Source File

Additions to Evan's version:

- create and initialize vectors for b_1 through b_6 (b_1 to b_3 belong to a_2 coefficient, b_4 to b_6 belong to a_4)
- create and initialize vectors for a_2 to a_{10} for *each* separate transition type (gg, ge, eg, ee)
- create and initialize vectors for *maxWTheta* for each separate transition type
- calculate mixing ratios for e^- transitions
- coefficients: modify first and second transition equations to include$b$, α parameters
  - b parameters are set to 1 for a γ transition
- modified `GenerateWThetaParameters` to accept b, α values given in multipole file
- calculate and output angular coefficients for all transition types
- suppress output for levels with $a_2(\gamma\gamma) = a_4(\gamma\gamma) = 0$
  - we were using the entire Photon Evaporation file. In theory, this shouldn't be necessary if we only include the levels that we're interested in
- calculate *maxWTheta* for all four cases – used for normalization purposes in `G4DiscreteGammaTransition`

G4NuclearLevelManager

This is where we read in the multipole file and save its data to the appropriate energy levels.
This is also where `GenerateWThetaParameters` is called.

Header File

Additions to Evan's version:

- declarations for b and α vectors
  - each vector consists of one element for each energy level in the Photon Evaporation file

Source File

Additions to Evan's version:

- read in b parameters and α values from multipole file
- fill b parameters and α values into the right level
  - it finds the right level by looping through all levels in the Photon Evaporation file and comparing the level energy in the multipole file to the current level energy
  - in checking that the level is compared correctly, I ran into some floating point precision errors, so I added the *firstcheck* and *secondcheck* variables to counteract them. The precision can probably increased further in these checks.
- retrieve the b and α values for each level in the Photon Evaporation file (two levels at a time – the current one and the one immediately above it in energy) and pass these values to `G4NuclearLevel::GenerateWThetaParameters`

G4NuclearLevelStore

This is just a storage module for variables used across the code.

Header File

Deletions from Evan's version:

- removed getter and setter for *FirstGammaDecay*

Additions to Evan's version:

- getter and setter for `G4bool` *FirstProduct*
  - every nuclear deexcitation results in a product chain inside the 'DoDecay' method of `G4VGammaDeexcitation`. *FirstProduct* simply stores whether or not the product list is empty. The first product of a deexcitation is isotropically emitted from the nucleus.
- getter and setter for `G4bool` *ElecFirst*
  - true if the particle generated first in the cascade was an e^-
  - its value is only relevant for the second particle in the cascade
- getter and setter for `G4int` *CaseNumber*
  - check the current particle type and the previous particle type to determine case (this is done in `G4VGammaDeexcitation` and saved here.
  * $>= 0$: only for second and subsequent particles
    ** case 0: $\gamma - \gamma$
    ** case 1: $\gamma - e^-$ (previous particle was a gamma, current particle is an electron)
    ** $e^- - \gamma$
    ** $e^- - e^-$
    -1 : for transitions which are not γ or e^- AND are not the first product
    ** emitted isotropically
  * -2 : for first particle (emitted isotropically)
  * -3 : for *NULL* transitions

Source File

Unchanged from Evan's version.

G4VGammaDeexcitation

This is where new particles in the cascade are generated.
This is also where `GetThetaFromWTheta` is called and the correlation is applied to the newly generated particle.

Header File

Unchanged from Evan's version

Source File

Additions to Evan's version:

- `DoDecay` method:
  - placed a check here to see if the product list is empty or not. This tells us whether or not the particle being generated is the first in the cascade (stored as `G4bool` *FirstProduct* in `G4NuclearLevelStore`)
- Case Selection
  - see *G4int CaseNumber* above
  - determine which case the currently generated particle belongs to, and pass this number to `G4DiscreteGammaTransition::GetThetaFromWTheta`
  - first particle in cascade is emitted isotropically
  - all non-γ and non-e^- particles are emitted isotropically
- subtract bond energy of electrons here (INSTEAD of in `G4DiscreteGammaTransition`)
- takes the θ produced from `GetThetaFromWTheta` and applies it to a polarization vector to simulate the correlation

---

G4RadioactiveDecay

This is where radioactive decay modes are accessed and the correct decay path is chosen for the source particle.

The entire module is unchanged from the original `Geant4` source files. Some deletions where made from Evan's version to get it back to this original state (removed *FirstGammaDecay* checks). This is only included in the package in case the user has Evan's extension already installed.

G4RadioactiveDecayMessenger

Not entirely sure what this module does.

The entire module is unchanged from Evan's version. Only included in this package so that the user does not have to download both the Gamma-Gamma *and* the Gamma-Electron extensions.