Synthèse des modifications du code Geant4

Différenciation des primaires directs
et des primaires redirigés par diffusion Compton

Simulation MiniX – Cône concentrateur en graphite

February 9, 2026

---

1 Contexte et motivation

Dans Geant4, lorsqu'un photon γ primaire subit une diffusion Compton, le **photon diffusé continue comme le même track** (même `trackID`, `parentID = 0`). Seul l'électron de recul est créé en tant que secondaire (`creator_process = "compt"`).

Conséquence : au ScorePlane ($z = 18$ mm), un photon redirigé par Compton dans le cône graphite est **indiscernable** d'un photon transmis directement à travers le canal. Les deux apparaissent comme `parentID = 0`, `creator_process = "primary"`.

Objectif : ajouter un marquage (flag) sur le track primaire pour distinguer les trois populations au ScorePlane :

Cat.	Description	`parentID`	`compton_in_cone`
A	Primaire transmis directement	0	0
B	Primaire redirigé par Compton	0	1
C	Secondaire (e^-, γ Brem…)	≥ 1	—

2 Fichiers modifiés

Fichier	Répertoire	Nature de la modification
`MyTrackInfo.hh`	`include/`	Ajout champs Compton
`MyTrackInfo.cc`	`src/`	Initialisation des nouveaux champs
`SteppingAction.cc`	`src/`	Détection Compton dans le cône
`AnalysisManagerSetup.cc`	`src/`	5 nouvelles colonnes ntuple
`SurfaceSpectrumSD.cc`	`src/`	Lecture flag + remplissage ntuple

Aucun autre fichier n'est modifié. Les fichiers `.hh` et `.cc` non listés ci-dessus restent inchangés.

3 Détail des modifications

3.1 `MyTrackInfo.hh / .cc` — Stockage de l'information Compton

Quatre nouveaux champs privés sont ajoutés à la classe `MyTrackInfo` (qui hérite de `G4VUserTrackInformation` et est attachée à chaque track) :

Champ	Type	Init.	Description
`fComptonInCone`	`G4bool`	`false`	Flag : ≥ 1 Compton dans le cône
`fNComptonInCone`	`G4int`	`0`	Nombre total de Compton dans le cône
`fLastComptonPos`	`G4ThreeVector`	`(0,0,0)`	Position de la dernière diffusion
`fLastComptonEkin`	`G4double`	`0.`	E_{kin} avant la dernière diffusion

Accesseurs publics correspondants : `Set/Get/Has` + `IncrementNComptonInCone()`.
L'include `G4ThreeVector.hh` est ajouté en en-tête.

3.2 `SteppingAction.cc` — Détection du Compton dans le cône

Un nouveau bloc est inséré dans `UserSteppingAction()`, juste après la récupération des noms de matériaux (`matNamePre`, `matNamePost`).

Condition de déclenchement (les 3 conditions doivent être vraies simultanément) :

1. Le processus défini au post-step est `"compt"`
2. Le volume logique au pré-step est `"logicConeCompton"`
3. Le track est un primaire (`parentID == 0`)

Actions :

1. `info->SetComptonInCone(true)`
2. `info->IncrementNComptonInCone()`
3. `info->SetLastComptonPos(postPoint->GetPosition())`
4. `info->SetLastComptonEkin(prePoint->GetKineticEnergy())`

Log : tag `[STEP][COMPTON_IN_CONE]`, 100 premiers puis 1 sur 10 000. Protégé par mutex (`G4AutoLock`) pour la compatibilité multi-thread.

Extrait de `SteppingAction.cc` (bloc ajouté)

```
const G4VProcess* procDefined = postPoint->GetProcessDefinedStep();
if (procDefined && track->GetParentID() == 0
    && procDefined->GetProcessName() == "compt"
    && namePre == "logicConeCompton")
{
    MyTrackInfo* info =
        static_cast<MyTrackInfo*>(track->GetUserInformation());
    if (info) {
        info->SetComptonInCone(true);
        info->IncrementNComptonInCone();
        info->SetLastComptonPos(postPoint->GetPosition());
        info->SetLastComptonEkin(prePoint->GetKineticEnergy());
    }
}
```

3.3 `AnalysisManagerSetup.cc` — Nouvelles colonnes du ntuple

Cinq colonnes sont ajoutées au ntuple `plane_passages` (indices 10 à 14), juste avant le `FinishNtuple` :

Colonnes ajoutées dans `AnalysisManagerSetup.cc`

```
// [ADD] Colonnes Compton dans le cone graphite
CreateNtupleIColumn(id, "compton_in_cone"); // 10
CreateNtupleIColumn(id, "n_compton_cone");  // 11
CreateNtupleDColumn(id, "compton_x_mm");    // 12
CreateNtupleDColumn(id, "compton_y_mm");    // 13
CreateNtupleDColumn(id, "compton_z_mm");    // 14
```

3.4 `SurfaceSpectrumSD.cc` — Lecture du flag et remplissage

Dans `ProcessHits()`, après le calcul de `is_secondary` :

1. Récupération du `MyTrackInfo` attaché au track
2. Lecture de `HasComptonInCone()`, `GetNComptonInCone()`, `GetLastComptonPos()`
3. Remplissage des colonnes 10–14 du ntuple

4. Log `[ScorePlane1][COMPTON_REDIRECTED]` (200 premiers puis 1 sur 5 000)
   L'include `MyTrackInfo.hh` est ajouté en en-tête du fichier.

4 Structure complète du ntuple `plane_passages`

Le ntuple `plane_passages` enregistre chaque particule quittant le ScorePlane ($z = 18$ mm) dans la direction $+z$. Ci-dessous la structure complète après modification (les colonnes 10–14 sont nouvelles) :

Table 1: Colonnes du ntuple `plane_passages` (ScorePlane, $z = 18$ mm). Les colonnes 0–9 sont inchangées. Les colonnes 10–14 (sur fond vert) sont ajoutées.

Col.	Type	Nom	Unité	Description
0	`int`	`pdg`	—	Code PDG (22 = γ, 11 = e^-)
1	`string`	`name`	—	Nom de la particule
2	`int`	`is_secondary`	—	0 = primaire, 1 = secondaire
3	`double`	`x_mm`	mm	Position x au plan
4	`double`	`y_mm`	mm	Position y au plan
5	`double`	`z_mm`	mm	Position z au plan (≈ 18)
6	`double`	`ekin_keV`	keV	Énergie cinétique au plan
7	`int`	`trackID`	—	Identifiant du track
8	`int`	`parentID`	—	ID du parent (0 = primaire)
9	`string`	`creator_process`	—	Processus créateur
addgreen 10	`int`	`compton_in_cone`	—	1 si ≥ 1 Compton dans le cône, 0 sinon
addgreen 11	`int`	`n_compton_cone`	—	Nombre de diffusions Compton dans le cône
addgreen 12	`double`	`compton_x_mm`	mm	x de la dernière diffusion Compton
addgreen 13	`double`	`compton_y_mm`	mm	y de la dernière diffusion Compton
addgreen 14	`double`	`compton_z_mm`	mm	z de la dernière diffusion Compton

Notes :
- Les colonnes 12–14 valent $(0, 0, 0)$ si `compton_in_cone = 0`.
- Si le photon subit plusieurs Compton dans le cône (`n_compton_cone` > 1), seule la position de la *dernière* diffusion est enregistrée.
- Les ntuples des autres plans (ScorePlane2, 3, 5) ne sont **pas modifiés**.

5 Classification des particules au ScorePlane

Avec les nouvelles colonnes, chaque entrée du ntuple peut être classée sans ambiguïté :

Table 2: Classification des particules au ScorePlane.

Catégorie	`parentID`	`is_sec`	`compton_in_cone`	Filtre ROOT
Primaire transmis directement ($\theta < 3.4$)	0	0	0	`is_secondary==0 && compton_in_cone==0`
Primaire redirigé Compton	0	0	1	`is_secondary==0 && compton_in_cone==1`
Secondaire (photoélectron, Brem…)	≥ 1	1	—	`is_secondary==1`

6 Nouveaux tags dans le fichier `.log`

Table 3: Tags de log ajoutés.

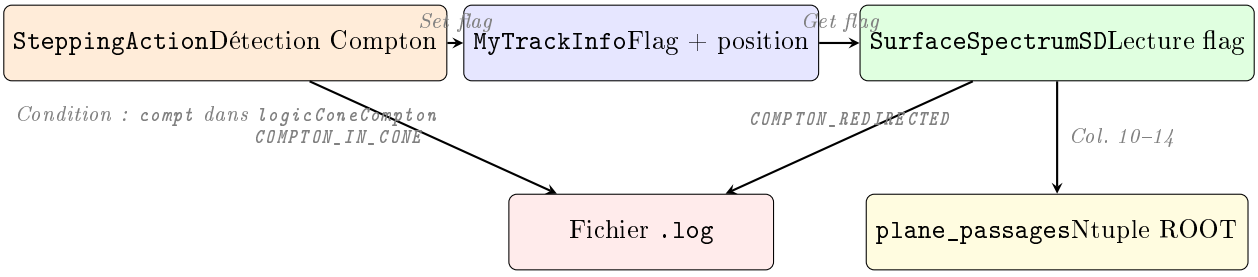
Tag	Source	Fréquence	Contenu
`[STEP][COMPTON_IN_CONE]`	`SteppingAction`	100 premiers, puis 1/10 000	Événement, trackID, n_{compt}, E_{avant}, $E_{\text{après}}$, position
`[ScorePlane1][COMPTON_REDIRECTED]`	`SurfaceSpectrumSD`	200 premiers, puis 1/5 000	Particule, trackID, E au plan, n_{compt}, position Compton, position au plan

7 Exemples d'analyse ROOT

Exemples de commandes ROOT/TTree::Draw

```
// Spectre en energie des photons transmis directement
plane_passages->Draw("ekin_keV",
    "pdg==22 && is_secondary==0 && compton_in_cone==0")

// Spectre en energie des photons rediriges par Compton
plane_passages->Draw("ekin_keV",
    "pdg==22 && is_secondary==0 && compton_in_cone==1")

// Distribution en z des lieux de diffusion Compton
plane_passages->Draw("compton_z_mm",
    "compton_in_cone==1")

// Distribution radiale r = sqrt(x^2+y^2) des Compton
plane_passages->Draw("sqrt(compton_x_mm^2+compton_y_mm^2)",
    "compton_in_cone==1")

// Perte d'energie fractionnelle par Compton (si >0)
plane_passages->Draw("n_compton_cone",
    "compton_in_cone==1")

// Comptages
plane_passages->Draw("compton_in_cone",
    "pdg==22 && is_secondary==0")
// -> bin 0 = transmis directs, bin 1 = Compton rediriges
```

8 Schéma synoptique du flux de données



9 Compatibilité et remarques

- **Multi-thread :** le log dans `SteppingAction` est protégé par `G4AutoLock`. Les compteurs statiques dans `SurfaceSpectrumSD` sont thread-local (`static` dans une fonction membre appelée par thread).
- **Ntuple :** les colonnes 0–9 sont inchangées ; les fichiers ROOT existants restent lisibles. Les nouveaux fichiers auront 15 colonnes au lieu de 10.
- **Autres plans :** ScorePlane2, 3 et 5 ne sont pas modifiés. Pour les instrumenter de façon analogue, il suffirait de répliquer la lecture du `MyTrackInfo` dans les SD correspondants.
- **Performances :** impact négligeable — un test de chaîne (`=="compt"`) et un cast statique par step.
- **Recompilation :** seuls les 5 fichiers listés doivent être remplacés. Un `make` incrémental suffit.