

Documentation des Histogrammes et Ntuples

Simulation Geant4 – Projet MiniX
Variables et conditions de remplissage

January 29, 2026

Contents

| | | |
|----------|--------------------------------------------------------------------|-----------|
| 1 | Histogrammes 1D | 2 |
| 1.1 | Vue d'ensemble | 2 |
| 1.2 | Histogrammes d'émission (H0, H1, H2) | 2 |
| 1.2.1 | H0 – Énergie des gammas primaires | 2 |
| 1.2.2 | H1 – Angle theta des gammas primaires | 2 |
| 1.2.3 | H2 – Angle phi des gammas primaires | 3 |
| 1.3 | Histogrammes de dose – Run complet (H3, H5–H9) | 3 |
| 1.3.1 | Formule de calcul de la dose | 3 |
| 1.3.2 | H3 – Dose totale dans l'eau (run complet) | 3 |
| 1.3.3 | H5–H9 – Dose par anneau (run complet) | 4 |
| 1.4 | Histogrammes de dose – Par 1000 événements (H4, H10–H14) | 4 |
| 1.4.1 | H4 – Dose totale dans l'eau (par 1000 événements) | 4 |
| 1.4.2 | H10–H14 – Dose par anneau (par 1000 événements) | 5 |
| 1.5 | Flux de données pour les histogrammes de dose | 5 |
| 2 | Ntuples | 7 |
| 2.1 | Vue d'ensemble | 7 |
| 2.2 | Ntuple 0 : <code>plane_passages</code> | 7 |
| 2.3 | Ntuples 1–2 : <code>ScorePlane2/3_passages</code> | 7 |
| 2.4 | Ntuple 3 : <code>WaterRings_passages</code> | 8 |
| 2.5 | Ntuple 4 : <code>ScorePlane5_passages</code> | 8 |
| 3 | Récapitulatif des fichiers sources | 10 |
| 4 | Exemple de run | 10 |

1 Histogrammes 1D

1.1 Vue d'ensemble

| ID | Nom | Bins | Min | Max |
|----|--------------------|------|-------|------------|
| 0 | E_emission | 150 | 0 | 50 keV |
| 1 | theta_emission | 180 | 0° | 180° |
| 2 | phi_emission | 90 | -180° | 180° |
| 3 | Dose_total_run | 200 | 0 | 1 μ Gy |
| 4 | Dose_total_1000evt | 200 | 0 | 1 μ Gy |
| 5 | Dose_ring0_run | 200 | 0 | 1 μ Gy |
| 6 | Dose_ring1_run | 200 | 0 | 1 μ Gy |
| 7 | Dose_ring2_run | 200 | 0 | 1 μ Gy |
| 8 | Dose_ring3_run | 200 | 0 | 1 μ Gy |
| 9 | Dose_ring4_run | 200 | 0 | 1 μ Gy |
| 10 | Dose_ring0_1000evt | 200 | 0 | 1 μ Gy |
| 11 | Dose_ring1_1000evt | 200 | 0 | 1 μ Gy |
| 12 | Dose_ring2_1000evt | 200 | 0 | 1 μ Gy |
| 13 | Dose_ring3_1000evt | 200 | 0 | 1 μ Gy |
| 14 | Dose_ring4_1000evt | 200 | 0 | 1 μ Gy |

1.2 Histogrammes d'émission (H0, H1, H2)

1.2.1 H0 – Énergie des gammas primaires

| Paramètre | Valeur |
|-------------|-----------------------------------------------------|
| Nom | E_emission |
| Description | Énergie cinétique des gammas primaires à l'émission |
| Bins | 150 |
| Range | [0, 50] keV |
| Unité | keV |

Condition de remplissage :

- **Quand** : À chaque génération d'un gamma primaire
- **Où** : PrimaryGeneratorAction1::GeneratePrimaries() et PrimaryGeneratorAction2::GeneratePrimaries()
- **Variable** : energy = fParticleGun->GetParticleEnergy()

```
analysisManager->FillH1(0, energy); // energy en keV
```

1.2.2 H1 – Angle theta des gammas primaires

| Paramètre | Valeur |
|-------------|---------------------------------------------------|
| Nom | theta_emission |
| Description | Angle polaire θ de la direction d'émission |
| Bins | 180 |
| Range | [0, 180] degrés |
| Unité | degrés |

Condition de remplissage :

- **Quand** : À chaque génération d'un gamma primaire
- **Où** : PrimaryGeneratorAction1/2::GeneratePrimaries()

- **Variable** : `thetaDeg = acos(cosTheta) / deg`

```
analysisManager->FillH1(1, thetaDeg); // theta en degres
```

1.2.3 H2 – Angle phi des gammas primaires

| Paramètre | Valeur |
|-------------|--------------------------------------------------|
| Nom | <code>phi_emission</code> |
| Description | Angle azimutal ϕ de la direction d'émission |
| Bins | 90 |
| Range | $[-180, 180]$ degrés |
| Unité | degrés |

Condition de remplissage :

- **Quand** : À chaque génération d'un gamma primaire
- **Où** : `PrimaryGeneratorAction1/2::GeneratePrimaries()`
- **Variable** : `phiDeg = phi / deg`

```
analysisManager->FillH1(2, phiDeg); // phi en degres
```

1.3 Histogrammes de dose – Run complet (H3, H5–H9)

1.3.1 Formule de calcul de la dose

$$\text{Dose } [\mu\text{Gy}] = \frac{E_{\text{dep}} [\text{keV}] \times 1.602 \times 10^{-7}}{\text{masse } [\text{g}]} \quad (1)$$

Masses des anneaux d'eau (épaisseur $Z = 3$ mm, $\rho_{\text{eau}} = 1$ g/cm³) :

| Anneau | Rayon (mm) | Volume (mm ³) | Masse (g) |
|--------------|------------|------------------------------------|-----------|
| 0 | 0 – 2 | $\pi \times 4 \times 3 = 37.70$ | 0.03770 |
| 1 | 2 – 4 | $\pi \times 12 \times 3 = 113.10$ | 0.11310 |
| 2 | 4 – 6 | $\pi \times 20 \times 3 = 188.50$ | 0.18850 |
| 3 | 6 – 8 | $\pi \times 28 \times 3 = 263.89$ | 0.26389 |
| 4 | 8 – 10 | $\pi \times 36 \times 3 = 339.29$ | 0.33929 |
| Total | 0 – 10 | $\pi \times 100 \times 3 = 942.48$ | 0.94248 |

1.3.2 H3 – Dose totale dans l'eau (run complet)

| Paramètre | Valeur |
|-------------|---------------------------------------------------|
| Nom | <code>Dose_total_run</code> |
| Description | Dose totale déposée dans toute l'eau du container |
| Bins | 200 |
| Range | $[0, 1]$ μGy |
| Unité | μGy |

Condition de remplissage :

- **Quand** : Une seule fois, à la fin du run
- **Où** : `RunAction::EndOfRunAction()`

- **Variable** : fTotalEdepWater (énergie accumulée sur tout le run)
- **Nombre d'entrées** : 1 par run

```
G4double dose_total_run = fTotalEdepWater * keV_to_uGy_per_gram /
    kMassTotalWater;
am->FillH1(3, dose_total_run);
```

1.3.3 H5–H9 – Dose par anneau (run complet)

| Paramètre | Valeur |
|-------------|-----------------------------------------------|
| Noms | Dose_ring0_run à Dose_ring4_run |
| Description | Dose déposée dans chaque anneau (run complet) |
| Bins | 200 |
| Range | [0, 1] μGy |
| Unité | μGy |

Condition de remplissage :

- **Quand** : Une seule fois, à la fin du run
- **Où** : RunAction::EndOfRunAction()
- **Variable** : fTotalEdepRing[i] pour $i \in [0, 4]$
- **Nombre d'entrées** : 1 par run pour chaque histogramme

```
for (G4int i = 0; i < kNbWaterRings; i++) {
    G4double dose_ring_run = fTotalEdepRing[i] * keV_to_uGy_per_gram /
        kMassRing[i];
    am->FillH1(5 + i, dose_ring_run); // H5 a H9
}
```

1.4 Histogrammes de dose – Par 1000 événements (H4, H10–H14)

1.4.1 H4 – Dose totale dans l'eau (par 1000 événements)

| Paramètre | Valeur |
|-------------|-------------------------------------------------------|
| Nom | Dose_total_1000evt |
| Description | Distribution des doses par tranche de 1000 événements |
| Bins | 200 |
| Range | [0, 1] μGy |
| Unité | μGy |

Condition de remplissage :

- **Quand** : Tous les 1000 événements (eventID % 1000 == 0)
- **Où** : RunAction::CheckAndFillDoseHistograms()
- **Variable** : fEdepWater1000 (énergie accumulée sur 1000 evt)
- **Nombre d'entrées** : $N_{\text{events}}/1000$ par run
- **Réinitialisation** : fEdepWater1000 = 0 après chaque remplissage

```

if (eventID > 0 && eventID % 1000 == 0 && eventID != fLastHistoFillEvent) {
    fLastHistoFillEvent = eventID;

    G4double dose_total = fEdepWater1000 * keV_to_uGy_per_gram /
        kMassTotalWater;
    analysisManager->FillH1(4, dose_total);

    fEdepWater1000 = 0.0; // Reinitialisation de l'accumulateur
}

```

1.4.2 H10–H14 – Dose par anneau (par 1000 événements)

| Paramètre | Valeur |
|-------------|--------------------------------------------------------------|
| Noms | Dose_ring0_1000evt à Dose_ring4_1000evt |
| Description | Distribution des doses par anneau et par tranche de 1000 evt |
| Bins | 200 |
| Range | [0, 1] μGy |
| Unité | μGy |

Condition de remplissage :

- **Quand** : Tous les 1000 événements ($\text{eventID} \% 1000 == 0$)
- **Où** : `RunAction::CheckAndFillDoseHistograms()`
- **Variable** : `fEdepRing1000[i]` pour $i \in [0, 4]$
- **Nombre d'entrées** : $N_{\text{events}}/1000$ par run pour chaque histo
- **Réinitialisation** : `fEdepRing1000[i] = 0` après chaque remplissage

```

if (eventID > 0 && eventID % 1000 == 0 && eventID != fLastHistoFillEvent) {
    for (G4int i = 0; i < kNbWaterRings; i++) {
        G4double dose_ring = fEdepRing1000[i] * keV_to_uGy_per_gram /
            kMassRing[i];
        analysisManager->FillH1(10 + i, dose_ring); // H10 à H14

        fEdepRing1000[i] = 0.0; // Reinitialisation
    }
}

```

1.5 Flux de données pour les histogrammes de dose

1. `SteppingAction::UserSteppingAction()`

- Détecte les dépôts d'énergie dans les volumes `logicWaterRing0` à `logicWaterRing4`
- Appelle `fEventAction->AddEdepToRing(ringIndex, edep)`

2. `EventAction::AddEdepToRing()`

- Accumule l'énergie dans `fEdepRing[ringIndex]` et `fEdepTotalWater`
- Ces variables sont réinitialisées au début de chaque événement

3. `EventAction::EndOfEventAction()`

- Transmet les énergies à `RunAction::AddEdepFromEvent()`
- Appelle `RunAction::CheckAndFillDoseHistograms()`

4. **RunAction::AddEdepFromEvent()**

- Accumule dans `fTotalEdepRing[i]` et `fTotalEdepWater` (pour fin de run)
- Accumule dans `fEdepRing1000[i]` et `fEdepWater1000` (pour 1000 evt)

5. **RunAction::CheckAndFillDoseHistograms()**

- Si `eventID % 1000 == 0` : remplit H4 et H10–H14, puis réinitialise les accumulateurs 1000 evt

6. **RunAction::EndOfRunAction()**

- Remplit H3 et H5–H9 avec les totaux du run

2 Ntuples

2.1 Vue d'ensemble

| ID | Nom | Position Z | Volume |
|----|----------------------|------------|-----------------|
| 0 | plane_passages | 18 mm | ScorePlane1 |
| 1 | ScorePlane2_passages | 28 mm | ScorePlane2 |
| 2 | ScorePlane3_passages | 38 mm | ScorePlane3 |
| 3 | WaterRings_passages | 65–68 mm | Couronnes d'eau |
| 4 | ScorePlane5_passages | 70 mm | ScorePlane5 |

2.2 Ntuple 0 : plane_passages

| Paramètre | Valeur |
|-------------|-----------------------------------|
| Nom | plane_passages |
| Description | Traversées +Z du plan ScorePlane1 |
| Position | $z = 18$ mm |

Variables :

| Col. | Variable | Type | Description |
|------|-----------------|--------|------------------------------|
| 0 | x_mm | Double | Position X (mm) |
| 1 | y_mm | Double | Position Y (mm) |
| 2 | z_mm | Double | Position Z (mm) |
| 3 | ekin_keV | Double | Énergie cinétique (keV) |
| 4 | pdg | Int | Code PDG de la particule |
| 5 | name | String | Nom de la particule |
| 6 | trackID | Int | ID de la trace |
| 7 | parentID | Int | ID du parent (0 si primaire) |
| 8 | creator_process | String | Processus créateur |

Condition de remplissage :

- **Quand** : À chaque traversée du plan dans la direction +Z
- **Où** : `SurfaceSpectrumSD::ProcessHits()`
- **Condition** : `postPoint->GetStepStatus() == fGeomBoundary` et direction +Z

2.3 Ntuples 1–2 : ScorePlane2/3_passages

| Paramètre | Valeur |
|-------------|----------------------------------------------------|
| Noms | ScorePlane2_passages, ScorePlane3_passages |
| Description | Traversées +Z des plans ScorePlane2 et ScorePlane3 |
| Positions | $z = 28$ mm et $z = 38$ mm |

Variables :

| Col. | Variable | Type | Description |
|------|-----------------|--------|------------------------------|
| 0 | pdg | Int | Code PDG de la particule |
| 1 | name | String | Nom de la particule |
| 2 | is_secondary | Int | 0 = primaire, 1 = secondaire |
| 3 | x_mm | Double | Position X (mm) |
| 4 | y_mm | Double | Position Y (mm) |
| 5 | ekin_keV | Double | Énergie cinétique (keV) |
| 6 | trackID | Int | ID de la trace |
| 7 | parentID | Int | ID du parent (0 si primaire) |
| 8 | creator_process | String | Processus créateur |

Condition de remplissage :

- **Quand :** À chaque traversée du plan dans la direction +Z
- **Où :** ScorePlane2SD::ProcessHits() et ScorePlane3SD::ProcessHits()
- **Condition :** postPoint->GetStepStatus() == fGeomBoundary et direction +Z

2.4 Ntuple 3 : WaterRings_passages

| Paramètre | Valeur |
|-------------|-----------------------------------------------------|
| Nom | WaterRings_passages |
| Description | Traversées dans les 5 couronnes d'eau concentriques |
| Position | $z = 65$ à 68 mm |

Variables : Identiques aux ntuples 1–2.

Condition de remplissage :

- **Quand :** À chaque traversée d'une couronne d'eau
- **Où :** ScorePlane4SD::ProcessHits()
- **Note :** Les 5 couronnes partagent le même ntuple

Identification des couronnes en post-traitement :

La position radiale $r = \sqrt{x^2 + y^2}$ permet d'identifier la couronne :

| Couronne | Condition sur r (mm) |
|----------|------------------------|
| 0 | $0 \leq r < 2$ |
| 1 | $2 \leq r < 4$ |
| 2 | $4 \leq r < 6$ |
| 3 | $6 \leq r < 8$ |
| 4 | $8 \leq r < 10$ |

2.5 Ntuple 4 : ScorePlane5_passages

| Paramètre | Valeur |
|-------------|---------------------------------------------------------------|
| Nom | ScorePlane5_passages |
| Description | Traversées +Z du plan ScorePlane5 (après les couronnes d'eau) |
| Position | $z = 70$ mm |

Variables : Identiques aux ntuples 1–2.

Condition de remplissage :

- **Quand** : À chaque traversée du plan dans la direction +Z
- **Où** : `ScorePlane5SD::ProcessHits()`
- **Condition** : `postPoint->GetStepStatus() == fGeomBoundary` et direction +Z

3 Récapitulatif des fichiers sources

| Fichier | Rôle |
|----------------------------|---------------------------------------------------|
| AnalysisManagerSetup.cc | Création des 15 histogrammes et 5 ntuples |
| PrimaryGeneratorAction1.cc | Remplissage H0, H1, H2 (mode 1) |
| PrimaryGeneratorAction2.cc | Remplissage H0, H1, H2 (mode 2) |
| SteppingAction.cc | Détection dépôts énergie dans anneaux |
| EventAction.cc/hh | Accumulation énergie par événement |
| RunAction.cc/hh | Accumulation run, calcul dose, remplissage H3–H14 |
| SurfaceSpectrumSD.cc | Remplissage ntuple 0 |
| ScorePlane2SD.cc | Remplissage ntuple 1 |
| ScorePlane3SD.cc | Remplissage ntuple 2 |
| ScorePlane4SD.cc | Remplissage ntuple 3 (WaterRings) |
| ScorePlane5SD.cc | Remplissage ntuple 4 |

4 Exemple de run

Pour un run de 10 000 événements :

| Histogramme | Nb entrées | Signification |
|------------------------------|------------|------------------------------------|
| H0 (E_emission) | 10 000 | Une entrée par gamma primaire |
| H1 (theta_emission) | 10 000 | Une entrée par gamma primaire |
| H2 (phi_emission) | 10 000 | Une entrée par gamma primaire |
| H3 (Dose_total_run) | 1 | Dose totale du run |
| H5–H9 (Dose_ringX_run) | 1 chacun | Dose par anneau (run) |
| H4 (Dose_total_1000evt) | 10 | Distribution des doses par tranche |
| H10–H14 (Dose_ringX_1000evt) | 10 chacun | Distribution par anneau et tranche |