2020 年度卒業論文

中性子過剰核の構造解明のための新たな β 遅発中性子検出器開発

川畑研究室 吉岡篤志

2021 年 4 月 27 日

概要

原子核、特に中性子が陽子に比べて非常に多い不安定な原子核（中性子過剰核）では安定核とは異なる構造が発見されてきた。その構造（形や運動）を解明する為、β 遅延中性子崩壊を用いて研究している。そのため、β 線、γ 線、中性子を測定して、崩壊様式を組み立て、その構造を議論する。β 遅発中性子は、数十 keV〜数 MeV と低エネルギーである為、未だ従来の検出器には課題が多々ある。そこで、β 遅発中性子を測定する新たな中性子検出器の開発を行った。

新たな中性子検出器には、従来使用していた magro と呼ばれる大型検出器と hile と呼ばれる小型検出器の２種類の中性子検出器の、双方の長所を満たすようなものが求められる。大型の magro は、高エネルギー中性子の検出効率が大きいが、低エネルギー中性子の検出効率が小さい。また、集光率の場所依存性が見られてしまうことや、エネルギー分解能が悪いという短所がある。対して小型の hile は、低エネルギー中性子の検出効率が大きいが、高エネルギー中性子の検出効率が小さい。集光率の場所依存性とエネルギー分解能に関しては問題ない。そこで、新たな中性子検出器には、magro 以上に高エネルギーに対して検出効率が大きく、且つ hile 以上に低エネルギーに対して検出効率が大きいこと、hile と同程度に中性子エネルギーの精度良い測定が可能なこと、集光率の場所依存性がないことが求められる。

以上に掲げた特徴が満たされるような検出器の作成を目指し、必要な条件や形状を導き出した。その結果、（１）低エネルギーの検出効率を大きくする為には、ノイズを落とす為に PMT を 2 個使用した同時計測を行うことと集光率の増大の為に小型にすることが重要である。（２）高エネルギーの検出効率を大きくする為には、シンチレータが厚いことが重要である。（３）中性子エネルギーの精度良い測定の為には、長い飛行距離と検出器が小型であることが重要である。（４）立体角を確保する為には、シンチレータの 形を六角形にし、ライトガイドをシンチレータの後方に接続することがそれぞれ重要だと理解した。

また、Geant4 によるシミュレーションを基に、集光率と集光率の場所依存性について評価し、細かい形状や反射材を決定し、最終的な検出器をデザインして製作した。

この新型中性子検出器を、β 線源と γ 線源を用いて、時間分解能とエネルギー閾値についての初期段階の性能評価実験を行った。その結果、新型中性子検出器の時間分解能が hile と同程度であること、またエネルギー閾値が hile より更に小さくなることがわかった。よって、従来の中性子検出器に比べて、低エネルギー中性子の検出効率の増大が期待できることがわかった。

目次

第 1 章

序論

1.1　中性子過剰核の構造解明に向けて

本グループは、原子核、特に中性子が陽子に比べて非常に多いような不安定な原子核について、その形や運動を解明していくことを目標としている。原子核の形や運動は、その原子核の準位構造から理解することができる。特に中性子過剰な不安定核の準位構造を調べてみると、その励起状態には複数の変形状態が共存していることが、これまでの研究でわかっている。例として、^{31}Mg の準位構造を示したものが図 1.1[1] である。よって、中性子過剰な不安定核の準位構造を調べることはとても興味深い研究である。

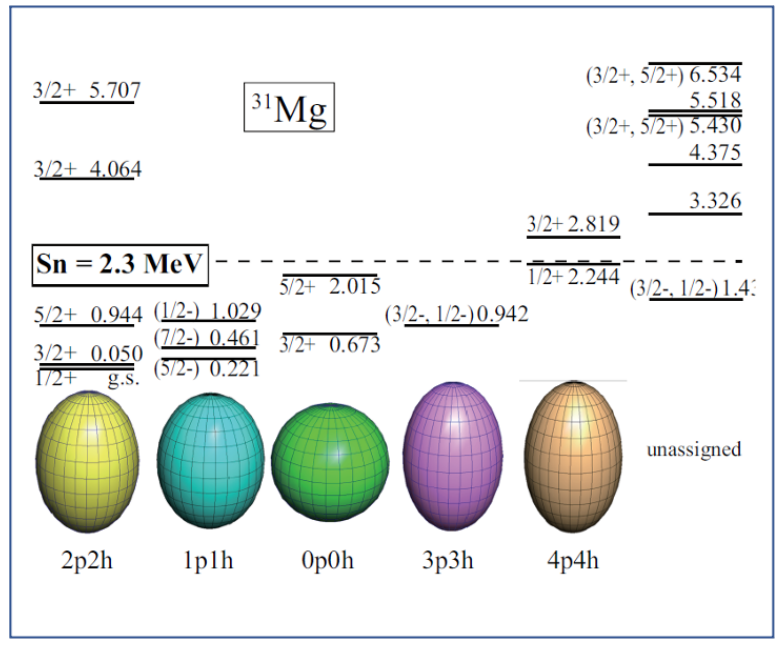


図 1.1　^{31}Mg の変形共存

1.2　β^- 崩壊

安定核近傍の原子核が β^- 崩壊後に、励起状態に崩壊した場合、γ 線などを放出して基底状態に脱励起する。それに対し、安定核から遠く離れた中性子過剰な原子核の場合は、Q_β が大きく、中性子１個を剥ぎ取るエネルギーである中性子分離エネルギー S_n が小さいため、中性子非束縛状態への β^- 崩壊も可能となる。中性子非束縛状態から脱励起するとき、β 遅発中性子を放出する。

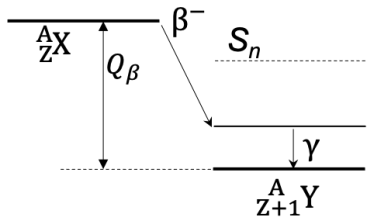


図 1.2　安定核近傍の原子核の β^- 崩壊

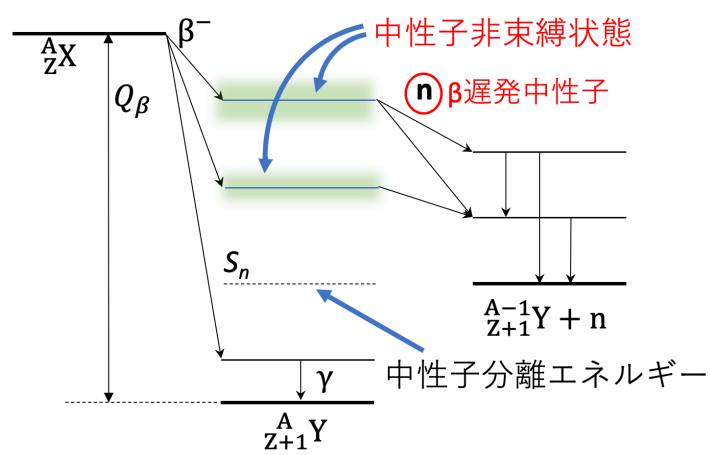


図 1.3　中性子過剰な原子核の β^- 崩壊

1.3　β 遅発中性子

β 遅発中性子とは、前項で触れたように、中性子過剰な原子核が中性子非束縛状態へ β^- 崩壊し、そこから脱励起するときに生じる中性子である。β 遅発中性子の特徴の一つがそのエネルギーである。およそ数十 keV〜数 MeV の広い範囲をもち、他の実験で一般的に測定対象となるような中性子に比べて低エネルギーである。よって、他の実験で使用される中性子検出器に比べて、低エネルギー中性子に対して感度のよい中性子検出器が必要となる。

第 2 章

中性子の測定方法

2.1　中性子の測定原理

　中性子は電荷を持たない粒子である。そのため電離作用がなく、エネルギーを直接測定することが困難である。そこで、中性子と原子核の弾性散乱を利用する。中性子がシンチレータに入射されると、ある確率でシンチレータ内の陽子などの原子核と弾性散乱を起こす。弾性散乱によってエネルギーを持った反跳陽子 (反跳原子核) は、エネルギーに依存したシンチレーション光を放出することで、そのエネルギーの一部を落とす。このシンチレーション光を検出することで中性子を検出する。このとき、得られる波高は、反跳陽子のエネルギーの大きさに比例する。

2.2　TOF 法

　前項で述べたように、中性子の測定には陽子との弾性散乱を利用する。このとき、衝突の角度によって、反跳陽子が得るエネルギー E_p は変化し、0 から中性子エネルギー E_n の間で連続分布する。よって、中性子のエネルギーを得られる波高から求めることはできない。そこで、TOF(Time of Flight) 法 [飛行時間法] を用いる。

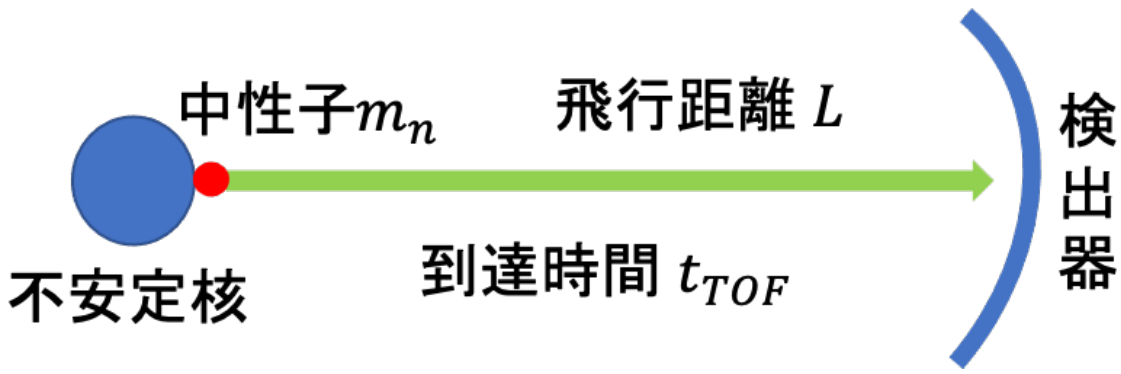


図 2.1　TOF 法の模式図

　TOF 法とは、中性子の飛行距離 L と、飛行に要した時間 t_{TOF} を測定することで、中性子エネル

ギー E_n を式 (2.1) を用いて決定する方法である。

$$E_n = \frac{1}{2} m_n \left(\frac{L}{t_{TOF}} \right)^2 \tag{2.1}$$

エネルギーを直接求める場合、通常、得られた波形全体を積分することで求めるが、TOF 法の場合は、波形のピークを用いる為、その分誤差が小さくなり、より正確に求めることができる。

TOF 法のエネルギー分解能 ΔE は、式 (2.2) のように表される。

$$\Delta E = 2E_n \sqrt{\left(\frac{\Delta L}{L} \right)^2 + \left(\frac{\Delta t}{t_{TOF}} \right)^2} \tag{2.2}$$

測定では、ある距離 L のところに検出器を固定して、その時の飛行時間 t_{TOF} を測定する為、検出器の設置の際の測定誤差 ΔL_{det} を除いて、その他全ての誤差は時間分解能 Δt_{TOF} として表れる。よって、検出器開発において考慮する誤差は、この時間分解能 Δt_{TOF} である。

$$\Delta E = 2E_n \sqrt{\left(\frac{\Delta L_{det}}{L} \right)^2 + \left(\frac{\Delta t_{TOF}}{t_{TOF}} \right)^2} \tag{2.3}$$

このとき、

$$\frac{\Delta L_{det}}{L} \ll \frac{\Delta t_{TOF}}{t_{TOF}} \tag{2.4}$$

である為、以下のように近似できる。

$$\Delta E \simeq 2E_n \left(\frac{\Delta t_{TOF}}{t_{TOF}} \right) \tag{2.5}$$

よって、時間分解能 Δt_{TOF} は、エネルギー分解能 ΔE に比例する。今後、測定するエネルギーの誤差について考えるとき、時間分解能 Δt_{TOF} という言葉を統一して用いる。式としては式 (2.2) を用いる。

第 3 章

新型中性子検出器に求められる性能

3.1　従来の中性子検出器

図 3.1　従来の中性子検出器

従来、本グループは、magro と呼ばれる大型検出器と hile と呼ばれる小型検出器の 2 種類の中性子検出器を併用していた。この 2 種類の中性子検出器はそれぞれ条件や特徴が異なり、それをまとめたものが以下の表 3.1 である。

表 3.1　従来の検出器の条件と特徴 まとめ

	magro	hile
厚さ	20 mm	12.5 mm
距離	1500 mm	700 mm
PMT の個数	2 個	1 個
低 E_n の検出効率	×	◯
高 E_n の検出効率	◯	×
時間分解能 Δt_{TOF}	×	◯
集光率の場所依存性	×	◯

2 つの検出器の条件としては、サイズの他に、厚さ、距離、PMT の個数に違いがある。これらの条件の違いによって、それぞれの検出器の特徴に違いが見られる。2 つの検出器の大きな特徴の違いとして、得意なエネルギー領域がある。大型の magro は、高エネルギーの中性子に対して検出効率が大きく、低エネルギーに対して小さい。反対に、小型の hile は、低エネルギーの中性子に対して検出効率が大きく、高エネルギーに対して小さい。また、magro は、時間分解能 Δt_{TOF} が悪いことや、集光率に場所依存性が見られるという特徴もある。集光率の定義については、次項で言及する。

3.2　新型中性子検出器に求める性能

従来の検出器の問題点として、高エネルギー中性子と低エネルギー中性子について、別々の検出器を用いているという事が挙げられる。2 種類の検出器それぞれについて、別々の解析を行う必要がある為、時間と労力が非常に必要となる。新たに、高エネルギーと低エネルギーどちらについても検出効率が大きい検出器が実現されれば、単純計算で作業効率が倍に上がる。よって、新型中性子検出器には、数十 keV〜数 MeV という幅広い領域で検出効率が大きいことが求められる。

このとき、当然ながらエネルギーを正確に求めることが必要となる。よって、時間分解能 Δt_{TOF} がよいことも求められる。

また、中性子非束縛状態のエネルギーだけでなく、その状態への崩壊強度も求める必要がある。崩壊強度について、以下の関係が成り立つ。

$$I \propto \frac{N}{\varepsilon} \tag{3.1}$$

I：崩壊強度、N：ピークのカウント数、ε：集光率

また、集光率の定義は、以下である。

$$\text{集光率} = \frac{\text{PMT へヒットした光子数}}{\text{シンチレータで発生した光子数}} \tag{3.2}$$

式 (3.1) からわかるように、崩壊強度を求めるには、集光率を一意に決める必要がある。言い換えれば、中性子のヒットポジションによって集光率が変化する場合、崩壊強度を求めることができない。よって、新型中性子検出器には、集光率に場所依存性が見られないことが必要である。

以上より、新型中性子検出器に求められる性能は、以下である。

- 低エネルギー中性子の検出効率が大きい
- 高エネルギー中性子の検出効率が大きい
- 時間分解能 Δt_{TOF} がよい
- 集光率の場所依存性が見られない

よって、次章以降では、以上 4 点について着目して行く。

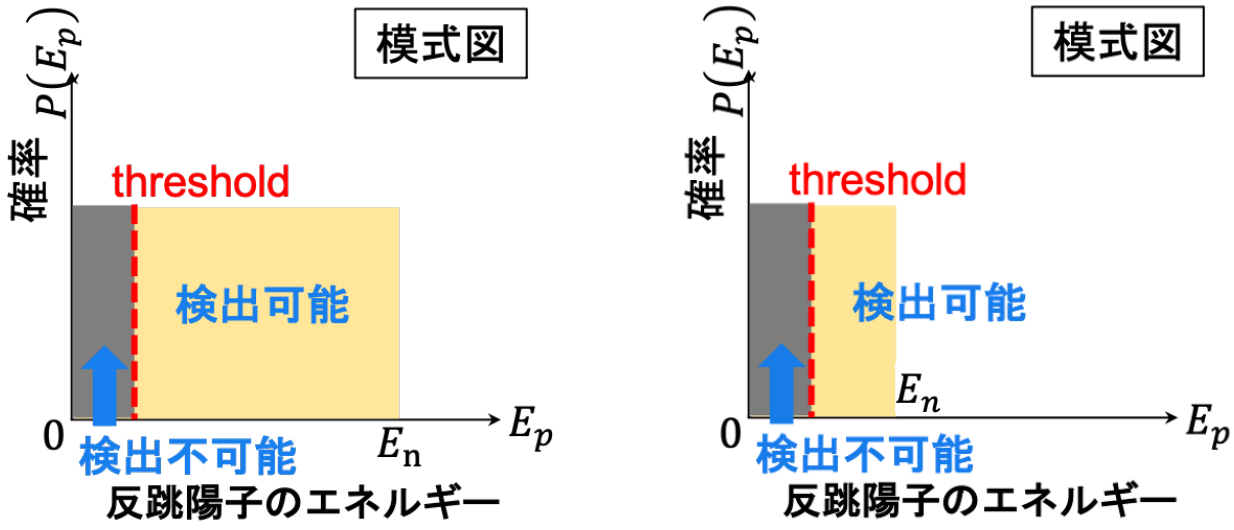
第 4 章

Geant4 を用いない検出器のデザイン

4.1　低エネルギー中性子の検出効率

2.1 項で記述したように、中性子とシンチレータ内の原子核 (主に陽子) との弾性散乱を利用して中性子を検出する。このとき、衝突の角度によって反跳陽子が得るエネルギー E_p は変化し、0〜E_n の間で連続分布する。得られる波高は、反跳陽子のエネルギー E_p の大きさに対応する為、同じ中性子エネルギー E_n の場合でも、様々な波高を取り得る。

また、ノイズなどを落とすために、threshold が必要である。そのとき、得られる波のうち、threshold を超える波高のもののみ検出される。高エネルギー E_n の場合、threshold の影響はさほど問題にならないが、図 4.1 からわかるように、低エネルギー E_n の場合は、threshold による検出効率への影響が大きくなる。



[1] 高 E_n の場合　　[2] 低 E_n の場合

図 4.1　E_p の分布と threshold の模式図

以上のことから、低エネルギー中性子の検出効率を大きくするために大事なことは、threshold に

よる影響をいかに抑えるかである。アプローチとしては、波高を大きくすることと threshold を下げることがある。前者については、集光率が大きいことが必要であり、その為にはシンチレータから PMT までの距離が短いことが重要である為、小型な検出器が求められる。threshold を下げることについては、PMT 由来のノイズを落とすために、PMT 2 個で同時計測を行うこととする。

以上をまとめると、低エネルギー中性子の検出効率が大きいために必要な条件は以下の 2 点である。

・小型の検出器

・PMT 2 個使用して同時計測

4.2 高エネルギー中性子の検出効率

$$\text{反応率} = \frac{\text{反応した中性子数}}{\text{シンチレータに入射した中性子数}} \tag{4.1}$$

式 (4.1) の反応率は、式 (4.2) で求めることができる。

$$\varepsilon = 1 - \exp[-\left((H_{CS} \times 10^{-22} \cdot \Delta L \cdot d_H) + (C_{CS} \times 10^{-22} \cdot \Delta L \cdot d_C)\right)] \tag{4.2}$$

ΔL：検出器の厚さ [mm]

H_{CS}：^{1}H の反応断面積 [barns]

d_H：シンチレータ中の ^{1}H の個数密度 [/mm^3]

C_{CS}：^{12}C の反応断面積 [barns]

d_C：シンチレータ中の ^{12}C の個数密度 [/mm^3]

今回、シンチレータには時間分解能がよい BC408 を用いるため、図 4.2 [11] にあるような BC408 についての個数密度を用いる。

Atomic Composition	BC-408
No. H Atoms per cc (x10^{22})	5.23
No. C Atoms per cc (x10^{22})	4.74
Ratio H:C Atoms	1.104
No. of Electrons per cc (x10^{23})	3.37

図 4.2　BC408 中の H 原子と C 原子の個数密度

また、^{1}H と ^{12}C の反応断面積の値には、図 4.3 と図 4.4 を用いる。[2]

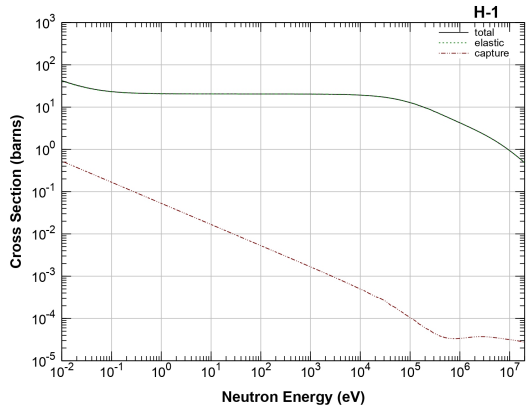


図 4.3　^{1}H の反応断面積

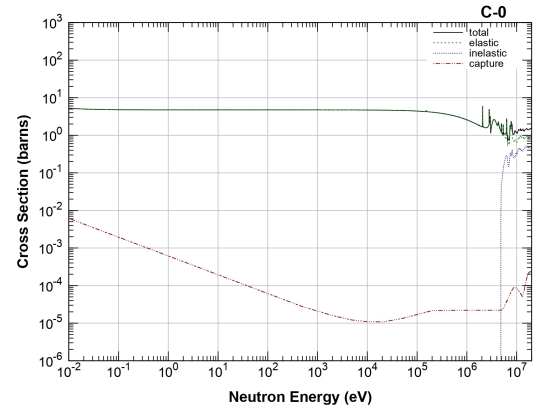


図 4.4　^{12}C の反応断面積

これらを用いて、$E_n = 10$ keV、100 keV、1 MeV、2 MeV、3 MeV それぞれの場合での、厚さと反応率の関係を求めたものが図 4.5 である。

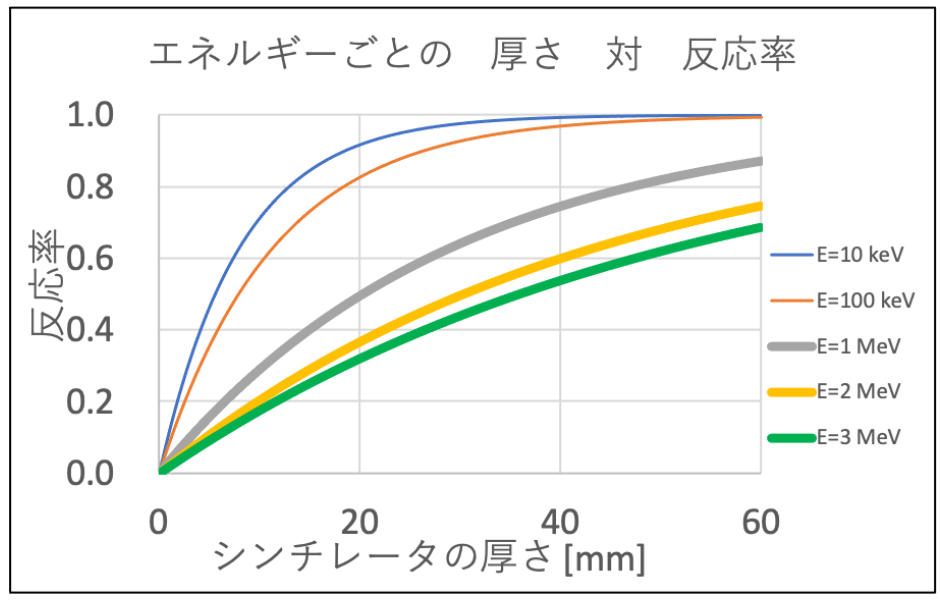


図 4.5　E_n ごとの厚さと反応率の関係

図 4.5 からわかるように、シンチレータの厚さが 20 mm 以上において、低エネルギー中性子は、厚くしてもさほど反応率が上がらないが、高エネルギー中性子の場合は、厚くすると反応率が急激に上がる。よって、高エネルギー中性子の検出効率が大きいために必要な条件は以下である。

・シンチレータの厚さが大きい。

4.3　時間分解能 Δt_{TOF}

式 (4.3) から、時間分解能 Δt_{TOF} がよくなる条件は、ΔL が小さい、L が大きい、Δt が小さい、である。

$$\Delta t_{TOF} \propto \Delta E = 2E_n \sqrt{\left(\frac{\Delta L}{L}\right)^2 + \left(\frac{\Delta t}{t_{TOF}}\right)^2} \tag{4.3}$$

L：中性子の飛行距離、t_{TOF}：飛行に要した時間、E_n：中性子エネルギー

前項で、シンチレータの厚さが大きいことが必要となったが、シンチレータの厚さが厚いほど、中性子の飛行距離の誤差 ΔL が大きくなるため、時間分解能 Δt_{TOF} が悪くなってしまう。この影響を抑えるには、中性子の飛行距離 L を大きくする必要がある。実際、$L = 1500$ mm という大きな値にした場合、式 (4.3) を用いて計算すると、図 4.6 のように、厚さを大きくしても、さほど時間分解能 Δt_{TOF} が悪くならないことがわかる。

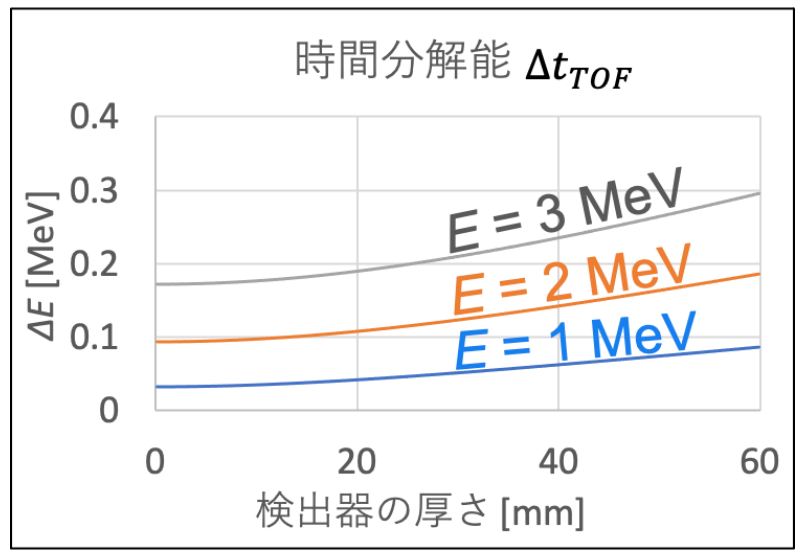


図 4.6　厚さと時間分解能 Δt_{TOF} の関係

よって、中性子の飛行距離 L を magro と同じ大きな値である $L = 1500$ mm とすることにした。この値は、実験施設で設置可能な最大の距離である。また、シンチレータの厚さを 20 mm よりさらに大きくすることとする。今回は、材料の面での限度である 29 mm とする。

・中性子の飛行距離 L=1500 mm

・シンチレータの厚さ：29 mm

また、Δt が小さくすることでも時間分解能 Δt_{TOF} はよくなる。そして、この Δt は検出器のサイズが影響しており、サイズが大きい検出器ほど、Δt は大きくなる。よって、サイズが小さい検出器にすることとした。

・検出器のサイズ：小型

4.4　検出効率（立体角の確保）

前項で、中性子の飛行距離 L を大きくし、シンチレータのサイズを小型にすることとしたが、この場合、シンチレータが覆う立体角が非常に小さい為、検出効率が小さくなってしまう。よって、図 4.7 のように複数個を組み合わせることで、大きな立体角を確保することとする。

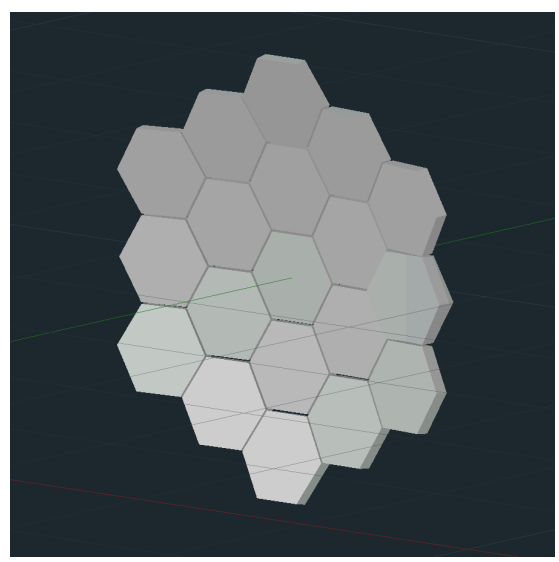


図 4.7　シンチレータの敷き詰めについての図

敷き詰めやすさ面から、シンチレータの形を六角形にすること、ライトガイドを後方に接続することとする。

・検出器を複数個組み合わせる
・シンチレータの形：六角形
・ライトガイドを後方に接続する

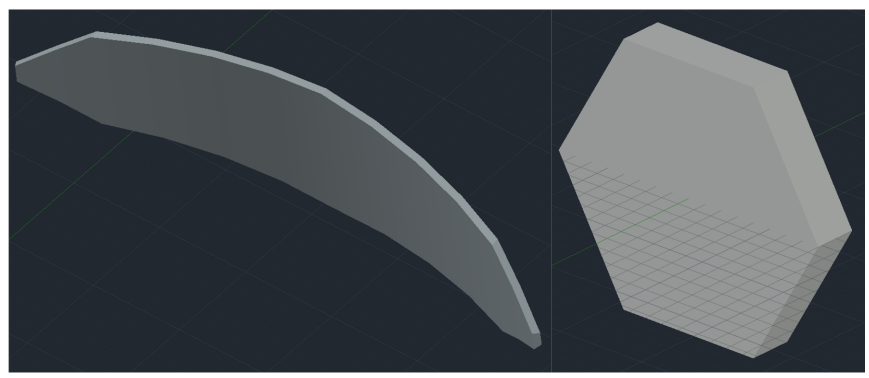


図 4.8　シンチレータの形状の違い

また、図 4.8 のように、従来の検出器はシンチレータを湾曲させていたが、今回は飛行距離が長く、且つシンチレータのサイズが小さいことから、平面板にすることとした。

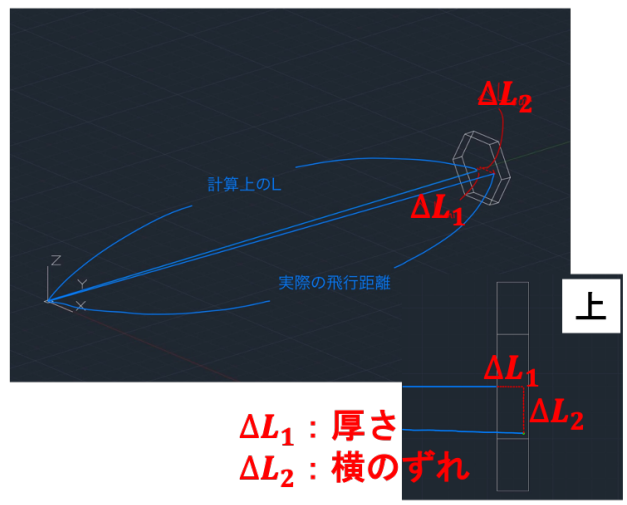


図 4.9　飛行距離の誤差 ΔL についての図

よって、図 4.9 のように、飛行距離の誤差 ΔL には、厚さによる誤差 ΔL_1 と、横のずれによる誤差 ΔL_2 の 2 種類の誤差がある。これらの関係は式 (4.4) のようになる。

$$\Delta L = \sqrt{(L + \Delta L_1)^2 + {\Delta L_2}^2} - L \tag{4.4}$$

湾曲板の場合は $\Delta L_2 = 0$ であるので、平面板にすることによる時間分解能 Δt_{TOF} への影響を考える。

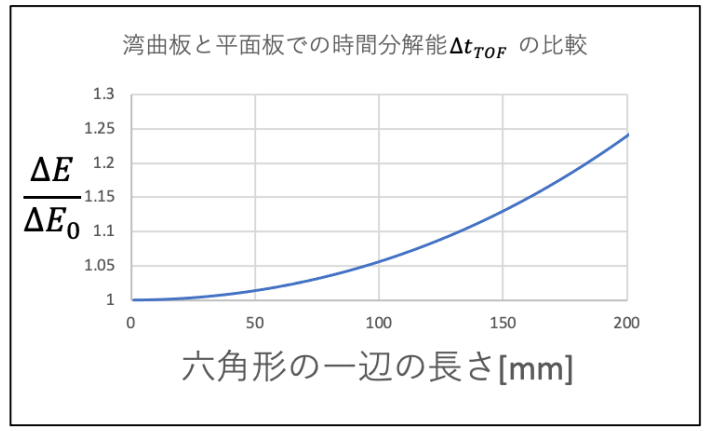


図 4.10　湾曲板と平面板での時間分解能 Δt_{TOF} の比較

図 4.10 からわかるように、六角形の一辺の長さが 130 mm を超えたあたりで、湾曲の場合との比率が 1.1 倍を超える。よって、130 mm 以下であれば平面板にすることによる影響は小さく、平面板を用いても問題ないと考えた。

4.5　新型中性子検出器の決定条件

今章での決定事項をまとめると以下のようになる。

・検出器のサイズ：小型

・PMT 2 個使用して同時計測

・中性子の飛行距離 L=1500 mm

・シンチレータの厚さが大きい：29 mm

・検出器を複数個組み合わせる

・シンチレータの形：六角形

・ライトガイドを後方に接続する

表としてまとめると以下のようになる。

表 4.1　検出器の条件と特徴 まとめ

	magro	hile	新型検出器
シンチレータのサイズ	大型	小型	小型
シンチレータの形	湾曲板	湾曲正方形板	平面六角形板
シンチレータの厚さ	20 mm	12.5 mm	29 mm
距離	1500 mm	700 mm	1500 mm
ライトガイド、PMT の接続部分	横方向	横方向	後方
PMT の個数	2 個	1 個	2 個
低 E_n の検出効率	×	◯	◎?
高 E_n の検出効率	◯	×	◎?
時間分解能 Δt_{TOF}	×	◯	◯?
集光率	×	◯	◯?
集光率の場所依存性	×	◯	◯(未検討)

図 4.11　新型中性子検出器と従来の検出器の比較

但し、集光率の場所依存性については、次章で検討する。

第 5 章

Geant4 を用いた検出器のデザイン

5.1　Geant4 の条件

Geant4 とは、物質中を通過する粒子の相互作用をシミュレーションするための計算コードである。[8]Geant4 の条件を以下に示す。1 MeV の中性子を 2×10^7 個を全方向へランダムに飛ばし、検出器を 1500 mm 離した位置に設置した。空間には真空、シンチレータには BC408、ライトガイドにはアクリルを定義した。図で示したものが、図 5.1 である。

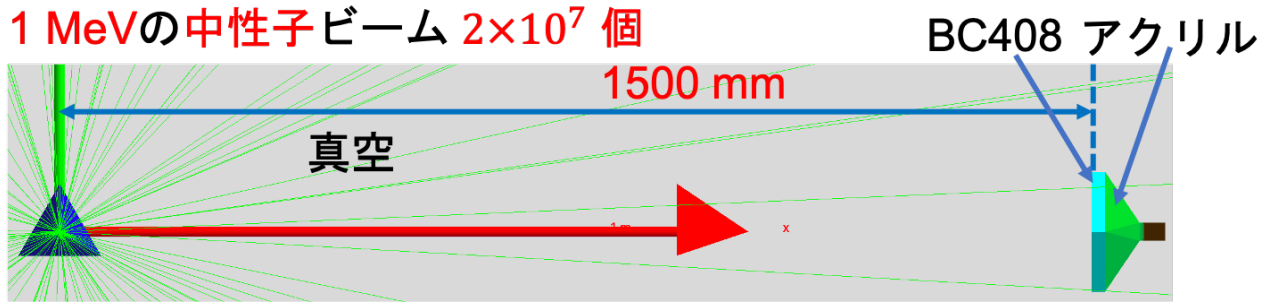
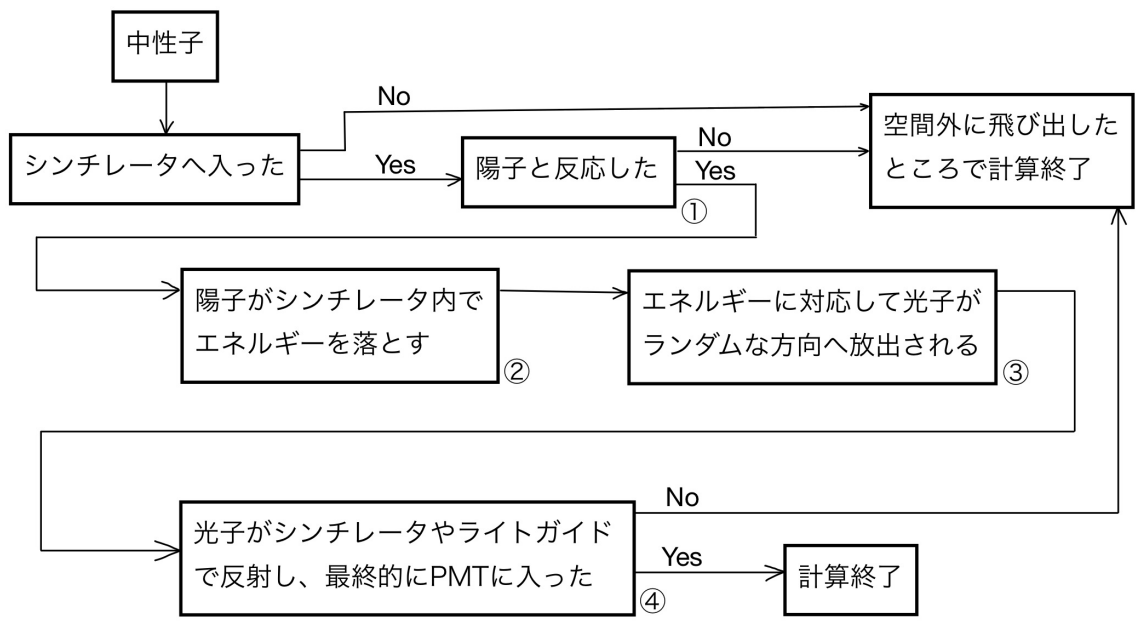


図 5.1　Geant4 の条件

また、フローチャートで表したものが図 5.2 である。今回、中性子が反応して出てくる光が、PMT の入口に到達するまでをシミュレーションした。検討することとしては、六角形の一辺の長さ、ライトガイドの長さ、反射材、の 3 つについてである。



①：中性子が陽子と反応した時間とポジションと反応の種類を記録
②：陽子がシンチレータ内で落としたエネルギーを記録
③：放出された光子数を記録
④：2つのPMTそれぞれについて、入った光子数と最初に到達した時間を記録

図 5.2　Geant4 のフローチャート

5.2　評価ポイント

Geant4 の結果によって評価するポイントは、前章で検討できていない集光率の場所依存性と、今回の検出器開発で最重要視する集光率の 2 点である。それぞれの評価方法について以下に記述する。

5.2.1　集光率の評価方法

集光率の評価には、横軸に集光率、縦軸に PMT ヒット数をとった 2 次元ヒストグラムを用いる。このヒストグラムについて、「最も密度が高い点が集光率 0.1 を超えている」という条件を用いる。

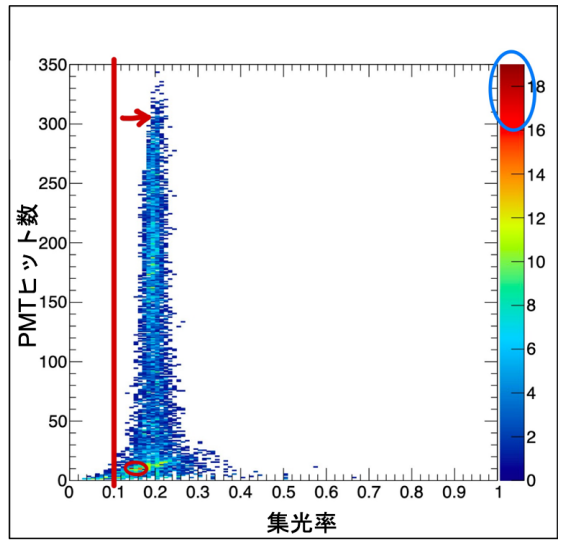


図 5.3　集光率と PMT ヒット数の 2 次元ヒストグラム

5.2.2　集光率の場所依存性の評価方法

集光率の場所依存性の評価には、横軸に六角形の中心からの距離（図 5.4 を参照）、縦軸に集光率をとった 2 次元ヒストグラムを用いる。

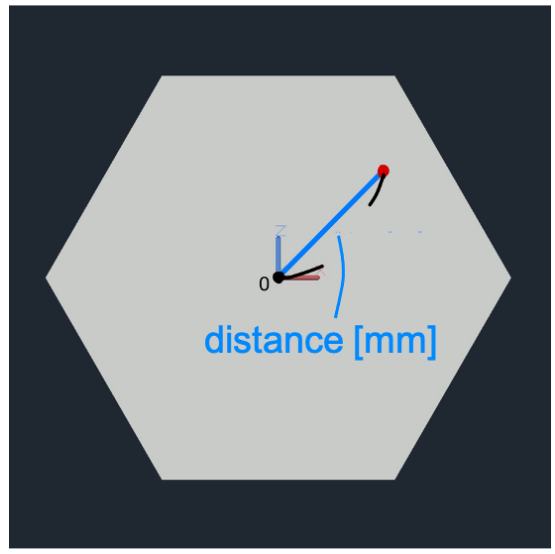


図 5.4　六角形の中心からの距離

このヒストグラムについて、図 5.5 のように折れ曲がっていないという条件を用いる。

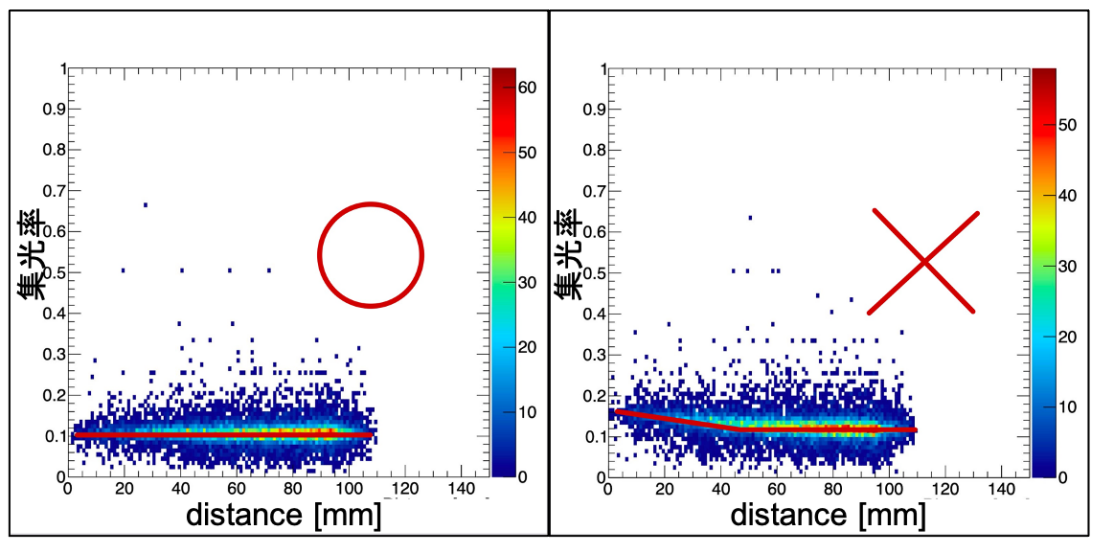


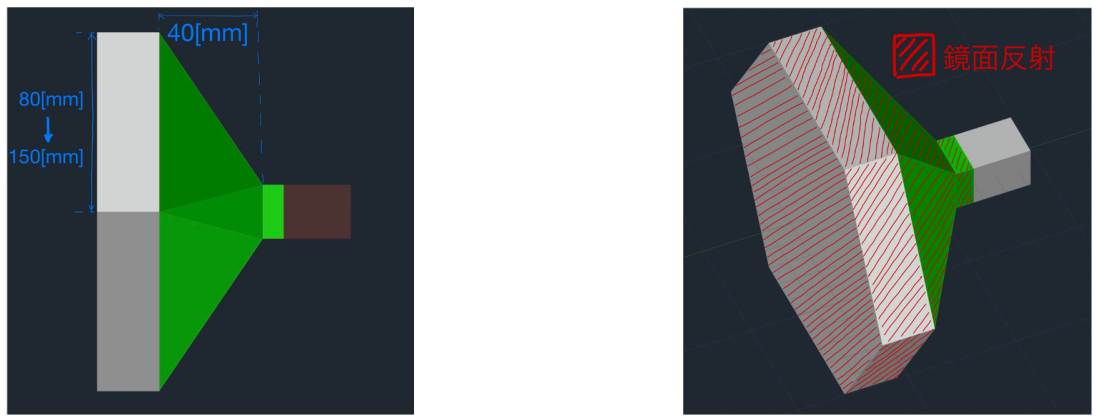
図 5.5　六角形の中心からの距離と集光率の 2 次元ヒストグラム

5.3　PMT １個の検出器について

PMT が 2 個の検出器で考える前に、初めに PMT が 1 個の検出器で考える。

5.3.1　六角形の一辺の長さとライトガイドの長さ

初めに、ライトガイドの長さを 40 mm に固定して、シンチレータ表面の六角形の一辺の長さを 80 mm〜150 mm まで 10 mm 刻みで変更して、各々の Geant4 の結果を比較した。なお、反射材の条件は全て鏡面反射（反射率：98%）に設定した。

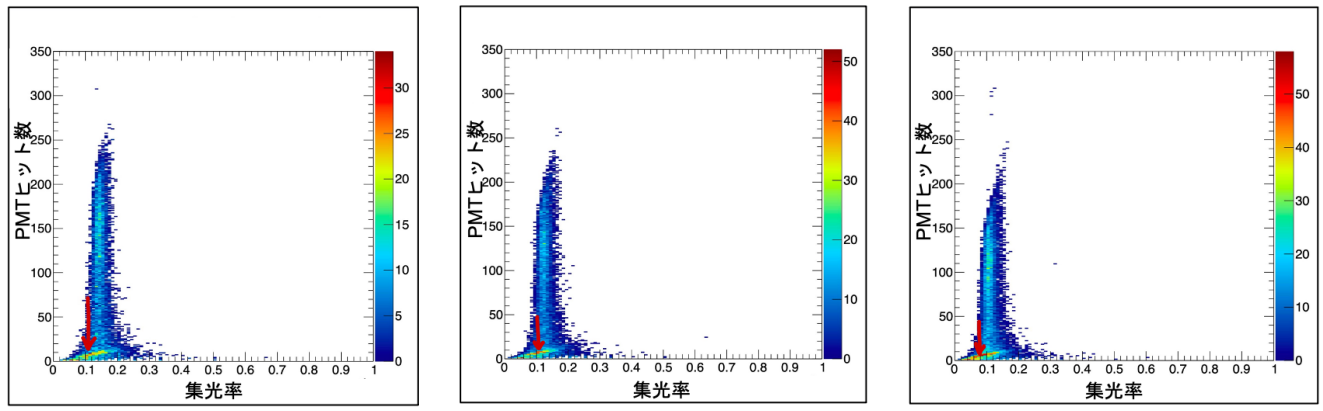


[1] 形状についての条件　　[2] 反射条件

図 5.6　シミュレーション条件

まず、集光率についてだが、図 5.7 のように、シンチレータ表面の六角形の一辺の長さを 120 mm にしたとき、密度の 1 番高い部分の集光度は 0.1 を下回ってしまう。よって、ライトガイドの長さが

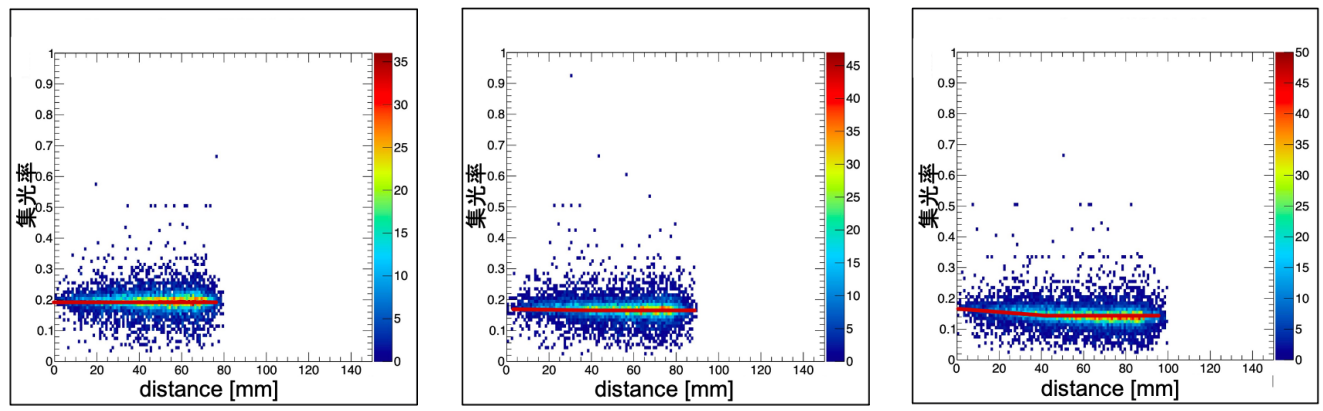
40 mm のとき、六角形の一辺の長さは 110 mm 以下にする必要がある。



[1]100 mm　　[2]110 mm　　[3]120 mm

図 5.7　六角形の一辺の長さと集光率

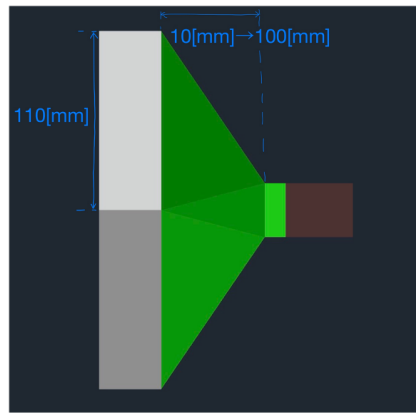
続いて、集光率の場所依存性についてだが、図 5.8 のように、シンチレータ表面の六角形の一辺の長さを 100 mm にしたとき、場所による集光率の違いが顕著になってくる。よって、ライトガイドの長さが 40 mm のとき、六角形の一辺の長さは 90 mm 以下にする必要がある。



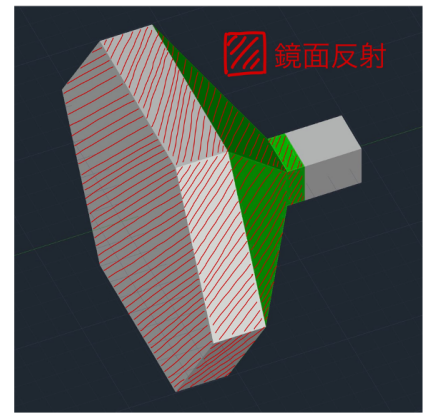
[1]80 mm　　[2]90 mm　　[3]100 mm

図 5.8　六角形の一辺の長さと集光率の場所依存性

次に、シンチレータ表面の六角形の一辺の長さを 110 mm に固定して、ライトガイドの長さを 10 mm ～100 mm まで 10 mm 刻みで変更して、各々の Geant4 の結果を比較した。なお、反射材の条件は全て鏡面反射（反射率：98%）に設定した。



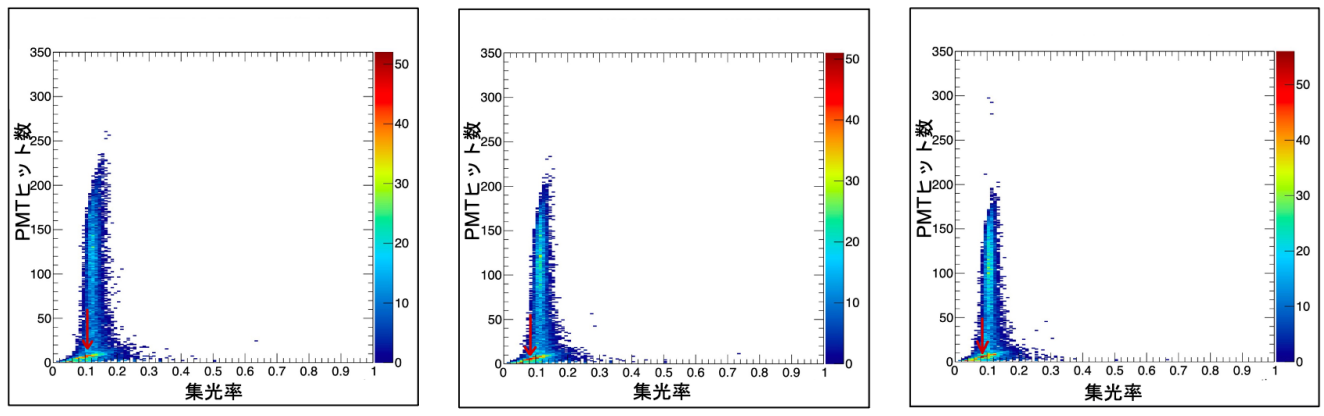
[1] 形状についての条件



[2] 反射条件

図 5.9　シミュレーション条件

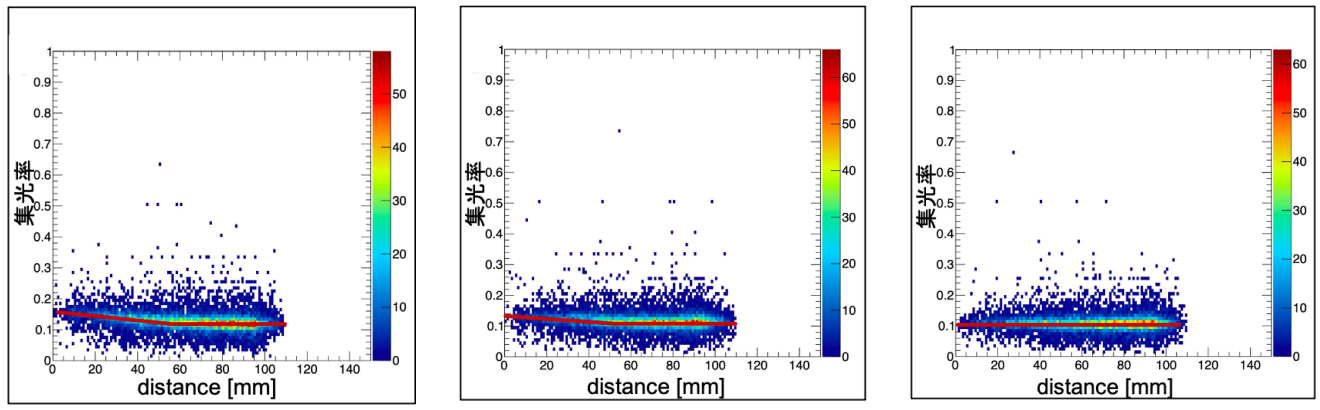
まず、集光率についてだが、図 5.10 のように、ライトガイドの長さを 50 mm にしたとき、密度の 1 番高い部分の集光度は 0.1 を下回ってしまう。よって、ライトガイドの長さが 110 mm のとき、六角形の一辺の長さは 40 mm 以下にする必要がある。



[1]40 mm　　[2]50 mm　　[3]60 mm

図 5.10　ライトガイドの長さと集光率

続いて、集光率の場所依存性についてだが、図 5.11 のように、ライトガイドの長さを 60 mm にしたとき、場所による集光率の違いが見られなくなる。よって、ライトガイドの長さが 110 mm のとき、六角形の一辺の長さは 60 mm 以上にする必要がある。



[1]80 mm　　[2]90 mm　　[3]100 mm

図 5.11　ライトガイドの長さと集光率の場所依存性

以上の結果をまとめたものが図 5.12 である。

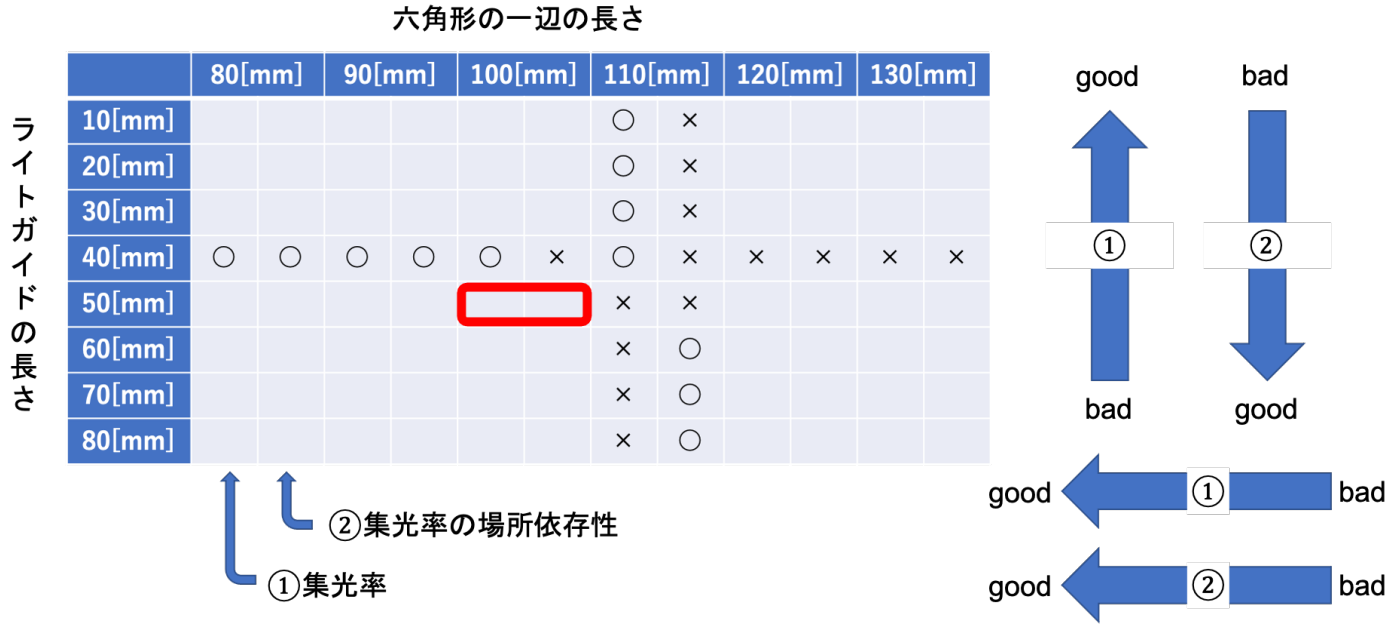
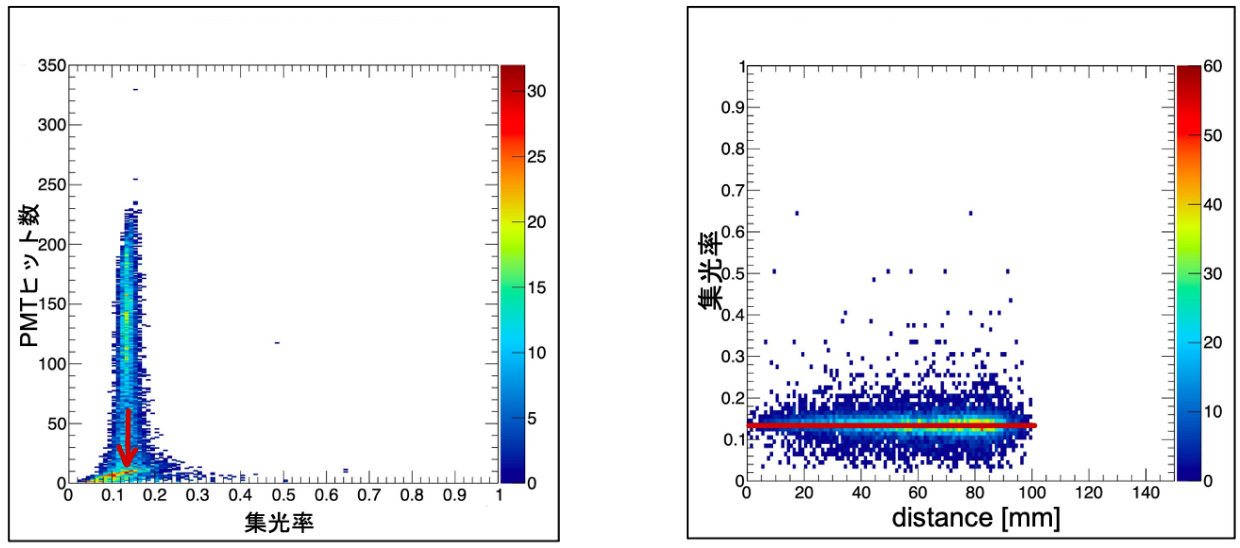


図 5.12　六角形の一辺の長さとライトガイドの長さ まとめ

今回目指すものは、図 5.12 においてともに ◯ となるようなものの中で、最も六角形の一辺の長さが長くなるようなものである。その理由としては、検出器を複数個組み合わせて利用するため、可能な限り必要な個数を少なくして、コストを抑えたいからである。これまでの Geant4 のシミュレーション結果から、図 5.12 で赤枠で囲った場合（六角形の一辺の長さ：100 mm、ライトガイドの長さ：50 mm）が最適となる可能性がある。よって、この場合についてシミュレーションを行なった。



[1] 集光率　　[2] 集光率の場所依存性

図 5.13　「六角形の一辺の長さ：100 mm、ライトガイドの長さ：50 mm 」のシミュレーション結果

図 5.13 からわかるように、集光率についても集光率の場所依存性についても条件を満たしている。よって、これが最適である。

・六角形の一辺の長さ：100 mm

・ライトガイドの長さ：50 mm

5.3.2　反射材

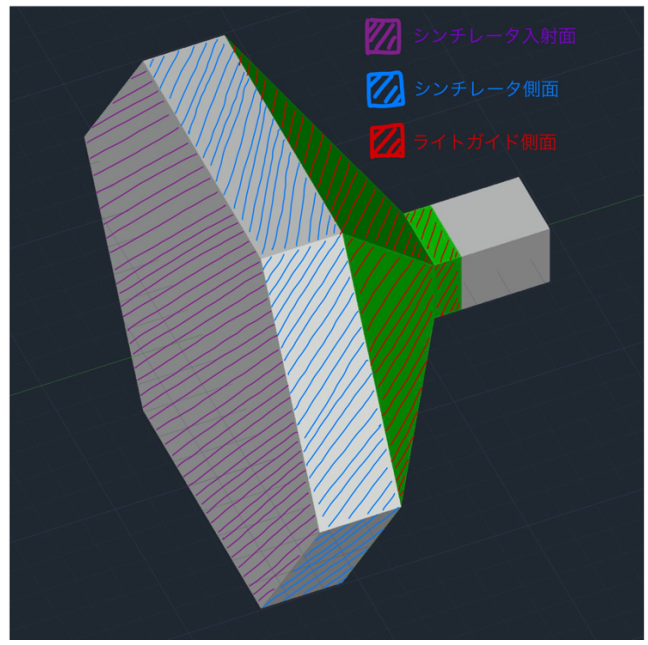


図 5.14　反射材の条件

図 5.14 のように、3 つの部分に分けて、それぞれについて反射材を別々に鏡面反射あるいは乱反射を定義していく。全 8 パターンをシミュレーションし、集光率の平均値で比較する。

表 5.1　反射材の条件番号

反射材の条件番号	シンチレータ入射面	シンチレータ側面	ライトガイド側面
1	鏡面反射	鏡面反射	鏡面反射
2	鏡面反射	鏡面反射	乱反射
3	鏡面反射	乱反射	鏡面反射
4	鏡面反射	乱反射	乱反射
5	乱反射	鏡面反射	鏡面反射
6	乱反射	鏡面反射	乱反射
7	乱反射	乱反射	鏡面反射
8	乱反射	乱反射	乱反射

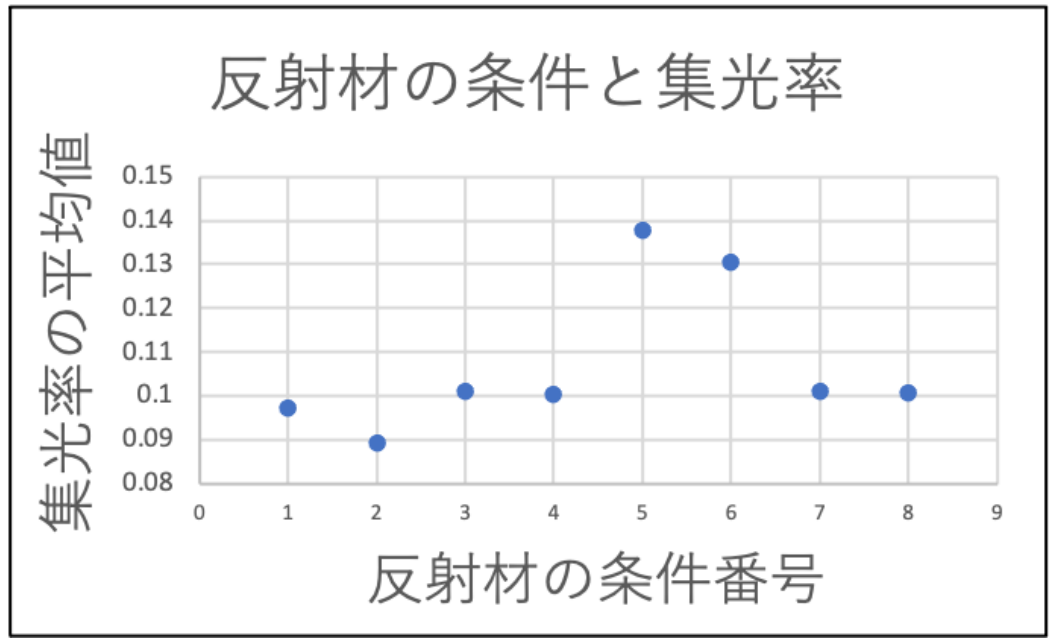


図 5.15　反射材の条件と集光率

図 5.15 からわかるように、反射材の条件番号：5 の場合（シンチレータ入射面：乱反射、シンチレータ側面：鏡面反射、ライトガイド側面：鏡面反射）が集光率の平均値が最も大きい。このとき、図 5.16 のように集光率の場所依存性も見られない。

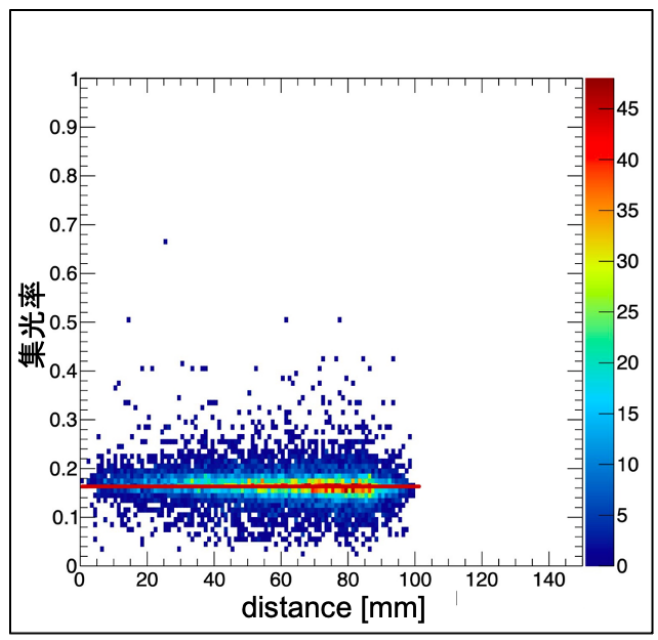


図 5.16　反射材の条件と集光率の場所依存性

よって、最適な反射材の条件は以下である。

・シンチレータ入射面：乱反射

・シンチレータ側面：鏡面反射

・ライトガイド側面：鏡面反射

5.4　PMT ２個の検出器について

5.4.1　PMT １個の場合との比較

第 4 章で述べたように、低エネルギー中性子の検出効率を大きくするために、PMT を 2 個用いるのだが、PMT を 2 個用いる場合、片方の PMT のみに光子が到達するようなイベントをカウントしないことになる為、むしろ検出効率が下がる可能性がある。また、PMT が 1 個の場合とは違った集光率の場所依存性が見られるようになる可能性もある。よって、初めにこの 2 点について、PMT が１個の場合での最適な場合のものを用いて確認する。

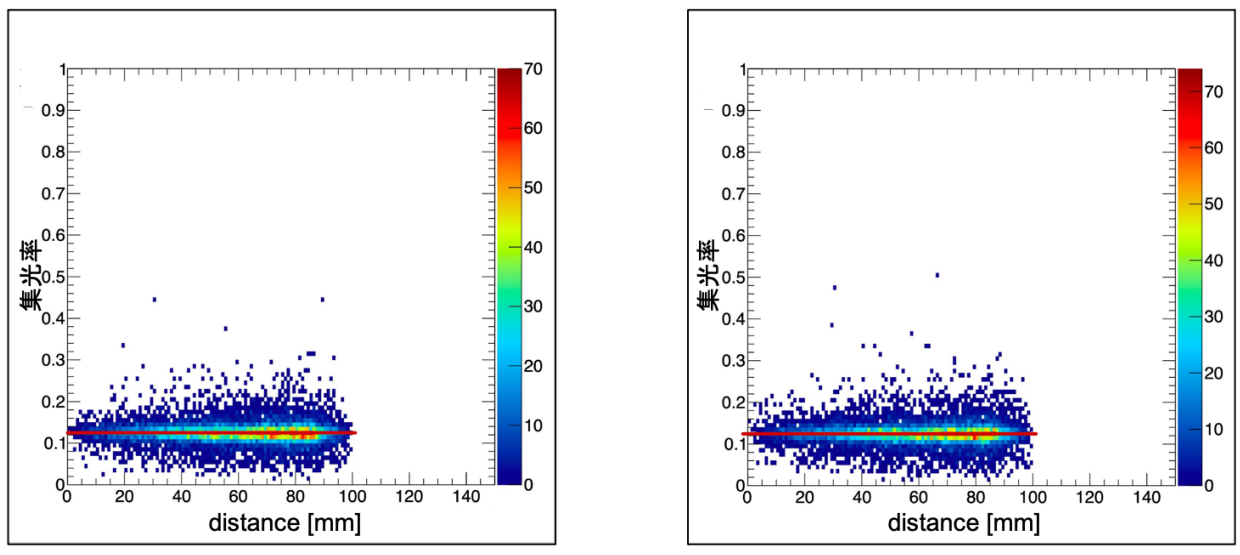
PMT １個の場合と PMT ２個の場合での検出効率についてまとめたものが表 5.2 である。

表 5.2　PMT 1 個の場合と PMT 2 個の場合での検出効率

	シンチレータに入射したイベント数	シンチレータで光子が発生したイベント数		PMT に到達したイベント数	total efficiency[%]
PMT 1 個	18264	12608		12325	67.48
PMT 2 個	18371	12796	coin	12328	67.11
			R only	124	0.67
			L only	118	0.64

$$\text{total efficiency} = \frac{\text{PMT に到達したイベント数}}{\text{シンチレータに入射したイベント数}} \tag{5.1}$$

表 5.2 からわかるように、PMT 2 個の同時計測を行なっても、total efficiency はほとんど変化しない。また、片方の PMT のみに光子が到達するようなイベントについての total efficiency は、合わせても僅か 1% 程度である。



[1]R についてのヒストグラム　　[2]L についてのヒストグラム

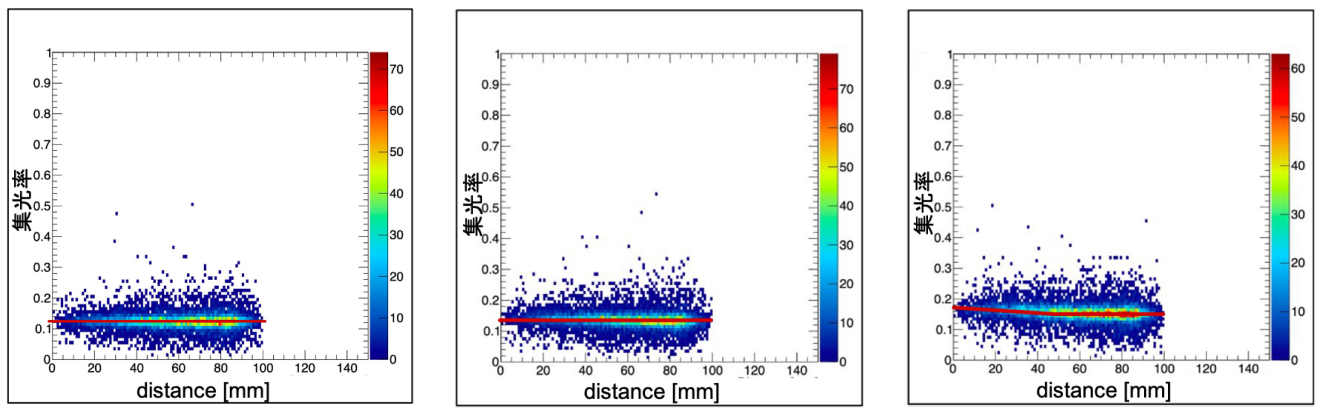
図 5.17　集光率の場所依存性

また、図 5.17 からわかるように、集光率の場所依存性も見られない。よって、PMT 2 個の同時計測を行なっても問題ない。

5.4.2　六角形の一辺の長さとライトガイドの長さ

PMT 1 個の場合とそこまで大きく形状は変わらないので、結果は大きく変わらないと考えられる。そこで、PMT 1 個の場合の結果を参考にして、六角形の一辺の長さを 100 mm とし、集光率を少し

でも大きくする為に、ライトガイドの長さを 50 mm より更に短くできるかをシミュレーションして確かめる。



[1]50 mm　　[2]40 mm　　[3]30 mm

図 5.18　ライトガイドの長さと集光率の場所依存性

集光率の場所依存性について、図 5.18 のように、ライトガイドの長さを 30 mm にしたとき、場所による集光率の違いが顕著になる。よって、ライトガイドの長さが 40 mm が適切である。以上より、最終的な検出器の形状は以下である。

・六角形の一辺の長さ：100 mm

・ライトガイドの長さ：40 mm

最後に、時間分解能 Δt_{TOF} を評価する。中性子非束縛状態は Γ 幅を持ち、その値は崩壊のしやすさに対応する。本グループが着目しているものについては、数 keV〜数百 keV である。これを時間空間で考える。但し、これはエネルギーに依存するので、今回はエネルギー 1 MeV の場合で考える。すると、約 0.05 nsec〜約 5 nsec となる。よって、時間分解能 Δt_{TOF} が、5 nsec 以上であると測定することができないため、これより小さいことが求められる。

最終的な検出器の形状の時の時間分解能について調べたところ、0.85 nsec であった。また、中性子エネルギーごとの Γ 幅の最大値と検出器の時間分解能を比較したものが図 5.19 である。ここから、約 6 MeV の中性子まで検出できることが予想される。

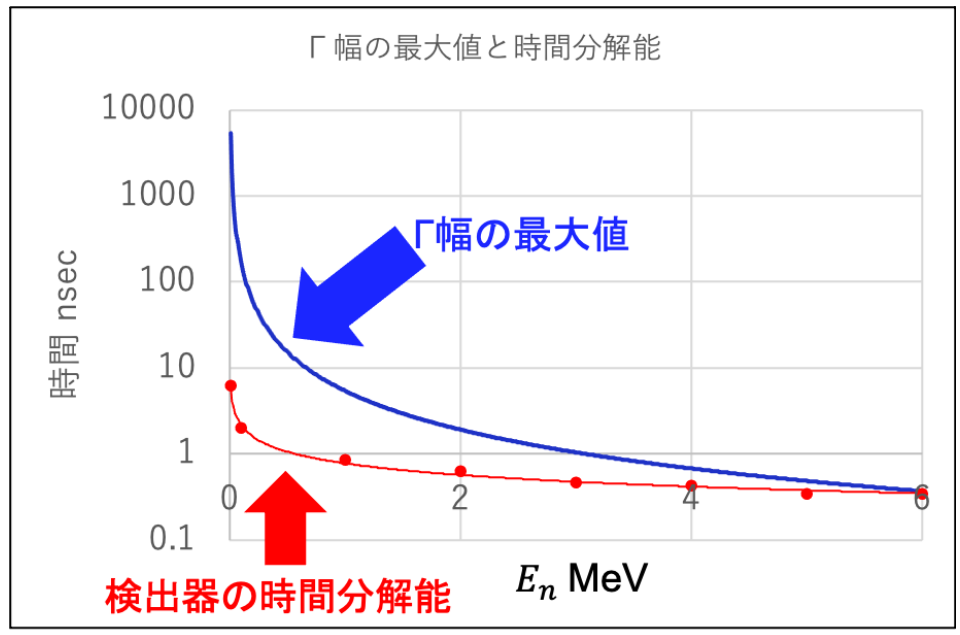
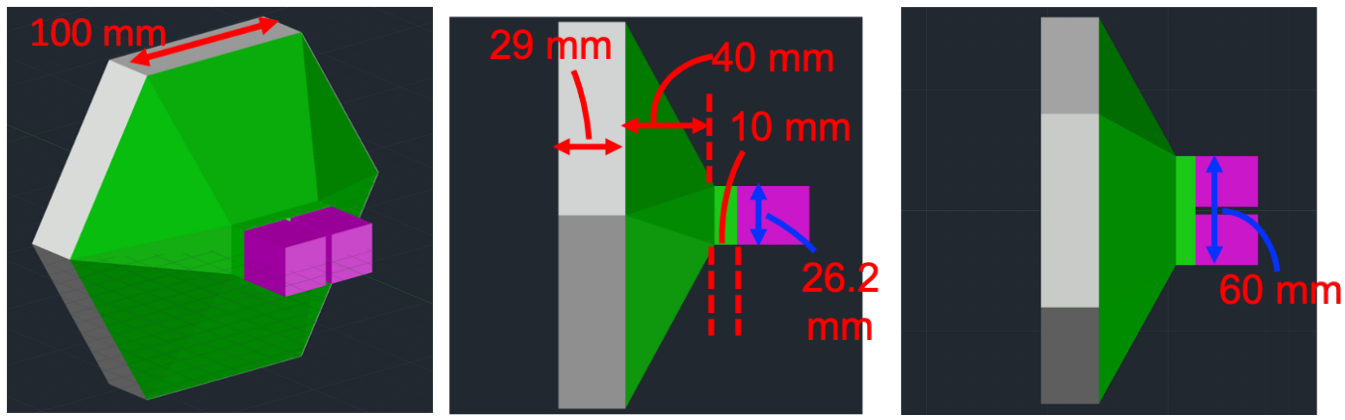


図 5.19　Γ 幅の最大値と時間分解能

第 6 章

最終的な検出器のデザイン

最終的な検出器のデザインは図 6.1、図 6.2 のようになる。白色部分がシンチレータ、緑色部分がライトガイド、紫色部分が PMT である。



[1] 後方斜めから見た図　　[2] 横から見た図　　[3] 上から見た図

図 6.1　新型中性子検出器のデザイン

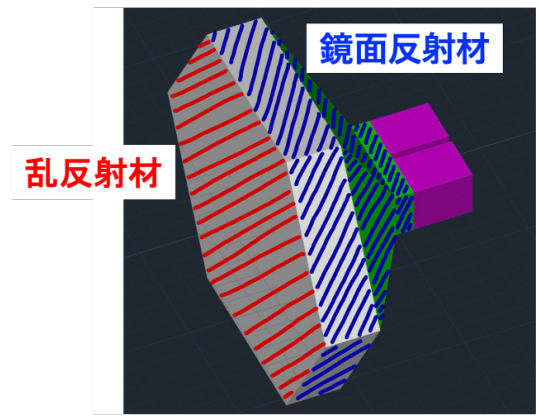
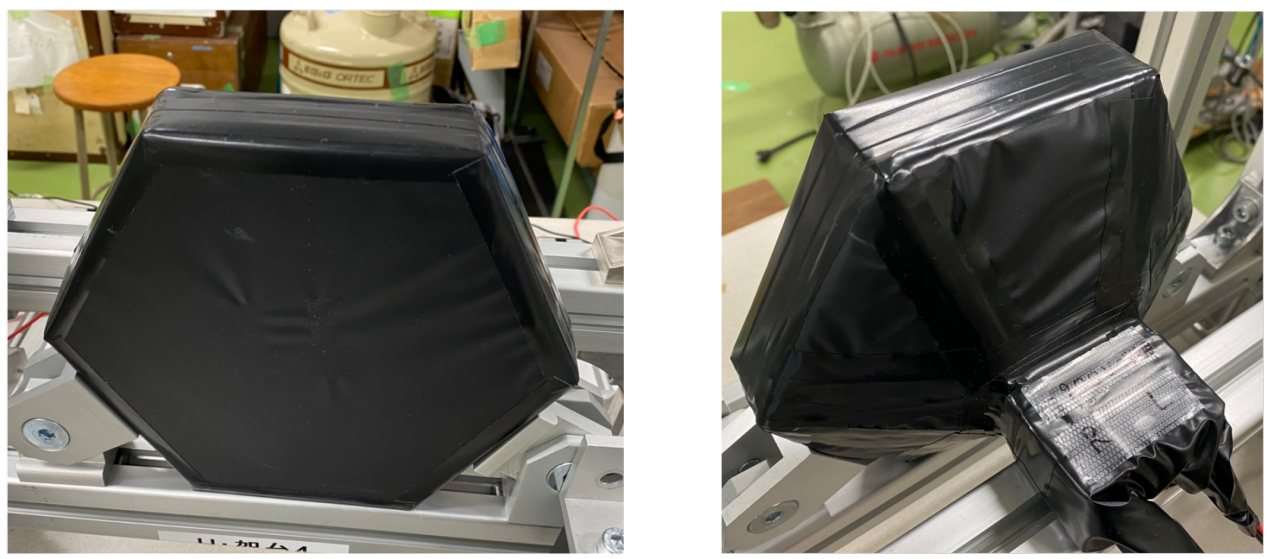


図 6.2　新型中性子検出器の反射材の条件

実物の写真が図 6.3 である。



[1] 前方　　[2] 後方

図 6.3　新型中性子検出器の実物の写真

第 7 章

性能評価実験

今章では、新型中性子検出器についての性能評価実験について記述する。今回は、非常に初期段階の実験として、主に 2 つのことについての実験を行った。それは、時間分解能についてとエネルギー閾値についてである。

7.1　時間分解能の測定

新型中性子検出器の時間分解能を測定する為に、β 線源である ^{90}Sr 線源を用いて、β 線検出器との同時計測実験を行った。具体的な回路の概略図が図 7.1 である。

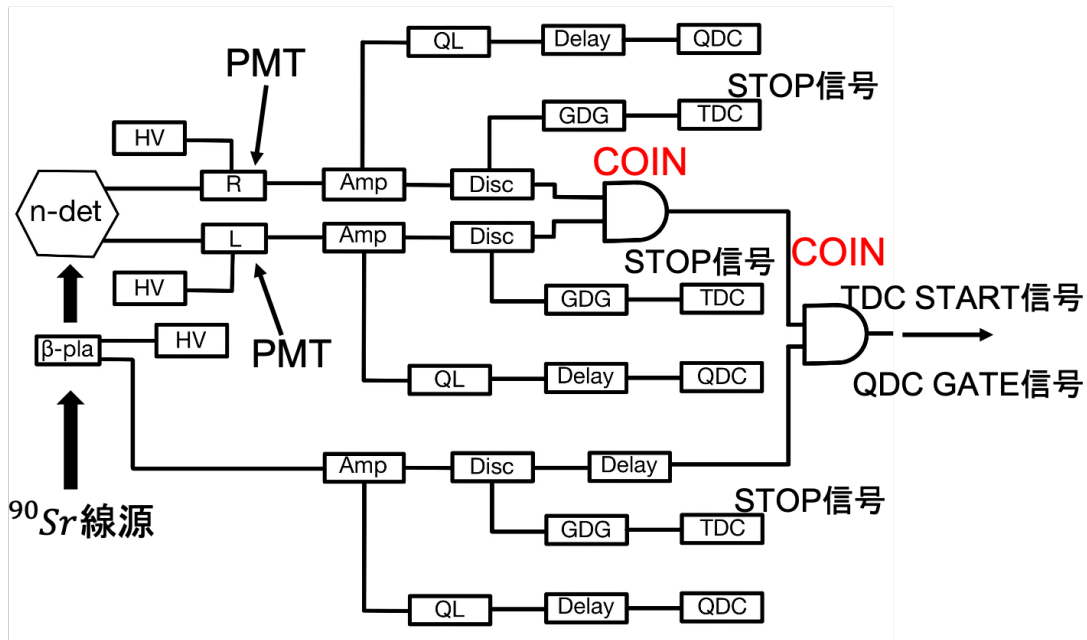
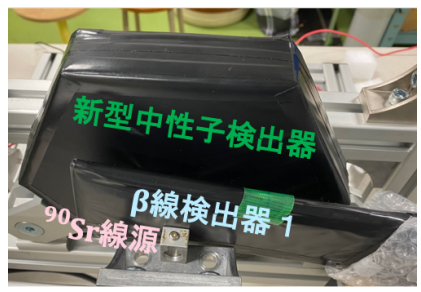


図 7.1　時間分解能の測定実験の回路

QL・・・Quad Linear Fan-in Fan-out

GDG・・・GATE&DELAY GENERATOR

今回、新型中性子検出器 (n-det) と 2 種類の β 線検出器 (β-1、β-2) の計 3 つの検出器について、全 3 パターンのペアでの同時計測を行った。トリガー条件は、中性子検出器の 2 つの PMT と β 線検出器の PMT の全てで同時に検出できた時である。(2 枚の β 線検出器の場合については、それぞれの PMT 両方で同時に検出できた時である。)



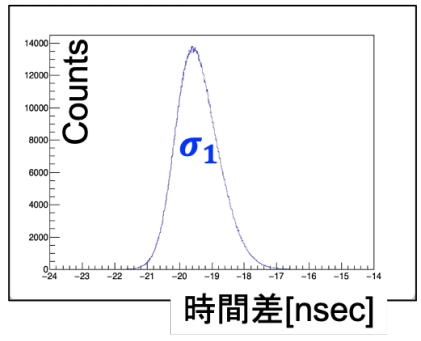
[1]n-det & β-1

[2]n-det & β-2

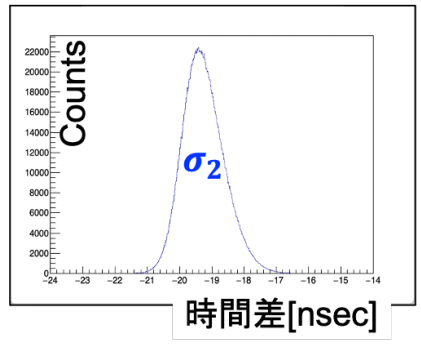
[3]β-1 & β-2

図 7.2　実験のセットアップ

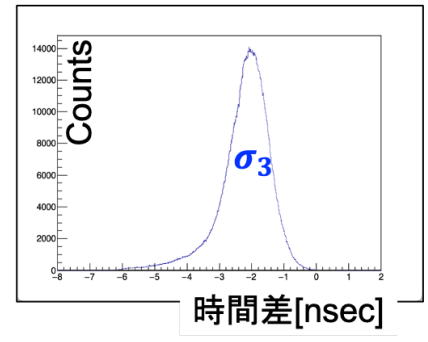
各計測ごとに 2 種類の検出器の TDC データについて引き算を行い、そのヒストグラムについての $\sigma(\sigma_1、\sigma_2、\sigma_3)$ を求める。但し、新型中性子検出器には 2 つの PMT（R と L）が接続されているため、TDC のデータも QDC のデータも 2 つずつ得られる。今回、新型中性子検出器の TDC データとしては、2 つの PMT の TDC データの平均値を使用した。



[1]n-det & β-1



[2]n-det & β-2



[3]β-1 & β-2

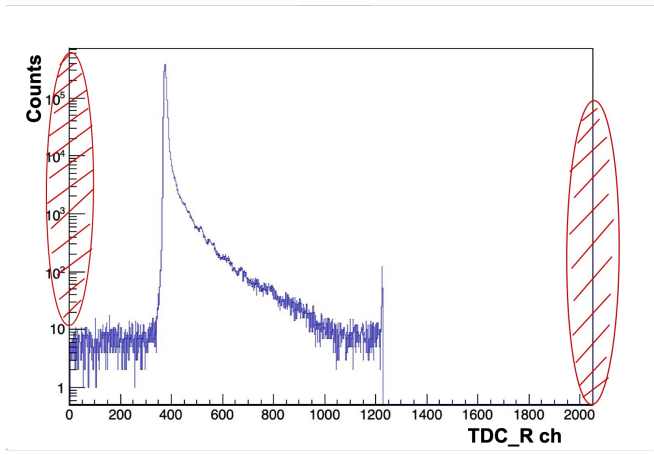
図 7.3　TDC データの引き算ヒストグラム

ここで、中性子検出器と β 線検出器の場合のヒストグラムの条件を以下に示す。 但し、_R、_L は新型中性子検出器に接続されている 2 つの PMT について、_b は β 線検出器に接続されている PMT についてのデータを意味する。

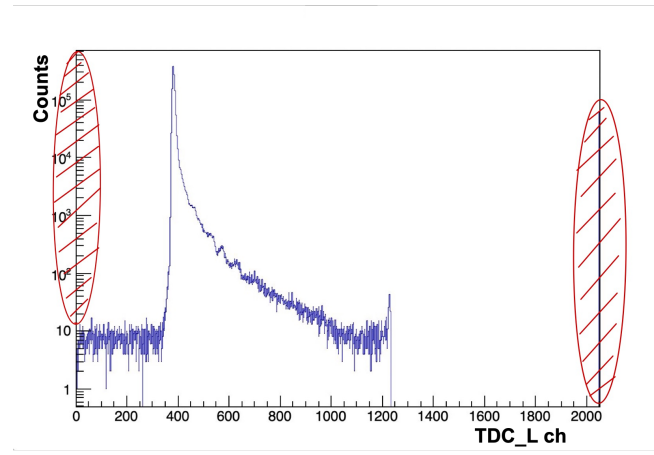
$$1 \leq \text{TDC\_R} < 2048,\ 1 \leq \text{TDC\_L} < 2048,\ \text{peak\_min} \leq \text{TDC\_b} < \text{peak\_max},$$
$$\text{QDC\_R} \geq 300,\ \text{QDC\_L} \geq 300,\ \text{QDC\_b} < 1300$$

peak_min：ピークの最小チャンネル、peak_max：ピークの最大チャンネル

条件のチャンネルの決め方について、新型中性子検出器の TDC データについては、両端を外す。



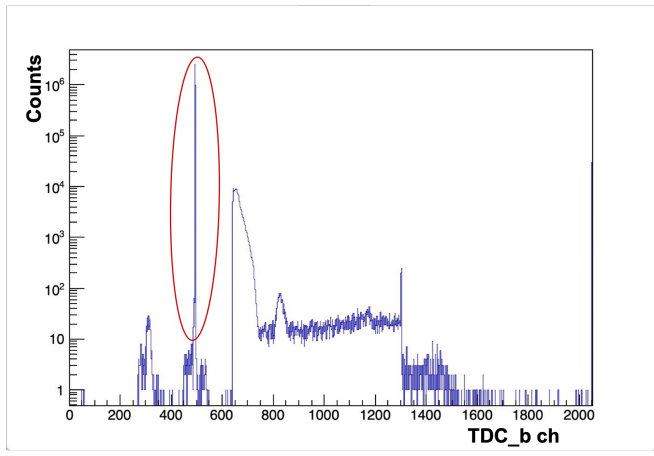
[1]R の TDC データ



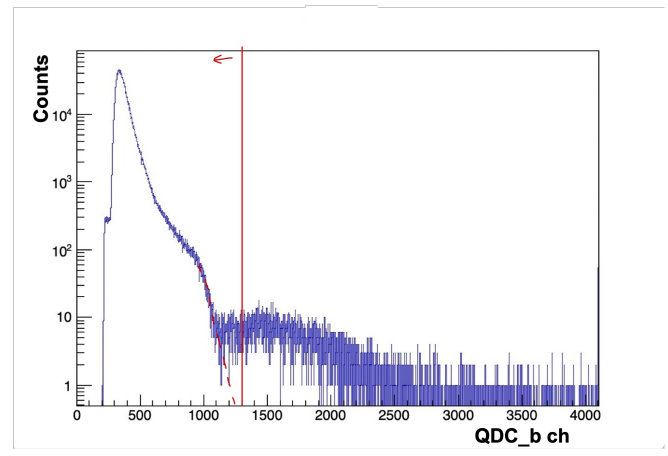
[2]L の TDC データ

図 7.4　新型中性子検出器の TDC ヒストグラム

β 線検出器の TDC データについては、ピークのみを取り出す。QDC データについては、ピーク右端の延長が 1200 ch〜1300 ch であるため、1300 ch 以下とする。



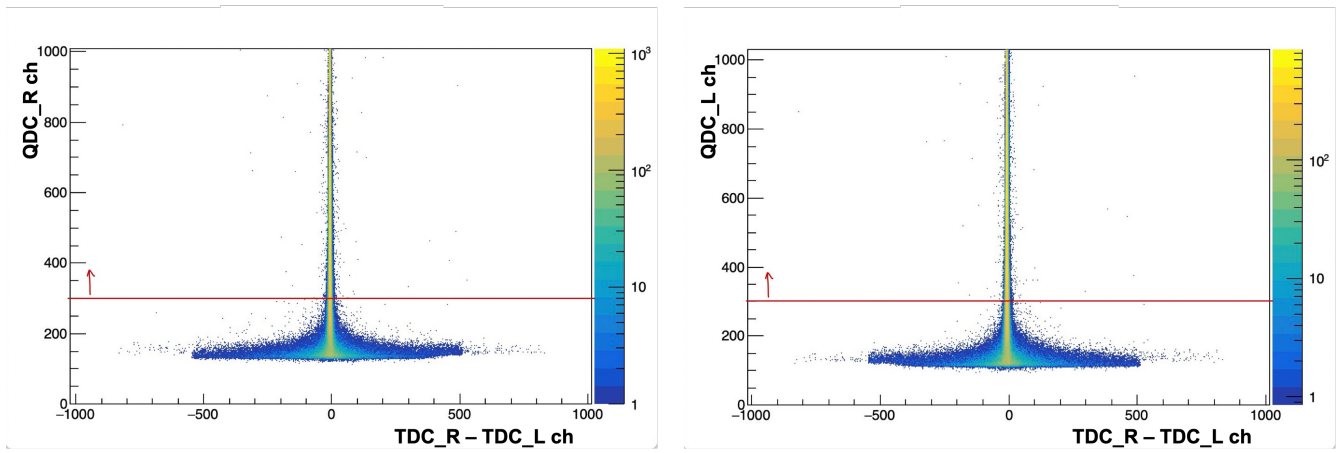
[1]TDC データ



[2]QDC データ

図 7.5　β 線検出器のヒストグラム

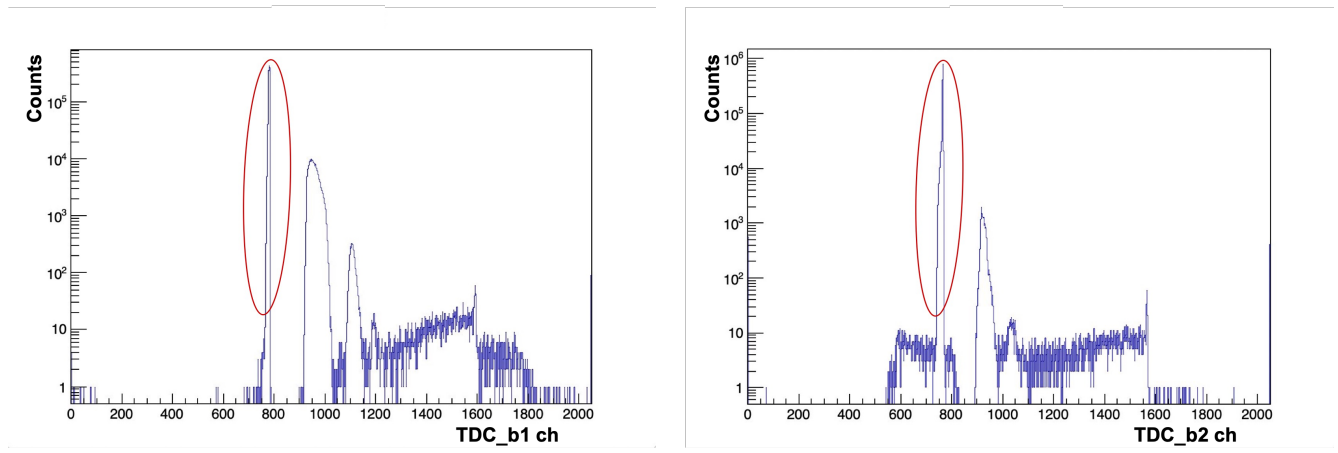
最後に新型中性子検出器の QDC データについてである。図 7.6 は、横軸に R と L のそれぞれの TDC データの引き算、縦軸に R 又は L の QDC データを取ったものである。図 7.6 からわかるように、あるエネルギー以下では TDC の引き算に広がりが見られる。これについては、本来 gain shift させるべきだが、今回は時間の都合で、300 ch 未満は切り捨てた。



[1]R の QDC データ　　[2]L の QDC データ

図 7.6　新型中性子検出器の QDC ヒストグラム

また、2 枚の β 線検出器の場合のヒストグラムの条件は、それぞれの TDC データのピークのみを取り出す。



[1]β-1 の TDC データ　　[2]β-2 の TDC データ

図 7.7　β 線検出器の TDC ヒストグラム

話を図 7.3 に戻すが、それぞれの σ について以下の連立方程式が成り立つ。

$$\begin{cases} \sigma_1 = \sqrt{{\sigma_n}^2 + {\sigma_{\beta 1}}^2} & (7.1) \\ \sigma_2 = \sqrt{{\sigma_n}^2 + {\sigma_{\beta 2}}^2} & (7.2) \\ \sigma_3 = \sqrt{{\sigma_{\beta 1}}^2 + {\sigma_{\beta 2}}^2} & (7.3) \end{cases}$$

これらを用いて、新型中性子検出器の時間分解能 σ_n を求める。結果が表 7.1 である。

表 7.1　時間分解能

	σ nsec
新型中性子検出器	0.47
β 線検出器 1	0.41
β 線検出器 2	0.37
hile	0.43

表 7.1 のように、新型中性子検出器は従来の小型中性子検出器 hile と同等の時間分解能を実現していることがわかる。[10]

7.2　エネルギー閾値の測定

7.2.1　エネルギー較正

エネルギー閾値を測定するために、初めにエネルギー較正を行う。今回は、^{241}Am、^{133}Ba、^{137}Cs、^{57}Co、^{22}Na の 5 種類の γ 線源を用いて、それらのコンプトン端や後方散乱ピーク、X 線ピーク、photo peak によって較正する。トリガー条件は、中性子検出器の 2 つの PMT 両方で同時に検出できた時である。

[1]^{137}Cs　　[2]^{241}Am

図 7.8　γ 線スペクトルの例

ヒストグラムの条件は、background の QDC データについて live time の比で較正したものを、差し引いたものである。

具体的な回路の概略図が図 7.9 である。

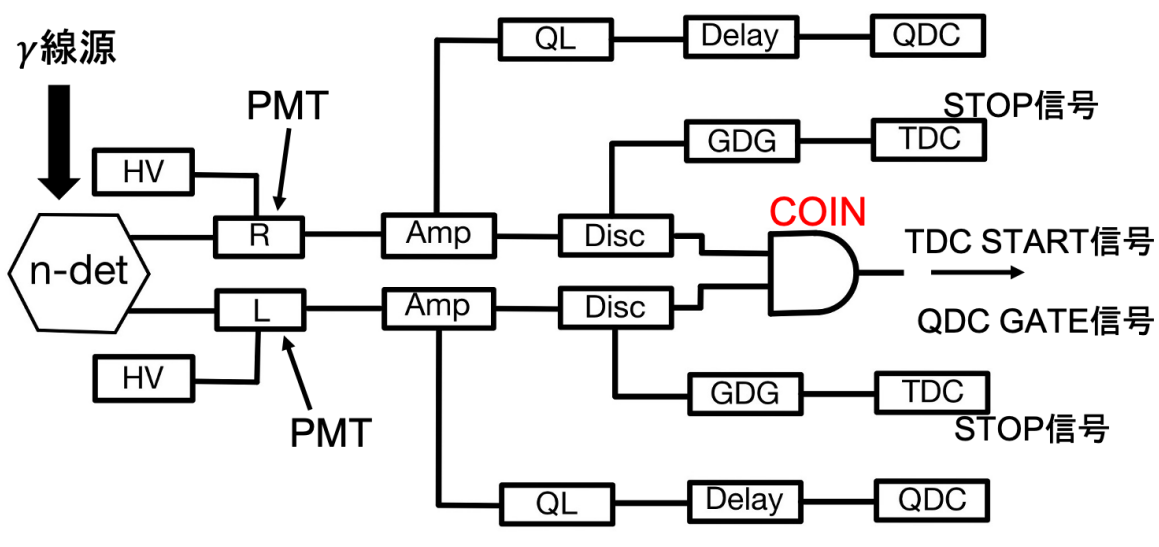


図 7.9　エネルギー較正実験の回路

エネルギー較正の結果が図 7.10 である。

[1] 低エネルギー領域　　[2] 高エネルギー領域

図 7.10　エネルギー較正

1：^{241}Am のコンプトン端、2：^{57}Co の photo peak
3：^{133}Ba の X 線ピーク、4：^{137}Cs の X 線ピーク
5：^{57}Co のコンプトン端、6：^{241}Am の後方散乱ピーク
7：^{57}Co の後方散乱ピーク、8：^{133}Ba のコンプトン端
9：^{22}Na のコンプトン端、10：^{137}Cs のコンプトン端
（低エネルギー順）

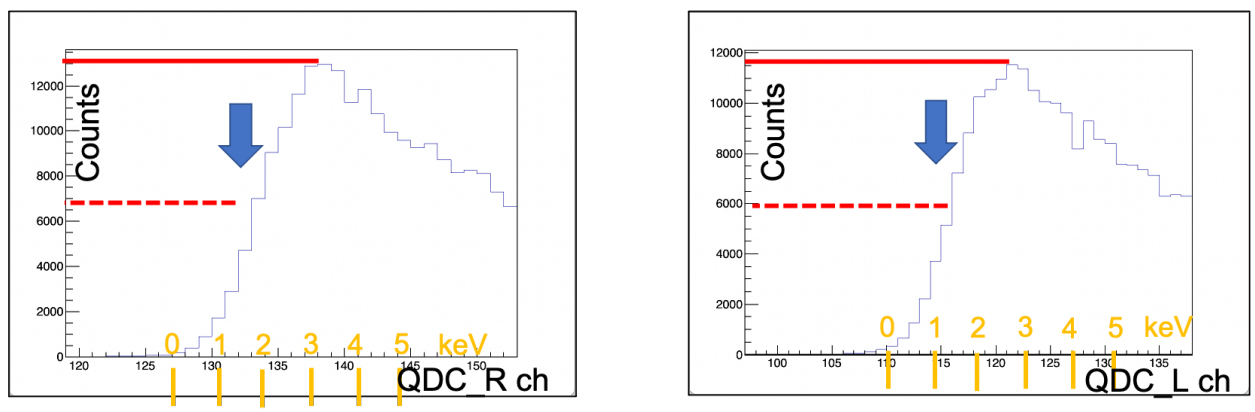
表 7.2　近似式

	低エネルギー領域	高エネルギー領域
PMT R	$y = -2.45 \times 10^{-4}x^2 + 0.295x$	$y = 0.197x - 15.9$
PMT L	$y = -1.86 \times 10^{-4}x^2 + 0.258x$	$y = 0.176x - 17.8$

図 7.10 のように、低エネルギー領域と高エネルギー領域で、別々の関数で近似せざるを得ないような結果が得られた。但し、誤差の決め方など定まっていない部分もあり、この結果については議論が必要である。今回は、この結果をそのまま用いる。

7.2.2　エネルギー閾値の測定

時間分解能の測定の実験の時に得られる QDC データを用いて、エネルギー閾値を測定する。



[1]PMT R　　[2]PMT L

図 7.11　エネルギー閾値

エネルギー較正の結果を用いると、表 7.3 のようになる。

表 7.3　エネルギー閾値

	QDC チャンネル [ch]	エネルギー閾値 [keV]
PMT R	133	1.8
PMT L	114	1.0

よって、エネルギー閾値は約 2 keV である。従来の小型中性子検出器 hile のエネルギー閾値が約 4 keV であるため、より低エネルギーの中性子の検出が期待される。

7.3　性能評価実験の現段階でのまとめ

新型中性子検出器の時間分解能は、0.47 nsec である。これは、従来の小型中性子検出器 hile と同等であったため、hile と同等の時間分解能を実現したといえる。

新型中性子検出器のエネルギー閾値は、約 2 keV である。陽子の発光率が電子の発光率のおよそ 1/10 であることを考慮して中性子のエネルギーで考えると、数十 keV 程度の低エネルギー中性子の測定が期待できる。

第 8 章

今後の課題

今回、新型中性子検出器の性能評価実験は、非常に初期段階のものしか行うことができていない。よって、今後もっと本格的な実験を行う必要がある。具体的には、実際の中性子を用いたテスト実験である。

また、エネルギー較正について、未だ方法が確立していない。今回、γ 線源を用いて行ったが、この方法を用いたのが本グループでは初めてであった為、測定方法や誤差の決め方など他の方法を試すなど、もっと試みるべきである。全エネルギー領域を一つの関数で近似できないという結果についての議論も必要である。

第 9 章

まとめ

中性子過剰核の構造を解明する為、β 遅延中性子崩壊を用いて研究している。そこで、数十 keV〜数 MeV のエネルギーをもつ β 遅発中性子のエネルギーの測定が必要な為、新型中性子検出器を開発した。新型中性子検出器に求められる性能は、幅広いエネルギー領域（数十 keV〜数 MeV）で検出効率が大きいこと、時間分解能 Δt_{TOF} がよいこと、集光率の場所依存性が見られないことである。それぞれの性能の為に必要な条件を考えた。

初めに、低エネルギーの検出効率を大きくする為、ノイズを落とすことを目的として PMT を 2 個使用した同時計測を行うこととし、また集光率の為に検出器を小型にすることとした。次に、高エネルギーの検出効率を大きくする為、シンチレータの厚さを今までで最も厚い 29 mm にした。続いて、よい時間分解能 Δt_{TOF} の為に、長い飛行距離（1500 mm）に設定し、小型なものを作成することとした。また、立体角を確保する為に、シンチレータの形を六角形にし、ライトガイドをシンチレータの後方に接続した。

集光率と集光率の場所依存性については、Geant4 によるシミュレーションを基に評価した。その結果、シンチレータ六角形の一辺の長さを 100 mm、ライトガイドの長さを 40 mm に決定した。また反射材について、シンチレータ入射面を乱反射材、シンチレータ側面とライトガイド側面を鏡面反射材を使用することとした。

以上の条件を満たす検出器を設計し、製作した。

β 線源や γ 線源を用いた新型中性子検出器の性能評価実験として、時間分解能とエネルギー閾値を測定した。その結果、時間分解能は、0.47 nsec、エネルギー閾値は、約 2 keV である。陽子の発光率が電子の発光率のおよそ 1/10 であることを考慮して中性子のエネルギーで考えると、数十 keV 程度の低エネルギー中性子の測定が期待できることがわかった。

但し、エネルギー較正の方法が確立していないことなど課題がある。また、高エネルギーの検出効率や集光率の場所依存性についての評価も必要である。そこで、実際の中性子を用いたテスト実験を行う必要がある。

謝辞

本研究を行うにあたり、多くの方々にご協力、ご助言を頂きました。本当に有り難う御座いました。

指導教員の小田原先生は、お忙しい中、多大なる時間を割いて指導して頂き、一緒に沢山の議論をして頂いたり、沢山のアドバイスや研究の方針を示して頂きました。物理や実験に関する知識の話だけでなく、今後の人生においてプラスになるような数多のことを教えて頂きました。また、本年度はコロナの影響により例年通りには行かないことも多々あり、本グループもカナダの TRIUMF へ行くことができませんでした。しかしながら、小田原さんの尽力もあり、個人的にはコロナ禍をほとんど感じずに研究を行うことができました。とても感謝しております。

共同研究者である下田先生や西畑先生には、普段の議論では得られないような的確な指摘やアドバイスを頂き、研究の方針が定まって行きました。本研究の重要な岐路では、毎回進むべき道筋をお二人のお言葉によって照らして頂きました。本当に有り難う御座いました。

D2 の飯村先輩には、ご自身の実験でとても忙しかったにも関わらず、メールや時にはビデオ会議ツールを利用して、Geant4 について右も左も分からない状態の私に、とても優しく丁寧に教えて頂きました。本研究は、飯村先輩のお力無しでは進めることができませんでした。本当に有り難う御座いました。M1 の大上先輩には、実験のご指導やアドバイス、更には研究に関する議論にもお付き合い頂きました。また、前回の実験を経験していない私に、前回の TRIUMF での実験について沢山教えて頂いたことは、新型中性子検出器の製作においてとても重要でした。有り難う御座いました。D1 の Fiza 先輩には、実験を手伝って頂きました。Fiza 先輩の協力なしでは、実験を行うことはできなかったと思います。有り難う御座いました。M2 の前島先輩には、TRIUMF での実験について、沢山教えて頂きました。また、沢山話しかけて頂き、気持ちが明るくなり研究への気持ちへの後押しとなりました。有り難う御座いました。

川畑研究室の他グループの先生や先輩、同期の皆さんにもとても感謝しております。特に、D2 の赤石先輩にはとてもお世話になりました。Geant4 の環境を整備して頂いたことや解析のマクロを教えて頂いたこと、とても感謝しております。本当に有り難う御座いました。

多くの方々の力に支えられて本研究を行うことができたこと、改めて深くお礼申し上げます。この 1 年間で学んだことを、今後の研究活動に活かしていきたいと思います。本当に有り難う御座いました。

参考文献

[1] H. Nishibata et al., Phys. Lett. B767, 81 (2017).

[2] JENDL-4.0 https://wwwndc.jaea.go.jp/jendl/j40/j40.html

[3] S. Agostinelli et al., NIM506, no. 3(2003) 250

[4] Allison et al., IEEE Trans. Nucl. Sci.53(2006) 270

[5] Geant4 Physics Reference Manual

[6] J. Allison et al., NIM A835(2016) 186

[7] M. Wang et. al., Chinese Phys. C 41, 3 (2017)

[8] https://geant4.web.cern.ch/node/1

[9] 飯村俊　大阪大学大学院　修士論文 (2019)

[10] 梅原基　大阪大学大学院　修士論文 (2019)

[11] https://www.crystals.saint-gobain.com/products/bc-408-bc-412-bc-416

付録 A

使用した Geant4 のコード

実行環境：CentOS Linux 7（バージョン 3.82.2)、コンパイラ：gcc（バージョン 4.8.5）
ライブラリ：Geant4.10.05

Geometry.cc シミュレーション空間に存在する構造物の定義

```
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
// Geometry.cc
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

//#include <CADMesh.hh>
#include "Geometry.hh"
#include "SensitiveDetectors.hh"
#include "MyMessenger.hh"

#include "G4Box.hh"
#include "G4Tubs.hh"
#include "G4EllipticalTube.hh"
#include "G4Polycone.hh"
#include "G4Polyhedra.hh"
#include "G4Trap.hh"
#include "G4Trd.hh"
#include "G4Tet.hh"
#include "G4Cons.hh"
#include "G4GenericTrap.hh"
#include "G4Sphere.hh"
#include "G4ExtrudedSolid.hh"
#include "G4ScaledSolid.hh"
#include "G4UnionSolid.hh"
#include "G4SubtractionSolid.hh"
#include "G4IntersectionSolid.hh"
#include "G4LogicalVolume.hh"
#include "G4PVPlacement.hh"
#include "G4VPhysicalVolume.hh"
#include "G4ThreeVector.hh"
#include "G4RotationMatrix.hh"
#include "G4Transform3D.hh"
#include "G4NistManager.hh"
```

```
#include "G4VisAttributes.hh"
#include "G4SystemOfUnits.hh"
#include "G4OpticalSurface.hh"
#include "G4LogicalBorderSurface.hh"
#include "G4LogicalSkinSurface.hh"
#include "G4UserLimits.hh"
#include "G4SDManager.hh"
//#include "CADMesh.hh"
#include "myParameter.hh"
//------------------------------------------------------------------------
Geometry::Geometry()
    : scinti_surf(0), reflector_surf(0), reflector_surf2(0), dielectric_surf(0), cath
ode_surf(0), fStepLimit(0), LV_world(NULL), fWorldMaterial(NULL), housing_ratio(1.000
1)
{
    //G4double minEkin = 1.0*eV;
    //fStepLimit = new G4UserLimits(DBL_MAX,DBL_MAX,DBL_MAX,minEkin);
    fMessenger = new MyMessenger(this);
}
//------------------------------------------------------------------------

//------------------------------------------------------------------------
Geometry::~Geometry()
{
    delete fStepLimit;
    delete fMessenger;
}
//------------------------------------------------------------------------
G4VPhysicalVolume *Geometry::Construct()
//------------------------------------------------------------------------
{
    using GVC = G4ThreeVector;
    using GTR = G4Transform3D;
    // Construct materials
    scinti_surf = new G4OpticalSurface("scinti_surface");
    reflector_surf = new G4OpticalSurface("reflector_surface");
    reflector_surf2 = new G4OpticalSurface("reflector_surface2");
    dielectric_surf = new G4OpticalSurface("dielectric_surface");
    cathode_surf = new G4OpticalSurface("cathode_surface");
    ConstructMaterials();
    ConstructSurfaces();
    //auto reflector = G4Material::GetMaterial("G4_TEFLON");
    //auto air = G4Material::GetMaterial("G4_AIR");
    auto BC408 = G4Material::GetMaterial("G4_PLASTIC_SC_VINYLTOLUENE");
    auto acrylic = G4Material::GetMaterial("G4_PLEXIGLASS");
    auto aluminum = G4Material::GetMaterial("G4_Al");
    auto vacuum = G4Material::GetMaterial("G4_Galactic");
    auto glass = G4Material::GetMaterial("G4_Pyrex_Glass");

    // Option to switch on/off checking of volumes overlaps
    G4bool checkOverlaps = true;

    // Define 'World Volume'
    // Define the shape of solid
    G4double leng_X_World = 4.0 * m; // X-full-length of world
```

```
    G4double leng_Y_World = 4.0 * m; // Y-full-length of world
    G4double leng_Z_World = 4.0 * m; // Z-full-length of world
    G4Box *SL_world = new G4Box("SL_world", leng_X_World / 2.0, leng_Y_World / 2.0, l
eng_Z_World / 2.0);
    // Define logical volume
    LV_world = new G4LogicalVolume(SL_world, vacuum, "LV_world", 0, 0, 0);
    // Placement of logical volume
    G4int copyNum_World = 0; // Set ID number of world
    G4VPhysicalVolume *PV_World = new G4PVPlacement(GTR(), "PV_World", LV_world, 0, f
alse, copyNum_World, checkOverlaps);

    // Define volumeID and variable

    G4int Mother_ID = 1;
    G4int Scinti_ID = 2;
    G4int Guide_ID = 3;
    G4int PMT_ID = 4;
    G4int Cathode_ID = 5;
    G4int PMT2_ID = 6;
    G4int Cathode2_ID = 7;

    GVC vct[10];
    vct[0] = GVC();
    vct[1] = GVC(0, 0, -127. * mm);
    vct[2] = GVC(-24. * mm, -24. * mm, -30.5 * mm);
    vct[3] = GVC(0, 0, 127. * mm);

    G4RotationMatrix rot[17], rot45y, rot90x, rot90y, rot31z, rot90z, rot180z, rot270
z;
    rot[0] = G4RotationMatrix();
    rot45y.rotateY(45 * deg);
    rot90x.rotateX(90 * deg);
    rot90y.rotateY(90 * deg);
    rot90z.rotateZ(90 * deg);
    rot180z.rotateZ(180 * deg);
    rot270z.rotateZ(270 * deg);

    // Build plastic scintillator

    //G4double Scinti_R = 10 * cm;
    G4double Thickness = 29 * mm;
    G4double PMT_W = 26.2 * mm;
    G4double PMT_L = 32.5 * mm;
    G4double PMT_C = 30 * mm;
    G4double Cathode_W = 23 * mm;
    G4double Cathode_T = 0.8 * mm;
    G4double Guide_L = 40 * mm;
    G4double Guide_S = 10 * mm;
    G4double Flight_length = 1.5 * m;
    G4double Hexagon_r = 100 * mm;
    G4double Hexagon_rr = Hexagon_r * sqrt(3) / 2 * mm;
    G4double Hexagon_length = sqrt(1500 * 1500 - Hexagon_r * Hexagon_r) * mm;
    G4double z1[] = {0, Thickness};
    G4double rI1[] = {0, 0};
    G4double rO1[] = {Hexagon_rr, Hexagon_rr};
```

```
    G4double z2[] = {-Guide_L / 2, Guide_L / 2};
    G4double PMT_W2 = (PMT_W + ((PMT_C - PMT_W) / 2)) * 2;
    G4double rO2[] = {Hexagon_rr, (PMT_W2 * sqrt(3) + PMT_W) / 4};

    G4Box *SL_Mother = new G4Box("SL_Mother", Hexagon_r * 1.1, Hexagon_r * 1.1, (Thic
kness + Guide_L + Guide_S + PMT_L) * 1.1);
    G4LogicalVolume *LV_Mother = new G4LogicalVolume(SL_Mother, LV_world->GetMaterial
(), "LV_Mother", 0, 0, 0);
    G4VPhysicalVolume *PV_Mother = new G4PVPlacement(GTR(rot90y, GVC(Flight_length, 0
, 0)), LV_Mother, "PV_Mother", LV_world, false, Mother_ID, checkOverlaps);

    G4Polyhedra *SL_Mother2 = new G4Polyhedra("SL_Mother2", 0 * degree, 360 * degree,
 6, 2, z1, rI1, rO1);
    G4LogicalVolume *LV_Mother2 = new G4LogicalVolume(SL_Mother2, LV_world->GetMateri
al(), "LV_Mother2", 0, 0, 0);

    auto rotatoin_1 = G4RotationMatrix(0., 0., 0.);
    auto position_1 = G4ThreeVector(0, 0, Hexagon_length - Flight_length - Thickness)
;
    auto trans3D_1 = G4Transform3D(rotatoin_1, position_1);
    G4VPhysicalVolume *PV_Mother2 = new G4PVPlacement(trans3D_1, LV_Mother2, "PV_Moth
er2", LV_Mother, false, Mother_ID, checkOverlaps);

    G4Polyhedra *SL_Scinti = new G4Polyhedra("SL_Scinti", 0 * degree, 360 * degree, 6
, 2, z1, rI1, rO1);
    G4LogicalVolume *LV_Scinti = new G4LogicalVolume(SL_Scinti, BC408, "LV_Scinti", 0
, 0, 0);

    auto rotatoin = G4RotationMatrix(0., 0., 0.);
    auto position = G4ThreeVector(0, 0, Hexagon_length - Flight_length);
    auto trans3D = G4Transform3D(rotatoin, position);
    G4VPhysicalVolume *PV_Scinti = new G4PVPlacement(trans3D, LV_Scinti, "PV_Scinti",
 LV_Mother, false, Scinti_ID, checkOverlaps);
    G4Trd *SL_Guide1 = new G4Trd("SL_Guide1", Hexagon_r, PMT_W2 / 2, (Hexagon_r * sqr
t(3)) / 2, PMT_W / 2, Guide_L / 2);
    G4Polyhedra *SL_Guide2 = new G4Polyhedra("SL_Guide2", 0 * degree, 360 * degree, 6
, 2, z2, rI1, rO2);
    G4IntersectionSolid *SL_Guide3 = new G4IntersectionSolid("SL_Guide3", SL_Guide1,
SL_Guide2);
    G4Box *SL_Guide4 = new G4Box("SL_Guide4", PMT_W2 / 2, PMT_W / 2, Guide_S / 2);

    auto rotatoin_4 = G4RotationMatrix(0., 0., 0.);
    auto position_4 = G4ThreeVector(0, 0, (Guide_L / 2) - Guide_S + (Thickness / 2));
    auto trans3D_4 = G4Transform3D(rotatoin_4, position_4);
    G4UnionSolid *SL_Guide = new G4UnionSolid("SL_Guide", SL_Guide3, SL_Guide4, trans
3D_4);
    G4LogicalVolume *LV_Guide = new G4LogicalVolume(SL_Guide, acrylic, "LV_Guide", 0,
 0, 0);

    auto rotatoin_5 = G4RotationMatrix(0., 0., 0.);
    auto position_5 = G4ThreeVector(0, 0, Hexagon_length - Flight_length + (Guide_L /
 2) + Thickness);
    auto trans3D_5 = G4Transform3D(rotatoin_5, position_5);
    G4VPhysicalVolume *PV_Guide = new G4PVPlacement(trans3D_5, LV_Guide, "PV_Guide",
LV_Mother, false, Guide_ID, checkOverlaps);
```

```
    G4Box *SL_PMT = new G4Box("SL_PMT", PMT_W / 2, PMT_W / 2, PMT_L / 2);
    G4LogicalVolume *LV_PMT = new G4LogicalVolume(SL_PMT, glass, "LV_PMT", 0, 0, 0);

    auto rotatoin_6 = G4RotationMatrix(0., 0., 0.);
    auto position_6 = G4ThreeVector(-PMT_C / 2, 0, Hexagon_length - Flight_length + T
hickness - 1.0 + Guide_L + Guide_S + (PMT_L / 2));
    auto trans3D_6 = G4Transform3D(rotatoin_6, position_6);
    G4VPhysicalVolume *PV_PMT = new G4PVPlacement(trans3D_6, LV_PMT, "PV_PMT", LV_Mot
her, false, PMT_ID, checkOverlaps);

    G4Box *SL_PMT_Cathode = new G4Box("SL_PMT_Cathode", Cathode_W / 2, Cathode_W / 2,
 Cathode_T / 2);
    G4LogicalVolume *LV_PMT_Cathode = new G4LogicalVolume(SL_PMT_Cathode, aluminum, "
LV_PMT_Carhode", 0, 0, 0);

    auto rotatoin_7 = G4RotationMatrix(0., 0., 0.);
    auto position_7 = G4ThreeVector(0, 0, -(PMT_L / 2) + 3 * (Cathode_T / 2));
    auto trans3D_7 = G4Transform3D(rotatoin_7, position_7);
    G4VPhysicalVolume *PV_PMT_Cathode = new G4PVPlacement(trans3D_7, LV_PMT_Cathode,
"PV_PMT_Cathode", LV_PMT, false, Cathode_ID, checkOverlaps);

    G4Box *SL_PMT2 = new G4Box("SL_PMT2", PMT_W / 2, PMT_W / 2, PMT_L / 2);
    G4LogicalVolume *LV_PMT2 = new G4LogicalVolume(SL_PMT2, glass, "LV_PMT2", 0, 0, 0
);

    auto rotatoin_8 = G4RotationMatrix(0., 0., 0.);
    auto position_8 = G4ThreeVector(PMT_C / 2, 0, Hexagon_length - Flight_length + Th
ickness - 1.0 + Guide_L + Guide_S + (PMT_L / 2));
    auto trans3D_8 = G4Transform3D(rotatoin_8, position_8);
    G4VPhysicalVolume *PV_PMT2 = new G4PVPlacement(trans3D_8, LV_PMT2, "PV_PMT2", LV_
Mother, false, PMT2_ID, checkOverlaps);

    G4Box *SL_PMT_Cathode2 = new G4Box("SL_PMT_Cathode2", Cathode_W / 2, Cathode_W /
2, Cathode_T / 2);
    G4LogicalVolume *LV_PMT_Cathode2 = new G4LogicalVolume(SL_PMT_Cathode2, aluminum,
 "LV_PMT_Carhode2", 0, 0, 0);

    auto rotatoin_9 = G4RotationMatrix(0., 0., 0.);
    auto position_9 = G4ThreeVector(0, 0, -(PMT_L / 2) + 3 * (Cathode_T / 2));
    auto trans3D_9 = G4Transform3D(rotatoin_9, position_9);
    G4VPhysicalVolume *PV_PMT_Cathode2 = new G4PVPlacement(trans3D_9, LV_PMT_Cathode2
, "PV_PMT_Cathode2", LV_PMT2, false, Cathode2_ID, checkOverlaps);

    // Define colors        -> SetVisAttributes( new G4VisAttributes(TRUE,G4Colour(re
d  ,green,blue ,Opacity)) );

    LV_world->SetVisAttributes(new G4VisAttributes(TRUE, G4Colour(1.0, 1.0, 1.0, 0.1)
));        //white 可視化FALSEのすると描画バグることあり。変な影が見えた。
    LV_Mother->SetVisAttributes(new G4VisAttributes(TRUE, G4Colour(1.0, 1.0, 1.0, 0.0
)));       //無色透明
    LV_Mother2->SetVisAttributes(new G4VisAttributes(TRUE, G4Colour(1.0, 1.0, 1.0, 0.
0)));      //無色透明
    LV_Scinti->SetVisAttributes(new G4VisAttributes(TRUE, G4Colour(0.0, 1.0, 1.0, 1.0
)));       //cyan
```

```
    LV_Guide->SetVisAttributes(new G4VisAttributes(TRUE, G4Colour(0.0, 1.0, 0.0, 0.4)
));        //green
    LV_PMT->SetVisAttributes(new G4VisAttributes(TRUE, G4Colour(0.45, 0.25, 0.0, 0.8)
));        //brown
    LV_PMT_Cathode->SetVisAttributes(new G4VisAttributes(TRUE, G4Colour(1.0, 1.0, 0.0
, 1.0)));  //yellow
    LV_PMT2->SetVisAttributes(new G4VisAttributes(TRUE, G4Colour(0.45, 0.25, 0.0, 0.8
)));        //brown
    LV_PMT_Cathode2->SetVisAttributes(new G4VisAttributes(TRUE, G4Colour(1.0, 1.0, 0.
0, 1.0))); //yellow

    /////// 有感検出器
    G4SDManager *SDman = G4SDManager::GetSDMpointer();
    SensitiveDetectors *MySensitiveDetectors = new SensitiveDetectors("MySensitiveDet
ectors");
    LV_Scinti->SetSensitiveDetector(MySensitiveDetectors);       // Add sensitivity t
o the logical volume
    LV_PMT_Cathode->SetSensitiveDetector(MySensitiveDetectors);  // Add sensitivity t
o the logical volume
    LV_PMT_Cathode2->SetSensitiveDetector(MySensitiveDetectors); // Add sensitivity t
o the logical volume
    SDman->AddNewDetector(MySensitiveDetectors);

    // ------------- Surfaces --------------

    new G4LogicalBorderSurface("Scinti_reflector", PV_Scinti, PV_Mother, reflector_su
rf);
    new G4LogicalBorderSurface("Guide_reflector", PV_Guide, PV_Mother, reflector_surf
2);
    new G4LogicalBorderSurface("Border_U1", PV_Scinti, PV_Guide, dielectric_surf);
    new G4LogicalBorderSurface("Border_D1", PV_Guide, PV_Scinti, dielectric_surf);
    new G4LogicalBorderSurface("Border_U2", PV_Guide, PV_PMT, dielectric_surf);
    new G4LogicalBorderSurface("Border_D2", PV_PMT, PV_Guide, dielectric_surf);
    new G4LogicalBorderSurface("Cathode_SF", PV_PMT, PV_PMT_Cathode, cathode_surf);
    new G4LogicalBorderSurface("Scinti_reflector2", PV_Scinti, PV_Mother2, scinti_sur
f);
    new G4LogicalBorderSurface("Border_U4", PV_Guide, PV_PMT2, dielectric_surf);
    new G4LogicalBorderSurface("Border_D4", PV_PMT2, PV_Guide, dielectric_surf);
    new G4LogicalBorderSurface("Cathode_SF2", PV_PMT2, PV_PMT_Cathode2, cathode_surf)
;

    return PV_World;
}
//-----------------------------------------------------------------------------
//-----------------------------------------------------------------------------
void Geometry::ConstructMaterials()
//-----------------------------------------------------------------------------
{
    auto nistManager = G4NistManager::Instance();

    G4double photonEnergy[] =
        {2.384 * eV, 2.431 * eV, 2.480 * eV, 2.530 * eV,
         2.583 * eV, 2.638 * eV, 2.695 * eV, 2.755 * eV,
         2.818 * eV, 2.883 * eV, 2.952 * eV, 3.024 * eV,
         3.100 * eV, 3.179 * eV, 3.263 * eV, 3.351 * eV,
```

```
        3.444 * eV};
    const G4int nEntries = sizeof(photonEnergy) / sizeof(G4double);

    G4Material *air = nistManager->FindOrBuildMaterial("G4_AIR"); // Air
    G4double refractiveIndexA[] = {1.00, 1.00, 1.00, 1.00,
                                   1.00, 1.00, 1.00, 1.00,
                                   1.00, 1.00, 1.00, 1.00,
                                   1.00, 1.00, 1.00, 1.00,
                                   1.00};
    assert(sizeof(refractiveIndexA) == sizeof(photonEnergy));
    G4MaterialPropertiesTable *MPT_air = new G4MaterialPropertiesTable();
    MPT_air->AddProperty("RINDEX", photonEnergy, refractiveIndexA, nEntries);
    G4cout << "Air G4MaterialPropertiesTable" << G4endl;
    MPT_air->DumpTable();
    air->SetMaterialPropertiesTable(MPT_air);

    G4Material *vacuum = nistManager->FindOrBuildMaterial("G4_Galactic"); // vacuum
    G4double refractiveIndexV[] = {1.00, 1.00, 1.00, 1.00,
                                   1.00, 1.00, 1.00, 1.00,
                                   1.00, 1.00, 1.00, 1.00,
                                   1.00, 1.00, 1.00, 1.00,
                                   1.00};
    assert(sizeof(refractiveIndexV) == sizeof(photonEnergy));
    G4MaterialPropertiesTable *MPT_vac = new G4MaterialPropertiesTable();
    MPT_vac->AddProperty("RINDEX", photonEnergy, refractiveIndexV, nEntries);
    G4cout << "Vacuum G4MaterialPropertiesTable" << G4endl;
    MPT_vac->DumpTable();
    vacuum->SetMaterialPropertiesTable(MPT_vac);

    G4Material *BC408 = nistManager->FindOrBuildMaterial("G4_PLASTIC_SC_VINYLTOLUENE"
);
    G4double refractiveIndexP[] = {1.58, 1.58, 1.58, 1.58,
                                   1.58, 1.58, 1.58, 1.58,
                                   1.58, 1.58, 1.58, 1.58,
                                   1.58, 1.58, 1.58, 1.58,
                                   1.58};
    assert(sizeof(refractiveIndexP) == sizeof(photonEnergy));
    G4double absorptionP[] = {3800 * mm, 3800 * mm, 3800 * mm, 3800 * mm,
                              3800 * mm, 3800 * mm, 3800 * mm, 3800 * mm,
                              3800 * mm, 3800 * mm, 3800 * mm, 3800 * mm,
                              3800 * mm, 3800 * mm, 3800 * mm, 3800 * mm,
                              3800 * mm};
    assert(sizeof(absorptionP) == sizeof(photonEnergy));
    G4double scintilFast[] = {0.02, 0.04, 0.07, 0.12,
                              0.18, 0.26, 0.41, 0.54,
                              0.82, 1.0, 0.82, 0.5,
                              0.21, 0.11, 0.07, 0.03,
                              0.02};
    assert(sizeof(scintilFast) == sizeof(photonEnergy));
    G4double scintilSlow[] = {0.02, 0.04, 0.07, 0.12,
                              0.18, 0.26, 0.41, 0.54,
                              0.82, 1.0, 0.82, 0.5,
                              0.21, 0.11, 0.07, 0.03,
                              0.02};
    assert(sizeof(scintilSlow) == sizeof(photonEnergy));
```

```
    G4MaterialPropertiesTable *MPT_pla = new G4MaterialPropertiesTable();
    MPT_pla->AddProperty("RINDEX", photonEnergy, refractiveIndexP, nEntries)
        ->SetSpline(true);
    MPT_pla->AddProperty("ABSLENGTH", photonEnergy, absorptionP, nEntries)
        ->SetSpline(true);
    MPT_pla->AddProperty("FASTCOMPONENT", photonEnergy, scintilFast, nEntries)
        ->SetSpline(true);
    MPT_pla->AddProperty("SLOWCOMPONENT", photonEnergy, scintilSlow, nEntries)
        ->SetSpline(true);
    MPT_pla->AddConstProperty("SCINTILLATIONYIELD", 8000. / MeV);
    //MPT_pla->AddConstProperty("PROTONSCINTILLATIONYIELD",17./MeV);
    MPT_pla->AddConstProperty("RESOLUTIONSCALE", 1.0);
    MPT_pla->AddConstProperty("FASTTIMECONSTANT", 2.1 * ns);
    MPT_pla->AddConstProperty("FASTSCINTILLATIONRISETIME", 0.9 * ns);
    MPT_pla->AddConstProperty("SLOWTIMECONSTANT", 2.1 * ns);
    MPT_pla->AddConstProperty("SLOWSCINTILLATIONRISETIME", 0.9 * ns);
    MPT_pla->AddConstProperty("YIELDRATIO", 1.0);
    G4cout << "BC-408 G4MaterialPropertiesTable" << G4endl;
    MPT_pla->DumpTable();
    BC408->SetMaterialPropertiesTable(MPT_pla);
    BC408->GetIonisation()->SetBirksConstant(birks_coefficient * mm / MeV);

    G4Material *PMT_glass = nistManager->FindOrBuildMaterial("G4_Pyrex_Glass");
    G4double refractiveIndexG[] = {1.473, 1.473, 1.473, 1.473,
                                   1.473, 1.473, 1.473, 1.473,
                                   1.473, 1.473, 1.473, 1.473,
                                   1.473, 1.473, 1.473, 1.473,
                                   1.473};
    assert(sizeof(refractiveIndexG) == sizeof(photonEnergy));
    G4double absorptionG[] = {100 * mm, 100 * mm, 100 * mm, 100 * mm,
                              100 * mm, 100 * mm, 100 * mm, 100 * mm,
                              100 * mm, 100 * mm, 100 * mm, 100 * mm,
                              100 * mm, 100 * mm, 100 * mm, 100 * mm,
                              100 * mm, 100 * mm, 100 * mm, 100 * mm,
                              100 * mm, 100 * mm, 100 * mm, 100 * mm,
                              100 * mm, 100 * mm, 100 * mm, 100 * mm,
                              100 * mm};
    assert(sizeof(absorptionG) == sizeof(photonEnergy));
    G4MaterialPropertiesTable *MPT_glass = new G4MaterialPropertiesTable();
    MPT_glass->AddProperty("RINDEX", photonEnergy, refractiveIndexG, nEntries)
        ->SetSpline(true);
    MPT_glass->AddProperty("ABSLENGTH", photonEnergy, absorptionG, nEntries)
        ->SetSpline(true);
    G4cout << "PMT glass G4MaterialPropertiesTable" << G4endl;
    MPT_glass->DumpTable();
    PMT_glass->SetMaterialPropertiesTable(MPT_glass);

    G4Material *acrylic = nistManager->FindOrBuildMaterial("G4_PLEXIGLASS"); // アク>
リル
    G4double refractiveIndexAC[] = {1.5, 1.5, 1.5, 1.5,
                                    1.5, 1.5, 1.5, 1.5,
                                    1.5, 1.5, 1.5, 1.5,
                                    1.5, 1.5, 1.5, 1.5,
                                    1.5};
    assert(sizeof(refractiveIndexAC) == sizeof(photonEnergy));
    G4double absorptionAC[] = {2000 * mm, 2000 * mm, 2000 * mm, 2000 * mm,
                               2000 * mm, 2000 * mm, 2000 * mm, 2000 * mm,
                               2000 * mm, 2000 * mm, 2000 * mm, 2000 * mm,
```

```
                                2000 * mm, 2000 * mm, 2000 * mm, 2000 * mm,
                                2000 * mm};
    assert(sizeof(absorptionAC) == sizeof(photonEnergy));
    G4MaterialPropertiesTable *MPT_acrylic = new G4MaterialPropertiesTable();
    MPT_acrylic->AddProperty("RINDEX", photonEnergy, refractiveIndexAC, nEntries)
        ->SetSpline(true);
    MPT_acrylic->AddProperty("ABSLENGTH", photonEnergy, absorptionAC, nEntries)
        ->SetSpline(true);
    G4cout << "Acrylic Light guide G4MaterialPropertiesTable" << G4endl;
    MPT_acrylic->DumpTable();
    acrylic->SetMaterialPropertiesTable(MPT_acrylic);

    nistManager->FindOrBuildMaterial("G4_TEFLON"); // TEFLON
    nistManager->FindOrBuildMaterial("G4_Al");     // Aluminium

    G4cout << G4endl << "The materials defined are : " << G4endl << G4endl;
    G4cout << *(G4Material::GetMaterialTable()) << G4endl;
}
//------------------------------------------------------------------------------
void Geometry::ConstructSurfaces()
//------------------------------------------------------------------------------
{
    const G4int num = 2;
    G4double ephoton[num] = {2.034 * eV, 4.136 * eV};

    scinti_surf->SetType(dielectric_dielectric);
    scinti_surf->SetModel(unified);
    //scinti_surf->SetFinish(polished); //without reflector
    scinti_surf->SetFinish(polishedbackpainted);
    //scinti_surf->SetFinish(groundbackpainted);
    G4double RefractiveIndex5[num] = {1.0, 1.0}; //無限に薄い層の屈折率
    //G4double reflectivity5[num] = {0.98, 0.98};  //反射材の反射率 //On_Specular
    G4double reflectivity5[num] = {0.96, 0.96}; //反射材の反射率 //On_lambert
    //G4double specularSpike5[num] = {1.0, 1.0}; //反射材は鏡面反射 //On_Specular
    G4double specularSpike5[num] = {0.0, 0.0}; //反射材は乱反射 //On_lambert
    G4double specularLobe5[num] = {0.0, 0.0};  //On_lambert
    G4double backScatter5[num] = {0.0, 0.0};   //On_lambert
    G4MaterialPropertiesTable *MPT_scinti = new G4MaterialPropertiesTable();
    MPT_scinti->AddProperty("RINDEX", ephoton, RefractiveIndex5, num);
    MPT_scinti->AddProperty("REFLECTIVITY", ephoton, reflectivity5, num);
    MPT_scinti->AddProperty("SPECULARSPIKECONSTANT", ephoton, specularSpike5, num);
    MPT_scinti->AddProperty("SPECULARLOBECONSTANT", ephoton, specularLobe5, num); //O
n_lambert
    MPT_scinti->AddProperty("BACKSCATTERCONSTANT", ephoton, backScatter5, num);   //O
n_lambert
    G4cout << "Scinti Surface G4MaterialPropertiesTable" << G4endl;
    MPT_scinti->DumpTable();
    scinti_surf->SetMaterialPropertiesTable(MPT_scinti);

    // 反射材で誘電体を覆っている表面。無限に薄い空気層を仮定している。
    reflector_surf->SetType(dielectric_dielectric);
    reflector_surf->SetModel(unified);
    reflector_surf->SetFinish(polishedbackpainted);
    //reflector_surf->SetFinish(polished); //without reflector
    //reflector_surf->SetFinish(groundbackpainted);
```

```
    G4double RefractiveIndex[num] = {1.0, 1.0}; //無限に薄い層の屈折率
    G4double reflectivity1[num] = {0.98, 0.98}; //反射材の反射率 //On_Specular
    //G4double reflectivity1[num] = {0.96, 0.96}; //反射材の反射率 //On_lambert
    G4double specularSpike1[num] = {1.0, 1.0}; //反射材は鏡面反射 //On_Specular
    //G4double specularSpike1[num] = {0.0, 0.0}; //反射材は乱反射 //On_lambert
    //G4double specularLobe1[num] = {0.0, 0.0};  //On_lambert
    //G4double backScatter1[num] = {0.0, 0.0};   //On_lambert
    //ランバート反射率は(1 -Lobe-Spike-back)
    G4MaterialPropertiesTable *MPT_reflector = new G4MaterialPropertiesTable();
    MPT_reflector->AddProperty("RINDEX", ephoton, RefractiveIndex, num);
    MPT_reflector->AddProperty("REFLECTIVITY", ephoton, reflectivity1, num);
    MPT_reflector->AddProperty("SPECULARSPIKECONSTANT", ephoton, specularSpike1, num)
;
    //MPT_reflector->AddProperty("SPECULARLOBECONSTANT", ephoton, specularLobe1, num)
; //On_lambert
    //MPT_reflector->AddProperty("BACKSCATTERCONSTANT", ephoton, backScatter1, num);
  //On_lambert
    G4cout << "Reflector Surface G4MaterialPropertiesTable" << G4endl;
    MPT_reflector->DumpTable();
    reflector_surf->SetMaterialPropertiesTable(MPT_reflector);

    // 反射材で誘電体を覆っている表面。無限に薄い空気層を仮定している。
    reflector_surf2->SetType(dielectric_dielectric);
    reflector_surf2->SetModel(unified);
    reflector_surf2->SetFinish(polishedbackpainted);
    //reflector_surf2->SetFinish(polished); //without reflector
    //reflector_surf2->SetFinish(groundbackpainted);
    G4double RefractiveIndex2[num] = {1.0, 1.0}; //無限に薄い層の屈折率
    G4double reflectivity4[num] = {0.98, 0.98};  //反射材の反射率 //On_Specular
    //G4double reflectivity4[num] = {0.96, 0.96}; //反射材の反射率 //On_lambert
    G4double specularSpike4[num] = {1.0, 1.0}; //反射材は鏡面反射 //On_Specular
    //G4double specularSpike4[num] = {0.0, 0.0}; //反射材は乱反射 //On_lambert
    //G4double specularLobe4[num] = {0.0, 0.0};  //On_lambert
    //G4double backScatter4[num] = {0.0, 0.0};   //On_lambert
    //ランバート反射率は(1 -Lobe-Spike-back)
    G4MaterialPropertiesTable *MPT_reflector2 = new G4MaterialPropertiesTable();
    MPT_reflector2->AddProperty("RINDEX", ephoton, RefractiveIndex2, num);
    MPT_reflector2->AddProperty("REFLECTIVITY", ephoton, reflectivity4, num);
    MPT_reflector2->AddProperty("SPECULARSPIKECONSTANT", ephoton, specularSpike4, num
);
    //MPT_reflector2->AddProperty("SPECULARLOBECONSTANT", ephoton, specularLobe4, num
); //On_lambert
    //MPT_reflector2->AddProperty("BACKSCATTERCONSTANT", ephoton, backScatter4, num);
   //On_lambert
    G4cout << "Reflector Surface2 G4MaterialPropertiesTable" << G4endl;
    MPT_reflector2->DumpTable();
    reflector_surf2->SetMaterialPropertiesTable(MPT_reflector2);

    // 誘電体同士の表面。フレネルの法則に従う。
    dielectric_surf->SetType(dielectric_dielectric);
    dielectric_surf->SetModel(unified);
    dielectric_surf->SetFinish(polished);
    //pla_glass->SetSigmaAlpha(sigma_alpha);
    G4double reflectivity2[num] = {0.999, 0.999};
    G4double specularSpike2[num] = {1.0, 1.0}; //物質の平均表面からの反射
```

```
    //ランバート反射率は(1 -Lobe-Spike-back)
    G4MaterialPropertiesTable *MPT_dielectric = new G4MaterialPropertiesTable();
    MPT_dielectric->AddProperty("REFLECTIVITY", ephoton, reflectivity2, num);
    MPT_dielectric->AddProperty("SPECULARSPIKECONSTANT", ephoton, specularSpike2, num
);
    G4cout << "dielectric - dielectric Surface G4MaterialPropertiesTable" << G4endl;
    MPT_dielectric->DumpTable();
    dielectric_surf->SetMaterialPropertiesTable(MPT_dielectric);

    // PMTのphotocathodeの表面。
    cathode_surf->SetType(dielectric_metal);
    cathode_surf->SetModel(glisur);
    cathode_surf->SetFinish(polished);
    G4double reflectivity3[num] = {0.0, 0.0};
    G4double photocath_EFF[num] = {1., 1.}; //Enables 'detection' of photons
    //G4double photocath_ReR[num]={1.92,1.92};
    //G4double photocath_ImR[num]={1.69,1.69};
    G4MaterialPropertiesTable *MPT_cathode = new G4MaterialPropertiesTable();
    MPT_cathode->AddProperty("REFLECTIVITY", ephoton, reflectivity3, num);
    MPT_cathode->AddProperty("EFFICIENCY", ephoton, photocath_EFF, num);
    //MPT_cathode->AddProperty("REALRINDEX",     ephoton,photocath_ReR,num);
    //MPT_cathode->AddProperty("IMAGINARYRINDEX",ephoton,photocath_ImR,num);
    G4cout << "Photo Cathode Surface G4MaterialPropertiesTable" << G4endl;
    MPT_dielectric->DumpTable();
    cathode_surf->SetMaterialPropertiesTable(MPT_cathode);
}
void Geometry::SetWorldMaterial(G4String materialName)
{
    G4NistManager *nistManager = G4NistManager::Instance();

    G4Material *pttoMaterial =
        nistManager->FindOrBuildMaterial(materialName);

    if (fWorldMaterial != pttoMaterial)
    {
        if (pttoMaterial)
        {
            fWorldMaterial = pttoMaterial;
            if (LV_world)
                LV_world->SetMaterial(fWorldMaterial);
            G4cout
                << G4endl
                << "----> The World is made of " << materialName << G4endl;
        }
        else
        {
            G4cout
                << G4endl
                << "-->  WARNING from SetWorldMaterial : "
                << materialName << " not found" << G4endl;
        }
    }
}
```

PhysicsList.cc 考慮する物理相互作用の選択

```
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
// PhysicsList.cc
//  [Note] Based on "G4 Basic Example: B3"
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
#include "PhysicsList.hh"
#include "G4HadronElasticPhysicsHP.hh"
#include "G4StoppingPhysics.hh"
#include "G4IonPhysics.hh"
#include "G4DecayPhysics.hh"
#include "G4RadioactiveDecayPhysics.hh"
#include "G4EmStandardPhysics.hh"
#include "G4EmLivermorePolarizedPhysics.hh"
#include "G4EmLivermorePhysics.hh"
#include "G4EmPenelopePhysics.hh"
#include "G4OpticalPhysics.hh"
#include "G4StepLimiterPhysics.hh"
//#include "GammaNuclearPhysics.hh"
#include "QGSP_BERT_HP.hh"
#include "G4HadronPhysicsQGSP_BERT_HP.hh"
#include "G4HadronicProcessStore.hh"
//#include "G4fissionEvent.hh"
#include "myParameter.hh"

//------------------------------------------------------------------------------
PhysicsList::PhysicsList()
        : G4VModularPhysicsList()
//------------------------------------------------------------------------------
{
    G4int verbose = 0;
    verboseLevel = 0;
// Default physics
    RegisterPhysics(new G4DecayPhysics(verbose));


// EM physics
    RegisterPhysics(new G4EmLivermorePhysics(verbose));
    //RegisterPhysics(new G4EmLivermorePolarizedPhysics(verbose));
    //RegisterPhysics(new G4EmPenelopePhysics(verbose));//included positron, used pil
s-san
    //RegisterPhysics(new G4EmLowEPPhysics(verbose));
    //RegisterPhysics(new G4EmDNAPhysics(verbose));

// Hadron Elastic scattering
    RegisterPhysics( new G4HadronElasticPhysicsHP(verbose) );

// Hadron Physics
    RegisterPhysics( new G4HadronPhysicsQGSP_BERT_HP(verbose));
    G4HadronicProcessStore::Instance()->SetVerbose(verbose);
```

```
// Stopping Physics
    RegisterPhysics( new G4StoppingPhysics(verbose));

// Ion Physics
    RegisterPhysics( new G4IonPhysics(verbose));

// Gamma-Nuclear Physics
     //RegisterPhysics( new GammaNuclearPhysics("gamma"));

// Optical physics
    G4OpticalPhysics* MyOpticalPhysics = new G4OpticalPhysics(verbose);
    RegisterPhysics( MyOpticalPhysics );
    MyOpticalPhysics->Configure(kCerenkov, _photon_tracking_);
    MyOpticalPhysics->Configure(kScintillation, _photon_tracking_);

// Radioactive decay
    RegisterPhysics(new G4RadioactiveDecayPhysics(verbose));

// User limits
    //RegisterPhysics(new G4StepLimiterPhysics(verbose));
}
//....oooOO0OOooo.....oooOO0OOooo.....oooOO0OOooo.....oooOO0OOooo.....oooOO0OOooo....
PhysicsList::~PhysicsList()
//....oooOO0OOooo.....oooOO0OOooo.....oooOO0OOooo.....oooOO0OOooo.....oooOO0OOooo....
{}

//....oooOO0OOooo.....oooOO0OOooo.....oooOO0OOooo.....oooOO0OOooo.....oooOO0OOooo....
void PhysicsList::SetCuts()
//....oooOO0OOooo.....oooOO0OOooo.....oooOO0OOooo.....oooOO0OOooo.....oooOO0OOooo....
{
    G4VUserPhysicsList::SetCuts();
}
```

SensitiveDetectors.cc　シミュレーション結果情報の取得

```
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
// SensitiveDetectors.cc
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
#include "SensitiveDetectors.hh"

#include "G4TouchableHistory.hh"
#include "G4Step.hh"
#include "G4VProcess.hh"
#include "G4HCofThisEvent.hh"
#include "G4RunManager.hh"
#include "Analysis.hh"
#include "G4SystemOfUnits.hh"
#include "G4ParticleGun.hh"
```

```
#include "PrimaryGenerator.hh"
#include "myParameter.hh"

//-----------------------------------------------------------------------------
SensitiveDetectors::SensitiveDetectors(G4String name) //each SESSION
        : G4VSensitiveDetector(name)
//-----------------------------------------------------------------------------
{
        event = 0;
        runtmp = 0;
}
//-----------------------------------------------------------------------------
SensitiveDetectors::~SensitiveDetectors()
//-----------------------------------------------------------------------------
{
}
//-----------------------------------------------------------------------------
void SensitiveDetectors::Initialize(G4HCofThisEvent *) //begin of each EVENT
//-----------------------------------------------------------------------------
{
        nPhoton = 0;
        hitPhoton = 0;
        init_E = 0.;
        En = 0.;
        edep = 0.;
        hittime = 100. * ns;
        firsttime = 100. * ns;
        decaytime = -100. * ns;
        preTrack = 0;
        event++;
        if (runid > runtmp)
        {
                runtmp = runid;
                event = 1;
        }
        mVol_ID = 0;
        neutron_ID = 1;
        Scinti_E = {0.};
        Scinti_E_ee = {0.};
        Scinti_T = {0.};
        Scinti_direct = {0};
        Scinti_phot = {0};
        Scinti_PMT_E = {0};
        Scinti_PMT_T = {0};
        Scinti_PMT2_E = {0}; //PMT_added
        Scinti_PMT2_T = {0}; //PMT_added
        Scinti_hitpos = {0., 0., 0.};
        //std::cout<<"+++++ eventID="<<event<<" +++++"<<std::endl;
}
//-----------------------------------------------------------------------------
void SensitiveDetectors::EndOfEvent(G4HCofThisEvent *) //end of each EVENT
//-----------------------------------------------------------------------------
{
        if (En > 0)
        {
```

```
                //PrimaryGenerator* mysetup;
                //Ekin = mysetup->fpParticleGun->GetParticleEnergy();
                G4AnalysisManager *analysisManager = G4AnalysisManager::Instance();
                analysisManager->FillNtupleIColumn(0, runid);
                analysisManager->FillNtupleIColumn(1, event);
                analysisManager->FillNtupleDColumn(2, En);
                analysisManager->AddNtupleRow();
                //std::cout << " emit photon is " << nPhoton << std::endl;
        }
}
//-----------------------------------------------------------------------------
G4bool SensitiveDetectors::ProcessHits(G4Step *aStep, G4TouchableHistory *) //each ST
EP
//-----------------------------------------------------------------------------
{
        //const G4Step* cStep = aStep;
        G4int nTrack = aStep->GetTrack()->GetTrackID();
        G4int volumeID = aStep->GetPreStepPoint()->GetTouchableHandle()->GetCopyNumbe
r();
        //G4cout<<volumeID<<G4endl;
        switch (volumeID)
        {
        case 2: //in the Scinti
                if (nTrack == 1)
                {
                        En = aStep->GetTrack()->GetVertexKineticEnergy();
                        if ((0 > aStep->GetDeltaEnergy()) && ((Scinti_T[0] == 0) || (
Scinti_T[0] > aStep->GetPostStepPoint()->GetGlobalTime())))
                        {
                                //if ((0 < aStep->GetTotalEnergyDeposit()) && ((Scint
i_T[0] == 0)||(Scinti_T[0] > aStep->GetPostStepPoint()->GetGlobalTime()))) {
                                Scinti_T[0] = aStep->GetPostStepPoint()->GetGlobalTim
e();
                                Scinti_hitpos[0] = aStep->GetPostStepPoint()->GetPosi
tion().x();
                                Scinti_hitpos[1] = aStep->GetPostStepPoint()->GetPosi
tion().y();
                                Scinti_hitpos[2] = aStep->GetPostStepPoint()->GetPosi
tion().z();
                                if (nTrack != neutron_ID)
                                {
                                        Scinti_direct[0] = 3; //inelastic, others
                                }
                                else if (aStep->GetPreStepPoint()->GetKineticEnergy()
 == aStep->GetTrack()->GetVertexKineticEnergy())
                                {
                                        Scinti_direct[0] = 1; //direct
                                }
                                else
                                {
                                        Scinti_direct[0] = 2; //elastic
                                }
                        }
                }
        ▒       Scinti_E[0] += aStep->GetTotalEnergyDeposit();
```

```
                                if ((Scinti_PMT_T[0] == 0) || (Scinti_PMT_T[0] > aSte
p->GetPostStepPoint()->GetGlobalTime()))
                                {
                                        Scinti_PMT_T[0] = aStep->GetPostStepPoint()->
GetGlobalTime();
                                }
                        }
                }
                break;

        //PMT_added
        case 6: //in the Scinti PMT2
                if ("opticalphoton" == aStep->GetTrack()->GetParticleDefinition()->Ge
tParticleName())
                {
                        //      G4cout<<aStep->GetPostStepPoint()->GetTouchable()->Ge
tCopyNumber()<<G4endl;
                        if (7 == aStep->GetPostStepPoint()->GetTouchable()->GetCopyNu
mber())
                        { //hit photocathode
                                Scinti_PMT2_E[0]++;
                                if ((Scinti_PMT2_T[0] == 0) || (Scinti_PMT2_T[0] > aS
tep->GetPostStepPoint()->GetGlobalTime()))
                                {
                                        Scinti_PMT2_T[0] = aStep->GetPostStepPoint()-
>GetGlobalTime();
                                }
                        }
                }
                break;
        }

        return true;
}
```