# Détecteur SDHCAL pour le signal $e^+e^- \longrightarrow \nu\nu h$ : Optimisation et Adaptation de l'analyse de données pour le Projet FCC

Alexia HOCINE

Étudiante en M2 Physique SUBA à l'UCBL1

Supervisé par Gérald Grenier

Maître de Conférence - UCBL1
Enseignant-Chercheur - IP2I, CNRS, IN2P3
Membre des collaborations CALICE, CMS et ILD
Corresponsable du groupe SDHCAL au sein de la collaboration CALICE



Rapport de Stage - Master 2 Physique SUBA
Université de Claude Bernard Lyon 1

Avril-Juillet 2022

# Préambule

## Remerciements

Je souhaite d'abord remercier Gérald Grenier pour m'avoir donner la chance de montrer ce que je peux faire. Et aussi pour son encadrement, son accompagnement et son temps.

Je souhaite plus largement remercier mon équipe, Gérald Grenier, Imad Laktineh et Clément Devanne, pour l'atmosphère positive, détendue et stimulante.

Et plus largement, les employés de l'IP2I pour leur gentillesse et leur accueil.

## Participation à la *Geek and Japan Touch*

Au cours de mon stage, j'ai participé à l'atelier tenu par l'IP2I à la *Geek and Japon Touch*, organisé par Stéphanie Beauceron (IP2I, CNRS, CMS).

Durant ce week-end, avec 2 autres chercheuses (non physiciennes), Florence Boyer et Liliane De Araujo, on a tenu un débat sur le film *Don't look up : Déni Cosmique* de Adam McKay, sur la crédibilité du discours scientifiques.

Ensuite sur le stand, j'ai pu expliquer les bases scientifiques et des recherches menées par le CNRS et CMS, ainsi que Virgo au près du grand publique.

De plus, j'ai aussi animé le stand de l'association ? ? dont l'objectif est d'expliquer les principes de base de la gravité en 2D avec un drap tenu.

# Table des matières

# Chapitre 1

# Introduction

## 1.1 Physique des collisionneurs

Le principe des collisionneurs est simple, on accélère des particules à des énergies cinétiques suffisantes pour provoquer des collisions inélastiques, et ainsi comprendre les interactions fondamentales et les constituants élémentaires de la physique.

On distingue 2 familles de collisionneurs en fonction des particules qui sont utilisés.

### 1.1.1 Collisionneurs hadroniques

Les collisionneurs hadroniques utilisent des hadrons, qui sont des particules complexes composées de 3 quarks et de gluons [1]. En pratique au LHC (Large Hadron Collider) du CERN, on utilise des protons, 2 quarks up et un quark down.

Comme il s'agit de particules composites, ce sont pas le protons qui collisionnent directement mais ces constituants, appelés partons. Chacun porte une fraction indéterminée de l'énergie du proton. Ce qui permet d'avoir des énergies de collisions inconnues en amont. C'est pourquoi, ils sont utiles pour la découverte de nouvelles particules de masse inconnue, puisqu'il permettent de balayer tout le spectre de masse sous la gamme d'énergie du collisionneur (au LHC $< 14$ `TeV`) [2].

### 1.1.2 Collisionneurs leptoniques

En revanche, les collisionneurs leptoniques utilisent des leptons, qui sont des particules élémentaires. Comme le LEP (Large Electron-Positron), le prédécesseur du LHC, qui collisionnait des électrons et des positrons [3].

Cette fois-ci, chaque lepton qui collisionne, possède une énergie complète. Donc on connait parfaitement leur énergie, puisque on peut leurs impulser une énergie choisie et ainsi augmenter la statistique pour un niveau d'énergie précis. Ces collisionneurs sont donc utilisés pour la recherche de précision.

Les prochaines générations de collisionneurs, comme ILC, CEPC, CLIC et FCC, ce sont des collisionneurs leptoniques. Leur objectif est de préciser les données du LHC, notamment sur le boson de Higgs découvert en 2012 par le LHC, qui était la pièce manquante du modèle standard des particules, car il permet aux particules d'acquérir une masse.

## 1.2 Physique du boson de Higgs

### 1.2.1 Production du boson de Higgs

Concrètement, on ne mesure pas directement le boson de Higgs mais les particules qu'il produit sous la forme de jets. Ainsi on veut améliorer la résolutions en énergie de ces jets que l'on détecte [8].

### 1.2.2 Détecteur

En physique des particules, on utilise des détecteurs appelés calorimètres pour mesurer l'énergie des particules. Cette énergie va être déposer par ionisation avec le matériau le long de la trajectoire des particules qui le

1. Gluon : boson médiateurs de l'interaction forte qui maintiennent les quarks ensembles.
2. D'où l'intérêt de nouveaux collisionneurs à des énergies plus élevées et donc des masses de particules produites plus lourdes.

traverse. Il faut donc des algorithmes de reconstruction pour déduire les énergies, les types de particules et les trajectoires.

Pour cela, on utilise des calorimètres à grande granularité qui permet une très bonne performance des Algorithmes de Flux de Particules (PFA) [8].

C'est dans ce cadre que la collaboration internationale CALICE, à développer le premier prototype de la famille de calorimètre granulaire SDHCAL, pour Semi-Digital Hadronic CALorimeter, qui a été développer en grande partie à l'IP2I dans l'équipe CMS, auquel j'appartiens pour ce stage.

### 1.2.3 Collisions

Au cours, de ce stage, je me concentrerais sur les collisions de type `nnh` pour neutrino-neutrino-higgs.
D'où voici les diagrammes de Feynman :

## 1.3 Présentation & Objectif du Stage

Pour ce stage, j'ai récupéré les codes de Guillaume Garillot, qui les a développé en 2021 au cours de son post-doctorat à l'IP2I. Ils sont en libre accès à l'adresse `https://github.com/ggarillot/nnhAnalysis/tree/refactor`.

Ce programme `nnhAnalysis` permet l'étude de fichiers `SLCIO` pour la collision :

$$e^- e^+ \longrightarrow \nu\nu h \tag{1.1}$$

Et l'analyse des canaux de désintégration :

$$h \longrightarrow WW^\star \longrightarrow qqqq \tag{1.2}$$
$$h \longrightarrow b\bar{b} \tag{1.3}$$

Pour cela, il a utilisé les suites logiciels de `iLCSoft`, `https://github.com/iLCSoft` (plus précisément `LCIO` et `Marlin`), qui sont les anciennes suites logicielles. Mais les nouveaux projets de collisionneurs changent de suites logicielles et passent à `Key4HEP` et `Gaudi`.

Mon objectif est double. Dans un premier temps, comprendre et optimiser les codes existants, c'est-à-dire les programmes `LCIO`, `Marlin` et ceux de `nnhAnalysis`. Puis, je vais devoir transformer `nnhAnalysis` pour qu'il puisque correspondre aux nouvelles normes `Key4HEP` et `Gaudi`.

# Chapitre 2

# Programme `ILC`

## 2.1 Présentation du Projet ILC

Le projet ILC (International Linear Collider) est un collisionneur linéaire, électron-positron, de 31 km conçu pour atteindre une énergie de centre de masse de 500 `GeV`[1].

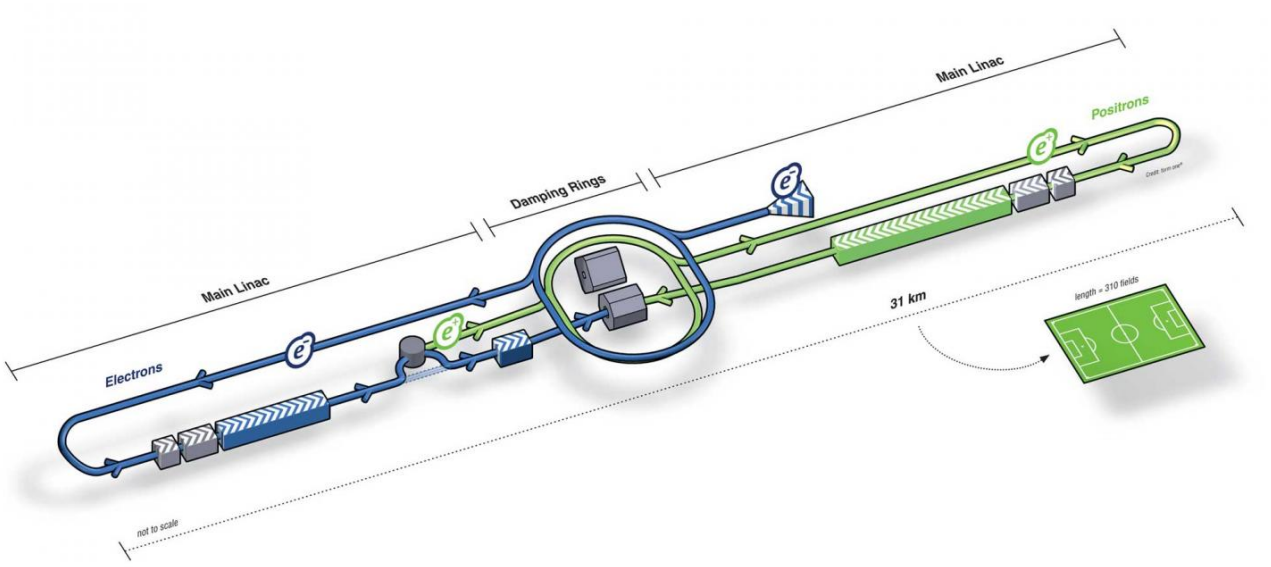


Figure 2.1 – Schéma ILC[1]

L'objectif de l'ILC est de produire beaucoup de boson de Higgs notamment pour découvrir s'il y en a d'autre génération du boson de Higgs. Et plus globalement pour rechercher de la nouvelle physique, par de nouveaux écarts avec le Modèle Standard.

Ce projet est toujours en attente pour commencer sa construction, probablement dans les montagne du Nord du Japon. Et le détecteur SDHCAL est en course pour y être installer. C'est pourquoi, l'IP2I développe des programme d'analyse en parallèle de ce détecteur.

## 2.2 Projet numérique : `original`

Pour ce stage j'ai récupéré les codes de Guillaume Garillot, qui les a développé en 2021 au cours de son post-doctorat à l'IP2I.

Son programme permet l'identification et l'analyse statistique de collision du détecteur SDHCAL.

### 2.2.1 Données initiales

La première étape est de récupérer les données (aujourd'hui simulées, plus tard obtenues dans le détecteur) sur le serveur distant où elles sont stockées, grâce au programme `miniDSTMaker`.

Mais pour le temps du stage je n'ai pas obtenu l'accès à ce serveur donc je n'ai pas pu utilisé cette partie de code. Je l'ai donc pas réutilisé dans ma propre version.

Initialement, on m'a mis à disposition certains de ces fichiers `SLCIO`, ceux des collisions de 250 `GeV`. Chaque fichier est rangés dans un des 66 dossiers (Figure 2.2), qui correspond au code du type de processus.

```
/gridgroup/ilc/nnhAnalysisFiles/AHCAL
(base) [          AHCAL]$ ls
402001  402007  402013  500006  500066  500078  500090  500101  500107  500115  500122
402002  402008  402014  500008  500068  500080  500092  500102  500108  500116  500124
402003  402009  402173  500010  500070  500082  500094  500103  500110  500117  500125
402004  402010  402176  500012  500072  500084  500096  500104  500112  500118  500126
402005  402011  402182  500062  500074  500086  500098  500105  500113  500119  500127
402006  402012  402185  500064  500076  500088  500100  500106  500114  500120  500128
```

FIGURE 2.2 – Les noms des dossiers qui correspondent aux numéros de processus

Ce dossier a été placé sur le serveur local de l'IP2I : `/gridgroup/ilc/nnhAnalysisFiles/AHCAL`.

| 2 leptoniques | 2 hadroniques | 4 hadroniques | |
|---|---|---|---|
| 500006, 500008 | 500010, 500012 | 500062, 500064, 500066, 500068, 500070, 500072 | 500074, 500076, 500078, 500080, 50 |

FIGURE 2.3 – Signification des codes des processus

### 2.2.2 Conversion des fichiers initiaux en fichiers `ROOT`

Grâce au programme `processor` on va pouvoir convertir les fichiers initiaux `SLCIO` en fichiers `ROOT` standards, afin de pouvoir les analyser.

On obtient ainsi pour chaque dossier de fichier de donnée `SLCIO` un fichier `ROOT` en sortie, c'est-à-dire que l'on obtiendra un arbre `ROOT` par type de processus.

Ce programme doit être robuste et donc à partir des mêmes fichiers toujours générer les fichiers `ROOT` strictement identiques.

### 2.2.3 Analyses des collisions

#### Étape 1

Avant de commencer l'analyse des fichiers `ROOT` générés précédemment, on va terminer ce que le programme `processor` avait commencé, et fusionner l'intégralité de ces fichiers en un seul gros fichier `DATA.root`, grâce à la commande `hadd`. Cette commande a été développé par le CERN et elle fusionne tous les histogrammes de différents fichiers en un seul.

Donc là aussi tous les fichiers `DATA.root` doivent être strictement identique.

#### Étape 2

Pour la deuxième étape de ce programme `analysis` on va de nouveau séparer nos données en 4 fichiers distincts. Mais au lieu de les séparer par leur numéro de processus, qui est un critère numérique, on va les diviser par le type de particules produit par le boson de Higgs, soit $b\bar{b}$, soit $WW^{\star}$, et par la polarisation de particules incidentes, c'est-à-dire l'électron et le positron avec une polarisation de -0,8 et 0,3 soit nulles. Ce qui va nous créer les 4 fichiers suivants :

Dans chacun de ces fichiers les arbres `ROOT`, `TTree`, auront les variables suivantes :

**`isSignal`** variable booléen qui indique si on considère l'évènement comme du signal ou du bruit de fond.

**`channelType`** c'est un entier qui représente le type de canaux de l'évènement :

- 2 fermions leptoniques ou hadronique
- 4 fermions leptoniques ou semi-leptonique ou hadroniques
- une autre type de résultat avec un boson de Higgs

split_bb_e-0.8_p+0.3.root
split_bb_e+0_p+0.root
split_ww_e-0.8_p+0.3.root
split_ww_e+0_p+0.root

FIGURE 2.4 – Fichiers où les évènements sont séparés en fonction des caractéristiques des polarisations des particules incidentes et des types de particules résultantes.

**isTrain** booléen qui confirme si l'évènement a été entrainé ou testé par la BDT.

**preSelected** booléen, si l'évènement a été pré-sélectionné par la BDT.

**weight** nombre flottant qui est le poids de l'évènement en $fb^{-1}$ de la luminosité intégrée.

Pour trier les données du fichier `DATA.root`, on va utiliser des arbres de décision, BDT pour *Boosted Decision Tree*.

Il s'agit d'une structure de donnée numérique binaire, qui permet de déterminer la nature des particules en répondant à des questions booléennes (FIGURE 2.5).

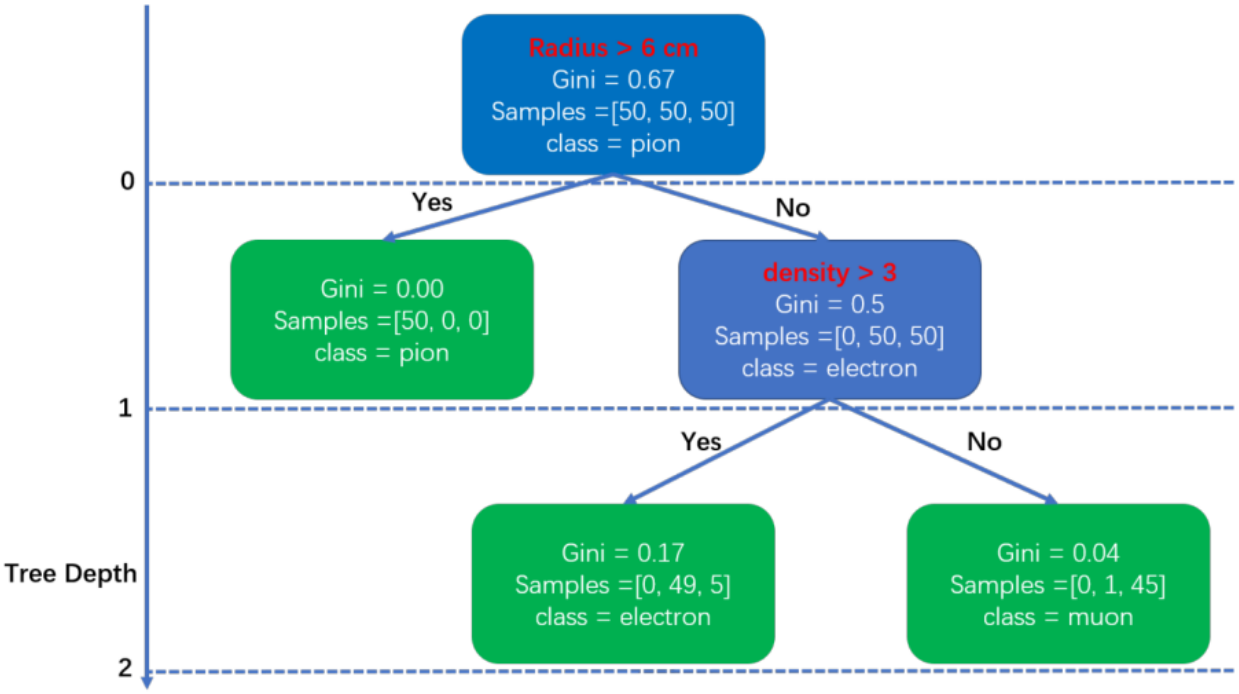


FIGURE 2.5 – Exemple de BDT [8]. Ici, elle permet de déterminer de le type d'une particule

Une fois encore l'API du CERN, nous aura été très utile. En effet, il existe déjà des méthodes (expliquées avec des tutoriels).

Mais cette fois-ci les fichiers créés sont équivalents mais pas identique. Car les BDT utilisent la génération de nombre aléatoire, ce qui engendre des variations dans les entraînements.

### Étape 3

Et pour terminer, on peut effectuer l'analyse à proprement parlé de nos données à partir des fichiers `split_XX.root` (FIGURE **??**).

model_bb_e-0.8_p+0.3.joblib
model_ww_e-0.8_p+0.3.joblib

FIGURE 2.6 – Fichiers du model de la BDT

scores_bb_e-0.8_p+0.3.root
scores_ww_e-0.8_p+0.3.root

FIGURE 2.7 – Les fichiers scores ont la réponse de la BDT : 2 variables pour chaque évènement, un booléen pour savoir s'il est sélectionné et la valeur retournée par la BDT

bestSelection_bb_e-0.8_p+0.3.root
bestSelection_ww_e-0.8_p+0.3.root

FIGURE 2.8

stats_bb_e-0.8_p+0.3.joblib
stats_ww_e-0.8_p+0.3.joblib

FIGURE 2.9 – Statistiques.

Et même si cette analyse sera toujours la même quelque soit les fichiers `split_XX.root`, ces derniers étant légèrement différent d'un entraînement à l'autre les résultats statistiques seront, là encore, légèrement différent mais doivent rester équivalent.

## 2.3 Projet numérique `ilcsoft`

### 2.3.1 Données

Initialement, on m'a mis à disposition des fichiers `SLCIO`, qui sera le format de fichier du détecteur. Chaque fichier est rangés dans un des 66 dossiers (Figure 2.10), qui correspond au code du type de processus.

```
/gridgroup/ilc/nnhAnalysisFiles/AHCAL
(base) [                AHCAL]$ ls
402001  402007  402013  500006  500066  500078  500090  500101  500107  500115  500122
402002  402008  402014  500008  500068  500080  500092  500102  500108  500116  500124
402003  402009  402173  500010  500070  500082  500094  500103  500110  500117  500125
402004  402010  402176  500012  500072  500084  500096  500104  500112  500118  500126
402005  402011  402182  500062  500074  500086  500098  500105  500113  500119  500127
402006  402012  402185  500064  500076  500088  500100  500106  500114  500120  500128
```

FIGURE 2.10 – Les noms des dossiers qui correspondent aux numéros de processus

### 2.3.2 Programme : `processor`

#### Méthodes

On cherche à convertir ces fichiers `SLCIO` en arbre `ROOT` par processus.

#### Résultats

Chaque dossier de fichier de donnée `SLCIO` produira un fichier `ROOT` en sortie, c'est-à-dire que l'on obtiendra un arbre `ROOT` par processus.

#### Interprétation

### 2.3.3 Programme `analysis`

#### Données

On récupère les fichiers `ROOT` du programme `processor` précédent.
*hadd* qui va créer le fichier DATA.root

### Méthodes

`BDT` Entrainement

### L'analyse

### Résultats

**Vérification des résultats** Comparaison entre les différents séries d'analyse, basée sur les même fichiers `ROOT`, mais un autre entraînement de BDT.

### Interprétation

# Chapitre 3

# Programme FCC

## 3.1 Projet FCC

### 3.1.1 Présentation

Le FCC (Futur Collisionneur Circulaire) est le projet du CERN pour remplacer leur collisionneur actuelle, le LHC (Large Hadronic Collider). Dont la fin de l'exploitation est prévu en 2040 [4]

Pour le FCC, on prévoit un anneau de 100 km, contre 27 km pour le LEP et le LHC (comme montrer Figure 3.1). Ce qui devrait nous permettra d'atteindre une énergie de 100 TeVcontre 13 TeVactuellement pour le LHC.

L'objectif est de rechercher d'une nouvelle physique, en mettant au jour de déviation avec le modèle standard.

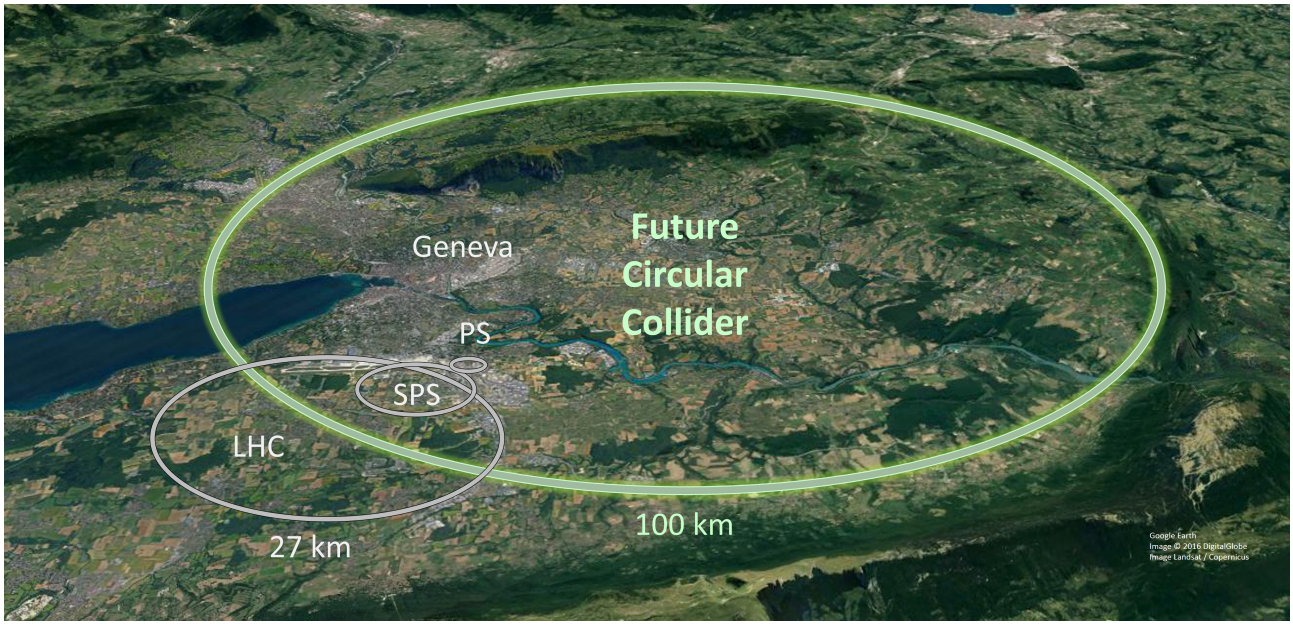


FIGURE 3.1 – https ://cds.cern.ch/images/OPEN-PHO-ACCEL-2019-001-2

## 3.2 Développement Numérique

Mon objectif dans ce stage est de transformer les codes développés par Guillaume GARILLOT lors de son post-doctorat pour le projet ILC pour ce projet qui n'utilise pas les mêmes suites logiciels.

`Gaudi`
`EDM4hep`

## 3.3 Travail de Stage

## 3.4 Comparaison avec iLCSoft

# Chapitre 4

# Outils Numériques

## 4.1 nnhScript

https://github.com/alexhxia/nnhAnalysis/tree/main/nnhScript

## 4.2 nnhTest

Pour tester les programmes générer avec nnhProgram, j'ai développé 4 programmes en python :

| • | Processus | Analysis |
|---|---|---|
| Completed | testProcessorCompleted.py | testAnalysisCompleted.py |
| Same | testProcessorSame.py | testAnalysisSame.py |

FIGURE 4.1 – Tableau récapitulatif des fonctions de tests

### 4.2.1 Programmes testXxCompleted.py

L'objectif de ce type de programme est de tester si tous les fichiers ont été généré.

#### Programmes testProcessorCompleted.py

Le processus est complet si tous les dossiers de

#### Programmes testAnalysisCompleted.py

### 4.2.2 Programmes testXxSame.py

#### Programmes testProcessorSame.py

#### Programmes testAnalysisSame.py

https://github.com/alexhxia/nnhAnalysis/tree/main/nnhTest

# Chapitre 5

# Résultats Physiques

## 5.1 Physique du Higgs

### 5.1.1 Nombre de Higgs attendu

### 5.1.2 Propriétés à Préciser

# Chapitre 6

# Conclusion

## 6.1 Travail effectué

## 6.2 Résultats attendus

## 6.3 Au delà

# Annexe A

# Résumé du travail effectué

## A.1 Bibliographie

**https://tel.archives-ouvertes.fr/tel-03405418**
— Étude du calorilmètre hadronique semi-digital et étude du canal physique

$$e^- e^+ \longrightarrow \nu\nu h \ (H \longrightarrow WW \longrightarrow qqqq)$$

au collisionneur circulaire électron positon (CEPC)
— Bing Liu, IP2I
— 2020

**https://tel.archives-ouvertes.fr/tel-02141420**
— Étude des gerbes hadroniques dans un calorimètre à grande granularité et étude du canal

$$e^- e^+ \longrightarrow HZ \ (Z \longrightarrow qq)$$

dans les futurs collisionneurs leptoniques
— Guillaume Garillot, IPNL
— 2019

## A.2 Tutoriels

**LCIO** https://github.com/iLCSoft/LCIO

**ILDConfig** https://github.com/iLCSoft/ILDConfig

**Marlin** https://github.com/iLCSoft/Marlin

**key4hep** https://github.com/key4hep/k4MarlinWrapper

## A.3 Code initial

https://github.com/ggarillot/nnhAnalysis/tree/refactor

## A.4 Code final

https://github.com/alexhxia/nnhAnalysis

# Annexe B

# Organisation du Projet

## B.1 Organisation initiale

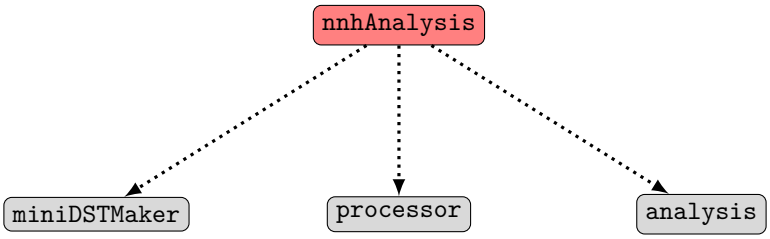


FIGURE B.1 – Organisation des dossiers de mon Projet - `https://github.com/ggarillot/nnhAnalysis/tree/refactor`

## B.2 Organisation finale

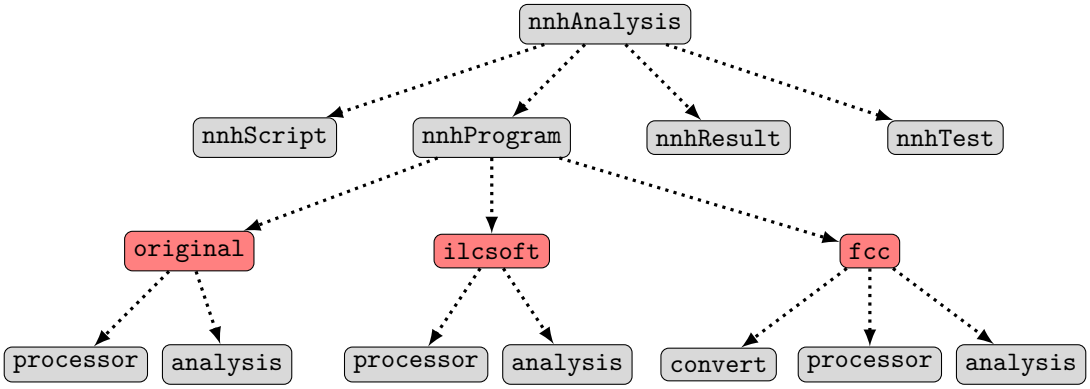


FIGURE B.2 – Organisation des dossiers de mon Projet - `https://github.com/alexhxia/nnhAnalysis`

## B.3 Le dossier `NNH_HOME`

Pour s'exécuter, le projet a besoin de la variable d'environnement `NNH_HOME` qui est le chemin du programme que vous souhaitez exécuter, mis en avant en rouge dans les Figure B.1 et Figure B.2.

Donc dans le projet initial, il s'agissait de `NNH_HOME=\nnhAnalysis` et dans le nouveau projet :

- `NNH_HOME = \nnhAnalysis\nnhProgram\original`
- `NNH_HOME = \nnhAnalysis\nnhProgram\ilcsoft`
- `NNH_HOME = \nnhAnalysis\nnhProgram\fcc`

# Annexe C

# Fichiers `ROOT` de sorties du programme `processor`

On a vu précédemment que `processor` converti une série de fichier `SLCIO` en fichiers `ROOT`.
Il considère deux canaux de désintégration du Higgs :

$$h \longrightarrow b\overline{b} \tag{C.1}$$
$$h \longrightarrow WW^{\star} \longrightarrow qqqq \tag{C.2}$$

La première équation (C.1) sera mesuré par le détecteur sous la forme de 2 jets reconstruit et identifié comme 2 quark `b`.
Alors que la seconde équation (C.2) sera reconstruit comme 4 jets, 2 par boson `W`[1].

1. `W` sur couche de masse, `Wstar` hors couche de masse

# Table des figures

# Bibliographie


[1] CERN. International linear collider ready for construction, jun 2013. https ://home.web.cern.ch/news/news/accelerators/international-linear-collider-ready-construction.

[2] CERN. Grand collisionneur de hadrons, juin 2022. https ://home.cern/fr/science/accelerators/large-hadron-collider.

[3] CERN. The large electron-positron collider, juin 2022. https ://home.cern/science/accelerators/large-electron-positron-collider.

[4] CERN. Le futur collisionneur circulaire, jun 2022. https ://home.cern/fr/science/accelerators/future-circular-collider.

[5] Nicolas Chadeau. Étude sur la mesure de la constante de couplage ghw w dans le cadre de l'expérience ilc à 250 gev. Master's thesis, UCBL1, 2021.

[6] Frank Gaede, Ties Behnke, Norman Graf, and Tony Johnson. Lcio : A persistency framework for linear collider simulation studies. *eConf*, C0303241 :TUKT001, 2003.

[7] Guillaume Garillot. *Étude des gerbes hadroniques dans un calorimètre à grande granularité et étude du canal e+e- → HZ (Z → qq) dans les futurs collisionneurs leptoniques.* Theses, Université de Lyon, February 2019.

[8] Bing Liu. *Etude du calorilmètre hadronique semi-digital et étude du canal physique e+ e- -H nu nu ( H -> WW->qq qq) au collisionneur circulaire electron positon (CEPC).* Theses, Université de Lyon ; Shanghai Jiao Tong University, November 2020.