HNL simulation and analysis in FairSHiP

Introduction

Elena Graverini

University of Zurich[UZH]

VII SHiP Collaboration Meeting
February 11, 2016

Tutorial: change of schedule

There was a change in todays schedule, due to overlapping with the LIGO/VIRGO announcements.

New timetable:

- Today: 15:00 to 16:30 (room 13-2-005)
  - Introduction (**Thomas**)
  - Introduction to HNL simulation and analysis (**Elena**)
  - How to implement detectors and material (**Annarita**)
- Tomorrow: 8:00 to 9:00 (Filtration Plant)
  - (Real) hands-on session: creating a detector (**laroslava**)

Please attend the early-morning Friday session, it will be the most useful towards the preparation of the CDR!

Introduction to HNL simulation and analysis

Reduced available time $\Longrightarrow$ ~~hands-on~~ quick introduction

- but the original tutorial is all on the twiki!

`https://twiki.cern.ch/twiki/bin/view/Ship/ShipSoftware`

This Twiki page contains:

- installation instructions (all platforms: AFS, Mac VM, ...)
- software tutorials
- useful links
- **Please contribute!**

FairShip starter's guide

Installation

Please refer to the FairShipInstallation page for detailed installation instructions.

Setup

Make sure you have the last FairShip version:

```
cd $FAIRSHIP
git pull
```

If something was updated (i.e. you do *not* get an `Already up-to-date` message), or if your operating system or GCC

HNL tutorial: getting started

Tutorial 1 (simulating HNL events)

Please avoid using the FairShip folder tree to produce rootfiles and develop custom scripts. It would mess up and possibly create versioning troubles.

```
cd ..
mkdir -p my_analysis
cd my_analysis
```

For this tutorial we are going to generate, reconstruct and analyse an ntuple of HNLs decaying to πμ.

The python script run_simScript.py has several options. It can be used to generate HNLs, muon background, neutrino background, and each one of these use cases can be customized with further options. For now we will just generate a sample of 100 HNL → πμ events generated in charmed meson decays, with the same couplings used for the studies in the Technical Proposal and the default mass of 1 GeV:

```
python $FAIRSHIP/macro/run_simScript.py --couplings '4.47e-10,7.15e-9,1.7e-9' -n 100
```

This script produced three files in the current directory:

- a data file: `ship.10.0.Pythia8-TGeant4.root`
- a parameters file: `ship.params.10.0.Pythia8-TGeant4.root` (it comes with the FairRoot framework, and it is not used by FairShip)
- a geometry file: `geofile_full.10.0.Pythia8-TGeant4.root`

Now we run the reconstruction procedure on the data file we just generated.

```
python $FAIRSHIP/macro/ShipReco.py -f ship.10.0.Pythia8-TGeant4.root -g geofile_full.10.0.Pythia8-TGeant4.root
```

In fact, we have provided the geometry file as additional input, in order to tell the reconstruction routines which version of the SHiP geometry to use. This is of particular concern in this initial period, when the software is often updated, adding new features to the detectors, changing the shape of the magnet, and so on.

FairShip macros

- `run_simScript.py`: generating HNLs, ν and μ backgrounds, μ DIS, cosmics
  - many configurable parameters
  - for HNLs: `DecaySelection.conf`
  - for HNLs: by default it accesses a file on EOS as input to get the correct charm mesons spectrum from cascade production
- `ShipReco.py`: takes simulated events as input, runs track fit and candidate reconstruction
- `ShipAna.py`: template for the analysis of reconstructed data, you can use it as-is, take it as example, expand it...
  - we are going to work on an interactive python shell
  - you can use exactly the same commands in a python script

HNL tutorial: exploring the ntuple

The ROOT PDG database allows quick identification of particles based on their code / name:

```
pdg = r.TDatabasePDG.Instance()
for track in mcTracks:
    print track.GetPdgCode()
    print pdg.GetParticle(track.GetPdgCode()).GetName()
```

There could be isotopes or other particles created by Geant4 in the event that are unknown to TDatabasePDG.

The particle with PDG code 9900015 is the HNL. You can add it to the PDG database with:

```
import pythia8_conf
pythia8_conf.addHNLtoROOT()
```

- exploring the ntuple
  - where are simulated particles?
  - where are reconstructed tracks and HNL candidates?
- accessing specific SHiP items (HNL)
- detector hits

HNL tutorial: geometry

```
geo = r.TFile('geofile_full.10.0.Pythia8-TGeant4.root', 'read')
geo.ls()
sGeo = geo.FAIRGeom
fGeo = r.gGeoManager
```

FindNode is a useful function. It takes x,y,z coordinates and tells you to what geometry node they correspond. You can for example scan the *z* axis:

```
for z in range(-3500, -2000, 10):
    print z, fGeo.FindNode(0, 0, z).GetName()
```

or see where the simulated particles originate:

```
for track in mcTracks:
    print fGeo.FindNode(track.GetStartX(), track.GetStartY(), track.GetStartZ()).GetName()
```

- importing and understanding the SHiP geometry
- using the positions of known detectors to define fiducial volume and event/candidate selections

HNL tutorial: HNL weights, candidate selection

Let's introduce *weights*. This is the recipe to compute the HNL acceptance. We produce a lot of HNLs, and we use the weights to model the exponential distribution of their decay position.

```
total = 0
selected = 0
wtotal = 0.
wselected = 0.

for event in t:
    for candidate in event.Particles:
        total += 1
        wtotal += event.MCTrack[1].GetWeight()
        vtx = r.TLorentzVector()
        mom = r.TLorentzVector()
        candidate.ProductionVertex(vtx)
        candidate.Momentum(mom)
        if in_vessel(vtx) and IPtoTarget(vtx, mom) < 250.:
            selected += 1
            wselected += event.MCTrack[1].GetWeight()

print total, selected
print wtotal/t.GetEntries(), wselected/t.GetEntries()
```

HNL tutorial: veto systems

Let's quickly implement online selections (VETOs).

```
import shipVeto
veto = shipVeto.Task()
total = 0
selected = 0
wtotal = 0.
wselected = 0.
for event in t:
    sbt, sbtw, sbtn = veto.SBT_decision(event)
    print 'SBT: veto decision ', sbt, 'with weight', sbtw
    uvt, uvtw, uvtn = veto.UVT_decision(event)
    print 'UVT: veto decision ', uvt, 'with weight', uvtw
    svt, svtw, svtn = veto.SVT_decision(event)
    print 'SVT: veto decision ', svt, 'with weight', svtw
    # If decision is True, event is vetoed!
    if sbt or uvt or svt: continue
    for candidate in event.Particles:
        if not match2MC(event, candidate): continue
        total += 1
        wtotal += event.MCTrack[1].GetWeight()
```

~~Don't~~ try this at home

- The tutorial will walk you through the basics of simulating HNL signal in FairShip.
- Advanced topics (not covered here, soon on the twiki!):
  - selecting the final states to simulate
  - changing the HNL parameters (mass, couplings)
- Please try it out by yourselves. It takes 10-40 minutes, depending on your expertise. During tomorrow's session we can rediscuss this and solve any problem / answer any question (hopefully).
- Please contribute. Once you manage to do something or solve a common problem, please share your knowledge by creating a simple tutorial for others to come!