

Delphes card & FCC Software Framework



Zbyněk Drásal

CERN

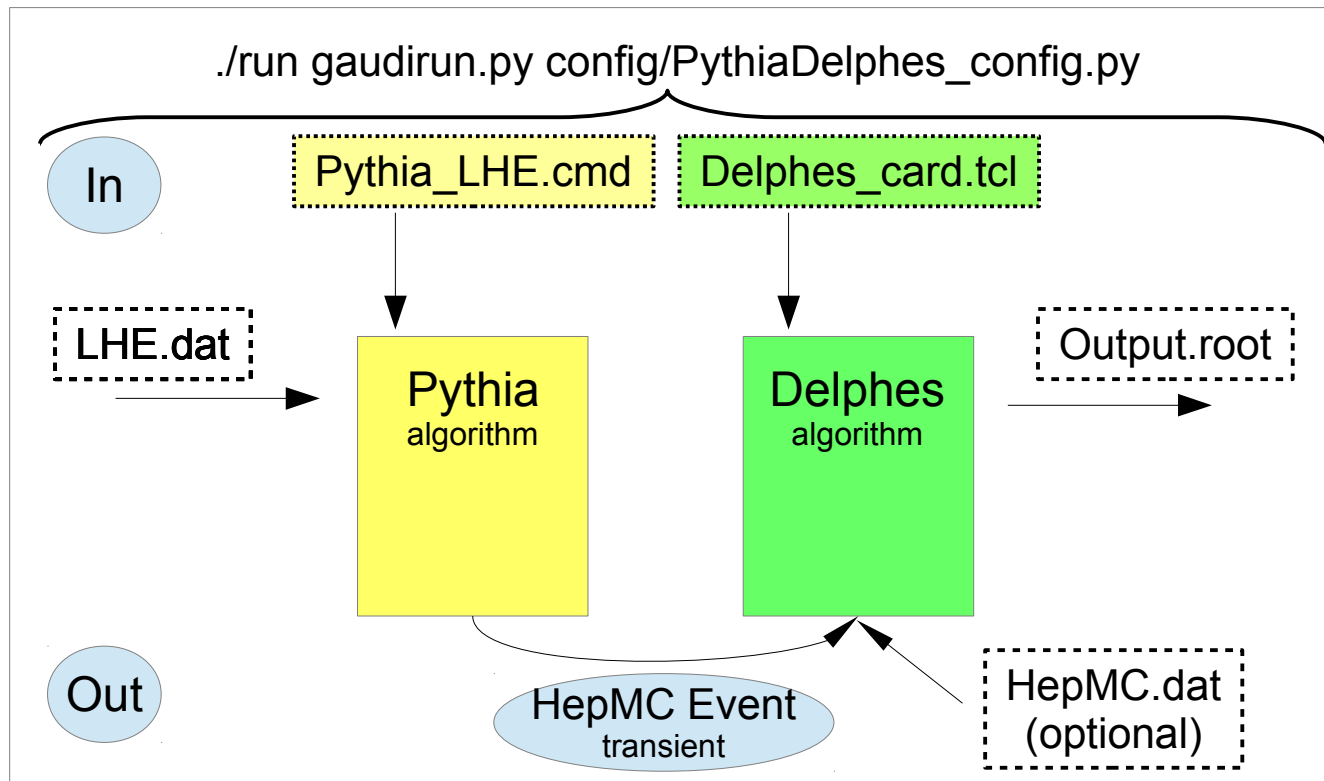
With: M. Mannelli, C. Helsen, W. Riegler, H. Gray & F. Moortgat

Overview

- How to use Delphes and simulate physics events within the FCC Software Framework?
 - Strategy & processing sequence
 - Input/Output data files
 - Configuration inputs
 - Tutorial
 - Example: $pp \rightarrow HH$
 - Issues?
- Summary & Outlook

Strategy & Processing Sequence

- FCCSW → based on modular structure of Gaudi framework
- Strategy:
 - modularize Pythia & Delphes as Gaudi algorithms
 - use configuration files to control processing sequence



Pythia Input/Output Data Files

- **Pythia Input:**

- The Pythia data input is required either through LHE (Les Houches Event) data file (from e.g. Madgraph, ...)
 - Already simulated Les Houches Events are processed by Pythia
 - Pythia performs MPI, ISR, FSR & hadronization
- Or no input data file provided to Pythia
 - Pythia will simulate physics events & perform MPI, ISR, FSR, ...

- **Pythia Output:**

- The Pythia data output is through transient memory data store using HepMC event data format.

- **Comments:**

- Current version of Pythia installed on **lxplus** supports LHE of version 1.0, not higher!
- The newest Pythia version 8.205 does support all LHE versions!

Delphes Input/Output Data Files

- Delphes Input:

- The Delphes data input is preferably from transient event data store
- In principal, the input event can be read-in from HepMC data file
 - both inputs read-in through: `DelphesExtHepMCReader` class

- Delphes Output:

- The Delphes data output is written out through the standard Delphes ROOT tree writer: `ExRootTreeWriter` class, so ROOT file is exported

FCCSW Configuration

- FCCSW configuration:
 - Run sequence of Gaudi framework controlled through a Python script
 - Use the following file: `${FCCSW}/config/PythiaDelphes_config.py`
 - Variables to be arranged:
 - `NEvents` → events to be simulated
 - `PythiaconfFile` → Pythia configuration, called command, file (use either `Pythia_LHEinput.cmd` or `Pythia_standard.cmd`)
 - `DelphesCard` → Delphes TCL configuration file (use official Delphes card)
 - `DelphesHepMCInFile` → Delphes input file (use "" to read HepMC directly from Pythia, i.e. from transient data store)
 - `DelphesRootOutFile` → Delphes output file (define name of Delphes output file)
If file not defined the data goes to transient data store → will be useful once the FCC event-data output is defined

Pythia Configuration

- Pythia configuration:
 - Control Pythia physics using command file → use the following files
 - `${FCCSW}/config/Pythia_LHEinput.cmd` → read-in LHE file
 - `${FCCSW}/config/Pythia_standard.cmd`
 - Variables to be arranged for LHE input:
 - `PartonLevel:MPI = on/off` → Switch on/off Multi-Particle-Interaction
 - `PartonLevel:ISR = on/off` → Switch on/off Initial-State-Radiation
 - `PartonLevel:FSR = on/off` → Switch on/off Final-State-Radiation
 - `HadronLevel:all = on/off` → Switch on/off Hadronization
 - `Beams:LHEF = data/name.lhe` → Define LHE event data file

Delphes Configuration

- Delphes configuration:

- `${FCCSW}/config/FCChh_DelphesCard_*vXX.tcl`

- Official FCC-hh Delphes configuration card
- Defines detector characterization and sequence of Delphes modules to be executed.
- All Delphes objects to be analyzed are written-out by Delphes internal module `TreeWriter`, via branches:

Array of particles in each event as `Particle` branch, each particle of `GenParticle` type:

```
Branch Delphes/allParticles Particle GenParticle
```

Array of electrons in each event as `Electron` branch, each particle of `Electron` type

```
Branch ElectronEnergySmearing/electrons Electron Electron
```

...

- To analyze data, read the branches through `ExRootTreeWriter` class → follow add hoc tutorial for more details

HowTo? Tutorial

- For additional information follow the HowTo? tutorial at FCC Twiki page:
 - <https://twiki.cern.ch/twiki/bin/view/FCC/FccPythiaDelphes>

Twiki >  FCC Web > [CommonTools](#) > [FccSoftware](#) > [FccPythiaDelphes \(2015-09-15, ZbynekDrasal\)](#)

FCC Pythia + Delphes Analysis (Tutorial)

Contents

- ↓ [FCC Pythia + Delphes Analysis \(Tutorial\)](#)
 - ↓ [Overview](#)
 - ↓ [Installation Procedure](#)
 - ↓ [How to Run?](#)
 - ↓ [How to Analyze Data?](#)
 - ↓ [Example: Read data from Delphes output ROOT file](#)

Overview

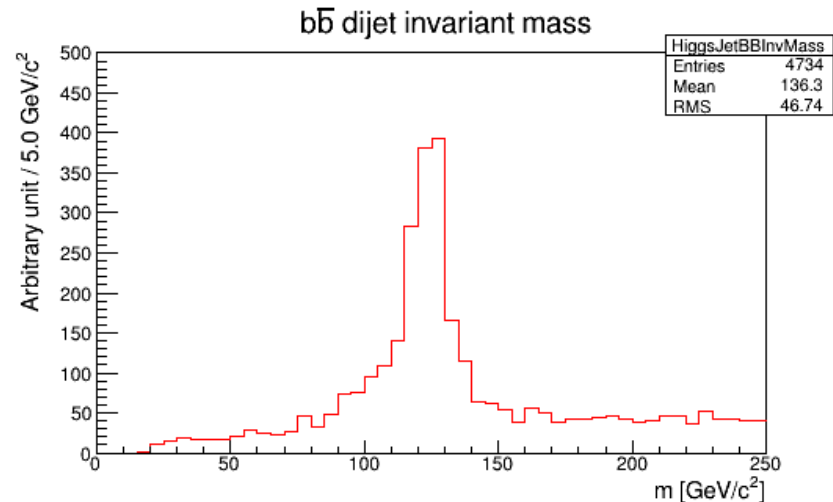
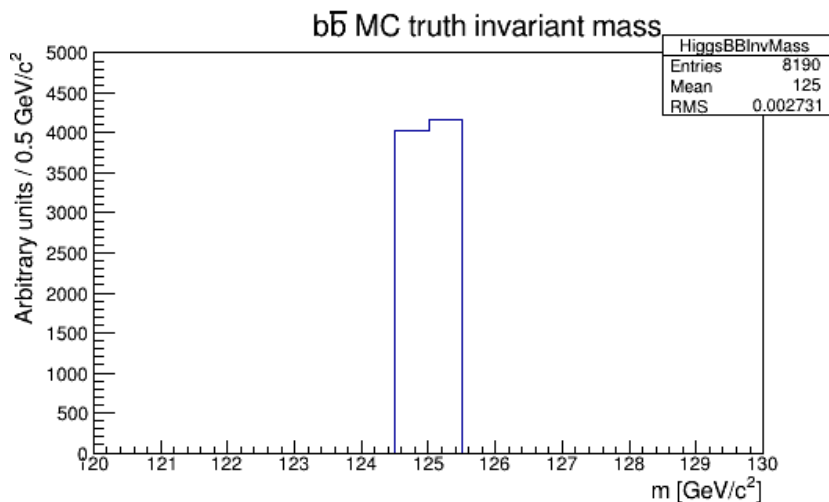
A small tutorial on how to study FCC-hh benchmark channels within the [FCCSW framework](#) with [Pythia](#) generator and [Delphes](#) simulation.

Input/Output:

- The Pythia input is required through [LHE](#) (Les Houches Event) data file together with a special Pythia config file or just standard Pythia config file.
- The Pythia output is through transient memory data store using [HepMC](#) event data format.
- The Delphes input is preferably from event data store, but can be in principal read-in from [HepMC](#) data file.
- The Delphes output is through Delphes ROOT tree writer, so via [ROOT](#) file.
- Support for FCC event-data format is under development --> will be available soon!

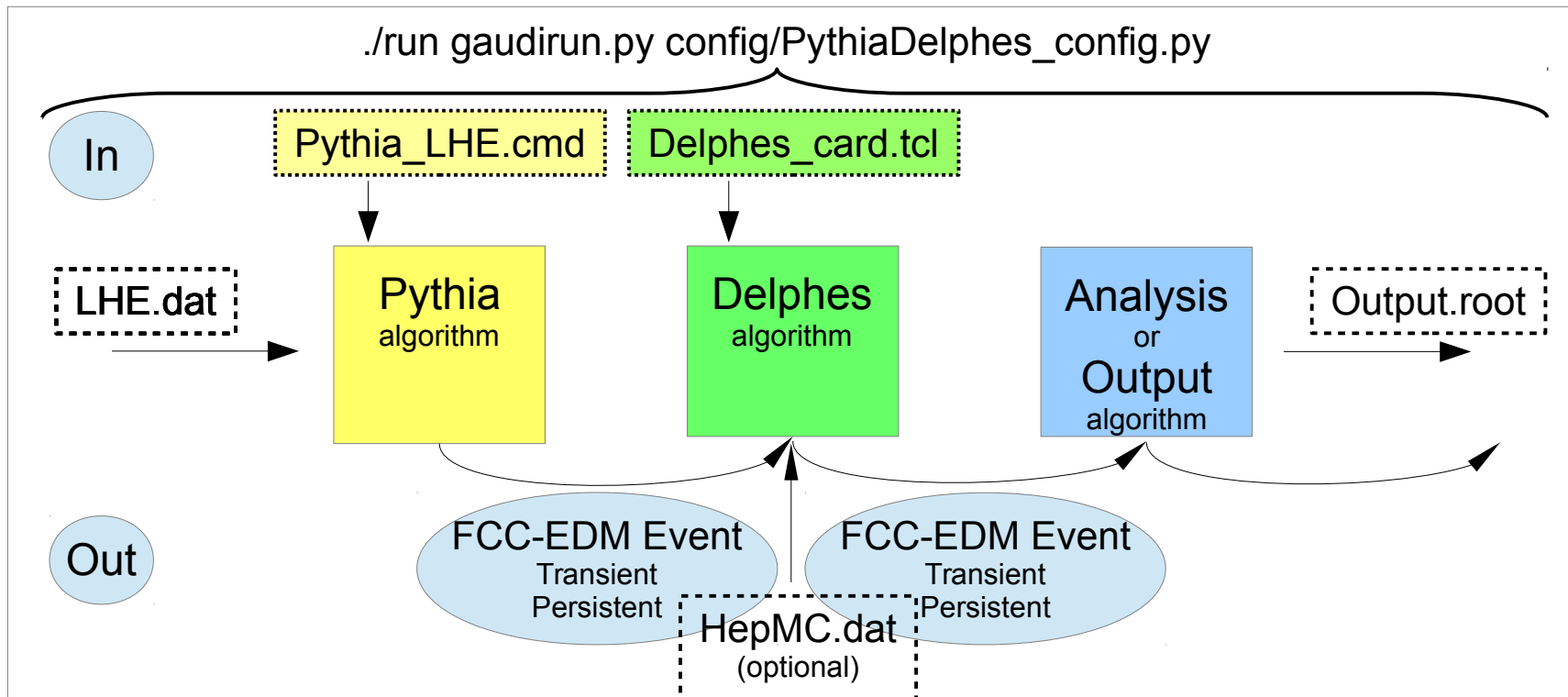
Example: $pp \rightarrow HH$

- Physics process: $pp \rightarrow HH$, analyze events with $H \rightarrow b\bar{b}$, draw inv. mass
 - Download LHE file (double higgs production) from:
<https://test-fcc.web.cern.ch/test-FCC/LHEevents.php>
 - Use `PythiaDelphes_config.py` with `Pythia_LHEconfig.py` set as `PythiaconfFile`.
 - Set `nEvents = 50000`, switch `MPI`, `ISR`, `FSR` on
 - Set the LHE file as an input in `Pythia_LHEconfig.py`
 - Use official Delphes card
 - Keep `DelphesHepMCInFile = ""`, set `DelphesRootOutFile = "DelphesOutput.root"`
 - Use Particle branch to find b, \bar{b} coming from Higgs and Jet branch with `BTag=1` to find b-jets



Issues?

- Current approach provides full generator & simulator chain, but ...
 - Missing output to the official FCC event-data structure, i.e. EDM, (transient/persistent) → needs to be implemented, e.g. for official simulation campaign etc.
 - An interplay between end-users and software developers needed, to define possibly missing parts in the official FCC EDM!



Summary

- **Summary:**

- Pythia & Delphes implemented as Gaudi algorithms into FCCSW framework (continue & extension of work by M. de Gruttola) → HepMC events kept in memory now (transient data store) → Pythia can be run in cooperation with Delphes within FCCSW now!
- Pythia&Delphes Gaudi config script prepared: `PythiaDelphes_config.py`
- Pythia starting configuration files arranged
- Official Delphes card included to the FCCSW (for details see W.Riegler's presentation @ <https://indico.cern.ch/event/443015>)
- Prepared tutorial on FCCSW Twiki for newcomers:
<https://twiki.cern.ch/twiki/bin/view/FCC/FccPythiaDelphes>

Outlook

- **Issues & Outlook:**

- Current approach doesn't allow to save in an efficient way e.g. Pythia output (HepMC files too large)
 - **Need for implementation of transient input/output using FCC EDM!**
- Not an issue for physics analysis → Delphes ROOT output given → use ROOT to analyse data, but...
- If one wants to organize e.g. an official campaign for simulation of benchmark channels, it makes sense to simulate Pythia outputs only → end-users might want to change Delphes configuration in order to understand an interplay between physics & detector performance ...