LATEST DEVELOPMENTS IN FCCANALYSES

Juraj Smieško for the FCCSW team

CERN

FCC Week 2023

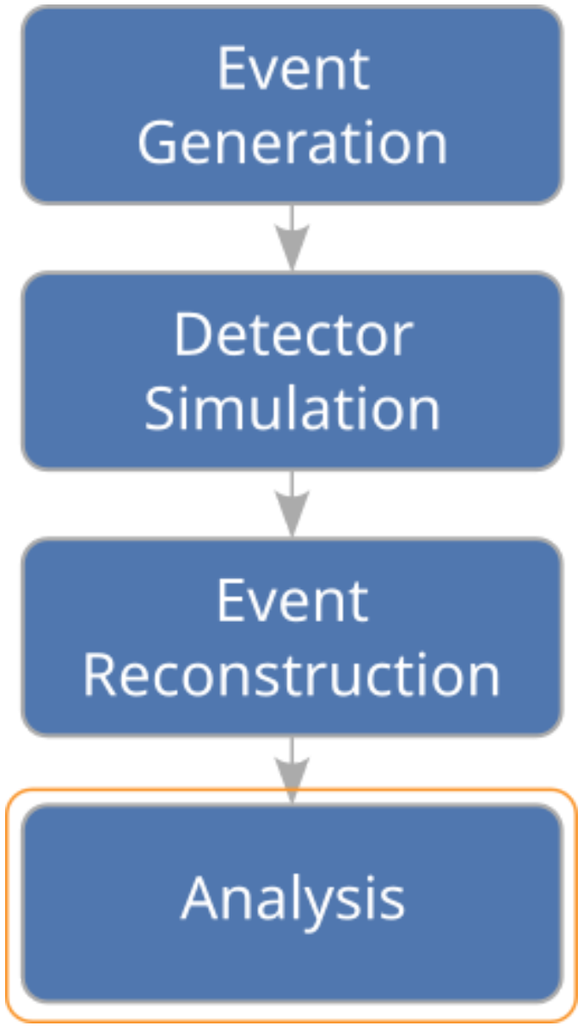
London, 06 Jun 2023

FCCANALYSES SCOPE

Goal of the framework is to aid the users in obtaining the desired physics results from the reconstructed objects

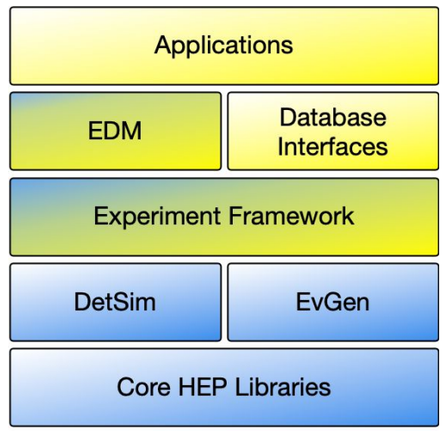
Framework requirements:

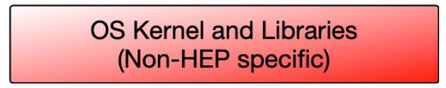
- Efficiency — Make quick turn-around possible
- Flexibility — Allow heavy customization
- Ease of use — Should not be hard to start using
- Scalable — Seamlessly handle from small to large datasets

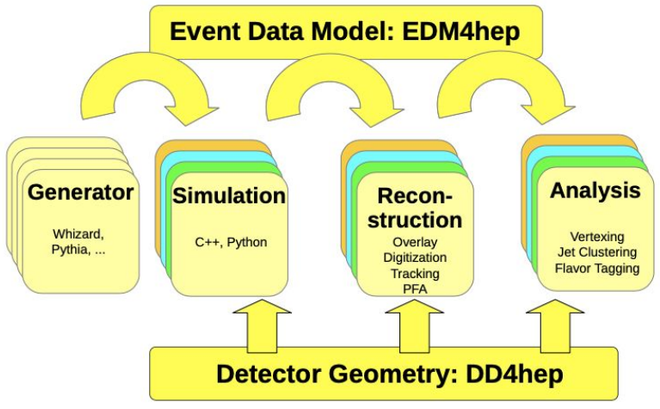


KEY4HEP

- Set of common software packages, tools, and standards for different Detector concepts
- Common for FCC, CLIC/ILC, CEPC, EIC, …
- Individual participants can mix and match their stack
- Main ingredients:
  - Data processing framework: Gaudi
  - Event data model: EDM4hep
  - Detector description: DD4hep
  - Software distribution: Spack



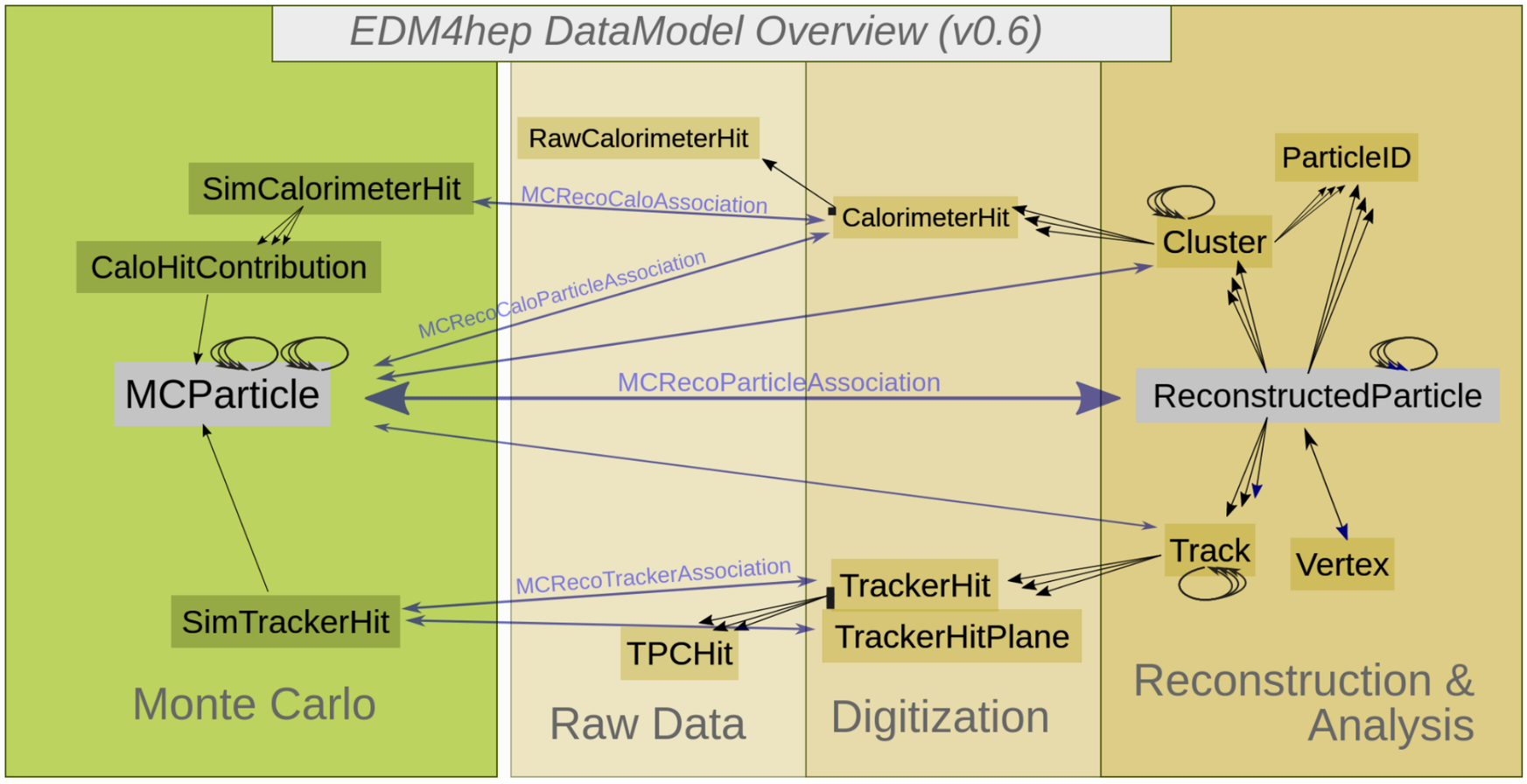




EDM4HEP I.

Describes event data with the set of standard objects.

- Specification in a single YAML file
- Generated with the help of Podio



EDM4HEP II.

Example object:

```
#-------------  CalorimeterHit
edm4hep::CalorimeterHit:
  Description: "Calorimeter hit"
  Author : "F.Gaede, DESY"
  Members:
    - uint64_t cellID             //detector specific (geometrical) cell id.
    - float energy                //energy of the hit in [GeV].
    - float energyError           //error of the hit energy in [GeV].
    - float time                  //time of the hit in [ns].
    - edm4hep::Vector3f position //position of the hit in world coordinates in [mm].
    - int32_t type                //type of hit. Mapping of integer types to names via coll
```

- Current version: `v0.8.0`
- Objects can be extended / new created
- Bi-weekly discussion: Indico

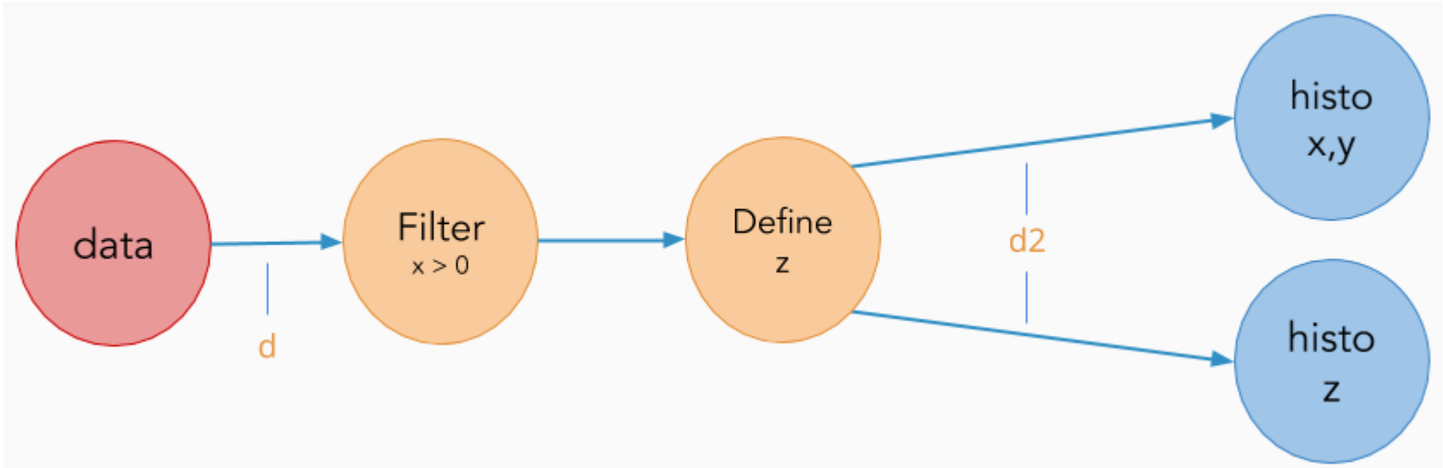
DATASETS

Plethora of processes are pre-generated and available from EOS

- Two main production campaigns in use:
  - Spring 2021
  - Winter 2023
- Processes are identified by its name, e.g.: `p8_ee_WW_ecm240`
- The production Database browsable at: fcc-physics-events.web.cern.ch
- Example: Delphes events, IDEA, FCCee, winter 2023
- EOS directory:

  `/eos/experiment/fcc/...`
- Generation handled by EventProducer
  - Heads up: Will change soon (Dirac, iLCDirac)

ROOT RDATAFRAME



- Describes processing of data as actions on table columns
  - Defines of new columns
  - Filter rules
  - Result definitions (histogram, graph)
- The actions are lazily evaluated
- Multi threading is available out of the box
- Optimized for bulk processing

INTEGRATION WITH EXISTING TOOLS

- Boundary between reconstruction and analysis blurred
  - Especially for full-sim
  - Plan: Develop algorithm on analysis side, then move to reconstruction
- Many tools/libraries created over the years
  - Most are integrated into the Key4hep stack
- RDataFrame C++ based, integrated into Python

```
// d2 is a new data-frame, a transformed version of d
auto d2 = d.Filter("x > 0")
           .Define("z","x*x + y*y");
// make histograms out of it
auto hz = d2.Histo1D("z");
auto hxy = d2.Histo2D({"hxy","hxy",16,-1,1,64,-1,1},"x","y");
```

AVAILABLE LIBRARIES

The physics analysis often depends on multitude of libraries

Libraries integrated into the framework:

- ROOT — together with RDataFrame
- ACTS — track reconstruction tools
- ONNX — neural network exchange format
- FastJet — jet finding package
- DD4hep — detector description
- Delphes — fast simulations

DISTRIBUTION

FCCAnalyses latest release `v0.7.0` can be found:

- As a package in the stable Key4hep stack
  - Allows to quickly put together small analysis
  - Limited options for customization
- As a tarball/tag from GitHub

Latest/development version of the FCCAnalyses can be found:

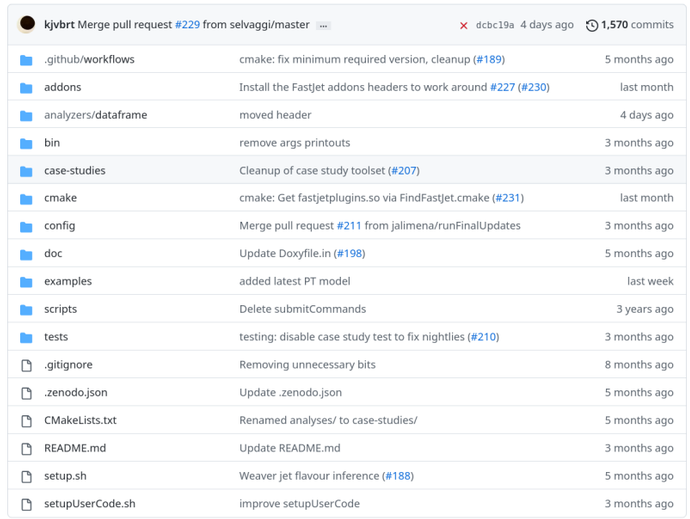
- As a package in the nightlies Key4hep stack
  - Might easily break
  - Latest `master`
- By checking out `master` branch
  - Allows greater customization
  - Requires discipline
  - Hint: Keep your master in sync with upstream (use rebase or merge)
  - Developments are welcome to be merged :)
  - `master` should be always buildable

Platforms: CentOS 7, AlmaLinux 9, Ubuntu 22.04

ECOSYSTEM

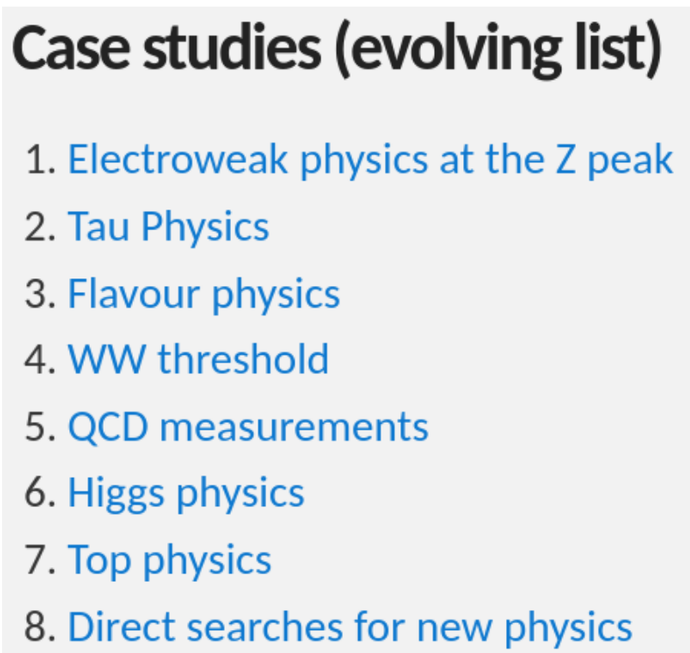
Analysis spread through two repositories:

- FCCAnalyses
  - Repository of common tools and algorithms
  - General analysis code in analyzers
  - Steering of the analysis (RDataFrame)
  - Access to the dataset (meta)data
  - Running over large datasets / on batch
  - Experimetal machinery for case studies
- FCCeePhysicsPerformance
  - Main place for the abstracts
  - Contains very specific analysis code
    - Or prototypes of tools of common interest to be eventually moved to FCCAnalysis
  - (Proto)package repository



kjvbrt Merge pull request #229 from selvaggi/master ... dcbc19a 4 days ago **1,570** commits

.github/workflows	cmake: fix minimum required version, cleanup (#189)	5 months ago
addons	Install the FastJet addons headers to work around #227 (#230)	last month
analyzers/dataframe	moved header	4 days ago
bin	remove args printouts	3 months ago
case-studies	Cleanup of case study toolset (#207)	3 months ago
cmake	cmake: Get fastjetplugins.so via FindFastJet.cmake (#231)	last month
config	Merge pull request #211 from jalimena/runFinalUpdates	3 months ago
doc	Update Doxyfile.in (#198)	5 months ago
examples	added latest PT model	last week
scripts	Delete submitCommands	3 years ago
tests	testing: disable case study test to fix nightlies (#210)	3 months ago
.gitignore	Removing unnecessary bits	8 months ago
.zenodo.json	Update .zenodo.json	5 months ago
CMakeLists.txt	Renamed analyses/ to case-studies/	5 months ago
README.md	Update README.md	3 months ago
setup.sh	Weaver jet flavour inference (#188)	5 months ago
setupUserCode.sh	improve setupUserCode	3 months ago



Case studies (evolving list)

1. Electroweak physics at the Z peak
2. Tau Physics
3. Flavour physics
4. WW threshold
5. QCD measurements
6. Higgs physics
7. Top physics
8. Direct searches for new physics

ANALYSIS ARCHITECTURE I.

One can write and run an analysis in several ways:

- Managed mode: `fccanalysis run my_ana.py`
  - The RDataFrame frame is managed by the framework
  - User provides Python analysis script with compulsory attributes
  - Libraries are loaded automatically
  - Dataset metadata are loaded from remote location — CVMFS/HTTP server
  - Batch submission on HTCondor
  - Customization: Possible at the level of analyzer functions
  - Intend for: Quick analysis, no advanced analyzer functions

ANALYSIS ARCHITECTURE II.

One can write and run an analysis in several ways:

- Standalone mode: `python my_ana.py`
  - The RDataFrame frame is managed by the user
  - Can leverage the FCCAnalyses library of analyzer functions
  - The analysis can be written as a Python script or C++ program
  - Loading of the libraries is handled by the user
  - Dataset metadata have to be handled manually
  - Batch submission is not provided
  - Customization: Creation and steering of the RDataFrame
  - Intended for: Advanced users
- Ntupleizer style:
  - Intend is to create flat trees and continue without the frameworks help

WRITING AN ANALYZER FUNCTION

- Analyzer function is a C++ function or struct
- Typically and analyzer is a `struct` which operates on an EDM4hep object
- Optional dependencies for analyzers can be: FastJet, DD4hep, ACTS and ONNX
- ROOT RDataFrame needs to be aware of the analyzer function
  - Provided as a string
  - Compiled in the library
  - Loaded and JITed by the `ROOT.gInterpreter`

```
128    /// Get the invariant mass in a given hemisphere (defined by it's angle wrt to axis).
129    struct getAxisMass {
130    public:
131      getAxisMass(bool arg_pos=0);
132      float operator() (const ROOT::VecOps::RVec<float> & angle,
133                        const ROOT::VecOps::RVec<float> & energy,
134                        const ROOT::VecOps::RVec<float> & px,
135                        const ROOT::VecOps::RVec<float> & py,
136                        const ROOT::VecOps::RVec<float> & pz);
137    private:
138      bool _pos; /// Which hemisphere to select, false/0=cosTheta<0 true/1=cosTheta>0. Default=0
139    };
```

FCCANALYSES LIBRARY

- Vertexing
- ACTS vertex finder
- Event variables
- Calorimeter hit/cluster variables
- Reconstructed/MC particle operations
- Flavour tagging
- Jet clustering/constituents

Case studies (evolving list)

1. Electroweak physics at the Z peak
2. Tau Physics
3. Flavour physics
4. WW threshold
5. QCD measurements
6. Higgs physics
7. Top physics
8. Direct searches for new physics

WORKFLOW

- The complete analysis in managed mode is divided into three steps (example):
  - `analysis_stage1.py`, ... — pre-selection stages, analysis dependent, usually runs on batch
  - `analysis_final.py` — final selection, produces final variables
  - `analysis_plots.py` — produces plots from histograms/TTrees
- or into two with the help of Histmaker (example):
  - The pre-selection stages and final stage are combined together
  - Plotting step
- Disclaimer: Plotting facilities are rudimentary, improvements are welcome :)

EOS SPACE

Various intermediate files of common interest can be stored at:

`/eos/experiment/fcc/ee/analyses_storage/...`

in four subfolders:

- BSM
- EW_and_QCD
- flavor
- Higgs_and_TOP

Access and quotas:

- Read access is is granted to anyone
- Write access needs to be granted: Ask your convener :)
- Total quota for all four directories is `200TB`
- ATM only part of the quota is allocated

RECENT CHANGES I.

Included in v0.7.0:

- External libraries as addons: PR#194
- EOS paths accessed through xrootd: PR#202
- Case studies (proto)packaging: PR#199
- Inclusion of ONNX + Jet flavour tools: PR#188, PR#224
- Inclusion of Delphes + Vertexing from Franco Bedeschi: PR#247
- New sub-commands — `build`, `pin`
- 2D and 3D histograms: PR#253
- Benchmarking and testing of the example analyses

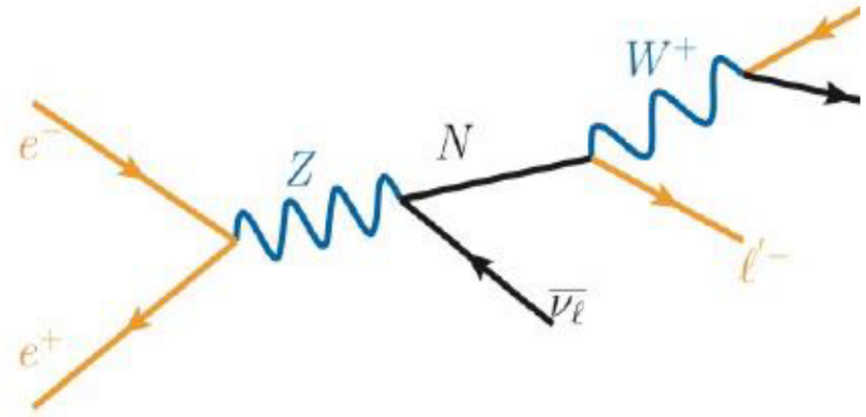
RECENT CHANGES II.

Available in the `master`:

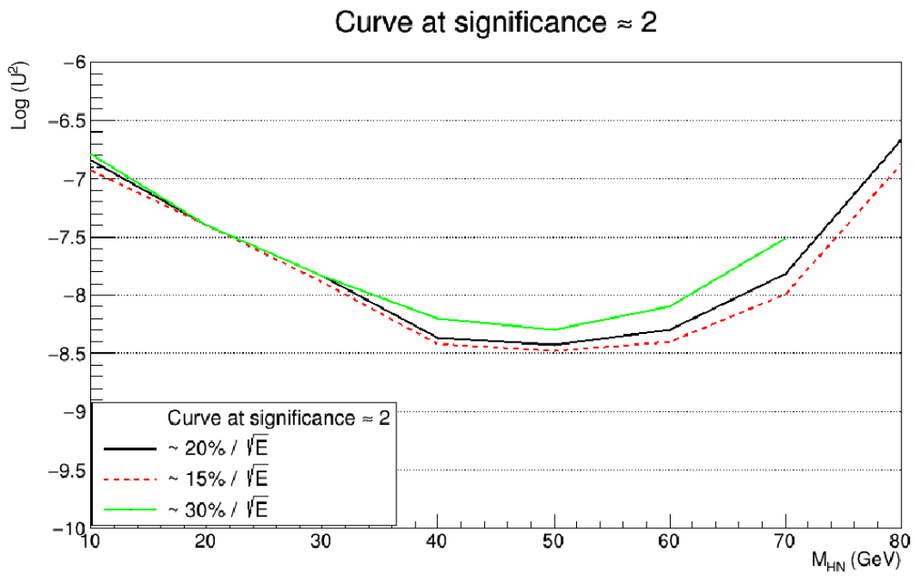
- Improvements in dNdx, time and energy smearing: PR#268
- Statistical uncertainty and rebin options in the plotter: PR#269
- Histmaker: PR#277
- Improved crash reports: PR#276
- More track utilities: PR#289
- Code formatting for the analyzers
- Modularization of the python machinery

PHYSICS RESULTS I.

Decay of an HNL into a muon and two jets



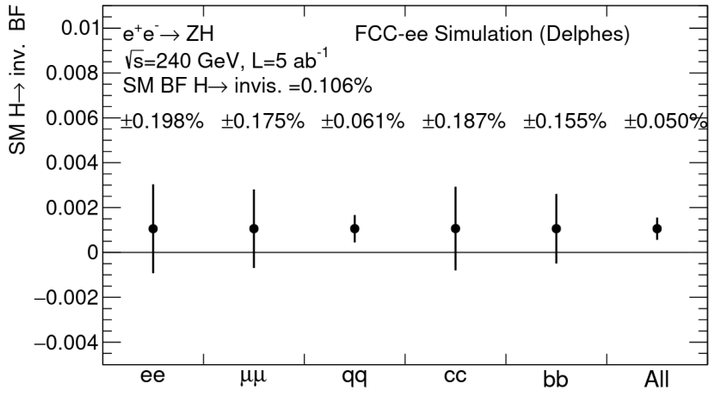
- BSM/LLP Analysis
- Private fork with the customizations applied on top
- Run in managed mode
- New analyzers, adjustments to the managed mode
- Uses mix of official and private productions

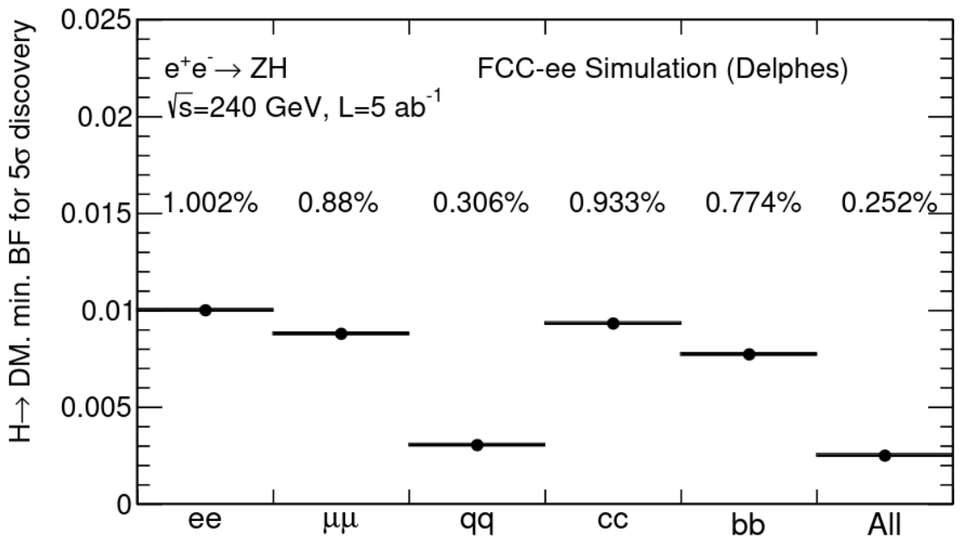


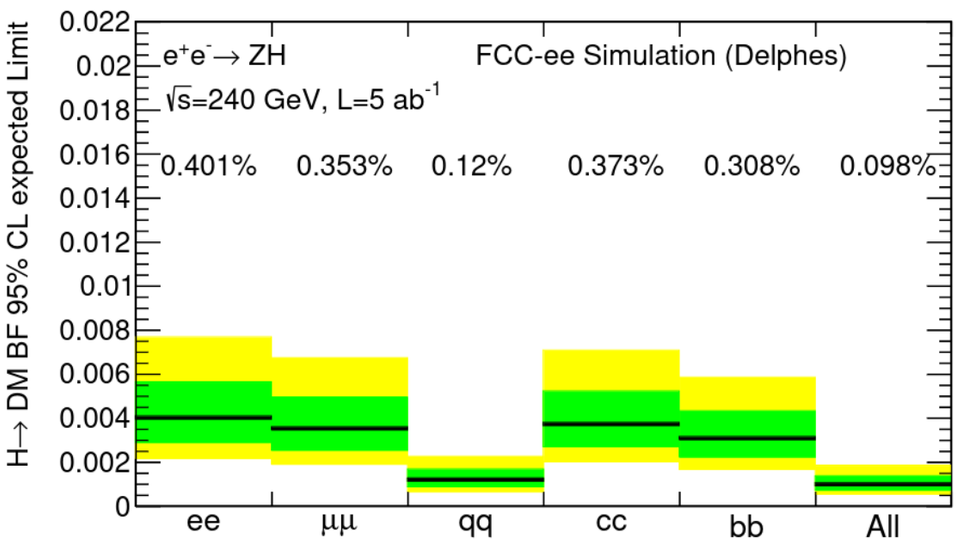
PHYSICS RESULTS II.

H to invisible

- Higgs Analysis
- Could not find the source code
- Uses officially produced samples



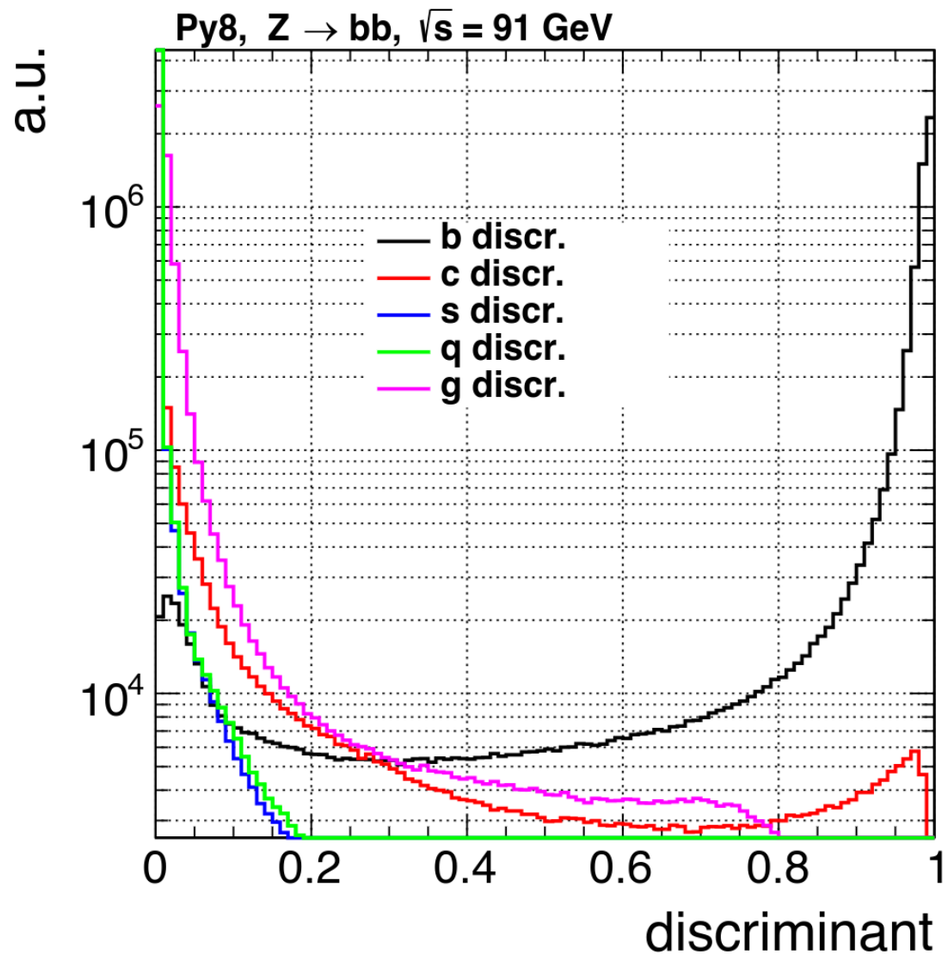




PHYSICS RESULTS III.

Tagger on Z to qq events

- Example of advanced usage of the framework
- Uses combination of managed mode and custom python scripts
- Leverages recently included libraries: Delphes, ONNX
- Uses officially produced samples



PLANS I.

Reliable framework to aid the physics performance studies for FSR

Heads up:

- Podio frame I/O in Gaudi, PR#100
  - One ROOT file can hold multiple event sets "frames"
- Podio collection IDs to hashes, PR#412
  - Less friendly referencing of the collections
  - Possible remedy: EDM4hep RDataSource
- RNTuple Podio backend, PR#359

PLANS II.

Reliable framework to aid the physics performance studies for FSR

- Make the framework free from lxplus/HTcondor
  - Localized access to the datasamples
- Overhaul "standard library" and disentangle the dependencies
- Prepare facilities to handle systematics
- Find distribution channels which allow as wide customization as possible
- Support fullsim detector studies
  - Initial place for the case-studies algorithms
  - Access to the detector description supported by the framework
- Support running on the distributed systems (Dirac)

DOCUMENTATION

There are several sources of documentation

- FCC Tutorials: https://hep-fcc.github.io/fcc-tutorials/
  - Focused on providing a tutorial on a specific topic
- Code reference: https://hep-fcc.github.io/FCCAnalyses/doc/latest/index.html
  - Provides details about implementation of individual analyzers
- Manual pages:
  - Info about commands directly in the terminal: `man fccanalysis`
- FCCAnalyses website, FCCSW website

CONCLUSIONS & OUTLOOK

- The combination of EDM4hep and RDataFrame works well
  - Possibility to integrate range of libraries
- Factorization at library and at analysis level under way
- Started focusing on the full simulation detector studies
  - Access to the detector description through the framework

More heads are welcome!

Babyface from Toy Story, Pixar

BACKUP

FCCANALYSES VS. COFFEA/COFFEA-CASA

- Provides similar set of features to FCCAnalyses
- Dataframe in coffea, Orchestration in coffea-casa
- User interface purely pythonic
- Integrated into python package ecosystem
- FCCAnalysis purpose build for FCC
- Integration with SWAN and Dask

FCCANALYSES BATCH SUBMISSIONS

- FCCAnalyses allows users to submit their jobs onto HTCondor
- It bootstraps itself with use of scripts in subprocesses
- Framework creates two files
  - Shell script with `fccanalysis` command
  - Condor configuration file
- There is also possibility to add user provided Condor parameters
- Condor environment now isolated from machine where the submission was done
- Revised tracking across chunks/stages done with the variable in the ROOT file

SUB-COMMAND ROUTING

- There are three ways to run the analysis
  - `fccanalysis run my_analysis.py`
  - `python config/FCCAnalysesRun.py my_analysis.py`
    - Can this way be dropped?
  - `python my_analysis.py`
- Removed reliance on try/catch for sub-command routing

CODE FORMATTING

- Currently, there is wide range of styles used
- End goal: Make the analyzers better organized
  - They are building blocks of the analysis
- Created CI to check every commit
- LLVM Style selected based on popularity
- Only changed lines are checked

UPDATED VERTEXING

- Vertexing done with the help of code from Franco B.
- Introduces dependency on Delphes
- Introduces new analyzers: `SmearedTracksdNdx`, `SmearedTracksTOF`
- Simplifies Delphes–EDM4hep unit gymnastic
- Adds examples for B_s to D_s K

BUILDING OF FCCANALYSES

- FCCAnalyses is a package in the Key4hep stack
- Advanced users can work directly on their forks
  - Allows to keep the analysis ''cutting edge''
  - Requires discipline
- Added helper sub-command: `fccanalysis build`
- Current distribution mechanisms:
  - Using released version in Key4hep stack
  - Separate git repository + stable Key4hep stack
  - Separate git repository + nightlies stack

KEY4HEP STACK PIN

- FCCAnalyses is developed on top of Key4hep stack
- Sometimes depends on specific version of the package
- Added helper sub-command: `fccanalysis pin`
- Will pin the analysis to a specific version of the Key4hep stack
  - There is no patch mechanism in the Key4hep stack