Software tutorial for FCC-hh studies

17.10.2024 FCC-hh Physics & Performance meeting

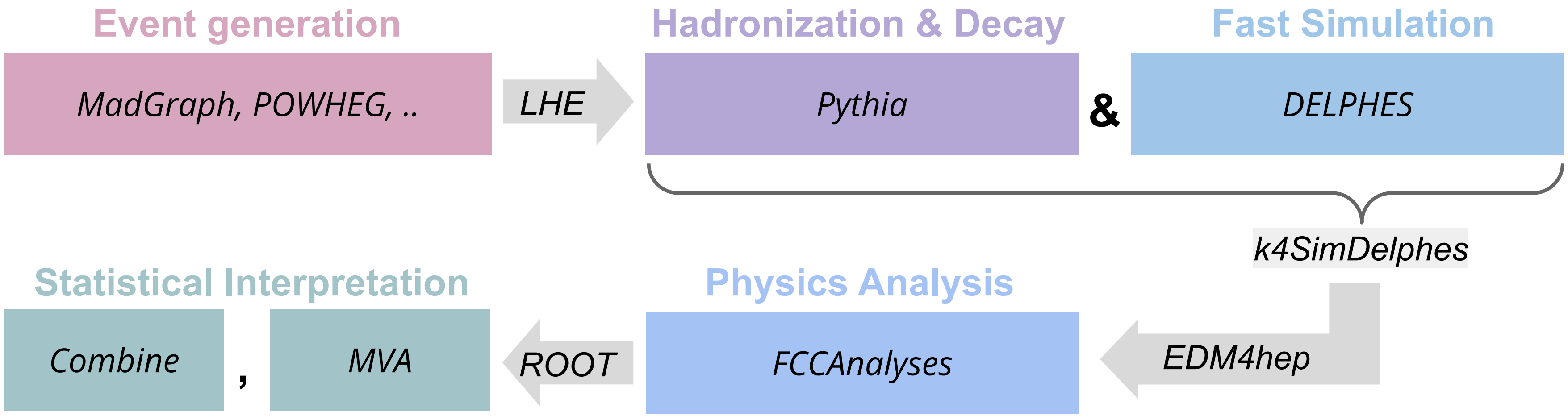
Birgit Stapf

Introduction

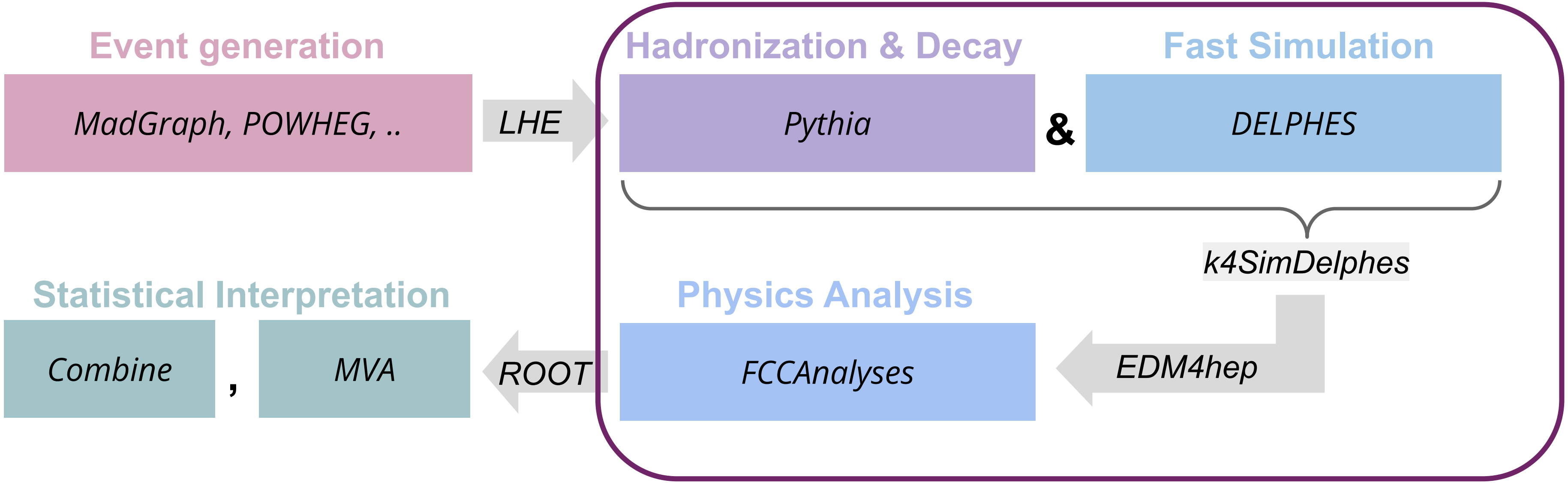
- Do you have an idea for a FCC-hh study, but find yourself wondering how you would go about putting it into practice?
  - After my overview presentation in the last general meeting, we will put what we learned into practice and run a small analysis example together

- *Note: All current and planned physics studies for FCC-hh rely on fast simulation with Delphes, there is also ongoing work and lots of opportunity for stand-alone full simulation studies, focussing e.g. on pile-up, tracking with timing information, flavour tagging → not part of this tutorial!*

Overview of technical workflow



Overview of technical workflow

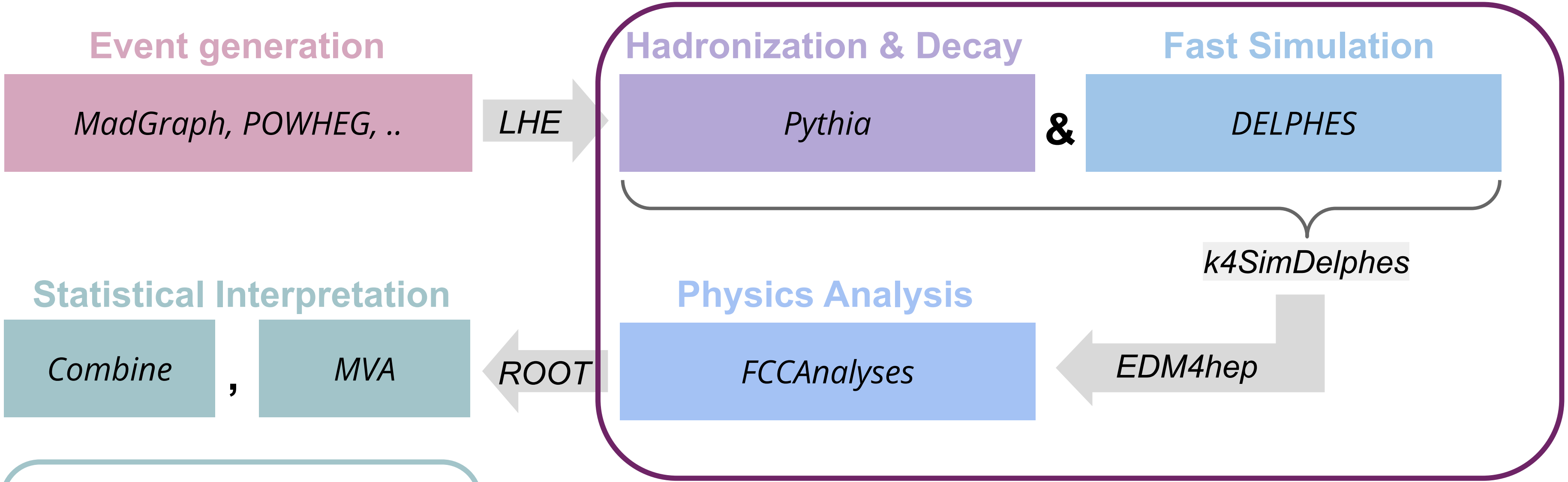


Rely on the key4hep project

Same approach as for FCC-ee studies

Content of the tutorial today

Overview of technical workflow



Please let us know if you would be interested in a combine tutorial for the next meeting!

Rely on the key4hep project

Same approach as for FCC-ee studies

Content of the tutorial today

Caveats & remarks

- This tutorial assumes no previous experience with FCC software or fast simulation with Delphes, tries to give a basic insight into the concepts
  - But it is not a simulation tutorial! Refer to Delphes documentation
- We will start from existing LHE for the process of interest
  - A tutorial on event generation is available here, but it is FCC-ee specific
  - If you have or need additional LHE, please get in touch!
- We will run a small file through Pythia + Delphes locally as an example
  - Normally this is done in a large scale production with EventProducer
  - We will start a new fast sim production campaign soon (v0.6), please get in touch about which samples you would need

Event generation

Step 1: Finding available LHE events in the database

https://fcc-physics-events.web.cern.ch/

FCC | FCC-hh | Gen | Les Houches | About

FCC-hh | Gen | Les Houches Samples

Additional stats about the production can be found here.

Name: lambda100

Expand table

No	Name	Nevents	Nfiles	Nbad	Neos	Size [GB]	Output Path	Main Process	Final States	Matching Param	Cross Section [pb]
128	mg_pp_hh_lambda100_5f	15,300,000	1530	0	1530	2.59	/eos/experiment/fcc/hh/generation/lhe//mg_pp_hh_lambda100_5f/	HH, H->bb, H undec., kl = 1.00			1.42752
142	mg_pp_hhj_lambda100_5f	8,750,000	875	0	875	1.25	/eos/experiment/fcc/hh/generation/lhe//mg_pp_hhj_lambda100_5f/	HH + 1 jet, pT(HH) > 200, kl = 1.00	inclusive		0.05644
315	mg_pp_tthh_lambda100_5f	1,000,000	100	0	100	0.17	/eos/experiment/fcc/hh/generation/lhe//mg_pp_tthh_lambda100_5f/	ttHH	inclusive		0.0595724055
375	mg_pp_vbfhh_lambda100_5f	1,000,000	100	0	100	0.17	/eos/experiment/fcc/hh/generation/lhe//mg_pp_vbfhh_lambda100_5f/	VBF HH (qq->jjHH)	inclusive		0.072176497
461	mg_pp_vhh_lambda100_5f	1,000,000	100	0	100	0.14	/eos/experiment/fcc/hh/generation/lhe//mg_pp_vhh_lambda100_5f/	VHH	inclusive		0.01159155
576	pw_pp_hh_lambda100_5f	10,329,977	1033	0	1033	1.64	/eos/experiment/fcc/hh/generation/lhe//pw_pp_hh_lambda100_5f/	gg->HH (NLO)	inclusive		1.13822

- For many processes, LHE events are available in the database
  - *Navigate to the FCC-hh LHE database*
- We will use a SM di-Higgs production sample named pw_pp_hh_lambda100_5f
  - *Search for the sample - What production mode is it? What is the cross-section?*

Event generation

Step 1: Finding available LHE events in the database

https://fcc-physics-events.web.cern.ch/

FCC FCC-ee FCC-hh

FCC Physics Events

Database of pre-generated samples for FCC-hh and FCC-ee physics performance studies.

Accelerators

The FCC integrated program includes two accelerator proposals:

- FCC-ee
- FCC-hh

FCC FCC-hh Gen Delphes Full Sim

FCC-hh Samples

- Gen | Les Houches
- Delphes | v0.2
- Delphes | v0.3
- Delphes | v0.4
- Full Sim | v0.3
- Full Sim | v0.3 ECal
- Full Sim | v0.4

FCC-hh | Gen | Les Houches Samples

Additional stats about the production can be found here.

Name pw_pp_hh_lambda100_5f

Expand table

No	Name	Nevents	Nfiles	Nbad	Neos	Size [GB]	Output Path	Main Process	Final States	Matching Param	Cross Section [pb]
576	pw_pp_hh_lambda100_5f	10,329,977	1033	0	1033	1.64	/eos/experiment/fcc/hh/generation/lhe//pw_pp_hh_lambda100_5f/	gg->HH (NLO)	inclusive		1.13822
609	pw_pp_hh_lambda100_5f_80TeV	110,030	1103	0	1033	0.80	/eos/experiment/fcc/hh/generation/lhe//pw_pp_hh_lambda100_5f_80TeV/	gg->HH (NLO)	inclusive		1.13822
612	pw_pp_hh_lambda100_5f_100TeV_testSA	24,000	8	0	0	0.00	/eos/experiment/fcc/hh/generation/lhe//pw_pp_hh_lambda100_5f_100TeV_testSA/	1	1	1	1
613	pw_pp_hh_lambda100_5f_100TeV_testSA_fixed	24,000	8	0	0	0.00	/eos/experiment/fcc/hh/generation/lhe//pw_pp_hh_lambda100_5f_100TeV_testSA_fixed/	1	1	1	1
622	pw_pp_hh_lambda100_5f_80TeV_SA	1,200,000	400	0	0	0.19	/eos/experiment/fcc/hh/generation/lhe//pw_pp_hh_lambda100_5f_80TeV_SA/	1	1	1	1
625	pw_pp_hh_lambda100_5f_120TeV_SA	1,191,000	397	0	0	0.19	/eos/experiment/fcc/hh/generation/lhe//pw_pp_hh_lambda100_5f_120TeV_SA/	1	1	1	1

Hadronization, decay & fast simulation

Step 2.1: Setting up the key4hep stack

```
mkdir EDM4HEP_prod
cd  EDM4HEP_prod

source /cvmfs/sw.hsf.org/key4hep/setup.sh

which DelphesPythia8_EDM4HEP
```

- Turnkey software for future projects, e.g. CEPC, ILC, muon collider, ..
- Complete workflow from generator to analysis (although for FCC we are not using every step)
- In practice: A complete software stack to set up in one simple step
- Will use the DelphesPythia8_EDM4HEP tool to run Pythia + Delphes + produce EDM4hep output file, from k4SimDelphes

Hadronization, decay & fast simulation

Step 2.2: Running k4SimDelphes

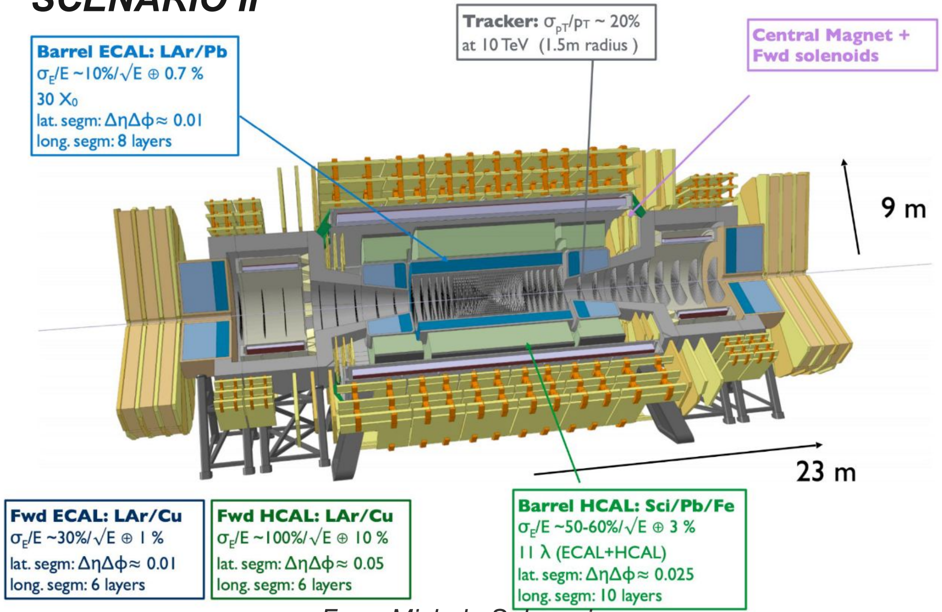
```
DelphesPythia8_EDM4HEP -h

ls $DELPHES_DIR/cards/FCC/scenarios/FCChh_I.tcl

ls $K4SIMDELPHES/edm4hep_output_config.tcl

ls /eos/experiment/fcc/hh/tutorials/lhe_unpacked_tester/

DelphesPythia8_EDM4HEP
$DELPHES_DIR/cards/FCC/scenarios/FCChh_I.tcl
$K4SIMDELPHES/edm4hep_output_config.tcl
/eos/experiment/fcc/hh/tutorials/lhe_unpacked_tester/tester_pwp8_pp_
hh_5f_hhbbyy.cmd pwp8_pp_hh_5f_hhbbyy.root
```

- Need to provide:
  - Config_file = Delphes card
  - Output_config_file for EDM4hep
  - pythia_card = Decay & Hadr.
  - output_file = Name of file
- Delphes cards ship with the Delphes installation: Can browse them here
- Standard EDM4hep output config file comes with key4hep stack
- Tester LHE file(s) & pythia cards provided in tutorial eos space

Hadronization, decay & fast simulation

Step 2.3: Understanding the Delphes card

SCENARIO II



From Michele Selvaggi

Note: Both scenarios implement fixes w.r.t the original, e.g. bremsstrahlung for electrons, multiple scattering, resolutions in forward region

- Two current Delphes scenarios for FCC-hh:
  - Scenario I: Idealistic scenario for ultimate precision
  - Scenario II: Baseline scenario based on FCC-hh detector concept from CDR
    - Should be default for new studies

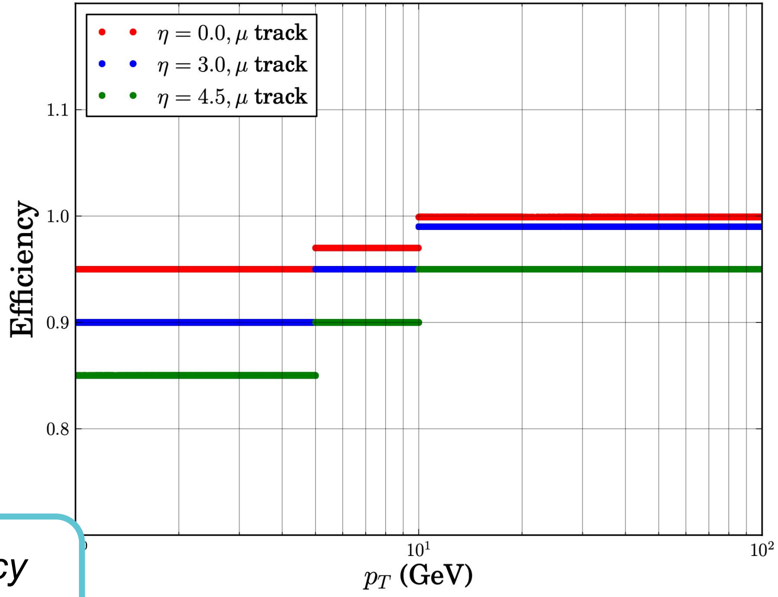
	Relative *p* resolution		Efficiency	
	Scenario I	Scenario II	Scenario I	Scenario II
Electrons	0.4-1%	0.8-3%	76-95%	72-90%
Muons	0.5-3%	1-6%	90-99%	88-97%
Medium b-tagging			80-90%	76-86%

Hadronization, decay & fast simulation

Step 2.3: Understanding the Delphes card

https://github.com/delphes/delphes/blob/master/cards/FCC/scenarios/FCChh_I.tcl

```
###################
# Muon efficiency
###################

module Efficiency MuonEfficiency {
  set InputArray MuonFilter/muons
  set OutputArray muons

  # set EfficiencyFormula {efficiency formula as a function of eta and pt}

  set EfficiencyFormula {
  (pt <= 1.0) * (0.00) + \
  (abs(eta) <= 2.5) * (pt > 1.0 && pt < 5.0)  * (0.95) +
  (abs(eta) <= 2.5) * (pt > 5.0 && pt < 10.0) * (0.97) +
  (abs(eta) <= 2.5) * (pt > 10.0)             * (0.999) +

  (abs(eta) > 2.5 && abs(eta) <= 4.0) * (pt > 1.0 && pt < 5.0)  * (0.90) +
  (abs(eta) > 2.5 && abs(eta) <= 4.0) * (pt > 5.0 && pt < 10.0) * (0.95) +
  (abs(eta) > 2.5 && abs(eta) <= 4.0) * (pt > 10.0)             * (0.99) +

  (abs(eta) > 4.0 && abs(eta) <= 6.0) * (pt > 1.0 && pt < 5.0)  * (0.85) + \
  (abs(eta) > 4.0 && abs(eta) <= 6.0) * (pt > 5.0 && pt < 10.0) * (0.90) + \
```



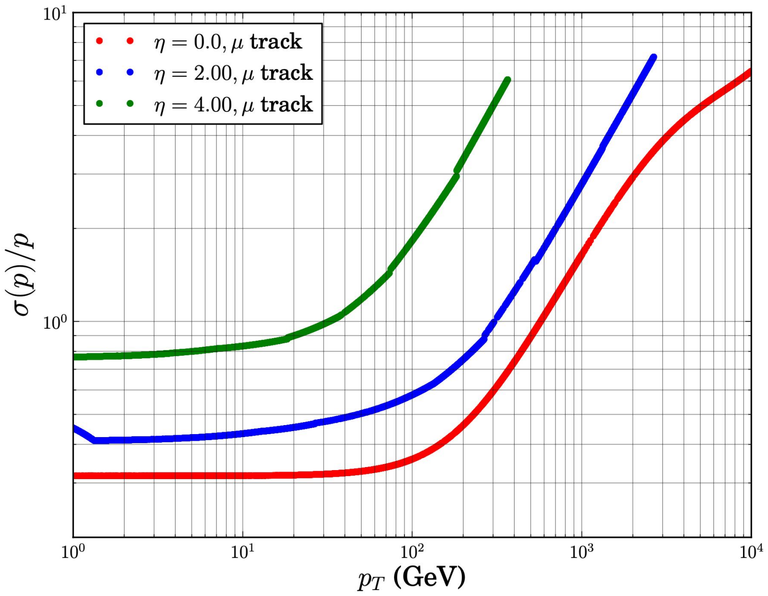
*Total efficiency = MuonTrackingEfficiency*MuonEfficiency*
In which |η| range are we reconstructing muon tracks?

Hadronization, decay & fast simulation

Step 2.3: Understanding the Delphes card

https://github.com/delphes/delphes/blob/master/cards/FCC/scenarios/muonMomentumResolution_I.tcl

```
#  Analytical formula


set ResolutionFormula {

( abs(eta) >= 0 && abs(eta) <= 1 ) *

    (sqrt(1e-5/sin(2*atan(exp(-abs(eta))))^2 + (
                          3*9.06262e-8 *pt^2* cosh(
         eta)^2 *(2.82074e-7/sin(2*atan(exp(-abs(eta))))^2 + (
          504.525 *(1/400000000 + (0.117945* 1/cosh(eta)^2)/(
             pt^2 *sin(2*atan(exp(-abs(eta))))^2)))/
          sin(2*atan(exp(-abs(eta))))^2) *sin(2*atan(exp(-abs(eta))))^2)/(
       0.00516429/sin(2*atan(exp(-abs(eta))))^2 + (
        96868.8 *(1/400000000 + 5*(0.117945 * 1/cosh(eta)^2)/(
           pt^2 *sin(2*atan(exp(-abs(eta))))^2)))/
        sin(2*atan(exp(-abs(eta))))^2))

    ) +


( abs(eta) > 1 && abs(eta) < 1.30 ) *

    (sqrt(1e-5/tan(2*atan(exp(-abs(eta))))^2 + (
                          3*9.06262e-8 *pt^2* cosh(
         eta)^2 *(2.82074e-7/sin(2*atan(exp(-abs(eta))))^2 + (
          504.525 *(1/400000000 + (0.117945* 1/cosh(eta)^2)/(
             pt^2 *sin(2*atan(exp(-abs(eta))))^2)))/
          sin(2*atan(exp(-abs(eta))))^2) *sin(2*atan(exp(-abs(eta))))^2)/(
       0.00516429/sin(2*atan(exp(-abs(eta))))^2 + (
        96868.8 *(1/400000000 + 5*(0.117945 * 1/cosh(eta)^2)/(
           pt^2 *sin(2*atan(exp(-abs(eta))))^2)))/
        sin(2*atan(exp(-abs(eta))))^2))

    ) +

```



Hadronization, decay & fast simulation

Step 2.3: Understanding the Delphes card

https://github.com/delphes/delphes/blob/master/cards/FCC/scenarios/FCChh_I.tcl

```
##################
# ROOT tree writer
##################

module TreeWriter TreeWriter {
# add Branch InputArray BranchName BranchClass
  add Branch Delphes/allParticles Particle GenParticle

  add Branch GenMissingET/momentum GenMissingET MissingET

  #add Branch TrackMerger/tracks Track Track
  #add Branch TowerMerger/towers Tower Tower

  #Temporary addition for manual isoVar validation/recalculation
  # add Branch EFlowFilter/eflow ParticleFlowCandidates ParticleFlowCandidate

  add Branch EFlowTrackMerger/eflowTracks EFlowTrack Track
  add Branch Calorimeter/eflowPhotons EFlowPhoton Tower
  add Branch Calorimeter/eflowNeutralHadrons EFlowNeutralHadron Tower

  add Branch UniqueObjectFinder/photons Photon Photon
  add Branch UniqueObjectFinder/electrons Electron Electron
  add Branch UniqueObjectFinder/muons Muon Muon
  add Branch UniqueObjectFinder/jets Jet Jet

  #collections for objects before isolation and uniqueobjectfinder
  add Branch PhotonEfficiency/photons PhotonNoIso Photon
  add Branch ElectronEfficiency/electrons ElectronNoIso Electron
  add Branch MuonEfficiency/muons MuonNoIso Muon
  add Branch JetEnergyScale/jets JetNoIso Jet

```

- Objects to write out are specified in the TreeWriter module
- *Which jets are we actually using? Which steps (i.e. modules) have they passed through? What is the minimum jet pT?*

Hadronization, decay & fast simulation

Step 2.4: Understanding the Pythia card

tester_pwp8_pp_hh_5f_hhbbyy.cmd

```
! 1) Settings that will be used in a main program.

Main:numberOfEvents = 10000            ! number of events to generate
Main:timesAllowErrors = 100        ! abort run after this many flawed events
#Random:seed = 1234                ! initialize random generator with a seed

! 2) Settings related to output in init(), next() and stat() functions.
Init:showChangedSettings = on      ! list changed settings
Init:showAllSettings = off         ! list all settings
Init:showChangedParticleData = on  ! list changed particle data
Init:showAllParticleData = off     ! list all particle data
Next:numberCount = 100              ! print message every n events
Next:numberShowLHA = 1             ! print LHA information n times
Next:numberShowInfo = 1            ! print event information n times
Next:numberShowProcess = 1         ! print process record n times
Next:numberShowEvent = 1           ! print event record n times
Stat:showPartonLevel = off         ! additional statistics on MPI

! 3) Tell Pythia that LHEF input is used
Beams:frameType             = 4
Beams:LHEF = /eos/experiment/fcc/hh/tutorials/lhe_unpacked_tester/ggHH_events_000068811.lhe ! the LHEF to read from

! 4) Exclusive decay with ResonanceDecayUserHook
25:onMode = off
25:onIfMatch = 5 -5
25:onIfMatch = 22 22
ResonanceDecayFilter:filter = on
ResonanceDecayFilter:exclusive = on
ResonanceDecayFilter:mothers = 25
ResonanceDecayFilter:daughters = 5,5,22,22

! 5) POWHEG parameters
! Number of outgoing particles of POWHEG Born level process
! (i.e. not counting additional POWHEG radiation)
POWHEG:nFinal = 2

```

- *How many events are we processing?*
- *Which LHE file are we reading from?*
- *Which final state of the Higgs boson decays are we requiring?*

Hadronization, decay & fast simulation

Step 2.4: Understanding the Pythia card

tester_pwp8_pp_hh_5f_hhbbyy.cmd

```
! 1) Settings that will be used in a main program.

Main:numberOfEvents = 10000          ! number of events to generate
Main:timesAllowErrors = 100        ! abort run after this many flawed events
#Random:seed = 1234                ! initialize random generator with a seed

! 2) Settings related to output in init(), next() and stat() functions.
Init:showChangedSettings = on      ! list changed settings
Init:showAllSettings = off         ! list all settings
Init:showChangedParticleData = on  ! list changed particle data
Init:showAllParticleData = off     ! list all particle data
Next:numberCount = 100              ! print message every n events
Next:numberShowLHA = 1             ! print LHA information n times
Next:numberShowInfo = 1            ! print event information n times
Next:numberShowProcess = 1         ! print process record n times
Next:numberShowEvent = 1           ! print event record n times
Stat:showPartonLevel = off         ! additional statistics on MPI

! 3) Tell Pythia that LHEF input is used
Beams:frameType             = 4
Beams:LHEF = /eos/experiment/fcc/hh/tutorials/lhe_unpacked_tester/ggHH_events_000068811.lhe ! the LHEF to read from

! 4) Exclusive decay with ResonanceDecayUserHook
25:onMode = off
25:onIfMatch = 5 -5
25:onIfMatch = 22 22
ResonanceDecayFilter:filter = on
ResonanceDecayFilter:exclusive = on
ResonanceDecayFilter:mothers = 25
ResonanceDecayFilter:daughters = 5,5,22,22

! 5) POWHEG parameters
! Number of outgoing particles of POWHEG Born level process
! (i.e. not counting additional POWHEG radiation)
POWHEG:nFinal = 2
```

- *How many events are we processing?*
- *Which LHE file are we reading from?*
- *Which final state of the Higgs boson decays are we requiring?*

Directory with official pythia cards is:
/eos/experiment/fcc/hh/utils/pythiacards

Hadronization, decay & fast simulation

Step 2.5: Understanding the EDM4hep configuration & file

Type of EDM4hep collection

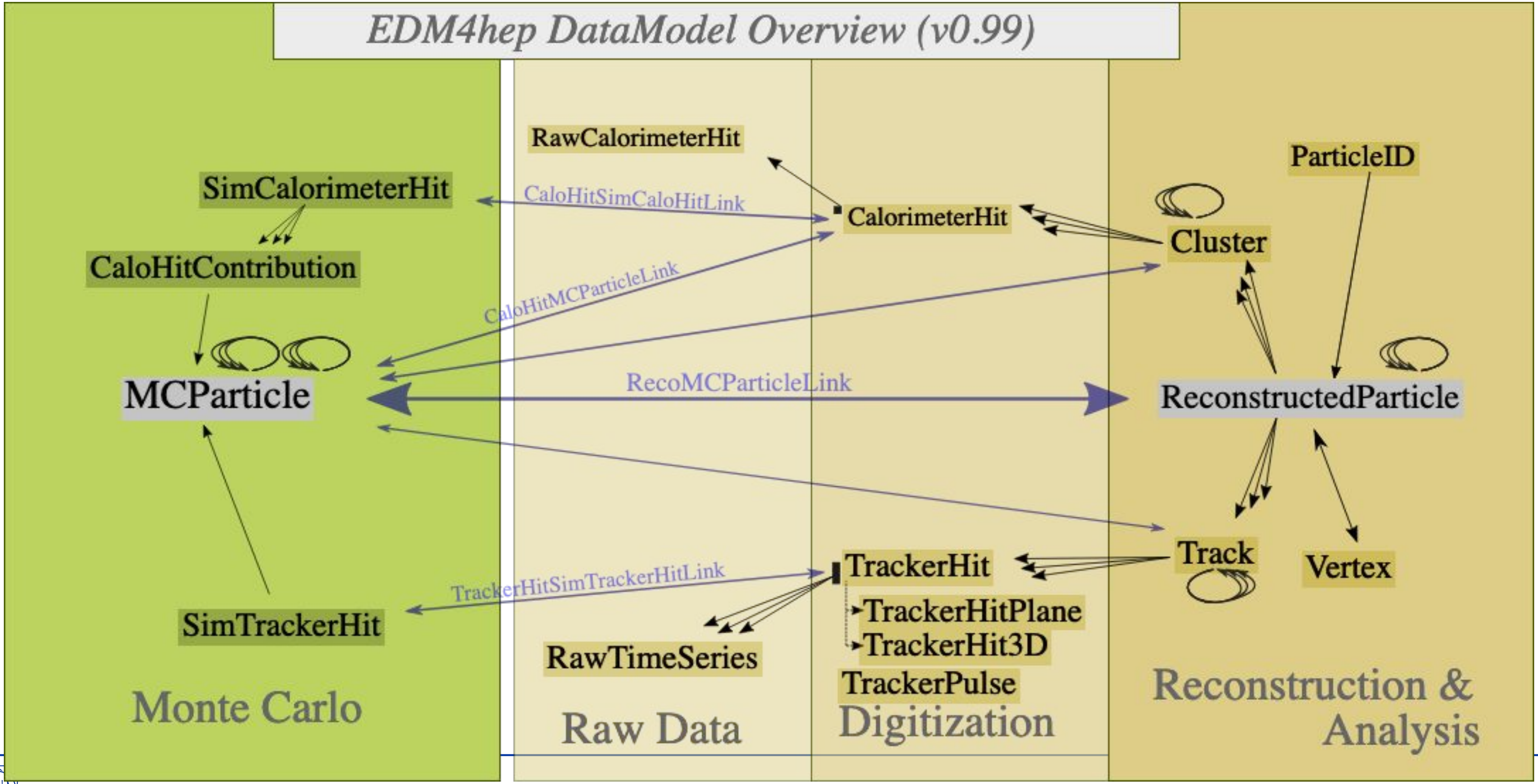
Name of collection (must match Delphes card names)

edm4hep_output_config.tcl

```
module EDM4HepOutput EDM4HepOutput {
    add ReconstructedParticleCollections EFlowTrack EFlowPhoton EFlowNeutralHadron
    add GenParticleCollections           Particle
    add JetCollections                   Jet
    add MuonCollections                  Muon
    add ElectronCollections              Electron
    add PhotonCollections                Photon
    add MissingETCollections             MissingET
    add ScalarHTCollections              ScalarHT
    set RecoParticleCollectionName       ReconstructedParticles
    set RecoMCParticleLinkCollectionName  MCRecoAssociations
}
```

Hadronization, decay & fast simulation

Step 2.5: Understanding the EDM4hep configuration & file



Hadronization, decay & fast simulation

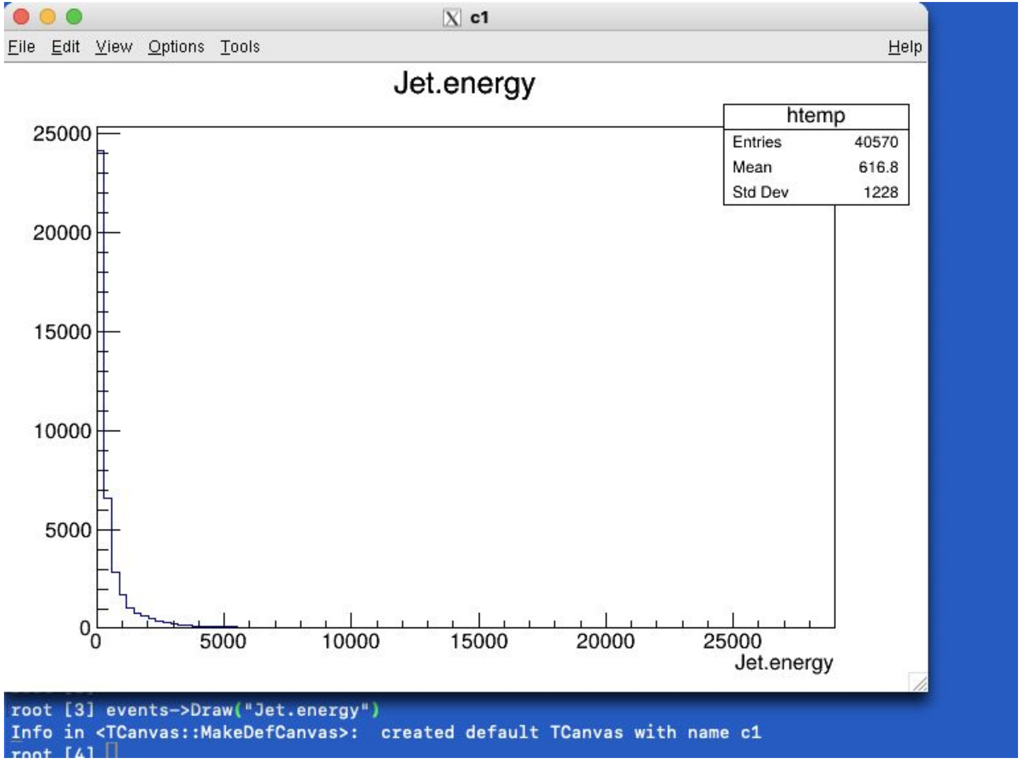
Step 2.5: Understanding the EDM4hep configuration & file

```
Attaching file /eos/experiment/fcc/hh/tutorials/edm4hep_tutorial_data/pwp8_pp_hh_5f_hhbbyy.root as _file0...
(TFile *) 0x3906b20
root [1] _file0->ls()
TNetXNGFile**           root://eospublic.cern.ch//eos/experiment/fcc/hh/tutorials/edm4hep_tutorial_data/pwp8_pp_hh_5f_hhbbyy.root
 TNetXNGFile*           root://eospublic.cern.ch//eos/experiment/fcc/hh/tutorials/edm4hep_tutorial_data/pwp8_pp_hh_5f_hhbbyy.root
  KEY: TTree    events;72       events data tree [current cycle]
  KEY: TTree    events;71       events data tree [backup cycle]
  KEY: TTree    podio_metadata;1        metadata tree for podio I/O functionality
root [2] events->Print()
******************************************************************************
*Tree    :events    : events data tree                                       *
*Entries :    10000 : Total =     11750891566 bytes  File  Size = 1860078386 *
*        :          : Tree compression factor =   6.32                       *
******************************************************************************
*Br    0 :CalorimeterHits : Int_t CalorimeterHits_                           *
*Entries :    10000 : Total  Size=     106955 bytes  File Size  =      77401 *
*Baskets :       72 : Basket Size=      32000 bytes  Compression=   1.12     *
*............................................................................*
*Br    1 :CalorimeterHits.cellID : ULong_t cellID[CalorimeterHits_]          *
*Entries :    10000 : Total  Size=   36733008 bytes  File Size  =     405688 *
*Baskets :       89 : Basket Size=     642560 bytes  Compression=  90.54     *
*............................................................................*
*Br    2 :CalorimeterHits.energy : Float_t energy[CalorimeterHits_]          *
*Entries :    10000 : Total  Size=   18390965 bytes  File Size  =     227436 *
*Baskets :       80 : Basket Size=     322560 bytes  Compression=  80.85     *
*............................................................................*
*Br    3 :CalorimeterHits.energyError : Float_t energyError[CalorimeterHits_]*
*Entries :    10000 : Total  Size=   18391385 bytes  File Size  =     227828 *
*Baskets :       80 : Basket Size=     322560 bytes  Compression=  80.72     *
*............................................................................*
*Br    4 :CalorimeterHits.time : Float_t time[CalorimeterHits_]              *
*Entries :    10000 : Total  Size=   18390797 bytes  File Size  =   18035847 *
*Baskets :       80 : Basket Size=     322560 bytes  Compression=   1.02     *
```

Hadronization, decay & fast simulation

Step 2.5: Understanding the EDM4hep configuration & file

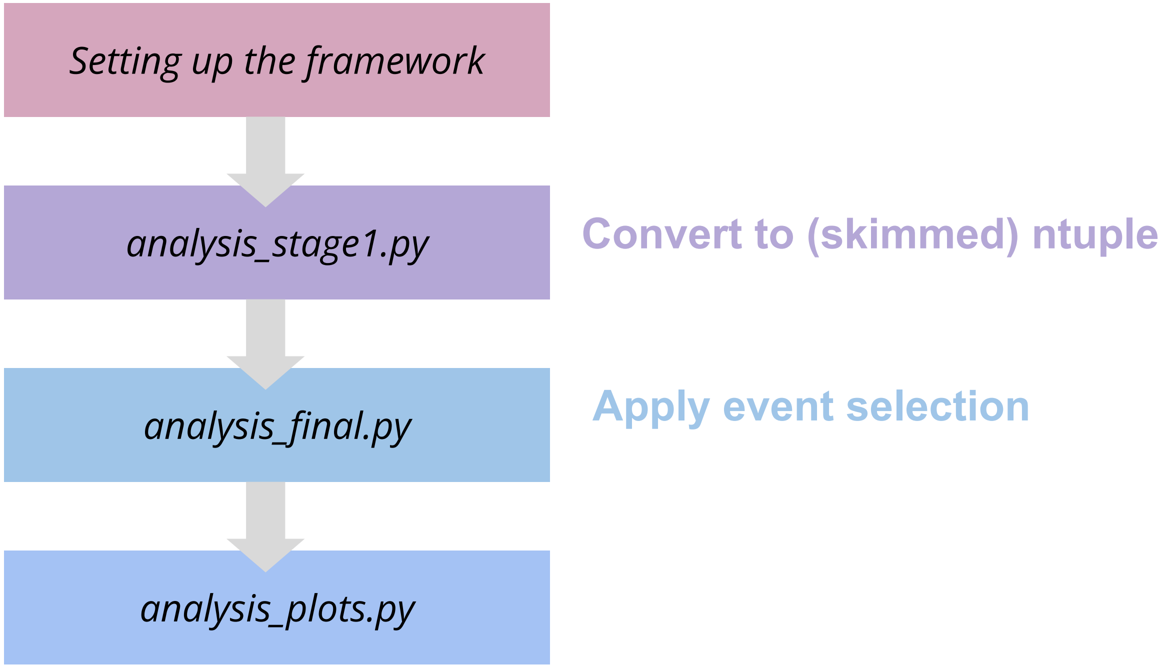
*Can analyse/plot from EDM4hep directly with [podio](), but for FCC we have the common **[FCCAnalyses]()** framework*



Physics Analysis

Step 3: Overview FCCAnalyses

- FCCAnalyses is a common software framework to analyse EDM4hep events using ROOT's RDataframe
  - Build an "analysis graph" with very simple syntax in python code
  - C++ libraries for the complex computations
  - Examples and tutorials available here



Physics Analysis

Step 3.1: Setting up the FCCAnalyses framework

```
mkdir FCCAnalyses_examples
cd FCCAnalyses_examples

source /cvmfs/sw-nightlies.hsf.org/key4hep/setup.sh

which fccanalysis

ls $FCCANALYSES/../share/examples/examples/FCChh/ggHH_bbyy/

cp -r $FCCANALYSES/../share/examples/examples/FCChh/ggHH_bbyy/ .
```

- Two ways to setup FW:
  - Get your local copy of FCCAnalyses git repo
  - Use build that ships with key4hep stack → for simplicity will do this for tutorial
- *Start a clean shell, we will need to setup the nightlies, and will copy the example files to local*

Physics Analysis

Step 3.2: Converting to (skimmed) analysis ntuple

analysis_stage1.py

```python
dframe2 = (
    dframe

    ############################################ DEFINITION OF VARIABLES ############################################

    # generator event weight
    .Define("weight",  "EventHeader.weight")

    ############################################ PHOTONS ############################################

    .Define("gamma",  "FCCAnalyses::ReconstructedParticle::get(Photon_objIdx.index, ReconstructedParticles)")
    .Define("selpt_gamma", "FCCAnalyses::ReconstructedParticle::sel_pt(30.)(gamma)")
    .Define("sel_gamma_unsort", "FCCAnalyses::ReconstructedParticle::sel_eta(4)(selpt_gamma)")
    .Define("sel_gamma", "AnalysisFCChh::SortParticleCollection(sel_gamma_unsort)") #sort by pT
```

Analysis_FCChh.cc

```cpp
// SortParticleCollection
//
ROOT::VecOps::RVec<edm4hep::ReconstructedParticleData>
AnalysisFCChh::SortParticleCollection(
    ROOT::VecOps::RVec<edm4hep::ReconstructedParticleData> particles_in) {
  if (particles_in.size() < 2) {
    return particles_in;
  } else {
    auto sort_by_pT = [&](edm4hep::ReconstructedParticleData part_i,
                          edm4hep::ReconstructedParticleData part_j) {
      return (getTLV_reco(part_i).Pt() > getTLV_reco(part_j).Pt());
    };
    std::sort(particles_in.begin(), particles_in.end(), sort_by_pT);
    return particles_in;
  }
}
```

- The stage1.py analysis script defines a RDataFrame with all the branches we want to store, and applies a pre-selection
  - To access the variables and do more complex calculations we use C++ libraries in the analyzers/dataframe directory

Physics Analysis

Step 3.2: Converting to (skimmed) analysis ntuple

```
cd ggHH_bbyy

fccanalysis run ggHH_bbyy/analysis_stage1.py
```

```
[bistapf@lxplus994 FCCAnalyses_examples]$ fccanalysis run ggHH_bbyy/analysis_stage1.py
----> INFO: Loading analyzers from libFCCAnalyses...
----> INFO: Loading analysis script:
            /afs/cern.ch/user/b/bistapf/FCChh_tutorial/FCCAnalyses_examples/ggHH_bbyy/analysis_stage1.py
----> INFO: No multithreading enabled. Running in single thread...
----> INFO: Using generator weights
----> INFO: Started processing sample "pwp8_pp_hh_5f_hhbbyy" ...
----> INFO: Number of the output files: 1
----> INFO: Running locally...
----> Warning: Input file is missing information about original number of events!
----> Warning: Input file is missing information about original sum of weights!
----> INFO: Creating dataframe object from files:
              - root://eospublic.cern.ch//eos/experiment/fcc/hh/tutorials/edm4hep_tutorial_data//pwp8_pp_hh_5f_hhbbyy.root

----> INFO: Number of local events: 10,000
----> INFO: Local sum of weights: 11,488.07
----> INFO: Output file path:
            outputs/FCChh/ggHH_bbyy/presel/pwp8_pp_hh_5f_hhbbyy.root
----> INFO: =================================== SUMMARY ===================================
            Elapsed time (H:M:S):     00:00:13
            Events processed/second: 718
            Total events processed:  10,000
            No. result events:        2,684
            Reduction factor local:  0.2684
            Total sum of weights processed:  11,488.07
            No. result weighted events :       3,094.37
            Reduction factor local, weighted:  0.2694
            ===============================================================================
```

- The stage1.py analysis script defines a RDataFrame with all the branches we want to store, and applies a pre-selection
  - To access the variables and do more complex calculations we use C++ libraries in the analyzers/dataframe directory
- *What is the efficiency of our pre-selection?*

Physics Analysis

Step 3.2: Converting to (skimmed) analysis ntuple

```
cd ggHH_bbyy

fccanalysis run ggHH_bbyy/analysis_stage1.py
```

```
[bistapf@lxplus994 FCCAnalyses_examples]$ fccanalysis run ggHH_bbyy/analysis_stage1.py
----> INFO: Loading analyzers from libFCCAnalyses...
----> INFO: Loading analysis script:
        /afs/cern.ch/user/b/bistapf/FCChh_tutorial/FCCAnalyses_examples/ggHH_bbyy/analysis_stage1.py
----> INFO: No multithreading enabled. Running in single thread...
----> INFO: Using generator weights
----> INFO: Started processing sample "pwp8_pp_hh_5f_hhbbyy" ...
----> INFO: Number of the output files: 1
----> INFO: Running locally...
----> Warning: Input file is missing information about original number of events!
----> Warning: Input file is missing inf
----> INFO: Creating dataframe objec
        - root://eospublic.cern

----> INFO: Number of local events
----> INFO: Local sum of weights:
----> INFO: Output file path:
        outputs/FCChh/ggHH_bby
----> INFO: ======================
        Elapsed time (H:M:S):
        Events processed/secon
        Total events processed
        No. result events:
        Reduction factor local
        Total sum of weights p
        No. result weighted ev
        Reduction factor local
        ======================
```

You now have a "standard" root ntuple that you can use in any framework for plotting, applying further selection and/or analysis (e.g. MVA) in the usual way.

- The stage1.py analysis script defines a RDataFrame with all the branches we want to store, and applies a pre-selection
  - To access the variables and do more complex calculations we C++ libraries in the yzers/dataframe directory
  - *ne efficiency of our ction?*

Physics Analysis

Step 3.3: Applying the event selection

```
#Link to the dictonary that contains all the cross section informations etc...
procDict = "/eos/experiment/fcc/hh/tutorials/edm4hep_tutorial_data/FCChh_procDict_tutorial.json"
#Note the numbeOfEvents and sumOfWeights are placeholders that get overwritten with the correct v

#How to add a process that is not in the official dictionary:
# procDictAdd={"pwp8_pp_hh_5f_hhbbyy": {"numberOfEvents": 4980000, "sumOfWeights": 4980000.0, "cr

# Expected integrated luminosity
intLumi = 30e+06  # pb-1

# Whether to scale to expected integrated luminosity
doScale = True

#Number of CPUs to use
nCPUS = 2

#produces ROOT TTrees, default is False
doTree = True
```

analysis_final.py

```
{
    "pwp8_pp_hh_5f_hhbbyy": {
    "numberOfEvents": 10000,
    "sumOfWeights": 10000.0,
    "crossSection": 0.0029844,
    "kfactor": 1.0753,
    "matchingEfficiency": 1.0
    }
}
```

FCChh_procDict_tutorial.json

- We can apply additional selection, define histograms, get a cutflow & store histograms/the ntuple at every selection cut with analysis_final.py
    - If we provide the cross-section, lumi, etc the histograms will be scaled to expected number of events
    - *Why is the cross-section here not the same as we saw in the LHE database?*

Physics Analysis

Step 3.3: Applying the event selection

```
fccanalysis final ggHH_bbyy/analysis_final.py
```

```
----> INFO: Running over process: pwp8_pp_hh_5f_hhbbyy
----> INFO: Generator scale factor for "pwp8_pp_hh_5f_hhbbyy": 0.003209
----> INFO:  - cross-section:                          0.002984 pb
----> INFO:  - kfactor:                                1.075
----> INFO:  - matching efficiency:                    1
----> INFO: Integrated luminosity: 3e+07 pb-1
----> INFO: Defining cuts and histograms
----> INFO: Evaluating...
----> INFO: Successfully applied event weights, got weighted events = 3,094.37
----> INFO: Done
----> INFO: Scaling cut yields...
----> INFO: Cutflow:
                                   Raw events    Scaled events
              - All events              2,684         2.59e+04
              - sel0_myy                2,678         2.58e+04
              - sel1_mbb                2,274         2.19e+04
----> INFO: Saving the outputs...
----> INFO: Scaling the histograms...
Updating file outputs/FCChh/ggHH_bbyy/final/pwp8_pp_hh_5f_hhbbyy_sel0_myy.root
Number of events processed: 10000
Sum of weights: 11488.0703125
Updating file outputs/FCChh/ggHH_bbyy/final/pwp8_pp_hh_5f_hhbbyy_sel1_mbb.root
Number of events processed: 10000
Sum of weights: 11488.0703125
----> INFO: Saving results in LaTeX tables to:
            outputs/FCChh/ggHH_bbyy/final/outputTabular.txt
----> INFO:
            =================================== SUMMARY ===================================
            Elapsed time (H:M:S):     00:00:09
            Events processed/second:  277
            Total events processed:   2,684
            ================================================================================
```

- We can apply additional selection, define histograms, get a cutflow & store histograms/the ntuple at every selection cut with analysis_final.py
  - If we provide the cross-section, lumi, etc the histograms will be scaled to expected number of events
  - *How many expected bbyy events pass our selection?*

Physics Analysis

Step 3.3: Applying the event selection

```
fccanalysis final ggHH_bbyy/analysis_final.py
```

```
---> INFO: Running over process: pwp8_pp_hh_5f_hhbbyy
---> INFO: Generator scale factor for "pwp8_pp_hh_5f_hhbbyy": 0.003209
---> INFO:  - cross-section:                              0.002984 pb
---> INFO:  - kfactor:                                    1.075
---> INFO:  - matching efficiency:                        1
---> INFO: Integrated luminosity: 3e+07 pb-1
---> INFO: Defining cuts and histograms
---> INFO: Evaluating...
---> INFO: Successfully applied event weights, got weighted events = 3,094.37
---> INFO: Done
---> INFO: Scaling cut yields...
---> INFO: Cutflow:

             - All events
             - sel0_myy
             - sel1_mbb
---> INFO: Saving the output
---> INFO: Scaling the histog
Updating file outputs/FCChh/gg
Number of events processed: 1
Sum of weights: 11488.0703125
Updating file outputs/FCChh/gg
Number of events processed: 1
Sum of weights: 11488.0703125
---> INFO: Saving results in
           outputs/FCChh/ggH
---> INFO:
           ==================
           Elapsed time (H:M:S):
           Events processed/second: 277
           Total events processed:  2,684
           ================================================================================
```

You now have an ntuple as well as a file of histograms (_histo.root) at every step of the selection sequence. (+A cutflow table)

- We can apply additional selection, define histograms, get a cutflow & store histograms/the ntuple at every selection cut with analysis_final.py
  - If we provide the cross-section, lumi, etc the histograms will be …ed to expected number of …nts
  - *… many expected bbyy …nts pass our selection?*

Physics Analysis

Step 3.4: Plotting distributions

analysis_plots.py

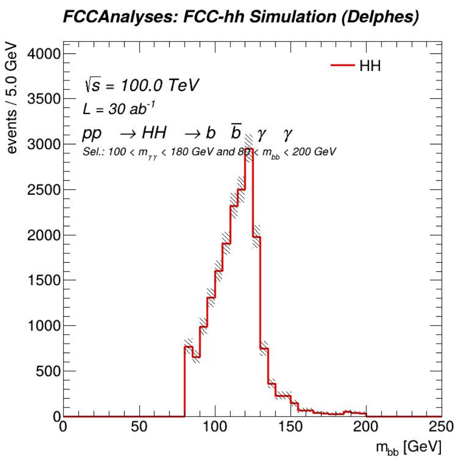
```
import ROOT

# global parameters
intLumi        = 30e+06 #in pb-1
ana_tex        = 'pp #rightarrow HH #rightarrow b #bar{b} #gamma #gamma '
delphesVersion = '3.4.2'
energy         = 100
collider       = 'FCC-hh'
inputDir       = 'outputs/FCChh/ggHH_bbyy/final/'
formats        = ['png','pdf']
yaxis          = ['lin','log']
stacksig       = ['nostack']
# stacksig       = ['stack','nostack']
outdir         = 'outputs/FCChh/ggHH_bbyy/plots/'
plotStatUnc    = True

variables = ['myy','myy_zoom', 'mbb', 'mbb_zoom', 'y1_pT','y2_pT']

# rebin = [1, 1, 1, 1, 2] # uniform rebin per variable (optional)

### Dictionary with the analysis name as a key, and the list of selections to be plotted for this analysis. Tl
selections = {}
selections['bbyy_analysis']   = ["sel0_myy","sel1_mbb"]

extralabel = {}
extralabel['sel0_myy'] = "Sel.: 100 < m_{#gamma#gamma} < 180 GeV"
extralabel['sel1_mbb'] = "Sel.: 100 < m_{#gamma#gamma} < 180 GeV and 80 < m_{bb} < 200 GeV"

colors = {}
colors['bbyy_signal'] = ROOT.kRed

plots = {}
plots['bbyy_analysis'] = {'signal':{'bbyy_signal':['pwp8_pp_hh_5f_hhbbyy']},
                }

legend = {}
legend['bbyy_signal'] = 'HH'
```

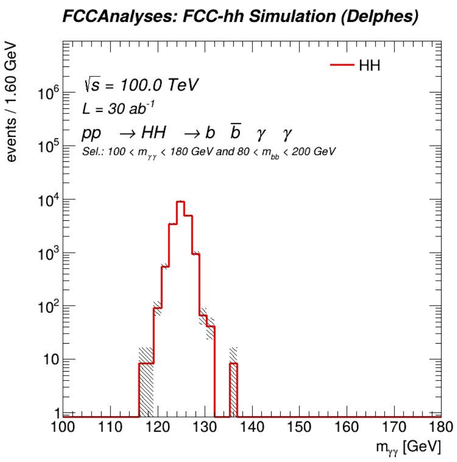
- With analysis_plots.py we can extract the histograms, and present them nicely on a canvas as usual (e.g. stacking backgrounds, adding uncertainties, legends & labels)

Physics Analysis

Step 3.4: Plotting distributions

```
fccanalysis plots ggHH_bbyy/analysis_plots.py
```





- With analysis_plots.py we can extract the histograms, and present them nicely on a canvas as usual (e.g. stacking backgrounds, adding uncertainties, legends & labels)

That's it for today :)

Summary

Key take-away messages

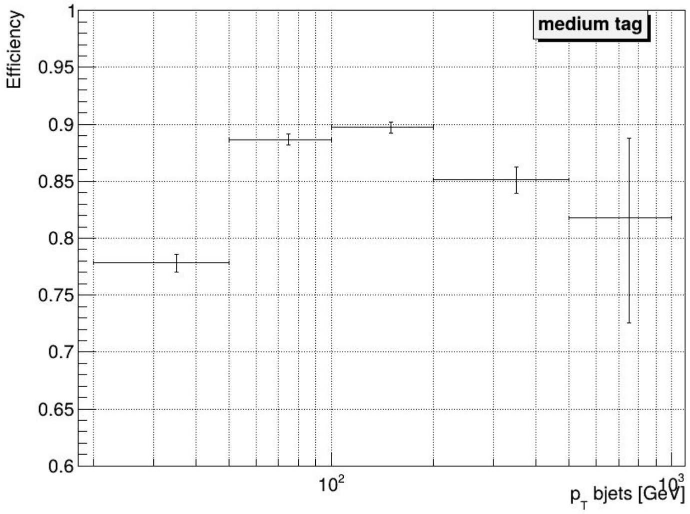
- We have a database of available LHE events
  - If you have or would need more processes added, please get in touch
- Hadronization, decay and Delphes fast simulation are run in one step, using key4hep tools
  - Two FCC-hh Delphes scenarios are available, FCChh_II.tcl is the baseline
  - We will start a new fast sim production campaign soon (v0.6), please get in touch about which samples you would need
- The EDM4hep files produced can be processed with the FCCAnalyses framework
  - *As usual: The software stack is under constant development, updates to the core code are in the pipeline, but the user code should not be affected (much)*
- Refer to our FCC-hh Physics & Performance documentation page
  - Join FCC software meetings & mailing lists (see FCC Software Documentation)

Thank you!

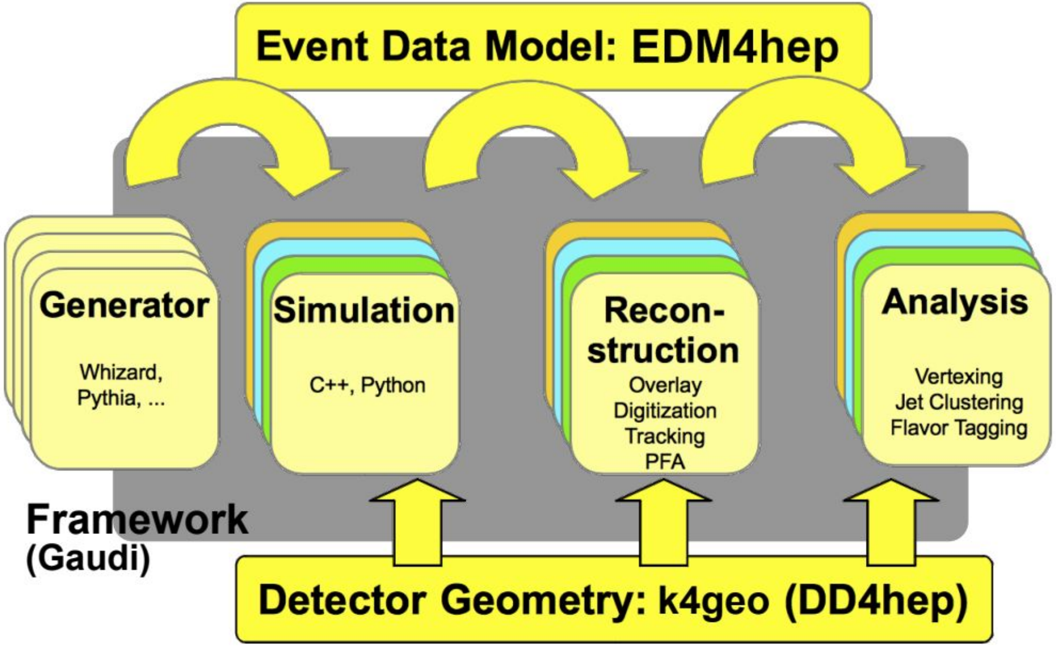
Physics Analysis

Bonus exercise: Checking Delphes b-tagging efficiencies

- The example directory `ggHH_bbyy` also contains two additional scripts:
  - `analysis_plot_tag_eff.py` to be run as a FCCAnalyses stage1 analysis
  - `plot_tag_eff.py` to be run standalone
- With which you can plot the b-tagging efficiency in our events in bins of pT and η, so that you can compare it to what is in the Delphes card



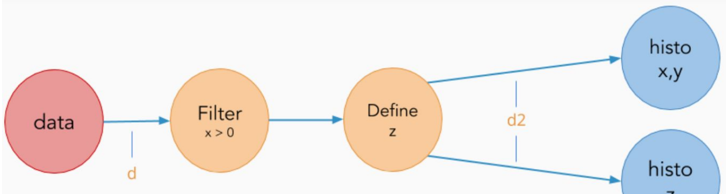
Key4hep project



- Turnkey software for future accelerators, used by different communities, e.g. CEPC, ILC, muon collider, ..
- Provides complete workflow from generator to analysis (although for FCC we are not using every step)
- In practice: A complete software stack to set up in one simple step

```
source /cvmfs/sw.hsf.org/key4hep/setup.sh
```

FCCAnalyses framework



From Juraj Smieško

```
# Mandatory: analyzers function to define the analysis graph, please make
# sure you return the dataframe, in this example it is dframe2
def analyzers(self, dframe):
    '''
    Analysis graph.
    '''

    dframe2 = (
        dframe

        .Define("weight",  "EventHeader.weight")

        ########################################### PHOTONS ###########################################

        .Define("gamma",  "FCCAnalyses::ReconstructedParticle::get(Photon_objIdx.index, ReconstructedParticles)")
        .Define("selpt_gamma", "FCCAnalyses::ReconstructedParticle::sel_pt(30.)(gamma)")
        .Define("sel_gamma_unsort", "FCCAnalyses::ReconstructedParticle::sel_eta(4)(selpt_gamma)")
        .Define("sel_gamma", "AnalysisFCChh::SortParticleCollection(sel_gamma_unsort)") #sort by pT

        .Define("ngamma",  "FCCAnalyses::ReconstructedParticle::get_n(sel_gamma)")
        .Define("g1_e",  "FCCAnalyses::ReconstructedParticle::get_e(sel_gamma)[0]")
        .Define("g1_pt",  "FCCAnalyses::ReconstructedParticle::get_pt(sel_gamma)[0]")
        .Define("g1_eta",  "FCCAnalyses::ReconstructedParticle::get_eta(sel_gamma)[0]")
        .Define("g1_phi",  "FCCAnalyses::ReconstructedParticle::get_phi(sel_gamma)[0]")
        .Define("g2_e",  "FCCAnalyses::ReconstructedParticle::get_e(sel_gamma)[1]")
        .Define("g2_pt",  "FCCAnalyses::ReconstructedParticle::get_pt(sel_gamma)[1]")
        .Define("g2_eta",  "FCCAnalyses::ReconstructedParticle::get_eta(sel_gamma)[1]")
        .Define("g2_phi",  "FCCAnalyses::ReconstructedParticle::get_phi(sel_gamma)[1]")
```

- FCCAnalyses is a common software framework to analyse EDM4hep events using ROOT's RDataframe
  - Build an "analysis graph" with very simple syntax in python code
  - C++ libraries for the complex computations
  - Examples and tutorials available here

What did we use for the ongoing HH studies?

Event generation	*Generators*: MG5_aMC, v 2.5.X (bkgs), POWHEG-BOX-V2 (sig) *PDF sets*: NN23LO1, NNPDF30_nlo_as_0118 from LHAPDF v6.1.6 *Production framework*: EventProducer from my fork, using custom key4hep release “2023-06-05-fcchh”
Detector simulation	*Delphes cards: Scenario I & II* *Framework:* Same EventProducer setup as above *Production Tags*: fcc_v05_scenarioI, fcc_v05_scenarioII
Samples (EDM4HEP)	*Edm4hep status:* Pre- official v1 release, v00-08
Physics analysis & statistical interpretation	*Analysis framework:* FCCAnalyses from my fork, with many custom fixes and additions, branched off in July 2023