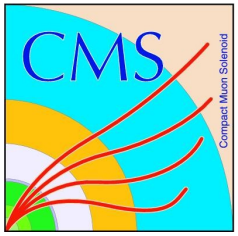
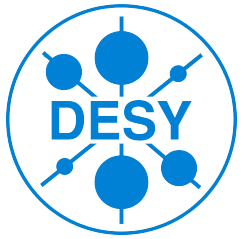


Particle Flow Short Exercise



Armin Burgmeier, Adrian Perieanu, Alexei Raspereza

CMS DAS School

DESY

January 2013

Outline

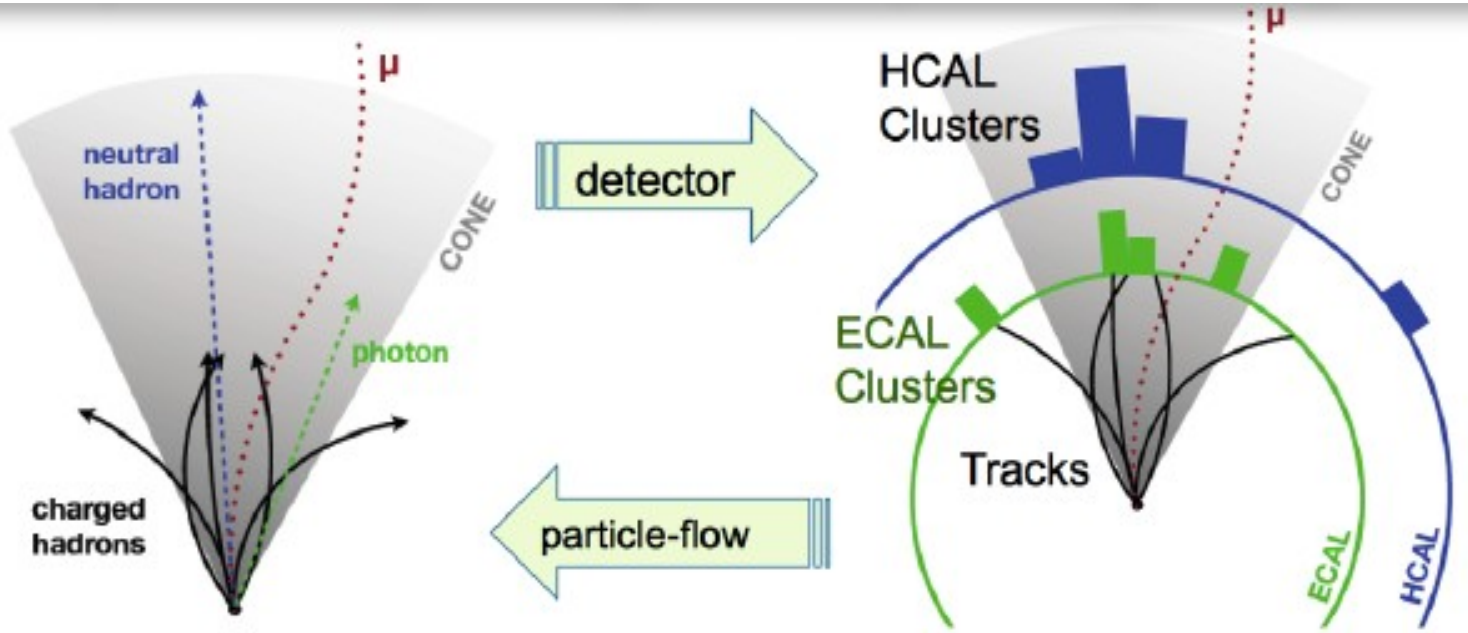
- The particle Flow Concept : Intro
- Exercises

  Part 1 : Analysis of Particle Flow objects

  - Inspecting PFlow Collection
  - Exploring PFlow constituents
    - Track and SuperCluster in PFElectrons
  - Identifying particles
  - Constructing PFlow Isolation
  - Selection of isolated leptons and plotting their mass

  Part 2 : Toy Particle Flow algorithm
- Goal : understanding particle flow concept and its applications

Particle Flow Concept

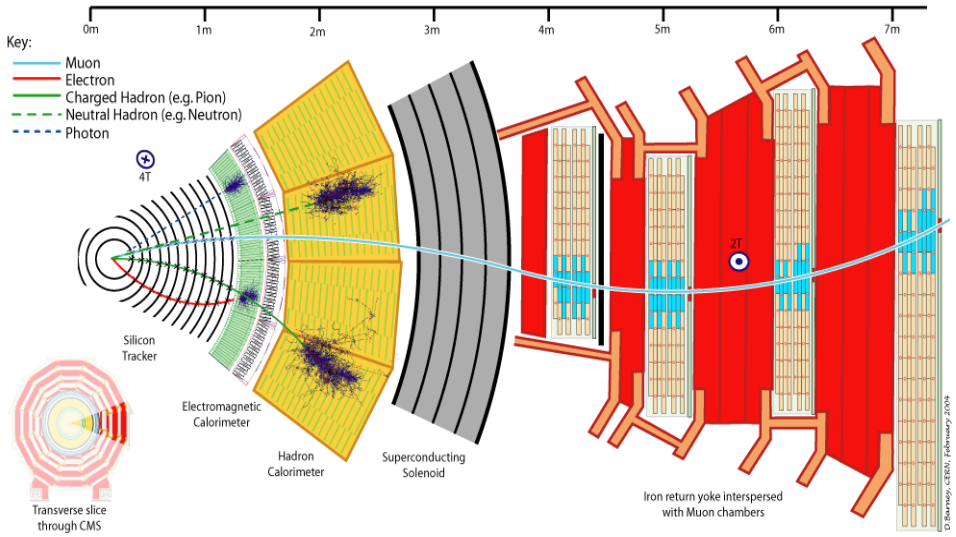


- Optimal combination of information from all subdetectors
- Returns a list of reconstructed particles
  - e,μ,γ, charged and neutral hadrons
    - Used in the analysis as if it came from a list of generated particles
    - Used as building blocks for jets, taus , missing transverse energy , isolation and PU particle identification

Particle Flow Concept

- Key ingredients : tracking, calorimeter clustering, track-cluster matching

Jet composition →



- Charged hadrons ≈65% of jet energy
  signature : track + Ecal/HCal cluster measured with tracker
- Photons ($\pi^0 \rightarrow \gamma\gamma$ decays) ≈25% of jet energy
  signature : ECal cluster w/o associated track measured with ECal
- Long-lived neutral hadrons (K^0_L , neutrons) ≈10% of jet energy
  signature : Ecal/HCal clusters w/o associated track ; measured with ECal and HCal

Pflow yields better jet reconstrcution than calorimeter based approach

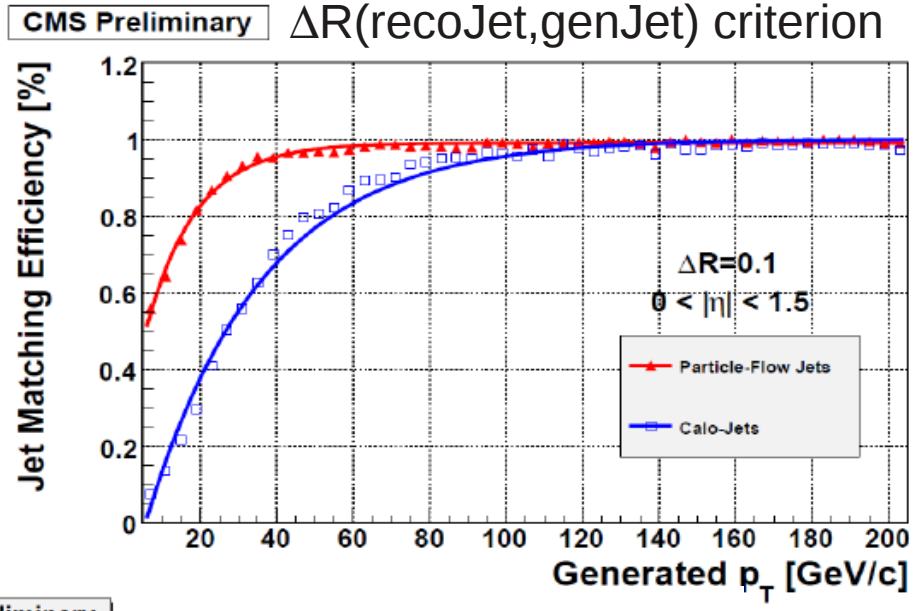
Why?

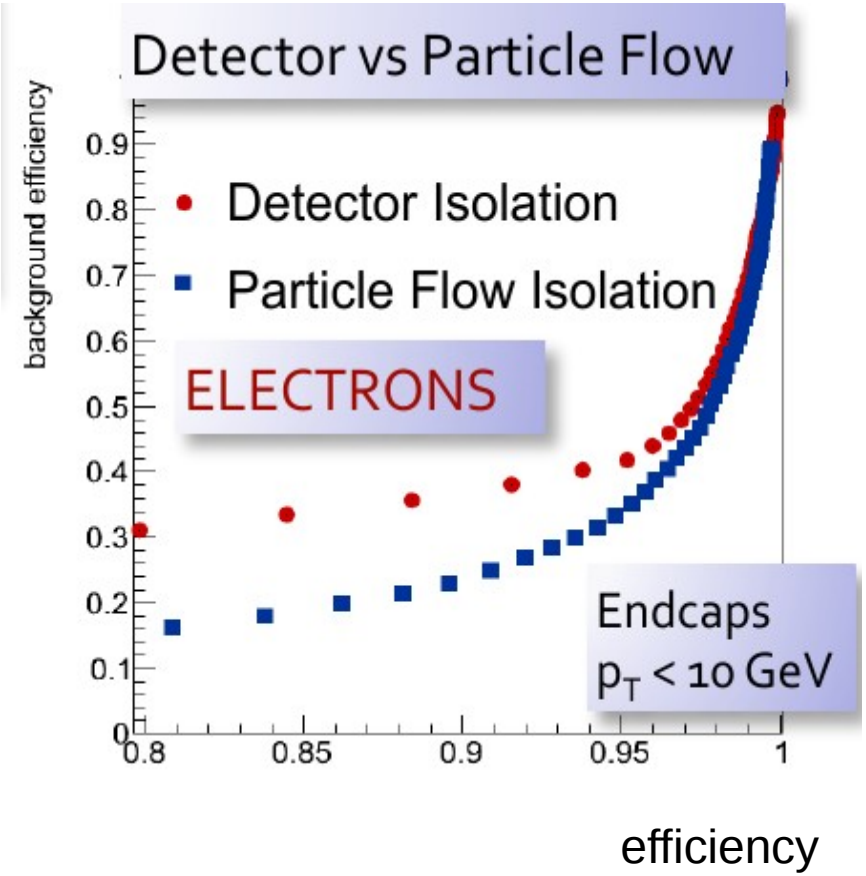
- **Reconstruction identification isolated leptons [e, μ]**
  - Electron : track matched with ECal cluster(s)
  - Muon : inner track matched with outer track / hits in muon stations, calorimeter clusters compatible with MIP signatures

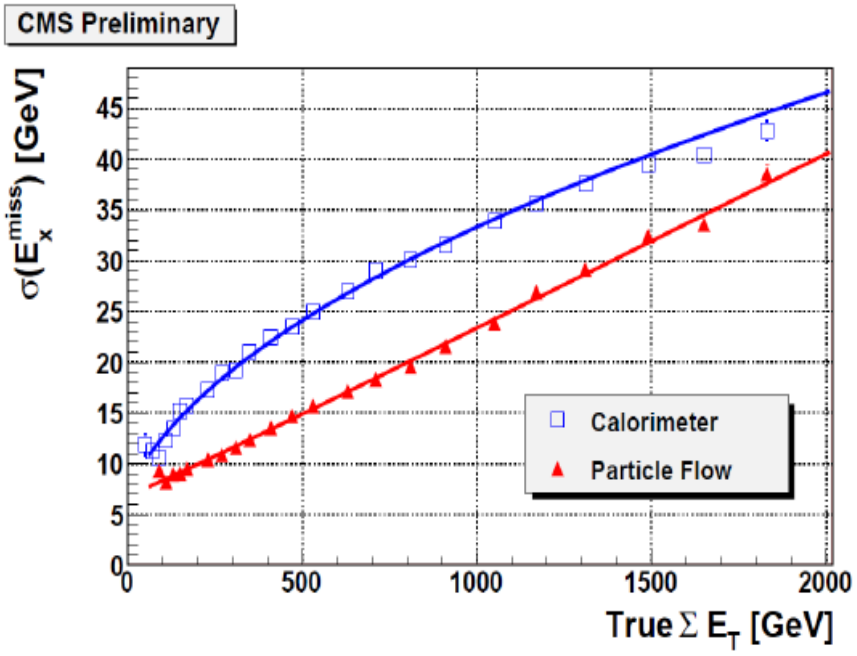
Advantages of PFlow

- more precise jet reconstruction
- more precise MET reconstruction
- more efficient lepton isolation

Matching efficiency based on ΔR(recoJet,genJet) criterion







Setup

- Connect to machine nafhh-cms011.desy.de
- Execute the following commands

  > ini cmssw

  > export SCRAM_ARCH=slc5_amd64_gcc462

- go to your working directory and create project area

  > cmsrel CMSSW_5_3_5 CMSSW

  > cd CMSSW_5_3_5/src

  > cmsenv

  - For the first exercise we will need the following package

    > cp -R /nfs/dust/test/cmsdas/schoola5/PFlowDAS ./

  - Compile the code

    > scramv1 b

Exercise 1
Analyzing PFlow collection

- Inspect EDAnalyzer
  - PFlowDAS/PFfowAnalyzer/src/PFfowAnalyzer.cc

  and corresponding python configuration script
  - PFlowDAS/PFfowAnalyzer/test/PFlowAnalyzer_cfg.py
- Note that python script is configured in such a way that code runs on simulated $Z \rightarrow ee/\mu\mu/\tau\tau$ events
- Modify PFlowAnalyzer.cc
  - access 3-momentum and PDG Id of selected particle flow objects and print out this information
  - You can use either printf() procedure or output to std::cout
- Execute cmsRun on 10 events and inspect the output of your code

Exercise 1
Analyzing PFlow collection

- When electrons are present in event CMS PFlow algorithm attempts to reconstruct superClusters, combining Ecal cluster created by electron and clusters induced by Bremstrahlung photons. Let us explore constituents of PFElectron objects
- Modify PFlowAnalyzer.cc
  - Comment out print out lines from previous exercise
  - Access super cluster and track of particle flow objects (check if they exist)
  - Compute ΔR between track and super cluster

    you can use method

    reco::deltaR(etaTrack, phiTrack, etaCluster, phiCluster)
  - Fill histogram with corresponding ΔR value
  - Compute p(track)/E(superCluster) and also $E_{ECal}/(E_{ECal} + E_{HCal})$

    (declare this histogram first in the scope of the PFlowAnalyzer::beginJob() method)
- Modify configuration file and run on 1000 Z events
- Open output RooT file and inspect created histogram

  (Use your findings for the next “ToyPF” exercise)

Exercise 1
Analyzing PFlow collection

- Develop PflowAnalyzer.cc code further and compute for each particle flow object isolation variables
  - Sum up transverse momentum of all PF candidates within <0.4 cone around current particle (excluding particle itself from this sum)
  - In the sum consider only those particle with p_T >1 GeV/c and $|\eta| < 2.5$
  - Define relative Pflow Isolation variable as a ratio of this sum and transverse momentum of the current particle
  - Fill histograms with PFlow isolation variable defined in this way separately for PF candidate identified as muons, electrons and all other particles
- Run code on 1000 MC events ($Z \rightarrow ee, \mu\mu, \tau\tau$)
- Open RooT file and inspect produced histograms (PFlow isolation)
- Explain your findings
- Select pairs of isolated opposite sign leptons
  - e^+e^- and $\mu^+\mu^-$

  and plot invariant mass of these pairs

Exercise 2
ToyPF Algorithm

- For this exercise we will need the following package

  > cd [working_dir]/CMSSW_5_3_5/src

  > cp /nfs/dust/test/cmsdas/schoola5/ToyPF ./

- Inspect the source file (EDProducer)
  - ToyPF/Configuration/plugins/ToyPF.cc
  - ToyPF/Configuration/plugins/ToyPF.h
- What kind of collection is this EDProducer supposed to produce?
- Take a look at configuration scripts
  - ToyPF/Configuration/python/ToyPF_cff.py
  - ToyPF/Configuration/test/ToyPF_cfg.py
- For this exercise we will use preselected sample of $Z \rightarrow \mu^{+}\mu^{-}$ Monte Carlo events

Exercise 2
ToyPF Algorithm

- This exercise is meant to teach you some of the basics of particle flow reconstruction by creating your own "toy" algorithm.
- Two key points now have to be addressed :
  - making links between elements

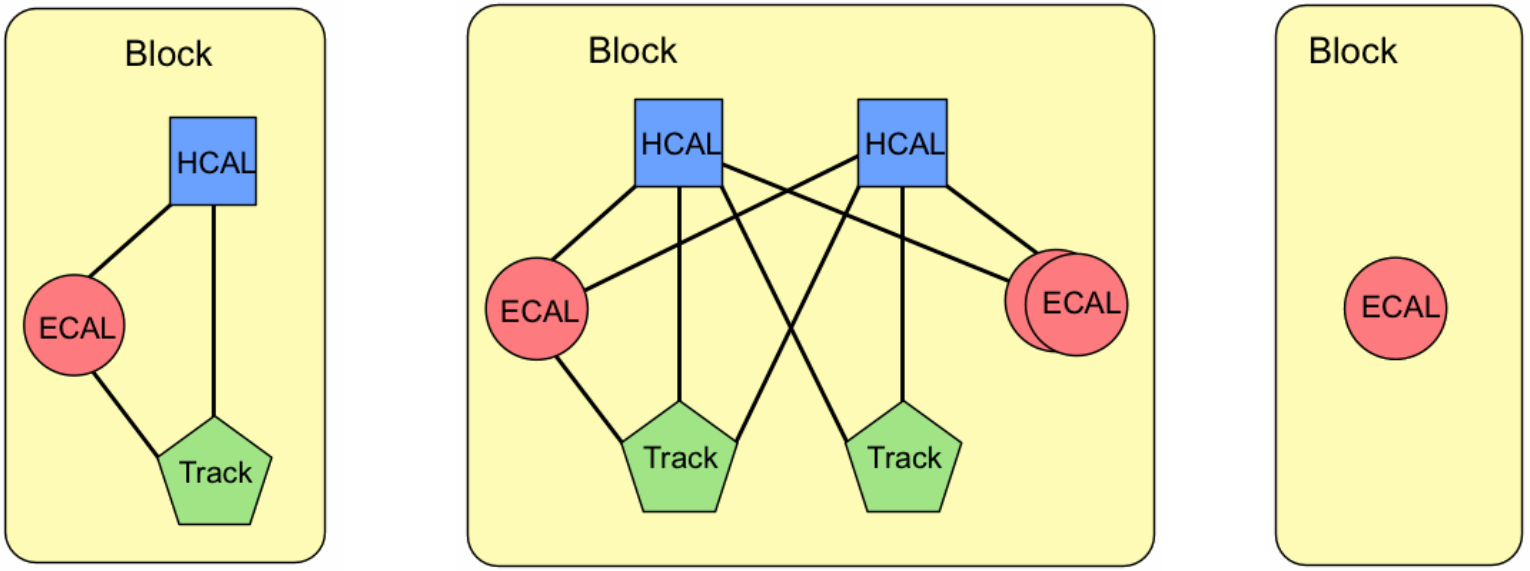
    (elements = tracks, ecal clusters, hcal clusters for our toy algorithm)
  - inferring particles from these sets of linked elements
- The EDAnalyzer code has part which prepares a list of Pflow building blocks
- This is done artificially
  - first by selecting PFCandidates with p_T > 10 GeV/c and $|\eta| < 2.5$
  - by splitting these PFCandidates into constituents
    - Tracks
    - Ecal Clusters
    - HCal Clusters
  - Matching tracks to reconstructed muons

ToyPF Algorithm : Part 1

- Part1 of deals with creation of links between elements
- You are offered to implement criterion which links
  - Tracks and ECal clusters
  - Tracks and HCal clusters
  - ECal and HCal clusters
- A hint
  - Implement linkage criterion based on “η,ϕ –proximity” for each pair of elements
  - You may use method reco::deltaR(eta1,phi1,eta2,phi2);
  - Recall previous exercise (ΔR(track-superCluster) distribution)
- It is recommended to use η and ϕ of the track at the entrance to the ECal
  - track.outerPhi()
  - track.outerEta()
- When establishing links between tracks and HCal clusters or ECal and Hcal clusters, use looser cut on ΔR

ToyPF Algorithm : Part 2

- Part2 of the ToyPF exersize offers your to make particles out of all linked elements
- To facilitate the construction of your toy algorithm, a very simplistic function which links elements together has been written for you.
- Set of linked element graphs (using the above isLinked function which you wrote in Part 1) is provided to help you to define your particles
- Example of graphs of linked elements



ToyPF Algorithm : Part 2

- Restrict yourself to consideration of 4 types of the partcicles
  - Interpret each type of block containing track either as
    - Charged pion (corresponding element of isMuon vector == false)
      - PDG ID = 211 (-211)
    - Muon (corresponding element of isMuon vector == true)
      - PDG ID = 13 (-13)
  - Block not containing track but linking ECAL and HCAL clusters
    - decide whether this block is based on ration of Ecal energy to the total energy of the ECal and HCal clusters
  - ECal cluster alone : photon (PDG ID = 22)
  - HCal cluster alone :
- When computing 4-momentum of neutral PFCandidates, assume that the reconstructed primary vertex is at (0,0,0)
- Think over what attributes you would like to assign to the reconstructed particles

Inspecting Output of ToyPF Algorithm

- Use PFlowAnalyzer.cc code to analyze collection of PFCandidates created by your ToyPF Algorithm
- Modify configuration file ToyPF/Configuration/test/ToyPF_cfg.py
  - Include PFlowAnalyzer module in your process.Path
  - Modify appropriately parameters of PFlowAnalyzer
- Implement parameters steering linking procedure (cuts on ΔR)
  - Track and ECal Cluster matching
  - Track and HCal Cluster matching
  - ECal and HCal cluster matching

  as an inputs parameters of EDProducer ToyPF
- Explore effect of this parameters on the performance of your Particle Flow algorithm
- Repeat first step of exercise1 but this time with PFCandidates provided by your ToyPF EDProducer
  - Print out energy, η and ϕ of PFCandidates provided by ToyPF

Why Pflow is Better than Calorimeter Based Approach

Perfect PFA : What theory predicts

- Jet energy resolution
  $\sigma^2(E_{jet}) = \sigma^2(ch.) + \sigma^2(\gamma) + \sigma^2(h^0) + \sigma^2(conf.)$
- Excellent tracker :
  $\sigma^2(ch.) << \sigma^2(\gamma) + \sigma^2(h^0) + \sigma^2(conf.)$
- Perfect PFA : $\sigma^2(conf.) = 0$
- $\sigma^2(E_{jet}) = A_\gamma^2 E_\gamma + A_h^2 E_{h0} = w_\gamma A_\gamma^2 E_{jet} + w_{h0} A_h^2 E_{jet}$

$$\sigma(E_{\gamma,h})/E_{\gamma,h} = A_{\gamma,h}/\sqrt{E_{\gamma,h}}$$

Typically $w_\gamma = 25\%$; $w_{h0} = 13\%$

Take typical CMS values
$\Delta E/E = A / \sqrt{E}$
ECal $\rightarrow A_\gamma$
HCal $\rightarrow A_{h0}$