Table of Contents

Particle Flow in PAT (PF2PAT)

Complete: (missing doc for the new top projection and for the isolation system)

Tutorial: WorkBookPF2PAT

Current Status (March 30)

- PF2PAT working in 3_5_6
- PF2PAT is being cleaned up, so that the PF2PAT concept can be ported to FWLite. The documentation might be slightly out-of-phase for a while

Open subjects

- neutral particles
    - vertex determination.
- muons
    - waiting for standalone and calo muons: must first be integrated to the PF algorithm (Matt is working on it : https://savannah.cern.ch/task/index.php?9788)
- electrons
    - isolation not optimal, being studied
    - cluster containment not optimal for particle-based electrons (Florian and Daniele working on it : https://savannah.cern.ch/task/index.php?14330)
- jet energy correction for particle flow jets from PF2PAT (volunteer welcome)
- b tagging
    - integrate b tagging in PF2PAT+PAT (volunteer welcome)

Introduction

The PAT (Physics Analysis Toolkit)

An essential goal of the PAT is to provide the analyst with a clean global view of the event, with no double counting of the energy between the various particles in the event. The edm::Event consists of various collections of objects, like electrons, jets, or taus. These collections are reconstructed independently, and can overlap. For example, an isolated electron will very often be reconstructed as a jet as well.

The PAT cleaning procedure for the standard reconstruction consists in:

- matching the reconstructed objects together
- decide what is the object to keep in case of overlap
- produce clean collections with no overlap

Particle flow in the PAT : PF2PAT

Particle flow can be used in the PAT in two different ways. The first way is to replace the standard reconstruction objects by the particle flow objects in the input of PAT layer 0, which will perform the standard Physics objects cleaning. This goal of this cleaning is to avoid that a given object appears in different collections. For example, an isolated electron will also give rise to a CaloJet. The cleaning removes the corresponding jet from the collection of CaloJets.

The second way is PF2PAT, which basically replaces the standard layer 0 of PAT, but is in fact much more than this.

PF2PAT starts from the collection of particles reconstructed by the particle flow.

From this input, it uses the standard reco algorithms to produce the following particle-based Physics objects:

- particle flow MET
- particle flow jets
- particle flow taus
- isolated particles
- pile-up particles
- remaining particles, which are not used in any jet, tau, and which did not give rise to an isolated or to a pile-up particle.

These objects can then be used in input of the PAT layer 1, which will convert them to PAT objects.

PF2PAT makes use of the following features of particle flow to provide clean collections of Physics objects:

- there is no double counting of the energy in the list of particles, if we assume a perfect particle flow reconstruction.
- all the particle-based Physics objects are built directly or indirectly from this list of particles.

If you do not want to replace the PAT Layer0 but add a new one for PF2PAT you can do that too (from CMSSW_3_6_X onward). Please see the example file in CVS.

In effect this keeps the standard PAT in tact and clones in for PF2PAT. To do this you have to assign a postfix in the `usePF2PAT` function call. This will be added to all the collections produced by this instance of PF2PAT.

For example the standard PAT muons will still be stored in `patMuons` and the muons found by PF2PAT will be stored in `patMuonsPOSTFIX` (assuming you chose postfix="POSTFIX" in your `usePF2PAT` function call)

Software Packages

- Producers and algorithms are located in CMS.PhysicsTools/PFCandProducer.
- This package makes use of data classes defined in DataFormats/PFCandidate.

Top projection, or avoiding double-counting

Event History

The edm::Event is an ensemble of collections, with an apparently flat structure. However, most of the objects stored in these collections keep track of the source objects used in their construction. For example:

- A PFJet keeps references to its constituents, that are the PFCandidates clustered in the jet.
- A PFTau keeps a reference to the corresponding PFJet.

This information constitutes the event history, which is made visible in a uniform way by the functions

```
unsigned             Candidate::numberOfSourceCandidatePtrs()  const;
CandidatePtr       Candidate::sourceCandidatePtr(unsigned i) const;
```

Which are overloaded in the child Candidate classes, as needed. The event history is analyzed in the so-called top projection.

Binary top projection

A binary top projection module is a producer:

- with two input collections: the top collection, and the bottom collection.
- which produces a subset of the bottom collection. An object in the bottom collection is said to be *masked* if it can be found in the history of at least one of the objects in the top collection. Unmasked objects are the only ones to be copied to the output collection.

Top projection producers are built from the template class TopProjector.

The template class is specialized in the TopProjector.cc file, which contains the top projection classes used in PF2PAT. The corresponding python configuration files can be found in the CMS.PhysicsTools/PFCandProducer/python/TopProjectors directory.

Other top projection classes can be added in the same way, in any client package.

Chained top projection

In PF2PAT, the collection of PFCandidates sent to jet clustering are the ones which:

- are not flagged as pile-up particles AND
- are not going to become a pat::Electron AND
- are not going to become a pat::Muon.

The collection of PFCandidates verifying these 3 conditions is obtained by chaining binary top projection modules. Please refer to the PF2PAT sequence.

Particle based algorithms

Particle selectors

The following particle selectors are implemented using the generic selector mechanism:

- **`PtMinPFCandidateSelector`** : select PFCandidates with a pT>pTmin
- **`PdgIdPFCandidateSelector`** : select PFCandidates from the given pdgIds

These selectors keep track of the source PFCandidate, which is necessary to preserve the event history.

Particle based MET

MET is computed by the PFMET module from a collection of PCandidates, by simply doing the vector sum of the PFCandidates transverse energy, and taking the opposite.

Particle based isolation

Particle based isolation in PF2PAT has been refurbished, and migrated to the IsoDeposit system. Documentation to be written.

- More information: Particle-based lepton isolation in PF2PAT, PF&Tau ID, Aug 24, 2009.

Pile-up candidate masking

Pile-up PFCandidates are identified in the PFPileUp module. This module reads

- a collection of PFCandidates
- a collection of reco::Vertex.

and associates a vertex to the charged PFCandidates. The association is done in two steps. First, the module tries to find a vertex that refers to the same reco::Track as the PFCandidate. If such a vertex can be found, it is associated to the charged PFCandidate. If not, the charged PFCandidate is associated to the closest reco::Vertex in z.

PFCandidates that are associated to a vertex that is not the primary vertex (which is the first vertex in the reco::Vertex collection) are considered as pile-up PFCandidates.

A PileUpPFCandidate is created for each of them, and put into the event. This object contains a reference to the vertex the PFCandidate is associated to.

Non pile-up PFCandidates do not have a reference to the primary vertex, which can anyway always be found at the beginning of the reco::Vertex collection.

Neutrals do not get flagged as PileUpPFCandidates, since there is currently no way to identify the neutral pile-up particles.

Electrons in the Particle Flow and in PF2PAT

Summary of the treatment in RECO

The electron reconstruction is based on the GSF track reconstruction. Two algorithms exist to seed the GSF tracks. The first one is the so-called ECAL-driven GSF track seeding which starts from the super-clusters in the ECAL. It is very efficient on high pT isolated electrons. In the context of the Particle Flow event reconstruction, an other seeding algorithm has been developed. It is known under the name "electron pre-identification" or more recently "tracker-driven seeding". It relies quite heavily on the track properties. The seeds found by these two algorithms are merged, and the GSF track reconstruction is run on the collection thus obtained. The Particle Flow electron reconstruction is using all the GSF track whatever their provenance. Inside the PFlow, the quality of an electron is evaluated by the "mva" variable. If mva>-0.1, the electron is kept as such. If it has mva<-0.1, the elements (track, clusters) of the electron are left free for the rest of the algorithm and will give rise to charged hadrons and photons.

Material: AN 2009-164 on the electron reconstruction in general and AN 2010-034 on the PFlow electron reconstruction.

What is the GsfElectronAlgo doing ? The GsfElectronAlgo is doing two main tasks. First, it runs the reconstruction of the ECAL-driven electrons, i.e. the reconstruction of the electron whose GSF track has been seeded by the ECAL-driven algorithm. It is followed by a loose pre-selection of the ECAL-driven electrons. Second, it collects the electrons reconstructed by the Particle Flow and adds them to the GsfElectron collection. There is no duplication of the objects, when an electron is found by the two algorithms, it makes only one GsfElectron candidate. The electrons found by only one of the two algorithms and passing the pre-selection criteria of the corresponding algorithm are saved. In other words, no electron is lost in the process. The GsfElectronAlgo does also other things: computation of sub-detector-based isolation and of shower shapes, but which are not relevant here.

All the electron PFCandidates have mva>-0.1 and are saved as GsfElectrons Electrons with mva<-0.1 are also saved in the collection. Some of them are there because they have been pre-selected by the ECAL-driven electron reconstruction, the others have been saved for commissioning purposes and have usually mva>-0.4. In any case, it is **strongly** recommended not to use them in an analysis, as they cannot be consistently used together with the collection of PFCandidates coming out of the PFAlgo.

Treatment of the electrons in PF2PAT

The selection of the PFCandidates is done in several steps. As far as the electrons are concerned, the candidates with Pt>5 GeV, isolated and passing some conversion rejection criteria are selected selected. Then, they are then turned into pat::Electrons by the PATElectronProducer in the following way: the PATElectronProducer looks for a GsfElectron corresponding to to the electron PFCandidate, on the basis of the Gsf track to create the pat::Electron. A reference to the PFCandidate is saved.
Important remark

- For versions before V08-01-04 of PAT, the `pat::Electron.pt()` method or `pat::Electron.momentum()` methods returns the momentum of the GsfElectron which should be similar to the one of the PFCandidate. It is recommended to use the PFCandidate momentum as to be consistent with the jets and MET determinations. It can be done with `pat::Electron.pfCandidateRef()->momentum()`. It is advised to do the same if the properties of the Gsf need to be accessed, for a reason explained below and even with the recent versions of PAT. The GsfTrack can be accessed with `pat::Electron.pfCandidateRef()->gsfTrackRef()`

Duplicates cleaning

It can happen, in particular in the case of an early Brem conversions, that several GSF tracks are reconstructed while there was only one electron, a cleaning is therefore applied. The cleaning of the GsfElectronAlgo and of the PFAlgo are very much similar, but in some rare cases (<1%), a different choice is done. Fortunately enough, the GsfElectronAlgo saves the list of the Gsf tracks discarded by the cleaning in a vector of ambiguous tracks. Therefore, when the PATElectronProducer does not find the electron with the same Gsf track in the GsfElectron collection, it looks into the list of the ambiguous tracks.

PF-Electron identification (in progress)

The mva>-0.1 criteria applied within the PFlow is too loose for the definition of a final state with high-pT, isolated electrons. Tighter criteria should be applied at the analysis level. Some information can be found in this talk. The identification of isolated PF-electron is very similar to the identification of the ECAL-driven electrons, and it consists in three steps

- Conversion rejection
- Electron identification
- Isolation

The conversion rejection criteria only used the properties of the Gsf track, and there is no therefore no reason to have specific criteria for PF electrons. For what concerns, the isolation, as to fully exploit the consistency of the treatment applied in the PFAlgo, it is recommended to use a particle-based isolation. As far as the electron identification is concerned, there are two main possibilities

- Using the standard electron identification criteria (i.e WP80, WP90, etc..), but not the isolation. As mentioned earlier, since a GsfElectron can be, at the same time, an ECAL-driven electron and a PF-electron; there is no problem. It should however be mentioned that the mva>-0.1 cut applied within the PFlow cannot be undone. The electron identification criteria of WP80, or of WP90 are therefore applied on top of the (loose) mva>-0.1 cut, causing an loss of 1-2%(tbc) of the electron candidates; but removing 10-20% (tbc) of additional background.
- Using a purely PF-based electron identification, i.e., apply a tighter cut on the mva. There is no recommended WP80PF or WP90PF set of identification/isolation cuts yet. It is being worked on.

External algorithms

Jet reconstruction

Standard PFJet reconstruction is used, but the reconstruction is driven by the PF2PAT configuration. This allows the user to decide which particles enter the jets. For example, one can exclude pile up particles from jet reconstruction in the following way:

```
# noPileUp is the name of a module that produces a collection
# of non--pile-up [[%PFCANDIDATE%][PFCandidates]]

include "FastSimulation/Configuration/data/FamosSequences.cff"
module kt10PFJetsNoPileUp = kt10PFJets from "RecoJets/JetProducers/data/kt10PFJets.cff"
replace kt10PFJetsNoPileUp.src = noPileUp
```

The jets in this collection will be used as an input to PAT layer 1, and mask PFCandidates used in the jets.

Tau ID

The PF2PAT jets are fed into the tandard PFTau identification, which produces PFTauDiscriminators. Then, a PFTauSelector is used to create a new collection of PFTaus containing only the PFTaus that passed the discimination.

The taus in this collection will be used as an input to PAT layer 1, and mask the PF2PAT jets they come from.

b tagging

Expected to be automatically filled when running the PAT part of PF2PAT+!PAT.

Output

The following documentation might not be up-to-date. If you see any problem, please contact Colin and Michal

It is essential that you learn how to understand the PF2PAT sequence, to be able to check by yourself what PF2PAT is doing.

Electrons

- Electron collection for PAT:
    - recoPFCandidates_pfIsolatedElectrons__PF2PAT : electrons with a pT > 5, and isolated
- Other, intermediate electron collections:
    - recoPFCandidates_pfAllElectrons__PF2PAT : all PFCandidates of type electron
    - recoPFCandidates_pfElectronsPtGt5__PF2PAT : same, with a pT > 5
- IsoDeposits, corresponding to the pfElectronsPtGt5 collection:
    - recoIsoDepositedmValueMap_isoDepElectronWithCharged__PF2PAT
    - recoIsoDepositedmValueMap_isoDepElectronWithNeutral__PF2PAT
    - recoIsoDepositedmValueMap_isoDepElectronWithPhotons__PF2PAT
- Isolation values, computed from IsoDeposits. These values correspond to pfElectronsPtGt5:
    - doubleedmValueMap_isoValElectronWithCharged__PF2PAT
    - doubleedmValueMap_isoValElectronWithNeutral__PF2PAT
    - doubleedmValueMap_isoValElectronWithPhotons__PF2PAT

Muons

- Muon collection for PAT:
    - recoPFCandidates_pfIsolatedMuons__PF2PAT : muons with a pT > 5, and isolated
- Other, intermediate muon collections:
    - recoPFCandidates_pfAllMuons__PF2PAT : all PFCandidates of type muon
    - recoPFCandidates_pfMuonsPtGt5__PF2PAT : same, with a pT > 5
- IsoDeposits, corresponding to the pfMuonsPtGt5 collection:
    - recoIsoDepositedmValueMap_isoDepMuonWithCharged__PF2PAT
    - recoIsoDepositedmValueMap_isoDepMuonWithNeutral__PF2PAT
    - recoIsoDepositedmValueMap_isoDepMuonWithPhotons__PF2PAT
- Isolation values, computed from IsoDeposits. These values correspond to pfMuonsPtGt5:
    - doubleedmValueMap_isoValMuonWithCharged__PF2PAT
    - doubleedmValueMap_isoValMuonWithNeutral__PF2PAT
    - doubleedmValueMap_isoValMuonWithPhotons__PF2PAT

Jets

- PFJets for PAT:
    - recoPFJets_pfNoTau__PF2PAT: jets not tagged as taus.
- Intermediate jet collections:
    - recoPFJets_allPfJets__PF2PAT: all jets
    - recoPFJets_pfJets__PF2PAT: jets with pT > a given threshold (selected from the previous collection).

Taus

- PFTaus for PAT:
    - recoPFTaus_allLayer0Taus__PF2PAT
- Tau by-products:
    - recoPFTauDiscriminator_allLayer0TausDiscrimination__PF2PAT
    - recoPFTaus_fixedConePFTauProducer__PF2PAT
    - recoPFTauTagInfos_pfRecoTauTagInfoProducer__PF2PAT
    - recoPileUpPFCandidates_pfPileUp__PF2PAT

MET

- PFMET for PAT:
    - recoMETs_pfMET__PF2PAT

Unmasked PFCandidates

- PFCandidates for PAT:
    - recoPFCandidates_pfNoJet__PF2PAT

Other particle objects

- For IsoDeposit creation:
    - recoPFCandidates_pfAllChargedHadrons__PF2PAT
    - recoPFCandidates_pfAllNeutralHadrons__PF2PAT
    - recoPFCandidates_pfAllPhotons__PF2PAT
- Intermediate objects, selected by top projection:
    - recoPFCandidates_pfNoElectron__PF2PAT

    - recoPFCandidates_pfNoMuon__PF2PAT
    - recoPFCandidates_pfNoPileUp__PF2PAT

Analysis Examples

VBFHtotautauPFlowTutorial

Tutorial

WorkBookPF2PAT

Links to more information

- Have a look at this presentation.