Table of Contents

PAT Exercise 07: PF2PAT Tutorial....1
Goal of this exercise....1
Setting up of the environment....1
Having a look at the PAT configuration file....2
Exercises....3
Exercise 7 a): Understand the PF2PAT sequence....3
Exercise 7 b): A small analysis of jets in Z -> μμ....4
Exercise 7 c): Run PF2PAT+PAT and standard PAT at the same time....6
Exercise 7 d): Understanding the PF2PAT muon collections....7
Exercise 7 e): Using the particle-based isolation computed in PF2PAT....8
Exercise 7 f): Decoupling particle-based isolation and apply dbeta correction in PF2PAT....9
Exercise 7 g): Investigate pile-up removal in PF2PAT....10
Additional information....11
Review status....12

PAT Exercise 07: PF2PAT Tutorial

Goal of this exercise

PF2PAT (now called PFBRECO) is a set of tools. It helps to work with particleFlow objects correctly.

In exercise 7a you will learn how to browse cmsRun configurations. As these can get quite complex, it is important to be able to track down all steps in the creation of an object, including cuts, corrections and whatever is applied. In order to browser the configs, two powerful tools are introduced: IPython and edmConfigEditor.

The other exercises will give you technical an physical understanding of PF2PAT. Your basic workflow is:

1. run, plot, check-out result
2. change something
3. goto 1.

From the physics point of view, particle isolation is studied most, as PF2PAT has a large influence on it. On-the-fly plotting with root and FWLite is used.

Note:

This web course is part of the PAT Tutorial, which takes regularly place at cern and in other places. When following the PAT Tutorial the answers of questions marked in RED should be filled into the exercise form that has been introduced at the beginning of the tutorial. Also the solutions to the Exercises should be filled into the form. The exercises are marked in three colours, indicating whether this exercise is basic (obligatory), continuative (recommended) or optional (free). The colour coding is summarized in the table below:

Color Code	Explanation
	Basic exercise, which is obligatory for the PAT Tutorial.
	Continuative exercise, which is recommended for the PAT Tutorial to deepen what has been learned.
	Optional exercise, which shows interesting applications of what has been learned.

Basic exercises ( ) are obliged and the solutions to the exercises should be filled into the exercise form during the PAT Tutorial.

Setting up of the environment

First of all connect to `lxplus` or `cmslpc` and go to some work directory. You can choose any directory, provided that you have enough space. You need ~100 MB of free disk space for this exercise. We recommend you to use your *~/scratch0* space. In case you don't have this (or do not even know what it is) check your quota typing `fs lq` and follow this link. If you don't have enough space, you may instead use the temporary space (*/tmp/your_user_name*), but be aware that this is lost once you log out of lxplus (or within something like a day). We will expect in the following that you have such a *~/scratch0* directory.

```
ssh lxplus.cern.ch
[ ... enter password ... ]
```

Create a directory and local release area for this exercise (to avoid interference with code from the other exercises) and build CMSSW.

```
cd scratch0/
mkdir exercise07
cd exercise07

# build CMSSW plus additional packages
cmsrel CMSSW_7_4_1_patch4
cd CMSSW_7_4_1_patch4/src
cmsenv
git cms-addpkg DataFormats/PatCandidates
git cms-addpkg PhysicsTools/PatAlgos
git cms-addpkg FWCore/GuiBrowsers
git cms-merge-topic -u CMS-PAT-Tutorial:CMSSW_7_1_0_patTutorial
scram b -j 9
```

Having a look at the PAT configuration file

The basic configuration file we are going to use for nearly all exercises is

```
PhysicsTools/PatAlgos/test/patTuple_PF2PAT_cfg.py
```

It will run the standard PF2PAT sequence on a standard AOD/RECO input. Having a look into this file with your favourite editor you will see (leaving out some comments and general things you should recognize from prior exercises):

```
# import skeleton process
from PhysicsTools.PatAlgos.patTemplate_cfg import *
# verbose flags for the PF2PAT modules
process.options.allowUnscheduled = cms.untracked.bool(True)
#process.Tracer = cms.Service("Tracer")

# Configure PAT to use PF2PAT instead of AOD sources
# this function will modify the PAT sequences.
from PhysicsTools.PatAlgos.tools.pfTools import *
postfix = "PFlow"
jetAlgo="AK4"
usePF2PAT(process,runPF2PAT=True, jetAlgo=jetAlgo, runOnMC=True, postfix=postfix)

# to turn on type-1 MET corrections, use the following call
#usePF2PAT(process,runPF2PAT=True, jetAlgo=jetAlgo, runOnMC=True, postfix=postfix, typeIMetCorrec

# to switch default tau (HPS) to old default tau (shrinking cone) uncomment
# the following:
# note: in current default taus are not preselected i.e. you have to apply
# selection yourself at analysis level!
#adaptPFTaus(process,"shrinkingConePFTau",postfix=postfix)

# Add PF2PAT output to the created file
from PhysicsTools.PatAlgos.patEventContent_cff import patEventContentNoCleaning
#process.load("CommonTools.ParticleFlow.PF2PAT_EventContent_cff")
#process.out.outputCommands =  cms.untracked.vstring('drop *')
process.out.outputCommands = cms.untracked.vstring('drop *',
                                                   'keep recoPFCandidates_particleFlow_*_*',
                'keep *_selectedPatJets*_*_*',
                'drop *_selectedPatJets*_caloTowers_*',
                'keep *_selectedPatElectrons*_*_*',
                'keep *_selectedPatMuons*_*_*',
                'keep *_selectedPatTaus*_*_*',
                'keep *_patMETs*_*_*',
                'keep *_selectedPatPhotons*_*_*',
                'keep *_selectedPatTaus*_*_*',
                                                   )

# top projections in PF2PAT:
```

```
getattr(process,"pfNoPileUpJME"+postfix).enable = True
getattr(process,"pfNoMuonJME"+postfix).enable = True
getattr(process,"pfNoElectronJME"+postfix).enable = True
getattr(process,"pfNoTau"+postfix).enable = False
getattr(process,"pfNoJet"+postfix).enable = True
# to use tau-cleaned jet collection uncomment the following:
#getattr(process,"pfNoTau"+postfix).enable = True

# verbose flags for the PF2PAT modules
getattr(process,"pfNoMuonJME"+postfix).verbose = False

# enable delta beta correction for muon selection in PF2PAT?
getattr(process,"pfIsolatedMuons"+postfix).doDeltaBetaCorrection = cms.bool(False)

# Temporary fix to have AK5 payloads until the AK4 payloads are ready
process.patJetCorrFactorsPFlow.payload = 'AK5PFchs'

## ------------------------------------------------------
#  In addition you usually want to change the following
#  parameters:
## ------------------------------------------------------
#
#   process.GlobalTag.globaltag =  ...    ##  (according to https://twiki.cern.ch/twiki/bin/view/
from PhysicsTools.PatAlgos.patInputFiles_cff import filesRelValProdTTbarAODSIM
process.source.fileNames = filesRelValProdTTbarAODSIM
process.maxEvents.input = 100
#   process.out.outputCommands = [ ... ]  ##  (e.g. taken from PhysicsTools/PatAlgos/python/patEv
process.out.fileName = 'patTuple_PF2PAT.root'
#   process.options.wantSummary = False   ##  (to suppress the long output at the end of the job)
```

Basically, what is done is the following:

- The standard PAT sequence which you have run several times already is loaded.
- The usePF2PAT function is used to create the PF2PAT+PAT sequence. This sequence contains:
  - the PF2PAT sequence
  - a modified PAT sequence. This sequence is obtained by cloning the standard PAT sequence, and by connecting the clone to the output of the PF2PAT sequence.
- The PF2PAT+PAT sequence is added to the path.

Note: Be aware that all your modifications to single modules need to made *after* applying the `usePF2PAT(...)` function, since it changes many parameters (e.g. input tags) on all of the pat modules.

Exercises

Exercise 7 a): Understand the PF2PAT sequence

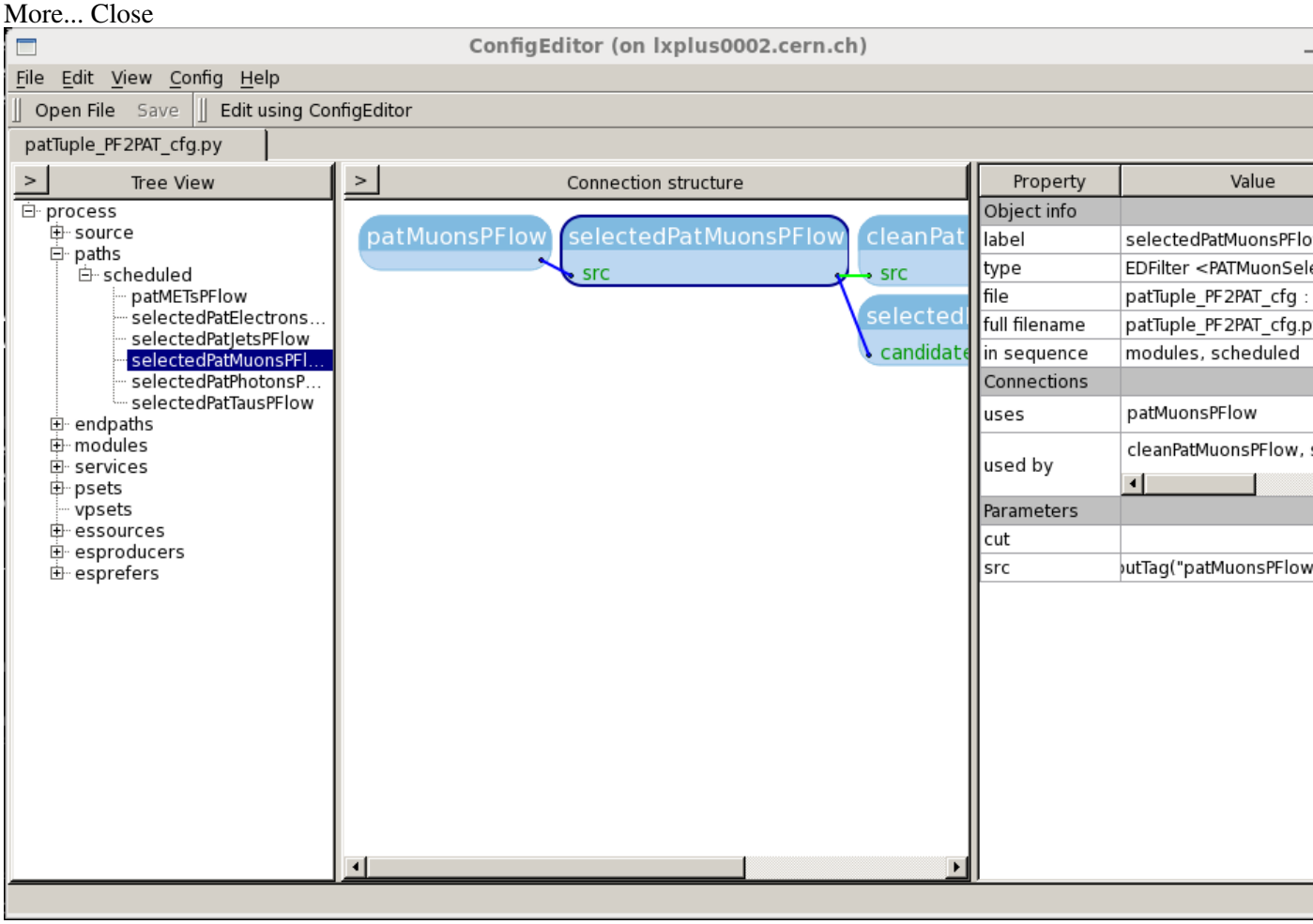
Use the edmConfigEditor to browse the config file:

```
cd PhysicsTools/PatAlgos/test
edmConfigEditorSSH patTuple_PF2PAT_cfg.py
```

In the left panel, open:

- paths (all paths)
  - scheduled
    - selectedPatMuonsPFlow (pat muon producer module configured with PF2PAT). You should select this module, then the connection structure to construct this module is appeared.

More... Close



Click on the various modules in the middle panel, and look at the parameters.

Alternatively it is possible to use the IPython shell and type an object of interest (use tab-completion!):

```
ipython patTuple_PF2PAT_cfg.py
print "Hello World"
process.pfIsolatedMuonsPFlow
```

Questions:

- (1) What are the various selections applied to obtain the pfIsolatedMuonsPFlow?
- (2) Double-click on pfAllMuonsPFlow, to navigate back. What are the selections applied to the muon PFCandidates before entering the pfMuonSequencePFlow sequence?
- (3) Navigate back as much as you can, to pfPileUpPFlow. What is the input collection of this module? Why don't we see the previous modules in the edmConfigEditor?

Exercise 7 b): A small analysis of jets in Z -> μμ

```
cd PhysicsTools/PatAlgos/test
cp patTuple_PF2PAT_cfg.py muons_PF2PAT_cfg.py
```

opening **`muons_PF2PAT_cfg.py`** in your favorite editor, find the following commented line, uncomment it, and change the path of the sample file as given below (or just append this at the end of the file, as it will overwrite any previous filename):

```
process.source.fileNames = [
 '/store/relval/CMSSW_7_1_0/RelValZmumuJets_Pt_20_300_13/GEN-SIM-RECO/PU25ns_POSTLS171_V15-v1/000
]
```

Now run the PF2PAT+PAT process on these Z mu mu events:

```
cmsRun muons_PF2PAT_cfg.py
```

A small hint by the way: to keep the shell unblocked, you can also pipe the output to a logfile and watch it with 'tail' (-f means follow, exit from tail with Crtl-C).

```
cmsRun muons_PF2PAT_cfg.py &> log_muons &
tail -f log_muons
```

Warnings of the following kind are harmless:

```
WARNING: called applyPostfix for module/sequence cleanPatTaus which is not in patDefaultSequenceP
MET: using  cms.InputTag("pfMETPFlow")
WARNING: called applyPostfix for module/sequence patCandidates which is not in patDefaultSequence
WARNING: called applyPostfix for module/sequence patCandidates which is not in patDefaultSequence
```

Open ROOT, load FWLite, and make a few plots (again, you can copy and paste the full box):

```
root
.L $CMSSW_RELEASE_BASE/src/CommonTools/ParticleFlow/test/Macros/rootlogon.C
loadFWLite()
TFile f("patTuple_PF2PAT.root")
initPAT("PAT")

new TCanvas("muons_pt","muons_pt")
Events.Draw("selectedPatMuonsPFlow.pt()")
new TCanvas("muons_iso","muons_iso")
Events.Draw("(selectedPatMuonsPFlow.chargedHadronIso() + selectedPatMuonsPFlow.photonIso() + sele
new TCanvas("jets_pt","jets_pt")
TH1D * histo = new TH1D("histo","",50,0,100)
Events.Draw("selectedPatJetsPFlow.pt()>>histo")
histo->Draw()
```

Questions:

- (1) Can you explain the shape of muon isolation distribution?
- (2) Can you explain the shape of the jet pT distribution?

Many questions become easier with higher statistics (here: maybe 300 events), because bumps get more obvious.

As you can see from here on you can use the PAT objects built during PF2PAT just like you have learned during the previous exercises. Now let us change the event hypothesis in the most basic way: only let global muons with pT > 50 GeV pass as muons. Also set the minimal isolation to 0.1. At the end of `muons_PF2PAT_cfg.py`, add the lines:

```
#global muon pt>50 GeV and isolation < 0.1
process.pfIsolatedMuonsPFlow.cut = "pt > 50 & muonRef.isAvailable() & muonRef.isGlobalMuon & muon
#new output file for new config
process.out.fileName = 'patTuple_PF2PAT_2.root'
```

Note: The selection in pfSelectedMuonsPFlow is done on a reco::PFCandidate and not on a pat::Muon. This means you have to be careful, for example, when cutting on muon quantities, to first check that the reference to the reco::Muon is available in the PFCandidate.

Run cmsRun again and repeat the plotting above.

Questions:

- (3) Can you explain the little bump that appeared in the jet pT distribution?
- (4) Can you explain the shape of the muon isolation distribution?
- (5) Conclusion: which PFCandidates of type muon end up as pat::Muons? in pat::Jets?

Now remove the last lines you have added at the end of your cfg and add this (the last parameter is already in the file, but as before, you can just overwrite it):

```
# relaxing completely the muon isolation in PF2PAT
process.pfIsolatedMuonsPFlow.cut = "pt > 50 & muonRef.isAvailable() & muonRef.isGlobalMuon"
#disable the NoMuon disambiguation
getattr(process,"pfNoMuon"+postfix).enable = False
#new output file for new config
process.out.fileName = 'patTuple_PF2PAT_3.root'
```

Questions:

- (6) Can you explain the little bump that appeared in the jet pT distribution?
- (7) Can you explain the shape of the muon isolation distribution?
- (8) Conclusion: which PFCandidates of type muon end up as pat::Muons? in pat::Jets?

Exercise 7 c): Run PF2PAT+PAT and standard PAT at the same time

```
cd PhysicsTools/PatAlgos/test
cp patTuple_PATandPF2PAT_cfg.py  muons_PATandPF2PAT_cfg.py
```

Edit **`muons_PATandPF2PAT_cfg.py`** and change the source to read the Z to mu mu RelVal sample, as explained before. If you like to see bumps more clearly, raise the number of events processed from 10 to 100. Then, run **`muons_PATandPF2PAT_cfg.py`**. A file called **`patTuple_PATandPF2PAT.root`** is produced. Check the collections stored in this file:

```
edmFileUtil -P -f file:patTuple_PATandPF2PAT.root
```

Result:

```
file:patTuple_PATandPF2PAT.root
file:patTuple_PATandPF2PAT.root (1 runs, 3 lumis, 332 events, 13649035 bytes)
Branch 0 of Events tree: EventAuxiliary Total size = 52474
Branch 1 of Events tree: EventBranchEntryInfo Total size = 9112112
Branch 2 of Events tree: EventSelections Total size = 29038
Branch 3 of Events tree: BranchListIndexes Total size = 7918
Branch 4 of Events tree: recoPFCandidates_particleFlow__RECO. Total size = 17766110
Branch 5 of Events tree: recoPFCandidates_particleFlow_AddedMuonsAndHadrons_RECO. Total size = 16
Branch 6 of Events tree: recoPFCandidates_particleFlow_CleanedCosmicsMuons_RECO. Total size = 160
...
```

All the branches for which the module name ends by PFlow are from PF2PAT+PAT . The other branches are from standard PAT. Let's compare the two pat::Muon collections using ROOT.

```
root
.L $CMSSW_RELEASE_BASE/src/CommonTools/ParticleFlow/test/Macros/rootlogon.C
loadFWLite()
TFile f("patTuple_PATandPF2PAT.root")

// Compare histograms
```

```
TCanvas c1("c1","c1")
Events.Draw("patMuons_selectedPatMuons__PAT.obj.pt()>>h1")
Events.Draw("patMuons_selectedPatMuonsPFlow__PAT.obj.pt()>>h2","","same")
h1.Draw()
h2.SetLineColor(2)
h2.Draw("same")

// Save Plot, so we can look at it after the next step
c1.SaveAs("Plot_muonpt_PATandPF2PAT.pdf")
c1.SaveAs("Plot_muonpt_PATandPF2PAT.root")
```

To understand these plots, one should note that:

- standard PAT pat::Muons are built from all reco::Muons in the event. The fake rate is rather large, and non-isolated muons are included.
- PF2PAT+PAT pat::Muons are built from isolated PFCandidates of type muon, with a pT larger than 5 GeV/c. These PFCandidates are reconstructed in the particle flow algorithm from the same reco::Muons as in the standard PAT case. The muon identification in the particle flow algorithm ensures a low fake rate, and the isolation requirement in PF2PAT removes muons in jets, for instance from heavy flavour leptonic decay. These muons are clustered in the jets together with the other particles, and don't give rise to pat::Muons. Instead, they end up in the pat::Jets.

Questions:

- (1) Did you notice any change in processing time with respect to PF2PAT+PAT alone? How can you explain it?
- (2) What is the cause for the peak at low pT in the standard pat::Muon plot?

Let us investigate the effect of the isolation cut in PF2PAT on the PF2PAT+PAT muons. Edit **`muons_PATandPF2PAT_cfg.py`** and add the following line **at the end of the file** to fully relax the muon isolation cut

```
#relaxed isolation
process.pfIsolatedMuonsPFlow.cut = "pt > 50 & muonRef.isAvailable() & muonRef.isGlobalMuon"
```

- backup Plot_muonpt_PATandPF2PAT.pdf and Plot_muonpt_PATandPF2PAT.root
- re-run PF2PAT+PAT,
- redo the plot, and compare to the previous one.

Question:

- (3) Why are there less muons in the PF2PAT+PAT than in the standard PAT sequence?

Exercise 7 d): Understanding the PF2PAT muon collections.

A small loss of muons is still visible at low pT for PF2PAT+PAT muons, with respect to standard PAT muons. The subject of the next exercise is to understand this loss.

Start edmConfigEditor on the file you have created in the previous exercise:

```
edmConfigEditorSSH muons_PATandPF2PAT_cfg.py
```

In the left panel, open:

- paths (all paths)
    - ♦ p (the p path)
        - ◊ scheduled

· selectedPatMuons
· selectedPatMuonsPFlow

Question:

- (1) Which module(s) could be the reason for the small loss observed at low muon pT in the previous exercise? Test your hypotheses by modifying the configuration of the module(s) you suspect to be responsible for the loss, before rerunning PF2PAT+PAT and redoing the plots.

It can be that the muons are lost before PF2PAT+PAT, in the particle flow algorithm. We're going to check the muon PFCandidates at the output of the particle flow algorithm, to see if some reco::Muons are already missing. To do that, we are going to use a module already present in the pfMuonSequencePFlow. We'll clone this module, and plug the clone where we need it.

```
# at the end of your cfg, clone the pfAllMuons module, and plug it to the collection of PFCandida
process.pfAllMuons2 = process.pfAllMuonsPFlow.clone()
process.pfAllMuons2.src = 'particleFlow'

# add it to your path:
process.selectedPatCandidates += process.pfAllMuons2

# keep the resulting collection in your output file:
from PhysicsTools.PatAlgos.patEventContent_cff import patEventContentNoCleaning
process.out.outputCommands.append('keep recoPFCandidates_pfAllMuons2_*_*')
```

Now, rerun the cfg.

Question:

- (2) Within ROOT, compare the pT spectrum of the muon PFCandidates you have selected to the pT spectrum of the standard PAT muons, which are directly built from the collection of reco::Muons. Is PF2PAT responsible for this small loss of muons at low pT? Where could this loss occur?

Exercise 7 e): Using the particle-based isolation computed in PF2PAT

First prepare your cfg.

```
cd PhysicsTools/PatAlgos/test
cp patTuple_PF2PAT_cfg.py isolation_PF2PAT_cfg.py
```

Edit **`isolation_PF2PAT_cfg.py`** and change the source to read the Z to mu mu RelVal sample, as explained before. Run the file.

Now let's have a look at the muon isolation. Plot the isolation values computed in the PF2PAT sequence, and stored in the pat lepton (A). Then plot the relative, combined isolation computed in the PF2PAT sequence, used to select the PF muons that will make it to pat::Muons (B).

```
root
.L $CMSSW_RELEASE_BASE/src/CommonTools/ParticleFlow/test/Macros/rootlogon.C
loadFWLite()
TFile f("patTuple_PF2PAT.root")

// (A) isolation stored in pat lepton
Events.Draw("patMuons_selectedPatMuonsPFlow__PAT.obj.chargedHadronIso()")

// (B) relative, combined isolation
new TCanvas
Events.SetAlias("mu","patMuons_selectedPatMuonsPFlow__PAT.obj")
Events.Draw("(mu.chargedHadronIso()+mu.photonIso()+mu.neutralHadronIso() ) / mu.pt()")
```

Question:

- (1) Set your canvas to log scale on the vertical axis. Where does the relative, combined isolation distribution stop? why?

The isolation value can also be recomputed on the fly, using the IsoDeposits stored in the pat lepton. Plot the sum pT of charged hadrons in a cone of 0.4 around the muon, computed from the IsoDeposits:

```
Events.Draw("patMuons_selectedPatMuonsPFlow__PAT.obj.isoDeposit(4).depositWithin(0.4)")
```

Note that the result is the same as when the pre-computed charged hadron isolation value is used.

The IsoDeposits made from particle-flow photons and neutral hadrons can be accessed in the same way, by changing the requested type of IsoDeposit (equal to 4 to access charged hadron IsoDeposits, and given as an argument of the isoDeposit function). The available types are defined in DataFormats/PatCandidates/interface/Isolation.h.

Exercises:

- plot the sum pT of neutral hadrons in a cone of 0.5 around the muon
- plot the sum pT of photons in a cone of 0.5 around the muon
- plot the sum pT of charged hadrons in a cone of 0.5 around the muon

Question

- (2) Which kind of particle (charged hadron, neutral hadron, photon) is contributing the most to the energy in the isolation cone? Why?

Exercise 7 f): Decoupling particle-based isolation and apply dbeta correction in PF2PAT

Note: What is dbeta?

Delta Beta is a correction of the isolation values of electrons and muons. Think of the isolation cone around a muon. Charged particles from vertices other than the primary vertex (let's call them 'aliens') can be in the cone and contribute to the isolation value of the muon. With Particle Flow, these pileup particles can be sorted out and just not be counted. But there are also neutral particles from pileup, how do we deal with them? Since there is no track, we cannot assign neutral particles to vertices. So as an estimate, we subtract one half neutral alien for every charged alien that we find in the cone. So to say. Have a look at the formula (we should not subtract a negative value, therefore max(0,...)): **`(selectedPatMuonsPFlow.chargedHadronIso() + max( 0, selectedPatMuonsPFlow.photonIso() + selectedPatMuonsPFlow.neutralHadronIso()-0.5*selectedPatMuonsPFlow.puChargedHadronIso()) ) / selectedPatMuonsPFlow.pt()`**

Got it? Great!

Now, first prepare your cfg:

```
cd PhysicsTools/PatAlgos/test
cp patTuple_PF2PAT_cfg.py noisolation_PF2PAT_cfg.py
```

open **`noisolation_PF2PAT_cfg.py`** in your favorite editor and add the path of the sample file as given below:

```
process.source.fileNames = [
  'file:/afs/cern.ch/sw/lcg/tmp/PAT_Tutorial_Summer14/exercise_07_data_F2178A41-D3D0-E311-82E2-00
]
```

This is data sample. So don't forget to set runOnMC=False

```
usePF2PAT(process,runPF2PAT=True, jetAlgo=jetAlgo, runOnMC=False, postfix=postfix)
```

Now replace isolated muons with all muons in PF2PAT:

```
# replace isolated muons with all muons in PF2PAT
process.patMuonsPFlow.pfMuonSource = "pfMuonsPFlow"
```

Run the file, then let's have a look at the muon isolation. Plot the relative, combined isolation computed in the PF2PAT sequence, used to select the PF muons that will make it to pat::Muons and apply dbeta correction (A). Also, plot the jet-pt distribution (B).

```
root
.L $CMSSW_RELEASE_BASE/src/CommonTools/ParticleFlow/test/Macros/rootlogon.C
loadFWLite()
TFile f("patTuple_PF2PAT.root")

// (A)
Events.Draw("(selectedPatMuonsPFlow.chargedHadronIso() + selectedPatMuonsPFlow.photonIso() + sele
Events.Draw("(selectedPatMuonsPFlow.chargedHadronIso() + max( 0, selectedPatMuonsPFlow.photonIso(
h2.SetLineColor(2)
h2.Draw()
h1.Draw("same")

// (B)
new TCanvas
Events.Draw("selectedPatJetsPFlow.pt()")
```

Questions:

- (1) Did you see the little bump that appeared in the jet pT distribution?
- (2) Can you explain the shape of the muon isolation distribution?
- (3) Conclusion: which PFCandidates of type muon end up as pat::Muons? in pat::Jets?

Exercise 7 g): Investigate pile-up removal in PF2PAT

By default, PF2PAT starts by identifiying the charged particles coming from a pile-up vertex, and masks them out so that they are not used in the remaining processing sequence, for example when:

- computing the particle-based isolation
- clustering the jets
- ...

In this exercise, we will create a second PF2PAT+PAT sequence, in which the pile-up identification and masking procedure is deactivated, so that all particles are used in the remaining processing sequence. Then, we will study the effect of the pile-up identification and masking.

```
cp patTuple_PF2PAT_cfg.py pileUp_PF2PAT_cfg.py
```

See the code snippets below. Merge them into **`pileUp_PF2PAT_cfg.py`** to:

- add the second PF2PAT+PAT sequence
- read a data sample, so some pile-up is present.
- disable pile-up removal in second sequence
- relax the combined isolation cut for both sequences

```
# reading a pile-up Data sample:
```

```
process.source.fileNames = [
 '/afs/cern.ch/sw/lcg/tmp/PAT_Tutorial_Summer14/exercise_07_data_F2178A41-D3D0-E311-82E2-00261894
]

# set runOnMC=False
usePF2PAT(process,runPF2PAT=True, jetAlgo=jetAlgo, runOnMC=False, postfix=postfix)

# adding the new PF2PAT sequence :
postfix2 = "PFlow2"
usePF2PAT(process,runPF2PAT=True, jetAlgo=jetAlgo, runOnMC=False, postfix=postfix2)

# disabling pile-up removal in the new PF2PAT+PAT sequence
getattr(process,"pfNoPileUp"+postfix2).enable = False

# Temporary fix to have AK5 payloads until the AK4 payloads are ready (do it also for PFlow2)
process.patJetCorrFactorsPFlow.payload = 'AK5PFchs'
process.patJetCorrFactorsPFlow2.payload = 'AK5PFchs'

# relaxing the muon isolation in both sequences:
getattr(process,"pfIsolatedMuons"+postfix).cut = "pt > 50 & muonRef.isAvailable() & muonRef.isGlo
getattr(process,"pfIsolatedMuons"+postfix2).cut =  "pt > 50 & muonRef.isAvailable() & muonRef.isG
```

Check your modifications:

```
edmConfigEditorSSH pileUp_PF2PAT_cfg.py
```

In particular, check that:

- the new PF2PAT+PAT sequence is there
- the pfNoPileUp module is configured correctly in each PF2PAT+PAT sequence
- the pfIsolatedMuons combinedIsolationCut is relaxed in each PF2PAT+PAT sequence

If you feel confident, run it!

Questions:

- (1) What distributions would you expect to change most dramatically?
- (2) In ROOT, investigate the effect of the pile-removal on all physics objects, by comparing the PFlow and PFlow2 collections.
- (3) Modify your cfg to keep the collection of offlinePrimaryVertices, and re-run. what is the effect of the PF2PAT pile-up removal on events containing only one primary vertex?

Note:

In case of problems don't hesitate to contact the SWGuidePAT#Support. Having successfully finished **Exercise 7** you might want to proceed to **Exercise 8** of the SWGuidePAT to learn more about how PAT supports collecting more information across objection collections, like the releation of jets to electrons directly in the jet object. For an overview you can go back to the WorkBookPATTutorial entry page.

Additional information

- SWGuidePF2PAT : PF2PAT Software Guide
- WorkBookPF2PAT : PF2PAT workbook
- Hermetic Top projection. This workflow is used to apply corrections to isolation of leptons, before clustering jets from the clean PF candidates in PFBRECO.

Review status

Show Hide

Reviewer/Editor and Date (copy from screen)	Comments
RogerWolf - 17 March 2012	Added color coding -- we still need a general adaptatino to the Question and Exercise model here.

Responsible: SalRappoccio

This topic: CMSPublic > SWGuidePATPF2PATExercise
Topic revision: r96 - 2014-07-02 - FelixHoehle