Table of Contents

4.2.6 Physics Analysis Toolkit (PAT): Glossary

Detailed Review status

Contents

A

addJetCollection:

pat::Tool to **add an arbitrary jet collection** to the pat::EventContent and PAT Workflow. The source code is located in the tools directory of the PatAlgos package. You can find a full description of the tool at Jet Tools.

B

C

pat::Candidate:

The CMSSW Event Data Model (EDM) is optimised for space consumption. The Data Formats of reconstructed objects contain only very basic information, while additional information might be added in form of extra Data Formats, which themselves might be distributed over the whole event content. This makes it difficult for the end user to access all information, which is necessary and available for analysis. The **pat:.Candidate is the common Data Format** of PAT. Each pat::Candidate is derived from a corresponding reco::Candidate including a user configurable set of extra information. This extra information might already be part of the event content (only being re-keyed) or newly created before being folded into the pat::Candidate Data Format. Examples for such information are:

- isoDeposits.
- electronId.
- b-tag information.
- Monte Carlo truth matching.
- object resolutions.
- jet energy correction factors.

Apart from the generic support of Monte Carlo truth matching and object disambiguation one of the key features of PAT is object embedding. At the moment the following reco::Candidate Data Formats are canonically supported:

reco collection type	pat collection type	selectedPatCandidate label	cleanPatCandidate label
reco::Photon	pat::Photon	selectedPatPhotons	cleanPatPhotons
reco::GsfElectron	pat::Electron	selectedPatElectrons	cleanPatElectrons
reco::Muon	pat::Muon	selectedPatMuons	cleanPatMuons
reco::Tau	pat::Tau	selectedPatTaus	cleanPatTaus
reco::Jet	pat::Jet	selectedPatJets	cleanPatJets
reco::MET	pat::MET	patMET	none

The pat::Candidtate collections are produced during the PAT Workflow. The main collections are the selectedPatCandidate and the cleanPatCandidate collections. Due to the feature of embedding the size of a pat::Candidate collection as well as the whole pat::EventContent is configurable, ranging between 6kB/evt and 36kB/evt for reasonable configurations. The persistent output of of the PAT Workflow is often referred to as pat::Tuple. To learn more about the size estimate of a pat::Tuple have a look to SWGuidePATEventSize. To learn more about the Data Formats have a look to WorkBookPATDataFormats.

---

cleanPatCandidates:

This is a common **reference to the collection labels** of a common PAT Candidate collection that contain extra information about object disambiguation. This information is fully user configurable. You can find all currently supported cleanPatCandidate collection labels under the description of the pat::Candidate. A suggested configuration exists in the cleaningLayer1 directory of the PatAlgos package. To learn more about the support of object disambiguation by PAT have a look to SWGuidePATCrossCleaning.

Cross Cleaning:

By construction energy deposits in the CMS detector may be interpreted differently depending on the analyst's view or the corresponding analysis purpose. For several analysis, which mostly aim for the interpretation of a combination of different objects it is necessary to resolve such ambiguities. **The disambiguation of overlapping objects is often referred to as object Cross Cleaning**. This name is a bit misleading as the disambiguation of objects does not necessarily mean, that elements will be erased from object collections. PAT supports object disambiguation by adding extra information about overlapping objects, which is well defined and completely user configurable. No elements are removed from the object collections in the default configuration. A typical example of overlapping objects is an electron, which may be interpreted as a photon or a jet at the same time. To learn more about the support of object disambiguation by PAT have a look to SWGuidePATCrossCleaning.

D

E

pat::Electron:

pat::Candidate corresponding to the reco::GsfElectron. You can learn more about the *pat::Electron* at WorkBookPATDataFormats#PatElectron.

Embedding:

The **Event Data Model (EDM)** is optimised for disc space consumption. High level analysis objects like electrons are reconstructed from basic reconstruction objects like super clusters or tracks that might again consist of base clusters or reconstructed hits. All higher level analysis objects consist of **smart links (pointers)** to the lower level reconstruction objects they are made up from. In addition extra information like track extras might be removed from the track object and kept in an extra data format in order to keep the track data format small. There is no other correlation between different object collections but via such smart pointers. As parts of these collections might be dropped from the event content at later steps of the event processing, these smart pointer correlations might be broken (*dangling*) and pointing nowhere. This architecture makes it extremely complicated for standard users to keep track of where the information of the high level analysis objects might be localised, as it might be distributed all over the event content, thus **complicating to drop information**, which might not be needed for the analysis. Moreover it reduces the flexibility of reducing the event size, which also has influence on the runtime performance of the analysis frame: the user might be interested in the calorimeter towers of which the jets in a certain jet collection are made up from. So he/she should keep the calorimeter tower collection in the event content, which requires a sizable amount of disc space even if the fraction of calorimeter towers of interest might be much smaller.

Embedding is the answer that PAT has to issues of this kind. Via configuration the user may **choose to embed certain object information** into the pat::Candidate during the production step. This information will be hard copied into the *pat::Candidate* and may now be dropped from the event content. In the example of the jet collection only the calorimeter towers, which are part of the jet will be kept. This process is fully transparent to the user: All member function of the *pat::Candidate* will be used the same way independent from whether the corresponding information has been embedded before or internally is still called by reference. To learn more about object embedding within PAT have a look to the WorkBookPATWorkflow or to the WorkBookPATConfiguration. To learn more about how embedding works have a look to the SWGuidePATEmbeddingExercise.

---

pat::EventContent:

Event content of a pat::Tuple as defined in the file https://github.com/cms-sw/cmssw/blob/CMSSW_5_3_X/PhysicsTools/PatAlgos/python/patEventContent_cff.py when making the event content persistent . In the default configuration all cleanPatCandidtates are written to the event. When removing pat::Cleaning from the workflow the selectedPatCandidates will be made persistent instead. To get an impression of what the size of a typical *pat::Tuple* as derived from the standard ttbar input files is have a look here. In the *patEventContent_cff.py* file also other vectors of useful event information are pre-defined. Of course the user is further free to customize the event content to his/her needs. To learn more about the customization of the event content of your personal *pat::Tuple* have a look at SWGuidePATConfigExercise. To learn more about tools how to estimate the size of your private *pat::Tuple* have a look to SWGuidePATEventSize.

F

G

H

I

J

pat::Jet:

pat::Candidate corresponding to the reco::Jet. The *pat::Jet* may carry the extra information of a *reco::CaloJet*, a *reco::PFJet* or a *reco::JPTJet*. It can carry much more information depending on the choice of the user. You can learn more about the *pat::Jet* at WorkBookPATDataFormats#PatJet.

K

L

M

Monte Carlo truth matching:

Association of generated event information to reconstructed object information for simulated events. Note that generator information (generator particles or generator particle based jets) are matched to reconstructed objects and not vice verse. The Analysis Tools packages provide sophisticated tools to cover all issues of matching. Have a look to WorkBookMCTruthMatch to learn more about these. To learn more about how PAT exploits the analysis tools of MC matching have a look to SWGuidePATMCMatching. You can find the files, that provide the default configuration of MC matching in the mcMatchLayer0 directory of the PatAlgos package.

---

pat::MET:

pat::Candidate corresponding to the reco::MET. You can learn more about the *pat::MET* at WorkBookPATDataFormats#PatMET.

---

pat::Muon:

pat::Candidate corresponding to the reco::Muon. You can learn more about the *pat::Muon* at WorkBookPATDataFormats#PatMuon.

N

O

P

PAT:

The Physics Analysis Toolkit (PAT) is a high-level analysis layer providing the Physics Analysis Groups (PAGs) with easy access to the algorithms developed by Physics Objects Groups (POGs) in the framework of the CMSSW offline software. It aims at fulfilling the needs of most CMS analyses, providing both ease-of-use for beginners and flexibility for advanced users. PAT is fully integrated into CMSSW and an integral part of any release of CMSSW. You can find more information about PAT in the WorkBookPAT and the complete documentation in the SWGuidePAT.

---

pat:

The minor letter abbreviation pat is the common namespace label for PAT. You mostly find it as namspace label for the pat::Candidates, but we also use it sometimes for more abstract expressions like pat::Tuple, pat::EventContent or pat::Workflow

---

Patification:

The act of creating a pat::Tuple or running the Pat Workflow on the fly. You can call this a true slang expression that should not be used. Nevertheless you might stumble over it from time to time.

PF2PAT:

Common abbreviation for a tool that makes pat::Candidates from particle flow objects only. Already without the use of PF2PAT you have the possibility to add jet collections made from particle flow constituents or to make use of of particle flow MET (pfMET). (Have a look to the pat::Tools to see how to do this.) But PF2PAT gives you the most consistent interface of the particle flow algorithms to PAT. To learn more about the use of PF2PAT have a look to the SWGuidePF2PAT.

---

pat::Photon:

pat::Candidate corresponding to the reco::Photon. You can learn more about the *pat::Photon* at WorkBookPATDataFormats#PatPhoton.

PU: - Pile Up

Q

R

S

selectedPatCandidates:

This is a common **reference to the collection labels** of a common PAT Candidate collection that has passed the selection step of the PAT Workflow but does not contain any extra information about object disambiguation. In the default configuration there is no selection applied to the reconstructed objects. To learn more about this phase of the PAT Workflow have a look to WorkBookPATWorkflow#SelectedCandidate.

You can find the configuration selectionLayer1 directory of the PatAlgos package. To learn more about tool to switch from the PAT default configurastion including information about object disambiguation ([cleanPatCandidates) to the *selectedPatCandidates* have a look to the SWGuidePATTools#CMS.Core_Tools.

---

switchJetCollection:

pat::Tool to **switch the default jet collection (ak5Calo)** to an arbitrary jet collection in the PAT Workflow. The source code is located in the tools directory of the PatAlgos package. You can find a full description of the tool at SWGuidePATTools#Jet_Tools.

T

pat::Tau:

pat::Candidate corresponding to the reco::PFTau. You can learn more about the *pat::Tau* at WorkBookPATDataFormats#PatTau.

---

pat::Tools:

A common expression for a large set off tools to modify and customize the PAT Workflow and/or event content. You can apply these tools just in your config file or in a more intuitive way via the edmConfigEditor. To learn more about the available *pat::Tools* have a look to SWGuidePATTools. It includes also tools to process input files from earlier versions of CMSSW.

---

pat::TriggerEvent:

In addition to the trigger::TriggerEvent PAT provides a *pat::TriggerEvent* to facilitate access to L1 and HLT trigger information. The pat::TriggerEvent is the common entry point to all trigger information in PAT. To learn more about it have a look to SWGuidePATTrigger.

---

pat::Tuple:

This is the common expression of the **Persistent Layer of PAT**, characterized by the pat::EventContent. Whenever you create an EDM file containing pat::Candidates during the workbook exercises we refer to it as a *pat::Tuple*. In real life the *pat::Tuple* is an EDM file that contains *pat::Candidates* and is taylored to the user's needs. It sustains full event provenance information and can be used with full CMSSW, FWLite, or plain *root*. As the content of a *pat::Tuple* can be very different from configuration to configuration (even down to the content of a *pat::Candidate* within a certain collection) it is not comparable to a data Tier like RECO or AOD. You should rather view it as **better replacement of a flat user ntuple**, that keeps all features of the EDM and can still be transformed into a flat ntuple at later stages of the analysis. Studies show that this is not necessary though: containing the same event information, there is hardly any performence differences in data access and runtime performance compared to a flat ntuple. Have a look here to get an idea of the current size of a *pat::Tuple* in the default configuration. It is of course not necessary to make the *pat::Tuple* persistent to work with *pat::Canddiate* collections. At your choice you can produce the *pat::Tuple* also on the fly.

U

V

W

pat::Workflow:

This is the common workflow for the creation of a *pat::Tuple*. It consists of a preparation phases followed by three further phases:

- **Phase 1**: creation of plain pat::Candidates.
- **Phase 2**: a first selection phase to create selectedPatCandidates.
- **Phase 3**: a object disambiguation phase to add object overlap information and to create cleanPatCandidates.

When adding PAT Triger information fo the *pat::Candidates* it might go through another phase. Have a look to WorkBookPATWorkflow to learn more about the details.

X

Y

Z

Review status

Show Hide

Reviewer/Editor and Date (copy from screen)	Comments
RogerWolf - 10 Mai 2010	Filled up to letter E plus P

Responsible: RogerWolf
Last reviewed by: RogerWolf - 10 Mai 2010

This topic: CMSPublic > WorkBookPATGlossary
Topic revision: r18 - 2014-07-31 - RogerWolf