Table of Contents

SpyRoot Tutorial

This twiki aims to be a resource of information and a tutorial of the SPyRoot package used by the AtlasCATSUSY group at CERN

Introduction

The aim of this page is to provide a basic introduction to SPyRoot data analysis package used by the AtlasCATSUSY group at CERN.

The code of SPyRoot is currently located under the following afs directory: **/afs/cern.ch/atlas/project/cern/susy1/SPyRoot/**

- the code is maintained using SVN

The part of the code that is common to all susy analysis is located under this directory:
/afs/cern.ch/atlas/project/cern/susy1/SPyRoot/common_susy11b/

1. For these common setup, instead of using a directory called "Cuts" we now use "Config" as default
2. Do not modify any *.py files under .../common_susy11b/ dir without being discussed beforehand with people maintaining this code (Till, D. Berge, D. Salek, C. Chavez)
3. If you want to make any modifications to the object definitions or scripts under .../common_susy11b/
    - First copy the file you want to modify to your workdir and modify this accordingly, python path is set such that will pick up any file in your workdir before looking under .../common_susy11b/

Main Selection Steeps

Preliminary steeps:

This section provides a set of instructions to be able to quickly produce basic plots using already made cat-susy ntuples.

1. Request access to **/afs/cern.ch/atlas/project/cern/susy1/** by contacting one of the AtlasCATSUSY organisers
2. login to lxplus :
   - ssh -Y lxplus.cern.ch
   - cd /afs/cern.ch/atlas/project/cern/susy1/SPyRoot/
3. Under this directory make your own copy of SPyRoot by following this steps :
   - cp -rp test_common_susy11b your-name_common_susy11b
   - for example : cp -rp test_common_susy11b Carlos_common_susy11b
   - cd your-name_common_susy11b
4. Edit setup.sh file and specify your local directory as workdir, change this line :
   - export WorkDir=/afs/cern.ch/atlas/project/cern/susy1/SPyRoot/your-name_comm
5. It is possible to define new configuration files (Cut_Files) under dir "Config"
6. Should be possible to run spyroot from command line
7. Select the configuration file , execute_&_load step0 ntuple processing (step0 = trigger_selection):
   - PyROOT> cutSet='myConfigFile'
   - PyROOT> execfile('LoadStep0.py')
8. If you want to make any modifications to the object definitions or scripts under .../common_susy11b/
   - First copy the file you want to modify to your workdir and modify this accordingly, python path is set such that will pick up any file in your workdir before looking under .../common_susy11b/

Data selection steeps:

1. Make pickled files from D3PD (in castor), this is known as step -1:
   - setupNtuples.py
   - [delete old pickle files to re-do]
   - this should have made a directory called "sampleDir/p543/ntuples/" (if not, please make it your self)

1. Split samples (at step -1) into sub-samples:
   - splitSample.py -f files cutset step sample1 sample2 ...
   - [no sample specified: all !]
   - splitSample.py -f 5 myConfigFile -1 JetTauEtmiss+
   - splitSample.py -f 10 myConfigFile -1 Wenu+
   - splitSample.py -f 10 myConfigFile -1 Zee+
   - [delete old pickle files to re-do]

Step 0:

1. Trigger selection

2. Run : submitStep.py myConfigFile 0 JetTauEtmiss+
    - this will run over all the JetTauEtmiss stream data ntuples in the batch system
3. Run : submitStep.py myConfigFile 0 Zee+
    - [overwrite castor ntuples, and pickle files]
    - [overwrite pickle files in afs group space]

1. Interactively this can be run with spyroot (after running spyroot in the command line):
    - PyROOT> cutSet='myConfigFile'
    - PyROOT> execfile('LoadStep0.py')

1. Merging of sub-samples into one can be done at any step for example:
    - mergeSamples.py [cutSet] [step] [sample] [numSamples]
    - mergeSamples.py myConfigFile 0 JetTauEtmiss00180400
    - need to figure out numSubsamples by looking at the number of subsamples that were originally split into
    - for examples by doing something like : ls sampleDir/p543/step0/zlSUSY11b/ | grep JetTauEtmiss00180400 | cut -f2 -d_| sort -n |tail -1| cut -f1 -d.

Step 1:

1. Run Step 1, object selection and general cuts:
    - Usage: submitStep.py [-q queue] cutSet step [sample1] [sample2]
    - submitStep.py [-q queue] cutSet step JetTauEtmiss00180400 Zee+
    - submitStep.py myConfigFile 1 JetTauEtmiss00180400 Zee+

1. Run mergeSamples.py in order to merge subsamples in to one sample (useful for data mainly): * mergeSamples.py myConfigFile 0 JetTauEtmiss00180400 * find numSamples by doing something like : ls sampleDir/p543/step1/zlSUSY11b/ | grep JetTauEtmiss00180400 | cut -f2 -d_| sort -n |tail -1| cut -f1 -d.

Step 2 :

1. Run Step 2, Final selection (signal region definition) and plotting (histogram definition):
    - submitStep.py myConfigFile 2 JetTauEtmiss00180400

Set Samples weight :

1. In order to set the correct luminosity to the MC (base on the data luminosity), a script sets a weight to each event (for data ==1)
    - setSampleWeight.py [-s] [-v] [cutset] [step] [sample1] [sample2]

Check samples :

1. Run checkSamples.py to check for missing subsamples and lost events in data and MC for each selection step:
    - checkSamples.py -s 0 myConfigFile JetTauEtmiss00180400
    - checkSamples.py -s 1 myConfigFile JetTauEtmiss00180400

- ♦ checkSamples.py -s 2 myConfigFile JetTauEtmiss00180400

Some info on our specific ntuples:

The ntuples that are loaded after step2 have had:

- GRL, MET cleaning cuts, Vtx cuts, and require exactly 1 selected isolated lepton with pt>10GeV to be in the event (after overlap removal etc...).
- In Step2 we apply some 'virtual' cuts which are applied and feature in the cut statistics - but events get into the ntuple even if they fail these.

these can then be applied by selecting on the bits:

- LeptonCut, JetNCut, JetPt1Cut, JetNCut2 , JetPt2Cut, METCut

details of the config/cuts can be seen in these configuration files for -lepton analysis (used in step0,1,2):

- /afs/cern.ch/atlas/project/cern/susy1/SPyRoot/zl_common_susy11b/Config/zlS
- /afs/cern.ch/atlas/project/cern/susy1/SPyRoot/zl_common_susy11b/Config/*

for cuts specific to el and mu channels in 1-lepton analysis:

- /afs/cern.ch/atlas/project/cern/susy1/SPyRoot/susy11b_prod/Cuts/elSUSY11b.
- /afs/cern.ch/atlas/project/cern/susy1/SPyRoot/susy11b_prod/Cuts/muSUSY11b.
- [the above files might need slight modification to be run with the common setup]

For monojet analysis :

- /afs/cern.ch/atlas/project/cern/susy1/SPyRoot/monojet_common_susy11b/Confi

Now you are ready to analyse cat-susy ntuples

1. Get a data sample handler object :
   - ♦ PyROOT> a=Data.Samples['JetTauEtmiss00180400']
2. Print efficiencies from all algorithms:
   - ♦ PyROOT> a.PrintCutEff()
3. Print efficiency from algorithms that contain the word 'Cut' in their name :
   - ♦ PyROOT> a.PrintCutEff(cutReq='Cut')

(usually, the last few lines from this printout show the same efficiency, this is due to virtual cuts)

tips and tricks:

1. you can get help by doing :
    - PyROOT> help(a.PrintCutEff)

which gives Help on method PrintCutEff in module Sample:

PrintCutEff(self, cutNames=[], UseEventWeight=True, applyAllPreviousCuts=True, cutReq='') method of Sample.Sample instance

---

or you can do a.[TAB_completion] which gives you the possible methods of object a:

1. PyROOT> a.[TAB_completion]

==>

a.AddDirectory	a.Files	a.Lumi	
a.AddFiles	a.GetCutEff	a.NormalizeHist	
a.AddFilesToChain	a.GetCutEffErr	a.Print	a.
a.AddFriendSample	a.GetCutEffNumbers	a.PrintCutEff	
a.AddStatsAfterAlgo	a.GetEntries	a.Project	
a.AddStatsBeforeAlgo	a.GetExpectedEvents	a.RegisterStatsAlgo	a.__n
a.Chain	a.GetLumi	a.RunNumber	
a.ChangeNtupleDir	a.GetStatistics	a.Scan	
a.CloneTree	a.GetTTreeCutEff	a.Statistics	
a.Copy	a.GetTTreeCutEffErr	a.UnsubscribeStatsAlgo	
a.Dir	a.GetTTreeCutEffNumbers	a.Weight	
a.Draw	a.GetWeight	a.XSection	
a.Draw2D	a.GetWeightedEntries	a._Sample__pathToPickleFile	
a.Draw2D_UB	a.GetXSection	a.__baseStr__	
a.Draw3D	a.Leaves	a.__class__	
a.Draw3D_UB	a.LiveTime	a.__delattr__	

It does not work If you tab from an object that you get from a list, example this will not work :

- Data.Samples['JetTauEtmiss00180400'].[TAB_completion]

You can get a TTree (TChain) by doing :

- a.Chain
    - (doesnt work for combined samples):

So you can do (for example):

- PyROOT> a.Chain.Draw("myEl_p_T")

or:

- PyROOT> a.Chain.Scan("myEl_p_T:myEl_eta:myEl_phi","abs(myEl_eta)<1.0")

*********	********	*************	**********	*********
Row	Instance	myEl_p_T	myEl_eta	myEl_phi
********	********	*************	***********	********
3	0	10566.992	-0.329333	-0.235800
10	0	11648.953	-0.548596	0.7006354
11	0	18206.357	-0.608094	1.3962770
13	0	13826.779	0.7104663	0.4839378
...	...	...	...	...

to find out what variables are available you can do:

- PyROOT> a.Leaves()

Variables in ntuple
AllMeff
AllMeffLep
EventNumber
FourMeff
FourMeffLep
...

from the same command you will also get group of variable names (mu_staco_E , mu_staco_allautthor , ... are also variables):

* mu_staco_ * : E allauthor author barrelSectors bestMatch charge d0_exPV endcapSectors energyLossErr energyLossPar eta etcone20 etcone30 etcone40 hastrack isCombinedMuon isLowPtReconstructedMuon isStandAloneMuon m matchchi2 matchndof n nBLHits nBLSharedHits nCSCEtaHits nCSCEtaHoles nCSCPhiHits nCSCPhiHoles nGangedPixels nMDTBEEHits nMDTBIHits nMDTBIS78Hits nMDTBMHits nMDTBOHits nMDTEEHits nMDTEIHits nMDTEMHits nMDTEOHits nMDTHits nMDTHoles nOutliersOnTrack nPixHits nPixHoles nPixSharedHits nRPCEtaHits nRPCEtaHoles nRPCLayer1EtaHits nRPCLayer1PhiHits nRPCLayer2EtaHits nRPCLayer2PhiHits nRPCLayer3EtaHits nRPCLayer3PhiHits nRPCPhiHits nRPCPhiHoles nSCTHits nSCTHoles nSCTSharedHits nTGCEtaHits nTGCEtaHoles nTGCLayer1EtaHits nTGCLayer1PhiHits nTGCLayer2EtaHits nTGCLayer2PhiHits nTGCLayer3EtaHits nTGCLayer3PhiHits nTGCLayer4EtaHits nTGCLayer4PhiHits nTGCPhiHits nTGCPhiHoles nTRTHighTHits nTRTHighTOutliers nTRTHits nTRTOutliers nucone20 nucone30 nucone40 phi phi_exPV pt ptcone20 ptcone30 ptcone40 px py pz qoverp_exPV theta_exPV trackd0 trackfitchi2 trackfitndof trackphi trackqoverp tracktheta trackz0 z0_exPV

This tries to print vector quantities on 1 line (as shown above for the staco muons) by splitting on the first _

You can also draw from SPyRoot itself (ie. without going through the TChain). this is good when you have weighted events, and also for combined samples). the method is:

- PyROOT> help(a.Draw)
    - Draw(self, expression, cut='1', opts='', bins=100, min=0, max=1000, HistName='Hist', Norm=1000.0, maxLumi=-1, DoDraw=True, title='', UseEventWeight=True, defaultSystematic=0.0, systematics={}, DontAllowNegativeBins=False, UseSampleWeight=True) method of

Sample.Sample instance
- ◊ The above is samples's Draw method to draw any TTree variable of this sample.
- ◊ maxLumi: if you want to plot only a sub-sample of the dataset, use this option. Note that if maxLumi > 0, then Norm is set to be -1 (ie. histogram is NOT normalized)
  - · maxLumi = -1 : run over full sample, apply normalization Norm
  - · maxLumi > 0 : run over sub-sample, do not normalize histogram

For example you can do:

1. PyROOT>hist=a.Draw("myEl_p_T/1000.","1","",100,0,100.,Norm=-1)

For data samples the sample knows its lumi. which you can get by doing :

1. PyROOT> a.GetLumi()
   - ♦ 16021.464999999998 *this is in microbarn (so divide by 1e6 for picobarn).

Combined Samples :

you can combine together different samples to make a combined sample (eg. All data runs into the full data set, or J0,J1,J2,J3,J4,J5,J6 MC pythia jet samples into a total jet background).

you can get this combined samples by doing :

1. PyROOT> AddCombinedSampleReg(CombinedSampleName='combData', SampleExp='____', Tags='data')
   - ♦ here you add together all samples with the tag 'data' (which all data samples have).

or alternatively you can do:

1. PyROOT> listOfSamples=['JetTauEtmiss00180400','JetTauEtmiss00180481','JetTauEtmiss
2. PyROOT> AddCombinedSample('combData',Data, listOfSamples)

for data the combined sample have all the events in the sub samples and also the combine lumi of these samples. but this is more useful for MC samples where you can combine samples with different weights, by doing for example:

1. PyROOT> AddCombinedSample('jets',Data,['J0','J1','J2','J3','J4','J5','J6'])

then you can make plots for the combines sample jets like:

1. PyROOT> hist = Data.Samples['jets'].Draw("myJet_p_T/1000.","1","",100,0,100,Norm=1)
   - ♦ here the plots are normalized to 1pb-1 of data (thats what Norm=1 means!).

another nice thing you can use is the SampleHandlers compare function:

1. PyROOT> hists = Data.Compare(['jets','J0','J1','J2','J3','J4','J5','J6'],"myJet_p_T[0]/100
   - so this makes the same plot for each of the given samples and Draws them on top of each other.

The ntuples contain vectors for selected objects:

ntuple object
myEl_
myMu_
myJet_
myMET_

For the El, Mu and Jets these contain some of the basic 4-vector information and the index of the object in the default list (also stored in the ntuple) - this index is called eg. myMu_idx.

- PyROOT> ds['MuonswBeam00159041'].Chain.Scan("myMu_p_T:myMu_eta:mu_staco_pt[myMu_id

********	*********	*************	************	********	*********
Row	Instance	myMu_p_T	myMu_eta	mu_staco_	mu_staco_
********	*********	**************	***********	***********	******
0	0	10686.618	0.0948067	10686.618	0.0948067
1	0	11085.493	0.1847464	11085.493	0.1847464
2	0	10201.325	-0.826803	10201.325	-0.826803

The above is printing the pt and eta of selected muons in 2 ways but getting the same values.

Major updates:

-- CarlosChavez - 6-June-2011

%RESPONSIBLE% CarlosChavezBarajas
%REVIEW% **Never reviewed**

This topic: Main > SpyRootTutorial
Topic revision: r7 - 2011-07-13 - CarlosChavezBarajas