Table of Contents

2018 CMS Data Analysis School Z->tautau Exercise

Complete:

Contacts

* Senior Facilitator: Abdollah Mohammadi (Kansas State U.), Isobel Ojalvo (Wisconsin-Madison U.), Indara Mayeli Suarez Silva (UCSB), Andrew melo (Vanderbilt).

Motivation

The goal of this exercise is to compute the cross section production of the Z to tautau in proton-proton collisions at 13 TeV centre-of-mass energy. The data are collected with CMS at LHC.

What we plan to do is to select Z-->tautau-->muTau events in both, data (single muon dataset) and MC (DYJets as signal and QCD, WJet and TTbar, and possibly diboson as backgrounds)

Then we will subtract all background contributions from data. The difference between data and the sum of all background contributions would be considered as ZTT contribution as the Higgs to ditau decay is negligible.

All background contributions are estimated from MC, except QCD, which is computed from data using muon-tau same-sign control region. The steps of the this exercise are:

1. Select events with a pair ParticleFlow muon and tau leptons
2. Fill the needed histogram: visible mass of muon and tau weighting the MC events according to the integrated Luminosity and MC generation
3. Extract the cross-section production of the Z to tautau
4. Compare the measured cross-section production with the theory prediction and other measurements

Documentation

Exercise script: AN and PAS from Run I

Tau Id Recommendations from Tau POG(Physics Object Group): TauIDRecommendation13TeV

Z->tautau Exercise in "23" lines:

* **Data**: integrated luminosity of 12.9 fb-1 collected in proton-proton collisions at 13 TeV in 2016 with CMS

* **Signatures**: final state with muon, hadronic tau and missing transverse momentum

* **Signal**: Drell Yan decaying into two taus, one tau decays into a muon and neutrinos, the second one decays into hadrons and neutrino

* **Backgrounds**:

- reducible with leptons from b/c quarks and/or W decays:
    - ttbar
    - diboson production WW, WZ, ZZ (small)
    - single top tW (very small, neglected)

- reducible with fake leptons
    - W+jets, QCD

* **Event selection**:

- well reconstructed, isolated and with opposite sign muon-tau pair
- muon-tau pair coming from the primary vertex

* **Analysis**:

- visible muon-tau pair mass
- background estimation from data and MC

* **Results**:

- standard model cross-section x branching fraction
- comparison with cross-section measured from the dimuon channel
- comparison with theory
- comparison with cross-section measured at 7 and 8 TeV centre-of-mass energy

* **Summary talk**:

- present, in a 15 minutes talk plus 5 minutes questions, your analysis in a simple, clear and easy to take home manner

Pre-requisites

- Basic knowledge of C++ and CMSSW
- Knowledge of ROOT and ability to write, compile and execute macros
- Short Exercises suggested:

1. Muons at link: here
2. Taus at link here
3. Pileup and MET: here
4. Statistics at link: here

Notes

The color scheme of the Exercise is as follows:

- Commands will be embedded in a grey box, e.g.

```
cmsRun testRun_cfg.py
```

- Output and screen printouts will be embedded in a green box, e.g.

```
OUTPUT
```

- Code to paste will be embedded in a pink box, e.g.

```
PASTE THIS CODE
```

- Formatted descriptions will be embedded in a blue box e.g.

```
FORMATTED DESCRIPTION OR EXPLANATION
```

- At each step of the Event Selection in this exercise it is expected that you code the selection of the muons, taus and of the dimuon veto, respectively.

Preparation

Login on cmslpc-sl6

In case you overlooked the first pre-exercise

Show Hide

```
go to
http://computing.fnal.gov/authentication/krb5conf/

and download the corresponding Fermilab Kerberos Configuration File for
Linux, OSX or Windows

the do:
sudo cp krb5.conf /etc/krb5.conf


then in:
~/.ssh/config

add the lines:
GSSAPIAuthentication yes
GSSAPIDelegateCredentials yes

then initialise:
kinit -f username@FNAL.GOV

as always: check, type in your shell:
klist

you should see something like:
Credentials cache: API:92B37754-DE6A-4B63-AF2E-6F717C63F95B
        Principal: perieanu@FNAL.GOV

  Issued                Expires               Principal
Jan  8 10:28:44 2016  Jan  9 12:28:42 2016  krbtgt/FNAL.GOV@FNAL.GOV

then this should work:
       SL6: ssh -K username@cmslpc-sl6.fnal.gov
```

Setup CMSSW at LPC

Setup a base CMSSW_7_4_5 environment on a LPC machine:

```
ssh -K yourloginname cmslpc-sl6.fnal.gov
source /cvmfs/cms.cern.ch/cmsset_default.sh

mkdir TauLongExercise
cd TauLongExercise

cmsrel CMSSW_7_4_5
cd CMSSW_7_4_5/src
cmsenv
```

Download code

In this area please run the command

```
wget https://raw.githubusercontent.com/abdollah110/Tau-CMSDAS/master/TreeReader.h
```

The output of this command is

Show Hide

```
--2015-12-23 16:02:58--  https://raw.githubusercontent.com/abdollah110/Tau-CMSDAS/master/TreeRead
Resolving raw.githubusercontent.com... 23.235.44.133
Connecting to raw.githubusercontent.com|23.235.44.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 28442 (28K) [text/plain]
Saving to:  TreeReader.h

100%[======================================>] 28,442      --.-K/s   in 0s

2015-12-23 16:02:59 (325 MB/s) -  TreeReader.h  saved [28442/28442]
```

Now run the command

```
wget https://raw.githubusercontent.com/abdollah110/Tau-CMSDAS/master/Make.sh
wget https://raw.githubusercontent.com/abdollah110/Tau-CMSDAS/master/WeightCalculator.h
chmod 700 Make.sh
```

Get Necessary Root Files

if you are using csh

```
xrdfsls ls -u /store/user/abdollah/CMSDAS2018
```

The output should resemble

```
root://131.225.204.161:1094//store/user/abdollah/CMSDAS2018/DYJetsToLL_M-50_Inc.root
root://131.225.204.161:1094//store/user/abdollah/CMSDAS2018/SingleElectron.root
root://131.225.204.161:1094//store/user/abdollah/CMSDAS2018/SingleMuon.root
root://131.225.204.161:1094//store/user/abdollah/CMSDAS2018/TTbar.root
root://131.225.204.161:1094//store/user/abdollah/CMSDAS2018/WJetsToLNu_Inc.root
root://131.225.204.161:1094//store/user/abdollah/CMSDAS2018/WW.root
root://131.225.204.161:1094//store/user/abdollah/CMSDAS2018/WZ.root
root://131.225.204.161:1094//store/user/abdollah/CMSDAS2018/ZZ.root
```

These file have been copied to the /afs area as well (in case you are running on lxplus):

/afs/cern.ch/work/a/abdollah/ROOTHadd

These files are called "ggNtuples", since they are flat trees that come from running MiniAOD with `ggNtuplizer`. They are now centrally produced and available. The code that produces the ggNtuples is here. This exercise could be done from MiniAOD ntuples, but would take much longer to run over the full statistics.

Sub-Exercise 1: Event Selection

The general process (selecting Z->TauTau->muTau_h events) is as follows:

1. Find a good muon in event (apply Pt, Eta, muon Identification, Muon Isolation and impact parameter cut)
2. Find a good tau in the event (apply Pt, Eta, decay Mode Finding, Isolation, Anti-Muon discriminator and Anti-Electron discriminator)
3. Check distance (dR) between muon and tau to greater than 0.5.
4. Make sure the Z->mumu process is not contaminating the mu-tau yield. Reject events with extra muons.

Step 1: Getting started

Create a new file named `ZTT_XSection.cc` and paste in the file

Show Hide

```
////////////////////////////////////////////////////////////////////////////////////////////
//   Compiling the code:   ./Make.sh ZTT_XSection.cc
//   Running the code:     ./ZTT_XSection.exe OutPut.root   Input.root
////////////////////////////////////////////////////////////////////////////////////////////
#include "TreeReader.h"
#include "WeightCalculator.h"
#include < string> //remove space for successful compilation
#include < ostream> //remove space for successful compilation

int main(int argc, char** argv) {
    using namespace std;

    std::string out = *(argv + 1);

    cout << "\n\n\n OUTPUT NAME IS:    " << out << endl;     //PRINTING THE OUTPUT FILE NAME
    TFile *fout = TFile::Open(out.c_str(), "RECREATE");

    std::string input = *(argv + 2);
    cout << "\n\n\n INPUT NAME IS:    " << input << endl;     //PRINTING THE INPUT FILE NAME
    TFile * myFile = TFile::Open(input.c_str());
    TH1F * HistoTot = (TH1F*) myFile->Get("hcount");



    //add the histrograms of muon and tau visible mass (both for opposite sign and same sign pair
        TH1F *    visibleMassOS = new TH1F ("visibleMassOS","visibleMassOS", 30, 0, 300);
        TH1F *    visibleMassSS = new TH1F ("visibleMassSS","visibleMassSS", 30, 0, 300);


    TTree *Run_Tree = (TTree*) myFile->Get("EventTree");
    cout.setf(ios::fixed, ios::floatfield);


    //end of analysis code, close and write histograms/file
    fout->cd();
    visibleMassOS->Write();
    visibleMassSS->Write();
    visibleMassOSRelaxedTauIso->Write();
    visibleMassSSRelaxedTauIso->Write();
    fout->Close();

}
```

The usage of cout.setf is described here:

Show Hide

```
cout.setf(): sets stream format flags
ios::fixed -  writes floating point values in fixed-point notation: number of decimals is fixed b
3.14

ios::scientific - writes floating point values in scientific notation: one digit before   the dec
and as many decimal digits as the precision field, ending with an exponential part e plus 3 expon
3.14e+000

ios::floatfield = ios::scientific | ios::fixed
```

After `cout.setf(ios::fixed, ios::floatfield);` add the following lines to access the branches you want. The names added here should match those found in `TreeReader.h`.

Show Hide

```
//########################################   General Info
      Run_Tree->SetBranchAddress("isData", &isData);
      Run_Tree->SetBranchAddress("run", &run);
      Run_Tree->SetBranchAddress("lumis", &lumis);
      Run_Tree->SetBranchAddress("event", &event);
      Run_Tree->SetBranchAddress("genWeight",&genWeight);
      Run_Tree->SetBranchAddress("HLTEleMuX", &HLTEleMuX);
      Run_Tree->SetBranchAddress("puTrue", &puTrue);
      Run_Tree->SetBranchAddress("nVtx",&nVtx);

      //########################################   MC Info
      Run_Tree->SetBranchAddress("nMC", &nMC);
      Run_Tree->SetBranchAddress("mcPID", &mcPID);
      Run_Tree->SetBranchAddress("mcStatus", &mcStatus);
      Run_Tree->SetBranchAddress("mcPt", &mcPt );
      Run_Tree->SetBranchAddress("mcEta", &mcEta );
      Run_Tree->SetBranchAddress("mcPhi", &mcPhi );
      Run_Tree->SetBranchAddress("mcE", &mcE );
      Run_Tree->SetBranchAddress("mcMass", &mcMass );
      Run_Tree->SetBranchAddress("mcMomPID", &mcMomPID );
      Run_Tree->SetBranchAddress("mcGMomPID", &mcGMomPID );

      //########################################   Tau Info
      Run_Tree->SetBranchAddress("nTau", &nTau);
      Run_Tree->SetBranchAddress("tauPt"  ,&tauPt);
      Run_Tree->SetBranchAddress("tauEta"  ,&tauEta);
      Run_Tree->SetBranchAddress("tauPhi"  ,&tauPhi);
      Run_Tree->SetBranchAddress("tauMass"  ,&tauMass);
      Run_Tree->SetBranchAddress("tauCharge"  ,&tauCharge);

      Run_Tree->SetBranchAddress("taupfTausDiscriminationByDecayModeFinding", &taupfTausDiscri

      Run_Tree->SetBranchAddress("tauByTightMuonRejection3", &tauByTightMuonRRejection3);
      Run_Tree->SetBranchAddress("tauByLooseMuonRejection3", &tauByLooseMuonRRejection3);

      Run_Tree->SetBranchAddress("tauByMVA6TightElectronRejection"  ,&tauByMVA6TightEElectronRe
      Run_Tree->SetBranchAddress("tauByMVA6MediumElectronRejection"  ,&tauByMVA6MediummElectron
      Run_Tree->SetBranchAddress("tauByMVA6LooseElectronRejection", &tauByMVA6LooseEElectronRej

      Run_Tree->SetBranchAddress("tauDxy",&tauDxy);
      Run_Tree->SetBranchAddress("tauDecayMode",&tauDecayMode);

      Run_Tree->SetBranchAddress("tauByLooseIsolationMVArun2v1DBoldDMwLT",&tauByLooseIsolaation
      Run_Tree->SetBranchAddress("tauByVLooseIsolationMVArun2v1DBoldDMwLT",&tauByVLooseIsollati
      Run_Tree->SetBranchAddress("tauByTightIsolationMVArun2v1DBoldDMwLT",&tauByTightIsolaation

      //########################################   Mu Info
      Run_Tree->SetBranchAddress("nMu", &nMu);
      Run_Tree->SetBranchAddress("muPt"  ,&muPt);
      Run_Tree->SetBranchAddress("muEta"  ,&muEta);
      Run_Tree->SetBranchAddress("muPhi"  ,&muPhi);
      Run_Tree->SetBranchAddress("muIsoTrk", &muIsoTrk);
      Run_Tree->SetBranchAddress("muCharge",&muCharge);
      Run_Tree->SetBranchAddress("muIDbit",&muIDbit);//NEW
      Run_Tree->SetBranchAddress("muPFChIso", &muPFChIso);
      Run_Tree->SetBranchAddress("muPFPhoIso", &muPFPhoIso);
      Run_Tree->SetBranchAddress("muPFNeuIso", &muPFNeuIso);
      Run_Tree->SetBranchAddress("muPFPUIso", &muPFPUIso);
      Run_Tree->SetBranchAddress("muD0",&muD0);
      Run_Tree->SetBranchAddress("muDz",&muDz);

      //########################################   Ele Info
      Run_Tree->SetBranchAddress("nEle", &nEle);
      Run_Tree->SetBranchAddress("elePt"  ,&elePt);
      Run_Tree->SetBranchAddress("eleEta"  ,&eleEta);
```

```
    Run_Tree->SetBranchAddress("elePhi"  ,&elePhi);
    Run_Tree->SetBranchAddress("elePFChIso", &elePFChIso);
    Run_Tree->SetBranchAddress("eleIDMVA", &eleIDMVA);//NEW
    Run_Tree->SetBranchAddress("eleCharge",&eleCharge);
    Run_Tree->SetBranchAddress("eleSCEta",&eleSCEta);
    Run_Tree->SetBranchAddress("elePFChIso", &elePFChIso);
    Run_Tree->SetBranchAddress("elePFPhoIso", &elePFPhoIso);
    Run_Tree->SetBranchAddress("elePFNeuIso", &elePFNeuIso);
    Run_Tree->SetBranchAddress("elePFPUIso", &elePFPUIso);
    Run_Tree->SetBranchAddress("eleD0",&eleD0);
    Run_Tree->SetBranchAddress("eleDz",&eleDz);
    Run_Tree->SetBranchAddress("eleMissHits", &eleMissHits);
    Run_Tree->SetBranchAddress("eleConvVeto", &eleConvVeto);
    Run_Tree->SetBranchAddress("eleSCEta", &eleSCEta );

    //########################################   Jet Info
    Run_Tree->SetBranchAddress("nJet",&nJet);
    Run_Tree->SetBranchAddress("jetPt",&jetPt);
    Run_Tree->SetBranchAddress("jetEta",&jetEta);
    Run_Tree->SetBranchAddress("jetPhi",&jetPhi);
    Run_Tree->SetBranchAddress("jetEn",&jetEn);
    Run_Tree->SetBranchAddress("jetCSV2BJetTags",&jetCSV2BJetTagss);
    Run_Tree->SetBranchAddress("jetPFLooseId",&jetPFLooseId);
    Run_Tree->SetBranchAddress("jetPUID",&jetPUID);
    Run_Tree->SetBranchAddress("jetRawPt",&jetRawPt);
    Run_Tree->SetBranchAddress("jetJECUnc",&jetJECUnc);
    Run_Tree->SetBranchAddress("jetRawEn",&jetRawEn);
    Run_Tree->SetBranchAddress("jetHadFlvr",&jetHadFlvr);

    //########################################   MET Info
    Run_Tree->SetBranchAddress("pfMET",&pfMET);
    Run_Tree->SetBranchAddress("pfMETPhi",&pfMETPhi);
    Run_Tree->SetBranchAddress("metFilters",&metFilters);
    Run_Tree->SetBranchAddress("genHT",&genHT);


float MuMass= 0.10565837;
float eleMass= 0.000511;
```

In order to look at all events in the file, a loop over all events must be added. Paste the following lines after initializing your branches and declaring the muon and electron masses.

Show Hide

```
Int_t nentries_wtn = (Int_t) Run_Tree->GetEntries();

cout<<"nentries_wtn====" << nentries_wtn << "\n";

for ( Int_t i = 0; i < nentries_wtn; i++) {

    Run_Tree->GetEntry(i);

    if (i % 1000 == 0) fprintf(stdout, "\r  Processed events: %8d of %8d ", i, nentries_wtn);

    fflush(stdout);

    TLorentzVector Mu4Momentum, Tau4Momentum;

   //////////////////////////////////////////////
   //Important Analysis Loop Will Happen Here!!!//
   //////////////////////////////////////////////


} //End Processing all entries
```

Event Weights

2 different weights are needed to be computed for MC when hostograms are filled. The weight for data is 1. So when computing these weights we need to make sure that weights are comouted for MC and NOT data: Here how you can get the isData from Tree:

```
        Run_Tree->SetBranchAddress("isData", &isData);
```

Lumi Weight

We need to rescale the MC events to the luminosity and cross-section. In order to do this, we need to know the total number of events of the MC sample. This weight is computed in a dedicated header file called: `WeightCalculator.h`. Take a look at this file to get an idea how this weight is computed.

```
    float LumiWeight = 1;
    if (HistoTot) LumiWeight = weightCalc(HistoTot, input);
    cout << "LumiWeight is " << LumiWeight << "\n";
```

Here is how one can compute Pileup Weight:

```
//pileup distribution for Data
wget  https://github.com/abdollah110/Tau-CMSDAS/blob/master/MyDataPileupHistogram2016.root
//pileup distribution for MC
wget https://github.com/abdollah110/Tau-CMSDAS/blob/master/mcMoriondPU.root
```

Before making loop over events:

```
    TFile * PUData= new TFile("MyDataPileupHistogram2016.root");
    TH1F * HistoPUData= (TH1F *) PUData->Get("pileup");
HistoPUData->Scale(1.0/HistoPUData->Integral());


    TFile * PUMC= new TFile("mcMoriondPU.root");
    TH1F * HistoPUMC= (TH1F *) PUMC->Get("pileup");
HistoPUMC->Scale(1.0/HistoPUMC->Integral());
```

It is better to get these root files from lxplus: Here are the root files in the lxplus

/afs/cern.ch/user/a/abdollah/public/dataMoriondPU.root

/afs/cern.ch/user/a/abdollah/public/mcMoriondPU.root

and then inside the loop:

```
                float PUWeight = 1;
 if (!isData){
                int puNUmmc=int(puTrue->at(0)*10);
                int puNUmdata=int(puTrue->at(0)*10);
                float PUMC_=HistoPUMC->GetBinContent(puNUmmc+1);
                float PUData_=HistoPUData->GetBinContent(puNUmdata+1);
                PUWeight= PUData_/PUMC_;
}
```

Note:

Here how you can get the isData, puTrue from Tree:

```
        Run_Tree->SetBranchAddress("puTrue", &puTrue);
```

Select a pair of muon and tau leptons

Within the *Important Analysis Loop* we want to select a pair of muon and tau. We want events with a well identified muon. First we make a loop over muons and taus for our *Important Analysis Loop*.

```
        for  (int imu=0 ; imu < nMu; imu++){
                 for  (int itau=0 ; itau < nTau; itau++){

                } // End of tau loop
        }  // End of muon loop
```

Step 2: Muon Selection

We will start by (1) selecting events that passed the single muon trigger, (2) looping over muons in that event and (3) selecting a *good* , *isolated* muon. with

```
 muPt->at(imu) > 30  && fabs(muEta->at(imu)) < 2.1
```

To select a single muon trigger, it can be applied

```
bool PassTrigger = (HLTEleMuX >> 19 & 1) == 1; //       else if (name.find("HLT_IsoMu24_v") != st
if (! PassTrigger) continue;
```

The trigger is selected in this manner, the muon trigger is here link.

The usage of bit operators is described here:

Show Hide

```
right shift operator ">>" shifts a bit pattern to the right
left shift operator "<<" shifts a bit pattern to the left

x >> y: returns x with the bits shifted to the right by y places
x = 00010111 (decimal +23)
x >> 2
x = 00000101 (decimal +5)

x << y:   returns x with the bits shifted to the left by y places
x = 00010111 (decimal +23)
x << 2
x = 01011100 (decimal +92)

x & y: does a "bitwise and". Each bit of the output is 1 if the corresponding bit of x AND of y i
    0110 (decimal 6)
AND 1101 (decimal 13)
  = 0100 (decimal 4)

x | y: does a "bitwise or". Each bit of the output is 1   if at least one   of the corresponding
    0110 (decimal 6)
OR  1101 (decimal 13)
  = 1111 (decimal 15)

bit operators order:
x >> y & 1 = ( x >> y) & 1

== logical equal
|| logical or
&& logical and
```

Apply muon Isolation as following:

```
float IsoMu=muPFChIso->at(imu)/muPt->at(imu);
if ( (muPFNeuIso->at(imu) + muPFPhoIso->at(imu) - 0.5* muPFPUIso->at(imu) )  > 0.0)
IsoMu= ( muPFChIso->at(imu)/muPt->at(imu) + muPFNeuIso->at(imu) + muPFPhoIso->at(imu) - 0.5* muPF
```

The medium working point of the muon Id can be applied as the following

```
(muIDbit->at(imu) >> 2 & 1) // 2 is tight
```

More info about muon ID working pint can be found here: https://github.com/cmkuo/ggAnalysis/blob/master/ggNtuplizer/plugins/ggNtuplizer_muons.cc#L166-L171

Step 3: Tau Selection

We also need to select good taus: The recommended pt cut for good taus starts at 20 GeV

```
Tau Pt > 30 GeV
```

and we want the electrons to have a good eta range as discussed in the short exercise

```
Tau |Eta| < 2.3
```

Tau should pass the following discriminators: =taupfTausDiscriminationByDecayModeFinding->at(itau) > 0.5 , tauByTightMuonRejection3->at(itau) > 0, and tauByMVA6LooseElectronRejection->at(itau) > 0, tauByTightIsolationMVArun2v1DBoldDMwLT->at(itau) > 0. From the short exercise, you found the fake rate and the tau efficiency from the medium working point.

To reject W events, require the Transverse Mass between muon and MET to be less than 40 GeV. It can be calculated by

```
float MuMetTranverseMass= TMass_F(muPt->at(imu), muPt->at(imu)*cos(muPhi->at(imu)),muPt->at(imu)*
```

We also need to check the charge of muon-Tau pair:

```
// Check charge of the muon and Taus

bool  OS = muCharge->at(imu) * tauCharge->at(itau) < 0;

bool  SS = muCharge->at(imu) * tauCharge->at(itau) > 0;
```

Step 4: Event Veto

We also need to make sure that there are no events with more than 1 good identified and isolated muons in the events. Z->mumu in particular will contaminate our phase space so we will veto events with 2 good muons. Why do we want to do this?

Show Hide

To reject Z->mumu events. One muon would be the muon we tagged, any extra jet in the event could fake a tau. The invariant mass between the muon and fake-tau would not reconstruct the Z mass, so we should reject these events.

```
        ////////////////////////////////////////////////////////////////////////////////////
        // Loop over Di-Mu events  We need to veto these events later
        ////////////////////////////////////////////////////////////////////////////////////
        bool IsthereDiMuon= false;

        for  (int imu=0 ; imu <nMu; imu++){
            for  (int jmu=0 ; jmu  <nMu; jmu++){
```

```
            // Select first good muon
            bool MuPtCut1 = muPt->at(imu) > 30 && fabs(muEta->at(imu))  <2.1 ;
            float IsoMu1=muPFChIso->at(imu)/muPt->at(imu);
            if ( (muPFNeuIso->at(imu) + muPFPhoIso->at(imu) - 0.5* muPFPUIso->at(imu) )  > 0.
                IsoMu1= ( muPFChIso->at(imu)/muPt->at(imu) + muPFNeuIso->at(imu) + muPFPhoIso
            bool MuIdIso1=((muIDbit->at(imu) >> 0 & 1)  && IsoMu1  <0.30 && fabs(muD0->at(imu


            // Select second good muon
            bool MuPtCut2 = muPt->at(jmu) > 15 && fabs(muEta->at(jmu))  <2.4 ;
            float IsoMu2=muPFChIso->at(jmu)/muPt->at(jmu);
            if ( (muPFNeuIso->at(jmu) + muPFPhoIso->at(jmu) - 0.5* muPFPUIso->at(jmu) )  > 0.
                IsoMu2= ( muPFChIso->at(jmu)/muPt->at(jmu) + muPFNeuIso->at(jmu) + muPFPhoIso
            bool MuIdIso2=((muIDbit->at(jmu) >> 0 & 1)  && IsoMu2  <0.30 && fabs(muD0->at(jmu


            bool  OS = muCharge->at(imu) * muCharge->at(jmu)  <0;

            if(MuIdIso1 && MuIdIso2 && OS)
                IsthereDiMuon=true;

        }
    }
```

Why do we ask for a different eta range for the muons in the event veto here?

Step 5: Veto Events including B-Jets

In order to supress the ttbar background, we veto events in case they have a jet tagged as b-jet.

```
vector  EveBJetPt;
                EveBJetPt.clear();

                for (int ijet= 0 ; ijet < nJet ; ijet++){

                    Jet4Momentum.SetPtEtaPhiE(jetPt->at(ijet),jetEta->at(ijet),jetPhi->at(ije
                    if (jetPt->at(ijet) > 20 && fabs(jetEta->at(ijet)) < 2.5 && Jet4Momentum.
                        EveBJetPt.push_back(jetPt->at(ijet));
                    }

                }
```

Now you can check the size of the EveBJetPt vector to apply b-jet Veto.

Step 5: Fill the visible mass of the muon and tau lepton pair

Now we need to fill two histograms-- one for Same Sign (SS) muTau and one for Opposite Sign (OS) muTau.

Please consider what contributes to the SS muon-Tau pair yield?

Show Hide
Tau Fakes, QCD, and Charge Mis-Identification of OS muon-tau pairs.

```
//Check if there is an OS  muTau pair with dR > 0.5 and TMass(mu.MET) < 40 and then fill the weig
visibleMassOS->SetDefaultSumw2();
visibleMassOS->Fill(Z4Momentum.M(),LumiWeight*PUWeight);
```

```
//Check if there is a SS  muTau pair with dR > 0.5 and TMass(mu.MET) < 40 and then fill the weigh
visibleMassSS->SetDefaultSumw2();
visibleMassSS->Fill(Z4Momentum.M(),LumiWeight*PUWeight);
```

Step 6: Compile and run your code

Sometimes compilation and running can take up the bulk of your time. But this should be easy! Now your code is all set up. You can

```
./Make.sh ZTT_XSection.cc
```

If there are no errors in the output and then you can run your compiled code on each root file we have prepared for you.

```
 ./ZTT_XSection.exe DYJetsToLL.root   root://cmseos.fnal.gov://store/user/abdollah/CMSDAS2018/DYJ

 ./ZTT_XSection.exe TTJets.root   root://cmseos.fnal.gov://store/user/abdollah/CMSDAS2018/TTbar.r

 ./ZTT_XSection.exe WJetsToLNu.root   root://cmseos.fnal.gov://store/user/abdollah/CMSDAS2018/WJe

 ./ZTT_XSection.exe SingleMu.root   root://cmseos.fnal.gov://store/user/abdollah/CMSDAS2018/Singl
```

Step 7: QCD Background Calculation

All backgrounds are estimated from MC. The only background estimated from data is QCD. It is estimated from Data using SameSign control region. There is a correction scale factor of 1.06 which is the OS/SS ratio. Can you propose a method to extract this value based one the current available codes?

Show Hide
ABCD METHOD Assign a phase space with 2 uncorrelated variables (isolation, and charge of the particles in this case). Split into 4 regions

1. Isolated muon and OS (signal region)
2. Isolated muon and SS
3. Anti-Isolated muon and OS
4. Anti-Isolated muon and SS

One can visualise the ABCD method as a matrix.

isolated	A	B
anti-isolated	C	D
	OS	SS

We take our shape/yield from a SS Isolated region. We need to know how to get from this (B) to our isolated OS signal (A).

```
 A/B=C/D so signal  yield A =B(C/D)
```

Essentially we see how the SS->OS yield changes in a different phase space, i.e. relaxed isolation.

Step 8: Make Invariant Mass Plot

Once you have all the root files prepared:

```
wget https://raw.githubusercontent.com/abdollah110/Tau-CMSDAS/master/xs_calculator_prefit_New.py
python xs_calculator_prefit.py
```

The `xs_calculator_prefit.py` script has an optional input of scaling the DY yield, running the above commands should produce the following output.

```
[user@cmslpc36 src]$ python xs_calculator_prefit.py
Usage:python xs_calculator_prefit.py DYCrossSection[optional]
loaded
Observation: XXX
ZTT (unscaled) Expected: XXX
TT Expected: XXX
W Expected: XXX
QCD Expected: XXX
Info in : pdf file xs.pdf has been created
DY->ll xs used: XXX pb
```

Open and look at the plot `xs.pdf`.

Parallel: Select a pair of electron and tau leptons

You may want to make your new file `ZTT_XSection_ETau.cc`. Say we wanted to select a Z->TauTau->eletronTau pair instead. This would be called a different *"channel."* Selecting an electron-Tau (eleTau or eTau) pair should be very similar to selecting the MuTau pair. The differences are:

1. Instead of a single muon trigger ask for a single electron trigger
2. Loop over electrons instead of muons
3. Use different isolation requirements. An electron isn't a muon
4. Di-electron veto (instead of diMuon veto).
5. Loose Electron rejection->Tight Electron Rejection
6. Tight Muon rejection -> Loose Muon Rejection

Within the *Important Analysis Loop* we want to select an electron and tau pair. We want events with a well identified electron. First we make a loop over electrons and taus for our *Important Analysis Loop*.

```
        for  (int iele=0 ; iele < nEle; iele++){
                 for  (int itau=0 ; itau < nTau; itau++){

                } // End of Tau loop
        }  // End of Electron Loop
```

Step 2b: Single Electron Trigger

bool PassTrigger = (HLTEleMuX >> 0 & 1) `= 1; //  if (name.find("HLT_Ele25_eta2p1_WPTight_Gsf_v")` string::npos) bitEleMuX = 0; if (! PassTrigger) continue;

Step 3b: Electron Isolation

Application of Electron ID is much different than muon ID. When doing analysis usually an official working point is suggested by the EGamma POG.

```
                    float IsoEle=elePFChIso->at(iele)/elePt->at(iele);
                    if ( (elePFNeuIso->at(iele) + elePFPhoIso->at(iele) - 0.5* elePFPUIso->at(iel
                    IsoEle= (elePFChIso->at(iele)/elePt->at(iele) + elePFNeuIso->at(iele) + elePF

                bool eleMVAId= false;
                if (fabs (eleSCEta->at(jele)) <= 0.8 && eleIDMVA->at(jele) >   0.941  ) eleMVAId=
                else if (fabs (eleSCEta->at(jele)) >  0.8 &&fabs (eleSCEta->at(jele)) <=  1.5 &&
                else if ( fabs (eleSCEta->at(jele)) >=  1.5 && eleIDMVA->at(jele) >  0.758  ) ele
                else eleMVAId= false;
```

Step 4b: Di-Electron Veto

We also need to make sure that there are no events with more than 1 good identified and isolated electron. Z->ee in particular will contaminate our phase space so we will veto events with 2 good electron. Why do we want to do this?

Show Hide
To reject Z->ee events. One muon would be the muon we tagged, any extra jet in the event could fake a tau. The invariant mass between the electron and fake-tau would not reconstruct the Z mass, so we should reject these events.

```
        for  (int iele=0 ; iele < nEle; iele++){
                 for  (int jele=0 ; jele < nEle; jele++){

//Both electrons should pass the following requirements:
  //Electron Pt > 15
  //Electron Eta < 2.4
  //eleIDMVA (as mentioned above)
  //IsoEle1 < 0.30
  //fabs(eleD0->at(iele)) < 0.045 && fabs(eleDz->at(iele)) < 0.2
//and they should have opposite charge:
  //eleCharge->at(iele) * eleCharge->at(jele) < 0;

                }
        }
```

Now one can go fill the Invariant Mass for the ETau pair à la Step 5.

Sub-Exercise 2: Compute cross section

Step 1: Cross Section Review

Cross-section for different samples are mentioned in the following google doc

SAMPLE NAME	CROSS-SECTION
DYJetsToLL	XXX
WJets	XXX
TTJets	XXX

Cross Section Formula

The goal of this exercise is to compare the computed ZTT cross-section with theory value. We will compute the ZTT cross-section by using this formula:

```
ZTT_XS= (Num_Data - Num_BG)/Lumi*Acceptance*Efficiency
```

The plots made already in the previous sections are all normalized to lumi, Acc*Eff. Assuming that Data matches MC Simulation, acceptance and efficiency don't need to be recalculated for this exercise.

So all we need is to subtract the background from Data and compare ZTT cross-section with theory value already considered when filling the histogram (i.e. 5765).

ZTT Cross-Section Scaling Factor

One can find the ZTT Cross-section using the following formula:

Within some visible mass region, e.g (25, 125)

```
Data_OST - xDY_OST - MC_None_DY_OST - QCD_OST = 0
```

```
1) X = DY_XS/5765
```

```
2) QCD_OST = 1.06(Data_SST - X * DY_SST - MC_None_DY_SST)
```

```
=> X = (Data_OST - MC_None_DY_OST - 1.06(Data_SST - MC_None_DY_SST))/(DY_OST - 1.06DY_SST)
```

```
=> DY_XS = X* 5765
```

`X` is our ZTT cross section scaling factor. In a Higgs->muTau analysis one would scale the ZTT MC simulation yield by this number `X`.

Splitting the ZLL from Ztautau

int numTau=0;

for (int igen=0;igen < nMC; igen++){

if (fabs(mcPID->at(igen)) ==15 && mcMomPID->at(igen)==23) numTau++;

}

// if (numTau <1) continue; // uncomment this line to get Ztautau contribution of DY

// if (numTau >1) continue; // uncomment this line to get Zll contribution of DY

```
 ./ZTT_XSection.exe DYJetsToTauTau.root   root://cmseos.fnal.gov://store/user/abdollah/CMSDAS2018
```

```
 ./ZTT_XSection.exe DYJetsToLL.root   root://cmseos.fnal.gov://store/user/abdollah/CMSDAS2018/DYJ
```

Step 2: Create Datacard

We want to include the proper uncertainties in finding `X`.

In the TauAnalysis directory do

```
export SCRAM_ARCH=slc6_amd64_gcc481 (bash) or  setnev SCRAM_ARCH slc6_amd64_gcc481 (tcsh)

scram project CMSSW CMSSW_7_4_7

cd CMSSW_7_4_7/src

cmsenv

git clone https://github.com/cms-analysis/HiggsAnalysis-CombinedLimit.git HiggsAnalysis/CombinedL

//Check the recommended tag on link above, a tag >= v5.0.2 is sufficient

cd $CMSSW_BASE/src/HiggsAnalysis/CombinedLimit

git fetch origin

git checkout v6.3.1

scramv1 b clean; scramv1 b # always make a clean build, as scram doesn't always see updates to sr
```

```
git clone https://github.com/cms-analysis/CombineHarvester.git CombineHarvester

scram b -j 8

cd CombineHarvester/CombineTools/bin

ls #these are the default datacard creators to make a datacard from a root file
```

This will set up an area for you to create and analyze "datacards". A final *datacard* is what you use to get expected limits, calculate significance, etc. The software guide for HiggsCombine is found at SWGuideHiggsAnalysisCombinedLimit.

In the `src` folder create a file datacard.txt

```
imax 1 number of bins
jmax 3 number of processes minus 1
kmax 7 number of nuisance parameters
------------------------------------------------------------------------------------------------
shapes *    ch1  FAKE
------------------------------------------------------------------------------------------------
bin          ch1
observation  9000.0
------------------------------------------------------------------------------------------------
bin                                ch1        ch1        ch1        ch1
process                            ZTT        TT         W          QCD
process                            0          1          2          3
rate                              5000.2000  200.6000   2400.4000  1500.3000
------------------------------------------------------------------------------------------------
CMS_eff_m               lnN        1.02       1.02       1.02       -
CMS_trigEff_m           lnN        1.02       1.02       1.02       -
CMS_eff_t               lnN        1.05        1.05        1.05        -
QCD_method              lnN        -          -          -          1.2
lumi_13TeV              lnN        1.025       1.025       1.025       -
theory_13TeV_TT         lnN        -          1.05        -           -
theory_13TeV_W          lnN        -          -          1.1        -
```

Step 3: Get from Datacard

These are estimated yields for Z-> MuTau. So please edit the `observation`, `rate`, and `yields` for your results. One edited it can be run

```
combine -M MaxLikelihoodFit datacard.txt --justFit
```

At the end of the output there should be something similar to

```
 --- MaxLikelihoodFit ---
Best fit r: 0.98145  -0.130604/+0.138899  (68% CL)
nll S+B -> -702.995  nll B -> -1
Done in 0.00 min (cpu), 0.00 min (real)
```

The `Best fit r` value, in the above example `0.98145` , tells you how much your signal (process 0) needs to be scaled for your computed cross section. This value `Best fit r` is `X` above. The uncertainty on the `X` is = -0.130604/+0.138899=.

Similarly for the eTau channel the uncertainty for the electron is about ~2%. How do you propose to edit the datacard for this?

Step 4: Combine Datacards

If you have two channels in two datacards, say `MuTau.txt` (a.k.a. datacard.txt) and `ETau.txt`. They can be combined

```
combineCards.py muTau_inclusive=MuTau.txt eTau_inclusive=ETau.txt > CombinedDatacard.txt
```

where eTau_inclusive will be the bin name in one and muTau inclusive wil be the bin name for the other. Any systematic uncertainty with the same name in the two datacards will be 100% correlated. When you open CombinedDatacard.txt it should resemble

```
Combination of muTau_inclusive=MuTau.txt  eTau_inclusive=ETau.txt
imax 2 number of bins
jmax 3 number of processes minus 1
kmax 7 number of nuisance parameters
----------------------------------------------------------------------------------------------------
shapes *                eTau_inclusive   FAKE
shapes *                muTau_inclusive  FAKE
----------------------------------------------------------------------------------------------------
bin          muTau_inclusive  eTau_inclusive
observation  8742.0           6618.0
----------------------------------------------------------------------------------------------------
bin                                 muTau_inclusive  muTau_inclusive  muTau_inclusive  muTau_inclu
process                             ZTT              TT               W                QCD
process                             0                1                2                3
rate                                4872.5300        281.5700         2367.4600        1442.7800
----------------------------------------------------------------------------------------------------
CMS_eff_e               lnN         -                -                -                -
CMS_eff_m               lnN         1.02             1.02             1.02             -
CMS_eff_t               lnN         1.05             1.05             1.05             -
QCD_method              lnN         -                -                -                1.2
lumi_13TeV              lnN         1.025             1.025             1.025             -
theory_13TeV_TT         lnN         -                1.05             -                -
theory_13TeV_W          lnN         -                -                1.10             -
```

The final Cross-Section can be found

```
combine -M MaxLikelihoodFit CombinedDatacard.txt --justFit
```

old commands below

Show Hide
Here how one can checkout the script to compute the cross-section and make the visible mass plots:

```
wget https://raw.githubusercontent.com/abdollah110/Tau-CMSDAS/master/xs_calculator.py
```

And here is how to run the script:

```
 python xs_calculator.py
```

Discussion and Conclusions

here is the link to Money plot

*

*

*

https://github.com/abdollah110/Tau-CMSDAS/blob/master/TauDAS.zip

*

*

*

Review Talk

Sample Presentations

In CMS presenting results is extremely important. While we work on our analyses, we present work in progress inside our "working meetings." When the analysis becomes more mature, we are required to give a "pre-approval" presentation at the appropriate Physics Analysis Group meeting (SUSY, EXO, Higgs, ...). This presentation is the beginning of the review process. During the review process the analysis and analysis documentation are reviewed by the PAG & POG conveners, Analysis Review Committee, other members of analysis group. After the review process, we present the "Approval" presentation and then the analysis will go through the collaboration wide review procedure before the results can be made public (paper publication or conference presentation). The pre-approval and approval presentations are internal to CMS. Therefor, they tend to me more technical and have more details about the analysis strategy. However, we results are presented outside of CMS it is good to avoid CMS-jargon and focus on the big picture.

When writing a presentation, it is important that you answer the What? Why? and How?

- What are you trying to study?
- Why is it important to do this search/measurement?
- How are you designing your analysis strategy? (what are the selections & why, how do you estimate your background?)

Finally, you should quantify your results by either providing a measurement or using a benchmark model to set a limit.

In https://www.dropbox.com/sh/53akrks6xn4ecae/AACVFD6F9TW_zph8BGF7tUB4a?dl=0 you will find a few examples of preapproval, approval, and conference presentations.

Here is the link to the presenatation in 2017:

https://indico.cern.ch/event/578454/sessions/218560/attachments/1395535/2127271/Ztautau_xsection.pdf

Responsible: AbdollahMohammadi

This topic: CMS > SWGuideCMSDataAnalysisSchoolLPC2018LongExerciseTau
Topic revision: r10 - 2018-01-12 - AbdollahMohammadi