Recreating the invariant mass of Z boson

Doris Lee

Supervised by Professor Dugan O'Neil

August 2013

Abstract

To analyze the leptonic decay of the Z and Higgs boson, I built PyROOT programs that recreate the invariant mass from the four-momentum data of the pair of leptons. The data used in this analysis is from the pp collision of the ATLAS experiment at LHC. The analysis of the distribution on the mass histogram and S/B cut flow tables showed the effectiveness of the cuts for QCD background elimination.

1 Introduction

1.1 ATLAS detector

The ATLAS liquid argon calorimeter contains an inner ring of EM calorimeter and an outer ring of hadronic calorimeter where the particles leave traces of shower and deposit their energy. The momenta of the particles P_t, P_x, P_y, P_z are recorded in the tracking chamber. Muons and neutrinos pass through these parts of the detector to the muon chamber where the four vectors are recorded. Since neutrinos only interact with the weak force, they can not be detected and is found through the missing energy and momentum in the collision.

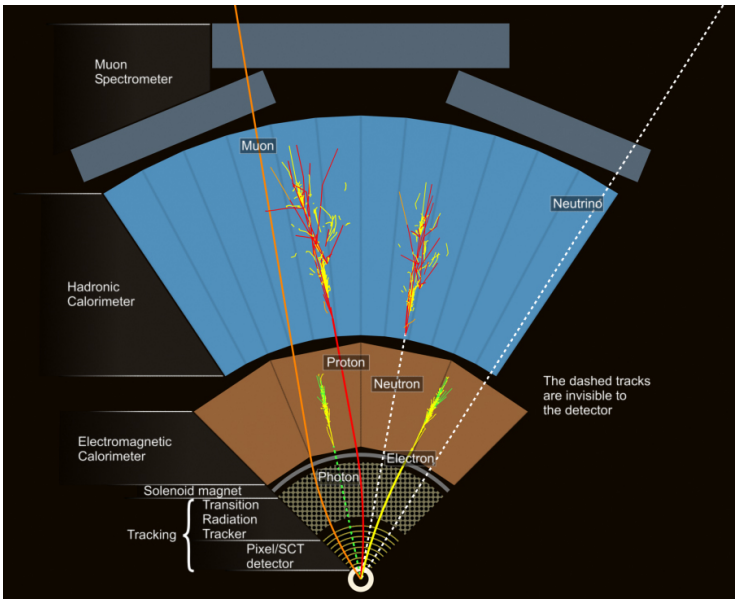


Figure 1: Cross sectional view of the ATLAS detector

1.2 Invariant mass calculation

The four-vector (E, P_x, P_y, P_z) defines a particle and is helpful in gaining other information about the particle, such as its invariant mass. Invariant mass is a particle's mass when its momentum is zero and is the same in all frames of reference.

We can calculate the mass of a particle using the energy-momentum relation from special relativity:

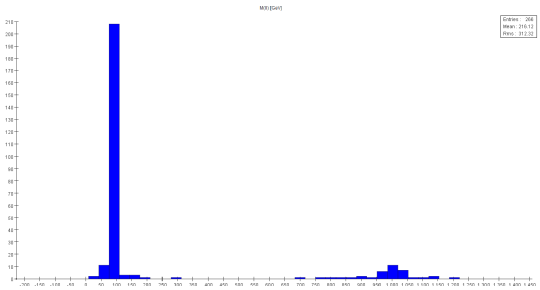
$$\begin{aligned} E^2 - (pc)^2 &= (mc^2)^2 \\ \text{let c=1 (unit-system)}, \quad E^2 - P^2 &= M^2 \\ M &= \sqrt{E^2 - P^2} \end{aligned} \tag{1}$$

where P is the resultant of the two particles' momenta.

2 HYPATIA

2.1 Z→ μ^+ μ^- simulation

Using the P_t cut in the Control Window to eliminate the tracks with lower P_t, it is easier to choose the muon pairs that correspond to the two lines that reach the muon spectrometer. Adding the muon pairs (which usually have two of the highest momenta), HYPATIA displays the mass in the Invariant Mass Window.



2.2 ≈ 1000s cluster

The distribution on the histogram indicates that a small number of masses fall in the region of around 1000GeV. Most of these pairs are almost collinear with each other and have relatively high momentum (>200GeV/c)compared to that of the muons from the ≈ 91 GeV Z bosons, which may be due to a head-on collision. These collisions are often associated with the release of a neutrino.

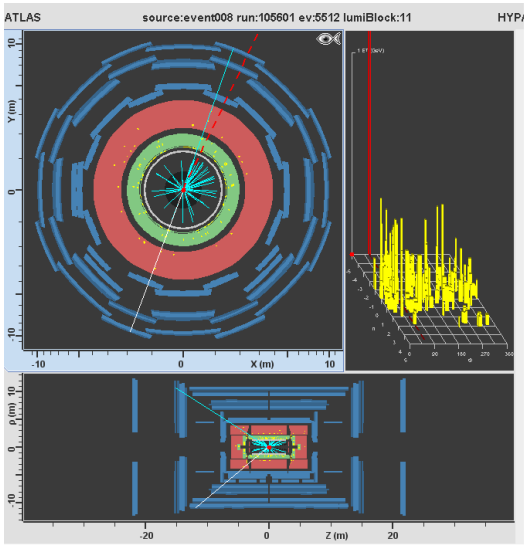


Figure 2: (μ^-, μ^+): P=(917.5,1013.7) ; P_t=(530.2,690.4) ; m=1216.819 The two muons are almost back to back with each other. The yellow bars on the right panel show the calorimetry-detected energy.

3 Python and ROOT

Learning basic Python programming assisted me in understanding PyROOT. The abundance of online resources made learning Python relatively easy. However, learning PyROOT was more challenging because there is less PyROOT documentations. I had to interpret the documentations of ROOT in C in order to understand the ROOT classes and methods, such as TChain, TH1F,and TCut. TTree was the most complicated part of PyROOT because at first the overwhelming amount of data, branches, and leaves was hard to work with. By practicing building, modifying, and reading TTrees, I eventually became familiar with using the information from various branches for analysis.

4 Analyzing Z$\rightarrow \mu^+ \mu^-$

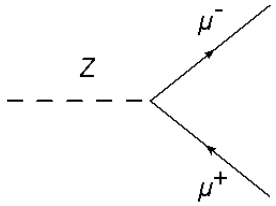


Figure 3: Z$\rightarrow \mu^+\mu^-$ Feynmann diagram

4.1 Analyzing data from HYPATIA

4.1.1 Python program

From HYPATIA's exported Z mass textfile, I created a Python program that calculates the average mass of the Z by summing the list of Z mass and dividing it by the total number of events. It yields an inaccurate result of 215.658080944 GeV; since it is the average Z mass, it takes into account of all the Z mass, including the 1000s cluster, making the result higher than the actual Z mass. Using Python, I defined a function called *histogram (L)* with many *elif* and *if* statements looped together by a *for* statement. I used the *elif* statements to manually define the bins. Looping through HYPATIA's Z mass list, the function categorizes the masses into each of the bins, which returned the values: 7,216,5,5,5,24,1,3,1,0. This function did not actually plot any graphical histograms.

4.1.2 PyROOT programs

Using PyROOT on the return values from the histogram function, I stored both the bin and the return values in a list, creating a histogram. Next, I created a simpler version of the histogram program that uses the TH1F class and looping the fill method. I also learned how to parse the textfile into the program using the readline command. Later I used the TCut, SetOptStat, Fit, and tried adjusting the bin size to modify the histogram. The program yields a mass of 201.4 GeV.

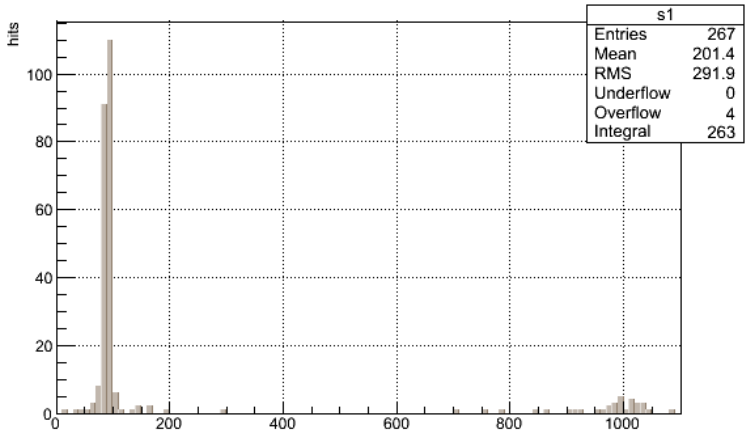


Figure 4: Z mass program

4.2 Calculating invariant mass of Z

The PyROOT program Athat calculates the invariant mass of Z runs through many loops. First, I set up a TChain called *mychain* where I loaded up the *tau* tree. Then, I stored all the names of the root files in the directory in a list called *listing.* By changing to the muon directory, I looped through all the items in the list to add it to *mychain* . Outside the loop I created two TLorentzVectors and declared *entries* as the number of collision events. The event loop goes through all the entries and loads up all the branches of the tree. I then created a list called *goodMuon* that is refreshed everytime it loops through an event. Incorporated in the muon loop is an *if* statement that adds only muons from events with two or more muons and muons with a transverse momentum restriction(initial program: $P_t >= 30000$) to the *goodMuon* list. If the *goodMuon* list had more than one good muon stored in it, I declared the first two muons in the list as *muon1* and *muon2* , respectively. Lastly, I set the four-vector to the TLorentzVectors. Then I calculated the mass of Z [1]using vector
addition and the mass function as shown below:

$$\begin{array}{rl} (E_1, Px_1, Py_1, Pz_1) & muon1(v1) \\ +\,(E_2, Px_2, Py_2, Pz_2) & muon2(v2) \\ \hline (E_1+E_2, Px_1+Px_2, Py_1+Py_2, Pz_1+Pz_2) & \longrightarrow V.m() \rightarrow massofZ \end{array} \tag{2}$$

By filling the mass into the histogram, I got:

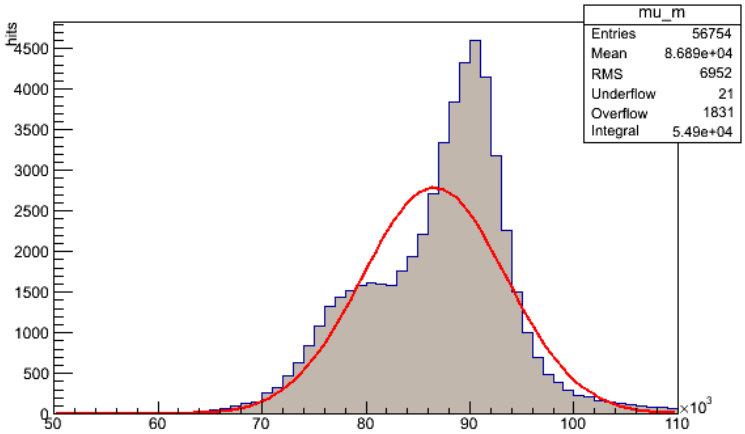


Figure 5: Using TChain for multiple root files

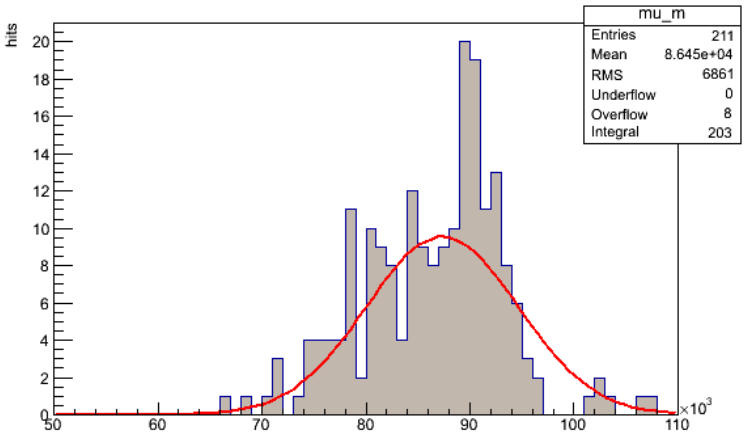


Figure 6: Using a single root file

[1]All plots in sections 4.2 and 5 are in MeV.

4.2.1 Challenges in creating the program

Loop

The multiple loops that are nested under one another makes it hard to distinguish which commands fall under which loop. Using the *if debug* printouts, the program's flow became more comprehensible.

m≈ 210 GeV

Initially, I made two TLorentzVectors without looping through a list. This yields the vector addition of two identical muons from which the invariant mass is calculated. Since each muon is 105.658369 MeV, the recreated mass was around 210 MeV. I fixed this problem by looping through the *goodMuon* list.

TLorentzVector

I thought that the 210 MeV problem was due to the v.m() because I was not sure if the function is programmed internally to do $M = \sqrt{E^2 - P^2}$.

TChain

I found out that I have to load the tree in the declaring of the TChain instead of using gDirectory.Get() which is used for a TFile.

5 Analyzing Z→ e^+e^-

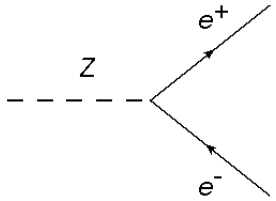


Figure 7: Z→ e^+ e^- Feynmann diagram

By using a similar program as Z→ μ^+ μ^-, I recreated a mass of 87020Mev (87 GeV) from the electrons. I tried using Z→ μ^+ μ^- file to find Z mass by adding electrons Z→ e^+e^- and I got an inaccurate result.I learned that even though ATLAS is a general detector, there are specific signal events in different files, just as how I could not recreate the mass of Z by adding electrons in HYPATIA. The electron results are more precise than the muon results because there were more events for Z→ e^+e^-.

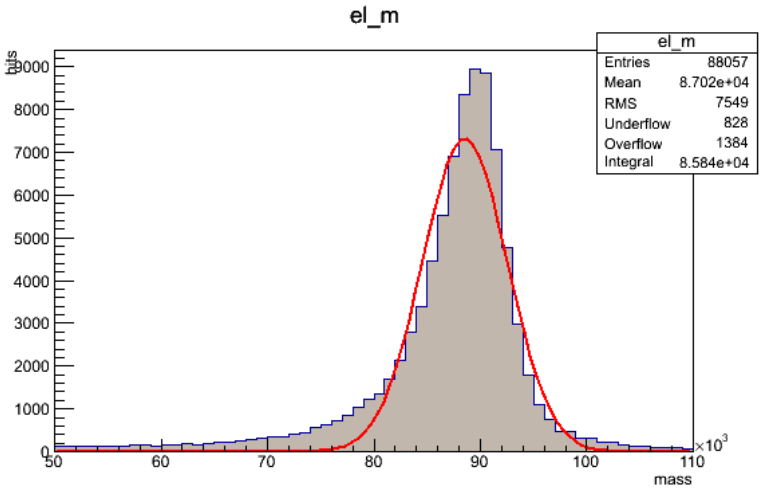


Figure 8: Invariant mass of Z from Z→ e^+ e^- decay data

6 QCD elimination

In this section, I try to investigate which combination of restrictions is most effective in eliminating the background from strong interactions. QCD elimination is important because background near the signal region can be superimposed with the real signal,which will give a less accurate distribution.The QCD elimination is not done on the muons because muons pass directly through the hadronic and EM calorimetry and can only be stopped by the muon spectrometer.

6.1 Quality cuts

The quality of the particle is determined by the shape of the shower it produces in the calorimetry. Each parameter (Loose,Medium, Tight) is specified by boolean value (where 1 is True and 0 is False).In the program, I specified seven combinations of the quality cuts in both the electron (el_LoosePP) and tau (tau_tauCutLoose).

$$\left\{\begin{array}{c}\text{Both leptons must pass loose (ll)}\\ \text{Both leptons must pass medium(mm)}\\ \text{Both leptons must pass tight(tt)}\end{array}\right\} \tag{3}$$

```
ex.)tau_tauCutLoose[tau1]+tau_tauCutLoose[tau2]==2
```

$$\left\{\begin{array}{c}\text{One must pass loose; another must pass medium(lm)}\\ \text{One must pass loose; another must pass tight(lt)}\\ \text{One must pass medium; another must pass tight(mt)}\end{array}\right\} \tag{4}$$

```
 ex.)tau_tauCutLoose[tau1]+tau_tauCutLoose[tau2]==2
tau_tauCutMedium[tau1]+tau_tauCutMedium[tau2]>=1
```

lm is also equivalent to "one is at least medium; both are loose" since all that passes medium must also pass loose, and vice versa. The *init* have no quality cuts; they only containP_t cuts ($P_t \geq 10000$ for electrons;$P_t \geq 20000$ for taus)

6.2 P_t Cuts

By analyzing the histogram of P_t restrictions from the data file, each incrementing by 5000, I found that there are peaks at 90 GeV (signal) and 60 GeV (background). The 60 GeV can be almost entirely eliminated by a higher P_t cut. Using this method, I determined the value for the common P_t cut used for both the tau and electron. The tau P_t is later adjusted to another value to allow more signal to pass through.

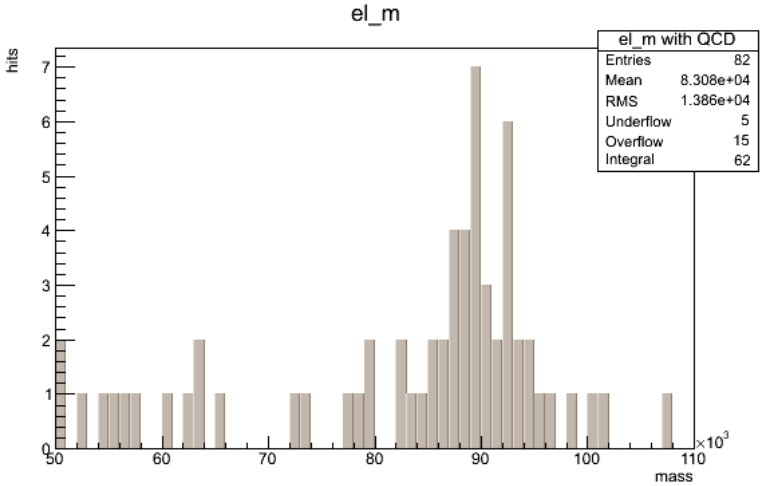


Figure 9: The background signal is most evident in the $P_t \geq 10000$ cut plot below since there is a consistent "base" on the plot.

7 Z→ e e QCD elimination

7.1 Discrepancies in Cut Flow Table results

7.1.1 Loose-Loose Cut

In some data files, most background are completely eliminated when the *ll* cut is applied(such as the simulated background). Chaining together more data files, there are some effects from the *lm* and *mm* .

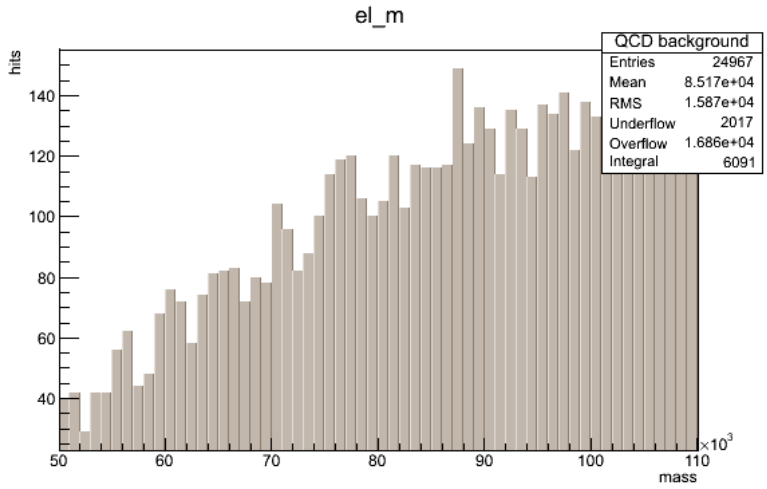


Figure 10: Monte Carlo simulated QCD background with no cuts. Subsequent *ll* cut yields zero entries.

7.1.2 Combinations involving tight cuts

The order at which *lt* and *mt* should be arranged on the cut flow table can vary because the tight cut is so strict that the S/B can start to decrease.

cut type	S	B	relative S/B	S/B
initial	9745	10646	0.9150	1
ll	8659	1478	5.8589	6.4002
lm	8657	1470	5.8890	6.4336
lt	8396	1446	5.8063	6.3432
mm	7733	1301	5.9439	6.4934
mt	7594	1289	5.8914	6.4361
tt	5964	1089	5.4766	5.9829

Background files=user.NoelDawe.HTauSkim.data11_7TeV.00178044.physics_JetTauEtmiss .merge.
NTUP_TAUMEDIUM.r2603_p659_p851.v4.120204173607
Tchain file:0001 to 0027
Total Entries=19040
Signal file=group.phys-higgs.73933_024864.107650._00256.LHSkim.mc11c_p851_lephad.root
entries=9851

Since the signal and background files are from different data files, the initial entries are not the same. I put a common factor (factor=10646/9745) on the relative S/B to obtain the actual S/B ratio. Some of these issues are also observed in the Z→ $\tau^+\tau^-$ files.

7.2 Summary

By chaining together the signal files and using the cut that yields the largest S/B ratio, I eliminated some of the background signal and obtained a mass of 87.11 GeV.

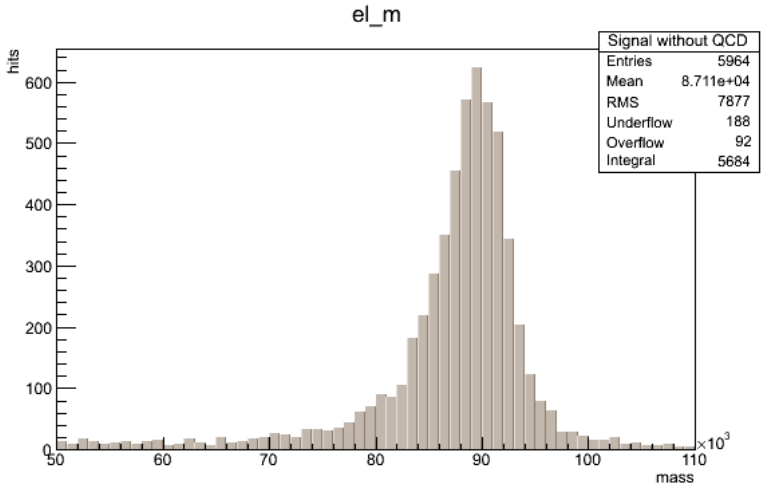
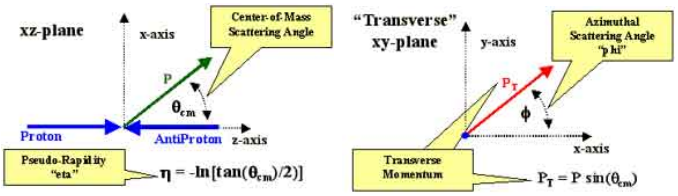


Figure 11: Mass signal from Tight-Tight cut

8 Z→ $\tau^+\tau^-$ QCD elimination

8.1 Initializing TLorentzVector with Eta,Phi,Pt

Instead of the P_x, P_y, P_z values measured in muons and electrons, taus' direction are specified by Eta (η) and Phi (ϕ). Phi is the polar angle relative to the transverse plane; eta is the psedorapidity, the angle relative to the beam axis. There are two ways to initialize the TLorentzVector given these values. They



where the z-axis is the beam axis

are convenient for working in the $\eta - \phi$ space. By initializing TLorentzVector with SetPtEtaPhiE(), the P_x, P_y, P_z of the tau can be calculated.

$$\left.\begin{aligned} P_x &= P_t cos\phi \\ P_y &= P_t sin\phi \\ P_z &= P_t sinh\eta \end{aligned}\right\} \qquad |P| = P_t cosh\eta \tag{5}$$

Thus the total momentum is also yield.

8.2 Missing mass in tau decay

When I calculated the Z mass in the modified tau program, I obtained a mass of around 50 to 60 GeV, significantly lower than the actual Z mass. Initially, I thought it was a logical error in the program, but I found that even manual calculation from a randomly selected tau yields such mass:

$$\begin{aligned} v.Px() \rightarrow Px &= 3.917e+04 \\ v.Py() \rightarrow Py &= 4.319e+04 \\ v.Pz() \rightarrow Pz &= 5.441e+04 \\ P = \sqrt{P_x^2 + P_y^2 + P_z^2} &= 79750.31724 \\ v.E() \rightarrow E &= 5.998e+04 \\ m^2 &= E^2 - P^2 \\ m &= 52559.61 MeV \end{aligned} \tag{6}$$

I learned that, unlike the muons and electrons, the tau decays produce neutrinos. Since their momentum and energy can not be recorded in the ATLAS detector, the products of the decay are not be fully recoved. Despite the energy lost, the effectiveness of the cuts can still be inferred from the S/B ratio.

8.3 Results

8.3.1 Cut Flow 1:

There is an evident decrease in both the signal and the background file as the quality cuts get stricter. The S/B ratio is increasing gradually (with the exception of the loose-tight cut) as the cuts get tighter. This shows that these cuts are working.

	S	B	S/B
init	1010	19040	1
ll	370	15128	0.461
lm	359	14269	0.47
lt	325	9866	0.62099
mm	245	9390	0.49
mt	239	7441	0.6055
tt	164	3365	0.91876

```
factor:18.85
signal file: group.phys-higgs.73919_024822.107670._00001.LHSkim.mc11c_p851_lephad.root
background file: TChain 0001-0027 in directory: user.NoelDawe.HTauSkim.data11_
7TeV.00178044.physics_JetTauEtmiss.merge.NTUP_TAUMEDIUM.r2603_p659_p851.v4.12020173607
```

From the histograms created, we can see that the P_t cut should be changed to 20 GeV (since both signal and background should have the same P_t cut) and the signal requires more files to be chained together.

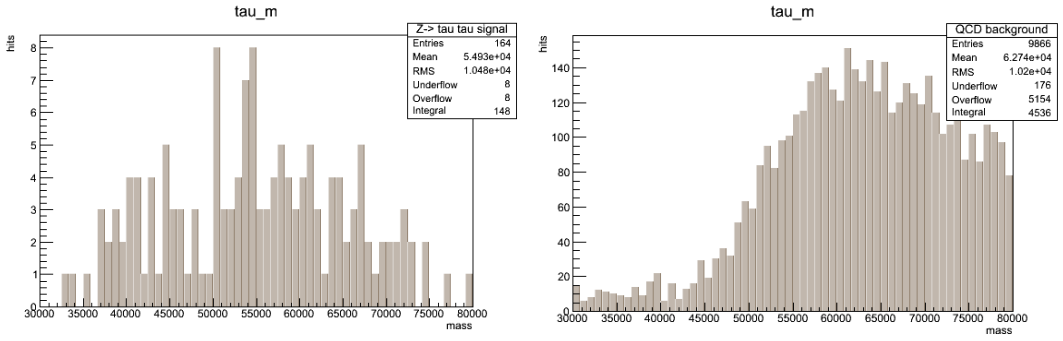


Table 1: Signal and background plots made from tt cuts

8.3.2 Cut Flow 2 (ELOG 39)

	S	B	S/B
init	3365	18423	1
ll	1829	15232	0.6574
lm	1789	14363	0.6819
mm	1309	9481	0.7558
mt	1264	7467	0.9268
lt	1650	9859	0.9163
tt	890	3367	1.4472

```
factor:18423/3365
signal file:group.phys-higgs.LHSkim.mc11_7TeV.107670.
AlpgenJimmyZtautauNp0_pt20.merge.NTUP_TAUMEDIUM.e835_s1299_s1300_r3043
_r2993_p851.small.v1.120505094954:TChain file(00001 to 00010): group.phys-
```

```
higgs.73919_024822.107670._00001.LHSkim.mc11c_p851_lephad.root
Background file: TChain 0001-0027 in directory: user.NoelDawe.HTauSkim.data11_
7TeV.00178044.physics_JetTauEtmiss.merge.NTUP_TAUMEDIUM.r2603_p659_p851.v4.120204173607
```

The fitted mass plots showed that the background files used in this trial is a signal file. The file contained events that had a different tau quality parameter. The medium parameter from the signal is different from the medium parameter in the background.

9 Working with TTrees

Histograms do not record the mass value but only the number of items in each defined bin. This makes it hard to change the properties of the histograms without having to rerun the whole program. Therefore, I created a program [2]that inputs the calculated mass values into a TTree. First, I created a double type array called n. Then, I created a branch that uses the values of the array inside the tree. By modifying the naming and conditional statement of each cut, each branch is a calulated mass list with its corresponding quality cut. The values calculated in the loop are then inputted inside this mass list. The values stored in the list are later appended in the array and back into the branch. Not only is the newly-created root file a storage for the mass values in each branch but each branch also has its own modifiable histogram in the TBrowser, as the binning can be adjusted later in the editor.

10 Implementing a new structure for analysis

When working with the large unskimmed files of H$\rightarrow \tau\tau$, how the program is written can significantly increase the speed of analysis. By separating the flag and analysis code, there is no need to run through the mass calculation again everytime something in the analysis is modified, therefore there is more flexibility for change.

Original program: access experimental data →loop calculates mass→mass list→loop each index into array →tree

New flag program:(run first) access experimental data→ assign flags→ loop calculates mass→new tree
New analysis program: access the flag tree to analyze data

10.1 Flag Code

The purpose of the flag code is to access the values from the original data tree, calculate the mass of the particle (H or Z boson), assign a flag value to each particle whose mass is recreated, and writing the mass and flag value into a tree that can be further used for analysis.

Implementation of the flag code offers:

1. Easier to adjust histograms details, since the calculated mass is not binned into the histogram but instead stored as values in of a single tree, the histogram is easily adjustable by the Editor in the ROOT Object Browser or by running a short script. This made the process of finding S/B, S/$\sqrt{B}$, S/$\sqrt{S+B}$ values significantly faster.
2. Easier way to modify flag requirements
3. Variables (such as P_x, P_y, P_z, P_t) do not need to be recopied into new variables of a new tree in order to be used
4. Faster runtime for analysis since mass calculation only have to be done once.

10.1.1 Definition of flags

I defined the flags as:
1→At least two taus in the event and tau's transverse momentum$\geq$ 200000(init)
2→Passing init and loose-loose or greater (ll)
3→Passing init and loose-medium or greater(lm)
4→Passing init and medium-medium or greater(mm)
5→Passing init and loose-tight or greater(lt)
6→Passing init and medium-tight or greater(mt)
7→Passing init and tight-tight (tt)

At first, I defined "0" as the ones that do not even pass "init" (pt >=20000and tau number >=2) by filling the whole array with 0 initally and overwriting with higher flag values by a series of if-else statements. However,

[2]Appendix B

since the mass is not calculated for events that do not pass "init" anyways, there is no point of using "0" flag for zero(uncalculated) mass value. So I defined the two arrays as one with non-zero flag and the other with corresponding non-zero mass values. I arranged my if-else statment from the loosest cut to the highest so that the flag correspond to the highest (tightest) cut possible since each slot can be continuously overwritten when it satisfies the requrement for a higher flag. Basing on the principle, all tt must have also passed all the quality cut below it (ll,lt,lm,mm...etc) and so on. However, this method may not work for some events in the lm,mm,lt region since there is no clear distinction of which is the tighter cut.

Initally, I used the variable EleBDT (Boosted Decision Tree)to define the flags, which resulted in mostly "init" cuts. I redefined the flag using JetBDT after learning that the EleBDTFlag distinguishes between electrons and taus whereas the JetBDTSig distinguishes between jets and taus.

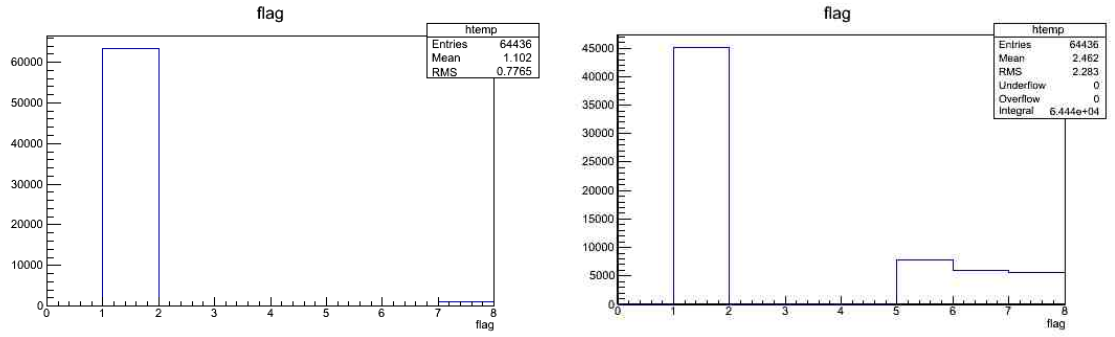


Table 2: EleBDTflagResult(Left); JetBDTflagResult(Right)

10.2 Analysis Code

The analysis code is a relatively short script that uses the data collected in the newly created tree. Since the mass is already calculated and stored in the tree, the run-time is sped up. This was particularly helpful when building cut-flow tables. Faster run-time enables different kinds of analysis to be done without having to access the original data tree. (Collecting signal and background entries, building histograms, making plots)The faster run-time also means more flexibility to modify plot details.

11 Finding best value to cut

11.1 Significance of S/B, S/$\sqrt{B}$, S/$\sqrt{S+B}$ values

- S/B: The signal and background entries are treated the same
- S/$\sqrt{B}$: $\sqrt{B}$ is related to the uncertainty of the background. If there is a particle at a certain mass then the S/$\sqrt{B}$ will exceed the $\sqrt{B}$ uncertainty.
- S/$\sqrt{S+B}$: is related to the signal's cross section.

11.2 Result

To find out the best mass value to cut the background without losing too much signal, I applied a mass cut every 10 GeV and stored the signal and background entries in two separate arrays. Then I wrote a short script that calculated the S/B, S/$\sqrt{B}$, S/$\sqrt{S+B}$ values and plotted them against the corresponding mass cut values using TGraph.

I thought that S/B ...etc ratio were relative so I chose used the original sample size from the original data file. However, since there was a significantly greater number of Higgs than the Z in the files, the number of Z "ran out" during the tighter cut as seen in the cut flow table , resulting an apparent "good cut": spike in the plot. However, by observing the pattern in the cut-flow table we know that the apparent boost in signal is only due to the lack of background to cut.

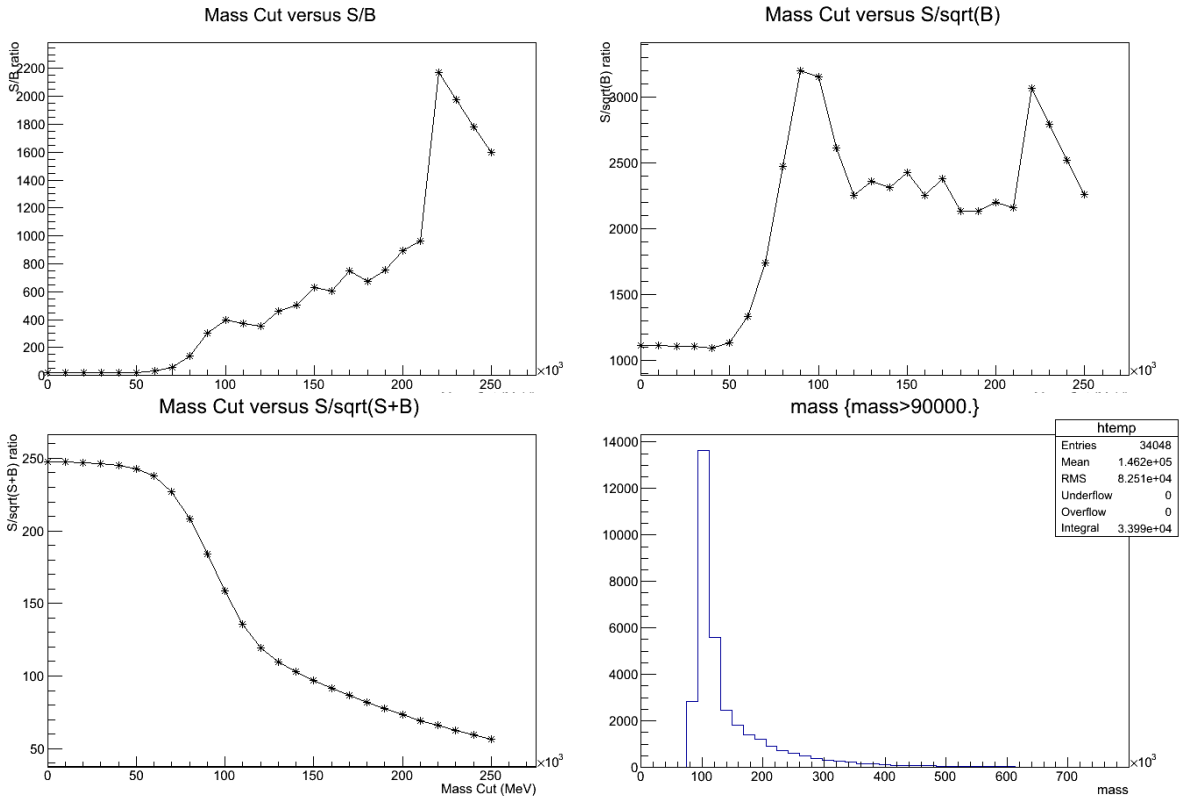
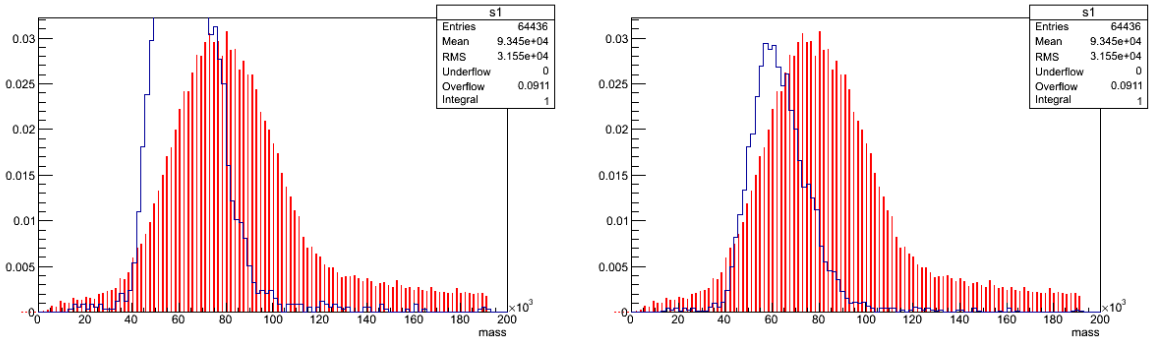


Figure 12: Although 90 GeV did not lie in the middle of the two peaks in the normalized, overlaid histogram, the S/sqrt(B) shows that 90 GeV is the best value to make the cut.

12 Scaling to Cross section

12.1 Normalizing Histogram

In the first attempt to overlay the Higgs and Z histograms, the significant difference in entry number made Z only a small peak which made it hard to distinguish the range between the peaks . Using the DrawNormalized[3] method, the two histogram is drawn with the same specified area. In the normalized histogram, the y-axis represents the number of event in the (bin/total number of events) multiplied by the specified area. The peak for the 120GeV Higg is around 80GeV and the peak for the Z boson is around 60 GeV.



Plot (right):
h1.DrawNormalized('bar1', 1)
h2.DrawNormalized('same',1)

Plot (left):
h1.DrawNormalized('bar1',1)
h2.DrawNormalized('same',.45)

[3]virtual TH1* DrawNormalized(Option_t* option = "", Double_t norm = 1) const

12.2 Cross Section

Cross section [4](σ)is the rate of production of a particle. It is measured in units of barn (1 bn= $10^{28}m^2$). Luminosity measures the number of interaction produced from an accelerator. The integrated luminosity (unit:fb^{-1} or m^{-2})is the total number of interactions in the accelerator in a given amount of time (a year). While looking at the cross section, luminosity ,branching ratio data, I noticed that the higher the center of mass energy($\sqrt{s}$), the larger the cross section; the higher the mass value of a particle, the smaller its cross section. The discrepancy between the number of 90 GeV Z and 120 GeV Higgs produced results from the difference in their respective cross section. The following calculation yields the value that can be used as the "weight" [5] in the normalization.

$$\begin{aligned}
&\text{Number of Z} \to \tau\tau\text{events at the LHC in 2012} \\
&= \text{Integrated luminosity at LHC for 2012} \times \text{cross section of Z} \times \text{branching ratio of} Z \to \tau\tau \\
&= (23.26 fb^{-1} \times 10^{15}\frac{fb}{b}) \times (41.541 nb \times 10^{-9}\frac{b}{nb}) \times 3.370\% \\
&= 32562411.342 \approx 3.26 \times 10^7
\end{aligned} \tag{7}$$

$$\begin{aligned}
&\text{Number of H} \to \tau\tau \text{ events at the LHC in 2012} \\
&= \mathcal{L} \times \sigma_{120GeVggFH} \times \text{BR}(Z \to \tau\tau) \\
&= (23.26 fb^{-1} \times 10^{15}\frac{fb}{b}) \times (53.49 pb \times 10^{-12}\frac{b}{pb}) \times 7.04\% \\
&= 8.759 \times 10^4
\end{aligned}$$

13 Useful ROOT Class and Methods

1. TTree: By trying to test whether the program is working and to understand the structure of the trees and branches, I learned new methods to try to display the data in the trees rather than just looking at the histogram display.
   - Scan[6]: returns values corresponding to each entry; useful for checking if values are properly written in the tree.

     ```
     f= TFile(‘ file_path’)
     t=f.Get(‘‘t1")
     t.Scan(‘‘flag")
     ```

   - Print[7]: returns the information about the tree (name, size, entries,branch...etc)
   - GetEntries[8]:returns number of mass entries
2. TGraph: plot specified points on the Cartesian coordinate system.
3. TCut: Draw only the masses (or any specified properties) with higher than a specified value

   ```
   ex) cut1 = TCut("mass>90000.")
   t.Draw("mass",cut1)
   ```

4. TH1F:1-D histogram with float variable
   - TH1F parameters
   - Draw options
   - Normalize
   - Fill with weight

[4]Calculations are made from data with center of mass energy of 8 TeV ($\sqrt{s}$=8 TeV)
[5]Appendix F
[6]Long64_t Scan(const char* varexp = “”, const char* selection = “”, Option_t* option = “”, Long64_t nentries = 1000000000, Long64_t firstentry = 0)
[7]virtual void Print(Option_t* option = “”) const
[8]Long64_t GetEntries() const

A Z→ e^+e^- invariant mass histogram program

```
from ROOT import *
import os
debug=False
gROOT.Reset();
gStyle.SetOptStat(1111111);
c1 = TCanvas( 'c1', 'c1', 200, 10, 600, 400 )
m=TH1F('el_m','el_m;mass;hits',80,40000,120000)
m.SetFillColor(11)
mychain=TChain('tau')
listing=os.listdir('/global/oneil/Doris-electrons/')
a=len(listing)
for x in xrange(a):
  os.chdir('/global/oneil/Doris-electrons/')
  mychain.AddFile(listing[x])
  if debug:print listing[x]
entries = mychain.GetEntries()
v1=TLorentzVector()
v2=TLorentzVector()

for jentry in xrange(entries):
  ientry = mychain.LoadTree( jentry)
  if ientry < 0:
    break
  nb = mychain.GetEntry(jentry)
  if nb <= 0:
    continue
  goodElectron=[]
  for ielectron in xrange(mychain.el_n):
    if debug: print "number of electron", mychain.el_n
    if debug: print "doing electron number ",ielectron
    if debug: print "pt is ",mychain.el_pt[ielectron]
    if mychain.el_n>=2 and mychain.el_tight==1 and  mychain.el_pt[ielectron]>20000:
      goodElectron.append(ielectron)
      if debug: print "I am adding a good electron to the list",ielectron
      if debug: print "the list is ",goodElectron
  if len(goodElectron)>1:
    electron1 = goodElectron[0]
    electron2 = goodElectron[1]
    if debug: print "the good electrons are numbers ",electron1,electron2
    v1.SetPxPyPzE(mychain.el_px[electron1],mychain.el_py[electron1],
    mychain.el_pz[electron1],mychain.el_E[electron1])
    v2.SetPxPyPzE(mychain.el_px[electron2],mychain.el_py[electron2],
    mychain.el_pz[electron2],mychain.el_E[electron2])
    v=v1+v2
    import math
    elmass=v.M()
    el=abs(elmass)
    m.Fill(el)
    print el
  if debug: print "--------------------------------"
  m.Draw('bar1')
m.Fit('gaus')
c1.SaveAs('el_m.png')
```

B Z→ $\tau^+\tau^-$ QCD elimination TTree program

```
\\Loose-Loose example
from ROOT import *
import os
from array import array
```

```
debug=False

f = TFile( 'QCD_tau_B.root', 'recreate' )
t = TTree( 't1', 'testTree')
n = array( 'd', [ 0 ] )
t.Branch( 'tau_m_ll', n, 'tau_m_ll/I' )

mass=[]
mychain=TChain('tau')
listing=os.listdir('/home/doris/Desktop/group.phys-higgs.HHSkim.mc11_7TeV.107701.
AlpgenJimmyWtaunuNp1_pt20.e835_s1299_s1300_r3043_r2993_p851.v1.120601000721')
a=len(listing)

for x in xrange(a):
  os.chdir('/home/doris/Desktop/group.phys-higgs.HHSkim.mc11_7TeV.107701.
  AlpgenJimmyWtaunuNp1_pt20.e835_s1299_s1300_r3043_r2993_p851.v1.120601000721' )
  mychain.AddFile(listing[x])
  if debug:print listing[x]

entries = mychain.GetEntries()
v1=TLorentzVector()
v2=TLorentzVector()

for jentry in xrange(entries):
  ientry = mychain.LoadTree( jentry)
  if ientry < 0:
    break
  nb = mychain.GetEntry(jentry)
  if nb <= 0:
    continue
  goodtau=[]

  for itau in xrange(mychain.tau_n):
    if debug: print "doing tau number ",itau
    if debug: print "pt is ",mychain.tau_pt[itau]
    if mychain.tau_n>=2 and mychain.tau_pt[itau]>=20000:
      goodtau.append(itau)
      if debug: print "I am adding a good tau to the list",itau
      if debug: print "the list is ",goodtau

  if len(goodtau)>1 :
    tau1 = goodtau[0]
    tau2 = goodtau[1]
    if debug: print "the good taus are numbers ",tau1,tau2
    if debug:print"the quality of tau is", mychain.tau_tauCutLoose[tau1],
     mychain.tau_tauCutLoose[tau2],mychain.tau_tauCutMedium[tau1],
     mychain.tau_tauCutMedium[tau2],mychain.tau_tauCutTight[tau1],
     mychain.tau_tauCutTight[tau2]

    if mychain.tau_tauCutLoose[tau1]+mychain.tau_tauCutLoose[tau2]==2:
      if debug:print "the taus pass quality control"
      v1.SetPtEtaPhiM(mychain.tau_pt[tau1],mychain.tau_eta[tau1],
    mychain.tau_phi[tau1], mychain.tau_m[tau1])
      v2.SetPtEtaPhiM(mychain.tau_pt[tau2],mychain.tau_eta[tau2],
    mychain.tau_phi[tau2], mychain.tau_m[tau2])
      v=v1+v2
      import math
      taumass=v.M()
      tau=abs(taumass)
      mass.append(tau)
      print tau
      if debug: print "-----------------------------------"
```

```
massnum=len(mass)
for i in range(massnum):
   n.append(mass[i])
   if debug:print "this is", i
   t.Fill()
f.Write()
```

C Flag Code H$\rightarrow \tau^+\tau^-$

The Flag Code of Z$\rightarrow \tau^+\tau^-$ has the same sturcture, but flags are defined by JetBDT.

```
from ROOT import *
import os
import glob
import math
from array import array
debug=False
gROOT.Reset();
gStyle.SetOptStat(1111111);

f = TFile( 'HtautauSignal_0428.root', 'recreate' )
t = TTree( 't1', 'testTree')
mychain=TChain('tau')
os.chdir('/home/doris/Desktop/mc12_8TeV.161576.PowHegPythia8_AU2CT10_ggH120_tautauhh.merge
.NTUP_TAU.e1217_s1469_s1470_r3542_r3549_p1344_tid01108681_00')
listing=os.listdir('/home/doris/Desktop/mc12_8TeV.161576.PowHegPythia8_AU2CT10_ggH120_tautauhh.merge.
NTUP_TAU.e1217_s1469_s1470_r3542_r3549_p1344_tid01108681_00')
a=len(listing)
for x in xrange(a):
  mychain.AddFile(listing[x])
  if debug:print listing[x]
entries = mychain.GetEntries()
v1=TLorentzVector()
v2=TLorentzVector()
#define flag array as int
flag=array('i',[0])
t1.Branch( 'flag', flag, 'flag/i' )
mass=array('f',[0])
t1.Branch('mass',mass, 'mass/F')
for jentry in xrange(entries):
  # mass and flag array reset
  mass[0]=0
  flag[0]=0
  ientry = mychain.LoadTree( jentry)
  if ientry < 0:
    break
  nb = mychain.GetEntry(jentry)
  if nb <= 0:
    continue
  goodtau=[]
  for itau in xrange(mychain.tau_n):
    if debug: print "doing tau number ",itau
    if debug: print "pt is ",mychain.tau_pt[itau]
    if mychain.tau_n>=2 and mychain.tau_pt[itau]>=20000:
      goodtau.append(itau)
      if debug: print "I am adding a good tau to the list",itau
      if debug: print "the list is ",goodtau
  if len(goodtau)>1 :
    tau1 = goodtau[0]
    tau2 = goodtau[1]
    if debug: print "the good taus are numbers ",tau1,tau2
    if debug:print"the quality of tau is", mychain.tau_
```

```
    JetBDTSigLoose[tau1], mychain.tau_JetBDTSigLoose[tau2],mychain.tau_JetBDT
    SigMedium[tau1],mychain.tau_JetBDTSigMedium[tau2],
    mychain.tau_JetBDTSigTight[tau1], mychain.tau_JetBDTSigTight[tau2]
    #init
    flag[0]=1
    #flag corresponds to the highest (tightest) cut possible since same element value can overwrite the slot
    #ll
    if mychain.tau_JetBDTSigLoose[tau1]+mychain.tau_JetBDTSigLoose[tau2]==2:
      if debug:print "the taus pass ll "
      flag[0]=2
    #lm
    if mychain.tau_JetBDTSigLoose[tau1]+mychain.tau_JetBDTSigLoose[tau2]==2
    and mychain.tau_JetBDTSigMedium[tau1]+ mychain.tau_JetBDTSigMedium[tau2]>=1:
      if debug:print "the taus pass lm "
      flag[0]=3
    #mm
    if mychain.tau_JetBDTSigMedium[tau1]+ mychain.tau_JetBDTSigMedium[tau2]==2:
      if debug:print "the taus pass mm"
      flag[0]=4
    #lt
    if mychain.tau_JetBDTSigLoose[tau1]+ mychain.tau_JetBDTSigLoose[tau2]==2
    and mychain.tau_JetBDTSigTight[tau1]+mychain.tau_JetBDTSigLoose[tau2]>=1:
      if debug:print "the taus pass lt"
      flag[0]=5
    #mt
    if mychain.tau_JetBDTSigMedium[tau1]+mychain.tau_JetBDTSigMedium[tau2]==2
    and mychain.tau_JetBDTSigTight[tau1]+ mychain.tau_JetBDTSigTight[tau2]>=1:
      if debug:print "the taus pass mt "
      flag[0]=6
    #tt
    if mychain.tau_JetBDTSigTight[tau1]+mychain.tau_JetBDTSigTight[tau2]==2:
      if debug:print "the taus pass tt "
      flag[0]=7
    v1.SetPtEtaPhiM(mychain.tau_pt[tau1],mychain.tau_eta[tau1],mychain.tau_phi[tau1], mychain.tau_m[tau1])
    v2.SetPtEtaPhiM(mychain.tau_pt[tau2],mychain.tau_eta[tau2],mychain.tau_phi[tau2], mychain.tau_m[tau2])
    #mass calculation
    v=v1+v2
    taumass=v.M()
    tau=abs(taumass)
    print tau
    mass[0]=tau
    #keeping only entries with non zero mass and flag values
    if mass[0]==0 and flag[0]==0:
      continue
    else:
      t1.Fill()

f.Write()
f.Close()
```

D Analysis Code H$\rightarrow \tau^+\tau^-$

```
from ROOT import *
import os
from array import array
debug=False
gROOT.Reset();
gStyle.SetOptStat(1111111);
#update overwrites the original file, read and recreate new tree
f = TFile( '/home/doris/Desktop/H->tautauSignal/HtautauSignal_0428flag.root' )
newF=TFile('H->tautauSignal0516Analysis.root','update')
newT=TTree ('analysis7','analysisTree')
```

```
mychain=TChain('t1')
mychain.AddFile('/home/doris/Desktop/H->tautauSignal/HtautauSignal_0428flag.root')
entries = mychain.GetEntries()
#init
tt = array( 'f', [ 0 ] )
newT.Branch( 'tau_tt', tt, 'tau_tt/F' )
for jentry in xrange(entries):
  tt[0]=0
  ientry = mychain.LoadTree( jentry)
  if ientry < 0:
    break
  nb = mychain.GetEntry(jentry)
  if nb <= 0:
    continue
  #Get number of entries that passes tt (Can be modified for different purposes)
  if mychain.flag >=7:
    print mychain.mass
    tt[0]=mychain.mass
    if debug:print tt[0]
  if tt[0]==0:
    continue
  else:
    newT.Fill()
newF.Write()
newT.GetEntries()
#newT.Scan("tau_m_tt")
#newT.Fit('gaus','tau_m_tt')
f.Close()
newF.Close()
```

E TGraph plotting and S/B calculation code

```
from ROOT import *
from array import *
debug=True
mass_cut=array('f',[0.,10000.,20000.,30000.,40000.,50000.,60000.,70000.,80000.,90000.,100000.,110000.
,120000.,130000.,140000.,150000.,160000.,170000.,180000.,190000.,200000.,210000.,220000.,230000.,240000.,250000.]
s=array('f', [64436.,64430.,64211.,63776.,63127.,61752.,58516.,52380.,43624.,34048.,25211.,18452.
,14226.,12027.,10585.,9401.,8428.,7521.,6742.,6029.,5380.,4829.,4341.,3947.,3561.,3198.])
b=array('f',[3365.,3365.,3365.,3346.,3321.,2960.,1932.,904.,311.,113.,64.,50.,40.,26.,21.,15.
,14.,10.,10.,8.,6.,5.,2.,2.,2.,2.])
s_b=array('f',[])
s_sqrt_b=array('f',[])
s_sqrt_s_b=array('f',[])
for x in range (len(s)):
    val=s[x]/b[x]
    if debug: print val
    s_b.append(val)
    val2= s[x]/ math.sqrt(b[x])
    if debug: print val2
    s_sqrt_b.append(val2)
    val3= s[x]/ math.sqrt(s[x]+b[x])
    if debug: print val3
    s_sqrt_s_b.append(val3)
if debug:print s_b
if debug:print s_sqrt_b
if debug:print s_sqrt_s_b
g = TGraph(26, mass_cut,s_b)
g.SetTitle("Mass Cut versus S/B ")
g.GetXaxis().SetTitle("Mass Cut (MeV)")
g.GetYaxis().SetTitle("S/B ratio")
g.Draw("AL*")
```

F Weighing Histogram

```
from ROOT import *
import os
from array import array
debug=False
gROOT.Reset();
gStyle.SetOptStat(1111111);
c1 = TCanvas( 'c1', 'c1', 200, 10, 600, 400 )
hf = TFile( '/home/doris/Desktop/H->tautauSignal/HtautauSignal_0428flag.root' )
h = TH1F( 'h1', 'H and Z mass (weighted)', 150, 0., 300000.)
h.SetFillColor(11)
hchain=TChain('t1')
hchain.AddFile('/home/doris/Desktop/H->tautauSignal/HtautauSignal_0428flag.root')
entries = hchain.GetEntries()
mass=[]
#Drawing Higgs Histogram
h_weight_factor=1./(8.759e4)
for jentry in xrange(entries):
  ientry = hchain.LoadTree( jentry)
  if ientry < 0:
    break
  nb = hchain.GetEntry(jentry)
  if nb <= 0:
    continue
  mass.append(hchain.mass)
for x in mass:
  h.Fill(x,h_weight_factor)
h.Draw('BAR1')
#Drawing Z Histogram
zh = TH1F( 'z', 'test', 150, 0., 300000.)
zh.SetFillColor(1)
zf = TFile('/home/doris/Desktop/tauSignal.root')
zchain=TChain('t1')
zchain.AddFile('/home/doris/Desktop/tauSignal.root')
z_entries = zchain.GetEntries()
zmass=[]
z_weight_factor=1./32562411.342
for jentry in xrange(z_entries):
  ientry = hchain.LoadTree( jentry)
  if ientry < 0:
    break
  nb = hchain.GetEntry(jentry)
  if nb <= 0:
    continue
  mass.append(hchain.mass)
for x in mass:
  zh.Fill(x,z_weight_factor)
zh.Draw('BAR1 SAME')
```

G Work Cited

- "A.2.Working with TTrees." *Python Scripting.*N.p., n.d. Web. 25 Oct. 2012.
- Alessio, Damato.*LaTeX.*N.p.: n.p.,n.d.Wikibooks. Web. 12 Aug. 2012.
- Applications of Special Relativity on Particle Physics.*High School Teachers at CERN.*CERN, n.d. Web. 7 Aug. 2012.
- Bock, Rudolf K. "Pseudorapidity." *Pseudorapidity.*CERN, 9 Apr. 1998. Web. 16 Nov. 2012.
- "Chapter 19. Python and Ruby Interfaces." *ROOT.* N.p., n.d. Web. 24 Sept. 2012.
- "Example 1: A Tree with Simple Variables." *ROOT* . N.p., n.d. Web. 25 Oct. 2012.
- Field, Rick, Craig Group, and David Wilson. *Pseudo-Rapidity, Azimuthal Angle, and Transverse Momentum.*Collider Detector Fermilab (CDF), 4 Nov. 2012. Web. 16 Nov. 2012.
- How a Particle Accelerator and Detector Work. Digital image. *Syracuse University Experimental High Energy Physics.*N.p.,n.d. Web.7 Aug. 2012.
- "How to Use Chains (List of Files)"*ROOT.*CERN, n.d. Web. 13 Aug. 2012.
- "Invariant Mass." *International Physics Masterclasses.*CERN, n.d. Web. 10 Aug. 2012.
- J. Beringer et al. (Particle Data Group), Phys. Rev. D86, 010001 (2012) and 2013 partial update for the 2014 edition
- "LaTeX Line and Page Breaking." *TeX in CEU.*N.p., n.d. Web. 13 Aug. 2012.
- "Latex Math Symbols." *Latex Math Symbols.*N.p., n.d. Web. 04 Nov. 2012.
- Metropolis, Kate. *Understanding Luminosity through 'barn', a Unit That Helps Physicists Count Particle Events.*Stanford Report, 21 July 2004. Web. 02 Aug. 2013.
- Murat, Pasha, and Peter Malzacher. "Class TLorentzVector: Public TObject." ROOT. CERN, 2 Dec. 1999. Web. 9 Oct. 2012.
- Muratori, Bruno, and Werner Herr. "Concept of Luminosity." CERN, n.d. Web.
- Ohl, Thorsten. *Drawing Feynman Diagrams with LaTeX and Metafont.*Tech. no. IKDA-95-20. N.p.: n.p., 1995. ArXiv. Web. 12 Aug. 2012.
- *Pt_def, Eta_def.*Digital image. *Pseudo-Rapidity, Azimuthal Angle, and Transverse Momentum.* Collider Detector Fermilab (CDF), n.d. Web. 17 Nov. 2012.
- Pivarski, Jim. *What Do We Mean by "cross section" in Particle Physics?* CMS Experiment, 18 Mar. 2013. Web. 02 Aug. 2013.
- "The ATLAS Calorimeter." Website.*Brookhaven and the LHC.*N.p., n.d. Web. 09 Aug. 2012.
- Roberts, Andrew. "Floats, Figures and Captions." *Getting to Grips with LaTeX.* N.p., n.d. Web. 13 Aug. 2012.
- "ROOT::Math::PtEtaPhiM4D Class Template Reference[3D and 4D Vectors]." MathCore: ROOT::Math:: PtEtaPhiM4D Class Template Reference. CERN, 14 Dec. 2005. Web. 19 Oct. 2012.
- "Working with TTrees." *PyROOT.*N.p., n.d. Web. 13 Aug. 2012.
- Yamanaka, Taku. "FeynMP / FeynMF Examples." Osaka University Physics Dept, 25 Mar. 2006. Web. 13 Aug. 2012.
- "8.6. Array Efficient Arrays of Numeric Values." Python 2.7.3 Documentation. *Python Software Foundation*, n.d. Web. 16 Nov. 2012.