This tutorial is for using root on OS X. It is based on other tutorials that I have found to be useful. The most helpful tutorial I have found is here: <http://www.nevis.columbia.edu/~seligman/root-class/RootClass2015.pdf>. The tutorial that is written here is based on the that tutorial. Many sections will be very similar. However, that tutorial is more general. The tutorial that I have written is more specific to working with KU CMS. This tutorial also assumes that you have some prior experience with C++. I recommend taking EECS 138 for learning the basics of C++. However, there are also many good tutorials online. Before beginning this tutorial make sure you have the latest version of XCode and OSX. In this case that is XCode 7.3.1 and OSX El Capitan 10.11.5. I have also created a google folder that can be accessed here that has some files that we will be working with: <https://drive.google.com/open?id=0B4-Ce8bwcKn1SlJxQVRnRzFyejQ>

You can either download all of those files now or download them as you progress through the tutorial.

Chapter 1: Installing Root

Here is a link to a good website for instructions on installing root. I will go over those steps in this tutorial but this link is also a good reference if you get lost. <https://alexpearce.me/2016/02/root-on-os-x-el-capitan/>

We will be installing root using a software known as homebrew. Homebrew will handle most of the downloading and installation for us. This link is for the homebrew installation page: http://brew.sh All it does is tell you to paste a script into the terminal[[1]](#footnote-1), which I have copied here: /usr/bin/ruby -e "$(curl –fsSL <https://raw.githubusercontent.com/Homebrew/install/master/install)>"

After installing homebrew close and reopen the terminal to start fresh.

Type the following commands into the terminal but wait for each process to finish before moving on

brew install cmake

brew tap homebrew/science

brew install root6

Now homebrew has done most of the heavy lifting there is still one more thing for us to do. Open Finder and type “configure” into the search bar. Click on “This Mac”. You should see a file (maybe multiple files) that looks like this:



Double click on this file to process it. Once it finishes then root has been installed!

Chapter 2: The basics of Root

Open a fresh terminal shell. Type “root” in the prompt and press enter. A graphic will pop up along with some other info about root (You can type “root -l” to skip the intro). You can type “.q” to quit root. Let’s begin by plotting a simple function. Type this into root: TF1 f1("func1","sin(x)/x",0,10). Then type f1.Draw() to plot the function.

Chapter 3: Using the Browser and Tree Viewer

Start with a new root shell and type “TBrowser b” to open the browser. The root browser is a great way to navigate through your files. It also allows root to know which files you are working with. Navigate to where you have the tutorial files downloaded and double click on the file “tp\_M1\_pythia\_lhe\_events.root”. The root terminal may pop up some warnings and should also say something like (TFile \*) 0x7f83dd597cb0. For those of you that are familiar with C++; this is a pointer. The root file has also now been loaded into memory. Double click on LHEF;1 to open the tree. Then open the Event branch and double click on “Event.Nparticles.” A histogram will appear on the right in the Canvas. This plot shows the number of particles in each event. There are a few other ways to generate this plot.

Right-Click on the tree folder (LHEF;1) and click on “StartViewer.” Drag the leaf Event.Nparticles to the x: variable slot. Click on the button next to the “SPIDER” button and the same histogram from earlier will appear on the canvas. Feel free to play around with other variables and the tree viewer. The tree viewer is great for simple plots and getting used to working with root. However, it is just the tip of the iceberg.

Open a new root terminal and load the file “tp\_M1\_pythia\_lhe\_events.root” into memory. Type “LHEF->Draw(“Event.Nparticles”)” into root and press enter. Our favorite plot from earlier should appear in the canvas.

Chapter 4: Simple Event Plot

Now comes the more advanced portion of the tutorial. For those of you not as familiar with C++ a macro is essentially a set of instructions for your computer to run through. Let’s start by generating a macro. Load the file “tp\_M1\_pythia\_lhe\_events.root” again (You should be a pro at this now). Type LHEF->MakeSelector(“Tutorial”) into the prompt. The files Tutorial1.C and Tutorial1.h have now been created in the folder that your root file is in. Open both of these files with XCode. NOTE: ONLY EDIT THE .C FILE. DO NOT EDIT THE .h FILE. NOTE: ONLY EDIT THE .C FILE. DO NOT EDIT THE .h FILE. NOTE: ONLY EDIT THE .C FILE. DO NOT EDIT THE .h FILE.

For those of you familiar with C++, the .h file is the header file. All of the information in the header file is included in the .C file with the line “#include “Tutorial1.h”.” The header file stores most of your variables and to make sure that your macro looks in the correct place it is important that you ONLY EDIT THE .C FILE. DO NOT EDIT THE .h FILE. Now let’s learn how to run the macro that you just created. Open a fresh root terminal and load the root file from earlier into root. Type in “LHEF->Process(“Tutorial1.C”)” to run the macro. You will see nothing happen so let’s make something happen!

Open the .C file in XCode if you have not done so already. The .C file is divided into multiple parts[[2]](#footnote-2). I will often refer to everything before the Begin function (void Tutorial::Begin(TTree \* /\*tree\*/)) as the Definition section. I may also refer to the Begin function as the Initialization section, the Process function as the Loop section and the Terminate function as the Wrap-Up section[[3]](#footnote-3)[[4]](#footnote-4). Let’s try creating the same plot of Nparticles that we have done so many times before.

In the definitions section under “#include <Tstyle.h>”, put “TH1\* M1\_NparticlesHist=NULL;”. This will define a one dimensional histogram and set it equal to NULL. In the Begin function before “TString option = GetOption();” put the following line:

M1\_NparticlesHist = new TH1F("M1 Nparticles","Histogram of Nparticles",15,14,29);

In the parentheses “Histogram of Nparticles” is the title of the histogram when it is drawn to the canvas. 15, is the number of bins in the histogram. 14 and 29 are the limits of the histogram on the x-axis[[5]](#footnote-5). In the loop section put this line: M1\_NparticlesHist->Fill(Event\_Nparticles[0]); This will fill the histogram that was defined earlier. Event\_Nparticles is found in the .h file. The .h file has each leaf from the tree stored in it. Notice that most of the .h file is arrays. The index that we are pulling from is set at 0. Most of the time Event arrays only have one element. Now that we have our histogram filled, we still need to plot it. Remember the draw command from earlier? We will now use a variation of this in the wrap-up section. Put this line in there: “M1\_NparticlesHist->Draw();” Go back and run this file[[6]](#footnote-6). The same plot from before should appear in the canvas.

You may be wondering how I knew what to put when defining my histogram in the initialization function. Open the plot of Nparticles from the browser (not by using the macro). Go to File->Save and click Canvas\_1.C. A file will appear in the folder that the root file is in. There is a line similar to the line we just used:

TH1F \*htemp\_\_3 = new TH1F("htemp\_\_3","Event.Nparticles",15,14,29);

This is a good way of finding information on how a histogram was generated.

Chapter 5: Simple Particle Plot

Let’s try plotting the transverse momentum (PT) from a new root file. Open up root and load the file “tprime\_13TeV\_Mass500\_events.root”. Enter this make selector code to create the framework for our analysis macro “LHEF->MakeSelector(“Tutorial2”)”. We will essentially write the same code that we used earlier but with a minor tweak. Let’s start by writing the code for everything except the process function.

In definitions: TH1\* PT\_Hist=NULL;

In Initialization: PT\_Hist = new TH1F("PT","Histogram of PT",100,0,1280);

In terminate: PT\_Hist->Draw();

All of that was very similar to what you did in chapter 4. The process section is a little bit different. In this root file are event are particle arrays. Events have only one element, while particle arrays will vary. In the event a variety of particles are generated. I have seen this range from around 15 to around 25. Most macros will have two loops. One for going through the events and another inside that loop that goes through the particles. Because the number of particles will vary we cannot have a set limit for the particle array. Also the process function is called 10,000 times so that in itself is the event loop. The particle loop for statement is this: for(int i=0; i<Event\_Nparticles[0]; i++)

{

PT\_Hist->Fill(Particle\_PT[i]);

}

This will loop through every particle in every event and plot their PT.

Chapter 6: Selecting part of an array.

Open up root and load the file “tprime\_13TeV\_Mass500\_events.root”. We are going to plot the eta distribution of the bottom (b) quarks from the events. Start by making a selector file and calling it “TutBQEta”. In the definitions section define the histogram with, “TH1\* BQEtaHist=NULL;”. In the initializations section define the parameters of the histogram with this line, “BQEtaHist = new TH1F("BQ Eta", "Eta from BQ",100,-15,15);”. Remember how each leaf is an array? We need to go through the entire array in order to plot the b quarks. We will start by using a “for” loop to go through each index in the array. Then we will have an “if” statement check if the particle ID is equal to 5 or -5 which would be a bottom quark. We will also have to use a new variable called status to check if the particle is in its final state. Here is the code:

Reader.SetEntry(entry);

for(int i=0; i<15; i++)

{

if((Particle\_PID[i]==5 || Particle\_PID[i]==-5) && Particle\_Status[i]==2)

{

BQEtaHist->Fill(Particle\_Eta[i]);

}

}

Finally, we need to plot this histogram with the draw command. “BQEtaHist->Draw();” (in the terminate function).

Chapter 7: Tprime and ttbar

This chapter is not necessary for learning root. This is just background information.

In Chapter 6, we found the Eta distribution of the bottom quarks. Now we are going to see where the bottom quarks are coming from. First, you need to know a little bit more about the top quark. At KU we look at two types of quark decay. We look at the standard model which is a top quark (t) and an anti-top quark (t with a bar on top or tbar). We are also looking for a hypothetical top quark called tprime (T). These quarks decay differently.

t->b,q,qbar

tbar->bbar,q,qbar

t’->tH

H->b,bbar

t->bW

W->q,qbar

Essentially the tprime creates three bottom quarks and ttbar creates two bottom quarks. We will start by looking at tprime.

Chapter 8: Mother Particle

The mother particle, is the particle that decays into another particle. In chapter 6, you learned how to plot the Eta from every bottom quark. Now you will only plot the Eta from the bottom quarks that come from the Higgs boson. If you haven’t already load the root file, “tp\_M1\_pythia\_lhe\_events.root”. Start by creating a new selector file. I’ll call mine Mother\_Tutorial.

1. To open the terminal go to: Applications->Utilities->Terminal in Finder. For those of you familiar with Windows this is similar to the command prompt. [↑](#footnote-ref-1)
2. If you aren’t familiar with C++ anything after // or between /\* and \*/ are comments highlighted in green. C++ compilers will skip this when running a code. [↑](#footnote-ref-2)
3. This is the same nomenclature that the nevis tutorial uses. [↑](#footnote-ref-3)
4. The SlaveBegin and SlaveTerminate functions are for running your code on multiple computers with multiple processors and is (for now) beyond the scope of this tutorial. [↑](#footnote-ref-4)
5. At the end of this section I will show how I knew how many bins and what the limit of the histogram should have been. Here is a hint: When graphing with the tree viewer, look at the save options and see if you can figure it out. [↑](#footnote-ref-5)
6. Remember to load the root file and use LHEF->Process(“Tutorial1.C”). You may want to do this in a new root terminal. [↑](#footnote-ref-6)