ROOT Part 1

What is ROOT

- https://root.cern
- ROOT is a set of libraries optimized for HEP research
- Developed and maintained at CERN
  - Written in C++ with interfaces to Python and R available
  - Successor to previous FORTRAN-based CERN Program Library
- Wide range of features such as simple calculation, relativistic calculations, histogramming, graphing, functional fitting, columnar analysis and efficient data storage/access
- Industry standard for HEP
  - Some movement towards using other tools such as uproot and numpy

Installing ROOT

- Installing ROOT is quite simple
  - A few years ago, it required building the binaries yourself
- Recommend using version 6.24.X or newer
  - New versions occasionally break backwards compatibility
- https://root.cern/install/
  - Pre-compiled binary available for Linux, MacOS and Windows (beta version)
  - Available through package managers
  - Check dependencies before installation
- Source thisroot.sh(.bat) to set environment variables
- Already available on lxplus

Launching ROOT

- Launch using root
  - Options can be passed using -
    - -l: suppress splash screen (-a to enable it now)
    - -b: batch mode (prevents pop ups)
    - -q: quit ROOT at the end of executing command (generally when running a macro)
  - Splash screen has been turned off by default since 6.20
- Arguments can be passed, usually used to run macros or open files
- Display a list of ROOT commands with .help or .? (don’t forget period)
- Exit ROOT session with .q

ROOT on lxplus

- ROOT is installed and set up by default on lxplus
  - Can run immediately upon login
- Opening X11 windows can be very slow
  - Add “X11.UseXft: no” to ~/.rootrc to speed it up a bit
- To set up a specific version of ROOT:

```
export ATLAS_LOCAL_ROOT_BASE=/cvmfs/atlas.cern.ch/repo/ATLASLocalRootBase
alias setupATLAS='source ${ATLAS_LOCAL_ROOT_BASE}/user/atlasLocalSetup.sh'
setupATLAS
lsetup "root 6.28.00-x86_64-centos7-gcc11-opt" (as an example version)
```

VSCode ROOT extension

- VSCode has a useful extension to view ROOT files
- ROOT File Viewer by Alberto Péraz de Rada Fiol
- Generally faster than integrated ROOT browser
    - Somewhat limited functionality, but great for quickly viewing files

ROOT Documentation

- Extensive documentation available on ROOT website
  - https://root.cern/manual/basics/ - good starting point
  - https://root.cern/doc/master/ - provides all class definitions
  - https://root.cern/doc/master/group__Tutorials.html - good tutorials
  - https://root-forum.cern.ch/ - ask questions to experts (or find existing questions)
- ROOT naming conventions:
  - Class/namespace and member functions are in UpperCamelCase (a.k.a. PascalCase)
  - Most classes/namespaces begin with T
  - Non-class types end in _t
- When using Google, begin search with “CERN ROOT”
  - ROOT refers to the top level directory in a file system or the name of an admin account

ROOT Calculations

- ROOT can be used as a simple calculator
  - The result of any math command is printed to the screen
- Variables can be declared, assigned and used
  - Variable type is implicit but can be done explicitly
- Can use cmath or TMath functions
- TMath namespace provides huge number of mathematical methods
  - https://root.cern/doc/master/namespaceTMath.html
  - Call using e.g., TMath::Sqrt(7)

ROOT Objects

- ROOT has classes for many different types of objects
  - Thorough system of inheritance - well documented
- Most classes do not have implicit (default) constructors
- Declare objects as pointers and call constructor with new
  - Be sure to delete your pointers when you are done
- Most classes inherit from TObject and TNamed
  - Have name and title string attributes
  - name is a unique identifier that ROOT uses to retrieve objects from memory
    - Multiple instances of the same name can lead to unexpected results
  - title should be descriptive but doesn't need to be unique

TF1 Class

- 1D functions use the TF1 class
  - https://root.cern/doc/master/classTF1.html
- Initialize with name and function expression
  - Optional arguments for the range of x
- Function can use cmath, TMath or user-defined functions

```
TF1 *f1 = new TF1("f1","sin(x)",0,10);
TF1 *f2 = new TF1("f2","TMath::Cos(x)",0,10)
Double_t myFunc(double x) {return x+sin(x);}
TF1 *f3 = new TF1("f3","myFunc(x)",0,10)
```

TCanvas and Drawing

- Many ROOT classes allow you to draw plots
  - Histograms, functions, graphs, etc.
- Plots are drawn on a TCanvas object
  - If no canvas is active, a default canvas is created
  - Use constructor to define size or just use default values
  - Change active canvas with cd() command: c1->cd()
- Use the Draw() command to draw to the active canvas
  - Use "same" as an argument to draw multiple things to the same canvas
  - Drawing captures a snapshot of the plot and canvas is not updated automatically
- Save TCanvas using SaveAs(“<filename>”) method
  - Most image formats supported

Histograms Overview

- A histogram is a binned representation of data
  - Histograms defined by bin edges
  - Each entry is placed in the bin with the corresponding range
  - Entries can be added with different weights
- ROOT offers 1, 2 and 3 dimensional histograms
- Different classes are available to be optimized for different types of data
  - TH1F, TH1S, TH2I, TH3D, etc
- Note that in ROOT, histogram bin numbers are indexed from 1!!!

TH1

```
TH1F *h1 = new TH1F("h1","h1",20,0,10);
h1->Fill(6.2);
h1->Fill(3.4,0.7);
h1->Draw();
h1->GetEntries();
h1->Integral();
h1->GetBinContent(4);
h1->FillRandom("gaus",1000);
```

ROOT Macros

- C++ macros (*.C files) can be used to call available ROOT functions
- Main function needs to have the same name as macro
- Header files for used classes need to be explicitly included

mymacro.C:

```
#include <TH1F.h>
void myMacro() {
        TH1F *h1 = new TH1F("h1","h1",20,0,10);
        h1->Fill(6.7);
        return;
}
```

Running ROOT Macros

- Macros can be called through the CLING interpreter or compiled
  - CLING interprets C++ similar to the way python is interpreted
- Within root, execute a macro using:

  .x mymacro.C
- Or call using (for CLING interpreter):

  root mymacro.C
- Or with (to compile the code):

  root mymacro.C+