MIT HEP Starter Kit

Yutaro Iiyama

March 16, 2015

Before you begin

The goal of the starter kit (`https://github.com/yiiyama/starterkit.git`) is to introduce the basic tools and ideas in high-energy physics analyses to intereted learners. We hope you enjoy the exercises and become comfortable with the basics by the time you are done!

A successful high-energy physics experiment requires an extremely broad range of expertise (hence the extraordinary collaboration size). It is motivated by the theory of particle physics, which is arguably one of the most esoteric field of study with no obvious practical application in human society. But to execute the experiment, one must also solve highly nontrivial civil engineering problems (e.g. drilling a vertical shaft while freezing the shaft wall to stop groundwater from flooding in), build superconductive magnets, cryogenics (magnets must be cooled down), and ultra-high vacuum, develop cutting-edge particle detectors and electronics, and finally know the statistical methods to interpret the data and write a software to implement those methods. This package is intended to help you get started with the last bit, i.e., writing the software for HEP analyses.

Please note that this document will contain mostly background information and not much of examples and solutions. Source code templates will be provided, but you are encouraged to modify and rewrite them as much as possible – if things cease to work, you can always recover the original file by `$ git checkout HEAD` *`path_to_file`*.
However, be careful when issuing above command, since the command removes all of the changes you made.

Requirements

You need a command-line terminal, an SSH program, and a text editor.

Installation

You have to first log in to one of the HEP cluster machines (MIT CMS tier-3 computing cluster):

```
$ ssh user_name@t3btch***.mit.edu
```

If your local user name is the same as your account name at T3, you can ommit the part `user_name@`.

Once you are in one of the cluster machines, download the starter kit package:

```
$ git clone https://github.com/yiiyama/starterkit.git
```

Install script will execute the necessary initial setups:

```
$ cd starterkit
$ ./install.sh
```

This is a one-time procedure. No more installation is necessary in future sessions. However, you always have to set up the environment after each log in (including the first):

```
$ cd starterkit
$ source init.sh
```

Exercise 1. Plotting out distributions

We will familiarize ourselves with ROOT in this exercise. ROOT is a software library developed at CERN specifically for HEP analysis. It is written in C++ and is intended to be mixed with user applications, rather than to be used as a ready-made program, to create softwares for specific analysis purposes. Thus we would often be writing C++ codes including some of ROOT objects, and compiling them together with ROOT.

Compilation can be done with g++ or any other C++ compiler by letting the compiler know where to find the ROOT source code and the pre-compiled ROOT binary libraries. Alternatively, one can make ROOT itself to compile your code using its internal compiler (which is actually g++):

```
$ root -q your_program.cc+
```

The plus sign at the end of the file name tells ROOT to compile the program (sometimes also called macro).

If you call ROOT without the plus sign, instead of compiling and creating a binary, ROOT will try to "interpret" your program line-by-line as a script written in C++. ROOT has a built-in interpreter named CINT to do this. CINT is fairly reliable but has some limitations (mainly because C++ was never intended to be used as a scripting language), so this mode of operation is not recommended unless the macro is very simple.

If ROOT is called without any argument, it will enter the interactive

mode, where command-line inputs are interpreted by CINT line-by-line. ROOT will enter this mode also after the macro execution if you call ROOT without the `-q` option. This mode is useful when e.g. making small changes to plots.

Another popular way to use ROOT functionalities is to import the ROOT module to python. We will see a lot of this in the following.