Pythia Mini-Session

Using Pythia 8 on the UCD Linux server with ROOT

Manuel Calderón de la Barca Sánchez

UCDAVIS

STAR Collaboration

Setting up: Getting code

- In your home directory, create a working directory for the Pythia installation
  - **mkdir pythia**
- Download Pythia from the Pythia website.
  - **http://home.thep.lu.se/~torbjorn/Pythia.html**
  - **Note: Pythia 8.244 tested and works.**
  - **Zipped Tarball: pythia8244.tgz**
  - **After unzipping and extracting tarball, should have a directory called "pythia8244"**

Interlude before setup: Documentation

- Documentation area:
  - **pythia8244/share/Pythia8/htmldoc/**
- Web documentation:
  - **Open the file Index.html in browser**
- PDF documentation:
  - **pythia8244/share/Pythia8/pdfdoc/pythia8200.pdf**
- Of intereset:
  - **“Study output”**
    - **Pythia Classes to manipulate the generated particles, momenta, decays etc.**
  - **“Setup Run Tasks”**
    - **Setting up the collision and beam parameters**
    - **Turning on the relevant processes processes (e.g. heavy quark production)**
    - **Phase space cuts, couplings, scales**

Compiling Pythia (standalone)

- Read the README files!
- Basic installation:
  - **type ./configure**
  - **type make**
  - **The libraries are now built!**
  - **(In OSX, need to do this as su)**

Running the first example

- go to the examples directory
- There are about 60 programs already given as examples, mainXX.cc
- Check the README file (in the examples area)
- Start with main01.cc
  - **Check that it works:**
    - **Type: make main01**
    - **Type: ./main01.exe**

To use ROOT, configure step

- When configuring PYTHIA, instead of the simple configure command, use:
- `./configure --with-root=$ROOTSYS/ --with-root-bin=$ROOTSYS/bin/ --with-root-lib=$ROOTSYS/lib/ --with-root-include=$ROOTSYS/include/`
- `make`
  - To remake the Makefile.inc in the examples directory
- **This assumes that you have the $ROOTSYS environment variable correctly set**

To use ROOT: try example main91

- main91.cc : version of main01 that uses ROOT
- Makefile will use environment variables set in Makefile.inc
  - **These were created when configuring and compiling Pythia in the previous step, when including ROOT**
- Correct flags for main91 should work out of the box, but they don't for Pythia 8.244.
- Edit the Makefile, in the instructions for target "main91" (around line 144), in the line:
- $(CXX) $< -o $@ -w -I$(ROOT_INCLUDE) $(CXX_COMMON)\
- Remove the -I$(ROOT_INCLUDE), because that variable is no longer set. It is obtained by running inline the root-config script, invoked in the next line.

Example main91, continued

- Makefile makes use of root-config script to obtain all the needed ROOT libraries.
- After editing the Makefile as in the previous slide, compile by doing
  - `make main91`
- Running main91 example:
  - Type: `./main91`
  - A window appears in OSX, but histogram doesn't show. When clicking the window, histogram appears and then the program quits.
  - To fix, after the `mult->Draw()` line in `main91.cc`, add the lines
    - `gPad->Modified();`
    - `gPad->Update();`
- Also, note that the program writes a file called hist.root.
- You can simply open the output file:
  - `root -l hist.root`
  - `mult->Draw()`

A possible problem using ROOT:

- If you are getting the error when running about “cannot find libCore”, the linker is not finding the right library at runtime. This is not a ROOT problem, it is a linking problem.
- To use ROOT libraries with any standalone C++ program in nuclear, at runtime the linker needs to know where the path to shared libraries are.
  - **This is the `-L$(ROOTSYS)/lib` directory, which we specified at compile time.**
- To fix this runtime problem, you need to set up the environment variable LD_LIBRARY_PATH to include this directory.
- This can be done in your .bashrc (or .cshrc)
  - **For .bashrc, add:**

```
export LD_LIBRARY_PATH=".:/usr/include:/usr/lib64:/lib/ssa:$ROOTSYS/lib”
```

  - **For .cshrc, add:**

```
setenv LD_LIBRARY_PATH .:/usr/include:/usr/lib64:/lib/ssa:$ROOTSYS/lib
```

A closer look at the main91 example

- main91 (also main01) does the following:
  - **Turned on all QCD Hard processes: HardQCD:all = on**
  - **Sampled phase-space with pTHat > 20**
    - **Check web documentation**
    - **Phase space cuts, cuts in 2 → 2 processes**
    - **pTHatMin : The minimum invariant *pT***
  - **Sets the center-of-mass energy (Beams:eCM) at 14 TeV**
    - **There are no explicit sets to the beam particles, Pythia uses proton-proton collisions by default, now.**
  - **In main91, the output file and the TH1F Histogram to save the output of the charged multiplicity are created**
    - **in main01, the Pythia histogram is created**
  - **Makes a loop to generate 100 events**
  - **Inside loop:**
    - **Generates the next event, pythia.next(), checks for errors**
    - **Loops over all particles in the event record, counts all final state particles that are charged**
    - **fills the multiplicity histogram for the event**
  - **Prints the statistics of the event (cross section, e.g.)**
  - **Draws the root histogram on a separate canvas window.**
    - **Fix with gPad->Modified(); gPad->Update(); in order to see it.**

Key Modifications to use ROOT histograms

- Note that the program will need ROOT Includes, e.g.:

```
#include "TROOT.h"
#include "TFile.h"
#include "TH1.h"
```

- Add root histogram (instead of Pythia histogram)

```
TH1D* multHist = new TH1D("multHist","Multiplicity",100,-0.5,99.5);
```

  - Can also change pTHatMin to 1 for MinBias

- Fill histogram (instead of Pythia histogram)

```
multHist->Fill(nCharged);
```

- Write output

```
TFile* outFile = new TFile("pythiaOutputHistosTest.root","RECREATE");
  multHist->Write();
outFile->Close();
```

Your turn:

- Add histograms to study:
- Multiplicities of:
  - **Final state protons. Count both protons and antiprotons. Make one histogram for each.**
  - **Final state pions. Look for pi+, pi-, and pi0.**
  - **You will need to look at the Pythia Particle class and check the particle's id(). Look for the relevant id numbers in the PDG website, via the Monte Carlo particle id numbering.**
  - **Do you see final state pi0's? If not, try to see in the event record what happens to them. Hint: do they have "daughters"? What are they? How can we measure the "daughters"?**