ROOT Introduction

Claudia Giuliani

Albert-Ludwigs-Universität Freiburg

APP WiSe13-14, October 31st 2013
Outline

Introduction and Documentation

Classes

- 4-Vectors
- Graphs
- Histograms
- Fit
- Trees
What is ROOT?

- ROOT is a software for **data analysis** developed by CERN
- Used heavily by experimental particle physicists
- ROOT is based on C++ classes
- Many additional classes
- Useful for
  - data analysis
  - plotting
  - fitting
- ROOT [website](#)
What is useful for you:

- **Discovering ROOT**: some more informations
- **Users Guide**: printable version and explanation of the various ROOT classes
- **ROOTPrimer**: compact ROOT introduction (50 pages)
- **Reference Guide**: your best friend to find out the available methods of each class
- **Tutorials and HowTo’s**: to use if you need help or inspiration
Starting ROOT

- Installation
  - Windows and Mac: binaries [here](#)
  - Ubuntu: `apt-get install root-system` or `debian` files
  - Fedora: `yum install root`

- Start ROOT
  - Open Terminal and type `root` (-l to not show ROOT logo)
  - Using the arrows you can get the previously used code lines
  - To close application type `.q`
My First Macro

- Download and have a look at `myfirstMacro.cxx`
- Run it with root:
  - `start root`
  - `.x myfirstMacro.cxx`
  - `close root`
  - or all together: `root -q myfirstMacro.cxx`

```cpp
void myfirstMacro() {
  gROOT->Reset();
gROOT->SetStyle("Plain");

  std::cout << "This is my first macro" << std::endl;
  int num = 2;

  std::cout << "number defined num= " << num << std::endl;
  std::cout << "second power of " "num" is: " "num*num" std::endl;
  std::cout << "second power of " "num" with TMath function is: " "TMath::Power(num,2)" std::endl;

  std::cout << "Multiplication Table" std::endl;
  for (int i=0; i<11; ++i) {
    std::cout << num << "*" << i << "=" << num*i std::endl;
  }
}
```

```
hepbook33:ROOTIntro claudia$ root root [0] .x myfirstMacro.cxx
This is my first macro
number defined num=2
second power of 2 is: 4
second power of 2 with TMath function is: 4
Multiplication Table
2*0=0
2*1=2
2*2=4
2*3=6
2*4=8
2*5=10
2*6=12
2*7=14
2*8=16
2*9=18
2*10=20
```

```
root [1] .q
hepbook33:ROOTIntro claudia$
```
Some important classes

- **4-Vectors**: ROOT has the possibility to save the 4-vector of an object
  - TLorentzVector
  - Flexible object to save properties of a particle

- **Graphs**: given a pair \((x,y)\) draw a point
  - ROOT class TGraph
  - ROOT class TGraphErrors

- **Functions**: 
  - ROOT class TF1
  - Fit a histogram with a function

- **Histograms**: represent the probability of one event to happen, if you want to visualise it look here
  - ROOT Base class TH1
  - We will use mainly TH1F (1-d histograms with float values) and TH2F (2-d histograms with float values)

- **Trees**: save same information for all events. Easy to manipulate and retrieve saved information.
  - ROOT class TTree
How we proceed

- Try some easy commands interactively
- Try to understand and run some macros
- Then modify them
- Simple macros - thanks F. Bührer
  http://wwwhep.physik.unifreiburg.de/~cgiulan/AppliedParticlePhysics/ROOTIntroduction/
- More complex macros taken from ROOT tutorials
- You can find them in $ROOTSYS/tutorials
- You can use them for further exercises
Particle Physics: we want to study particles
Save the particle properties in a sensible way
Particle Energy-Momentum vector is an important quantity
ROOT gives us an easy and flexible tool: TLorentzVector
we can set, and later retrieve, the vector components
e.g. we have a particle with \( P_x=100 \) GeV, \( P_y=30 \) GeV, \( P_z=50 \) GeV and \( E=200 \) GeV

```cpp
TLorentzVector * particle = new TLorentzVector();
particle->SetPxPyPzE(100.,30.,50.,200.);
also possible: SetPtEtaPhiE(), SetPtEtaPhiM()
[ or also SetXYZT() for a space-time vector ]
particle->Pt() : to get the transverse momentum of the particle
particle->M() : to get the mass
```
Graph: plot a pair \((x,y)\), e.g. \(x\) and \(y\) coordinates of an object moving on a plane

ROOT class: TGraph

TCanvas: canvas where the plots are drawn

```c
{ 
    TCanvas *c1 = new TCanvas("c1","A Simple Graph Example",200,10,700,500);
    Double_t x[100], y[100];
    Int_t n = 20;
    for (Int_t i=0;i<n;i++) {
        x[i] = i*0.1;
        y[i] = 10*sin(x[i]+0.2);
    }
    gr = new TGraph(n,x,y);
    gr->Draw("AC*");
    return c1;
}
```
Histograms: TH1

- Download `dices.cxx`
- Open the macro and try to understand what is done
- Run it with: `root dices.cxx`

```cpp
void dices(){
    TRandom *Rnd = new TRandom(); // random number generator
    TH1F *myHisto = new TH1F("myHisto","Distribution",34,-0.5,13.5); //histogram definition
    // 34 bins from -0.5 to 13.5
    TCanvas *c1 = new TCanvas("c1","c1",800,600); //canvas definition
    for(int i=0; i<10000; i++){
        int dice1 = (int)(Rnd->Rndm()*6)+1; //Rndm generates random numbers between 0 and 1
        int dice2 = (int)(Rnd->Rndm()*6)+1;
        myHisto->Fill(dice1+dice2);
        cout << "<" << dice1 << "<" << dice2 << "=" << dice1+dice2 << endl;
        myHisto->Draw();
        Tstring title = "Hist"; //string ROOT type
        title << i; title = ".png";
        c1->SaveAs(title);
    }
}
myHisto->Draw();
c1->SaveAs("Hist_10000.png"); //equivalently ->Print()
myHisto->Scale1().myHisto->GetEntries();
c1->SaveAs("Hist_10000scaled.png");
```

Distribution

<table>
<thead>
<tr>
<th></th>
<th>Entries</th>
<th>Mean</th>
<th>RMS</th>
</tr>
</thead>
<tbody>
<tr>
<td>myHisto</td>
<td>10000</td>
<td>6.987</td>
<td>2.423</td>
</tr>
</tbody>
</table>

0  2  4  6  8  10  12
| 0  | 200 | 400 | 600 | 800 | 1000 | 1200 | 1400 | 1600 |

0  2  4  6  8  10  12
Fit a Histogram with GUI

Data Set: TH1F::my-Histo

Fit Function:
Type: gaus

Operation: Add

Selected:
gaus

Fit Settings:
Method: Chi-square

Fit Options:
Integral: Off
Best errors: No
All weights = 1
Empty bins, weights=1

Draw Options:
SAME
No drawing
Do not store/draw

FCN=0.00137742 FROM MIGRAD STATUS-CONVERGED 36 CALLS 37 TOTAL
EDM=3.30363e-07 STRATEGY= 1 ERROR MATRIX ACCURATE

<table>
<thead>
<tr>
<th>EXT PARAMETER</th>
<th>VALUE</th>
<th>ERROR</th>
<th>SIZE</th>
<th>STRATEGY</th>
<th>FIRST</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Constant</td>
<td>1.51548e-01</td>
<td>1.9064e-01</td>
<td>7.4153e-05</td>
<td>1.1700e-03</td>
<td></td>
</tr>
<tr>
<td>2 Mean</td>
<td>6.98605e+00</td>
<td>3.1298e+00</td>
<td>1.5276e-05</td>
<td>3.0703e-05</td>
<td></td>
</tr>
<tr>
<td>3 Sigma</td>
<td>2.75862e+00</td>
<td>2.9612e+00</td>
<td>1.4668e-04</td>
<td>2.3663e-03</td>
<td></td>
</tr>
</tbody>
</table>
Files and Trees: TFile and TTree

- Download Top.root
- To open the file: root Top.root
- To visualise the content of the file: TBrowser b