

ROOT

An Object-Oriented
Data Analysis Framework



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](#)

ROOT Website

ROOT

```
create the file, the tree and the browser  
File f1("root", "tree")  
Tree t1("t1", "a simple tree with 1000  
t1.Branch("pc", "Apc", "p");  
t1.Branch("pp", "Apc", "p");
```

Home What's New About Screenshots Download Documentation Support Forum Developers

Screenshots
Get a taste of ROOT's capabilities by sampling some screenshots.

Documentation
Go ahead and discover the inside scoop on how to fully utilize ROOT. Also, search the Reference Guide, the HowTo's and the user forums.

Discovering ROOT

Architectural Overview
Discovering ROOT
User's Guide
Reference Guide
Tutoriale
HowTo's
FAQ
Other Resources

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**

```
bash ... bash bash bash bash >>
hepbook33:~ claudia$ root
*****
*                               *
*      W E L C O M E to R O O T   *
*                               *
*   Version  5.34/10   29 August 2013  *
*                               *
*   You are welcome to visit our Web site
*       http://root.cern.ch
*                               *
*****

ROOT 5.34/10 (heads/v5-34-00-patches@v5-34-10-5-g0e8bac8, Sep 04 2013,
11:52:19 on macosx64)

CINT/ROOT C/C++ Interpreter version 5.18.00, July 2, 2010
Type ? for help. Commands must be C++ statements.
Enclose multiple statements between { }.
root [0] 7+5
(const int)12
root [1] 12/4
(const int)3
root [2] 12./4.
(const double)3.0000000000000000e+00
root [3] int b = 2
root [4] b * 4
(int)8
root [5] ++b
(int)3
root [6] .q
hepbook33:~ claudia$
```

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`

```
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;  
    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. Bühner
<http://wwwhep.physik.uni-freiburg.de/~cgiulian/AppliedParticlePhysics/ROOTIntroduction/>
- ▶ More complex macros taken from ROOT tutorials
- ▶ You can find them in \$ROOTSYS/tutorials
- ▶ You can use them for further exercises

4-Vectors: TLorentzVector

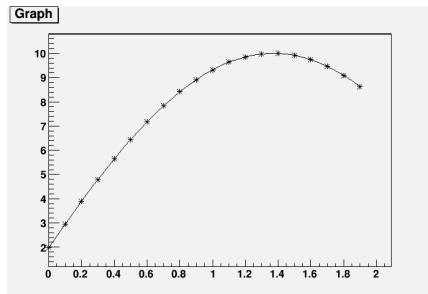
- ▶ Particle Physics: we want to study particles
- ▶ Save the particle properties in a sensible way
- ▶ Particle Energy-Momentum vector is an important quantity
- ▶ ROOT gives us an easy and flexible tool: TLorentzVector
- ▶ we can set, and later retrieve, the vector components
- ▶ e.g. we have a particle with $P_x=100$ GeV, $P_y=30$ GeV, $P_z=50$ GeV and $E=200$ GeV

```
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
```

TGraph

- ▶ 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

```
{  
  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;  
}
```



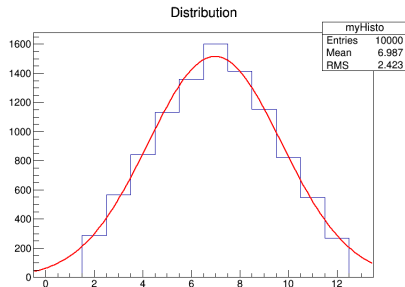
Histograms: TH1

- ▶ Download [dices.cxx](#)
- ▶ Open the macro and try to understand what is done
- ▶ Run it with: `root dices.cxx`

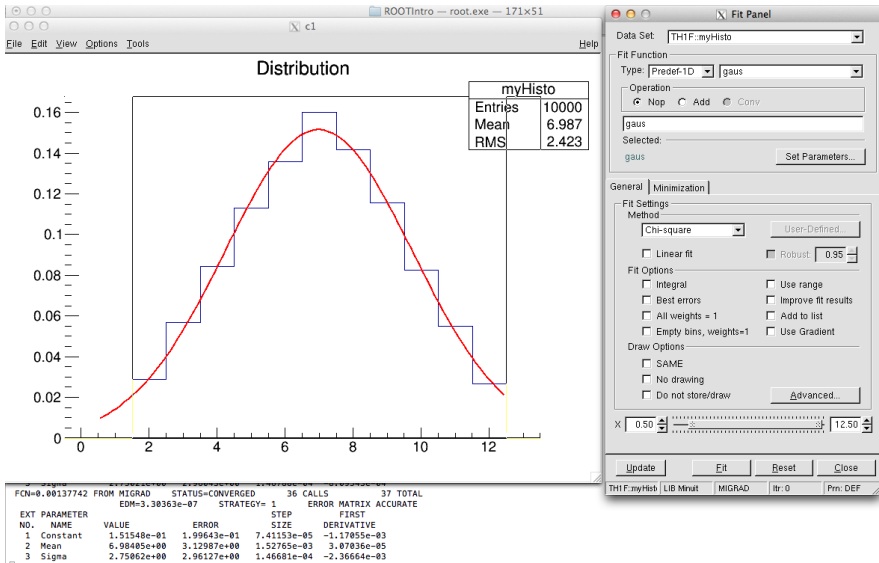
```
void dicesO{
  TRandom3 *rnd = new TRandom3(); // random number generator
  TH1F* myHist=new TH1F("myHisto", "Distribution",14,-0.5,13.5); //histogram definition:
  14 bins from -0.5 to 13.5
  TCanvas *c1=new TCanvas("c1","c1",800,600); //canvas definition
  for(int i=0; i<10000; i++){
    int dice1=(int)(rnd->Rndm()*6)+1; //Rndm generates random numbers between 0 and 1
    int dice2=(int)(rnd->Rndm()*6)+1;

    myHist->Fill(dice1+dice2);
    if(i<10){
      cout <<dice1<<" " <<dice2<<" " <<dice1+dice2<<endl;
      myHist->Draw();
      TString title="hist_"; //string ROOT type
      title+=i; title+=".png";
      c1->SaveAs(title);
    }
  }
  myHist->Draw();
  c1->SaveAs("hist_10000.png"); //equivalently ->Print()
  myHist->Scale(1./myHist->GetEntries());
  c1->SaveAs("hist_10000scaled.png");
}
```

-U:--- dices.cxx All L9 (C++/L Abbrev)



Fit a Histogram with GUI



Files and Trees: TFile and TTree

- ▶ Download [Top.root](#)
- ▶ To open the file: root Top.root
- ▶ To visualise the content of the file: TBrowse b

