ROOT introduction
Functions, basic plot options

Anna Simon

October 14, 2015

Outline for the rest of the semester

GOAL: Write a complete data analysis code that uses ROOT libraries.

- intro, functions, macros, TLegend
- histograms, random number generators
- fitting histograms
- TGraph
- writing histograms to a file, reading from a file
- trees
- creating an analysis macro (MakeClass())
- using ROOT within a C++ code, including ROOT libraries, compiling the code
- TChain, TFriends
- TStyle

ROOT

- ROOT - open source project available under GNU LGPL licence
- set of libraries for visualisation, statistical studies, data reduction and multivariate techniques
- comes with CINT interpreter (versions ≤ 5) or cling (version 6) - allows for using C/C++ commands directly from the command line
- understands macros - not compiled programs $\rightarrow$ quick prototyping, easy to reuse/modify scripts for data analysis
- set of mathematical libraries and tools needed for event reconstruction, simulation and statistical data analysis
- source code, installation instructions, manuals and examples are available at root.cern.ch

ROOT prompt

To launch ROOT on CRC machines:

- login to a CRC machine (regulus or canopus):

```
ssh -Y netID@regulus.crc.nd.edu
```

- load the ROOT module:

```
module load root
```

- start ROOT

```
root
```

- quit ROOT

```
.q
```

- end CRC cession

```
exit
```

NOTE: once you ssh to a CRC machine you are operating in a linux environment when in the terminal or any window opened through that terminal. All the keyboard shortcuts are now linux-like.

ROOT prompt - a pocket calculator

```
root [2] sqrt(2)
(const double)1.4142135623730951e+00

root [3] double var = 7
root [4] var
(double)7.0000000000000000e+00

root [5] sin(var)
(const double)6.5698659871878906e-01

root [6] log(var)
(const double)1.9459101490553132e+00
```

Macros

- Each command that is passed to the ROOT command line can be written into a "macro" file that is executed from the ROOT command line
- the sequence of commands in a macro file has to be enclosed in curly brackets {}

```
{
double var = 3;
cout << "This is my variable: \t" << var << endl;
var = var + .1415;
cout << "Here is a sine of that variable: " << sin(var) << endl;
}
```

- to execute the macro within ROOT

```
.x myMacro.C
```

- NOTE: the code is not compiled, it will run until it encouters a problem and then throw an error

ROOT macros

- macro can contain functions

```
1 int AddValues(int x, int y){
2 return x+y;
3 }
4
5 int SubtractValues(int x, int y){
6 return x-y;
7 }
```

- such macro can be loaded via ROOT prompt and then the functions are accessible from the terminal:

```
1 root [3] .L myFunctions.C
2 root [4] AddValues(3,2)
3 (int)5
4 root [5] SubtractValues(4,1)
5 (int)3
6 root [6] .U myFunctions.C
```

Useful prompt commands:

```
root [1] .q // quit ROOT
/* if it doesn't work try:
        .qqq        : quit cint - mandatory
        .qqqqq      : exit process immediately
        .qqqqqqq    : abort process
*/
root [2] .? // list all available commands

root [3] .! <OS_command> // access to the shell of the OS

root [4] .x <filename> // execute a macro

root [5] .L <filename> // load a macro
```

Functions (TF1)

- define a function

```
1 TF1 *f1 = new TF1("f1","sin(x)/x",0,10);
```

- draw a function

```
1 root [3] f1->Draw();
```

- A lot of options available for the function from the command line, e.g.:

```
1 f1->Eval(3);
2 f1->Integral(1,7);
3 f1->DrawDeriviative();
```

Editing the plot

- Context menu available to change the properties of the plot
- Check: View/Toolbar View/Event Statusbar View/Editor
- Latex within Toolbar: #gamma, #alpha, ^{3} _{4}, #frac{}{}, #splitline{topline}{bottomline}
- TLatex

```
1  TLatex *latex = new TLatex (5,0.3,"#splitline{top}{bottom}");
2  latex->Draw()
```

(5,0.3 → x,y coordinates of the text)

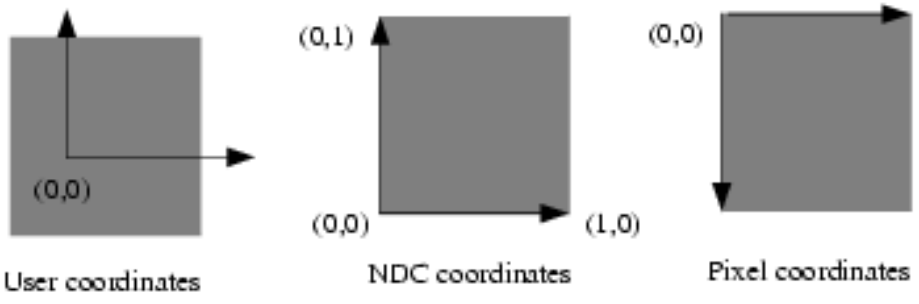
Editing the plot

Each option from the context menu can be accessed via the command line. The index corresponding to a given line style/color can be found in the Editor toolbar or here: http://root.cern.ch/root/html/TAttLine.html

```
f1->SetLineColor(3);
f1->SetLineStyle(2);
f1->GetXaxis()->SetTitle("this is an x-axis");
f1->SetTitle("my ROOT plot");
```

To build a legend for the plot:

```
TLegend *leg = new TLegend(0.7,0.7,1,1);
leg->AddEntry("fSum","fSum","l");
leg->Draw();
```



Functions with parameters

- parameters are denoted with [0], [1], ... Default parameter names are: p0, p1, ...
- numbering of the parameters starts with zero
- all the parameters are by default initialized to zero
- to define a function with parameters:

```
TF1 *f2 = new TF1("f2","[0]*sin([1]*x)/x,0,10)
```

- to set the value of the parameter

```
f2->SetParameter(0,2); //sets parameter [0]
f2->SetParameter(1,3); //sets parameter [1]
f2->SetParameters(2,3); //sets the parameters: [0]=2, [1]=3
```

- some predefined functions exist; gaus (3-param), expo (2-param), pol$N$ ([0] + [1]*x + ... [N-1]*x^{N-1}), landau
- formula can be created using any math function available in ROOT (e.g. from TMath library)

TMath

TMath: a namespace providing the following functionality (http://root.cern.ch/root/htmldoc/TMath.html):

- Numerical constants.
- Trigonometric and elementary mathematical functions.
- Functions to work with arrays and collections (e.g sort, min max of arrays,...)
- Statistic Functions (e.g. Gaus())
  ```
  TMath::Gaus( x, mean, sigma, norm=kFalse);
  ```
  Calculate a gaussian function with mean and sigma. If norm=kTRUE (default is kFALSE) the result is divided by sqrt(2*Pi)*sigma.
- Special Mathematical Functions (e.g. Bessel functions) – For more details, see the reference documentation of TMath.

Exercise

Your first macro

Complete the following tasks within a macro file (test the commands from the command line first if needed):

- Define three functions (all in the range 0 to 20) and set their parameters:
  fPeak1 - Gaussian: area=30, mean=5, sigma=1
  fPeak2 - Gaussian: area=20, mean=15, sigma=1,
  fBgrd - linear: slope=-2, y-intercept=30
- Define fSum - a function that is a sum of the fPeak1, fPeak2 and fBgrd
- Set the line style to dashed for the peaks and to dotted for fBgrd
- Set the line color for the peaks to green and blue, and to black for fSum
- Draw all four functions in the same canvas
- Create a legend for the plot
- Print out to the screen the integral (0,20), derivative (x=6), fSum value for x= 15