Introdução ao ROOT

Professores: Sandro Fonseca, Eliza da Costa, Maurício Thiel   *Name:* Eduarda de Barros Silva

Lista 3

EXERCICIO 1

Create a function with parameters, $p_0 * \sin (p_1 * x) / x$, and also draw it for different parameter values. Set the colour of the parametric function to blue. After having drawn the function, compute for the parameter values ($p_0 = 1$, $p_1 = 2$):

    a. Function value for x=1;

    b. Function derivative for x=1;

    c. Integral of the function between 0 and 3:

```cpp
#include "TF1.h"
void plotFunction()
 {
    TF1 * f1 = new TF1("f1","sin(x)/x",0,10);
    f1->Draw();

    TF1 * fp = new TF1("fp","[0]*sin([1]*x)/x",0,10);
    fp->SetParameters(1,2);
    fp->Draw("same");
    fp->SetLineColor(kBlue);

    f1->SetMaximum(2);
    f1->SetMinimum(-2);

    std::cout << "f(x) para x = 1: " << fp->Eval(1.) << std::endl;
    std::cout << "Derivada de f(x) para x = 1: " << fp->Derivative(1.) << std::endl;
    std::cout << "Integral de f(x) definida em (0,3): " << fp->Integral(0,3) << std::
        endl;
 }
```
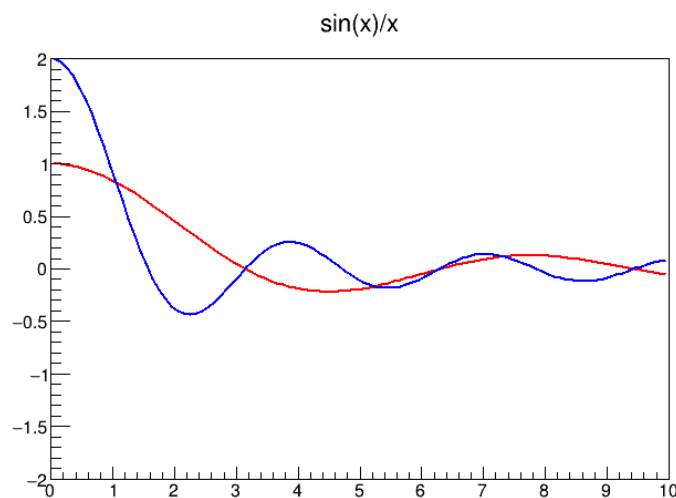

Figura 1: Histograma 1

EXERCICIO 2

Suppose you have this set of points defined in the attached file graphdata.txt. Plot these points using the TGraph class. Use as marker point a black box. Looking at the possible options for drawing the TGraph in TGraphPainter, plot a line connecting the points. Make a TGraphError and display it by using the ttached data set, graphdata_error.txt, containing error in x and y.

```
1  void plotGraph()
2   {
3     TGraph * g = new TGraph("graphdata.txt");
4     g->Draw("AP");
5     g->SetMarkerStyle(21);
6   }
```
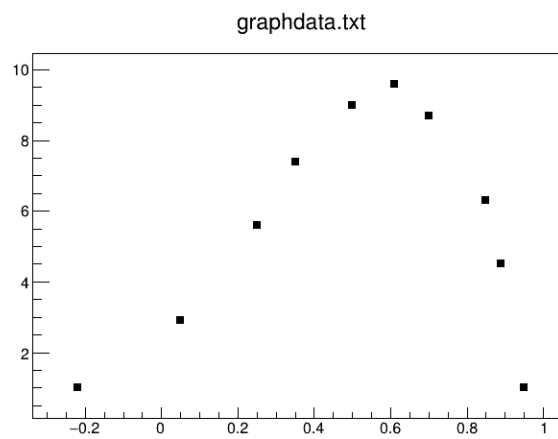

Figura 2: Histograma 2

```
1  void plotGraphError()
2   {
3     TGraphErrors * g = new TGraphErrors("graphdata_error.txt");
4     g->Draw("AP");
5     g->SetMarkerStyle(21);
6   }
```
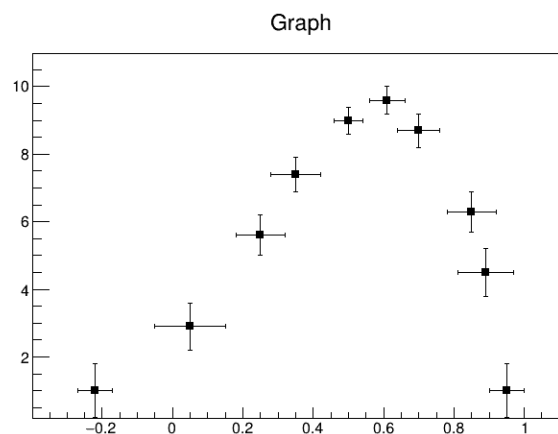

Figura 3: Histograma 3

EXERCICIO 3

Create a one-dimensional histogram with 50 bins between 0 to 10, and fill it with 10000 gaussian distributed random numbers with mean 5 and sigma 2. Plot the histogram and, looking at the documentation in the THistPainter, show in the statistic box the number of entries, the mean, the RMS, the integral of the histogram, the number of underflows, the number of overflows, the skewness and the kurtosis.

```cpp
#include "TH1.h"
#include "TRandom.h"
#include "TStyle.h"

void plotHistogram()

{
  TH1D * h1 = new TH1D("h1","h1",50,0,10);
  for (int i = 0; i < 10000; ++i)

 {
  double x = gRandom->Gaus(5,2);
  h1->Fill(x);
 }

  h1->Draw();
  gStyle->SetOptStat(111111110);
}
```
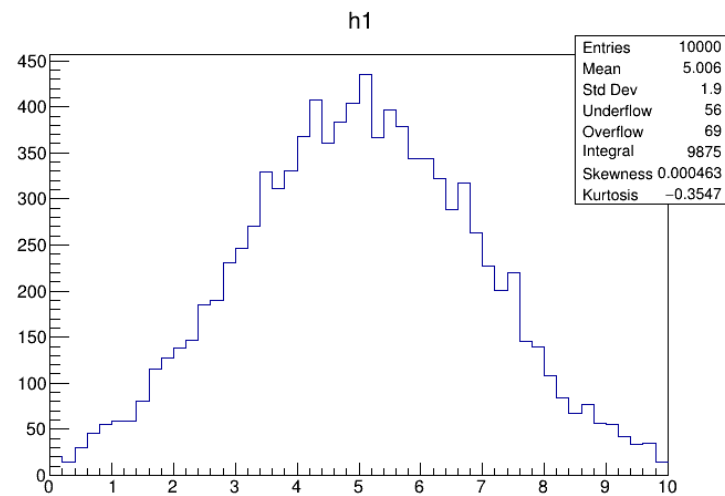

Figura 4: Histograma 4

EXERCICIO 4

Using the tree contained in tree.root make a distribution of the total momentum of each whose beam energy was outside of the mean by more than 0.2. Use TCut objects to make your events selections. Project this distribution into a histogram, draw it and save it to a file.

```
1  root [0] TFile *f1 = new TFile("tree.root");
2  root [1] f1->ls();
3        TFile**          tree.root
4         TFile*          tree.root
5          KEY: TTree     tree1;1 Reconstruction ntuple
6  root [2] TTree *mytree = (TTree *)f1->Get("tree1");
7  root [3] mytree->Print();
8  root [4] TCanvas *c2 = new TCanvas("c2", "Tree canvas", 300, 600);
9  root [5] c2->Divide(1,2);
10 root [6] c2->cd(1);
11 root [7] gStyle->SetOptStat(1);
12 root [8] mytree->Draw("ebeam");
13 root [9] c2->cd(2);
14 root [10] mytree->Draw("px:py", "ebeam>150.0");
15 root [11] TCut*px_plane = new TCut("px/log(px*2 + py**2) > 0.20");
16 root [12] TCut*py_plane = new TCut("py/log(px*2 + py**2) > 0.20");
17 root [14] mytree->Draw("ebeam", *px_plane && *py_plane);
```
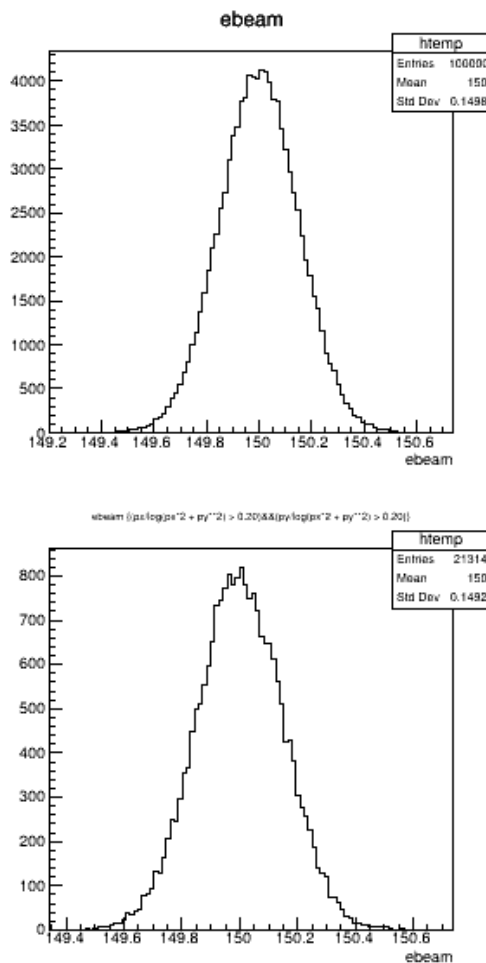
Figura 5: Histograma 5

Observação

Foi pedido para fazer a instalação do root, segue um vídeo da aluna fazendo a instalação do root:
https://www.youtube.com/watch?v=45SS4IvUODE&list=PLtoL3h1NQ-yqed4XFMo4E9m_AskXq5nPs

Segue também um vídeo da aluna fazendo um histograma simples:
https://www.youtube.com/watch?v=Znr6XUslJr0&list=PLtoL3h1NQ-yqed4XFMo4E9m_AskXq5nPs&index=11