

# Exercicios da aula 3 de estatística - ROOT

Sérgio da Silva dos Santos Júnior

Professores: Dilson Damião, Eliza Melo e Mauricio Thiel

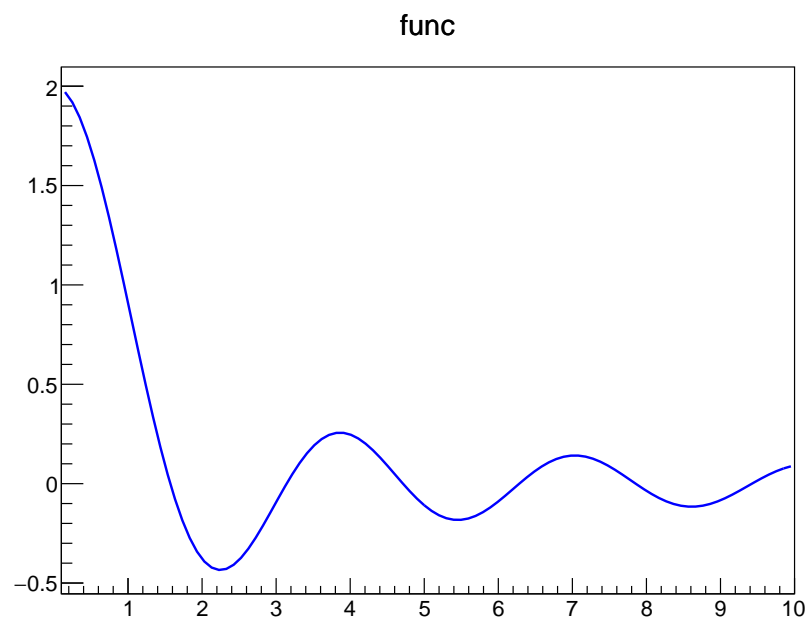
2024/2

# 1 Exercício 1

O código para o exercício foi:

```
1 #include <iostream>
2 #include <TCanvas.h>
3 #include <TF1.h>
4 #include <TGraph.h>
5 #include <TMath.h>
6 #include <TRandom.h>
7
8 double f(double* x, double* par) {
9     double p0 = par[0];
10    double p1 = par[1];
11    return (p0 * TMath::Sin(p1 * x[0])) / x[0];
12 }
13
14 void integral() {
15     double p0 = 1;
16     double p1 = 2;
17
18     // Criar a funcao
19     TF1 *func = new TF1("func", f, 0.1, 10, 2);
20     func->SetParameters(p0, p1);
21
22     // Criar um canvas para o grafico
23     TCanvas *c1 = new TCanvas("c1", "Grafico da funcao", 800, 600);
24     func->SetLineColor(kBlue);
25     func->Draw();
26
27     // a) Valor da funcao para x=1
28     double x = 1;
29     double valorFuncao = func->Eval(x);
30
31     // b) Derivada da funcao para x=1
32     double derivadaFuncao = func->Derivative(x);
33
34     // c) Integral da funcao entre 0 e 3
35     double integralFuncao = func->Integral(0.1, 3);
36
37     std::cout << "a) Valor da funcao para x=1: " << valorFuncao <<
38     std::endl;
39     std::cout << "b) Derivada da funcao para x=1: " <<
40     derivadaFuncao << std::endl;
41     std::cout << "c) Integral da funcao entre 0 e 3: " <<
42     integralFuncao << std::endl;
43
44     c1->SaveAs("funcao.pdf");
45 }
```

Executando o código, obtivemos como resultado:



E valores obtidos no terminal:

- a) Valor da função para  $x=1$ : 0.909297
- b) Derivada da função para  $x=1$ : -1.74159
- c) Integral da função entre 0 e 3: 1.22513

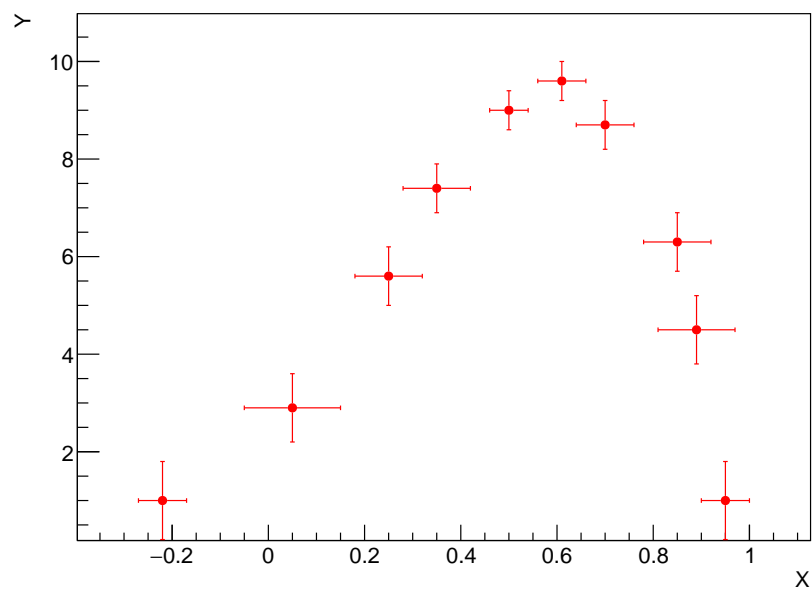
## 2 Exercício 2

O código para o exercício foi:

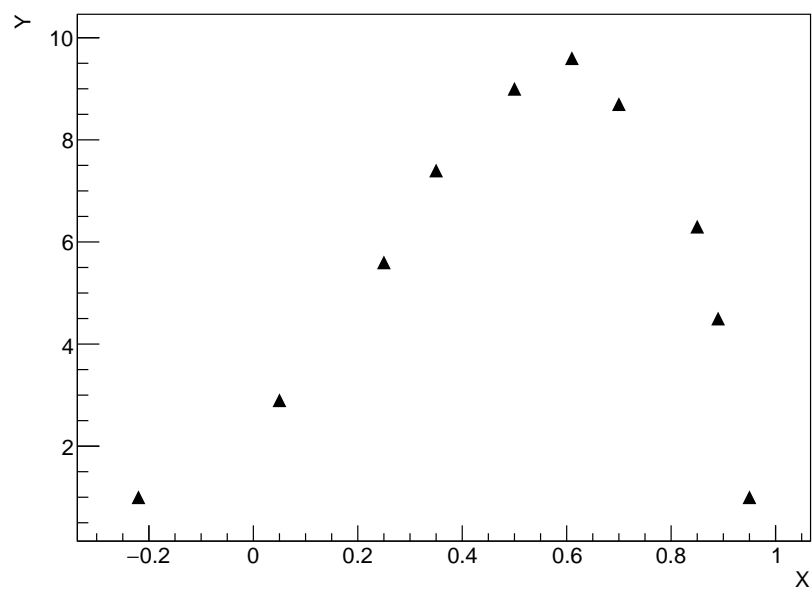
```
1 #include <iostream>
2 #include <TCanvas.h>
3 #include <TGraph.h>
4 #include <TGraphErrors.h>
5
6 void graphdata() {
7
8     TCanvas *c1 = new TCanvas("c1", "Grafico de Pontos", 800, 600);
9
10    TGraph *graph = new TGraph("graphdata.txt");
11    graph->SetMarkerStyle(22);
12    graph->SetMarkerSize(1.5);
13    graph->SetLineColor(kBlue);
14    graph->SetTitle("Grafico de Dados;X;Y");
15    graph->Draw("AP");
16
17    c1->SaveAs("grafico_pontos.pdf");
18
19    TCanvas *c2 = new TCanvas("c2", "Grafico de Erros", 800, 600);
20
21    TGraphErrors *graphErrors = new TGraphErrors("graphdata_error.
22    txt");
23    graphErrors->SetMarkerStyle(20);
24    graphErrors->SetMarkerColor(kRed);
25    graphErrors->SetLineColor(kRed);
26    graphErrors->SetTitle("Grafico de Erros;X;Y");
27    graphErrors->Draw("AP");
28
29    c2->SaveAs("grafico_erros.pdf");
30 }
```

Executando o código, obtivemos como resultado:

**Grafico de Erros**



**Grafico de Dados**

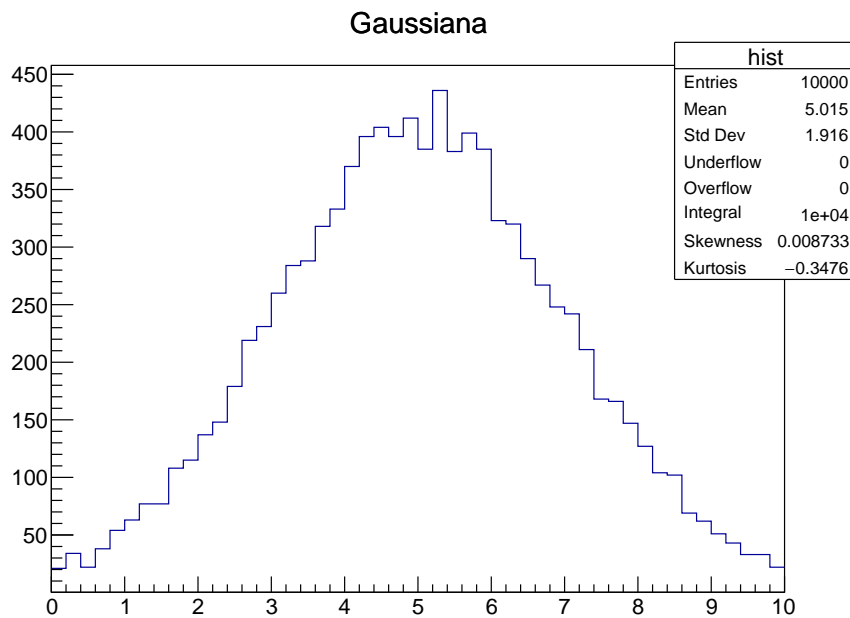


### 3 Exercício 3

O código para o exercício foi:

```
1 #include "TH1.h"
2 #include "TF1.h"
3 #include "TCanvas.h"
4
5 void histograma() {
6   TH1F * histGaus = new TH1F (" hist ", " Gaussiana ", 50 , 0 , 10) ;
7
8   TF1 * gaussiana = new TF1 (" gaussiana ", " gauss ", 0 , 10) ;
9   gaussiana->SetParameters (1 , 5 , 2) ; // amplitude , media e
      desvio p a d r o
10  histGaus->FillRandom (" gaussiana ", 10000) ;
11  gStyle->SetOptStat (11111111) ;
12
13  TCanvas *c1 = new TCanvas ("c1", " Gauss ", 800 , 600) ;
14  histGaus->SetMarkerStyle (21) ;
15  histGaus->SetMarkerColor (1) ;
16
17  histGaus->Draw () ;
18      c1->SaveAs("histograma.pdf");
19 }
```

Executando o código, obtivemos como resultado:

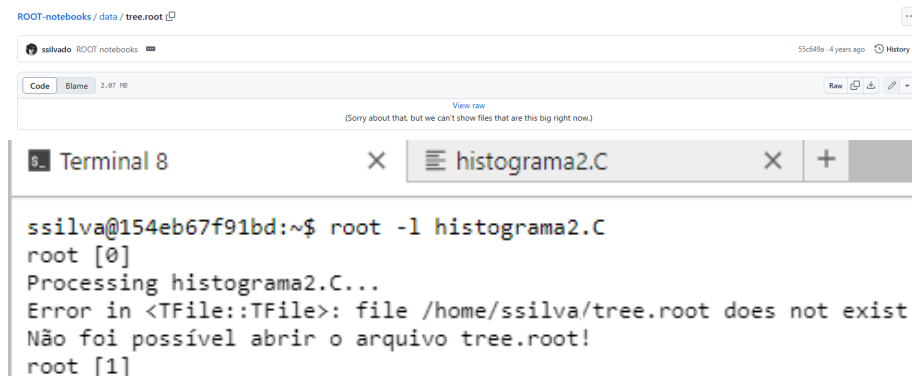


## 4 Exercício 4

O código feito para o exercício foi:

```
1 #include <TFile.h>
2 #include <TTree.h>
3 #include <TH1F.h>
4 #include <TCanvas.h>
5 #include <TCut.h>
6
7 void histograma2() {
8
9     TFile *file = TFile::Open("tree.root");
10
11     if (!file || file->IsZombie()) {
12         std::cerr << "Nao foi possivel abrir o arquivo tree.root!"
13         << std::endl;
14         return;
15     }
16
17     TTree *tree = (TTree*)file->Get("tree");
18     if (!tree) {
19         std::cerr << "Nao foi possivel acessar tree!" << std::endl;
20         return;
21     }
22
23     TCut cut = "abs(beamEnergy - mean(beamEnergy)) > 0.2";
24
25     TH1F *hist = new TH1F("momentumHist", "Distribuicao do Momento
26     Total", 100, 0, 10);
27
28     tree->Draw("momentum >> momentumHist", cut);
29
30     TCanvas *c1 = new TCanvas("c1", "Distribuicao do Momento Total"
31     , 800, 600);
32     hist->Draw();
33
34     TFile *outFile = new TFile("resultado_momentum.root", "RECREATE
35     ");
36     hist->Write();
37     outFile->Close();
38
39     file->Close();
40
41     std::cout << "Analise concluida e histograma salvo como
42     resultado_momentum.root." << std::endl;
43
44     c1->SaveAs("histograma2.pdf");
45 }
```

Porém, não consegui concluir a questão por erro no "tree.root" que não consegui resolver. O arquivo não era mostrado e, mesmo baixando, o terminal não reconheceu. A princípio, baixei normalmente e executei, com o terminal alegando que o arquivo não existe.



The screenshot shows a JupyterLab interface. At the top, there's a breadcrumb navigation bar: "ROOT-notebooks / data / tree.root". Below it, a header bar shows the user "ssilva", the notebook name "ROOT-notebooks", and a timestamp "55c48a 4 years ago" with a "History" link. The main area is divided into two panes. The left pane is a "Code" editor showing a file named "tree.root" with a "Blame" button and a timestamp "2:07 PM". The right pane is a "Terminal" window titled "Terminal 8" with a tab for "histograma2.C". The terminal shows the following output:

```
ssilva@154eb67f91bd:~$ root -l histograma2.C
root [0]
Processing histograma2.C...
Error in <TFile::TFile>: file /home/ssilva/tree.root does not exist
Não foi possível abrir o arquivo tree.root!
root [1]
```

E, seguida, coloquei "tree.root" no JupyterLab e, novamente, não obtive sucesso.

```
ssilva@154eb67f91bd:~$ root -l histograma2.C
root [0]
Processing histograma2.C...
Não foi possível acessar tree!
root [1]
```

Não sei dizer se estou esquecendo de fazer algo para que o código funcione corretamente, porém não consegui obter sucesso nesta questão.