Introdução à Análise de dados em FAE **(19/04/2024)**

Lista de exercícios 3

Professores: Sandro Fonseca, Eliza Melo, Maurício Thiel e Diego Torres *Name:* Miguel Lopes

EXERCÍCIO 0

O probelma 0 pedia para usar a ntupla encontrada em https://opendata.cern.ch/record/12353, e usar o MakeClass para fazer histogramas de algumas distribuições. Infelizmente não consegui baixar o arquivo, nem utilizando o wget no terminal.

EXERCÍCIO 1

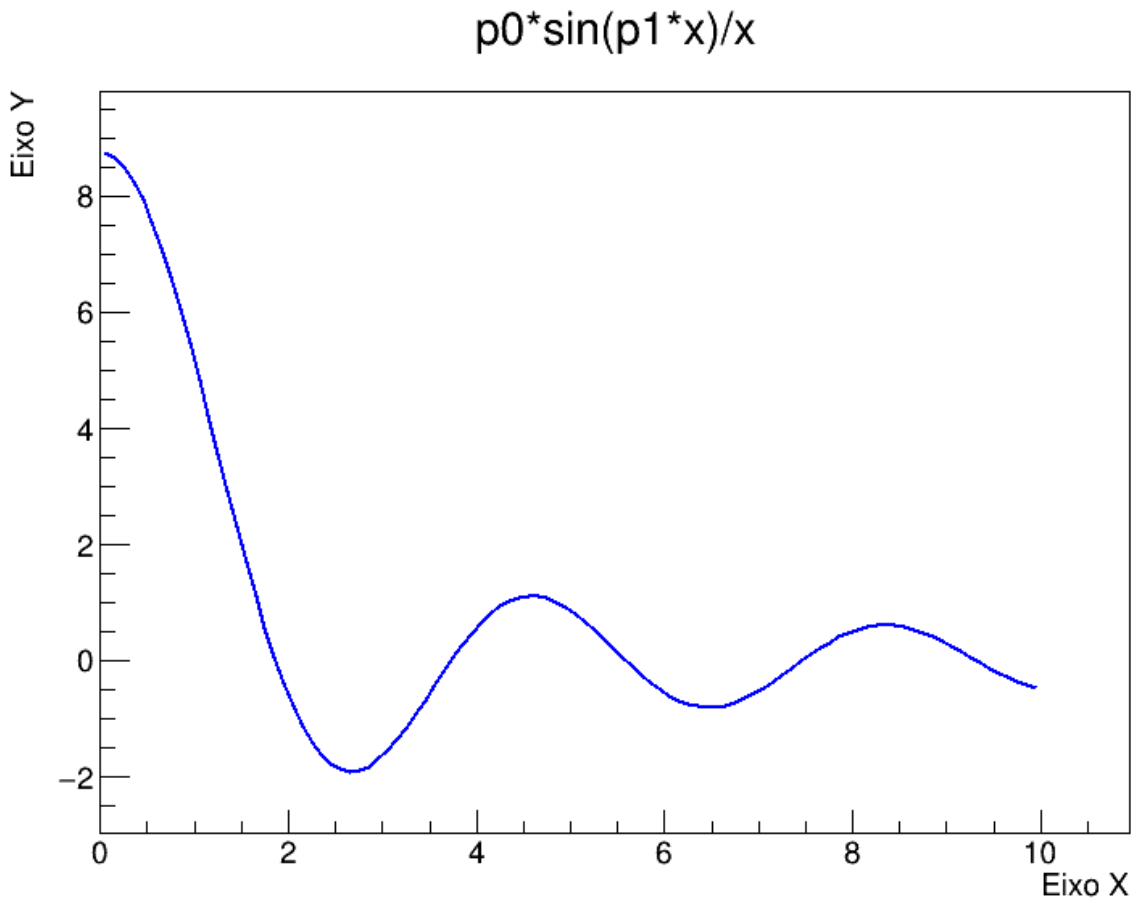
Create a function with parameters, p0 * sin (p1 * x) / x, and also draw it for different parameter values. Set the colour of the parametric function to blue. After having drawn the function, compute for the parameter values (p0 = 1, p1 = 2):

a : Function value for x=1
b : Function derivative for x=1
c : Integral of the function between 0 and 3

Para a primeira questão elaborei esse código:

```
1
2  #include <iostream>
3  #include <cmath>
4  #include "TCanvas.h"
5  #include "TF1.h"
6  #include "TMath.h"
7  #include "TGraph.h"
8
9  using namespace std;
10
11 Double_t Fseno(Double_t *x, Double_t *par) {
12     if (x[0] == 0) return 0;
13     return par[0] * sin(par[1] * x[0]) / x[0];
14 }
15
16 void Q1() {
17     TCanvas *canvas = new TCanvas("canvas", "Function Plot", 800, 600);
18
19     Double_t randomN1 = gRandom->Uniform(0,10);
20     Double_t randomN2 = gRandom->Uniform(0,10);
21
22     TF1 *func = new TF1("func", Fseno, 0., 10., 3);
23     func->SetParameters(randomN1, randomN2);
24
25     TGraph *graph = new TGraph(func);
26     graph->SetTitle("p0*sin(p1*x)/x");
27     graph->GetXaxis()->SetTitle("Eixo X");
28     graph->GetYaxis()->SetTitle("Eixo Y");
29
30     canvas->cd();
31     graph->Draw("AL");
32
```

```
    func->SetParameters(1, 2);

    Double_t xValue = 1.0;
    Double_t ValorFuncao = func->Eval(xValue);
    Double_t derivada = func->Derivative(xValue);
    Double_t integral = func->Integral(0, 3);

    cout << "Fun   o com o valor de x=1: " << ValorFuncao << endl;
    cout << "Derivada com x=1: " << derivada << endl;
    cout << "Integral entre 0 e 3: " << integral << endl;

    return 0;
}
```



Histograma 1: f(x) = $p0 * sin(p1 * x)/x$

E como saída do terminal obtive com os valores expecificos de x:

```

Funcao com o valor de x=1: 0.909297
Derivada com x=1: -1.74159
Integral entre 0 e 3: 1.42469
```

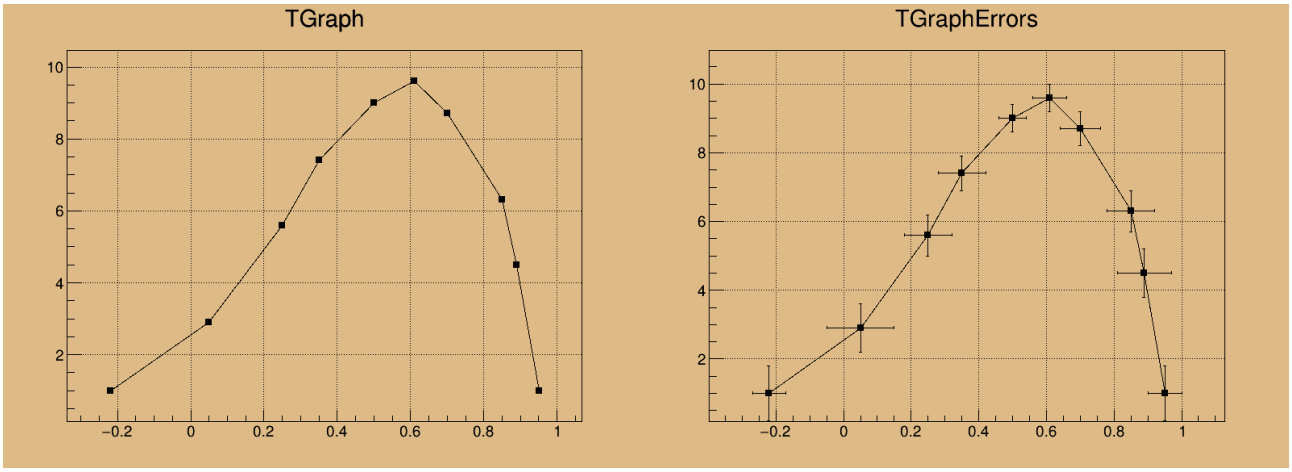
EXERCÍCIO 2

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 attached data set, graphdata$_e$*rror.txt*, *containingerrorinxandy*.

Meu código foi:

```
#include <TGraph.h>
#include <TGraphErrors.h>
#include <TCanvas.h>
#include <TFile.h>
#include <iostream>
#include "RtypesCore.h"

void Q2() {

    TGraph *graph = new TGraph();

    fstream file1;
    file1.open("graphdata.txt", ios::in);

    while(1)
    {
        double x, y;
        file1 >> x >> y;
        graph->SetPoint(graph->GetN(), x, y);
        if(file1.eof()) break;
    }
    file1.close();

    TCanvas *c1 = new TCanvas("c1", "TGraph", 800, 600);
    graph->SetMarkerStyle(21);
    graph->SetMarkerColor(1);
    c1->SetFillColor(42);
    c1->SetGrid();
    c1->GetFrame()->SetFillColor(21);
    c1->GetFrame()->SetBorderSize(12);
    graph->SetTitle("TGraph");
    graph->Draw("APL"); // A for markers, P for connecting lines, L for line color

//-------------------------------------------------------------------

    std::ifstream arquivo("graphdata_erro2.txt");
    if (!arquivo) {
        std::cerr << "Erro ao abrir o arquivo." << std::endl;
        return 1;
    }

    std::vector<float> coluna1;
    std::vector<float> coluna2;
    std::vector<float> coluna3;
    std::vector<float> coluna4;
    float valor1, valor2, valor3, valor4;

    while (arquivo >> valor1 >> valor2 >> valor3 >> valor4) {
        coluna1.push_back(valor1);
        coluna2.push_back(valor2);
        coluna3.push_back(valor3);
        coluna4.push_back(valor4);
    }
    arquivo.close();

    float x[coluna1.size()];
    float y[coluna2.size()];
    float ex[coluna3.size()];
    float ey[coluna4.size()];
    std::copy(coluna1.begin(), coluna1.end(), x);
    std::copy(coluna2.begin(), coluna2.end(), y);
    std::copy(coluna3.begin(), coluna3.end(), ex);
    std::copy(coluna4.begin(), coluna4.end(), ey);
```

```
64
65
66      auto c2 = new TCanvas("c2","TGraphErrors",800,600);
67      const Int_t n = 10;
68      auto gr = new TGraphErrors(n,x,y,ex,ey);
69      c2->SetFillColor(42);
70      c2->SetGrid();
71      c2->GetFrame()->SetFillColor(21);
72      c2->GetFrame()->SetBorderSize(12);
73      gr->SetTitle("TGraphErrors");
74      gr->SetMarkerColor(1);
75      gr->SetMarkerStyle(21);
76      gr->Draw("ALP");
77  }
```



(a) (b)

a Histograma 2: TGraph. b Histograma 3: TGraphErrors.

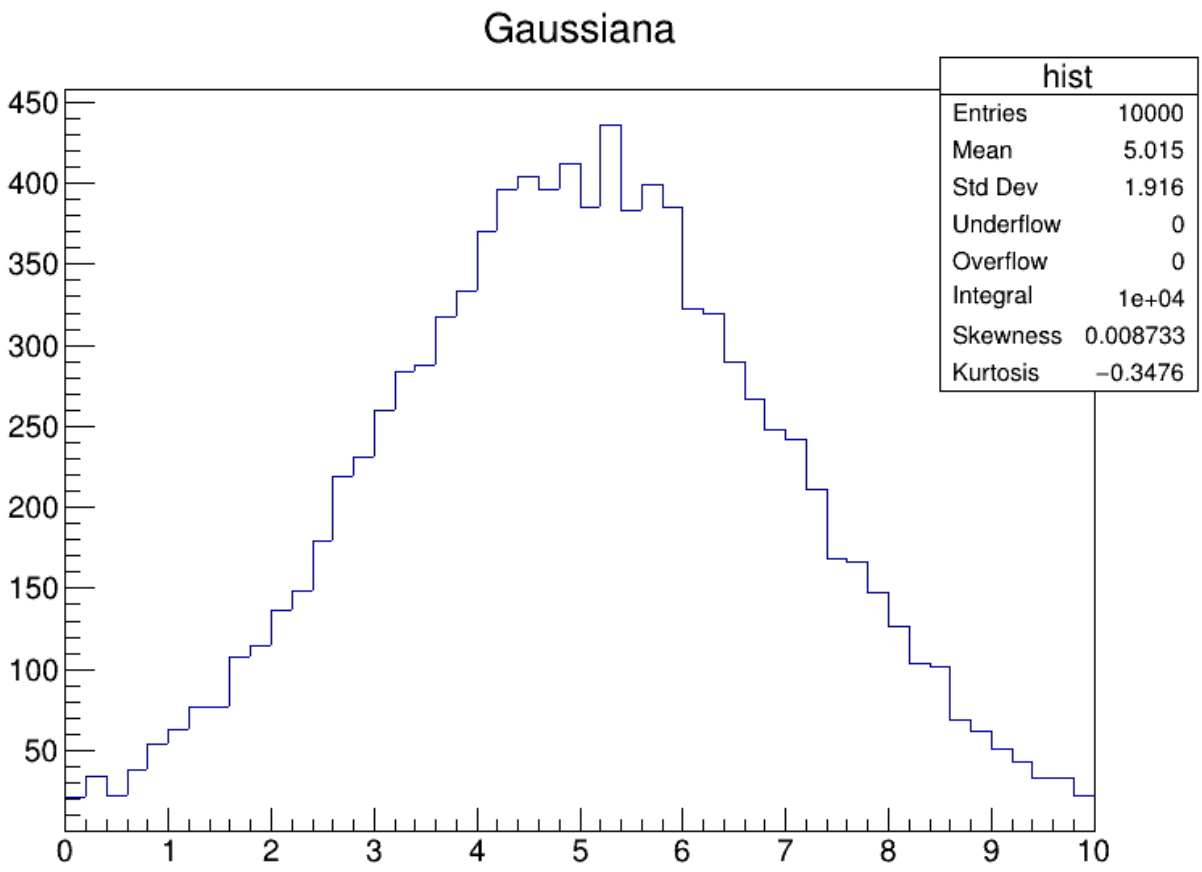
EXERCÍCIO 3

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.

Meu código para esse exercício é:

```
1  #include "TH1.h"
2  #include "TF1.h"
3  #include "TCanvas.h"
4
5  void Q3() {
6      TH1F* histGaus = new TH1F("hist", "Gaussiana", 50, 0, 10);
7
8      TF1* gaussiana = new TF1("gaussiana", "gaus", 0, 10);
9      gaussiana->SetParameters(1, 5, 2); // amplitude, media e desvio padr o
10     histGaus->FillRandom("gaussiana", 10000);
11     gStyle->SetOptStat(111111111);
12
13     TCanvas *c1 = new TCanvas("c1", "Gauss", 800, 600);
14     histGaus->SetMarkerStyle(21);
15     histGaus->SetMarkerColor(1);
16
17     histGaus->Draw();
18 }
```

7cm



Histograma 4: Gausiana

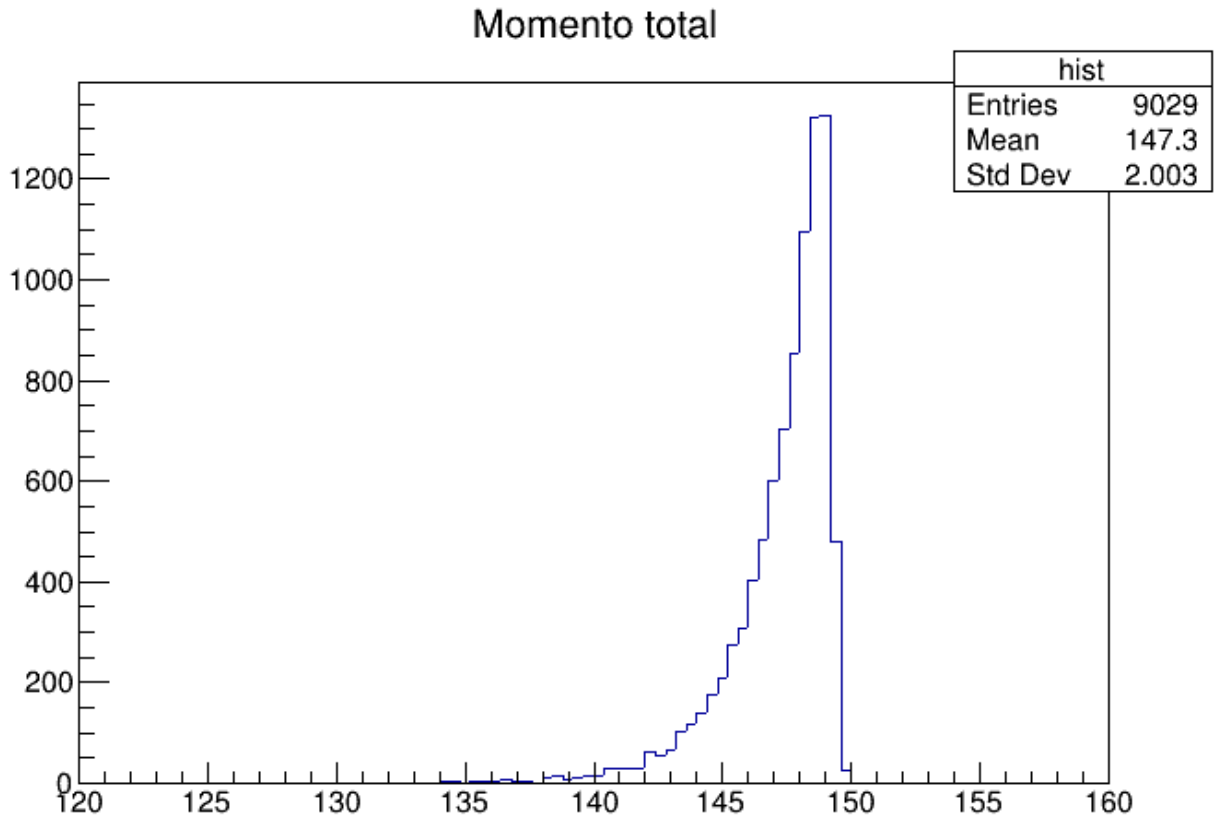
EXERCÍCIO 4

sing 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.

Para esse exercício eu tive um problema, não consegui usar o Tcut para realizar os cortes. Fiz os corte de uma outra maneira, apenas para apresentrar.

```
1
2  #include "TFile.h"
3  #include "TTree.h"
4  #include "TH1F.h"
5  #include "TCut.h"
6  #include "TCanvas.h"
7  #include "TMath.h"
8
9  void Q4() {
10     TFile *file = TFile::Open("tree.root");
11     TTree *tree = (TTree*)file->Get("tree1");
12     TH1F *hist = new TH1F("hist", "Total Momentum Distribution", 100, 120, 160);
13
14     //Definindo variaveis e acessando as branches.
15     Int_t           event;
16     Float_t         ebeam;
17     Float_t         px;
```

```
    Float_t         py;
    Float_t         pz;
    Float_t         zv;
    Float_t         chi2;
    Float_t         Momento;

    tree->Branch("event",&event,"event");
    tree->Branch("ebeam",&ebeam,"ebeam/F");
    tree->Branch("px",&px,"px/F");
    tree->Branch("py",&py,"py/F");
    tree->Branch("pz",&pz,"pz/F");
    tree->Branch("zv",&zv,"zv/F");
    tree->Branch("chi2",&chi2,"chi2/F");

    tree->SetBranchAddress("px",&px);
    tree->SetBranchAddress("py",&py);
    tree->SetBranchAddress("pz",&pz);
    tree->SetBranchAddress("zv",&zv);
    tree->SetBranchAddress("ebeam",&ebeam);
    tree->SetBranchAddress("event",&event);
    tree->SetBranchAddress("chi2",&chi2);


float Mean_ebeam = 0;
for (int i = 0; i < tree->GetEntries(); i++) {
    tree->GetEntry(i);

      Mean_ebeam = Mean_ebeam + ebeam;

   }
    Mean_ebeam = Mean_ebeam/tree->GetEntries();

for (int i = 0; i < tree->GetEntries(); i++) {
    tree->GetEntry(i);
    if ((ebeam - Mean_ebeam) > 0.2){
      Momento = TMath::Sqrt(px*px + py*py + pz*pz);
      hist->Fill(Momento);
    }
  }
 TCanvas *canvas = new TCanvas("canvas", "Momento total", 800, 600);
  hist->Draw();
    canvas->SaveAs("Q4.png");
}
```



Histograma 5: Momento total.