Introdução à Análise de dados em FAE **(DATA)**

Introdução ao ROOT

Professores: Dilson de Jesus Damião, Eliza Melo da Costa e Maurcio Thiel
Name: Bruno Kron Guandalini e Pedro Oliveira

EXERCICIO 1

Crie uma função com parâmetros, $p_0 \cdot \frac{\sin(p_1 \cdot x)}{x}$, e desenhe-a para diferentes valores de parâmetros. Defina a cor da função paramétrica como azul. Após desenhar a função, calcule para os valores dos parâmetros ($p_0 = 1$, $p_1 = 2$):

1. Valor da função para $x = 1$
2. Derivada da função para $x = 1$
3. Integral da função entre 0 e 3

RESPOSTA:

```
#include <TCanvas.h>
#include <TMath.h>
#include <TF1.h>

void exercicio_1() {

    double p0 = 1, p1 = 2;

    TF1 *f = new TF1("f", "[0] * sin([1] * x) / x", 0.1, 10);
    f->SetParameters(p0, p1);

    TCanvas *c1 = new TCanvas("c1", "Fun   o", 800, 600);
    f->SetLineColor(kBlue);
    f->Draw();

    c1->SaveAs("exercicio_1.png");
}

    void calcu_val() {

    double p0 = 1, p1 = 2;
    TF1 *f = new TF1("f", "[0] * sin([1] * x) / x", 0.1, 10);
    f->SetParameters(p0, p1);

    double x_value = 1;
    double func_value = f->Eval(x_value);
    std::cout << "Valor da fun   o para x = " << x_val << ": " << func_val << std::
        endl;

    double derivative_value = f->Derivative(x_value);
    std::cout << "Derivada da fun   o para x = " << x_val << ": " << der_val << std
        ::endl;

    double integral_value = f->Integral(0, 3);
    std::cout << "Integral da fun   o entre 0 e 3: " << integral_val << std::endl;
```

```
34 }
```

1. Valor da função para x = 1: 0.909297
2. Derivada da função para x = 1: -1.74159
3. Integral da função entre 0 e 3: 1.42469

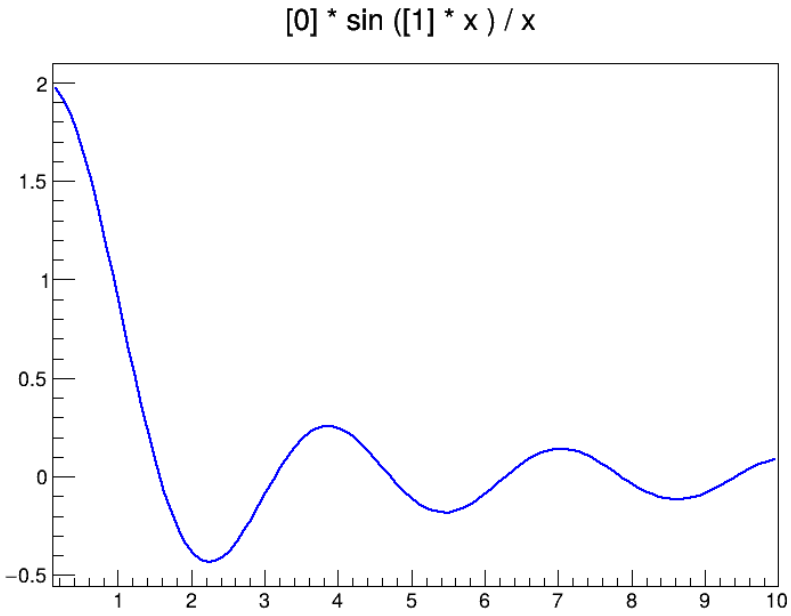

Figura 1: Gráfico da função

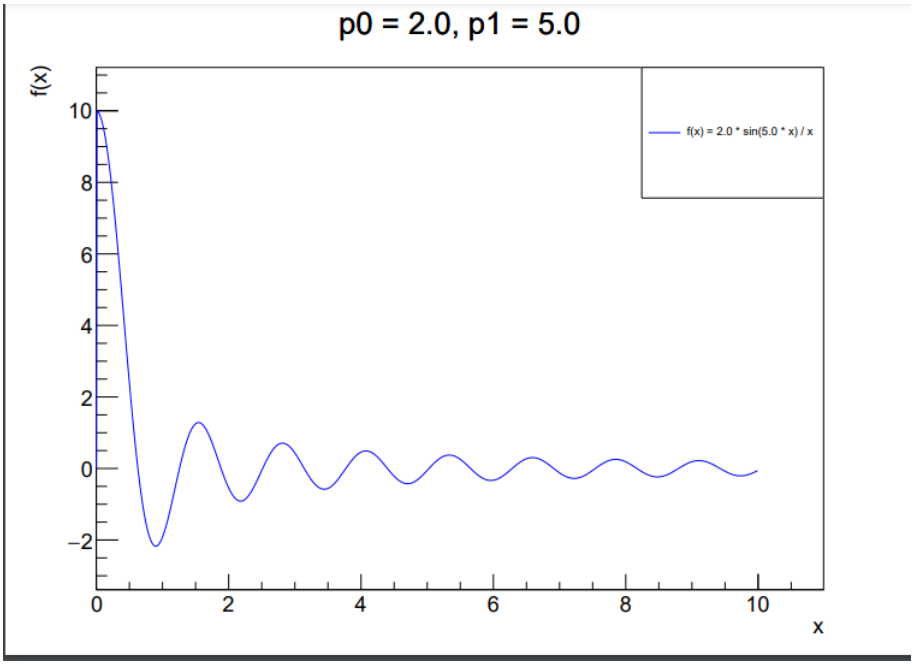

Figura 2: $p_0 = 2.0$ e $p_1 = 5.0$

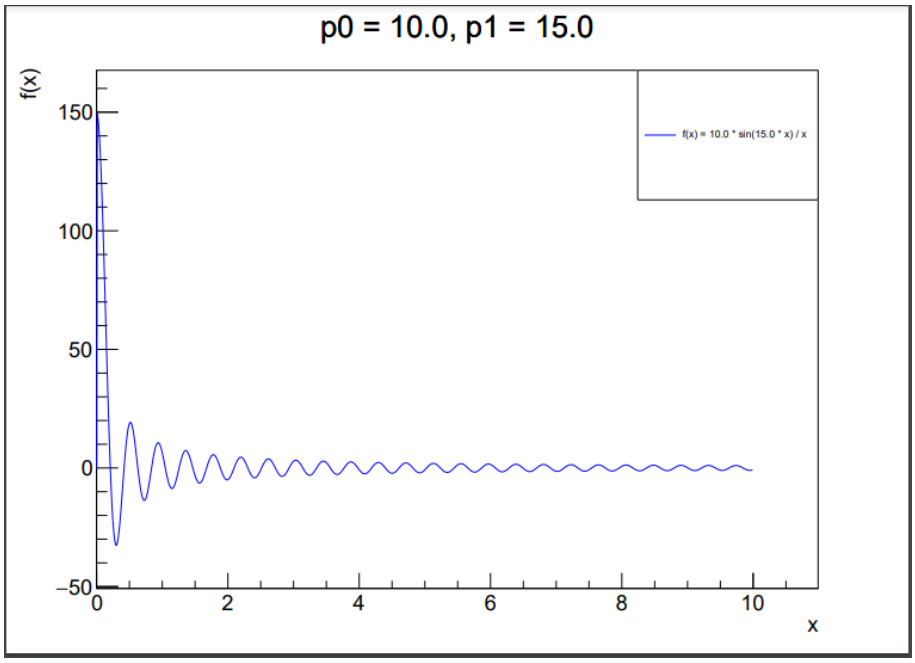

Figura 3: $p_0 = 10.0$ e $p_1 = 15.0$

EXERCICIO 2

Suponha que você tenha este conjunto de pontos definido no arquivo anexado `graphdata.txt`. Plote esses pontos utilizando a classe `TGraph`. Use como marcador de ponto uma caixa preta. Observando as opções possíveis para desenhar o `TGraph` no `TGraphPainter`, plote uma linha conectando os pontos. Crie um `TGraphError` e exiba-o utilizando o conjunto de dados anexado, `graphdata_error.txt`, contendo erro em x e y.

RESPOSTA:

```
#include <TGraph.h>
#include <TGraphErrors.h>
#include <TCanvas.h>
#include <TStyle.h>
#include <iostream>

void exercicio_2() {

    TCanvas *canvas = new TCanvas("canvas", "Graph with Errors", 800, 600);


    TGraph *graph = new TGraph("graphdata.txt");
    if (!graph) {
        std::cerr << "Erro ao carregar graphdata.txt" << std::endl;
        return;
    }


    TGraphErrors *graphErrors = new TGraphErrors("graphdata_error.txt");
    if (!graphErrors) {
        std::cerr << "Erro ao carregar graphdata_error.txt" << std::endl;
        return;
    }


    graph->SetMarkerStyle(22);
    graph->SetMarkerColor(kBlack);
    graph->SetTitle("Grafico com Erros;eixo-X;eixo-Y");
```

```
29      graph->Draw("ALP");
30
31
32      graphErrors->SetMarkerStyle(21);
33      graphErrors->SetMarkerColor(kRed);
34      graphErrors->Draw("P");
35
36
37      canvas->Print("exerc cio_2.pdf");
38
39
40  }
41
42  void grafico() {
43      graph();
44  }
```

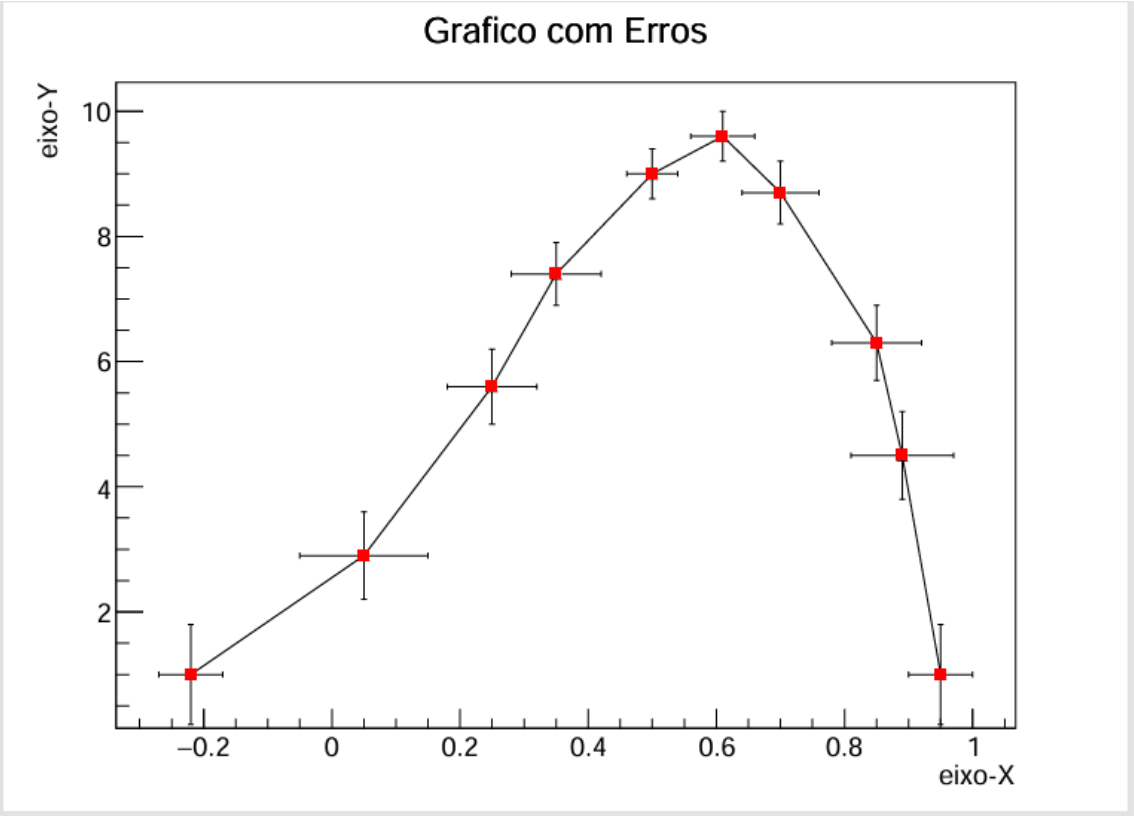

Figura 4: Gráfico com erros

EXERCICIO 3

Crie um histograma unidimensional com 50 intervalos entre 0 e 10, e preencha-o com 10.000 números aleatórios distribuídos de forma gaussiana com média 5 e desvio padrão 2. Plote o histograma e, observando a documentação no `THistPainter`, mostre na caixa de estatísticas o número de entradas, a média, o RMS, a integral do histograma, o número de underflows, o número de overflows, a assimetria e a curtose.

RESPOSTA:

```
#include <TH1F.h>
#include <TCanvas.h>
#include <TRandom3.h>
#include <TStyle.h>
#include <TApplication.h>

void exercicio_3() {
    TRandom3 random;

    TH1F *hist = new TH1F("hist", "Histogram of Gaussian Random Numbers;Value;Entries
        ", 50, 0, 10);

    for (int i = 0; i < 10000; i++) {
        double value = random.Gaus(5, 2);
        hist->Fill(value);
    }

    TCanvas *canvas = new TCanvas("canvas", "Gaussian Histogram", 800, 600);

    gStyle->SetOptStat("kseiorum");

    hist->Draw();

    canvas->Print("exerc cio_3.pdf");
}
```

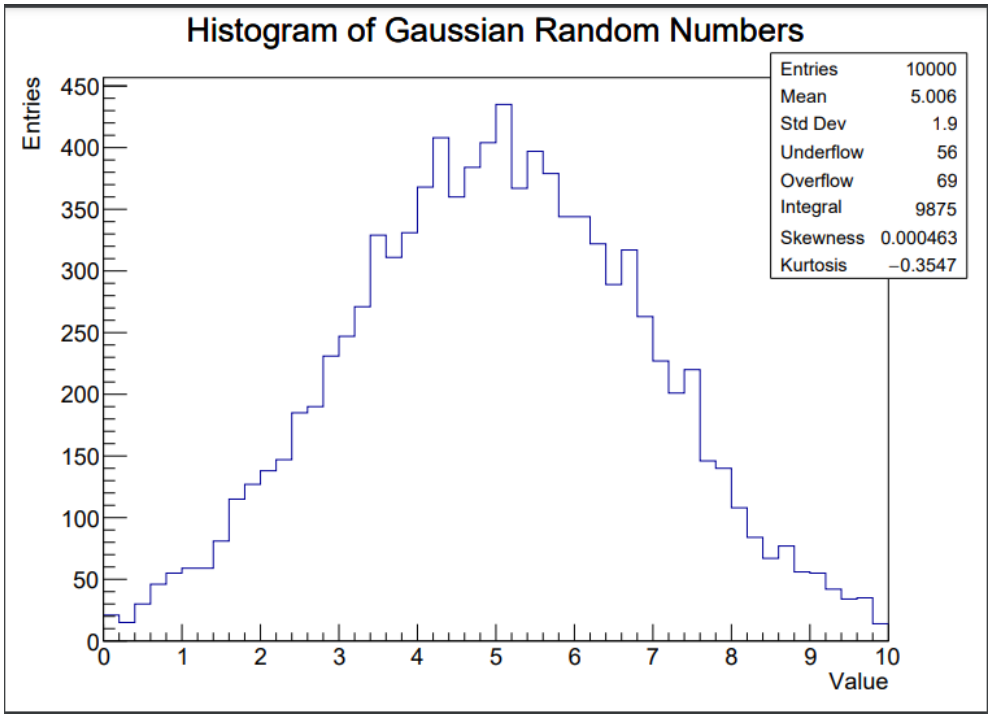

Figura 5: Gaussiana

EXERCICIO 4

Usando a árvore contida no arquivo `tree.root`, faça uma distribuição do momento total de cada partícula cujo energia do feixe estava fora da média em mais de 0,2. Use objetos `TCut` para fazer a seleção dos eventos. Projete essa distribuição em um histograma, desenhe-o e salve-o em um arquivo.

RESPOSTA:

```
#include <TFile.h>
#include <TTree.h>
#include <TBranch.h>
#include <TCut.h>
#include <TH1F.h>
#include <TMath.h>
#include <TCanvas.h>
#include <TString.h>
#include <iostream>


void exercicio_4() {
    TFile *file = TFile::Open("tree.root");
    TTree *tree = (TTree*)file->Get("tree1");

    float ebeam, px, py, pz;
    tree->SetBranchAddress("ebeam", &ebeam);
    tree->SetBranchAddress("px", &px);
    tree->SetBranchAddress("py", &py);
    tree->SetBranchAddress("pz", &pz);

    double totalEnergy = 0;
    Long64_t nEntries = tree->GetEntries();

    for (Long64_t i = 0; i < nEntries; i++) {
        tree->GetEntry(i);
        totalEnergy += ebeam;
    }

    if (nEntries > 0) {
        double meanEnergy = totalEnergy / nEntries;
        double lowerCut = meanEnergy - 0.2;
        double upperCut = meanEnergy + 0.2;

        TCut cut = Form("ebeam < %f || ebeam > %f", upperCut, lowerCut);

        TH1F *histogram = new TH1F("h_total_momentum", "Total Momentum Distribution",
            200, 130, 160);

        tree->Draw("TMath::Sqrt(px*px + py*py + pz*pz)>>h_total_momentum", cut);

        TCanvas *c2 = new TCanvas("c2", "Total Momentum Distribution", 800, 600);
        histogram->Draw();

        c2->SaveAs("exerc cio_4.pdf");
    }
    file->Close();
}
```

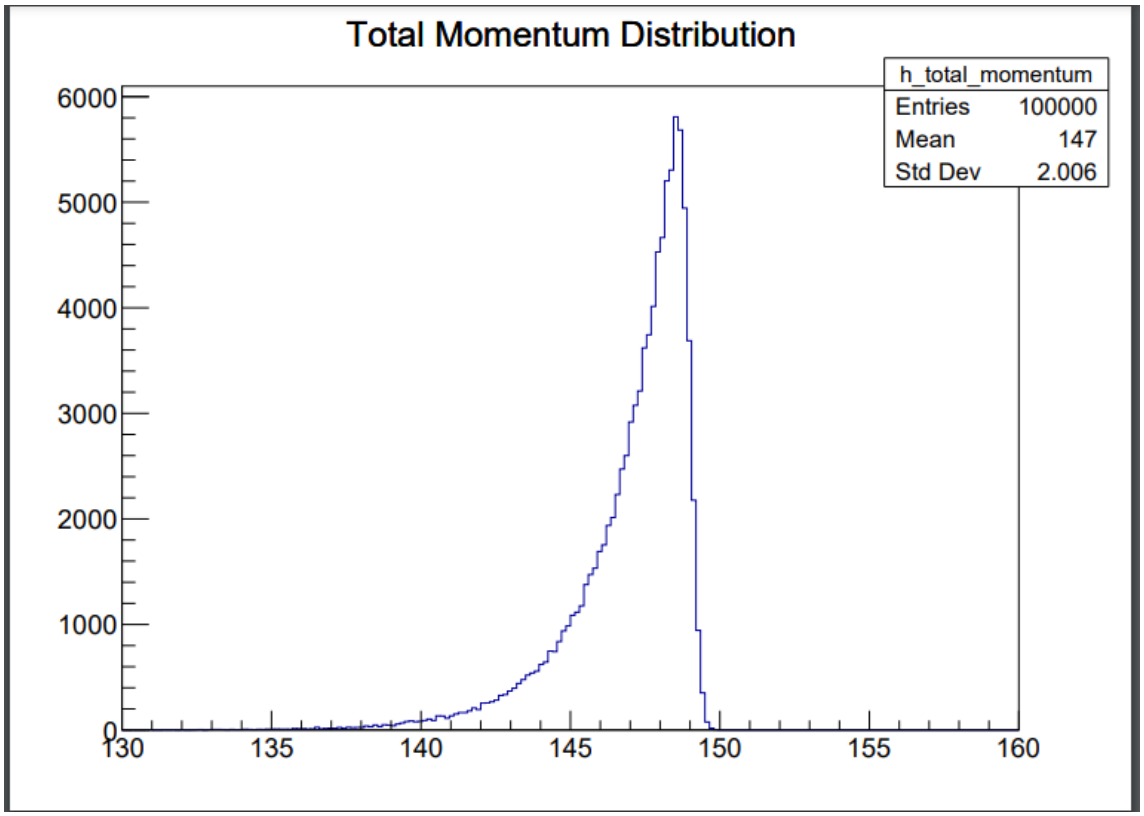

Figura 6: Distribuição do Momentum Total