Introdução à Análise de dados em FAE **(26/04/2024)**

Lista de exercícios 4

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

EXERCÍCIO 1

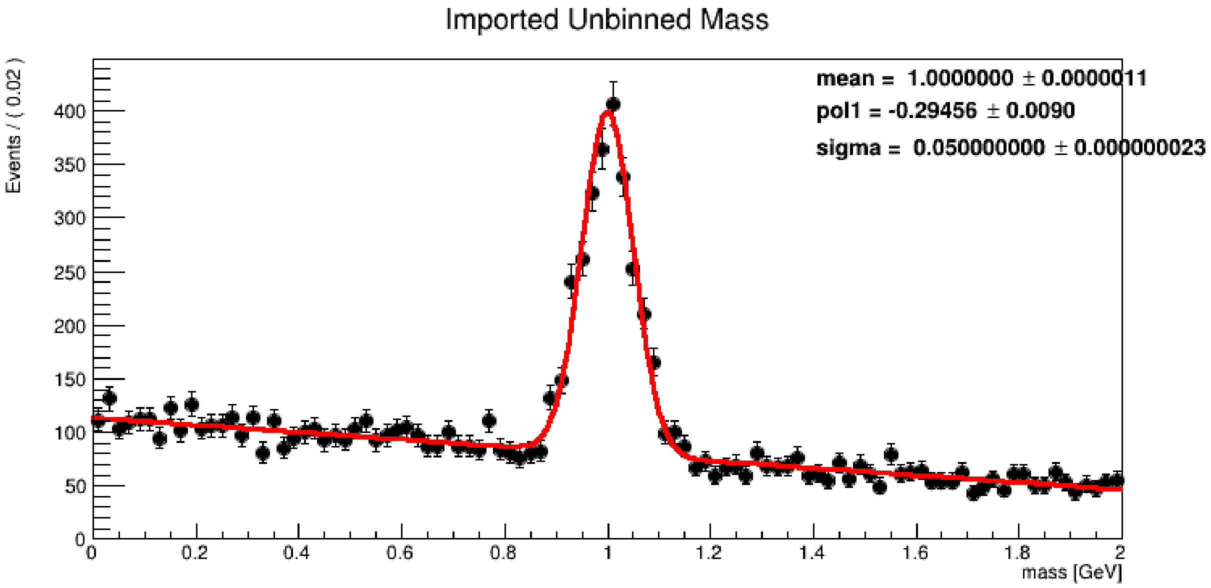
Para melhorar o Fit, adicionei uma reta linear ao ajuste, também determinei a contribuição que cada função (Gausiana e linear) deveriam ter no Fit.

Para a primeira questão elaborei esse código:

```
import sys
sys.path.append("/content/root_build/")
sys.path.append("/content/root_build/bin/")
sys.path.append("/content/root_build/include/")
sys.path.append("/content/root_build/lib/")
import ctypes
ctypes.cdll.LoadLibrary(’/content/root_build/lib//libCore.so’)
ctypes.cdll.LoadLibrary(’/content/root_build/lib//libThread.so’)
ctypes.cdll.LoadLibrary(’/content/root_build/lib//libTreePlayer.so’)

#Block to import all the ROOT functions that we will be using throughout this template
from ROOT import TFile
from ROOT import TLorentzVector
from ROOT import TH1F
from ROOT import TF1
import numpy as np
from ROOT import RooRealVar
from ROOT import RooDataHist
from ROOT import RooDataSet
from ROOT import RooExponential
from ROOT import RooGaussian
from ROOT import RooVoigtian
from ROOT import RooPolynomial
from ROOT import RooArgList
from ROOT import RooArgSet
from ROOT import RooAddPdf
from ROOT import RooPlot
from ROOT import TLegend
from ROOT import RooFit
from ROOT import TLatex
from ROOT import RooChi2Var
from ROOT import TStyle
from ROOT import TCanvas, TFile, TPaveText, TH1F, TLegend, TTree
from ROOT import gStyle, TGraphErrors, TF1, TGraph, gPad, gRandom
from ROOT import kRed, kBlue
from ROOT import TFitResultPtr, TMatrixD

# Criando uma TTree chamada "tree" e adicionando dois ramos (branches) a ela: "x" e "y".
# Em seguida, preenchemos a TTree com dados aleat rios usando gRandom.Gaus e gRandom.Uniform.
from array import array

tree = TTree("tree", "tree")
px = array(’d’, [0])
py = array(’d’, [0])
```

```
tree.Branch("x", px, "x/D")
tree.Branch("y", py, "y/D")
#
for i in range(100):
    px[0] = gRandom.Gaus(0, 3)
    py[0] = gRandom.Uniform() * 30 - 15
    tree.Fill()
#Depois disso, estamos utilizando o RooFit para importar os dados do TTree para um RooDataSet chamado "ds".
# Define 2nd observable y
x = RooRealVar("x", "x", -3, 3)
y = RooRealVar("y", "y", -10, 10)
ds = RooDataSet("ds", "ds", RooArgSet(x, y),RooFit.Import(tree))
ds.Print()

# Plot dataset with multiple binning choices
# ------------------------------------------------------------------------------------

# Print unbinned dataset with default frame binning (100 bins)
frame = y.frame(RooFit.Title("Unbinned data shown in default frame binning"))
ds.plotOn(frame)

# Print unbinned dataset with custom binning choice (20 bins)
frame2 = y.frame(RooFit.Title("Unbinned data shown with custom binning"))
ds.plotOn(frame2, RooFit.Binning(20))

# Draw all frames on a canvas
c = TCanvas("dataimport", "dataimport", 800, 800)
c.Divide(2)
c.cd(1)
gPad.SetLeftMargin(0.15)
frame.GetYaxis().SetTitleOffset(1.4)
frame.Draw()
c.cd(2)
gPad.SetLeftMargin(0.15)
frame2.GetYaxis().SetTitleOffset(1.4)
frame2.Draw()
c.Draw()
c.SaveAs("dataimport.png")

# Make a plot of unbinned dataset (ROOT.RooFit # default)
frame3 = mass.frame(RooFit.Title("Imported Unbinned Mass"))
data.plotOn(frame3)
frame3.SetStats(0)

# Fit a Gaussian p.d.f to the data
mean = RooRealVar("mean", "mean", 1.0, 1., 1.2)

sigma = RooRealVar("sigma", "sigma",0.05, 0., 0.05)

pol1 = RooRealVar("pol1", "Constant of the polynomial", 0, -10.0, 10.0)

gauss = RooGaussian("gauss", "gauss", mass, mean, sigma)
Linear = RooPolynomial("Linear", "Linear", mass, pol1)

sinal = RooRealVar("sinal", "sinal", 0.2)
sum_pdf = RooAddPdf("sum_pdf", "Sum of Gaussian and Linear", RooArgList(gauss, Linear), RooArgList(sinal))


gauss.fitTo(data)
Linear.fitTo(data)
```

```

#gauss.plotOn(frame3, LineColor="g")
#Linear.plotOn(frame3, LineColor="b")
sum_pdf.plotOn(frame3, LineColor="r")

sum_pdf.paramOn(frame3,data)
#sum_pdf.paramOn(frame3,data)
#gauss.paramOn(frame3,data)
#Linear.paramOn(frame3,data)
#data.statOn(frame3)
c3 = TCanvas("exemplo03","exemplo03",800,400)

frame3.Draw()
c3.Draw()
gauss.Print("t")
LineColor="r"

}
```



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

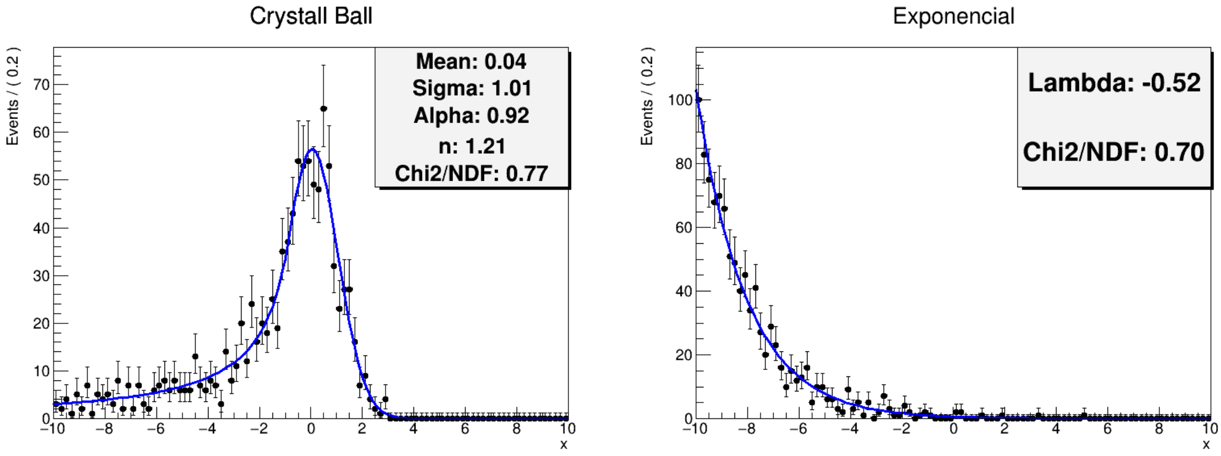
EXERCÍCIO 2

Fiz dois ajustes! Um deles utilizando uma crystall Ball e o outro um exponencial.

Meu código foi:

```
#include "RooRealVar.h"
#include "RooDataSet.h"
#include "RooPlot.h"
#include "RooGaussian.h"
#include "RooCBShape.h" // Crystall Ball
#include "RooExponential.h" // Exponential
#include "RooFit.h"
#include "RooAddPdf.h"
#include "RooRandom.h"
#include "TCanvas.h"
#include "TPaveText.h"

using namespace RooFit;

```

```
int Q2() {


// variaveis-------------------------------------------------------------
    RooRealVar x("x", "x", -10, 10);
    RooRealVar mean("mean", "Mean", 0, -10, 10);//CBS
    RooRealVar sigma("sigma", "Sigma", 1, 0.1, 5);//CBS
    RooRealVar alpha("alpha", "Alpha", 1, 0.1, 10);//CBS
    RooRealVar n("n", "n", 1, 0.1, 10);//CBS

    RooCBShape CBS("CBS", "Crystall Ball", x, mean, sigma, alpha, n);

    RooDataSet *data = CBS.generate(x, 1000); // gerando os eventos


    RooFitResult* fit_result = CBS.fitTo(*data, Save());//junta CBS com os dados


    TCanvas *c1 = new TCanvas("c1", "Crystall Ball", 800, 600);
    RooPlot *frame = x.frame();
    frame->SetTitle("Crystall Ball");
    data->plotOn(frame);
    CBS.plotOn(frame);
    frame->Draw();

    TPaveText *Leg = new TPaveText(0.6, 0.6, 0.9, 0.9, "NDC");//legenda
    Leg->AddText(Form("Mean: %.2f", mean.getVal()));
    Leg->AddText(Form("Sigma: %.2f", sigma.getVal()));
    Leg->AddText(Form("Alpha: %.2f", alpha.getVal()));
    Leg->AddText(Form("n: %.2f", n.getVal()));
    Leg->AddText(Form("Chi2/NDF: %.2f", frame->chiSquare()));
    Leg->Draw();

//Exponencial----------------------------------------------------------------------

    TCanvas *c2 = new TCanvas("c2", "Exponencial", 800, 600);


    RooRealVar lambda("lambda", "lambda", -0.5, -10., 0.);//exp
    RooExponential exp("exp", "Exponential ", x, lambda);//exp

    RooDataSet *exp_data = exp.generate(x, 1000);//junta exp com os dados

    RooFitResult* exp_fit_result = exp.fitTo(*exp_data, Save());

    c2->Draw();
    RooPlot *exp_frame = x.frame();
    exp_data->plotOn(exp_frame);
    exp.plotOn(exp_frame);
    exp_frame->SetTitle("Exponencial");
    exp_frame->Draw("same");

    TPaveText *exp_statsBox = new TPaveText(0.6, 0.6, 0.9, 0.9, "NDC");
    exp_statsBox->AddText(Form("Lambda: %.2f", lambda.getVal()));
    exp_statsBox->AddText(Form("Chi2/NDF: %.2f", exp_frame->chiSquare()));
    exp_statsBox->Draw();

    c1->SaveAs("CrystallBall.png");
    c2->SaveAs("Exponencial.png");

    return 0;
 }
```



(a)

(b)

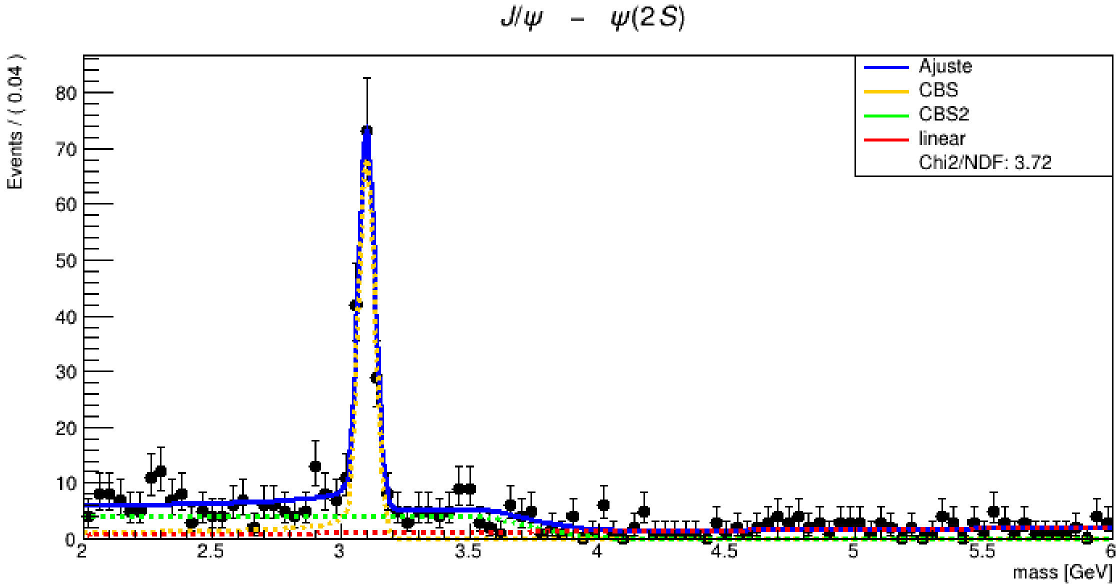
EXERCÍCIO 3

Não consegui fazer esse código em c++, pois tive dificuldade para acessar o arquivo .root. Juntei duas Crystal Balls e uma polinomial para fazer o ajuste. Meu código para esse exercício é:

```python

from ROOT import TFile
from ROOT import TLorentzVector
from ROOT import TH1F
from ROOT import TF1
#import numpy as np
from ROOT import RooRealVar
from ROOT import RooDataHist
from ROOT import RooDataSet
from ROOT import RooExponential
from ROOT import RooGaussian
from ROOT import RooVoigtian
from ROOT import RooPolynomial
from ROOT import RooCBShape
from ROOT import RooArgList
from ROOT import RooArgSet
from ROOT import RooAddPdf
from ROOT import RooPlot
from ROOT import TLegend
from ROOT import RooFit
from ROOT import TLatex
from ROOT import RooChi2Var
from ROOT import TStyle
from ROOT import TCanvas, TFile, TPaveText, TH1F, TLegend, TTree
from ROOT import gStyle, TGraphErrors, TF1, TGraph, gPad, gRandom
from ROOT import kRed, kBlue
from ROOT import TFitResultPtr, TMatrixD
import ROOT

#lendo o arquivo
file0 = TFile("DataSet_lowstat.root")
Data = file0.Get("data")


#cores na legenda
green = TH1F("h2","Ex",1,-10,10)
green.SetLineColor(ROOT.kGreen)
green.SetLineStyle(1)
green.SetLineWidth(2)

blue = TH1F("h2","Ex",1,-10,10)
blue.SetLineColor(ROOT.kBlue)
blue.SetLineStyle(1)
blue.SetLineWidth(2)

orange = TH1F("h2","Ex",1,-10,10)
orange.SetLineColor(ROOT.kOrange)
orange.SetLineStyle(1)
orange.SetLineWidth(2)

Dashed = TH1F("h2","Ex",1,-10,10)
Dashed.SetLineColor(ROOT.kDashed)
Dashed.SetLineStyle(1)
Dashed.SetLineWidth(2)


#variaveis
mass = RooRealVar("mass", "mass [GeV]", 2, 6);
frame3 = mass.frame(RooFit.Title(r"$J/\psi$-$\psi(2S)$"))
Data.plotOn(frame3)

c3 = TCanvas("c3","c3",800,400)

mean_CBS = RooRealVar("mean_CBS", "mean_CBS",3.1, 3.1, 3.1)
sigma_CBS = RooRealVar("sigma_CBS", "sigma_CBS", 0.09, 0.0, 0.2)
alpha = RooRealVar("alpha_CBS", "alpha_CBS", 1.5, 0, 6)
n = RooRealVar("n_CBS", "nCBS",1.5, 0, 6)


mean_CBS2 = RooRealVar("mean_CBS2", "mean_CBS2",3.5, 3.5, 3.5)
sigma_CBS2 = RooRealVar("sigma_CBS2", "sigma_CBS2", 0.2, 0.0, 0.3)
alpha2 = RooRealVar("alpha_CBS2", "alpha_CBS2", 1.5, 0, 6)
n2 = RooRealVar("n_CBS2", "nCBS2",1.5, 0, 1)


pol1 = RooRealVar ( " pol1 " , " pol1 " , 0 , -10.0 , 10.0)

sinal_1=RooRealVar("sinal_1", "sinal_1", 0.5, 0, 1.);
sinal_2=RooRealVar("sinal_1", "sinal_1", 0.5, 0, 1.);

#Fun  es
CBS = RooCBShape("CBS", "CBS", mass, mean_CBS, sigma_CBS, alpha,n)
CBS2 = RooCBShape("CBS2", "CBS2", mass, mean_CBS2, sigma_CBS2, alpha2,n2)
linear = RooPolynomial("linear", "linear", mass, pol1)



sum_pdf = RooAddPdf("sum_pdf", "Soma dos ajustes", RooArgList(CBS, CBS2, linear), RooArgList(sinal_1, sinal_2))


sum_pdf.fitTo(Data)


sum_pdf.plotOn(frame3, RooFit.LineColor(kBlue))
sum_pdf.plotOn(frame3, RooFit.Components("CBS"), RooFit.LineStyle(ROOT.kDashed), RooFit.LineColor(ROOT.kOrange))
sum_pdf.plotOn(frame3, RooFit.Components("CBS2"), RooFit.LineStyle(ROOT.kDashed), RooFit.LineColor(ROOT.kGreen))
sum_pdf.plotOn(frame3, RooFit.Components("linear"), RooFit.LineStyle(ROOT.kDashed), RooFit.LineColor(ROOT.kDashed))
```

```

#legenda
legend = TLegend(0.7, 0.7, 0.9, 0.9)
legend.AddEntry(blue, "Ajuste", "l")
legend.AddEntry(orange, "CBS", "l")
legend.AddEntry(green, "CBS2", "l")
legend.AddEntry(Dashed, "linear", "l")
chi2_ndf = frame3.chiSquare()
legend.AddEntry(sum_pdf,"Chi2/NDF: {:.2f}".format(chi2_ndf), "")




frame3.Draw()
c3.Draw()
legend.Draw()

c3.Print("mass.png")
```

7cm


J/ψ – $\psi(2S)$
Events / (0.04)
mass [GeV]
Ajuste
CBS
CBS2
linear
Chi2/NDF: 3.72