下载文章

Lesson 0 install root and software on your compute

> *"If you get the THU's computer cluster account, use following commands to get into.*
>
> *`$ ssh -XY -p 48571 -o ServerAliveInterval=5  yourname@hepthu.com`*
>
> *You can also run code on your pc if you have installed root, it's up to you. "*

homebrew Click this link to go to the Homebrew page, follow the instructions to install Homebrew on your Mac.
Homebrew is a software package manager that allows you to download many applications on your Mac!
After finishing the installation, paste the following command in a macOS terminal to install CERN ROOT software: `$ brew install root` . This way, you can also install applications like `firefox` which you can't find in the App Store.
Additionally, `$` indicates a shell command, so drop the `$` sign when you copy the command.

Lesson 1 base commands for Linux

Some simple commands you can have a try!

```
pwd  #print work directory
cd /path/to/directory #change directory
cd ..  #back to parent directory
ls    #list directory contents
mkdir test #make a directory
touch test.txt #creat a new txt file
cp test.txt copy.txt #copy and rename a file
mv copy.txt copy1.txt #move and rename a file
rm copy.txt #remove
rm -r test  #delete a directory
vi filename    #open or make a file with vim editor
```

Vim editor

```
i     #insert, edit mode
Esc    #exit edit mode
:wq    #save and exit
:q    #exit, will mention you if you have modified the file
:q!    #exit !without! save
```

you will use the next commands from time to time
vi somefile -> press i into edit mode -> do something ->esc exit edit mode -> :wq

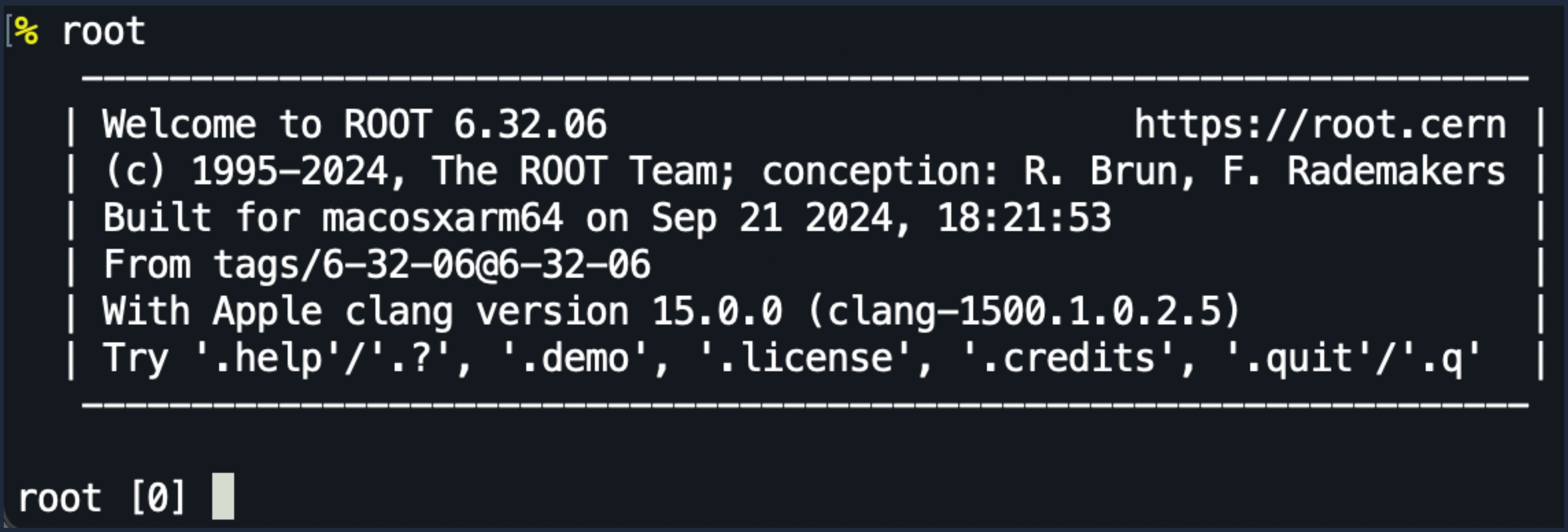
Some tricks that can help you:

Press the `TAB` key to automatically complete some of your commands, such as file names. Pressing it twice will show you the available commands/file names you can input.
Press `↑` to retrieve previous commands.

Lesson 2 root 基础练习

进入root环境

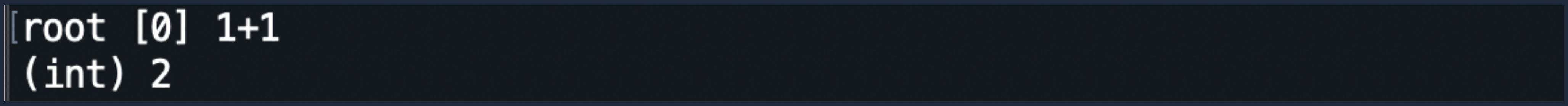
```
$ root
```



```
% root
   ------------------------------------------------------------------
  | Welcome to ROOT 6.32.06                        https://root.cern |
  | (c) 1995-2024, The ROOT Team; conception: R. Brun, F. Rademakers |
  | Built for macosxarm64 on Sep 21 2024, 18:21:53                   |
  | From tags/6-32-06@6-32-06                                        |
  | With Apple clang version 15.0.0 (clang-1500.1.0.2.5)             |
  | Try '.help'/'.?', '.demo', '.license', '.credits', '.quit'/'.q'  |
   ------------------------------------------------------------------

root [0]
```

`root [0] .q` 退出root

root能输入一句解释一句



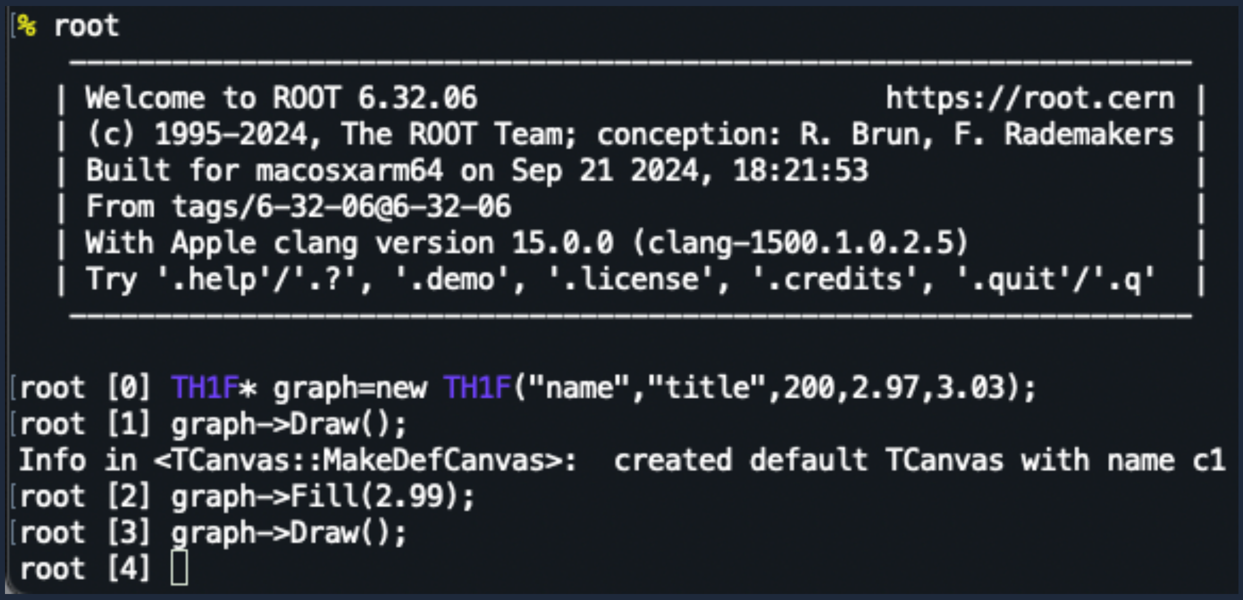
```
root [0] 1+1
(int) 2
```

也可以边解释边执行或者编译后执行一个程序文件，用 `root yourprogram.C`

例如：可以在root中输入下面的程序用来创建一个空白的直方图

```
TH1F* graph=new TH1F("name","title",200,2.97,3.03);
graph->Draw();
```

但一个空白的直方图是很无趣的，你还需要用 `Fill` 来填入数据



```
% root
   ------------------------------------------------------------------
  | Welcome to ROOT 6.32.06                        https://root.cern |
  | (c) 1995-2024, The ROOT Team; conception: R. Brun, F. Rademakers |
  | Built for macosxarm64 on Sep 21 2024, 18:21:53                   |
  | From tags/6-32-06@6-32-06                                        |
  | With Apple clang version 15.0.0 (clang-1500.1.0.2.5)             |
  | Try '.help'/'.?', '.demo', '.license', '.credits', '.quit'/'.q'  |
   ------------------------------------------------------------------

root [0] TH1F* graph=new TH1F("name","title",200,2.97,3.03);
root [1] graph->Draw();
Info in <TCanvas::MakeDefCanvas>:  created default TCanvas with name c1
root [2] graph->Fill(2.99);
root [3] graph->Draw();
root [4]
```

此外我们还能直接创建一个程序文件，以放入更长的程序

Get start!

```
$ vi test.C
```

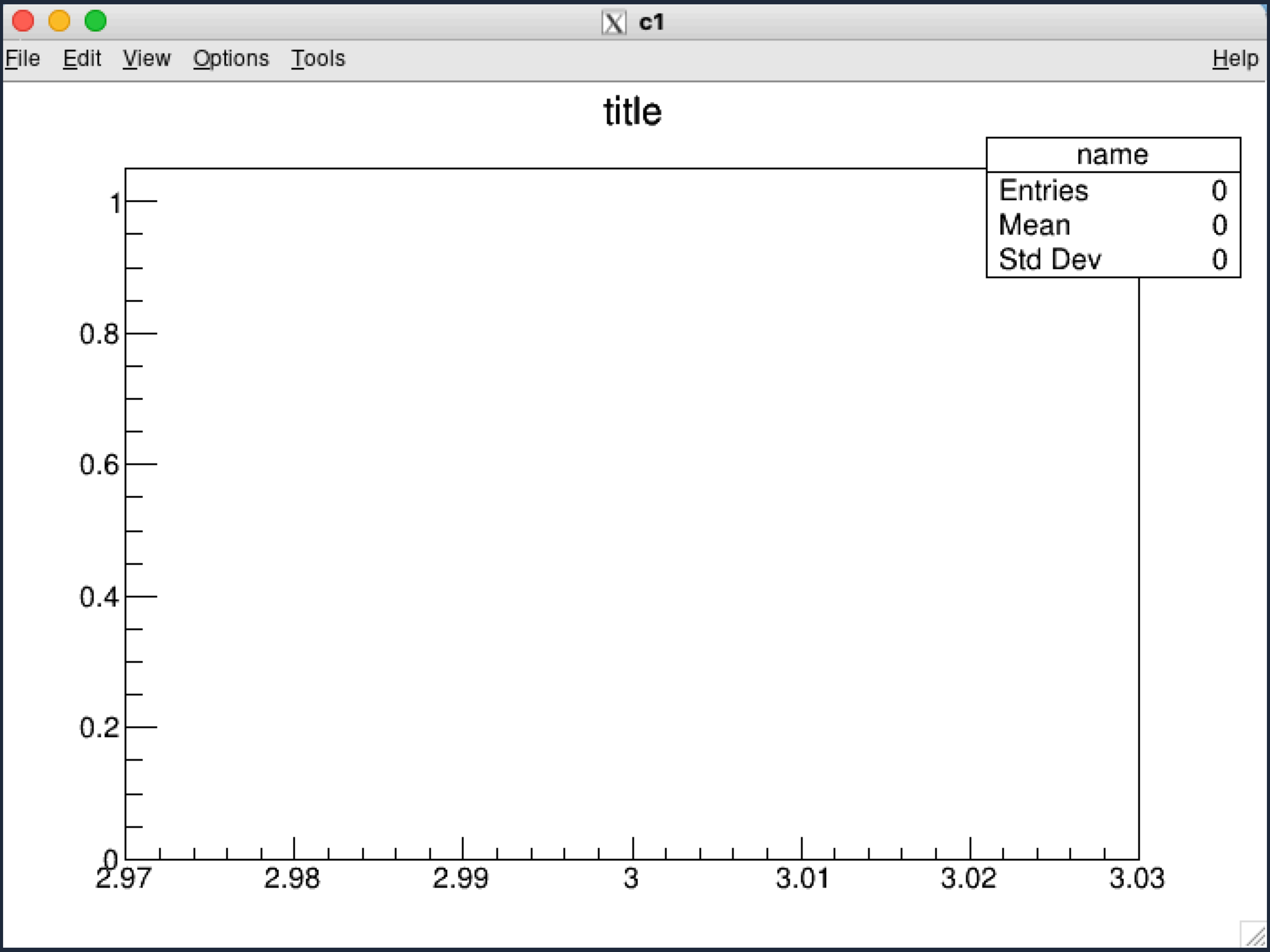
//push `i` *to enter the insert mode//*

```
void test() //be sure it is similar with your file name

 {

  TH1F* graph=new TH1F("name","title",200,2.97,3.03);
  //TH1F* graph name=new TH1F("name","title",bins, low, high)

  graph->Draw();  //draw your graph

  }
```

//push `ESC` *to end the insert mode//*

`:wq` *//save and exit*

`$ root test.C` *//run your code*

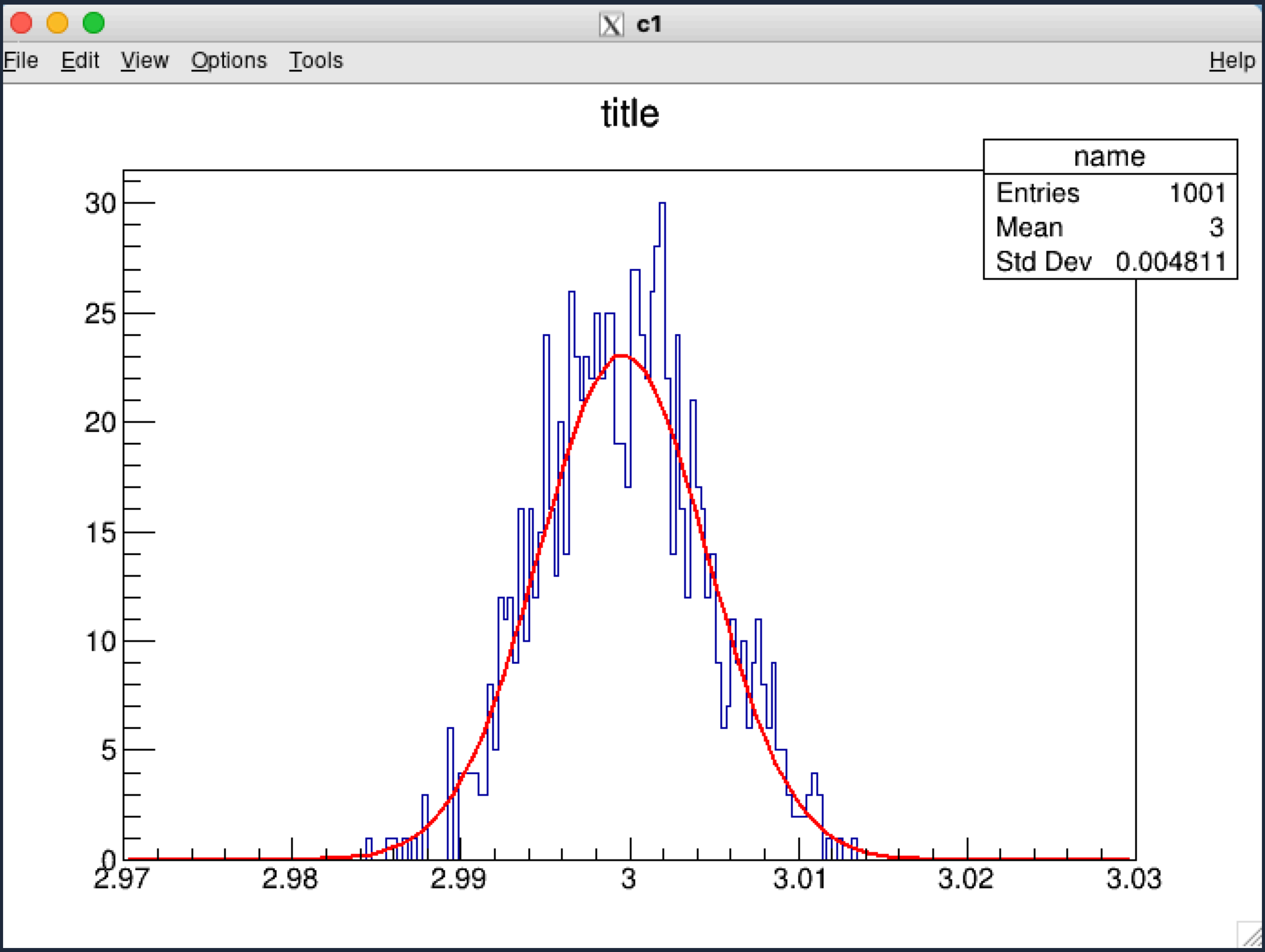


///successful!///

`.q` *//exit the program*

Let's *add something into your graph!*

```
void test(){

 //TCanvas* c1=new TCanvas("c1","fitting with Gaussian function");  //if you don't set the canvas's name, it will created default canvas with name c1

 //c1->SetGrid();  // set grid

 TH1F* graph=new TH1F("name","title",200,2.97,3.03);

 graph->Fill(2.99); // you can add just one data in your graph using "Fill" command and try to draw your graph.It's boring, right? We need more!

 TRandom n; // define a random variable n

 for (int i=0;i<1000;++i){

 graph->Fill(n.Gaus(3,0.005));  //Determine variable n using Gaussian distribution and fill to graph

 }

 graph->Draw();

 graph->Fit("gaus");  //fit your graph with Gaussian function

 }
```



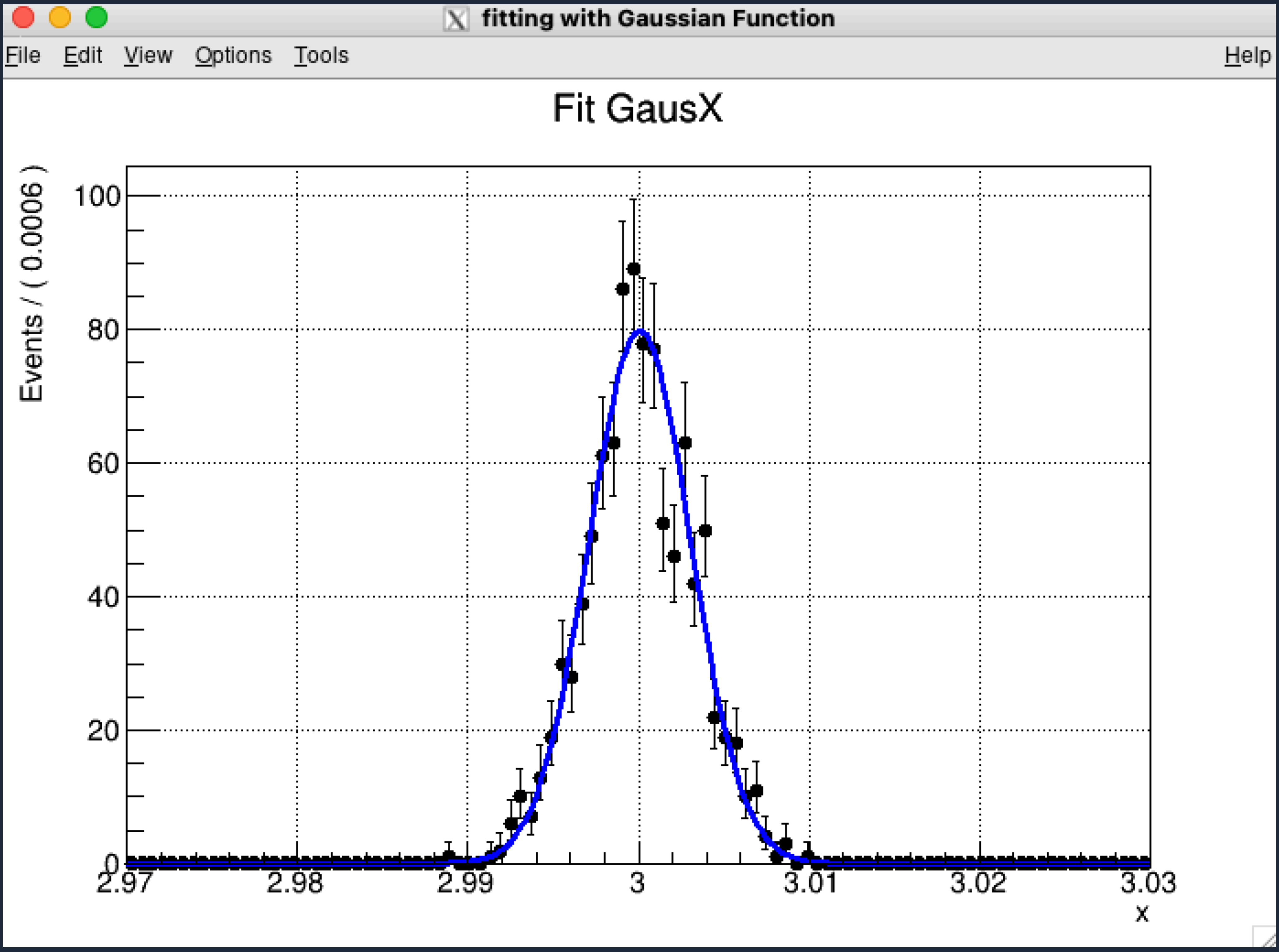
Wow! now you can use root to do some simple fit works. but we usually use RooFit to do more complex job! So, ready for more programs!

以上只是一些基础的拟合，很多参数我们并不能自己去定义，没有灵活性， `RooFit` 中则提供了更加专业的拟合函数。

Lesson 2 Gaussian fitting using RooFit

```
#include "RooAbsReal.h"
#include "RooRealVar.h"
#include "RooGaussian.h"
#include "RooChebychev.h"
#include "RooAddPdf.h"
//#include "RooProdPdf.h"
#include "RooDataSet.h"
//#include "RooDataHist.h"
//#include "RooFitResult.h"
#include "RooPlot.h"
//#include "RooArgList.h"
//#include "RooArgSet.h"
//#include "RooRandom.h"
//#include "RooPrintable.h"
using namespace RooFit;
```

```
void test() {
    RooRealVar x("x", "x", 2.97, 3.03);
    // Declares a real-valued variable 'x' with a range from 2.97 to 3.03.

    RooRealVar mean("mean", "mean", 3.0, 2.8, 3.2);
    // Declares a real-valued variable 'mean' with an initial value of 3.0 and a range from 2.8 to 3.2.

    RooRealVar sigma("sigma", "sigma", 0.003, 0.002, 0.003);
    // Declares a real-valued variable 'sigma' with an initial value of 0.003 and a range from 0.002 to 0.003.

    RooAbsPdf* gaus = new RooGaussian("gaus", "gaus", x, mean, sigma);
    // Creates a Gaussian probability density function (PDF) named 'gaus' with 'x' as the observable, 'mean' as the mean, and 'sigma' as the standard deviation.

    RooRealVar n("n", "n", 0, 0, 50000);
    // Declares a real-valued variable 'n' with an initial value of 0 and a range from 0 to 50000.

    RooExtendPdf* exp = new RooExtendPdf("exp", "exp", *gaus, n);
    // Creates an extended PDF named 'exp' that combines the Gaussian PDF 'gaus' with the variable 'n'.

    RooAddPdf total("total", "total", RooArgList(*exp), RooArgList(n));
    // Creates a composite PDF named 'total' that consists of the extended PDF 'exp' and the variable 'n'.

    RooDataSet* data;
    // Declares a pointer to a RooDataSet object named 'data'.

    data = gaus->generate(RooArgSet(x), 1000);
    // Generates a dataset 'data' with 1000 events based on the Gaussian PDF 'gaus' and the observable 'x'.

    RooFitResult* result = total.fitTo(*data, Save());
    // Fits the composite PDF 'total' to the dataset 'data' and saves the fit result in 'result'.

    TCanvas* c1 = new TCanvas("c1", "fitting with Gaussian function");
    // Creates a new canvas named 'c1' with the title "fitting with Gaussian function" for plotting.

    RooPlot* xframe = x.frame(RooFit::Title("Fit GausX"));
    // Creates a frame for the observable 'x' with the title "Fit GausX".

    data->plotOn(xframe);
    // Plots the dataset 'data' on the frame 'xframe'.

    total.plotOn(xframe);
    // Plots the composite PDF 'total' on the frame 'xframe'.

    xframe->Draw();
    // Draws the frame 'xframe' on the canvas.
}
```



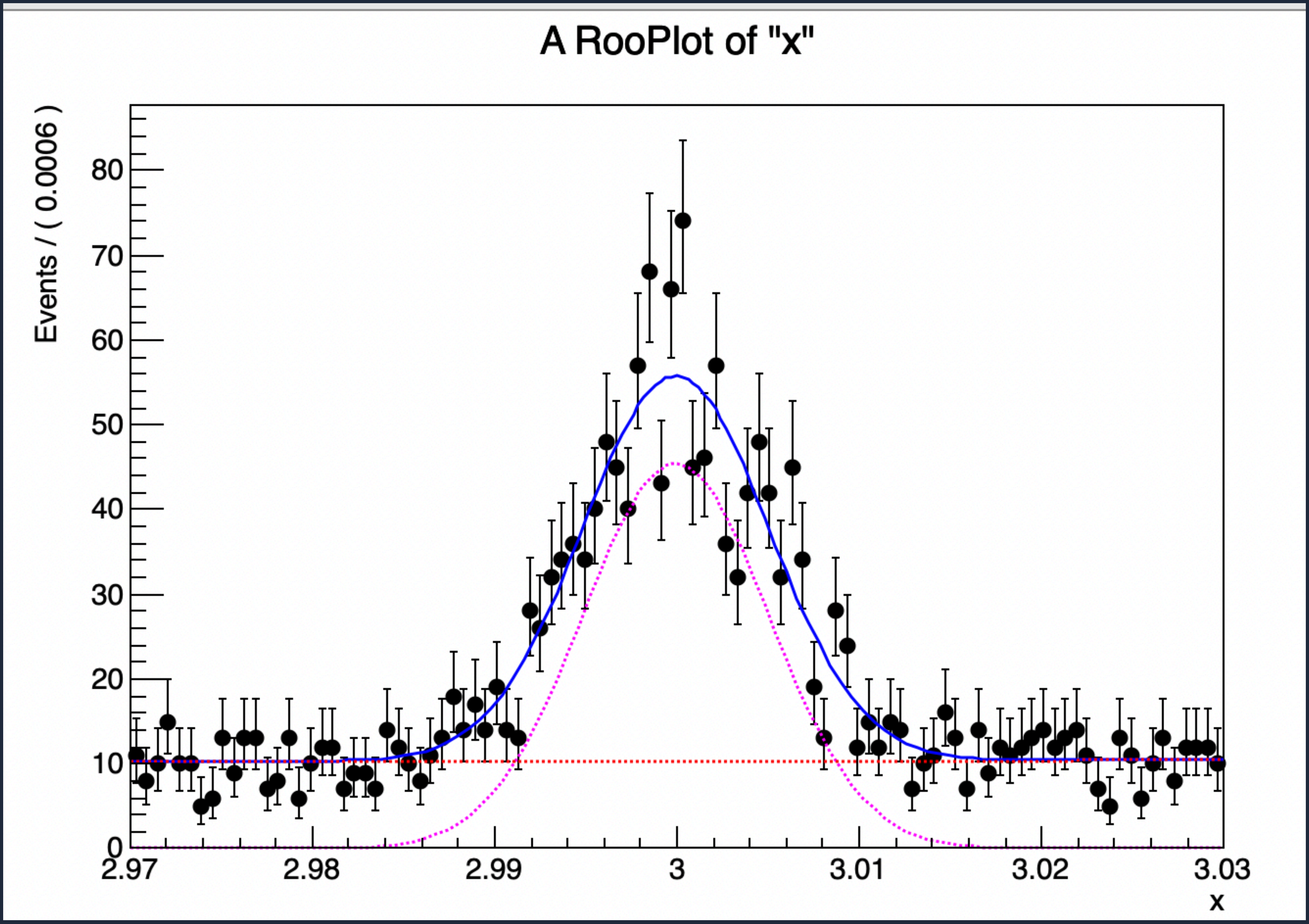
一个函数看起来有些孤单，我们可以用 RooAddPdf 这个函数把两个函数放在一起

RooRealVar fsig("fsig","fsig",0.2,0,1.);

//model(x) = fsig*gaus(x) + (1-fsig)*chev(x)

RooAddPdf model("fsig","fsig",RooArgList(gaus,chev),fsig);

```
void test(){
using namespace RooFit;
//build gaussian and shev pdf
RooRealVar x("x","x",2.97,3.03);
RooRealVar mean("mean","mean",3.0,2.8,3.2);
RooRealVar sigma("sigma","sigma",0.005,0.003,0.007);
RooRealVar a0("a0","a0",0.005,0.004,0.006);
RooAbsPdf* gaus=new RooGaussian("gaus","gaus",x,mean,sigma);
RooAbsPdf* shev=new RooChebychev("shev","shev",x,a0);
RooRealVar n1("n1","n1",0,0,50000);
RooRealVar n2("n2","n2",0,0,50000);
RooExtendPdf*signal1=new RooExtendPdf("sig1","sig1",*gaus,n1);
RooExtendPdf*signal2=new RooExtendPdf("sig2","sig2",*shev,n2);
RooAddPdf totalPdf("total","total",RooArgList(*signal1,*signal2),RooArgList(n1,n2));//totalPdf(x)=signal1(x)+signal2(x)

//generate data point
RooDataSet*data1;
RooDataSet*data2;
data1=gaus->generate(RooArgSet(x),1000);
data2=shev->generate(RooArgSet(x),1000);
data1->append(*data2);

//fit
RooFitResult*result= totalPdf.fitTo(*data1,Save());
RooPlot*xframe=x.frame();
data1->plotOn(xframe);
totalPdf.plotOn(xframe);
totalPdf.plotOn(xframe, Components(*signal1),LineStyle(kDashed),LineColor(6));
totalPdf.plotOn(xframe, Components("sig2"),LineStyle(kDashed),LineColor(2));//two different ways set the Components name
xframe->Draw();
}
```



> *"以上是一维拟合，但很多时候我们会研究两个不同组分，这样就需要进行二维拟合。二维拟合就不单单是把两个PDF加和在一起了，而是需要乘积，可以理解为同时出现两个组分的概率。举例来说，对于x事件，他有发生和不发生两个概率设为p(x) q(x)，对于y事件同样的p(y) q(y)，那么对于所有的事件就有四种情况，这四种情况的概率相对应的就是p(x)*p(y) 是x，y同时发生的概率，其他的相同。所以我们需要构建四个概率函数来表示对于这两个事件不同组合的各种情况出现的概率。"*

Note: 默认的随机生成的种子号是 `0`，一般会根据时间设置种子号，这样每次生成都是不一样的，但是这里可能是为了调试的方便，让种子号为 `0` 时是一个确定的值，所以如果你的程序跟我一样的话生成的数据也是一模一样的，并不是那么随机。你可以 `RooRandom::randomGenerator()->SetSeed(seed)` 来自定义种子号，还有另外一种方式 `gRandom->SetSeed(seed)`，前者是对RooFit中的随机数生成器进行设置，RooFit中的一些需要随机数的函数都会依赖这个值，后者是面向ROOT的全局随机数生成器，当除RooFit之外的其他库中也需要随机数生成的时候，会依赖这个seed。简单来说，后者可以包含前者，你可以使用 `gRandom->GetSeed()` 去查看当前的seed。

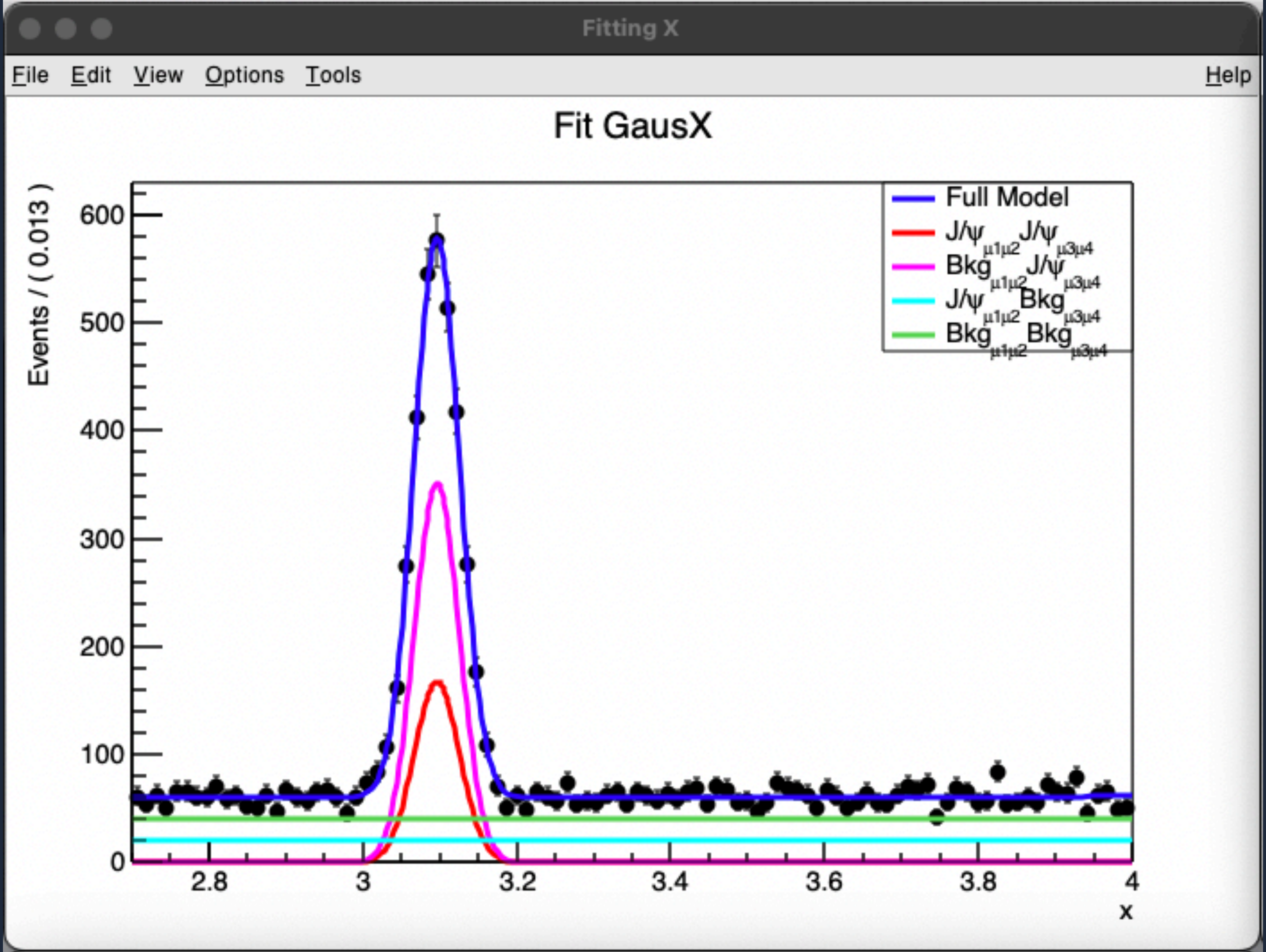
```
TDatime* starttime=new TDatime();
Int_t today=starttime->GetDate();
Int_t clock=starttime->GetTime();
Int_t seed=today+clock;
RooRandom::randomGenerator()->SetSeed(seed);
```

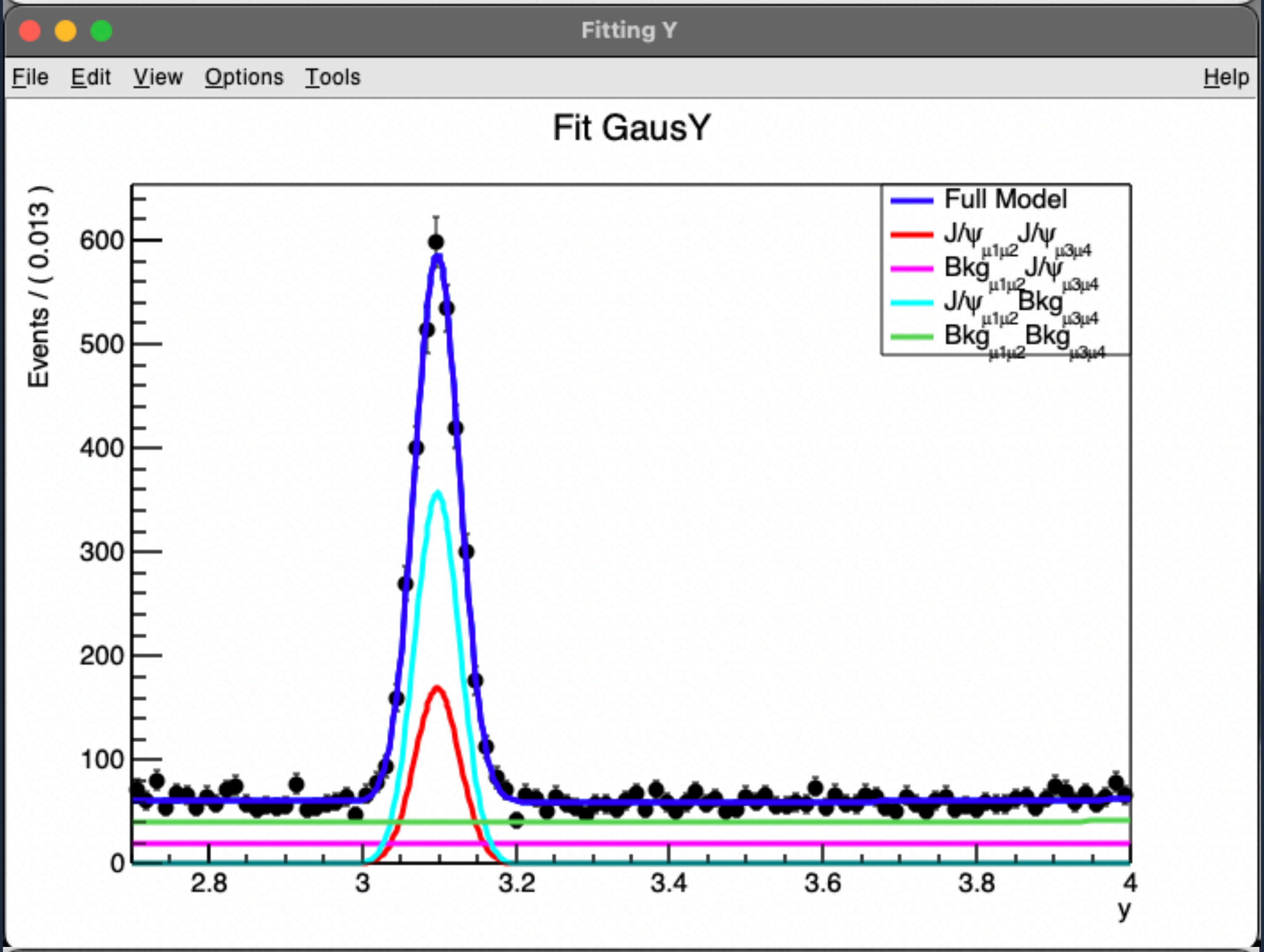
Lesson 3 2D fitting

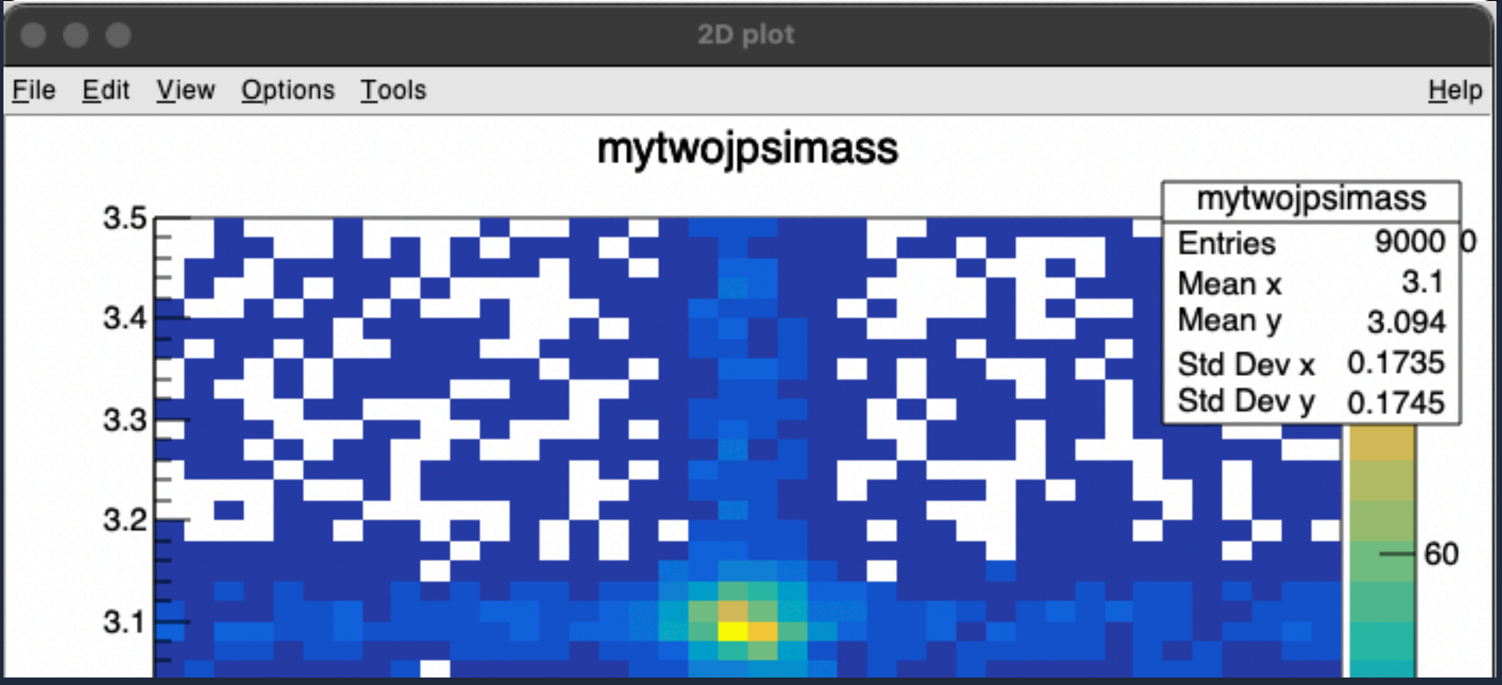
下面展示了两个二维拟合

```
#include <TH2.h>
#include <TStyle.h>
#include <TCanvas.h>
#include <iostream>
#include <fstream>
#include "TF1.h"
#include "TText.h"
#include "TLorentzVector.h"
#include "TVector3.h"
#include "TLorentzRotation.h"
#include "RooNumIntConfig.h" //no matter what, copy directly!

using namespace RooFit;
void test(){
//build gaussian pdf in two dimension
RooRealVar x("x","x",2.7,4);

RooRealVar meanx("meanx","meanx",3.0967,2.9,3.2);
RooRealVar sigmax("sigmax","sigmax",0.03,0.01,0.06);
RooAbsPdf* gausx=new RooGaussian("gausx","gausx",x,meanx,sigmax);

RooRealVar ax0("ax0","ax0",0.005,0.004,0.006);
RooRealVar ax1("ax1","ax1",0.01,0.01,0.02);
RooAbsPdf* shevx=new RooChebychev("shevx","shevx",x,RooArgSet(ax0,ax1));

RooRealVar y("y","y",2.7,4);

RooRealVar meany("meany","meany",3.0967,2.9,3.2);
RooRealVar sigmay("sigmay","sigmay",0.03,0.01,0.06);
RooAbsPdf* gausy=new RooGaussian("gausy","gausy",y,meany,sigmay);

RooRealVar ay0("ay0","ay0",0.005,0.004,0.006);
RooRealVar ay1("ay1","ay1",0.01,0.01,0.02);
RooAbsPdf* shevy=new RooChebychev("shevy","shevy",y,RooArgSet(ay0,ay1));

//constructor with 2 pdf
RooProdPdf* sigxsigy=new RooProdPdf("sigxsigy","sigxsigy",*gausx,*gausy);
RooProdPdf* bkgxsigy=new RooProdPdf("bkgxsigy","bkgxsigy",*shevx,*gausy);
RooProdPdf* sigxbkgy=new RooProdPdf("sigxbkgy","sigxbkgy",*gausx,*shevy);
RooProdPdf* bkgxbkgy=new RooProdPdf("bkgxbkgy","bkgxbkgy",*shevx,*shevy);

RooRealVar n_sigxsigy("n_sigxsigy","n_sigxsigy",0,0,50000);
RooRealVar n_bkgxsigy("n_bkgxsigy","n_bkgxsigy",0,0,50000);
RooRealVar n_sigxbkgy("n_sigxbkgy","n_sigxbkgy",0,0,50000);
RooRealVar n_bkgxbkgy("n_bkgxbkgy","n_bkgxbkgy",0,0,50000);


RooExtendPdf*signal1=new RooExtendPdf("signal1","signal1",*sigxsigy,n_sigxsigy);
RooExtendPdf*signal2=new RooExtendPdf("signal2","signal2",*sigxbkgy,n_sigxbkgy);
RooExtendPdf*signal3=new RooExtendPdf("signal3","signal3",*bkgxsigy,n_bkgxsigy);
RooExtendPdf*signal4=new RooExtendPdf("signal4","signal4",*bkgxbkgy,n_bkgxbkgy);

RooAddPdf totalPdf("total","total",RooArgList(*signal1,*signal2,*signal3,*signal4),RooArgList(n_sigxsigy,n_sigxbkgy,n_bkgxsigy,n_bkgxbkgy));

//generate data point

RooDataSet*data1;
RooDataSet*data2;
RooDataSet*data3;
RooDataSet*data4;
data1=sigxsigy->generate(RooArgSet(x,y),1000);
data2=sigxbkgy->generate(RooArgSet(x,y),2000);
data3=bkgxsigy->generate(RooArgSet(x,y),2000);
data4=bkgxbkgy->generate(RooArgSet(x,y),4000);

data1->append(*data2);
data1->append(*data3);
data1->append(*data4);

//fit
RooFitResult*result= totalPdf.fitTo(*data1,Save());
//x dimension

//create two canvases, because you have to know two dimensions fitting.
TCanvas*c1 =new TCanvas("c1","Fitting X");
c1->cd();
RooPlot*xframe=x.frame(RooFit::Title("Fit GausX"));
data1->plotOn(xframe);
totalPdf.plotOn(xframe,Name("fullModel"));
totalPdf.plotOn(xframe,Components(*signal1),LineColor(2),LineStyle(1),Name("JpsiJpsi"));
totalPdf.plotOn(xframe,Components(*signal2),LineColor(6),LineStyle(1),Name("BkgJpsi"));
totalPdf.plotOn(xframe,Components(*signal3),LineColor(7),LineStyle(1),Name("JpsiBkg"));
totalPdf.plotOn(xframe,Components(*signal4),LineColor(8),LineStyle(1),Name("BkgBkg"));

TLegend leg(0.7, 0.7, 0.9, 0.9);
leg.AddEntry(xframe->findObject("fullModel"), "Full Model", "L");
leg.AddEntry(xframe->findObject("JpsiJpsi"), "J/#psi_{#mu1#mu2}J/#psi_{#mu3#mu4}", "L");
leg.AddEntry(xframe->findObject("BkgJpsi"), "Bkg_{#mu1#mu2}J/#psi_{#mu3#mu4}", "L");
leg.AddEntry(xframe->findObject("JpsiBkg"), "J/#psi_{#mu1#mu2}Bkg_{#mu3#mu4}", "L");
leg.AddEntry(xframe->findObject("BkgBkg"), "Bkg_{#mu1#mu2}Bkg_{#mu3#mu4}", "L");

```

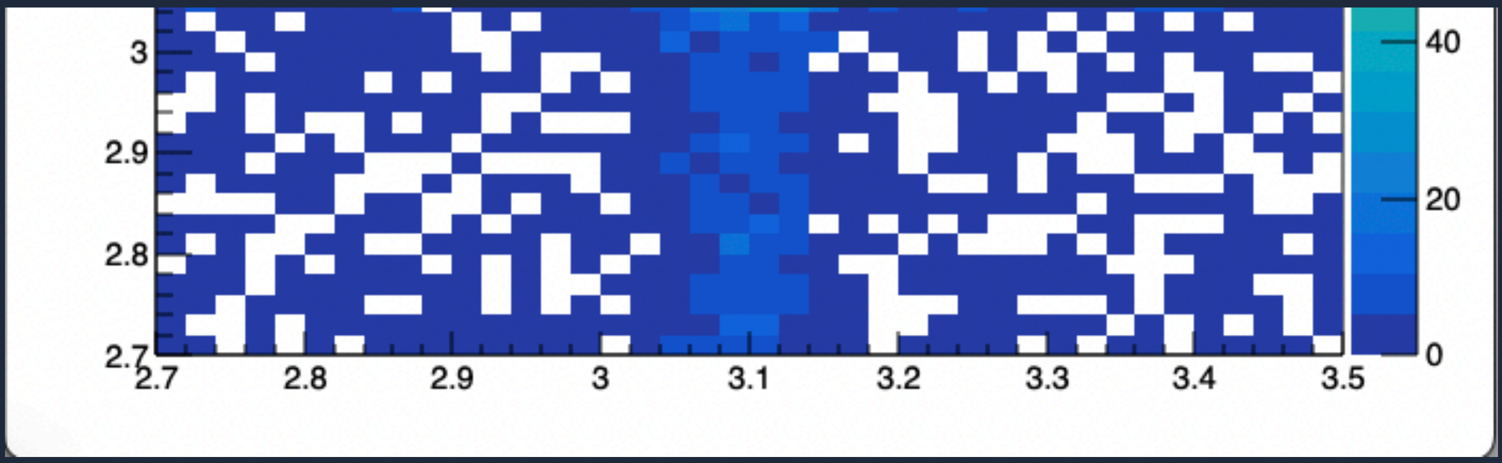
```
xframe->Draw();
leg.DrawClone();

//y dimension
TCanvas*c2 =new TCanvas("c2","Fitting Y");
c2->cd();
RooPlot*yframe=y.frame(RooFit::Title("Fit GausY"));
data1->plotOn(yframe);
totalPdf.plotOn(yframe);
totalPdf.plotOn(yframe,Name("fullModel"));
totalPdf.plotOn(yframe,Components(*signal1),LineColor(2),LineStyle(1),Name("JpsiJpsi"));
totalPdf.plotOn(yframe,Components(*signal2),LineColor(6),LineStyle(1),Name("BkgJpsi"));
totalPdf.plotOn(yframe,Components(*signal3),LineColor(7),LineStyle(1),Name("JpsiBkg"));
totalPdf.plotOn(yframe,Components(*signal4),LineColor(8),LineStyle(1),Name("BkgBkg"));

TLegend leg2(0.7, 0.7, 0.9, 0.9);
leg2.AddEntry(yframe->findObject("fullModel"), "Full Model", "L");
leg2.AddEntry(yframe->findObject("JpsiJpsi"), "J/#psi_{#mu1#mu2}J/#psi_{#mu3#mu4}", "L");
leg2.AddEntry(yframe->findObject("BkgJpsi"), "Bkg_{#mu1#mu2}J/#psi_{#mu3#mu4}", "L");
leg2.AddEntry(yframe->findObject("JpsiBkg"), "J/#psi_{#mu1#mu2}Bkg_{#mu3#mu4}", "L");
leg2.AddEntry(yframe->findObject("BkgBkg"), "Bkg_{#mu1#mu2}Bkg_{#mu3#mu4}", "L");

yframe->Draw();
leg2.DrawClone();

TCanvas*c3 =new TCanvas("c3","2D plot");
c3->cd();
TH2F* mytwojpsimass = new TH2F("mytwojpsimass","mytwojpsimass",40,2.7,3.5,40,2.7,3.5);
data1->fillHistogram(mytwojpsimass,RooArgList(x,y));
mytwojpsimass->Draw("colz");

}
```


Fitting X
File Edit View Options Tools Help
Fit GausX
Events / (0.013)
600
500
400
300
200
100
0
2.8
3
3.2
3.4
3.6
3.8
4
x
Full Model
$J/\psi_{\mu1\mu2}$ $J/\psi_{\mu3\mu4}$
$Bkg_{\mu1\mu2}$ $J/\psi_{\mu3\mu4}$
$J/\psi_{\mu1\mu2}$ $Bkg_{\mu3\mu4}$
$Bkg_{\mu1\mu2}$ $Bkg_{\mu3\mu4}$


Fitting Y
File Edit View Options Tools Help
Fit GausY
Events / (0.013)
600
500
400
300
200
100
0
2.8
3
3.2
3.4
3.6
3.8
4
y
Full Model
$J/\psi_{\mu1\mu2}$ $J/\psi_{\mu3\mu4}$
$Bkg_{\mu1\mu2}$ $J/\psi_{\mu3\mu4}$
$J/\psi_{\mu1\mu2}$ $Bkg_{\mu3\mu4}$
$Bkg_{\mu1\mu2}$ $Bkg_{\mu3\mu4}$

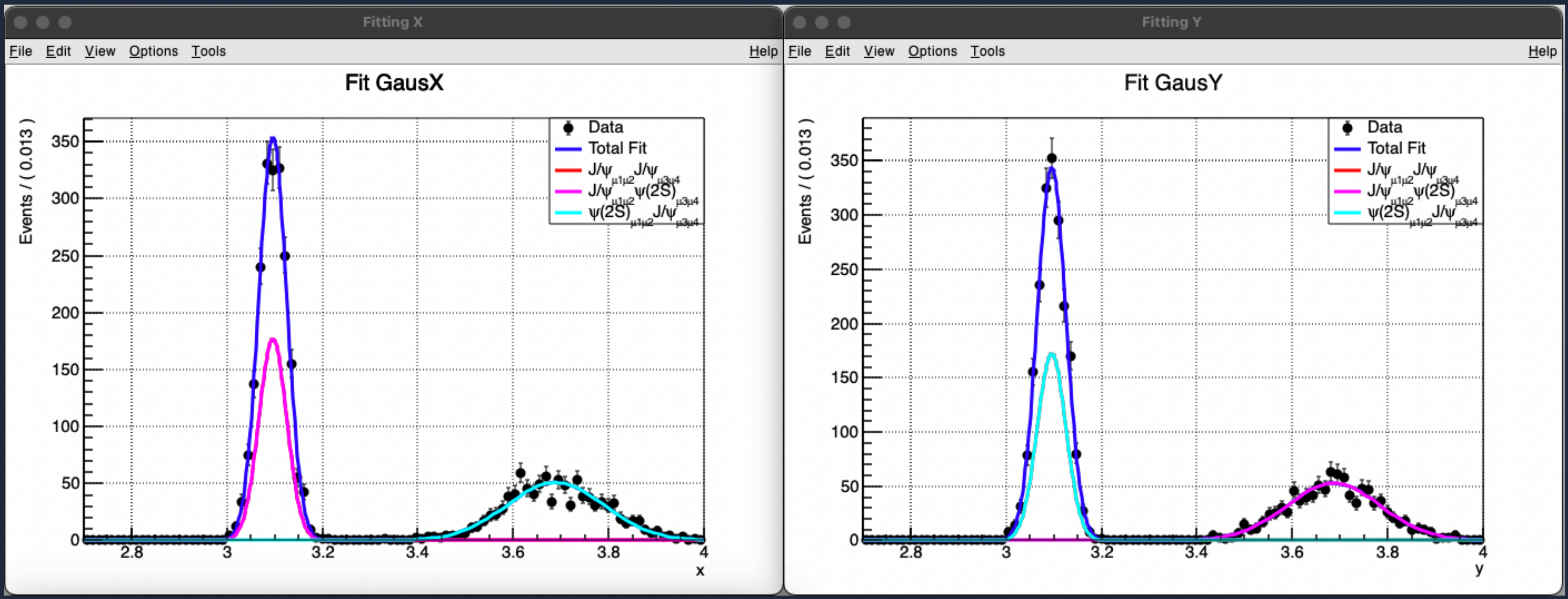


```
mytwojpsimass
Entries      9000 0
Mean x          3.1
Mean y        3.094
Std Dev x    0.1735
Std Dev y    0.1745
```



```cpp
#include <TH2.h>
#include <TStyle.h>
#include <TCanvas.h>
#include <iostream>
#include <fstream>
#include "TF1.h"
#include "TText.h"
#include "TLorentzVector.h"
#include "TVector3.h"
#include "TLorentzRotation.h"
#include "RooNumIntConfig.h" //no matter what, copy directly!
using namespace RooFit;
void test(){
//build gaussian pdf in two dimension
RooRealVar x("x","x",2.7,4);
RooRealVar meanx1("meanx1","meanx1",3.0967,2.9,3.2);
RooRealVar sigmax1("sigmax1","sigmax1",0.03,0.01,0.06);
RooAbsPdf* gausx1=new RooGaussian("gausx1","gausx1",x,meanx1,sigmax1);

RooRealVar meanx2("meanx2","meanx2",3.686,3.5,3.8);
RooRealVar sigmax2("sigmax2","sigmax2",0.1,0.01,0.2);
RooAbsPdf* gausx2=new RooGaussian("gausx2","gausx2",x,meanx2,sigmax2);

RooRealVar y("y","y",2.7,4);

RooRealVar meany1("meany1","meany1",3.0967,2.9,3.2);
RooRealVar sigmay1("sigmay1","sigmay1",0.03,0.01,0.06);
RooAbsPdf* gausy1=new RooGaussian("gausy1","gausy1",y,meany1,sigmay1);

RooRealVar meany2("meany2","meany2",3.686,3.5,3.8);
RooRealVar sigmay2("sigmay2","sigmay2",0.1,0.01,0.2);
RooAbsPdf* gausy2=new RooGaussian("gausy2","gausy2",y,meany2,sigmay2);

//constructor with 2 pdf
RooProdPdf* gausx1y1=new RooProdPdf("gausx1y1","gausx1y1",*gausx1,*gausy1);
RooProdPdf* gausx1y2=new RooProdPdf("gausx1y2","gausx1y2",*gausx1,*gausy2);
RooProdPdf* gausx2y1=new RooProdPdf("gausx2y1","gausx2y1",*gausx2,*gausy1);

RooRealVar n_x1y1("n1","n1",0,0,50000);
RooRealVar n_x1y2("n2","n2",0,0,50000);
RooRealVar n_x2y1("n3","n3",0,0,50000);

RooAddPdf totalPdf("total","total",RooArgList(*gausx1y1,*gausx1y2,*gausx2y1),RooArgList(n_x1y1,n_x1y2,n_x2y1));

//generate data point

RooDataSet*data1;
RooDataSet*data2;
RooDataSet*data3;
data1=gausx1y1->generate(RooArgSet(x,y),1000);
data2=gausx1y2->generate(RooArgSet(x,y),1000);
data3=gausx2y1->generate(RooArgSet(x,y),1000);

data1->append(*data2);
data1->append(*data3);

//fit
RooFitResult*result= totalPdf.fitTo(*data1,Save());
//x dimension
//create two canvases, because you have to know two dimensions fitting.
TCanvas*c1 =new TCanvas("c1","Fitting X");
c1->cd();
RooPlot*xframe=x.frame(RooFit::Title("Fit GausX"));
data1->plotOn(xframe,Name("data1"));
totalPdf.plotOn(xframe,Name("fullModel"));
totalPdf.plotOn(xframe,Components(*gausx1y1),LineColor(2),LineStyle(1),Name("JpsiJpsi"));
totalPdf.plotOn(xframe,Components(*gausx1y2),LineColor(6),LineStyle(1),Name("JpsiPsi2S"));
totalPdf.plotOn(xframe,Components(*gausx2y1),LineColor(7),LineStyle(1),Name("Psi2SJpsi"));

TLegend* leg= new TLegend(0.7, 0.7, 0.9, 0.9);
leg->AddEntry(xframe->findObject("data1"), "Data", "pe");
leg->AddEntry(xframe->findObject("fullModel"), "Total Fit", "L");
leg->AddEntry(xframe->findObject("JpsiJpsi"), "J/#psi_{#mu1#mu2}J/#psi_{#mu3#mu4}", "L");
leg->AddEntry(xframe->findObject("JpsiPsi2S"), "J/#psi_{#mu1#mu2}#psi(2S)_{#mu3#mu4}", "L");
leg->AddEntry(xframe->findObject("Psi2SJpsi"), "#psi(2S)_{#mu1#mu2}J/#psi_{#mu3#mu4}", "L");

xframe->Draw();
leg->Draw();



//y dimension
TCanvas*c2 =new TCanvas("c2","Fitting Y");
c2->cd();
RooPlot*yframe=y.frame(RooFit::Title("Fit GausY"));
```

```
data1->plotOn(yframe,Name("data1"));
totalPdf.plotOn(yframe,Name("fullModel"));
totalPdf.plotOn(yframe,Components(*gausx1y1),LineColor(2),LineStyle(1),Name("JpsiJpsi"));
totalPdf.plotOn(yframe,Components(*gausx1y2),LineColor(6),LineStyle(1),Name("JpsiPsi2S"));
totalPdf.plotOn(yframe,Components(*gausx2y1),LineColor(7),LineStyle(1),Name("Psi2SJpsi"));


TLegend leg2(0.7, 0.7, 0.9, 0.9);
leg2.AddEntry(yframe->findObject("data1"), "Data", "pe");
leg2.AddEntry(yframe->findObject("fullModel"), "Total Fit", "L");
leg2.AddEntry(yframe->findObject("JpsiJpsi"), "J/#psi_{#mu1#mu2}J/#psi_{#mu3#mu4}", "L");
leg2.AddEntry(yframe->findObject("JpsiPsi2S"), "J/#psi_{#mu1#mu2}#psi(2S)_{#mu3#mu4}", "L");
leg2.AddEntry(yframe->findObject("Psi2SJpsi"), "#psi(2S)_{#mu1#mu2}J/#psi_{#mu3#mu4}", "L");

yframe->Draw();
leg2.DrawClone();

TCanvas*c3 =new TCanvas("c3","2D plot");
c3->cd();
TH2F* mytwojpsimass = new TH2F("mytwojpsimass","mytwojpsimass",40,2.7,4,40,2.7,4);
data1->fillHistogram(mytwojpsimass,RooArgList(x,y));
mytwojpsimass->Draw("colz");
```

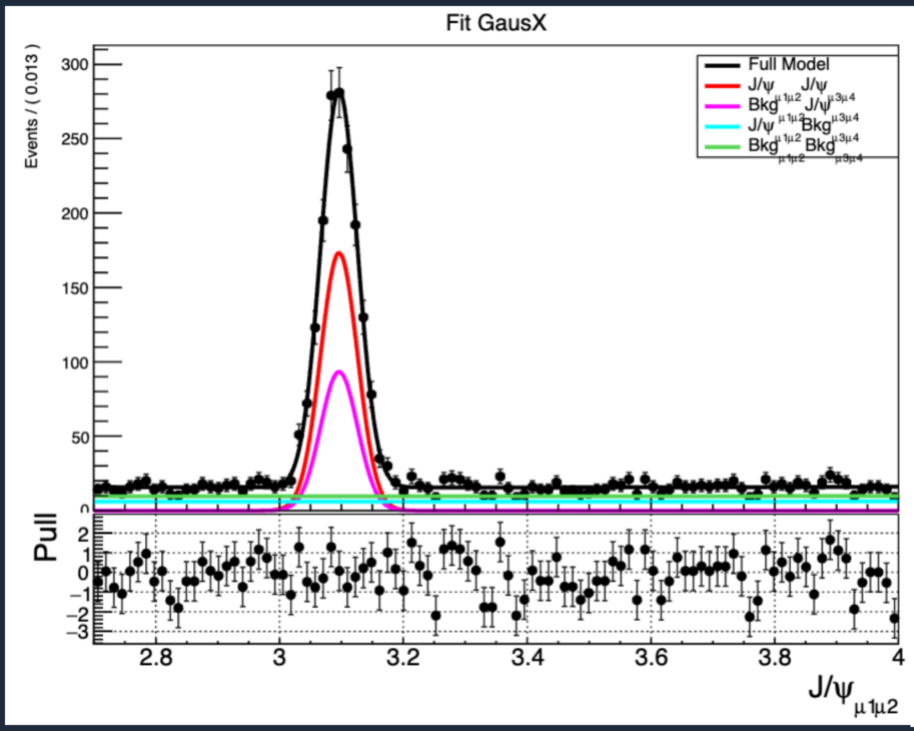


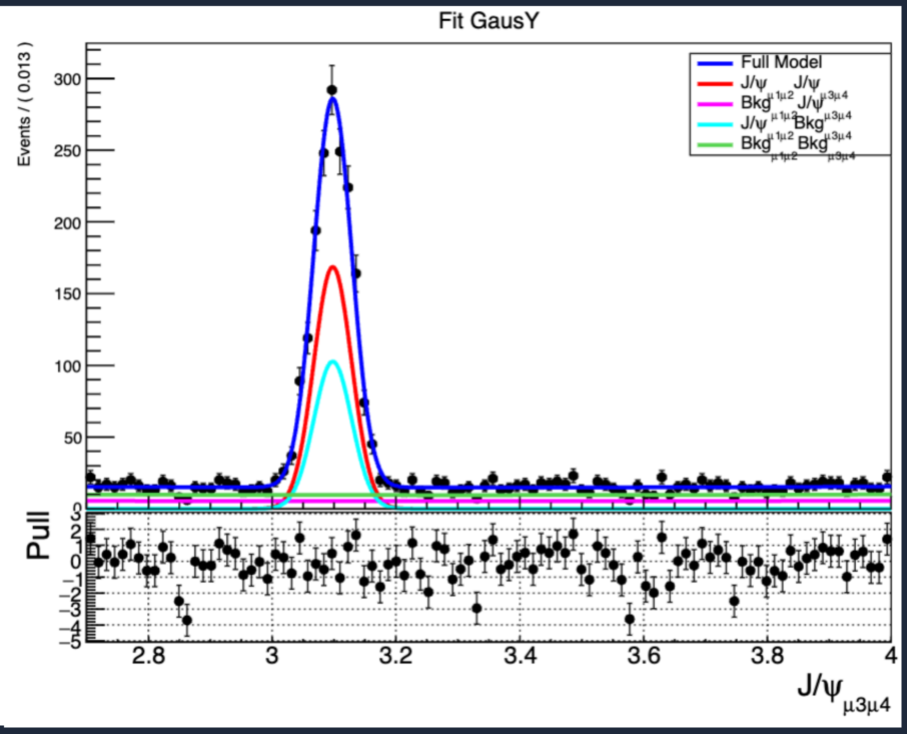
Lesson 3+ Pull Plot

当我们想看拟合曲线和数据点之间拟合是否拟合得很好的话，就需要看pull分布，就相当于将拟合曲线拉直看数据点相对于曲线的距离

```
RooPlot *xfpull=x.frame(RooFit::Title("    "));//创建 RooPlot 对象 xfpull，用于绘制 x 维度的 pull 分布图
```

```
RooHist *pullx=xframe->pullHist("data","all"); //从 xframe 中获取数据和模型的 pull 分布，并将其添加到 xfpull 中，这里要对应你在xframe画的对象的Name()，比如totalPdf.plotOn(xframe,Name("fullModel"));你的pullHist()里面就应该是"fullModel"
xfpull->addPlotable(pullx,"p");
xfpull->GetXaxis()->SetTitle("J/#psi_{#mu1#mu2}");
xfpull->GetXaxis()->SetTitleSize(0.15);
xfpull->GetXaxis()->SetLabelSize(0.12);
xfpull->GetYaxis()->SetTitle("Pull");
xfpull->GetYaxis()->SetTitleSize(0.15);
xfpull->GetYaxis()->SetLabelSize(0.1);
xfpull->GetYaxis()->SetTitleOffset(0.2);

//在画布 c2 上创建两个子画布 pad21 和 pad22，分别用于绘制拟合结果和 pull 分布图。
//在 pad11 上绘制 y 维度的拟合结果 xframe 和图例 leg，TPad的位置和TLegend的类似，都是相对于TCanvas的位置关系
TCanvas c1("c1","c1",800,600);
c1.cd();
TPad pad11("pad11","pad11",0,0.3,1,1.0);
pad11.SetTopMargin(0.08);
pad11.SetBottomMargin(0.005);
pad11.Draw();
pad11.cd();
xframe->Draw();
leg.Draw();
//在 pad12 上绘制 y 维度的 pull 分布图 yfpull
c1.cd();
TPad pad12("pad12","pad12",0,0.0,1,0.3);
pad12.SetTopMargin(0.005);
pad12.SetBottomMargin(0.4);
pad12.SetGridx();
pad12.SetGridy(2);
pad12.Draw();
pad12.cd();
xfpull->Draw();

c1.Update();
c1.SaveAs("c1.pdf");
```


Fit GausX
Full Model
J/ψ J/ψ
Bkg J/ψ
J/ψ Bkg
Bkg Bkg
Events / (0.013)
300
250
200
150
100
50
0
Pull
2
1
0
−1
−2
−3
2.8
3
3.2
3.4
3.6
3.8
4
$J/\psi_{\mu1\mu2}$


Fit GausY
Full Model
J/ψ J/ψ
Bkg J/ψ
J/ψ Bkg
Bkg Bkg
Events / (0.013)
300
250
200
150
100
50
0
Pull
3
2
1
0
−1
−2
−3
−4
−5
2.8
3
3.2
3.4
3.6
3.8
4
$J/\psi_{\mu3\mu4}$