ROOT: Functions & Fitting

Harinder Singh Bawa
California State University Fresno

Mar 31, 2016

Review of previous Exercise*:

* Good to practice some exercises side by side in order to understand

Exercise:

1)

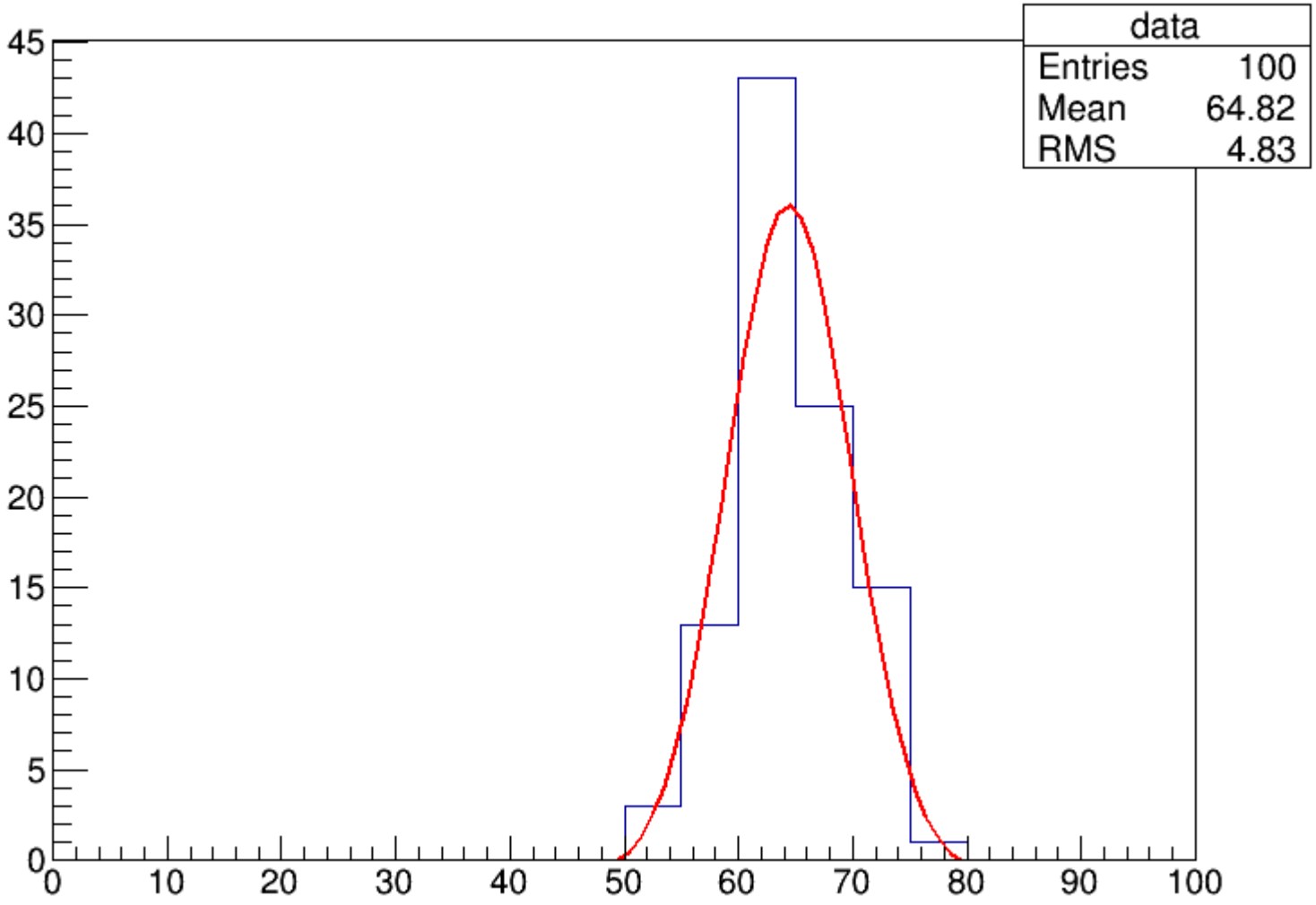
- Create a gaussian function using **TF1** class of Root, set its parameters(500.,65.,5.), plot it and finally save the plot.

Hint: Gaussian function:

TF1 f1("gauss", "[0] / sqrt(2.0 * TMath::Pi()) / [2] * exp(-(x-[1])*(x-[1])/2./[2]/[2])", 0, 100)

- Create a gaussian distributed random numbers using the Random number generator class **TRandom3** and using the provided basic Random distribution "Gaus(mean=65,sigma=5)". Create a 1-dimensional histogram **TH1D** and fill in with the generated random numbers. Finally book a canvas **TCanvas** and plot the distribution and save it
- Fit the histogram from (2) with function (1)

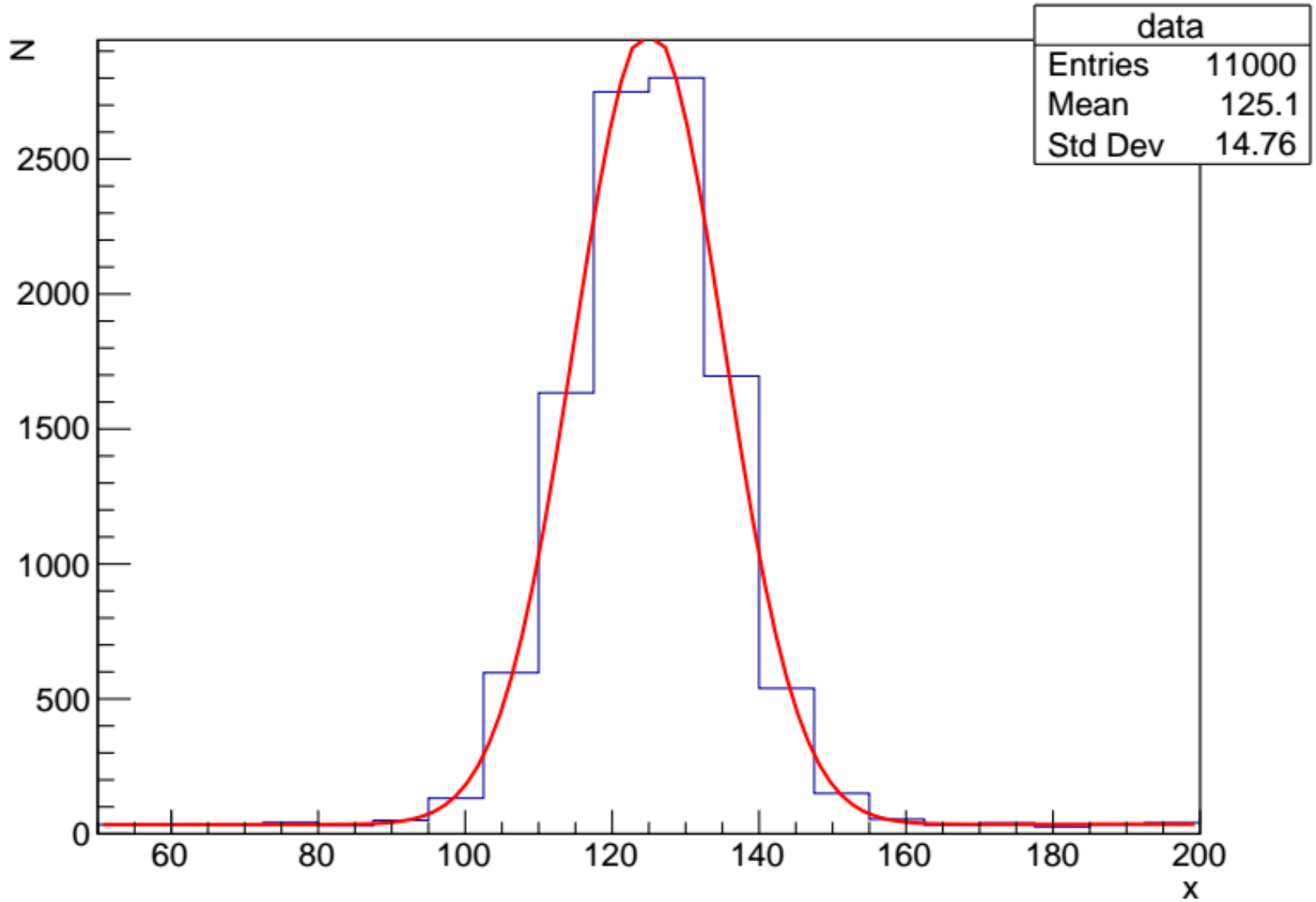
```
void fit_bgsig()
{

gRandom=new TRandom3();

//Define a histogram
TH1D *hist= new TH1D("data","",20,0.0,100.0);

//Fill histogram with Random Number using random distribution "Gaus"
for(int i=0;i<100;i++)
hist->Fill(gRandom->Gaus(65.0,5.0));

//Define Gaussian function with Mean=65.& RMS=5.(Same as histogram)
TF1 *f1=new TF1("myfunc","[0]/sqrt(2.0*TMath::Pi())/[2]*exp(-(x-[1])*(x-[
1])/2./[2]/[2])+[3]",0,100);

f1->SetParNames("Constant","Mean","Sigma","Flat");
f1->SetParameters(500.,hist->GetMean(),hist->GetRMS(),2.);


TCanvas *c1=new TCanvas("c1","fit sig+bg", 5,5,800,600);
TFile f("fitting.root","RECREATE");
hist->Fit("myfunc");
hist->Draw();
hist->Write();
f.Close();
hist->SaveAs("fit_ngsig.eps");
}
```


data
Entries 100
Mean 64.82
RMS 4.83
45
40
35
30
25
20
15
10
5
0
0
10
20
30
40
50
60
70
80
90
100

2) Write a root macro that creates randomly generated data as a signal peak (gaussian) with mean= 125.0 & sigma=10.0. Perform fit with a gaussian function and inspect the parameters .

- Add background as uniform
- Fit using function gaus+pol2
- Write down the good parameters.

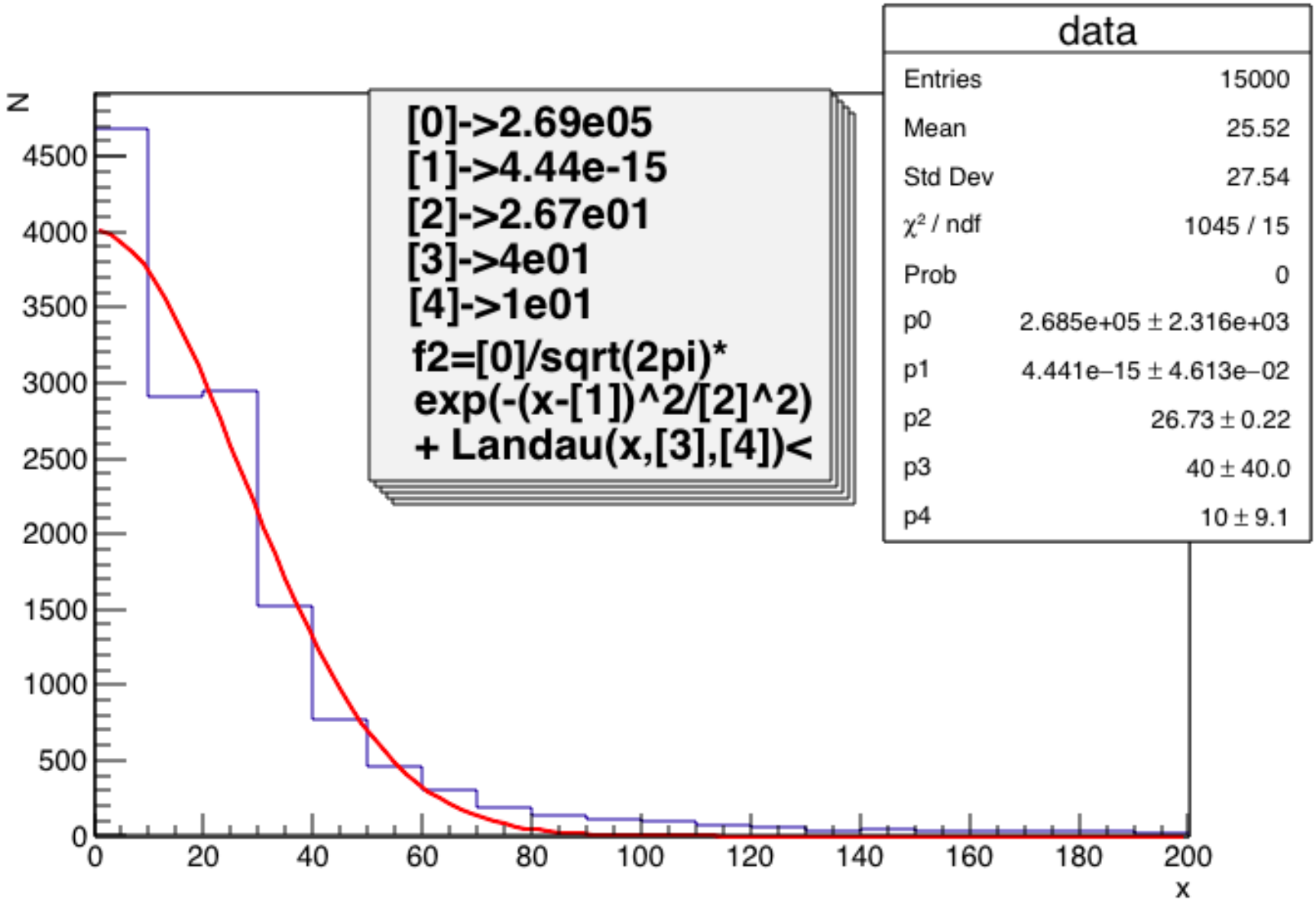
```
root [0] gRandom=new TRandom3()
(class TRandom *) 0x7fda455c0600
root [1] TH1D * hist = new TH1D("data",";x;N",20,50,200)
(class TH1D *) 0x7fda458b9010
root [2] for (int i = 0; i < 1000; ++i)hist->Fill(200.0 * gRandom->Uniform())
root [3] for (int i = 0; i < 10000; ++i)hist->Fill(gRandom->Gaus(125,10))
root [4] TF1 *f2 = new TF1("myfunc","[0] / sqrt(2.0 * TMath::Pi()) / [2] * exp(-(x-[1])*(x-[1
])/2./[2]/[2]) + [3] + [4] * x * x", 0, 200)
(class TF1 *) 0x7fda43533a40
root [5] f2->SetParNames("Constant","Mean","Sigma","Flat","pol2")
root [6] f2->SetParameters(11000,hist->GetMean(),hist->GetRMS(),20,0.0005)
root [7] f2->SetParLimits(0,9000,13000)
root [8] f2->SetParLimits(1,100,150)
root [9] f2->SetParLimits(2,20,60)
root [10] f2->SetParLimits(3,0,50)
root [11] f2->SetParLimits(4,0,1)
root [12] f2->SetLineWidth(2)
root [13] f2->SetLineColor(2)
root [14] TCanvas * c1 = new TCanvas("c1", "fit",5,5,800,600)
(class TCanvas *) 0x7fda4354cd30
root [15] hist->Fit("myfunc")
Info in ROOT::Math::ParameterSettings>: lower/upper bounds outside current parameter value. T
he value will be set to (low+up)/2
 FCN=7476.63 FROM MIGRAD    STATUS=CONVERGED     228 CALLS          229 TOTAL
                     EDM=3.69806e-09    STRATEGY= 1      ERROR MATRIX ACCURATE
  EXT PARAMETER                                   STEP         FIRST
  NO.   NAME      VALUE            ERROR          SIZE      DERIVATIVE
   1  Constant     1.30000e+04   7.55087e+00   3.67019e-03** at limit **
   2  Mean         1.25314e+02   4.13529e-01   6.98473e-04   1.13793e-03
   3  Sigma        2.00000e+01   8.43365e-03   1.22623e-03** at limit **
   4  Flat         3.42013e+01   1.69370e+00   3.07688e-03   1.60653e-04
   5  pol2         1.32894e-13   8.59659e-05   7.82025e-04** at limit **
(TFitResultPtr) @0x7fda45c353c0
```


data
Entries 11000
Mean 125.1
Std Dev 14.76
N
2500
2000
1500
1000
500
0
60
80
100
120
140
160
180
200
x

Full Exercise upto now

- Fill a histogram randomly (n=~10,000) with a Landau distribution with a most probable value at 20 and a “width” of 5 (use the ROOT website to find out about the Landau function)
- Fill the same histogram randomly (n=~5,000) with a Gaussian distribution centered at 5 with a “width” of 3
- Write a compiled script with a fit function that describes the total histogram nicely (it might be a good idea to fit both peaks individually first and use the fit parameters for a combined fit)
- Add titles to x- and y-axis
- Include a legend of the histogram with number of entries, mean, and RMS values
- Add text to the canvas with the fitted function parameters
- Draw everything on a square-size canvas (histogram in blue, fit in red)
- Save as png, eps, and root file

```
[root [0] gRandom=new TRandom3()
 (class TRandom *) 0x7fc96a526800
[root [1] TH1D * hist = new TH1D("data",";x;N",20,0,200)
 (class TH1D *) 0x7fc96b97d510
[root [2] for (int i = 0; i < 10000; ++i)hist->Fill(gRandom->Landau(20,5))
[root [3] hist->Draw()
 Info in <TCanvas::MakeDefCanvas>:  created default TCanvas with name c1
[root [4] for (int i = 0; i < 5000; ++i)hist->Fill(gRandom->Gaus(5,3))
[root [5] hist->Draw()
 root [6] TF1 *f2 = new TF1("myfunc","[0] / sqrt(2.0 * TMath::Pi()) / [2] * exp(-(x-[1])*(x-[1
 ])/2./[2]/[2]) + TMath::Landau(x,[3],[4])", 0, 200)
 (class TF1 *) 0x7fc96969bed0
[root [7] f2->SetParameters(70000,hist->GetMean(),hist->GetRMS(),20,5)
[root [8] f2->SetParLimits(0,60000,80000)
[root [9] f2->SetParLimits(1,10,100)
[root [10] f2->SetParLimits(1,0,10)
[root [11] f2->SetParLimits(2,0,40)
[root [12] f2->SetParLimits(3,0,40)
[root [13] f2->SetParLimits(4,0,10)
[root [14] hist->Fit("myfunc")
 Info in ROOT::Math::ParameterSettings>: lower/upper bounds outside current parameter value. T
 he value will be set to (low+up)/2
  FCN=5446.85 FROM MIGRAD    STATUS=CONVERGED     237 CALLS         238 TOTAL
                     EDM=6.0734e-08    STRATEGY= 1      ERROR MATRIX ACCURATE
   EXT PARAMETER                                   STEP         FIRST
   NO.   NAME      VALUE            ERROR          SIZE      DERIVATIVE
    1  p0           8.00000e+04   1.43744e+01   1.93264e-03** at limit **
    2  p1           1.00000e+01   4.96284e-03   1.60588e-03** at limit **
    3  p2           1.32694e+01   2.91448e-01   5.57703e-04   6.07687e-03
    4  p3           4.00000e+01   3.81453e+01   9.09214e-02** at limit **
    5  p4           1.00000e+01   7.81224e+00   6.11331e-02** at limit **
 (TFitResultPtr) @0x7fc96bc5fc20
```


data
Entries 15000
Mean 25.52
Std Dev 27.54
χ² / ndf 1045 / 15
Prob 0
p0 2.685e+05 ± 2.316e+03
p1 4.441e−15 ± 4.613e−02
p2 26.73 ± 0.22
p3 40 ± 40.0
p4 10 ± 9.1
[0]->2.69e05
[1]->4.44e-15
[2]->2.67e01
[3]->4e01
[4]->1e01
f2=[0]/sqrt(2pi)*
exp(-(x-[1])^2/[2]^2)
+ Landau(x,[3],[4])<
N
x
0
500
1000
1500
2000
2500
3000
3500
4000
4500
0
20
40
60
80
100
120
140
160
180
200

Simple Straight Line fitting

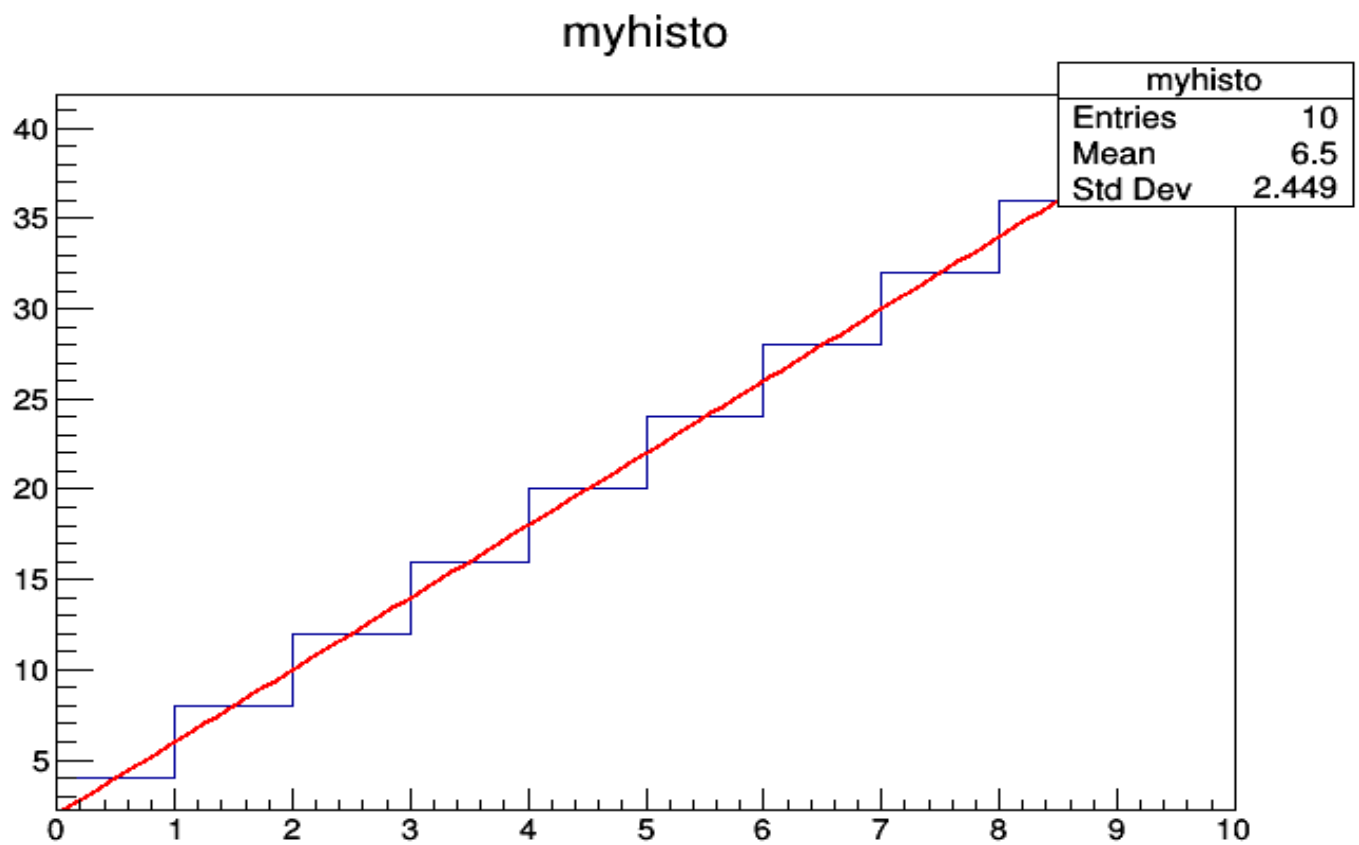
Straight line fit: y=mx + c

```
#include <iostream>
#include "TH1D.h"
#include <TMath.h>
#include <TROOT.h>
#include "TF1.h"
using namespace std;

//Declare a user defined function for fitting
Double_t fitf(Double_t *x, Double_t *par)
{
  Double_t value=par[0]+(par[1]*x[0]); //Here par[0]=constant/intercept & par[1] is Gradient
  return value;
}

int run(){
  //Create a histogram to be fitted
  TH1D* myhisto=new TH1D("myhisto","myhisto",10,0,10);
  //Fill the histogram
  for (int i=1; i<=10; i++){
    myhisto->SetBinContent(i,i*4);  //Randomly
}
```

Contd:

```
//Create a TF1 object with the function defined above..
 //The last three parameters specify the range and number of parameters of the function.
 TF1 *func=new TF1("fit", fitf,0,10,2);

 //Set the initial paramters of the function
func->SetParameters(0.04,0.1);

//Give the parameters meaningful name

 func->SetParNames("intercept","gradient");

 //Call TH1::Fit with the name of TF1 Object
 myhisto->Fit("fit");

 //Access the Fit results directly
 cout<<"The y-intercept is:  "<<func->GetParameter(0)<<endl;
 cout<<"The gradient(constant) is: "<<func->GetParameter(1)<<endl;
 cout<<"Fitting completed...."<<endl;

 return 0; }
```

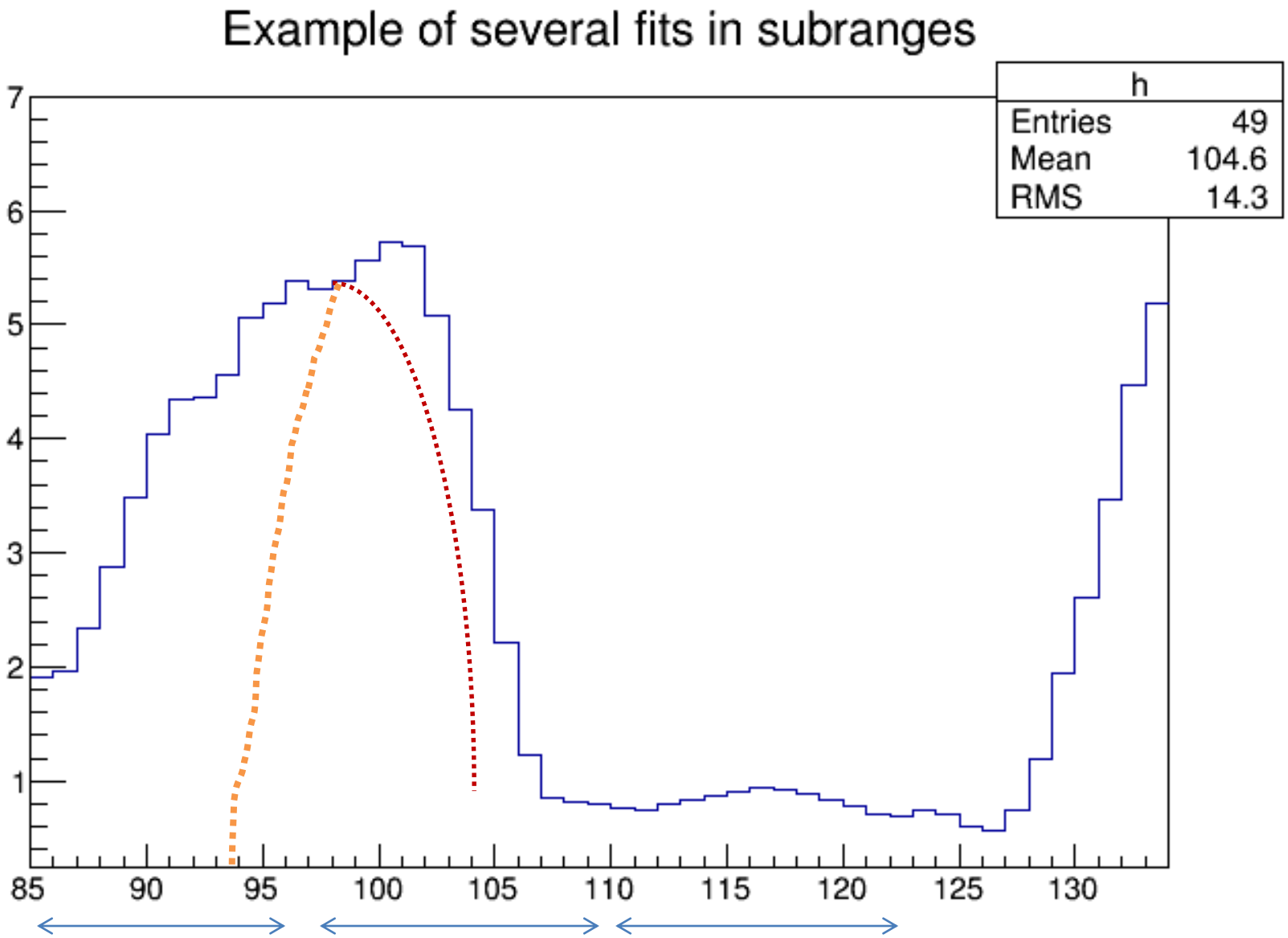
```
harinder@harinder-Latitude-E6430:~/root_examples$ root -l run.C
root [0]
Processing run.C...
Info in <TCanvas::MakeDefCanvas>:  created default TCanvas with name c1
 FCN=1.43736e-23 FROM MIGRAD    STATUS=CONVERGED      30 CALLS          31 TOTAL
                     EDM=2.87472e-23    STRATEGY= 1      ERROR MATRIX ACCURATE
  EXT PARAMETER                                   STEP         FIRST
  NO.   NAME      VALUE            ERROR          SIZE      DERIVATIVE
   1  intercept    2.00000e+00   1.73039e+00   5.70614e-04   1.95947e-12
   2  gradient     4.00000e+00   4.37916e-01   1.44407e-04  -1.07963e-11
The y-intercept is:  2
The gradient(constant) is: 4
Fitting completed....
(int) 0
```



Complex fitting

Multi Peak Histogram Fitting

```
#include "TH1.h"
#include "TF1.h"
void multifit() {
  const Int_t np = 49;
  Float_t x[np] = {1.913521, 1.953769, 2.347435, 2.883654, 3.493567,
                   4.047560, 4.337210, 4.364347, 4.563004, 5.054247,
                   5.194183, 5.380521, 5.303213, 5.384578, 5.563983,
                   5.728500, 5.685752, 5.080029, 4.251809, 3.372246,
                   2.207432, 1.227541, 0.8597788,0.8220503,0.8046592,
                   0.7684097,0.7469761,0.8019787,0.8362375,0.8744895,
                   0.9143721,0.9462768,0.9285364,0.8954604,0.8410891,
                   0.7853871,0.7100883,0.6938808,0.7363682,0.7032954,
                   0.6029015,0.5600163,0.7477068,1.188785, 1.938228,
                   2.602717, 3.472962, 4.465014, 5.177035};
  TH1F *h = new TH1F("h","Example of several fits in subranges",np,85,134);
  h->SetMaximum(7);
  for (int i=0;i<np;i++) {
    h->SetBinContent(i+1,x[i]);
  }
h->Draw();
```

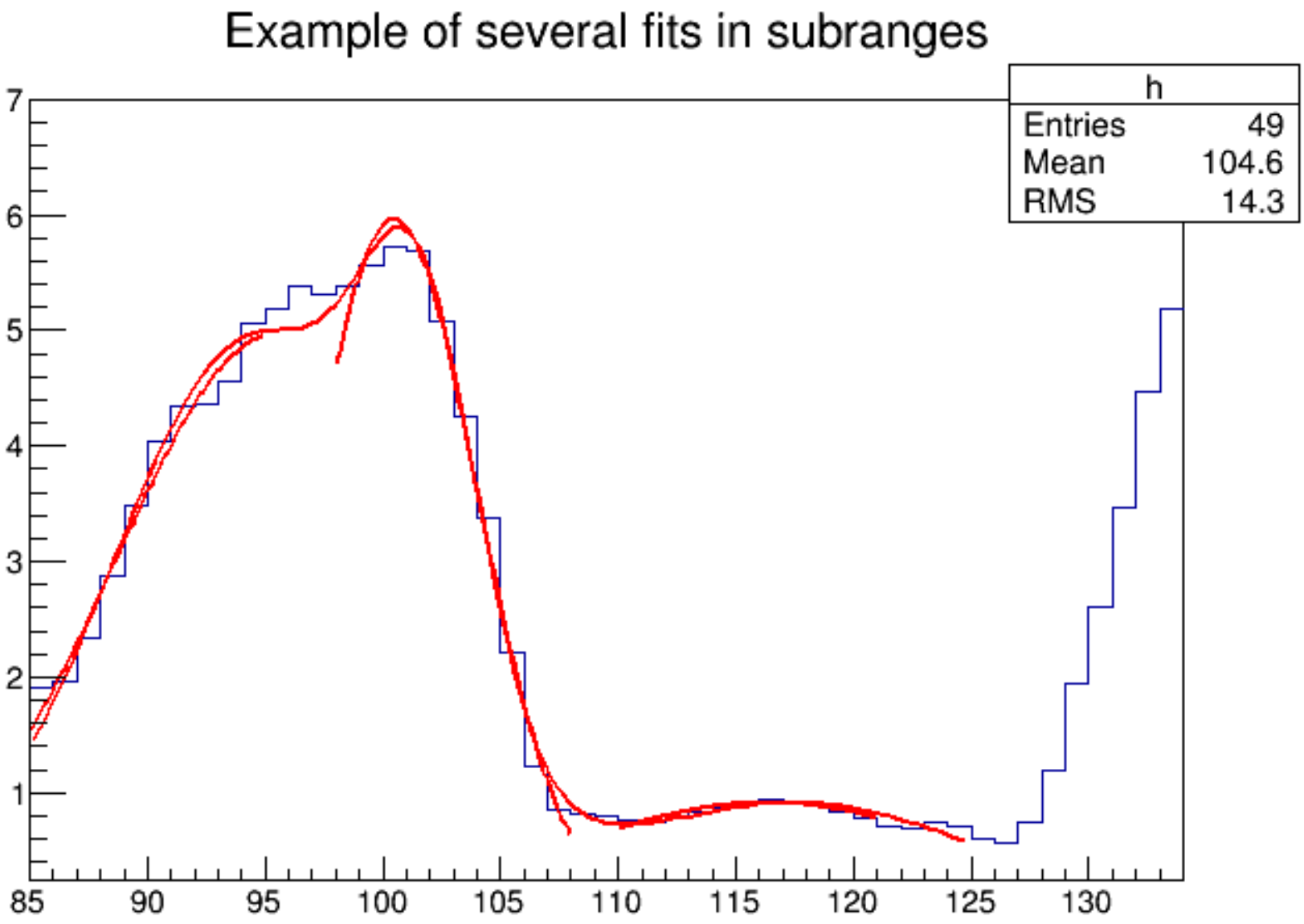

Example of several fits in subranges
h
Entries 49
Mean 104.6
RMS 14.3
7
6
5
4
3
2
1
85
90
95
100
105
110
115
120
125
130

Fitting (Fitting multiple functions to different ranges of a 1-D histogram)

```
Double_t par[9];
//3 gaussians are fitted in sub-ranges of this histogram
  TF1 *g1    = new TF1("g1","gaus",85,95);//sub-Range
  TF1 *g2    = new TF1("g2","gaus",98,108);
  TF1 *g3    = new TF1("g3","gaus",110,121);
//A new function (a sum of 3 gaussians) is fitted on another subrange
  TF1 *total = new TF1("total","gaus(0)+gaus(3)+gaus(6)",85,125);
  total->SetLineColor(2);
  h->Fit(g1,"R"); //Fit according to sub-Range only
  h->Fit(g2,"R+");
  h->Fit(g3,"R+");
  //Note that when fitting simple functions, such as gaussians, the initial values of
parameters are automatically computed by ROOT.
//  In the more complicated case of the sum of 3 gaussians, the initial values of
parameters must be given. In this particular case, the initial values are taken from the
result of the individual fits
    g1->GetParameters(&par[0]);
    g2->GetParameters(&par[3]);
    g3->GetParameters(&par[6]);
    total->SetParameters(par);
   h->Fit(total,"R+"); }
```

https://root.cern.ch/doc/master/classTF1.html

```
root [1] .x multifit.C
Warning in <TROOT::Append>: Replacing existing TH1: h (Potential memory leak).
Info in <TCanvas::MakeDefCanvas>:  created default TCanvas with name c1
 FCN=0.0848003 FROM MIGRAD    STATUS=CONVERGED     105 CALLS          106 TOTAL
                     EDM=1.77382e-007    STRATEGY= 1      ERROR MATRIX ACCURATE

  EXT PARAMETER                                   STEP         FIRST
  NO.   NAME      VALUE            ERROR          SIZE      DERIVATIVE
   1  Constant    4.96664e+000  2.83221e+000  4.26889e-004  1.67619e-004
   2  Mean        9.54663e+001  1.23905e+001  7.53972e-004 -2.63161e-004
   3  Sigma       6.82779e+000  7.49131e+000  5.87496e-005  3.68521e-003
 FCN=0.0771026 FROM MIGRAD    STATUS=CONVERGED      72 CALLS           73 TOTAL
                     EDM=2.00364e-007    STRATEGY= 1      ERROR MATRIX ACCURATE

  EXT PARAMETER                                   STEP         FIRST
  NO.   NAME      VALUE            ERROR          SIZE      DERIVATIVE
   1  Constant    5.96312e+000  1.14355e+000  4.82019e-004  1.52951e-004
   2  Mean        1.00467e+002  1.53372e+000  3.74926e-004  6.69980e-004
   3  Sigma       3.54806e+000  1.16899e+000  3.22077e-005  3.86167e-003
 FCN=0.0087702 FROM MIGRAD    STATUS=CONVERGED      93 CALLS           94 TOTAL
                     EDM=5.57239e-007    STRATEGY= 1      ERROR MATRIX ACCURATE

  EXT PARAMETER                                   STEP         FIRST
  NO.   NAME      VALUE            ERROR          SIZE      DERIVATIVE
   1  Constant    9.12665e-001  4.37176e-001  1.46528e-004  2.91010e-004
   2  Mean        1.16309e+002  8.37408e+000  3.57386e-003 -3.17966e-005
   3  Sigma       8.38413e+000  1.84577e+001  4.99414e-004 -4.98793e-004
 FCN=0.312817 FROM MIGRAD    STATUS=CONVERGED     515 CALLS          516 TOTAL
                     EDM=1.73245e-007    STRATEGY= 1      ERROR MATRIX ACCURATE

  EXT PARAMETER                                   STEP         FIRST
  NO.   NAME      VALUE            ERROR          SIZE      DERIVATIVE
   1  p0          4.91145e+000  1.41387e+000  3.61239e-004 -3.22790e-004
   2  p1          9.44525e+001  3.71612e+000  5.60861e-004 -6.78941e-005
   3  p2          5.94796e+000  2.41732e+000  4.25396e-004  2.68176e-005
   4  p3          3.22134e+000  3.11650e+000  5.86729e-004 -1.82620e-004
   5  p4          1.01663e+002  1.67863e+000  5.56527e-004  3.95769e-004
   6  p5          2.48454e+000  1.91461e+000  3.85832e-004  7.23818e-005
   7  p6          9.11463e-001  3.68235e-001  1.45489e-004  5.77239e-004
   8  p7          1.17582e+002  5.06329e+000  2.01798e-003 -8.25382e-005
   9  p8          7.58627e+000  8.76000e+000  2.12468e-003  2.02614e-005
root [2]
```


Example of several fits in subranges
h
Entries 49
Mean 104.6
RMS 14.3
7
6
5
4
3
2
1
85
90
95
100
105
110
115
120
125
130

Exercise

- First create an histogram with a double gaussian peaks and then we fit it.
- Create an histogram with 100 bins between 0 and 10.
- Fill with 5000 Gaussian number with mean 3 and width 1.5.
- Fill with 5000 Gaussian number with a mean of 8 and width of 1.
- Create a function composed of the sum of two Gaussian and fit to the histogram.
- Do the fit works ? What do you need to do to make the fit working ?

Hint

Before fitting you need to set sensible parameter values. You can do this by fitting first a single Gaussian in the range [0,5] and then a second Gaussian in the range [5,10].If you don't set good initial parameter values, (for example if you set the amplitude of the Gaussians to 1, the means of the Gaussians to 5 and widths to 1), the fit will probably not converge.

BACKUP

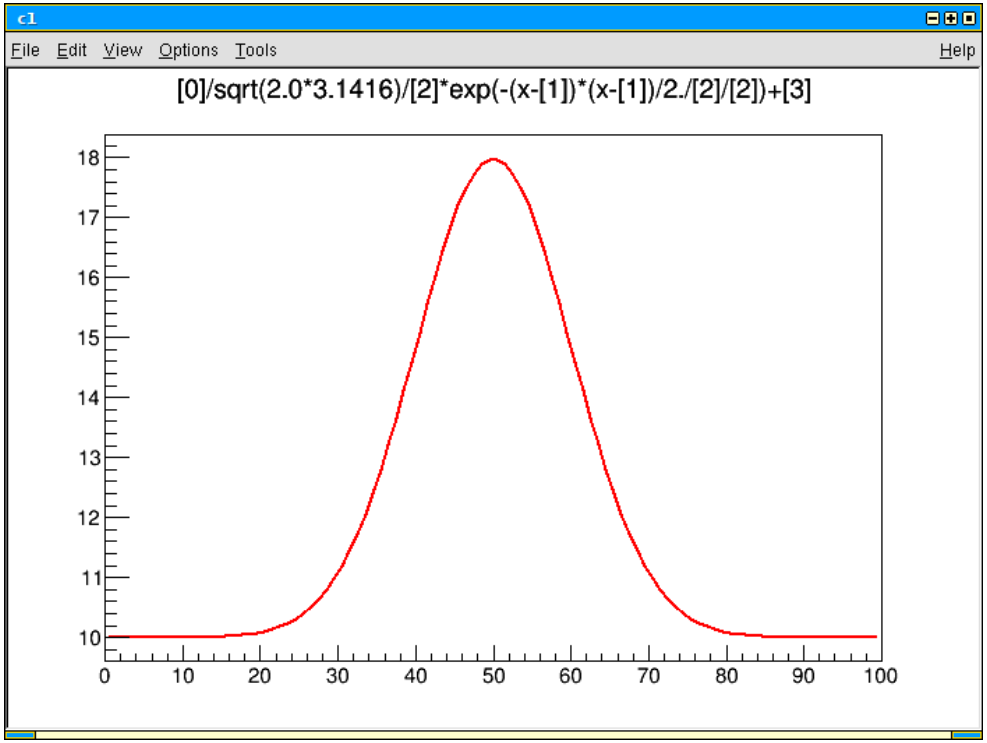
Functions

TF1 with parameters

A function can have **parameters** (e.g. floating parameters for fits...)

```
{

  TF1 * f1 = new TF1("f1",
                     "[0] / sqrt(2.0 * 3.1416) / [2] * exp(-(x-[1])*(x-[1])/2./[2]/[2]) + [3]",
                     0., 100.);
  f1->SetParameter(0, 200.0);
  f1->SetParameter(1, 50.0);
  f1->SetParameter(2,  10);
  f1->SetParameter(3,  10);
  f1->Draw();

}
```

```
root [6] f1->GetParameter(0)
(const Double_t)2.00000000000000000e+02
root [7] f1->GetParameter(1)
(const Double_t)5.00000000000000000e+01
root [8] f1->GetParameter(2)
(const Double_t)1.00000000000000000e+01
root [9] f1->GetParameter(3)
(const Double_t)1.00000000000000000e+01
```



Let's draw a TF1 on a TCanvas

Like most objects in ROOT, functions can be **drawn** on a **canvas**

a **TCanvas** is an object too...

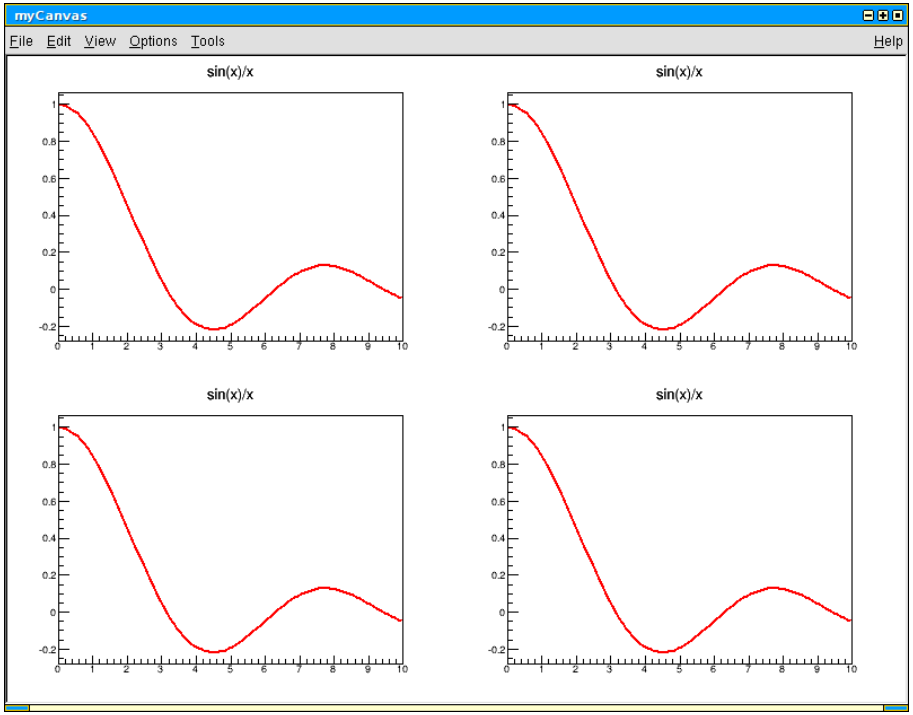
```
root [10] TCanvas c("myCanvas", "myCanvas"
, 800, 600)
```

```
root [3] TF1 of("myFunction","sin(x)/x",0,10)
root [4] of.Draw()
Info in <TCanvas::MakeDefCanvas>:  created default TCanvas
with name_c1
```

...it can be divided in **TPads**

```
root [1] c.Divide(2,2)
```

```
root [3] c.cd(2)
(class TVirtualPad*)0x242a890
root [4] of.Draw()
root [5] c.cd(1)
(class TVirtualPad*)0x242a510
root [6] of.Draw()
root [7] c.cd(3)
(class TVirtualPad*)0x242ac30
root [8] of.Draw()
root [9] c.cd(4)
(class TVirtualPad*)0x242afb0
root [10] of.Draw()
```



...and saved as an image

```
root [6] c.SaveAs("myFunction.png")
Info in <TCanvas::Print>: png file myFunction.png has
 been created
```

Functions and Histograms

- Define a function:

```
TF1 *myfunc=new TF1("myfunc","gaus",0,3);
Myfunc->SetParameters(10., 1.5, 0.5);
Myfunc->Draw();
```

- Generate histograms from functions:

```
(Myfunc->GetHistogram())->Draw();
```

- Generate histograms with random numbers from a function:

```
TH1F *hgaus=new TH1F("hgaus","histo from gaus", 50, 0,3);
hgaus->FillRandom("myfunc",10000);
hgaus->Draw();
```

- Write histogram to a rootfile:

```
TFile *myfile= new Tfile("hgaus.root","RECREATE");
hgaus->Write();
Myfile->Close();
```

Fitting Histograms

Let us try to fit the histogram created by the previous step:

Interactively:

- Open rootfile containing histogram:
  root –l hgaus.root
- Draw histogram
  hgaus->Draw()
- Right click on the histogram and select "Fit Panel"
- Check to ensure:
- "gaus" is selected in the Function->Predefined pop-menu
- "Chi-square" is selected in the Fit Settings->Method menu
- Click on "Fit" at the bottom

[Display fit parameters: Select Options->FitParameters]

A couple of words about Fitting

Fitting in a nut-shell:

A mathematical procedure to find parameter values of a TFn function, ***f***, describing *the best* your histogram or graph ***(x,y)***. The most commonly used criteria for an optimum fit is to *minimize* the χ^2-function:

$$\chi^2 = \sum \left(\frac{y_i - f(x_i)}{\sigma_i} \right)^2$$

Thumb rule for *Goodness of Fit*: **χ^2/NDF ~ 1**

…but there is *much much more* to fitting and statistical data analysis, which fills a course by itself…

Fitting contd:

Using user defined functions:

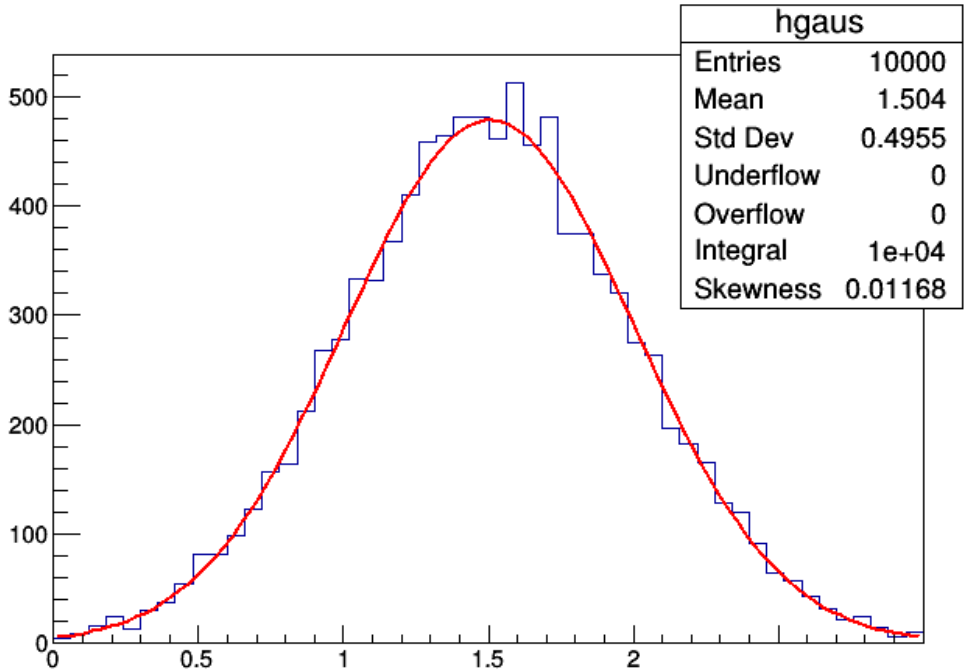
- Create a file called user_func.C with the following contents:

harinder@harinder-Latitude-E6430: ~/ROOT

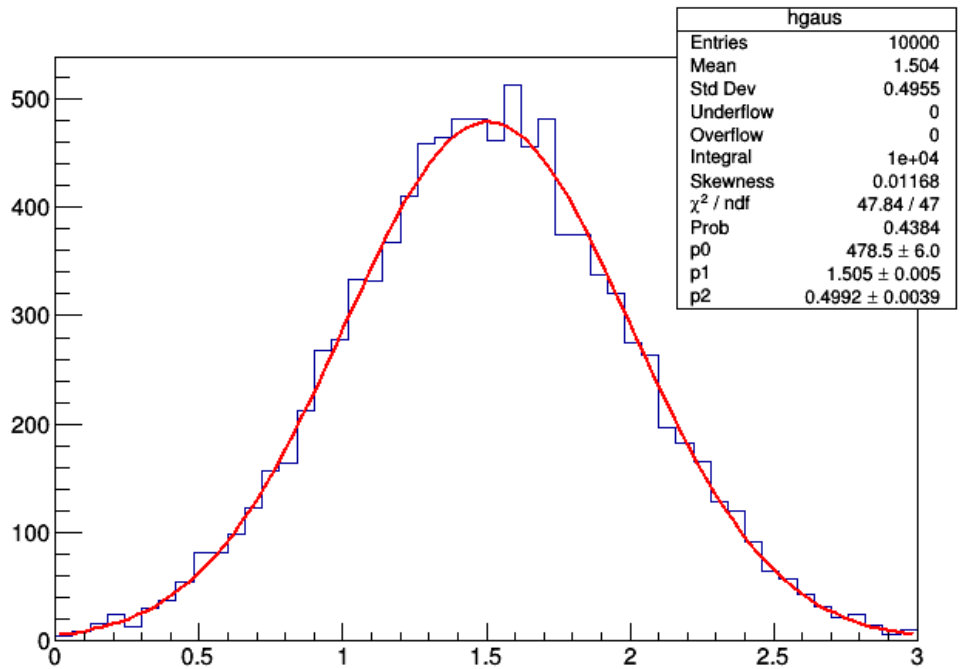
```
double user_func(double *x, double *par){
double arg=0;
if(par[2])arg=(x[0]-par[1])/par[2];
return par[0]*TMath::Exp(-0.5*arg*arg);
}
```

```
harinder@harinder-Latitude-E6430:~/ROOT$ ls
gaus.root  histo.C  user_func.C
harinder@harinder-Latitude-E6430:~/ROOT$ root -l gaus.root
root [0]
Attaching file gaus.root as _file0...
(class TFile *) 0x9136958
root [1] .L user_func.C
root [2] TF1 *f1=new TF1("f1", user_func, 0, 3, 3);
root [3] hgaus->Draw()
Info in <TCanvas::MakeDefCanvas>:  created default TCanvas with name c1
root [4] f1->SetParameters(10, hgaus->GetMean(), hgaus->GetRMS());
root [5] hgaus->Fit("f1");
 FCN=28.5714 FROM MIGRAD    STATUS=CONVERGED     172 CALLS          173 TOTAL
                     EDM=4.37095e-11    STRATEGY= 1      ERROR MATRIX ACCURATE
  EXT PARAMETER                                   STEP         FIRST
  NO.   NAME      VALUE            ERROR          SIZE      DERIVATIVE
   1  p0           7.97277e+02   9.78840e+00   2.11998e-02  -5.33649e-07
   2  p1           1.50390e+00   5.00504e-03   1.32879e-05  -1.59987e-03
   3  p2           4.99048e-01   3.55395e-03   7.69647e-06  -1.76223e-04
root [6]
```

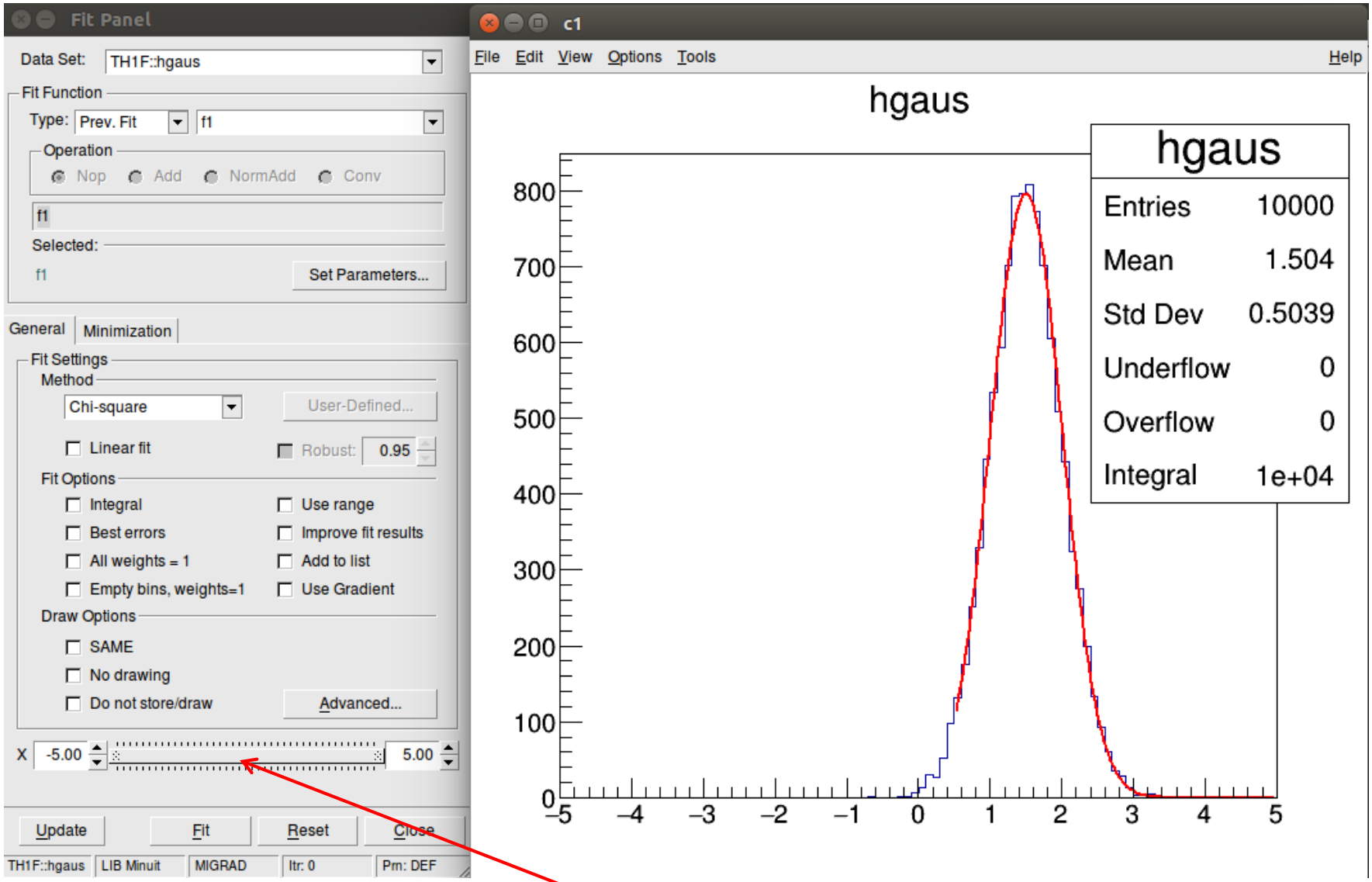
Last 3 parameters specify the number of parameters for the function



gStyle->SetOptStat(1111111)
Also can try :
gStyle->SetOptFit(1111)



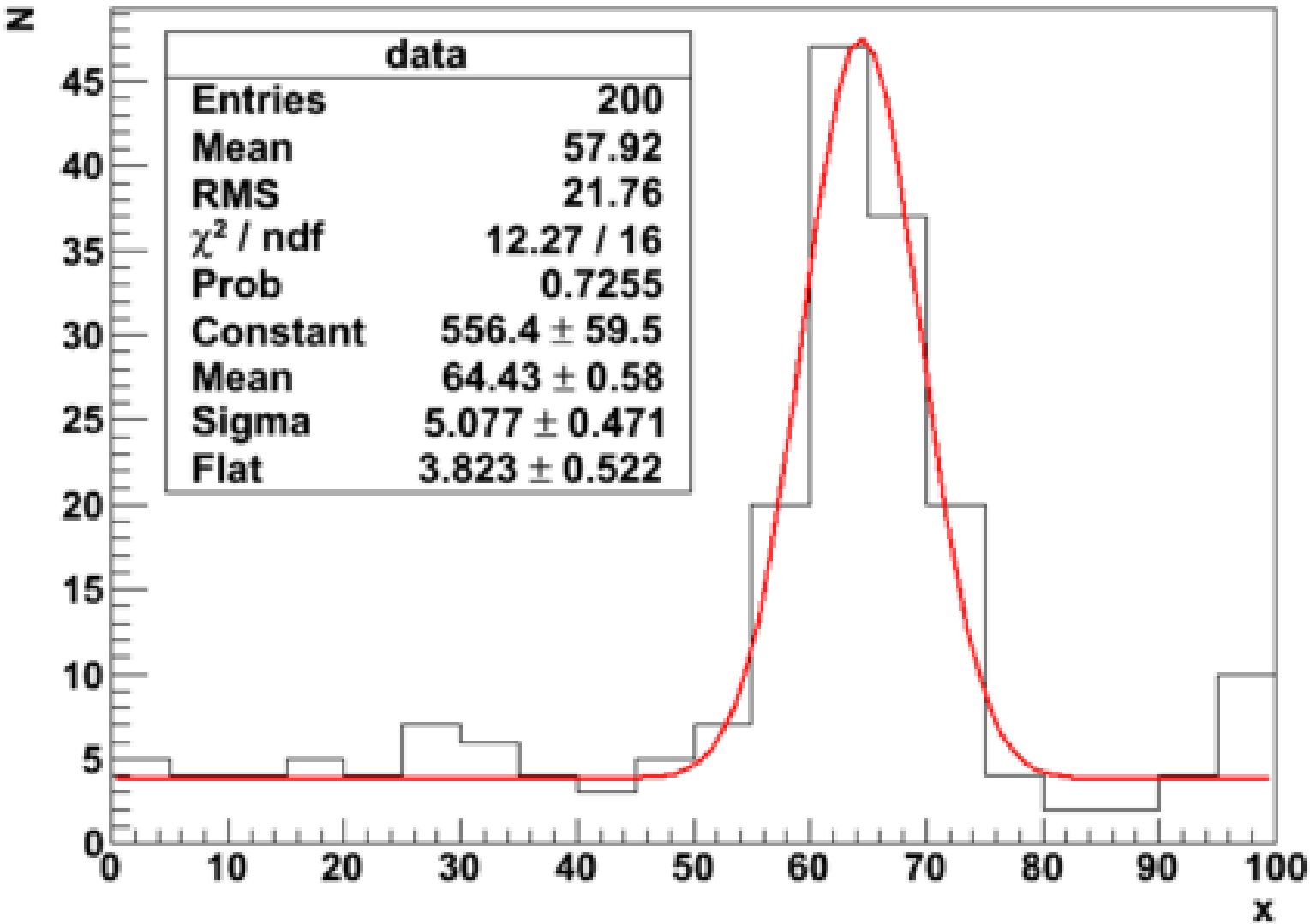
Fitting contd:



Move the slider to change fit Range

```
void fit_bgsig()
{
        // style setting
        gROOT->SetStyle("Plain");
        gStyle->SetOptFit(111111);
        gStyle->SetFrameBorderMode(0);
        gStyle->SetFillColor(0);
        gStyle->SetCanvasColor(0);
        gStyle->SetFrameFillColor(0);
        gStyle->SetCanvasBorderMode(0);

        // create a random number generator
        gRandom = new TRandom3();

        // create a histogram
        TH1D * hist = new TH1D("data", ";x;N", 20, 0.0, 100.0);

        // fill in signal
        for (int i = 0; i < 100; ++i)
                hist->Fill(gRandom->Gaus(65.0, 5.0));
        // fill background
        for (int i = 0; i < 100; ++i)
                hist->Fill(100.0 * gRandom->Uniform());

        // define a fit function = gauss + pol0
        TF1 * f1 = new TF1("myfunc", "[0] / sqrt(2.0 * TMath::Pi()) / [2] * exp(-(x-[1])*(x-[1])/2./[2]/[2]) + [3]", 0, 100);

        //set parameter start values (mandatory).
        f1->SetParNames("Constant","Mean","Sigma","Flat");
        f1->SetParameters(700.,hist->GetMean(),hist->GetRMS(),2.);
        f1->SetParLimits(0,  100.0, 700.0);
        f1->SetParLimits(1, 50.0,  90.0);
        f1->SetParLimits(2,  0.1,  10.0);
        f1->SetParLimits(3,  0.0,   10.0);
        f1->SetLineWidth(2);
        f1->SetLineColor(2);

        // create a canvas to draw the histogram
        TCanvas * c1= new TCanvas("c1", "fit sig+bg",5,5,800,600);
        // perform fit
        hist->Fit("myfunc");
        hist->Draw();
        hist->SaveAs("fit_bgsig.eps");

}
```

Mean=65, sigma=5.


data
Entries 200
Mean 57.92
RMS 21.76
χ² / ndf 12.27 / 16
Prob 0.7255
Constant 556.4 ± 59.5
Mean 64.43 ± 0.58
Sigma 5.077 ± 0.471
Flat 3.823 ± 0.522
N
x
0
5
10
15
20
25
30
35
40
45
0
10
20
30
40
50
60
70
80
90
100

Generating with standard PDF's

- Provided methods of TRandom$N$ objects:
  - `Exp(tau)`
  - `Integer(imax)`
  - `Gaus(mean, sigma)`
  - `Rndm()`
  - `RndmArray(n, x)`
  - `Uniform(x)`
  - `Uniform(x1, x2)`
  - `Landau(mpv, sigma)`
  - `Poisson(mean)`
  - `Binomial(ntot, prob)`

https://root.cern.ch/doc/master/classTRandom.html