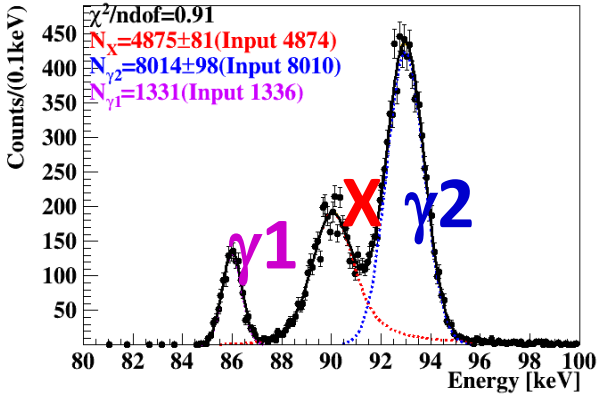
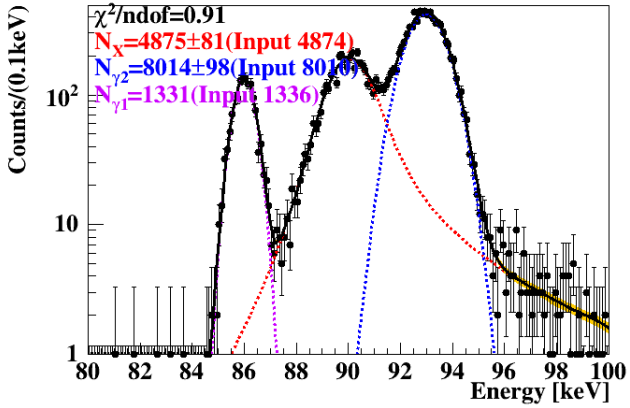


ROOT介绍



王思广

Email: siguang@pku.edu.cn

北京大学物理学院

The 2nd International Summer School on TeV Experimental Physics

Shandong University, Aug. 12, 2015

报告内容

- ROOT自由软件网页
- 利用RooFit进行重叠峰拟合
- ROOT 部分功能展示
- ROOT的安装
- 安装后的运行
- A ROOT Guide For Beginners
- 结束语

ROOT自由软件网页

http://root.cern.ch/drupal/

Screenshots

Get a taste of ROOT's capabilities by sampling some screenshots.

Download

Go ahead and download the latest build of ROOT.

Documentation

Get the inside scoop on how to fully utilize ROOT. Also, search the Reference Guide, the HowTo's and the user forums.

What's New

- July 14, 2015, 13:11 Patch release 6.04/02 - 2015-07-14
- June 24, 2015, 15:41 Patch release 6.02/12 - 2015-06-24
- June 23, 2015, 17:18 Patch release 5.34/32 - 2015-06-23

ROOT Workshop Registration Open!

workshop

Registration for the ROOT Users' Workshop, Sept 15-18 in Saas-Fee, Switzerland is now open at http://indico.cern.ch/event/root20. Please join us to celebrate ROOT's 20th birthday, and discuss what its future will be.

Read more

Patch release 6.04/02 - 2015-07-14

ROOT自由软件网页

ROOT 6:

- Pro, version 6.04/00
- Dev, version 6.03/04
- Old, version 6.02/12
- Old, version 6.00/00

目前为止太多的代码基于ROOT5，故我们学习ROOT5
点击进入

ROOT 5:

- Pro, version 5.34/32 (see also the release notes)
- Old, version 5.32/04 (see also the release notes and development notes)
- Old, version 5.30/06 (see also the release notes and development notes)
- Old, version 5.28/00h (see also the release notes and development notes)
- Old, version 5.26/00 (see also the release notes and development notes)
- Old, version 5.24/00 (see also the release notes and development notes)

ROOT自由软件网页

Production Version 5.34

production version

Availability

ROOT is available in binary and source form. The binaries are available for most supported platforms. The source is available as a tarball or from git and can easily be compiled on any supported platform/compiler combination.

For what is new in this version see the release notes.

Source

- ROOT 5.34.32 complete source tree for all systems (72 MB).
  After unpacking read "Installing ROOT From Source" or the file README/INSTALL.

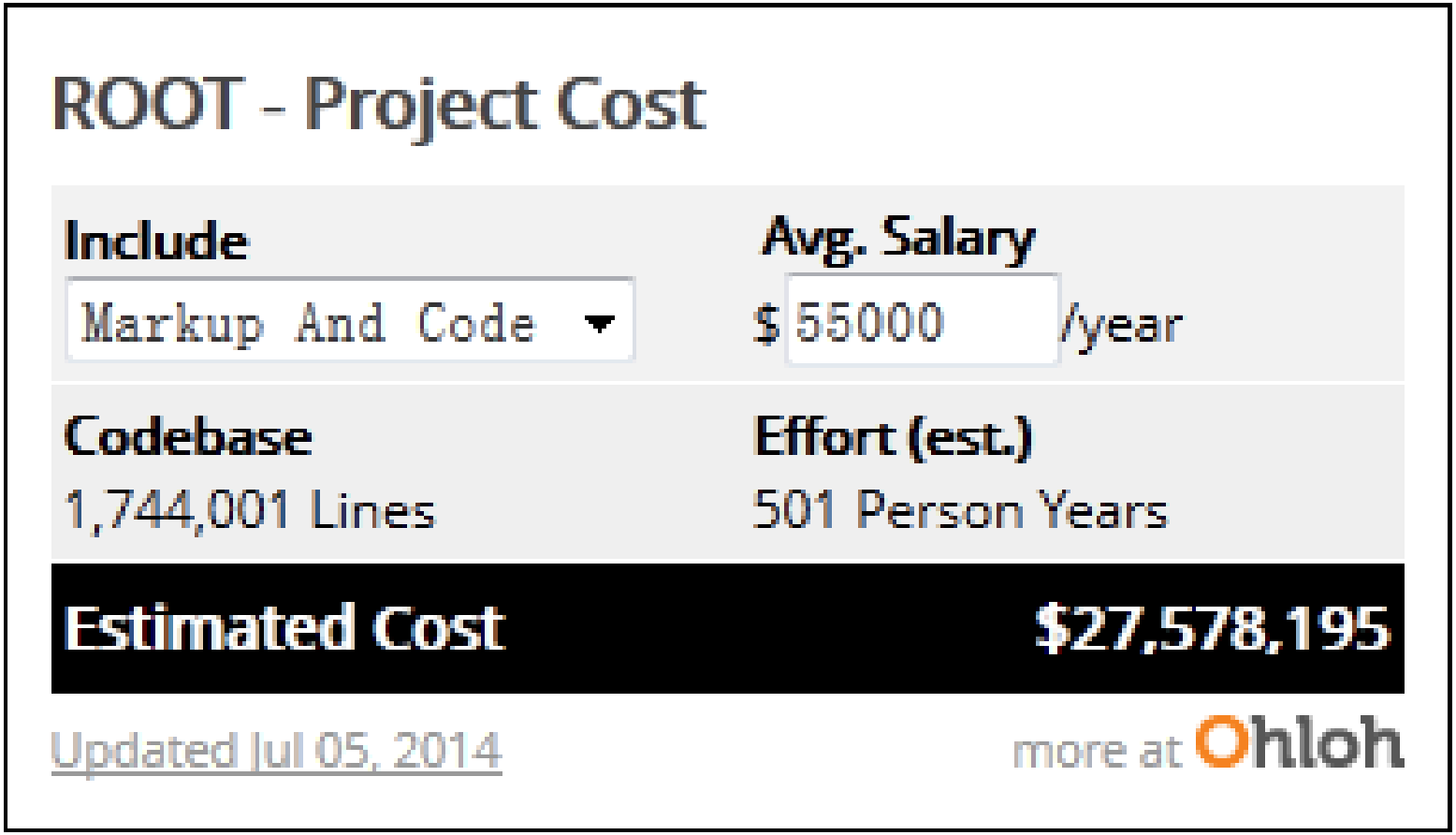
Documentation

- ROOT 5.34.32 classes html documentation compressed tar file (885 MB).

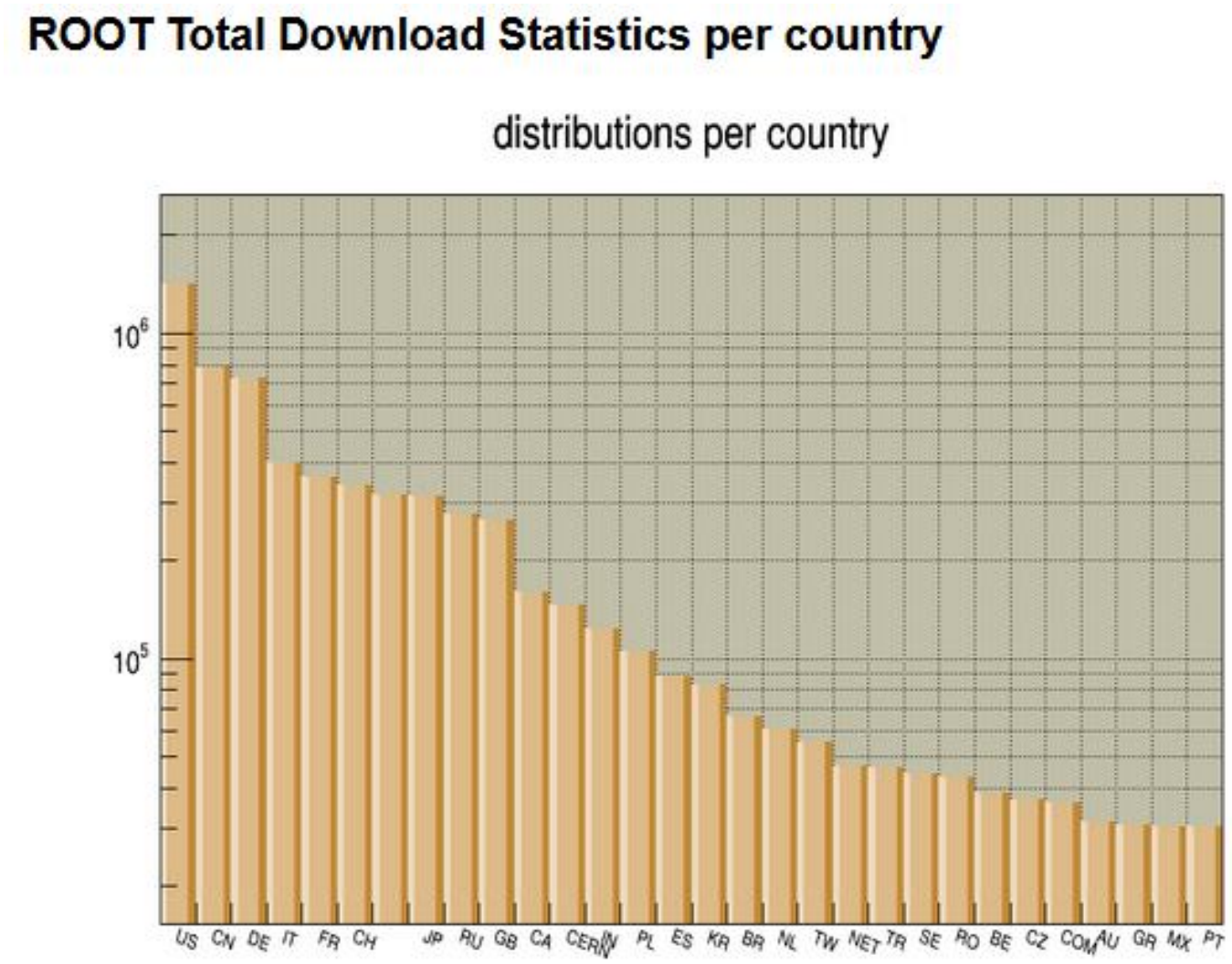
https://root.cern.ch/drupal/content/production-version-534

Project Statistics

http://root.cern.ch/drupal/content/project-statistics



各国下载量



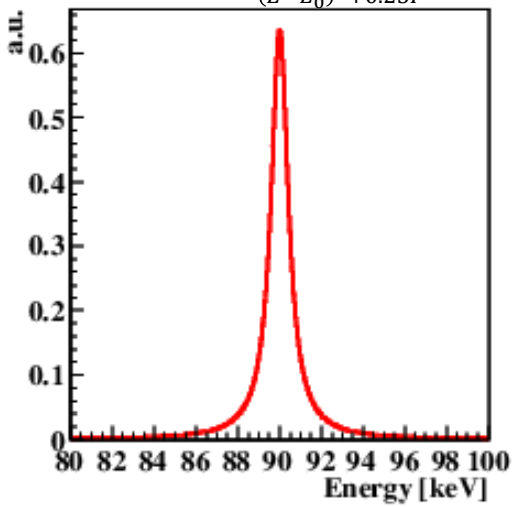
https://root.cern.ch/drupal/content/download-statistics

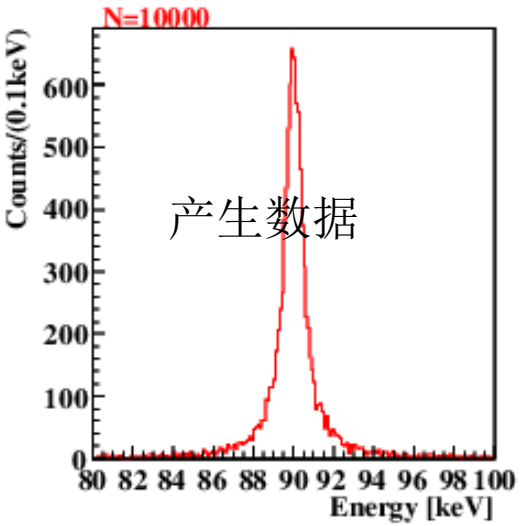
利用RooFit进行重叠峰拟合

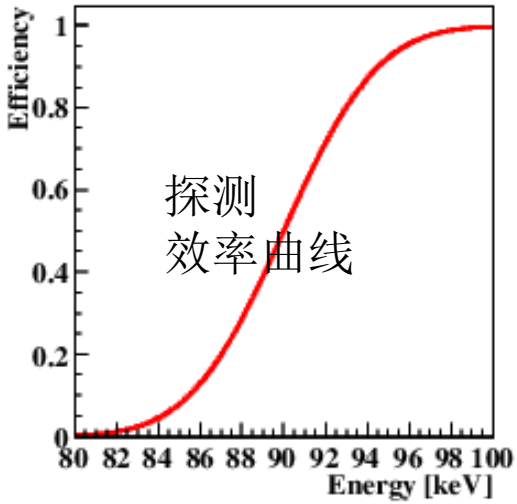
Siguang@pku.edu.cn

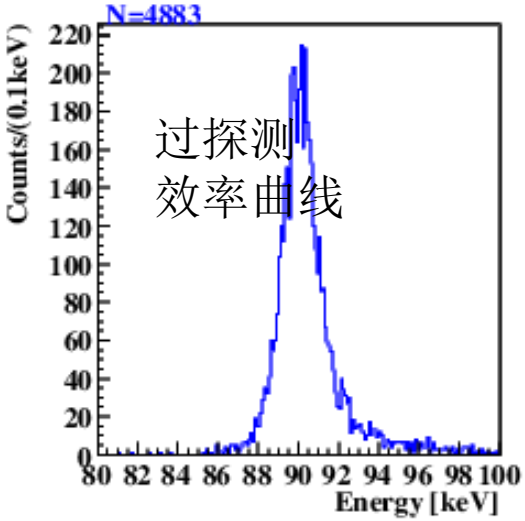
X射线能峰形状

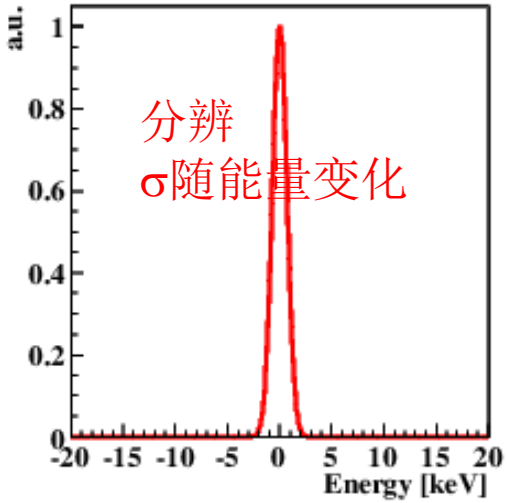
本征形状: $f(E) \propto \frac{1}{(E-E_0)^2+0.25\Gamma^2}$

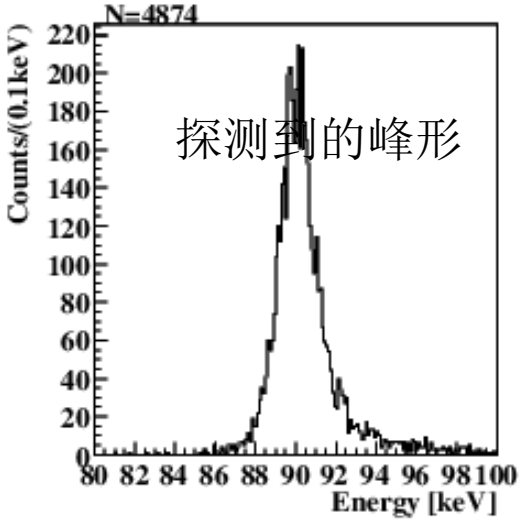






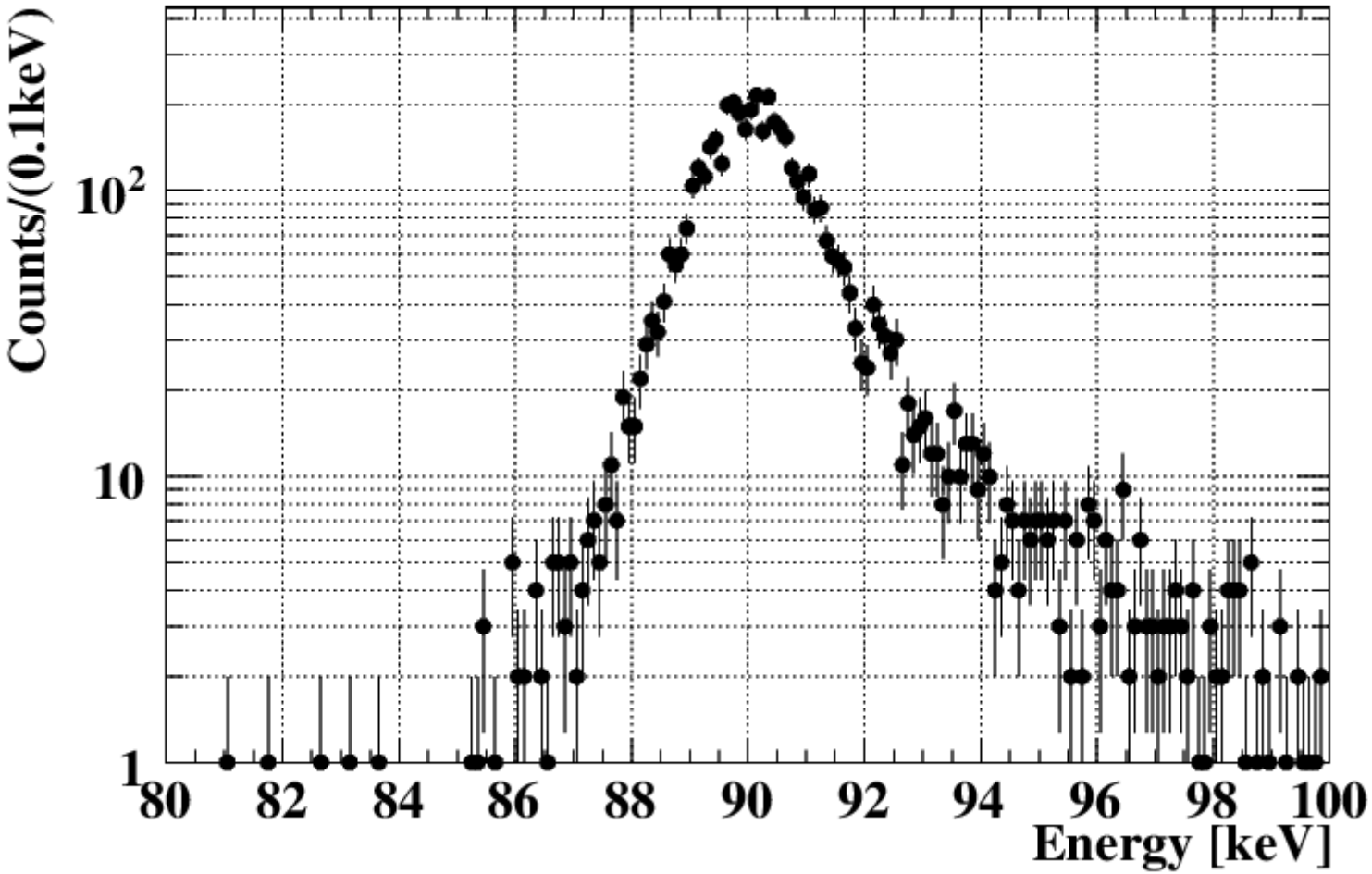






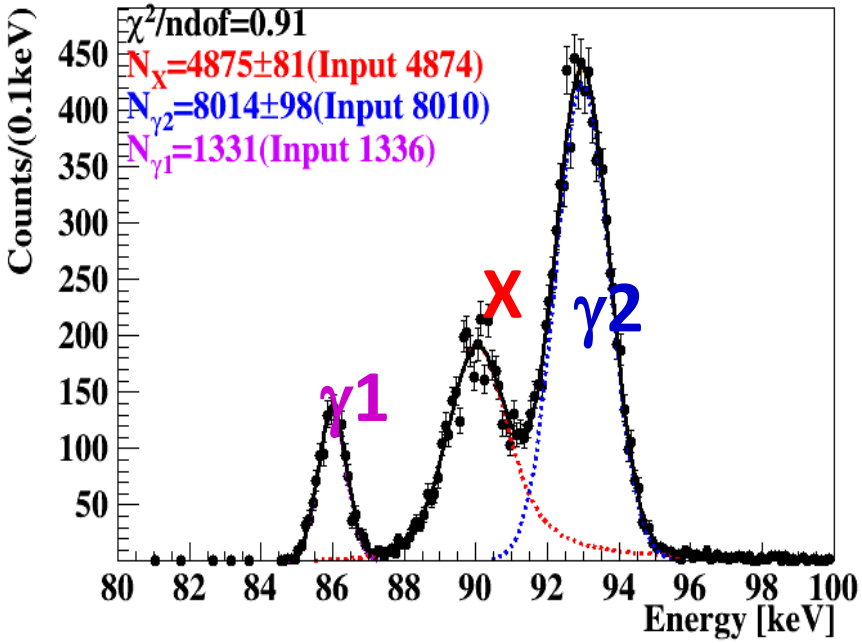
Siguang@pku.edu.cn

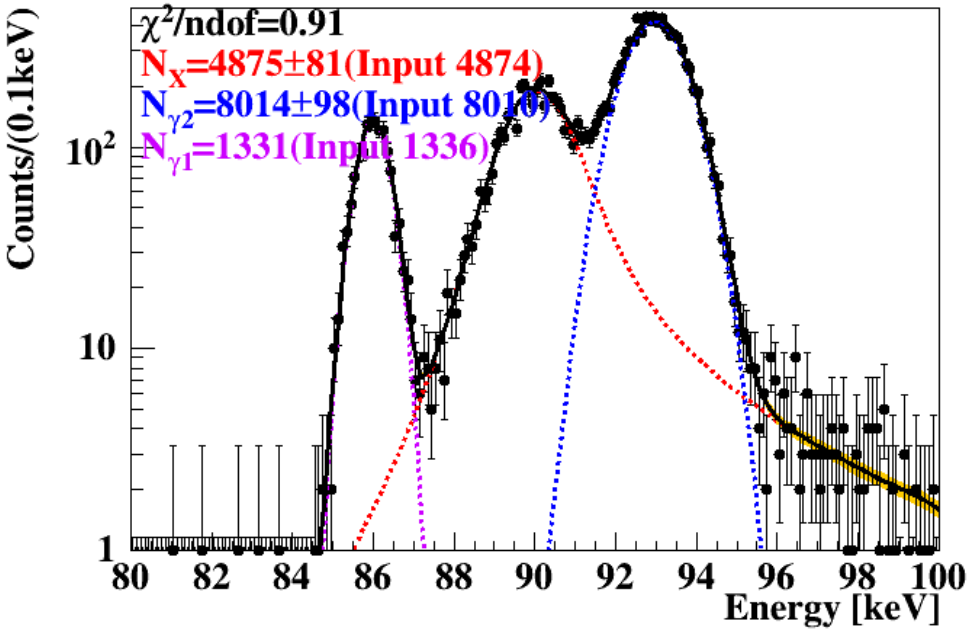
X射线能峰形状(产生)



Siguang@pku.edu.cn

产生并拟合的谱





Siguang@pku.edu.cn

拟合函数

$$N_1 \cdot G(\mathrm{x}, E_1, \sigma(E_1)) \cdot \varepsilon(\mathrm{E}_1) +$$
$$N_2 \cdot \left\{ \left[BW(x, E_2, \Gamma) \cdot \varepsilon(\mathrm{x}) \right] \otimes G(\mathrm{x}, 0, \sigma(E_2)) \right\} +$$
$$N_3 \cdot G(\mathrm{x}, E_3, \sigma(E_3)) \cdot \varepsilon(\mathrm{E}_3)$$

说明：

1） ⊗----卷积功能。利用FFTW （"**Fastest Fourier Transform in the West.**" ）

2）分辨是射线能量的函数；对于X射线这里近似用峰位分辨.

3）效率是能量的函数

实现代码 （读入能谱）

```
//
//Set plot style----
SetSgStyle();

//
//Read the spectrum.....
TH1D *hSpec = 0;
TH1D *hL = 0;
TH1D *hXray = 0;
TH1D *hR = 0;

Double_t fMeanG1,fMeanG2,fMeanX;
TFile fin("spec.root");
hSpec = (TH1D *)fin.Get("h3Peaks");
hSpec->SetDirectory(0);
hL = (TH1D *)fin.Get("hL");
hL->SetDirectory(0);
hR = (TH1D *)fin.Get("hR");
hR->SetDirectory(0);
hXray = (TH1D *)fin.Get("hXray");
hXray->SetDirectory(0);
fMeanG1 = hL -> GetMean();
fMeanG2 = hR -> GetMean();
fMeanX = hXray -> GetMean();
fin.Close();
Double_t xmin = hSpec->GetXaxis()->GetXmin();
Double_t xmax = hSpec->GetXaxis()->GetXmax();
```

实现代码 （定义X-射线峰）

核心代码

```
RooRealVar x("x","x",xmin,xmax);

//X-ray peak
RooRealVar MeanX("MeanX","Mean of X-ray Peak",fMeanX,fMeanX-5,fMeanX+5);
RooRealVar WidthX("WidthX","Width of X-Ray Peak",1.);//we know the width
RooBreitWigner peakX0("peakX0","X-ray Peak Before Detected",x,MeanX,WidthX);

//Efficiency
RooRealVar a("a","1st coefficency of eff", 90.0,80,100);
RooRealVar b("b","2nd coefficency of eff", 5,1.0,8.0);
RooFormulaVar effFun("effFun","0.5*(TMath::Erf((x-a)/b)+1)",RooArgList(a,b,x)) ;

//Resolution
RooRealVar mg("mg","mg",0) ;
RooRealVar c1("c1","1st coefficency of sg",0.1,0,4);
RooRealVar c2("c2","2nd coefficency of sg",0.05,0,1.);
RooFormulaVar sg("sg",Form("(c1+c2*(MeanX-%g))",xmin),RooArgList(c1,c2,MeanX)) ;
RooGaussian R("R","resolution",x,mg,sg) ;

// Multiply pdf(x) with efficiency in x
RooEffProd peakX0eff("peakX0eff","peakX0 with efficiency",peakX0,effFun) ;

// Construct peakXeff (x) R
// Set #bins to be used for FFT sampling to 10000
x.setBins(10000,"cache") ;
RooFFTConvPdf peakX("peakX","(peakX0*Eff) (X) gauss",x,peakX0eff,R) ;
```


Siguang@pku.edu.cn

实现代码（定义γ_1、γ_2；3峰拟合）

核心代码

```
//1st gamma peak
RooRealVar Mean1("Mean1","Mean of 1st gamma Peak",fMeanG1,fMeanG1-5,fMeanG1+5);
RooFormulaVar Sigma1("Sigma1",Form("(c1+c2*(Mean1-%g))",xmin),RooArgList(c1,c2,Mean1)) ;
RooGaussian peak1("peak1","1st Peak)",x,Mean1,Sigma1);

//
//2nd gamma peak
RooRealVar Mean2("Mean2","Mean of 2nd gamma Peak",fMeanG2,fMeanG2-5,fMeanG2+5);
RooFormulaVar Sigma2("Sigma2",Form("(c1+c2*(Mean2-%g))",xmin),RooArgList(c1,c2,Mean2)) ;
RooGaussian peak2("peak2","2nd Peak)",x,Mean2,Sigma2);

Double_t fN1,fNX,fN2;
fN1 = hSpec->GetEntries()*.33;  fNX = fN1;  fN2 = fN1;
RooRealVar NX("NX","Count under 2nd Peak",fNX,0,3*fNX);
RooRealVar N2("N2","Count under 3rd Peak",fN2,0,3*fN2);
//Using N1 and N2 have same branch ratio and are from same isotope: N1=Eff(x1)/Eff(x2)*N2
RooFormulaVar N1("N1","(TMath::Erf((Mean1-a)/b)+1)/(TMath::Erf((Mean2-a)/b)+1)*N2",RooArgList(a,b,Mean1,Mean2,N2));

RooAddPdf model("model","model",RooArgList(peak1,peakX,peak2),RooArgList(N1,NX,N2)) ;

RooDataHist dh("dh","dh",x,Import(*hSpec)) ;

RooFitResult *frlt = model.fitTo(dh,Save());
```

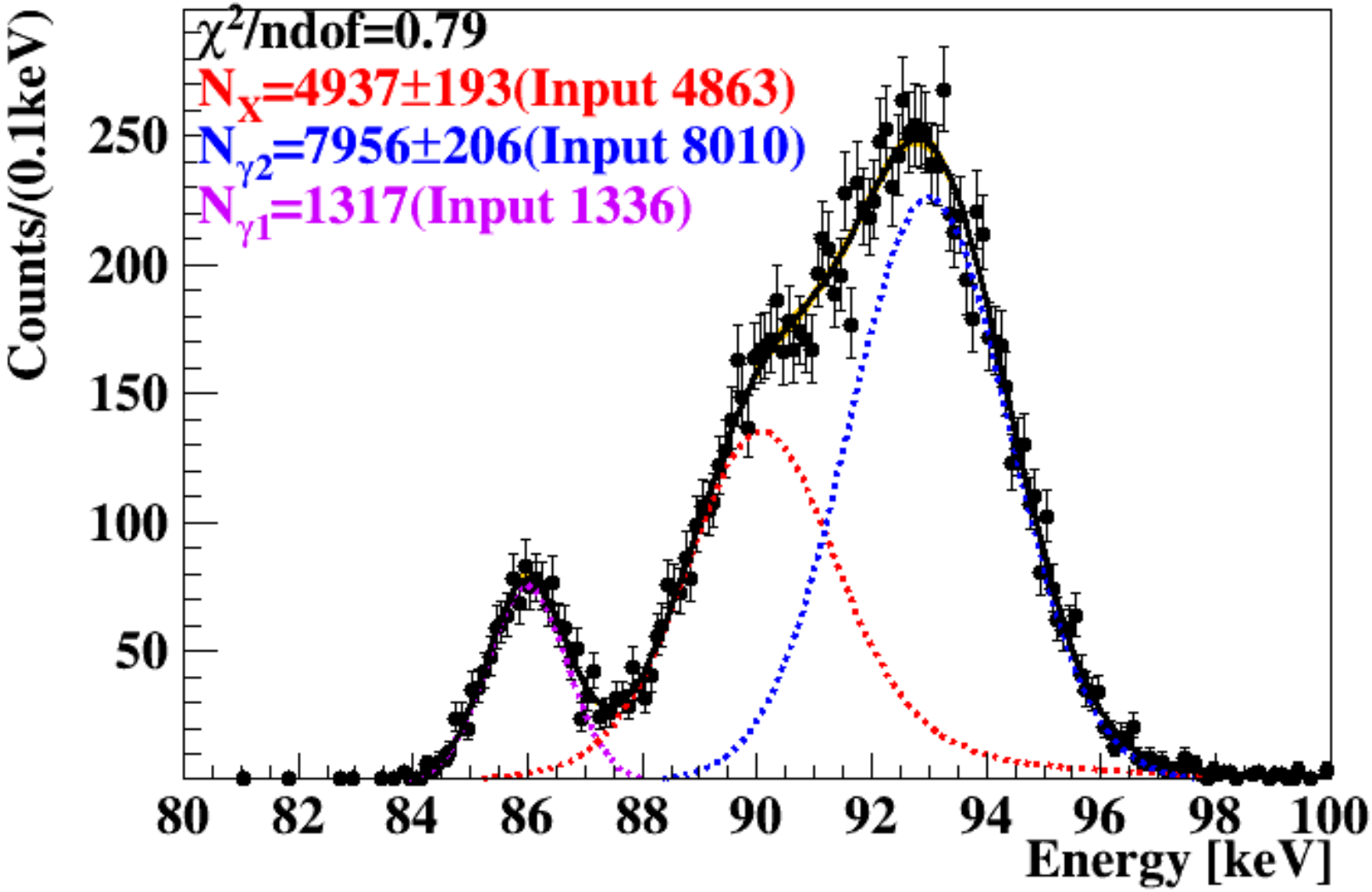
也可以用多CPU同时拟合(unbin条件下)：

RooFitResult *frlt = model.fitTo(dh,Save(),**NumCPU(2)**);

实现代码（作图）

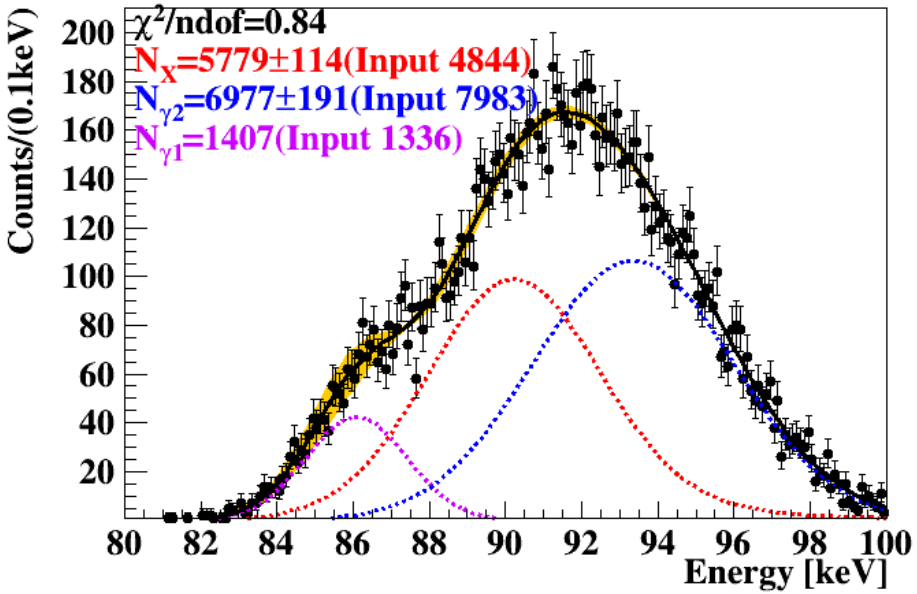
```
RooPlot* frame = x.frame() ;
dh.plotOn(frame) ;
model.plotOn(frame,VisualizeError(*frlt,1),FillColor(kOrange));
dh.plotOn(frame) ;
model.plotOn(frame, Components(peak1),LineStyle(kDashed),LineColor(kViolet));
model.plotOn(frame, Components(peakX),LineStyle(kDashed),LineColor(kRed));
model.plotOn(frame, Components(peak2),LineStyle(kDashed),LineColor(kBlue));
model.plotOn(frame,LineStyle(kSolid),LineColor(kBlack));

TCanvas *cPlot = new TCanvas("cPlot","");
frame->SetMinimum(1);
frame->Draw();

Int_t nParsToFit = (frlt->floatParsFinal()).getSize();
Double_t chi2_red = frame->chiSquare(nParsToFit);//reduced chi-squared = chi2/ndof
txt(0.16,0.9,Form("#chi^{2}/ndof=%.2f",chi2_red));
txt(0.16,0.84,Form("N_{X}=%.0f#pm%.0f(Input %.0f)",NX.getVal(),NX.getError(),hXray->GetEntries()),kRed);
txt(0.16,0.78,Form("N_{#gamma2}=%.0f#pm%.0f(Input %.0f)",N2.getVal(),N2.getError(),hR->GetEntries()),kBlue);
txt(0.16,0.72,Form("N_{#gamma1}=%.0f(Input %.0f)",N1.getVal(),hL->GetEntries()),kViolet);

frame->SetTitle(Form(";%s;%s",hSpec->GetXaxis()->GetTitle(),hSpec->GetYaxis()->GetTitle()));
frlt->Print();
cPlot->Modified();
cPlot->Update();
cPlot->cd();
cPlot->SaveAs("cPlot.pdf");
```

较差分辨下能谱的拟合

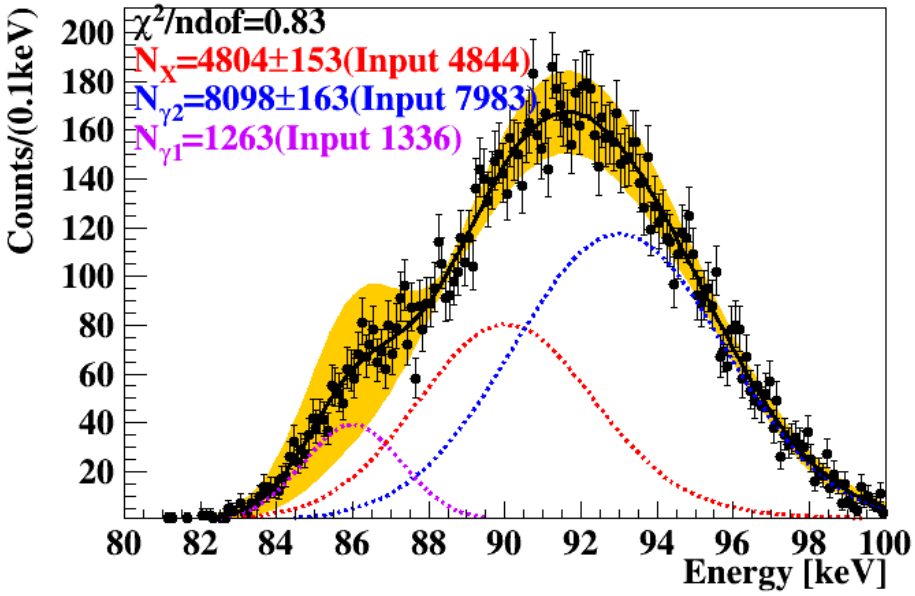


Siguang@pku.edu.cn

更差分辨下能谱的拟合



自由浮动峰位



峰位固定

Siguang@pku.edu.cn

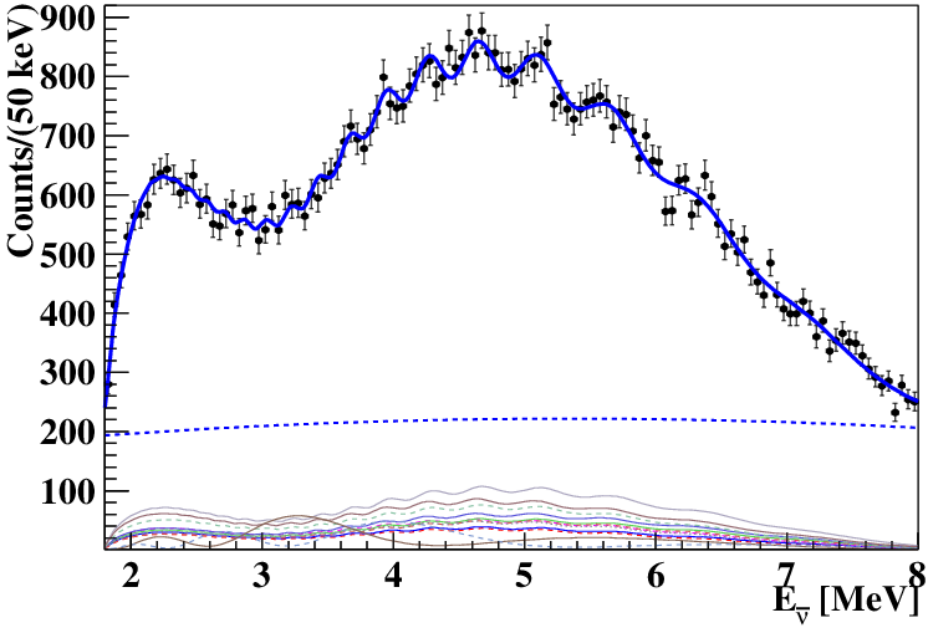
周期： 10^{6000}

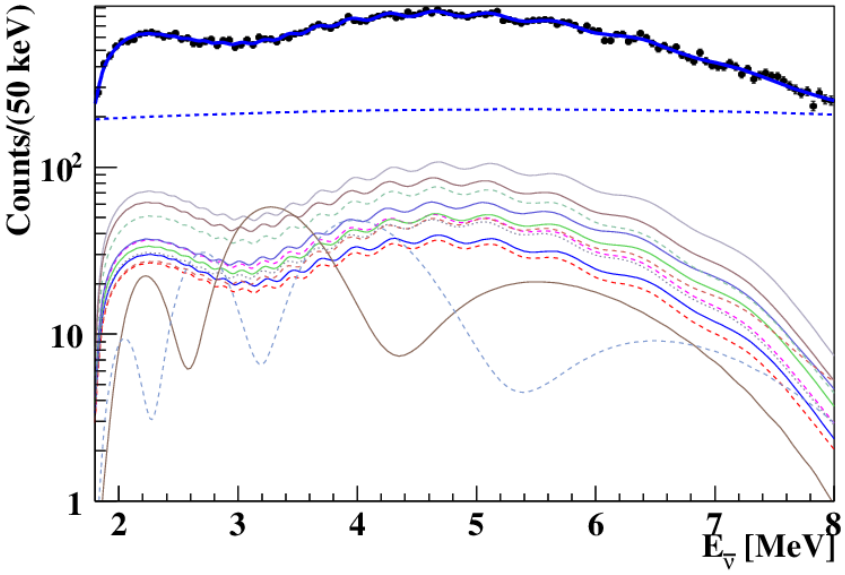
```
TRandom3   r;
r.SetSeed(0);
Double_t val = r.Rndm();
```

Machine independent random number generator.
Produces uniformly-distributed floating points in (0,1)

分辨随能量变化拟合例子

分辨随能量的变化: $R = \frac{3\%}{\sqrt{E_{vis}}}$



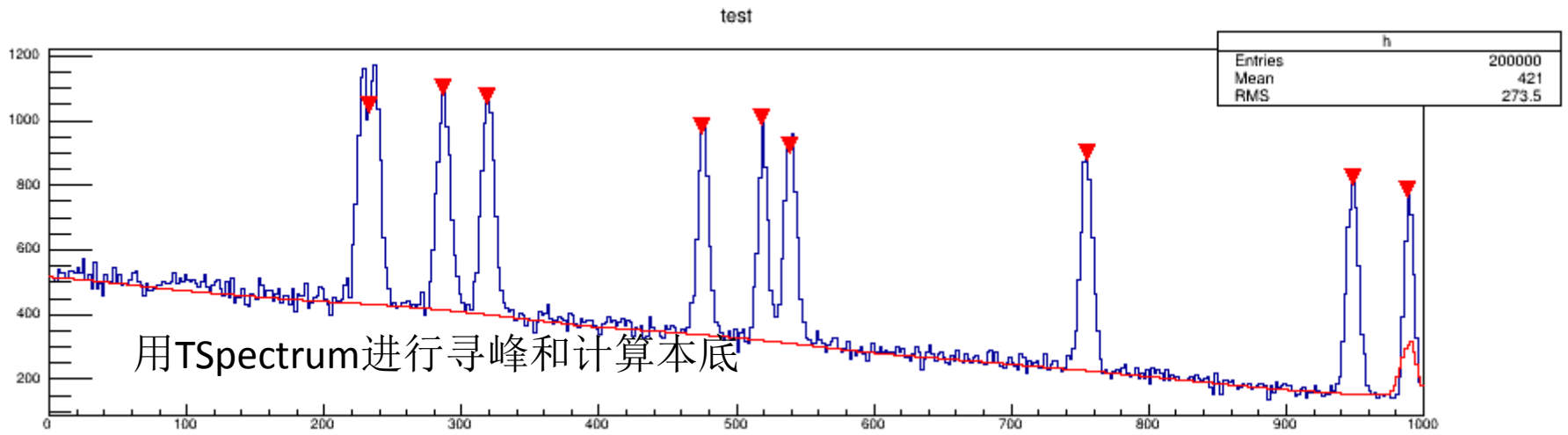


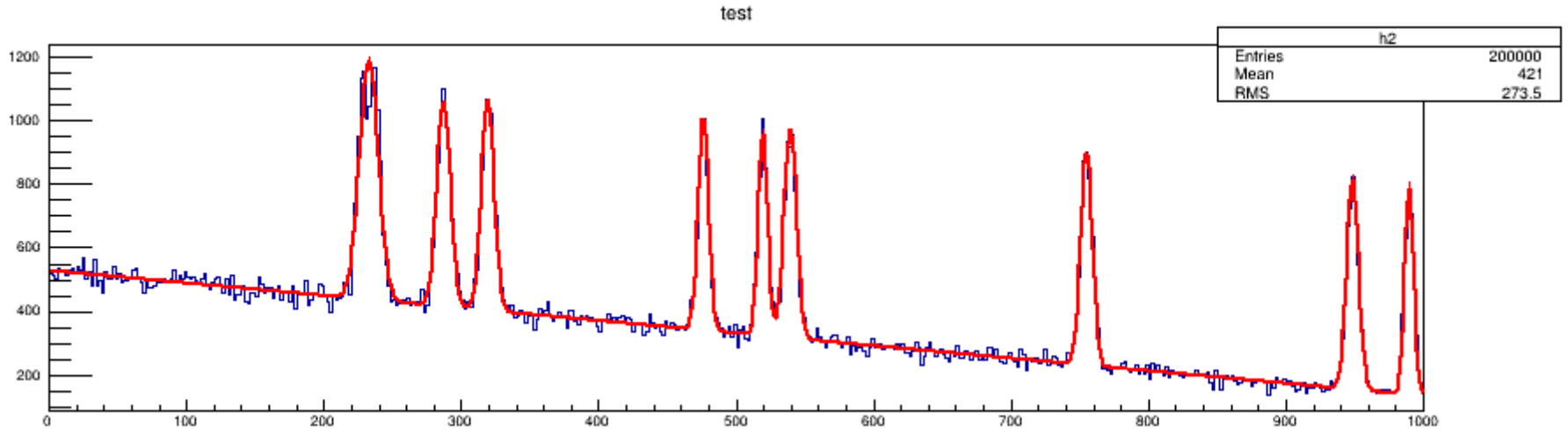
55个自由参数的拟合;
需要根据卷积的定于自己写程序

Siguang@pku.edu.cn

ROOT 部分功能展示

ROOT自带的能谱分析类TSpectrum





$ROOTSYS/tutorials/spectrum $ root peaks.C

TSpectrum类基本功能之本底估算

const char * Background(float* spectrum, Int_t ssize, Int_t numberIterations, Int_t direction, Int_t filterOrder, bool smoothing, Int_t smoothWindow, bool compton)

Parameters:

spectrum: pointer to the vector of source spectrum

ssize: length of the spectrum vector

numberIterations: maximal width of clipping window,

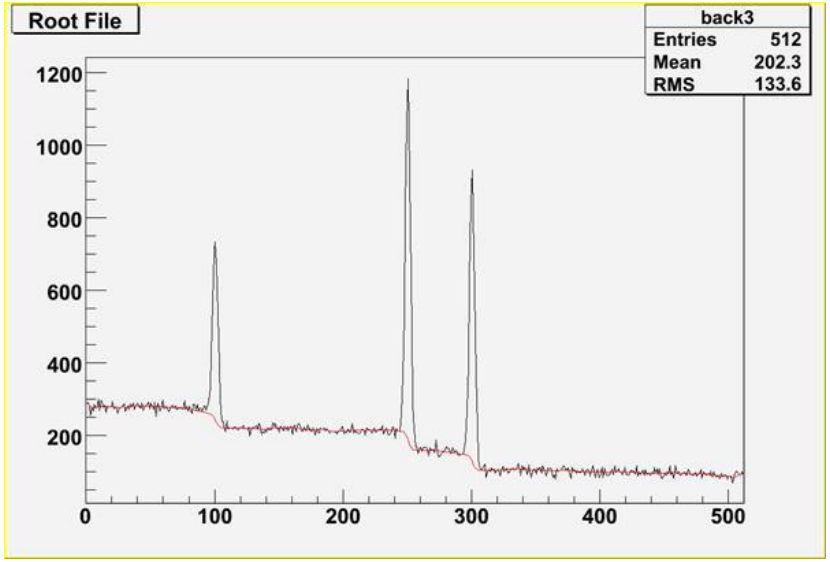
direction: direction of change of clipping window. Possible values: kBackIncreasingWindow, kBackDecreasingWindow

filterOrder: order of clipping filter. Possible values: kBackOrder2, kBackOrder4, kBackOrder6, kBackOrder8

smoothing: logical variable whether the smoothing operation in the estimation of background will be included. Possible values: kFALSE, kTRUE

smoothWindow: width of smoothing window. Possible values: kBackSmoothing3, kBackSmoothing5, kBackSmoothing7, kBackSmoothing9, kBackSmoothing11, kBackSmoothing13, kBackSmoothing15.

compton: logical variable whether the estimation of Compton edge will be included. Possible values: kFALSE, kTRUE.



http://root.cern.ch/root/html534/TSpectrum.html#TSpectrum:Background

本底估算参考代码

```
// Example to illustrate the background estimator (class TSpectrum) including
// Compton edges. To execute this example, do:
// root > .x Background_compton.C

void Background_compton() {
   Int_t i;
   Double_t nbins = 512;
   Double_t xmin  = 0;
   Double_t xmax  = (Double_t)nbins;
   Float_t * source = new float[nbins];
   TH1F *h = new TH1F("h","",nbins,xmin,xmax);
   TH1F *d1 = new TH1F("d1","",nbins,xmin,xmax);
   TFile *f = new TFile("spectra\\TSpectrum.root");
   h=(TH1F*) f->Get("back3;1");
   TCanvas *background = gROOT->GetListOfCanvases()->FindObject("background");
   if (!background) background = new TCanvas("background",
   "Estimation of background with Compton edges under peaks",10,10,1000,700);
   h->Draw("L");
   TSpectrum *s = new TSpectrum();
   for (i = 0; i < nbins; i++) source[i]=h->GetBinContent(i + 1);
   s->Background(source,nbins,10,kBackDecreasingWindow,kBackOrder8,kTRUE,
   kBackSmoothing5,kTRUE);
   for (i = 0; i < nbins; i++) d1->SetBinContent(i + 1,source[i]);
   d1->SetLineColor(kRed);
   d1->Draw("SAME L");
}
```

TSpectrum类基本功能之能谱光滑

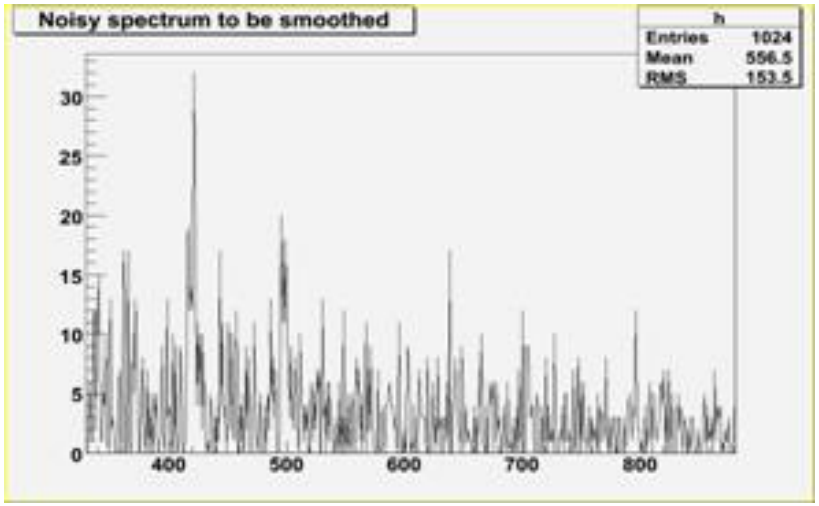
const char* SmoothMarkov(float* source, Int_t ssize, Int_t averWindow)

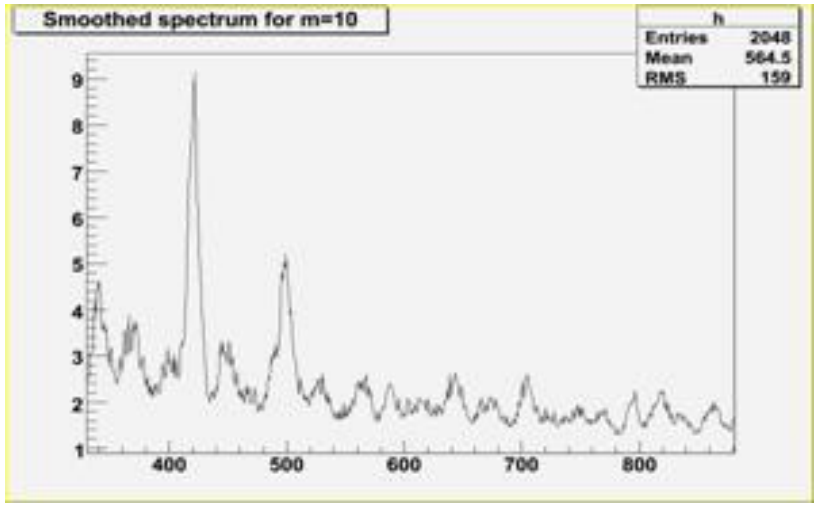
Parameters:

source: pointer to the array of source spectrum

ssize: length of source array

averWindow: width of averaging smoothing window



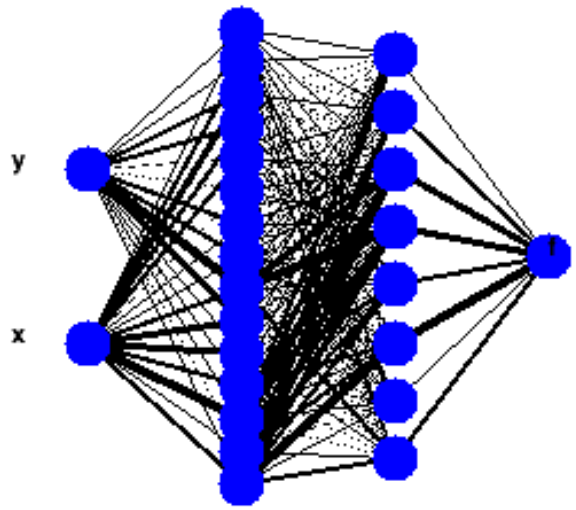


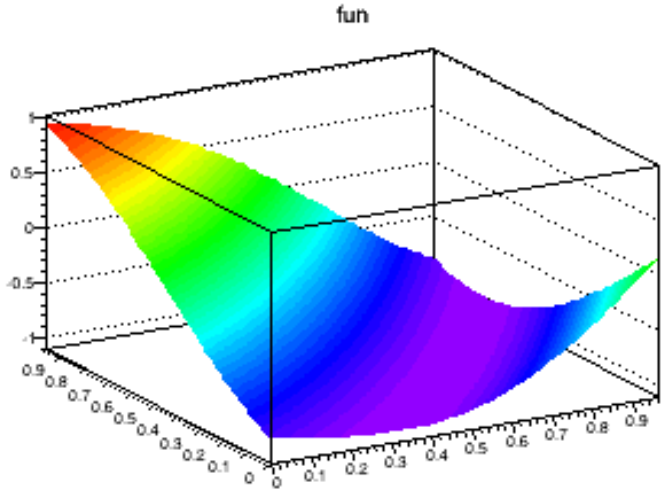
Siguang@pku.edu.cn

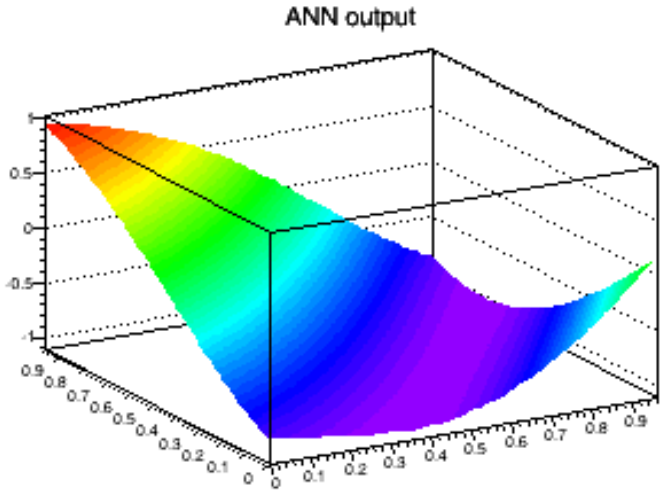
能谱光滑参考代码

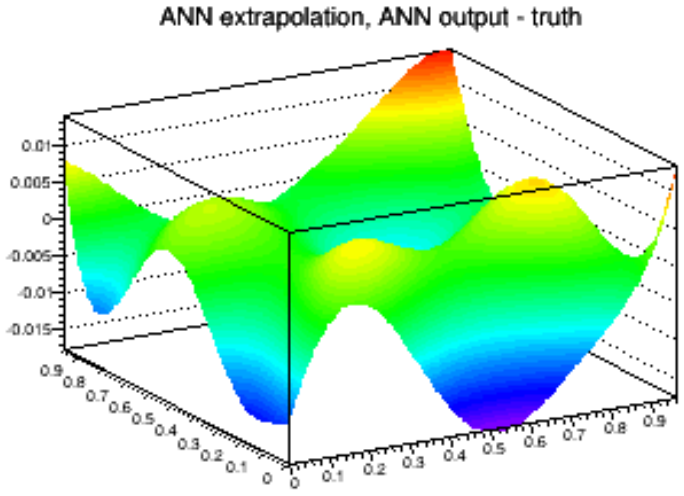
```
// Example to illustrate smoothing using Markov algorithm (class TSpectrum).
// To execute this example, do
// root > .x Smoothing.C
void Smoothing() {
   Int_t i;
   Double_t nbins = 1024;
   Double_t xmin  = 0;
   Double_t xmax  = (Double_t)nbins;
   Float_t * source = new float[nbins];
   TH1F *h = new TH1F("h","Smoothed spectrum for m=3",nbins,xmin,xmax);
   TFile *f = new TFile("spectra\\TSpectrum.root");
   h=(TH1F*) f->Get("smooth1;1");
   for (i = 0; i < nbins; i++) source[i]=h->GetBinContent(i + 1);
   TCanvas *Smooth1 = gROOT->GetListOfCanvases()->FindObject("Smooth1");
   if (!Smooth1) Smooth1 = new TCanvas("Smooth1","Smooth1",10,10,1000,700);
   TSpectrum *s = new TSpectrum();
   s->SmoothMarkov(source,1024,3);  //3, 7, 10
   for (i = 0; i < nbins; i++) h->SetBinContent(i + 1,source[i]);
   h->SetAxisRange(330,880);
   h->Draw("L");
}
```

利用TMultiLayerPerceptron内插曲面









Siguang@pku.edu.cn

利用TMultiLayerPerceptron内插曲面

```
Double_t theUnknownFunction(Double_t x, Double_t y) {
   return sin((1.7+x)*(x-0.3)-2.3*(y+0.7));
}

void mlpRegression() {
   // create a tree with train and test data.
   // we have two input parameters x and y,
   // and one output value f(x,y)
   TNtuple* t=new TNtuple("tree","tree","x:y:f");
   TRandom r;
   Double_t Wd = 1.0;
   for (Int_t i=0; i<10000; i++) {
      Float_t x=r.Rndm()*Wd;
      Float_t y=r.Rndm()*Wd;

      // fill it with x, y, and f(x,y) - usually this function
      // is not known, and the value of f given an x and a y comes
      // e.g. from measurements
      t->Fill(x,y,theUnknownFunction(x,y));
   }

   // create ANN
   TMultiLayerPerceptron* mlp=new TMultiLayerPerceptron("x,y:15:8:f",t,
      "Entry$%2==1","(Entry$%2)==0");
   mlp->Train(150,"text,graph,update=10");
   mlp->Export("testRlt","C++");
```

剩下的是使用和画图。。。

利用TMultiLayerPerceptron内插曲面

```
// draw difference of ANN's output for (x,y) vs f(x,y) assuming
// the ANN can extrapolate
Int_t N = 30;
const Int_t NN = N*N;
Double_t *vx= new Double_t[NN];
Double_t *vy= new Double_t[NN];
Double_t *delta= new Double_t[NN];
Double_t *vfOrg= new Double_t[NN];
Double_t *vfmlp= new Double_t[NN];
Double_t v[2];
Int_t idx=0;
for (Int_t ix=0; ix<N; ix++) {
   v[0]=Wd*ix/N;
   for (Int_t iy=0; iy<N; iy++) {
      v[1]=Wd*iy/N;
      vx[idx]=v[0];
      vy[idx]=v[1];
      vfOrg[idx] = theUnknownFunction(v[0],v[1]);
      vfmlp[idx] = mlp->Evaluate(0, v);
      delta[idx] = vfmlp[idx]  - vfOrg[idx];
      idx++;
   }
}
```

利用TMultiLayerPerceptron内插曲面

```
   TCanvas *c1 = new TCanvas("c1","");
   c1->Divide(2,2);
   c1->cd(1);
   mlp->Draw();
   c1->cd(2);
   TGraph2D* g2Org=new TGraph2D("g2Org",
                                "fun",
                                idx, vx, vy, vfOrg);
   g2Org->Draw("TRI2");
   c1->cd(3);
   TGraph2D* g2mlp=new TGraph2D("g2mlp",
                                "ANN output",
                                idx, vx, vy, vfmlp);
   g2mlp->Draw("TRI2");
   c1->cd(4);
   TGraph2D* g2Extrapolate=new TGraph2D("ANN extrapolation",
                                "ANN extrapolation, ANN output - truth",
                                idx, vx, vy, delta);
   g2Extrapolate->Draw("TRI2");

   delete[] vx;
   delete[] vy;
   delete[] delta;
   delete[] vfOrg;
   delete[] vfmlp;
}
```

注：在$ROOTSYS/tutorials/mlp$ root mlpRegression.C 的基础上修改，
原程序x，y测试区间[-1 ,2)与函数学习区间(0,1)有差异

利用TMultiLayerPerceptron内插曲面

可输出C++类作为结果提供使用

testRlt.h

```
class testRlt {
public:
   testRlt() {}
   ~testRlt() {}
   double Value(int index,double in0,double in1);
   double Value(int index, double* input);
private:
   double input0;
   double input1;
   double neuron0x2cd6450();
   double neuron0x2cd6790();
   double input0x2ce9850();
   double neuron0x2ce9850();
   double input0x2ce9b80();
   double neuron0x2ce9b80();
   double input0x2ce9f40();
   double neuron0x2ce9f40();
   double input0x2cea300();
   double neuron0x2cea300();
   double input0x2cea6c0();
   double neuron0x2cea6c0();
   double input0x2ceaa80();
   double neuron0x2ceaa80();
   double input0x2ceae40();
   double neuron0x2ceae40();
   double input0x2ceb200();
   double neuron0x2ceb200();
```

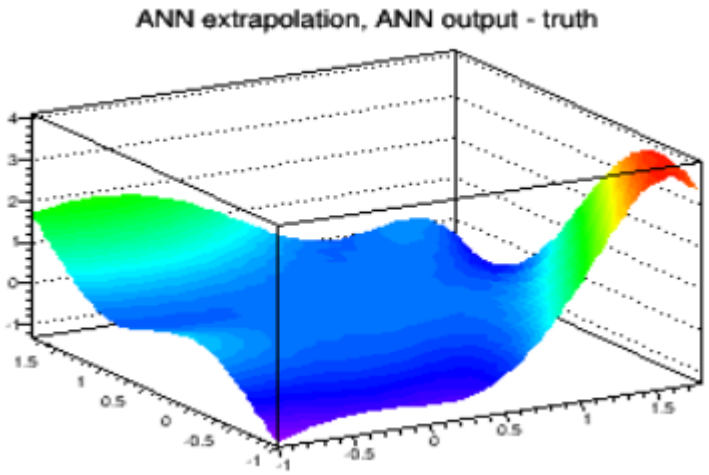
testRlt.cxx

```
double testRlt::Value(int index,double in0,double in1) {
   input0 = (in0 - 0)/1;
   input1 = (in1 - 0)/1;
   switch(index) {
     case 0:
         return neuron0x2cf0fa0();
     default:
         return 0.;
   }
}

double testRlt::Value(int index, double* input) {
   input0 = (input[0] - 0)/1;
   input1 = (input[1] - 0)/1;
   switch(index) {
     case 0:
         return neuron0x2cf0fa0();
     default:
         return 0.;
   }
}

double testRlt::neuron0x2cf0fa0() {
   double input = input0x2cf0fa0();
   return (input * 1)+0;
}

double testRlt::input0x2cf0fa0() {
   double input = -1.04701;
   input += synapse0x2cf12e0();
   input += synapse0x2cf1320();
   input += synapse0x2cf1360();
   input += synapse0x2cf13a0();
   input += synapse0x2cf13e0();
   input += synapse0x2cf1420();
   input += synapse0x2cf1460();
   input += synapse0x2cf14a0();
   return input;
}
```


Siguang@pku.edu.cn

$ROOTSYS/tutorials/mlp$ root mlpRegression.C

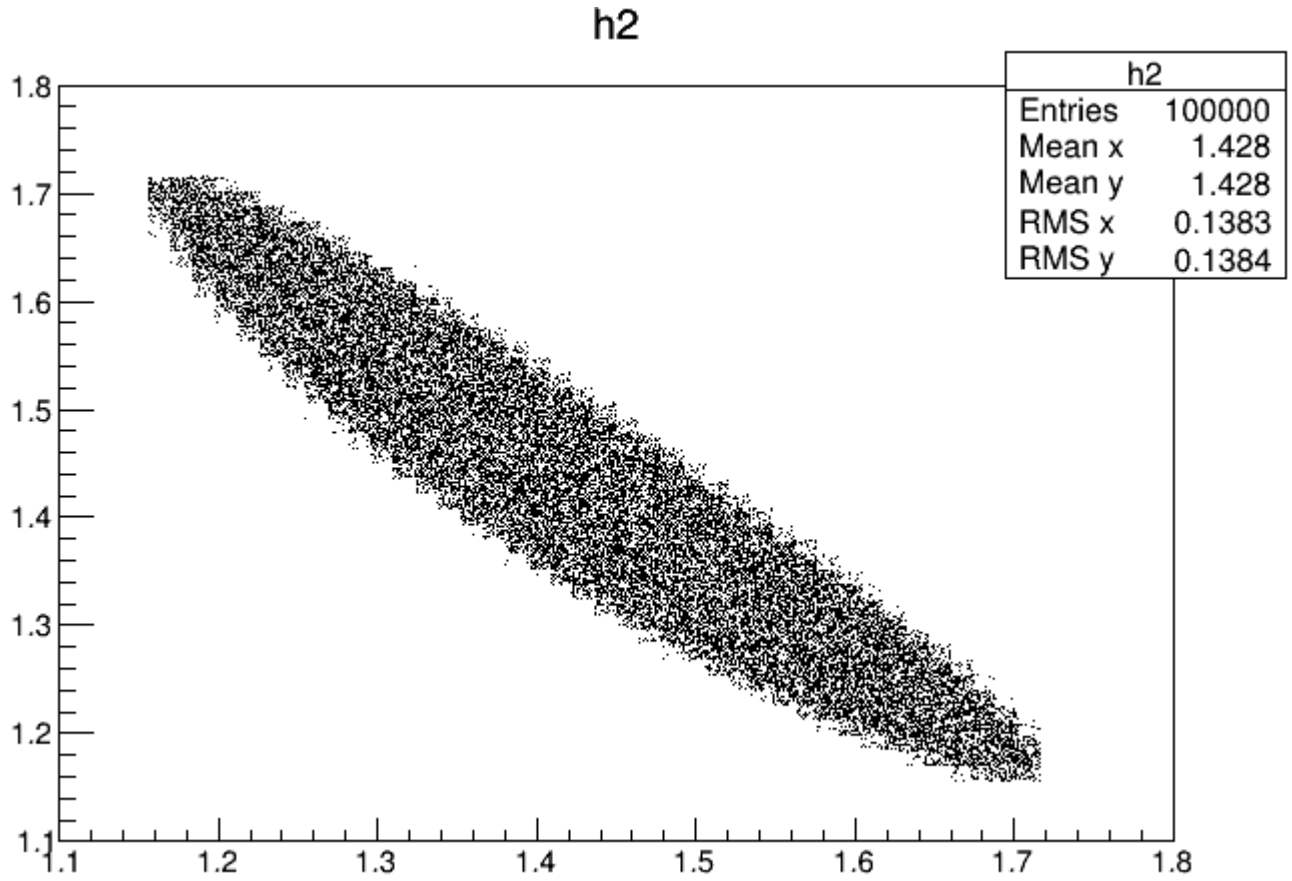
```
   // draw difference of ANN's output for (x,y) vs
   // the ANN can extrapolate
   Double_t vx[225];
   Double_t vy[225];
   Double_t delta[225];
   Double_t v[2];
   for (Int_t ix=0; ix<15; ix++) {
     v[0]=ix/5.-1.;//problem!
     //v[0]=ix/15.;//right one
      for (Int_t iy=0; iy<15; iy++) {
         v[1]=iy/5.-1.;//problem!
         //v[1]=iy/15.;//right one
         Int_t idx=ix*15+iy;
         vx[idx]=v[0];
         vy[idx]=v[1];
         delta[idx]=mlp->Evaluate(0, v)-theUnknownFunction(v[0],v[1]);
      }
   }
   TGraph2D* g2Extrapolate=new TGraph2D("ANN extrapolation",
                                        "ANN extrapolation, ANN output
- truth",
                                        225, vx, vy, delta);
   g2Extrapolate->Draw("TRI2");
}
```



x,y的学习区间是(0,1),外推到[-1,2）差异很大，故ANN的使用不要外推

PhaseSpace

$ROOTSYS/tutorials/physics/PhaseSpace.C



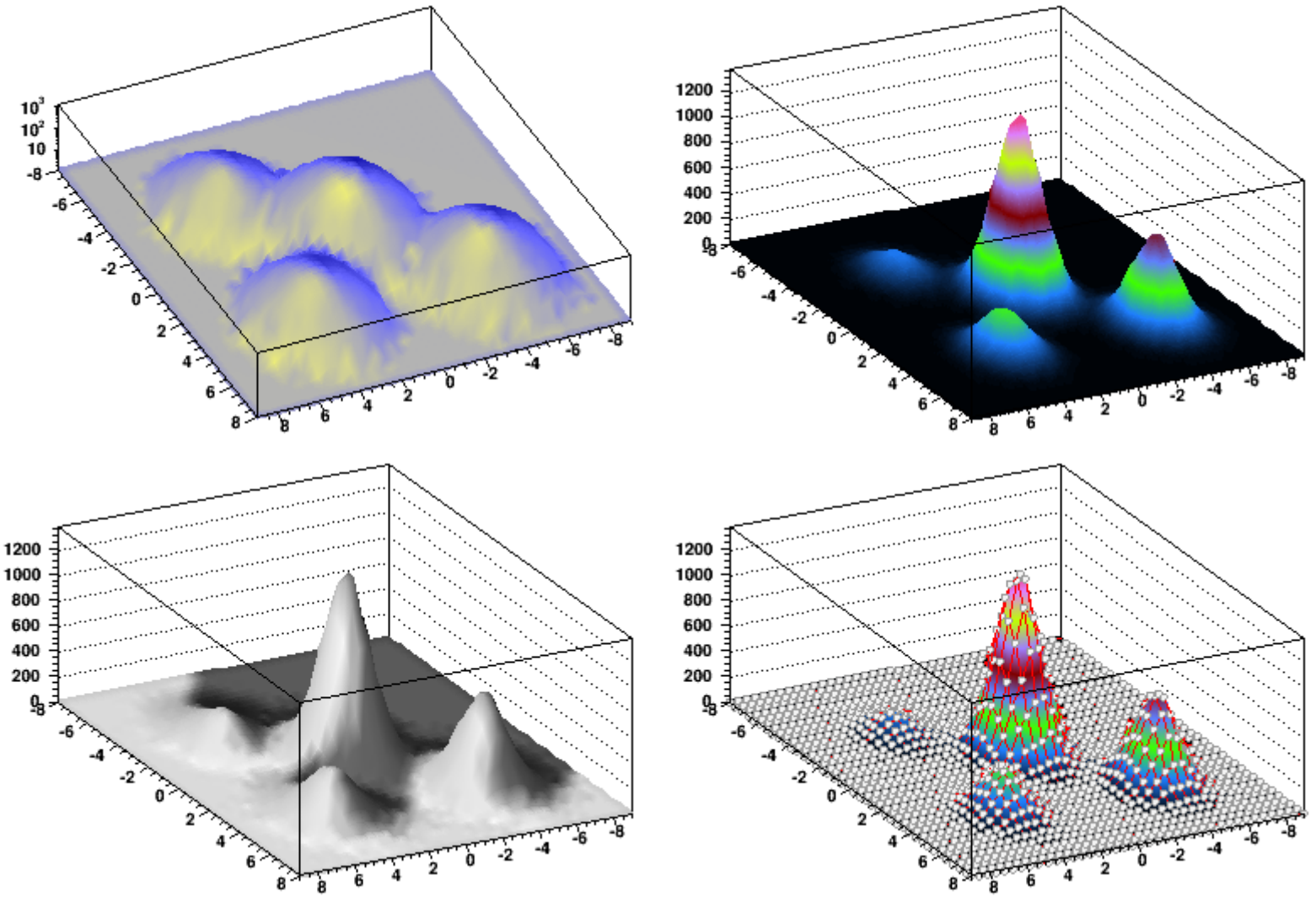
Siguang@pku.edu.cn

$ROOTSYS/tutorials/physics/PhaseSpace.C

```
void PhaseSpace() {
// example of use of TGenPhaseSpace
//Author: Valerio Filippini

   if (!gROOT->GetClass("TGenPhaseSpace")) gSystem.Load("libPhysics");

   TLorentzVector target(0.0, 0.0, 0.0, 0.938);
   TLorentzVector beam(0.0, 0.0, .65, .65);
   TLorentzVector W = beam + target;

   //(Momentum, Energy units are Gev/C, GeV)
   Double_t masses[3] = { 0.938, 0.139, 0.139} ;

   TGenPhaseSpace event;
   event.SetDecay(W, 3, masses);

   TH2F *h2 = new TH2F("h2","h2", 50,1.1,1.8, 50,1.1,1.8);

   for (Int_t n=0;n<100000;n++) {
      Double_t weight = event.Generate();

      TLorentzVector *pProton = event.GetDecay(0);

      TLorentzVector *pPip    = event.GetDecay(1);
      TLorentzVector *pPim    = event.GetDecay(2);

      TLorentzVector pPPip = *pProton + *pPip;
      TLorentzVector pPPim = *pProton + *pPim;

      h2->Fill(pPPip.M2() ,pPPim.M2() ,weight);
   }
   h2->Draw();
}
```

$$\gamma + H \rightarrow p + \pi^+ + \pi^-$$

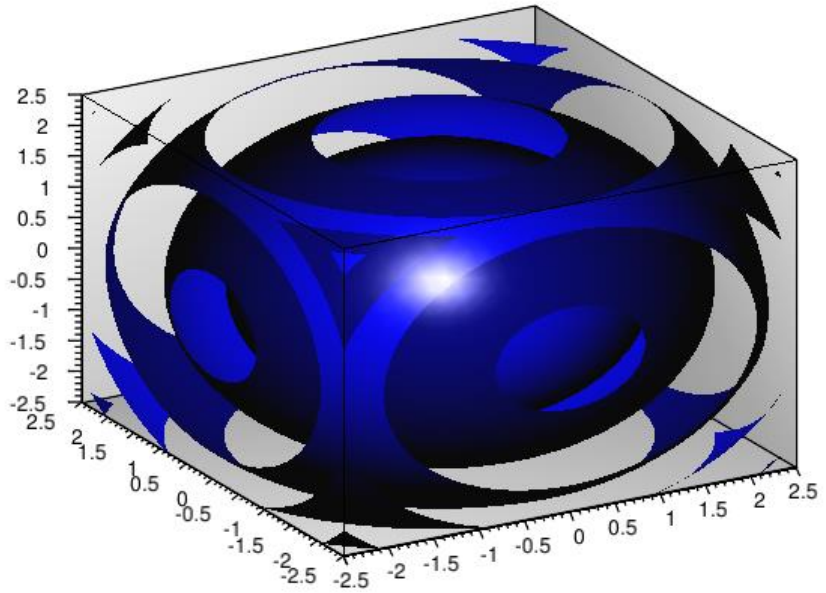
2D 显示



$ROOTSYS/tutorials/spectrum$ root spectrumpainter.C

Siguang@pku.edu.cn

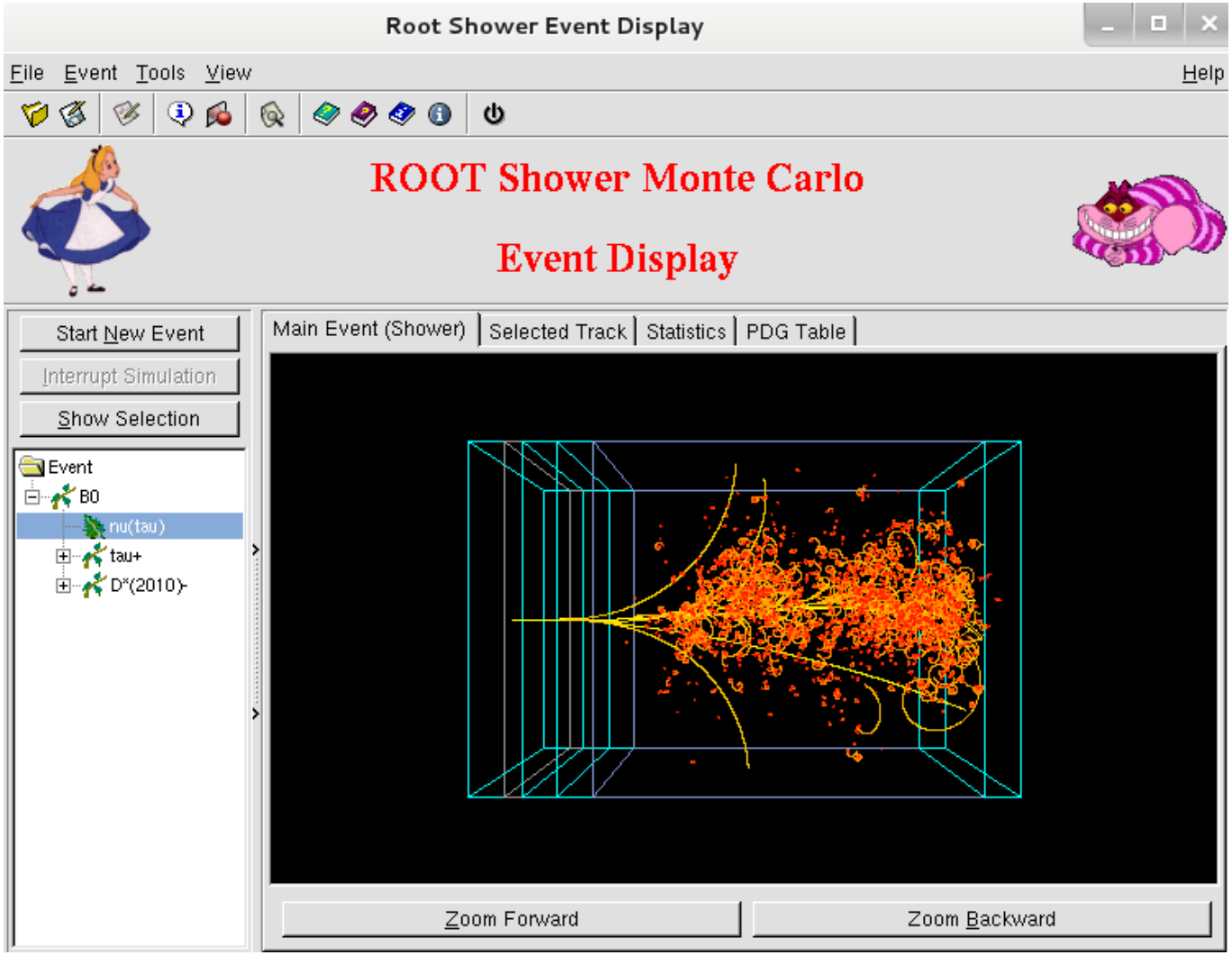
画OpenGL图



ROOT 替我们做了大量工作!

```
void glFun(){
   // after this command all legos surfaces are automatically rendered with OpenGL.
   gStyle->SetCanvasPreferGL(kTRUE);
   TCanvas *c2 = new TCanvas("glc2","");
   TF3 *fun4 = new TF3("fun4","sin(x * x + y * y + z * z - 4)",
                       -2.5, 2.5, -2.5, 2.5, -2.5, 2.5);
   fun4->SetFillColor(kBlue);
   fun4->Draw("gl");
}
```

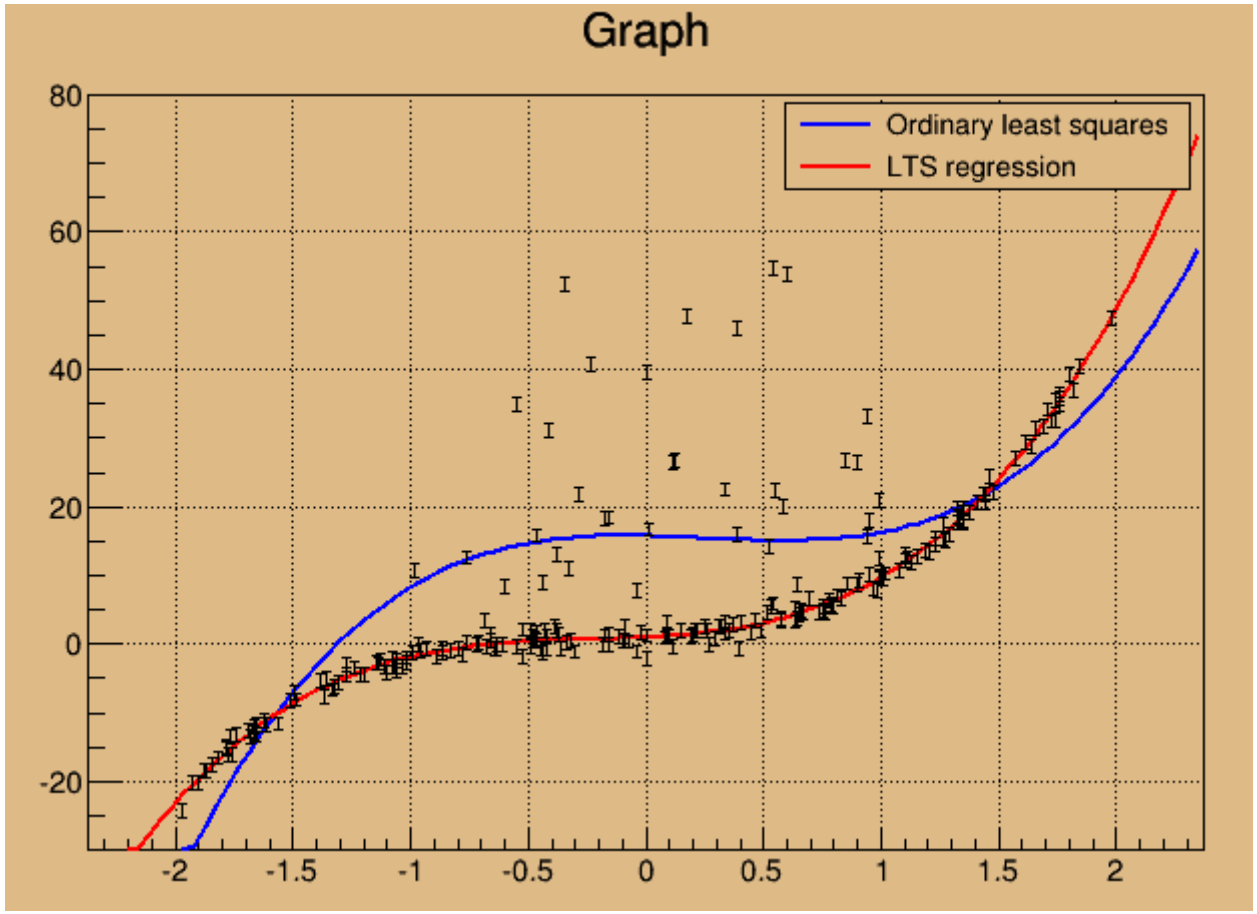
图形界面



$ROOTSYS/test/RootShower$ ./RootShower

Siguang@pku.edu.cn

$ROOTSYS/tutorials/fit/fitLinearRobust.C



Siguang@pku.edu.cn

$ROOTSYS/tutorials/fit/fitLinearRobust.C

数据准备

```
#include "TRandom.h"
#include "TGraphErrors.h"
#include "TF1.h"
#include "TCanvas.h"
#include "TLegend.h"

void fitLinearRobust()
{
//This tutorial shows how the least trimmed squares regression,
//included in the TLinearFitter class, can be used for fitting
//in cases when the data contains outliers.
//Here the fitting is done via the TGraph::Fit function with option "rob":
//If you want to use the linear fitter directly for computing
//the robust fitting coefficients, just use the TLinearFitter::EvalRobust
//function instead of TLinearFitter::Eval
//Author: Anna Kreshuk

   //First generate a dataset, where 20% of points are spoiled by large
   //errors
   Int_t npoints = 250;
   Int_t fraction = Int_t(0.8*npoints);
   Double_t *x = new Double_t[npoints];
   Double_t *y = new Double_t[npoints];
   Double_t *e = new Double_t[npoints];
  TRandom r;
  Int_t i;
  for (i=0; i<fraction; i++){
     //the good part of the sample
     x[i]=r.Uniform(-2, 2);
     e[i]=1;
     y[i]=1 + 2*x[i] + 3*x[i]*x[i] + 4*x[i]*x[i]*x[i] + e[i]*r.Gaus();
  }
  for (i=fraction; i<npoints; i++){
     //the bad part of the sample
     x[i]=r.Uniform(-1, 1);
     e[i]=1;
     y[i] = 1 + 2*x[i] + 3*x[i]*x[i] + 4*x[i]*x[i]*x[i] + r.Landau(10, 5);
  }
```

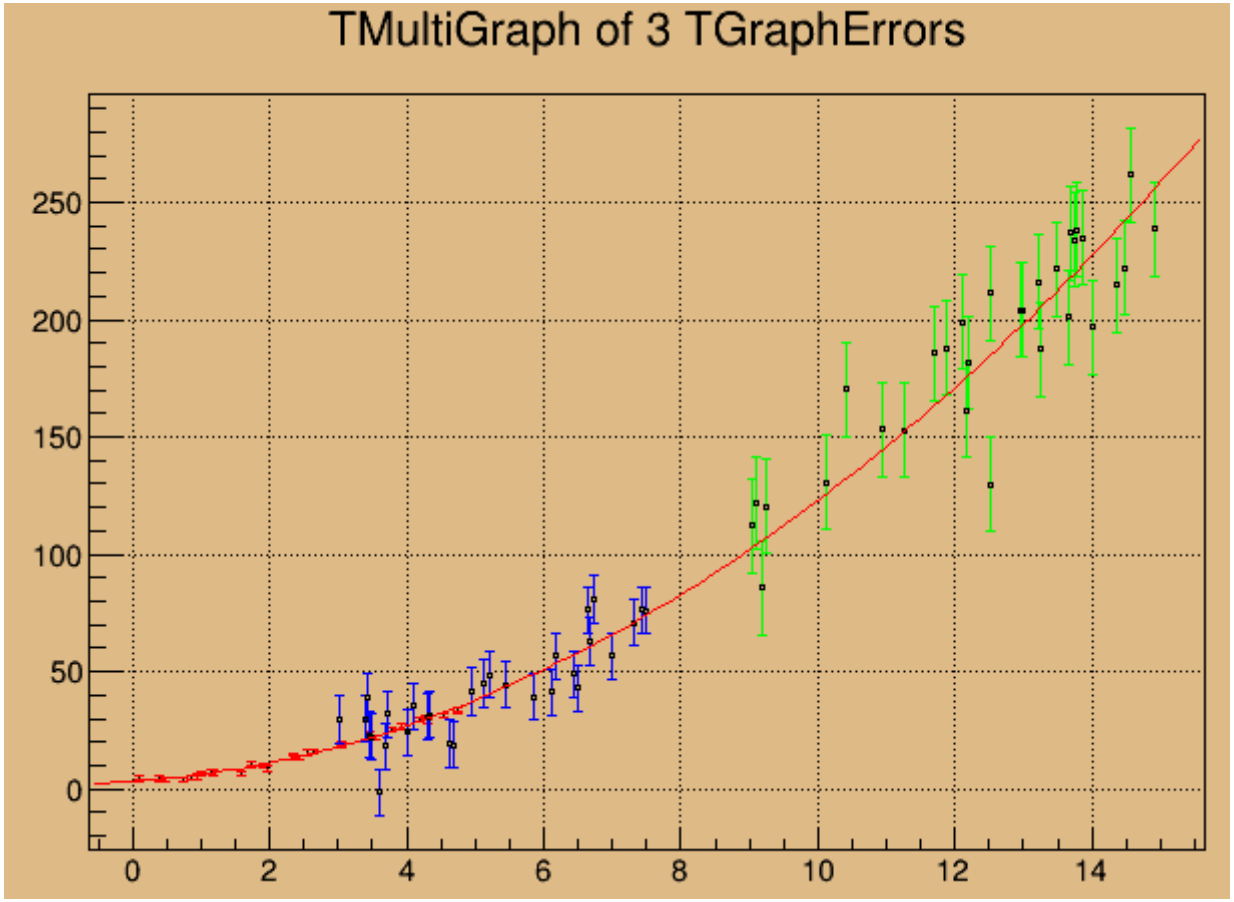
$ROOTSYS/tutorials/fit/fitLinearRobust.C

```
TGraphErrors *grr = new TGraphErrors(npoints, x, y, 0, e);
grr->SetMinimum(-30);
grr->SetMaximum(80);
TF1 *ffit1 = new TF1("ffit1", "pol3", -5, 5);
TF1 *ffit2 = new TF1("ffit2", "pol3", -5, 5);
ffit1->SetLineColor(kBlue);
ffit2->SetLineColor(kRed);
TCanvas *myc = new TCanvas("myc", "Linear and robust linear fitting");
myc->SetFillColor(42);
myc->SetGrid();
grr->Draw("ap");
//first, let's try to see the results of ordinary least-squares fit:
printf("Ordinary least squares:\n");
grr->Fit(ffit1);
//the fitted function doesn't really follow the pattern of the data
//and the coefficients are far from the real ones
```

$ROOTSYS/tutorials/fit/fitLinearRobust.C

```
printf("Resistant Least trimmed squares fit:\n");
//Now let's try the resistant regression
//The option "rob=0.75" means that we want to use robust fitting and
//we know that at least 75% of data is good points (at least 50% of points
//should be good to use this algorithm). If you don't specify any number
//and just use "rob" for the option, default value of (npoints+nparameters+1)/2
//will be taken
grr->Fit(ffit2, "+rob=0.75");
//
TLegend *leg = new TLegend(0.6, 0.8, 0.89, 0.89);
leg->AddEntry(ffit1, "Ordinary least squares", "l");
leg->AddEntry(ffit2, "LTS regression", "l");
leg->SetFillColor(42);
leg->Draw();

delete [] x;
delete [] y;
delete [] e;
```

$ROOTSYS/tutorials/fit/fitMultiGraph.C



Siguang@pku.edu.cn

$ROOTSYS/tutorials/fit/fitMultiGraph.C

```
void fitMultiGraph()
{
   //fitting a parabola to a multigraph of 3 partly overlapping graphs
   //with different errors
   //Author: Anna Kreshuk

   Int_t n = 30;
   Double_t *x1 = new Double_t[n];
   Double_t *x2 = new Double_t[n];
   Double_t *x3 = new Double_t[n];
   Double_t *y1 = new Double_t[n];
   Double_t *y2 = new Double_t[n];
   Double_t *y3 = new Double_t[n];
   Double_t *e1 = new Double_t[n];
   Double_t *e2 = new Double_t[n];
   Double_t *e3 = new Double_t[n];
```

$ROOTSYS/tutorials/fit/fitMultiGraph.C

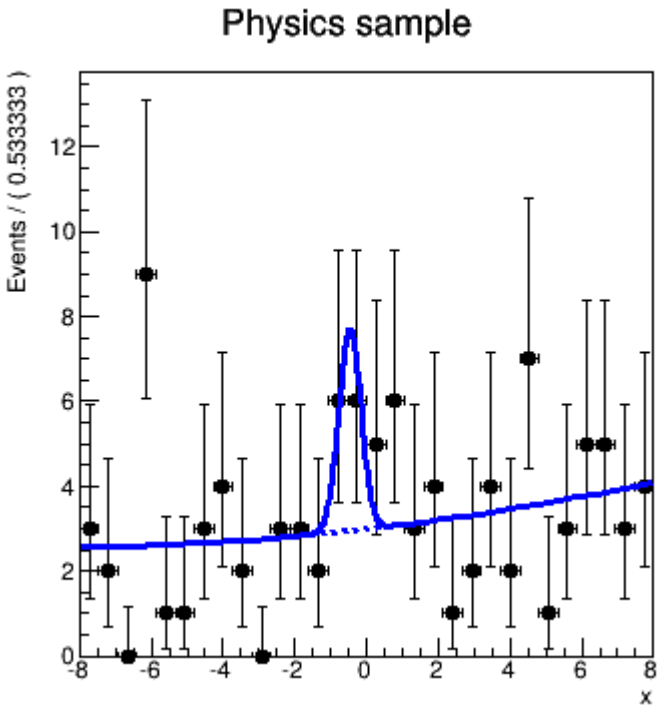
```
//generate the data for the graphs
TRandom r;
Int_t i;
for (i=0; i<n; i++) {
   x1[i] = r.Uniform(0.1, 5);
   x2[i] = r.Uniform(3, 8);
   x3[i] = r.Uniform(9, 15);
   y1[i] = 3 + 2*x1[i] + x1[i]*x1[i] + r.Gaus();
   y2[i] = 3 + 2*x2[i] + x2[i]*x2[i] + r.Gaus()*10;
   e1[i] = 1;
   e2[i] = 10;
   e3[i] = 20;
   y3[i] = 3 + 2*x3[i] + x3[i]*x3[i] + r.Gaus()*20;
}

//create the graphs and set their drawing options
TGraphErrors *gr1 = new TGraphErrors(n, x1, y1, 0, e1);
TGraphErrors *gr2 = new TGraphErrors(n, x2, y2, 0, e2);
TGraphErrors *gr3 = new TGraphErrors(n, x3, y3, 0, e3);
gr1->SetLineColor(kRed);
gr2->SetLineColor(kBlue);
gr2->SetMarkerStyle(24);
gr2->SetMarkerSize(0.3);
gr3->SetLineColor(kGreen);
gr3->SetMarkerStyle(24);
gr3->SetMarkerSize(0.3);
```

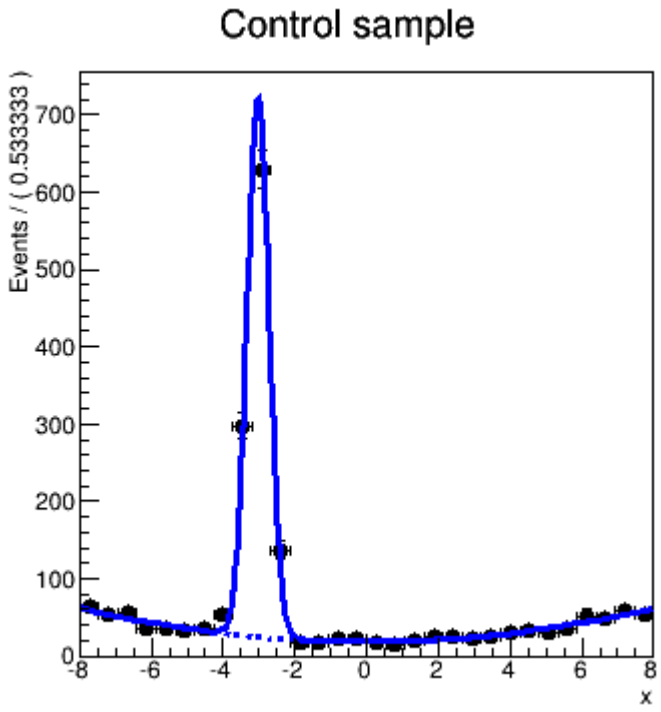
$ROOTSYS/tutorials/fit/fitMultiGraph.C

```
   TCanvas *myc = new TCanvas("myc",
      "Fitting a MultiGraph of 3 TGraphErrors");
   myc->SetFillColor(42);
   myc->SetGrid();

   mg->Draw("ap");

   //fit
   mg->Fit("pol2", "F");
   //mg->Fit("pol2");

   //access to the fit function
   TF1 *fpol = mg->GetFunction("pol2");
   fpol->SetLineWidth(1);

}
```

"F" If fitting a polN, switch to minuit fitter

$ROOTSYS/tutorials/roofit/rf501_simultaneouspdf.C





Siguang@pku.edu.cn

$ROOTSYS/tutorials/roofit/rf501_simultaneouspdf.C

```
void rf501_simultaneouspdf()
{
  // C r e a t e   m o d e l   f o r   p h y s i c s   s a m p l e
  // ---------------------------------------------------------

  // Create observables
  RooRealVar x("x","x",-8,8) ;

  // Construct signal pdf
  RooRealVar mean("mean","mean",0,-8,8) ;
  RooRealVar sigma("sigma","sigma",0.3,0.1,10) ;
  RooGaussian gx("gx","gx",x,mean,sigma) ;

  // Construct background pdf
  RooRealVar a0("a0","a0",-0.1,-1,1) ;
  RooRealVar a1("a1","a1",0.004,-1,1) ;
  RooChebychev px("px","px",x,RooArgSet(a0,a1)) ;

  // Construct composite pdf
  RooRealVar f("f","f",0.2,0.,1.) ;
  RooAddPdf model("model","model",RooArgList(gx,px),f) ;
```

$ROOTSYS/tutorials/roofit/rf501_simultaneouspdf.C

```
// C r e a t e   m o d e l   f o r   c o n t r o l   s a m p l e
// ------------------------------------------------------------

// Construct signal pdf.
// NOTE that sigma is shared with the signal sample model
RooRealVar mean_ctl("mean_ctl","mean_ctl",-3,-8,8) ;
RooGaussian gx_ctl("gx_ctl","gx_ctl",x,mean_ctl,sigma) ;

// Construct the background pdf
RooRealVar a0_ctl("a0_ctl","a0_ctl",-0.1,-1,1) ;
RooRealVar a1_ctl("a1_ctl","a1_ctl",0.5,-0.1,1) ;
RooChebychev px_ctl("px_ctl","px_ctl",x,RooArgSet(a0_ctl,a1_ctl)) ;

// Construct the composite model
RooRealVar f_ctl("f_ctl","f_ctl",0.5,0.,1.) ;
RooAddPdf model_ctl("model_ctl","model_ctl",RooArgList(gx_ctl,px_ctl),f_ctl) ;
```

共用sigma

$ROOTSYS/tutorials/roofit/rf501_simultaneouspdf.C

```
  // Generate 1000 events in x and y from model
  RooDataSet *data = model.generate(RooArgSet(x),100) ;
  RooDataSet *data_ctl = model_ctl.generate(RooArgSet(x),2000) ;


  // C r e a t e   i n d e x   c a t e g o r y   a n d   j o i n   s a m p l e s
  // ------------------------------------------------------------------------

  // Define category to distinguish physics and control samples events
  RooCategory sample("sample","sample") ;
  sample.defineType("physics") ;
  sample.defineType("control") ;

  // Construct combined dataset in (x,sample)
  RooDataSet combData("combData","combined data",x,Index(sample),Import("physics",*data),Import("control"
,*data_ctl)) ;
// C o n s t r u c t   a   s i m u l t a n e o u s   p d f   i n   ( x , s a m p l e )
// ---------------------------------------------------------------------------------

// Construct a simultaneous pdf using category sample as index
RooSimultaneous simPdf("simPdf","simultaneous pdf",sample) ;

// Associate model with the physics state and model_ctl with the control state
simPdf.addPdf(model,"physics") ;
simPdf.addPdf(model_ctl,"control") ;



// P e r f o r m   a   s i m u l t a n e o u s   f i t
// ---------------------------------------------------

// Perform simultaneous fit of model to data and model_ctl to data_ctl
simPdf.fitTo(combData) ;
```


Siguang@pku.edu.cn

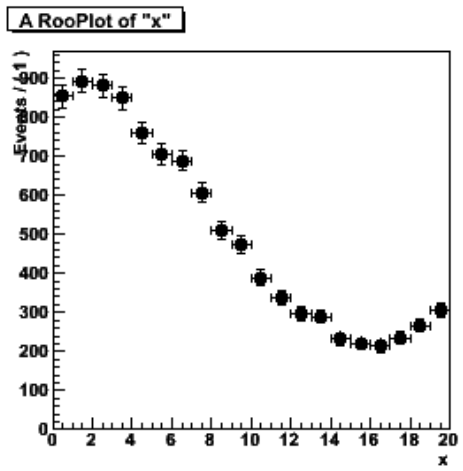
$ROOTSYS/tutorials/roofit/rf501_simultaneouspdf.C

```
  // P l o t   m o d e l   s l i c e s   o n   d a t a   s l i c e s
  // ------------------------------------------------------------

  // Make a frame for the physics sample
  RooPlot* frame1 = x.frame(Bins(30),Title("Physics sample")) ;

  // Plot all data tagged as physics sample
  combData.plotOn(frame1,Cut("sample==sample::physics")) ;

  // Plot "physics" slice of simultaneous pdf.
  // NBL You _must_ project the sample index category with data using ProjWData
  // as a RooSimultaneous makes no prediction on the shape in the index category
  // and can thus not be integrated
  simPdf.plotOn(frame1,Slice(sample,"physics"),ProjWData(sample,combData)) ;
  simPdf.plotOn(frame1,Slice(sample,"physics"),Components("px"),ProjWData(sample,combData),LineStyle(kDas
hed)) ;


    // The same plot for the control sample slice
    RooPlot* frame2 = x.frame(Bins(30),Title("Control sample")) ;
    combData.plotOn(frame2,Cut("sample==sample::control")) ;
    simPdf.plotOn(frame2,Slice(sample,"control"),ProjWData(sample,combData)) ;
    simPdf.plotOn(frame2,Slice(sample,"control"),Components("px_ctl"),ProjWData(sample,combData),LineStyle(
 kDashed)) ;



    TCanvas* c = new TCanvas("rf501_simultaneouspdf","rf403_simultaneouspdf",800,400) ;
    c->Divide(2) ;
    c->cd(1) ; gPad->SetLeftMargin(0.15) ; frame1->GetYaxis()->SetTitleOffset(1.4) ; frame1->Draw() ;
    c->cd(2) ; gPad->SetLeftMargin(0.15) ; frame2->GetYaxis()->SetTitleOffset(1.4) ; frame2->Draw() ;


}
```

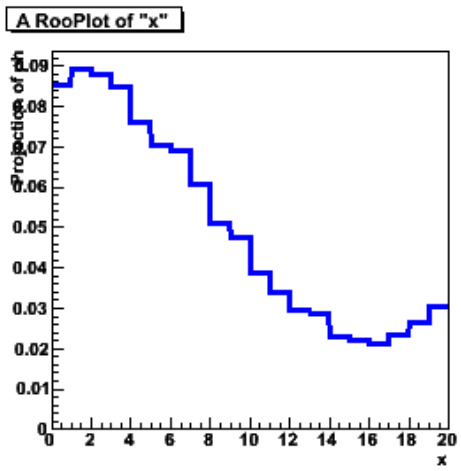
Highlight of non-parametric shapes - histograms

- Class **RooHistPdf** – a p.d.f. described by a histogram

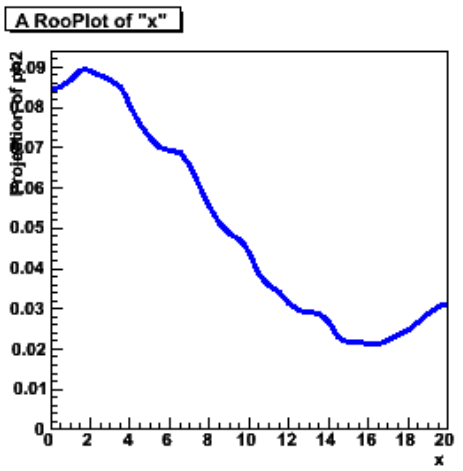
dataHist



RooHistPdf(N=0)



RooHistPdf(N=4)

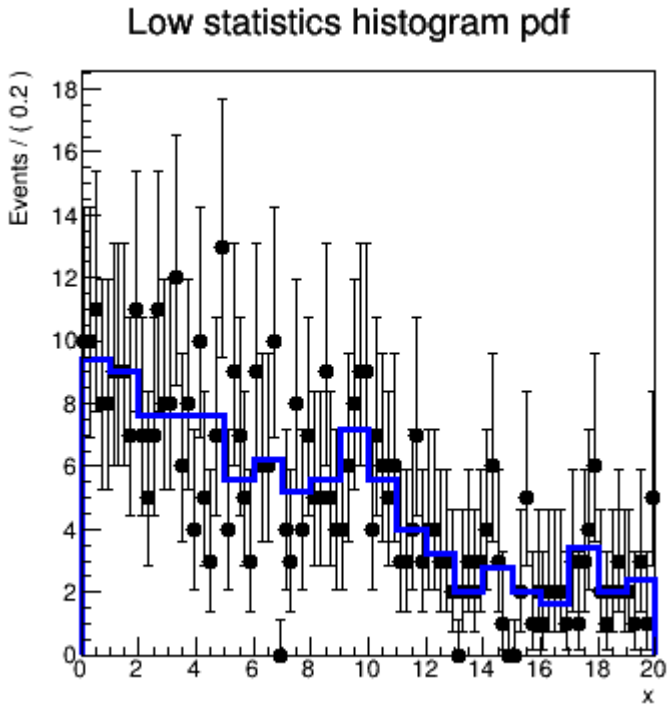


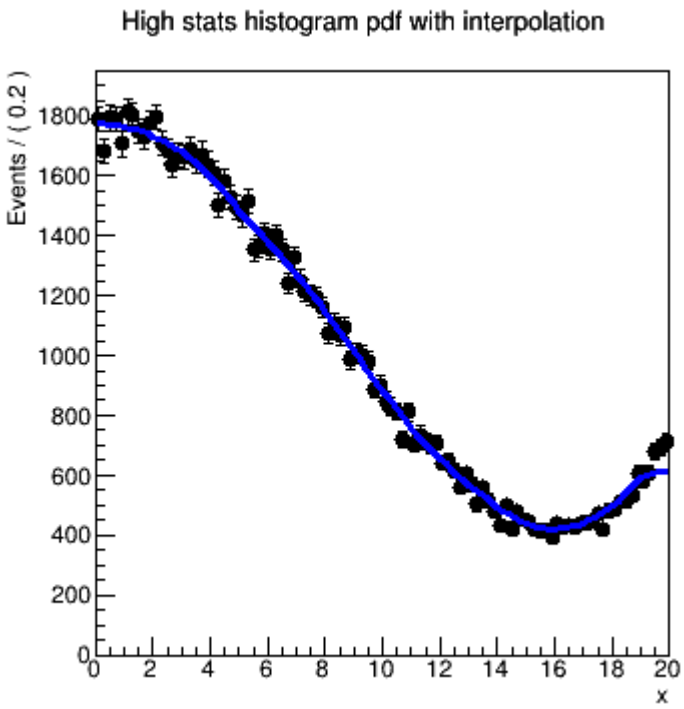

```
// Histogram based p.d.f with N-th order interpolation
RooHistPdf ph("ph","ph",x,*dataHist,N) ;
```

- Not so great at low statistics (especially problematic in >1 dim)

Siguang@pku.edu.cn

RooHistPdf





$ROOTSYS/tutorials/roofit/rf706_histpdf.C

$ROOTSYS/tutorials/roofit/rf706_histpdf.C

```
void rf706_histpdf()
{
  // C r e a t e   p d f   f o r   s a m p l i n g
  // ---------------------------------------------

  RooRealVar x("x","x",0,20) ;
  RooPolynomial p("p","p",x,RooArgList(RooConst(0.01),RooConst(-0.01),RooConst(0.0004))) ;



  // C r e a t e   l o w   s t a t s   h i s t o g r a m
  // ---------------------------------------------------

  // Sample 500 events from p
  x.setBins(20) ;
  RooDataSet* data1 = p.generate(x,500) ;

  // Create a binned dataset with 20 bins and 500 events
  RooDataHist* hist1 = data1->binnedClone() ;

  // Represent data in dh as pdf in x
  RooHistPdf histpdf1("histpdf1","histpdf1",x,*hist1,0) ;

  // Plot unbinned data and histogram pdf overlaid
  RooPlot* frame1 = x.frame(Title("Low statistics histogram pdf"),Bins(100)) ;
  data1->plotOn(frame1) ;
  histpdf1.plotOn(frame1) ;
```

$ROOTSYS/tutorials/roofit/rf706_histpdf.C

```
  // C r e a t e   h i g h   s t a t s   h i s t o g r a m
  // ---------------------------------------------------

  // Sample 100000 events from p
  x.setBins(10) ;
  RooDataSet* data2 = p.generate(x,100000) ;

  // Create a binned dataset with 10 bins and 100K events
  RooDataHist* hist2 = data2->binnedClone() ;

  // Represent data in dh as pdf in x, apply 2nd order interpolation
  RooHistPdf histpdf2("histpdf2","histpdf2",x,*hist2,2) ;

  // Plot unbinned data and histogram pdf overlaid
  RooPlot* frame2 = x.frame(Title("High stats histogram pdf with interpolation"),Bins(100)) ;
  data2->plotOn(frame2) ;
  histpdf2.plotOn(frame2) ;


  TCanvas* c = new TCanvas("rf706_histpdf","rf706_histpdf",800,400) ;
  c->Divide(2) ;
  c->cd(1) ; gPad->SetLeftMargin(0.15) ; frame1->GetYaxis()->SetTitleOffset(1.4) ; frame1->Draw() ;
  c->cd(2) ; gPad->SetLeftMargin(0.15) ; frame2->GetYaxis()->SetTitleOffset(1.8) ; frame2->Draw() ;


}
```

Nuclear Instruments and Methods in Physics Research A 600 (2009) 445–452

Contents lists available at ScienceDirect

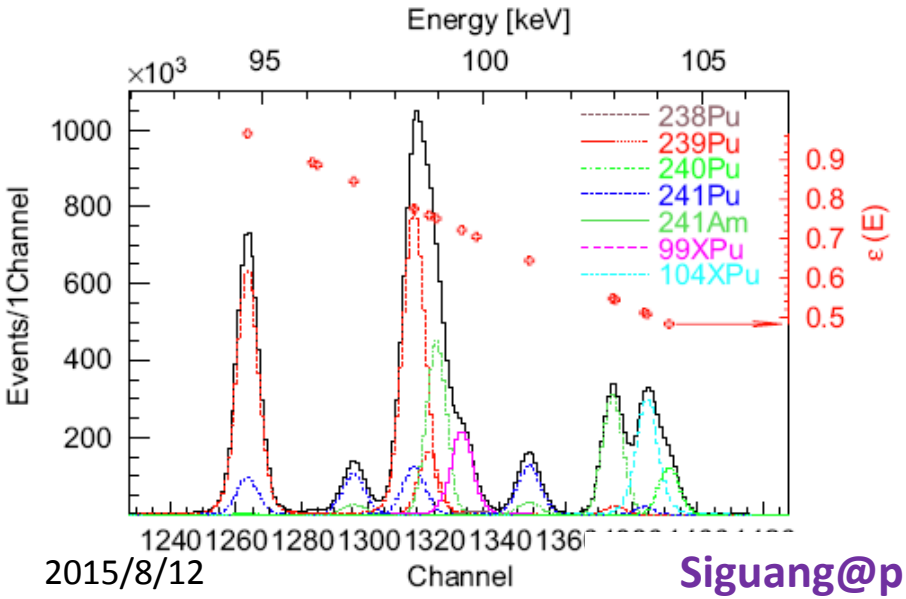
Nuclear Instruments and Methods in Physics Research A

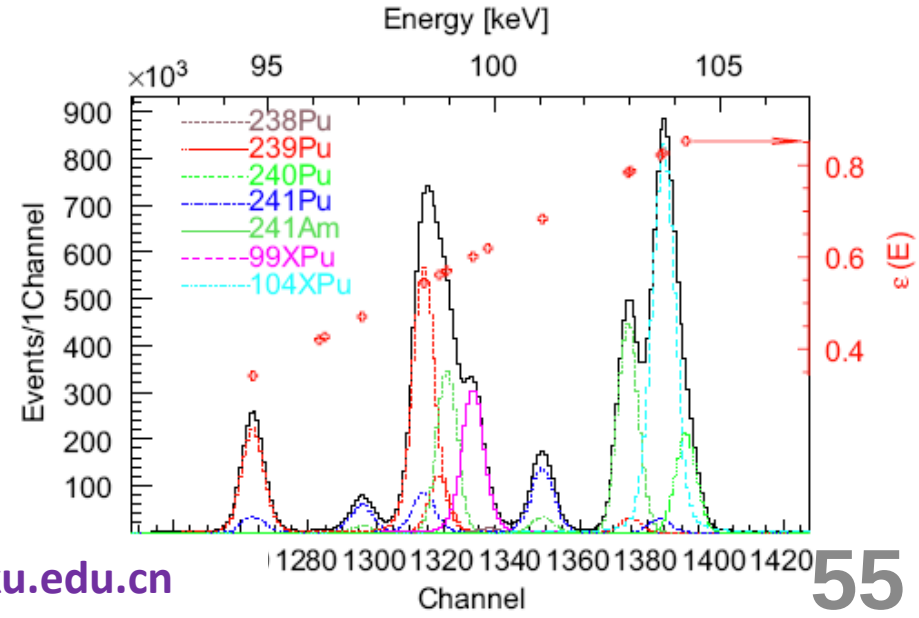
journal homepage: www.elsevier.com/locate/nima

Method for unfolding multiplet regions of X- and γ-ray spectra with a detection efficiency constraint avoiding inflecting the peak shapes for correction results ☆

Si-guang Wang [a,*], Ya-jun Mao [a], Pei-jia Tang [b]

[a] School of Physics and State Key Laboratory of Nuclear Physics and Technology, Peking University, Beijing 100871, PR China
[b] Department of Chemistry, China Institute of Atomic Energy, Beijing 102413, PR China





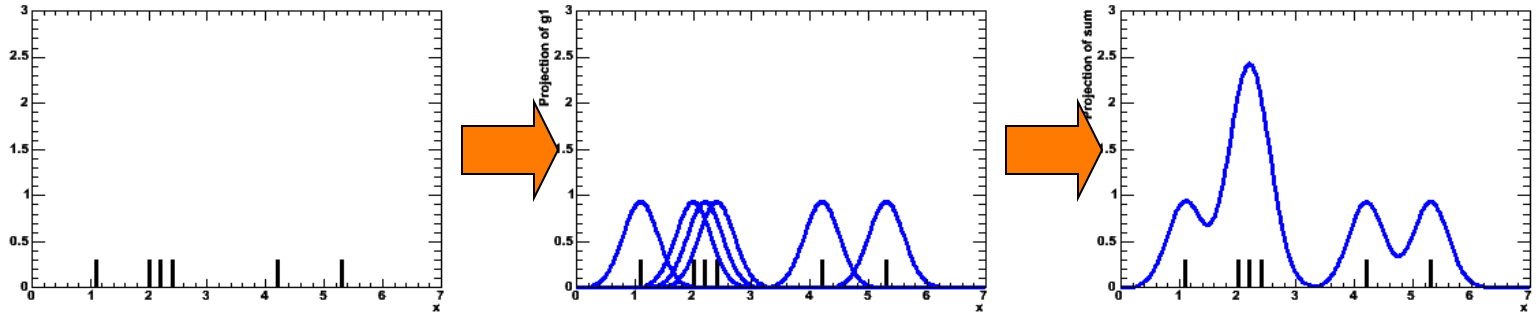
Siguang@pku.edu.cn

- Class `RooKeysPdf` – A kernel estimation p.d.f.
  - Uses *unbinned* data
  - Idea represent each event of your MC sample as a Gaussian probability distribution
  - Add probability distributions from all events in sample

Sample of events

Gaussian probability distributions for each event

Summed probability distribution for all events in sample

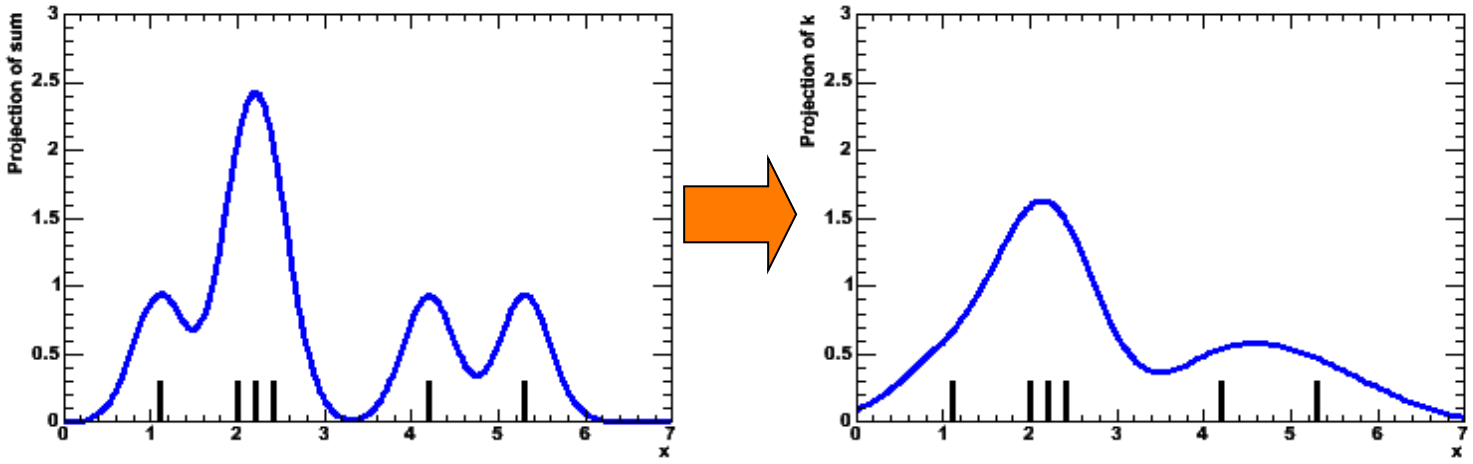


Highlight of non-parametric shapes – kernel estimation

- Width of Gaussian kernels need not be the same for all events
  - As long as each event contributes 1/N to the integral
- Idea: 'Adaptive kernel' technique(Automatically calculated）
  - Choose **wide** Gaussian if local density of events is **low** -->Preserves small features in high statistics areas,
  - Choose **narrow** Gaussian if local density of events is high→ minimize jitter in low statistics areas

Static Kernel
(with of all Gaussian identical)

Adaptive Kernel
(width of all Gaussian depends on local density of events)



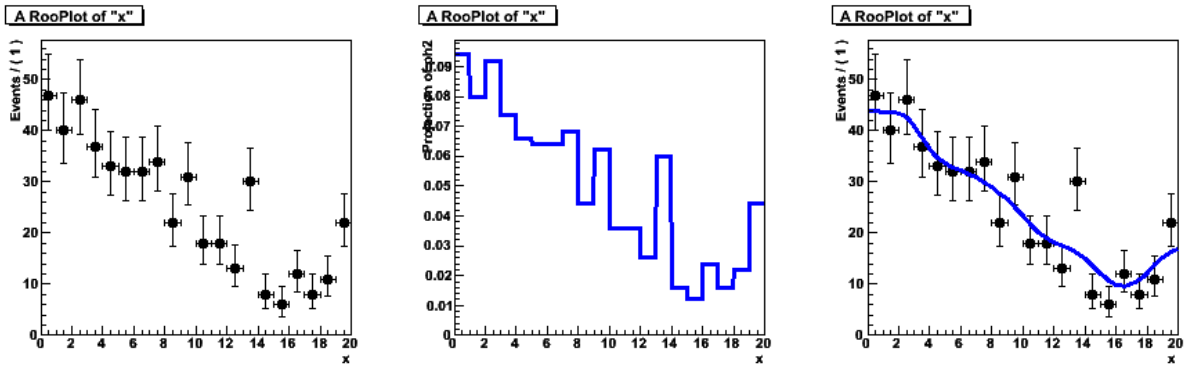
Siguang@pku.edu.cn

Highlight of non-parametric shapes – kernel estimation

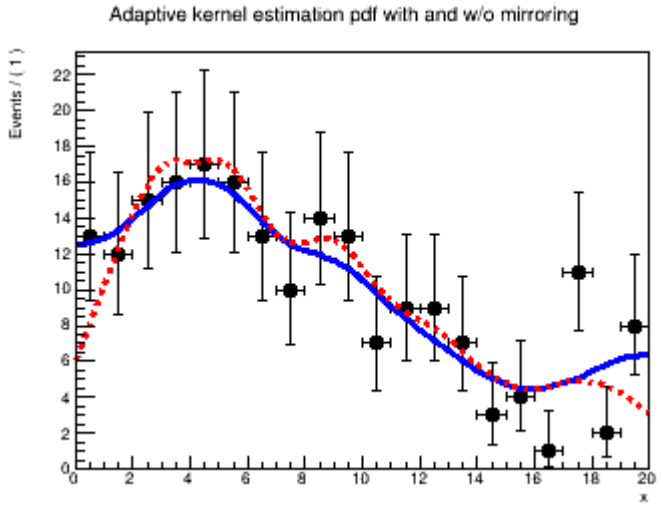
```
// Adaptive kernel estimation p.d.f
 RooKeysPdf k("k","k",x,*d,RooKeysPdf::MirrorBoth) ;
```

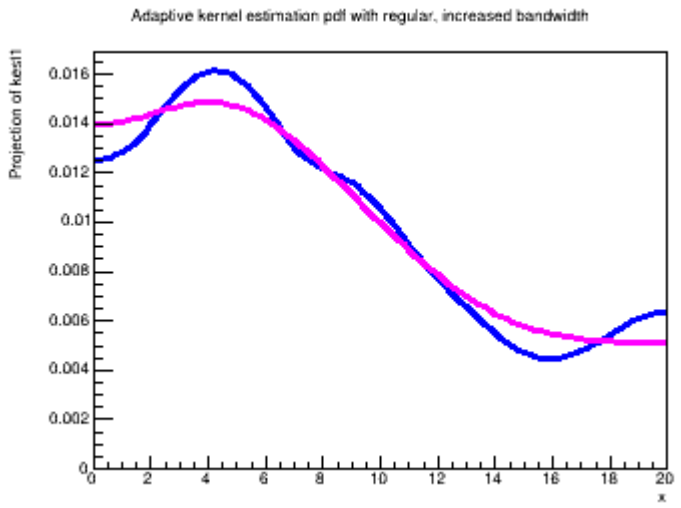
- Example with comparison to histogram based p.d.f
  - Superior performance at low statistics
  - Can mirror input data over boundaries to reduce 'edge leakage'
  - Works also in >1 dimensions (class **RooNDKeysPdf**)

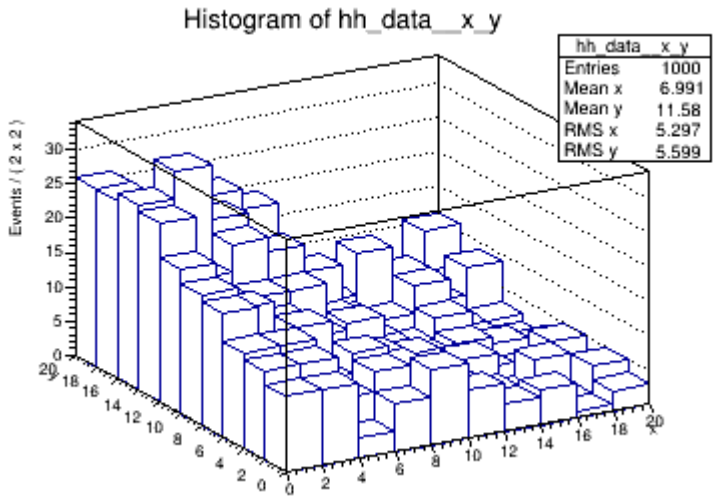
Data (N=500) RooHistPdf(data) RooKeysPdf(data)

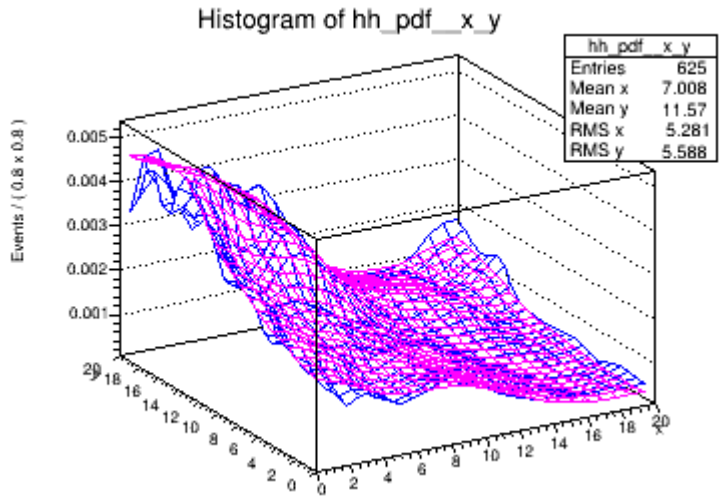


$ROOTSYS/tutorials/roofit/rf707_kernelestimation.C









Siguang@pku.edu.cn

$ROOTSYS/tutorials/roofit/rf707_kernelestimation.C

```
void rf707_kernelestimation()
{
  // C r e a t e   l o w   s t a t s   1 - D   d a t a s e t
  // ---------------------------------------------------

  // Create a toy pdf for sampling
  RooRealVar x("x","x",0,20) ;
  RooPolynomial p("p","p",x,RooArgList(RooConst(0.01),RooConst(-0.01),RooConst(0.0004))) ;

  // Sample 500 events from p
  RooDataSet* data1 = p.generate(x,200) ;



  // C r e a t e   1 - D   k e r n e l   e s t i m a t i o n   p d f
  // -------------------------------------------------------------

  // Create adaptive kernel estimation pdf. In this configuration the input data
  // is mirrored over the boundaries to minimize edge effects in distribution
  // that do not fall to zero towards the edges
  RooKeysPdf kest1("kest1","kest1",x,*data1,RooKeysPdf::MirrorBoth) ;

  // An adaptive kernel estimation pdf on the same data without mirroring option
  // for comparison
  RooKeysPdf kest2("kest2","kest2",x,*data1,RooKeysPdf::NoMirror) ;
```

$ROOTSYS/tutorials/roofit/rf707_kernelestimation.C

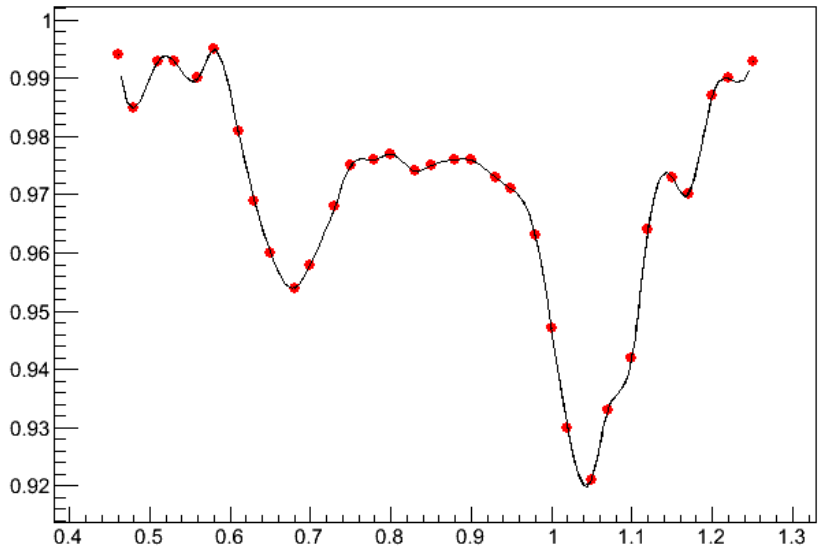
```
// Adaptive kernel estimation pdf with increased bandwidth scale factor
// (promotes smoothness over detail preservation)
RooKeysPdf kest3("kest1","kest1",x,*data1,RooKeysPdf::MirrorBoth,2) ;


// Plot kernel estimation pdfs with and without mirroring over data
RooPlot* frame = x.frame(Title("Adaptive kernel estimation pdf with and w/o mirroring"),Bins(20)) ;
data1->plotOn(frame) ;
kest1.plotOn(frame) ;
kest2.plotOn(frame,LineStyle(kDashed),LineColor(kRed)) ;


// Plot kernel estimation pdfs with regular and increased bandwidth
RooPlot* frame2 = x.frame(Title("Adaptive kernel estimation pdf with regular, increased bandwidth")) ;
kest1.plotOn(frame2) ;
kest3.plotOn(frame2,LineColor(kMagenta)) ;
```

.....

TSpline

```
void test(){
   TGraph *gr = new TGraph("test.txt");
   gr->SetMarkerStyle(20);
   gr->SetMarkerColor(kRed);
   gr->Draw("AP");
   TSpline3 *sp = new TSpline3("sp",gr);
   sp->Draw("lcpsame");
   for(Int_t i=0; i<10; i++){
     Double_t x = 0.4+i*(1.3-0.4)/10.;
     printf("x=%f Eff = %f\n",x,sp->Eval(x));
   }
}
```

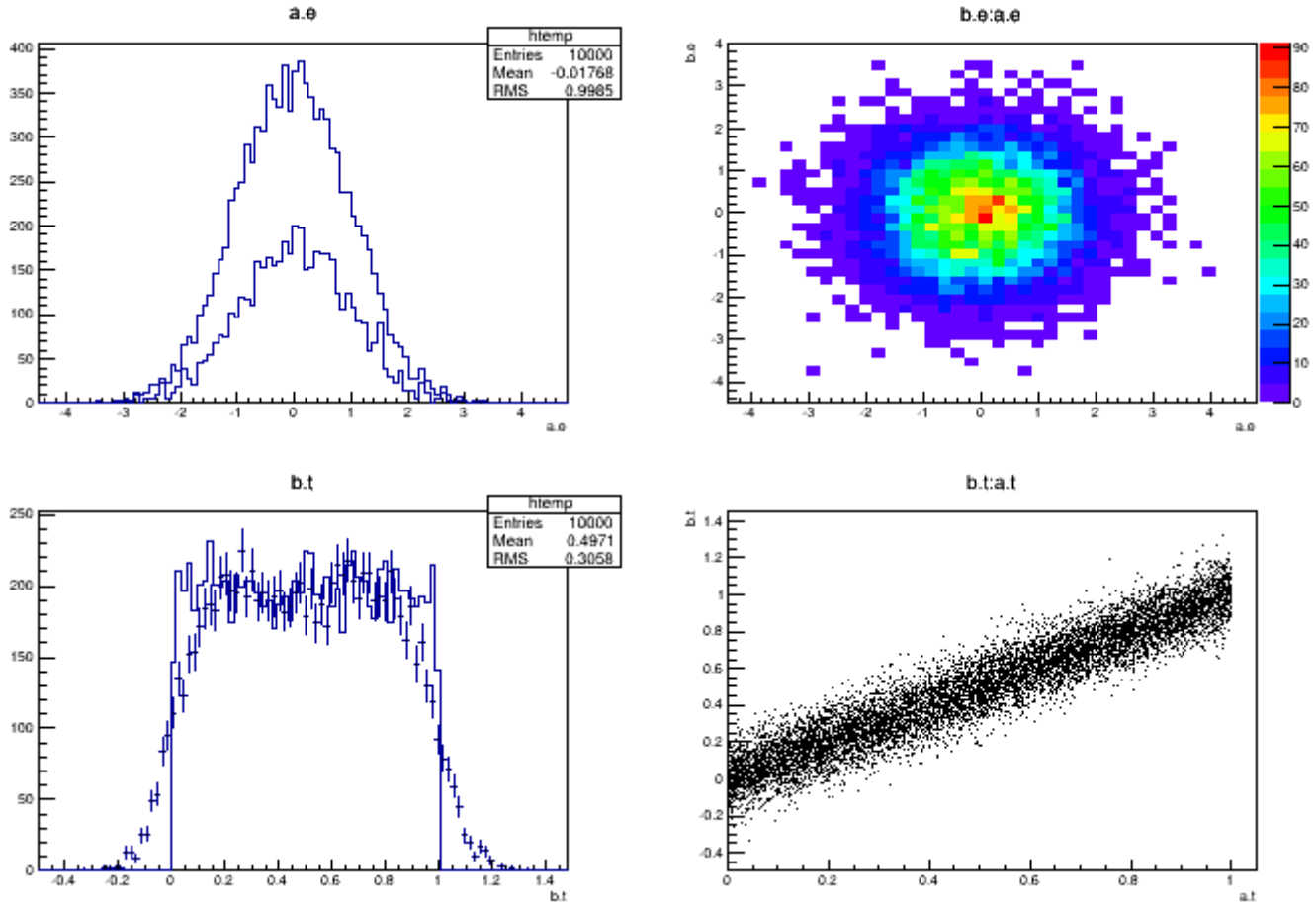
test.txt



0.46	0.994
0.48	0.985
0.51	0.993
0.53	0.993
0.56	0.99
0.58	0.995
0.61	0.981
0.63	0.969
0.65	0.96
0.68	0.954
0.7	0.958
0.73	0.968
0.75	0.975
0.78	0.976
0.8	0.977
0.83	0.974
0.85	0.975
0.88	0.976
0.9	0.976
0.93	0.973
0.95	0.971
0.98	0.963
1	0.947
1.02	0.93

Siguang@pku.edu.cn

$ROOTSYS/tutorials/tree/tree0.C



Siguang@pku.edu.cn

$ROOTSYS/tutorials/tree/tree0.C

```
#include <TRandom.h>
#include <TTree.h>
#include <TCanvas.h>
#include <TStyle.h>

#include <Riostream.h>

//class Det   : public TObject  {
class Det {  // each detector gives an energy and time signal
public:
  Double_t e; //energy
  Double_t t; //time

//  ClassDef(Det,1)
};

//ClassImp(Det)

//class Event { //TObject is not required by this example
class Event : public TObject {
public:

  Det a; // say there are two detectors (a and b) in the experiment
  Det b;
  ClassDef(Event,1)
};

ClassImp(Event)
```

$ROOTSYS/tutorials/tree/tree0.C

```
void tree0() {
  // create a TTree
  TTree *tree = new TTree("tree","treelibrated tree");
  Event *e = new Event;

  // create a branch with energy
  tree->Branch("event",&e);

  // fill some events with random numbers
  Int_t nevent=10000;
  for (Int_t iev=0;iev<nevent;iev++) {
    if (iev%1000==0) cout<<"Processing event "<<iev<<"..."<<endl;

    Float_t ea,eb;
    gRandom->Rannor(ea,eb); // the two energies follow a gaus distribution
    e->a.e=ea;
    e->b.e=eb;
    e->a.t=gRandom->Rndm();  // random
    e->b.t=e->a.t + gRandom->Gaus(0.,.1);  // identical to a.t but a gaussian
                                           // 'resolution' was added with sigma .1

    tree->Fill();  // fill the tree with the current event
  }

  // start the viewer
  // here you can investigate the structure of your Event class
  tree->StartViewer();

  //gROOT->SetStyle("Plain");   // uncomment to set a different style
  gStyle->SetPalette(1);        // use precomputed color palette 1
```

$ROOTSYS/tutorials/tree/tree0.C

```
  // now draw some tree variables
  TCanvas *c1 = new TCanvas();
  c1->Divide(2,2);
  c1->cd(1);
  tree->Draw("a.e");  //energy of det a
  tree->Draw("a.e","3*(-.2<b.e && b.e<.2)","same");  // same but with condition on energy b; scaled by 3
  c1->cd(2);
  tree->Draw("b.e:a.e","","colz");        // one energy against the other
  c1->cd(3);
  tree->Draw("b.t","","e");    // time of b with errorbars
  tree->Draw("a.t","","same"); // overlay time of detector a
  c1->cd(4);
  tree->Draw("b.t:a.t");       // plot time b again time a

  cout<<endl;
  cout<<"You can now examine the structure of your tree in the TreeViewer"<<endl;
  cout<<endl;
}
```

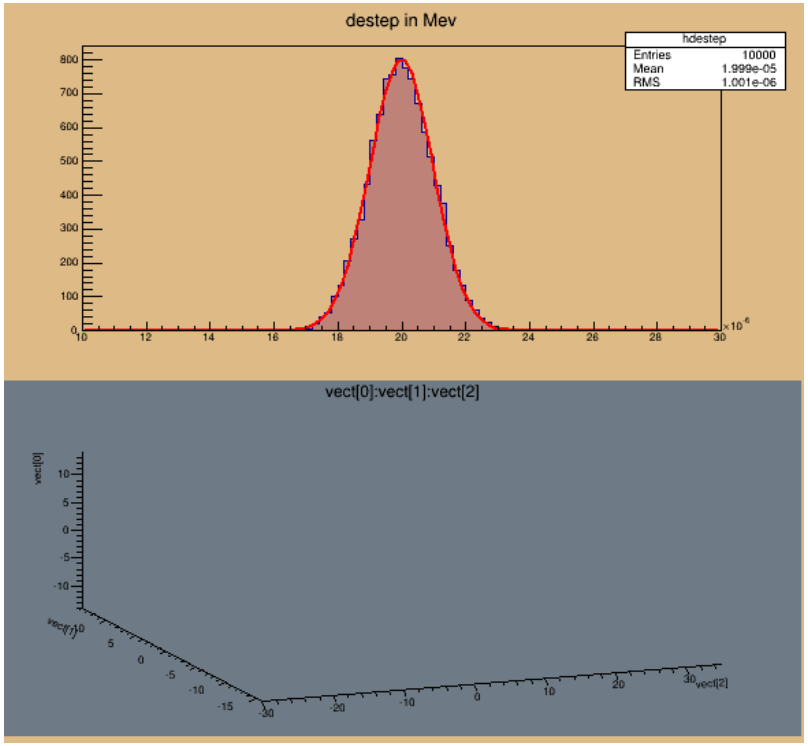
$ROOTSYS/tutorials/tree/tree1.C

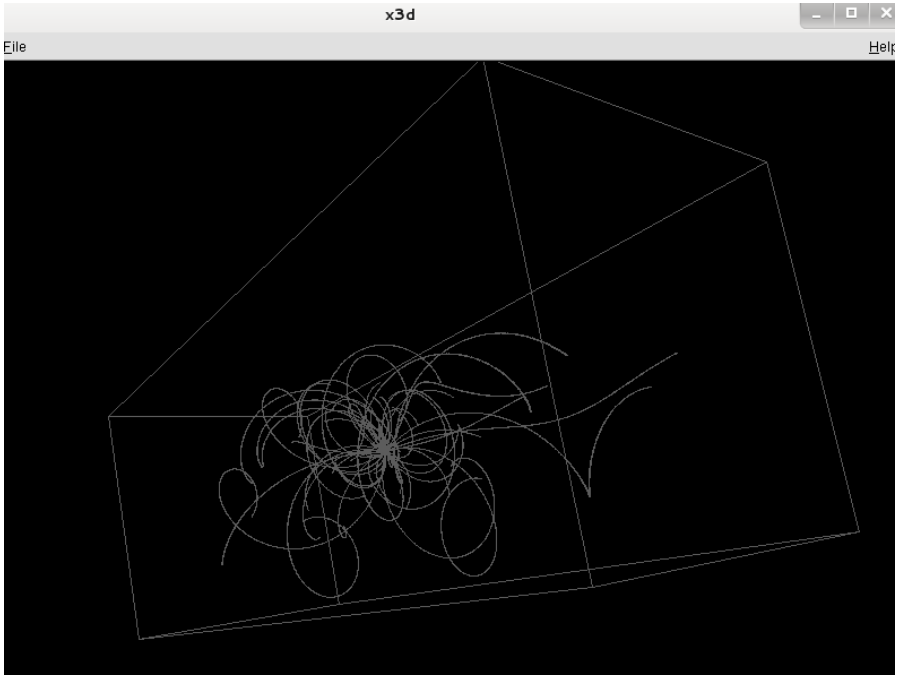
```
void tree1w()
{
   //create a Tree file tree1.root

   //create the file, the Tree and a few branches
   TFile f("tree1.root","recreate");
   TTree t1("t1","a simple Tree with simple variables");
   Float_t px, py, pz;
   Double_t random;
   Int_t ev;
   t1.Branch("px",&px,"px/F");
   t1.Branch("py",&py,"py/F");
   t1.Branch("pz",&pz,"pz/F");
   t1.Branch("random",&random,"random/D");
   t1.Branch("ev",&ev,"ev/I");

   //fill the tree
   for (Int_t i=0;i<10000;i++) {
     gRandom->Rannor(px,py);
     pz = px*px + py*py;
     random = gRandom->Rndm();
     ev = i;
     t1.Fill();
  }

  //save the Tree header. The file will be automatically closed
  //when going out of the function scope
  t1.Write();
}
```

$ROOTSYS/tutorials/tree/tree1.C

```
void tree1r()
{
   //read the Tree generated by tree1w and fill two histograms

   //note that we use "new" to create the TFile and TTree objects !
   //because we want to keep these objects alive when we leave this function.
   TFile *f = new TFile("tree1.root");
   TTree *t1 = (TTree*)f->Get("t1");
   Float_t px, py, pz;
   Double_t random;
   Int_t ev;
   t1->SetBranchAddress("px",&px);
   t1->SetBranchAddress("py",&py);
   t1->SetBranchAddress("pz",&pz);
   t1->SetBranchAddress("random",&random);
   t1->SetBranchAddress("ev",&ev);

   //create two histograms
   TH1F *hpx   = new TH1F("hpx","px distribution",100,-3,3);
   TH2F *hpxpy = new TH2F("hpxpy","py vs px",30,-3,3,30,-3,3);

   //read all entries and fill the histograms
   Long64_t nentries = t1->GetEntries();
   for (Long64_t i=0;i<nentries;i++) {
     t1->GetEntry(i);
     hpx->Fill(px);
     hpxpy->Fill(px,py);
  }

  //we do not close the file. We want to keep the generated histograms
  //we open a browser and the TreeViewer
  if (gROOT->IsBatch()) return;
  new TBrowser();
  t1->StartViewer();
  // in the browser, click on "ROOT Files", then on "tree1.root".
  //     you can click on the histogram icons in the right panel to draw them.
  // in the TreeViewer, follow the instructions in the Help button.
}
```

$ROOTSYS/tutorials/tree/tree2.C





$ROOTSYS/tutorials/tree/tree2.C

```
const Int_t MAXMEC = 30;

class Gctrak : public TObject {
public:
  Float_t  vect[7];
  Float_t  getot;
  Float_t  gekin;
  Float_t  vout[7];   //! not persistent
  Int_t    nmec;
  Int_t   *lmec;      //[nmec]
  Int_t   *namec;     //[nmec]
  Int_t    nstep;     //! not persistent
  Int_t    pid;
  Float_t  destep;
  Float_t  destel;    //! not persistent
  Float_t  safety;    //! not persistent
  Float_t  sleng;     //! not persistent
  Float_t  step;      //! not persistent
  Float_t  snext;     //! not persistent
  Float_t  sfield;    //! not persistent
  Float_t  tofg;      //! not persistent
  Float_t  gekrat;    //! not persistent
  Float_t  upwght;    //! not persistent

  Gctrak() {lmec=0; namec=0;}

  ClassDef(Gctrak,1)
};
```

$ROOTSYS/tutorials/tree/tree2.C

```
void helixStep(Float_t step, Float_t *vect, Float_t *vout)
{
  // extrapolate track in constant field
   Float_t field = 20;       //magnetic field in kilogauss
   enum Evect {kX,kY,kZ,kPX,kPY,kPZ,kPP};
   vout[kPP] = vect[kPP];
   Float_t h4    = field*2.99792e-4;
   Float_t rho   = -h4/vect[kPP];
   Float_t tet   = rho*step;
   Float_t tsint = tet*tet/6;
   Float_t sintt = 1 - tsint;
   Float_t sint  = tet*sintt;
   Float_t cos1t = tet/2;
   Float_t f1 = step*sintt;
   Float_t f2 = step*cos1t;
   Float_t f3 = step*tsint*vect[kPZ];
   Float_t f4 = -tet*cos1t;
   Float_t f5 = sint;
   Float_t f6 = tet*cos1t*vect[kPZ];
   vout[kX]   = vect[kX]  + (f1*vect[kPX] - f2*vect[kPY]);
   vout[kY]   = vect[kY]  + (f1*vect[kPY] + f2*vect[kPX]);
   vout[kZ]   = vect[kZ]  + (f1*vect[kPZ] + f3);
   vout[kPX]  = vect[kPX] + (f4*vect[kPX] - f5*vect[kPY]);
   vout[kPY]  = vect[kPY] + (f4*vect[kPY] + f5*vect[kPX]);
   vout[kPZ]  = vect[kPZ] + (f4*vect[kPZ] + f6);
}
```

$ROOTSYS/tutorials/tree/tree2.C

```
void tree2aw()
{
   //create a Tree file tree2.root

   //create the file, the Tree and a few branches with
   //a subset of gctrak
   TFile f("tree2.root","recreate");
   TTree t2("t2","a Tree with data from a fake Geant3");
   Gctrak *gstep = new Gctrak;
   t2.Branch("track",&gstep,8000,1);

   //Initialize particle parameters at first point
   Float_t px,py,pz,p,charge=0;
   Float_t vout[7];
   Float_t mass  = 0.137;
   Bool_t newParticle = kTRUE;
   gstep->lmec    = new Int_t[MAXMEC];
   gstep->namec   = new Int_t[MAXMEC];
   gstep->step    = 0.1;
   gstep->destep  = 0;
   gstep->nmec    = 0;
   gstep->pid     = 0;
```

Branch(branchname, &object, bufsize, splitlevel)

$ROOTSYS/tutorials/tree/tree2.C

```
//transport particles
for (Int_t i=0;i<10000;i++) {
   //generate a new particle if necessary
   if (newParticle) {
      px = gRandom->Gaus(0,.02);
      py = gRandom->Gaus(0,.02);
      pz = gRandom->Gaus(0,.02);
      p  = TMath::Sqrt(px*px+py*py+pz*pz);
      charge = 1; if (gRandom->Rndm() < 0.5) charge = -1;
      gstep->pid    += 1;
      gstep->vect[0] = 0;
      gstep->vect[1] = 0;
      gstep->vect[2] = 0;
      gstep->vect[3] = px/p;
      gstep->vect[4] = py/p;
      gstep->vect[5] = pz/p;
      gstep->vect[6] = p*charge;
      gstep->getot   = TMath::Sqrt(p*p + mass*mass);
      gstep->gekin   = gstep->getot - mass;
      newParticle = kFALSE;
   }

   // fill the Tree with current step parameters
   t2.Fill();

   //transport particle in magnetic field
   helixStep(gstep->step, gstep->vect, vout); //make one step
```

$ROOTSYS/tutorials/tree/tree2.C

```
      //apply energy loss
      gstep.destep = gstep.step*gRandom->Gaus(0.0002,0.00001);
      gstep.gekin -= gstep.destep;
      gstep.getot  = gstep.gekin + mass;
      gstep.vect[6] = charge*TMath::Sqrt(gstep.getot*gstep.getot - mass*mass);
      gstep.vect[0] = vout[0];
      gstep.vect[1] = vout[1];
      gstep.vect[2] = vout[2];
      gstep.vect[3] = vout[3];
      gstep.vect[4] = vout[4];
      gstep.vect[5] = vout[5];
      gstep.nmec    = (Int_t)(5*gRandom->Rndm());
      for (Int_t l=0;l<gstep.nmec;l++) gstep.lmec[l] = 1;
      if (gstep.gekin < 0.001)              newParticle = kTRUE;
      if (TMath::Abs(gstep.vect[2]) > 30) newParticle = kTRUE;
   }

   //save the Tree header. The file will be automatically closed
   //when going out of the function scope
   t2.Write();
}
```

$ROOTSYS/tutorials/tree/tree2.C

```
void tree2ar()
{
   //read the Tree generated by tree2w and fill one histogram
   //we are only interested by the destep branch.

   //note that we use "new" to create the TFile and TTree objects !
   //because we want to keep these objects alive when we leave
   //this function.
   TFile *f = new TFile("tree2.root");
   TTree *t2 = (TTree*)f->Get("t2");
   Gctrak *gstep = 0;
   t2->SetBranchAddress("track",&gstep);
   TBranch *b_destep = t2->GetBranch("destep");

   //create one histogram
   TH1F *hdestep   = new TH1F("hdestep","destep in Mev",100,1e-5,3e-5);

   //read only the destep branch for all entries
   Long64_t nentries = t2->GetEntries();
   for (Long64_t i=0;i<nentries;i++) {
      b_destep->GetEntry(i);
      hdestep->Fill(gstep->destep);
   }
```

$ROOTSYS/tutorials/tree/tree2.C

```
   //we do not close the file.
   //We want to keep the generated histograms
   //We fill a 3-d scatter plot with the particle step coordinates
   TCanvas *c1 = new TCanvas("c1","c1",600,800);
   c1->SetFillColor(42);
   c1->Divide(1,2);
   c1->cd(1);
   hdestep->SetFillColor(45);
   hdestep->Fit("gaus");
   c1->cd(2);
   gPad->SetFillColor(37);
   t2->SetMarkerColor(kRed);
   t2->Draw("vect[0]:vect[1]:vect[2]");
   if (gROOT->IsBatch()) return;

   // invoke the x3d viewer
   gPad->GetViewer3D("x3d");
}

void tree2a() {
   tree2aw();
   tree2ar();
}
```

ROOT的安装

ROOT的安装

快速安装：在联网的状态下，在Ubuntu、Debian操作系统下执行
apt-get install root-system
即可

编译源代码安装

从root.cern.ch获取源代码后,执行如下代码解压缩

tar –zxvf root root_v5.34.18.source.tar.gz

执行结果是在当前目录下产生一root子目录，然后顺序执行如下：

```
#install root under debian----
export ROOTSYS=/home/wsg/work/root/534/
mkdir -p $ROOTSYS
cd root
#fftw
apt-get install fftw3

#For Debian to compile
apt-get install build-essential
apt-get instal ldpkg-dev
#OpenGL
apt-get install libxft-dev
apt-get install libgl1-mesa-dev
apt-get install libxt-dev
apt-get install libxmu-dev
apt-get install libxmu-headers
apt-get install libxmu6
apt-get install libxmu6-dbg
apt-get install libxmuu-dev
apt-get install libxmuu1
apt-get install  libxmuu1-dbg &

#
apt-get intall libghc-sdl-ttf-dev &

cmake ../root -DCMAKE_INSTALL_PREFIX=~/work/root/534 -Droottest=ON -Dfftw3=ON -Droofit=ON -Dqt=ON
make -j2
make install
```

注：不同环境已经安装的包不同，在运行cmake时，如果缺XXX库什么系统会提醒，用apt-cache search XXX查找，用apt-get install XXX 安装

ROOT 需要的包

根据
https://root.cern.ch/drupal/content/build-prerequisites
需要如下的包：

sudo apt-get install git dpkg-dev make g++ gcc binutils libx11-dev libxpm-dev libxft-dev libxext-dev

Optional packages:

sudo apt-get install gfortran libssl-dev libpcre3-dev xlibmesa-glu-dev libglew1.5-dev libftgl-dev libmysqlclient-dev libfftw3-dev cfitsio-dev graphviz-dev libavahi-compat-libdnssd-dev libldap2-dev python-dev libxml2-dev libkrb5-dev libgsl0-dev libqt4-dev

如果有ROOT权限，执行apt。。。。即可

编译源代码安装(更省事的安装)

```
export FFTW_DIR=/scratch/other/wangsg/root64/fftw
mkdir tmpRootCompile
cd tmpRootCompile
cmake ../root -DCMAKE_INSTALL_PREFIX =/home/wsg/work/root/534 -Dall=on  -Dfail-on-missing=OFF

make -j40
make install
```

其中:
../root 指向root解压缩后的代码目录
-DCMAKE_INSTALL_PREFIX =/home/wsg/work/root/534指向安装目录
-Dall=on 打开所有选项
-Dfail-on-missing=OFF 如果没有找到需要的外挂库,继续执行其余安装
make –j40 用40个核同时编译
make install 进行安装

具体安装参考:
https://root.cern.ch/drupal/content/installing-root-source

设置环境

检查运行环境： echo $0

如果返回bash

```
wsg@debian:~$ cd
wsg@debian:~$ emacs .bashrc &
在文件中加入：
export ROOTSYS=/home/wsg/work/root/534
export PATH=$ROOTSYS/bin:$PATH
export LD_LIBRARY_PATH=$ROOTSYS/lib:$LD_LIBRARY_PATH
```

如果返回-tcsh

```
wsg@debian:~$ cd
wsg@debian:~$ emacs .tcshrc &
在文件中加入：
setenv ROOTSYS   /home/wsg/work/root/534
setenv PATH   $ROOTSYS/bin:$PATH
setenv LD_LIBRARY_PATH   $ROOTSYS/lib:$LD_LIBRARY_PATH
```

重新开窗口即可输入root

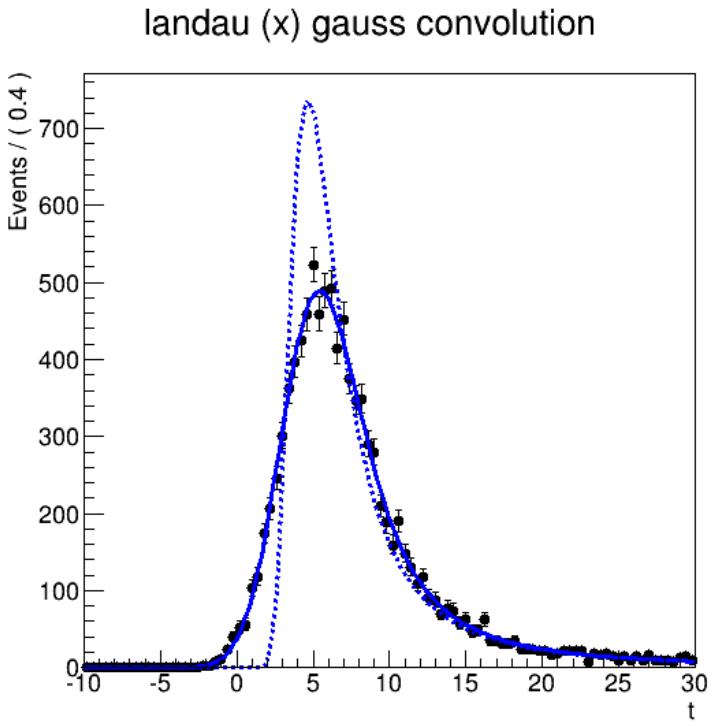
安装后的运行

$ROOTSYS/tutorials/

在$ROOTSYS/tutorials/下有很多例子程序，运行方法为：

$cd $ROOTSYS/tutorials/roofit

$root rf208_convolution.C



看能否出现左图。
如果能，说明你的root、roofit软件包、fftw软件安装成功。

如果不能。。。。。或许系统差异大。。。。直接安装虚拟机好了。

Siguang@pku.edu.cn

编译运行

红色为您输入的文字

```
wsg@debian:~/work/root/534/tutorials/roofit$ root
root [0] .L rf208_convolution.C++
Info in <TUnixSystem::ACLiC>: creating shared library
/home/wsg/work/root/534/tutorials/roofit/./rf208_convolution_C.so

RooFit v3.60 -- Developed by Wouter Verkerke and David Kirkby
                Copyright (C) 2000-2013 NIKHEF, University of California & Stanford University
                All rights reserved, please read http://roofit.sourceforge.net/license.txt

root [1] rf208_convolution()
```

$ROOTSYS/tutorials/rf208_convolution.C

编译运行头文件要包含

```
///////////////////////////////////////////////////////////////////////////
//
// 'ADDITION AND CONVOLUTION' RooFit tutorial macro #208
//
// One-dimensional numeric convolution
// (require ROOT to be compiled with --enable-fftw3)
//
// pdf = landau(t) (x) gauss(t)
//
//
// 07/2008 - Wouter Verkerke
//
///////////////////////////////////////////////////////////////////////////

#ifndef __CINT__
#include "RooGlobalFunc.h"
#endif
#include "RooRealVar.h"
#include "RooDataSet.h"
#include "RooGaussian.h"
#include "RooLandau.h"
#include "RooFFTConvPdf.h"
#include "RooPlot.h"
#include "TCanvas.h"
#include "TAxis.h"
#include "TH1.h"
using namespace RooFit ;
```

rf208_convolution.C

```
void rf208_convolution()
{
  // S e t u p   c o m p o n e n t   p d f s
  // ---------------------------------------

  // Construct observable
  RooRealVar t("t","t",-10,30) ;

  // Construct landau(t,ml,sl) ;
  RooRealVar ml("ml","mean landau",5.,-20,20) ;
  RooRealVar sl("sl","sigma landau",1,0.1,10) ;
  RooLandau landau("lx","lx",t,ml,sl) ;

  // Construct gauss(t,mg,sg)
  RooRealVar mg("mg","mg",0) ;
  RooRealVar sg("sg","sg",2,0.1,10) ;
  RooGaussian gauss("gauss","gauss",t,mg,sg) ;


  // C o n s t r u c t   c o n v o l u t i o n   p d f
  // -------------------------------------------------

  // Set #bins to be used for FFT sampling to 10000
  t.setBins(10000,"cache") ;

  // Construct landau (x) gauss
  RooFFTConvPdf lxg("lxg","landau (X) gauss",t,landau,gauss) ;
```

rf208_convolution.C

```
  // S a m p l e ,   f i t   a n d   p l o t   c o n v o l u t e d   p d f
  // ---------------------------------------------------------------------

  // Sample 1000 events in x from gxlx
  RooDataSet* data = lxg.generate(t,10000) ;

  // Fit gxlx to data
  lxg.fitTo(*data) ;

  // Plot data, landau pdf, landau (X) gauss pdf
  RooPlot* frame = t.frame(Title("landau (x) gauss convolution")) ;
  data->plotOn(frame) ;
  lxg.plotOn(frame) ;
  landau.plotOn(frame,LineStyle(kDashed)) ;


  // Draw frame on canvas
  new TCanvas("rf208_convolution","rf208_convolution",600,600) ;
  gPad->SetLeftMargin(0.15) ; frame->GetYaxis()->SetTitleOffset(1.4) ; frame->Draw() ;

}
```

可以把ROOT作为普通库函数进行连接

```
wsg@debian:/media/sf_UbuntuShare/rootEdu/CmakeTest/cmakeRunCode/GuiRunCode$ ll
total 12
-rwxrwx--- 1 root vboxsf 1834 Aug 12 15:23 CMakeLists.txt
-rwxrwx--- 1 root vboxsf 9645 Jul  8 18:51 FindROOT.cmake
drwxrwx--- 1 root vboxsf    0 Jul  7 23:15 include
-rwxrwx--- 1 root vboxsf  411 Aug 12 15:25 runTest.C
drwxrwx--- 1 root vboxsf    0 Jul  7 23:09 src
```

runTest.C

```
#include "Tools.hh"
#include <TCanvas.h>
#include <TApplication.h>
#include <TRint.h>
int main(int argc, char *argv[]){
  TApplication *theApp = new TRint("App", &argc, argv);

  SetSgStyle();
  printf("run...\n");
  TH1D *h = newTH1D("h",0.1,-5,5);
  h->FillRandom("gaus",10000);
  TCanvas *c1 = new TCanvas("c1","");
  h->Draw();
  txtM(0.2,0.94,h);
  c1->SaveAs("test.eps");

  theApp->Run();
  return 0;
}
```

FindROOT.cmake (http://root.cern.ch/drupal/sites/default/files/event.tgz)
直接放在主目录下

参照CMakeLists.txt建立自己的文件

Include目录放头文件(Tools.hh)
Src目录放.cc文件（Tools.cc）

可以把ROOT作为普通库函数进行连接

CMakeLists.txt

```
#----------------------------------------------------------------------------
# Setup the project
#
cmake_minimum_required(VERSION 2.6 FATAL_ERROR)
project(runTest)



#----------------------------------------------------------------------------
# Find ROOT (Modified it as an option in future)
#set(CMAKE_MODULE_PATH $ENV{G4MODULES} ${CMAKE_MODULE_PATH})
#set(CMAKE_MODULE_PATH $ENV{ROOTSYS}/cmake/modules/  ${CMAKE_MODULE_PATH})
set(CMAKE_MODULE_PATH ${CMAKE_SOURCE_DIR}  ${CMAKE_MODULE_PATH})

find_package(ROOT)
if(ROOT_FOUND)
  message(STATUS "ROOT found.")
else()
    message(STATUS "ROOT not found")
endif()


#----------------------------------------------------------------------------

include_directories(${PROJECT_SOURCE_DIR}/include
                    ${ROOT_INCLUDE_DIR} )
```

```
#----------------------------------------------------------------
# Locate sources and headers for this project
# NB: headers are included so they will show up in IDEs
#
file(GLOB sources ${PROJECT_SOURCE_DIR}/src/*.cc
          )
file(GLOB headers ${PROJECT_SOURCE_DIR}/include/*.hh
          )

#----------------------------------------------------------------
# Add the executable, and link it to the Geant4 libraries
#
add_executable(runTest runTest.C ${sources} ${headers})
target_link_libraries(runTest  ${ROOT_LIBRARIES} )
```

可以把ROOT作为普通库函数进行连接

编译及运行

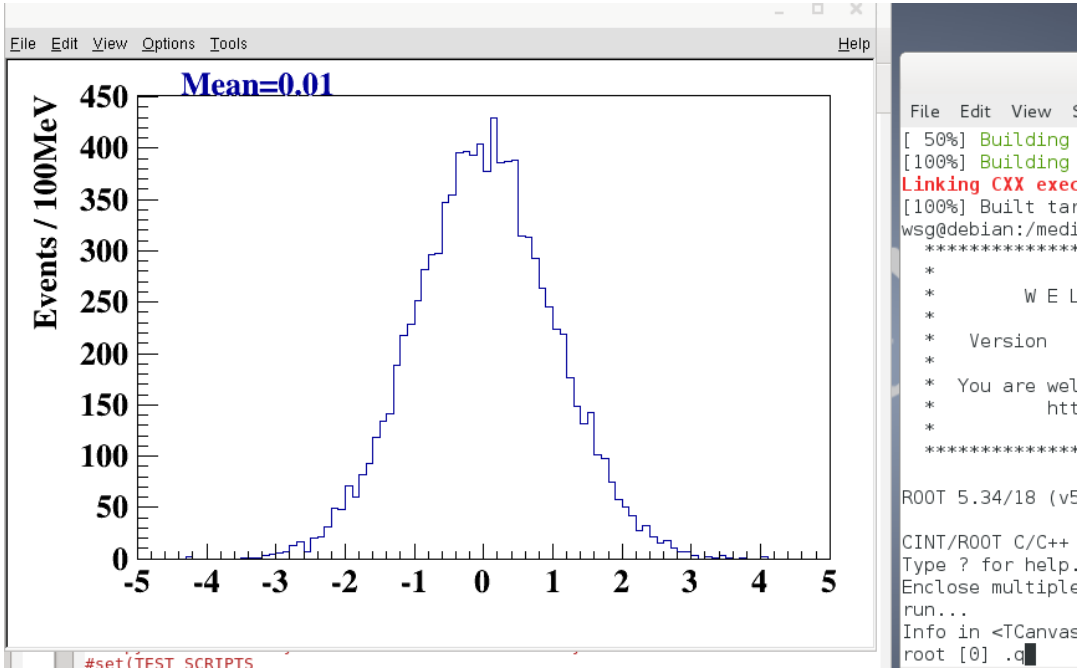
```
>cmake ../GuiRunCode/
>make
>./runTest
root[0].q
```

注：后者为CmakeLists.txt 所在的目录

注：生成可执行程序 runTest

注：运行

注： .q 退出



Siguang@pku.edu.cn

A ROOT Guide For Beginners

来自：

https://root.cern.ch/drupal/content/users-guide#primer

2.1 ROOT as calculator

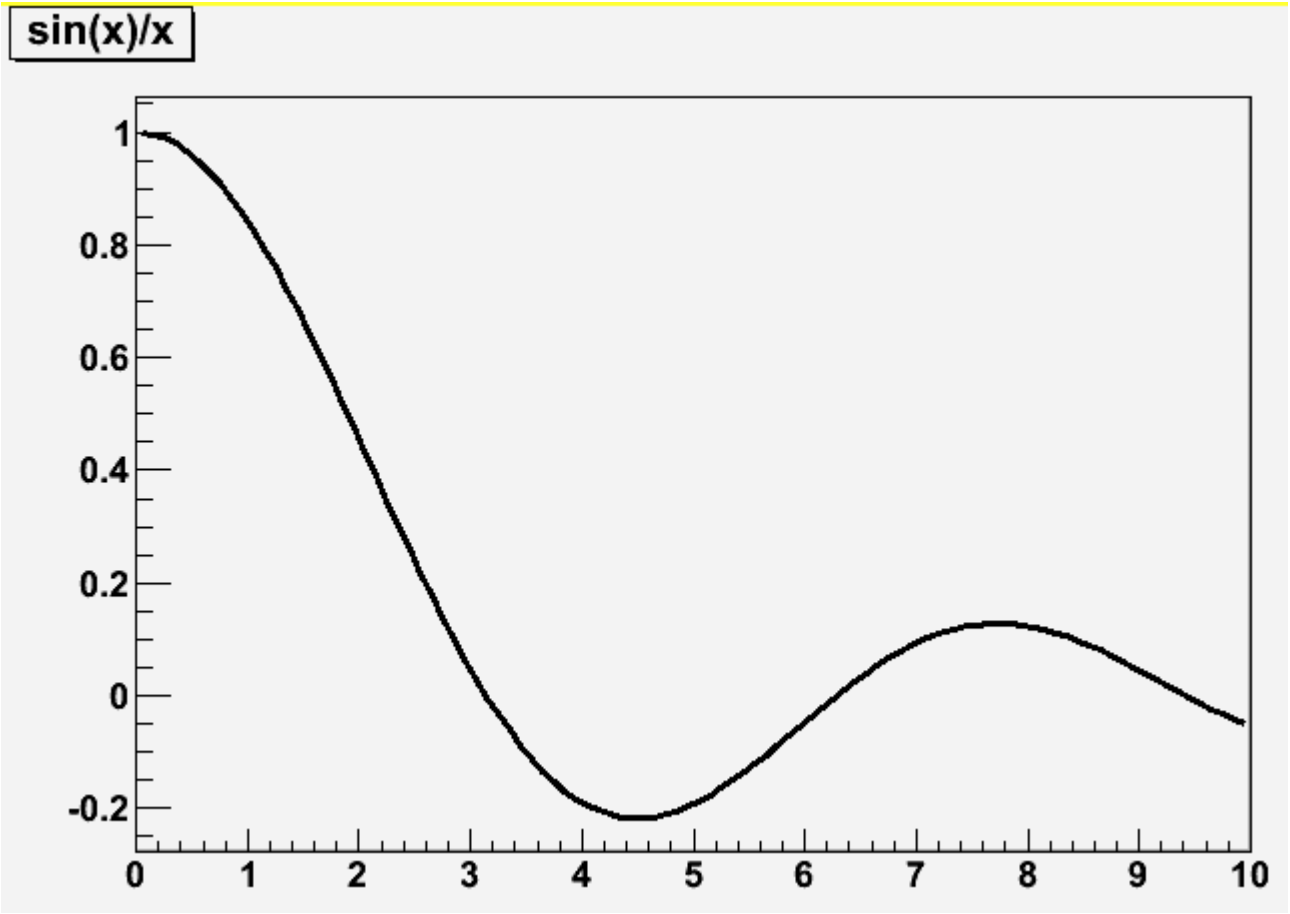
```
root [0] 1+1
(const int)2
root [1] 2*(4+2)/12.
(const double)1.00000000000000e+00
root [2] sqrt(3)
(const double)1.73205080756887719e+00
root [3] 1 > 2
(const int)0
root [4] TMath::Pi()
(Double_t)3.14159265358979312e+00
root [5] TMath::Erf(.2)
(Double_t)2.22702589210478447e-01
```

ROOT as calculator [Continue]

```
root [6] double x=.5
root [7] int N=30
root [8] double geom_series=0
root [9] for (int i=0;i<N;++i)geom_series+=TMath::Power(x,i)
root [10] TMath::Abs(geom_series - (1-TMath::Power(x,N-1))/(1-x))
(Double_t)1.8626451492309570e-09
```

2.2 ROOT as Function Plotter

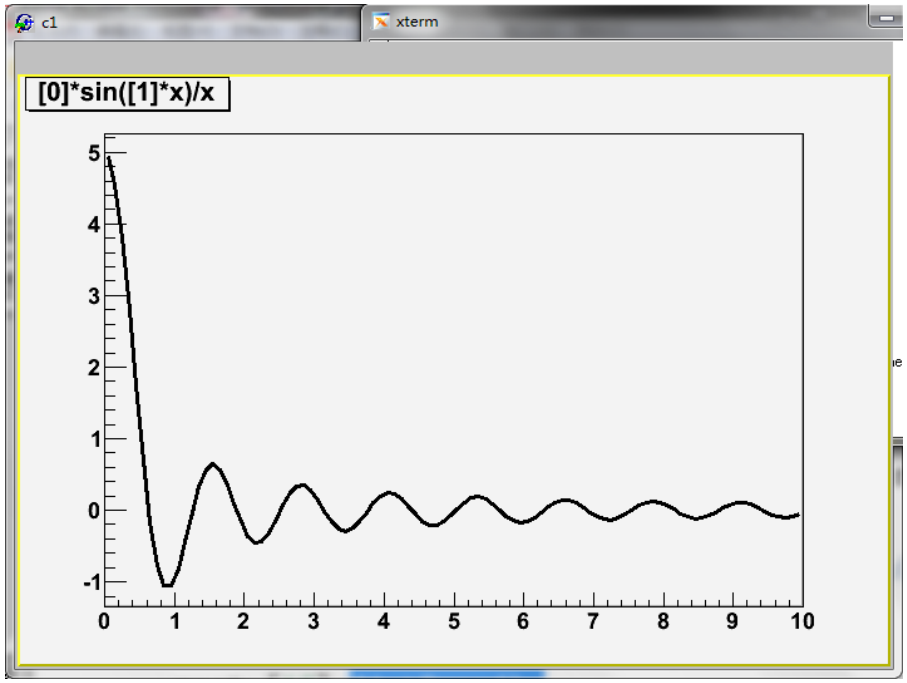
```
root [11] TF1 *f1 = new TF1("f1","sin(x)/x",0.,10.);
root [12] f1->Draw();
```



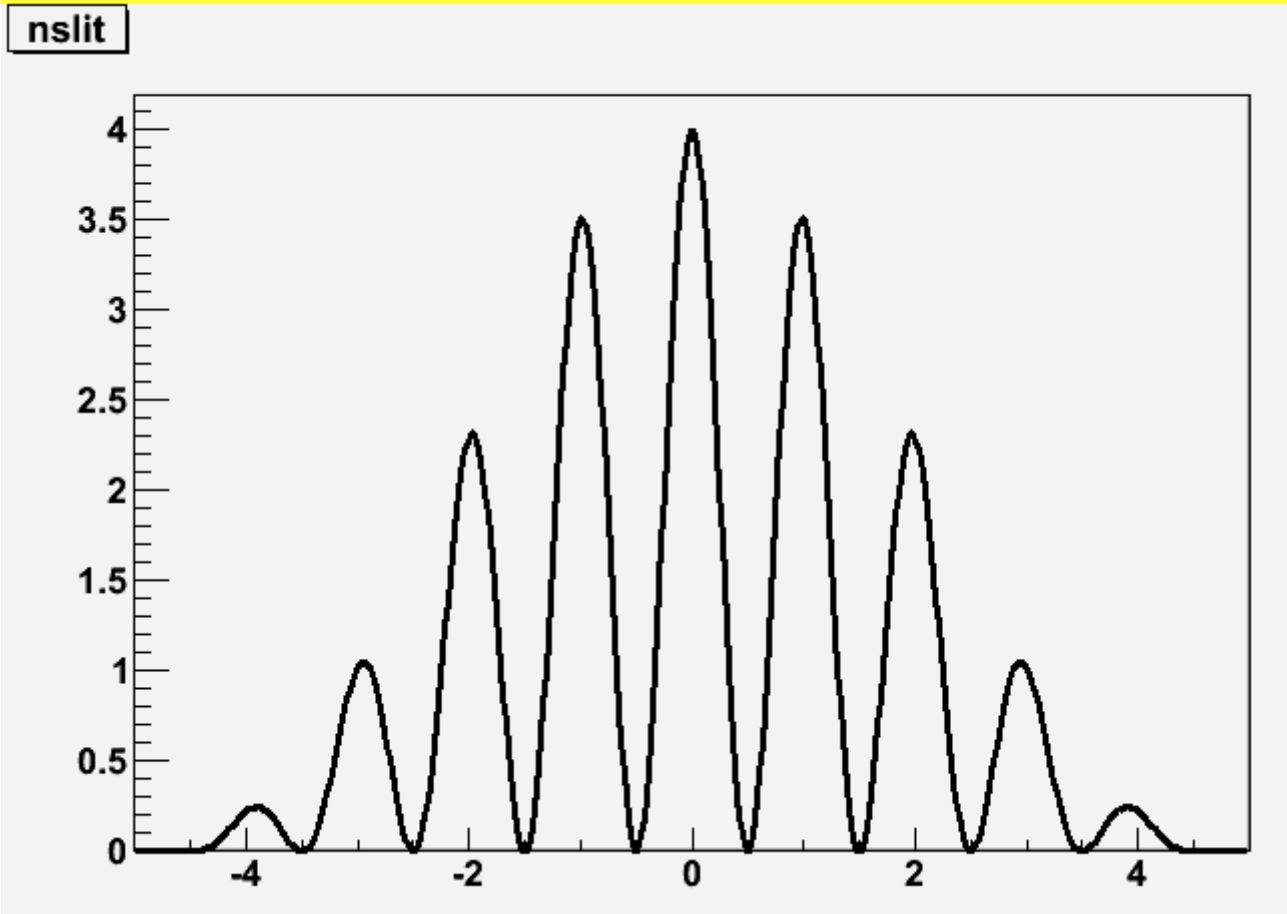
ROOT as Function Plotter [Continue]

```
root [13] TF1 *f1 = new TF1("f2","[0]*sin([1]*x)/x",0.,10.);
root [14] f1->SetParameter(0,1);
root [15] f1->SetParameter(1,1);
root [16] f1->Draw();
```

root [] f1->SetParameter(1,5);
root [] f1->Draw()



写成代码运行



代码名： slits.C

siguang@pku.edu.cn

slits.C

```
// Example drawing the interference pattern of light
// falling on a grid with n slits and ratio r of slit
// width over distance between slits.

// function code in C
double single(double *x, double *par) {
  double const pi=4*atan(1.);
  return pow(sin(pi*par[0]*x[0])/(pi*par[0]*x[0]),2);
}

double nslit0(double *x,double *par){
  double const pi=4*atan(1.);
  return pow(sin(pi*par[1]*x[0])/sin(pi*x[0]),2);
}

double nslit(double *x, double *par){
  return single(x,par) * nslit0(x,par);
}

// This is the main program
void slits() {
  float r,ns;
  // request user input
  cout << "slit width: r=0.2? ";
  scanf("%f",&r);
  cout << "# of slits ns=2?";
  scanf("%f",&ns);
  cout <<"interference pattern for "<< ns
       <<" slits, width/distance: "<<r<<endl;

  // define function and set options
  TF1 *Fnslit = new TF1("Fnslit",nslit,-5.001,5.,2);
  Fnslit->SetNpx(500);
```

slits.C [continue]

```
  // set parameters, as read in above
  Fnslit->SetParameter(0,r);
  Fnslit->SetParameter(1,ns);

  // draw the interference pattern for a grid with n slits
  Fnslit->Draw();
}
```

练习：

1）在上面画箭头然后保存成c1.C文件；保存成.eps 等文件

2）增加坐标轴的标题

Controlling ROOT

在root[]输入如下命令看看啥反应？

.q

.?

.!ls

.!pwd

.x slits.C　要在当前目录下有这个文件奥

.L slits.C

slits()　记不全打tab键看看

要想编译运行，slits.C加头文件

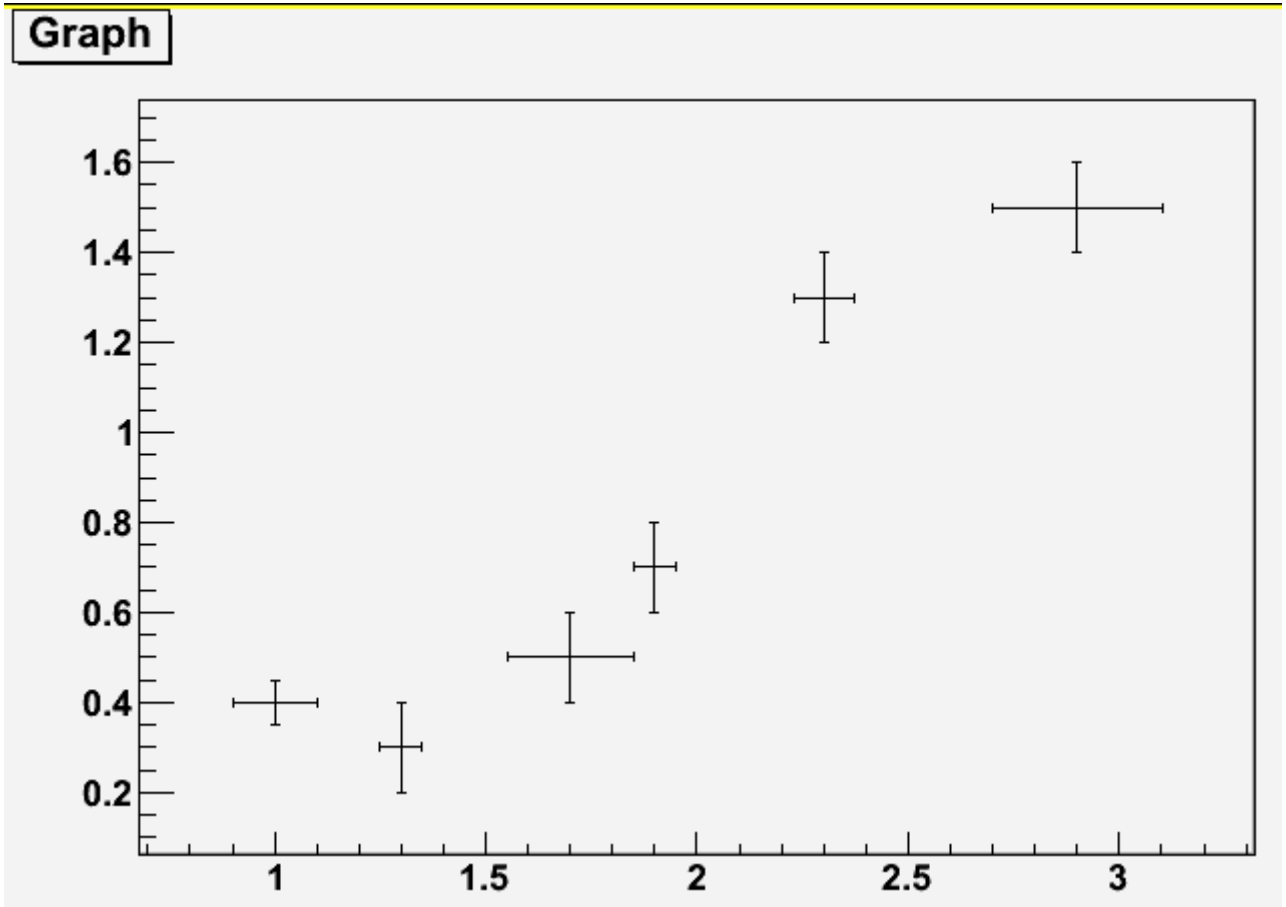
```
#include <iostream>
#include <TF1.h>

using namespace std;
```

后

.L slits.C++

slits()

Plotting Measurements




```
root [0] TGraphErrors *gr=new TGraphErrors("ExampleData.txt")
root [1] gr->Draw("AP")
```

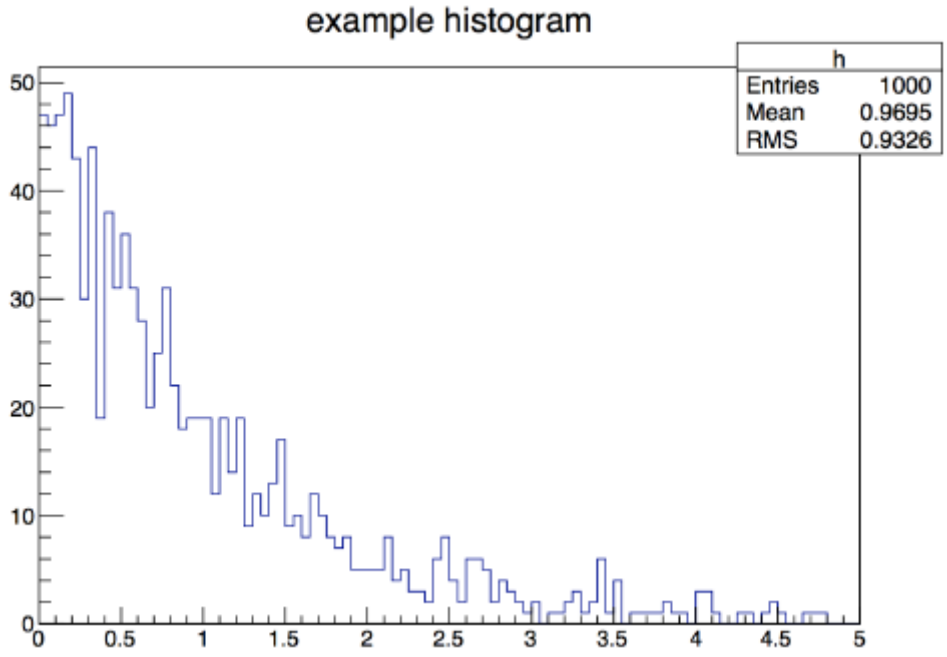
ExampleData.txt

[] cat ExampleData.txt
fake data to demonstrate the use of TGraphErrors

x y ex ey
1. 0.4 0.1 0.05
1.3 0.3 0.05 0.1
1.7 0.5 0.15 0.1
1.9 0.7 0.05 0.1
2.3 1.3 0.07 0.1
2.9 1.5 0.2 0.1

Histograms in ROOT

```
root [0] TF1 efunc("efunc","exp([0]+[1]*x)",0.,5.);
root [1] efunc.SetParameter(0,1);
root [2] efunc.SetParameter(1,-1);
root [3] TH1F* h=new TH1F("h","example histogram",100,0.,5.);
root [4] for (int i=0;i<1000;i++) {h->Fill(efunc.GetRandom());}
root [5] h->Draw();
```



读文件填充

```
root [1] TH1F* h=new TH1F("h","example histogram",100,0.,5.);
root [2] ifstream inp; double x;
root [3] inp.open("expo.dat");
root [4] while (inp >> x) { h->Fill(x); }
root [5] h->Draw();
root [6] inp.close();
```

expo.data
为每行一个数的文本文件

rootlogon.C

```
[hepfarm02]  cat $ROOTSYS/tutorials/rootlogon.C
{
   printf("\nWelcome to the ROOT tutorials\n\n");
   printf("\nType \".x demos.C\" to get a toolbar from which to execute the demos\n");
   printf("\nType \".x demoshelp.C\" to see the help window\n\n");
   printf("==> Many tutorials use the file hsimple.root produced by hsimple.C\n");
   printf("==> It is recommended to execute hsimple.C before any other script\n\n");
}
```

rootlogon.C

```
// This is the file rootlogon.C
{
   TStyle *myStyle  = new TStyle("MyStyle","My Root Styles");

   // from ROOT plain style
   myStyle->SetCanvasBorderMode(0);
   myStyle->SetPadBorderMode(0);
   myStyle->SetPadColor(0);
   myStyle->SetCanvasColor(0);
   myStyle->SetTitleColor(1);
   myStyle->SetStatColor(0);

   myStyle->SetLabelSize(0.03,"xyz"); // size of axis values

   // default canvas positioning
   myStyle->SetCanvasDefX(900);
   myStyle->SetCanvasDefY(20);
   myStyle->SetCanvasDefH(550);
   myStyle->SetCanvasDefW(540);

   myStyle->SetPadBottomMargin(0.1);
   myStyle->SetPadTopMargin(0.1);
   myStyle->SetPadLeftMargin(0.1);
   myStyle->SetPadRightMargin(0.1);
   myStyle->SetPadTickX(1);
   myStyle->SetPadTickY(1);
   myStyle->SetFrameBorderMode(0);

   // Din letter
   myStyle->SetPaperSize(21, 28);
```

rootlogon.C [continue]

```
    myStyle->SetOptStat(111111);// Show overflow and underflow as well
    myStyle->SetOptFit(1011);
    myStyle->SetPalette(1);

    // apply the new style
    gROOT->SetStyle("MyStyle"); //uncomment to set this style
    gROOT->ForceStyle(); // use this style, not the one saved in root files

    printf("\n Beginning new ROOT session with private TStyle \n");

}
```

.rootrc

实现方式，按顺序检查：

- .rootrc //local directory
- $HOME/.rootrc //user directory
- $ROOTSYS/etc/system.rootrc //global ROOT directory

内有

```
# Rint (interactive ROOT executable) specific alias, logon and logoff macros.
Rint.Load:              rootalias.C
Rint.Logon:             rootlogon.C
Rint.Logoff:            rootlogoff.C
```

ROOT command history

```
cat ~/.root_hist
```

俺喜欢的

```
#include <TStyle.h>

// Set the general style options
void SetSgStyle(){
  // No Canvas Border
  gStyle->SetCanvasBorderMode(0);
  gStyle->SetCanvasBorderSize(0);
  // White BG
  gStyle->SetCanvasColor(10);
  // Format for axes
  gStyle->SetLabelFont(22,"xyz");
  gStyle->SetLabelSize(0.06,"xyz");
  gStyle->SetLabelOffset(0.01,"xyz");
  gStyle->SetNdivisions(510,"xyz");
  gStyle->SetTitleFont(22,"xyz");
  gStyle->SetTitleColor(1,"xyz");
  gStyle->SetTitleSize(0.06,"xyz");
  gStyle->SetTitleOffset(0.91);
  gStyle->SetTitleYOffset(1.1);
  // No pad borders
  gStyle->SetPadBorderMode(0);
  gStyle->SetPadBorderSize(0);
  // White BG
  gStyle->SetPadColor(10);
  // Margins for labels etc.
  gStyle->SetPadLeftMargin(0.15);
  gStyle->SetPadBottomMargin(0.15);
  gStyle->SetPadRightMargin(0.05);
  gStyle->SetPadTopMargin(0.06);
  // No error bars in x direction
  gStyle->SetErrorX(0);
  // Format legend
  gStyle->SetLegendBorderSize(0);
  // gStyle->SetLegendFont(22); not In root5.28
  gStyle->SetFillStyle(0);
}
```

用法：

1)在一个文件中例如useful.h敲入左边代码

2) emacs testStyle.C &

```
#include "useful.h"
void testStyle(){
  TH1F *h1 = new TH1F("h1","",100,-10,10);
  SetSgStyle();
  TH1F *h2 = new TH1F("h2","",100,-10,10);
  h1->FillRandom("gaus",1000);
  h2->FillRandom("gaus",1000);
  TCanvas *c1 = new TCanvas("c1","");
  c1->Divide(2,1);
  c1->cd(1);
  h1->Draw();
  c1->cd(2);
  h2->Draw();
}
```

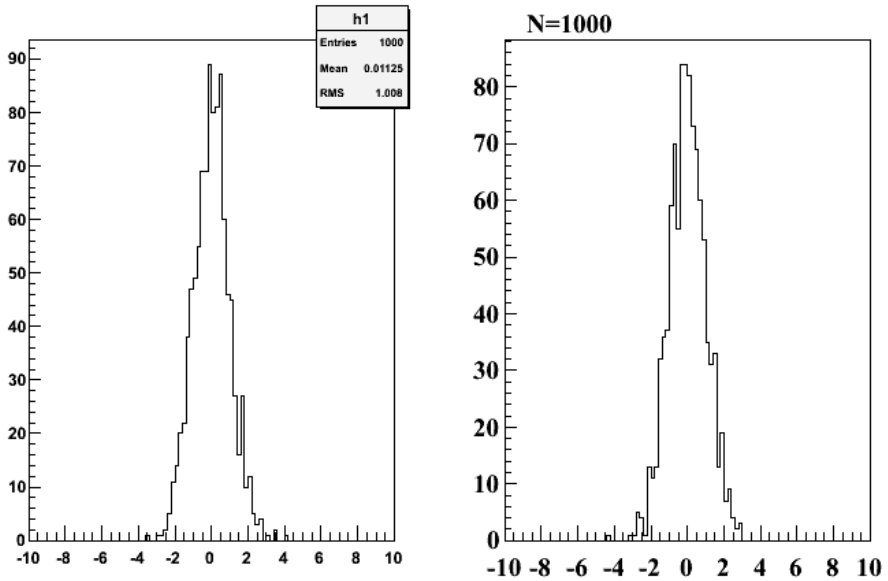
txtN

在useful.h中加入：

```
void txtN(Double_t x0, Double_t y0,TH1 *h,Char_t sName[]="N=%.0f",Double_t sizeTxt=0.06){
  h->SetStats(kFALSE);
  TLatex *ltx = new TLatex();
  ltx->SetNDC(kTRUE);
  ltx->SetTextColor(h->GetLineColor());
  ltx -> SetTextFont(22);
  ltx->SetTextSize(sizeTxt);
  ltx->DrawLatex(x0,y0,Form(sName,h->GetEntries()));
  gPad->Modified();
  gPad->Update();
}
```

在testStyle.C中加入

```
txtN(0.2,0.95,h2);
```



siguang@pku.edu.cn

newTH1F

```
TH1F * newTH1F(Char_t name[]="h1",Double_t binw=0.01, Double_t LowBin=0.0, Double_t HighBin=3.0,Bo
_t MevTitle = kTRUE,Int_t iMode=-1){
  Int_t nbin = TMath::Nint( (HighBin - LowBin)/binw );
  HighBin = binw*nbin + LowBin;

  TH1F *h = new TH1F(name,"",nbin,LowBin,HighBin);
  if(MevTitle) h->GetYaxis()->SetTitle(Form("Events / (%.0fMeV/c^{2})",h->GetBinWidth(1)*1000));
  h->SetMinimum(0.0);
  h->GetYaxis()->SetTitleOffset(1.1);
  if(iMode>=0 && iMode<14){
    Int_t iMarker[] = {20,21,24,25,28,29,30,27,3,  5,2, 26,22,23};
    Int_t iColor[]  = { 2, 4, 6, 9, 1,50,40,31,41,35,44,38,47,12};
    h ->SetMarkerStyle(iMarker[iMode]);
    h ->SetMarkerColor(iColor[iMode]);
    h ->SetLineColor(iColor[iMode]);
  }

  return h;
}
```

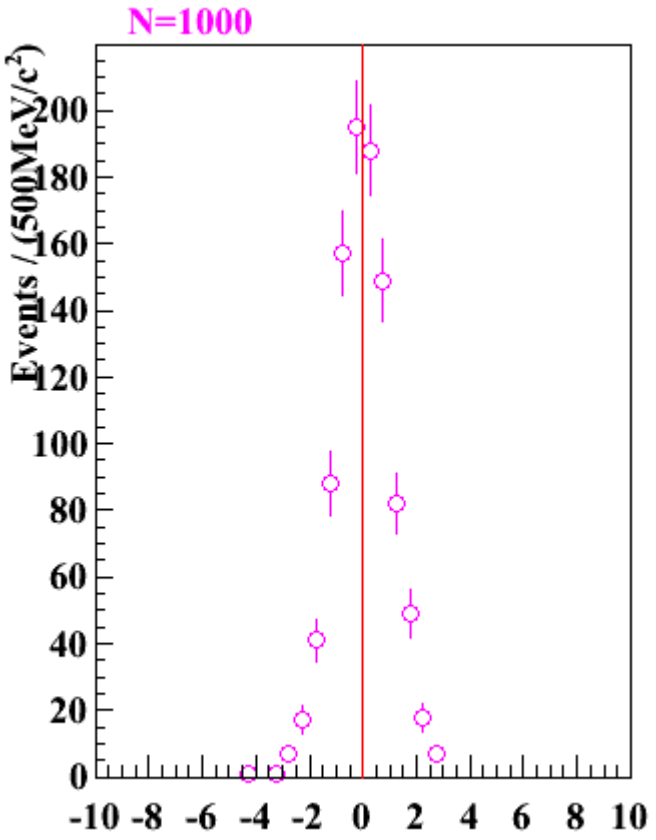
```
void testStyle2(){
  TH1F *h1 = newTH1F("h1",0.5,-10,10,kTRUE,1);
  SetSgStyle();
  TH1F *h2 = newTH1F("h2",0.5,-10,10,kTRUE,2);
  h1->FillRandom("gaus",1000);
  h2->FillRandom("gaus",1000);
  TCanvas *c1 = new TCanvas("c1","");
  c1->Divide(2,1);
  c1->cd(1);
  h1->Draw("EP");
  c1->cd(2);
  h2->Draw("EP");
  txtN(0.2,0.95,h2);
}
```


siguang@pku.edu.cn

LineX1

```
void LineX1(Double_t atX,Int_t iColor=kRed,Int_t iStyle=1,Double_t iWidth=1){
  gPad->Modified();
  gPad->Update();
  TLine *l1 = new TLine(atX,gPad->GetUymin(),atX,gPad->GetUymax());
  l1->SetLineColor(iColor);
  l1->SetLineStyle(iStyle);
  l1->SetLineWidth(iWidth);
  l1->Draw();
}
```

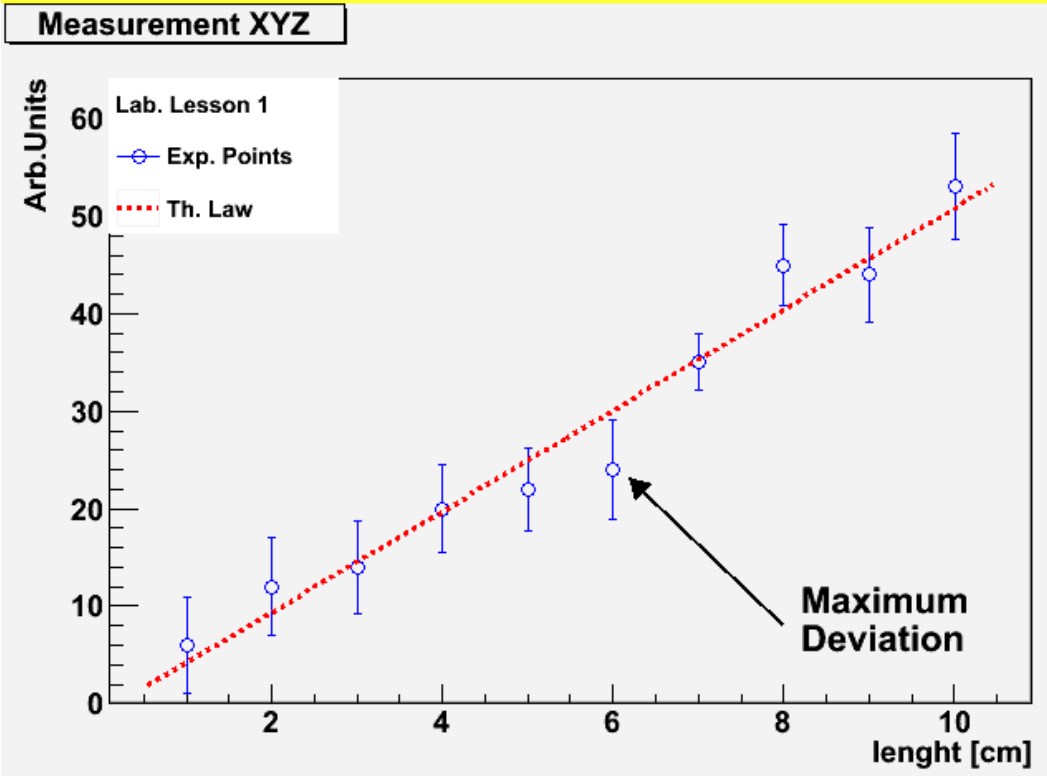
```
void testStyle2(){
  TH1F *h1 = newTH1F("h1",0.5,-10,10,kTRUE,1);
  SetSgStyle();
  TH1F *h2 = newTH1F("h2",0.5,-10,10,kTRUE,2);
  h1->FillRandom("gaus",1000);
  h2->FillRandom("gaus",1000);
  TCanvas *c1 = new TCanvas("c1","");
  c1->Divide(2,1);
  c1->cd(1);
  h1->Draw("EP");
  c1->cd(2);
  h2->Draw("EP");
  txtN(0.2,0.95,h2);
  LineX1(0.0);
}
```



避免多次引用保护

```
#ifndef USEFUL_H
#define USEFUL_H
#include <TStyle.h>

// Set the general style options
void SetSgStyle(){
  // No Canvas Border
  gStyle->SetCanvasBorderMode(0);
  gStyle->SetCanvasBorderSize(0);
  // White BG
  gStyle->SetCanvasColor(10);
  // Format for axes
  gStyle->SetLabelFont(22,"xyz");
  gStyle->SetLabelSize(0.06,"xyz");


...

...

...


#endif
```

A more complete example



siguang@pku.edu.cn

Marco1.C

```
// Builds a graph with errors, displays it and saves it as
// image. First, include some header files (within CINT,
// these will be ignored).

#include "TCanvas.h"
#include "TROOT.h"
#include "TGraphErrors.h"
#include "TF1.h"
#include "TLegend.h"
#include "TArrow.h"
#include "TLatex.h"

void macro1(){
    // The values and the errors on the Y axis
    const int n_points=10;
    double x_vals[n_points]=
            {1,2,3,4,5,6,7,8,9,10};
    double y_vals[n_points]=
            {6,12,14,20,22,24,35,45,44,53};
    double y_errs[n_points]=
            {5,5,4.7,4.5,4.2,5.1,2.9,4.1,4.8,5.43};

    // Instance of the graph
    TGraphErrors graph(n_points,x_vals,y_vals,NULL,y_errs);
    graph.SetTitle("Measurement XYZ;lenght [cm];Arb.Units");

    // Make the plot estetically better
    graph.SetMarkerStyle(kOpenCircle);
    graph.SetMarkerColor(kBlue);
    graph.SetLineColor(kBlue);

    // The canvas on which we'll draw the graph
    TCanvas* mycanvas = new TCanvas();

    // Draw the graph !
    graph.DrawClone("APE");
```

Marco1.C [continue]

```
    // Define a linear function
    TF1 f("Linear law","[0]+x*[1]",.5,10.5);
    // Let's make the funcion line nicer
    f.SetLineColor(kRed); f.SetLineStyle(2);
    // Fit it to the graph and draw it
    graph.Fit(&f);
    f.DrawClone("Same");

    // Build and Draw a legend
    TLegend leg(.1,.7,.3,.9,"Lab. Lesson 1");
    leg.SetFillColor(0);
    graph.SetFillColor(0);
    leg.AddEntry(&graph,"Exp. Points");
    leg.AddEntry(&f,"Th. Law");
    leg.DrawClone("Same");

    // Draw an arrow on the canvas
    TArrow arrow(8,8,6.2,23,0.02,"|>");
    arrow.SetLineWidth(2);
    arrow.DrawClone();

    // Add some text to the plot
    TLatex text(8.2,7.5,"#splitline{Maximum}{Deviation}");
    text.DrawClone();

    mycanvas->Print("graph_with_law.pdf");
}

#ifndef __CINT__
int main(){
    macro1();
    }
#endif
```

Interpretation and Compilation

3.4.1 Compile a Macro with ACLiC (The Automatic Compiler of Libraries for CINT--C/C++ interpreter)

```
root [0] .L macro1.C++
root [1] macro1()
```

3.4.2 Compile a Macro with the Compiler

- **g++ -o macro1.exe macro1.C `root-config --cflags --libs`**
- **./macro1.exe**

- **acroread graph_with_law.pdf &**

ExampleMarco.C

```
/*
 * Note that this file can be either used as a compiled program
    or as a ROOT macro.
 * If it is used as a compiled program, additional include statements
   and the definition of the main program have to be made. This is
   not needed if the code is executed at the ROOT prompt.
                                                                    */

//#ifndef __CINT__       // These include-statements are needed if the program is
#include "TFile.h"      // run as a "stand-alone application", i.e. if it is not
#include "TH1F.h"       // called from an interactive ROOT session.
#include "TCanvas.h"
#include "TMath.h"
// eventually, load some C libraries
#include <math.h>

void ExampleMacro();

//_____________________________________________________________________________
int main()
{
  ExampleMacro();
  return 0;
}
//#endif
```

ExampleMarco.C [continue]

```
//_____________________________________________________________________

/*
   * From here on, the code can also be used as a macro
   * Note though, that CINT may report errors where there are none
     in C++. E.g. this happens here where CINT says that f1 is
     out of scope ...

     ==>>  put your code here
      (remember to update the name of you Macro in the
       lines above if you intend to comile the code)
                                                                    */
void ExampleMacro() {
  // Create a histogram, fill it with random gaussian numbers
  TH1F *h = new TH1F ("h", "example histogram", 100, -5.,5.);
  h->FillRandom("gaus",1000);

  // draw the histogram
  h->DrawClone();

/* - Create a new ROOT file for output
   - Note that this file may contain any kind of ROOT objects, histograms,
     pictures, graphics objects etc.
   - the new file is now becoming the current directory */
  TFile *f1 = new TFile("ExampleMacro.root","RECREATE","ExampleMacro");

  // write Histogram to current directory (i.e. the file just opened)
  h->Write();

  // Close the file.
  //   (You may inspect your histogram in the file using the TBrowser class)
  f1->Close();
}
```

Compile&Run ExampleMarco.C [continue]

```
>g++ -o ExampleMacro.exe ExampleMacro.C `root-config --cflags --libs`
> ./ExampleMacro.exe
Info in <TCanvas::MakeDefCanvas>:  created default TCanvas with name c1
```

ExampleMacro_GUI.C

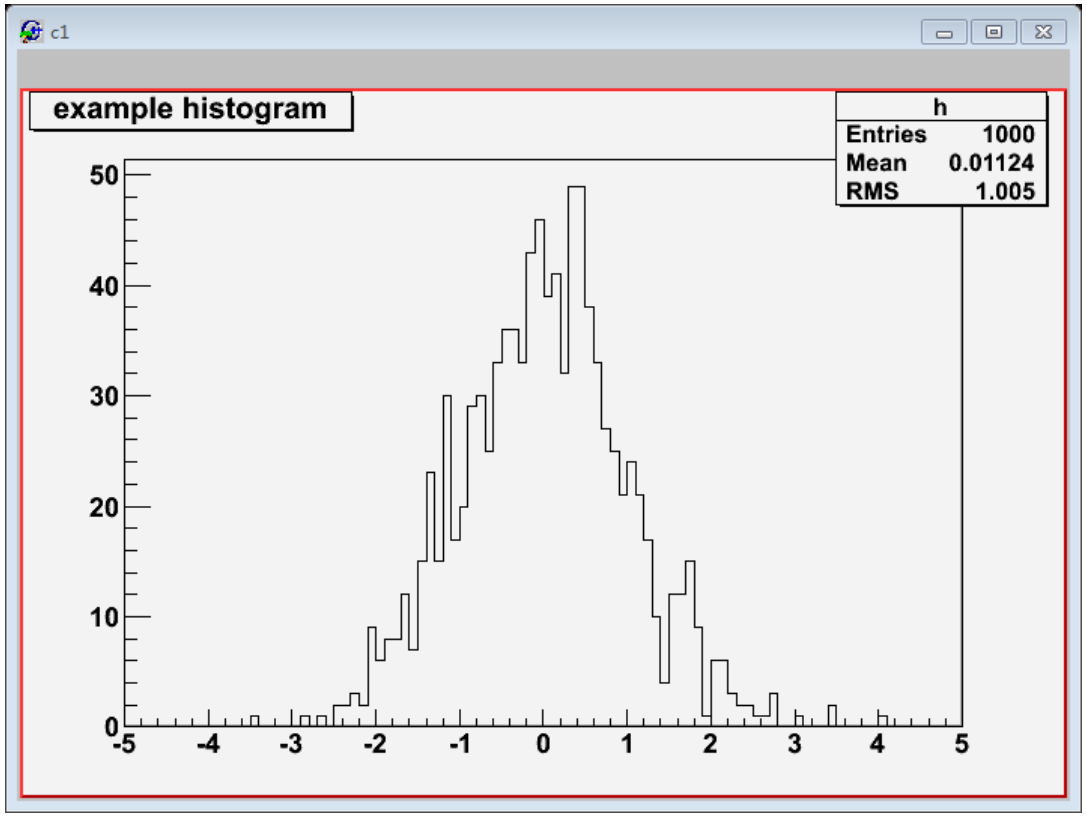
```
/*
 This piece of code demonstrates how a root macro is used as a standalone
    application with full acces the grapical user interface (GUI) of ROOT   */

// include ALL header files needed
#ifndef __CINT__
#include "TROOT.h"
#include "TApplication.h"
#include "TBrowser.h"
#include "TFile.h"
#include "TH1F.h"
#include "TCanvas.h"
#include "TMath.h"
#endif
// eventually, include some additoinal C or C++ libraries
#include <math.h>

//     ==>>  put the code of your macro here
void ExampleMacro_GUI() {
  // Create a histogram, fill it with random gaussian numbers
  TH1F *h = new TH1F ("h", "example histogram", 100, -5.,5.);
  h->FillRandom("gaus",1000);

  // draw the histogram
  h->DrawClone();
```

ExampleMacro_GUI.C [continue]

```
/* - Create a new ROOT file for output
   - Note that this file may contain any kind of ROOT objects, histograms,
     pictures, graphics objects etc.
   - the new file is now becoming the current directory */
  TFile *f1 = new TFile("ExampleMacro.root","RECREATE","ExampleMacro");

  // write Histogram to current directory (i.e. the file just opened)
  h->Write();

  // Close the file.
  //   (You may inspect your histogram in the file using the TBrowser class)
  f1->Close();
}

// the "dressing" code for a stand-alone ROOT application starts here
#ifndef __CINT__
void StandaloneApplication(int argc, char** argv) {
  // ==>> here the ROOT macro is called
  ExampleMacro_GUI();
}

// This is the standard main of C++ starting a ROOT application
int main(int argc, char** argv) {
   TApplication app("Root Application", &argc, argv);
   StandaloneApplication(app.Argc(), app.Argv());
   app.Run();
   return 0;
}
#endif
```

Compile&Run ExampleMarco_GUI.C [continue]

```
>g++ -o ExampleMacro_GUI.exe ExampleMacro_GUI.C `root-config --cflags --libs`
> ./ExampleMacro_GUI.exe
Info in <TCanvas::MakeDefCanvas>:  created default TCanvas with name c1
```



Read Graph Points from File

```
TGraphErrors(const char *filename,
const char *format="%lg %lg %lg %lg", Option_t *option="");
```

The format string can be:

- "\%lg \%lg" read only 2 first columns into X,Y
- "\%lg \%lg \%lg" read only 3 first columns into X,Y and EY
- "\%lg \%lg \%lg \%lg" read only 4 first columns into X,Y,EX,EY

macro2.C

macro2_input.txt

```
# Measurement of Friday 26 March
# Experiment 2 Physics Lab

1   6   5
2   12  5
3   14  4.7
4   20  4.5
5   22  4.2
6   24  5.1
7   35  2.9
8   45  4.1
9   44  4.8
10  53  5.43
```

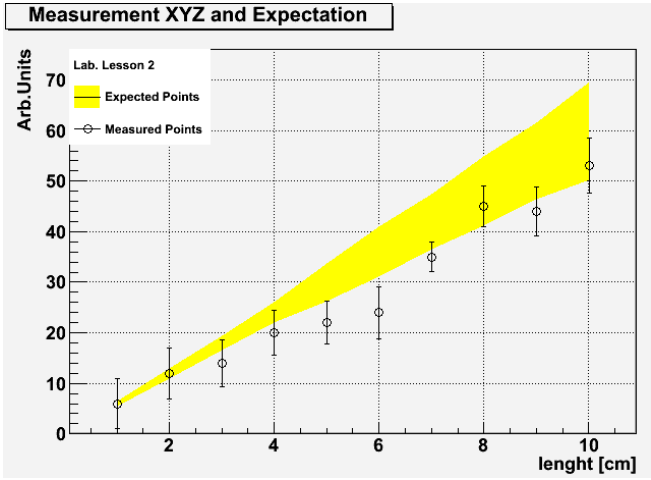
macro2_input_expected.txt

```
# Measurement of Friday 26 March
# Experiment 2 Physics Lab
# Expected points from theory predictions

1   6   0.5
2   12  1.
3   18  1.5
4   24  2.0
5   30  3.7
6   36  4.9
7   42  5.4
8   48  6.8
9   54  7.5
10  60  9.7
```

marco2.C

```
// Reads the points from a file and produces a simple graph.
int macro2(){
    TCanvas* c=new TCanvas();
    c->SetGrid();

    TGraphErrors graph_expected("./macro2_input_expected.txt",
                                "%lg %lg %lg");
    graph_expected.SetTitle(
       "Measurement XYZ and Expectation;
        lenght [cm];
        Arb.Units");
    graph_expected.SetFillColor(kYellow);
    graph_expected.DrawClone("E3AL"); // E3 draws the band

    TGraphErrors graph("./macro2_input.txt","%lg %lg %lg");
    graph.SetMarkerStyle(kCircle);
    graph.SetFillColor(0);
    graph.DrawClone("PESame");

    // Draw the Legend
    TLegend leg(.1,.7,.3,.9,"Lab. Lesson 2");
    leg.SetFillColor(0);
    leg.AddEntry(&graph_expected,"Expected Points");
    leg.AddEntry(&graph,"Measured Points");
    leg.DrawClone("Same");

    graph.Print();
}
```

graph.Print()显示数据信息

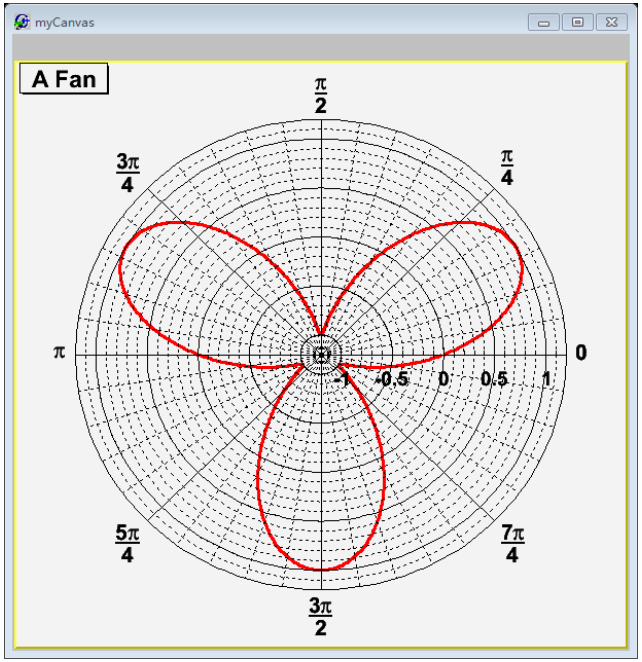


siguang@pku.edu.cn

Polar Graphs

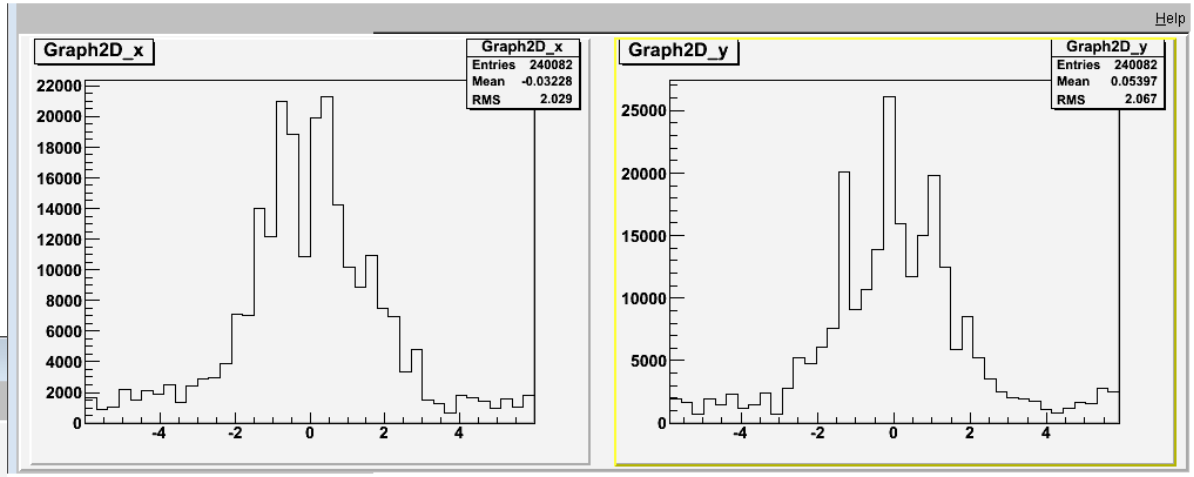
macro3.C

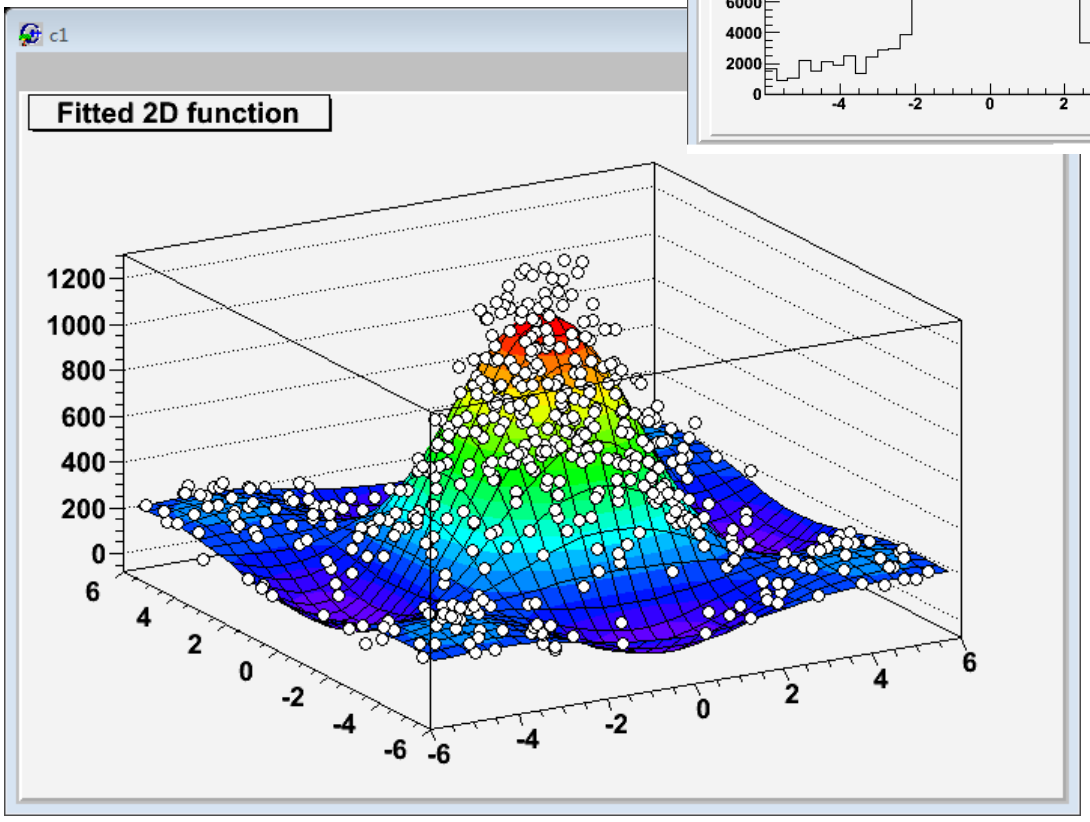
```
// Builds a polar graph in a square Canvas.

void macro3(){
     TCanvas* c = new TCanvas("myCanvas","myCanvas",600,600);
     double rmin=0;
     double rmax=TMath::Pi()*6;
     const int npoints=1000;
     Double_t r[npoints];
     Double_t theta[npoints];
     for (Int_t ipt = 0; ipt < npoints; ipt++) {
          r[ipt] = ipt*(rmax-rmin)/npoints+rmin;
          theta[ipt] = TMath::Sin(r[ipt]);
     }
     TGraphPolar grP1 (npoints,r,theta);
     grP1.SetTitle("A Fan");
     grP1.SetLineWidth(3);
     grP1.SetLineColor(2);
     grP1.DrawClone("AOL");
}
```



2D Graphs

macro4.C





siguang@pku.edu.cn

macro4.C

```
// Create, Draw and fit a TGraph2DErrors
void macro4(){
   gStyle->SetPalette(1);
   const double e = 0.3;
   const int nd = 500;

   TRandom3 my_random_generator;
   TF2 *f2 = new TF2("f2",
                     "1000*(([0]*sin(x)/x)*([1]*sin(y)/y))+200",
                     -6,6,-6,6);
   f2->SetParameters(1,1);
   TGraph2DErrors *dte = new TGraph2DErrors(nd);
   // Fill the 2D graph
   double rnd, x, y, z, ex, ey, ez;
   for (Int_t i=0; i<nd; i++) {
      f2->GetRandom2(x,y);
      // A random number in [-e,e]
      rnd = my_random_generator.Uniform(-e,e);
      z = f2->Eval(x,y)*(1+rnd);
      dte->SetPoint(i,x,y,z);
      ex = 0.05*my_random_generator.Uniform();
      ey = 0.05*my_random_generator.Uniform();
      ez = TMath::Abs(z*rnd);
      dte->SetPointError(i,ex,ey,ez);
   }
   // Fit function to generated data
   f2->SetParameters(0.7,1.5);  // set initial values for fit
   f2->SetTitle("Fitted 2D function");
   dte->Fit(f2);
```

macro4.C [continue]

```
    // Plot the result
    TCanvas *c1 = new TCanvas();
    f2->Draw("Surf1");
    dte->Draw("P0 Same");
    // Make the x and y projections
    TCanvas* c_p= new TCanvas("ProjCan",
                              "The Projections",1000,400);
    c_p->Divide(2,1);
    c_p->cd(1);
    dte->Project("x")->Draw();
    c_p->cd(2);
    dte->Project("y")->Draw();
}
```

Fit Options

```
void TH1::Fit(const char *fname, Option_t *option, Option_t *goption,
Axis_t xxmin, Axis_t xxmax)
```

- *option:The second parameter is the fitting option. Here is the list of fitting options:
- “W” Set all weights to 1 for non empty bins; ignore error bars
- “WW” Set all weights to 1 including empty bins; ignore error bars
- “I” Use integral of function in bin instead of value at bin center
- “L” Use log likelihood method (default is chi-square method)
- “U” Use a user specified fitting algorithm
- “Q” Quiet mode (minimum printing)
- “V” Verbose mode (default is between Q and V)
- “E” Perform better errors estimation using the Minos technique
- “M” Improve fit results
- “R” Use the range specified in the function range
- “N” Do not store the graphics function, do not draw
- “0” Do not plot the result of the fit. By default the fitted function is drawn unless the option “N” above is specified.

Fit Options

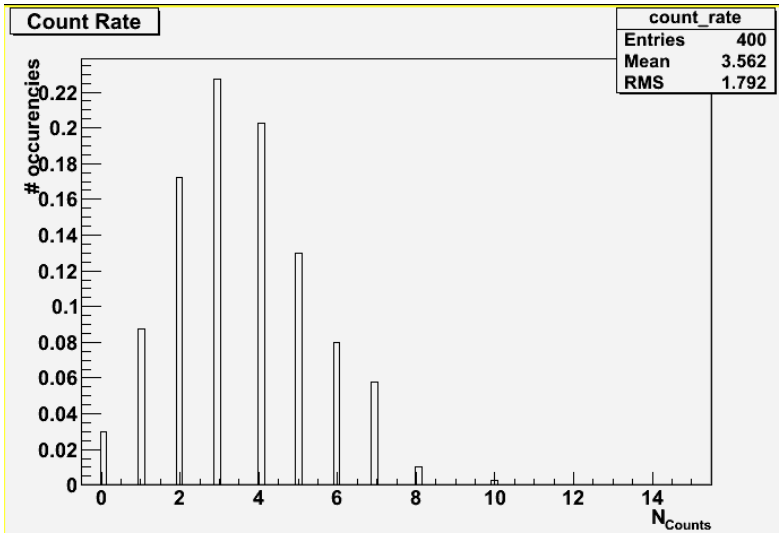
• “+” Add this new fitted function to the list of fitted functions (by default, the previous function is deleted and
only the last one is kept)
• “B”Use this option when you want to fix one or more parameters and the fitting function is like polN, expo,
landau, gaus.
• “LL”An improved Log Likelihood fit in case of very low statistics and when bincontentsare not integers. Do not
use this option if bin contents are large (greater than 100).
• “C”In case of linear fitting, don’t calculate the chisquare (saves time).
• “F”If fitting a polN, switch to Minuit fitter (by default, polN functions are fitted by the linear fitter).

• *goption:The third parameter is the graphics option that is the same as in the **TH1**::Draw (see the chapter Draw Options).
• xxmin, xxmax:Thee fourth and fifth parameters specify the range over which to apply the fit.

a Histogram

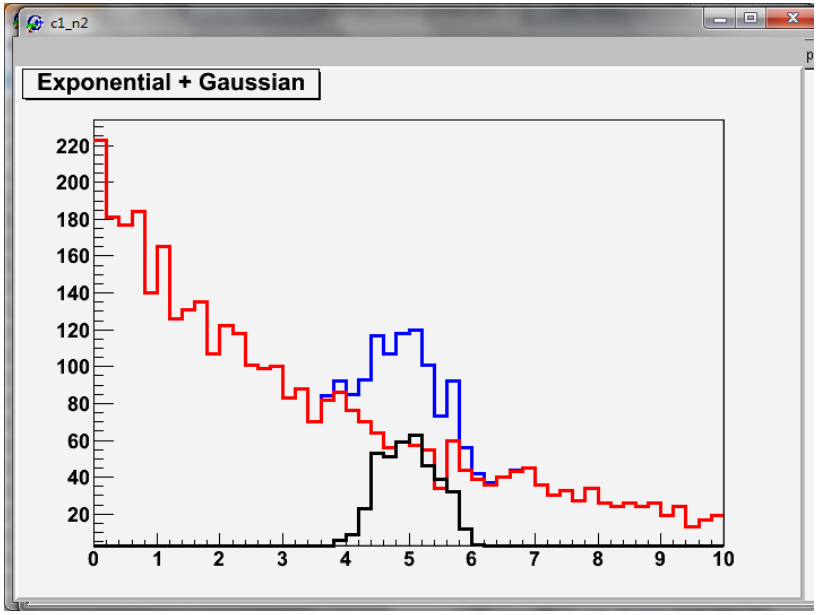
macro.C

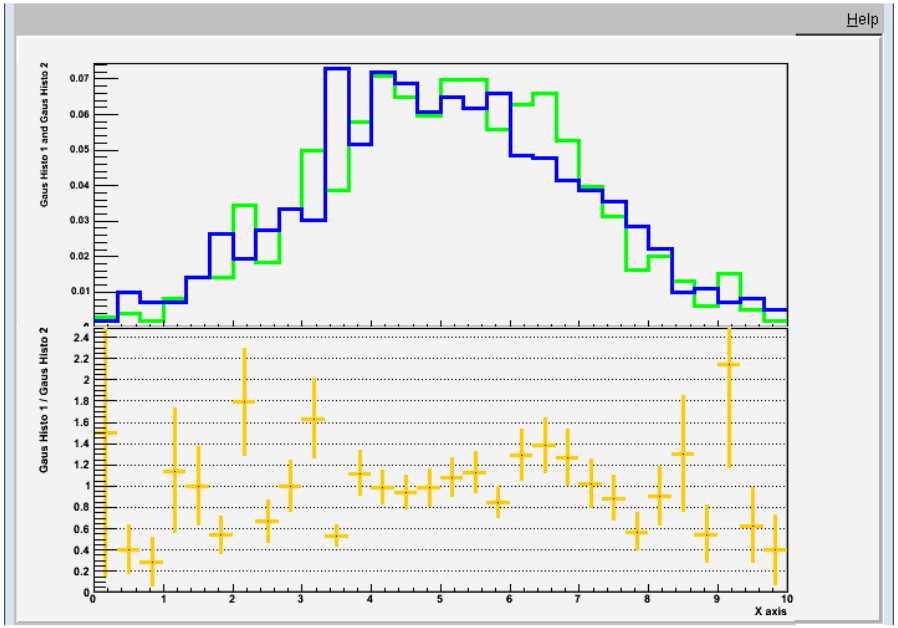
```
// Create, Fill and draw an Histogram which reproduces the
// counts of a scaler linked to a Geiger counter.

void macro5(){
    TH1F* cnt_r_h=new TH1F("count_rate",
                "Count Rate;N_{Counts};# occurencies",
                100, // Number of Bins
                -0.5, // Lower X Boundary
                15.5); // Upper X Boundary

    const float mean_count=3.6;
    TRandom3 rndgen;
    // simulate the measurements
    for (int imeas=0;imeas<400;imeas++)
        cnt_r_h->Fill(rndgen.Poisson(mean_count));

    TCanvas* c= new TCanvas();
    cnt_r_h->Draw();

    TCanvas* c_norm= new TCanvas();
    cnt_r_h->DrawNormalized();

    // Print summary
    cout << "Moments of Distribution:\n"
         << " - Mean = " << cnt_r_h->GetMean() << " +- "
                         << cnt_r_h->GetMeanError() << "\n"
         << " - RMS = " << cnt_r_h->GetRMS() << " +- "
                        << cnt_r_h->GetRMSError() << "\n"
         << " - Skewness = " << cnt_r_h->GetSkewness() << "\n"
         << " - Kurtosis = " << cnt_r_h->GetKurtosis() << "\n";
}
```



Add and Divide Histograms

macro6.C





右上图的y轴可用SetRangeUser(0.001,0.08)改进
消除露出一半的数字

siguang@pku.edu.cn

macro6.C

```
// Divide and add 1D Histograms

void format_h(TH1F* h, int linecolor){
    h->SetLineWidth(3);
    h->SetLineColor(linecolor);
    }

void macro6(){

    TH1F* sig_h=new TH1F("sig_h","Signal Histo",50,0,10);
    TH1F* gaus_h1=new TH1F("gaus_h1","Gauss Histo 1",30,0,10);
    TH1F* gaus_h2=new TH1F("gaus_h2","Gauss Histo 2",30,0,10);
    TH1F* bkg_h=new TH1F("exp_h","Exponential Histo",50,0,10);

    // simulate the measurements
    TRandom3 rndgen;
    for (int imeas=0;imeas<4000;imeas++){
        exp_h->Fill(rndgen.Exp(4));
        if (imeas%4==0) gaus_h1->Fill(rndgen.Gaus(5,2));
        if (imeas%4==0) gaus_h2->Fill(rndgen.Gaus(5,2));
        if (imeas%10==0)sig_h->Fill(rndgen.Gaus(5,.5));}

    // Format Histograms
    TH1F* histos[4]={sig_h,bkg_h,gaus_h1,gaus_h2};
    for (int i=0;i<4;++i){
        histos[i]->Sumw2(); // *Very* Important
        format_h(histos[i],i+1);
        }
```

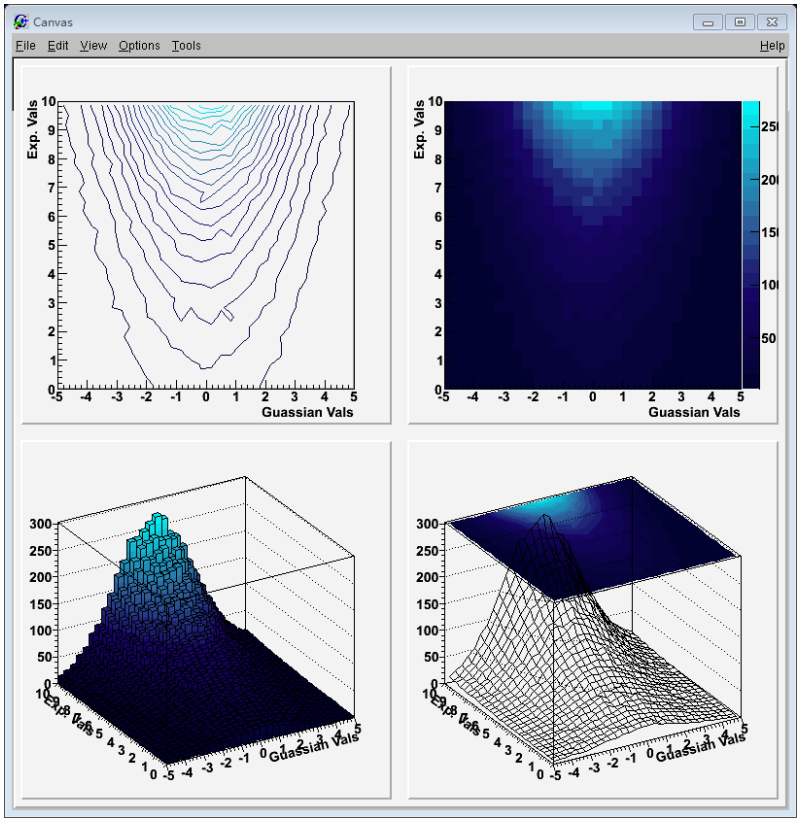

siguang@pku.edu.cn

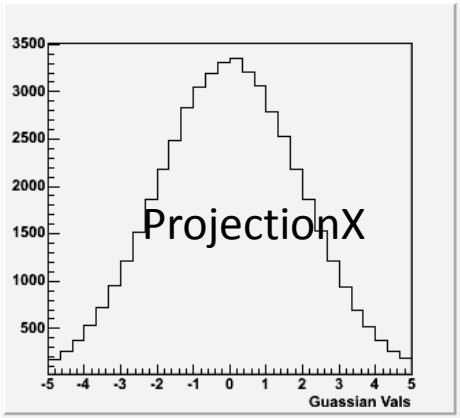
macro6.C

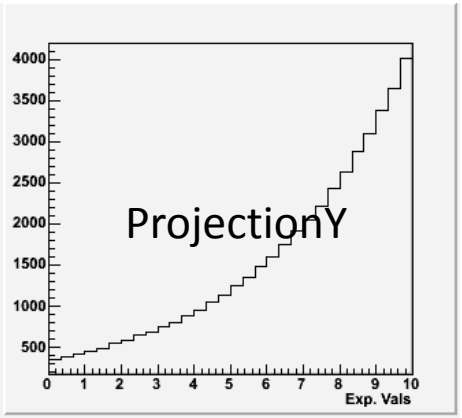
```
// Sum
TH1F* sum_h= new TH1F(*bkg_h);
sum_h->Add(sig_h,1.);
sum_h->SetTitle("Exponential + Gaussian");
format_h(sum_h,kBlue);

TCanvas* c_sum= new TCanvas();
sum_h->Draw("hist");
bkg_h->Draw("SameHist");
sig_h->Draw("SameHist");

// Divide
TH1F* dividend=new TH1F(*gaus_h1);
dividend->Divide(gaus_h2);

// Graphical Maquillage
dividend->SetTitle(";X axis;Gaus Histo 1 / Gaus Histo 2");
format_h(dividend,kOrange);
gaus_h1->SetTitle(";;Gaus Histo 1 and Gaus Histo 2");
gStyle->SetOptStat(0);

TCanvas* c_divide= new TCanvas();
c_divide->Divide(1,2,0,0);
c_divide->cd(1);
c_divide->GetPad(1)->SetRightMargin(.01);
gaus_h1->DrawNormalized("Hist");
gaus_h2->DrawNormalized("HistSame");
c_divide->cd(2);
dividend->GetYaxis()->SetRangeUser(0,2.49);
c_divide->GetPad(2)->SetGridy();
c_divide->GetPad(2)->SetRightMargin(.01);
dividend->Draw();
}
```

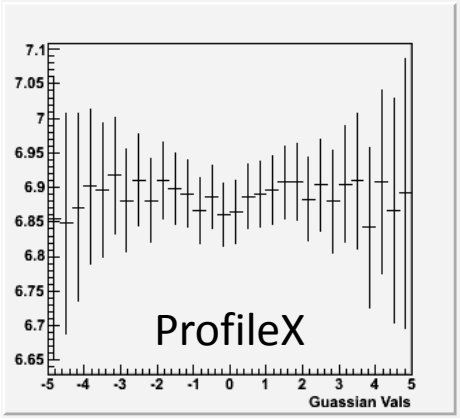
Two-dimensional Histograms

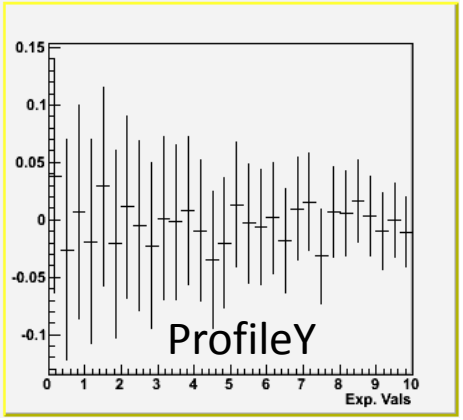
Output of macro7.C











siguang@pku.edu.cn

macro7.C

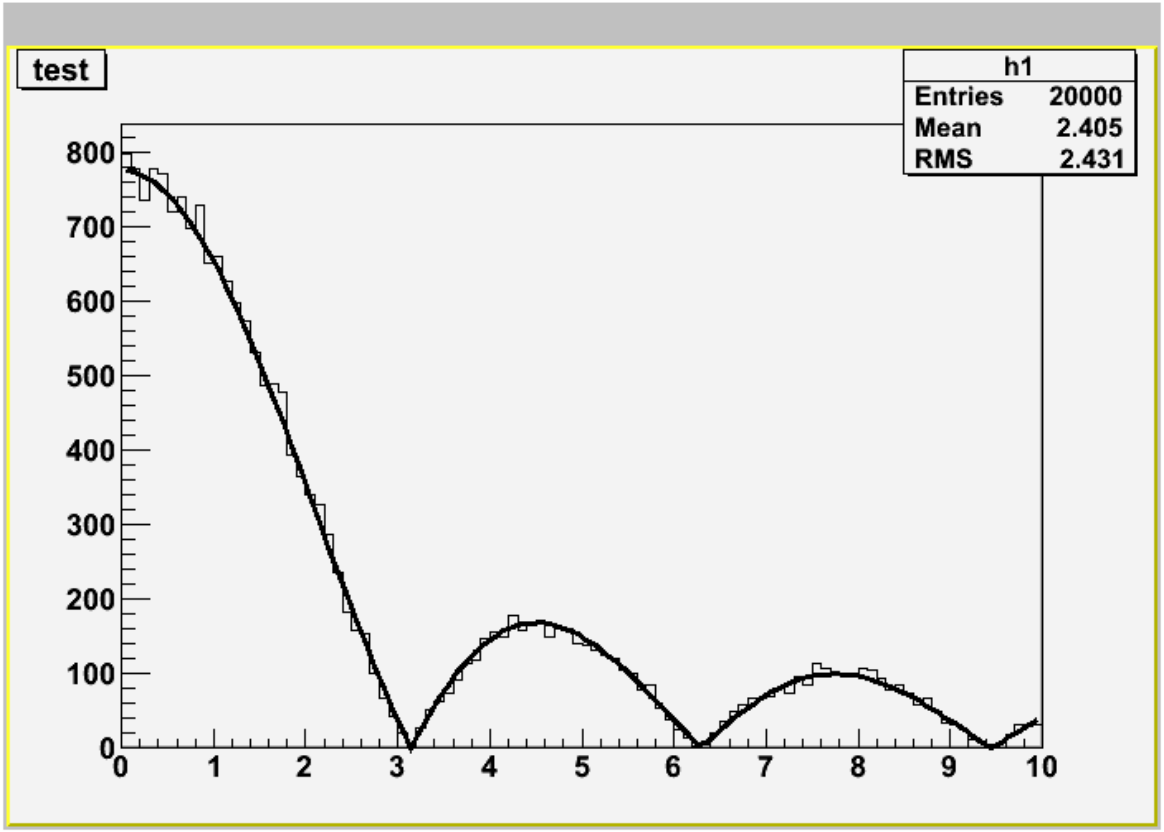
```
// Draw a Bidimensional Histogram in many ways
// together with its profiles and projections

void macro7(){
    gStyle->SetPalette(53);
    gStyle->SetOptStat(0);
    gStyle->SetOptTitle(0);

    TH2F bidi_h("bidi_h","2D Histo;Guassian Vals;Exp. Vals",
                30,-5,5,  // X axis
                30,0,10); // Y axis

    TRandom3 rgen;
    for (int i=0;i<500000;i++)
        bidi_h.Fill(rgen.Gaus(0,2),10-rgen.Exp(4),.1);

    TCanvas* c=new TCanvas("Canvas","Canvas",800,800);
    c->Divide(2,2);
    c->cd(1);bidi_h.DrawClone("Cont1");
    c->cd(2);bidi_h.DrawClone("Colz");
    c->cd(3);bidi_h.DrawClone("lego2");
    c->cd(4);bidi_h.DrawClone("surf3");

    // Profiles and Projections
    TCanvas* c2=new TCanvas("Canvas2","Canvas2",800,800);
    c2->Divide(2,2);
    c2->cd(1);bidi_h.ProjectionX()->DrawClone();
    c2->cd(2);bidi_h.ProjectionY()->DrawClone();
    c2->cd(3);bidi_h.ProfileX()->DrawClone();
    c2->cd(4);bidi_h.ProfileY()->DrawClone();
}
```

自定义函数 (Cmyfit.C)

```
#include <TF1.h>
#include <TH1F.h>
#include <TROOT.h>

// Macro myfunc.C
   Double_t myfunction(Double_t *x, Double_t *par)
   {
      Float_t xx =x[0];
      Double_t f = TMath::Abs(par[0]*sin(par[1]*xx)/xx);
      return f;
   }
   void myfunc()
   {
      TF1 *f1 = new TF1("myfunc",myfunction,0,10,2);
      f1->SetParameters(2,1);
      f1->SetParNames("constant","coefficient");
      f1->Draw();
   }
   void myfit()
   {
      TH1F *h1=new TH1F("h1","test",100,0,10);
      h1->FillRandom("myfunc",20000);
      TF1 *f1=(TF1 *)gROOT->GetFunction("myfunc");
      f1->SetParameters(800,1);
      h1->Fit("myfunc");
   }
```

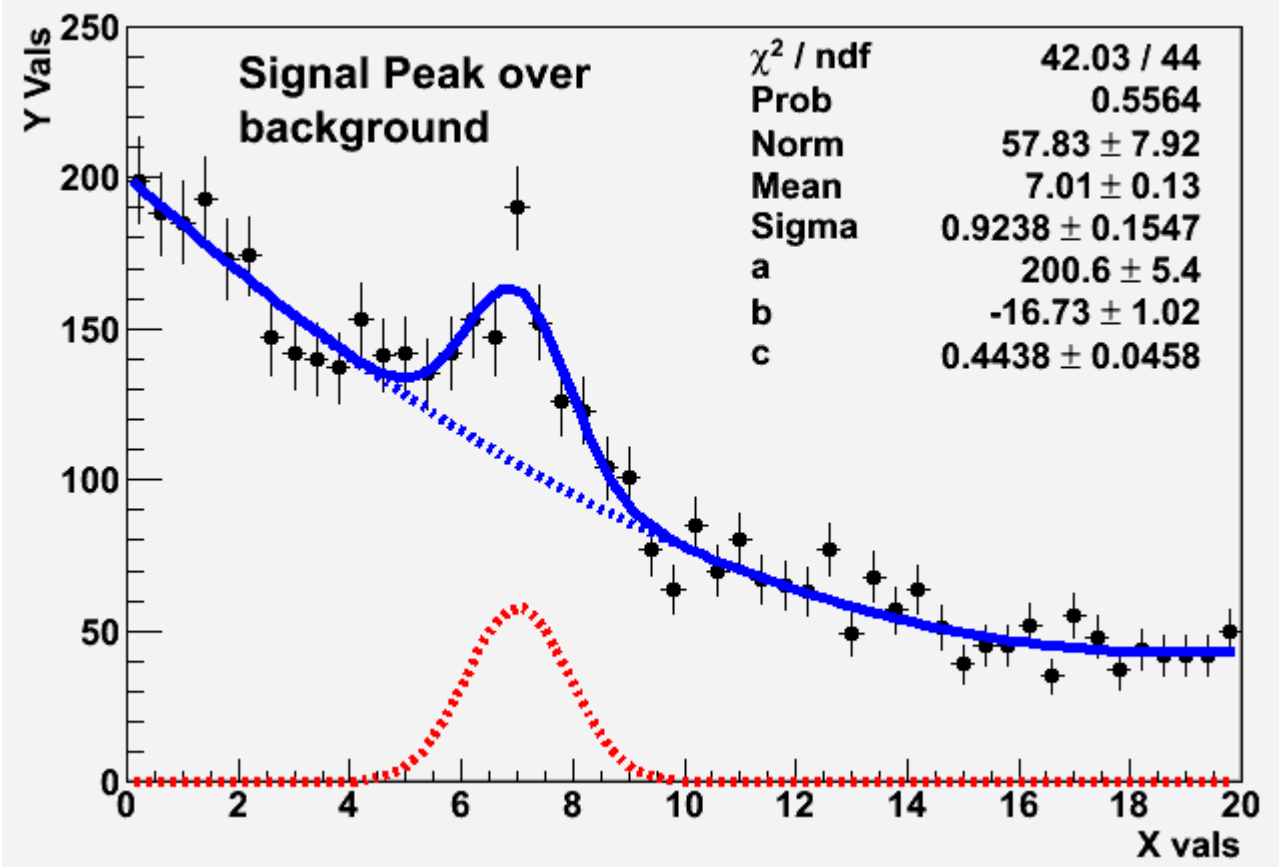
run

root [1].L Cmyfit.C
root [1] myfunc()
root [2] myfit()



Functions and Parameter Estimation

Output of macro8.C



siguang@pku.edu.cn

macro8.C

```
void format_line(TAttLine* line,int col,int sty){
    line->SetLineWidth(5); line->SetLineColor(col);
    line->SetLineStyle(sty);}

double the_gausppar(double* vars, double* pars){
    return pars[0]*TMath::Gaus(vars[0],pars[1],pars[2])+
        pars[3]+pars[4]*vars[0]+pars[5]*vars[0]*vars[0];}

int macro8(){
    gStyle->SetOptTitle(0); gStyle->SetOptStat(0);
    gStyle->SetOptFit(1111); gStyle->SetStatBorderSize(0);
    gStyle->SetStatX(.89); gStyle->SetStatY(.89);

    TF1 parabola("parabola","[0]+[1]*x+[2]*x**2",0,20);
    format_line(&parabola,kBlue,2);

    TF1 gaussian("gaussian","[0]*TMath::Gaus(x,[1],[2])",0,20);
    format_line(&gaussian,kRed,2);
```

macro8.C [continue]

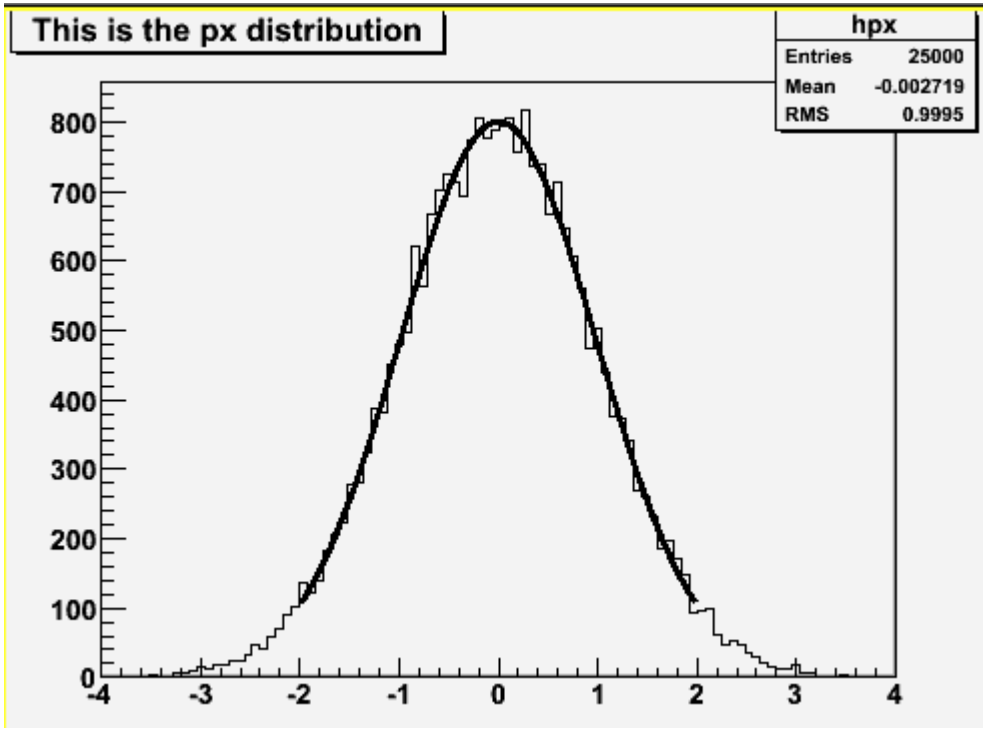
```
TF1 gausppar("gausppar",the_gausppar,-0,20,6);
double a=15; double b=-1.2; double c=.03;
double norm=4; double mean=7; double sigma=1;
gausppar.SetParameters(norm,mean,sigma,a,b,c);
gausppar.SetParNames("Norm","Mean","Sigma","a","b","c");
format_line(&gausppar,kBlue,1);

TH1F histo("histo","Signal plus background;X vals;Y Vals",
           50,0,20);
histo.SetMarkerStyle(8);

// Fake the data
for (int i=1;i<=5000;++i) histo.Fill(gausppar.GetRandom());

// Reset the parameters before the fit and set
// by eye a peak at 6 with an area of more or less 50
gausppar.SetParameter(0,50);
gausppar.SetParameter(1,6);
int npar=gausppar.GetNpar();
for (int ipar=2;ipar<npar;++ipar)
    gausppar.SetParameter(ipar,1);

// perform fit ...
TFitResultPtr frp = histo.Fit(&gausppar, "S");

// ... and retrieve fit results
frp->Print(); // print fit results
// get covariance Matrix an print it
TMatrixDSym covMatrix (frp->GetCovarianceMatrix());
covMatrix.Print();
```

macro8.C [continue]

```
    // Set the values of the gaussian and parabola
    for (int ipar=0;ipar<3;ipar++){
        gaussian.SetParameter(ipar,
                              gausppar.GetParameter(ipar));
        parabola.SetParameter(ipar,
                              gausppar.GetParameter(ipar+3));}

    histo.GetYaxis()->SetRangeUser(0,250);
    histo.DrawClone("PE");
    parabola.DrawClone("Same"); gaussian.DrawClone("Same");
    TLatex latex(2,220,
                 "#splitline{Signal Peak over}{background}");
    latex.DrawClone("Same");
}
```

用户自定义函数

$ROOTSYS/tutorials/fit/myfit.C



siguang@pku.edu.cn

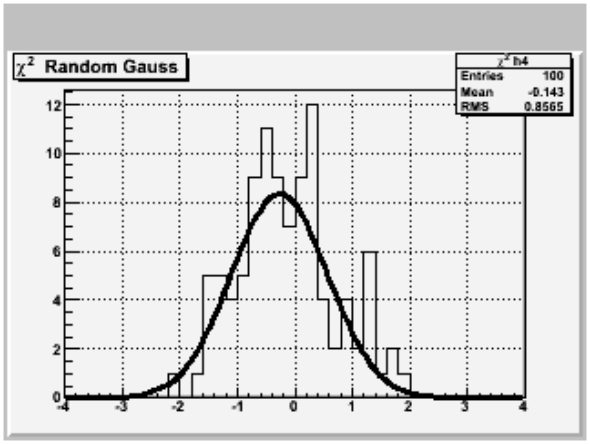
myfit.C

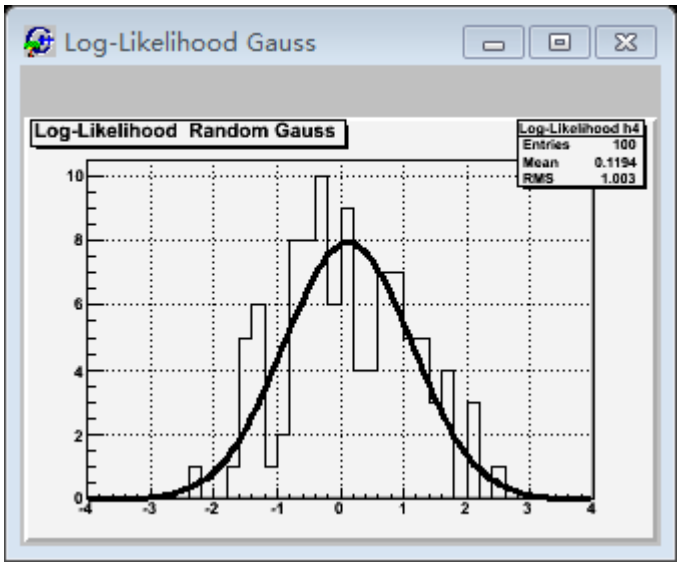
```
// Get in memory an histogram from a root file and fit a user defined function.
// Note that a user defined function must always be defined
// as in this example:
//  - first parameter: array of variables (in this example only 1-dimension)
//  - second parameter: array of parameters
// Note also that in case of user defined functions, one must set
// an initial value for each parameter.
//Author: Rene Brun

Double_t fitf(Double_t *x, Double_t *par)
{
   Double_t arg = 0;
   if (par[2] != 0) arg = (x[0] - par[1])/par[2];

   Double_t fitval = par[0]*TMath::Exp(-0.5*arg*arg);
   return fitval;
}
void myfit()
{
   TString dir = gSystem->UnixPathName(gInterpreter->GetCurrentMacroName());
   dir.ReplaceAll("myfit.C","../hsimple.C");
   dir.ReplaceAll("/./","/");
   if (!gInterpreter->IsLoaded(dir.Data())) gInterpreter->LoadMacro(dir.Data());

   TFile *hsimple = (TFile*)gROOT->ProcessLineFast("hsimple(1)");
   if (!hsimple) return;

   TCanvas *c1 = new TCanvas("c1","the fit canvas",500,400);

   TH1F *hpx = (TH1F*)hsimple->Get("hpx");
```

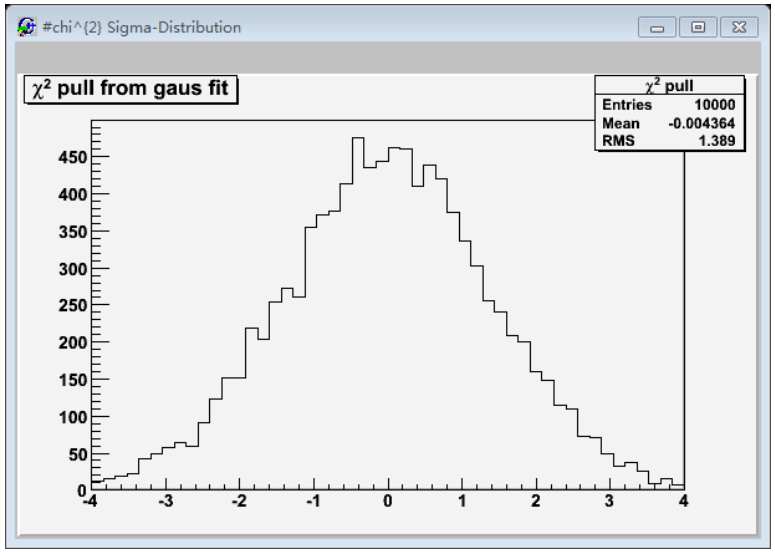
myfit.C [continue]

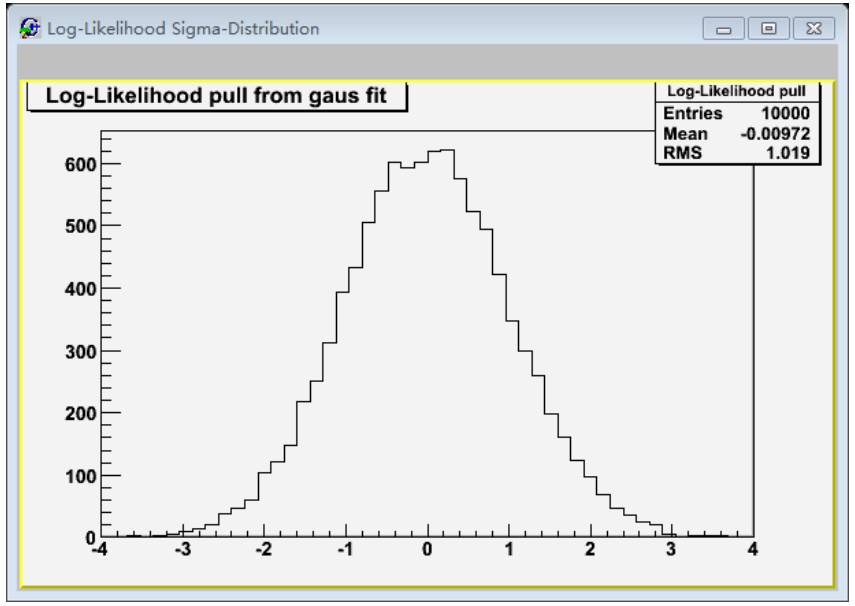
```
// Creates a Root function based on function fitf above
   TF1 *func = new TF1("fitf",fitf,-2,2,3);

// Sets initial values and parameter names
   func->SetParameters(100,0,1);
   func->SetParNames("Constant","Mean_value","Sigma");

// Fit histogram in range defined by function
   hpx->Fit(func,"r");

// Gets integral of function between fit limits
   printf("Integral of function = %g\n",func->Integral(-2,2));
}
```

Output of macro9.C









siguang@pku.edu.cn

macro9.C

```
// Toy Monte Carlo example.
// Check pull distribution to compare chi2 and binned
// log-likelihood methods.

pull( int n_toys = 10000,
      int n_tot_entries = 100,
      int nbins = 40,
      bool do_chi2=true ){

  TString method_prefix("Log-Likelihood ");
  if (do_chi2)
    method_prefix="#chi^{2} ";

  // Create histo
  TH1F* h4 = new TH1F(method_prefix+"h4",
                      method_prefix+" Random Gauss",
                      nbins,-4,4);
  h4->SetMarkerStyle(21);
  h4->SetMarkerSize(0.8);
  h4->SetMarkerColor(kRed);

  // Histogram for sigma and pull
  TH1F* sigma = new TH1F(method_prefix+"sigma",
                         method_prefix+"sigma from gaus fit",
                         50,0.5,1.5);
  TH1F* pull = new TH1F(method_prefix+"pull",
                        method_prefix+"pull from gaus fit",
                        50,-4.,4.);

  // Make nice canvases
  TCanvas* c0 = new TCanvas(method_prefix+"Gauss",
                            method_prefix+"Gauss",0,0,320,240);
  c0->SetGrid();
```

macro9.C [continue]

```
// Make nice canvases
TCanvas* c1 = new TCanvas(method_prefix+"Result",
                          method_prefix+"Sigma-Distribution",
                          0,300,600,400);
c0->cd();

float sig, mean;
for (int i=0; i<n_toys; i++){
  // Reset histo contents
  h4->Reset();
  // Fill histo
  for ( int j = 0; j<n_tot_entries; j++ )
    h4->Fill(gRandom->Gaus());
  // perform fit
  if (do_chi2) h4->Fit("gaus","q"); // Chi2 fit
  else h4->Fit("gaus","lq"); // Likelihood fit
  // some control output on the way
  if (!(i%100)){
    h4->Draw("ep");
    c0->Update();}

  // Get sigma from fit
  TF1 *fit = h4->GetFunction("gaus");
  sig = fit->GetParameter(2);
  mean= fit->GetParameter(1);
  sigma->Fill(sig);
  pull->Fill(mean/sig * sqrt(n_tot_entries));
} // end of toy MC loop
```

macro9.C [continue]

```
  // print result
  c1->cd();
  pull->Draw();
}

void macro9(){
  int n_toys=10000;
  int n_tot_entries=100;
  int n_bins=40;
  cout << "Performing Pull Experiment with chi2 \n";
  pull(n_toys,n_tot_entries,n_bins,true);
  cout << "Performing Pull Experiment with Log Likelihood\n";
  pull(n_toys,n_tot_entries,n_bins,false);
}
```

怎样编译运行？试一试

As a very simple yet powerful quantity to check the quality of the fit results, we construct for each pseudo-data set the so-called "pull", the difference of the estimated and the true value of a parameter, normalised to the estimated error on the parameter, $\frac{(p_{estim}-p_{true})}{\sigma_p}$. If everything is OK, the distribution of the pull values is a standard normal distribution, i.e. a Gaussian distribution centred around zero with a standard deviation of one.

Storing ROOT Objects

```
void write_to_file(){

    // Instance of our histogram
    TH1F h("my_histogram","My Title;X;# of entries",100,-5,5);

    // Let's fill it randomly
    h.FillRandom("gaus");

    // Let's open a TFile
    TFile out_file("my_rootfile.root","RECREATE");

    // Write the histogram in the file
    h.Write();

    // Close the file
    out_file.Close();
}
```

read_from_file.C

```
void read_from_file(){

    // Let's open the TFile
    TFile* in_file= new TFile("my_rootfile.root");

    // Get the Histogram out
    //  TH1F* h = (TH1F*) in_file->GetObjectChecked("my_histogram","TH1F");

    TH1F* h = (TH1F*) in_file->Get("my_histogram");

    // Draw it
    h->Draw();
}
```

Codes in Tutorials

$ROOTSYS/tutorials/hist$ root hstack.C
$ROOTSYS/tutorials/hist$ root hstack.C
$ROOTSYS/tutorials/tree/
tree0.C tree1.C tree2.C tree3.C tree4.C
hvector.C
$ROOTSYS/tutorials/physics$ root PhaseSpace.C

结束语

今天仅仅汇报了我所了解的ROOT部分功能，希望能对您有一些可取之处。

谢谢！

Siguang@pku.edu.cn

Tools.cc 的部分函数

```
void SetSgStyle(){
  // No Canvas Border
  gStyle->SetCanvasBorderMode(0);
  gStyle->SetCanvasBorderSize(0);
  // White BG
  gStyle->SetCanvasColor(10);
  // Format for axes
  gStyle->SetLabelFont(22,"xyz");
  gStyle->SetLabelSize(0.06,"xyz");
  gStyle->SetLabelOffset(0.01,"xyz");
  gStyle->SetNdivisions(510,"xyz");
  gStyle->SetTitleFont(22,"xyz");
  gStyle->SetTitleColor(1,"xyz");
  gStyle->SetTitleSize(0.06,"xyz");
  gStyle->SetTitleOffset(0.91);
  gStyle->SetTitleYOffset(1.1);
  // No pad borders
  gStyle->SetPadBorderMode(0);
  gStyle->SetPadBorderSize(0);
  // White BG
  gStyle->SetPadColor(10);
  // Margins for labels etc.
  gStyle->SetPadLeftMargin(0.15);
  gStyle->SetPadBottomMargin(0.15);
  gStyle->SetPadRightMargin(0.05);
  gStyle->SetPadTopMargin(0.06);
  // No error bars in x direction
  gStyle->SetErrorX(0);

  // Format legend
  gStyle->SetLegendBorderSize(0);
  gStyle->SetLegendFont(22);
  gStyle->SetFillStyle(0);
}
```

```
TH1D * newTH1D(TString name,Double_t binw, Double_t LowBin, Double_t Hig
hBin,Bool_t MevTitle,Int_t iMode){
  Int_t nbin = TMath::Nint( (HighBin - LowBin)/binw );
  HighBin = binw*nbin + LowBin;

  TH1D *h = new TH1D(name.Data(),"",nbin,LowBin,HighBin);
  if(MevTitle) h->GetYaxis()->SetTitle(Form("Events / %.0fMeV",h->GetBin
Width(1)*1000));
  h->SetMinimum(0.0);
  h->GetYaxis()->SetTitleOffset(1.1);
  if(iMode>=0 && iMode<14){
    Int_t iMarker[] = {20,21,24,25,28,29,30,27,3,  5,2, 26,22,23};
    Int_t iColor[]  = { 2, 4, 6, 9, 1,50,40,31,41,35,44,38,47,12};
    h ->SetMarkerStyle(iMarker[iMode]);
    h ->SetMarkerColor(iColor[iMode]);
    h ->SetLineColor(iColor[iMode]);
  }

  return h;
}
```

How to Install Root Under Linux

Siguang WANG

一步步知道自己在干啥的安装

Step by Step

[hepfarm02] /ClusterDisks/HDN05/WorkSpace/testroot/ForStudent >

cp Download/root_v5.32.00.source.tar.gz ./

[hepfarm02] /ClusterDisks/HDN05/WorkSpace/testroot/ForStudent >

tar -zxvf root_v5.32.00.source.tar.gz

会在当前目录下产生
root目录

[hepfarm02] /ClusterDisks/HDN05/WorkSpace/testroot/ForStudent >

emacs autoInstall.sh &

输入文件内容如下

Step by Step

```
#!/bin/bash

export ROOTSYS=/ClusterDisks/HDN05/WorkSpace/testroot/ForStudent/root532
cd root
./configure  --enable-roofit
make -j8
make install
```

将文件属性改成可执行文件

```
chmod u+x autoInstall.sh
```

在当前建立目录(确保/ClusterDisks/HDN05/WorkSpace/testroot/ForStudent/root532存在):

```
mkdir root532
```

Step by Step

在/home/testroot/.bashrc下添加

```
export ROOTSYS=/ClusterDisks/HDN05/WorkSpace/testroot/ForStudent/root532
export PATH=$ROOTSYS/bin:$PATH
export LD_LIBRARY_PATH=$ROOTSYS/lib:$LD_LIBRARY_PATH
```

Intall ROOT with Pythia6 and Pythia8

```
#!/bin/bash
export PYTHIA8=/scratch/other/wangsg/root64/pythia8176
export PYTHIA8DATA=$PYTHIA8/xmldoc
export PYTHIA6=/scratch/other/wangsg/root64/pythia6
export ROOTSYS=/scratch/other/wangsg/root64/534
export FFTW3=/scratch/other/wangsg/root64/fftw

cd root/

 ./configure  --enable-fftw3  --enable-tmva  --enable-unuran  --enable-roofit
 --with-fftw3-incdir=/scratch/other/wangsg/root64/fftw/include --with-fftw3-
libdir=/scratch/other/wangsg/root64/fftw/lib
--enable-pythia8 --with-pythia8-incdir=$PYTHIA8/include  --with-pythia8-
libdir=$PYTHIA8/lib
--enable-pythia6 --with-pythia6-libdir=$PYTHIA6 --with-pythia6-uscore=$PYTHIA6


make -j60
make install
```