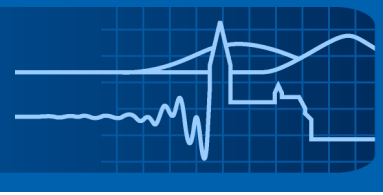
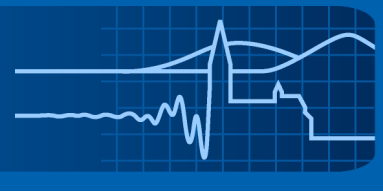


ROOT Einführung im Rahmen des FP

Julian Glatzer
julian.glatzer@physik.uni-freiburg.de

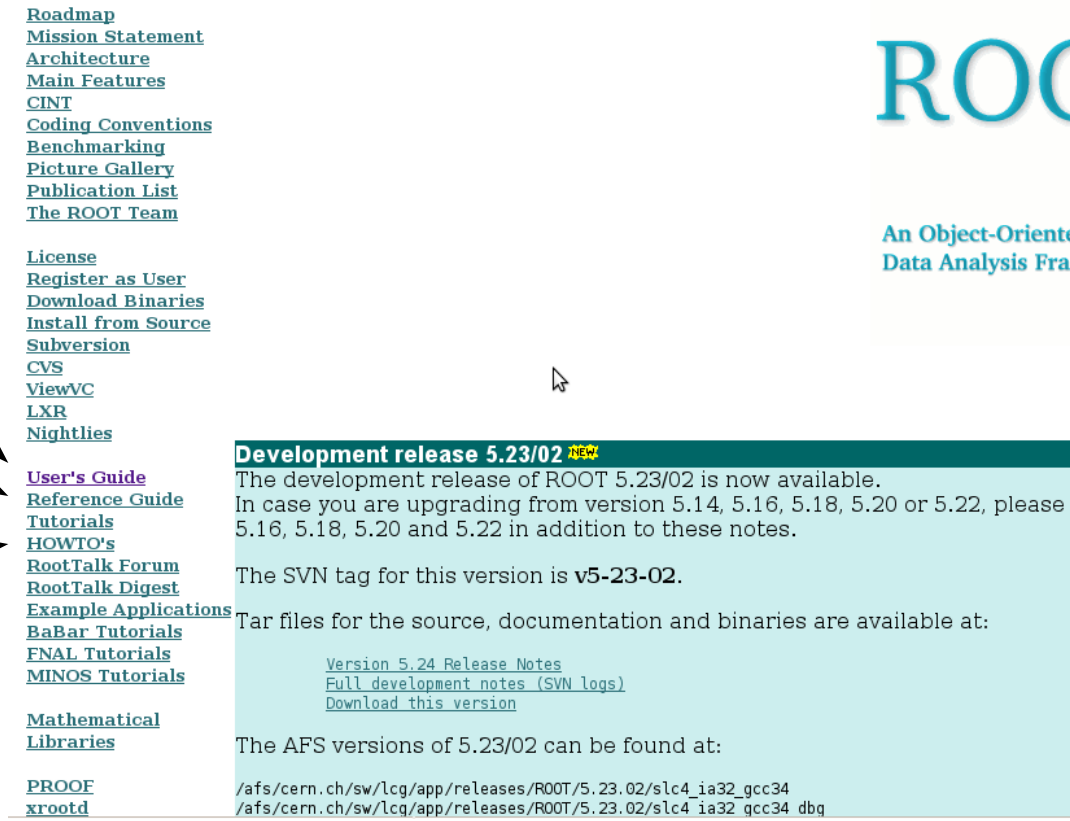
Was ist ROOT?



- ROOT ist ein am CERN entwickeltes Softwarepaket zur Datenanalyse (insbesondere in der Teilchenphysik). ROOT basiert auf C++ Klassen.
- Aber keine Angst: Es ist nicht notwendig C++ im Detail zu können. Kenntnisse schaden aber nicht.
- Wobei kann mir ROOT helfen?
  - Histogramme, Fitten, u.v.m.

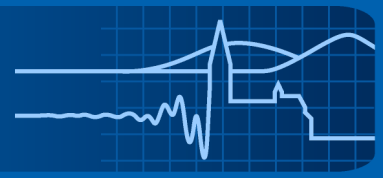
Hilfe

- http://root.cern.ch
  - User's Guide
    in “Buchform”
  - Reference Guide
    Erklärung aller Klassen
  - HowTos
- Google Suche
  - TH1F site:root.cern.ch



- Bei mir: Westbau 02-009, Tel: 8406, E-Mail
- Alle Beispiele sind auf der FP Webseite

Installation



- Im CIP-Pool bereits vorinstalliert
- Downloadlinks auf http://root.cern.ch/root/Version522.html
  - Linux: tar.gz unter Linux SLC5 downloaden (Ubuntu: Quellen)
    - tar xvfz root_v5.22.00.Linux-slc5-gcc4.3.tar.gz
    - export ROOTSYS=<path to root> z.B. /home/jglatzer/root
    - export PATH=$ROOTSYS/bin:$PATH
    - export LD_LIBRARY_PATH=$ROOTSYS/lib:$LD_LIBRARY_PATH
  - Windows: WindowsXP/NT/w2000 with VC++ 7.1 (runs with VC++6 and VC++8.0), version 5.22/00
    - Windows Installer, eventuell ftp://root.cern.ch/root/msvcp71.dll, msvcr71.dll nach C:\root\bin downloaden (VISTA, XP)
    - Mit Icon auf Desktop starten oder MSDOS Eingabeaufforderung “root”

Erste Schritte

- Starten durch Eingabe von root / Klick auf Desktop Icon Root
- Root hat einen eingebauten C++ Interpreter
- Mit den Tasten ↑ ↓ erreicht man die zuletzt benutzten Befehle
- Beenden mit .q

```
File  Edit  View  Terminal  Tabs  Help
jglatzer@pico:~$ root.exe
   ****************************************
   *                                      *
   *       W E L C O M E  to  R O O T      *
   *                                      *
   *   Version  5.18/00b     10 March 2008  *
   *                                      *
   *  You are welcome to visit our Web site  *
   *          http://root.cern.ch         *
   *                                      *
   ****************************************

ROOT 5.18/00b (branches/v5-18-00-patches@22563, Oct 19
 2008, 22:04:00 on linux)

CINT/ROOT C/C++ Interpreter version 5.16.29, Jan 08, 2
008
Type ? for help. Commands must be C++ statements.
Enclose multiple statements between { }.
root [0] 2+2
(const int)4
root [1] int i=1
root [2] i++
(int)1
root [3] i
(int)2
root [4] .q
jglatzer@pico:~$
```

Macros

- Makros automatisieren wiederkehrende Arbeiten
- Werden in Textdateien geschrieben (z.B. beispiel.C) und mit .x beispiel.C ausgeführt
- Der Dateiname hallo.C muss dem Namen der Funktion void hallo() entsprechen.

```
void beispiel(){
  gROOT->Reset();
  cout << "Beispiel 1: "<<endl;
  cout << "Wurzel aus 2 = "
       << sqrt(2.) << endl;
}
```

```
File  Edit  View  Terminal  Tabs  Help
jglatzer@pico:~/root$ root.exe
  ****************************************
  *                                      *
  *       W E L C O M E  to  R O O T     *
  *                                      *
  *   Version  5.18/00b     10 March 2008  *
  *                                      *
  *  You are welcome to visit our Web site  *
  *          http://root.cern.ch         *
  *                                      *
  ****************************************

ROOT 5.18/00b (branches/v5-18-00-patches@22563, Oct 19
 2008, 22:04:00 on linux)

CINT/ROOT C/C++ Interpreter version 5.16.29, Jan 08, 2
008
Type ? for help. Commands must be C++ statements.
Enclose multiple statements between { }.
root [0] .x beispiel.C
Beispiel 1:
Wurzel aus 2 = 1.41421
root [1] .x beispiel2.C
Error: Function beispiel2() is not defined in current
scope  :0:
*** Interpreter error recovered ***
root [2] .x beispiel2.C(2.)
Beispiel 2:
Wurzel aus 2 = 1.41421
root [3]
```

Macros

- Makros automatisieren wiederkehrende Arbeiten
- Werden in Textdateien geschrieben (z.B. beispiel.C) und mit .x beispiel.C ausgeführt
- Der Dateiname hallo.C muss dem Namen der Funktion void hallo() entsprechen.

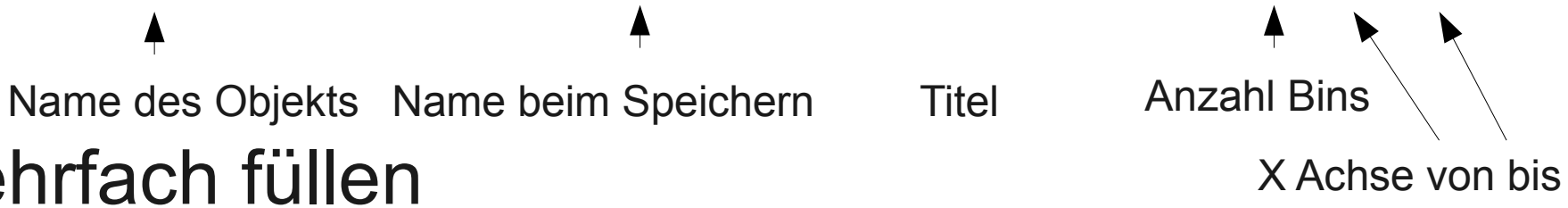
```
void beispiel2(double x){
  gROOT->Reset();
  cout << "Beispiel 2: "<<endl;
  cout << "Wurzel aus " << x
       << " = " << sqrt(x) << endl;
}
```

```
File  Edit  View  Terminal  Tabs  Help
jglatzer@pico:~/root$ root.exe
  ****************************************
  *                                      *
  *       W E L C O M E  to  R O O T      *
  *                                      *
  *   Version  5.18/00b     10 March 2008  *
  *                                      *
  *  You are welcome to visit our Web site  *
  *          http://root.cern.ch         *
  *                                      *
  ****************************************

ROOT 5.18/00b (branches/v5-18-00-patches@22563, Oct 19
 2008, 22:04:00 on linux)

CINT/ROOT C/C++ Interpreter version 5.16.29, Jan 08, 2
008
Type ? for help. Commands must be C++ statements.
Enclose multiple statements between { }.
root [0] .x beispiel.C
Beispiel 1:
Wurzel aus 2 = 1.41421
root [1] .x beispiel2.C
Error: Function beispiel2() is not defined in current
scope  :0:
*** Interpreter error recovered ***
root [2] .x beispiel2.C(2.)
Beispiel 2:
Wurzel aus 2 = 1.41421
root [3]
```

Histogramme

- Buchen

```
TH1F* myHist=new TH1F("myHisto","Distribution",10,0.,1.);
```



- Mehrfach füllen

```
myHist->Fill(x);
```

- Zeichnen

```
myHist->Draw("Option");
```

wobei `Option=""`, `"E"` (√N Fehlerbalken), `"SAME"` (Histogramm über altes Histogramm), `"C"` (glatte Kurve), `"L"` (Linie) oder Kombinationen z.B. `"SAME,E"`.

Histogramme

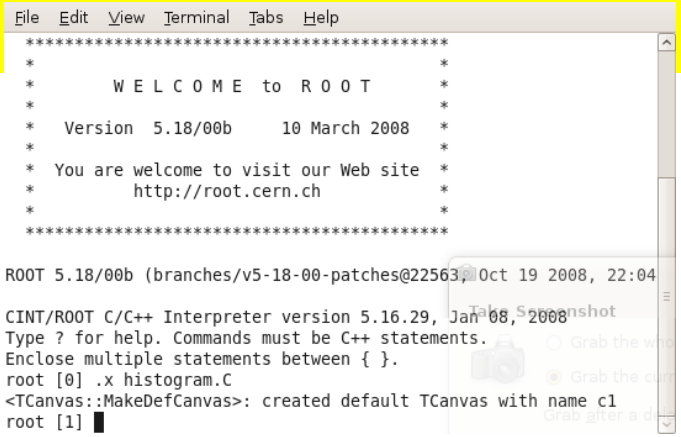
```
void histogram(){
  gROOT->Reset();
  gROOT->SetStyle("Plain");
  TH1F* myHist = new TH1F("myHist","Distribution"
   ,10,0.,1.);
  myHist->Fill(0.37); //Bin 4
  myHist->Fill(0.35); //Bin 4
  myHist->Fill(0.78); //Bin 8
  myHist->Fill(0.51); //Bin 6
  myHist->Draw();
}
```

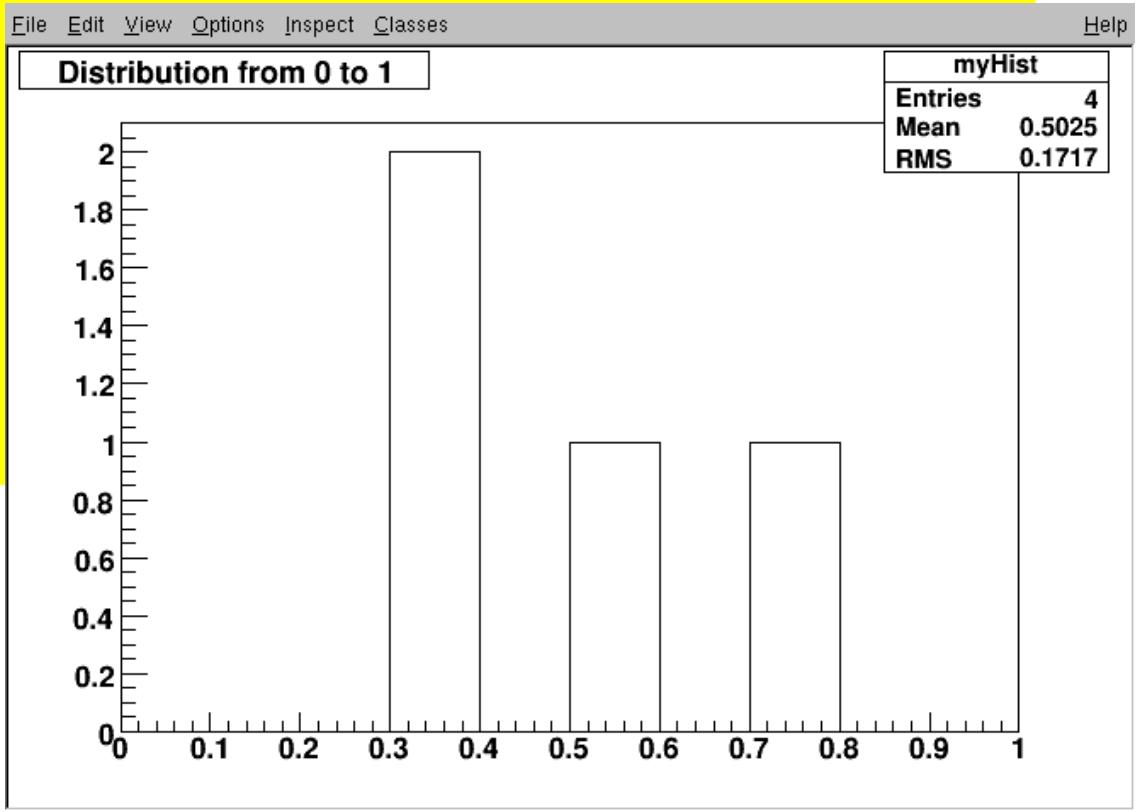
Für schönere Grafik

Histogramme

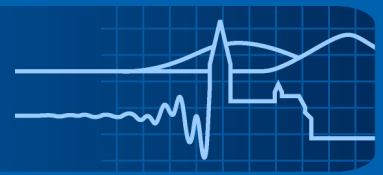
```
void histogram(){
   gROOT->Reset();
   gROOT->SetStyle("Plain");
   TH1F* myHist = new TH1F("myHist","Distribution"
    ,10,0.,1.);
   myHist->Fill(0.37);
   myHist->Fill(0.35);
   myHist->Fill(0.78);
   myHist->Fill(0.51);
   myHist->Draw();
}
```

Für schönere Grafik





Optionen



- Für Linien

  Hist->SetLineStyle(x); Hist->SetLineColor(x); Hist->SetLineWidth(x);

- Für Marker

  Hist->SetMarkerStyle(x); Hist->SetMarkerColor(x); Hist->SetMarkerSize(x);

- Für Flächen

  Hist->SetFillStyle(x); Hist->SetFillColor(x);

- Für Achsen

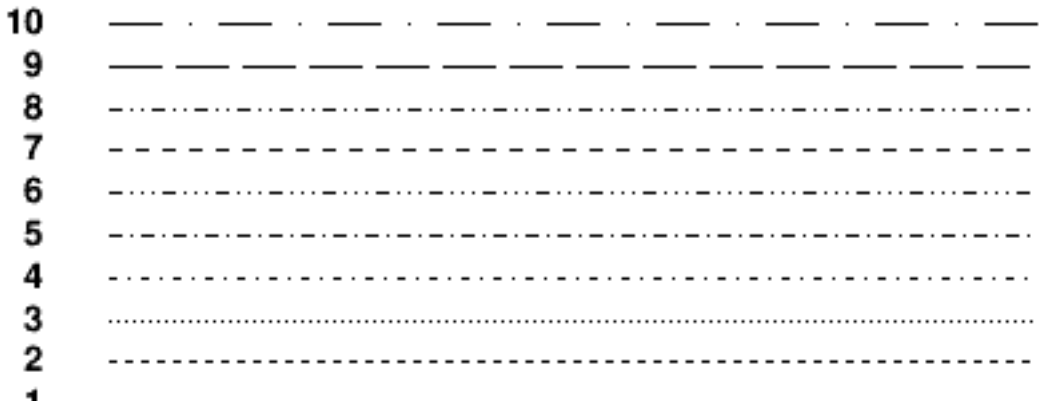
  Hist->GetXaxis()->SetTitle("#phi_{2}^{2}"); ← Latex mit # statt \

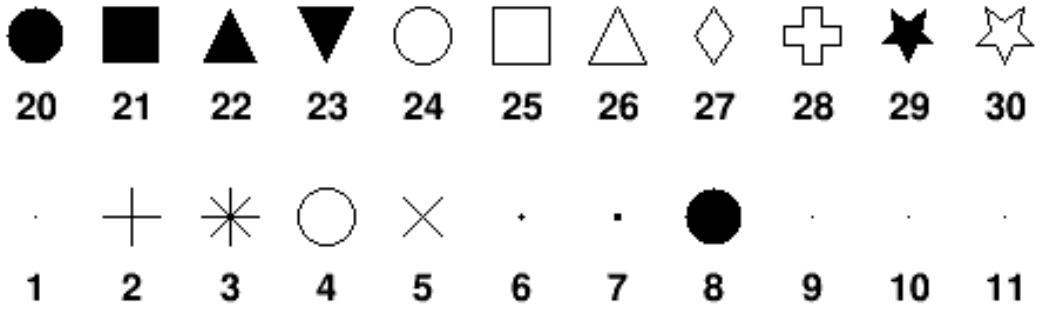
  Hist->GetYaxis()->SetRangeUser(-1.,1.);

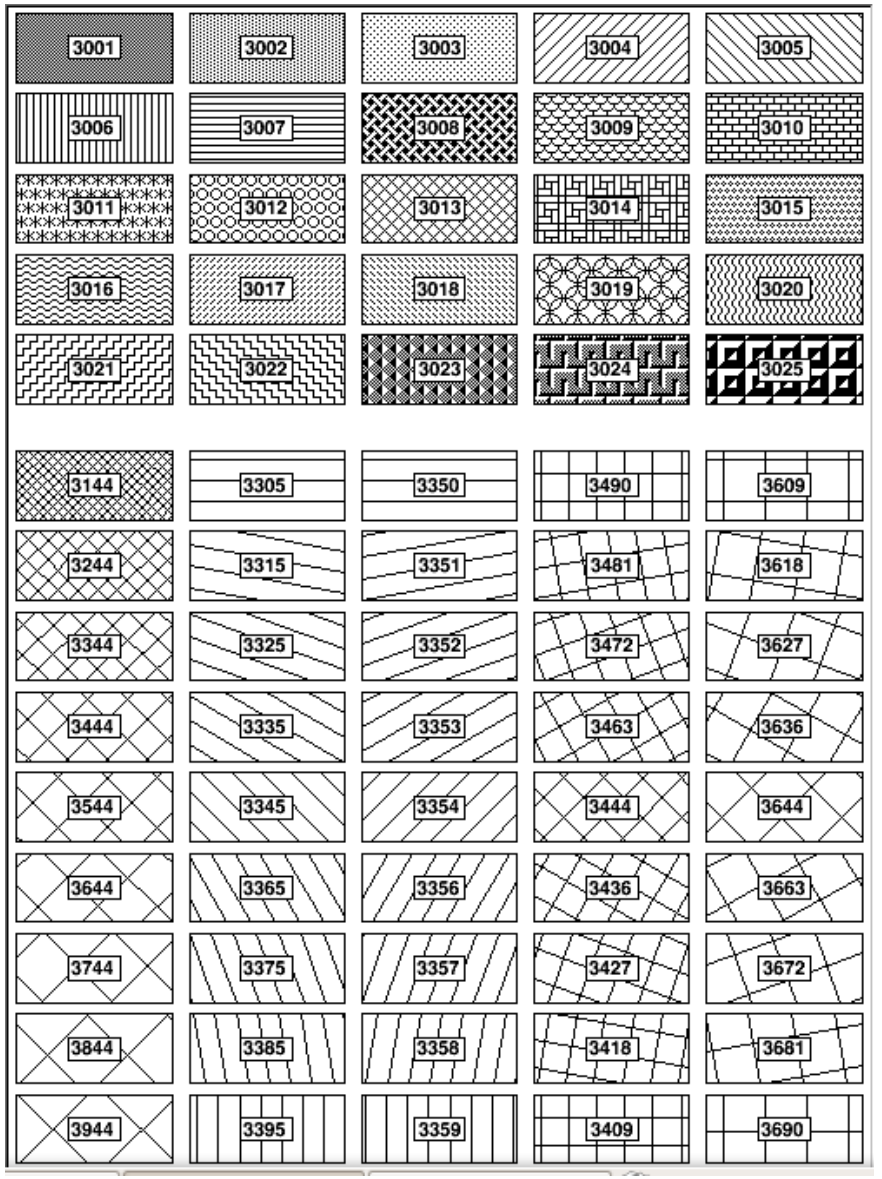
- Für den Titel

  Hist->SetTitle("Titel");

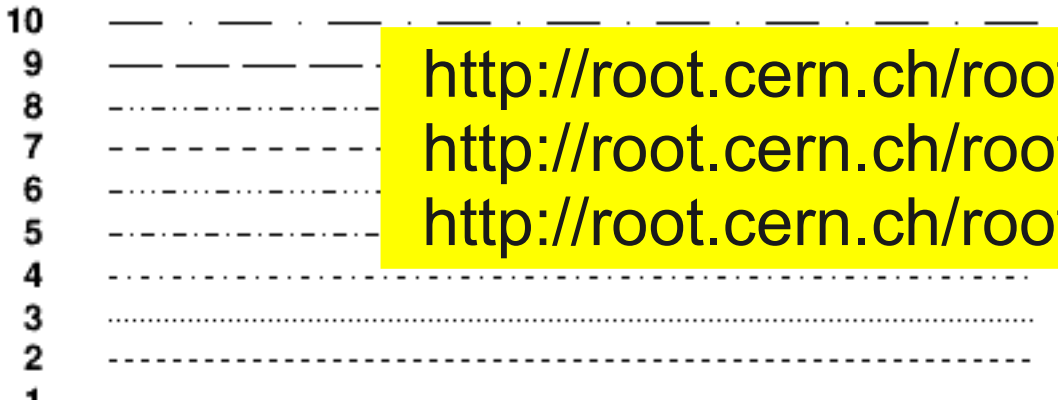
Farben, Linien, Marker, ...

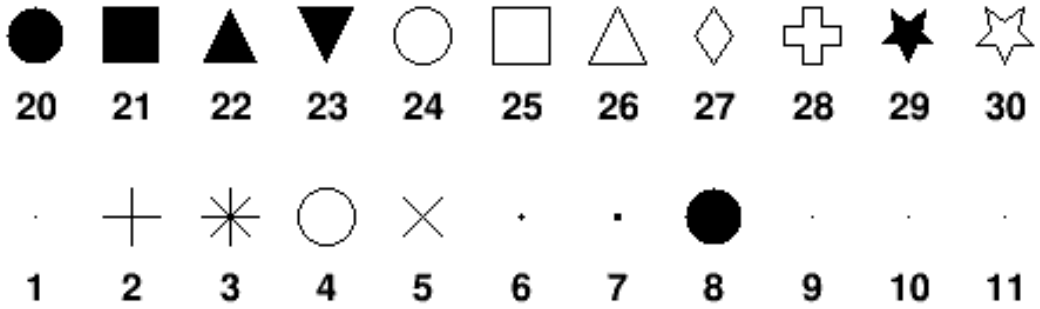


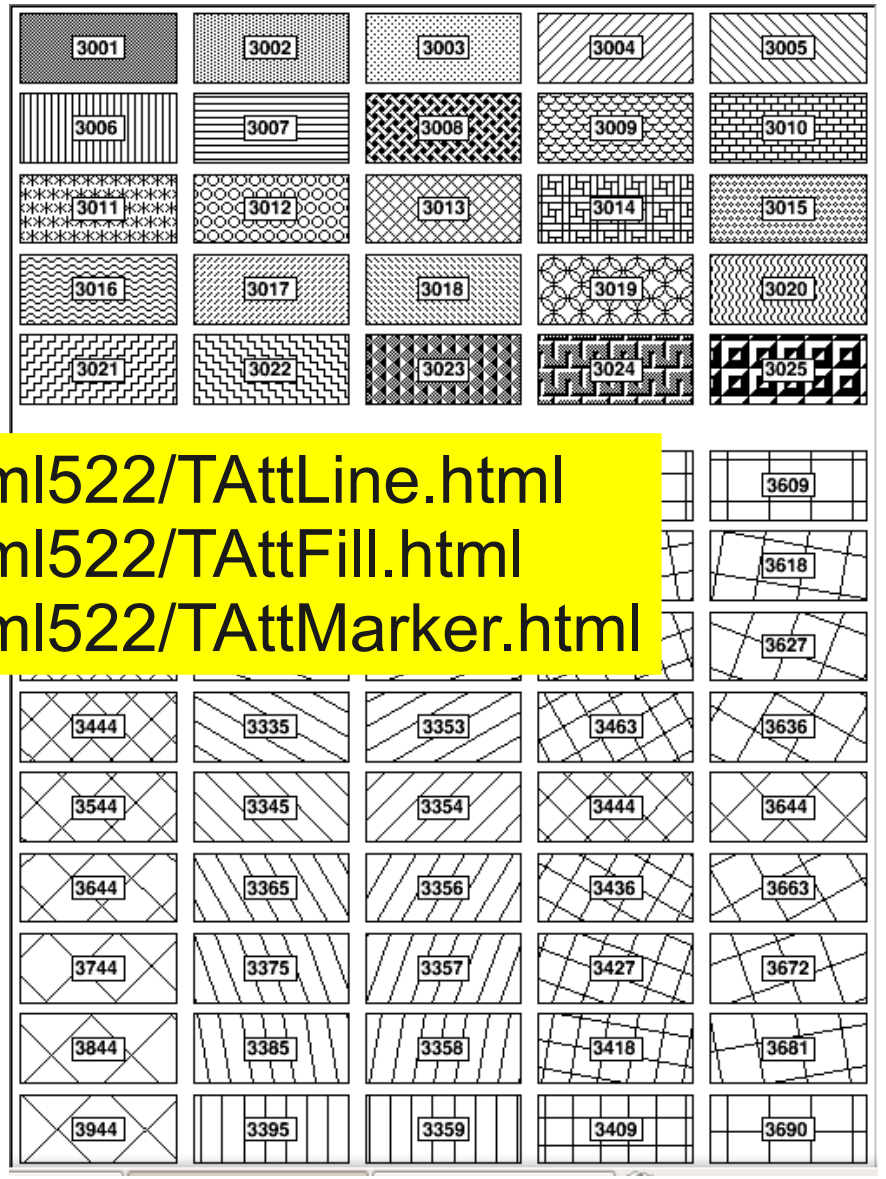




Farben, Linien, Marker, ...

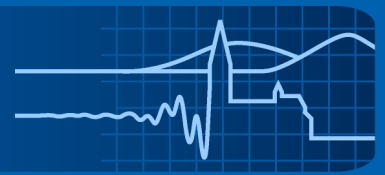






http://root.cern.ch/root/html522/TAttLine.html
http://root.cern.ch/root/html522/TAttFill.html
http://root.cern.ch/root/html522/TAttMarker.html

Root Dateien



Histogramm speichern

```
void writeHisto(){
  gROOT->Reset(); gROOT->SetStyle("Plain");
  TFile* _file=new TFile("histo.root","RECREATE");
  TH1F* myHist = new TH1F("myHisto","Distribution",10,0.,1.);
  myHist->Fill(0.37); myHist->Fill(0.35);
  myHist->Fill(0.78); myHist->Fill(0.51);
  myHist->Write();
  _file->Close();
}
```

Histogramm laden

```
void readHisto(){
  gROOT->Reset();
  gROOT->SetStyle("Plain");
  TFile* _file=new TFile("histo.root","OPEN");
  TH1F* _myH1 = (TH1F*)_file->Get("myHisto");
  _myH1->Draw();
}
```

oder

TBrowser br

Funktionen

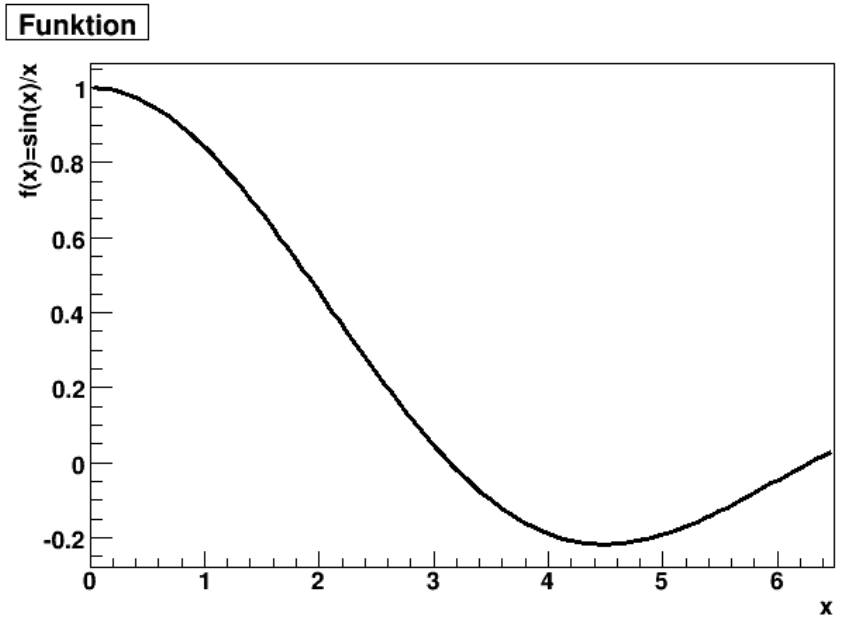
ROOT kann

- exp(x), sin(x), log(x), sqrt(x)
- gaus (mit 1 s!), Landau,...
- TMath Funktionen

http://root.cern.ch/root/html522/TMath.html)

- Beliebige Funktionen

(http://root.cern.ch/root/html/tutorials/fit/myfit.C.html)




```
void function(){
  gROOT->Reset();
  gROOT->SetStyle("Plain");
  TF1 *myFunc = new TF1("myFunction", "sin(x)/x",0.,6.5.);
  myFunc->Draw();
  myFunc->GetXaxis()->SetTitle("x");
  myFunc->GetYaxis()->SetTitle("f(x)=sin(x)/x");
  myFunc->SetTitle("Funktion");
}
```

Einlesen von Daten

```
void readFile(){
  gROOT->Reset();
  gROOT->SetStyle("Plain");
  ifstream in;              //Input Stream
  in.open("peaks.dat"); //Oeffnen der Datei
  Float_t xi;
  Int_t nlines = 0;

  TH1F* _histo = new TH1F("_histo","Peaks",
   1250, 0., 125 );

  while( !in.eof() ){    //Bis zum Ende der Datei
    if(in >> xi){        //Einlesen einer Zeile
      _histo->SetBinContent( nlines, xi );
      //Setzen des Bin Inhalts
      nlines++;
      cout << nlines << ": " << xi <<endl;
    }
  }

  cout<<"found "<<nlines<<" data points"<<endl;
  in.close();
  _histo->Draw();
}
```

z.B. Spannung in 0.1s Schritten

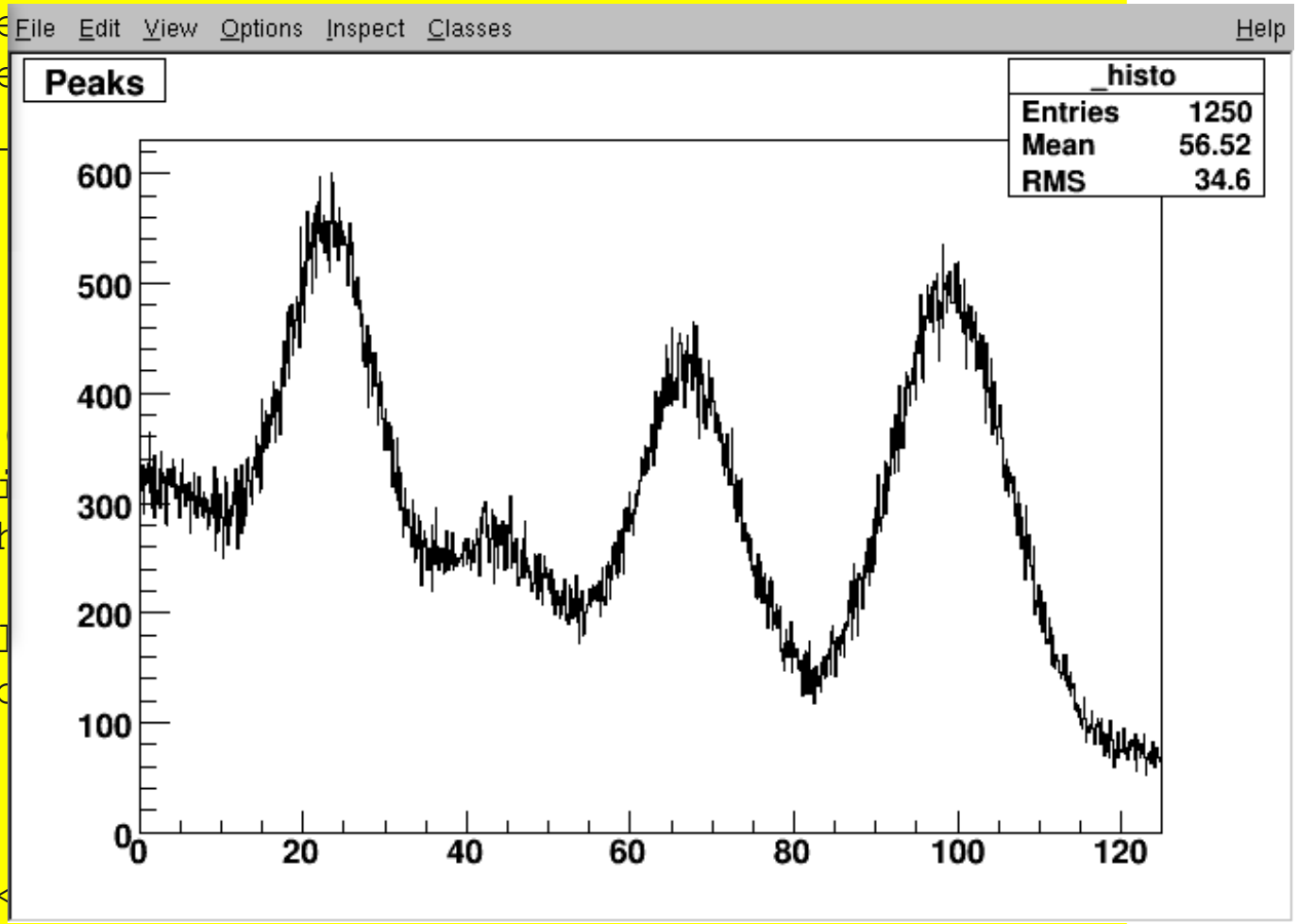
peaks.dat:

322
323
322
312
314
335
291
331
329
317
318
337
336
364
310
etc.

Einlesen von Daten

```
void readFile(){
  gROOT->Reset();
  gROOT->SetStyle("Plain");
  ifstre
  in.ope
  Float_
  Int_t

  TH1F*
   1250

  while
    if(i
      _h
       /
      nl
      co
    }
  }

  cout<<
  in.close();
  _histo->Draw();
}
```

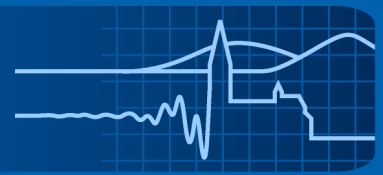
z.B. Spannung in 0.1s Schritten



peaks.dat:

322
323
322
312
314
335
291
331
329
317
318
337
336
364
310
etc.

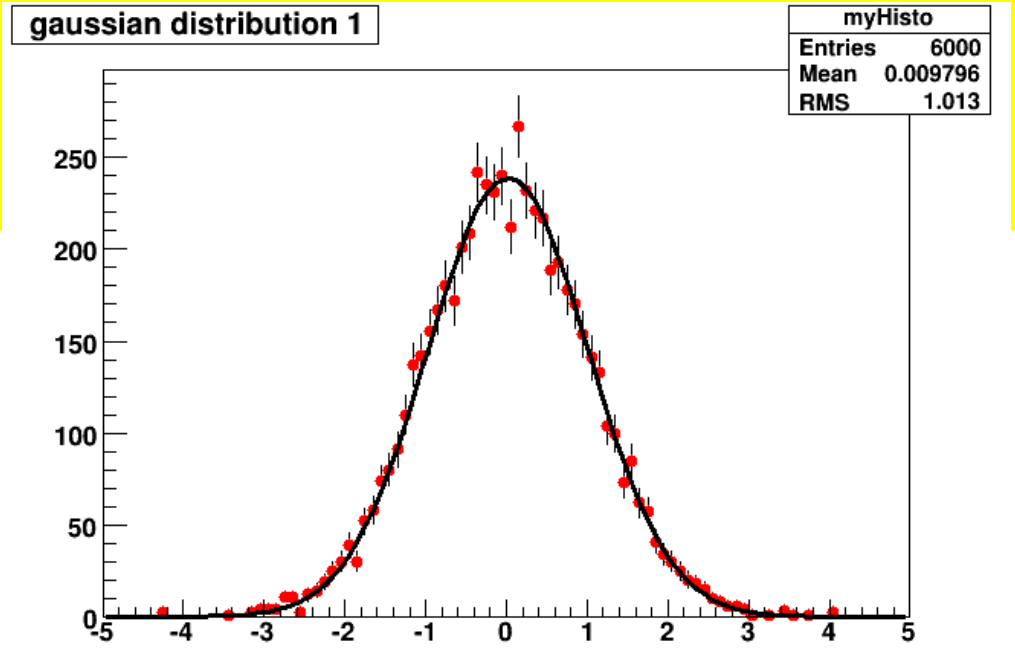
1D-Fits



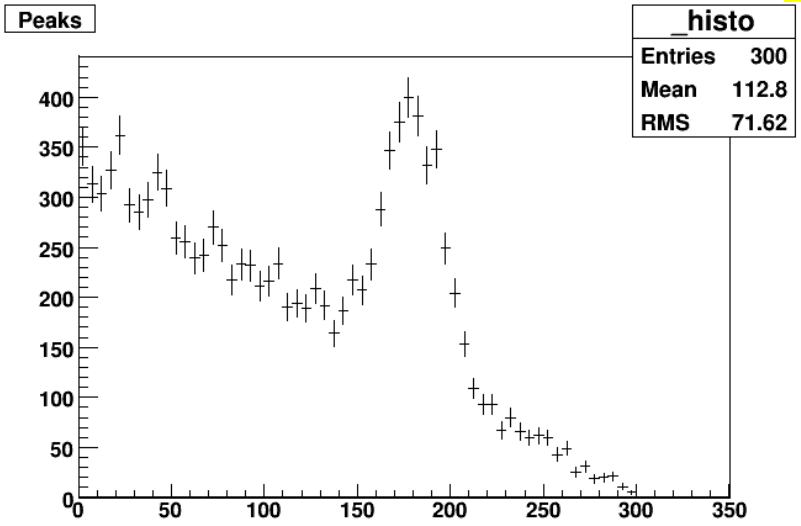
```
void fit(){
  gROOT->Reset();
  gROOT->SetStyle("Plain");
  TH1F* myH1 = new TH1F("myHisto","gaussian distribution",100,-5.,5.);
  TF1* myGaus = new TF1("myGaus","gaus",-5,5); //Fitbereich [-5,5]
    // Gauss "gaus" 3 Fitparameter: Konstante, mu, sigma
    // Polynom "pol0", "pol1", "pol2", ...
    // Eigene Funktionen: z.B. "[0]*sin([1]*x)"
  myH1->FillRandom("gaus",6000);//Histogramm zufaellig nach Gauss
  myH1->SetMarkerColor(2);
  myH1->SetMarkerStyle(20);

  myH1->Fit("myGaus");
  myH1->Draw("E");
  cout<<" ------------------------------" <<endl;
  cout<<" chi2/dof: "<< myGaus->GetChisquare()/myGaus->GetNDF()<<endl;
  cout<<" mean: "<< myGaus->GetParameter(1)
      <<"+/-"<<myGaus->GetParError(1)<<endl;
  cout<<" sigma: "<< myGaus->GetParameter(2)
      <<"+/-"<<myGaus->GetParError(2)<<endl;
}
```

Ausgabe des 1D-Fits

```
root [17] .x fit.C
 FCN=65.2561 FROM MIGRAD    STATUS=CONVERGED      62 CALLS          63 TOTAL
                     EDM=1.03954e-09    STRATEGY= 1      ERROR MATRIX ACCURATE
  EXT PARAMETER                                   STEP         FIRST
  NO.   NAME      VALUE            ERROR          SIZE      DERIVATIVE
   1  Constant     2.38701e+02   3.81138e+00   1.23151e-02  -6.02893e-06
   2  Mean         2.07507e-02   1.29856e-02   5.15153e-05   2.51545e-03
   3  Sigma        9.92900e-01   9.33484e-03   1.00065e-05   3.36906e-03
<TCanvas::MakeDefCanvas>: created default TCanvas with name c1
 ---------------------------------
 chi2/dof: 0.973971
 mean: 0.0207507+/-0.0129856
 sigma: 0.9929+/-0.00933484
root [18]
```

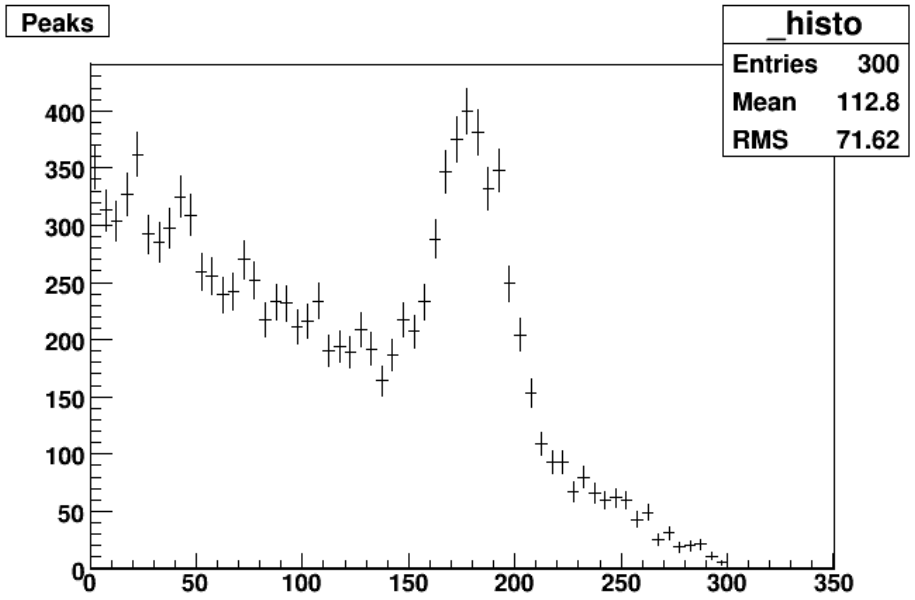


Ein komplizierterer Fit




```
void fit2_nurhist(){
  gROOT->Reset();
  gROOT->SetStyle("Plain");
  ifstream in;          //Input Stream
  in.open("peak.dat"); //Oeffnen der Datei
  Float_t xi;
  Float_t yi;
  Int_t nlines = 0;
  TH1F* _histo =
    new TH1F("_histo","Peaks", 350, 0., 350 );
  while( !in.eof() ){
    if(in >> xi >> yi){
      _histo->SetBinContent( xi, yi );
      nlines++;
      cout << nlines << ": "
           << xi << " " << yi << endl;
    }
  }
  cout << "found "<< nlines
       << " data points."<<endl;
  in.close();
  gStyle->SetOptFit();
  _histo->Rebin(5);
  _histo->Draw("E");
}
```

Ein komplizierterer Fit



- Funktion

$$f(x) = p_1 + p_2 x + p_3 \exp -\frac{1}{2}\left(\frac{x - p_4}{p_5}\right)^2$$

- In Root

```
pol1(0)+gauss(2)=[0]+[1]*x+
[2]*exp(-((x-[3])/(2*[4]))^2)
```

- Parameter...
  - Startwert:

```
func->SetParameter(Index,Wert);
```

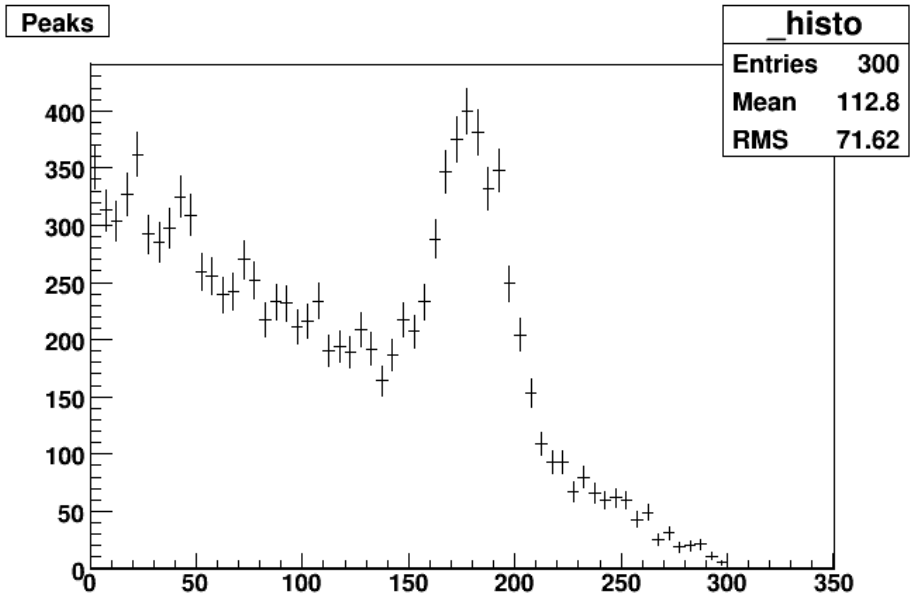
  - Bereich:

```
func->SetParLimits(Index, von, bis);
```

  - Anzeige aller Parameter

```
gStyle->SetOptFit(1111);
```

Ein komplizierterer Fit



Format pcev (Standard 111 und 11 statt 0011)
p = 1; Probability
c = 1; Chisquare/Number of DOF
e = 1; errors (if e=1, v must be 1)
v = 1; name/values of parameters

- Funktion

$$f(x) = p_1 + p_2 x + p_3 \exp -\frac{1}{2}\left(\frac{x - p_4}{p_5}\right)^2$$

- In Root

```
pol1(0)+gauss(2)=[0]+[1]*x+
[2]*exp(-((x-[3])/(2*[4]))^2)
```

- Parameter...
  - Startwert:

```
func->SetParameter(Index,Wert);
```

  - Bereich:

```
func->SetParLimits(Index, von, bis);
```

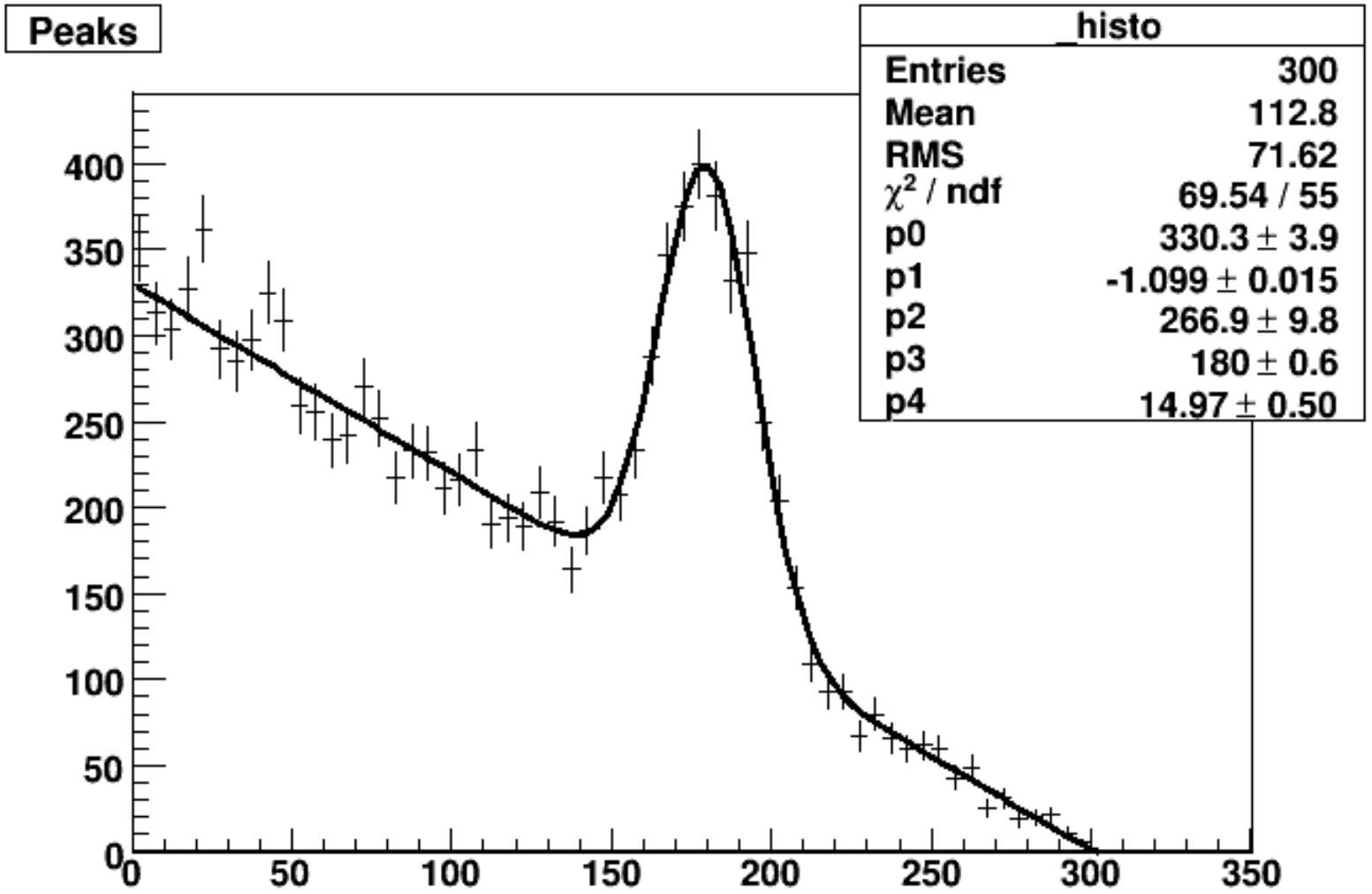
  - Anzeige aller Parameter

```
gStyle->SetOptFit(1111);
```

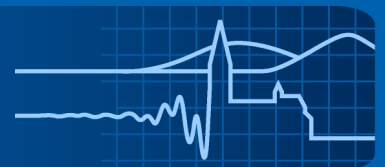
Ein komplizierterer Fit

```
void fit2(){

  [... Initialisierung ...]

  TH1F* _histo = new TH1F("_histo","Peaks", 350, 0., 350 );

  [... Daten einlesen ...]

 TF1* fitFunc = new TF1("fitFunc","pol1(0)+gaus(2)",0,300);
  //Polynom ersten Grades [0]+[1]*x
  //(Parameternummerierung startet bei 0)
  //multipliziert mit Gaussfunktion [2]*exp(-((x-[3])/(2*[4]))^2)
  //(Parameternummerierung startet bei 2)
  fitFunc->SetParameter(3,175);
  fitFunc->SetParameter(4,20);
  in.close();
  gStyle->SetOptFit();
  _histo->Rebin(5);
  _histo->Fit("fitFunc");
  _histo->Draw("E");
}
```

Ein komplizierterer Fit



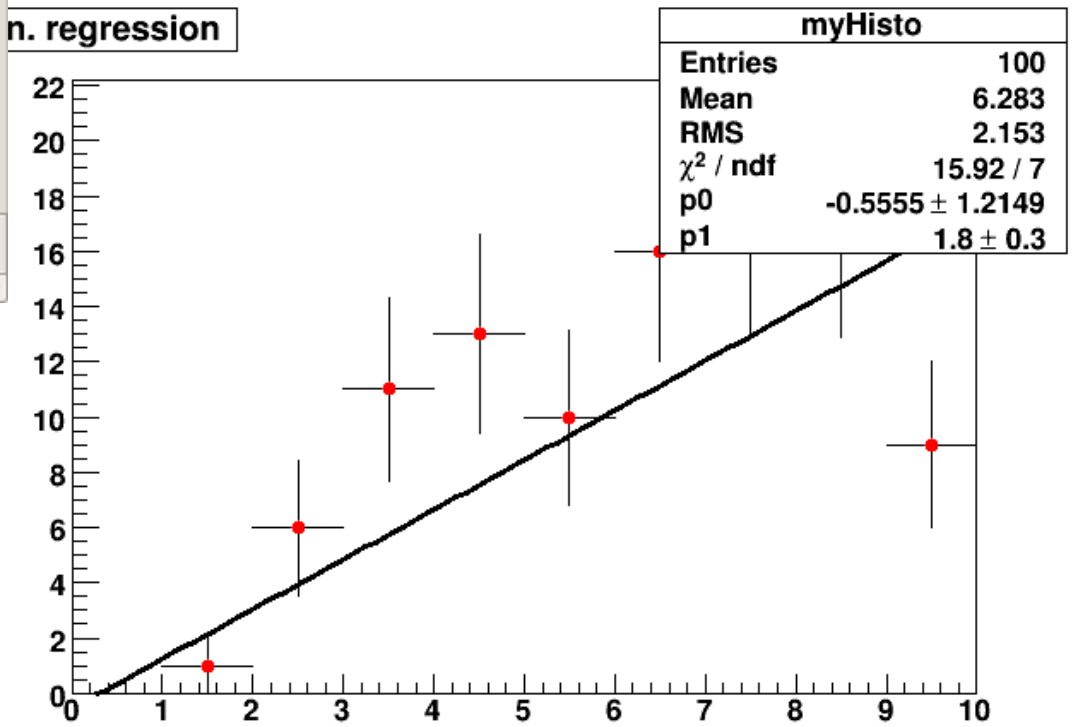
Kovarianzmatrix



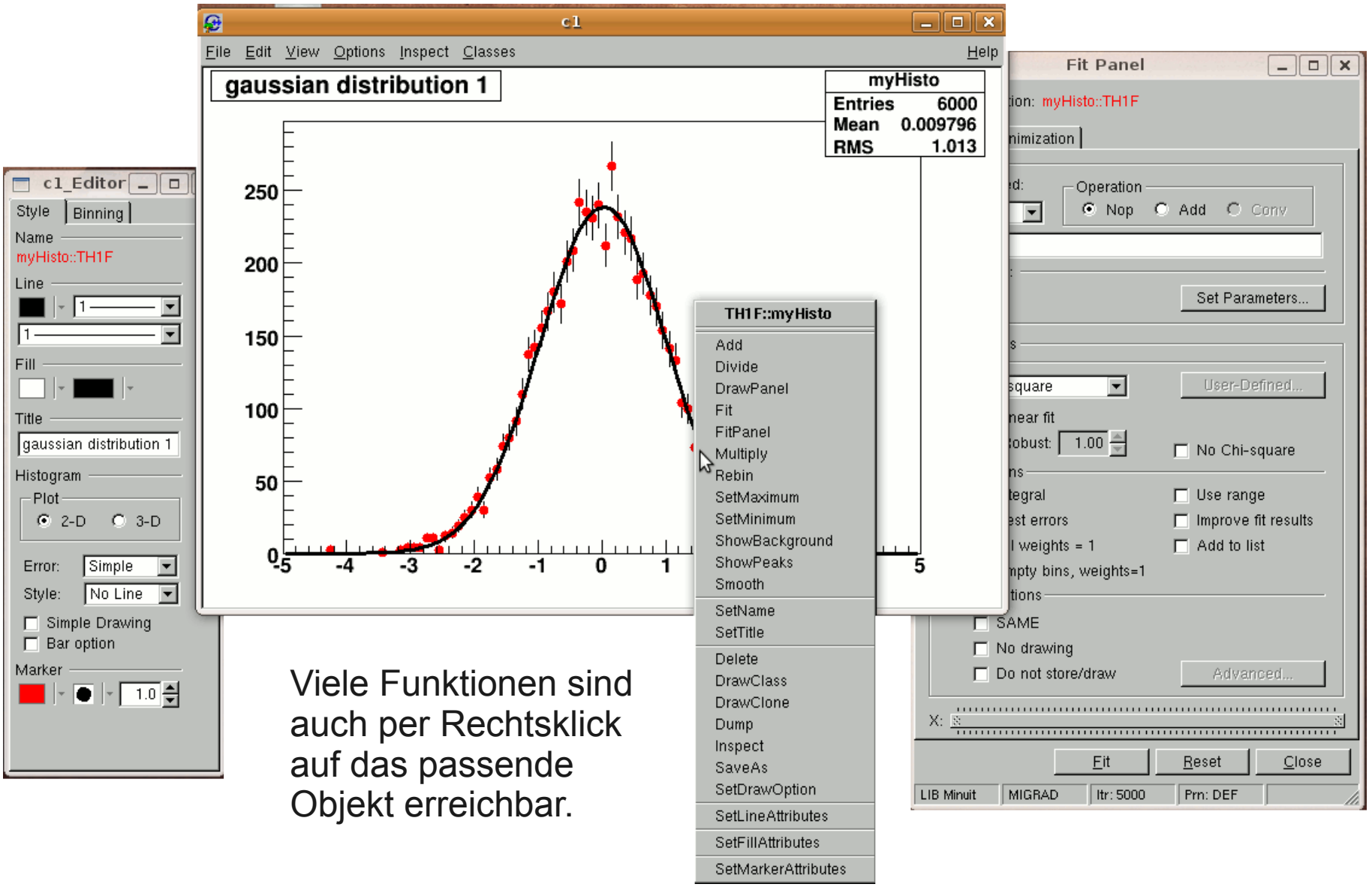
```
void linRegression(){
  gROOT->Reset();
  gROOT->SetStyle("Plain");
  gStyle->SetOptFit();
  TH1F* myH1 = new TH1F("myHisto","lin. regression",10,0.,10.);
  TF1* myPol1 = new TF1("myPol1","2*x",0.,10.);
  myH1->FillRandom("myPol1",100);
  myH1->SetMarkerColor(2);
  myH1->SetMarkerStyle(20);
  myH1->Fit("pol1");
  myH1->Draw("E");
  TVirtualFitter *fitter = TVirtualFitter::GetFitter();
  TMatrixD *matrix = new TMatrixD(2,2,fitter->GetCovarianceMatrix());
  matrix->Print();
}
```

Kovarianzmatrix

```
File  Edit  View  Terminal  Tabs  Help
CINT/ROOT C/C++ Interpreter version 5.16.29, Jan 08, 20
Type ? for help. Commands must be C++ statements.
Enclose multiple statements between { }.
root [0] .x linRegression.C
Fitting results:
Parameters:
NO.              VALUE            ERROR
0       -5.555001e-01   1.214911e+00
1       1.799546e+00    2.952771e-01
<TCanvas::MakeDefCanvas>: created default TCanvas with

2x2 matrix is as follows

     |      0    |      1    |
--------------------------------
   0 |      1.476     -0.2796
   1 |    -0.2796     0.08719

root [1]
```

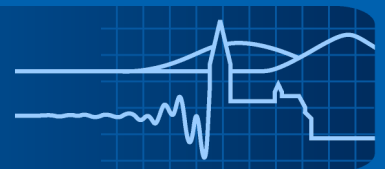


Rechtsklickmenüs



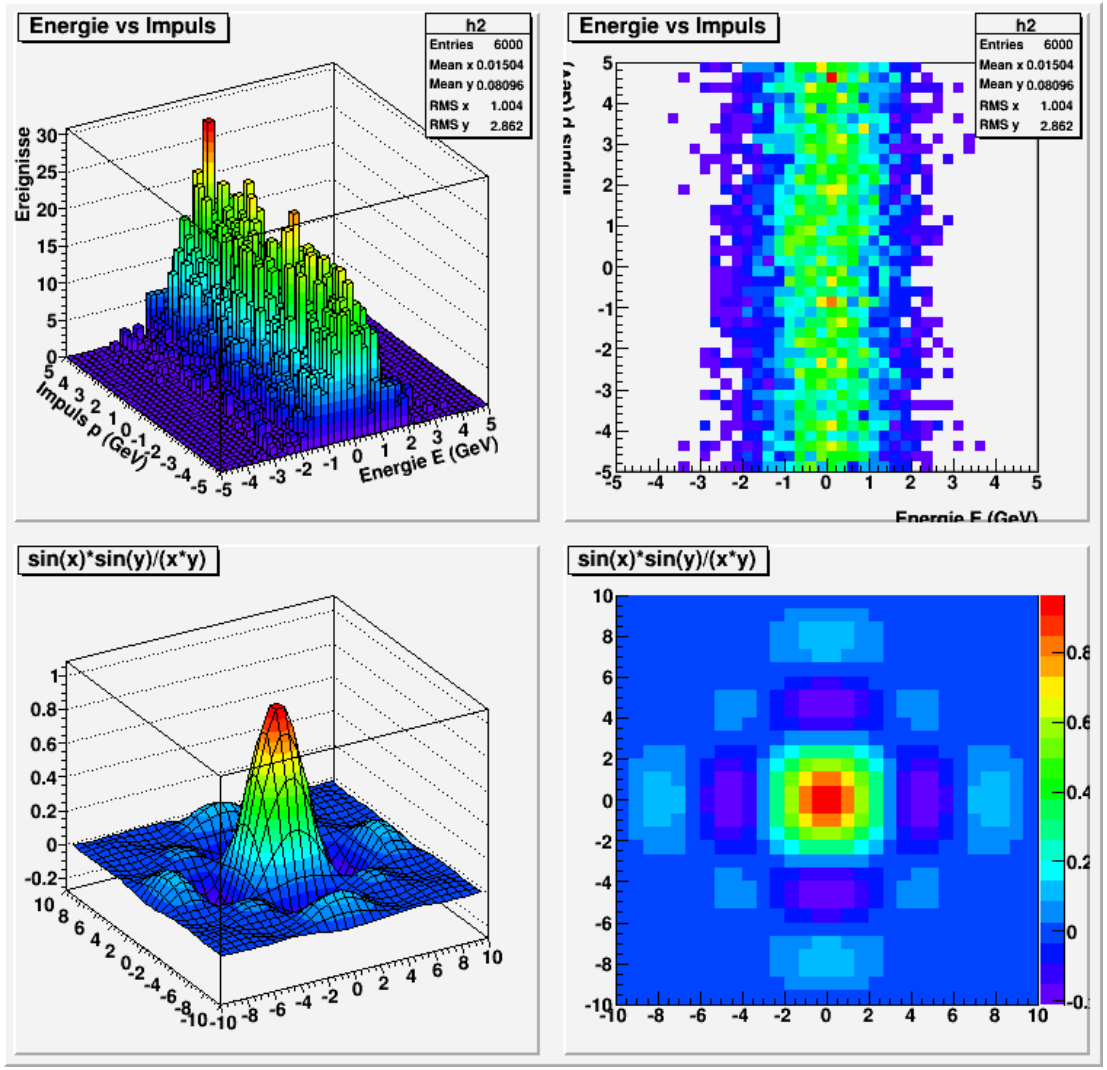
Viele Funktionen sind auch per Rechtsklick auf das passende Objekt erreichbar.

2D-Histogramme



```
void twodhistos(){
  gROOT->Reset();
  gStyle->SetPalette(1);
  TCanvas *c1 = new TCanvas("c1","Canvas fuer viele Histogramme",800,800);
  c1->Divide(2,2);
  TH2F *h2 = new TH2F("h2","Energie vs Impuls",40,-5.,5.,40,-5.,5.);
  h2->FillRandom("gaus",6000);
  h2->GetXaxis()->SetTitle("Energie E (GeV)");
  h2->GetYaxis()->SetTitle("Impuls p (GeV)");
  h2->GetZaxis()->SetTitle("Ereignisse");
  TF2* f2=new TF2("func2","sin(x)*sin(y)/(x*y)",-10.,10.,-10.,10.);
  c1->cd(1);
  h2->GetXaxis()->SetTitleOffset(1.5);
  h2->GetYaxis()->SetTitleOffset(1.5);
  h2->GetZaxis()->SetTitleOffset(1.2);
  h2->Draw("LEGO2");
  c1->cd(2);
  h2->Draw("COL");
  c1->cd(3);
  f2->Draw("SURF1");
  c1->cd(4);
  f2->Draw("COLZ");
}
```

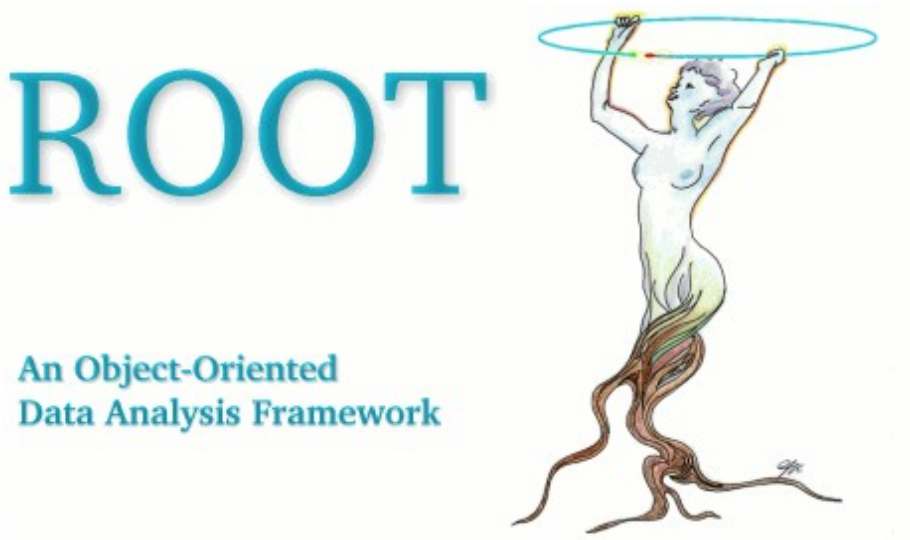
2D-Histogramme



Zusammenfassung

- Root zu benutzen ist eigentlich gar nicht so schwer
- Die Online-Hilfe ist empfehlenswert.
- Ansonsten dürft ihr auch gerne mich fragen.

Danke.



http://root.cern.ch