Root 2009

www.if.usp.br/suaide

Alexandre Suaide

aula 2

Programa

- **Aula 1**
  - Introdução ao c++ e ROOT
  - Conceito de classe e objeto
  - Básico de gráficos e funções no ROOT
- **Aula 2**
  - Mais gráficos e funções
  - Histogramas de 1 e 2D
  - Ajustes de funções, legendas, etc.
  - Escrevendo programas simples: Monte Carlo e simulações
- **Aula 3**
  - Referências e ponteiros
  - Nomes e memória
  - Programação mais complexa: mais Monte Carlo
- **Aula 4**
  - I/O no ROOT
  - Mais programação no ROOT
  - Compilando com o ROOT

Algumas variáveis internas do ROOT

- gSystem
  - TSystem – Classe que controla interface com S.O.
    - gSystem->cd("c:\root");
    - gSystem->pwd();
    - gSystem->Exec("emacs meu_programa.C");
- gStyle
  - TStyle – Define vários padrões básicos gráficos, por exemplo, cor padrão de tela, linha, fonte padrão, etc.
- gROOT
  - TROOT – Controla os atributos globais do ROOT
- gRandom
  - TRandom – Gerador de números aleatórios
- gPad
  - Variável da classe TPad, sempre corresponde à tela gráfica ativa atualmente
    - gPad->SetLogy(); // log Y na tela ativa

Gráficos e funções

$$f(x) = \frac{A}{Bx\sqrt{C^2 + \left(Dx - \frac{1}{Bx}\right)^2}}$$

fit.dat - /\

File Edit Search
Preferences Shell
Macro Windows
Help

```
3500  18.6  0.8
3600  17.3  0.9
3700  19.1  0.9
3800  18.7  1.0
3900  20.1  1.0
4000  21.0  1.1
4100  25.1  1.2
4200  25.0  1.2
4300  28.9  1.4
4400  30.1  1.5
4500  37.4  1.6
4600  35.6  1.8
4700  45.6  2.1
4800  48.9  2.5
4900  59.4  2.9
5000  72.4  3.7
5100  96.0  4.8
5200 146.2  6.6
5300 177.6  8.9
5400 190.2  8.6
5500 124.2  6.2
5600  86.9  4.4
5700  72.5  3.3
5800  53.8  2.6
5900  47.5  2.2
6000  39.7  1.8
6100  32.2  1.6
6200  28.2  1.4
6300  24.9  1.2
6400  22.6  1.1
```

exemplo oRoot/

File Edit Search Pr Help

```
// fora das funcoes ou em
TGraphErrors *g;
TF1 *f;

exemplo_fit()
{
  g = new TGraphErrors("fit.dat","%lg %lg %lg");

  g->Draw("a p");
  g->SetMarkerStyle(20);
  g->SetMarkerColor(2);
  g->SetLineColor(2);
  g->SetTitle("Ressonancia em carga de um RLC");
  g->GetXaxis()->SetTitle("#omega [rad/s]");
  g->GetYaxis()->SetTitle("V_{C} [V]");

  f = new TF1("funcao","[0]/(x*[1]*sqrt([2]^2+(x*[3]-1/(x*[1]))^2))",3000,7000);
  f->SetParameters(5,1e-6,5,30e-3);

  f->Draw("same l");
  f->SetLineWidth(1);
  f->SetLineColor(4);

}
```

Variáveis definidas fora de funções são globais e podem ser manipuladas diretamente no prompt

O same faz com que o gráfico seja desenhado em superposição ao anterior. O l desenha linha

Ajustando funções

No prompt do ROOT digite

Root [0] .x exemplo fit.C

Root [1] g->
(const Doul

Root [2] g->
Aparece ur

Root [3] g->
(const Doul

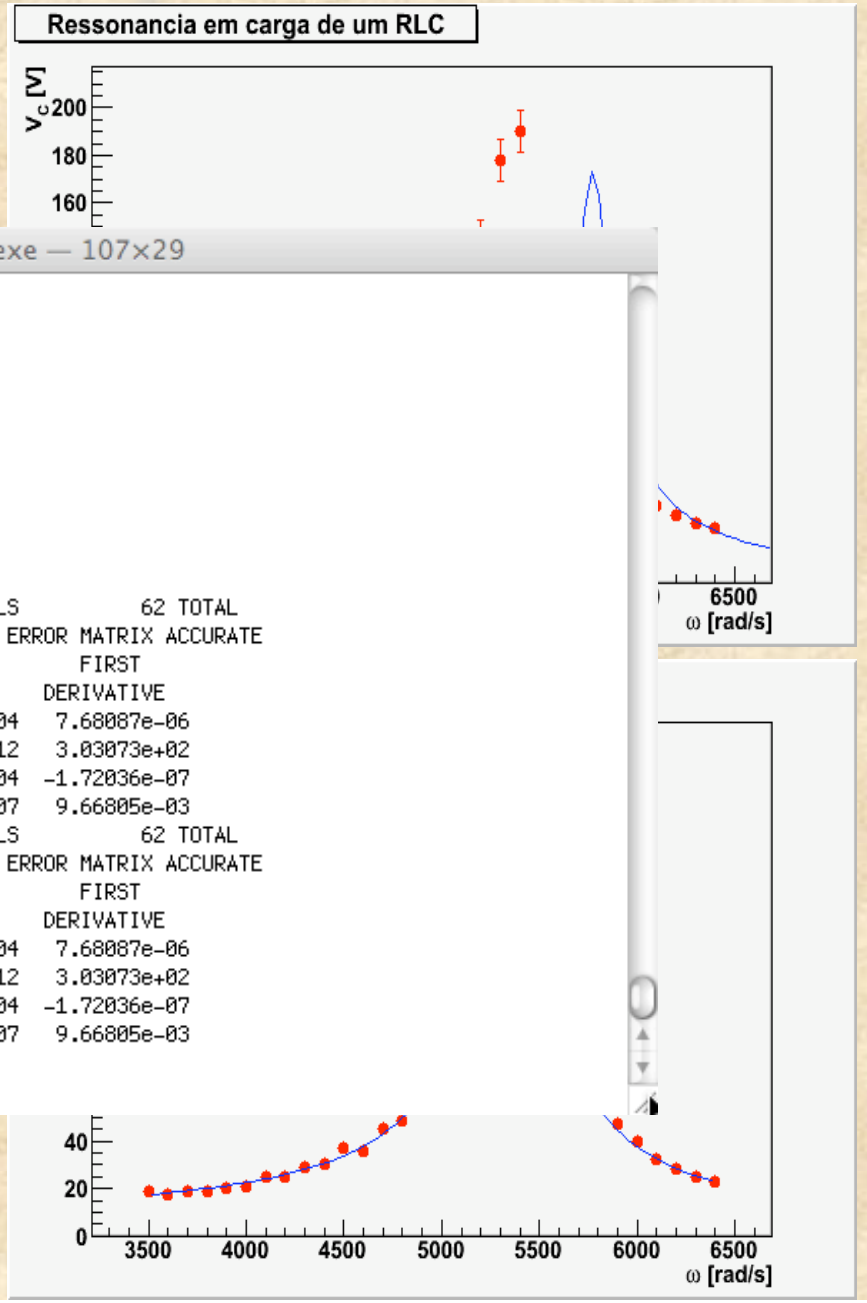
Root [4] g->
(const doub

Root [5] f->
(const Doul

Root [6] f->GetParError(2)
(const Double_t)1.782507e+00

Terminal — root.exe — 107×29

```
root [9]
root [9]
root [9]
root [9]
root [9]
root [9]
root [9] g->Fit(f)
 **********
 **   10 **MIGRAD        5000   0.0003616
 **********
 MIGRAD MINIMIZATION HAS CONVERGED.
 MIGRAD WILL VERIFY CONVERGENCE AND ERROR MATRIX.
 FCN=29.6721 FROM MIGRAD    STATUS=CONVERGED      61 CALLS          62 TOTAL
                     EDM=4.58254e-13    STRATEGY= 1      ERROR MATRIX ACCURATE
  EXT PARAMETER                                          STEP         FIRST
  NO.   NAME      VALUE            ERROR          SIZE      DERIVATIVE
   1  p0           9.97522e+00   1.01255e-01   2.99257e-04   7.68087e-06
   2  p1           1.05316e-06   2.06076e-07   5.12089e-12   3.03073e+02
   3  p2           8.94788e+00   1.78250e+00   7.87576e-04  -1.72036e-07
   4  p3           3.31875e-02   6.49395e-03   1.62045e-07   9.66805e-03
 FCN=29.6721 FROM MIGRAD    STATUS=CONVERGED      61 CALLS          62 TOTAL
                     EDM=4.58254e-13    STRATEGY= 1      ERROR MATRIX ACCURATE
  EXT PARAMETER                                          STEP         FIRST
  NO.   NAME      VALUE            ERROR          SIZE      DERIVATIVE
   1  p0           9.97522e+00   1.01255e-01   2.99257e-04   7.68087e-06
   2  p1           1.05316e-06   2.06076e-07   5.12089e-12   3.03073e+02
   3  p2           8.94788e+00   1.78250e+00   7.87576e-04  -1.72036e-07
   4  p3           3.31875e-02   6.49395e-03   1.62045e-07   9.66805e-03
(Int_t)(0)
root [10]
```



TLegend

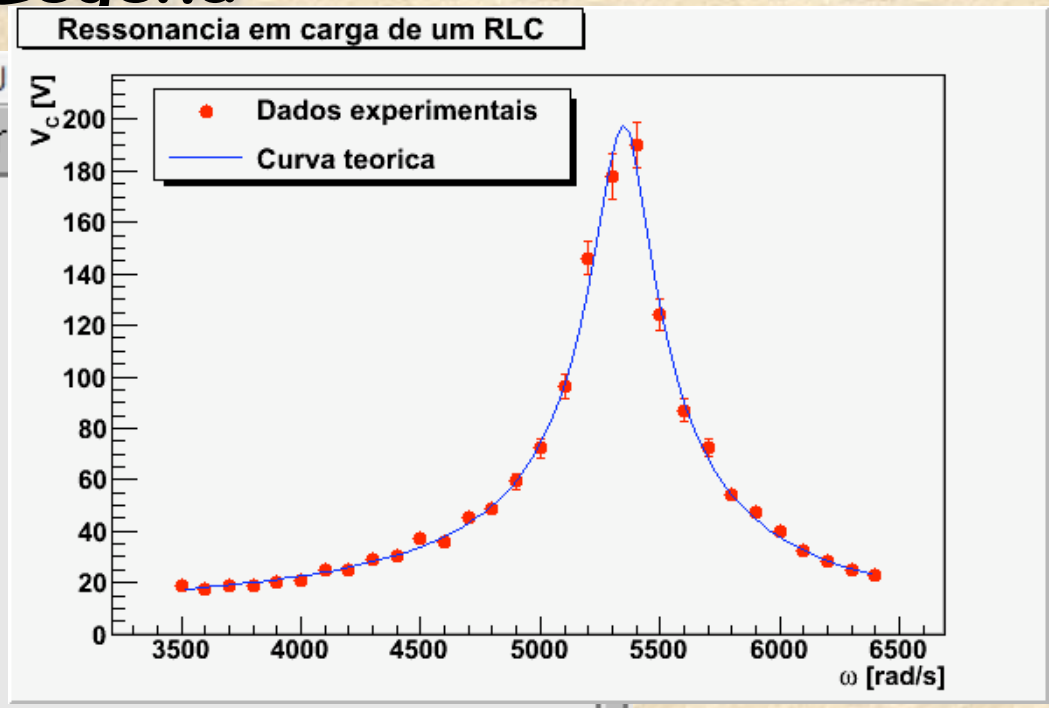
- A classe Tlegend cria legenda de gráficos
  - … = new Tlegend(xi, y1, xf, yf);
  - Onde xi, yi, xf, yf são as coordenadas na tela.
    - Números reais (float) entre 0 e 1
  - Ex:
    - TLegend *L = new TLegend(0.25, 0.30, 0.70, 0.50);
- Adicionando ítem na legenda
  - AddEntry(*objeto, "Título", "modo");
  - Onde modo = p (ponto), l (linha), f (fill, caixa)

TLegend

exemplo_fit_2.C - /Volumes/Data/U

File Edit Search Preferences Shell Macr

```
// variaveis globais. podem ser manipuladas
// fora das funcoes ou em outras funcoes
TGraphErrors *g;
TF1 *f;

exemplo_fit_2()
{
  g = new TGraphErrors("fit.dat","%lg %lg %lg");

  g->Draw("a p");
  g->SetMarkerStyle(20);
  g->SetMarkerColor(2);
  g->SetLineColor(2);
  g->SetTitle("Ressonancia em carga de um RLC");
  g->GetXaxis()->SetTitle("#omega [rad/s]");
  g->GetYaxis()->SetTitle("V_{C} [V]");

  f = new TF1("funcao","[0]/(x*[1]*sqrt([2]^2+(x*[3]-1/(x*[1]))^2))",3000,7000);
  f->SetParameters(5,1e-6,5,30e-3);

  f->Draw("same l");
  f->SetLineWidth(1);
  f->SetLineColor(4);

  g->Fit(f);

  TLegend *l = new TLegend(0.14, 0.75, 0.54, 0.88);
  l->AddEntry(g,"Dados experimentais","p");
  l->AddEntry(f,"Curva teorica","l");
  l->Draw();

}
```



Parte nova adicionada ao programa. Faz o ajuste dos dados e desenha a legenda

Monte Carlo no ROOT

- Sorteio de números aleatórios simples
  - TRandom, TRandom1, TRandom2, TRandom3
    - A única diferença é o algorítimo de geração
  - Geradores básicos

    Exp(tau)
    - Integer(imax)
    - Gaus(mean,sigma)
    - Rndm()
    - Uniform(x1)
    - Landau(mpv,sigma)
    - Poisson(mean)
    - Binomial(ntot,prob)
- Ou usar funções (TF1,etc.) para outras distribuições de probabilidade

Gráficos e histogramas no ROOT

- **Gráficos e histogramas**
  - O ROOT possui uma quantidade enorme de classes para tratar objetos gráficos
  - Histogramas
    - TH1 – Histogramas de 1 dimensão
      - TH1I, TH1S, TH1F, TH1D, ... (estabelece a precisão do eixo)
    - TH2 – Histogramas de 2 dimensões
    - TH3 – Histogramas de 3 dimensões

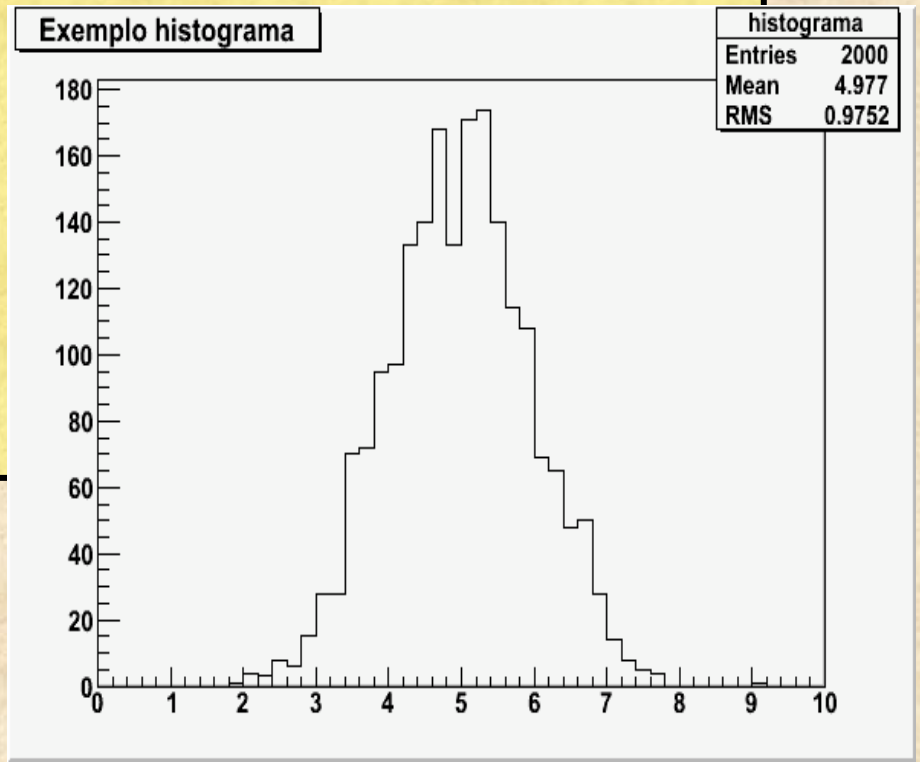
Histogramas de 1 dimensão

- Criando um Histograma de 1 dimensão
  - TH1F *h = new TH1F("nome","título", Nbins, Xmin, Xmax);
  - TH1F h ("nome","título", Nbins, Xmin, Xmax);

```
void exemplo_TH1()
{
  TRandom *r = new TRandom();
  TH1F *h1 = new TH1F("histograma","Exemplo histograma",50,0,10);
  for(int i = 0;i<2000;i++)
  {
    float x = r->Gaus(5,1);
    h1->Fill(x);
  }
  h1->Draw();
}
```

Para rodar esse exemplo, assim como os Seguintes, salve-o em um arquivo, por Exemplo, exemplo_TH1.C e digite, no prompt do ROOT

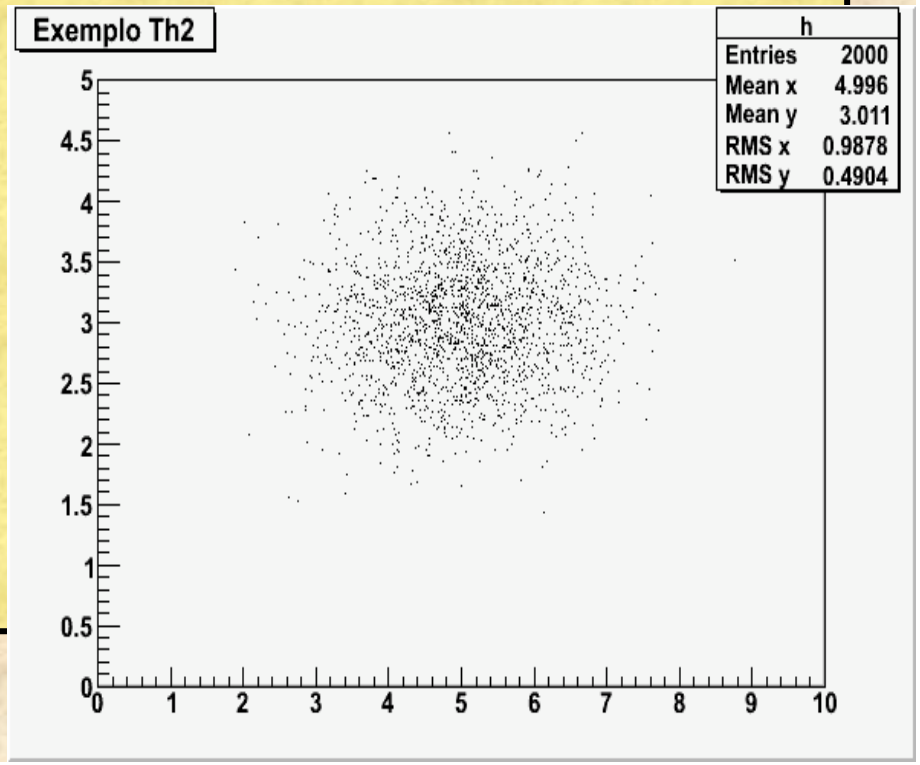
root [0] .x exemplo_TH1.C



Histogramas de 2 dimensões

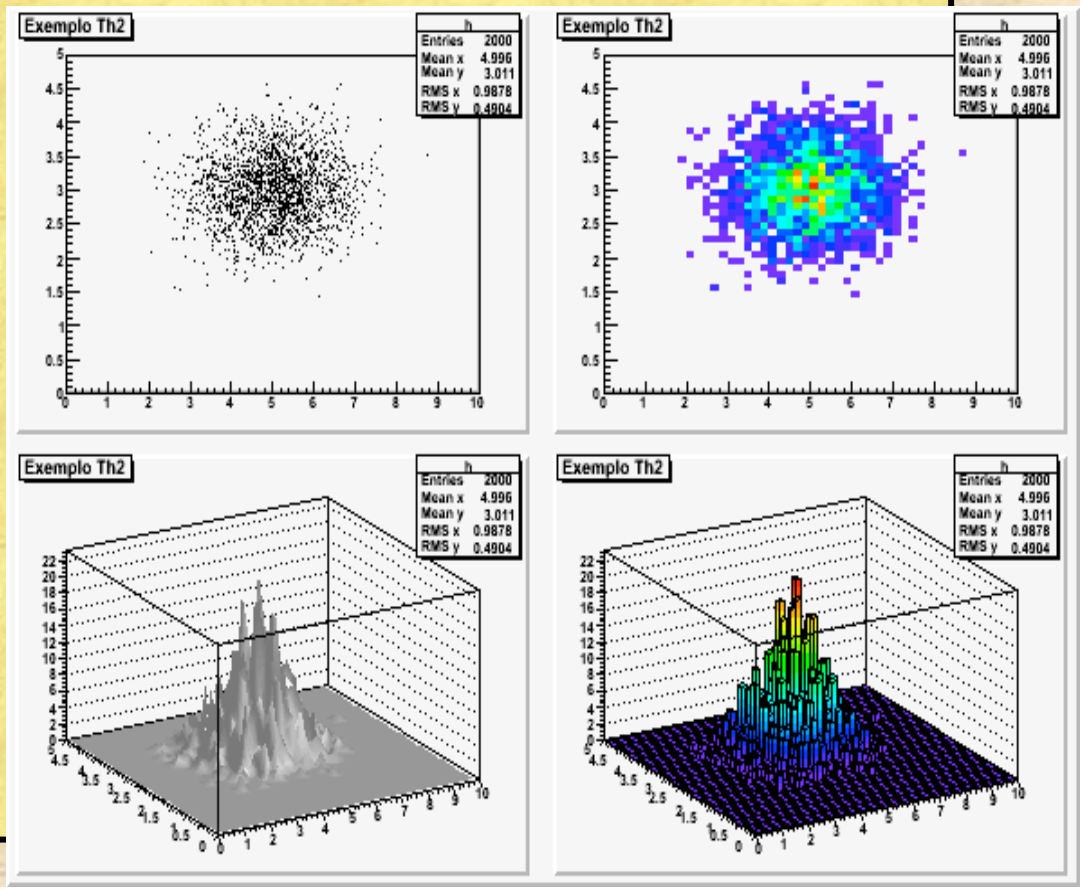
- **Muito similar ao TH1**
  - TH2F *h = new TH2F("nome","título", NbinsX, Xmin, Xmax, NBinsY, Ymin, Ymax);
  - TH2F h ("nome","título", NbinsX, Xmin, Xmax, NbinsY, Ymin,Ymax);

```
void exemplo_TH2()
{
   TRandom *r = new TRandom();
   TH2F *h2 = new TH2F("h","Exemplo Th2",50,0,10,50,0,5);
   for(int i = 0;i<2000;i++)
   {
      float x = r->Gaus(5,1);
      float y = r->Gaus(3,0.5);
      h2->Fill(x,y);
   }
   h2->Draw();
}
```



Dividindo uma tela

```
void exemplo_TH2_2()
 {
      gStyle->SetPalette(1,0);
      TRandom *r = new TRandom();
      TH2F *h2 = new TH2F("h","Exemplo Th2",50,0,10,50,0,5);
      for(int i = 0;i<2000;i++)
      {
        float x = r->Gaus(5,1);
        float y = r->Gaus(3,0.5);
        h2->Fill(x,y);
      }
      TCanvas *c = new TCanvas();
      c->Divide(2,2);
      c->cd(1);
      h2->Draw();
      c->cd(2);
      h2->Draw("col");
      c->cd(3);
      h2->Draw("surf4");
      c->cd(4);
      h2->Draw("lego2");
 }
```



Exemplo: Propagação de incertezas com Monte Carlo

- Distância focal de uma lente convergente

$$\frac{1}{f} = \frac{1}{i} + \frac{1}{o}$$

- i = 10.5 ± 0.8 cm
- o = 25.3 ± 0.2 cm

- Incerteza f = ????

MC_lente.C - /Volumes/Data/Users/suaide/cursoRoot/

File Edit Search Preferences Shell Macro Windows Help

```
 1 TH1F *hist;
 2 float MC_lente(float o, float so, float i, float si, int N)
 3 {
 4   hist = new TH1F("f","foco",100,0,20);
 5
 6   TRandom *r = new TRandom();
 7   for(int k = 0; k<N; k++)
 8   {
 9     float I = r->Gaus(i,si);
10     float O = r->Gaus(o,so);
11     float F = I*O/(I+O);
12     hist->Fill(F);
13   }
14   float RMS =hist->GetRMS();
15   return RMS;
16 }
17
```

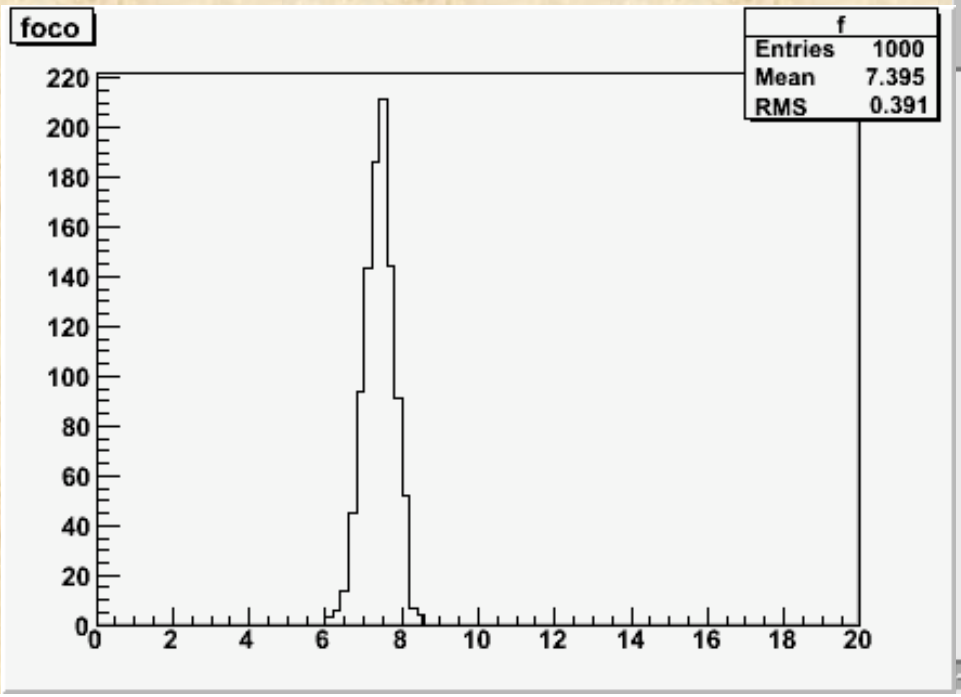
Exemplo: Propagação de incertezas com Monte Carlo

- **No ROOT digite:**

root [0] .L MC_lente.C

root [1] MC_lente(25.3, 0.2, 10.5, 0.8, 1000)
(float)3.910020885171509e-01

root [2] hist->Draw()



MC_lente.C - /Volumes/Data/Users/suaide/cursoRoot/

File Edit Search Preferences Shell Macro Windows Help

```
1 TH1F *hist;
2 float MC_lente(float o, float so, float i, float si, int N)
3 {
4   hist = new TH1F("f","foco",100,0,20);
5
6   TRandom *r = new TRandom();
7   for(int k = 0; k<N; k++)
8   {
9     float I = r->Gaus(i,si);
10    float O = r->Gaus(o,so);
11    float F = I*O/(I+O);
12    hist->Fill(F);
13  }
14  float RMS =hist->GetRMS();
15  return RMS;
16 }
17
```

Monte Carlo com TF1

- Eu sei a distribuição de probabilidade

```
void MC_TF1()
{
  TF1 *f = new TF1("f","[0]*exp([1]*x)+gaus(2)+
  f->SetParameter(0,40);
  f->SetParameter(1,-0.3);
  f->SetParameter(2,20);
  f->
  f->
  f->
  f->
  f->

  TCa
  c->
  c->
  f->Draw();

  TH1F* h = new TH1F("hist", "teste",100,0,10);
  for(int i=0; i<8000; i++) h->Fill(f->GetRando

  c->cd(2);
  h->Draw();
}
```

Funções pré-definidas.

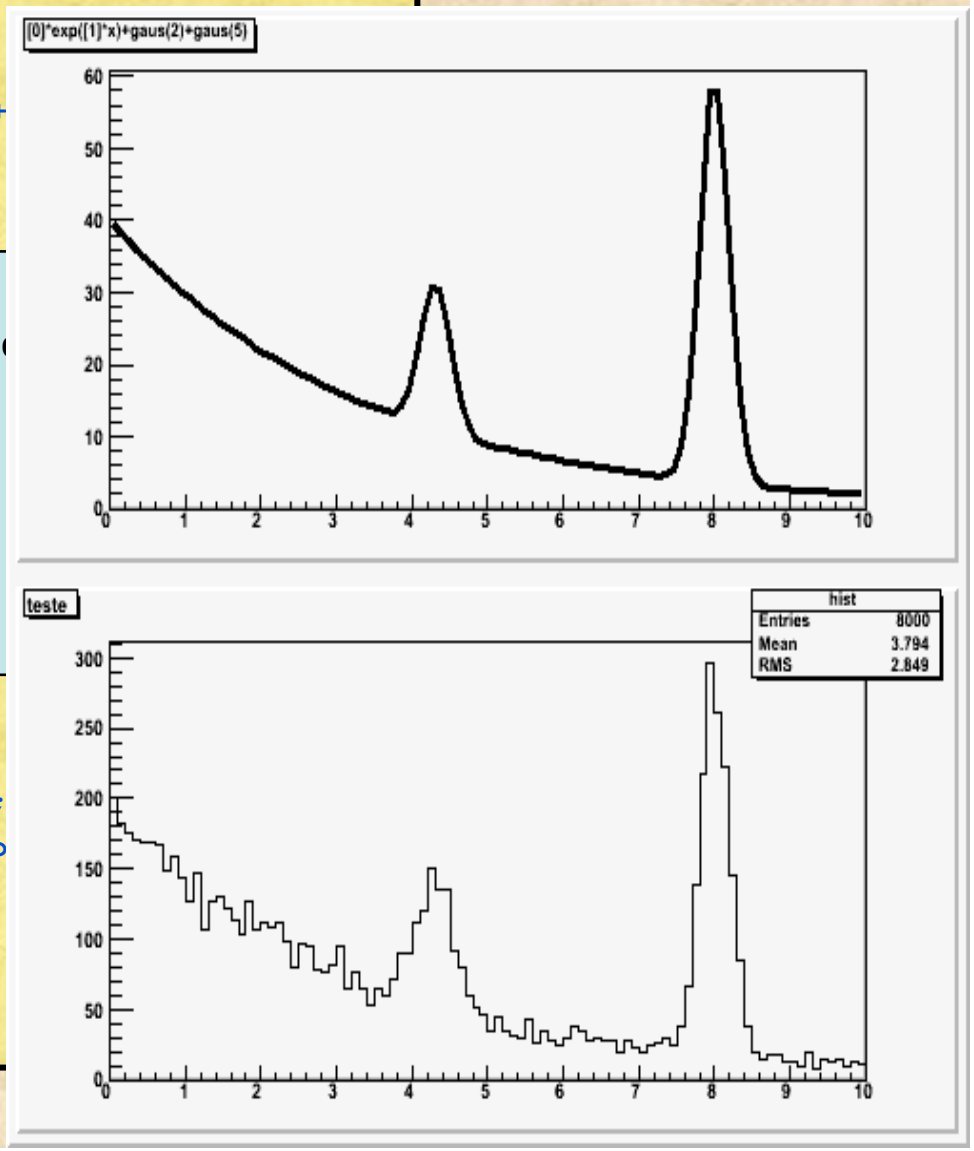
O Root possui algumas funções pré-defini tais como:

`gaus` – Gaussiana

`polN` – polinômio de grau N (N = 0...9)

`landau` – distribuição de Landau

`expo` – exp([0]+[1]*x)



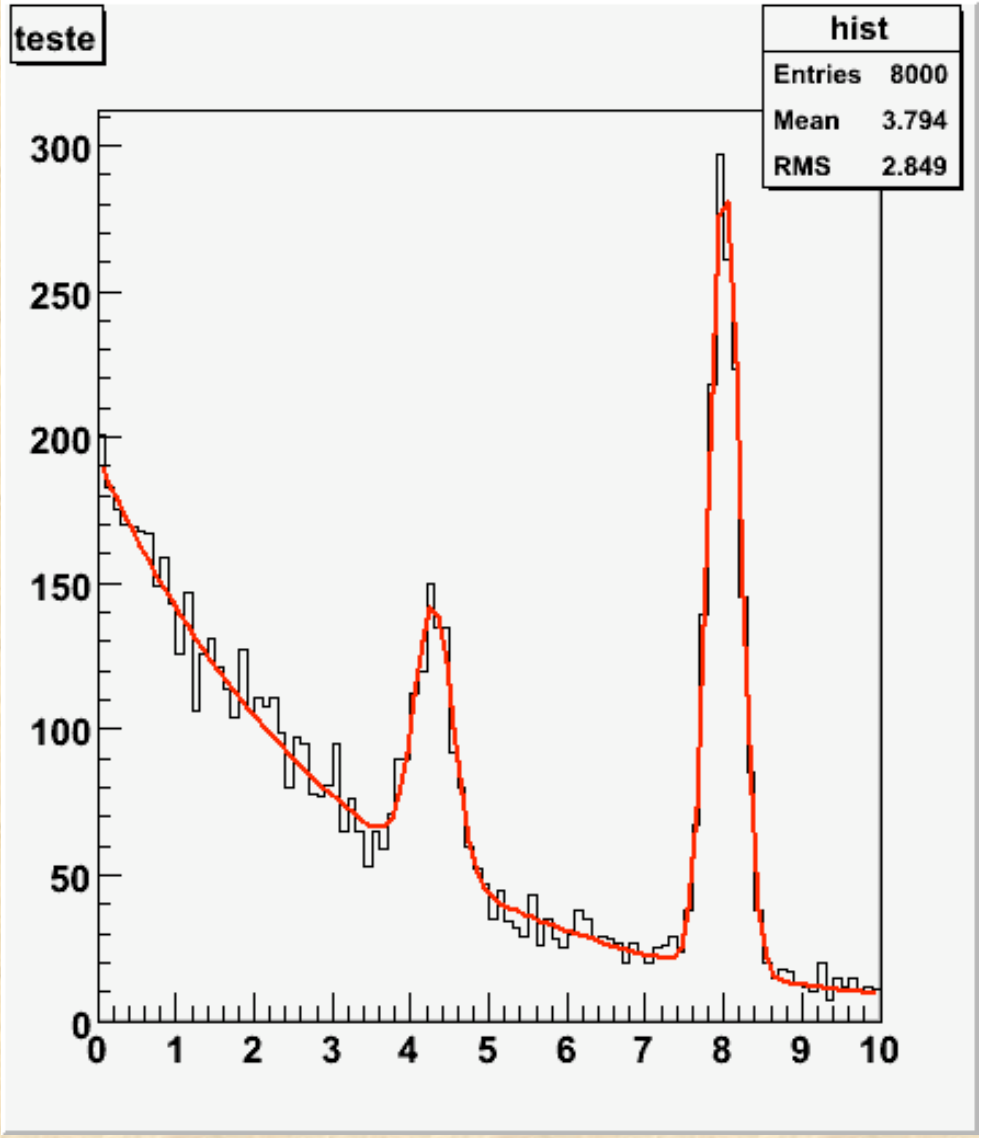
Outro exemplo de fit (com histogramas)

```
[0]*exp([1]*x)+gaus(2)+gaus(5)
```

- Vamos ajustar o histograma

```
new Tcanvas();
h->Draw();
f->SetLineColor(2);
f->SetLineWidth(2);
h->Fit(f);
```

- Como extrair informações da função ajustada que não seja pela tela?



Outro exemplo de fit (com histogramas)

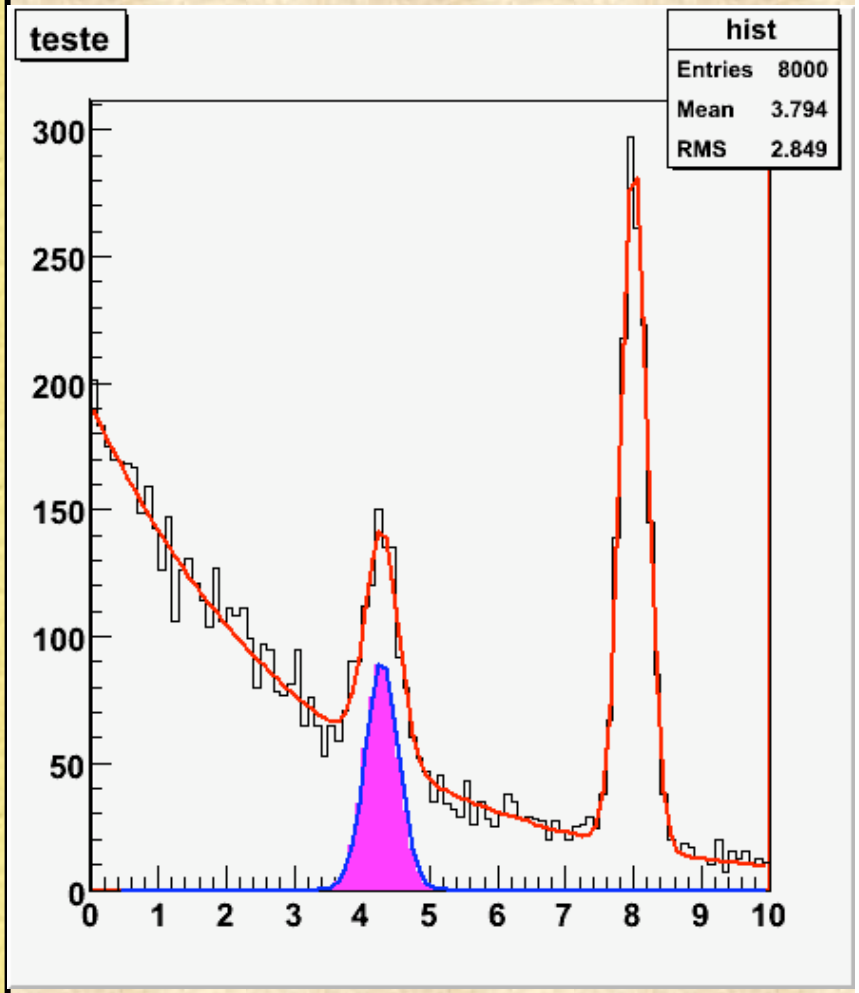
```
[0]*exp([1]*x)+gaus(2)+gaus(5)
```

- Qual o X^2_{red} do ajuste?

```
f->GetChisquare()/f->GetNDF();
1.36299
```

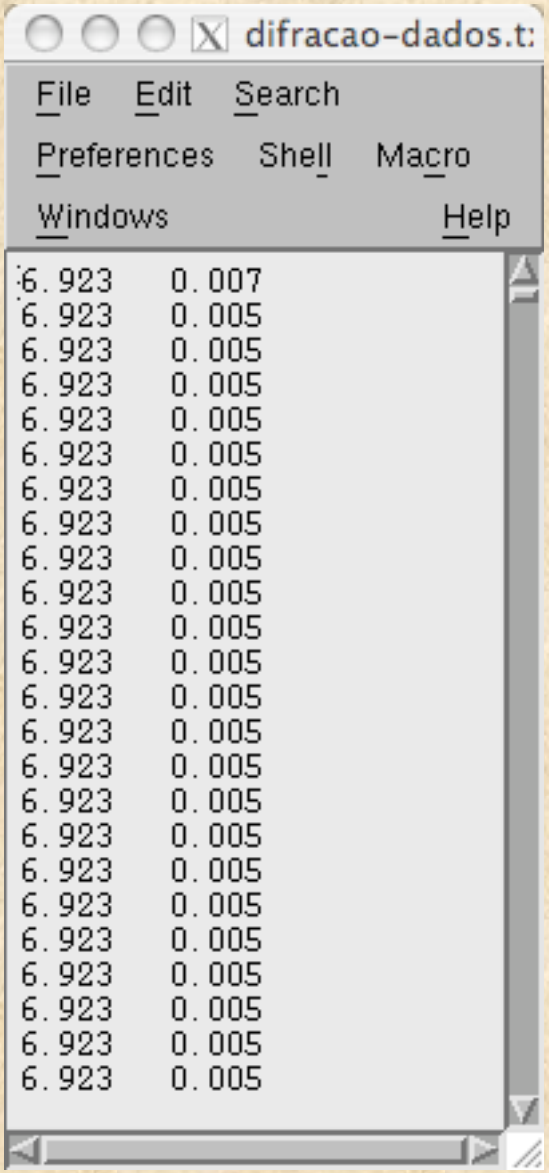
- Qual e a integral no primeiro pico?

```
TF1 *peak = new TF1("peak","gaus(0)",0,10);
peak->SetParameter(0,f->GetParameter(2));
peak->SetParameter(1,f->GetParameter(3));
peak->SetParameter(2,f->GetParameter(4));
peak->SetLineColor(4);
peak->SetLineWidth(2);
peak->SetFillColor(6);
peak->SetFillStyle(1000);
peak->Draw("same l f");

peak->Integral(0,10);
53.5103
```



Manipulando conteúdo de gráficos

- Como eu posso manipular os dados de um gráfico ?
  - GetX(), GetY(), GetEx(), GetEy()
- Exemplo:
  - Calculando incertezas e convertendo difração para radianos



Manipulando conteúdo de gráficos

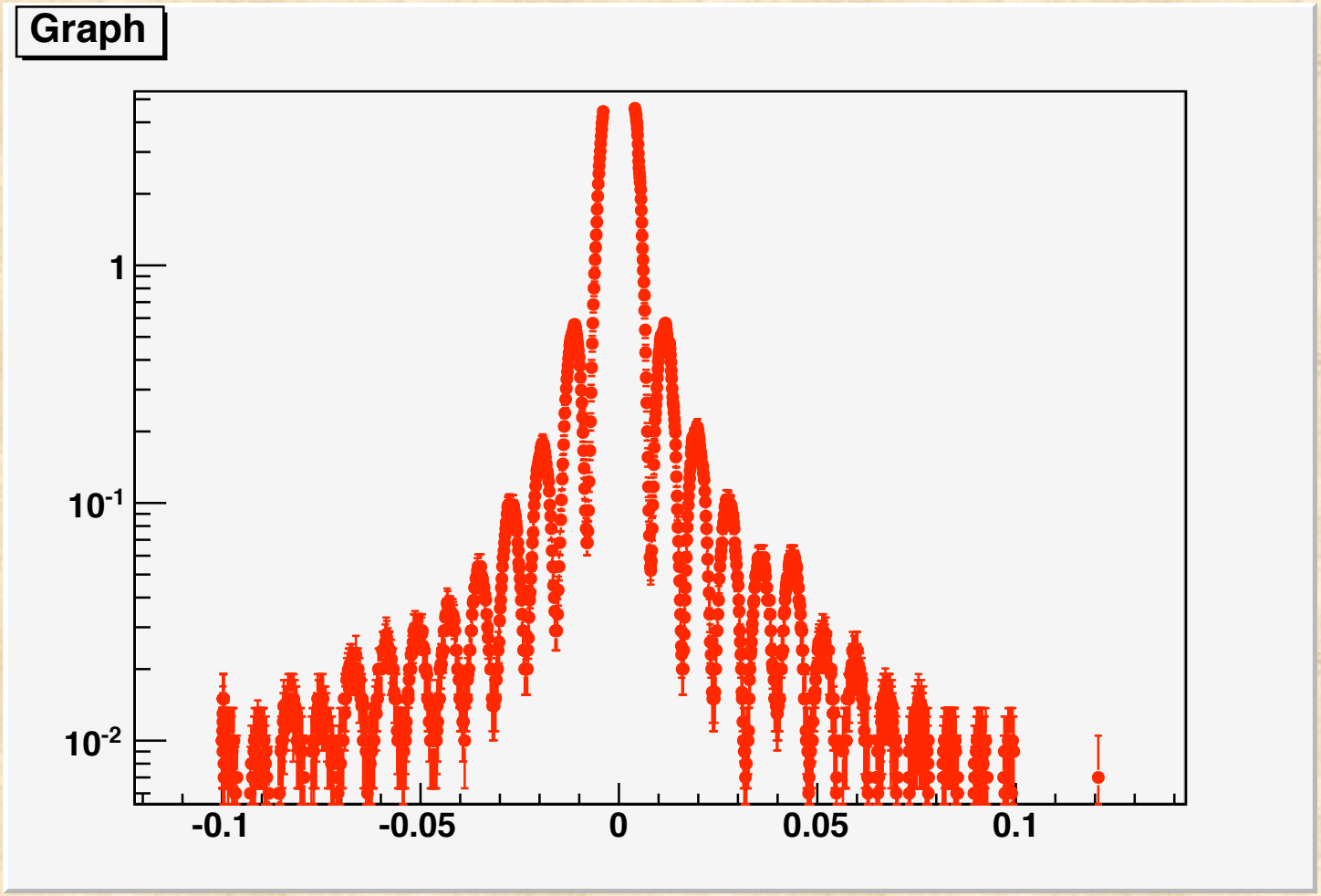
fit.C – /Users/suaide/Desktop/

File Edit Search Preferences Shell Macro Windows Help

```
void fit()
{
  // primeiro le o arquivo de dados e faz o grafico experimental
  // o arquivo so contem x e y. Temos que calcular os erros na mao
  TGraphErrors* g = new TGraphErrors("difracao-dados.txt");

  // pega os vetores de dados do grafico para manipular os dados
  double *e = g->GetEY(); // pega o vetor de erros em y do grafico
  double *y = g->GetY();  // pega o vetor de  y do grafico
  double *x = g->GetX();  // pega o vetor de  x do grafico
  int np = g->GetN();     // pega o numero de pontos no grafico

  // loop sobre todos os pontos do grafico
  for(int i =0; i<np; i++)
  {
    e[i] = 0.07*y[i]+0.003;  // calcula erro-y = 7% de y + 0.003
    x[i] = x[i]*3.1415/180;  // converte o angulo para radianos
    // pode-se tambem fazer as correcoes de angulo neste loop se
    // os dados ainda nao foram corrigidos
  }

  // agora eu defino o integrador da funcao
  f = new TF1("f","[0]*(sin([1]*sin(x+[2]))/([1]*sin(x+[2])))^2",-0.0
```

Arquivo fit.C

Manipulando conteúdo de gráficos



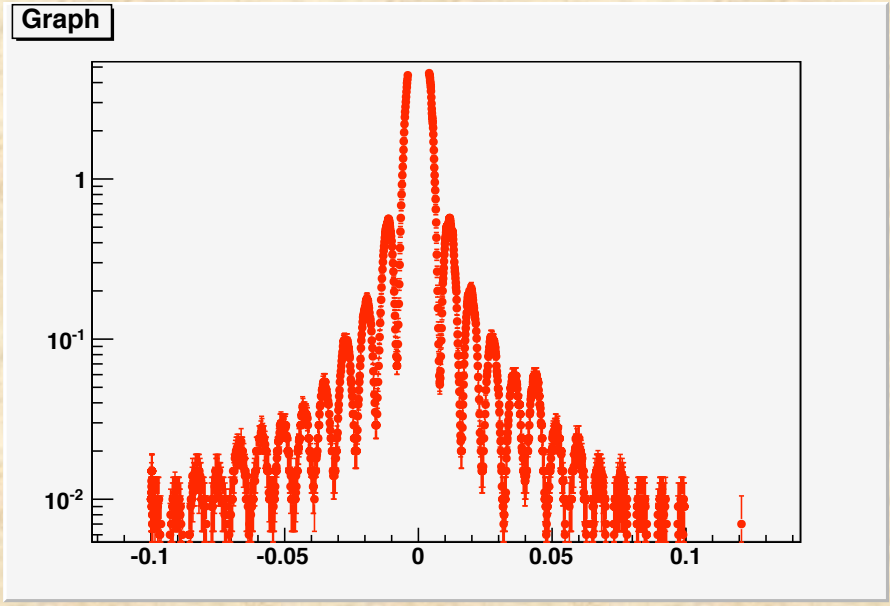
Arquivo fit.C

Fazendo ajustes mais sofisticados

- Se eu quiser fazer um ajuste de função mais complexo, que não seja possível escrever uma fórmula?
- Exemplo:
  - Ajuste contendo uma integral

$$I(\theta) = I_0 \int_{\theta-\delta/2}^{\theta+\delta/2} \left(\frac{\sin\left(\beta \sin(\phi+\theta_0)\right)}{\beta \sin(\phi+\theta_0)} \right)^2 d\phi + f$$

$$I(\theta) = [0] \int_{\theta-[3]/2}^{\theta+[3]/2} \left(\frac{\sin\left([1] \sin(\phi+[2])\right)}{[1] \sin(\phi+[2])} \right)^2 d\phi + [4]$$



Fazendo ajustes mais sofisticados

- Posso definir uma função no ROOT (TF1, 2, 3) cujo cálculo seja feita em uma função em c++
  - TF1(const char* name, void* fcn, Double_t xmin, Double_t xmax, Int_t npar)
- Primeira parte, definir a função a ser integrada

fit.C – /Users/suaide/Desktop/

File Edit Search Preferences Shell Macro Windows Help

```
// agora eu defino a funcao a ser integrada
f = new TF1("f","[0]*(sin([1]*sin(x+[2]))/([1]*sin(x+[2])))^2",-0.07, 0.07);

// agora eu defino a funcao integral
TF1 *f_int = new TF1("f_int", funcao, -0.07, 0.07, 5);

// inicializo os parametros da funcao
f_int->SetParameters(12,400,0,0.002,0);

// mando fazer o fit. Ver site do root para saber o que siginifcam as letras
g->Fit(f_int,"R0");
```

Fazendo ajustes mais sofisticados

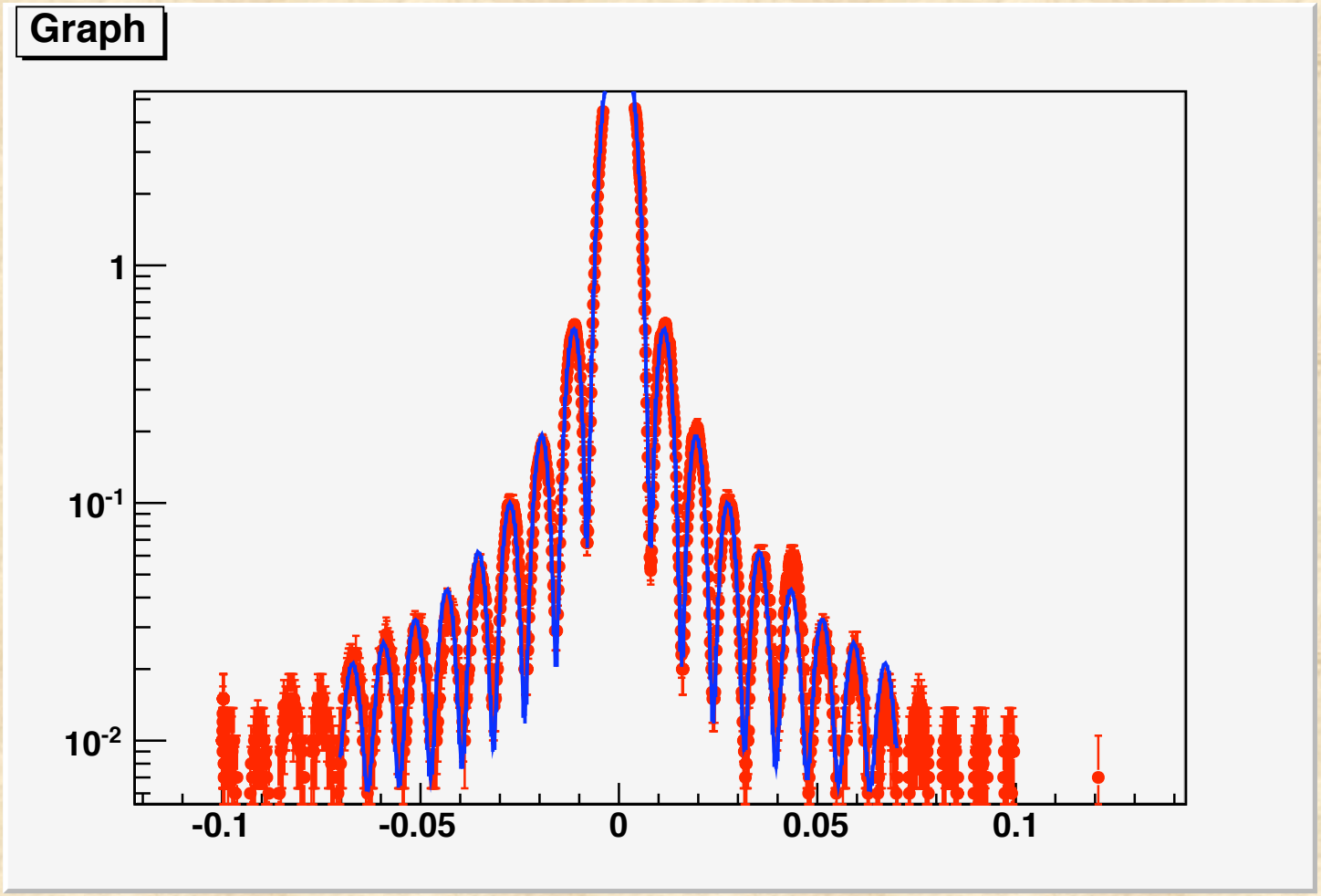
- **Mas quem é 'funcao'?**
  - **É um código em c++ que retorna o valor da integral**

fit.C – /Users/suaide/Desktop/

File Edit Search Preferences Shell Macro Windows Help

```
TF1 *f;

// aqui a gente define a funcao de integral
// como nao da para definir como uma formula
// simples, precisamos defini-la como funcao
// o vetor X contem as variaveis x,y,z, caso a
// funcao tenha mais de uma dimensao. o vetor
// par contem os parametros da funcao

// par[0] = I0
// par[1] = pi*d/lambda
// par[2] = theta0
// par[3] = largura do colimador
// par[4] = luz de fundo

double funcao(double *X, double *par)
{
  double x = X[0];
  f->SetParameters(par); // tranfere parametros para a funcao f
  double valor = f->Integral(x-par[3]/2,x+par[3]/2)/par[3]+par[4];
  return valor;
}
```

Fazendo ajustes mais sofisticados



Arquivo fit.C

Cálculo vetorial no ROOT

- **O Root possui classes para cálculos vetoriais em Física (ver http://root.cern.ch)**

Several documents describing these classes are listed below:

- The main histogram class is documented in class TGeoManager.
- The Chapter about the Physics Vectors classes in the Users Guide

TFeldmanCousins	calculate the CL upper limit using the Feldman-Cousins method
TGenPhaseSpace	Simple Phase Space Generator
TLorentzRotation	Lorentz transformations including boosts and rotations
TLorentzVector	A four vector with (-,-,-,+) metric
TQuaternion	a quaternion class
TRobustEstimator	Minimum Covariance Determinant Estimator
TRolke	calculate confidence limits using the Rolke method
TRotation	Rotations of TVector3 objects
TVector2	A 2D physics vector
TVector3	A 3D physics vector

Alguns exemplos de vetores

No prompt do ROOT, digite:

```
root [0] TVector3 A(3,2,6)
root [1] A.Print()
TVector3 A 3D physics vector (x,y,z)=(3.000000,2.000000,6.000000)
(rho,theta,phi)=(7.000000,31.002719,33.690068)

root [2] TVector3 B(2,7,8)
root [3] B.Print()
TVector3 A 3D physics vector (x,y,z)=(2.000000,7.000000,8.000000)
(rho,theta,phi)=(10.816654,42.302625,74.054604)

root [4] TVector3 C = A-B
root [5] C.Print()
TVector3 A 3D physics vector (x,y,z)=(1.000000,-5.000000,-2.000000)
(rho,theta,phi)=(5.477226,111.416714,-78.690068)

root [6] A.Dot(B)
(const Double_t)6.80000000000000000e+01

root [7] TVector3 D=A.Cross(B)
root [8] D.Print()
TVector3 A 3D physics vector (x,y,z)=(-26.000000,-12.000000,17.000000)
(rho,theta,phi)=(33.301652,59.303846,-155.224859)
```

Resumindo

- Vimos algumas funcionalidades do ROOT
  - Ajuste de funções
  - Histogramas em 1 e 2 dimensões
  - TLegend
  - Vetores (TVector2 e TVector3)

- O ROOT é muito versátil para análise de dados e simulações. Usem!!!