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

ajuste.C (modified) - /Users/suaide/

File Edit Search Preferences Shell Macro Windows Help

```
void ajuste()
{

  //
  // cria um grafico a partir de dados em um arquivo.
  //
  TGraphErrors *g = new TGraphErrors("dados.txt","%lg %lg %lg %lg");
  g->Draw("AP");
  g->SetLineColor(2);
  g->SetMarkerColor(2);
  g->SetMarkerStyle(20);
  g->GetXaxis()->SetTitle("T [K]");
  g->GetYaxis()->SetTitle("P [W]");
  g->SetTitle("Potencia em uma lampada de automovel");
```

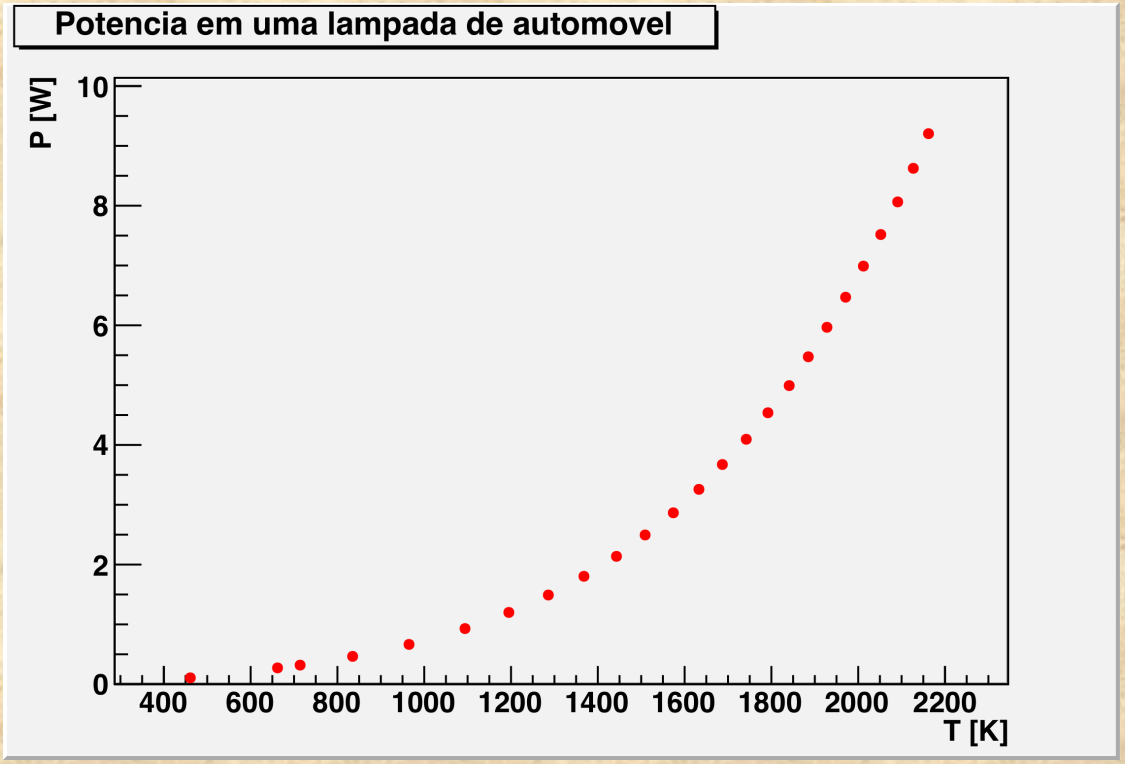
dados.txt (modified) - /Users/suaide/

File Edit Search Preferences Shell Macro Windows Help

ajuste.C | *dados.txt

461.0	0.1064	2.7	0.0006
662	0.2728	4	0.0012
714	0.3192	4	0.0014
835	0.4653	5	0.0018
965	0.6650	5	0.0024
1094	0.9300	7	0.0051
1195	1.2000	7	0.0058
1286	1.4910	8	0.0066
1368	1.8040	8	0.0073
1443	2.1375	9	0.0081
1509	2.4950	9	0.0088
1574	2.8655	9	0.0096
1633	3.258	10	0.010
1687	3.673	8	0.011

Gráficos e funções

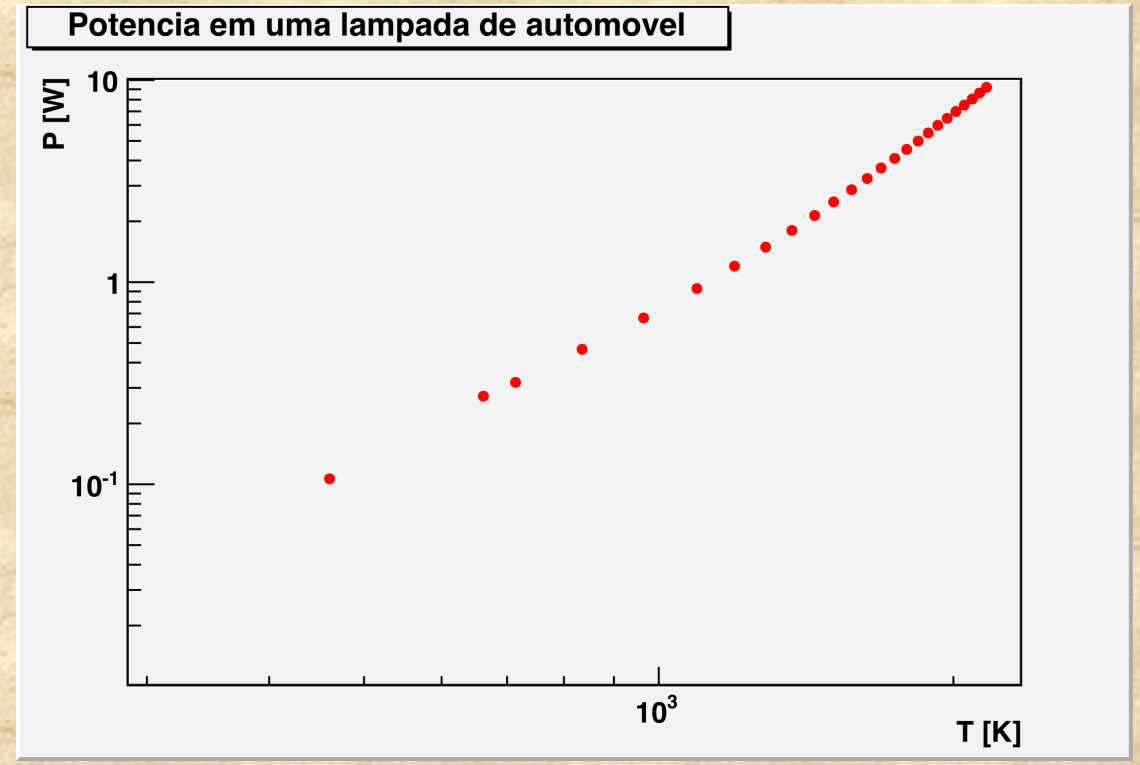


Vamos acrescentar as seguintes linhas ao programa

```
gPad->SetLogy();
gPad->SetLogx();
```

A variável gPad corresponde SEMPRE à tela gráfica ativa e pode ser utilizada para alterar suas características

Gráficos e funções

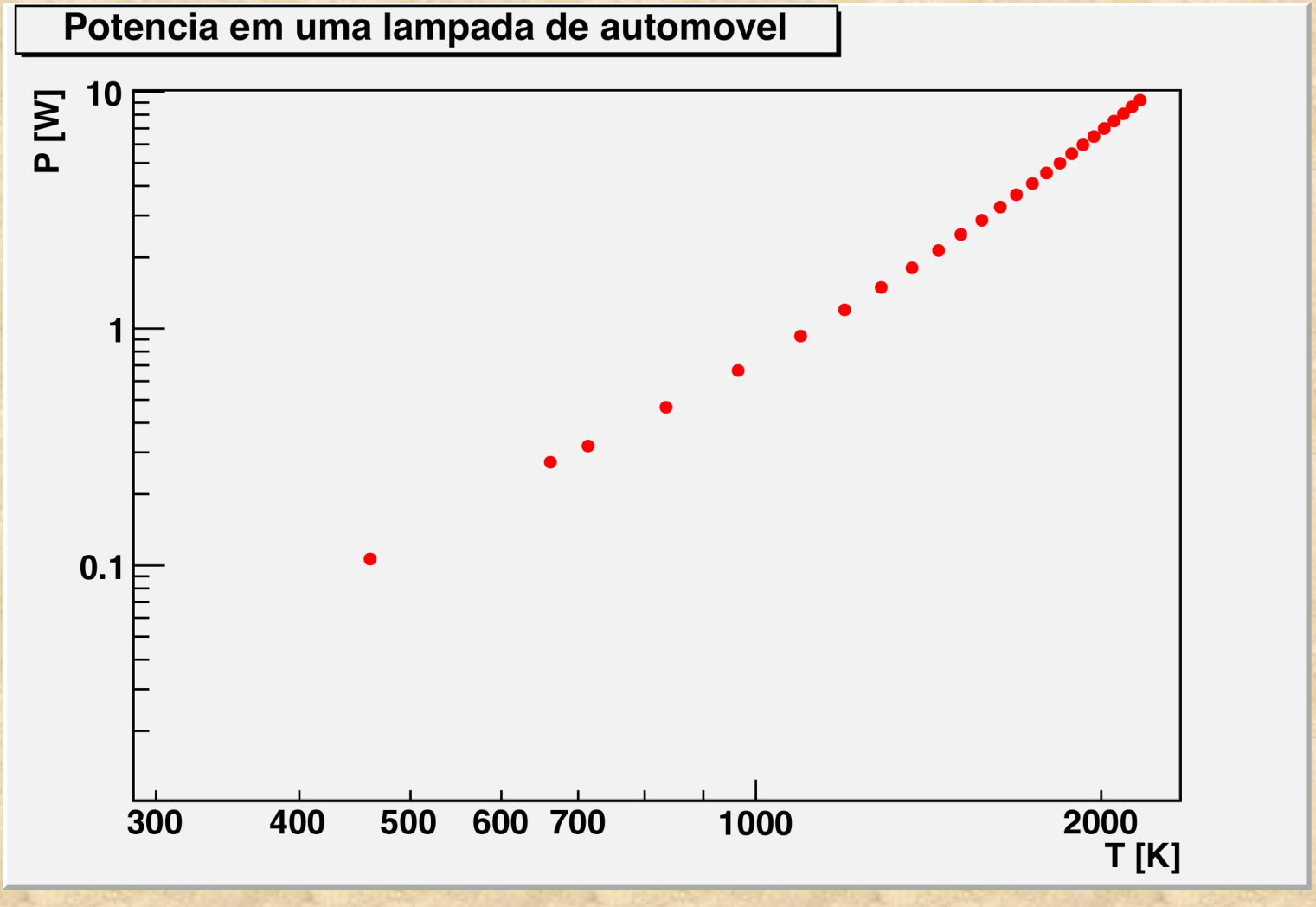


Só que as escalas não estão bem distribuídas. Faltam valores no eixo-x e a notação exponencial não é muito agradável, nesse caso

Vamos acrescentar as seguintes linhas ao programa

```
g->GetXaxis()->SetMoreLogLabels();
   g->GetXaxis()->SetNoExponent();
   g->GetYaxis()->SetNoExponent();
```

Gráficos e funções



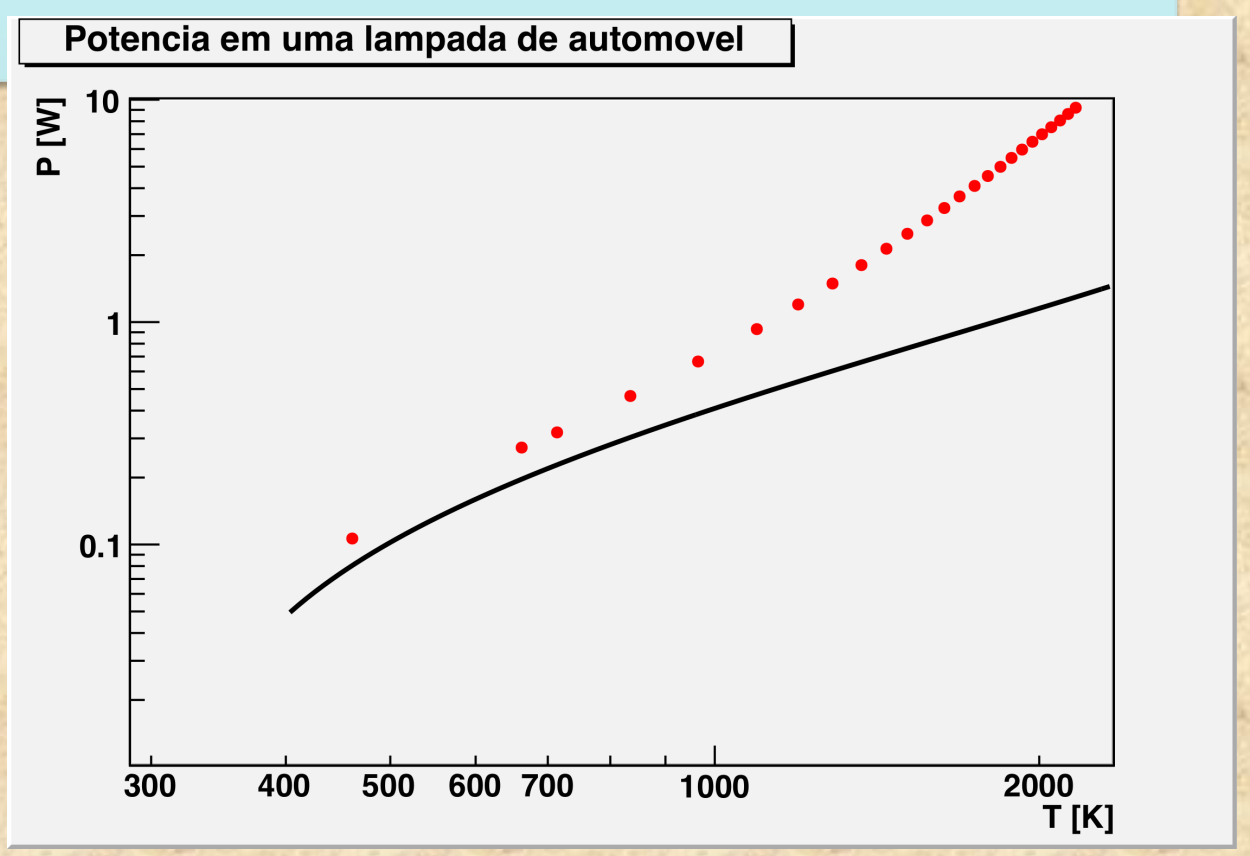
Gráficos e funções

$$P(T) = A\left(T - T_0\right)^{\alpha} + B\left(T^{\beta} - T_0^{\beta}\right)$$

Vamos acrescentar as seguintes linhas ao programa

```
TF1 *f = new TF1("teoria",
                 "[0]*(x-300)^[1]+[2]*(x^[3]-300^[3])",
                 400,3000);
f->SetParameters(3e-4, 1.1, 5e-15, 4);
f->Draw("sameL");
```

O same significa superposto e o L, desenha uma linha

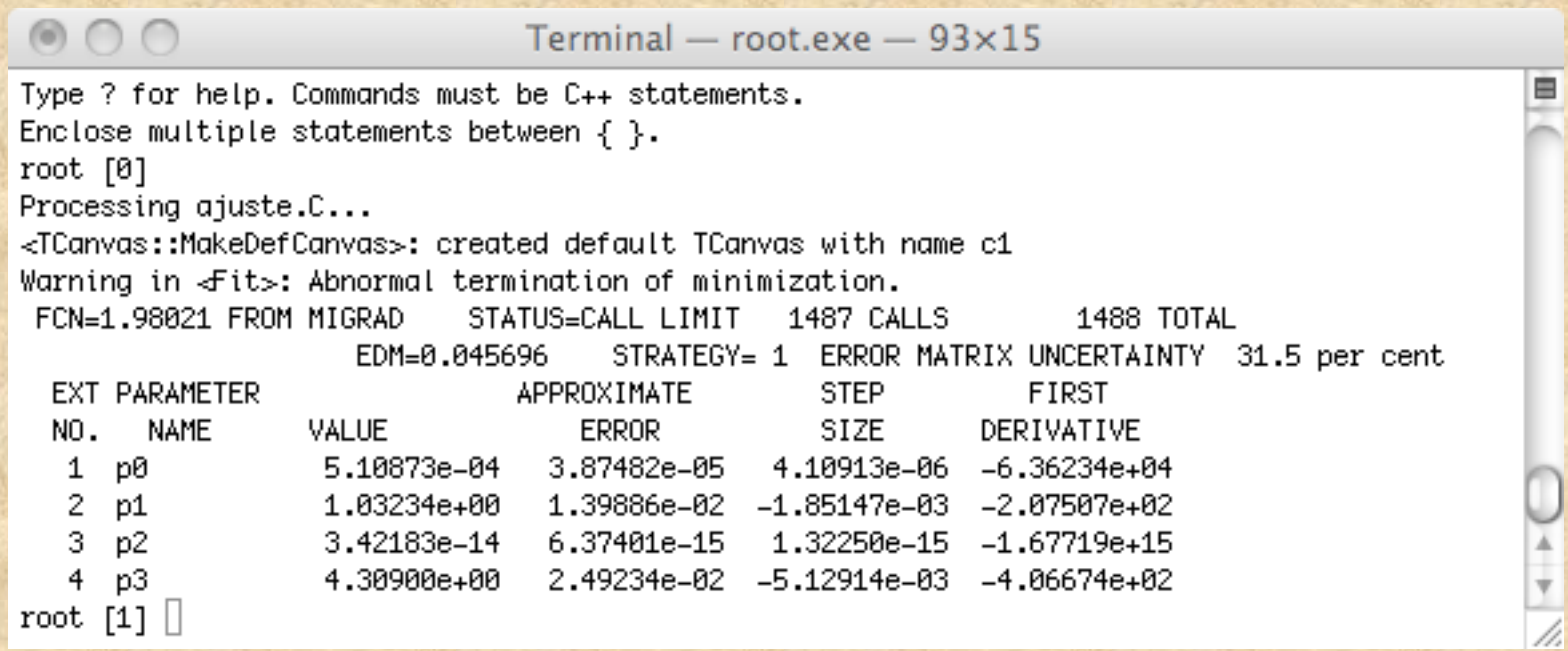


Ajustando funções

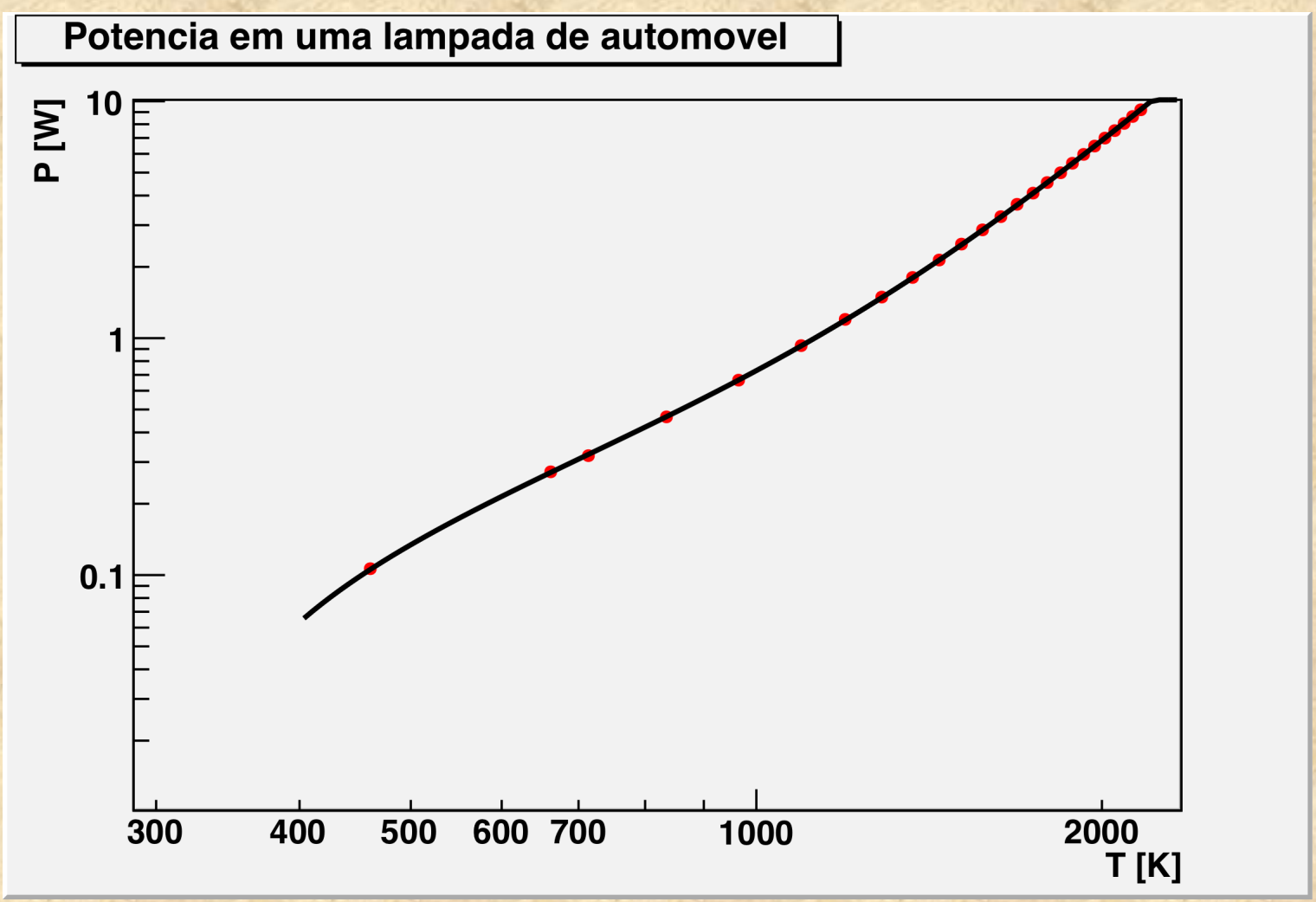
Para ajustar a função aos dados basta acrescentar o seguinte linha ao programa:

```
g->Fit(f,"RN");
```

O R significa para usar os limites especificados na função e o N significa para não associar a função ao gráfico.



Ajustando funções



Ajustando funções

Obtendo informações do ajuste

```
cout <<"Chi2 = "<<f->GetChisquare()<<endl;
cout <<"NDF =  "<<f->GetNDF()<<endl;
```

Obtendo matriz de covariância

```
TMatrixD C(4,4);
gMinuit->mnemat(C.GetMatrixArray(),4);
C.Print();
```

Terminal — root.exe — 93×14

```
   4  p3           4.30900e+00   2.49234e-02  -5.12914e-03  -4.06674e+02
Chi2 = 1.98021
NDF =  21

4x4 matrix is as follows

     |      0    |      1    |      2    |      3    |
--------------------------------------------------------
   0 |  1.501e-09  -5.366e-07   5.622e-20  -1.847e-07
   1 | -5.366e-07   0.0001957  -2.932e-17   0.0001014
   2 |  5.622e-20  -2.932e-17   4.063e-29  -1.586e-16
   3 | -1.847e-07   0.0001014  -1.586e-16   0.0006212

root [1]
```

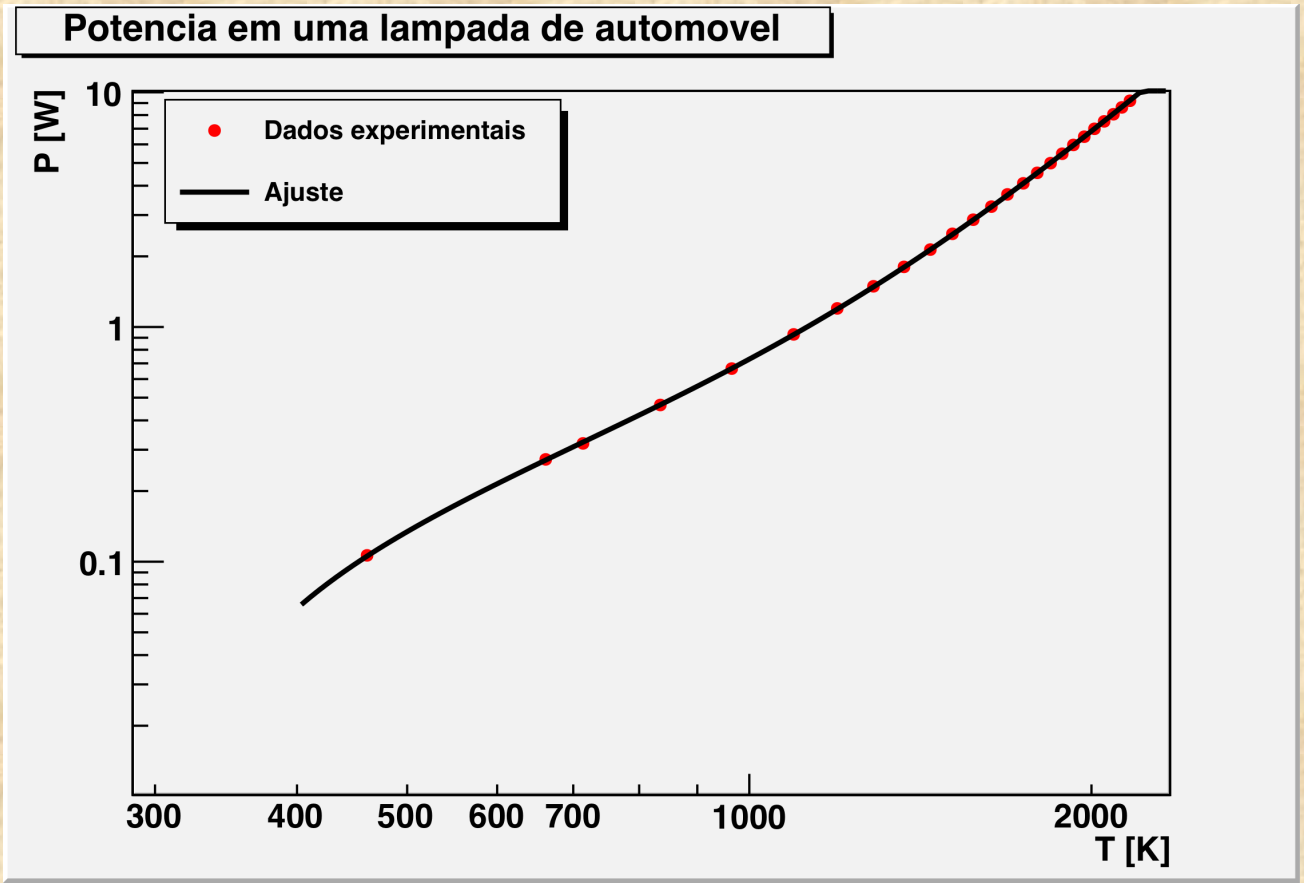
TLegend

- 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

Criando legenda no gráfico

```
TLegend *L = new TLegend(0.125,0.75,0.43,0.89);
L->AddEntry(g,"Dados experimentais","p");
L->AddEntry(f,"Ajuste","l");
L->Draw();
```



Manipulando conteúdo de gráficos

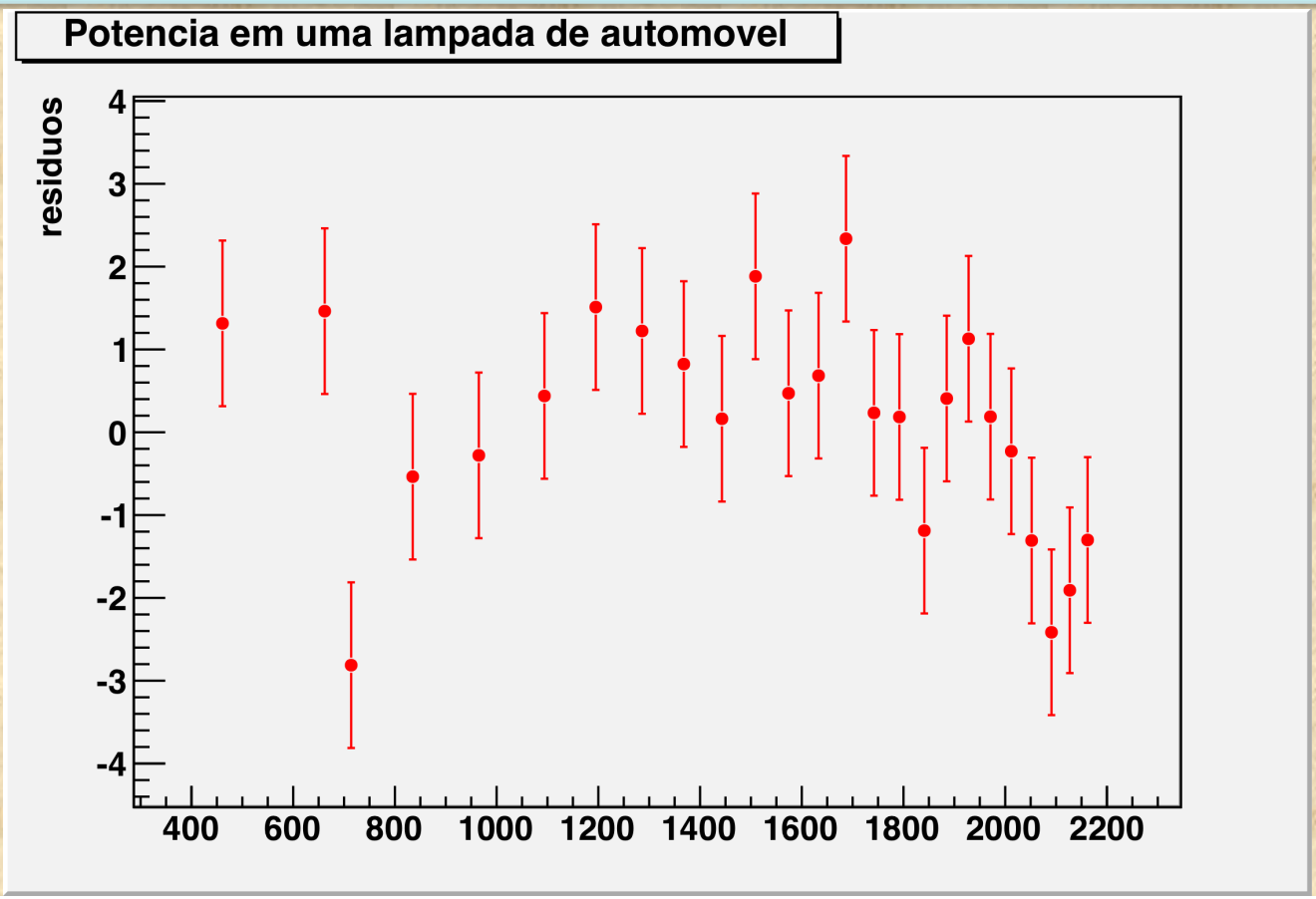
- Como eu posso manipular os dados de um gráfico ?
  - GetX(), GetY(), GetEX(), GetEY()
- Exemplo:
  - Gráfico de resíduos

```
TGraphErrors* residuo(TGraphErrors* g, TF1* f)
{
  int n = g->GetN();
  TGraphErrors *r = new TGraphErrors(*g);
  double *x = r->GetX();
  double *y = r->GetY();
  double *e = r->GetEY();
  for(int i = 0; i<n; i++)
  {
    y[i] = (y[i]-f(x[i]))/e[i];
    e[i] = 1;
  }
  r->GetYaxis()->SetTitle("residuos");
  return r;
}
```

Manipulando conteúdo de gráficos

Vamos acrescentar ao programa a função anterior e as seguintes linhas

```
new TCanvas();
TGraphErrors *res = residuo(g,f);
res->Draw("AP");
```



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

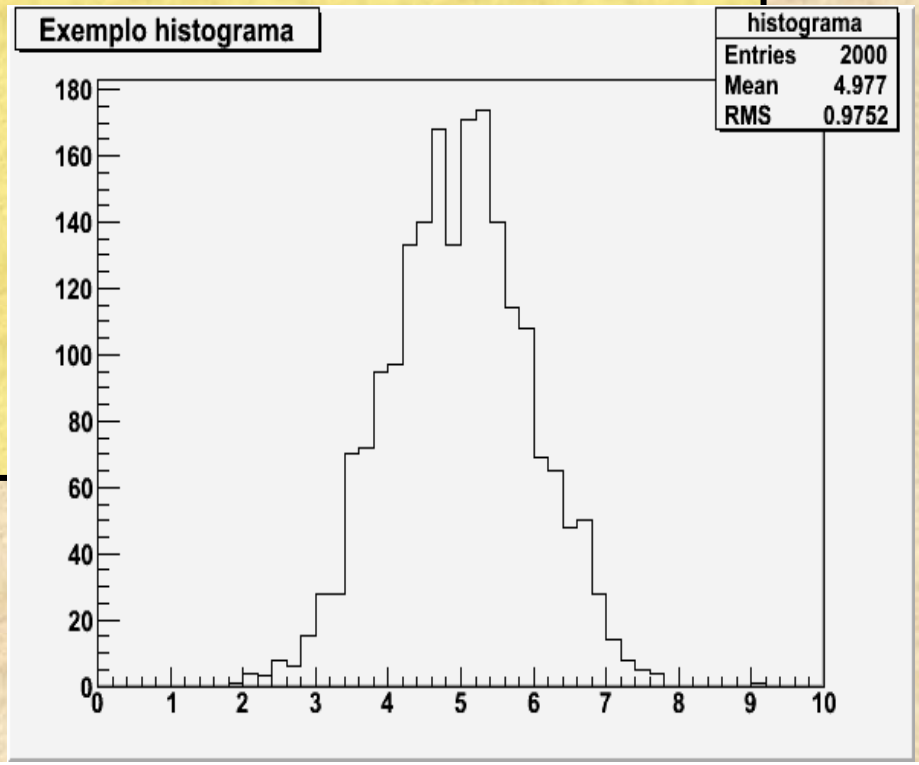
Histogramas de 1 dimensão

- 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

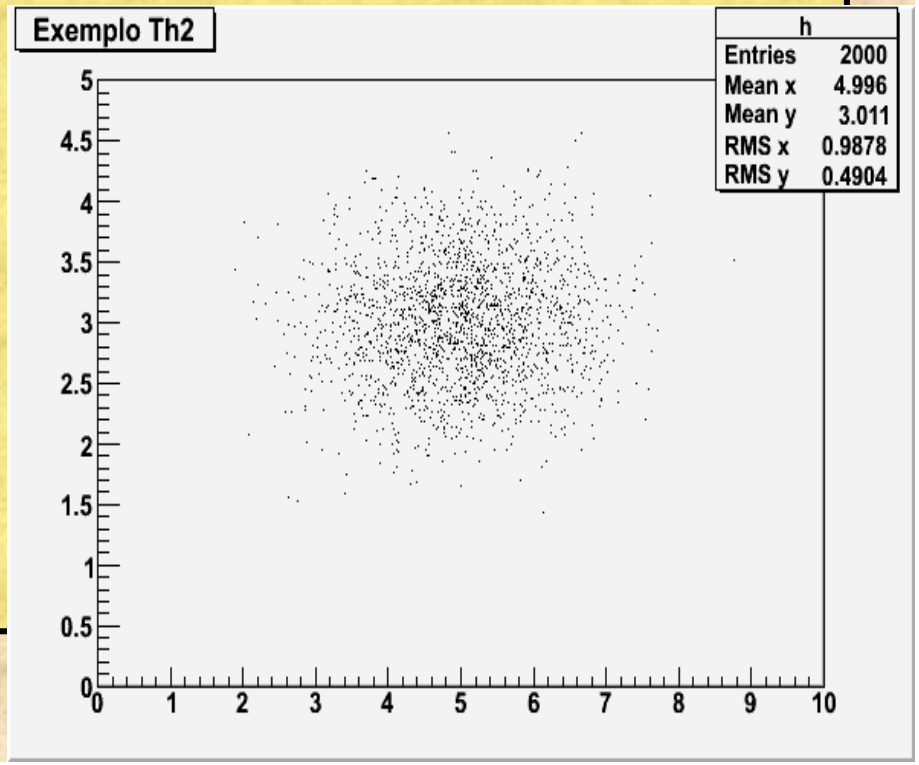
root [0] .x exemplo_TH1.C



Histogramas de 2 dimensões

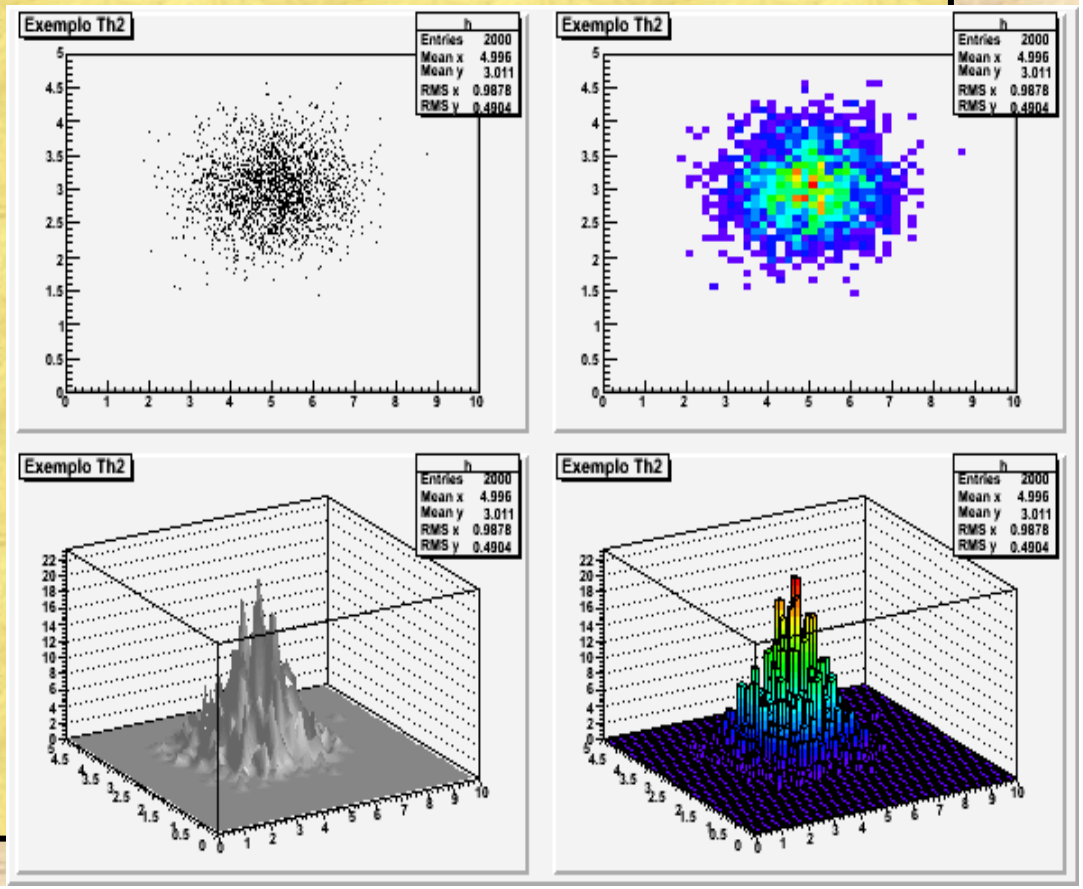
- **Muito similar ao TH1**
  - TH2F *h = new TH2F("nome","título", NbinsX, Xmin, Xmax, NBinsY, Ymin, Ymax);
  - TH2F h ("nome","título", NbinsX, Xmin, Xmax, NbinsY, Ymin,Ymax);

```
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
```

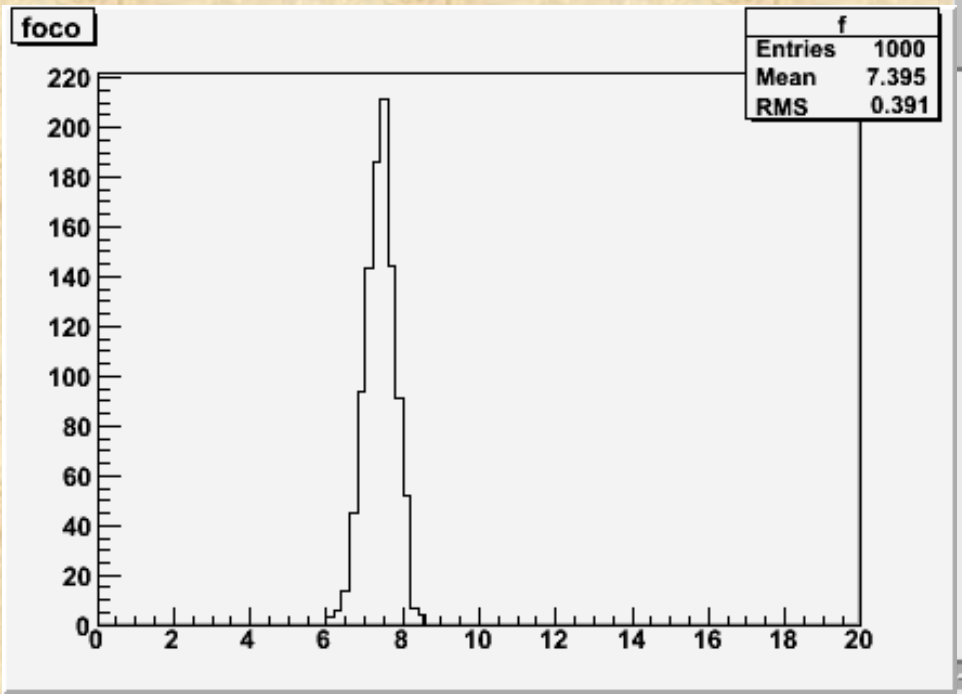
Exemplo: Propagação de incertezas com Monte Carlo

- **No ROOT digite:**

root [0] .L MC_lente.C

root [1] MC_lente(25.3, 0.2, 10.5, 0.8, 1000)
(float)3.91002088785171509e-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.

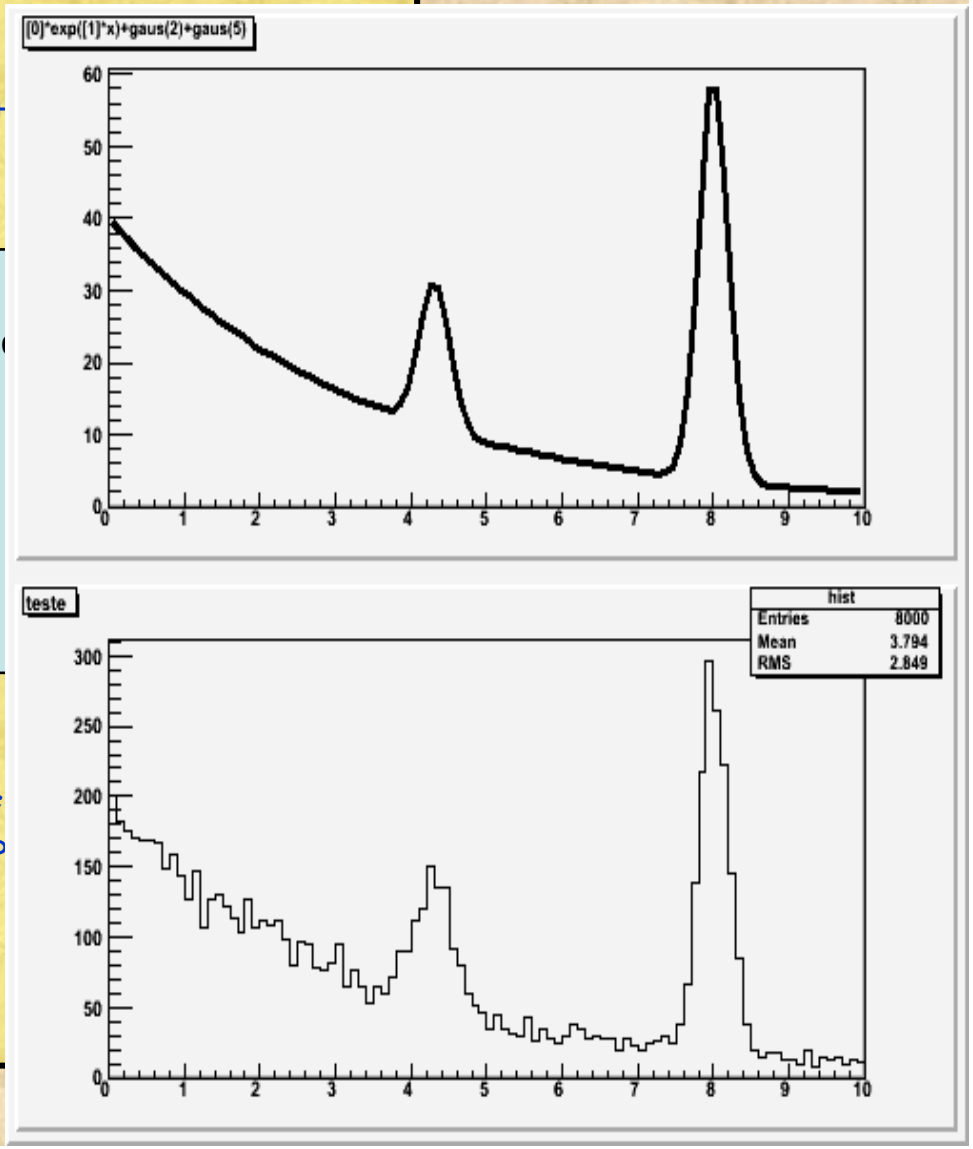
O Root possui algumas funções pré-defini tais como:

`gaus` – Gaussiana

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

`landau` – distribuição de Landau

`expo` – exp([0]+[1]*x)



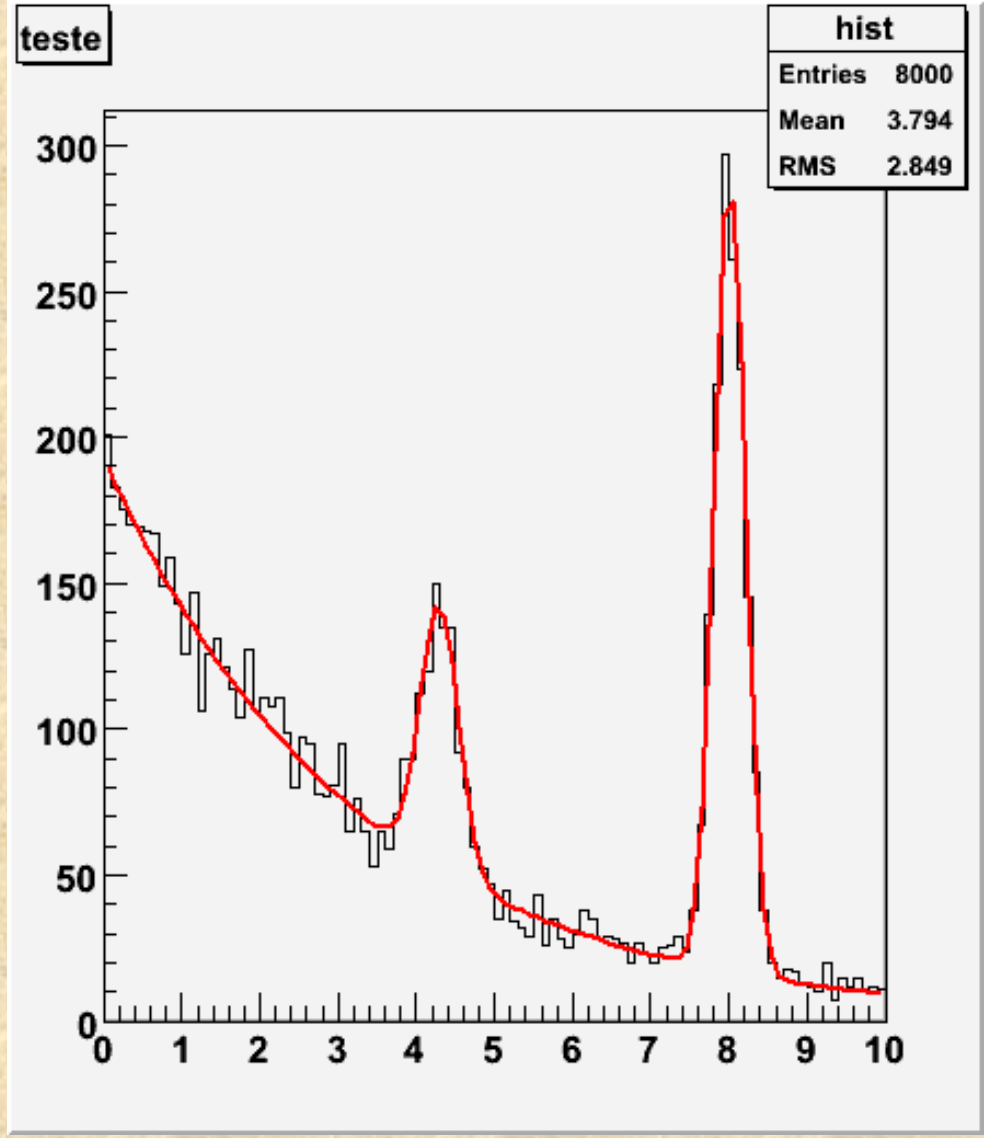
Outro exemplo de fit (com histogramas)

`[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)
```

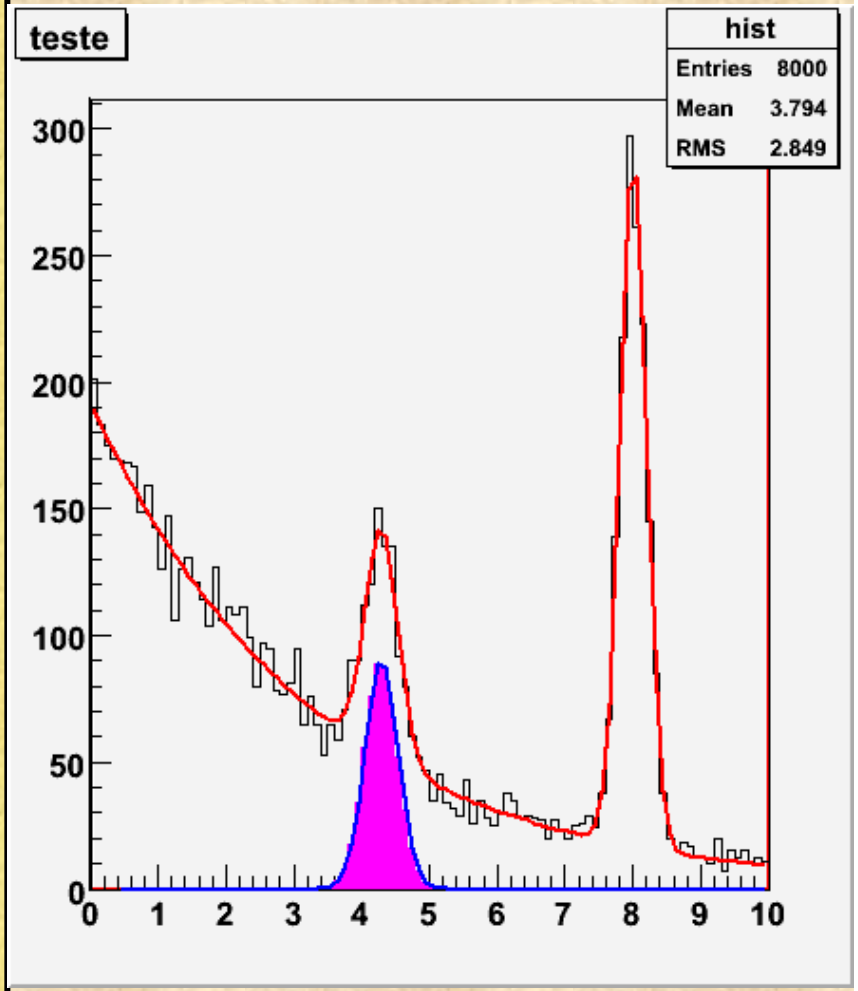
- Qual o X^2_{red} do ajuste?

```
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
```



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)6.8000000000000000e+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!!!