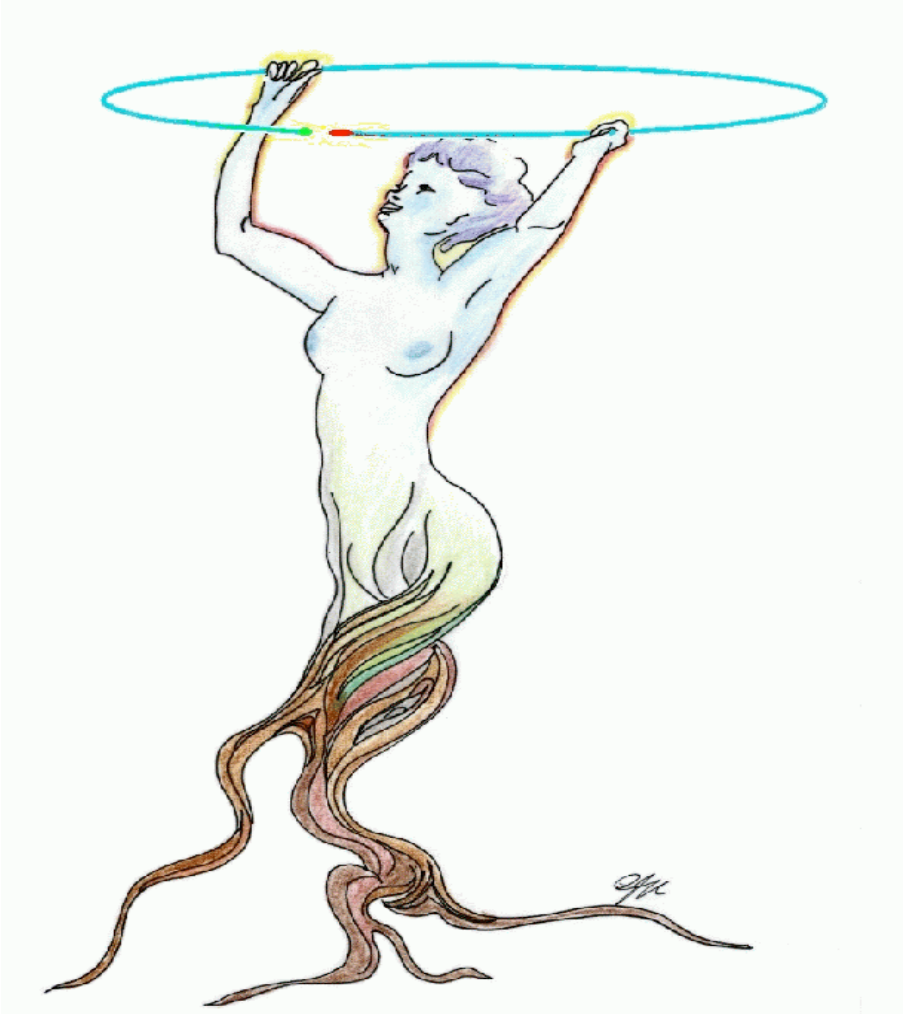
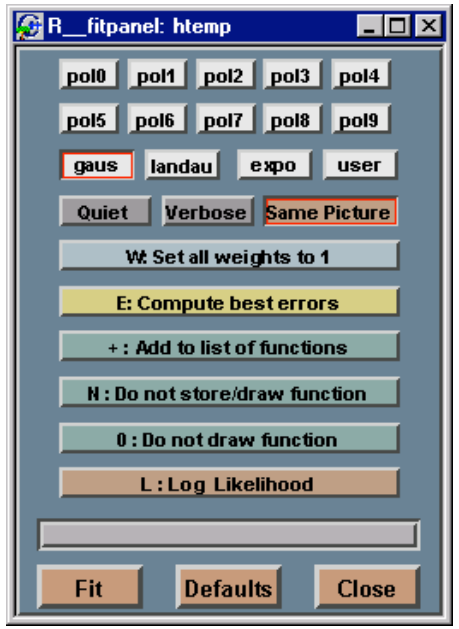
Formation ROOT pour débutants

Troisième Jour

Les fits



Les Fits



- On a vu comment faire un fit sur un spectre avec l'interface graphique...
- Comment fitter à partir de la ligne de commande ? ou dans un script ?

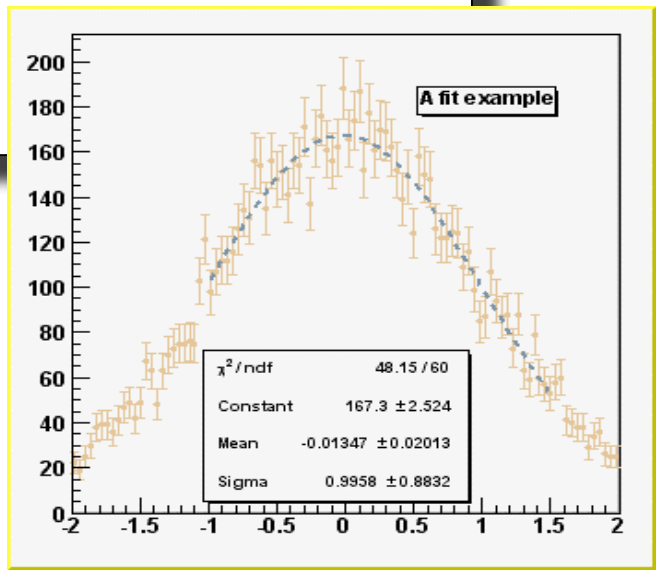
```
gStyle->SetOptFit(kTRUE)
TH1F *h = new TH1F("hg","Un exemple de fit",100,-2,2)
h->FillRandom("gaus",10000)
h->Fit("gaus","V","E1",-1,1.5)
```

nom de la fonction

options du fit

options d'affichage du spectre

limites du fit

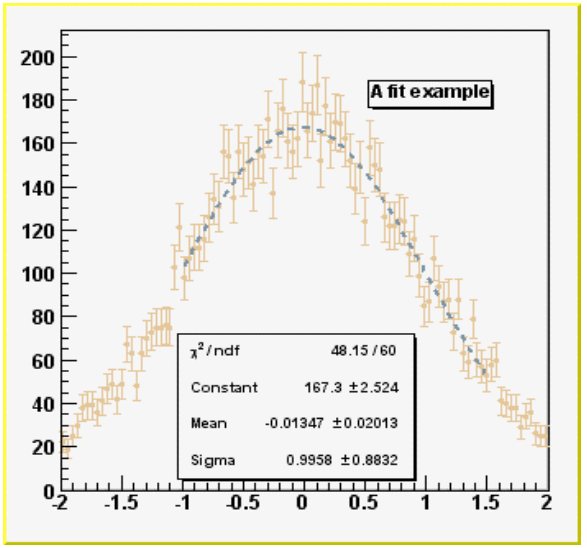


Les fits d'en bas…

Quelles fonctions pour le fit ?

- Les fonctions prédéfinies:
  - "gaus" `= p0*exp(-0.5*pow((x-p1)/p2),2)`
  - "expo" `= exp(p0+p1*x)`
  - "polN" `= p0 + p1*x + p2*pow(x,2) + p3*...`
  - "landau" (avis aux amateurs!)
- Comment accéder aux paramètres du fit ?

```
TF1 *gfit = (TF1 *)h->GetFunction("gaus")
gfit->GetParameter(0)
gfit->GetParameter(1) ...
gfit->GetParError(0) ...
Double_t par[3]
gfit->GetParameters(par)
```

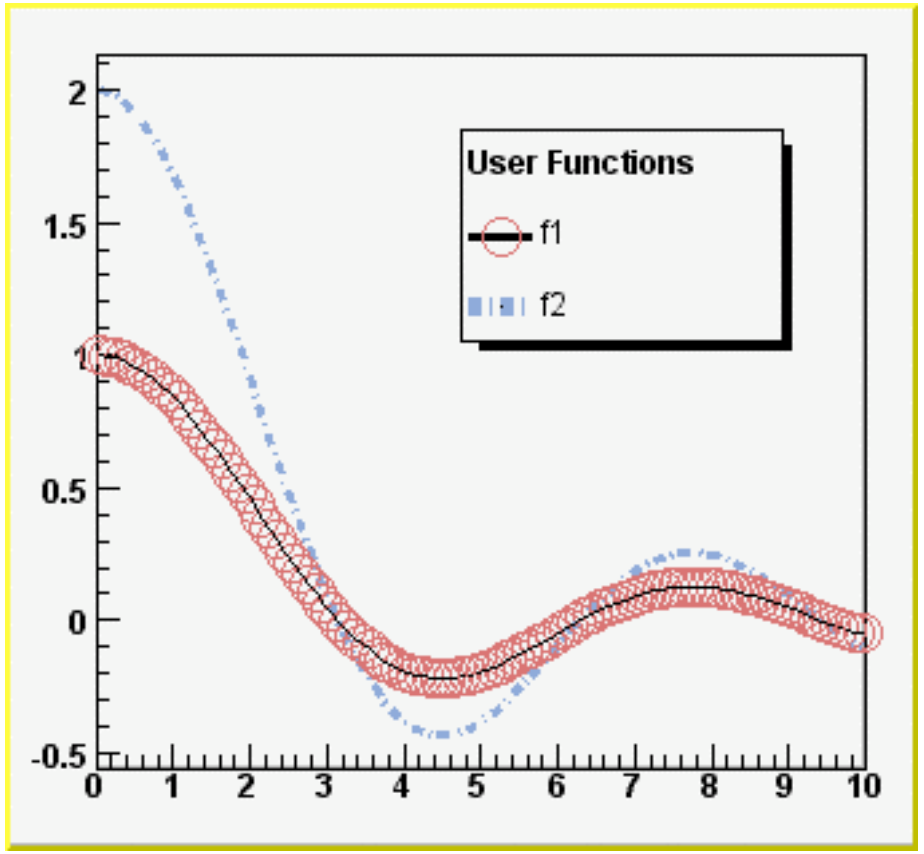


Définir ses propres fonctions

```
TF1 *fu = new TF1("f1", "sin(x)/x", 0, 10)
TF1 *fd = new TF1("f2", "f1 * 2", 0, 10)
fu->Draw()
fd->Draw("same")
```

C'est le nom qui compte!

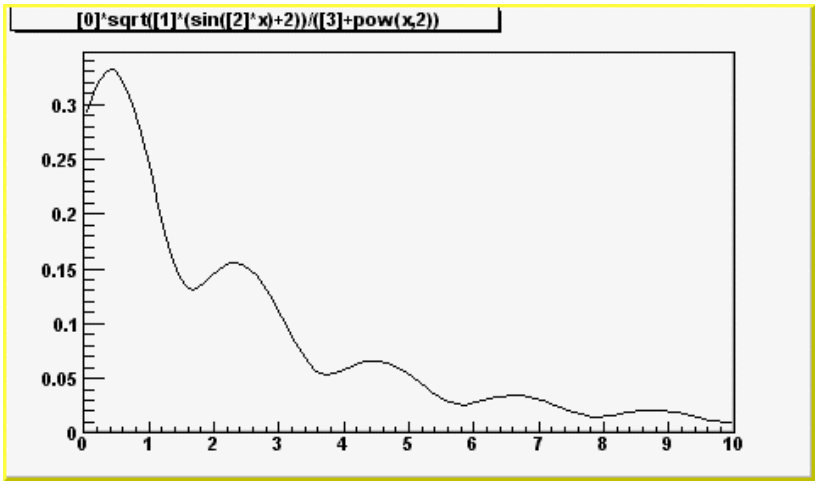
Et tout plein de combinaisons !



Introduire des paramètres

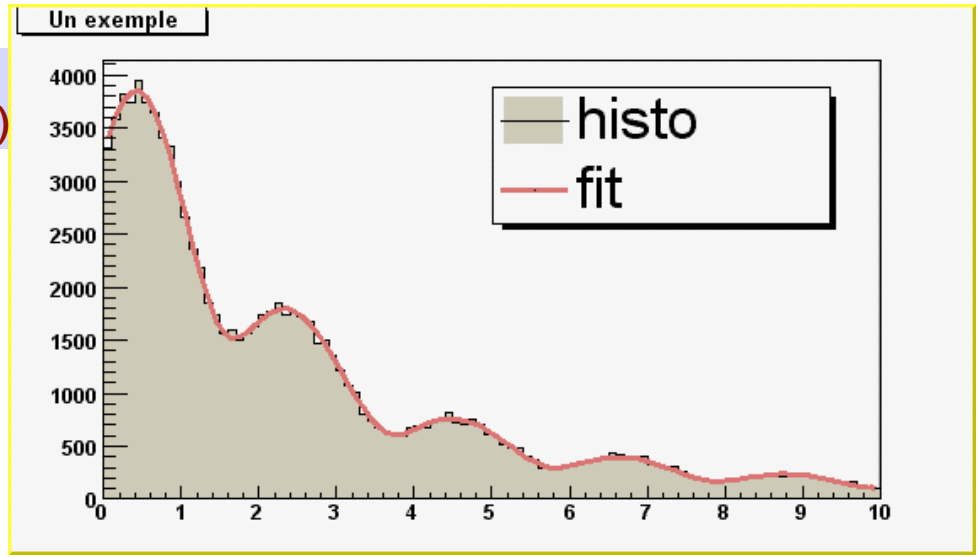
```
TF1 *ft = new TF1("f3", "[0]*sqrt([1]*(sin([2]*x)+2))
   /([3]+pow(x,2))", 0, 10)
ft->SetParameters(1,1,3,5)
ft->Draw()
```

indice	0	1	2	3
contenu	1	1	3	5




```
TH1F *hd = new TH1F("h2","Un exemple", 100, 0, 10)
hd->FillRandom("f3",100000)
ft->SetParameters
   (hd->GetMaximum(),1,2.8,6.)
hd->Fit("f3")
```

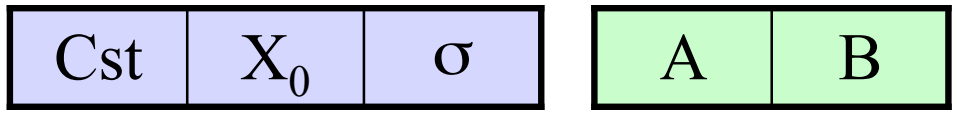
indice	0	1	2	3
contenu	`h2->GetMaximum()`	1	2.8	6



Mélange de fonctions

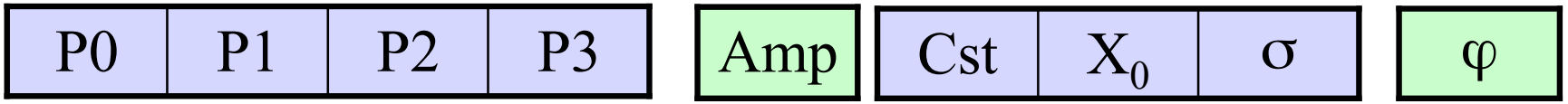
- On peut mélanger les fonctions de base

```
TF1 *fq=new TF1("f4","gaus(2)+expo(0)",0,10)
```



- Un autre exemple

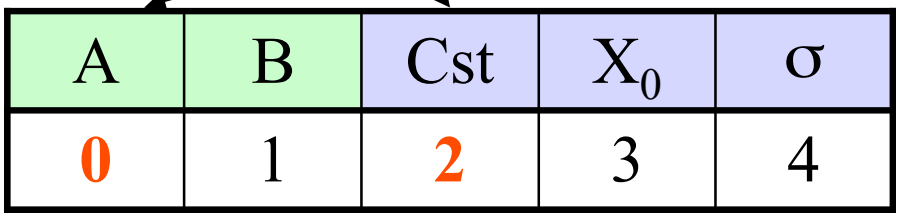
```
TF1 *fc=new TF1("f5","pol3(0)+[4]*sin(gaus(5)+[8])",0,10)
```



Mélange de fonctions

- On peut mélanger les fonctions de base

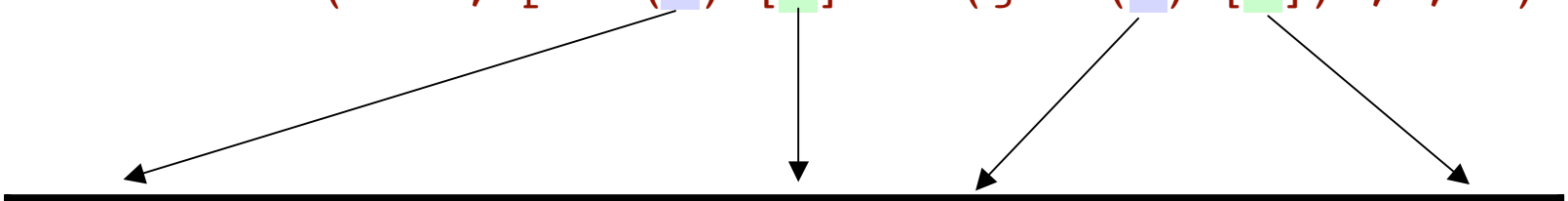
```
TF1 *fq=new TF1("f4","gaus(2)+expo(0)",0,10)
```



A	B	Cst	X_0	σ
0	1	**2**	3	4

- Un autre exemple

```
TF1 *fc=new TF1("f5","pol3(0)+[4]*sin(gaus(5)+[8])",0,10)
```



P0	P1	P2	P3	Amp	Cst	X_0	σ	φ
0	1	2	3	**4**	**5**	6	7	**8**

Les fits d'en haut

Un exemple de fit compliqué

- Fitter un spectre avec une maxwellienne:

  3 étapes:

  - Et 1: Définir la fonction
  - Et 2: L'intégrer dans un TF1
  - Et 3: Réaliser le fit

http://caeinfo.in2p3.fr/root/Formation/fr/Jour3/FitMaxwell.root

Et 1: définir la fonction

http://caeinfo.in2p3.fr/root/Formation/fr/Jour3/Maxwell.C

```
//
// Fonction de fit Maxwell
//
#include “TMath.h”

Double_t Maxwell(Double_t *x, Double_t *par)
{
if(x[0] > par[1] && par[2] > 0 && par[0] > 0)
 {
 return par[0]*(x[0]-par[1])/par[2]*
        TMath::Exp(-(x[0]-par[1])/par[2]);
 }
else
 {
 return 0.;
 }
}
```

Tableau d'arguments

Tableau des paramètres

(E-B)/T*exp(-(E-B)/T)

x	E
	0

par	Cst	B	T
	0	1	2

Et 2: intégrer la fonction à ROOT

```
root[0] .L Maxwell.C+
```

Compilation et chargement de la fonction

```
root[1] TF1 *mw=new
   TF1("maxwell",Maxwell,0,200,3)
```

Création du TF1

Portée

Nombre de parametres

```
root[2] mw->SetParNames("Const","B","T")
```

Noms des parametres

```
root[3] mw->SetParameters(100,5,10)
```

Valeurs de départ des parametres

```
root[4] mw->Draw()
```

Dessin de la fonction (pour voir)

Ce qui se passe en mémoire…

Maxwell(x,par)

```
.L Maxwell.C+

TF1 *mw=
 new TF1("maxwell",Maxwell,0,200,3)
```

maxwell

P0	P1	P2
?	?	?

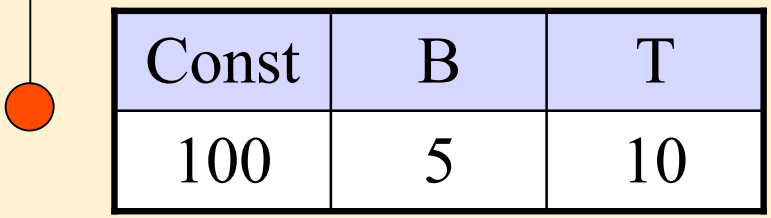
Ce qui se passe en mémoire…

Maxwell(x,par)

```
mw->SetParNames("Const","B","T")

mw->SetParameters(100,5,10)
```

maxwell



Const	B	T
100	5	10

Et 3: fitter

```
root[0] TH1F *h1=(TH1F *)gROOT->FindObject("TestMaxwell")
```
On cherche l'histogramme a fitter

```
root[1] h1->Fit("maxwell")
```
On réalise le fit

```
root[2] mw->GetParameter(2)
```
On récupère le 3ème paramètre (T)

```
root[2] mw->GetParameter("B")
```
On récupère le paramètre de nom "B"

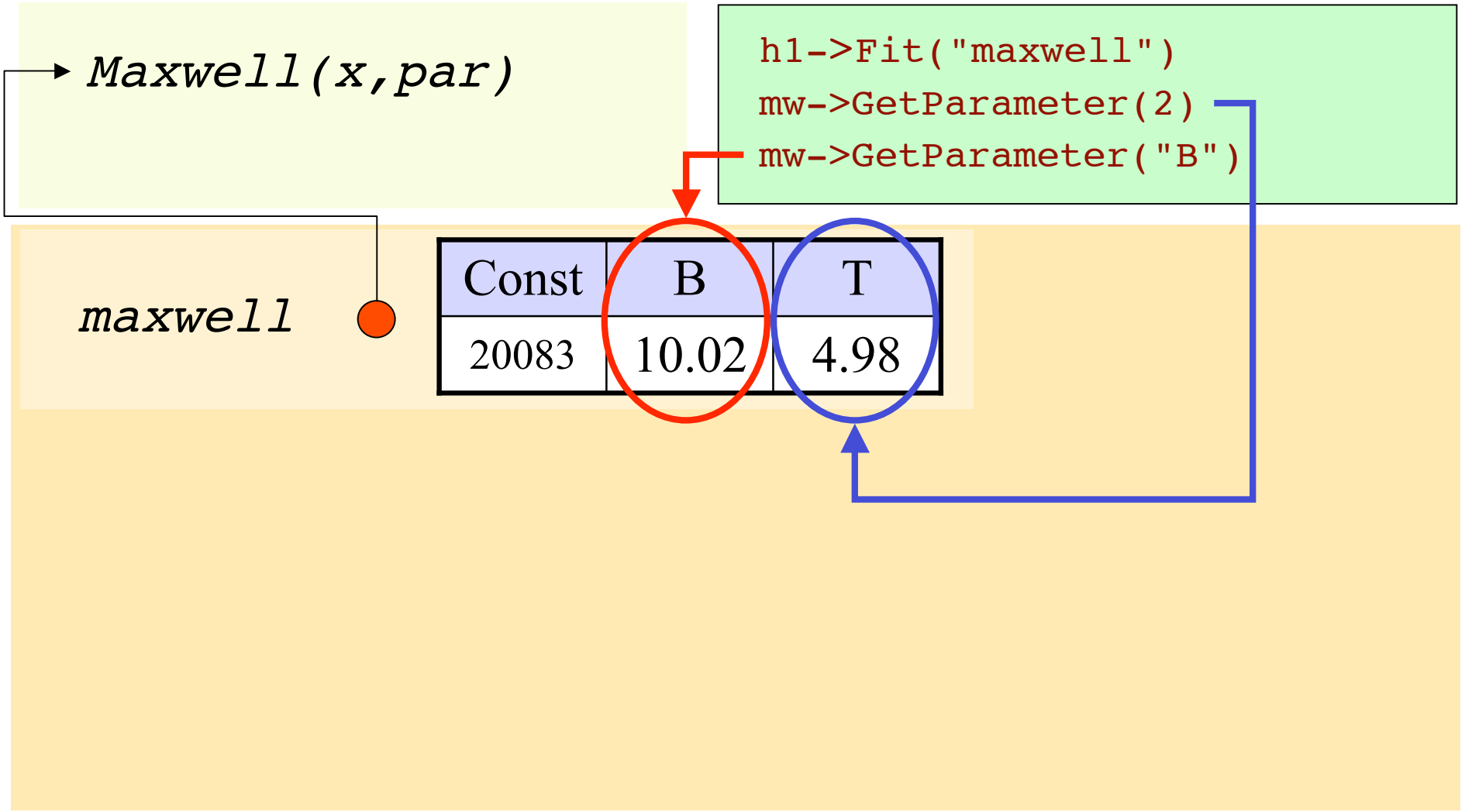
```
root[3] Double_t para[3]
```
On déclare un tableau de 3 réels

```
root[4] mw->GetParameters(para)
```
On récupère les paramètres dans le tableau

```
root[5] mw->GetChisquare()
```
On récupère le Chi2

```
root[6] mw->GetNDF()
```
On récupère le nombre de points utilises dans le fit

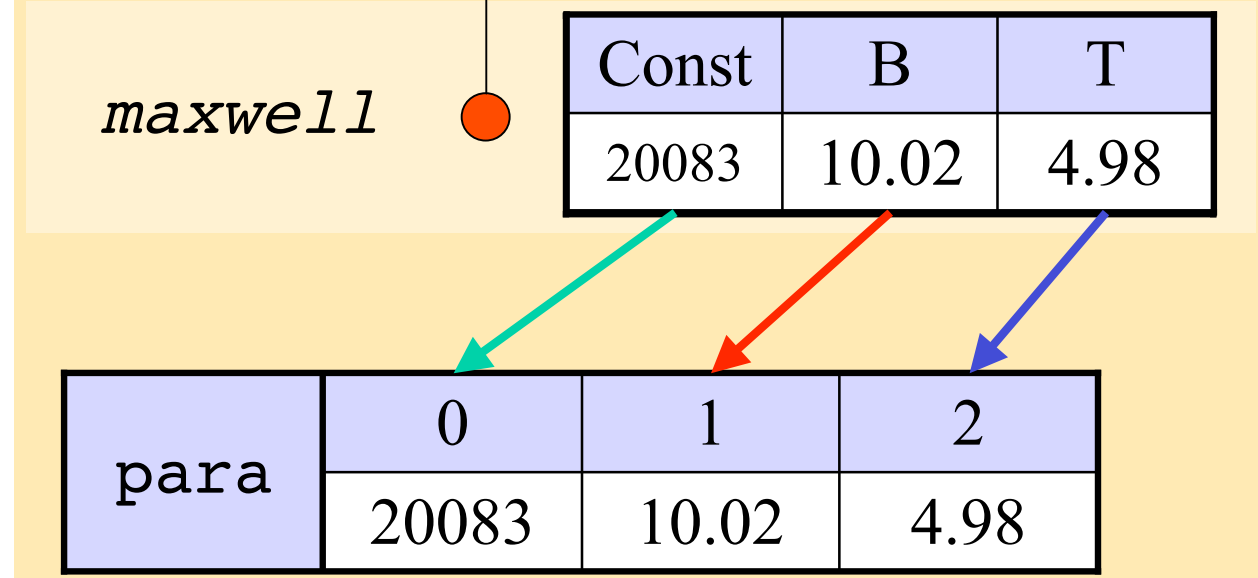
Ce qui se passe en mémoire…



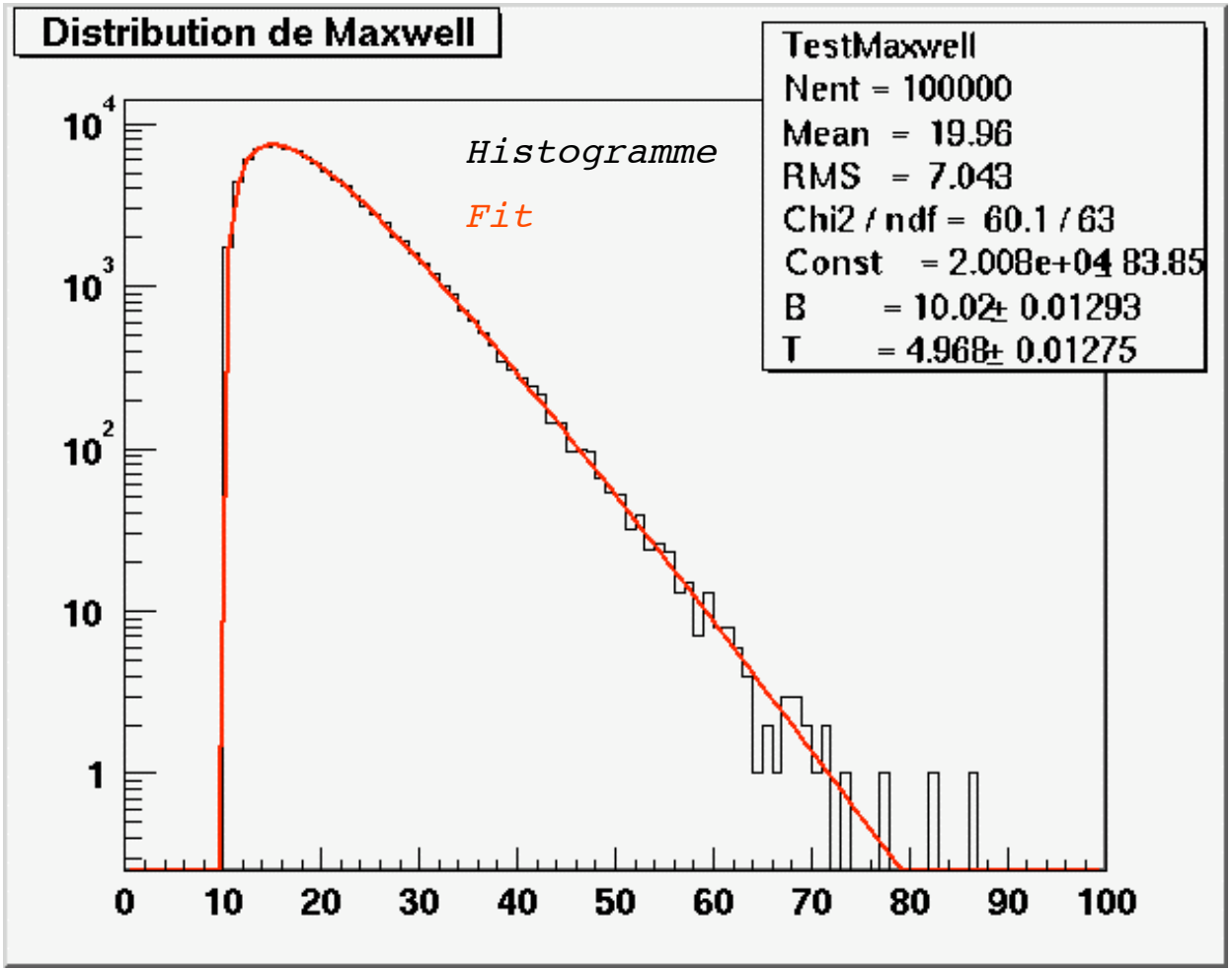
Ce qui se passe en mémoire…

```
Maxwell(x,par)
```

```
Double_t para[3]
mw->GetParameters(para)
```



Oh le beau résultat!



Plus compliqué: gaussienne 2D

http://caeinfo.in2p3.fr/root/Formation/fr/Jour3/Gaus2D.C

```
//
// Fonction de fit Gaussienne 2D
//
#include "TMath.h"

Double_t Gaus2D(Double_t *x, Double_t *par)
{
if(par[2] > 0 && par[4] > 0)
 {
 Double_t rx=(x[0]-par[1])/par[2];
 Double_t ry=(x[1]-par[3])/par[4];
 return par[0]*TMath::Exp(-(rx*rx+ry*ry)/2.);
 }
else
 return 0.;
}
```

Tableau des paramètres

Tableau d'arguments

x	X	Y
	0	1

par	Cst	X_0	σ_X	Y_0	σ_Y
	0	1	2	3	4

Intégrer la fonction

```
root[0] .L Gaus2D.C+
root[1] TF2 *g2D=new TF2("g2d",Gaus2D,-10,10,
                                      -10,10,5)


root[2] g2D->SetParNames("Const","X_{0}","#sigma_{x}",
                                 "Y_{0}","#sigma_{y}")

root[3] g2D->SetParameters(100,5,10,2,3)


root[4] g2D->Draw("surf")
```

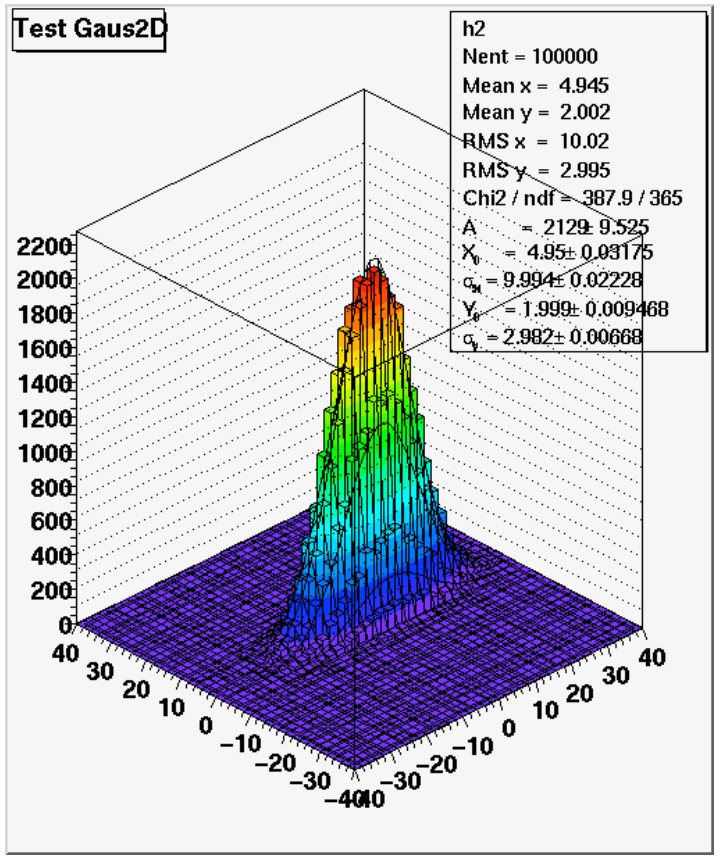
Réaliser le fit

```
root[0] TH2F *h2=(TH2F *)gROOT->FindObject("TestGaus2D")

root[1] h2->Fit("g2d","V")
root[2] h2->Draw("lego2")
root[3] g2D->Draw("surf,same")
```

Pour avoir de belles couleurs!

```
root[3] gStyle->SetPalette(1)
```



Les fits à plusieurs…

De plus en plus fort: mélanger 2 fonctions

Maxwell.C

```
//
// Fonction de fit de 2 Maxwelliennes
//
Double_t DeuxMaxwell(Double_t *x, Double_t *par)
{
  return Maxwell(x,par)+Maxwell(x,&par[3]);
}
```

Maxwell avec par[0], par[1] et par[2]

Maxwell avec par[3], par[4] et par[5]

x	E
	0

par	C_1	B_1	T_1	C_2	B_2	T_2
	0	1	2	3	4	5

Réalisation du fit

```
root[0] .L Maxwell.C+                                  Compilation et chargement des fonctions
root[1] TF1 *deuxmw=new TF1("deuxmax",DeuxMaxwell,0,200,6)
                                                       Définition du TF1
root[2] deuxmw->SetParNames("C1","B1","T1","C2","B2","T2")
                                                  Noms des paramètres
root[3] deuxmw->SetParameters(1,1,1,2,2,2)
                                                  Valeurs de départ
root[4] gStyle->SetOptFit(kTRUE)    Pour afficher le résultat du fit dans les statistiques
Root[5] TH1F *h2m=(TH1F *)gROOT->FindObject("Test2Maxwell")
                                                  On récupère le pointeur de l'histogramme à fitter
root[6] h2m->Fit("deuxmax")                       Réalisation du fit
root[7] Double_t param[6]                         Tableau de réels doubles
root[8] deuxmw->GetParameters(param)              Obtention des paramètres
root[9] mw->SetParameters(param)                  Valeurs pour la première Maxwellienne
root[10] mw->SetLineColor(kRed)                   On la met en rouge
root[11] mw->DrawClone("same")                    On affiche une copie (pourquoi?)
root[12] mw->SetParameters(&param[3])             Valeurs pour la deuxième Maxwellienne
root[13] mw->SetLineColor(kBlue)                  On la met en bleu
root[14] mw->DrawClone("same")                    On affiche une copie
```

Ce qui se passe en mémoire…

Maxwell(x,par)

DeuxMaxwell(x,par)

```
.L Maxwell.C+
TF1 *deuxmw=new TF1("deuxmax",DeuxMaxwell,0,200,6)
deuxmw->SetParNames("C1","B1","T1","C2","B2","T2")
h2m->Fit("deuxmax")
```

maxwell

Const	B	T
?	?	?

deuxmax

C1	B1	T1	C2	B2	T2
901.02	10.00	5.02	1990.26	5.08	24.58

Ce qui se passe en mémoire…

Maxwell(x,par)

DeuxMaxwell(x,par)

```
Double_t param[6]

deuxmw->GetParameters(param)
```

maxwell

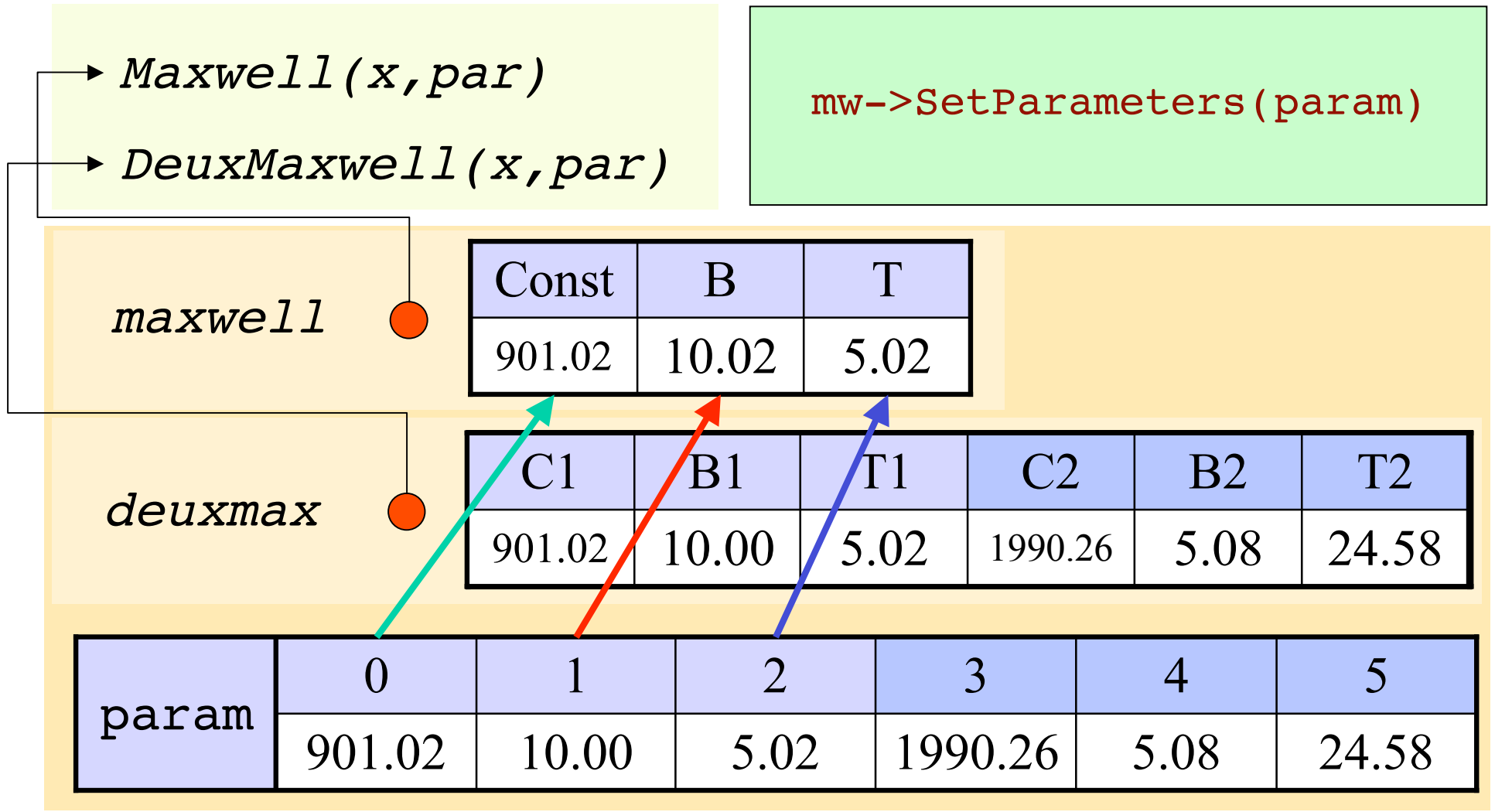
Const	B	T
?	?	?

deuxmax

C1	B1	T1	C2	B2	T2
901.02	10.00	5.02	1990.26	5.08	24.58

param	0	1	2	3	4	5
	901.02	10.00	5.02	1990.26	5.08	24.58

Ce qui se passe en mémoire…



Ce qui se passe en mémoire…

Maxwell(x,par)

DeuxMaxwell(x,par)

```
mw->SetParameters(&param[3])
```

maxwell

Const	B	T
1990.26	5.08	24.58

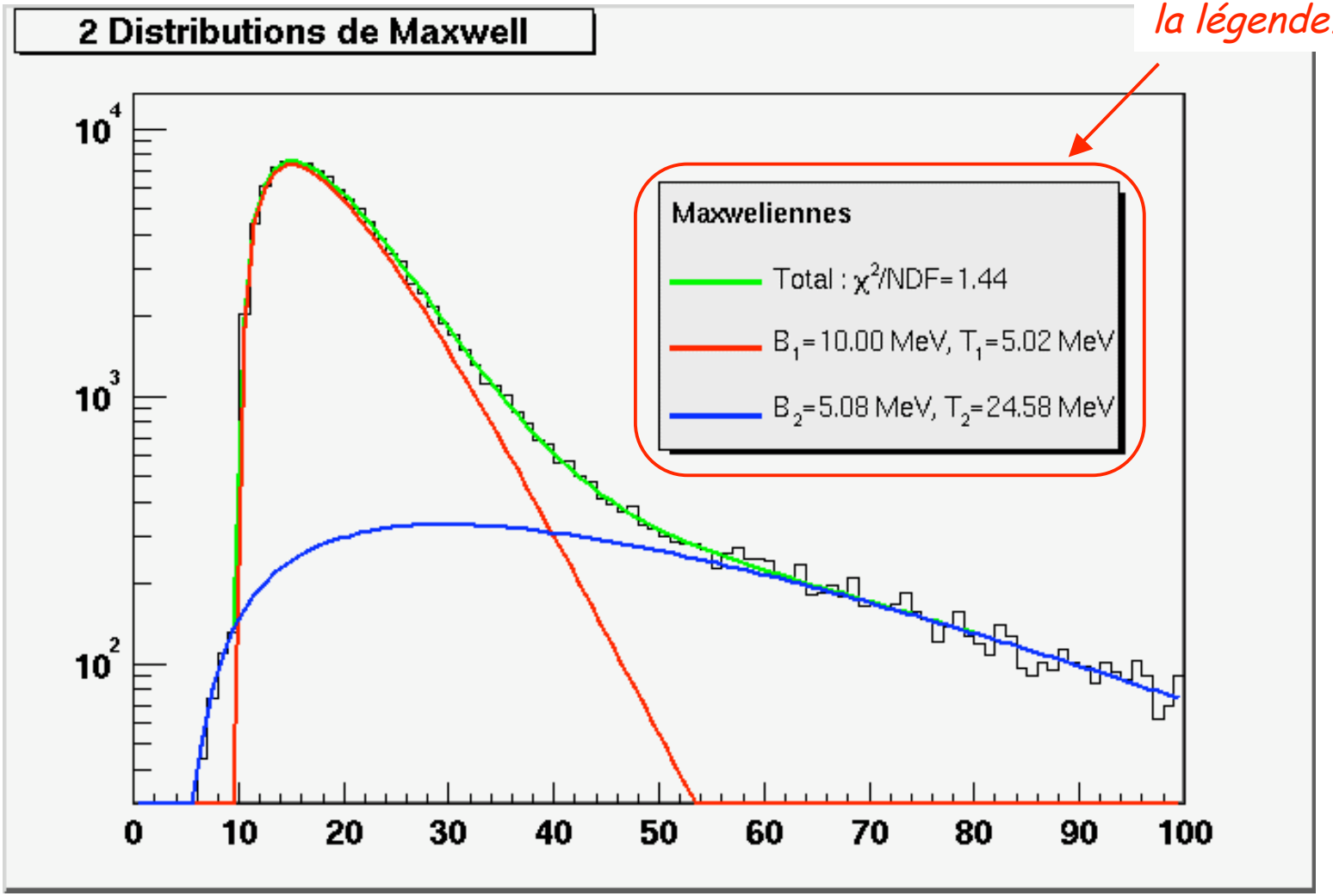
deuxmax

C1	B1	T1	C2	B2	T2
901.02	10.00	5.02	1990.26	5.08	24.58

param	0	1	2	3	4	5
	901.02	10.00	5.02	1990.26	5.08	24.58

C'est vraiment très beau!

Comment avoir la légende?



Une figure de légende

- Il suffit de rajouter un objet **TLegend**
- On ajoute les messages avec la méthode

```
TLegendEntry *TLegend::AddEntry(TObject*, const char*, Option_t*)
```

- Deux petits problèmes:
  - Comment récupérer les bons pointeurs?
    - TF1 nommé deuxmax
    - Les deux clones du TF1 nommé maxwell
  - Comment générer les chaines de caractères avec les informations issues des fonctions?

Regardons les listes (TList) des objets dans le TCanvas

- Liste des objets graphiques dans le TPad courant:

```
root[1] gPad->GetListOfPrimitives()->ls()   (ou gPad->ls())
OBJ: TH1F     Test2Maxwell    2 Distributions de Maxwell : 0 at: 0x593a000
OBJ: TF1      maxwell  : 0 at: 0x342a330
OBJ: TF1      maxwell  : 0 at: 0x344db30
```

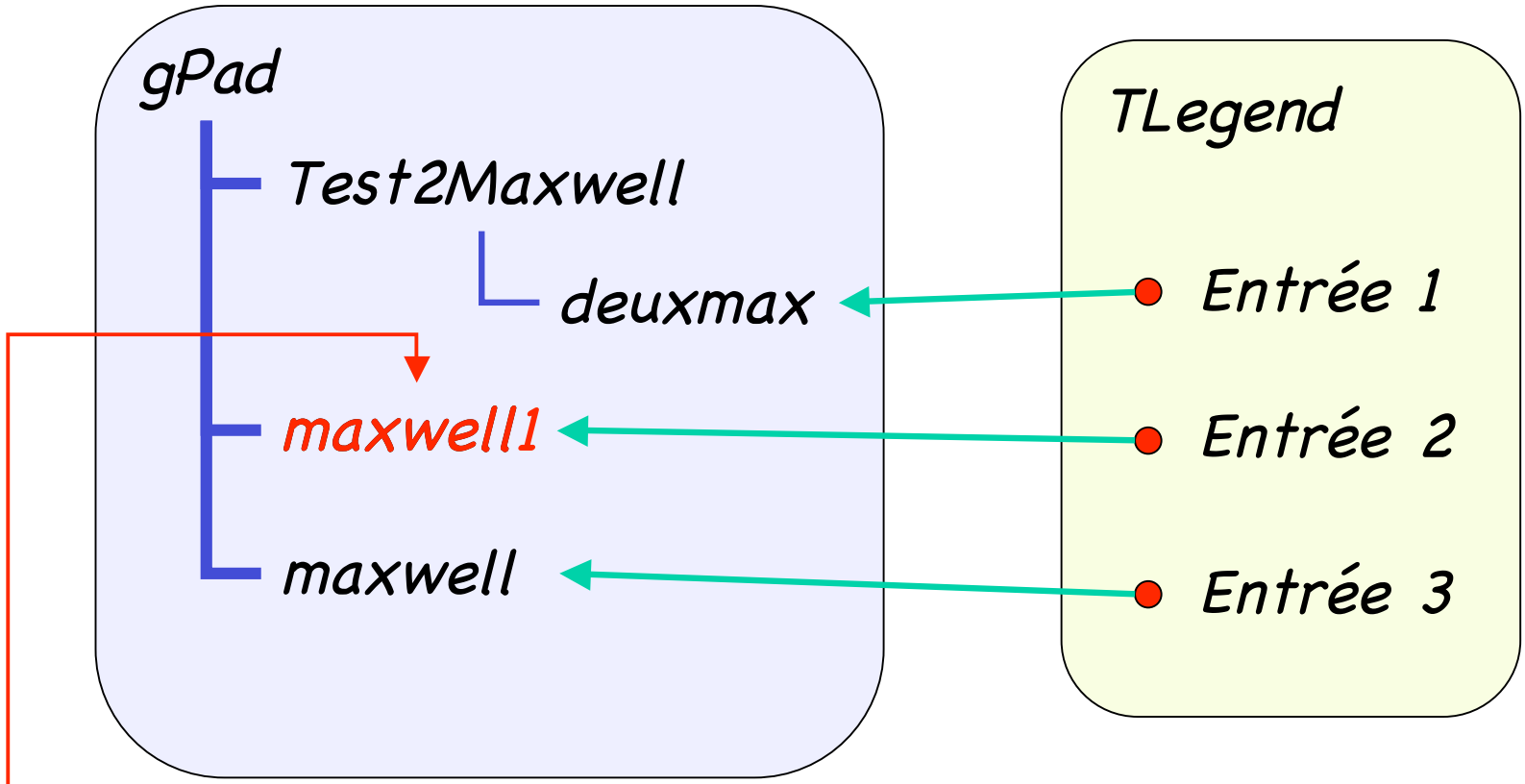
- Que se passe-t-il s'il y a deux objets avec le même nom?

```
root[2] gPad->GetListOfPrimitives()->FindObject("maxwell")
(const class TObject*)0x342a330
```

- Ou est donc passé la fonction deuxmax?

```
root[3] TH1F *h=(TH1F *)gPad->FindObject("Test2Maxwell")
root[4] h->GetListOfFunctions()->ls()
OBJ: TF1        deuxmax  : 0 at: 0x3457af0
```

Ce que l'on va faire



On change le nom du 1er TF1 nommé "maxwell" pour pouvoir retrouver le pointeur de l'autre TF1 nommé "maxwell"

Générer les chaines de caractères

Deux solutions:

- Utilisation de sprintf (fonction C/C++)

```
Char_t chaine[80];
sprintf(chaine,"I=%d, D=%f, S=%s",2,3.14159,"Coucou!");
(Char_t* 0x3462fd0)"I=2, D=3.141590, S=Coucou!"
```

- Utilisation de Form (fonction ROOT)

```
Form("I=%03d, D=%6.3f, S=%-10s",2,3.14159,"Coucou!")
(char* 0x22f4249)"I=002, D= 3.142, S=   Coucou!"
```

- Explication des formats dans:

  http://www.cplusplus.com/ref/cstdio/printf.html

Une figure de légende: le code

http://caeinfo.in2p3.fr/root/Formation/fr/Jour3/MakeFits.C

```
TLegend *legend=new TLegend(0.5,0.5,0.8,0.8,"Maxwelliennes");
Char_t message[80];
TF1 *fun=h2m->GetFunction("deuxmax");
sprintf(message,"Total : #chi^{2}/NDF = %.2f",fun->GetChisquare()/fun->GetNDF());
legend->AddEntry(fun,message);
TList *liste = gPad->GetListOfPrimitives();
for(Int_t i=0;i<2;i++)
 {
 fun=(TF1 *)liste->FindObject("maxwell");
 fun->SetName(Form("maxwell%d",i+1));
 sprintf(message,"%s = %.2f MeV, %s = %.2f MeV",
                 deuxmax->GetParName(3*i+1),fun->GetParameter(1),
                 deuxmax->GetParName(3*i+2),fun->GetParameter(2));
 legend->AddEntry(fun,message);
 }
legend->Draw();
```

Titre

Coordonnées du cadre

Fonction de fit associée au spectre

Ajout de la première ligne

Liste des objets dans le TPad courant

Boucle sur les 2 fonctions "maxwell"

Recherche d'un objet nommé "maxwell"

On change son nom

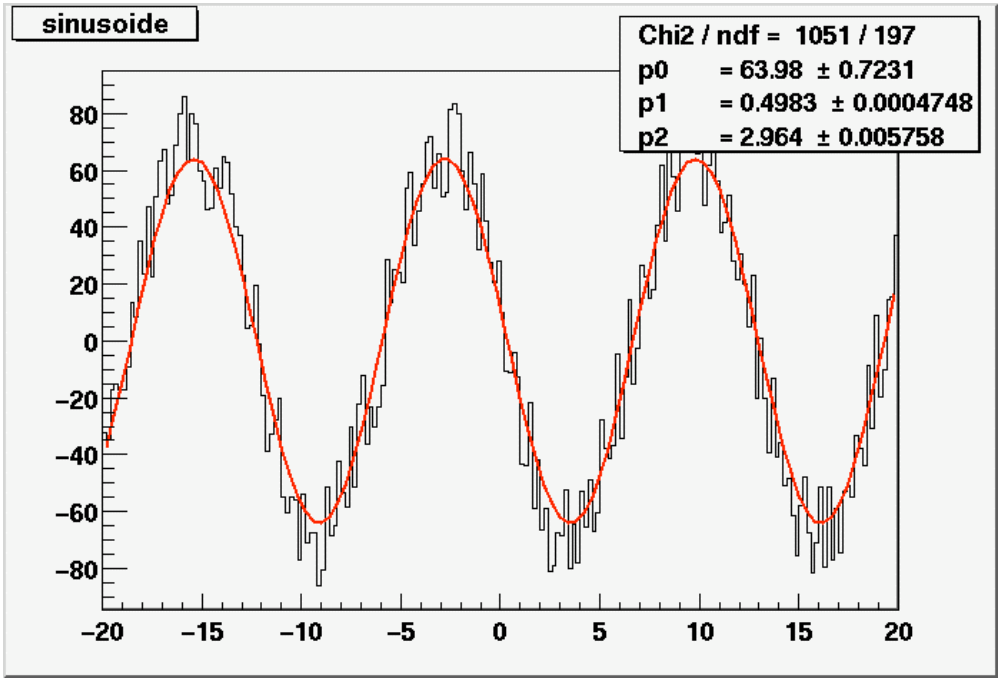
Ajout de la ligne à la TLegend

Dessin de la TLegend

Exercice 1

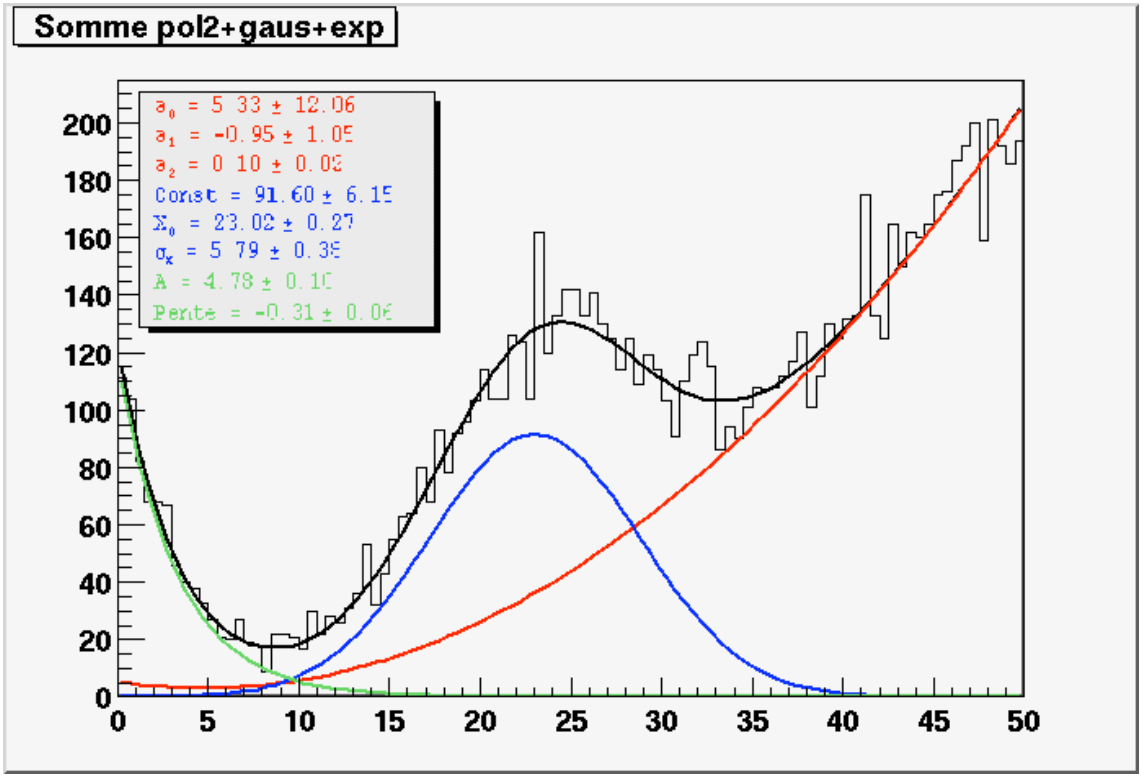
`http://caeinfo.in2p3.fr/root/Formation/fr/Jour3/Fits.root`

- Fitter le spectre `hfs` du fichier `Fits.root` suivant avec une fonction sinusoïdale et afficher les paramètres du fit.



Exercice 2

- Fitter le spectre `hfsomme` du fichier `Fits.root` avec la somme d'une exponentielle (expo), d'une gaussienne (gaus) et d'un polynôme de degré 2 (pol2). Afficher sur le spectre le fit global et les 3 fonctions individuellement.



Exercice 3

- Fitter le spectre **`hData`** du fichier **`Fits.root`** avec la somme
- d'un fond

$$Fond(x) = \frac{Ax}{1 + \exp\left[(x - B)/C\right]}$$

- d'un signal

$$Signal(x) = \begin{cases} G\exp\left(-\frac{1}{2}\left(\frac{x - x_0}{\sigma_g}\right)^2\right) & si\ x \le x_0 \\ G\exp\left(-\frac{1}{2}\left(\frac{x - x_0}{\sigma_d}\right)^2\right) & si\ x > x_0 \end{cases}$$

- Afficher sur le spectre le fit global et les 2 fonctions individuellement avec une légende (**`TLegend`**) pour chacune des fonctions

Résultat de l'exercice 3