ROOT and statistics tutorial

Exercise: Discover the Higgs, part 1

Attilio Andreazza

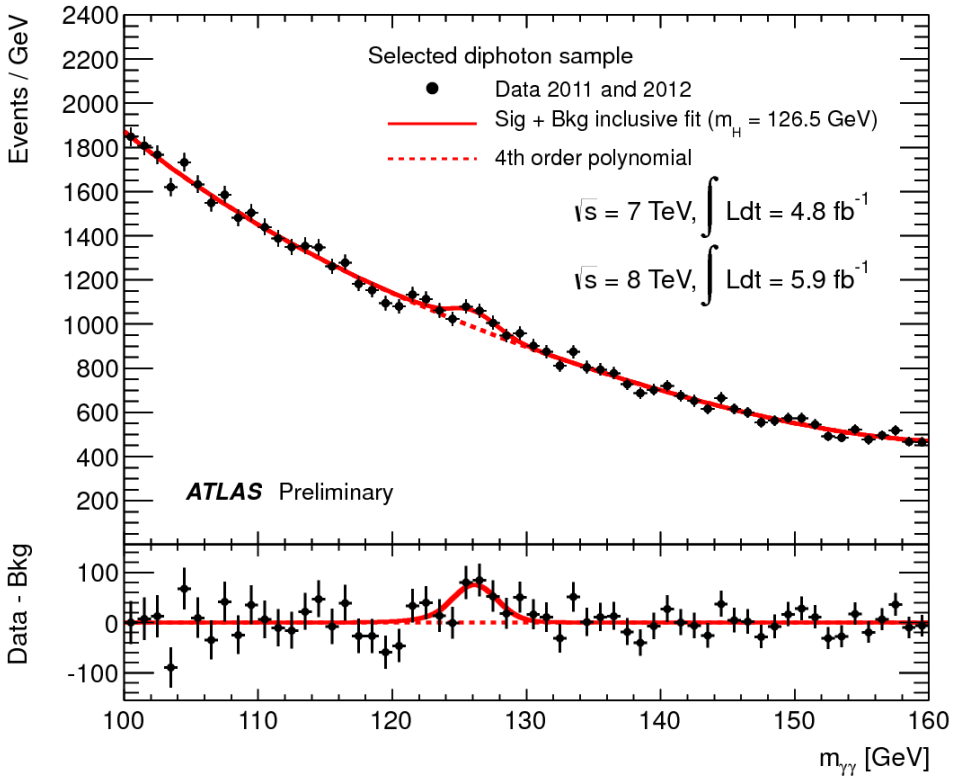
Università di Milano and INFN

Caterina Doglioni

Université de Genève

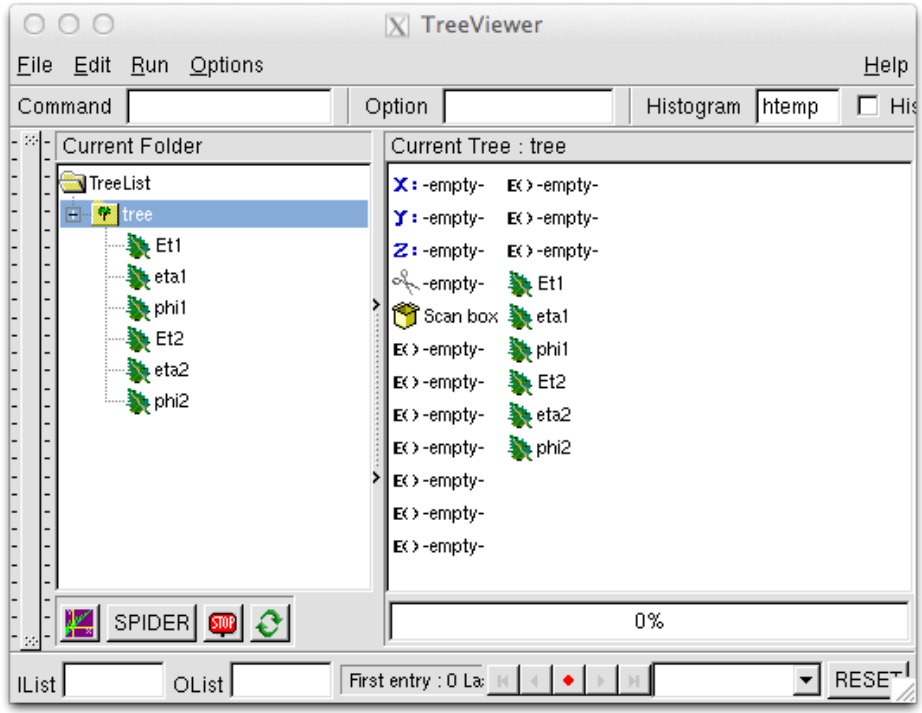
Outline

- What will we do today:
  - **Discover the Higgs boson of course!** ...in a small toy simulation ☺
- What we will learn!
  - Opening and read a root file
  - Fill an histogram with $\gamma\gamma$ invariant masses
  - Fit this histogram with increasing level of complexity:
    1. Estimate resolution from simulation
    2. Understand background shape
    3. Look at a "data" sample: where is the Higgs boson?
- For people running fast:
  - Signal strength plot
  - Solve using RooFit (or PyROOT)



What's in the TTree

- Download from the indico page the zip file containing the material for this exercise: **`Exercise_Hgamgam.zip`**. Pick up **`Higgs130.root`**
  - This file contains 10000 simulated photon pairs coming from a Higgs with mass m_H=130 GeV. For each pair it containt E_T, η, ϕ of the leading and sub-leading γs.
- Enter in Root
- Open the file:
  ```
  TFile* f = TFile::Open("Higgs130.root");
  ```
- List the file content:
  ```
  f->ls();
  ```
- You can see there is a TTree called tree.
  To see its content, open it in a viewer:
  tree->StartViewer();
  In *interactive* ROOT you can use
  object names as pointers.
  But do not try to use that in a macro!
- To see the content of the variables, either double click on them or use the `Draw` method:
  ```
  tree->Draw("phi2:phi1","abs(eta1)<1.");
  ```

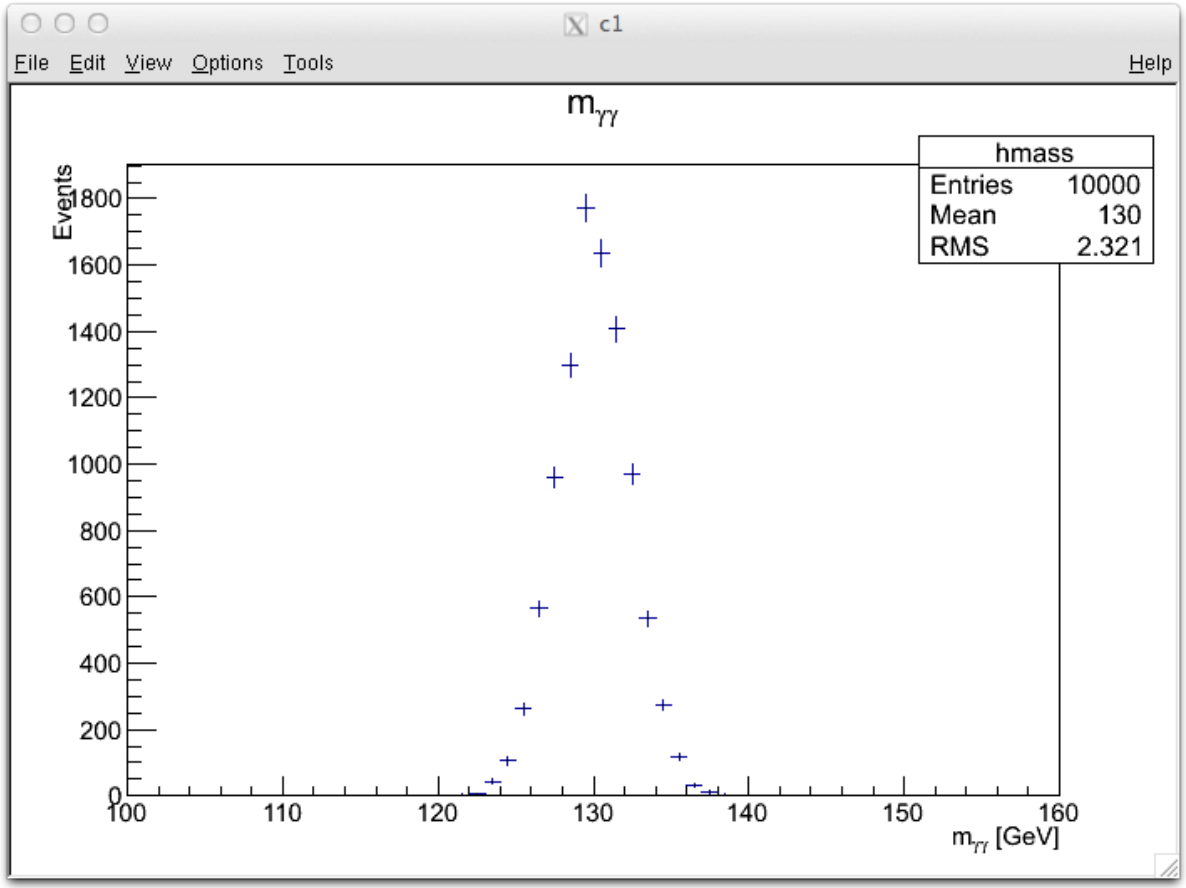


More complex operations

- To do any more serious manipulation on the TTree content, one needs to use at least a macro.
- Among the dounloaded files threse is a small macro example: **`readTree_basic.C`**

- Execute this macro:
  `root -l -x readTree_basic.C`
  it will display the distribution of $\gamma\gamma$ invariant mass.

- **Let's see what's inside!**



The basic macro: part 1

```
void readTree_basic() {
  Char_t *filename = "Higgs130.root";
  // Retrieve the TTree
  TFile* myFile = TFile::Open(filename);
  TTree* tree = (TTree*)(myFile->Get("tree"));
  Double_t Et1, eta1, phi1, Et2, eta2, phi2;
  tree->SetBranchAddress("Et1" ,&Et1 );
  tree->SetBranchAddress("eta1",&eta1);
  tree->SetBranchAddress("phi1",&phi1);
  tree->SetBranchAddress("Et2" ,&Et2 );
  tree->SetBranchAddress("eta2",&eta2);
  tree->SetBranchAddress("phi2",&phi2);
  // Book histograms
  TH1F* hmass = new TH1F("hmass","m_{#gamma#gamma}",60,100.,160.);
  hmass->GetXaxis()->SetTitle("m_{#gamma#gamma} [GeV]");
  hmass->GetYaxis()->SetTitle("Events");
```

To executed automatically same name as file

File to read (it changes during the exercise)

In a macro need to retrieve the object properly (still using their names

This is where information read from file is stored

Book the histogram

Label the axes! (otherwise old professors complain)

The basic macro: part 2

```
// Loop over the events
  Long64_t events = tree->GetEntries();
  for (int i=0; i<events; i++) {
    tree->GetEntry(i);
    TLorentzVector g1,g2;
    g1.SetPtEtaPhiM(Et1,eta1,phi1,0.);
    g2.SetPtEtaPhiM(Et2,eta2,phi2,0.);
    TLorentzVector gg=g1+g2;
    hmass->Fill( gg.M() );
  }
  hmass->Draw("e");
}
```

Get the number of events in the TTree

Load the event i in memory

`TLorentzVector` is a powerful class, implementing the whole Lorentz vector alegebras and many, many useful methods.

See all of them at: `http://root.cern.ch/root/html/TLorentzVector.html`

Fitting is easy when using some predefined functions, like a Gaussian.
Add before:
`hmass->Fit("gaus");`
Get the resolution of γγ mass reconstrution.

Step 1: which is the resolution on $m_{\gamma\gamma}$?

- Compute the resolution on the mass peak for the following samples (the file name indicates the simulated Higgs boson mass):
  - **Higgs110.root**
  - **Higgs120.root**
  - **Higgs130.root**
  - **Higgs140.root**
- After the fit the resulting parameters can be retrived by the instructions:

```
hmass->GetFunction("gaus")->GetParameter(2);
hmass->GetFunction("gaus")->GetParError(2);
```

Function name

- **Provide a reasonable parameterization of the mass resolution as a function of the m_H.**

Hint

In this mass region the Higgs boson is a narrow resonance, so its observed mass is dominated by the experimental resolution.

$$m_H = \sqrt{2E_{g1}E_{g2}\left(1 - \cos\theta_{g1,g2}\right)}$$

The main contribution is the uncertainty in energy reconstruction, which is usually parameterized as:

$$\frac{\sigma_E}{E} = A \oplus \frac{B}{\sqrt{E}} \oplus \frac{C}{E}$$

Calibration | Sampling | Noise

In our simulation one of these three terms will dominate.

Background model

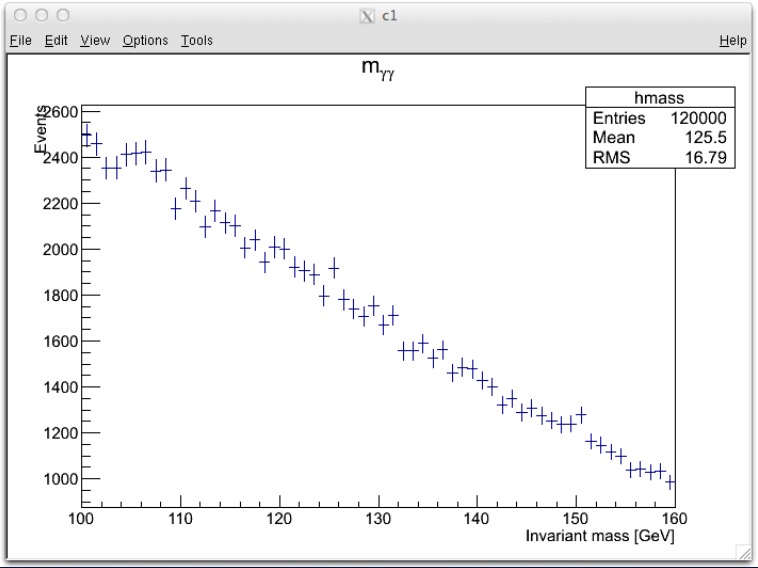
- Use the file: **`Background.root`**
- Content of this file are di-photons from background events: our aim is to define a model for their $m_{\gamma\gamma}$ distribution.
- Let's fit the resulting histograms with different functions:
  - Exponential $p_0*\exp(-m_{\gamma\gamma}/p_1)$
  - 1st, 2nd and 3rd degree polinomials
- Use a more general approach:

```
TF1* myF = new TF1("myF","[0]*exp(-x/[1])",100.,160.);
myF->SetParameter(0,1000.);
myF->SetParameter(1,30.);
hmass->Fit(myF);
```

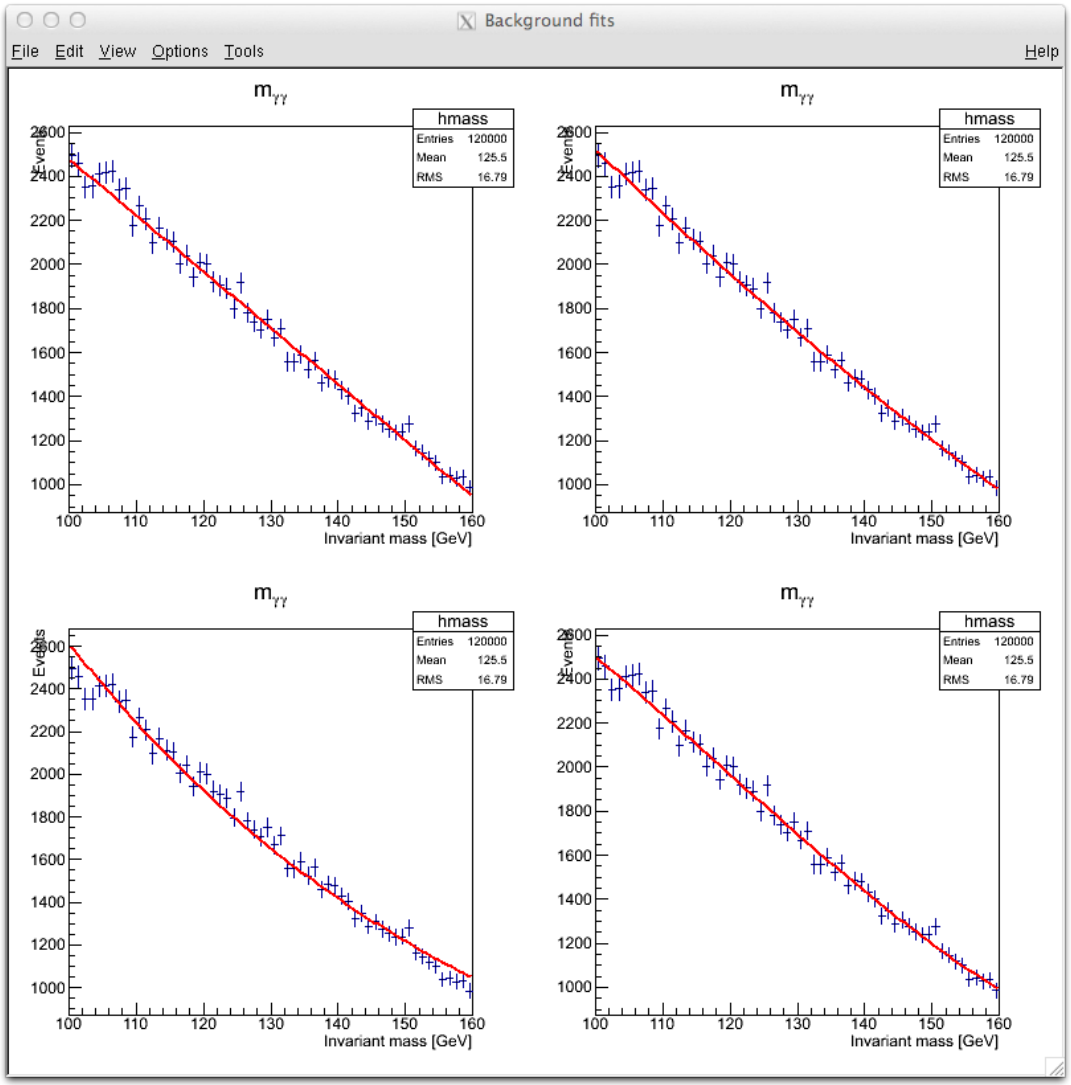
Parameters

Need to set initial values for all parameters

Fit using a pointer to the TF1 fit function



How to discriminate these models?



Hint

If wanting to have multiple plots of the same histogram, use the method `DrawClone` instead of `Draw`.

You need a TCanvas for each clone:

```
TCanvas *c1 = new TCanvas();
hmass->Fit(myF1);
hmass->DrawClone("e");
TCanvas *c2 = new TCanvas();
hmass->Fit(myF2);
hmass->DrawClone("e");
```

Or to divide a Tcanvas in zones:

```
TCanvas *c1 = new TCanvas();
c1->Divide(2,2);
c1->cd(1)
hmass->Fit(myF1);
hmass->DrawClone("e");
c1->cd(2);
hmass->Fit(myF2);
hmass->DrawClone("e");
```

Step 2: choose the background model.

- When invoked with option "S", the `Fit` method gives back a pointer to a `TFitResult` object:

```
TFitResultPtr fit1 = hmass->Fit(myF1,"S");
```

- This object has many accessors to the result and quality of the fit:
  - `Double_t par = fit1->Parameter(0);`
  - `Double_t spar= fit1->ParError(0);`
  - `Double_t chi2= fit1->Chi2();`
  - `Int_t NDF = fit1->Ndf();`

Rules of thumb

1. The p-value of the χ^2 fit must be reasonable (it cannot be perfect if we don't know the real functional form of the background.)
2. If there are parameters compatible with 0, it indicates they are not useful in improving quality of the data descripton.

Retrieve the value of fitted parameter

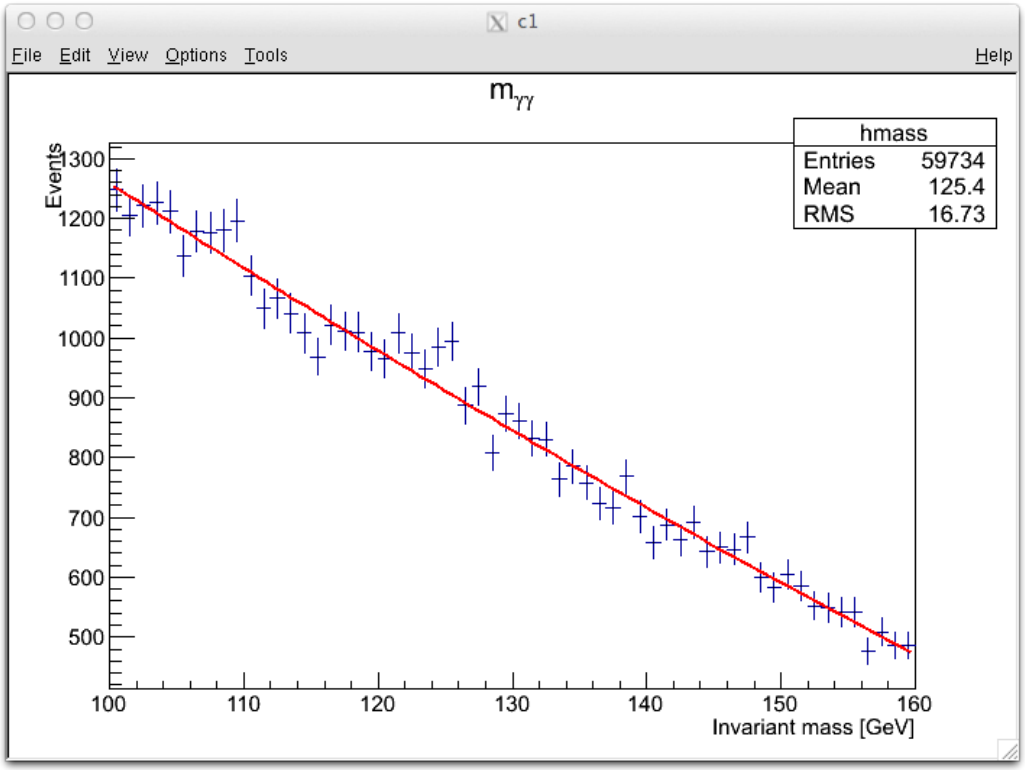
Get its uncertainty

χ^2 and number of degrees of freedom of the fit

- The function `TMath::Prob(chi2,NDF)` provides the p-value for the χ^2 fit (probability that, if the disribution follows the given parameterization, the χ^2 will be worse than the observed one).
- **Which function will you choose?**

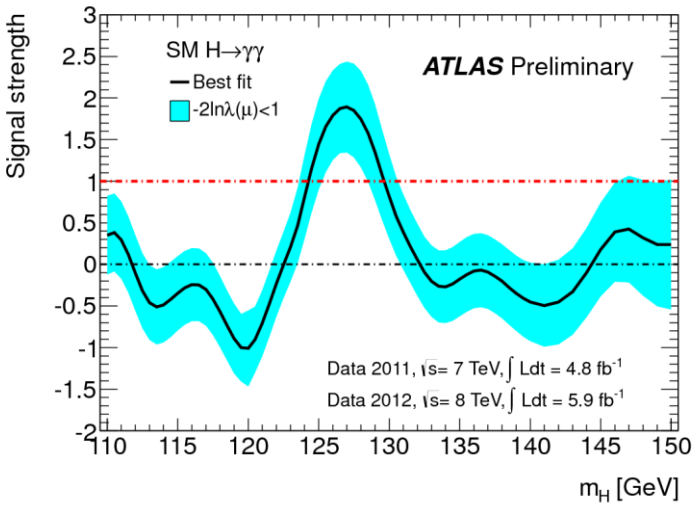
Opening the black box

- Use the file: **`Data.root`**
- Content of this file are di-photons with both signal and background:
  - But you do not know the mass, nor the amount of the signal (it may be zero).
- First, fit it just with the background model found in the previous step:
  - What's the value of the χ^2 and its p-value?
- Fit adding to your background a Gaussian whose width depends on m_H (from your initial study)
  - Be careful in setting initial values of the parameters!
  - To be more specific:
    - How much is the mass of the excess about 110 GeV?
    - How much is its significance? (as significance take the amplitude of the fitted Gaussian divided by its uncertainty)



The signal strength plot

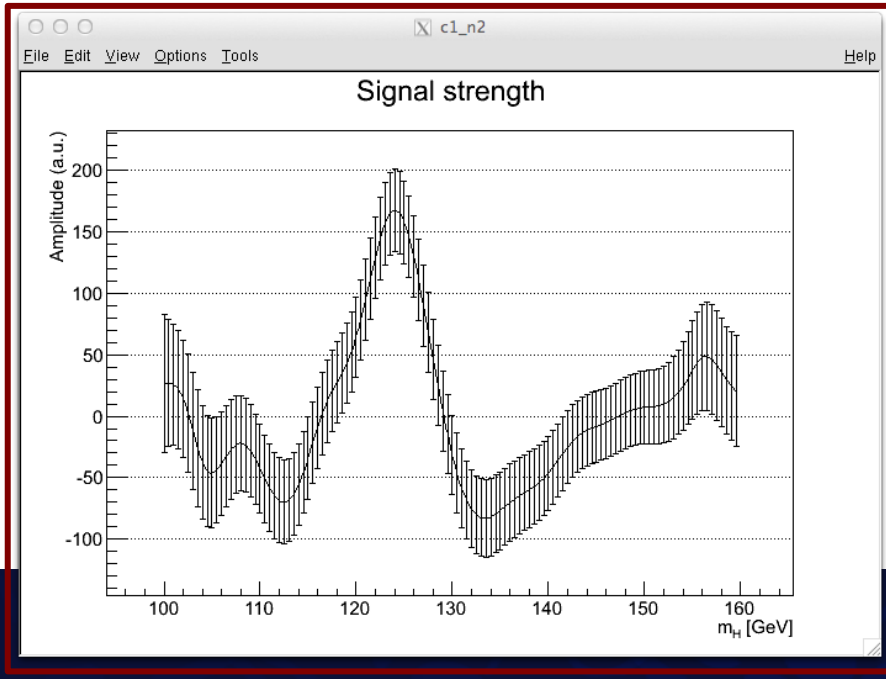
- A more systematic way to proceed is the so-called signal strength:
  - Fix a mass hypothesis
  - Fit the shape for that hypothesis
  - Look at the signal amplitude vs. mass hypothesis
  - *Normally divided for expected signal amplitude* (so that our model as strenght=1): we do not do it here.




```
TGraphErrors* myGraph = new TGraphErrors();
for (Int_t i=0; i<120; i++) {
   ...
   myHiggs->FixParameter(4,mH);
   TFitResultPtr result = hmass->Fit(myHiggs,"SQ");
   myGraph->SetPoint(i,mH,result->Parameter(3));
   myGraph->SetPointError(i,0.,result->ParError(3));
   ...
}
myGraph->Draw("ACP");
```

Fix mass parameter

Retrieve amplitude and its error



If using RooFit

- You can use the same approach, but before fitting you need to translate the **`TH1`** histogram with the mass distribution in a **`RooDataHist`** object.

- To fix a **`RooRealVar`** (you may need that in a loop):

```
Variable = value;
Variable.setConstant(true);
```