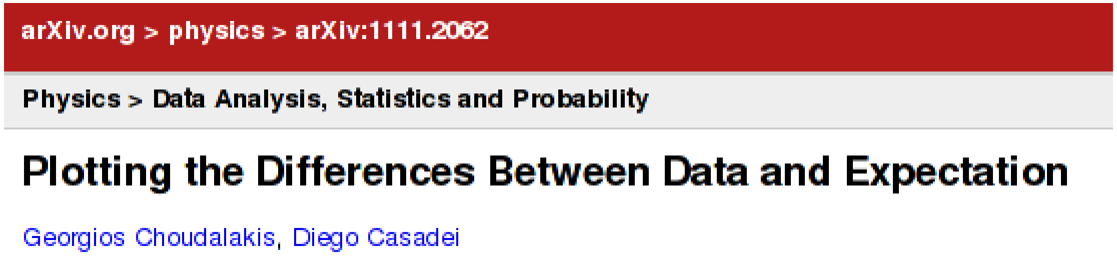
(py)ROOT tutorial, hands-on

Attilio Andreazza Università di Milano
Caterina Doglioni Université de Genève

HASCO school – 18/07/2012

What we're going to do



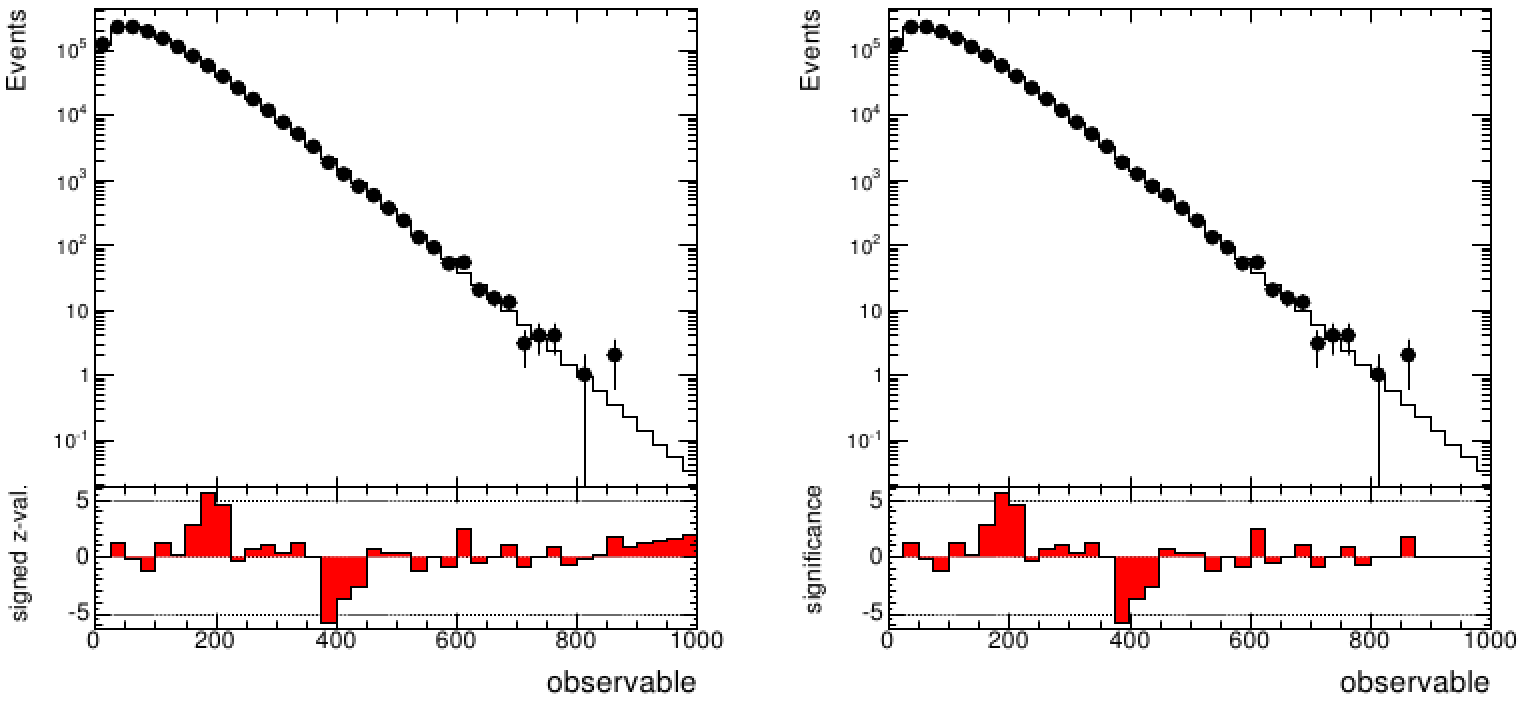
arXiv.org > physics > arXiv:1111.2062

Physics > Data Analysis, Statistics and Probability

Plotting the Differences Between Data and Expectation

Georgios Choudalakis, Diego Casadei

arXiv 1111.2062



+ plots of

p-value
z-value
significance

(...only for the brave)

...in pyROOT!

Step 0: If you want to play with strings, prepare S and B starting from text files

1) Read the following inputs from the text files:

- data.txt
- bkg.txt

binXLow, binXHigh, binContent, binError

hint: these files are the output of TH1->Print(“all”) and come from histograms with constant bin width

2) prepare a .root file containing a histogram for data and a histogram for the background

Since you'll have to do 1) for both signal and background, use a function that takes in the file name and returns a histogram

Step 1: Overlay S and B

- Prepare the data/bg histograms for plotting

  1) open file histograms.root
  (if you haven't followed Step 0...in that case use what you obtained there)

  2) pick out the *data* and *bkg* histograms

  3) format them (choose different colors...)

  4) add them to a THStack

  5) prepare a TLegend

Step 2: Get the relative difference

- Prepare the (D-B)/B histogram

  1) make a copy of the data histogram (D)

  2) subtract the bkg histogram (B) from the copy of D

  3) divide the result of point 2) by B

Step 3: Get (D-B)/sqrt(B)

- Prepare the (D-B)/B histogram

  1) make a copy of the data histogram

  2) subtract B from the copy of D

  3) divide the result of point 2) by sqrt(B)

Step 4: Make and save the plots

1) create two TPads

2) format the plots so axes etc look nice

3) plot the two histograms (THStack) on the top pad, the ratios on the bottom pad

4) save the two plots as png/eps files

Step 5: Think...

1) in the original data histogram, we had an excess and a deficit in the data wrt the background. Where can you see them best?

2) what about the high mjj bins?

Ideally, we would like to see how '*statistically significant*' those deviations are

Caveat: probability and statistics experts alike might have differing opinions on the matter and we could easily digress...
so for now let's take this exercise as something
commonly used at LHC experiments that is good to know about

Step 6: (optional) calculate z-values

1) calculate the Poisson z-value according to the paper (arXiv **1111.2062**) in a ratio-like plot to place in the bottom pad. This will require the calculation of the *p*-value as detailed in the paper.

2) save also result 1) as a histogram that only has bins filled when *p*-value<0.5

Step 7: (optional) plot signed z-value

1) create two TPads

2) format the plots so axes etc look nice

3) plot the two histograms (THStack) on the top pad, the ratios on the bottom pad.

4) save the two plots as png/eps files

Step 8: Think more...

1) what are the advantages of showing a z-value?

2) what are the advantages and disadvantages of showing a signed z-value?

3) what are the advantages of showing a signed z-value showing excess/deficit, but only if p-value < 50%?

The following will pretty much tell you how to do the exercise
...suggestion: use only if you're really stuck

D. Casadei's code (for something slightly more complicated):
http://svn.cern.ch/guest/psde/

Hints to step 0: Prepare S and B

1) Read the following inputs from the text files:

- data.txt
- bkg.txt

binXLow, binXHigh, binContent, binError

Hints:

To open the file, use the infile=open(“filename.txt”, “r”) method
To read in a line, simply use the loop for line in infile: (do sth with line)
To 'tokenize', use tokens = line.split(“ “)
To look for a substring, use myString.find(“substring”):
it will return the index of the first place where the string is found

Hints to step 0: Prepare S and B

2) prepare a .root file containing a histogram for data and a histogram for the background

Hints:

To open the ROOT file for writing,
use outfile=TFile.Open(“filename.root”, “RECREATE”)
To loop on a histogram, use for iBin in xrange(1, hist.GetNbinsX()+1) :
To set bin content/error, use hist.SetBinContent() and hist.SetBinError()
To write an object on a file, use object.Write(“objectName”)

do things in a function

Hints:

To reuse code in functions, define your functions as:

```
def myFunction(argumentName=arg, ...)
  (do something here)
  return hist
```

Hints to step 1: Overlay S and B

- Prepare the data/bg histograms for plotting

1) open file histograms.root

To open the ROOT file for reading,
use infile=TFile.Open(“filename.root”, “READ”)

2) pick out the *data* and *bkg* histograms

To get objects out of a ROOT file, use
object=infile (“filename.root”, “READ”)

3) format them (choose different colors...)

To format histograms to your liking, use functions as:
histo.GetXAxis().SetLineColor(kColor), histo.GetXAxis().SetLineWidth(lineWidth)

Hints to step 1: Overlay S and B

- Prepare the data/bg histograms for plotting

4) add them to a THStack

To add histograms to a THStack, use myStack.Add(histo)

5) prepare a TLegend

To make a TLegend, use TLegend myLegend(xLow, yLow, xHigh, yHigh)
To add entries to a TLegend, use myLegend.AddEntry(histo, “text”, “option”)

Step 2: Get the relative difference

- Prepare the (D-B)/B histogram

1) make a copy of the data histogram (D)

To make a copy of a histogram, use the function
clonedHisto=histo.Clone(“newname”)

2) subtract the bkg histogram (B) from the copy of D

To subtract a histogram from another histo.Add(secondHisto, -1)

3) divide the result of point 2) by B

To divide a histogram numeratorHisto.Divide(denominatorHisto)

Step 3: Get (D-B)/sqrt(B)

- Prepare the (D-B)/B histogram

1) make a copy of the data histogram

To make a copy of a histogram, use the function

```
clonedHisto=histo.Clone(“newname”)
```

2) subtract B from the copy of D

To subtract a histogram from another histo.Add(secondHisto, -1)

3) divide the result of point 2) by sqrt(B)

Since there's no ROOT function that does this quickly, we'll need to loop over all bins of the existing histogram and do it ourselves

```
for iBin in xrange(1,histo.GetNbinsX()+1) :
  num=histo1.GetBinContent(); den = sqrt(histo2.GetBinContent());
  newHisto.SetBinContent(num/den)
```

Step 4: Make the plots

1) create two TPads

```
pad1 = TPad("pad1","pad1",0.05,0.30,1,1);
pad2 = TPad("pad2","pad2",0.05,0.05,1,0.30);
```

2) format the plots so axes etc look nice

For the formatting, see macro TPadExample.C

3) plot the two histograms (THStack) on the top pad, the ratios on the bottom pad

To plot something on a pad, just select it before Draw() using mypad.cd()

4) save the two plots as png/eps files

To save a canvas in a file, use myCanvas.SaveAs(“myFileName.png”)

Step 6: (optional) calculate z-values

1) calculate the Poisson *z*-value according to the paper (arXiv **1111.2062**) in a ratio-like plot to place in the bottom pad. This will require the calculation of the *p*-value as detailed in the paper.

Here you'll have to work with the following functions:

```
from ROOT import Math
gammaValue = Math.inc_gamma_c(nObs, nExp)
```

and

```
pvalue = Math.normal_quantile(p,1)
```

2) save also result 1) as a histogram that only has bins filled when *p*-value<0.5

we'll need to loop over all bins of the existing histogram and check what the p-value of each bin is – if it is >0.5, set bin content to zero

Step 7: (optional) plot signed z-value

1) create two TPads

```
pad1 = TPad("pad1","pad1",0.05,0.30,1,1);
pad2 = TPad("pad2","pad2",0.05,0.05,1,0.30);
```

2) format the plots so axes etc look nice

For the formatting, see macro TPadExample.C

3) plot the two histograms (THStack) on the top pad, the ratios on the bottom pad

To plot something on a pad, just select it before Draw() using mypad.cd()

4) save the two plots as png/eps files

To save a canvas in a file, use myCanvas.SaveAs(“myFileName.png”)