Discover a particle for fun and profit

Mario Pelliccioni
INFN Torino

INFN School of Statistics – Paestum 2022

Let’s do this!

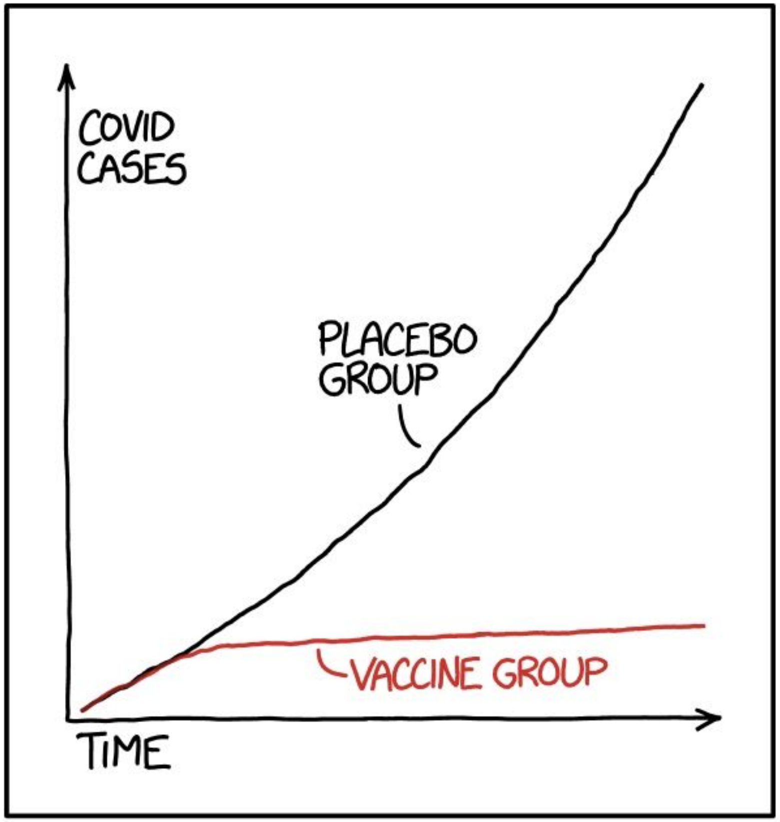
A 4.5 hours class

Cover few relevant cases for statistical analysis in HEP
→ Using RooFit and RooStats as main tools

You can use your laptop for this (provided you installed ROOT and python)
→ Exercises will be in PyROOT

CERN/other labs central clusters usually work too

I will flash a few introductory slides for each topic



STATISTICS TIP: ALWAYS TRY TO GET DATA THAT'S GOOD ENOUGH THAT YOU DON'T NEED TO DO STATISTICS ON IT

Disclaimer

The point of this class is to introduce you to some libraries that let you use different statistical tools

I'll try to present as many different approaches as I can

These are not the best (or most appropriate) ways to approach **any** statistical problem

It's your responsibility to find (or build) the best tool for the job!

RooFit, RooStats and friends

RooFit: a ROOT library containing classes that allow to perform multi-dimensional (un)binned maximum likelihood/chi2 fits, toy-MC generation, plotting, etc

RooStats: a ROOT library that uses RooFit and provides classes to perform statistical interpretation of your results

Combine: an interface to RooFit+RooStats (with some very nifty tools!) created by and for the ATLAS and CMS collaborations

Documentation

For most of what I do, I refer to the ROOT reference guide:

https://root.cern.ch/doc/master/classes.html

This includes RooFit and RooStats reference

RooFit manual (a bit outdated):

https://root.cern.ch/download/doc/RooFit_Users_Manual_2.91-33.pdf

RooStats documentation

https://twiki.cern.ch/twiki/bin/view/RooStats/WebHome

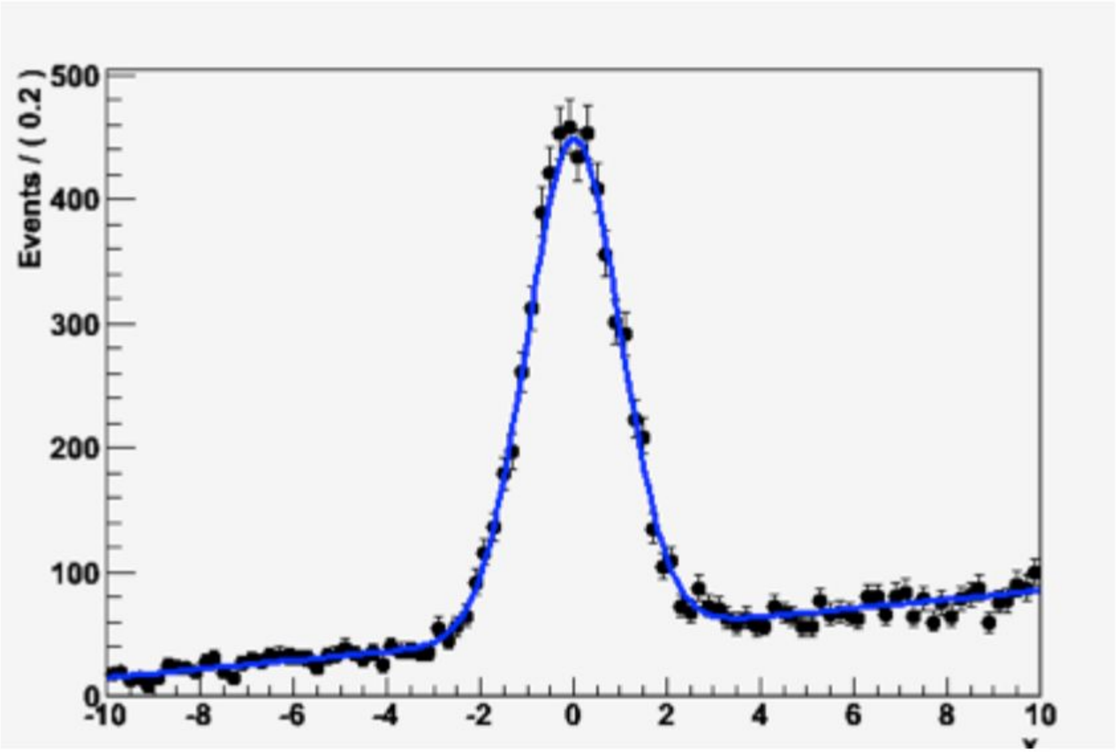
More RooFit/RooStats examples

https://github.com/pellicci/UserCode/tree/master/RooFitStat_class (C++ based)

https://github.com/pellicci/UserCode/tree/master/RooFitStat_class_python

Why do we need RooFit?

- Focus on one practical aspect of many data analysis in HEP: How do you formulate your p.d.f. in ROOT
  - For ‘simple’ problems (gauss, polynomial) this is easy



  - But if you want to do unbinned ML fits, use non-trivial functions, or work with multidimensional functions you quickly find that you need some tools to help you

The origins

- BaBar experiment at SLAC: Extract sin(2β) from time_ dependent CP violation of B decay: $e^+e^- \rightarrow \Upsilon(4s) \rightarrow BB$
  - Reconstruct both Bs, measure decay time difference
  - Physics of interest is in decay time dependent oscillation

$$f_{sig} \cdot \left[\mathrm{SigSel}(m; \overline{p}_{sig}) \cdot \left(\mathrm{SigDecay}(t; q_{sig}, \sin(2\beta)) \otimes \mathrm{SigResol}(t \mid \vec{dt}; r_{sig}) \right) \right] +$$
$$(1 - f_{sig}) \left[\mathrm{BkgSel}(m; \overline{p}_{bkg}) \cdot \left(\mathrm{BkgDecay}(t; q_{bkg}) \otimes \mathrm{BkgResol}(t \mid dt; r_{bkg}) \right) \right]$$

- Many issues arise
  - Standard ROOT function framework clearly insufficient to handle such complicated functions → must develop new framework
  - Normalization of p.d.f. not always trivial to calculate → may need numeric integration techniques
  - Unbinned fit, >2 dimensions, many events → computation performance important → must try optimize code for acceptable performance
  - Simultaneous fit to control samples to account for detector performance

"Dictionary"

- Mathematical objects are represented as C++ objects

Mathematical concept		RooFit class
variable	x	`RooRealVar`
function	$f(x)$	`RooAbsReal`
PDF	$f(x)$	`RooAbsPdf`
space point	$\vec{x}$	`RooArgSet`
integral	$\int_{x_{min}}^{x_{max}} f(x)dx$	`RooRealIntegral`
list of space points		`RooAbsData`

RooFit uses MINUIT for most of its work, it just provides an easy to use interface and optimizations

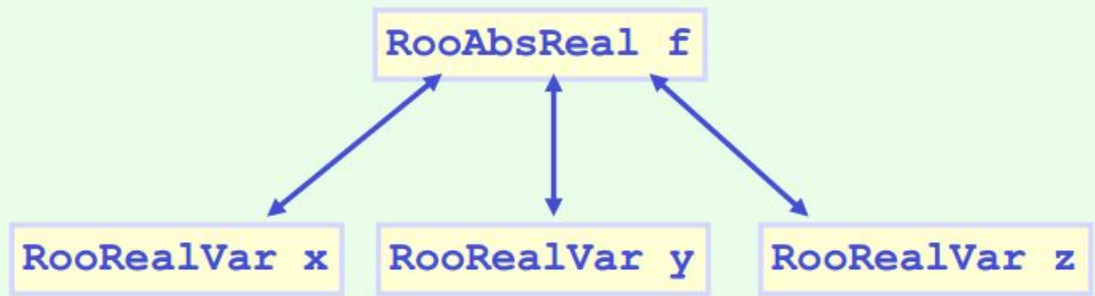
Design philosophy

- Represent relations between variables and functions as client/server links between objects

Math

$$f(x,y,z)$$

RooFit diagram



RooFit code

```
RooRealVar x("x","x",5) ;
RooRealVar y("y","y",5) ;
RooRealVar z("z","z",5) ;
RooBogusFunction f("f","f",x,y,z) ;
```

Variables

All variables (observables or parameters) are defined as **RooRealVar**

Several constructors available, depending on the needs:

```
var1 = ROOT.RooRealVar("var1","My first var",4.15)        #constant variable
var2 = ROOT.RooRealVar("var2""My second var",1.,10.); #range, no initial value
var3 = ROOT.RooRealVar("var3""My third var",3.,1.,10.);  #valid range, initial value
```

You can also specify the unit (mostly for plotting purposes)

```
time = ROOT.RooRealVar("time","Decay time",0.,100.,"[ps]");
```

You can change the properties of your RooRealVar later (setRange, setBins, etc.)

If you want to be 100% sure a variable will stay constant, use RooConstVar

For discrete variables, use RooCategory

Probability Density Functions

Each PDF in RooFit must inherit from RooAbsPdf

RooAbsPdf provides methods for numerical integration, events generation (hit & miss), fitting methods, etc.

RooFit provides extensive list of predefined functions (RooGaussian, RooPolynomial, RooCBShape, RooExponential, RooLandau, etc…)

If possible, use a predefined function (if analytical integration or inversion method for generation available, will speed your computation)

You can define a custom function using RooGenericPdf

Data Handling

Two basic classes to handle data in RooFit:

- **RooDataSet**: an unbinned dataset (think of it as a TTree). An ntuple of data

- **RooDataHist**: a binned dataset (think of it as a THXF)

Both types of data handlers can have multiple dimensions, contain discrete variables, weights, etc.

The perfect container

In order to “move” information among different RooFit/RooStats programs, one can use the RooWorkspace class

A **RooWorkspace** can contain:

- Variables
- PDFs
- DataSets

A RooWorkspace can be saved into a ROOT file

We’ll see how to use it

The problem at hand

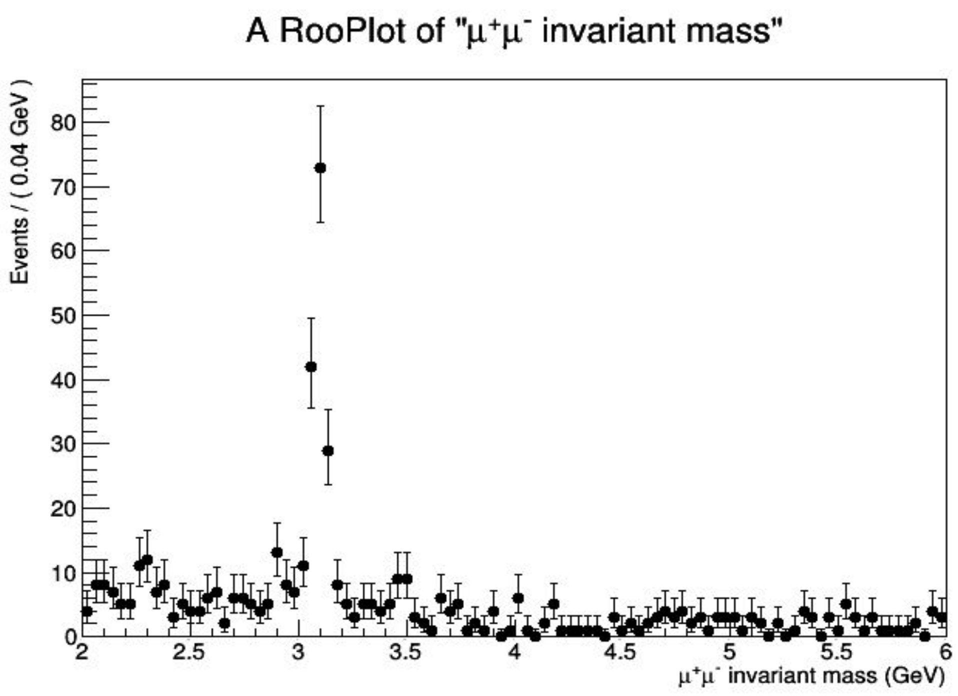
We'll be analyzing a sample from the 2010 CMS dataset

All CMS data from Run1-2 is <u>public</u> → opendata.cern.ch

- Events with two opposite sign muons
- Calculated the invariant mass of the system
- Saved it into a RooDataSet (a 1D ntuple containing "mass" variable)

First, let's look at the first three weeks of data taking (corresponds to about half a pb^{-1} of integrated lumi)

We'll be studying this distribution



Exercise #0

The first exercise involves RooFit only

- Construct a J/ψ and $\psi(2S)$ + background PDF
  - J/ψ with a Crystal Ball function
  - $\psi(2S)$ with a “similar” Crystal Ball
  - Background with a polynomial
- For now, the $\psi(2S)$ will involve a very small amount of signal events
- Fit it, plot it, save it

We are going to use this program all the way through the exercises

Parameter of interest

→ a variable you want to know to the best precision and accuracy possible. Depends on problem

Number of ψ(2S) could be considered the POI

In reality, probably more interested in cross section of ψ(2S) production → real connection with theory

$$\sigma(pp \rightarrow \psi(2S)) * BR(\psi(2S) \rightarrow \mu\mu) = \frac{N_{\psi(2S)}}{\mathcal{L} * \epsilon_{\mu\mu}}$$

How do we express our problem in this way?

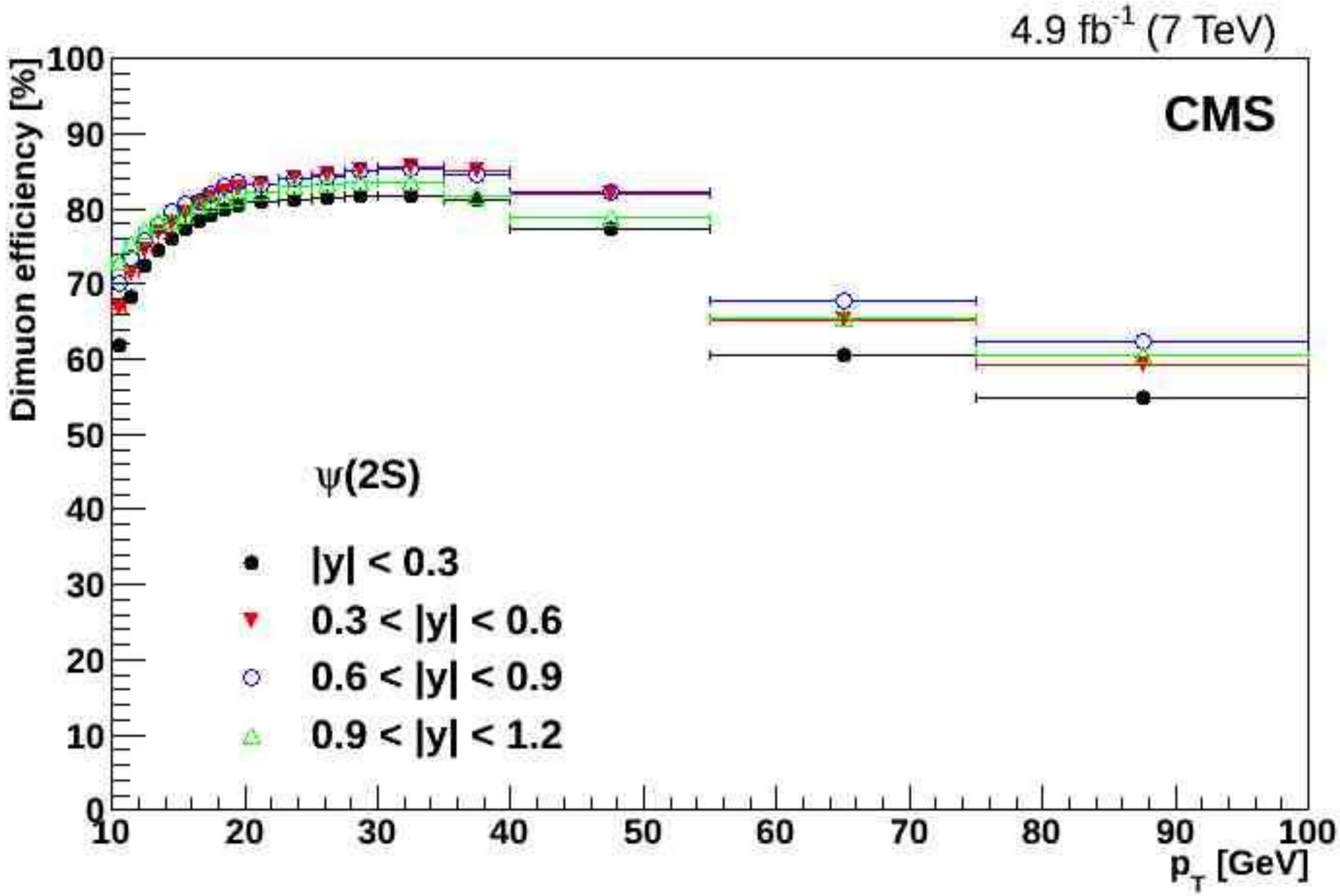
We'll assume:

75% total efficiency
A luminosity of 0.64 pb^{-1}

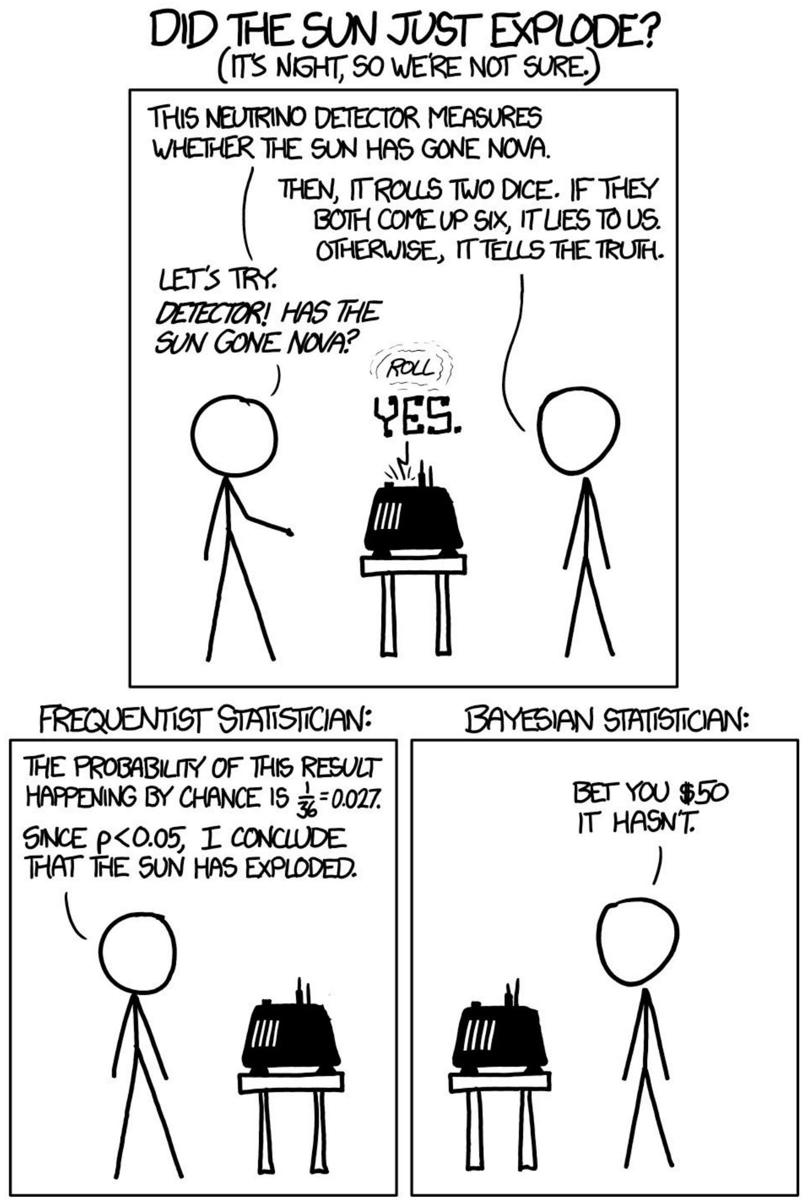
Both efficiency and luminosity uncertainties are negligible (for now!)

Dimuon efficiency

From CMS-BPH-14-001

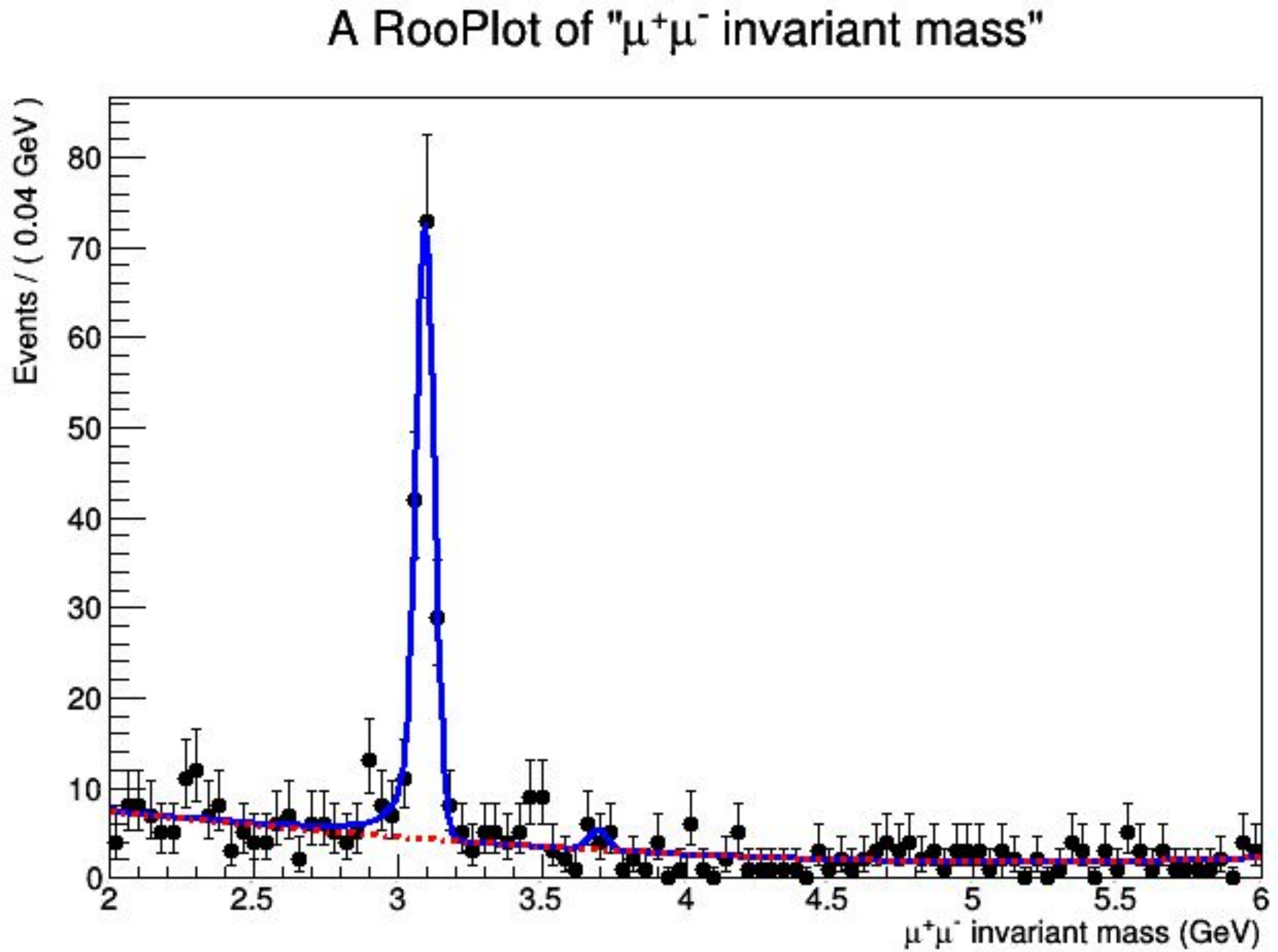


Intermezzo



DID THE SUN JUST EXPLODE?
(IT'S NIGHT, SO WE'RE NOT SURE.)
THIS NEUTRINO DETECTOR MEASURES WHETHER THE SUN HAS GONE NOVA.
THEN, IT ROLLS TWO DICE. IF THEY BOTH COME UP SIX, IT LIES TO US. OTHERWISE, IT TELLS THE TRUTH.
LET'S TRY.
DETECTOR! HAS THE SUN GONE NOVA?
ROLL
YES.
FREQUENTIST STATISTICIAN:
THE PROBABILITY OF THIS RESULT HAPPENING BY CHANCE IS $\frac{1}{36}=0.027$.
SINCE $p<0.05$, I CONCLUDE THAT THE SUN HAS EXPLODED.
BAYESIAN STATISTICIAN:
BET YOU $50 IT HASN'T.

Result of exercise #0



Complicating the problem

Our current parametrization actually sensitive to

$$\sigma(pp \to \psi(2S) + X) \cdot BR(\psi(2S) \to \mu^+\mu^-)$$

Additional exercise: modify $N_{\psi(2S)}$ parametrization to actually measure the cross section!

Assume $BR(\psi(2S) \to \mu^+\mu^-) \sim 8*10^{-3}$

Exercise #1: test fit with toy-MCs

RooFit has tools to test robustness of model via toy-MC generation

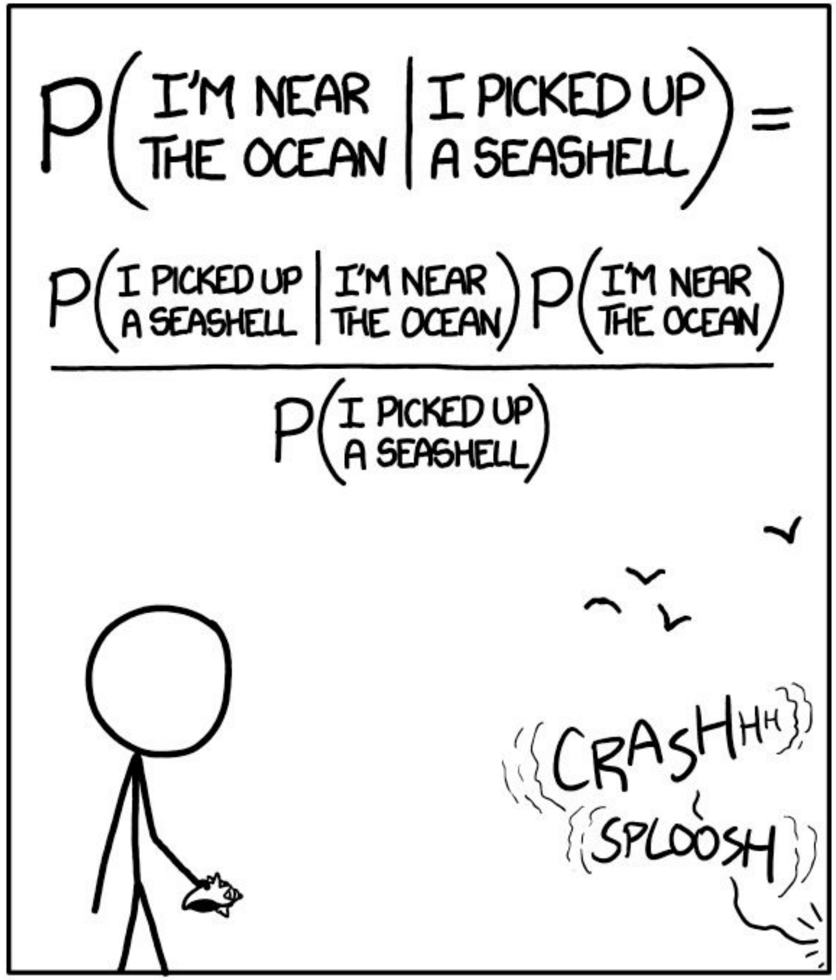
Usually healthy to perform, especially on POI

We do this in exercise 1

Approach:
- Use result of fit #0 as *true* model
- Generate 1k experiments, with same stats as CMS, using *true* model
    - Probably not enough, but time is limited
- Refit the 1k experiments with same model
- Compare with the *true* value of the parameters

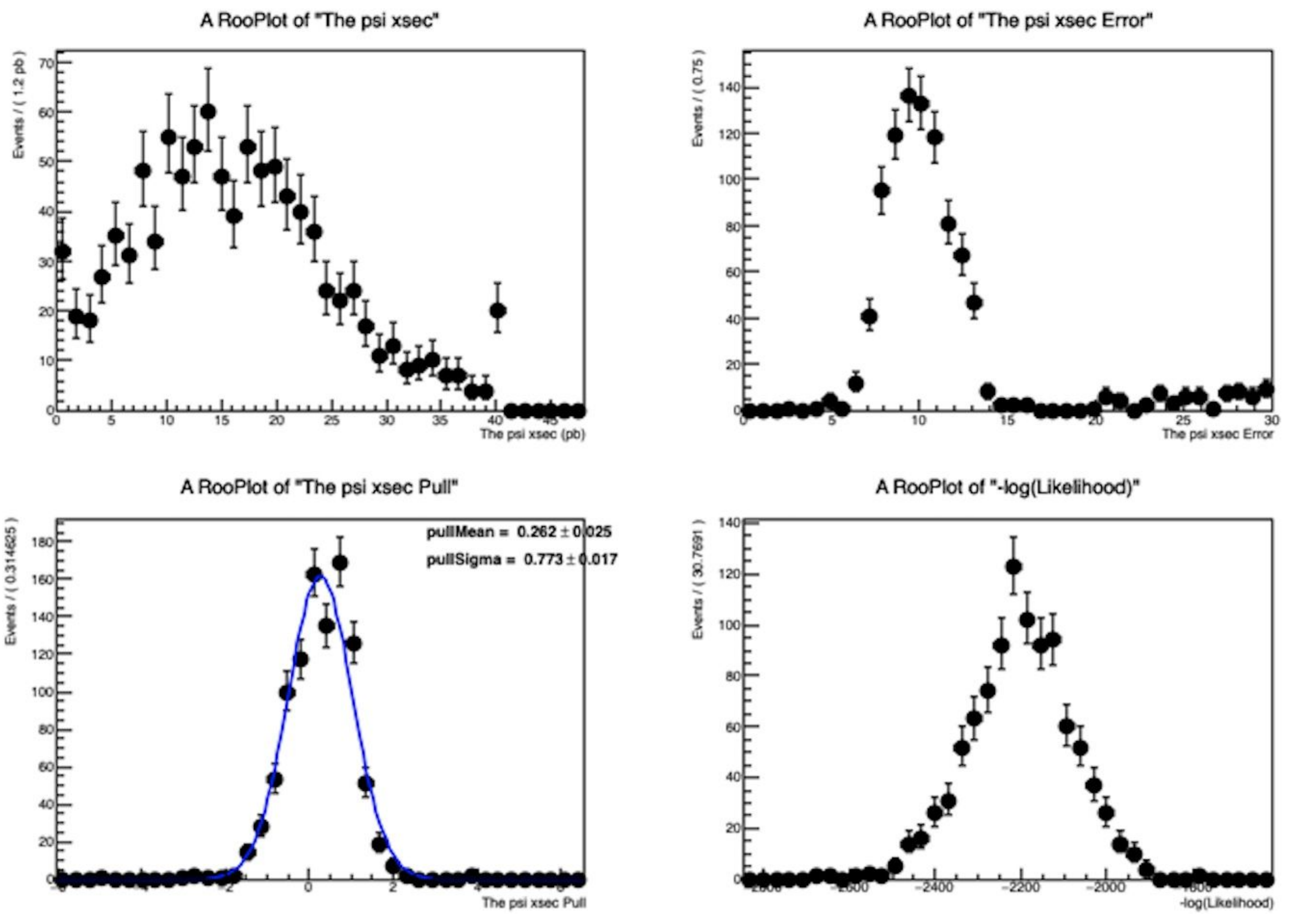
RooFit can do this pretty easily

Intermezzo

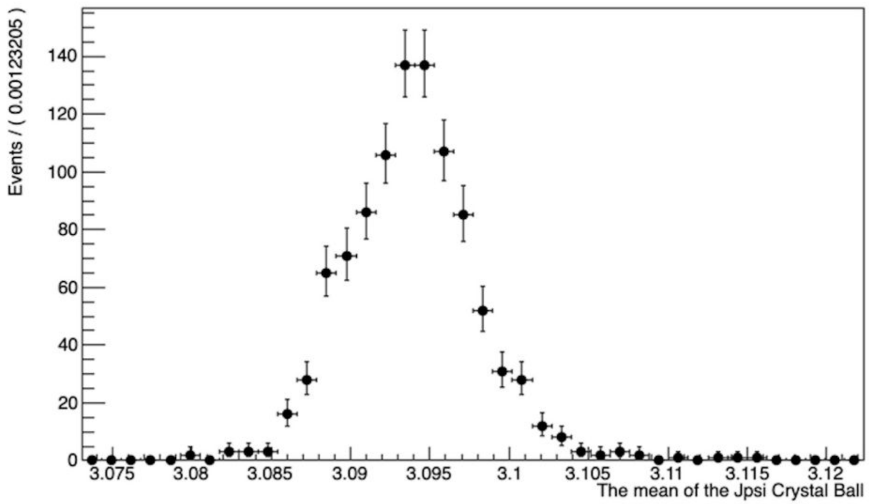


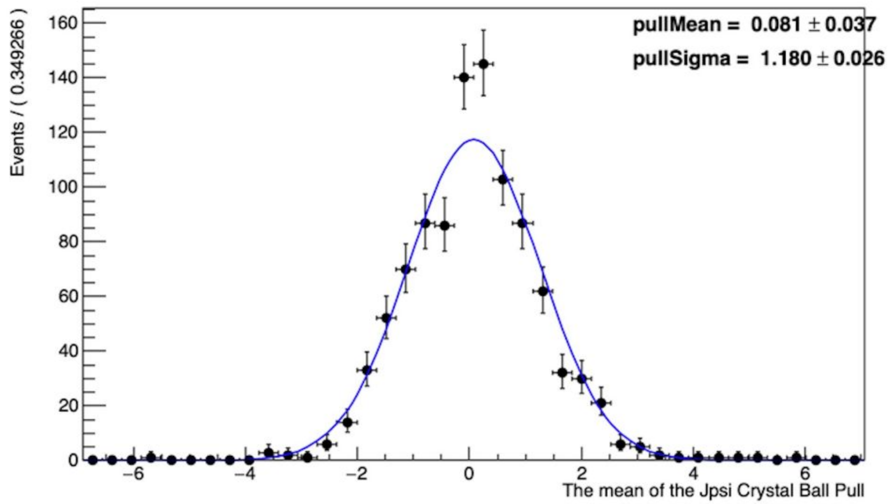
STATISTICALLY SPEAKING, IF YOU PICK UP A SEASHELL AND *DON'T* HOLD IT TO YOUR EAR, YOU CAN PROBABLY HEAR THE OCEAN.

Results of exercise #1



Comparing with higher stats





Additional exercise: what happens if

- you increase number of experiments?
- you increase statistics of each experiment?

→ different effect on what toy-MCs can tell you

RooStats

Set of libraries for statistical interpretation of your results
→ communicates with RooFit via RooWorkspace

RooStats does essentially two things:

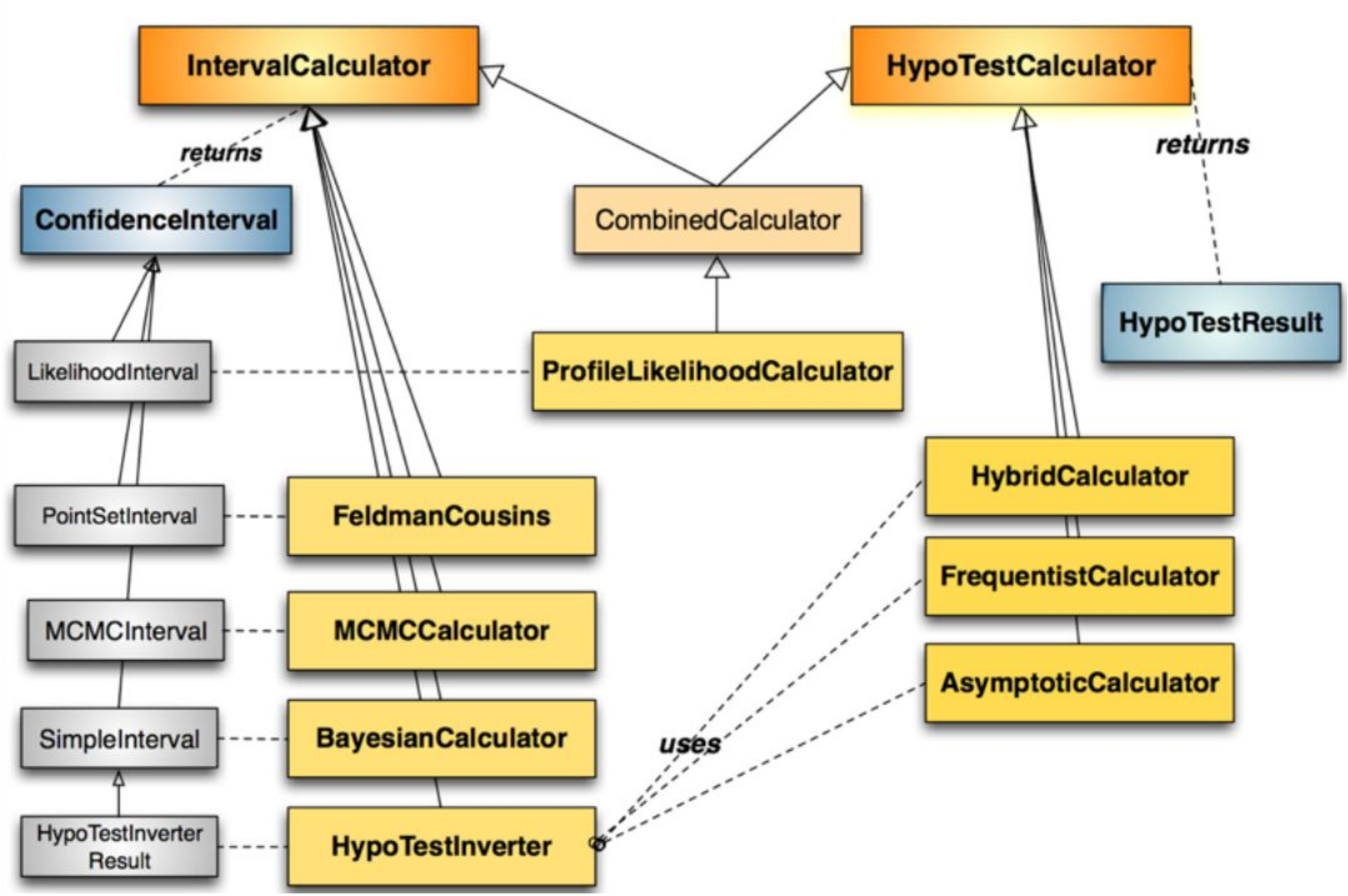
Interval calculation

Hypothesis testing

To do this, it uses "calculators"

RooStats design

C++ classes that reproduce statistical concepts



Main RooStats calculators

ProfileLikelihood calculator

- interval estimation using asymptotic properties of the likelihood function

Bayesian calculators

- interval estimation using Bayes theorem

BayesianCalculator (analytical or adaptive numerical integration)

MCMCCalculator (Markov-Chain Monte Carlo)

HybridCalculator, FrequentistCalculator

- frequentist hypothesis test calculators using toy data (difference in treatment of nuisance parameters)

AsymptoticCalculator

- hypothesis tests using asymptotic properties of likelihood function

HypoTestInverter

- invert hypothesis test results (from Asympototic, Hybrid or FrequentistCalculator) to estimate an interval
- main tools used for limits at LHC (limits using CLs procedure)

NeymanConstruction and FeldmanCousins

- frequentist interval calculators

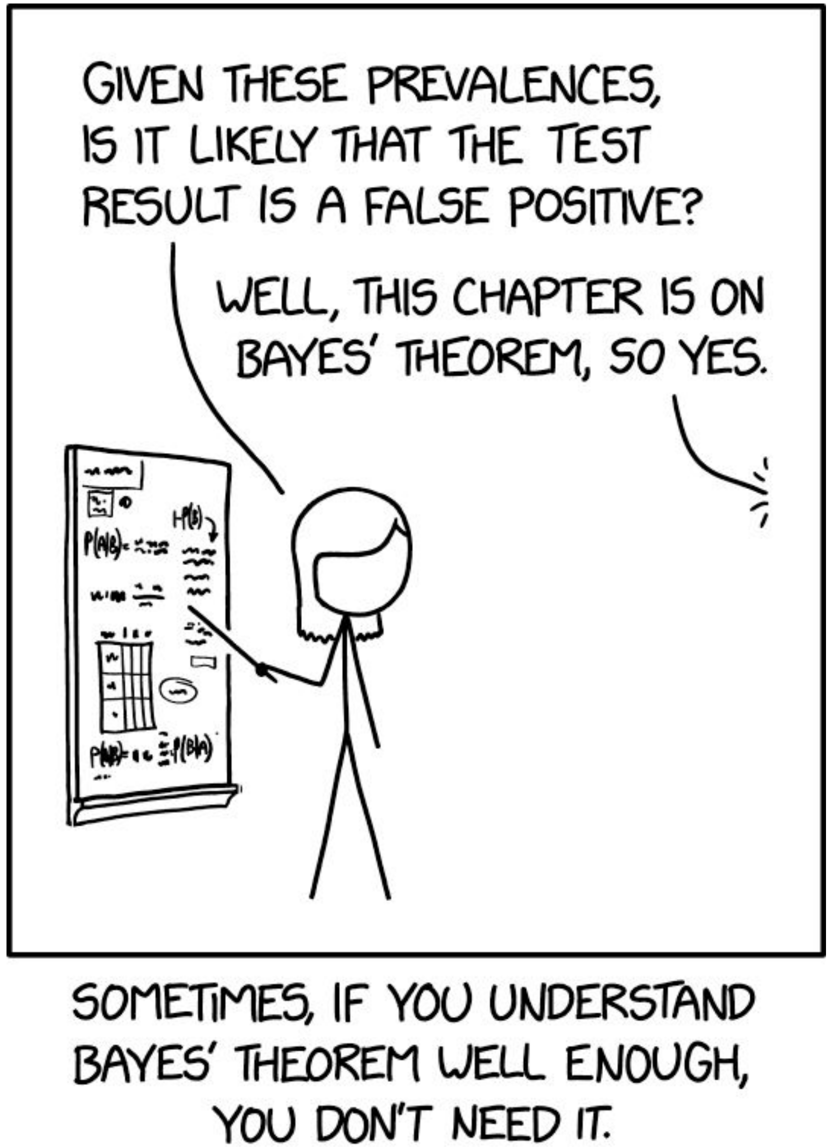
Exercise #2

Exercise #0 told us that there's clearly no significant peak in the distribution

Is this actually clear? How do we quantify?

Let's use the likelihood ratio!

Intermezzo


GIVEN THESE PREVALENCES, IS IT LIKELY THAT THE TEST RESULT IS A FALSE POSITIVE?
WELL, THIS CHAPTER IS ON BAYES' THEOREM, SO YES.
SOMETIMES, IF YOU UNDERSTAND BAYES' THEOREM WELL ENOUGH, YOU DON'T NEED IT.

Exercise #3

From exercise #2 we know that our excess is “not significant”.

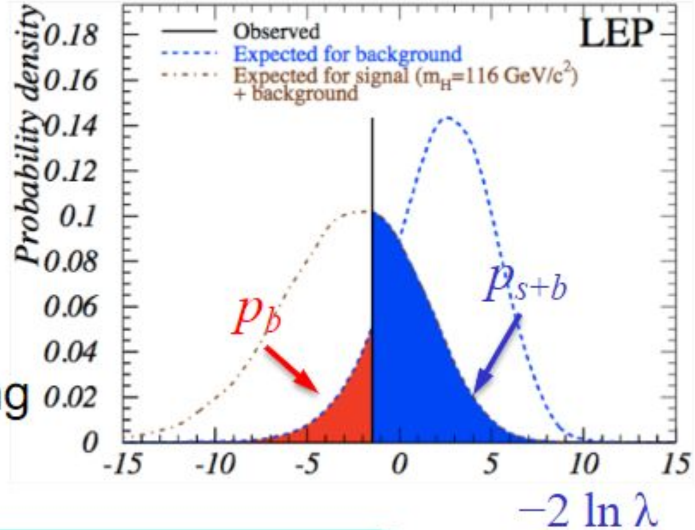
The normal procedure here is to evaluate an upper limit on our parameter of interest.

For the frequentist method, we will use CL_s...

Understanding CL_s

- A modified approach was proposed for the first time when combining the limits on the Higgs boson search from the four LEP experiments, ALEPH, DELPHI, L3 and OPAL
- Given a test statistic $\lambda(x)$, determine its distribution for the two hypotheses $H_1(s+b)$ and $H_0(b)$, and compute:

$$\begin{cases} p_{s+b} = P\left(\lambda(x|H_1) \leq \lambda^{\text{obs}}\right) \\ p_b = P\left(\lambda(x|H_0) \geq \lambda^{\text{obs}}\right) \end{cases}$$



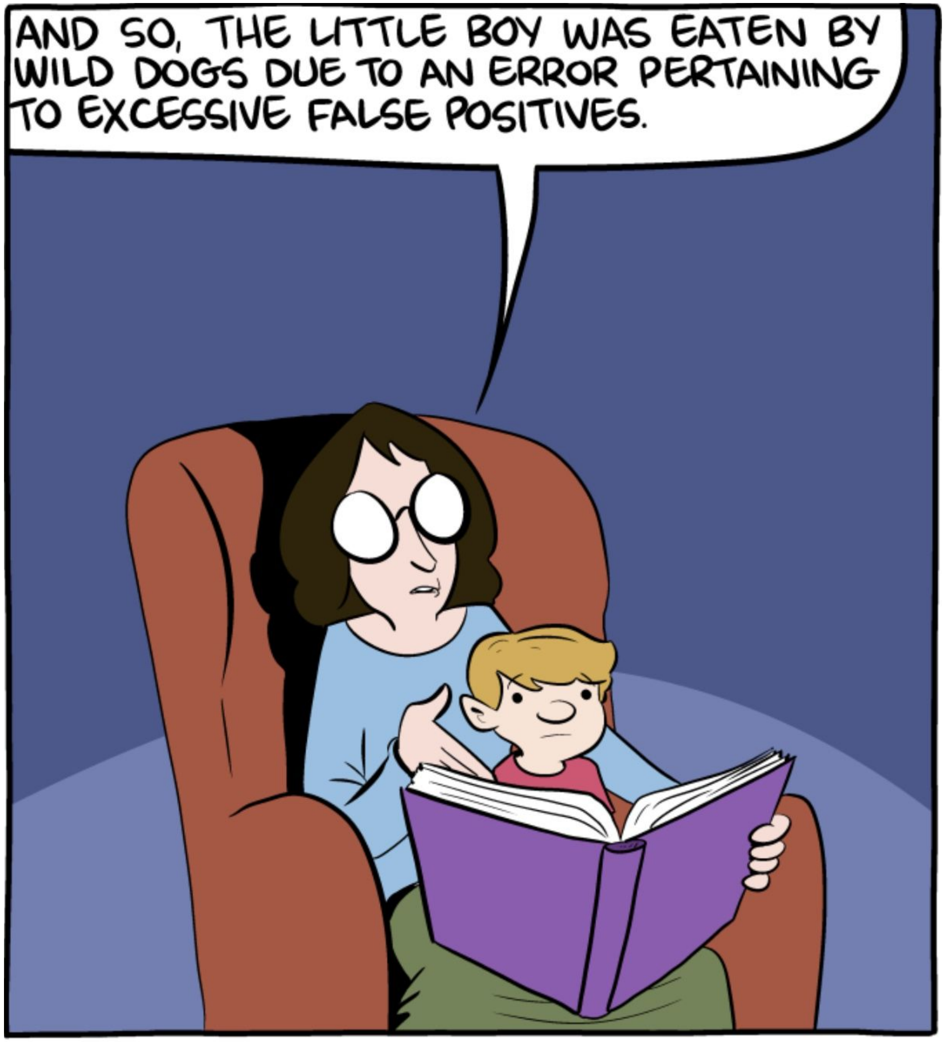
- The upper limit is computed, instead of requiring $p_{s+b} \leq \alpha$, on the modified statistic $CL_s \leq \alpha$:

$$CL_s = \frac{p_{s+b}}{1 - p_b}$$

- Since $1 - p_b \leq 1$, $CL_s \geq p_{s+b}$, hence upper limits computed with the CL_s method are always **conservative**

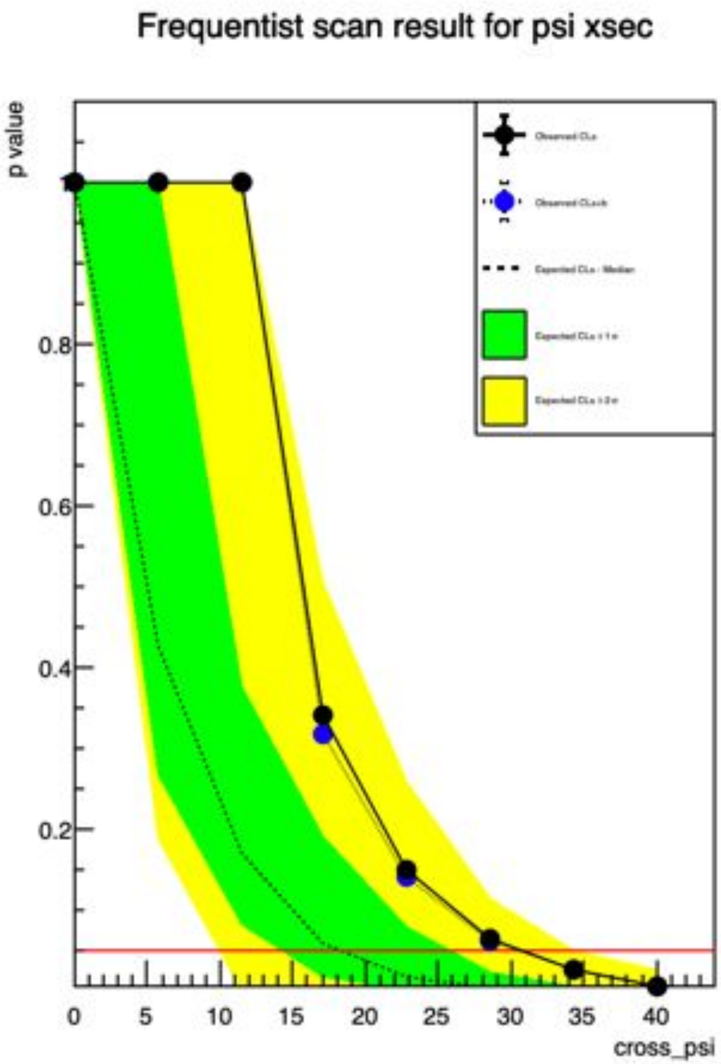
Note: $\lambda \leq \lambda^{\text{obs}}$ implies $-2\ln\lambda \geq \lambda^{\text{obs}}$

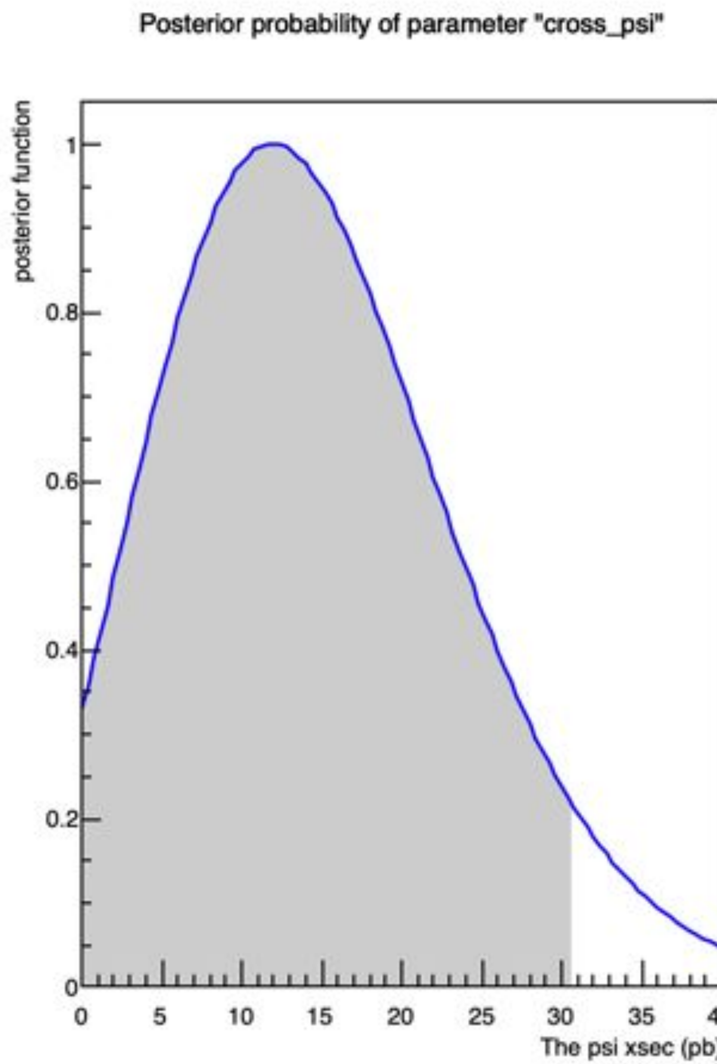
Intermezzo



Statisticians have a different version of
The Boy Who Cried Wolf.

Result of exercise #3





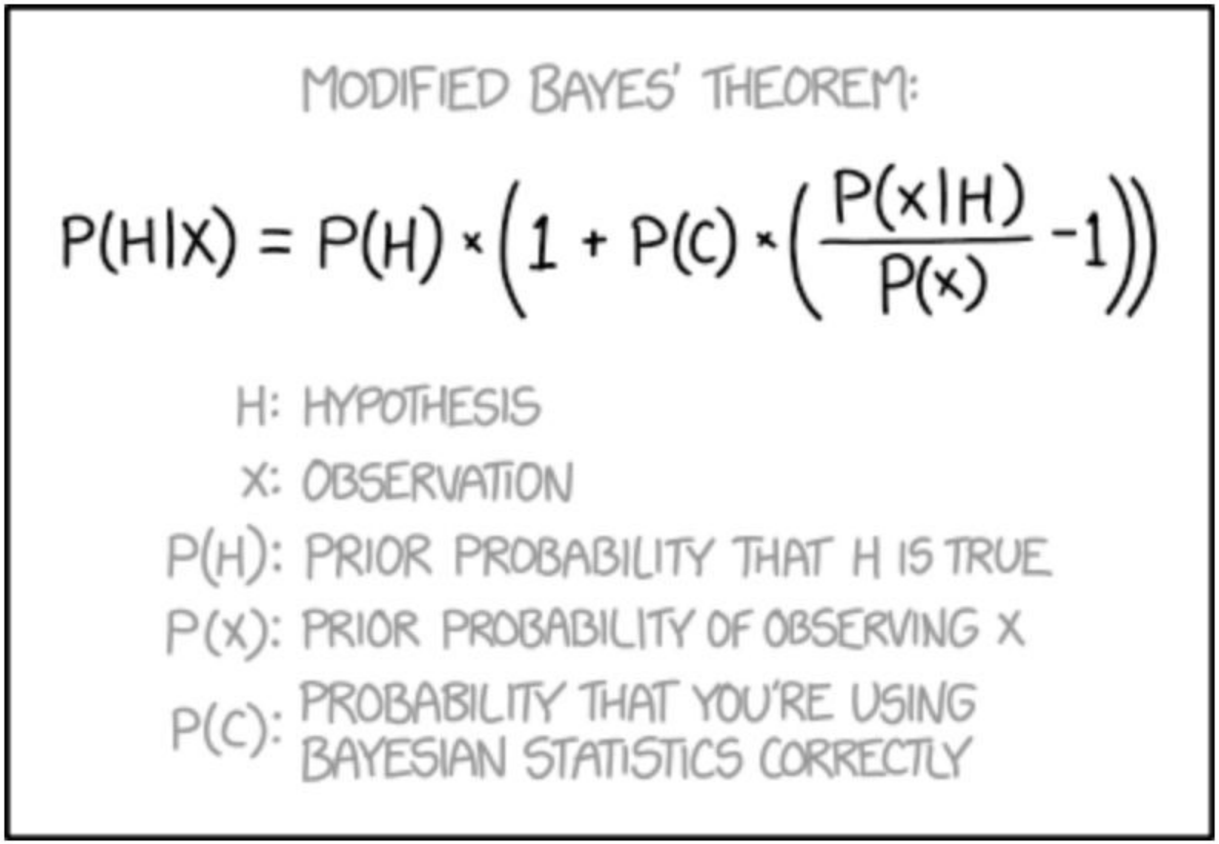
Exercise #4

Let's now go to a scenario where we have a significant excess

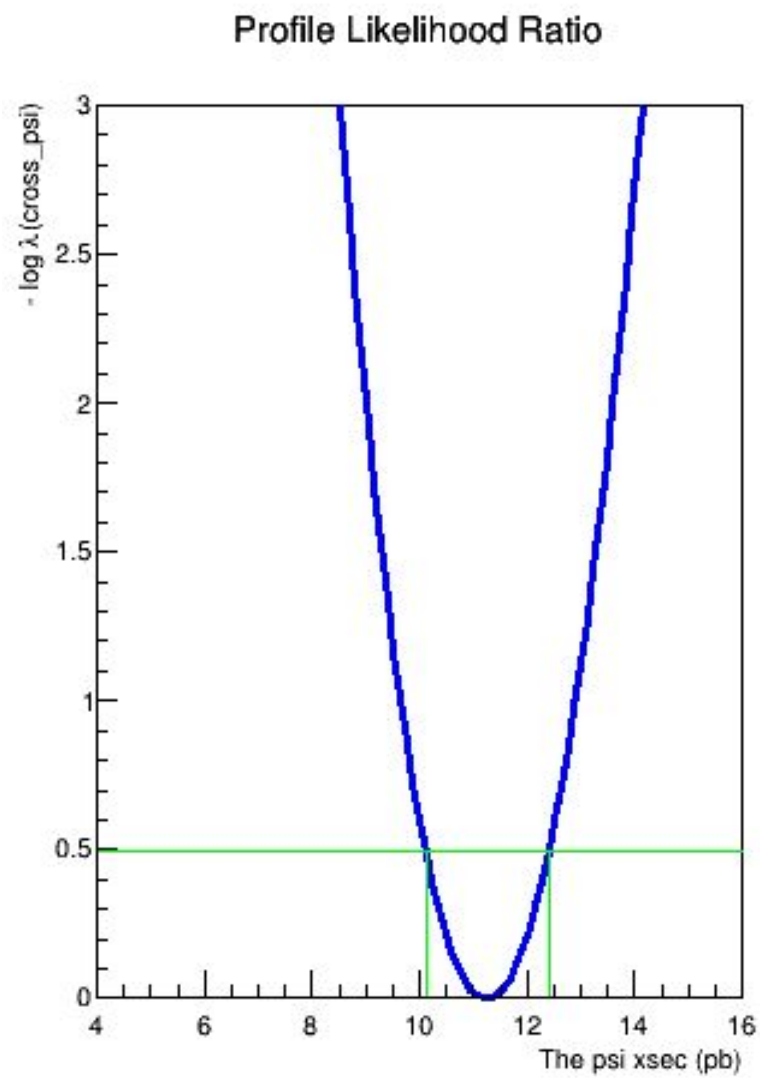
- Get the full 2010 statistics file
- Rerun exercise 0 and 2 to recreate the workspace and calculate the new significance

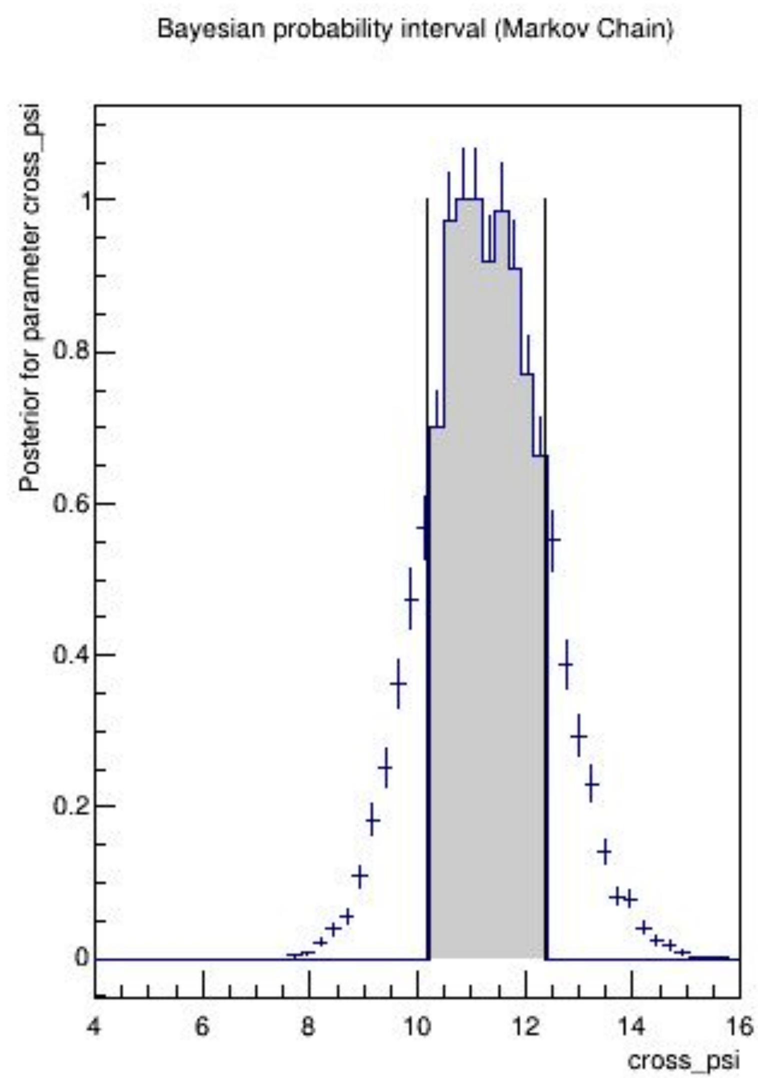
Now we can measure the properties of our discovery

Intermezzo


MODIFIED BAYES' THEOREM:
$$P(H|X) = P(H) \times \left(1 + P(C) \times \left(\frac{P(X|H)}{P(X)} - 1\right)\right)$$
H: HYPOTHESIS
X: OBSERVATION
P(H): PRIOR PROBABILITY THAT H IS TRUE
P(X): PRIOR PROBABILITY OF OBSERVING X
P(C): PROBABILITY THAT YOU'RE USING BAYESIAN STATISTICS CORRECTLY

Result of exercise #4





Blinding

In general, should not run interpretation on data before analysis strategy is decided

→ Check out the ~ 80 GeV top quark “discovery” for a cautionary tale…

RooFit has tools to ease blinding. For example

```
var1 = ROOT.RooUnblindOffset(“var1”,”blinded var”,”Daredevil”,1.0,my_poi)
```

my_poi shifted by unknown quantity (seeded by “Daredevil” of same order)

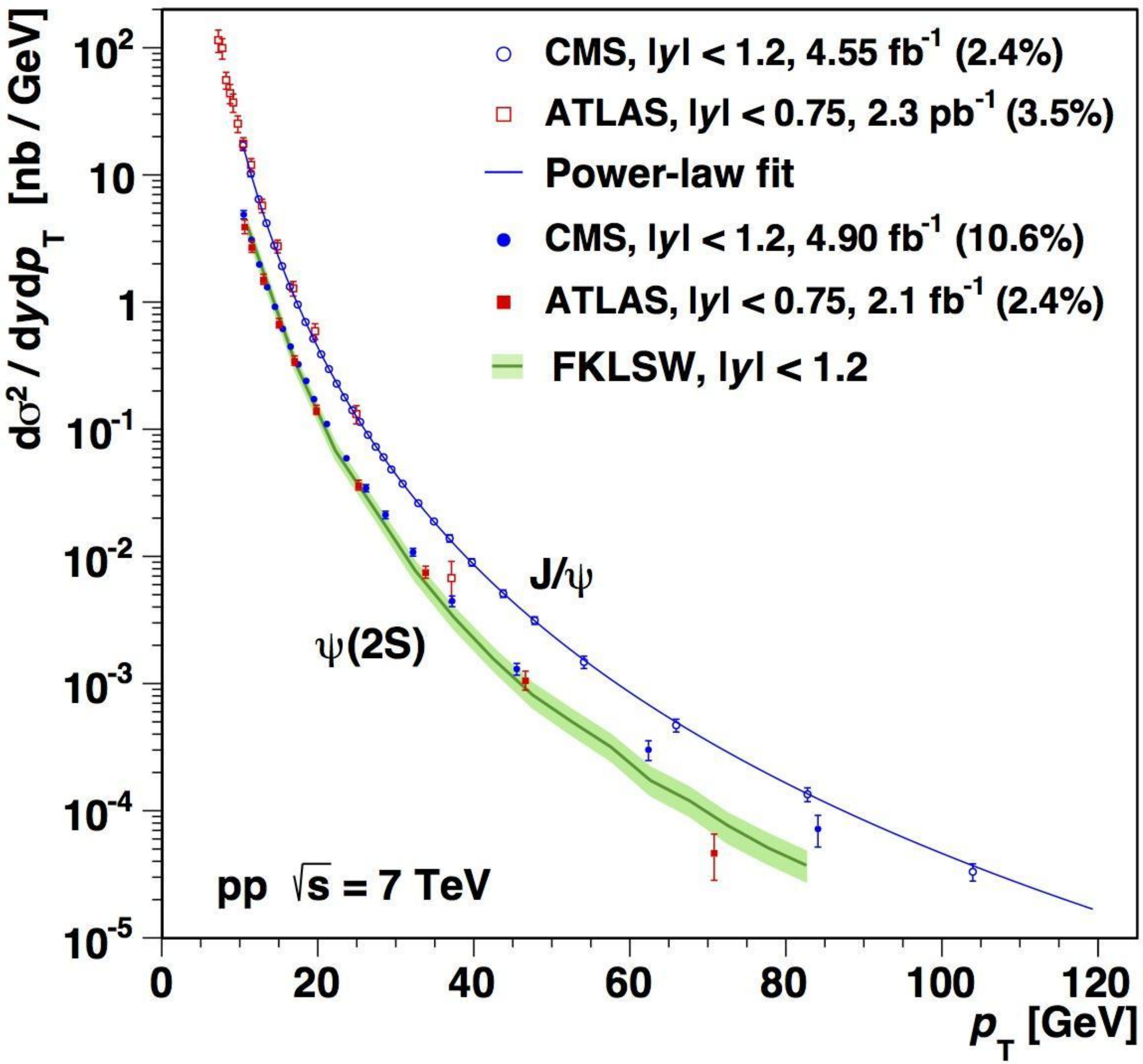
Can be used to study shifts (systematics!) directly on data while remaining blind

ψ(2S) cross section

From CMS-BPH-14-001

Remember that

BR(Ψ(2S)→μμ) ~ 8*10^{-3}



Exercise #5

Let's see how to incorporate systematic uncertainties in this workflow

Let's assume we have a 10% uncertainty on the efficiency

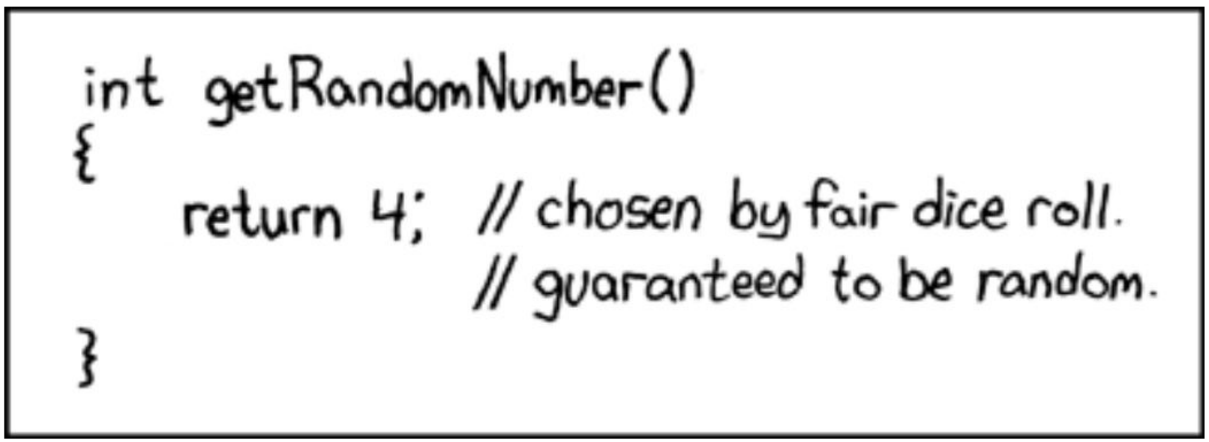
One possible way is to reparametrize the efficiency as

$$\sigma_{eff} = k * \sigma$$

Scale factor k==1 for no uncertainty

Assuming a Gaussian behavior for this uncertainty, one can add this term to the total PDF

Intermezzo


int getRandomNumber()
{
return 4; // chosen by fair dice roll.
// guaranteed to be random.
}

That’s all folks!