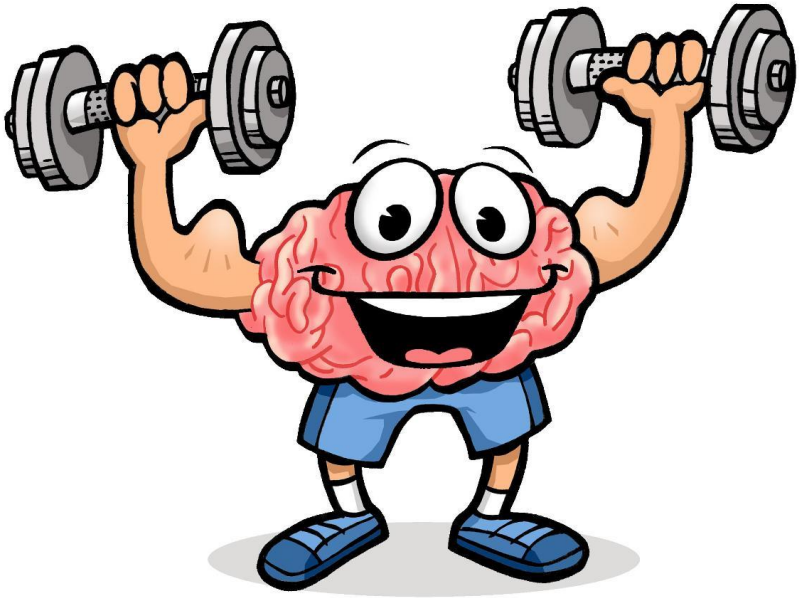
Perugia, June 8th 2017

Statistics for data analysis: Part 2, with more root exercises



Tommaso Dorigo
INFN Padova

Contents of the hands-on lesson

1. Probabilities of Poisson-distributed data
   - with and without **nuisances**
2. Modeling distributions: the **F-test**
3. Understanding confidence intervals
   - bounded parameter, Gaussian measurement
   - **(flip-flopping** and undercoverage)
4. Hypothesis testing and goodness of fit
5. Stuff I won't discuss, but you still find in this file for reference

Mind the underscores → they are where you see a space in the name

Code for exercises in:
http://www.pd.infn.it/%7Edorigo/Poisson_prob_fix.C
http://www.pd.infn.it/%7Edorigo/Poisson_prob_fluct.C
http://www.pd.infn.it/%7Edorigo/F_test_commented_exercise.C
http://www.pd.infn.it/%7Edorigo/F_test_commented.C
http://www.pd.infn.it/%7Edorigo/FlipFlop_exercise.C
http://www.pd.infn.it/%7Edorigo/FlipFlop.C
http://www.pd.infn.it/%7Edorigo/Coverage.C
http://www.pd.infn.ig/%7Edorigo/Die3a.C (and Die.C and Die2.C)

1 – Probabilities of Poisson data

Exercise 1 – Poisson probabilities

We want to write a root macro that inputs expected background counts B (with no error) and observed events N, and computes the probability of observing at least N, and the corresponding number of sigma Z for a Gaussian one-tailed test.

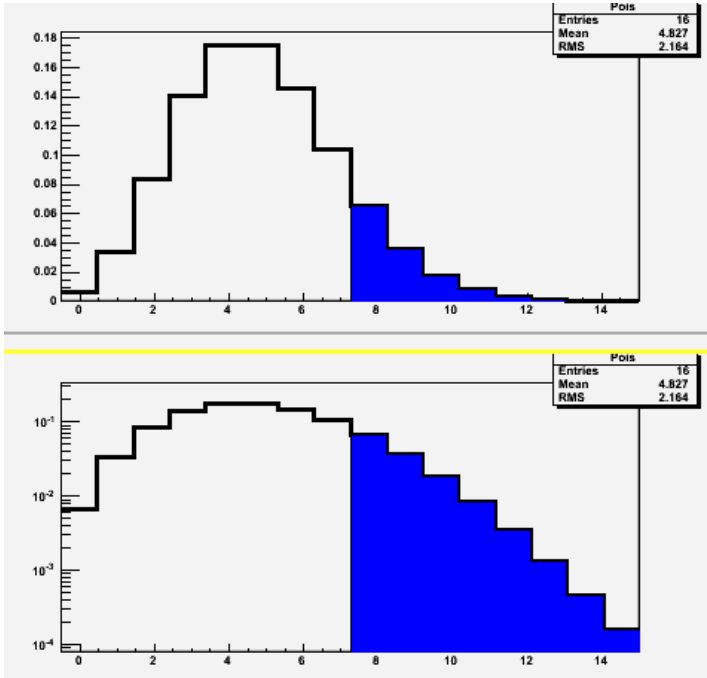
The p-value calculation should be straightforward: just sum from 0 to N-1 the values of the Poisson (computing the factorial as you go along in the cycle), and derive p as 1-sum.

RECALL:

$$P(n;\mu) = \frac{\mu^n e^{-\mu}}{n!}$$

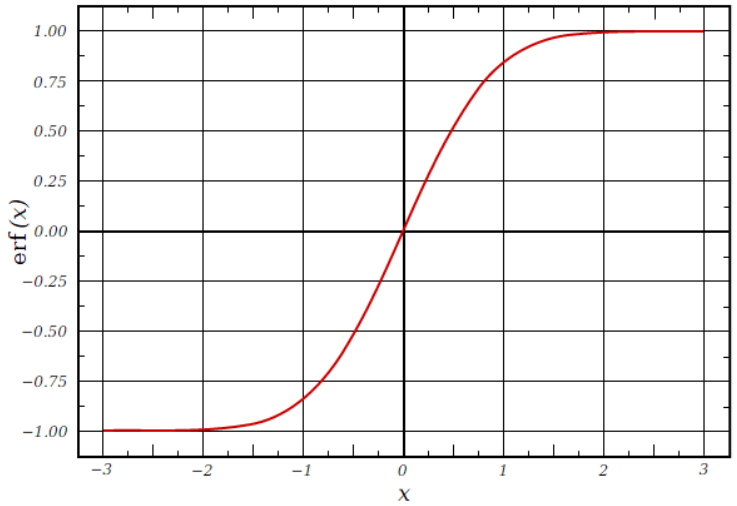
Deriving the number of sigmas that p corresponds to requires the inverse
error function ErfInverse(x) as
Z = sqrt(2) * ErfInverse(1-2p)
(it should be available as TMath::ErfInverse(double))

You can also fill two distributions, one with the Poisson(B), the other with only the bins >=N filled (and with SetFillColor(kBlue) or something) and plot them overimposed, to get something like the graph on the right (top: linear y scale; bottom: log y scale)



Parenthesis – Erf and ErfInverse

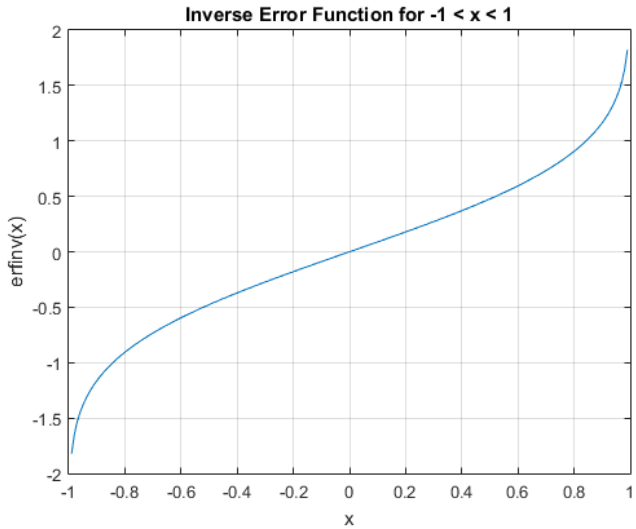
$$\operatorname{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2}\,\mathrm{d}t.$$



- The error function and its inverse are useful tools in statistical calculations – you will encounter them frequently.
- The Erf can be used to obtain the integral of a Gaussian as

$$\Phi(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} e^{\frac{-t^2}{2}}\,\mathrm{d}t = \frac{1}{2}\left[1 + \operatorname{erf}\left(\frac{x}{\sqrt{2}}\right)\right]$$

The erfinverse function is used to convert alpha values into number of sigmas. We will see examples of that later on.



One possible implementation

```
// Macro that computes p-value and Z-value
// of N observed vs B predicted Poisson counts
// ----------------------------------------------------------------------
void Poisson_prob_fix (double B, double N) {

  int maxN = N*3/2; // extension of x axis
  if (N<20) maxN=2*N;
  TH1D * Pois = new TH1D ("Pois", "", maxN, -0.5, maxN-0.5);
  TH1D * PoisGt = new TH1D ("PoisGt", "", maxN, -0.5, maxN-0.5); // we also fill a “highlighted” portion

  double sum=0.;
  double fact=1.;
  for (int i=0; i<maxN; i++) {
   if (i>1) fact*=i;  // calculate factorial
   poisson = exp(-B)*pow(B,i)/fact;
   if (i<N) sum+= poisson; // calculate 1-tail integral
   Pois->SetBinContent(i+1,poisson);
   if (i>=N) PoisGt->SetBinContent(i+1,poisson);
  }
  double P=1-sum;  // get probability of >=N counts
  double Z = sqrt(2) * TMath::ErfInverse(1-2*P);

   cout << "P of observing N=" << N << " or more events if B="  << B << " : P= " << 1-sum << endl;
   cout << "This corresponds to " << Z << " sigma for a Gaussian one-tailed test." << endl;

  Pois->SetLineWidth(3);
  PoisGt->SetFillColor(kBlue);
  TCanvas* T = new TCanvas ("T","Poisson distribution", 500, 500);
  // Plot the stuff
  T->Divide(1,2);
  T->cd(1);
  Pois->Draw();
  PoisGt->Draw("SAME");
  T->cd(2);
  T->GetPad(2)->SetLogy();
  Pois->Draw();
  PoisGt->Draw("SAME");
 }
```

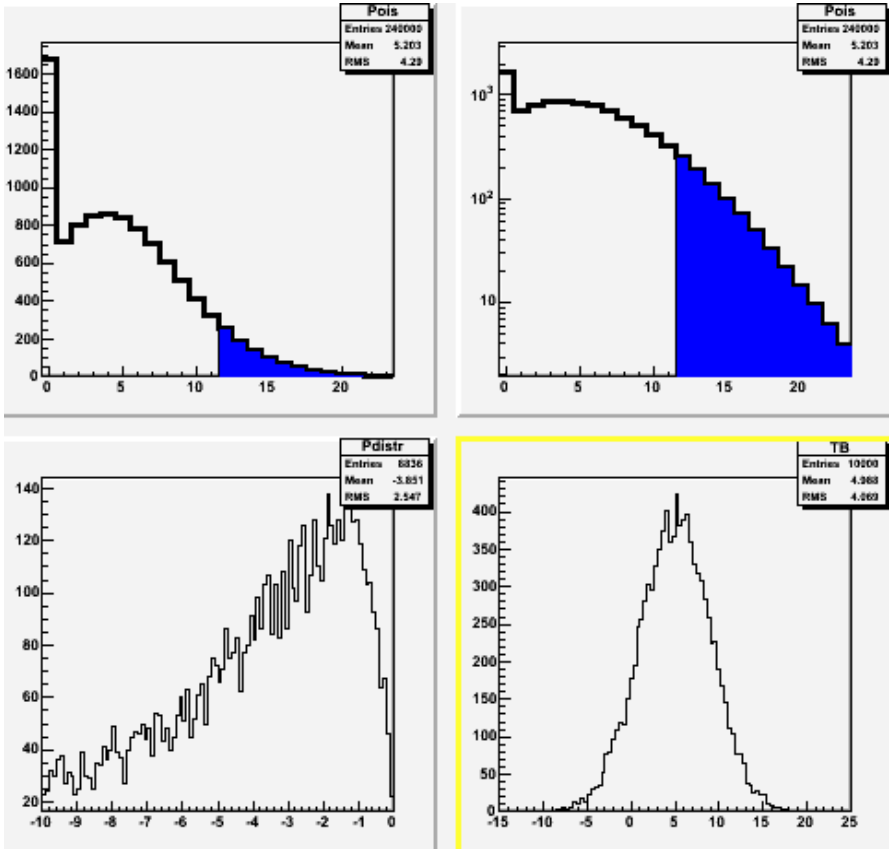
Adding a nuisance

- Let us assume now that B' is not fixed, but known to some accuracy σ_B. We want to add that functionality to our macro. We can start with a Gaussian uncertainty.

You just have to throw a random number $B=G(B',\sigma_B)$ to set B, and collect a large number (say 10k) of p-values as before, then take the average of them.
(why the average ? Would the median be better ?)

Upon testing it, you will discover that you need to enforce that B be non-negative. What we do with the negative B determines the result we get, so we have to be careful, and ask ourselves what exactly do we mean when we say, e.g., "B=2.0+-1.0"

Example below: B=5+-4, N=12



Possible implementation

```
void Poisson_prob_fluct (double B, double SB, double N) {
 double Niter=10000;
 int maxN = N*3/2;
 if (N<20) maxN=2*N;
 TH1D * Pois = new TH1D ("Pois", "", maxN, -0.5, maxN-0.5);
 TH1D * PoisGt = new TH1D ("PoisGt", "", maxN, -0.5, maxN-0.5);
 // We throw a random Gaussian smearing SB to B, compute P,
 // and iterate Niter times; we then study the distribution
 // of p-values, extracting the average
 double Psum=0;
 TH1D * Pdistr = new TH1D ("Pdistr", "", 100, -10., 0.);
 TH1D * TB = new TH1D ("TB", "",100, B-5*SB,B+5*SB);
 cout << "Start of cycle" << endl;
 for (int iter=0; iter<Niter; iter++) {
  // Extract B from G(B,SB)
  double thisB = gRandom->Gaus(B,SB);
  TB->Fill(thisB);  // We keep track of the pdf of the background
  if (thisB<=0) thisB=0.; // Note this – what if we had rethrown it ?
  double sum=0.;
  double fact=1.;
  for (int i=0; i<maxN; i++) {
   if (i>1) fact*=i;
   double poisson = exp(-thisB)*pow(thisB,i)/fact;
   if (i<N) sum+= poisson;
   Pois->Fill((double)i,poisson);
   if (i>=N) PoisGt->Fill((double)i,poisson);
  }
  double thisP=1-sum;
  if (thisP>0) Pdistr->Fill(log(thisP));
  Psum+=thisP;
 }
 double P = Psum/Niter; // we use the average for our inference here
 double Z = sqrt(2) * ErfInverse(1-2*P);
 cout << "Expected P of observing N=" << N << " or more events if
      B="
    << B << "+-" << SB << " : P= " << P << endl;
 cout << "This corresponds to " << Z << " sigma for a Gaussian one-
      tailed test." << endl;

 // Plot the stuff
 Pois->SetLineWidth(3);
 PoisGt->SetFillColor(kBlue);
 TCanvas* T = new TCanvas ("T","Poisson distribution", 500, 500);
 T->Divide(2,2);
 T->cd(1);
 Pois->DrawClone();
 PoisGt->DrawClone("SAME");
 T->cd(2);
 T->GetPad(2)->SetLogy();
 Pois->DrawClone();
 PoisGt->DrawClone("SAME");
 T->cd(3);
 Pdistr->DrawClone();
 T->cd(4);
 TB->Draw();
}
```

2 – Finding the right model

Finding the right model

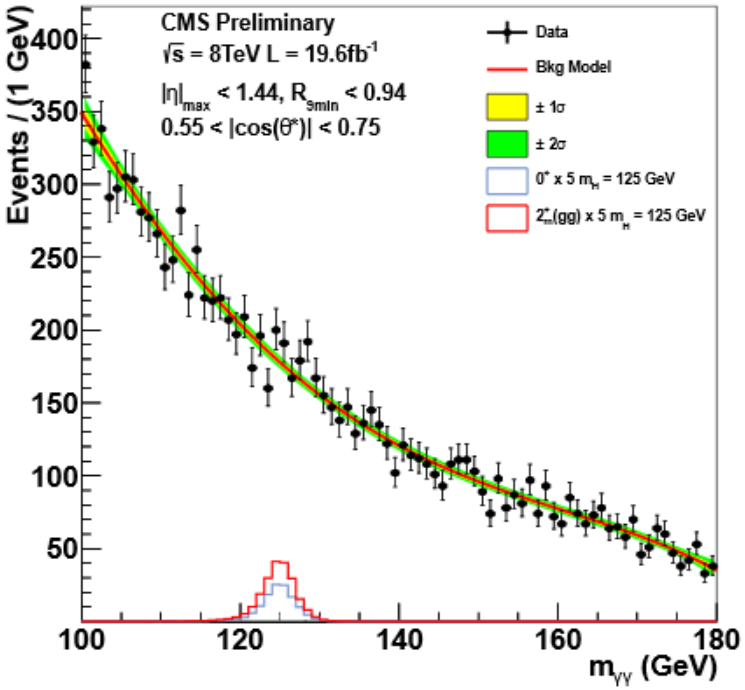
- Often in HEP, astro-hep etc. we do not know what is the true functional form the data are drawn from
  - Can in specific cases use MC simulations; not always
- Extracting inference from a spectrum is thus limited:
  - "I see a deformation in the spectrum"
  - "A deformation from what ?"

Nonetheless, we routinely use e.g. mass spectra to search for new particles and we "guess" the data shape

EG: Higgs H→γγ searches in ATLAS and CMS !

These searches have trouble simulating the reconstructed mass spectrum so families of possible "background shapes" are used

The modeling of the background shape is thus a difficult problem



Fisher's F-test

- Suppose you have no clue of the real functional form followed by your data (**n** points)
  - or even suppose you know only its general form (e.g. polynomial, but do not know the degree)
- You may try a function $\mathbf{f_1(x;\{p_1\})}$ and find it produces a good fit; however, you are unsatisfied about some additional feature of the data that appear to be systematically missed by the model
- You may be tempted to try a more complex function –usually by adding one or more parameters to $\mathbf{f_1}$
  - **this ALWAYS improves the absolute χ^2**, as long as the new model "**embeds**" the old one (the latter means that given **any** choice of $\mathbf{\{p_1\}}$, there exists a set $\mathbf{\{p_2\}}$ such that $\mathbf{f_1(x;\{p_1\})==f_2(x;\{p_2\})}$

How to decide whether $\mathbf{f_2}$ is more motivated than $\mathbf{f_1}$, or rather, that the added parameters are doing something of value to your model ?

Don't use your eye! *Doing so may result in choosing more complicated functions than necessary to model your data, with the result that your statistical uncertainty (e.g. on an extrapolation or interpolation of the function) may abnormally shrink, at the expense of a modeling systematics which you have little hope to estimate correctly.*

→ Use the F-test: the function F

$$F = \frac{\dfrac{\sum_i (y_i - f_1(x_i))^2 - \sum_i (y_i - f_2(x_i))^2}{p_2 - p_1}}{\dfrac{\sum_i (y_i - f_2(x_i))^2}{n - p_2}}$$

has a Fisher distribution **if** the added parameter is **not** improving the model.

$$f(F;\nu_1,\nu_2) = \frac{\nu_1^{\nu_1/2}\nu_2^{\nu_2/2}\Gamma\left(\frac{\nu_1+\nu_2}{2}\right)}{\Gamma(\nu_1/2)\Gamma(\nu_2/2)} \frac{F^{\frac{\nu_1}{2}-1}}{(\nu_1+\nu_2 F)^{\frac{\nu_1+\nu_2}{2}}}$$

Example of F-test

Imagine you have the data shown on the right, and need to pick a functional form to model the underlying p.d.f.

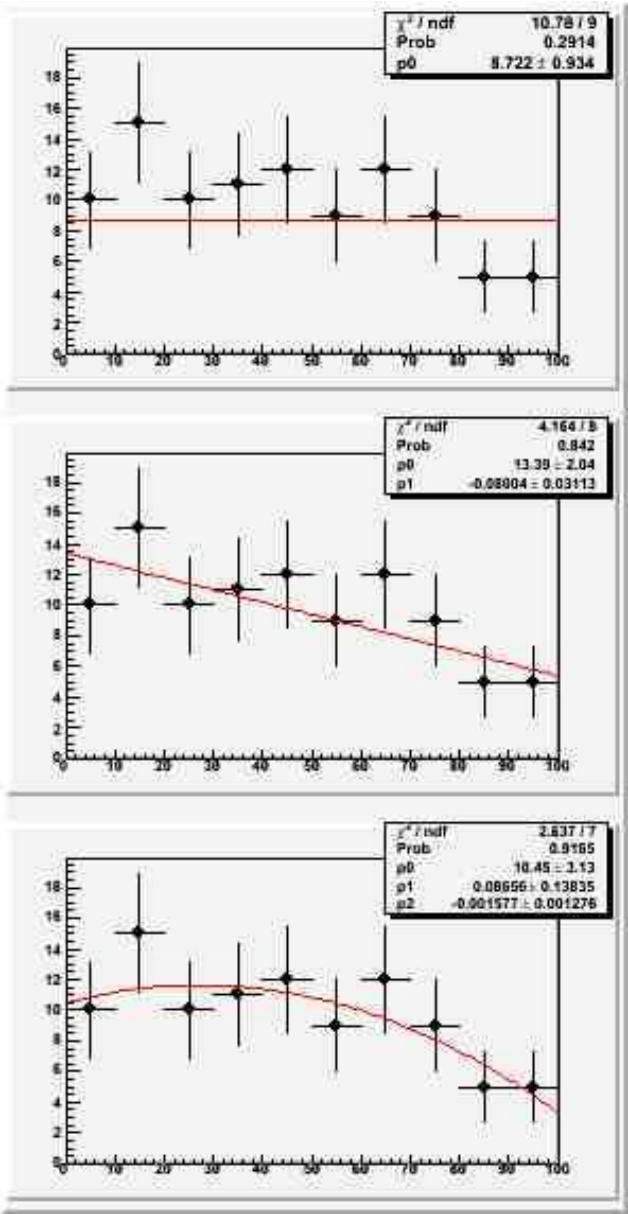
At first sight, any of the three choices shown produces a meaningful fit. P-values of the respective χ^2 are all reasonable (0.29, 0.84, 0.92)

The F-test allows us to pick the right choice, by determining whether the additional parameter in going from a constant to a line, or from a line to a quadratic, is really needed.

We need to pre-define a **size α** of our test: we will reject the "null hypothesis" that the additional parameter is useless if **p<α**. **Let us pick α=0.05 (ARBITRARY CHOICE!)**.

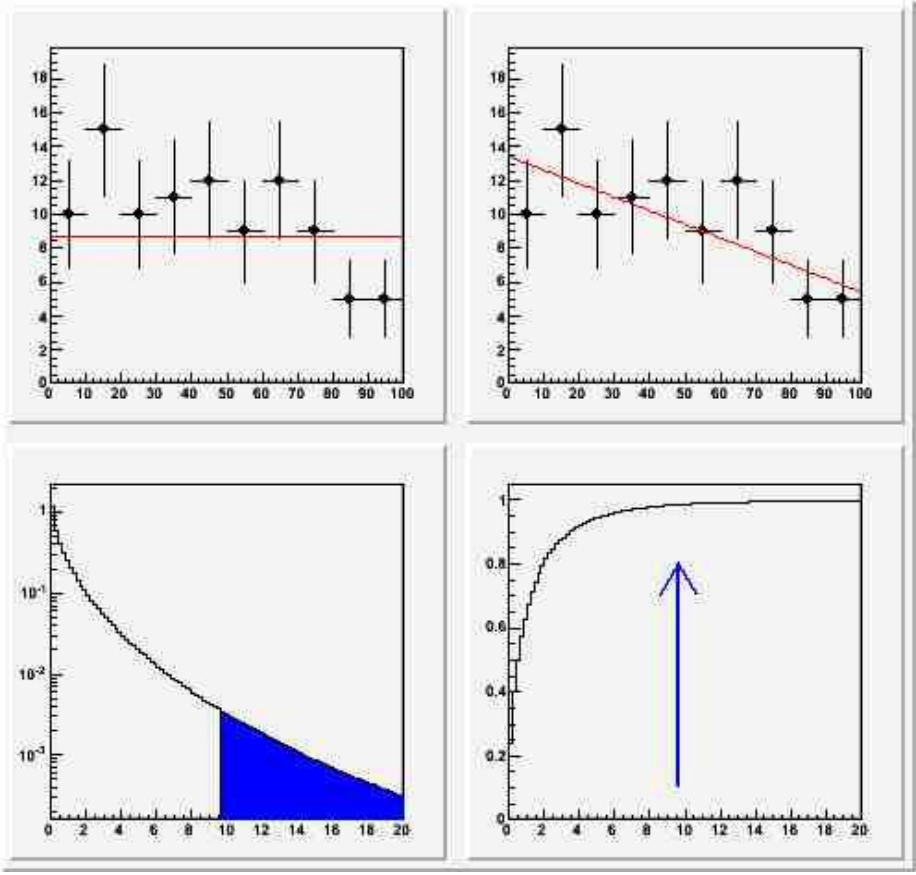
We define p as the *probability that we observe a F value at least as extreme as the one in the data*, if it is drawn from a Fisher distribution with the corresponding degrees of freedom.

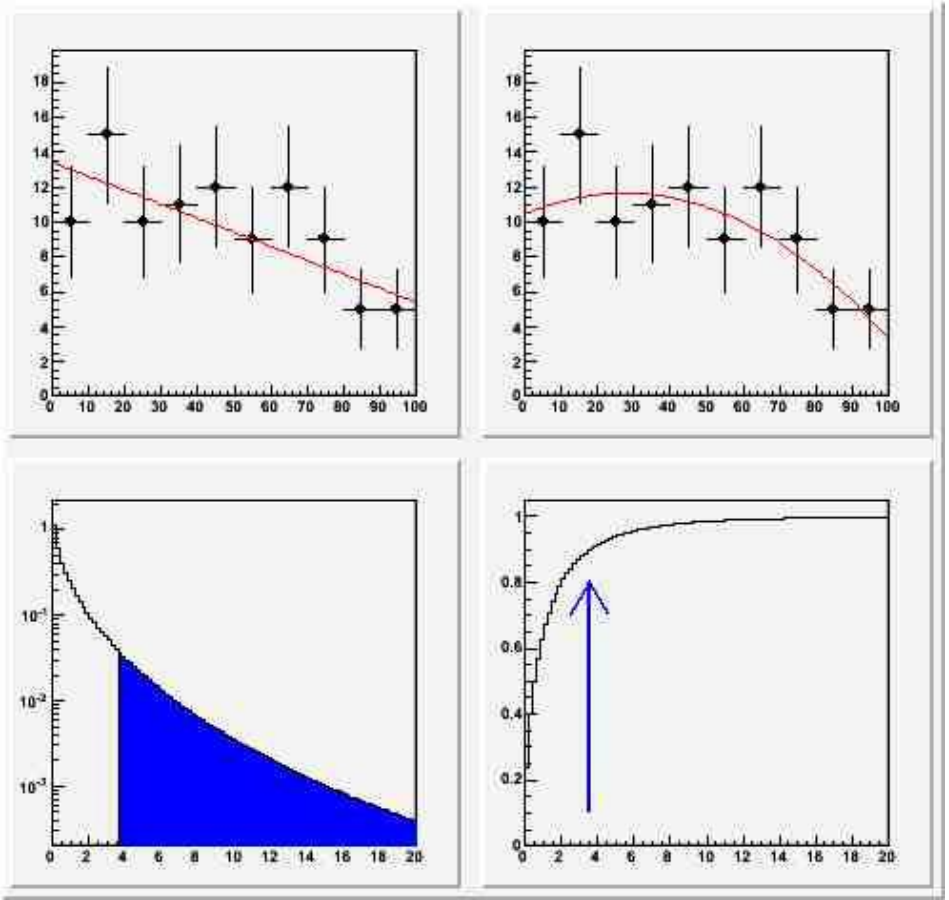
Note that we are implicitly also selecting a "**region of interest**" (large values of F)!



How many of you would pick the constant model ?
The linear ? The quadratic ?
Would your choice change if α=0.318 (1-sigma)?

The test between constant and line yields **p=0.0146**: there is *evidence* (according to our choice of α) against the null hypothesis (that the additional parameter is useless), so we reject the constant pdf and take the linear fit





The test between linear and quadratic fit yields **p=0.1020**: there is no evidence against the null hypothesis (that the additional parameter is useless). We therefore keep the linear model.

Playing with the F test

- The provided code can be used to get familiar with the use of the F test.
- Simple exercise: add functionality to generate exponentially falling data; check when linear model breaks down, when quadratic model also breaks down, etcetera, as a function of
  - number of events in histogram
  - number of bins in histogram
  - size of the test

What you need:

1) understand what the code does
2) understand how to generate exponentially falling data
3) code it
4) choose suitable upper range of histogram

In particular, you need to use the integral function of the pdf (we assume gRandom only provides uniformly-distributed random numbers!)

3 - Confidence intervals

The simplest confidence interval: +- 1 standard error

- The **standard deviation** is used in most simple applications as a *measure of the uncertainty of a point estimate*
- For example: N observations $\{x_i\}$ of random variable x with hypothesized pdf $f(x;\theta)$, with θ unknown. The set $X=\{x_i\}$ allows to construct an estimator $\theta^*(X)$
- Using an analytic method, or the RCF bound, or a MC sampling, one can estimate the standard deviation of θ^*
- The value $\theta^* \pm \sigma^*_{\theta^*}$ is then reported. What does this mean ?
- It means that in repeated estimates based on the same number N of observations of x, θ^* would distribute according to a pdf $G(\theta^*)$ centered around a true value θ with a true standard deviation σ_{θ^*}, **respectively estimated** by θ^* and $\sigma^*_{\theta^*}$

Pay att'n

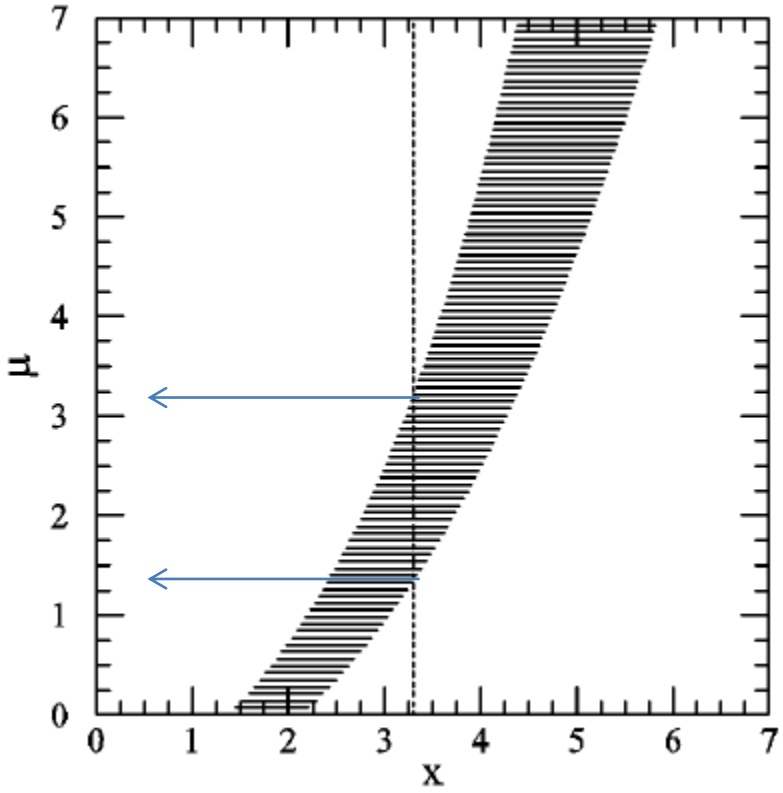
- *In the large sample limit G() is a (multi-dimensional) Gaussian function*
- In most interesting cases for physics G() is not Gaussian, the large sample limit does not hold, 1-sigma intervals do not cover 68.3% of the time the true parameter, and we have better be a bit more tidy in constructing intervals. But we need to have a hunch of the pdf $f(x;\theta)$ to start with!

Neyman's Confidence interval recipe

- Specify a model which provides the probability density function of a particular observable x being found, for each value of the unknown parameter of interest: $p(x|\mu)$
- *Also choose a Type-I error rate α (e.g. 32%, or 5%)*
- For each μ, draw a horizontal *acceptance interval* $[x_1,x_2]$ *such that*

  $$p\,(x \in [x_1,x_2] \mid \mu) = 1 - \alpha.$$

  ***There are infinitely many ways of doing this**: it all depends on what you want from your data*
  - *for upper limits, integrate the pdf from x to infinity*
  - *for lower limits do the opposite*
  - *might want to choose central intervals*
  - *or shortest intervals ?*
- *In general: **an ordering principle** is needed to well-define.*
- Upon performing an experiment, you measure $x=x^*$. *You can then draw a vertical line through it.*



→ The vertical *confidence interval $[\mu_1,\mu_2]$ (with Confidence Level C.L. = 1 -α) is the union of all values of μ for which the corresponding acceptance interval is intercepted by the vertical line.*

Important notions on C. I.'s

What is a vector ? A vector is an element of a vector space (a set with certain properties).

Similarly, a ***confidence interval*** *is defined to be "an element of a confidence set", the latter being a set of intervals defined to have the property of frequentist coverage under sampling!*

Let the unknown true value of μ be μ_t . In repeated experiments, the confidence intervals obtained will have different endpoints $[\mu_1, \mu_2]$, depending on the random variable x.

A fraction C.L. = 1 –α of intervals obtained by Neyman's contruction will contain ("cover") the fixed but unknown μ_t *:* $P(\mu_t \in [\mu_1, \mu_2]) = \text{C.L.} = 1 - \alpha$.

It is important thus to realize two facts:

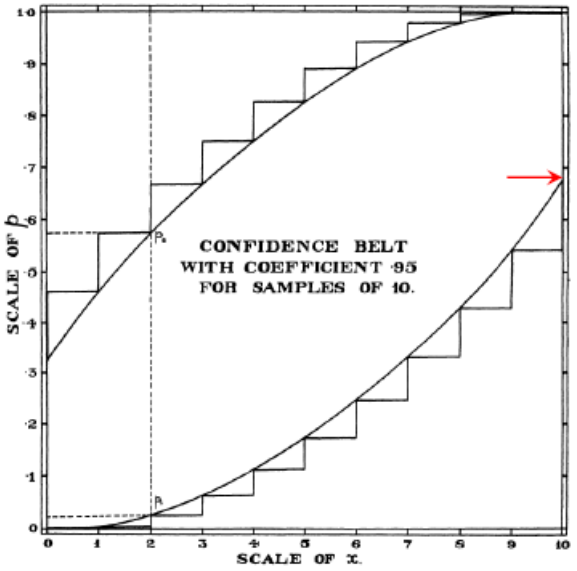
1) **the random variables in this equation are $\mu_1$and μ_2, and not μ_t !**
2) **Coverage is a property of the set, not of an individual interval !** For a Frequentist, the interval either covers or does not cover the true value, regardless of α.

→ Classic **FALSE statement** you should avoid making:
"The probability that the true value is within μ_1 and μ_2 is 68%" !

The confidence interval instead does consist of those values of μ for which the observed x is among the most probable (in sense specified by ordering principle) to be observed.

Also note: "repeated sampling" does not require one to perform the same experiment all of the times for the confidence interval to have the stated properties. Can even be different experiments and conditions! A big issue is what is the **relevant space** of experiments to consider.

More on coverage

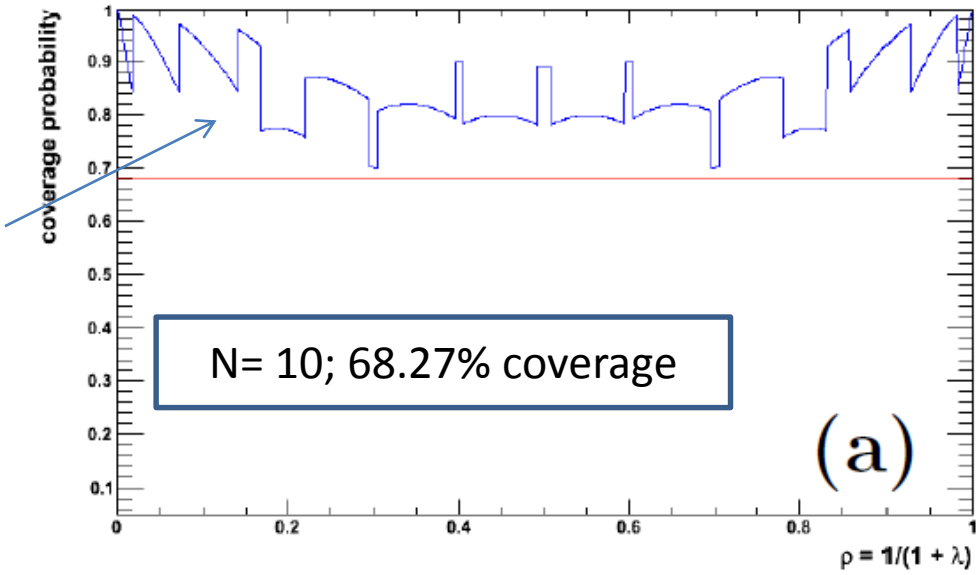


- Coverage is usually guaranteed by the frequentist Neyman construction. But there are some *distinguos* to make
- **Over-coverage**: sometimes the pdf $p(x|\theta)$ is discrete → it may not be possible to find exact boundary values x_1, x_2 for each θ; one thus errs conservatively by including x values (according to one's ordering rule) until $\Sigma_i p(x_i|\theta) > 1-\alpha$

  → θ_1 and θ_2 will **overcover**

- Classical example: Binomial error bars for a small number of trials. A complex problem!
- The (true) variance is $\sigma=\sqrt{\rho(1-\rho)/N}$, but its ESTIMATE $\sigma^*=\sqrt{\rho^*(1-\rho^*)/N}$ fails badly for small N and $\rho^* \to 0,1$
- Clopper-Pearson: intervals obtained from Neyman's construction with a *central interval* ordering rule. They overcover sizeably for some values of the trials/successes.
- Lots of technology to improve properties

  → See [Cousins and Tucker, 0905.3831]



Best practical advice: use "Wilson's score interval" (few lines of code)

Confidence Intervals and Flip-Flopping

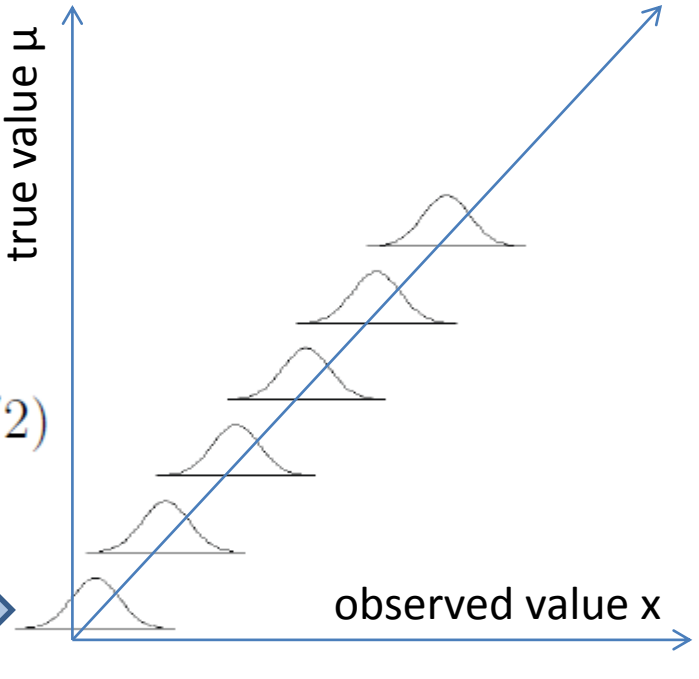
- Here we want to understand a couple of issues that the Neyman construction can run into, for a very common case: the measurement of a bounded parameter and the derivation of upper limits on its value

- Typical observables falling in this category: cross section for a new phenomenon; or neutrino mass

- We take the simplifying assumption that we do a unbiased Gaussian-resolution measurement; we also renormalize measured values such that the variance is 1.0. In that case if μ is the true value, our experiment will return a value x which is distributed as

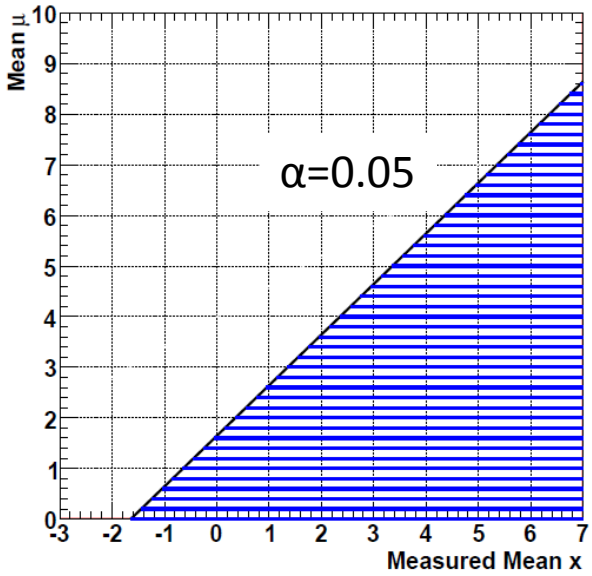
$$P(x|\mu) = \frac{1}{\sqrt{2\pi}} \exp(-(x-\mu)^2/2)$$

Nota bene: x may assume negative values!



Example of Neyman construction

- Gaussian measurement with known sigma (σ=1 assumed in graph) of **bounded parameter** μ>=0
- Classical method for α=0.05 produces upper limit μ<x+1.64σ (or μ<x+1.28σ for α=0.1) (blue lines)
  - for x<-1.64 this results in the **empty set**!
    - in violation of one of Neyman's own demands (confidence set does not contains empty sets)
  - Also note: x<<0 casts doubt on σ=1 hypothesis → rather than telling about value of μ the result could be viewed as a GoF test (analogy with contract bridge). Another possibility is to widen the model to allow σ>1



Flip-flopping: "*since we observe no significant signal, we proceed to derive upper limits…*" As a result, the upper limits undercover ! (Unified approach by Feldman and Cousins solves the issue.)

The attitude that one might take, upon measuring, say, a higgs cross section which is negative (say if your backgrounds fluctuated up such that $N_{obs}<B_{exp}$), is to **quote zero, and report an upper limit** which, in units of sigma, is

$$x^{up}=\text{sqrt}(2)*\text{ErfInverse}(1-2\alpha)$$

where α is the desired confidence level. X^{up} is such that the integral of the Gaussian from minus infinity to x^{up} is 1-α (one-tailed test).

If, however, one finds x>D, where D is one's discovery threshold (say, 3-sigma or 5-sigma), one feels entitled to say one has "measured" a non-zero value of the parameter – a discovery of the Higgs, or a measurement of a non-zero neutrino mass. What the physicist will then report is rather an interval: to be consistent with the chosen test size α, he will then quote central intervals which cover at the same level: $x_{meas}+-E(\alpha/2)$, with

$$E(\alpha) = \text{sqrt}(2)*\text{ErfInverse}(1-2*\alpha).$$

The confidence belt may then take the form shown on the graph on the right.

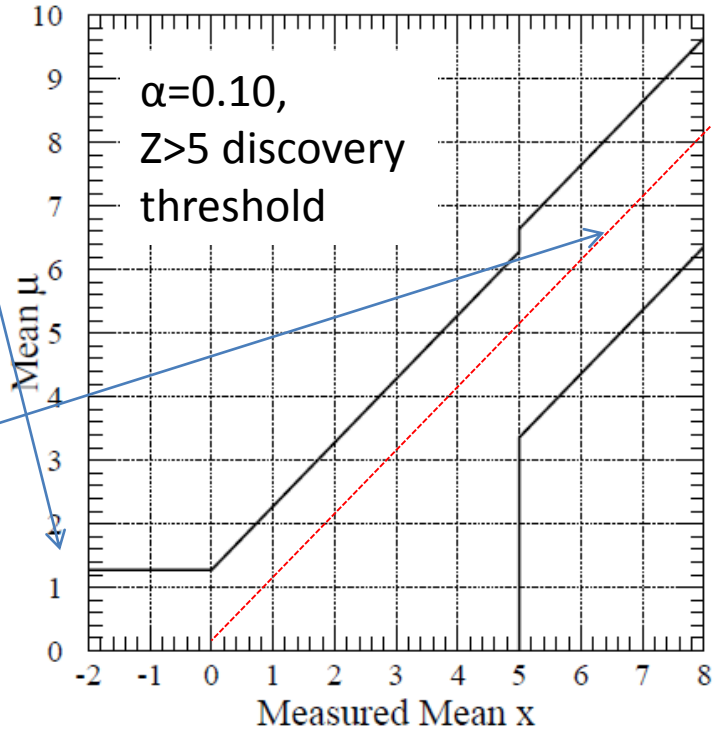
$$\Phi(x)=\frac{1}{2}+\frac{1}{2}erf\left(\frac{x^{up}}{\sqrt{2}}\right)$$

$$2\Phi(x)-1=erf\left(\frac{x^{up}}{\sqrt{2}}\right)$$

$$\frac{x^{up}}{\sqrt{2}}=erfinv\left(2\Phi(x)-1\right)$$

$$x^{up}=\sqrt{2}erfinv[2(1-\alpha)-1]=$$

$$=\sqrt{2}erfinv(1-2\alpha)$$



Coverage of flip-flopping experiment

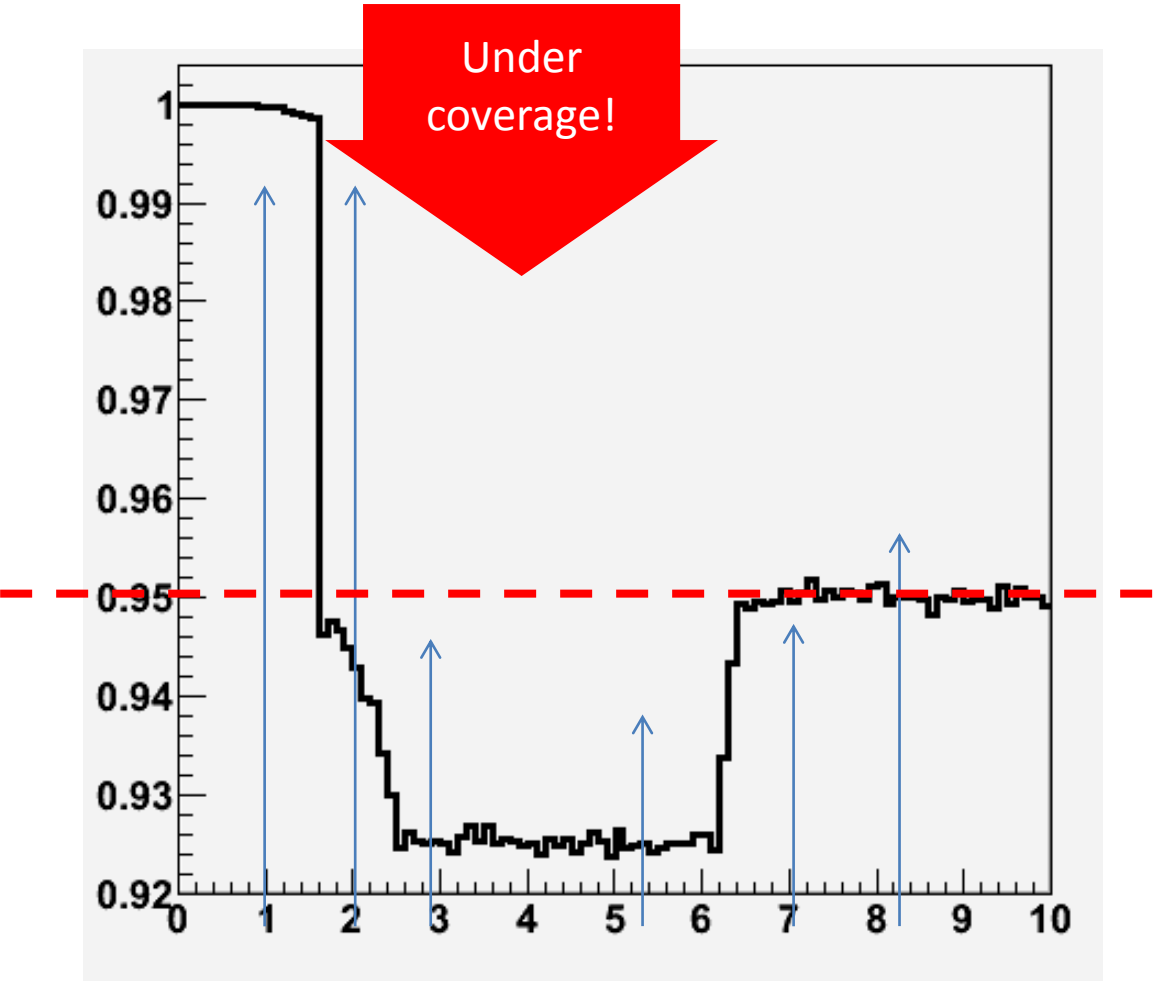
- We want to write a routine that determines the true coverage of the procedure discussed above for a Gaussian measurement of a bounded parameter:
  - $x_{meas}<0$ → quote size-α upper limit as if $x_{meas}=0$, **x^{up}=sqrt(2)*ErfInverse(1-2α)**
  - $0<=x_{meas}<D$→ quote size-α upper limit, **x^{up}=sqrt(2)*ErfInverse(1-2α) + x_{meas}**
  - $x_{meas}>=D$ → quote central value +-α/2 error bars, **x_{meas}+-sqrt(2)*ErfInverse(1-α)**

Guidelines:

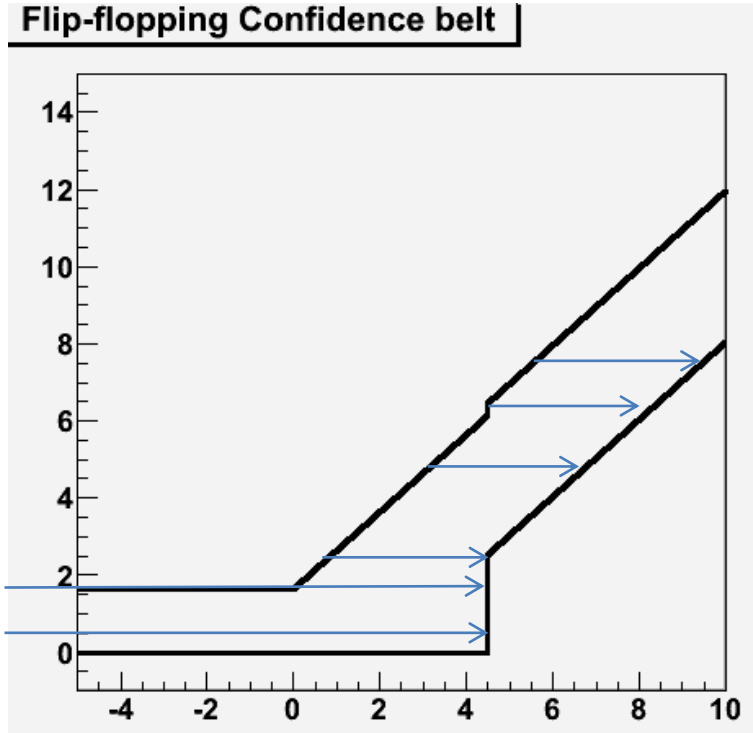
1. insert proper includes (we want to compile it or it'll be too slow)
2. header: pass through it alpha, D, and N_pexp
3. define useful variables and histogram containing coverage values
4. loop on x_true values from 0 to 10 in 0.1 steps → i=0...<100 steps, x_true=0.05+0.1*i
5. for each x_true:
    1. zero a counter C
    2. loop many times (eg. N_pexp, defined in header)
    3. throw x_meas = gRandom->Gaus(x_true,1.)
    4. derive x_down and x_up depending on x_meas:
        1. if x_meas<0 then x_down=0 and x_up = sqrt(2)*ErfInverse(1-2*alpha)
        2. if 0<=x_meas<D then x_down=0 and x_up=x_meas+sqrt(2)*EI(1-2*alpha)
        3. if x_meas>=D then x_down,up = x_meas +- sqrt(2)*EI(1-alpha)
    5. if x_true is in [x_down,x_up] C++
6. fill histogram of coverage at x_true with C/N_pexp
7. plot and enjoy

Results

- Interesting typical case: alpha=0.05 – 0.1, D=4-5
- E.g. alpha=0.05, D=4.5, with N_pexp=100000:



The coverage, for this special case, can actually be computed analytically...
Just determine the integral of the covered area for each region of the belt – see next slide



Coverage.C

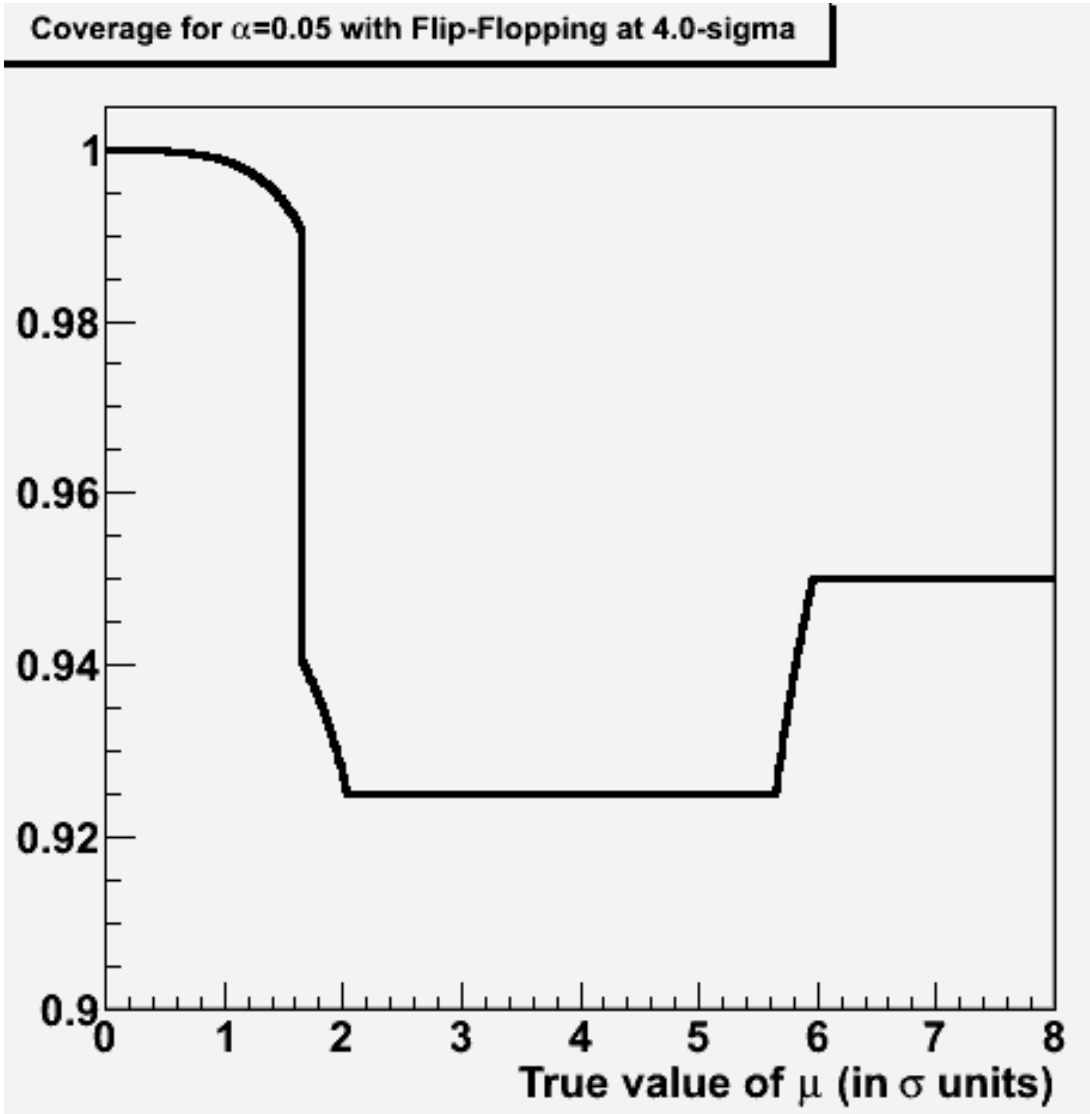
(add at the top the #include commands needed to compile it)

```
void Coverage (double alpha, double disc_threshold=5.) {
// Only valid for the following:
// ----------------------------
 if (disc_threshold-sqrt(2)*ErfInverse(1.-2*alpha/2.)<
    sqrt(2)*ErfInverse(1.-2*alpha)) {
  cout << "Too low discovery threshold, code not suitable. " << endl;
  cout << "Try a larger threshold" << endl;
  return;
 }
 char title[100];
 int idisc_threshold=disc_threshold;
 int fracdiscthresh =10*(disc_threshold-idisc_threshold);
 if (alpha>=0.1) {
  sprintf (title, "Coverage for #alpha=0.%d with Flip-Flopping at %d.%d-
sigma",  (int)(10.*alpha),idisc_threshold, fracdiscthresh);
 } else {
  sprintf (title, "Coverage for #alpha=0.0%d with Flip-Flopping at %d.%d-
            sigma",  (int)(100.*alpha),idisc_threshold, fracdiscthresh);
 }
 TH1D * Cov = new TH1D ("Cov", title, 1000, 0., 2.*disc_threshold);
 Cov->SetXTitle("True value of #mu (in #sigma units)");

// Int Gaus-1:+1 sigma is TMath::Erf(1./sqrt(2.))
 // To get 90% percentile (1.28): sqrt(2)*ErfInverse(1.-2*0.1)
 // To get 95% percentile (1.64): sqrt(2)*ErfInverse(1.-2*0.05)
double cov;
 for (int i=0; i<1000; i++) {
  double mu = (double)i/(1000./(2*disc_threshold))+
   0.5*(2*disc_threshold/1000);
 if (mu<sqrt(2)*ErfInverse(1.-2*alpha)) { // 1.28, so mu within upper 90%
CL
    cov = 0.5*(1+TMath::Erf((disc_threshold-mu)/sqrt(2.)));
  } else if (mu< disc_threshold-sqrt(2)*ErfInverse(1.-2*alpha/2.)) { //
<3.36
    cov = 1.-alpha-0.5*(1.-TMath::Erf((disc_threshold-mu)/sqrt(2.)));
  } else if (mu<disc_threshold+
                sqrt(2)*ErfInverse(1.-2*alpha)) { // 6.28
    cov = 1.-1.5*alpha;
  } else if (mu<disc_threshold+sqrt(2)*ErfInverse(1.-2*alpha/2.) ) { //
6.64) {
    cov = 1.-alpha/2.-0.5*(1+TMath::Erf((disc_threshold-mu)/sqrt(2.)));
  } else { cov = 1.-alpha; }
  Cov->Fill(mu,cov);
 }
 char filename[40];
 if (alpha>=0.1) {
  sprintf(filename,"Coverage_alpha_0.%d_obs_at_%d_sigma.eps",
(int)(10.*alpha),idisc_threshold);
 } else {
  sprintf(filename,"Coverage_alpha_0.0%d_obs_at_%d_sigma.eps",
(int)(100.*alpha),idisc_threshold);
 }
 TCanvas * C = new TCanvas ("C","Coverage", 500,500);
 C->cd();
 Cov->SetMinimum(1.-2*alpha);
 Cov->SetLineWidth(3);
 Cov->Draw();
 C->Print(filename);

 // Now plot confidence belt
 // ------------------------

}
```

Here is e.g. the exact calculation of coverage for flip-flopping at 4-sigma and a test size alpha=0.05

Can get it by running:

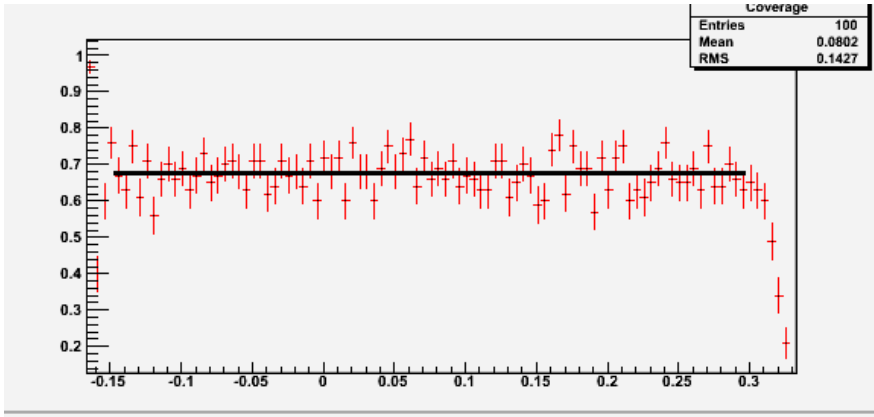
root> .L Coverage.C+;
root> Coverage(0.05,4.);



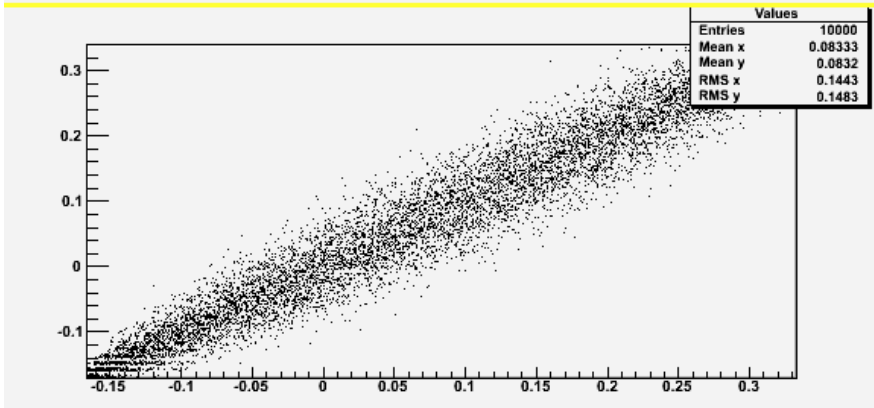
One further example of coverage

- Recall the "loaded die" example. We solved it with a likelihood maximization.
- You may modify it to compute the coverage of the likelihood intervals. → Die3a.C

Just add a TH1D* called “Coverage” and a cycle on the true parameter values, taking care of simulating the die throws correctly taking into account the bias t. Then you count how often the likelihood has the true value within its interval, as a function of the true value.



By running it you will find that the coverage is only approximate for small number of throws, especially when your true value of the parameter t (the “increase in probability” of throws giving a 6) lies close to the boundaries -1/6, 1/3.



Summary of previous slides

- Handling nuisances is easy with toy MC; playing with them in specific problems allows you to get a feeling of how dependent your results are on the size and shape of systematic uncertainties

- Modeling a distribution is a **unsolvable problem** in principle; to do a fair job and get an approximately valid solution one needs at the very least to consider a wide spectrum of functional forms and a **disciplined method** to choose the right one

- **Undercoverage** is equivalent to reporting a smaller uncertainty than what you should have – it **is BAD** especially if you are applying Frequentist tools and standpoint

- Root has tons of built-in functions and tools; getting to know them will make you stronger in your capability of doing statistical analysis on the fly

Possible solutions

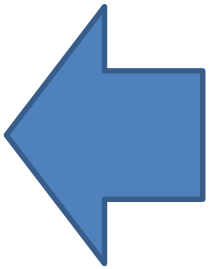
Log-normal nuisance in Poisson test

```
// Macro that computes p-value and Z-value of N observed vs B predicted
// Poisson counts
// ------------------------------------------------------------------
void Poisson_prob_fluct (double B, double SB, double N, int opt=1) {

 double Niter=10000;

 if (opt!=0 && opt!=1) {
  cout << "Please put fourth argument either =0 (Gaussian nuisance)" << endl;
  cout << "or =1 (LogNormal nuisance)" << endl;
  return;
 }

 int maxN = N*2;
 TH1D * Pois = new TH1D ("Pois", "", maxN, -0.5, maxN-0.5);
 TH1D * PoisGt = new TH1D ("PoisGt", "", maxN, -0.5, maxN-0.5);

 // We throw a random Gaussian smearing SB to B, compute P,
 // and iterate Niter times; we then study the distribution
 // of p-values, extracting the average

 double Psum=0;
 TH1D * Pdistr = new TH1D ("Pdistr", "", 100, -10., 0.);
 TH1D * TB = new TH1D ("TB", "",100, B-5*SB,B+5*SB);

 if (opt==0) { // nornal
  mu = B;
  sigma = SB;
 } else { // lognormal
  mu = log(B); // median! omitting the convexity correction -sigma*sigma/2;
  sigma = SB/B;
 }

 for (int iter=0; iter<Niter; iter++) {
  // Extract B from G(B,SB)
  double thisB = gRandom->Gaus(mu,sigma);  // normal
  if (opt==1) thisB = exp(thisB); // lognormal

  TB->Fill(thisB);
  if (thisB<=0) thisB=0.;
  double sum=0.;
  double fact=1.;
  for (int i=0; i<maxN || (opt==0 && i<B+6*SB) || (opt==1 &&
       i<mu+10*sigma); i++) {
   if (i>1) fact*=i;
   double poisson = exp(-thisB)*pow(thisB,i)/fact;
   if (i<N) sum+= poisson;
   Pois->Fill((double)i,poisson);
   if (i>=N) PoisGt->Fill((double)i,poisson);
  }
  double thisP=1-sum;
  if (thisP>0) Pdistr->Fill(log(thisP));
  Psum+=thisP;
 }
 double P = Psum/Niter;
 double Z = sqrt(2) * ErfInverse(1-2*P);

 cout << "Expected P of observing N=" << N << " or more events if B="
    << B << "+-" << SB << " : P= " << P << endl;
 cout << "This corresponds to " << Z << " sigma for a Gaussian one-tailed
       test." << endl;
```

Exponential model in F-test

```
double y = gRandom->Uniform(0.,1.);
  // Generate histogram of data according to different pdfs
  // ------------------------------------------------------
  if (option==0) {
    // int(0:x) dt = x
    // quindi genero y=uniform(0:1) e prendo
    // x=y*xmax
    Data0->Fill(y*xmax);
    Data1->Fill(y*xmax);
    Data2->Fill(y*xmax);
    Data3->Fill(y*xmax);
  } else if (option==1) {
    // int(0:x) t dt = x^2/2
    // quindi genero y=uniform(0:1) e prendo
    // x=sqrt(2*y*xmax^2/2)
    Data0->Fill(sqrt(y*xmax*xmax));
    Data1->Fill(sqrt(y*xmax*xmax));
    Data2->Fill(sqrt(y*xmax*xmax));
    Data3->Fill(sqrt(y*xmax*xmax));
  } else if (option==2) {
    // int(0:x) t^2 dt = x^3/3
    // quindi genero y=uniform(0:1) e prendo
    // x=pow(y,1/3)*xmax
    Data0->Fill(pow(y,0.33333)*xmax);
    Data1->Fill(pow(y,0.33333)*xmax);
    Data2->Fill(pow(y,0.33333)*xmax);
    Data3->Fill(pow(y,0.33333)*xmax);
  } else if (option==3) {
    // int(0:x) e(-t) dt = (1-e^-x)
    // quindi genero y=uniform e prendo
    // x=-log(1-y*(1-exp(-xmax)))
    Data0->Fill(-log(1-y*(1-exp(-xmax))));
    Data1->Fill(-log(1-y*(1-exp(-xmax))));
    Data2->Fill(-log(1-y*(1-exp(-xmax))));
    Data3->Fill(-log(1-y*(1-exp(-xmax))));
  }
 }
```

For full code, see
http://www.pd.infn.it/%7Edorigo/F_test_commented.C



Piece to be added to former version of code

Coverage of Flip-flopping measurement

```
void FlipFlop (double alpha=0.05, double D=4.5, double Npexp=1000) {

 double x_true;
 double x_meas;
 double sigma = 1;
 double x_down;
 double x_up;
 double covers=0.;
 double EIa = sqrt(2)*TMath::ErfInverse(1-alpha);
 double EI2a= sqrt(2)*TMath::ErfInverse(1-2*alpha);

 TH1D * Coverage_vs_xtrue = new TH1D("Coverage_vs_xtrue", "Coverage vs x_true", 100, 0., 10.);
 TH1D * BeltUp = new TH1D ("BeltUp", "Flip-flopping Confidence belt", 15000, -5.,10.);
 TH1D * BeltDo = new TH1D ("BeltDo", "Flip-flopping Confidence belt", 15000, -5.,10.);

 cout << "Critical values: " << endl;
 cout << "For xmeas < 0 : 0 < xtrue < " << EI2a*sigma << endl;
 cout << "For 0<xmeas<" << D << " : 0 < xtrue < xmeas+"
    << EI2a*sigma << endl;
 cout << "For xmeas>=D : xmeas-" << EIa*sigma << " < xtrue < xmeas+"
    << EIa*sigma << endl;
 cout << endl;
 for (int ix=0; ix<100; ix++) {

  x_true = 0.05 + 0.1*ix;
  covers=0;
  for (int pexp=0; pexp<Npexp; pexp++) {

   // A Gaussian measurement with uncertainty sigma
   x_meas = gRandom->Gaus(x_true,sigma);

   if (x_meas<D) {  // Not significantly different from zero, will report upper limit
        x_down = 0;
        x_up = EI2a*sigma;
        if (x_meas>0) x_up = x_meas + x_up;
   } else {                  // will report an interval
        x_down = x_meas-EIa*sigma;
        x_up = x_meas+EIa*sigma;
   }

// compute coverage
    if (x_true>=x_down && x_true<x_up) covers++;
  }

  Coverage_vs_xtrue->Fill(x_true,covers/Npexp);
 }

 // Belt plot
 for (int i=0; i<15000; i++) {
  x_meas = -4.9995 + i*0.001;
  if (x_meas<0) {
   BeltUp->Fill(x_meas,EI2a);
   BeltDo->Fill(x_meas,0.);
  } else if (x_meas<D) {
   BeltUp->Fill(x_meas,x_meas+EI2a);
   BeltDo->Fill(x_meas,0.);
  } else {
   BeltUp->Fill(x_meas,x_meas+EIa);
   BeltDo->Fill(x_meas,x_meas-EIa);
  }
 }

 gStyle->SetOptStat(0);

 TCanvas * W2 = new TCanvas ("W2", "Coverage of flip-flopping NP construction", 500, 500);
 W2->cd();
 Coverage_vs_xtrue->SetLineWidth(3);
 Coverage_vs_xtrue->Draw();

 TCanvas * W = new TCanvas ("W", "Confidence belt", 500, 500);
 W->cd();
 BeltUp->SetMinimum(-1);
 BeltUp->SetMaximum(15);
 BeltUp->SetLineWidth(3);
 BeltDo->SetLineWidth(3);
 BeltUp->Draw();
 BeltDo->Draw("SAME");
}
```

Exact calculation of coverage

```
void Coverage (double alpha, double disc_threshold=5.) {

 gStyle->SetOptStat(0);

 // Only valid for the following:
 // ----------------------------
 if (disc_threshold-sqrt(2)*ErfInverse(1.-2*alpha/2.)<
    sqrt(2)*ErfInverse(1.-2*alpha)) {
  cout << "Too low discovery threshold, code not suitable. " << endl;
  cout << "Try a larger threshold" << endl;
  return;
 }

 char title[100];
 int idisc_threshold=disc_threshold;
 int fracdiscthresh =10*(disc_threshold-idisc_threshold);
 if (alpha>=0.1) {
  sprintf (title, "Coverage for #alpha=0.%d with Flip-Flopping at %d.%d-sigma",
            (int)(10.*alpha),idisc_threshold, fracdiscthresh);
 } else {
  sprintf (title, "Coverage for #alpha=0.0%d with Flip-Flopping at %d.%d-sigma",
            (int)(100.*alpha),idisc_threshold, fracdiscthresh);

 }
 TH1D * Cov = new TH1D ("Cov", title,
                                          1000, 0., 2.*disc_threshold);
 Cov->SetXTitle("True value of #mu (in #sigma units)");

 // Int Gaus-1:+1 sigma is TMath::Erf(1./sqrt(2.))
 // To get 90% percentile (1.28): sqrt(2)*ErfInverse(1.-2*0.1)
 // To get 95% percentile (1.64): sqrt(2)*ErfInverse(1.-2*0.05)

double cov;
 for (int i=0; i<1000; i++) {
  double mu = (double)i/(1000./(2*disc_threshold))+
   0.5*(2*disc_threshold/1000);
  if (mu<sqrt(2)*ErfInverse(1.-2*alpha)) { // 1.28, so mu within upper 90% CL
   cov = 0.5*(1+TMath::Erf((disc_threshold-mu)/sqrt(2.)));
  } else if (mu< disc_threshold-sqrt(2)*ErfInverse(1.-2*alpha/2.)) { // <3.36
   cov = 1.-alpha-0.5*(1.-TMath::Erf((disc_threshold-mu)/sqrt(2.)));
  } else if (mu<disc_threshold+
              sqrt(2)*ErfInverse(1.-2*alpha)) { // 6.28
   cov = 1.-1.5*alpha;
  } else if (mu<disc_threshold+sqrt(2)*ErfInverse(1.-2*alpha/2.) ) { // 6.64) {
   cov = 1.-alpha/2.-0.5*(1+TMath::Erf((disc_threshold-mu)/sqrt(2.)));
  } else {
   cov = 1.-alpha;
  }
  Cov->Fill(mu,cov);
 }

 char filename[40];
 if (alpha>=0.1) {
  sprintf(filename,"Coverage_alpha_0.%d_obs_at_%d_sigma.eps",
        (int)(10.*alpha),idisc_threshold);
 } else {
  sprintf(filename,"Coverage_alpha_0.0%d_obs_at_%d_sigma.eps",
        (int)(100.*alpha),idisc_threshold);
 }
 TCanvas * C = new TCanvas ("C","Coverage", 500,500);
 C->cd();
 Cov->SetMinimum(1.-2*alpha);
 Cov->SetLineWidth(3);
 Cov->Draw();
 C->Print(filename);
```

Testing Hypotheses

Hypothesis testing: generalities

We are often concerned with proving or disproving a theory, or comparing and choosing between different hypotheses the most credible one, based on some data.

In general this is a different problem than that of estimating a parameter, but the two are tightly connected.

If nothing is known a priori about a parameter, naturally one uses the data to **estimate** it; if however a theoretical prediction exists on a particular value, the problem is more proficuously formulated as a **test of hypothesis**.

Within the realm of hypothesis testing one must distinguish what are more aptly called **goodness-of-fit tests**: in that case there is only one hypothesis (e.g. a particular value of a parameter as opposed to any other value), so some of the possible techniques are not applicable

A hypothesis is **simple** if it is completely specified; otherwise (e.g. if depending on the unknown value of a parameter) it is called **composite**.

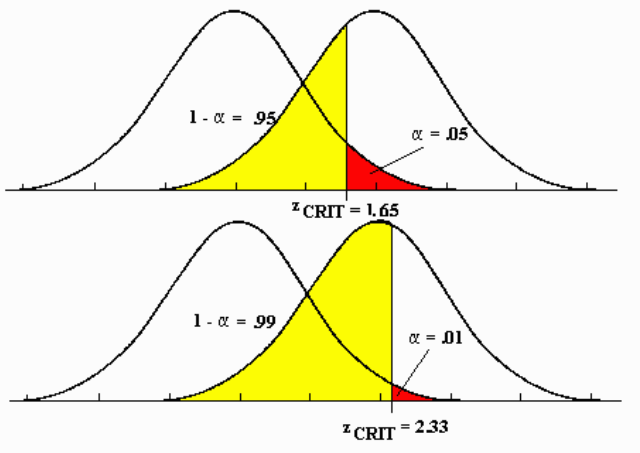
Hypothesis Testing: Ingredients

- H_0: null hypothesis
- H_1: alternate hypothesis
- Three main parameters in the game:
  - α: **type-I error rate**; probability that H_0 is true although you accept the alternative hypothesis
  - β: **type-II error rate**; probability that you fail to claim a discovery (accept H_0) when in fact H_1 is true
  - θ, parameter of interest (describes a continuous hypothesis, for which H_0 is a particular value). E.g. θ=0 might be a zero cross section for a new particle
- Common for H_0 to be nested in H_1

Can compare different methods by plotting α vs β vs the parameter of interest

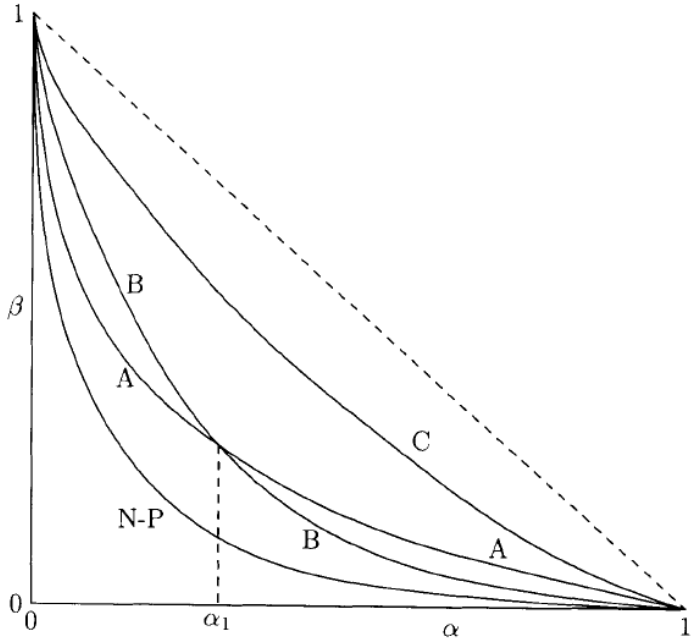
- Usually there is a tradeoff between α and β; often a subjective decision, involving cost of the two different errors.
- Tests may be more powerful in specific regions of an interval (e.g. a Higgs mass)

NB: There is a 1-to-1 correspondence between hypothesis tests and interval construction. In fact a test of σ=0 for a new particle signal equates to asking whether σ=0 is in the confidence interval.



Above, a smaller α is paid for by a larger type-II error rate (yellow area)
→ smaller power 1-β

Alpha vs Beta and power graphs



- Choice of α and β is conflicting: where to stay in the curve provided by your analysis method highly depends on habits in your field
- What makes a difference is the test statistic. N-P likelihood-ratio test, when available, outperforms others (NP lemma, see next slide)
- As data size increases, power curve becomes closer to step function

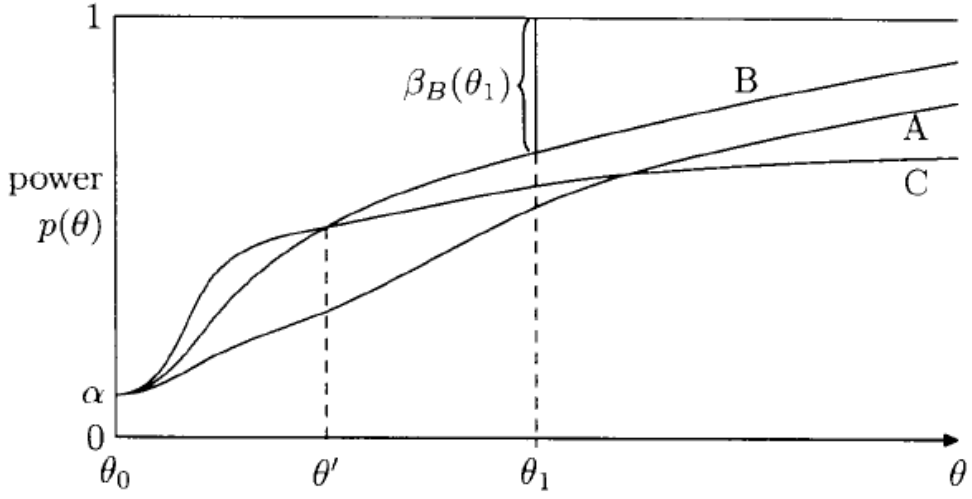


Fig. 10.3. Power functions of tests A, B, and C at significance level α. Of these three tests, B is the best for $\theta > \theta'$. For smaller values of θ, C is better.

The power of a test usually also depends on the parameter of interest: different methods may have better performance in different parameter space points UMP (uniformly most powerful): has the highest power for any θ

The Neyman-Pearson Lemma

- For simple hypothesis testing there is a recipe to find the most powerful test. It is based on the likelihood ratio.
- Take data X={X_1…X_N} and two hypotheses depending on the values of a discrete parameter: H_0={θ=$θ_0$} vs H_1{θ=$θ_1$}. If we write the expressions of size α and power 1-β we have

$$\int_{w_\alpha} f_N(X \mid \theta_0) dX = \alpha$$

$$1-\beta = \int_{w_\alpha} f_N(X \mid \theta_1) dX$$

The problem is then to find the critical region w_α such that 1-β is maximized, given α. We rewrite the expression for power as

$$1-\beta = \int_{w_\alpha} \frac{f_N(X \mid \theta_1)}{f_N(X \mid \theta_0)} f_N(X \mid \theta_0) dX$$

which is an expectation value:

$$= E_{w_\alpha}\left[\frac{f_N(X \mid \theta_1)}{f_N(X \mid \theta_0)} \mid \theta = \theta_0 \right]$$

This is maximized if we accept in w_α all the values for which

$$l_N(X, \theta_0, \theta_1) = \frac{f_N(X \mid \theta_1)}{f_N(X \mid \theta_0)} \geq c_\alpha$$

So one chooses H_0 if $l_N(X, \theta_0, \theta_1) > c_\alpha$

and H_1 if instead $l_N(X, \theta_0, \theta_1) \leq c_\alpha$

In order for this to work, the likelihood ratio must be defined in all space; hypotheses must be **simple**. The test above is called Neyman-Pearson test, and a test with such properties is the **most powerful**.

Treatment of Systematic Uncertainties

- Statisticians call these *nuisance parameters*
- Any measurement in HEP is affected by them: the turning of an observation into a measurement requires assumptions about parameters and other quantities whose exact value is not perfectly known → their uncertainty affects the main measurement
  - Going from a event count to a cross section requires knowing N_b, L, ε_{sel}, ε_{trig} …
  - measurements which are subsidiary to the main result

- Inclusion of effect of nuisances in interval estimation and hypothesis testing introduces complications. Each of the methods has recipes, but not universal nor always applicable
  - **Bayesian treatment**: one constructs the multi-dimensional prior pdf $p(\theta)\Pi_i p(\lambda_i)$ including all the parameters λ_i, multiplies by $p(X_0|\theta,\lambda)$, and integrates all of the nuisances out, remaining with $p(\theta|X_0)$
  - **Classical frequentist treatment**: scan the space of nuisance parameters; for each point do Neyman construction, obtaining multi-dimensional confidence region; project on parameter of interest
  - **Likelihood ratio**: for each value of the parameter of interest θ^*, one finds the value of nuisances that globally maximizes the likelihood, and the corresponding $L(\theta^*)$. The set of such likelihoods is called the **profile likelihood.**

- Each "method" has problems (B: multi-D priors; C: overcoverage and intractability; L: undercoverage) – will not discuss them here, but note that this is a topic at the forefront of research, for which no general recipe is valid.
- Often used are "hybrid" methods for integrating nuisance parameters out: for instance, treat nuisance parameters in a Bayesian way while treating the parameter of interest in a frequentist way, or "profile away" the nuisance parameters and then use any method. Also possible is using Bayesian techniques and then evaluate their coverage properties.

Notes on Goodness-of-fit tests

- If H_0 is specified but the alternative H_1 is not, then only the Type I error rate α can be calculated, since the Type II error rate β depends on having specified a particular H_1.

  In this case the test is called a test for *goodness-of-fit (to H_0).*

- Hence the question "Which g.o.f. test is best?" is ill-posed, since the power depends on the alternative hypothesis, which is not given.

- In spite of the popularity of tests which give a statistic from which a p-value can more readily be computed (in particular χ^2 and Kolomogorov tests), their ability to discriminate against variations with respect to H_0 may be poor, i.e. they may have small power $(1-\beta)$ against relevant alternative hypotheses
  - χ^2 throws away information (sign, ordering)
  - Kolmogorov –Smirnov test only sensitive to biases, not to shape variations, and has terrible performance on tails

- It is in general hard to define what is random and what is not. *Imagine you get three p-values of the null hypothesis: would you like to see them evenly spaced in [0,1] ? Would it induce you to doubt of the null if they all came out within 0.01 of 0.5 ? What if they are all close to 0.624 ? Or all close to zero ?*

More on GoF

- Note the duality with confidence intervals: one might test the hypothesis $\theta=\theta_{test}$ using θ^* as test statistic. If we define the region $\theta^*>=\theta^*_{obs}$ as having equal or less agreement with the hypothesis than the result obtained, then the p-value of the test is α.
  - but for the c.i. the probability α is specified first, and the value θ_{test} is the random variable (depends on data); in a G.o.F. test for θ_{test}, we specify θ_{test} and the p-value is the result.

- In HEP, despite their limitations, Goodness-of-Fit tests are useful for a number of applications:
  - consistency checks
  - defining a control region
  - model testing
- The job of the experimenter is to find a suitable test statistic, *and* a region of interest of the latter. An example will clarify matters.

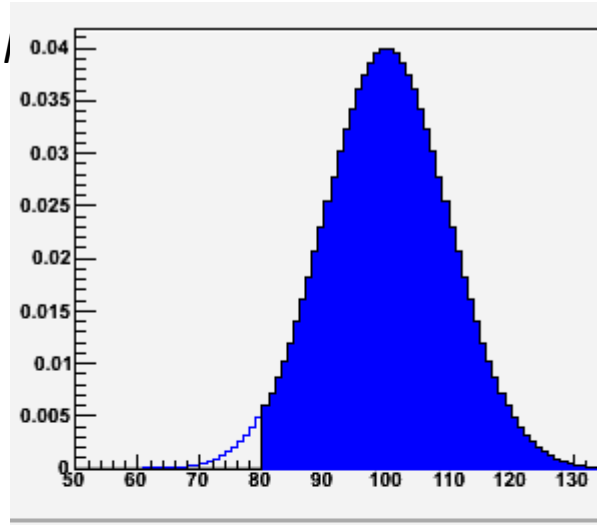
Choosing the region of interest

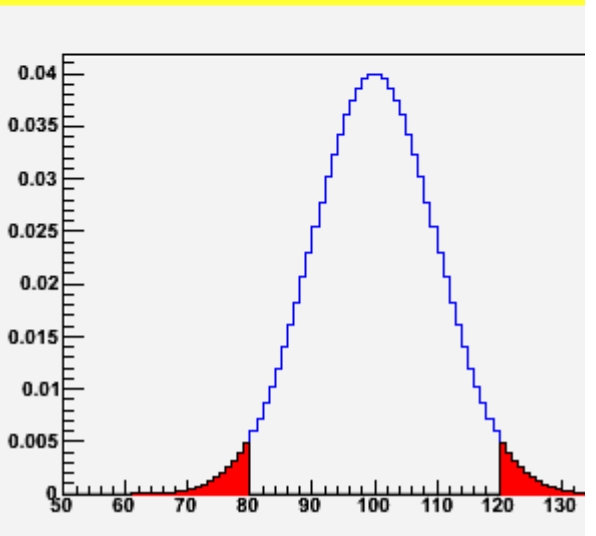
- Feynman's example:

  "*Upon walking here this morning, the strangest thing ever happened to me. A car passed by, and I could read the plate:* ***JKZ 0533****. How weird is that ??! The probability that I saw such a combination of letters and numbers (assuming they are all used in this country) is one in 10000*26^3, or one in eighty-eight millions!*"

  Correct… The paradox arises from not having defined beforehand the region of interest!

- A more common one: you have a counting experiment where background is predicted to be 100 events. You observe 80 events. How rare is that ?





  - **Ill-posed question** ! Depends, to say the least, on whether you are interested only in excesses or in absolute departures!
  - In the first case the region of interest is N>=x, which, for x=80, corresponds to a fractional area p = 0.977.
  - In the second case, the region of interest is |N-100|>=|x-100| which for x=80 has an integral p = 0.0455.
  - And one might imagine other ways to answer – a no-brainer being $p=e^{-100}\ 100^{80}/80!$

Combination of p-values

- Suppose you have several p-values, derived from different, independent tests. You may ask yourself several questions with them.
  - What is the probability that the smallest of them is as small as the one I got ?
  - What is the probability that the largest one is as small as the largest I observed ?
  - What is the probability that the product is as small as the one I can compute with these N values ?
- Please note! Your inference on the data at hand **strongly** depends on what test you perform, for a given set of data. In other words, you cannot choose which test to run only upon seeing the data…

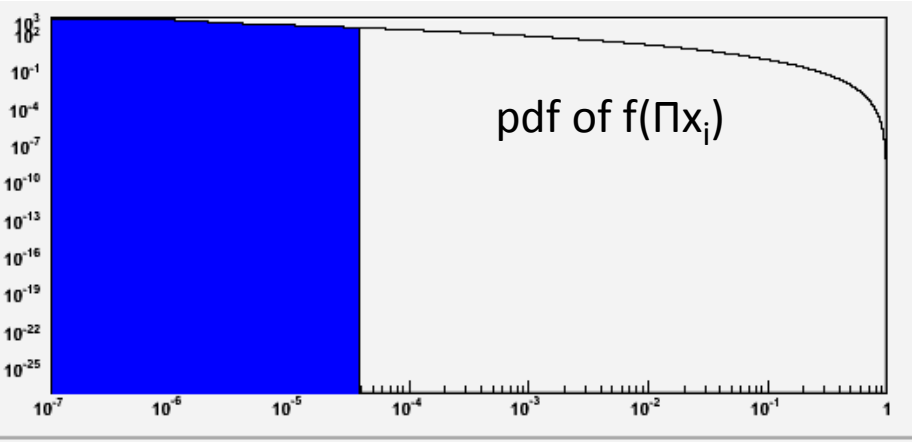
- Suppose anyway you believe that each p-value tells something about the null hypothesis you are testing, so you do not want to discard any of them. Then one reasonable (not the optimal!) thing to do is to use the product of the N values. The formula providing the cumulative distribution of the density of $x=\Pi x_i$ can be derived by induction (see [Roe 1992], p.129) and is

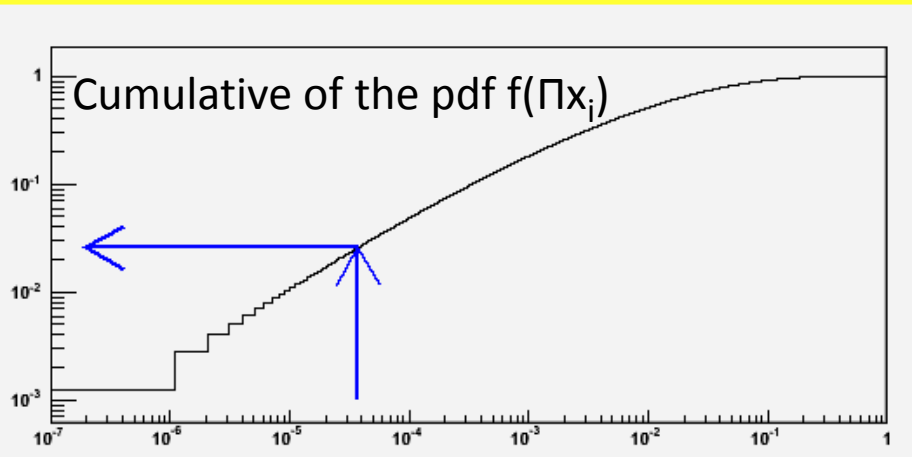
$$F_N(x) = x\sum_{j=0}^{N-1} \frac{1}{j!} |\log^j(x)|$$

This accounts for the speed with which the product of N numbers in [0,1] tends to zero as N grows.

Note, this is just one of MANY ways to construct a single statistic from several p-values.

Some examples





To start let us take five *really uniformly* distributed p-values, x_1=0.1, x_2=0.3, x_3=0.5, x_4=0.7, x_5=0.9. Their product is 0.00945, and with the formula just seen we get P(0.00945)=0.5017. As expected.

And what if instead x_1=0.00001, x_2=0.3, x_3=0.5, x_4=0.7, x_5=0.9 ? The result is P($9.45*10^{-7}$) =0.00123, which is rather large: one might think that the chance of getting one in five numbers as small as 10^{-5} must occur only a few times in 10^{-5}. But we are testing the product, not the smallest of the five numbers !

And if now we let x_1=0.05, x_2=0.10, x_3=0.15, x_4=0.20, x_5=0.25, the test for the product yields P($3.75*10^{-5}$)=0.0258 (see picture on the right).

Also not a compelling rejection of the null…

Compare with what you would get if you had asked "*what is the chance that five numbers are all smaller than 0.25 ?*", whose answer is $(0.25)^5$=0.00098. This demonstrates that the a-posteriori choice of the test is to be avoided !

GoF tests with Max Likelihood

- The maximum likelihood is a powerful method to estimate parameters, but **no measure of GoF is given**, because the expected value of L at maximum is not known, even under the hypothesis that the data are indeed sampled from the pdf model used in the fit
- The distribution of L_{max} can be studied with toy MC → one derives a p-value that a value as small as the one observed in the data arises, under the given assumptions
- Alternatively, one can bin the data, obtaining estimated mean values of entries per bin from the ML fit:

$$\hat{\nu}_i = n_{tot} \int_{x_i^{min}}^{x_i^{max}} f(x;\hat{\theta})dx$$

Then one can derive a χ^2_L statistic using the ratio of likelihoods $\lambda = \frac{L(n|\nu)}{L(n|n)}$

and computing $\chi^2 = -2\log\lambda$

since in this case the latter follows a χ^2 distribution.

The quantity $\lambda(\nu)=L(n|\nu)/L(n|n)$ differs from the likelihood function by a normalization factor, and can thus be used for both parameter estimation and Goodness of fit.

Conclusions

- **Statistics is NOT trivial**. Not even in the simplest applications!
- A understanding of the different methods to derive results (eg. for upper limits) is crucial to make sense of the often conflicting results one obtains even in simple problems
  - The key in HEP is to try and derive results with different methods –if they do not agree, we get wary of the results, plus we learn something

- Making the right choices for what method to use is an expert-only decision, so…

  You should become an **expert in Statistics**, if you want to be a good particle physicist (or even if you want to make money in the financial market)
- The slide of this course are nothing but an appetizer. To really learn the techniques, you must put them to work

- Be careful about what statements you make based on your data! You should now know how to avoid:
  - Probability inversion statements: "The probability that the SM is correct given that I see such a departure is less than x%"
  - Wrong inference on true parameter values: "The top mass has a probability of 68.3% of being in the 171-174 GeV range"
  - Apologetic sentences in your papers: "Since we observe no significant departure from the background, we proceed to set upper limits"
  - Improper uses of the Likelihood: "the upper limit can be obtained as the 95% quantile of the likelihood function"

References

[James 2006] F. James, *Statistical Methods in Experimental Physics* (IInd ed.), World Scientific (2006)

[Cowan 1998] G. Cowan, *Statistical Data Analysis*, Clarendon Press (1998)

[Cousins 2009] R. Cousins, HCPSS lectures (2009)

[D'Agostini 1999] G. D'Agostini, *Bayesian Reasoning in High-Energy Physics: Principles and Applications*, CERN Yellow Report 99/03 (1999)

[Stuart 1999] A. Stuart, K. Ord, S. Arnold, *Kendall's Advanced Theory of Statistics*, Vol. 2A, 6th edition (1999)

[Cox 2006] D. Cox, *Principles of Statistical Inference*, Cambridge UP (2006)

[Roe 1992] B. P. Roe, Probability and Statistics in Experimental Physics, Springer-Verlag (1992)

[Tucker 2009] R. Cousins and J. Tucker, 0905.3831 (2009)

[Cousins 2011] R. Cousins, Arxiv:1109.2023 (2011)

[Cousins 1995] R. Cousins, "Why Isn't Every Physicist a Bayesian ?", Am. J. Phys. 63, n.5, pp. 398-410 (1995)

[Gross 2010] E. Gross, "Look Elsewhere Effect", Banff (2010) (see p.19)

[Vitells 2010] E. Gross and O. Vitells, "Trials factors for the look elsewhere effects in High-Energy Physics", Eur.Phys.J.C70:525-530 (2010)

[Dorigo 2000] T. Dorigo and M. Schmitt,"On the significance of the dimuon mass bump and the greedy bump bias", CDF-5239 (2000)

[ATLAS 2011] ATLAS and CMS Collaborations, ATLAS-CONF-2011-157 (2011); CMS PAS HIG-11-023 (2011)

[CMS 2011] ATLAS Collaboration, CMS Collaboration, and LHC Higgs Combination Group, "Procedure for the LHC Higgs boson search combination in summer 2011", ATL-PHYS-PUB-2011-818, CMS NOTE-2011/005 (2011).

Also cited (but not on statistics):

[McCusker 1969] C.McCusker, I.Cairns, PRL 23, 658 (1969)

[MINOS 2011] P. Adamson et al., Arxiv:1201.2631 (2011)

Stuff I Should Perhaps Skip

Eye fitting: Sensitivity to bumps

- I will discuss the quantification of a signal's significance later on. For now, let us only deal with our perception of it.
- In our daily job as particle physicists, we develop the skill of seeing bumps *–even where there aren't any*
- It is quite important to realize a couple of things:

  1) a likelihood fit is better than our eye at spotting these things → we should avoid getting enamoured with a bump, because we run the risk of fooling ourselves by biasing our selection, thus making it impossible to correctly estimate the significance of a fluctuation

  2) we need to always account for the look-elsewhere effect before we even caress the idea that what we are seeing is a real effect

     - Note that, on the other hand, a theorist with a model in his or her pocket (e.g. one predicting a specific mass) **might not need to account for a LEE –** we will discuss the issue later on

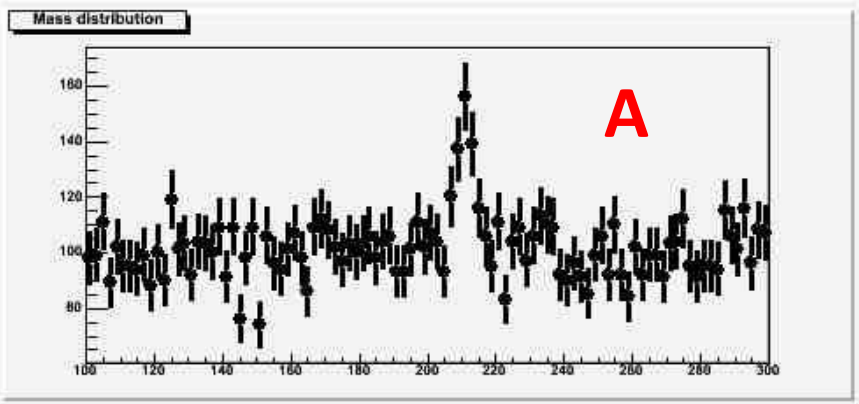
  3) our eye is typically more likely to pick up a tentative signal in some situations rather than others – see point one.

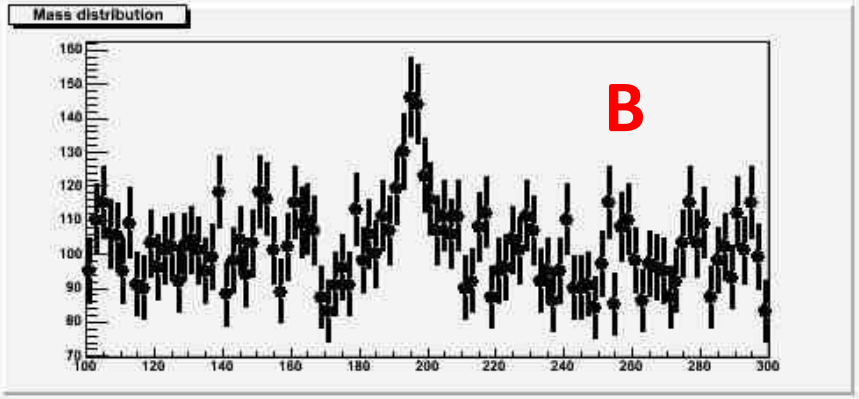
  4) I will try a practical demonstration of the above now.

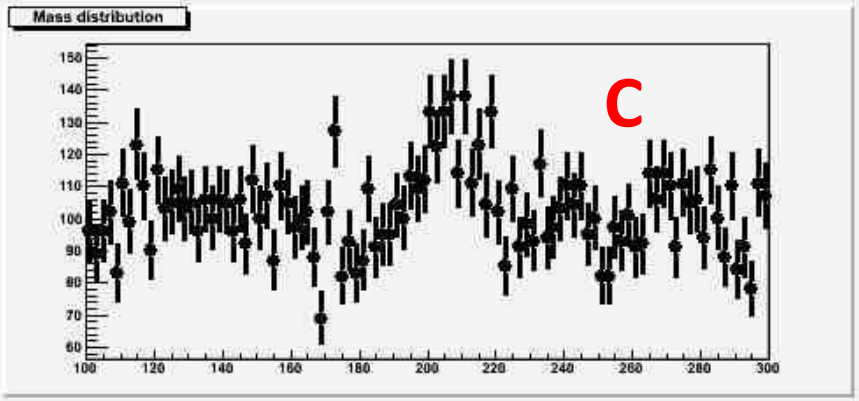
Order by significance:

- Assume the background is flat. Order the three bumps below in descending order of significance (first=most significant, last=least significant)
- Don't try to act smart – I know you can. I want you to examine each histogram and decide which would honestly get you the most excited…

- Let's take stock.







Issues with eye-spotting of bumps

- We tend to want all the data points to agree with our imagined bump hypothesis
  - easier for a few-bin bump than for a many-bin one
  - typical "eye-pleasing" size: a three-bin bump
  - We give more importance to outliers than needed
- We usually forget to account for the multiplicity of places where a bump could build up (correctable part of Look-Elsewhere Effect)

- In examples of previous page, all bumps had the same local significance (5 sigma); however, the most significant one is actually the widest one, if we specified in advance the width of the signal we were looking for! That's because of the smaller number of places it could arise.

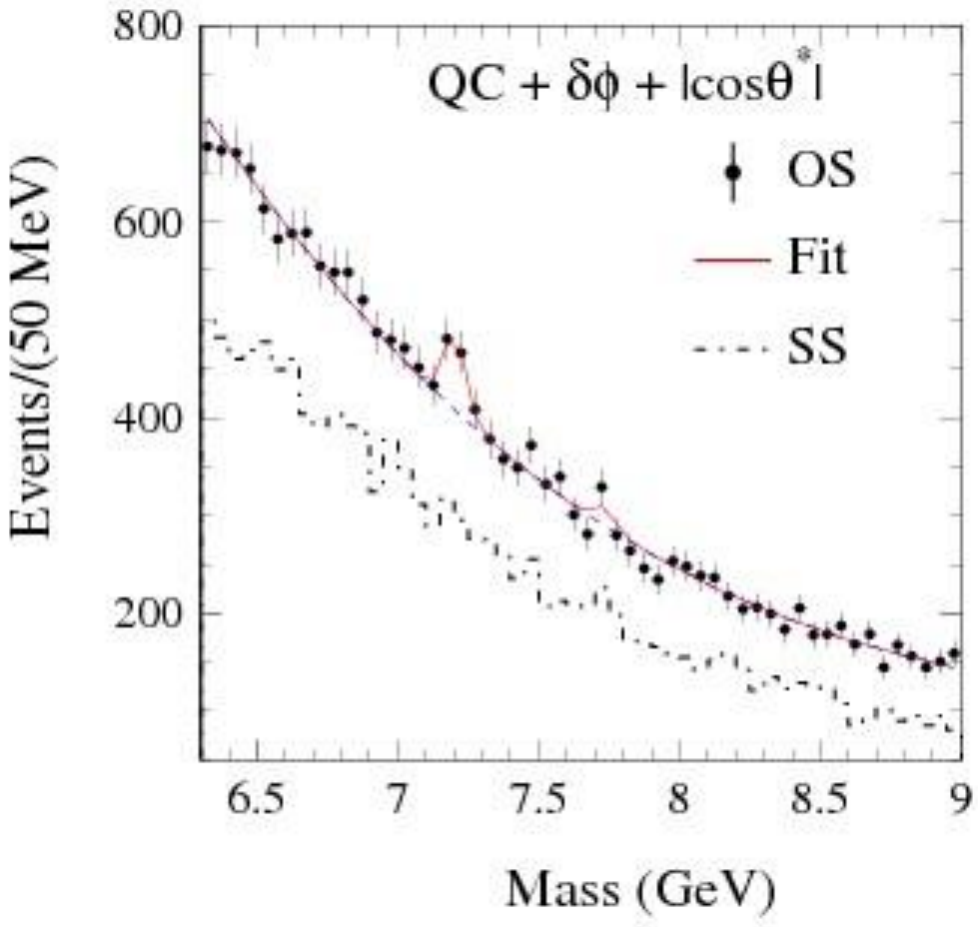
- The nasty part: *we typically forget to account for the multiplicity of histograms and combinations of cuts we have inspected*
  - this is **usually impossible to correct for**!

- The end result: before internal review, 4-sigma effects happen about 1000 times more frequently than they should.

- And some survive review and get published!

One example: the Girominium

- CDF, circa 2000
- Tentative resonance found in proton-antiproton collisions. Fundamental state has mass 7.2 GeV
- Decays to muon pairs; hypothesized bound state of scalar quarks with 1^{--} properties
- Narrow natural width → observable width comparable to resolution
- Significance: 3.5σ
- Issue: statistical fluctuation, wide-context LEE



Phys.Rev.D72:092003,2005

Evaluating significance: one note

- In HEP and astro-HEP a common problem is the evaluation of a significance in a counting experiment. Significance is usually measured in "number of sigmas"

- We have already seen examples of this. It is common to cast the problem in terms of a Goodness-of-Fit test of a null hypothesis H_0

- Expect *b* events from background, test for a signal contributing *s* events by a Poisson experiment: then

$$f(n|b+s) = (b+s)^n e^{-(b+s)}/n!$$

- Upon observing N_{obs}, can assign a probability to the observation as

$$P(n \geq N_{obs}) = 1 - \sum_{n=0}^{N_{obs}-1} \frac{b^n e^{-b}}{n!}$$

- **Of course,** this is not the probability of H_0 being true !! It is the probability that, H_0 being true, we observe N_{obs} events or more

- Take b=1.1, N_{obs}=10: then p=2.6E-7 → a 5σ discovery. Similar for b=0.05, N_{obs}=4.

- **Please note:** if you use a small number of events to measure a cross section, you will have large error bars (whatever your method of evaluating a confidence interval for the true mean!). For instance if b=0, N=5, Likelihood-ratio intervals give 3.08 < s < 7.58, i.e. $s=5_{-1.92}^{+2.58}$. Does that mean we are less than 3-sigma away from zero ? NO !

Bump hunting: Wilks' theorem

- A typical problem: test for the presence of a Gaussian signal on top of a smooth background, using a fit to B(M) (H_0: null hypothesis) and a fit to B(M)+S(M) (H_1: alternative hypothesis)
- This time we have both H_0 and H_1. One can thus easily derive the local significance of a peak from the likelihood values resulting from fits to the two hypotheses. The standard recipe uses **Wilks' theorem**:
  - get L_0, L_1
  - evaluate -2ΔLogL
  - Obtain p-value from probability that $\chi^2(N_{dof})>-2\Delta LogL$
  - Convert into number of sigma for Gaussian distribution using the inverse of the error function
  - Four lines of code !
- Convergence of -2ΔlnL to χ^2 distribution is fast. But certain regularity conditions need to hold! In particular, models need to be nested, and we need to be away from a boundary in the parameter of interest.
  - In principle, allowing the mass of the unknown signal to vary in the fit violates the conditions of Wilks' theorem, since for zero signal normalization H_0 corresponds to any $H_1(M)$ (mass is undefined under H_0: it is a *nuisance parameter present only in the alternative hypothesis*);
  - But it can be proven that approximately Wilks' theorem still applies (see [Gross 2010])
  - Typically one runs toys to check the distribution of p-values
  - but this is not always practical
- Upon obtaining the local significance of a bump, one needs to account for the multiplicity of places where the signal might have arisen by chance.
  - Is rule of thumb valid ? $TF = (M_{max}-M_{min})/\sigma_M$

More on the Look-Elsewhere Effect

- The problem of accounting for the multiplicity of places where a signal could have arisen by chance is apparently easy to solve:
  - Rule of thumb ?
  - Run toys by simulating a mass distribution according to H_0 alone, with $N=N_{obs}$ (remember: **thou shalt condition!**), deriving the distribution of $-2\Delta\ln L$

- Running toys is sometimes impractical (see Higgs combination); it is also illusory to believe one is actually accounting fully for the trials factor
  - In typical analyses one has looked at a number of distributions for departures from H_0
  - Even if the observable is just one (say a M_{jj}) one often is guilty of having checked many possible cut combinations
  - If a signal appears in a spectrum, it is often natural to try and find the corner of phase space where it is most significant; then "a posteriori" one is often led into justifying the choice of selection cuts
  - A HEP experiment runs O(100) analyses on a given dataset and O(1000) distributions are checked for departures. A departure may occur in any one of 20 places in a histogram → trials factor is O(20k)
  - This means that one should expect a 4-sigma bump to naturally arise by chance in any given HEP experiment ! (→ Well borne out by past experience…) Beware of quick conclusions!

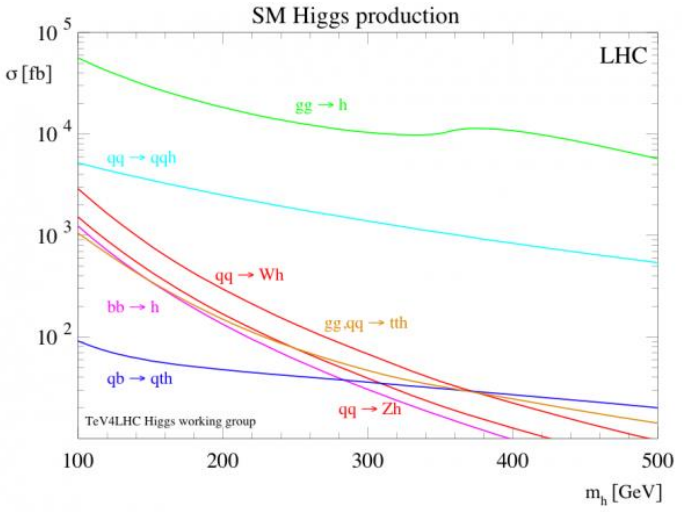
- In reality the trials factor depends also on the significance of the local fluctuation (which can be evaluated by fixing the mass, such that $\Delta N_{dof}=1$). Gross and Vitells [Vitells 2010] demonstrate that a better "rule of thumb" is provided by the formula

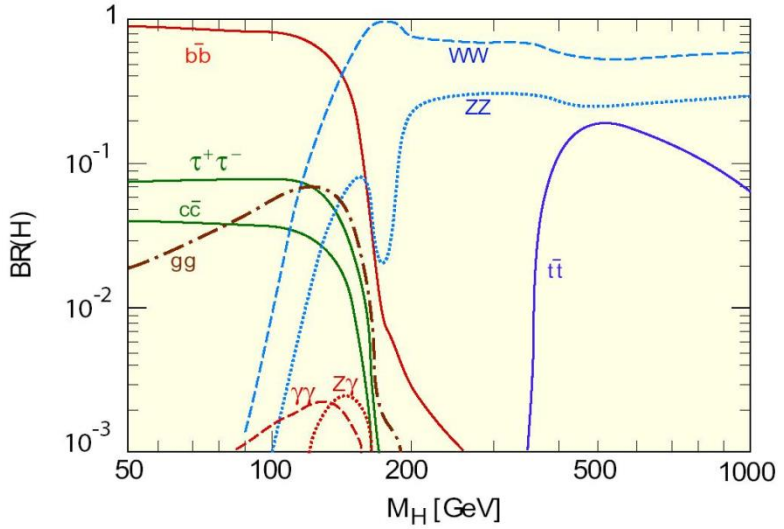
$$TF = k\frac{M_{\max} - M_{\min}}{\sigma_M} Z_{fix}$$

where k is typically 1/3 and can be estimated by counting the average number of local minima $\langle N\rangle = k\,(M_{max}-M_{min})/\sigma_M$

Higgs Searches at LHC

- The Higgs boson has been sought by ATLAS and CMS in all the main production processes and in a number of different final states, resulting from the varied decay modes:
  - qq→Hqq
  - gg→H
  - qq(‘)→VH
  - H→ZZ
  - H→WW
  - H→gg
  - H→tt
  - H→bb





- The importance of the goal brought together some of the best minds of CMS and ATLAS, to define and refine the procedures to combine the above many different search channels, most of which have marginal sensitivity by themselves
- The method used to set upper limits on the Higgs boson cross section is called CL_s and the test statistics is a profile log-likelihood ratio. Dozens of nuisance parameters, with either 0% or 100% correlations, are considered
- Results have been produced as a combined upper limit on the “strength modifier” $\mu=\sigma/\sigma_{SM}$, as well as a “best fit value” for μ, and a combined p-value of the null hypothesis. All of these are produced as a function of the unknown Higgs boson mass.
- The technology is an advanced topic. We can give a peek at the main points, including the construction of the CL_s statistics and the treatment of nuisances, to understand the main architecture

Nuts and Bolts of Higgs Combination

The recipe must be explained in steps. The first one is of course the one of writing down extensively the likelihood function!

1) One writes a global likelihood function, whose parameter of interest is the strength modifier μ. If s and b denote signal and background, and θ is a vector of systematic uncertainties, one can generically write for a single channel:

$$\mathcal{L}(\text{data}\,|\,\mu,\theta) = \text{Poisson}\,(\,\text{data}\,|\,\mu\cdot s(\theta)+b(\theta)\,)\cdot p(\tilde{\theta}|\theta)$$

Note that θ has a "prior" coming from a hypothetical auxiliary measurement.
In the LHC combination of Higgs searches, nuisances are treated in a frequentist way by taking for them the likelihood which would have produced as posterior, given a flat prior, the PDF one believes the nuisance is distributed from. This differs from the Tevatron and LEP Higgs searches.

In L one may combine many different search channels where a counting experiment is performed as the product of their Poisson factors:

$$\prod_i \frac{(\mu s_i + b_i)^{n_i}}{n_i!}\; e^{-\mu s_i - b_i}$$

or from a unbinned likelihood over k events, factors such as:

$$k^{-1}\prod_i (\mu S f_s(x_i) + B f_b(x_i))\;\cdot\; e^{-(\mu S+B)}$$

2) One then constructs a profile likelihood test statistic q_μ as $\tilde{q}_\mu = -2 \ln \frac{\mathcal{L}(\text{data}|\mu, \hat{\theta}_\mu)}{\mathcal{L}(\text{data}|\hat{\mu}, \hat{\theta})}$

Note that the denominator has L computed with the values of $\mu\hat{}$ and $\theta\hat{}$ that globally maximize it, while the numerator has $\theta=\theta\hat{}_\mu$ computed as the conditional maximum likelihood estimate, given μ.

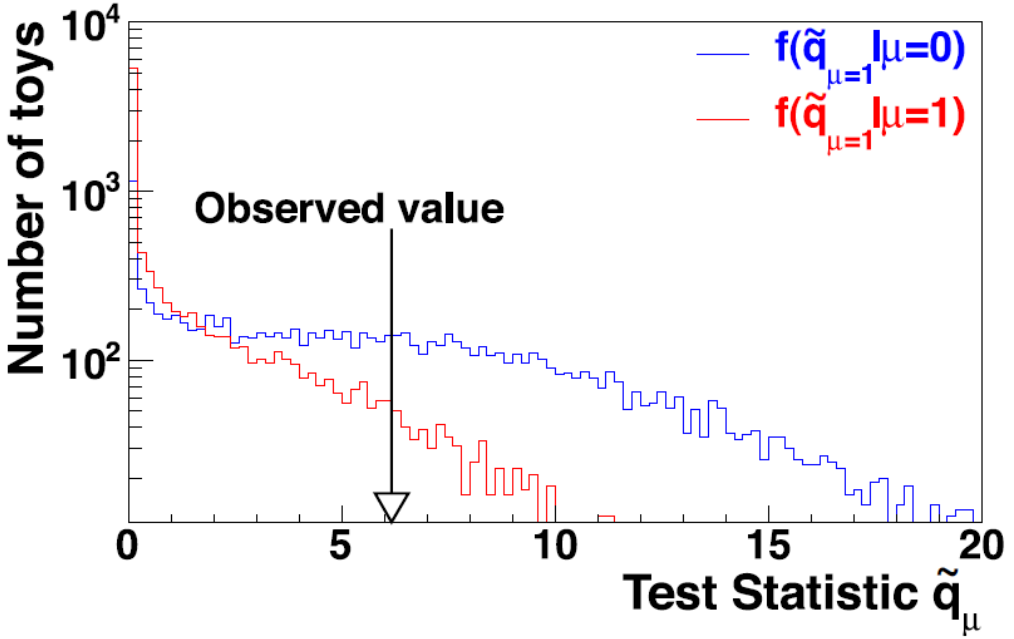
A constraint is posed on the MLE $\mu\hat{}$ to be confined in $0<=\mu\hat{}<=\mu$: this avoids negative solutions for the cross section, and ensures that best-fit values *above* the signal hypothesis μ are not counted as evidence against it.

The above definition of a test statistic for CL_s in Higgs analyses differs from earlier instantiations

- LEP: no profiling of nuisances
- Tevatron: μ=0 in L at denominator

3) ML values $\theta_\mu\hat{}$ for H_1 and $\theta_0\hat{}$ for H_0 are then computed, given the data and μ=0 (bgr-only) and μ>0

4) Pseudo-data is then generated for the two hypotheses, using the above ML estimates of the nuisance parameters. With the data, one constructs the pdf of the test statistic given a signal of strength μ (H_1) and μ=0 (H_0). This way has good coverage properties.



5) With the pseudo-data one can then compute the integrals defining p-values for the two hypotheses. For the signal plus background hypothesis H_1 one has

$$p_\mu = P(\tilde{q}_\mu \geq \tilde{q}_\mu^{obs} \,|\, \text{signal+background}) = \int_{\tilde{q}_\mu^{\text{obs}}}^{\infty} f(\tilde{q}_\mu|\mu, \hat{\theta}_\mu^{\text{obs}})\, d\tilde{q}_\mu$$

and for the null, background-only H_0 one has

$$1 - p_b = P(\tilde{q}_\mu \geq \tilde{q}_\mu^{obs} \,|\, \text{background-only}) = \int_{q_0^{\text{obs}}}^{\infty} f(\tilde{q}_\mu|0, \hat{\theta}_0^{\text{obs}})\, d\tilde{q}_\mu$$

6) Finally one can compute the value called CL_s as

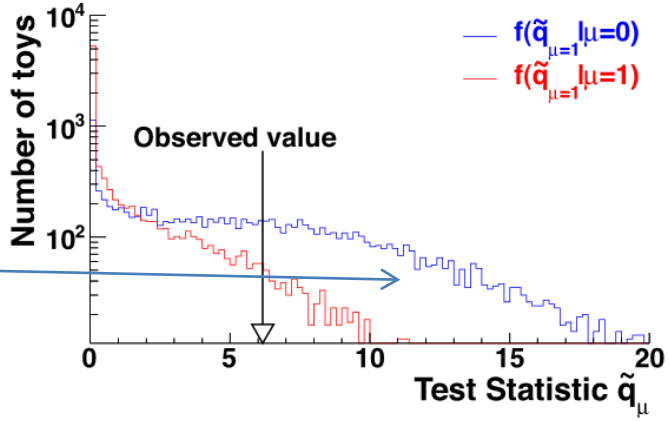
$$CL_s = p_\mu/(1-p_b)$$

CL_s is thus a "modified" p-value, in the sense that it describes how likely it is that the value of test statistic is observed under the alternative hypothesis by also accounting for how likely the null is: the drawing incorrect inferences based on extreme values of p_μ is "damped", and cases when one has no real discriminating power, approaching the limit $f(q|\mu)=f(q|0)$, are prevented from allowing to exclude the alternate hypothesis.

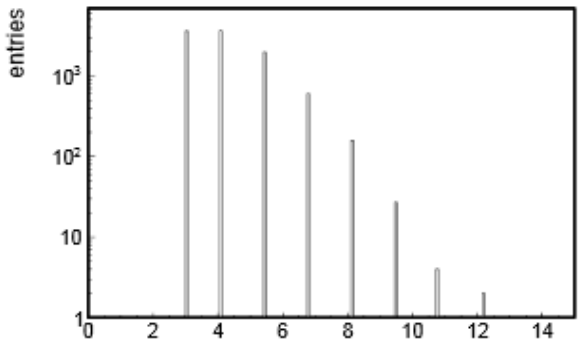
7) We can then **exclude H_1 when $CL_s < \alpha$**, the (defined in advance !) *size* of the test. In the case of Higgs searches, all mass hypotheses $H_1(M)$ for which $CL_s<0.05$ are said to be excluded (one would rather call them "disfavoured"…)

Derivation of expected limits

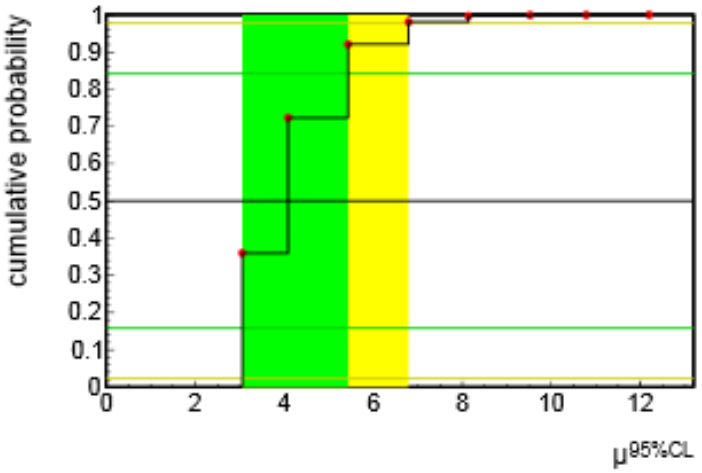
One starts with the **background-only hypothesis μ=0**, and determines a distribution of possible outcomes of the experiment with toys, obtaining the CLs test statistic distribution for each investigated Higgs mass point



From CLs one obtains the PDF of upper limits μ^{UL} on μ for each M_h. [*E.g. on the right we assumed b=1 and s=0 for μ=0, whereas μ=1 would produce <s>=1*]



Then one computes the cumulative PDF of μ^{UL}



Finally, one can derive the median and the intervals for μ which correspond to 2.3%, 15.9%, 50%, 84.1%, 97.7% quantiles. These define the "expected-limit bands" and their center.

Quantifying the significance of a signal in the Higgs search

- To test for the significance of an excess of events, given a Mh hypothesis, one uses the bgr-only hypothesis and constructs a modified version of the q test statistic:

$$q_0 = -2\ln\frac{\mathcal{L}(\text{data}|0,\hat{\theta}_0)}{\mathcal{L}(\text{data}|\hat{\mu},\hat{\theta})} \quad \text{and } \hat{\mu} \geq 0.$$

- This time we are testing any $\mu>0$ versus the H_0 hypothesis. One builds the distribution $f(q_0|0,\theta_0^{\hat{}obs})$ by generating pseudo-data, and derives a p-value corresponding to a given observation as

$$p_0 = P(q_0 \geq q_0^{obs}) = \int_{q_0^{obs}}^{\infty} f(q_0|0,\hat{\theta}_0^{\text{obs}})\, dq_0.$$

- One then converts p into Z using the relation

$$p = \int_Z^{\infty} \frac{1}{\sqrt{2\pi}} \exp(-x^2/2)\, dx = \frac{1}{2} P_{\chi_1^2}(Z^2)$$

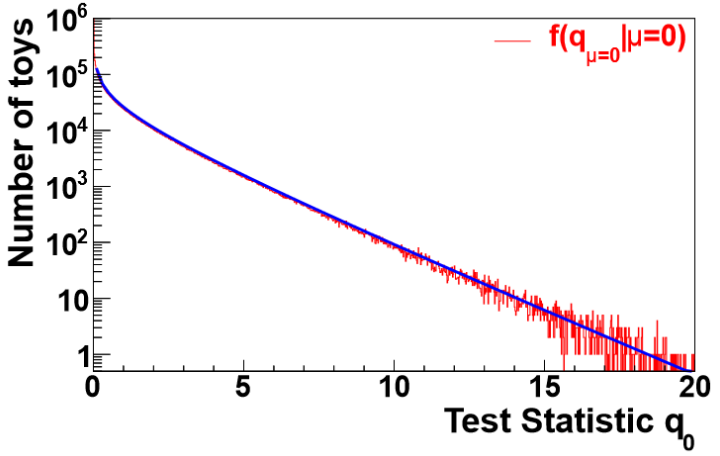
where p_χ^2 is the survival function for the 1-dof chisquared.

- Often it is impractical to generate large datasets given the complexity of the search (dozens of search channels and sub-channels, correlated among each other). One then relies on a very good asymptotic approximation:

$$p^{estimate} = \frac{1}{2}\left[1 - \text{erf}\left(\sqrt{q_0^{\text{obs}}/2}\right)\right]$$

- The derived p-value and the corresponding Z value are "local": they correspond to the specific hypothesis that has been tested (a specific M_h) as q_0 also depends on M_h (the search changes as M_h varies)

- When dealing with many searches, one needs to get a global p-value and significance, i.e. evaluate a trials factor. How to do it in complex situations is explained in the next slide.



Trials factors in the Higgs search

When dealing with complex cases (Higgs combination), a study comes to help.

Wilks' theorem does not apply, and the complication of combining many different search channels makes the option of throwing huge number of toys impractical

Fortunately it has been shown how the trials factor can be counted in. First of all one defines a test statistic encompassing all possible Higgs mass values:

$$q_0(\hat{m}_H) = \max_{m_H} q_0(m_H)$$

This is the maximum of the test statistic defined above for the bgr-only, across the many tests performed at the various possible masses of the Higgs boson.

One can use an asymptotic "regularity" of the distribution of the above q to get a global p-value by using a technique derived by Gross and Vidells [Vitells 2010].

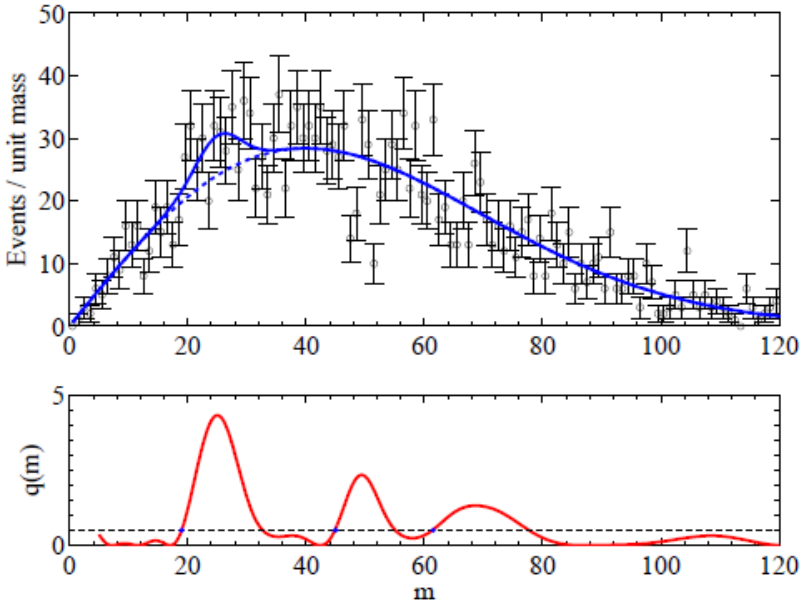
Local minima and upcrossings

One counts the number of "upcrossings" of the distribution of the test statistic, as a function of mass. Its wiggling tells you how many independent places you have been searching in.

The number of local minima in the fit to a distribution is closely connected to the freedom of the fit to pick signal-like fluctuations in the investigated range

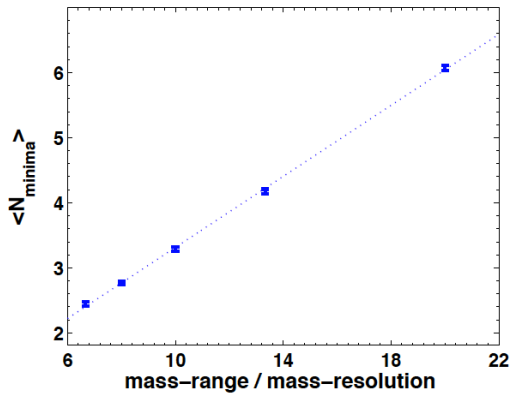
The number of times that the test statistic (below, the likelihood ratio between H_1 and H_0) crosses some reference point is a measure of the trials factor. One estimates the global p-value with the number N_0 of upcrossings from a minimal value of the q_0 test statistics (for which $p=p_0$) by the formula

$$p_b^{global} = P(q_0(\hat{m}_H) > u) \leq \langle N_u \rangle + \frac{1}{2} P_{\chi_1^2}(u)$$



The number of upcrossings can be best estimated using the data themselves at a low value of significance, as it has been shown that the dependence on Z is a simple negative exponential:

$$\langle N_u \rangle = \langle N_{u_o} \rangle e^{-(u-u_o)/2}$$



Example

- Imagine that you scan the Higgs mass and find a maximum q_0 of 9, which according to

$$p^{estimate} = \frac{1}{2}\left[1 - \mathrm{erf}\left(\sqrt{q_0^{\mathrm{obs}}/2}\right)\right]$$

corresponds to a local p-value of 0.13% and a local Z-value of 3σ, the latter computed using

$$p = \int_Z^\infty \frac{1}{\sqrt{2\pi}} \exp(-x^2/2)\, dx \;=\; \frac{1}{2} P_{\chi^2_1}(Z^2)$$

- You then look at the distribution of q_0 as a function of M_h and count the number of upcrossings at a level u_0=1 (where the significance is Z=1 as per above formulas) finding that there are 8 of them. You can then get $\langle N_u \rangle$ for u=9 using

$$\langle N_u \rangle \;=\; \langle N_{u_o} \rangle\, e^{-(u-u_o)/2}$$

which gives <Nu>=0.1465

- The global p-value can be then computed as p_{glob}=0.1465+0.0013 using the formula below. One concludes that the trial factor is about 100 in this case.

$$p_b^{global} \;=\; P(q_0(\hat{m}_H) > u) \;\leq\; \langle N_u \rangle + \frac{1}{2} P_{\chi^2_1}(u)$$