

LECTURE 3: PARAMETER ESTIMATION

STATISTICAL ANALYSIS IN EXPERIMENTAL PARTICLE PHYSICS

Kai-Feng Chen
National Taiwan University

THEORY OF ESTIMATORS

- Estimation may be considered as the measurement of a parameter (*which is assumed to be fixed, but unknown value*) based on a limited number of experimental observations.
- **Point estimation:** determines a single value as close as possible to the true value — for example a measurement of physics parameter, such a mass, cross section, branching fraction.
- **Interval estimation:** determines a range of values most likely to include the true parameter value — for example an estimation of upper/lower limits.
- The main subject here is what is the exact sense in which “*close*” and “*likely to include*”!

STATISTICAL INFERENCE

**THEORY
MODEL**



Probability

Data fluctuate according to process randomness



DATA

**THEORY
MODEL**



Inference

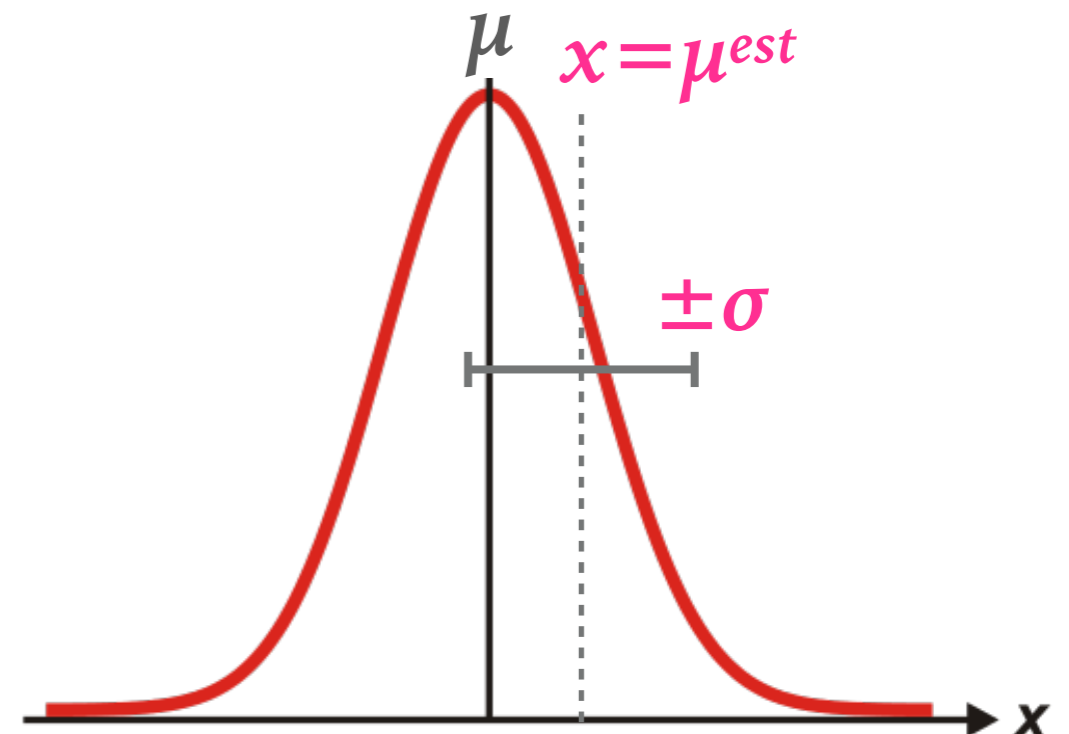
Model uncertainty due to fluctuations of the data sample

BASIC CONCEPTS

- To estimate a parameter, one first chooses a function of the observations = a method for proceeding from the observations to the estimate = **the estimator**.
- The numerical value yield by the estimator for a particular set of observations is the **estimate**.
- Basic properties of estimator of concern:
 - **Consistency** — if an estimator converge toward the true value, as the number of observation increases.
 - **Unbiasedness** — the deviation of the expectation of the estimate from the true value is zero.
 - (*and more: robustness, efficiency, ...*)

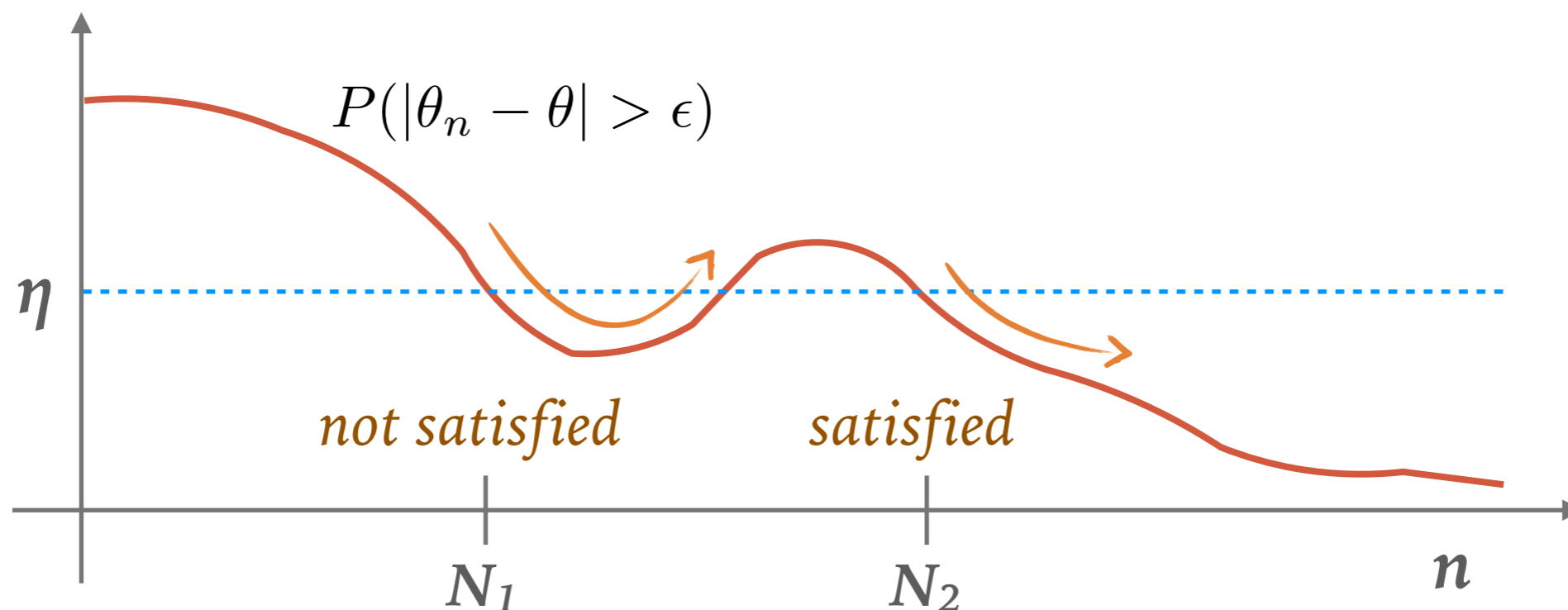
SIMPLEST EXAMPLE

- Remember the estimator is a function of a given sample whose statistical properties are known, and is related to some parameters (denoted as θ) in PDF.
- A minimal example:
 - Assume we have a Gaussian PDF with a **known** σ and an **unknown** μ
 - A single experiment gives a measurement x , thus we estimate μ as $\mu^{est} = x$
 - The distribution of μ^{est} (repeating the experiment many times) should give the original Gaussian.
 - On average 68.27% of the experiments will provide an estimate within the range: $\mu - \sigma < \mu^{est} < \mu + \sigma$
 - And one can determine: $\mu = \mu^{est} \pm \sigma$



CONSISTENCY AND CONVERGENCE

- An estimator is called a consistent estimator if its estimates converge toward the true value of the parameter as the number of observations increases.
- θ_n is an estimator of the random parameter θ based on n observables. For any $\epsilon > 0$, any $\eta > 0$, and large number N exists, the **convergence in probability**:
$$P(|\theta_n - \theta| > \epsilon) < \eta \quad \text{for all } n > N$$
- This means that the θ_n converges in probability toward θ as n increases.



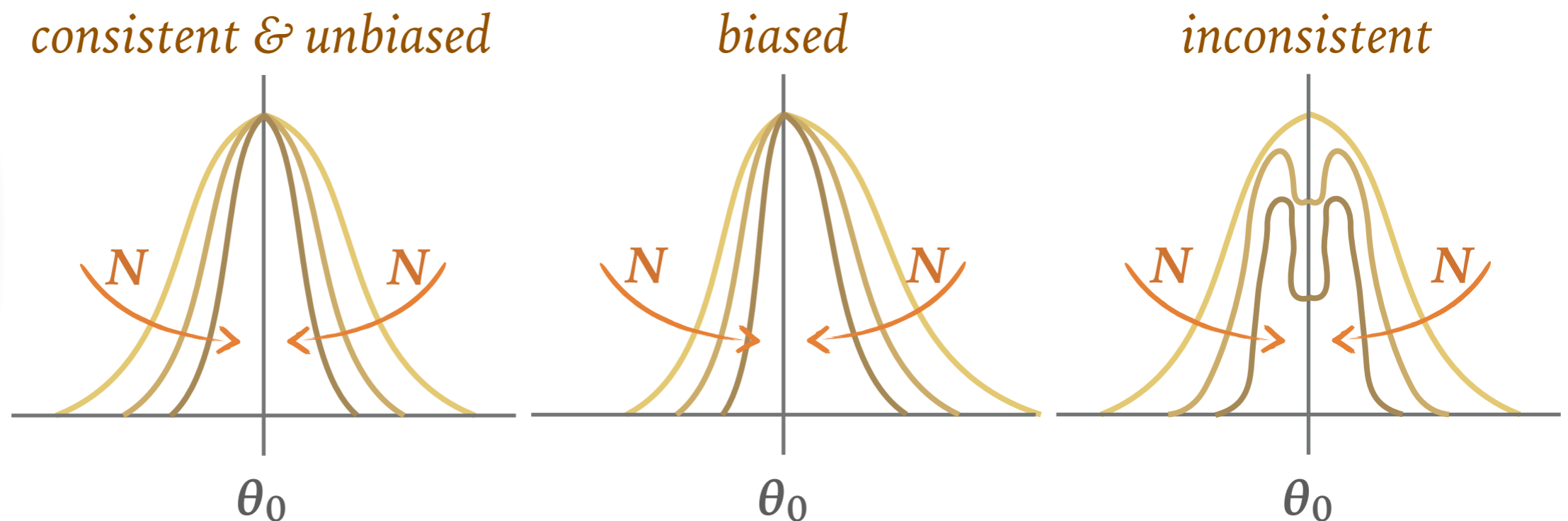
BIAS AND CONSISTENCY

- Then how about unbiasedness? Let's define the bias b of the estimator θ based on N observation as the deviation of the expectation of θ from the true value θ_0 :

$$b_N(\theta) = E(\theta) - \theta_0 = E(\theta - \theta_0)$$

- An estimator is unbiased if b is zero for all N and θ_0 , or $E(\theta) = \theta_0$.

PDF as
functions
of θ



THE LEAST SQUARES METHOD / CHI-SQUARE METHOD

- Consider a set of N independent observations of X_1, X_2, \dots, X_N , from a distribution with expectations of $E(X_i, \theta)$ and the covariance matrix V . By minimizing the covariance form:

$$\begin{aligned} Q^2 &= \sum_{i=1}^N \sum_{j=1}^N [X_i - E(X_i, \theta)] (V^{-1})_{ij} [X_j - E(X_j, \theta)] \\ &= [X - E(X, \theta)]^T V^{-1} [X - E(X, \theta)] \end{aligned}$$

it provides an estimate of the unknown parameters .

- The least squares estimator is consistent, and unbiased.
- The covariance matrix V is not diagonal in general case. However if the observations are independent, the covariance matrix is diagonal. In this case the covariance form can be simplified to just sum of squares:

$$Q^2 = \sum_{i=1}^N \frac{[X_i - E(X_i, \theta)]^2}{\sigma_i^2(\theta)} \quad \text{where } \sigma_i^2(\theta) = V_{ii}$$

EXAMPLE: CORRELATED LEAST SQUARES

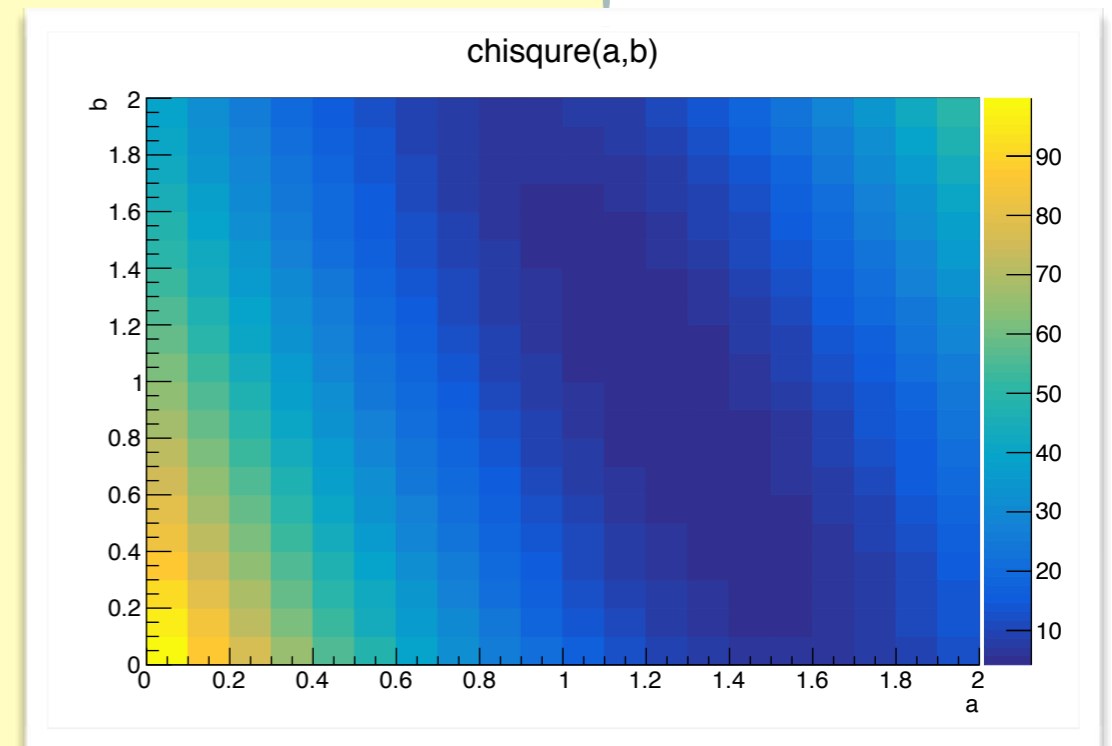
- Here are an easy example for calculating the value of χ^2 with covariance matrix:

example_01.cc

```
const double var_x[5] = {0.000, 1.000, 2.000, 3.000, 4.000};
const double var_y[5] = {1.951, 1.995, 3.088, 4.220, 6.553};
const double cov_y[5][5] = {
    {1.000, 0.020, 0.000, 0.000, 0.000},
    {0.020, 0.485, 0.103, 0.000, 0.000},
    {0.000, 0.103, 0.740, 0.170, 0.000},
    {0.000, 0.000, 0.170, 0.316, 0.214},
    {0.000, 0.000, 0.000, 0.214, 0.520}
};

double model(double x, double a, double b)
{ return a*x + b; }

double chisq(double a, double b)
{
    TVectorD vec(5);
    for (int r=0; r<5; r++) {
        double func = model(var_x[r],a,b);
        vec(r) = func-var_y[r];
    }
    TMatrixD cov(5,5);
    for (int r=0; r<5; r++) {
        for(int c=0;c<5;c++) {
            cov(r,c) = cov_y[r][c];
        }
    }
    return cov.Invert()*vec*vec;
}
```



Just make a 2D scan over the chisq() function!

THE MAXIMUM LIKELIHOOD METHOD

- Consider a set of N independent observations of X : X_1, X_2, \dots, X_N . They can be N events found in an experiment, and the joint PDF of X is

$$P(X|\theta) = P(X_1, X_2, \dots, X_N|\theta) = \prod_{i=1}^N f(X_i|\theta)$$

where $f(X, \theta)$ is the PDF of any observation X .

- When the variable X is replaced the observed data X^0 , then P is no longer a PDF. It becomes the **likelihood function L** , as a function of θ :

$$L(\theta) = P(X|\theta) \Big|_{X=X^0}$$

- The maximum likelihood estimate of the parameter θ is that value for which **L has its maximum given the particular observation X^0** .
- If the number of observations N is also a random variable, the **extended likelihood function** is can be introduced:

$$L(\theta) = p(N|\theta) \prod_{i=1}^N f(X_i|\theta)$$

In the most common case p is a Poisson distribution

THE MAXIMUM LIKELIHOOD METHOD (CONT.)

- In many cases it is convenient to take the logarithm, hence the production of probability can be converted to a summation:

$$L(\theta) \Rightarrow \ln L(\theta) \quad \prod_{i=1}^N f(X_i|\theta) \Rightarrow \sum_{i=1}^N \ln f(X_i|\theta)$$

- The “best fit” parameters can be obtained by maximizing the (log) likelihood function, or solving the *likelihood equation* as below:

$$\frac{\partial}{\partial \theta} \sum_{i=1}^N \ln f(X_i, \theta) = \frac{\partial}{\partial \theta} \ln L(X|\theta) = 0$$

- ML has “very good” statistical properties: **it’s consistency, efficient, and robust.**
- ML estimators may have some bias, but they should decrease as N increases, if the selected PDF model is the correct one!
- Nevertheless the **ML is the widest used parameter estimator.**

INTERFACE WITH MINUIT

- Both the least squares estimator and maximum likelihood estimator requires minimizing or maximizing a certain function. This operation can be performed analytically, for the simplest cases, and numerically for most of the cases.
- **Minuit** is historically (*and still the case nowadays*) the most used minimization engine in the high energy physics.
- For the least squares estimator, this can be carried out by simply supplying the corresponding Q^2 (χ^2) function.
- For the ML estimator, it is common to supply $-2\ln L$ instead. The negative sign is required since Minuit always does minimizing, and the factor of two will matches the supplied function as just sum of squares, if the PDFs are all Gaussians:

$$-2 \ln L = \sum_{i=1}^N \frac{(X_i - \mu)^2}{\sigma^2} + \text{Const.}$$

remark: one can either gives minuit the scaled $-2\ln L$, or ask minuit to do it for you!

LOWER BOUNDS FOR THE VARIANCE

➤ Let X be observations from a distribution, and the estimator θ . Hence the likelihood function is denoted as $L = L(X|\theta)$. We obtained the expression for the bias, $b = E(\theta) - \theta_0$, where θ_0 is the true value.

➤ The variance of any consistent estimator is subject a lower bound, ie. the **Cramér-Rao bound**,

$$V(\theta) \geq \frac{\left[1 + \frac{\partial b(\theta)}{\partial \theta}\right]^2}{E\left[\left(\frac{\partial \ln L}{\partial \theta}\right)^2\right]}$$

➤ “Efficiency of estimator” is the ratio of this Cramér-Rao bound over the variance.

➤ The **efficiency of ML estimator is asymptotically 1** (when the size of observations approaches infinite: $N \rightarrow \infty$).

- No other asymptotically unbiased estimator has asymptotic mean-squared error smaller than the ML estimator.

ROBUSTNESS

- What kind of parameters can be estimated without an assumptions about of the PDF form?
- How reliable are the parameter estimates if the form assumed for the PDF is not quite correct? If the sample distribution has (slight?) deviations from the model, some estimators may deviate more or less than others from the true value.
 - for example if the data includes some misinterpreted observations (unexpected tails, etc).
- **Robustness** is taken to imply **insensitivity to small deviations from the underlying distribution** assumed.
- Although robustness is practically important, but a full treatment is beyond the scope of this lecture (as well as the reference books!)

ROBUSTNESS (CONT.)

- As an example, how could one estimate (*in a robust manner*) the centre of a distribution? Some examples of location estimators are:
 - The **mean** is the expectation of the variable X ;
 - The **median** is the value X for which the cumulative distribution reached 0.5;
 - The **mode** is that value of X for which the PDF has a maximum;
 - The **midrange** is defined when the values of X are limited to $[X_{\min}, X_{\max}]$, and the midrange is $(X_{\min} + X_{\max})/2$.
- Only for Gaussians, the **mean** is the optimal estimator (*unbiased and minimal variance*). If the underlying distribution is not Gaussian, the **mean** is not be best estimator anymore. Sometimes the **median** actually works better.
- **Trimming** — to avoid the effects of “tails”, one can remove the $n/2$ highest and $n/2$ lowest values, and compute the centre of the remaining $N-n$ observations.

COMMENTS: MAXIMUM LIKELIHOOD PROPERTIES

- In general it is important to distinguish between the asymptotic properties which hold for sufficiently large number of observations ($N \rightarrow \infty$), and the finite sample properties.
- As mentioned already, the ML estimator is consistent, *asymptotically* close to the true value, *asymptotically* unbiased. The estimator is also *asymptotically* Normally distributed with minimal variance, *asymptotically* efficient.
- In many cases, the asymptotic limit where these optimal properties hold, but will be approached slowly. For the case of finite N , ML estimator may not always have those optimal properties, except the parent distribution of the observations is of the exponential form.

COMMENTS: GAUSSIAN APPROXIMATION

- If we have a set of N independent measurements, whose PDFs are identical and are Gaussian, we have the model

$$f(X; \mu, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} \exp \left[-\frac{(X - \mu)^2}{2\sigma^2} \right]$$

- The likelihood function is

$$-2 \ln L = \sum_{i=1}^N \frac{(X_i - \mu)^2}{\sigma^2} + N(\ln 2\pi + 2 \ln \sigma)$$

- The maximum likelihood estimate can be performed by an analytical minimization on μ (assuming σ is known):

$$\mu^{\text{est}} = \frac{1}{N} \sum_{i=1}^N X_i \quad (\text{Basically the sampling mean})$$

- If σ^2 is also unknown, the ML estimate of σ^2 is:

$$(\sigma^{\text{est}})^2 = \frac{1}{N} \sum_{i=1}^N (X_i - \mu^{\text{est}})^2 \quad (\text{mean-squares})$$

This estimate can be demonstrated to have a bias of σ^2/N .

ERROR ESTIMATION

- There are two approaches to determine parameter uncertainties.
- **Local error – the 2nd order partial derivatives with respect to the fit parameters around the minimum:**

$$C_{ij}^{-1} = -\frac{\partial^2 \ln L}{\partial \theta_i \partial \theta_j}$$

- Under Gaussian approximation it equals to the covariance matrix;
- May lead to underestimated errors with finite samples.
- **Evaluation of $-2\ln L$ values around the maximum point of likelihood function.**
 - Leads to usual error matrix in a Gaussian model
 - May lead to asymmetric errors.

MIGRAD/HESSE
command under
minuit

MINOS
command under
minuit

ERROR ON MEAN?

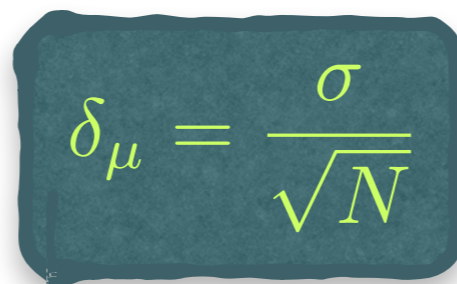
- Let's practice the calculation with second derivatives!
- Remember the likelihood function with the assumption of Gaussian models:

$$-2 \ln L = \sum_{i=1}^N \frac{(X_i - \mu)^2}{\sigma^2} + N(\ln 2\pi + 2 \ln \sigma)$$

- The error on the mean μ can be estimated by

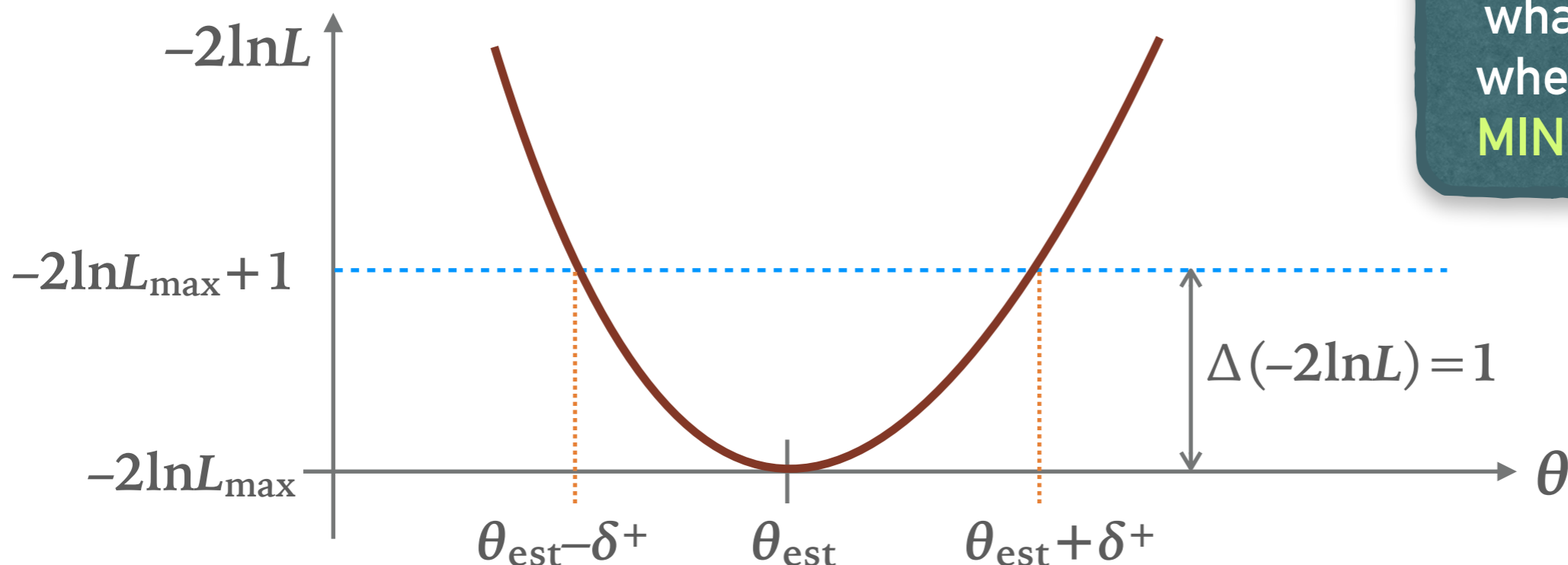
$$C_{\mu}^{-1} = \frac{1}{\delta_{\mu}^2} = -\frac{\partial^2 \ln L}{\partial \mu^2} = \frac{N}{\sigma^2}$$

- And it just gives us the usual estimation of “error on mean”:


$$\delta_{\mu} = \frac{\sigma}{\sqrt{N}}$$

ASYMMETRIC ERROR

- If the $-2\ln L$ function is close to a parabolic shape, the derivatives can be approximated by parameter excursion ranges.
- A “ $n\text{-}\sigma$ ” error can be determined by the range around the Likelihood maximum for which the $-2\ln L$ value increases by n^2 :
 - The errors can be asymmetric for the positive and negative side!
 - It is identical to the σ of Gaussian PDF.



Basically this is what minuit does when you call the **MINOS** command.

EXAMPLE: SCAN OVER LIKELIHOOD FUNCTION

-
- Here are an example how to do a manual likelihood scan over a selected parameter (*take the ending example from the previous lecture*):

partial example_02.cc

```
using namespace RooFit;

TFile *fin = new TFile("example_data.root");
TNtupleD* nt = (TNtupleD *)fin->Get("nt");

RooRealVar mass("mass", "mass", 0., 2.);
RooDataSet data("data", "data", nt, RooArgSet(mass));

RooRealVar mu("mu", "mu", 1.0, 0.5, 1.5);
RooRealVar sigma("sigma", "sigma", 0.05, 0.001, 0.15);
RooGaussian gaus("gaus", "gaus", mass, mu, sigma);
RooRealVar slope("slope", "slope", -0.3, -10., 10.);
RooPolynomial linear("linear", "linear", mass, RooArgSet(slope));

RooRealVar frac("frac", "frac", 0.2, 0., 1.);
RooAddPdf model("model", "model", RooArgList(gaus, linear), RooArgList(frac));

RooFitResult *res = model.fitTo(data, Save(true), Minos(true));

RooAbsReal* nll = model.createNLL(data); ↪ create -log(L)

TH1D *nll_scan = new TH1D("nll_scan", "Likelihood scan", 200, 0.185, 0.22);
for(int bin=1; bin<=nll_scan->GetNbinsX(); bin++) {
    frac.setVal(nll_scan->GetBinCenter(bin));
    nll_scan->SetBinContent(bin, (nll->getVal()-res->minNll())*2.);
}
}
```

Unbinned ML fit
example as before

Fill the histogram w/ $-2*\log(L/L_{\max})$

EXAMPLE: SCAN OVER LIKELIHOOD FUNCTION (CONT.)

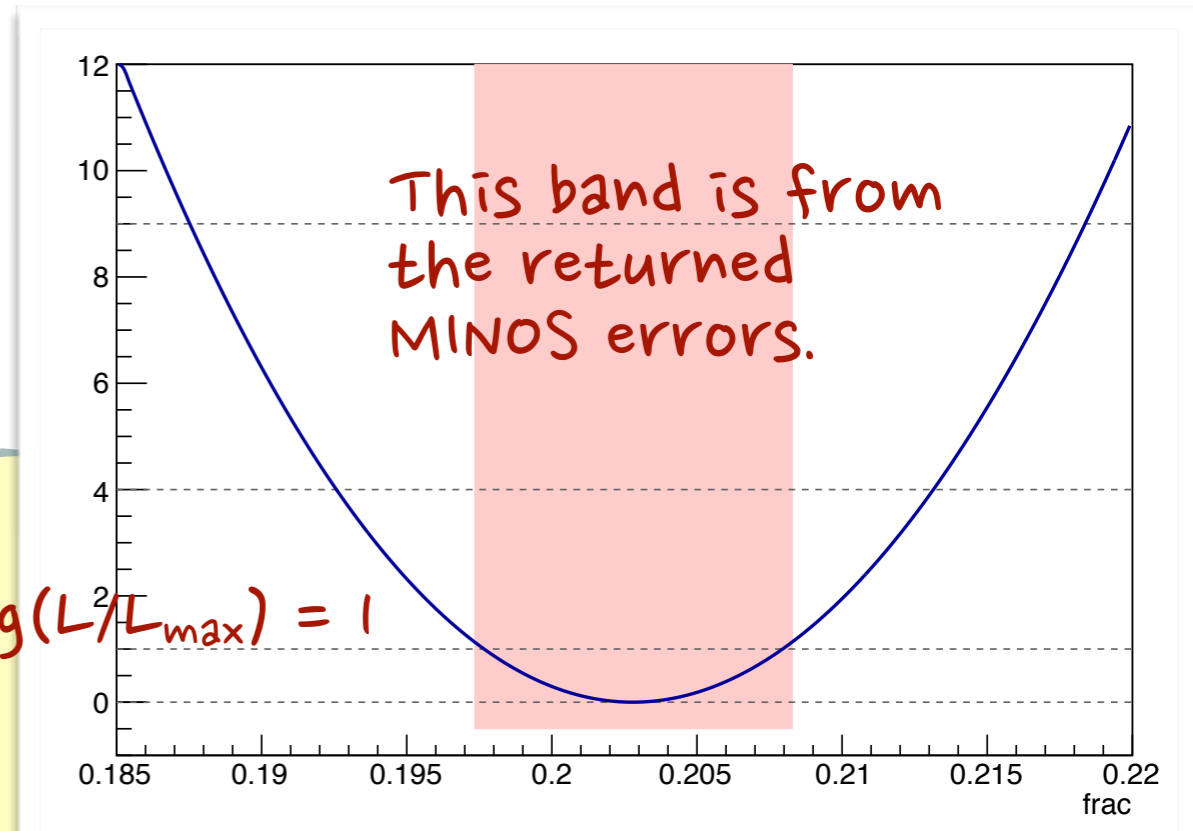
- Plot it, and “verify” the **MINOS errors**.
- Everything good/matched? (*Be aware!*)

partial example_02.cc

```
nll_scan->SetStats(false);  
nll_scan->SetLineWidth(2);  
nll_scan->SetMinimum(-1.);  
nll_scan->SetMaximum(12.);  
nll_scan->GetXaxis()->SetTitle("frac");  
nll_scan->Draw("axis");
```

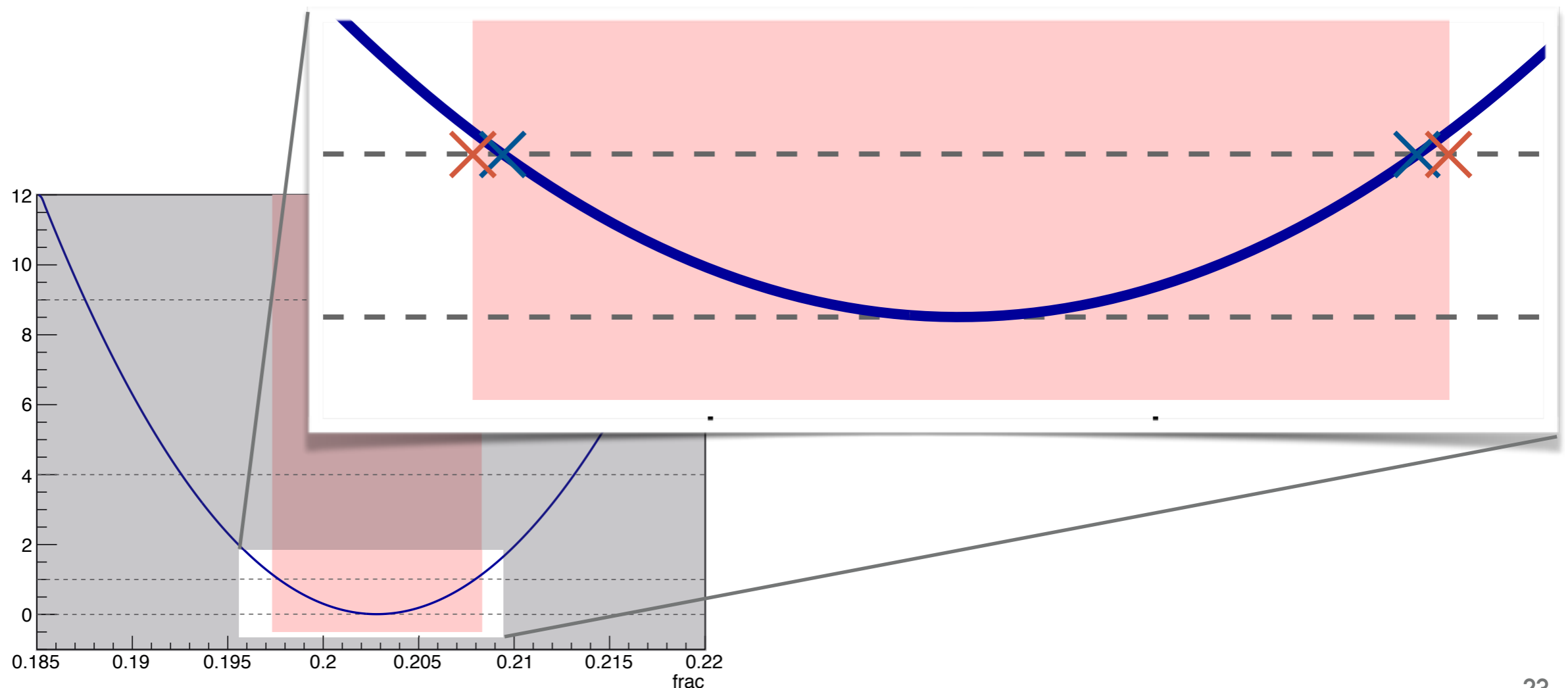
```
TBox box;  
box.SetFillColor(kRed-10);  
RooRealVar* best_frac = (RooRealVar*)res->floatParsFinal().find("frac");  
box.DrawBox(best_frac->getVal()+best_frac->getErrorLo(), -0.5,  
            best_frac->getVal()+best_frac->getErrorHi(), +12.);
```

```
TLine lin;  
lin.SetLineColor(kGray+2);  
lin.SetLineStyle(kDashed);  
for(int n=0; n<=3; n++) lin.DrawLine(0.185, n*n, 0.22, n*n);  
nll_scan->Draw("csame");
```



EXAMPLE: SCAN OVER LIKELIHOOD FUNCTION (ZOOM-IN)

- ▶ You may find the **Minos error band** does not fully match with the cross points of $-2\ln(L/L_{\max})=1$, if you zoom-in to the figure.
- ▶ This is due to the fact that we need to calculate **profile likelihood** instead of a direct scan.



PROFILE LIKEHOOD

- When the likelihood function depends on multiple parameters, we might be interested in only a subset of them. It is often possible to reduce the uninteresting (*nuisance*) parameters by writing them as functions of the parameters of interest only.
- This procedure is called concentration of the parameters and results in the concentrated likelihood function, and most often called the **profile likelihood function**.
- The idea of profile likelihood can also be used to compute confidence intervals that often have better small-sample properties than those based on asymptotic standard errors calculated from the full likelihood.
- Results from profile likelihood analysis can be incorporated in uncertainty analysis of model predictions.

PROFILE LIKEHOOD (CONT.)

- When some of the parameters are “**profiled**” in this likelihood scan, this means some (*or all*) other floated parameters are changed according to the “**maximized likelihood**” during the likelihood scan.
- In the previous example, all other parameters are staying at the values from the first best fit with every parameter floated.
- In order to produce a **proper profile likelihood function**, one can simply ask the Minuit to maximize the likelihood function with respect to those uninterested parameters, while the interested parameter set to its target value.

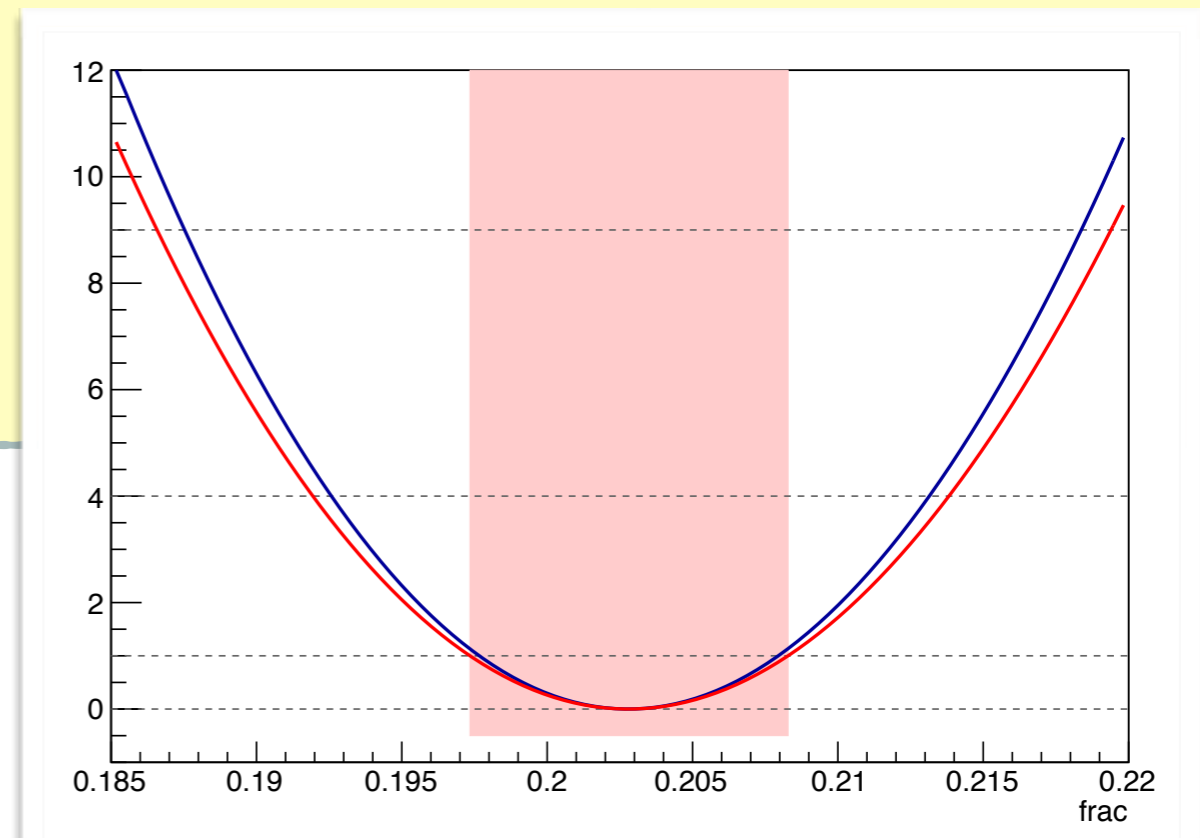
EXAMPLE: PROFILE LIKELIHOOD SCAN

- Just need to modify the code slightly, and run it *for a while*:

partial example_03.cc

```
TH1D *nll_prof = new TH1D("nll_prof", "Profile likelihood", 100, 0.185, 0.22);  
for(int bin=1; bin<=nll_scan->GetNbinsX(); bin++) {  
    frac.setVal(nll_scan->GetBinCenter(bin));  
    frac.setConstant(true); ↪ need to "fit" at each step!  
    model.fitTo(data);  
    nll_prof->SetBinContent(bin, (nll->getVal()-res->minNll())*2.);  
}  
  
nll_scan->Draw("csame");  
nll_prof->SetStats(false);  
nll_prof->SetLineWidth(2);  
nll_prof->SetLineColor(kRed);  
nll_prof->Draw("csame");
```

Now you can see a match
between the likelihood
function & MINOS errors!



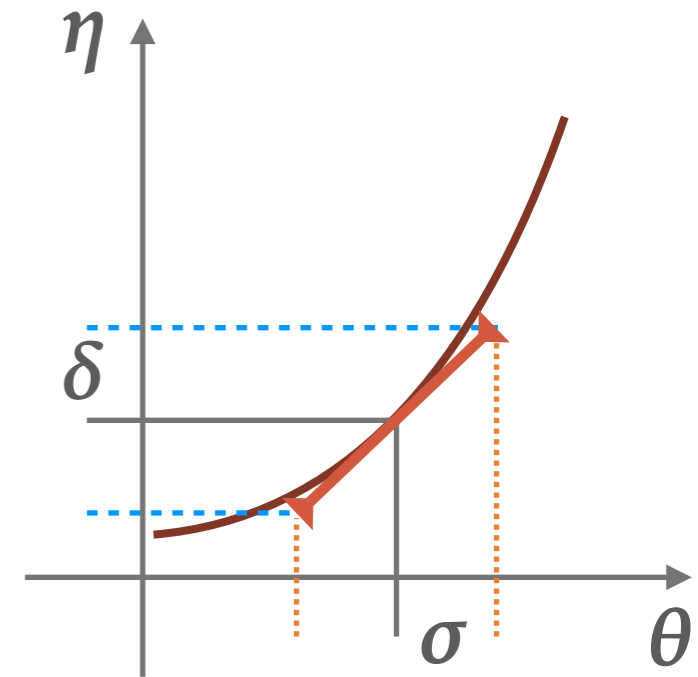
ERROR PROPAGATION

- Consider the case of estimating a set of parameter set and the corresponding covariance matrix:

$$\theta = (\theta_1, \theta_2, \dots, \theta_n) \quad V = \begin{pmatrix} \sigma_1^2 & \sigma_{12} & \dots \\ \sigma_{12} & \sigma_2^2 & \\ \vdots & & \ddots \end{pmatrix}$$

- And would like to convert to another set of parameters

$$\eta = (\eta_1, \eta_2, \dots, \eta_m) \quad H = \begin{pmatrix} \delta_1^2 & \delta_{12} & \dots \\ \delta_{12} & \delta_2^2 & \\ \vdots & & \ddots \end{pmatrix}$$



where they are known functions of input parameter set θ .

- One can take a linear approximation around the given central values of θ , and estimate the slopes by Taylor expansion.

$$H_{ij} = \sum_{k,l} \frac{\partial \eta_i}{\partial \theta_k} \frac{\partial \eta_j}{\partial \theta_l} V_{kl} \quad \text{or} \quad H = A^T V A \quad \text{where } \mathbf{A} \text{ is the Jacobian, as } \partial \eta_j / \partial \theta_l \text{ in the matrix form}$$

ERROR PROPAGATION (II)

- Example: consider a differentiable function $f(a,b)$ of two variables a and b . It can be expanded as

$$f \approx f_0 + \frac{\partial f}{\partial a} a + \frac{\partial f}{\partial b} b + \dots$$

- If the variables a and b has the error of σ_a and σ_b , the combined error on σ_f can be given by

$$\sigma_f^2 \approx \left(\frac{\partial f}{\partial a}\right)^2 \sigma_a^2 + \left(\frac{\partial f}{\partial b}\right)^2 \sigma_b^2 + 2 \frac{\partial f}{\partial a} \frac{\partial f}{\partial b} \sigma_{ab}$$

according to the formulae given in the previous slide.

- Note if no correlation between a and b , the variance formula can be simplified to the usual quadrature sum:

$$\sigma_f \approx \sqrt{\left(\frac{\partial f}{\partial a}\right)^2 \sigma_a^2 + \left(\frac{\partial f}{\partial b}\right)^2 \sigma_b^2}$$

ERROR PROPAGATION (III)

- In most of the cases this is not so trivial, in particular if asymmetric errors are in the consideration.
 - Naive sum in quadrature of positive/negative errors leads to a wrong estimate. Could be biased!
 - A model of the non-linear dependence may be needed for quantitative calculations.
- It would be much better to know the **original PDF** and propagate/combine the information properly!
- Have to be very careful about interpreting the meaning of the result since only the mean value and variance can be propagated in the linear way (as introduced earlier), but not the most probable value.
- **Tips:** whenever possible, it would be better to **use a single fit** rather than multiple cascade fits, and quote the final asymmetric errors only.



Different constructions of the estimator are required to solve issues in different cases!

EXTENDED LIKELIHOOD

- As introduced earlier, if the number of observations N is also a random variable, the **extended likelihood function** is the one to be used.
- In the case of Poissonian, with number of signal (S) and number of background (B) processes:

$$L(X_i; S, B, \theta) = \frac{(S + B)^N e^{-(S+B)}}{N!} \prod_{i=1}^N [f_S P_S(X_i; \theta) + f_B P_B(X_i; \theta)]$$

$$= \frac{e^{-(S+B)}}{N!} \prod_{i=1}^N [S P_S(X_i; \theta) + B P_B(X_i; \theta)]$$

Note the change of fractions (f_S, f_B) to the yields (S, B)

- The fit to be carried out simultaneously on S, B , and θ :

$$-2 \ln L = 2 \left\{ S + B + \sum_{i=1}^N \ln[S P_S(X_i; \theta) + B P_B(X_i; \theta)] - \ln N! \right\}$$

N is a constant during minimization. Can be dropped!

EXAMPLE: SCAN WITH EXTENDED LIKELIHOOD

-
- Let's make a comparison the scanned curves between standard likelihood and extended likelihood using the same model in the previous example.

partial example_04.cc

```
▪ ▪ ▪ ▪ ▪
RooRealVar mass("mass", "mass", 0., 2.);
RooDataSet data("data", "data", nt, RooArgSet(mass));

RooRealVar mu("mu", "mu", 1.0, 0.5, 1.5);
RooRealVar sigma("sigma", "sigma", 0.05, 0.001, 0.15);
RooGaussian gaus("gaus", "gaus", mass, mu, sigma);
RooRealVar slope("slope", "slope", -0.3, -10., 10.);
RooPolynomial linear("linear", "linear", mass, RooArgSet(slope));

▪ ▪ ▪ ▪ ▪
RooRealVar ns("ns", "# of signal", 2000., 0., 20000.);
RooRealVar nb("nb", "# of background", 8000., 0., 20000.);
RooAddPdf model_ext("model_ext", "model w/ ext likelihood",
                    RooArgList(gaus, linear), RooArgList(ns, nb));

▪ ▪ ▪ ▪ ▪
RooFitResult *res_ext = model_ext.fitTo(data, Save(true), Minos(true));
RooAbsReal* nll_ext = model_ext.createNLL(data);
```

the extended likelihood need ns & nb!

EXAMPLE: SCAN WITH EXTENDED LIKELIHOOD (CONT.)

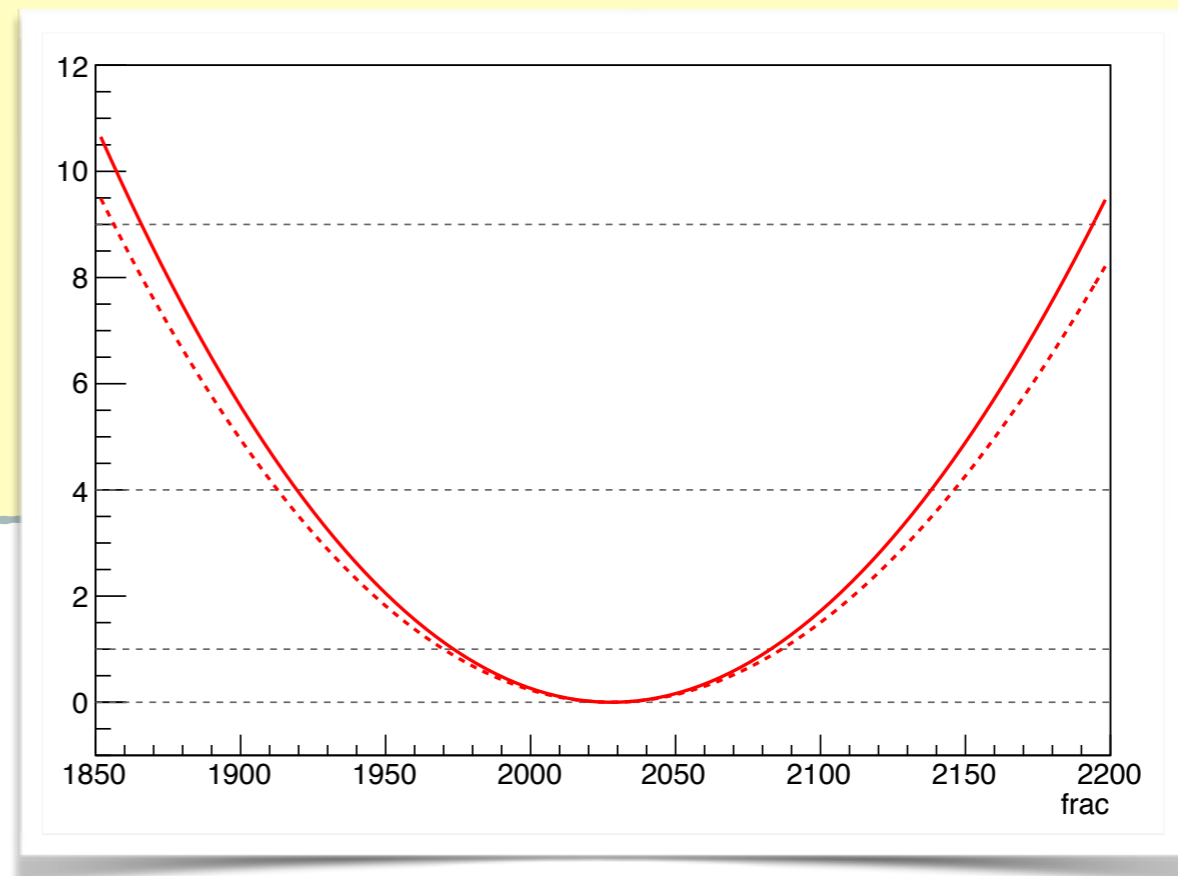
- Perform the scan over **number of signal** instead of the fraction, and let the rest of parameters profiled.

partial example_04.cc

```
TH1D *scan_ext = new TH1D("scan_ext", "Extended likelihood", 100, 1850., 2200.);  
for(int bin=1; bin<=scan_ext->GetNbinsX(); bin++) {  
    ns.setVal(scan_ext->GetBinCenter(bin));  
    ns.setConstant(true);  
    model_ext.fitTo(data);  
    scan_ext->SetBinContent(bin, (nll_ext->getVal()-res_ext->minNll())*2.);  
}
```

```
scan_std->Draw("csame");  
scan_ext->SetStats(false);  
scan_ext->SetLineWidth(2);  
scan_ext->SetLineColor(kRed);  
scan_ext->SetLineStyle(kDashed);  
scan_ext->Draw("csame");
```

The extended likelihood should give a lightly larger (correct!) error!



ADD CONSTRAINTS TO THE LIKELIHOOD

- If knowledge is available for some of the fit parameters, one can also include this information into the likelihood.
- This can be incorporated by multiplying the constrained PDF to the original likelihood. i.e. $L' = L(X|\theta) \times p(\lambda)$.
- As an example, consider the parameter λ is to be constrained with a Gaussian PDF with mean μ_λ and σ_λ :

$$p(\lambda) = f(\lambda; \mu_\lambda, \sigma_\lambda) = \frac{1}{\sqrt{2\pi}\sigma_\lambda} \exp \left[-\frac{(\lambda - \mu_\lambda)^2}{2\sigma_\lambda^2} \right]$$

- The minimization to be performed on the global likelihood function:

$$-2 \ln L' = -2 \ln L - 2 \ln f(\lambda; \mu_\lambda, \sigma_\lambda) = -2 \ln L + \frac{(\lambda - \mu_\lambda)^2}{\sigma_\lambda^2} + \dots$$

- In this particular case the constraint term is nothing more than a sum of squares. For non-Gaussian PDF, it will result a different form for sure.

EXAMPLE: ADDING CONSTRAINT WITH ROOFIT

- An example of adding an external constraint to the likelihood fit within RooFit framework:

The signal is wider than it should be because we constrained it to a wrong value!

partial example_05.cc

```
TFile *fin = new TFile("example_data.root");
TNtupleD* nt = (TNtupleD *)fin->Get("nt");

RooRealVar mass("mass", "mass", 0., 2.);
RooDataSet data("data", "data", nt, RooArgSet(mass));

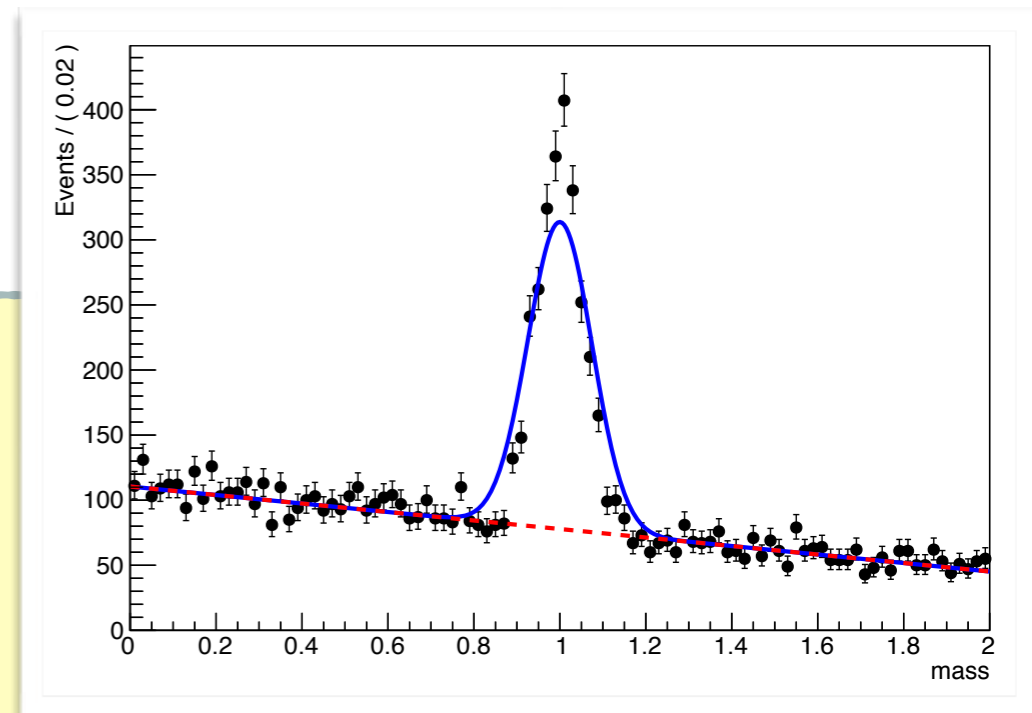
RooRealVar mu("mu", "mu", 1.0, 0.5, 1.5);
RooRealVar sigma("sigma", "sigma", 0.05, 0.001, 0.15);
RooGaussian gaus("gaus", "gaus", mass, mu, sigma);
RooRealVar slope("slope", "slope", -0.3, -10., 10.);
RooPolynomial linear("linear", "linear", mass, RooArgSet(slope));

RooGaussian cons("cons", "const. model", sigma, RooConst(0.08), RooConst(0.001));

RooRealVar ns("ns", "ns", 2000, 0., 20000.);
RooRealVar nb("nb", "nb", 8000, 0., 20000.);
RooAddPdf model("model", "model", RooArgList(gaus, linear), RooArgList(ns, nb));

model.fitTo(data, Minos(true), ExternalConstraints(cons));
```

A Gaussian constraint for sigma



BINNED DATA

- Consider the case when the observations X_i are numbers of events in histogram bins.
- Generally the observations X_i will have a multinomial distribution; in the large number limit, the distribution becomes asymptotically Gaussian.

- Neglecting the correlations, the quantity to be minimized is

$$Q^2 = \sum_{i=1}^N \frac{[X_i - E(\theta)]^2}{\sigma_i^2(\theta)} \approx \sum_{i=1}^N \frac{[X_i - E(\theta)]^2}{E(\theta)} \quad \begin{array}{l} \text{In the case of many bins, } p \ll 1 \\ \sigma_i^2 \theta \approx E(\theta) \end{array}$$

- For practical reason, it is often to adopt the **modified minimum chi-square estimator**, which is consistent with the estimator above in the limit of large N :

$$Q^2 = \sum_{i=1}^N \frac{[X_i - E(\theta)]^2}{X_i}$$

BINNED LIKELIHOOD METHOD

- Apply the maximum likelihood method to the binned observations, the binned likelihood estimator can be realized by maximizing

$$\ln L = \sum_{i=1}^N X_i \ln E(\theta)$$

- Asymptotically this estimator approaches to the previous chi-square estimator. Generally the ML estimator has a better property (faster converge, no problems with null bins, etc).
- When the Poisson distribution is considered for the yields, the extended binned likelihood (given by Baker-Cousins paper) can be expressed as

$$\ln L = \sum_{i=1}^N \left[E(\theta) - X_i + X_i \ln \left(\frac{X_i}{E(\theta)} \right) \right]$$

Remark: this does not work for the bins with fractional numbers.

WEIGHTED DATA

- For the least squares method, the formulation is more-or-less straightforward: just imagine you have some “*bigger*” events and “*smaller*” events:

$$Q^2 = \sum_{i=1}^N \frac{1}{\sigma_i^2} \left(\sum_{j=1}^{n_i} w_{ij} - b_i \right)^2$$

where b_i is the expected content for i^{th} bin, w_{ij} is the weight for j^{th} out of n_i events in the i^{th} bin.

- However it is rather tricky for maximum likelihood method with weighted events. Even the likelihood function can be written as

$$\ln L = \sum_{i=1}^N w_i \ln P(X_i; \theta)$$

But the **uncertainties of the floated parameters (covariance matrix) cannot be calculated directly.**

WEIGHTED DATA (CONT.)

- The leading correction of the covariance can be obtained by

$$V' = V \cdot C^{-1} \cdot V$$

where V is the covariance calculated with the likelihood function given in the previous slide, and C is the covariance calculated with the likelihood function with the squared weights:

$$\ln L' = \sum_{i=1}^N w_i^2 \ln P(X_i; \theta)$$

- And this is the reason why you may see this (*strange?*) message when you carried out such a weighted likelihood fit with RooFit:

```
[#0] WARNING:InputArguments -- RooAbsPdf::fitTo(p2) WARNING: a likelihood fit is request of what appears to be weighted data.
```

While the estimated values of the parameters will always be calculated taking the weights into account, there are multiple ways to estimate the errors on these parameter values...

For more details, please check the F.James's book

EXAMPLE: FIT TO A WEIGHTED DATA

- Here are an example for how to deal with weighted data within RooFit:

partial example_06.cc

```
RooRealVar x("x","x",-5.,5.);  
RooRealVar w("w","w",0.,1.);
```

```
RooDataSet data("data","nomial data",RooArgSet(x));  
RooDataSet data_wgt("data_wgt","weighted data",RooArgSet(x,w),"w");
```

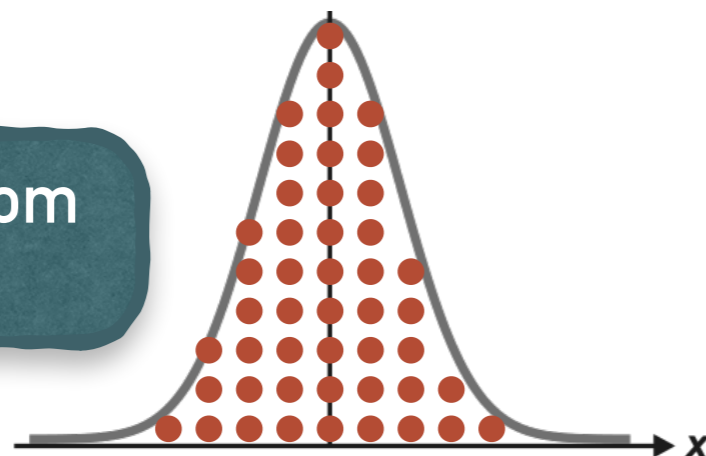
```
TRandom3 rnd;  
for (int i=0;i<1000;i++) {  
  x.setVal(rnd.Gaus(0.,1.));  
  data.add(x);
```

using variable "w"
as the weight of the event

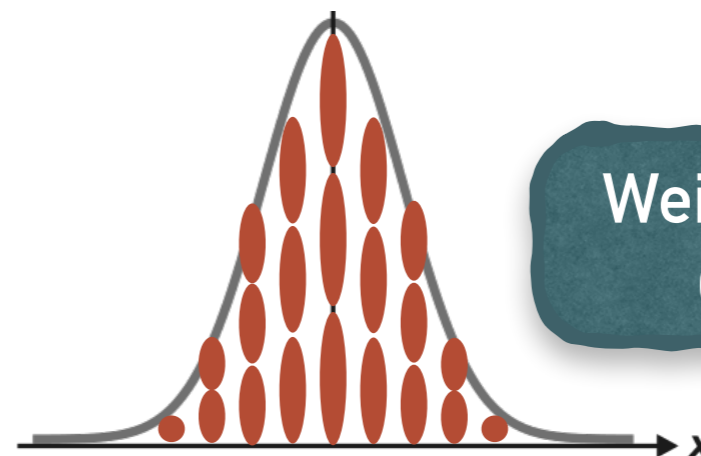
```
  x.setVal(rnd.Uniform(-5.,5.));  
  data_wgt.add(x,TMath::Gaus(x.getVal(),0.,1.));  
}
```

"x" is actually
uniformly distributed

Nominal random
distribution



Weighted random
distribution



EXAMPLE: FIT TO A WEIGHTED DATA (CONT.)

- Then if we fit to these two different data sets:

partial example_06.cc

```
RooRealVar mu("mu", "mu", 0., -0.5, +0.5);
RooRealVar sigma("sigma", "sigma", 1., 0.8, 1.2);
RooGaussian gaus("gaus", "gaus", x, mu, sigma);

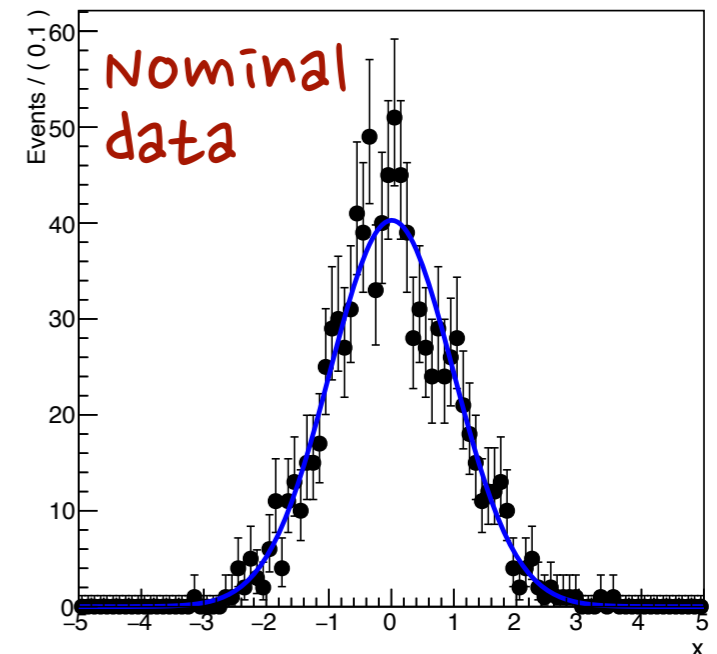
TCanvas *c1 = new TCanvas("c1", "c1", 400, 800);
c1->Divide(1, 2);

c1->cd(1);
gaus.fitTo(data, Minos(true));
RooPlot* frame1 = x.frame();
data.plotOn(frame1);
gaus.plotOn(frame1);
frame1->Draw();

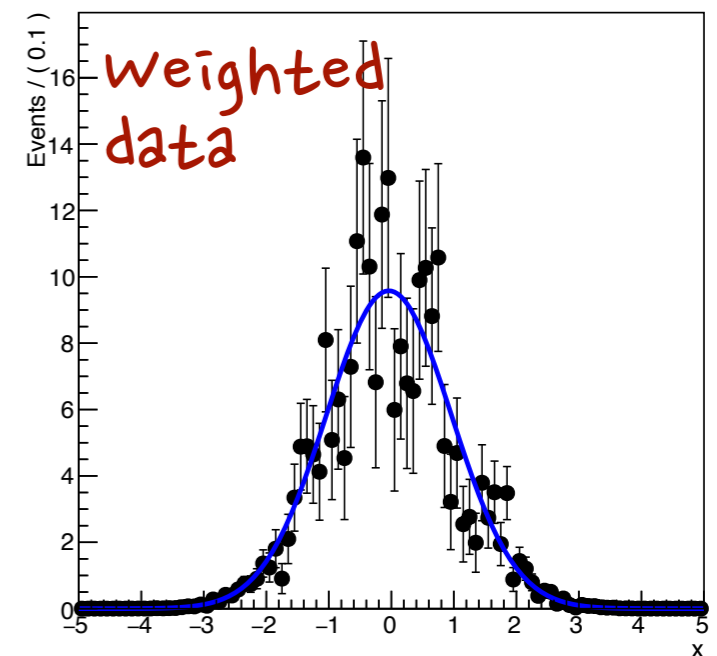
c1->cd(2);
gaus.fitTo(data_wgt, Minos(true));
RooPlot* frame2 = x.frame();
data_wgt.plotOn(frame2);
gaus.plotOn(frame2);
frame2->Draw();
```

You should be able to find the "warning" during the fit to weighted data.

A RooPlot of "x"



A RooPlot of "x"





Case Study

Time for a couple of extended examples!

SIMULTANEOUS FIT

- It is very common to perform a fit with several different data sets together, in order to extract **a single common (*physics*) parameter**.
- In this case, **a simultaneous fit** is the common/good solution.
- Surely one can, in principle, perform individual fits to different set of events and take the average of the results.
 - Depends on the chosen method, but the most simple way is to assume the results are Normally distributed and average with standard error propagation.
 - This does not work with complicated (*or multi-dimensional*) cases obviously.
- Within RooFit, a simultaneous fit can be implemented easily with a **RooSimultaneous** model.

EXAMPLE: SIMULTANEOUS FIT

- A simple RooFit example code is as following, assuming to extract a common “production cross section” out of two different channels:

partial example_07.cc

```
// observables
```

```
RooRealVar mass("mass", "mass", 0., 2.);  
RooCategory channel("channel", "channel");  
channel.defineType("decay1", 1);  
channel.defineType("decay2", 2);
```

↪ require at least one discrete category for defining “channels”

```
// models for decay #1
```

```
RooGaussian ch1_gaus("ch1_gaus", "gaus", mass, ch1_mu, ch1_sigma);  
RooPolynomial ch1_linear("ch1_linear", "linear", mass, RooArgSet(ch1_slope));
```

```
// models for decay #2
```

```
RooGaussian ch2_gaus("ch2_gaus", "gaus", mass, ch2_mu, ch2_sigma);  
RooPolynomial ch2_linear("ch2_linear", "linear", mass, RooArgSet(ch2_slope));
```

```
RooRealVar XS("XS", "cross section", 0.2, 0.0, 0.5);  
RooRealVar ch1_norm("ch1_norm", "decay1 norm", 4000.);  
RooRealVar ch2_norm("ch2_norm", "decay2 norm", 6000.);  
RooProduct ch1_ns("ch1_ns", "decay1 ns", RooArgList(XS, ch1_norm));  
RooProduct ch2_ns("ch2_ns", "decay2 ns", RooArgList(XS, ch2_norm));
```

```
RooRealVar ch1_nb("ch1_nb", "decay1 nb", 4000, 0., 20000.);  
RooRealVar ch2_nb("ch2_nb", "decay2 nb", 7000, 0., 20000.);
```

Relate signal yields with a common cross section value

```
RooAddPdf ch1_model("ch1_model", "decay1 model",  
    RooArgList(ch1_gaus, ch1_linear), RooArgList(ch1_ns, ch1_nb));  
RooAddPdf ch2_model("ch2_model", "decay2 model",  
    RooArgList(ch2_gaus, ch2_linear), RooArgList(ch2_ns, ch2_nb));
```

EXAMPLE: SIMULTANEOUS FIT (II)

partial example_07.cc

```
// now build the simultaneous model by adding two channels
```

```
RooSimultaneous model("model","model",channel);  
model.addPdf(ch1_model,"decay1");  
model.addPdf(ch2_model,"decay2");
```

Build simultaneous
PDF model

```
RooDataSet* ch1_data = ch1_model.generate(mass);  
RooDataSet* ch2_data = ch2_model.generate(mass);
```

```
RooDataSet data("data","joint data",mass,Index(channel),  
    Import("decay1",*ch1_data),Import("decay2",*ch2_data));
```

combine two
data sets

```
model.fitTo(data,Minos(true));
```

```
TCanvas *c1 = new TCanvas("c1","c1",1200,400);  
c1->Divide(3);
```

```
c1->cd(1); // sum of the two channels
```

```
RooPlot* frame1 = mass.frame();  
data.plotOn(frame1);  
model.plotOn(frame1,ProjWData(channel,data));  
frame1->Draw();
```

```
c1->cd(2); // decay1 only
```

```
RooPlot* frame2 = mass.frame();  
data.plotOn(frame2,Cut("channel==1"));  
model.plotOn(frame2,Slice(channel,"decay1"),ProjWData(channel,data));  
frame2->Draw();
```

Projection w/ one of the channels

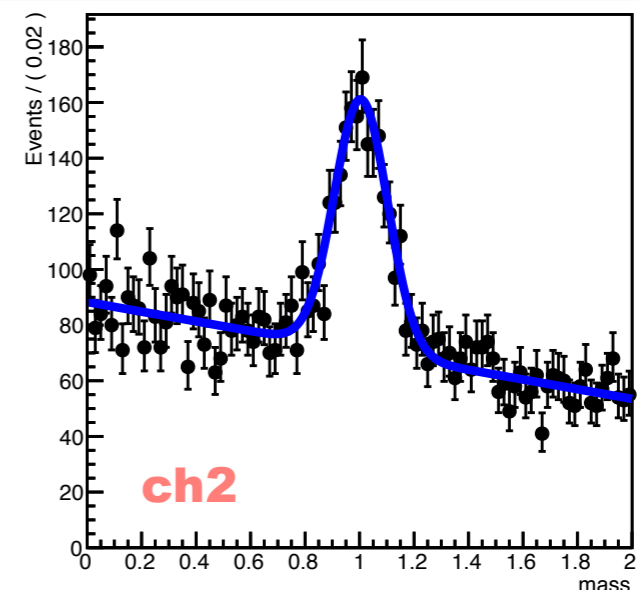
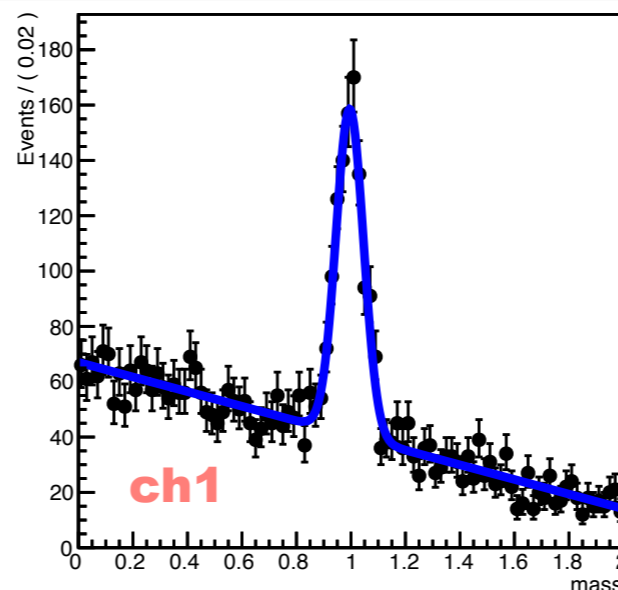
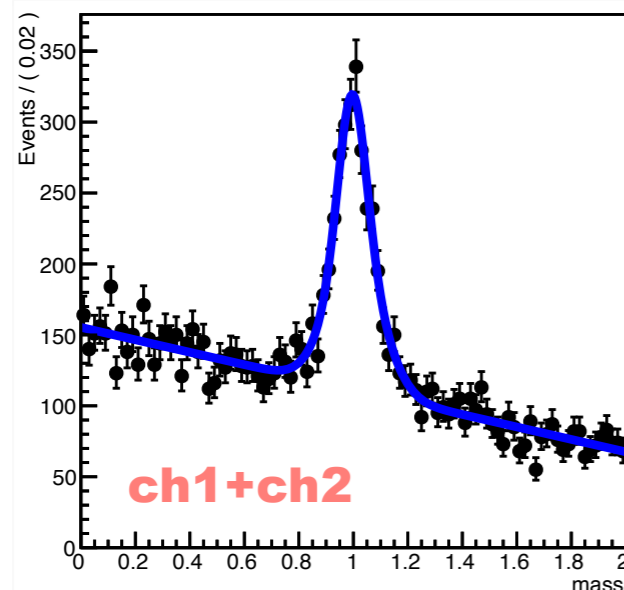
```
• • • • •
```

EXAMPLE: SIMULTANEOUS FIT (III)

➤ The results:

One common "XS"
out of two channels!

```
FCN=-84653.2 FROM MINOS      STATUS=SUCCESSFUL      144 CALLS      1380 TOTAL
EDM=7.41671e-05      STRATEGY= 1      ERROR MATRIX ACCURATE
EXT PARAMETER
NO.  NAME      VALUE      PARABOLIC      MINOS ERRORS
      NAME      VALUE      ERROR      NEGATIVE      POSITIVE
  1  XS      1.85632e-01  6.92234e-03  -6.87829e-03  6.97093e-03
  2  ch1_mu    9.95607e-01  2.76611e-03  -2.77373e-03  2.76121e-03
  3  ch1_nb    4.03445e+03  6.75923e+01  -6.69704e+01  6.82316e+01
  4  ch1_sigma 5.05144e-02  2.43081e-03  -2.37884e-03  2.48783e-03
  5  ch1_slope -3.94938e-01  8.49552e-03  -8.31853e-03  8.67485e-03
  6  ch2_mu    1.00538e+00  5.70435e-03  -5.71690e-03  5.70313e-03
  7  ch2_nb    7.10900e+03  9.34638e+01  -9.31037e+01  9.38420e+01
  8  ch2_sigma 9.80912e-02  5.63931e-03  -5.47038e-03  5.83389e-03
  9  ch2_slope -1.97217e-01  1.30299e-02  -1.27859e-02  1.32827e-02
```

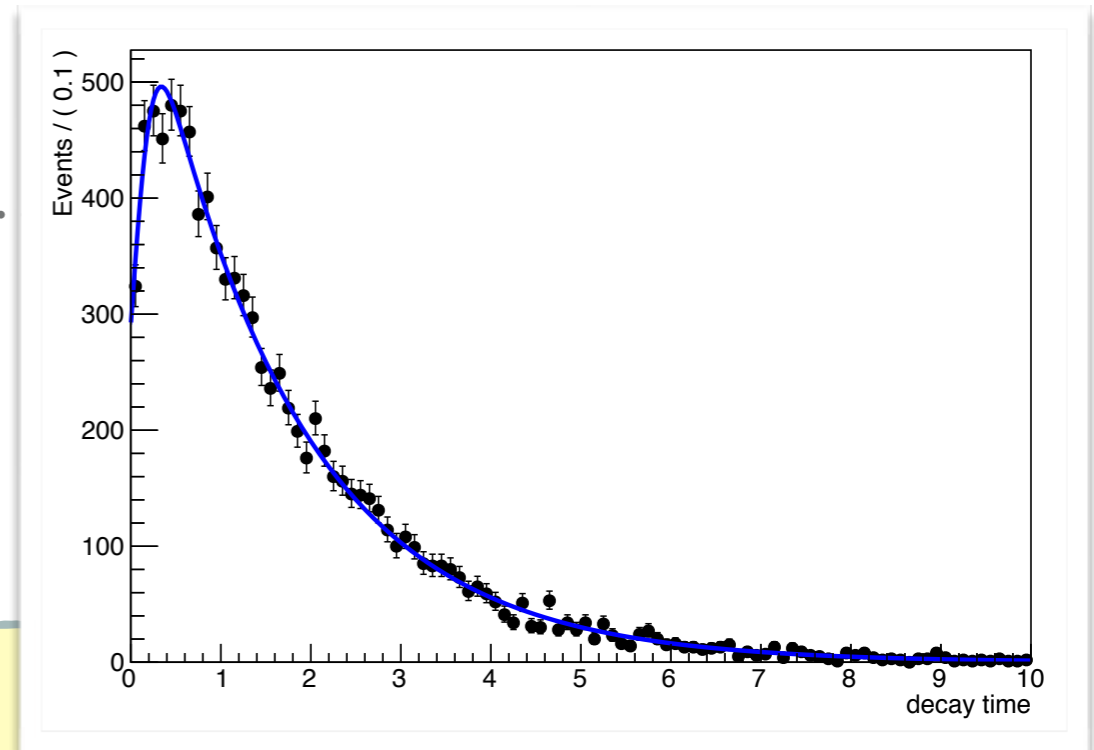


CONDITIONAL OBSERVABLE

- Sometimes if some of the PDF parameters are actually depending on other observables.
- This requires a fit with **conditional observable(s)**. A common practice is the **resolution**, or the uncertainty, of your key observable. For example, measurement of invariant mass or decay time of a particle, accompany with *the estimate of the mass or time resolution*.
- In such a situation, the model of the key observable can in principle taking the resolution parameter into account. e.g.
 - An event with poor (better) resolution, a wider (narrower) PDF can be introduced.
 - It is generally nothing wrong to use an averaged resolution value, but since you have a good estimate of it for each event, why not to use it?

EXAMPLE: FIT WITH RESOLUTION

- Here is an example to illustrate such a case, using the convoluted exponential as discussed before!



example_08.cc

```
// observables, decay time and its error
RooRealVar t("t", "decay time", 0., 10.);
RooRealVar terr("terr", "decay time error", 0.001, 0.4);

// model for decay time error
RooRealVar terr_mu("terr_mu", "terr mu", 0.2);
RooRealVar terr_sigma("terr_sigma", "terr sigma", 0.02);
RooGaussian terr_model("terr_model", "terr model", terr, terr_mu, terr_sigma);

// model for decay time (w/ time error in the resolution model)
RooRealVar tau("tau", "Lifetime", 1.6, 1.2, 2.0);
RooGaussModel res("res", "Resolution model", t, RooConst(0.), terr);
RooDecay t_model("t_model", "t_model", t, tau, res, RooDecay::SingleSided);

RooProdPdf model("model", "full model", t_model, terr_model);
RooDataSet* data = model.generate(RooArgSet(t, terr), 10000);

model.fitTo(*data, Minos(true), ConditionalObservables(terr));

RooPlot *frame = t.frame();
data->plotOn(frame);
model.plotOn(frame);
frame->Draw();
```

every event has a different resolution function.

"terr" is a conditional observable, not really in the fit but as a parameter.

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

Let's revisit the **Bayes theorem** and the **"Bayesian"** estimation now.

REMINDER: BAYES THEOREM

- The form of Bayes theorem used in Bayesian parameter estimation for a particular data set X^0 :

$$p(\theta|X^0) = \frac{p(X^0|\theta)p(\theta)}{\int p(X^0|\theta)p(\theta)d\theta}$$

- $p(\theta|X^0)$ is **posterior probability density** for θ .
- $p(X^0|\theta)$ is the **likelihood function** for N independent observation of a variable X_i and PDF $f_i(X_i|\theta)$. The joint density function can be expressed as

$$p(X|\theta) = \prod_{i=1}^N f_i(X_i|\theta)$$

- $p(\theta)$ is the **prior probability** density for θ .
- The integration in the denominator is the normalization factor.

PRIORS AND POSTERIORS

- The **prior PDF** represents your **personal, subjective, degree of belief** about parameter θ before you do any experiments.
 - If you already have some experimental knowledge about θ (e.g. *from a previous experiment*), the posterior PDF from the previous experiment can be introduced as the prior for the new one.
 - But this implies that, somewhere in the beginning, there must be a prior which contained no experimental evidence!
- **The very first prior** can be thought of as a kind of phase space, or density of possible states of nature. But there is no law of nature that tells us what this density is!
- On the other hand, **the posterior density** already represents all our knowledge about θ , so there is no need to process this PDF any further. But since we want a point estimate here, further operations does require.

BAYESIAN INFERENCE

- The posterior probability is proportional to the product of likelihood function times the prior probability for the unknown parameters θ :

$$p(\theta|X^0) \propto \prod_{i=1}^N f_i(X_i|\theta) \cdot p(\theta)$$

- Based on the **posterior probability** one can evaluate then the average and variance of θ , as well as the point with highest posterior density (HPD)!

- **Note the value which gives the highest posterior density and the average don't coincide in general!**

- By looking for the highest posterior density point (*maximizing the posterior probability*), it is just the **maximum likelihood estimator with a flat prior $p(\theta)$** :

$$L(\theta|X^0) \propto \prod_{i=1}^N f_i(X_i|\theta)$$

BAYESIAN INFERENCE: POISSONIAN CASE

- ▶ Let's calculate the **posterior probability** with Bayes theorem, assuming a prior $p(\mu)$ and the likelihood function $p(n|\mu)$ is Poisson.

$$p(n|\mu) = \frac{\mu^n e^{-\mu}}{n!} \Rightarrow p(\mu|n) = \frac{\frac{\mu^n e^{-\mu}}{n!} \cdot p(\mu)}{\int_0^\infty \frac{\mu^n e^{-\mu}}{n!} p(\mu) d\mu}$$

- ▶ If the prior is uniform, the normalization calculation is basically straightforward; this gives the form of posterior probability as Poisson distribution as well (*here it means given observed n counts, the probability for finding the value of μ*):

$$\int_0^\infty \frac{\mu^n e^{-\mu}}{n!} p(\mu) d\mu = 1 \Rightarrow p(\mu|n) = \frac{\mu^n e^{-\mu}}{n!}$$

- ▶ Here **the value μ with highest posterior density is n** , by maximizing $p(\mu|n)$.

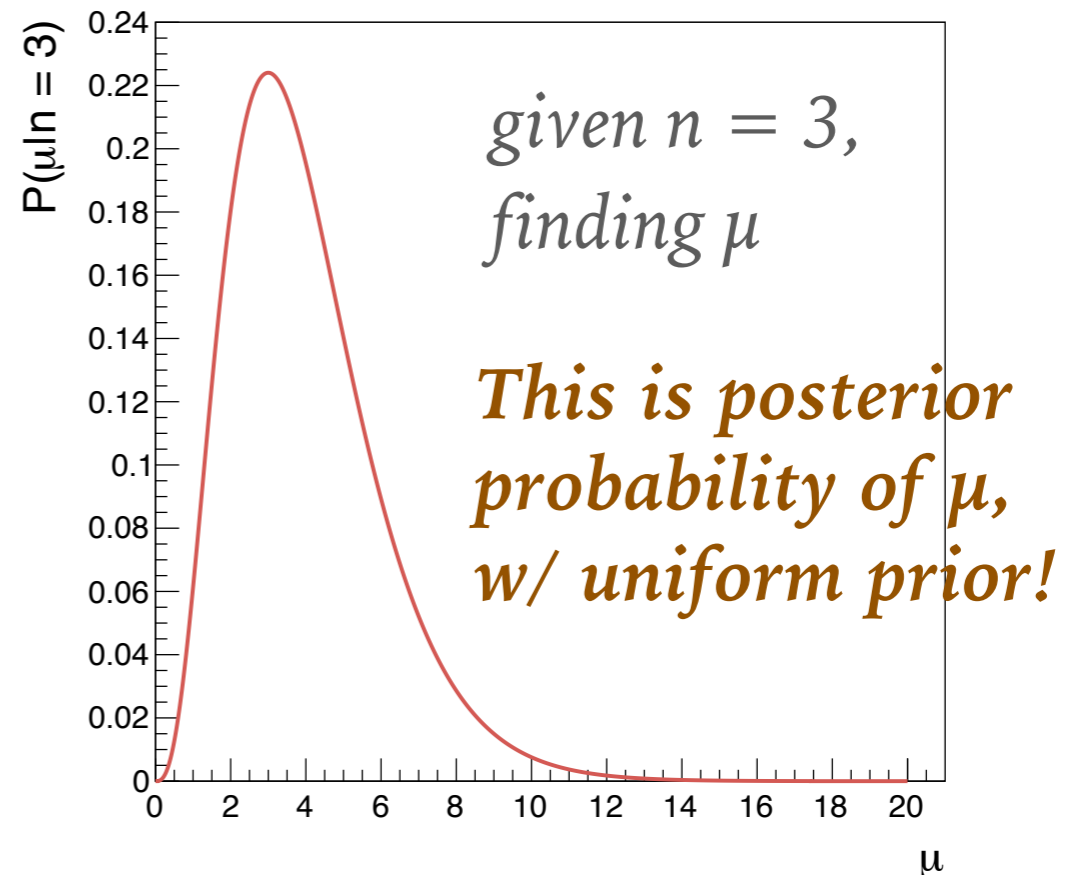
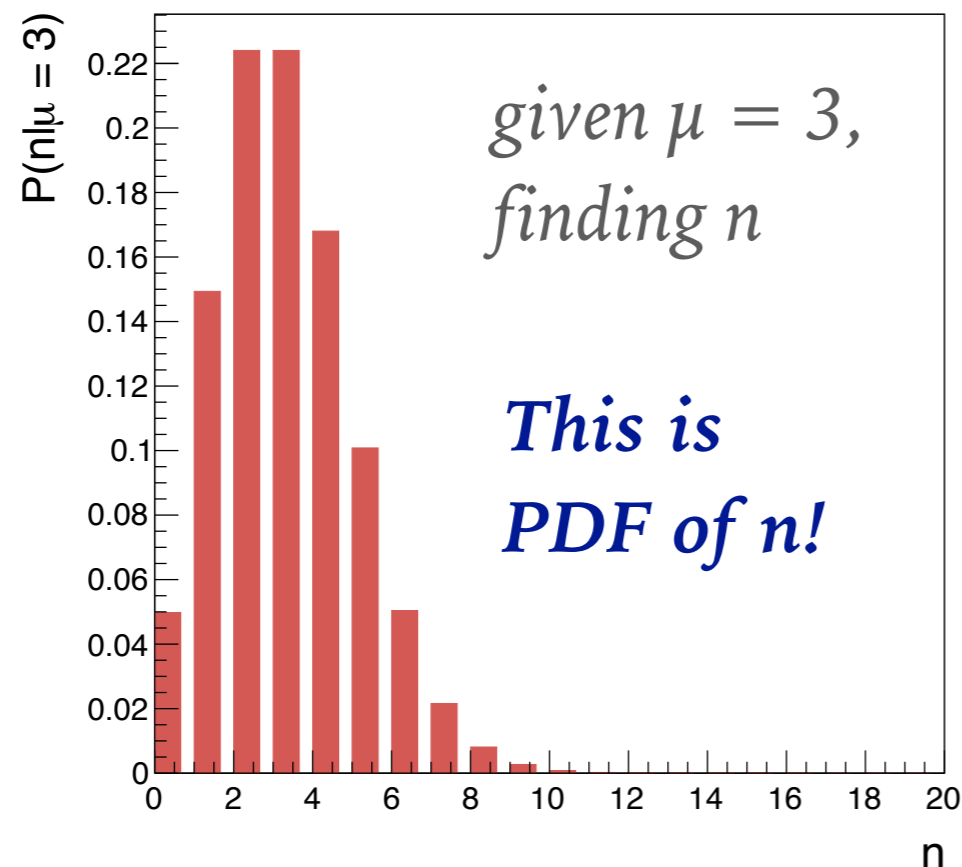
BAYESIAN INFERENCE: POISSONIAN CASE (CONT.)

- Poisson PDF versus posterior probability?

$$p(n|\mu) = \frac{\mu^n e^{-\mu}}{n!}$$



$$p(\mu|n) = \frac{\mu^n e^{-\mu}}{n!}$$



Remark: the right plot is **NOT** just a scanning over μ with given n with original Poisson distribution (which is not a PDF in that case, but just a likelihood)!

BAYESIAN POINT ESTIMATES

- One can see that the method discussed above gives the expected result for the Bayesian point estimate $\hat{\mu}$
 - ⇒ **with n events observed, $\hat{\mu} = n$.**
- However, there were two suspicious things being carried out:
 - **A flat prior was introduced** (*uniformly distributed between zero and positive infinity!*). ie. our prior belief of the true value of μ , integrated between any two finite numbers is zero. That is not really physical.
 - **Our point estimate was the mode** (*position of the peak*) of the PDF, which is not invariant under change of variables. e.g. if we want to estimate μ^2 , we will not obtain $\hat{\mu}^2$.

Any alternative choices?

ANY ALTERNATIVES?

- Another possible Bayesian estimate of μ would be to use the expectation $E(\mu)$ with the posterior density (*average of μ*). With a uniform prior on μ , when n events are observed, this gives $E(\mu) = n + 1$ in fact!

$$E(\mu) = \int_0^{\infty} \mu p(\mu|n) d\mu = n + 1 \quad V(\mu) = n + 1$$

- Since $E(n) = \mu$, one might be happier to see $E(\mu) = n$ also!
- In fact, one can get $E(\mu) = n$ if the **prior PDF** $P(\mu) = 1/\mu$ is chosen. But the $1/\mu$ prior also has other advantages:
 - It could represent somebody's prior belief, since it goes to zero at $\mu = \infty$, and it produces a uniform density on a log scale (*good*). However, it is still not a proper density since it cannot be normalized in the range $(0, \infty)$.
 - It is one of the **Jeffreys priors**, proposed by physicist Harold Jeffreys as being "objective" and it is scale invariant.

CHOICE OF PRIOR

- Unfortunately the $1/\mu$ prior also has some other problems:
 - If one observes zero event ($n=0$), $P(\mu | 0)$ is a delta-function at $\mu=0$!
 - When there is background contributing to the Poisson process, the point estimate breaks down. Have to introduce a different one!
- Nevertheless, as a summary of the common choice of Bayesian estimate:

The choice of priors (*in the case with Poisson estimation*):

- Uniform – gives some reasonable estimates, but has issues.
- Jeffreys $1/\mu$ – Better in theory, but not in practice due to other problems!

The choice of point estimators:

- Take the maximum posterior density gives reasonable results, but not invariant under change of variable;
- Take the expectation $E(\mu)$ is possible, but not invariant either.
- Median of the posterior density is invariant, but is not used much.

COMMENT: JEFFREYS PRIOR

- The Jeffreys prior is a non-informative (*objective*) prior distribution for a parameter space; it is proportional to the square root of the determinant of the **Fisher information matrix $I(\theta)$** :

$$p(\theta) \propto \sqrt{\det \mathcal{I}(\theta)} \quad \mathcal{I}(\theta)_{ij} = E \left[\left(\frac{\partial}{\partial \theta_i} \ln L \right) \left(\frac{\partial}{\partial \theta_j} \ln L \right) \right]$$

- It has the key feature that it is invariant under reparameterization. Some common choices:

Poissonian mean

$$p(\theta) \propto 1/\sqrt{\mu}$$

Gaussian mean

$$p(\theta) \propto 1$$

Poissonian mean w/ background b

$$p(\theta) \propto 1/\sqrt{\mu + b}$$

Gaussian standard deviation

$$p(\theta) \propto 1/\sigma$$

Binomial parameter

$$p(\theta) \propto 1/\sqrt{\epsilon(1 - \epsilon)}$$

Obvious generalization to multi-parameter models is problematic!

BAYESIAN ESTIMATION SUMMARY

- Remember: **Bayesian probability** means the probability is defined as **personal, subjective, degree of belief**.
- Bayesian point estimation is a coherent method which provides a reasonable way to estimate parameters. But it involves two arbitrary choices (*issues*):
 - Which prior PDF to use, and how sensitive is the result to the choice?
 - How to connect the posterior probability to the point estimate?
- Well, one can increase the observations, **the prior probability is significantly modified by data** — then the final posterior probability (*tends toward a Gaussian in most of the cases*) will depend much less from the initial prior probability.
 - But under such a condition, using frequentist or Bayesian approaches does not make much difference.

COMMENT: BAYES THEOREM AS “LEARNING”

➤ **Recall the typical Bayes theorem formula:**

$$p(A|B) = \frac{p(B|A)p(A)}{p(B)}$$

- Before the observation of data B , our degree of belief of A is $p(A)$, as the **prior probability**.
- After observing data B , our degree of belief changes into $p(A|B)$, as the **posterior probability**.
- Probability can be expressed also as a property of non-random variables, e.g. associated with unknown parameter or unknown events;
- As an approach to extend knowledge with subsequent observations, e.g. combine experiment/observations \Rightarrow **just multiply probabilities**.

COMMENT: REPEATED USE OF BAYES THEOREM

- Bayes theorem can be applied sequentially for repeated data observations: **posterior** \Leftrightarrow **learning from experiments**.



$$P_0 = \text{Prior} \quad P_1 \propto P_0 \times L_1 \quad P_2 \propto P_0 \times L_1 \times L_2 \quad P_3 \propto P_0 \times L_1 \times L_2 \times L_3$$

...accumulating more and more observations = **multiply probabilities**

- The observation modifies the prior knowledge of the unknown parameters as if L is a probability distribution function.
- Note applying Bayes theorem directly from prior to multiple observables leads to the same result:

$$P_{1+2+3} = P_0 \times L_{1+2+3} = P_0 \times L_1 \times L_2 \times L_3 = P_3$$

NON-UNIFORM PRIOR AND MAXIMUM LIKELIHOOD

- As discussed in the previous slide, the ML estimator is a special case of Bayes theorem with flat prior.
- Similarly and in practice, **constrained ML estimator** can be seen as the special case of Bayes theorem with non-uniform prior in fact.
- Recall the constraint can be incorporated by multiplying the **constrained PDF to the original likelihood**. i.e.

$$L' = L(X|\theta) \times p(\lambda) = \prod_{i=1}^N f_i(X_i|\theta) \cdot p(\lambda) \quad \lambda \text{ is subset of } \theta$$

- You may find that it matches how we treat the prior PDF in Bayes theorem:

$$p(\theta|X^0) \propto \prod_{i=1}^N f_i(X_i|\theta) \cdot p(\theta)$$

TOOLS FOR COMPUTING POSTERIOR PDF

- The usual problem: parameters with priors needed to be integrated out \Rightarrow multidimensional integration!
- One can perform analytical integration
 - But it is only feasible in very few cases
- Or one can use numerical integration
 - It would be (very) CPU intensive
- Integration with **Markov Chain Monte Carlo**
 - Sampling parameter space efficiently using *random walks*, heading to the regions of higher probability;
 - “Metropolis” algorithm to do sampling according to a PDF.

RooStats::BayesianCalculator
available

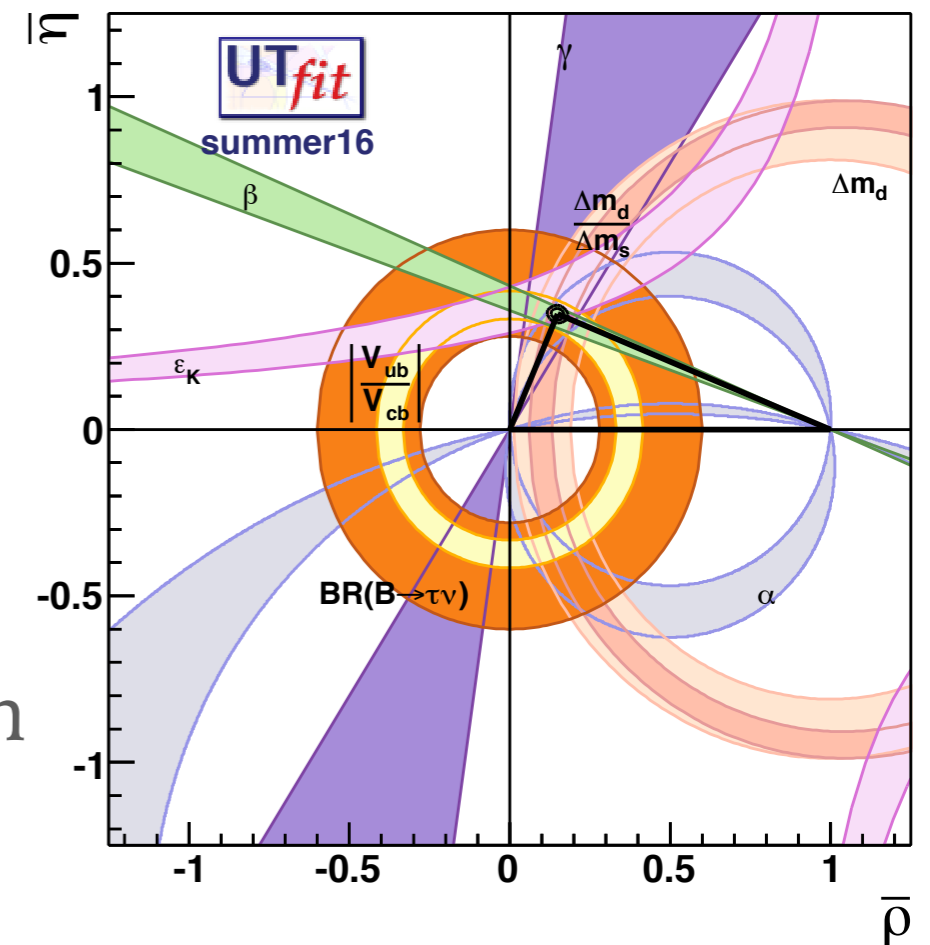
RooStats::MCMCCalculator
available

MARKOV CHAIN MONTE CARLO

- **MCMC** methods are primarily used for calculating numerical approximations of **multi-dimensional integrals**.
- In Bayesian statistics, it has been a key step in making it possible to compute large hierarchical models that require integrations over hundreds/thousands of parameters.
- **Typical algorithm:**
 - 1) Starting from a random point X_i in the parameter space;
 - 2) Generating a proposal point X_p in the vicinity of X_i ;
 - 3) If $p(X_p) > p(X_i)$ accept it as the next point $X_{i+1} = X_p$ else, accept the new point only with a probability $p = p(X_p) / p(X_i)$;
 - 4) Repeating the procedures from 2)
- *Convergence criteria* and *step size* needed to be defined.

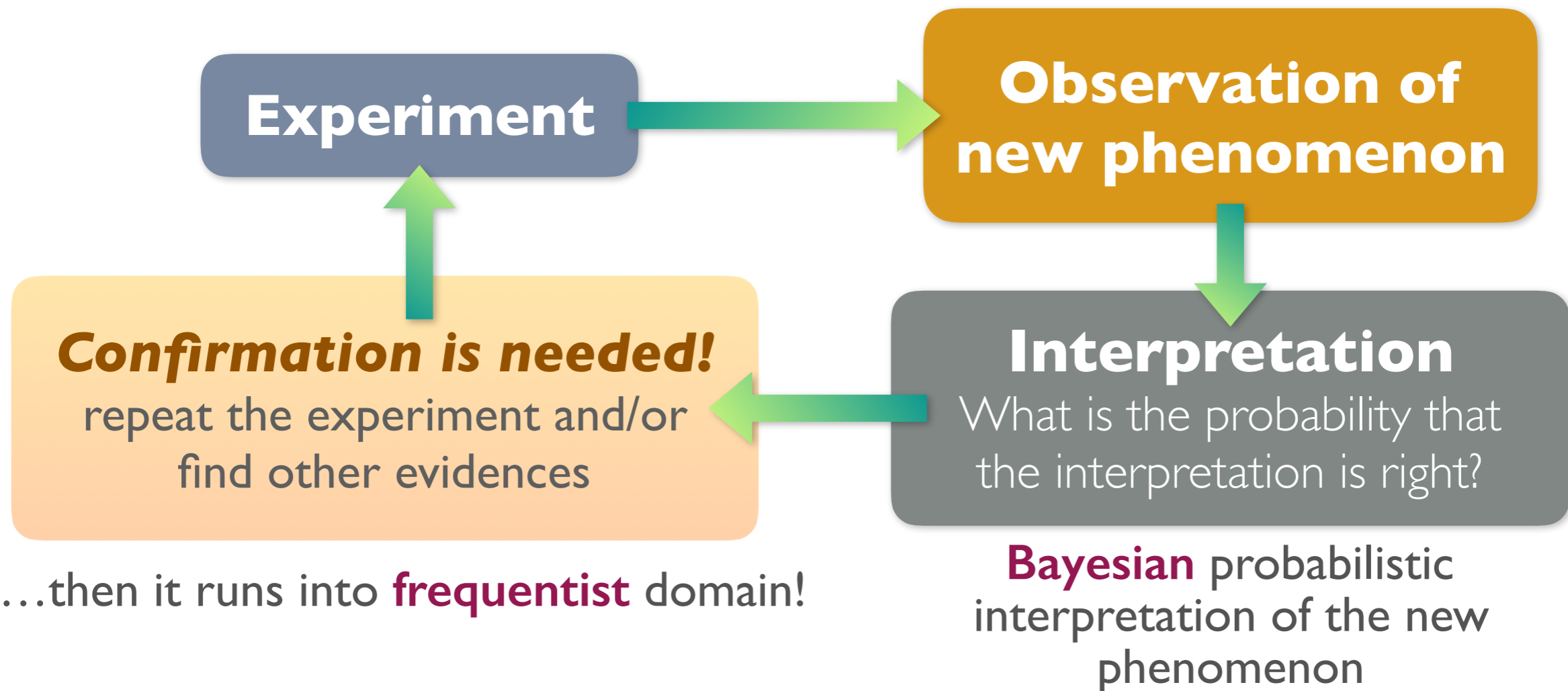
COMMENT: BAYESIAN VERSUS FREQUENTIST

- The frequentist approach was largely favored by scientists, due to its “**experimental-driven definitions**”.
- However, recently Bayesian estimates are getting more popular and provide simpler mathematical methods to perform complicated estimates.
- Also Bayesian estimators properties can be studied with a *frequentist approach* using the toy Monte Carlos, and it is feasible with today’s computers.
- Also preferred by some TH community, e.g. **UT Fit**, with Bayesian determination of the CKM unitarity triangle.



COMMENT: BAYESIAN VERSUS FREQUENTIST (CONT.)

- Bayesian and frequentist approaches have complementary roles in the scientific process in fact:



So both approaches are mandatory!

SUMMARY

- In this lecture we went through the theory (*mathematics*) behind the point estimations.
- Hopefully this gives you a little bit more deeper idea about the maximum likelihood estimator, least-square estimator, and Bayesian inference.
- For the next lecture, we are going to discuss another estimation: how to extract the intervals out of your distributions.