Using priors to exclude unphysical regions

- Do you want publish (only) results restricted to the physical region?
  - It depends very much to what further analysis and/or combinations is needed…
- An interval / parameter estimate that in includes unphysical still represents the best estimate of *this* measurement
  - Straightforward to combined with future measurements, new combined result might be physical (and more precise)
  - You need to decide between 'reporting outcome of this measurement' vs 'updating belief in physics parameter'
- Typical issues with unphysical results in confidence intervals
  - 'Low fluctuation of background' → 'Negative signal' → 95% confidence interval excludes *all* positive values of cross-section.
  - Correct result (it should happen 5% of the time), but people feel 'uncomfortable' publishing such a result
- Can you also exclude unphysical regions in confidence intervals?
  - No concept of prior…But yes, it can be done!

Physical boundaries frequentist confidence intervals

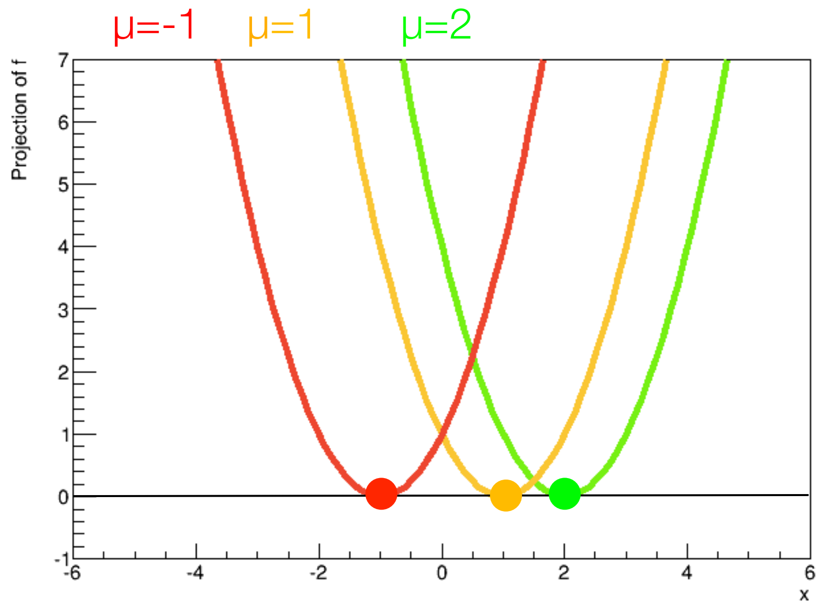
- Solution is to modify the statistic to avoid unphysical region

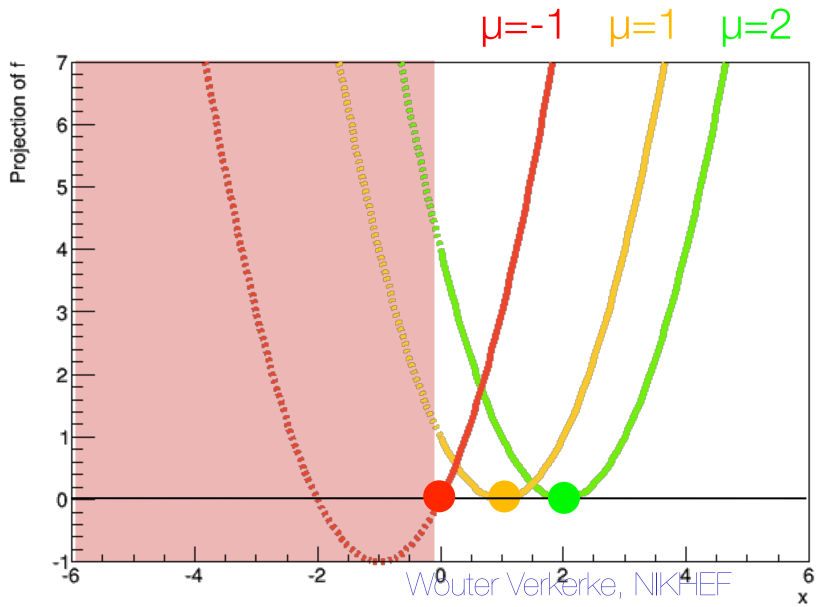
$$t_{\mu}(x) = -2\log\frac{L(x|\mu)}{L(x|\hat{\mu})}$$

Introduce "physical bound" μ>0

$$\tilde{t}_{\mu}(x) = \begin{cases} -2\log\dfrac{L(x|\mu)}{L(x|\hat{\mu})} & \forall\hat{\mu}\geq 0 \\ -2\log\dfrac{L(x|\mu)}{L(x|0)} & \forall\hat{\mu}<0 \end{cases}$$

If μ<0, use 0 in denominator
→ *Declare data maximally compatible with hypothesis μ=0*





Wouter Verkerke, NIKHEF

Physical boundaries in frequentist confidence intervals

- What is effect on *distribution* of test statistic?

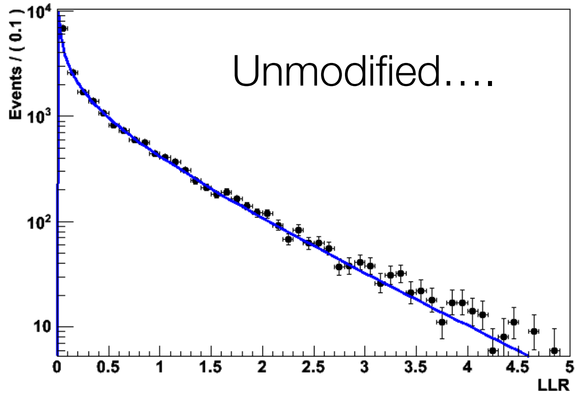
$$t_\mu(x) = -2\log\frac{L(x\,|\,\mu)}{L(x\,|\,\hat{\mu})}$$

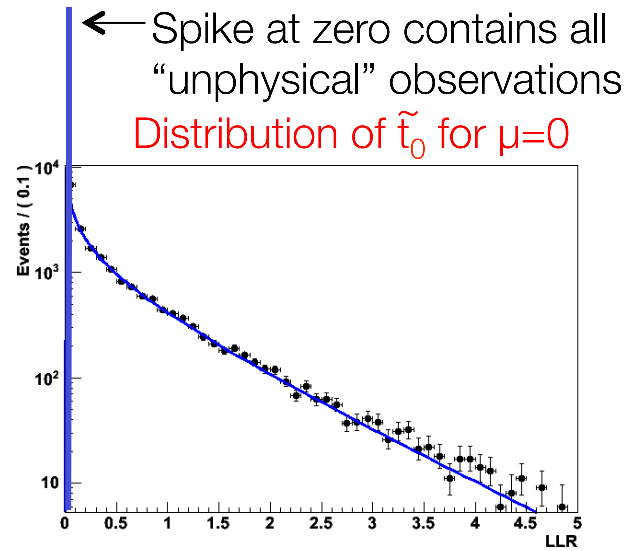
Introduce “physical bound” μ>0

$$\tilde{t}_\mu(x) = \begin{cases} -2\log\dfrac{L(x\,|\,\mu)}{L(x\,|\,\hat{\mu})} & \forall\hat{\mu} \geq 0 \\[2ex] -2\log\dfrac{L(x\,|\,\mu)}{L(x\,|\,0)} & \forall\hat{\mu} < 0 \end{cases}$$

If μ<0, use 0 in denominator
→ *Declare data maximally compatible with hypothesis μ=0*

Distribution of $\tilde{t}_0$ for μ=2

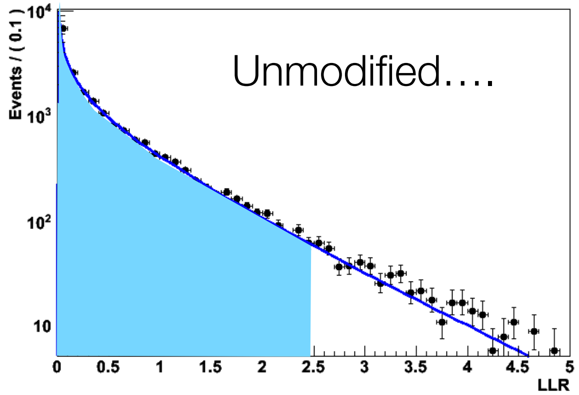




Physical boundaries frequentist confidence intervals

- What is effect on *acceptance interval* of test statistic?

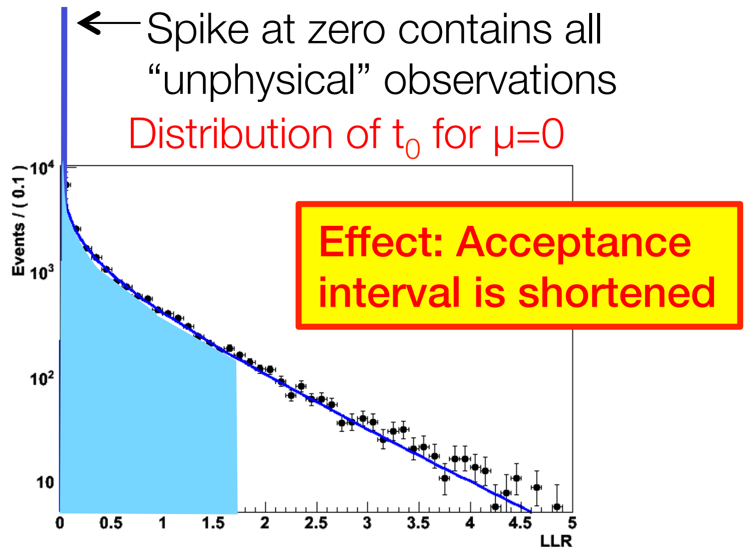
$$t_\mu(x) = -2\log\frac{L(x\,|\,\mu)}{L(x\,|\,\hat{\mu})}$$



Introduce "physical bound" μ>0

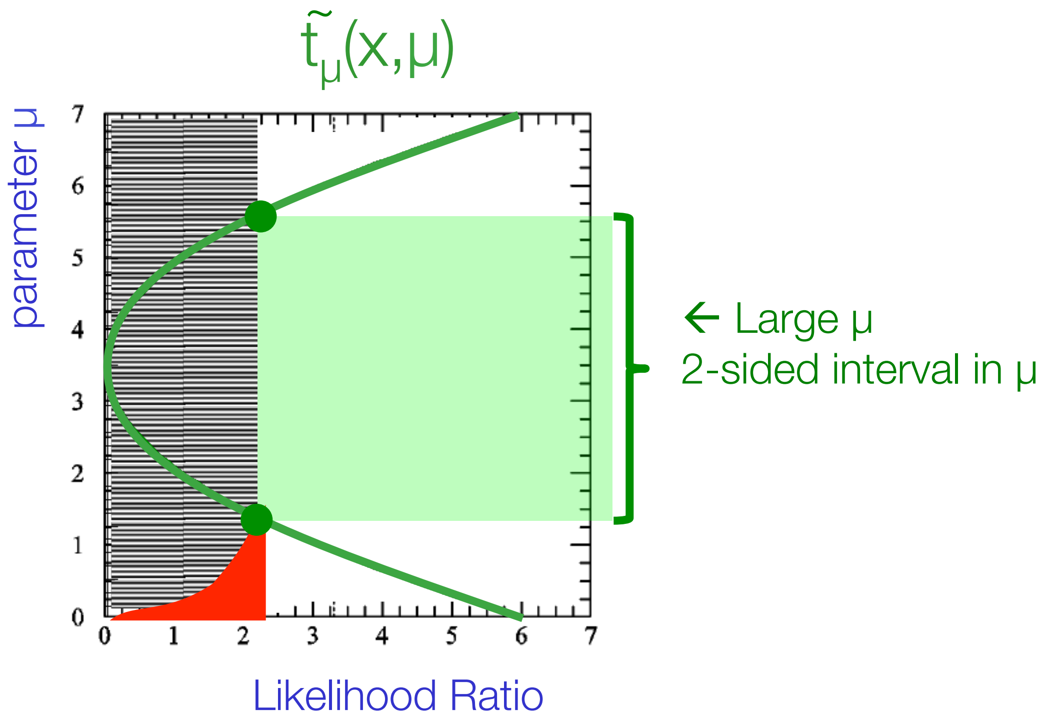
$$\tilde{t}_\mu(x) = \begin{cases} -2\log\dfrac{L(x\,|\,\mu)}{L(x\,|\,\hat{\mu})} & \forall\hat{\mu}\geq 0 \\[2ex] -2\log\dfrac{L(x\,|\,\mu)}{L(x\,|\,0)} & \forall\hat{\mu}<0 \end{cases}$$

If μ<0, use 0 in denominator
→ *Declare data maximally compatible with hypothesis μ=0*



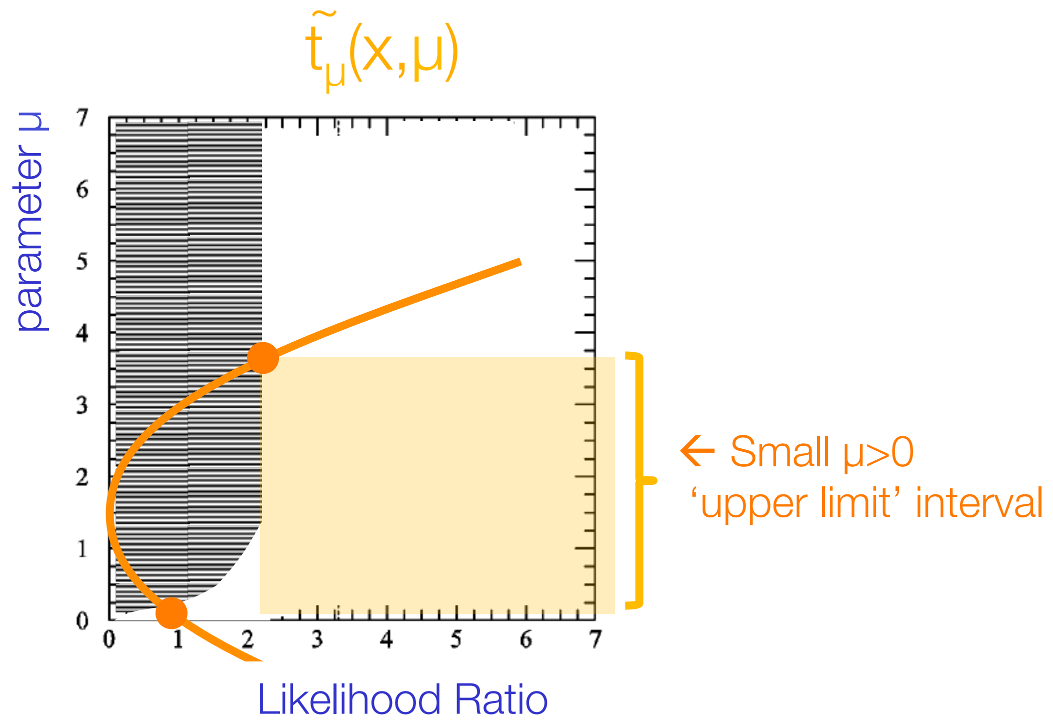
Physical boundaries frequentist confidence intervals

- Putting everything together – the confidence with modified t_μ
- Confidence belt ‘pinches’ towards physical boundary
- Offsetting of likelihood curves for measurements that prefer $\mu<0$



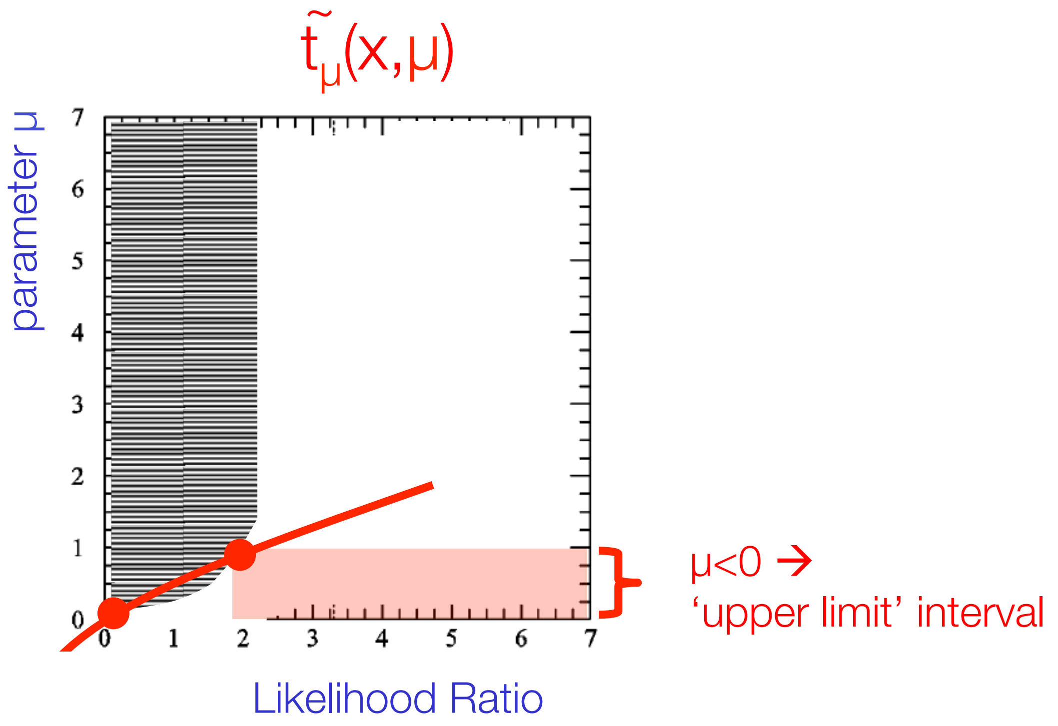
Physical boundaries frequentist confidence intervals

- Putting everything together – the confidence with modified t_μ
- Confidence belt 'pinches' towards physical boundary
- Offsetting of likelihood curves for measurements that prefer μ<0



Physical boundaries frequentist confidence intervals

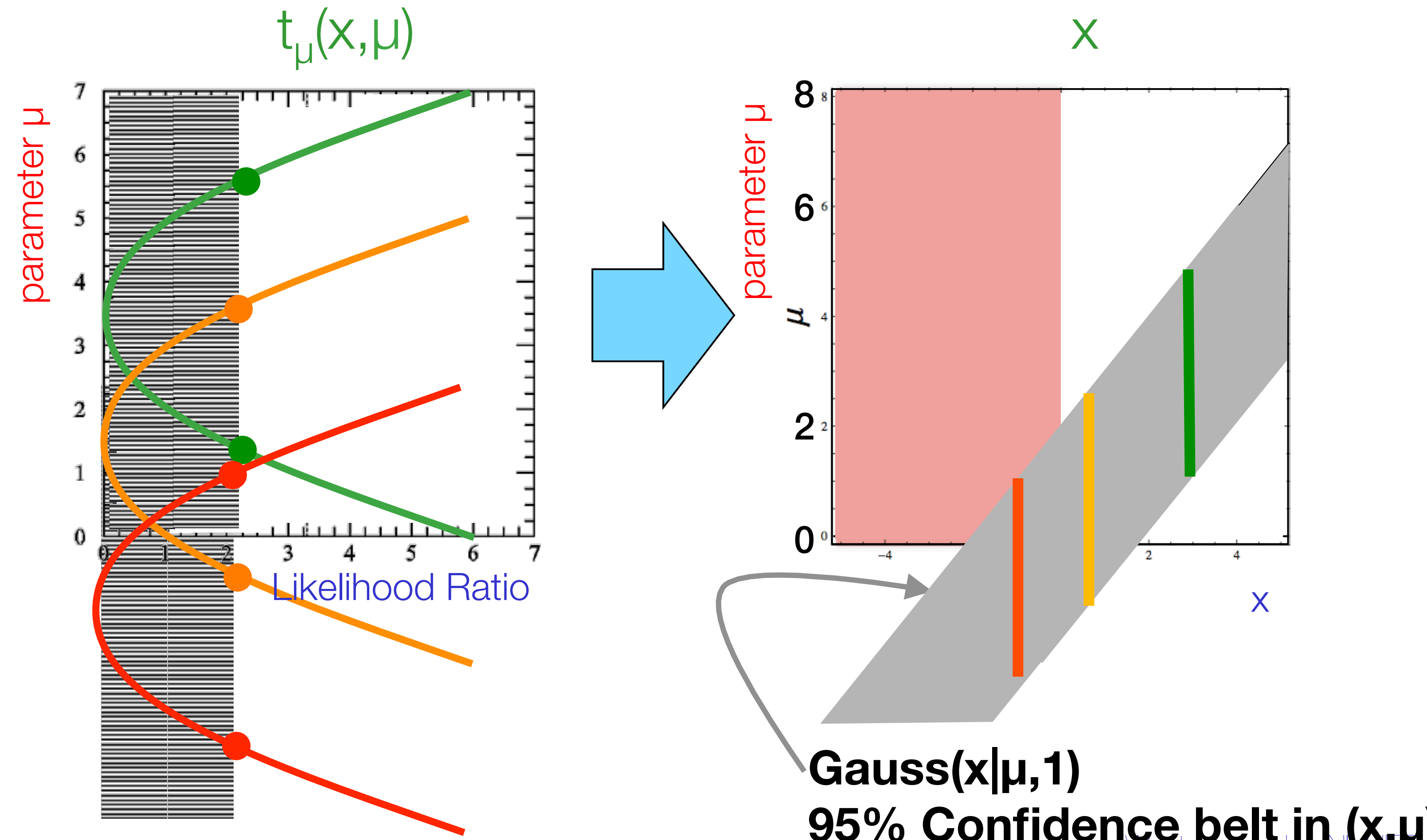
- Putting everything together – the confidence with modified t_μ
- Confidence belt 'pinches' towards physical boundary
- Offsetting of likelihood curves for measurements that prefer $\mu<0$



Physical boundaries frequentist confidence intervals

- Example for *unconstrained* unit Gaussian measurement

$$L = Gauss(x \mid \mu, 1)$$

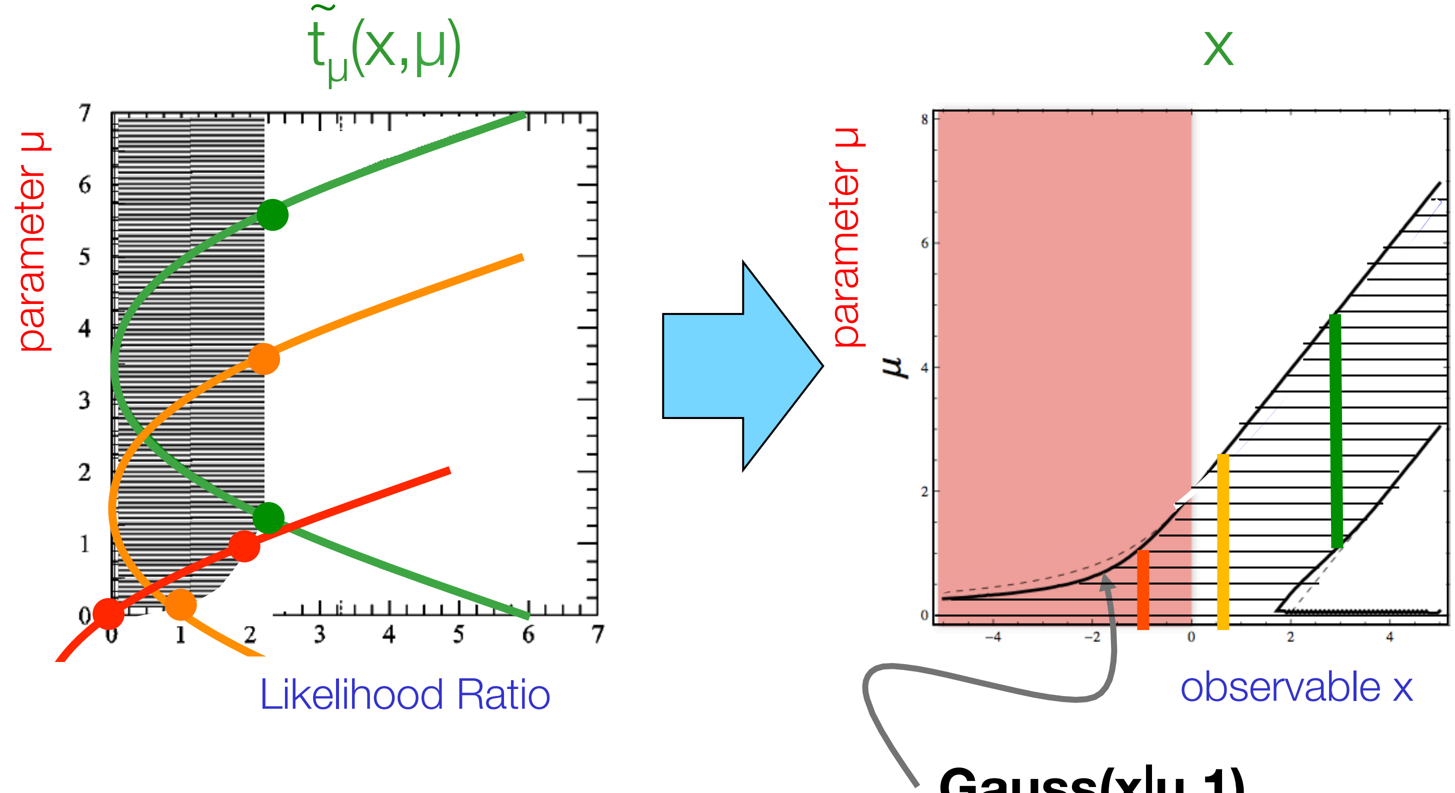


Gauss(x|μ,1)
95% Confidence belt in (x,μ)
defined by cut on t_μ

Physical boundaries frequentist confidence intervals

- First map back horizontal axis of confidence belt from $t_\mu(x) \rightarrow x$

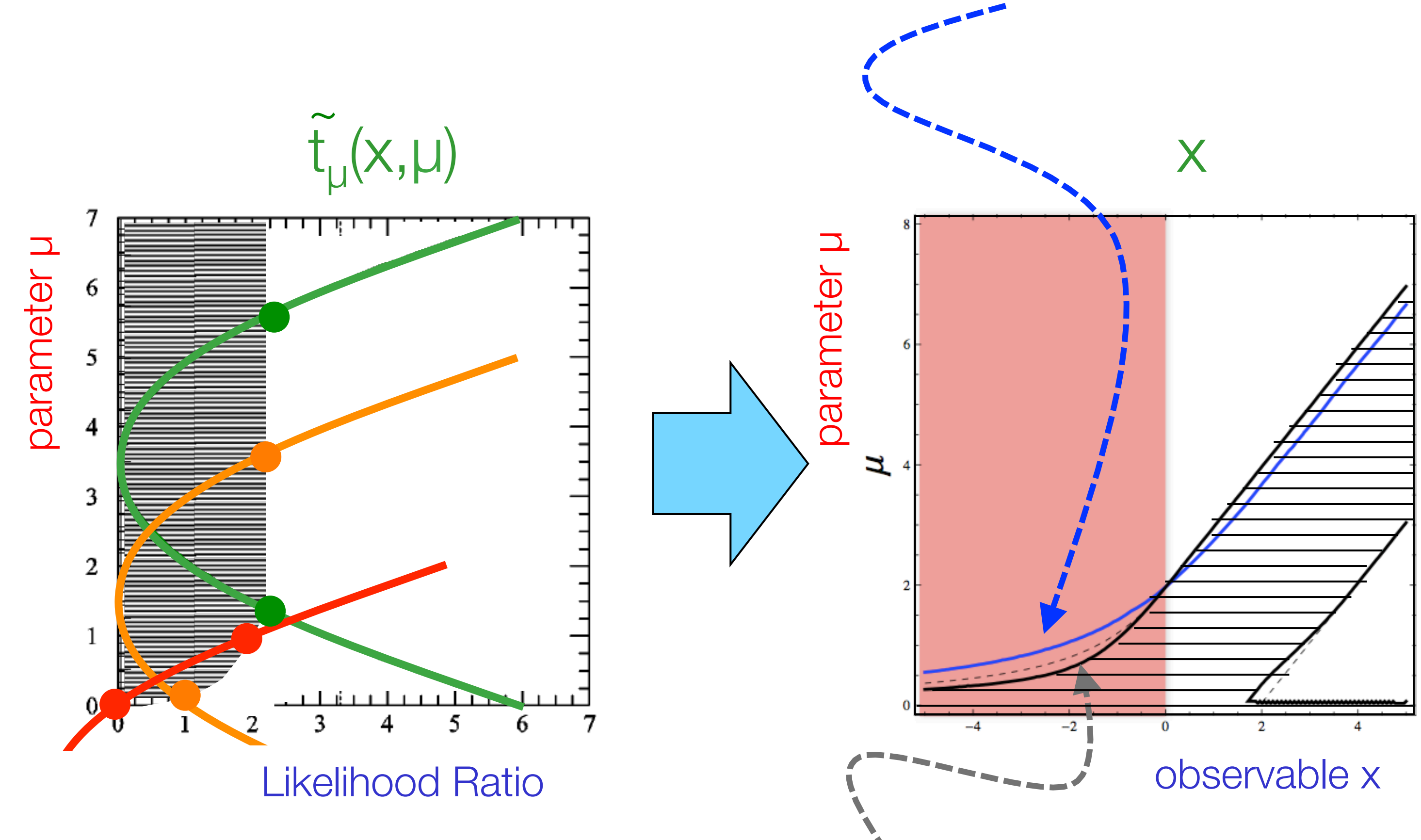
"Feldman-Cousins"



Gauss(x|μ,1)
95% Confidence belt in (x,μ)
defined by cut on $\tilde{t}_\mu$

Comparison of Bayesian and Frequentist limit treatment

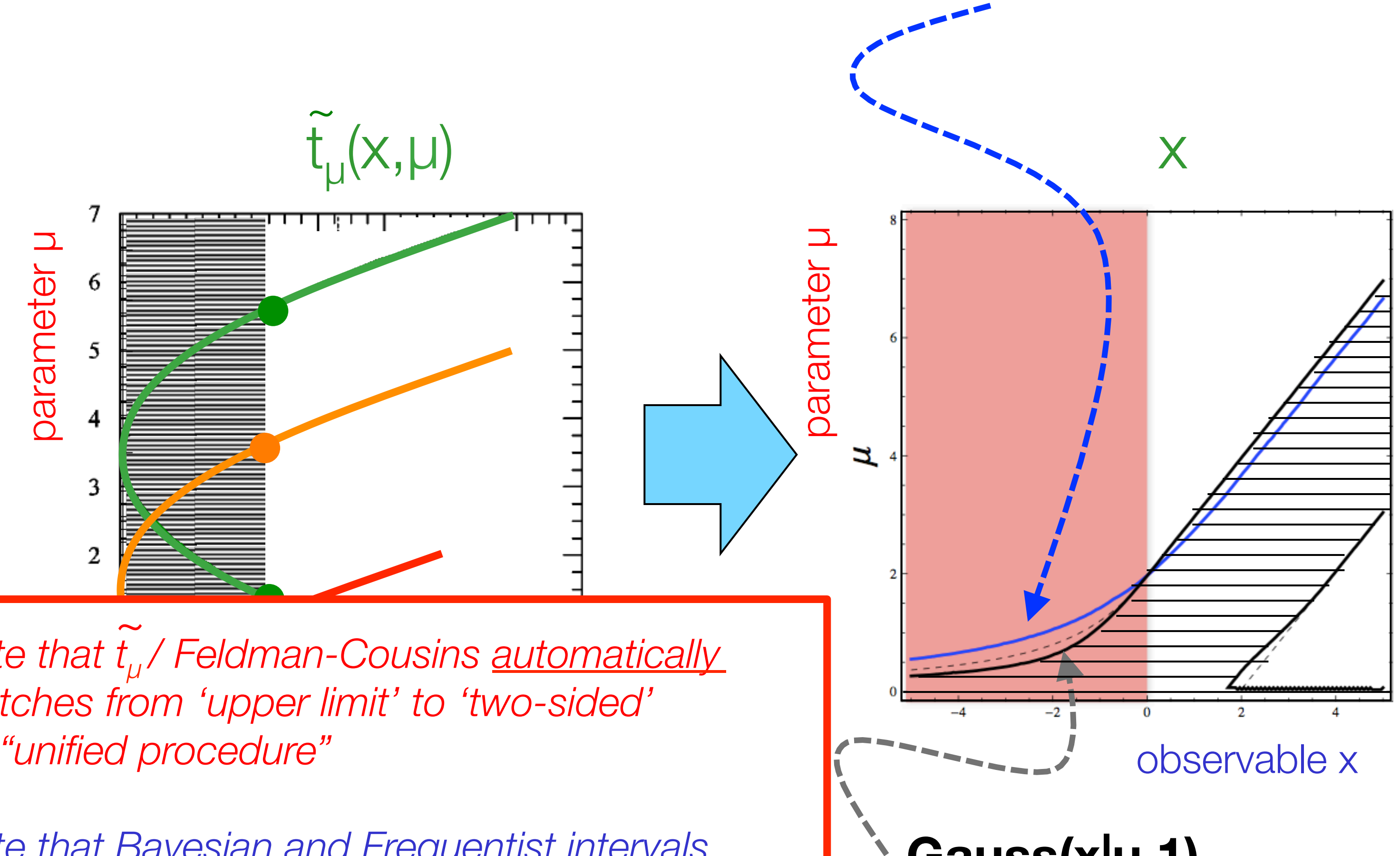
- Bayesian 95% credible upper-limit interval with flat prior μ>0



Gauss(x|μ,1)
95% Confidence belt in (x,μ)
defined by cut on t_μ for

Comparison of Bayesian and Frequentist limit treatment

- Bayesian 95% credible upper-limit interval with flat prior μ>0



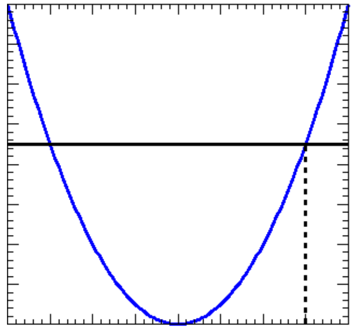
Note that $\tilde{t}_{\mu}$ / Feldman-Cousins <u>automatically</u> switches from 'upper limit' to 'two-sided'
→ "unified procedure"

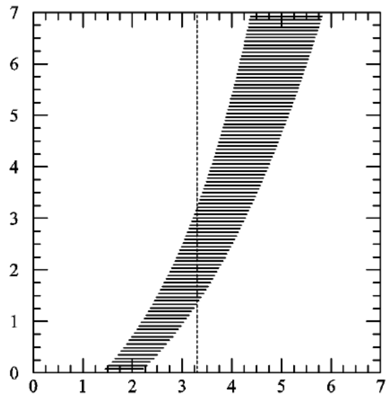
Note that Bayesian and Frequentist intervals at x>2 would agree exactly for Gaussian example if both would be taken as 'two-sided'

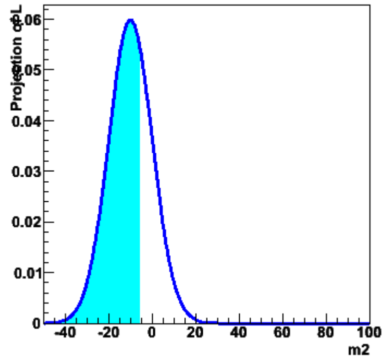
Gauss(x|μ,1)
95% Confidence belt in (x,μ) defined by cut on t_{μ} for

Summary

- Maximum Likelihood
  - Point and variance estimation
  - Variance estimate assumes normal distribution. No upper/lower limits
- Frequentist confidence intervals
  - Extend hypothesis testing to composite hypothesis
  - Neyman construction provides exact “coverage” = calibration of quoted probabilities
  - Strictly p(data|theory)
  - Asymptotically identical to likelihood ratio intervals (MINOS errors, *does not assume parabolic L*)
- Bayesian credible intervals
  - Extend P(theo) to p.d.f. in model parameters
  - Integrals over posterior density → credible intervals
  - Always involves prior density function in parameter space







Next subject…

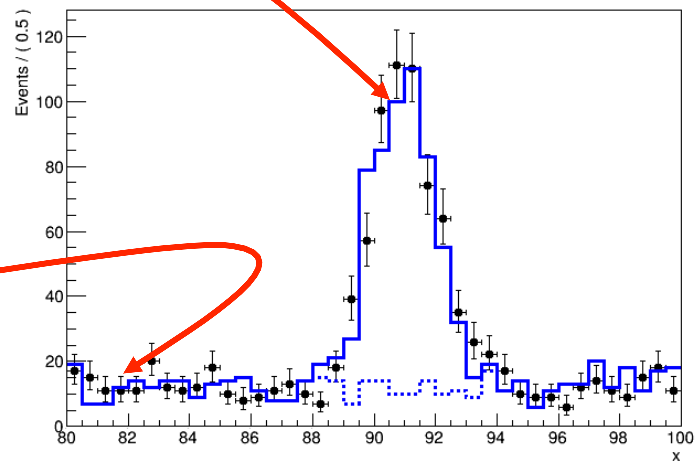
- Start with basics, gradually build up to complexity of

Statistical tests with simple hypotheses for counting data	*"What do we mean with probabilities"*
↓	
Statistical tests with simple hypotheses for distributions	*"p-values"*
↓	
Hypothesis testing as basis for event selection	*"Optimal event selection & machine learning"*
↓	
Composite hypotheses (with parameters) for distributions	*"Confidence intervals, Maximum Likelihood"*
↓	
Statistical inference with nuisance parameters	*"Fitting the background"*
↓	
Response functions and subsidiary measurements	*"Sideband fits and systematic uncertainties"*

So far we've only considered the *ideal* experiment

- The "only thing" you need to do (as an experimental physicist) is to formulate the likelihood function for your measurement
- For an ideal experiment, where signal and background are assumed to have perfectly known properties, this is trivial

$$L(\vec{N} \mid \mu) = \prod_{bins} Poisson(N_i \mid \mu \cdot \tilde{s}_i + \tilde{b}_i)$$

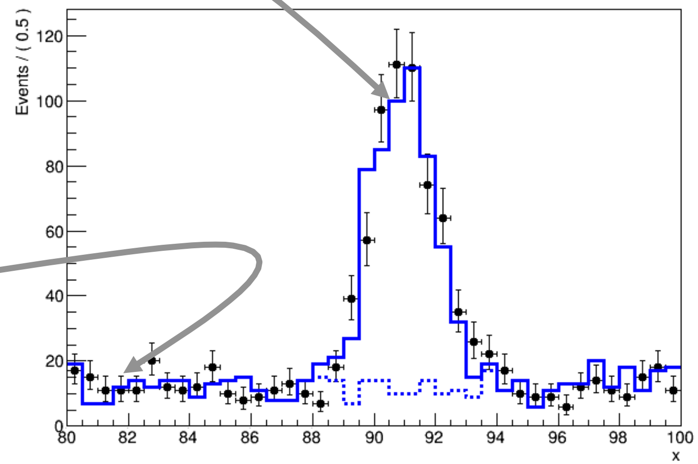


- So far only considered a single parameter in the likelihood: the physics *parameter of interest,* usually denoted as μ

The imperfect experiment

- In realistic measurements many effect that we don't control exactly influence measurements of parameter of interest
- How do you model these uncertainties in the likelihood?

$$L(\vec{N} \mid \mu) = \prod_{bins} Poisson(N_i \mid \mu \cdot \tilde{s}_i + \tilde{b}_i)$$

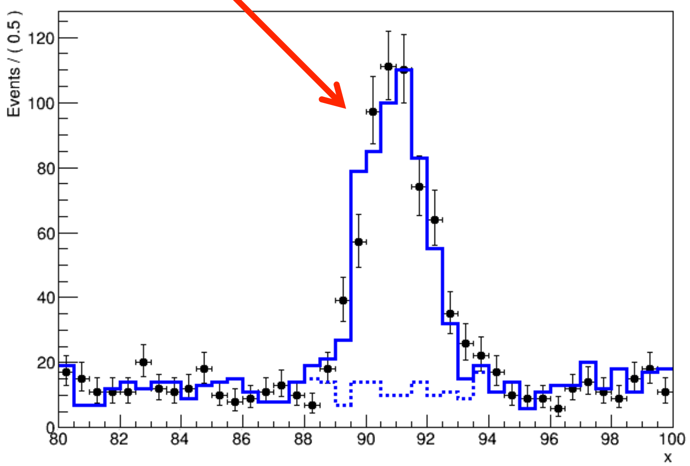


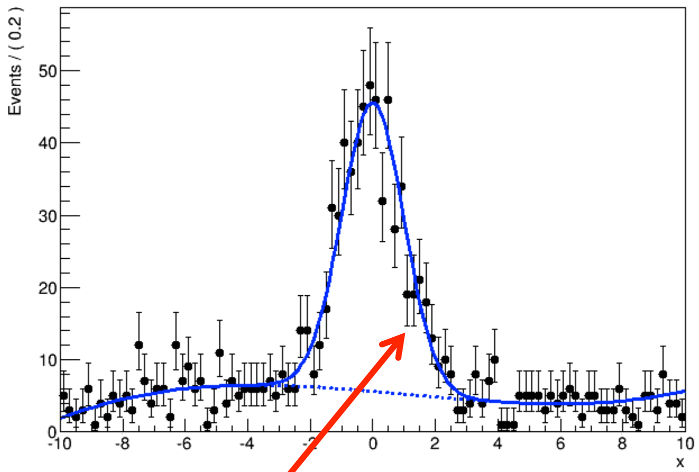
Signal and background predictions are affected by (systematic) uncertainties

Adding parameters to the model

- We can describe uncertainties in our model by adding new parameters of which the value is uncertain

$$L(\vec{N}\,|\,\mu) = \prod_{bins} Poisson(N_i\,|\,\mu\cdot\tilde{s}_i + \tilde{b}_i)$$



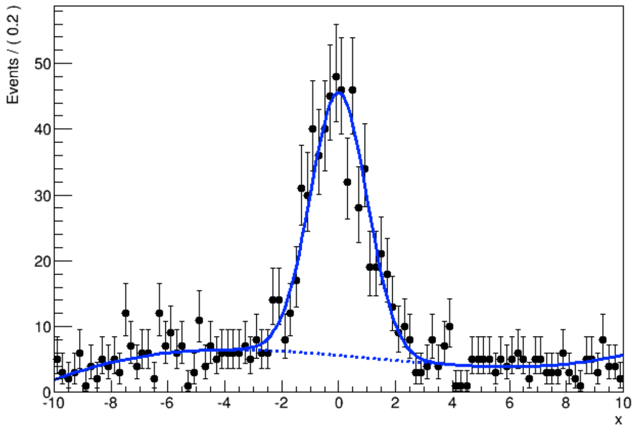


$$L(x\,|\,f,m,\sigma,a_0,a_1,a_2) = fG(x,m,\sigma) + (1-f)Poly(x,a_0,a_1,a_2)$$

- These additional model parameters are not 'of interest', but we need them to model uncertainties → 'Nuisance parameters'

What are the nuisance parameters of your *physics model?*

- *Empirical* modeling of uncertainties, e.g. polynomial for background, Gaussian for signal, is easy to do, but may lead to hard questions

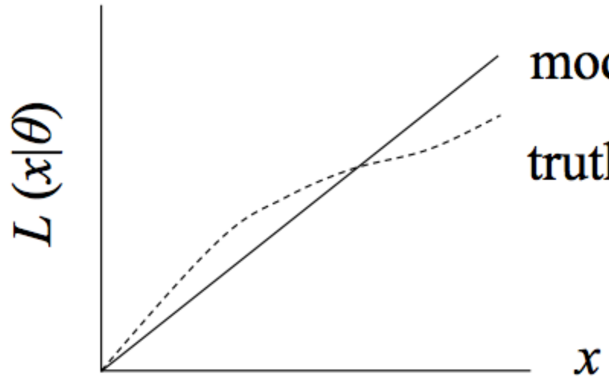


$$L(x \mid f, m, \sigma, a_0, a_1, a_2) = fG(x, m, \sigma) + (1 - f)Poly(x, a_0, a_1, a_2)$$

- Is your model correct? (Is true signal distr. captured by a Gaussian?)
- Is your model flexible enough? (4th order polynomial, or better 6th)?
- How do model parameters connect to known detector/theory uncertainties in your distribution?
  - what conceptual uncertainty do your parameters represent?

The statisticians view on nuisance parameters

- In general, our model of the data is not perfect



model: $L(x|\theta) = \theta x$

truth: $L(x|\theta) = \theta x + \alpha x^2 + \beta x^3 + \cdots$

- Can improve modeling by including additional adjustable parameters
- Goal: some point in the parameter space of the enlarged model should be "true"
- Presence of nuisance parameters decreases the sensitivity of the analysis of the parameter(s) of interest

Treatment of nuisance parameters in parameter estimation

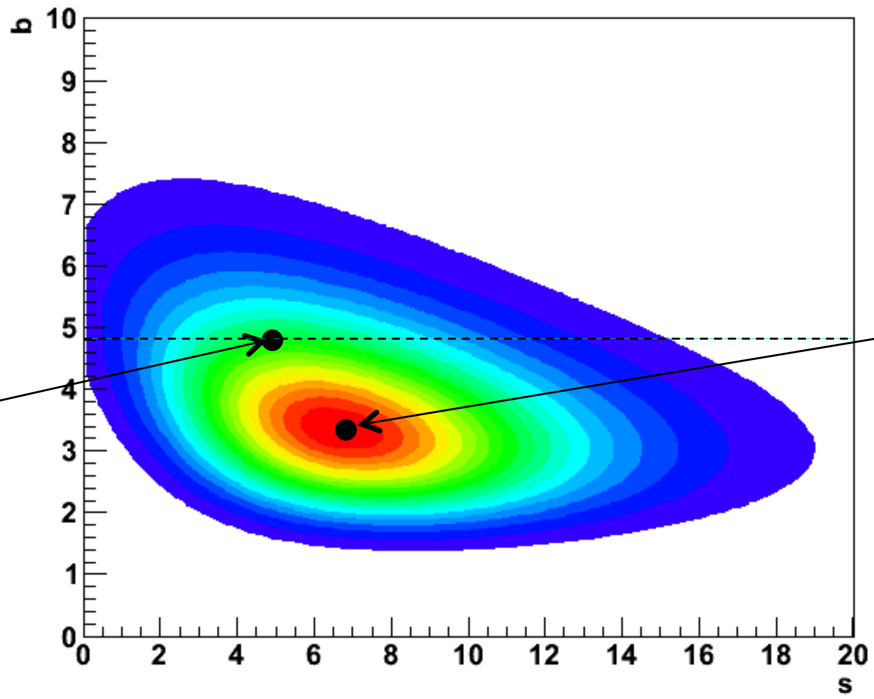
- In POI parameter estimation, the effect of NPs incorporated through *unconditional minimization*
  - I.e. minimize Likelihood w.r.t all parameter simultaneously.
- Simple example with 2-bin Poisson counting experiment

$$L(s) = Poisson(10 \mid s+5)$$

$$L(s,b) = Poisson(10 \mid s+b)Poisson(10 \mid 3\cdot b)$$

Conditional minimum in s (condition: b=5)

$$\hat{\hat{s}}\Big|_{b=5}$$



Unconditional minimum in s,b

$$(\hat{s},\hat{b})$$

Wouter Verkerke, NIKHEF

Treatment of nuisance parameters in variance estimation

- Maximum likelihood estimator of parameter variance is based on 2nd derivative of Likelihood
  - For multi-parameter problems this 2nd derivative is generalized by the **Hessian Matrix** of partial second derivatives

$$\hat{\sigma}(p)^2 = \hat{V}(p) = \left(\frac{d^2 \ln L}{d^2 p}\right)^{-1} \quad \Rightarrow \quad \hat{\sigma}(p_i)^2 = \hat{V}(p_{ii}) = \left(H^{-1}\right)_{ii}$$

$$H(f) = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1\,\partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_1\,\partial x_n} \\ \frac{\partial^2 f}{\partial x_2\,\partial x_1} & \frac{\partial^2 f}{\partial x_2^2} & \cdots & \frac{\partial^2 f}{\partial x_2\,\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_n\,\partial x_1} & \frac{\partial^2 f}{\partial x_n\,\partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_n^2} \end{bmatrix}.$$

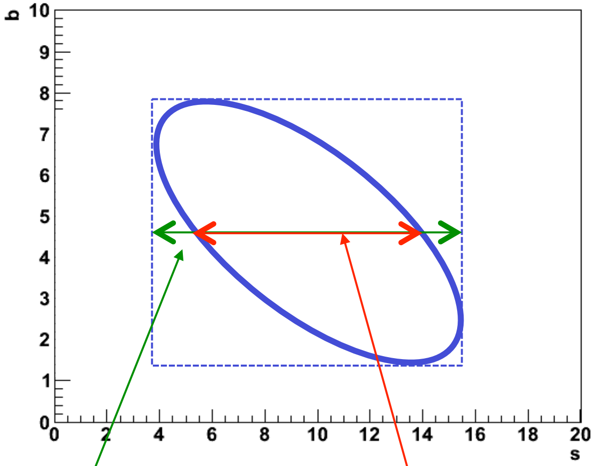
- For multi-parameter likelihoods estimate of covariance V_{ij} of pair of 2 parameters in addition to variance of individual parameters
  - Usually re-expressed in terms dimensionless correlation coefficients ρ

$$V_{ij} = \rho_{ij}\sqrt{V_{ii}V_{jj}}$$

Treatment of nuisance parameters in variance estimation

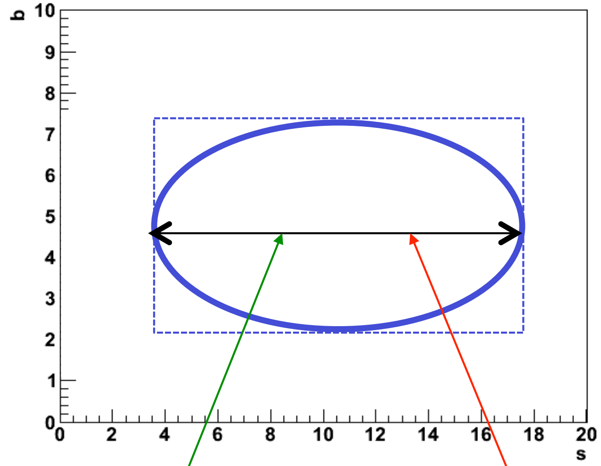
- Effect of NPs on variance estimates visualized

Scenario 1
Estimators of
POI and NP correlated
i.e. ρ(s,b)≠0



$$\hat{V}(s)\ from \begin{bmatrix} \frac{\partial^2 L}{\partial s^2} & \frac{\partial^2 L}{\partial s \partial b} \\ \frac{\partial^2 L}{\partial s \partial b} & \frac{\partial^2 L}{\partial b^2} \end{bmatrix}^{-1} \qquad \hat{V}(s)\ from \left[\frac{\partial^2 L}{\partial s^2}\right]^{-1}_{b=\hat{b}}$$

Scenario 2
Estimators of
POI and NP correlated
i.e. ρ(s,b)=0



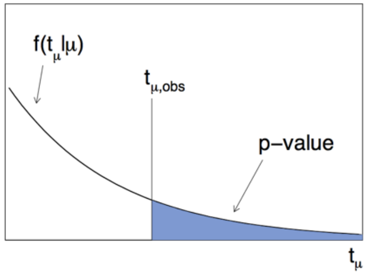
$$\hat{V}(s)\ from \begin{bmatrix} \frac{\partial^2 L}{\partial s^2} & \frac{\partial^2 L}{\partial s \partial b} \\ \frac{\partial^2 L}{\partial s \partial b} & \frac{\partial^2 L}{\partial b^2} \end{bmatrix}^{-1} \qquad \hat{V}(s)\ from \left[\frac{\partial^2 L}{\partial s^2}\right]^{-1}_{b=\hat{b}}$$

Uncertainty on background increases uncertainty on signal

Treatment of NPs in hypothesis testing and conf. intervals

- We've covered frequentist hypothesis testing and interval calculation using likelihood ratios based on a likelihood with a single parameter (of interest) L(μ)
  - Result is p-value on hypothesis with given μ value, or
  - Result is a confidence interval $[\mu_-,\mu_+]$ with values of μ for which p-value is at or above a certain level (the confidence level)
- How do you do this with a likelihood L(μ,θ) where θ is a nuisance parameter?
  - With a test statistics q_μ, we calculate p-value for hypothesis θ as

$$p_\mu = \int_{q_{\mu,obs}}^{\infty} f(q_\mu \mid \mu,\theta)\, dq_\mu$$



- But what values of θ do we use for $f(q_\mu|\mu,\theta)$?
  Fundamentally, we want to reject μ only if p<α for all θ
  → Exact confidence interval

Hypothesis testing & conf. intervals with nuisance parameters

- The goal is that the parameter of interest should be covered at the stated confidence for every value of the nuisance parameter
- if there is *any value* of the nuisance parameter which makes the data consistent with the parameter of interest, that value of the POI should be considered:
  - e.g. don't claim discovery if any background scenario is compatible with data
- But: technically very challenging and significant problems with over-coverage
  - Example: how broadly should 'any background scenario' be defined? Should we include background scenarios that are clearly incompatible with the observed data?

The profile likelihood construction as compromise

- For LHC the following prescription is used:

  NPs

  Given L(μ,θ)

  POI

  perform hypothesis test for each value of μ (the POI),

  using values of nuisance parameter(s) θ that best fit the data under the hypothesis μ

- Introduce the following notation

$$\hat{\hat{\theta}}(\mu)$$

M.L. estimate of θ for a given value of μ
(i.e. a conditional ML estimate)

- The resulting confidence interval will have exact coverage for the points $(\mu, \hat{\hat{\theta}}(\mu))$
  - Elsewhere it may overcover or undercover (but this can be checked)

The profile likelihood ratio

- With this prescription we can construct the profile likelihood ratio as test statistic

Likelihood for given μ

Maximum Likelihood for given μ

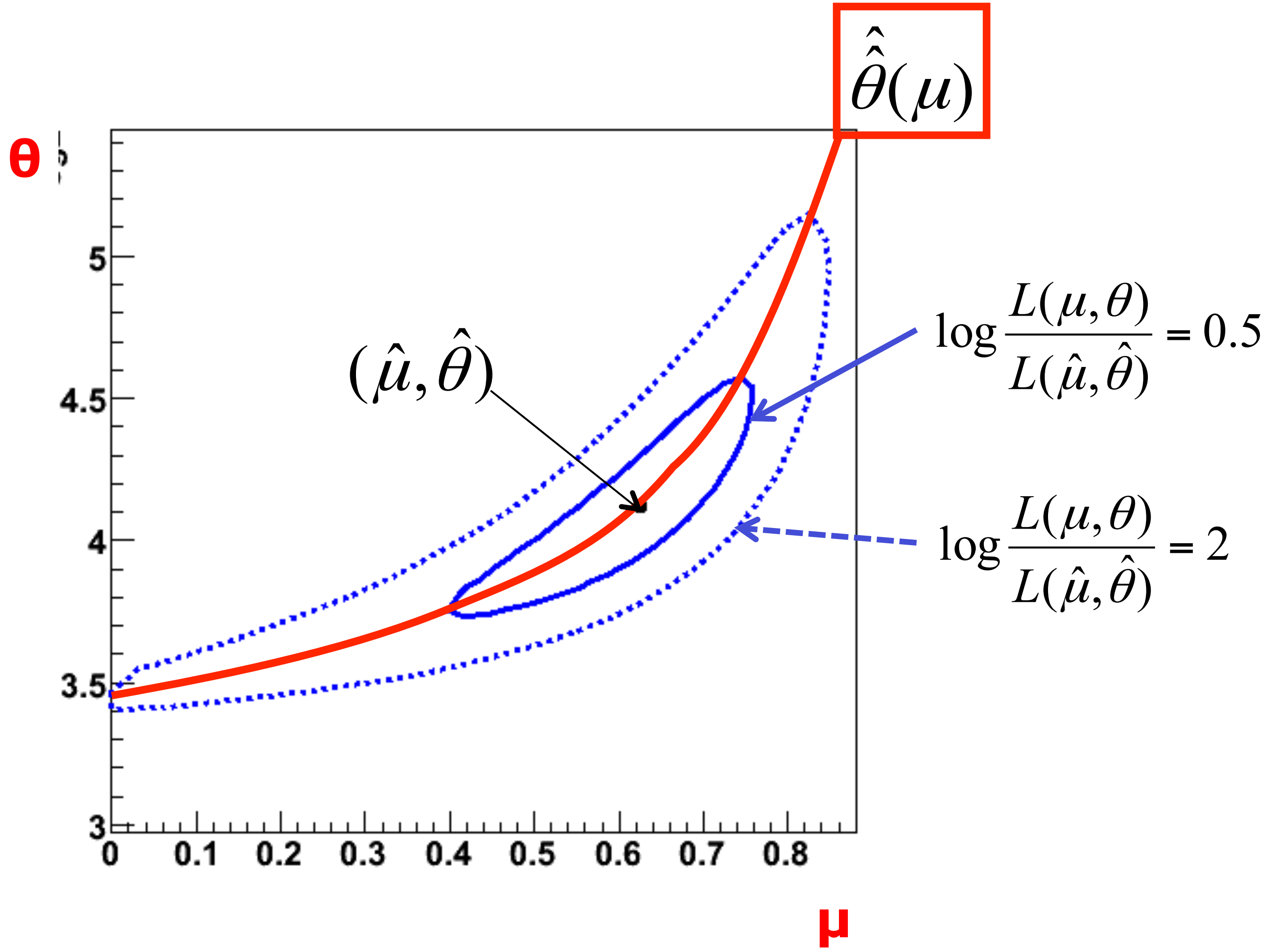
$$\lambda(\mu) = \frac{L(\mu)}{L(\hat{\mu})} \Rightarrow \lambda(\mu) = \frac{L(\mu, \hat{\hat{\theta}}(\mu))}{L(\hat{\mu}, \hat{\theta})}$$

Maximum Likelihood

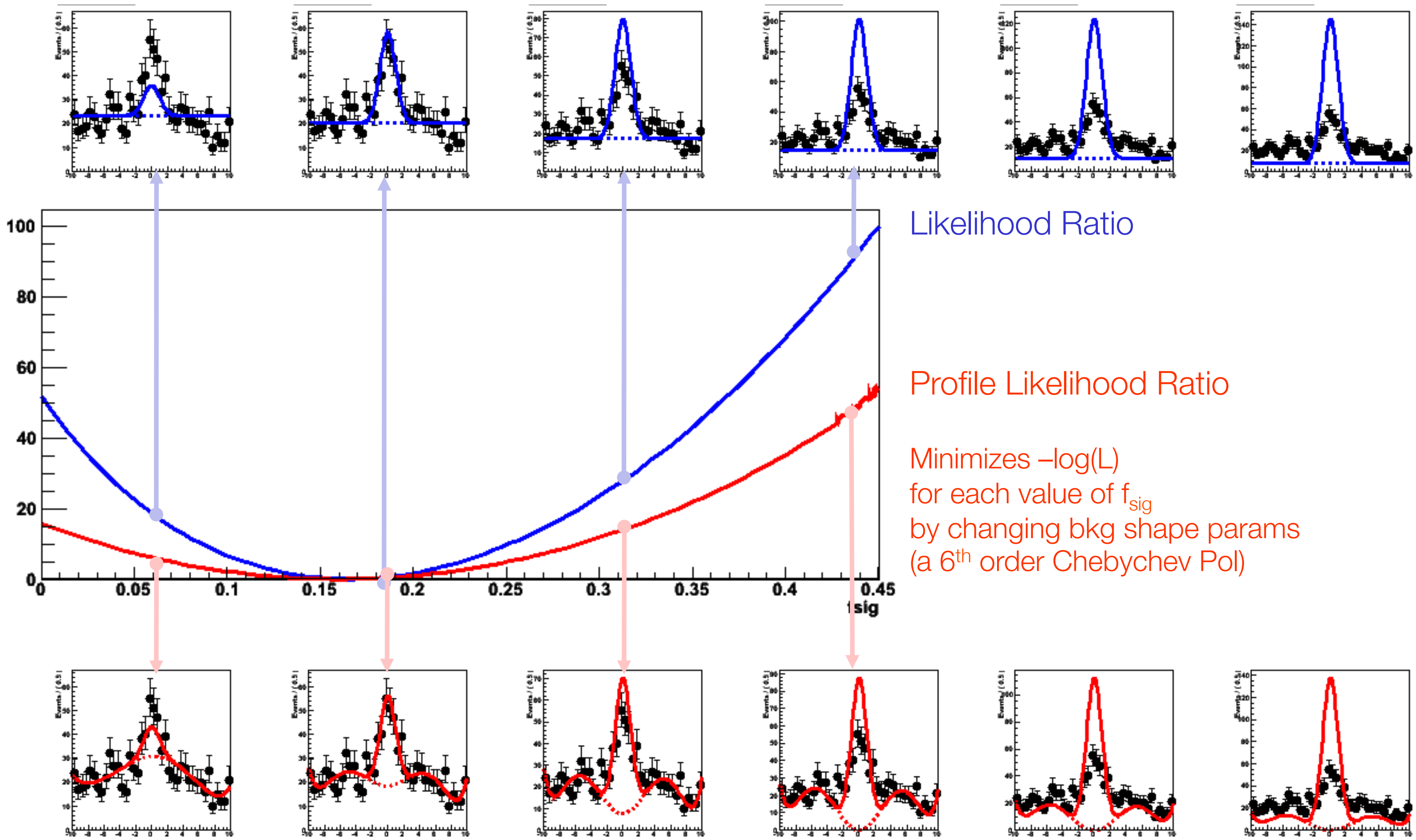
Maximum Likelihood

- NB: value profile likelihood ratio does *not* depend on θ

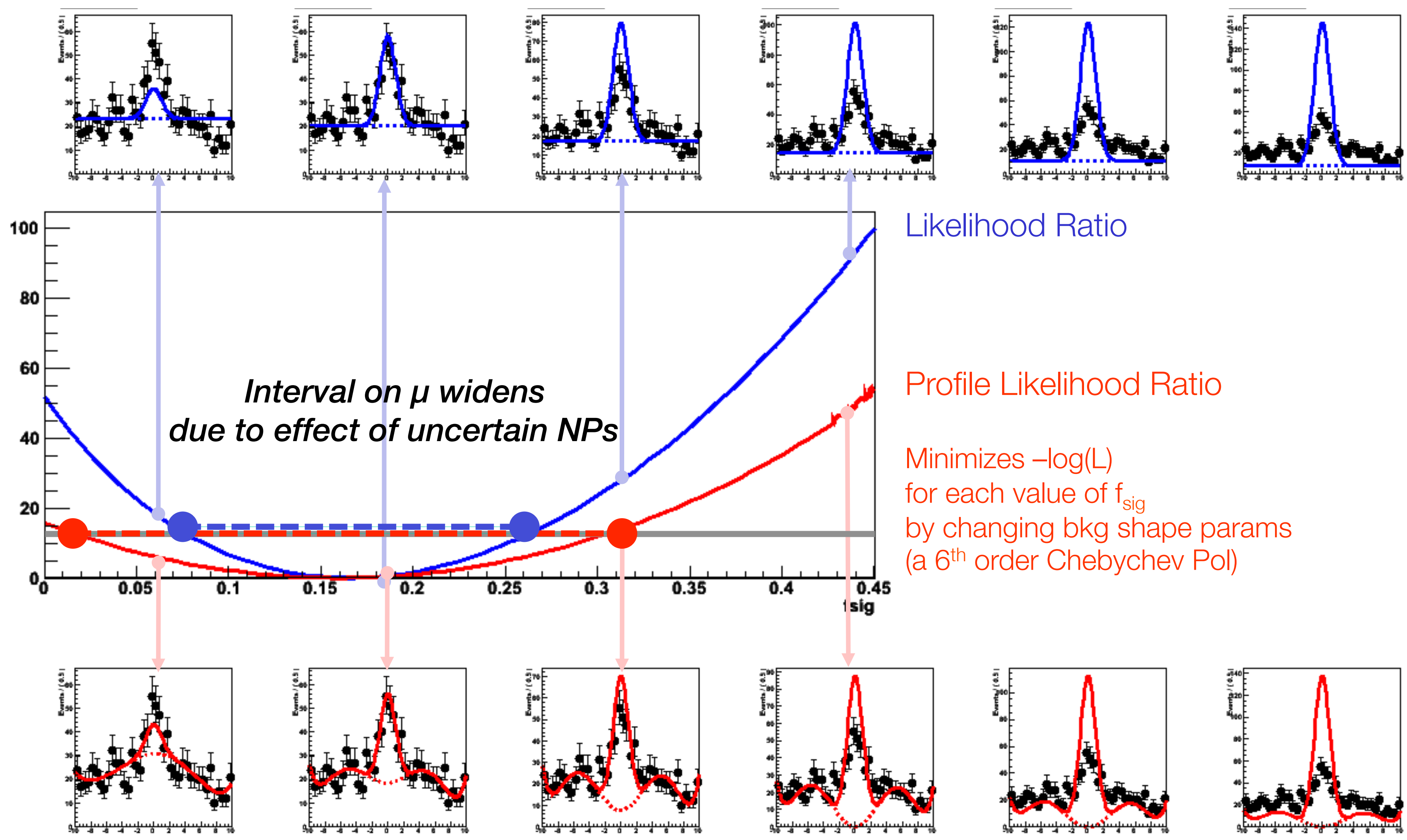
Profiling illustration with one nuisance parameter



Profile scan of a Gaussian plus Polynomial probability model



Profile scan of a Gaussian plus Polynomial probability model

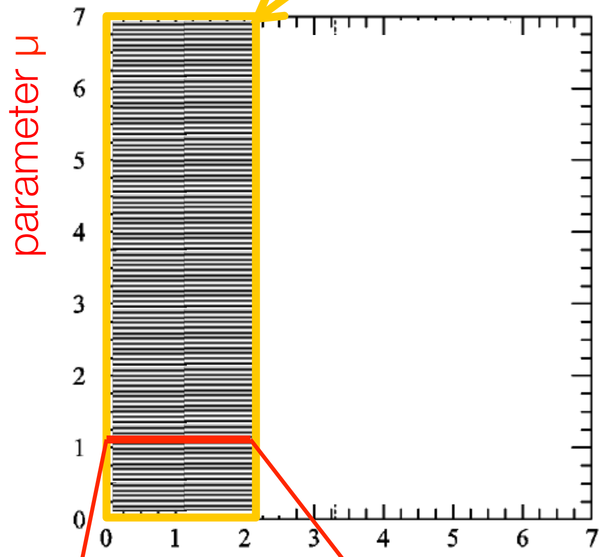


Minimizes –log(L)
for each value of f_{sig}
by changing bkg shape params
(a 6th order Chebychev Pol)

PLR Confidence interval vs MINOS

Confidence belt now range in PLR

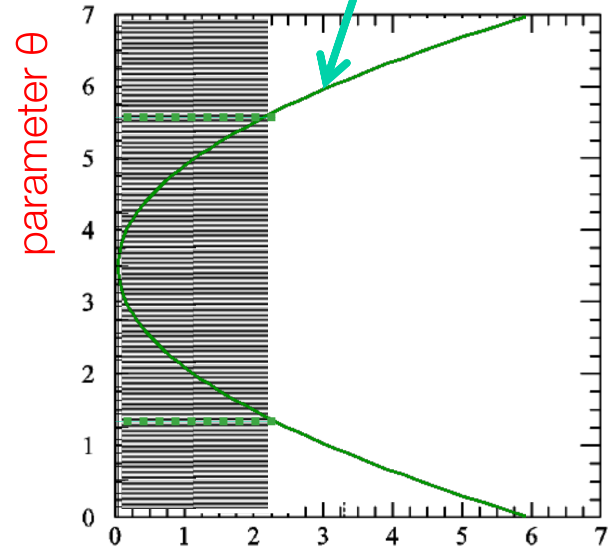
$t_\mu(x,\mu)$



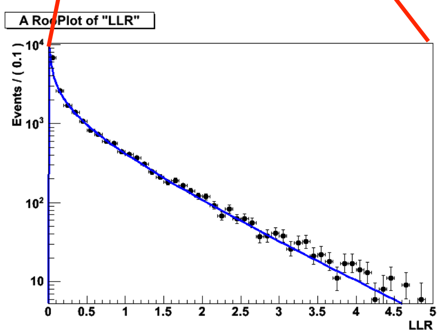
Profile Likelihood Ratio

Measurement = $t_\mu(x_{obs},\mu)$ is now a function of μ

$t_\mu(x,\mu)$



Profile Likelihood Ratio



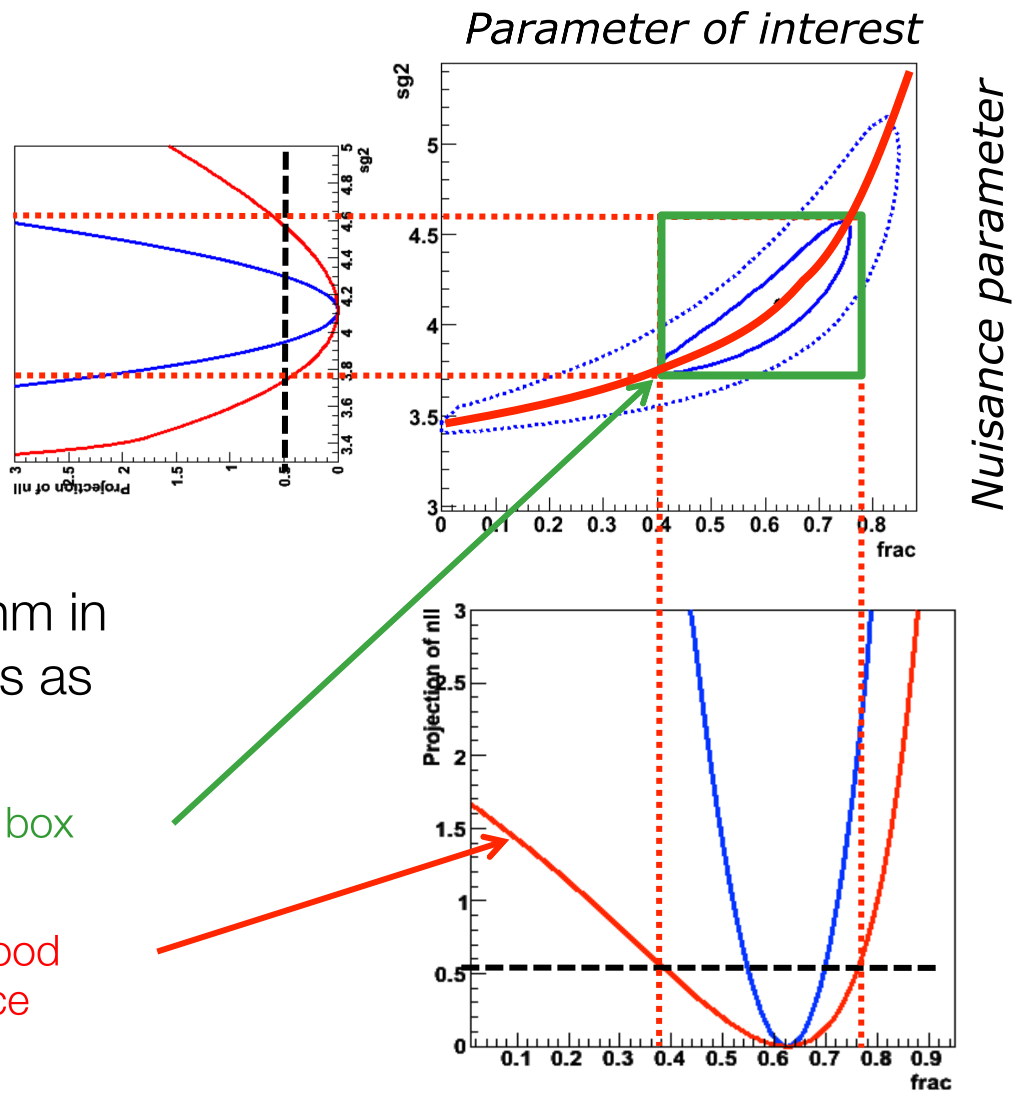
Asymptotically, distribution is identical for all μ

NB: asymptotically, distribution is also independent of true values of θ

$$f(t_\mu;\Lambda) = \frac{1}{2\sqrt{t_\mu}}\frac{1}{\sqrt{2\pi}}\left[\exp\left(-\frac{1}{2}\left(\sqrt{t_\mu}+\sqrt{\Lambda}\right)^2\right)+\exp\left(-\frac{1}{2}\left(\sqrt{t_\mu}-\sqrt{\Lambda}\right)^2\right)\right]$$

$$\Lambda = \frac{(\mu-\mu')^2}{\sigma^2}\ .$$

Link between MINOS errors and profile likelihood



- Note that MINOS algorithm in MINUIT gives same errors as Profile Likelihood Ratio
  - MINOS errors is bounding box around λ(s) contour
  - Profile Likelihood = Likelihood minimized w.r.t. all nuisance parameters

NB: Similar to graphical interpretation of variance estimators, but those always assume an elliptical contour from a perfectly parabolic likelihood

Summary on NPs in confidence intervals

- Exact confidence intervals are difficult with nuisance parameters
  - Interval should cover for any value of nuisance parameters
  - Technically difficult and significant over-coverage common
- LHC solution Profile Likelihood ratio → Guaranteed coverage at *measured* values of nuisance parameters only
  - Technically replace likelihood ratio with profile likelihood ratio
  - Computationally more intensive (need to minimize likelihood w.r.t all nuisance parameters for each evaluation of the test statistic), but still very tractable
- Asymptotically confidence intervals constructed with profile likelihood ratio test statistics correspond to (MINOS) likelihood ratio intervals
  - As distribution of profile likelihood becomes asymptotically independent of θ, coverage for all values of θ restored

Dealing with nuisance parameters in Bayesian intervals

- Elimination of nuisance parameters in Bayesian interval: Integrate over the full subspace of all nuisance parameters;

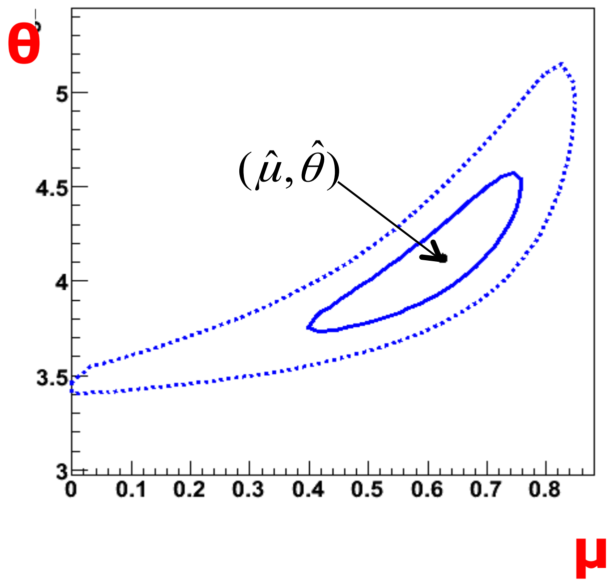
$$P(\mu \mid x) \propto L(x \mid \mu) \cdot \pi(\mu)$$

$$\Downarrow$$

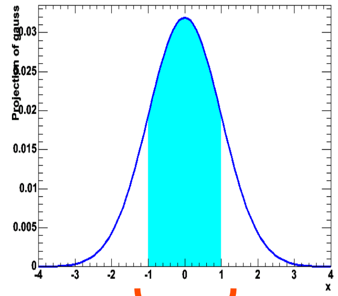
$$P(\mu \mid x) \propto \int \Big(L(x \mid \mu, \vec{\theta}) \pi(\mu) \pi(\vec{\theta}) \Big) d\vec{\theta}$$

- You are left with posterior pdf for μ

$\int$



$\times\ \pi(\mu, \theta)\ =$



Credible interval:
area that integrates
X% of posterior

Computational aspects of dealing with nuisance parameters

- Dealing with many nuisance parameters is computationally intensive in both Bayesian and (LHC) Frequentist approach
- Profile Likelihood approach
  - Computational challenge = ***Minimization*** *of likelihood w.r.t. all nuisance parameters for every point in the profile likelihood curve*
  - Minimization can be a difficult problem,
    e.g. if there are strong correlations, or multiple minima
- Bayesian approach
  - Computational challenge = ***Integration*** *of posterior density of all nuisance parameters*
  - Requires sampling of very potentially very large space.
  - Markov Chain MC and importance sampling techniques can help, but still very CPU consuming

Other procedures that have been tried*

- Hybrid Frequentist-Bayesian approach ('Cousins-Highland / Z_N')
  - Integrate likelihood over nuisance parameters

$$L_m(\mu) = \int \left(L(\mu, \vec{\theta}) \pi(\vec{\theta}) \right) d\vec{\theta}$$

  - Then treat integrated L_m as test statistic → obtain p-value from its distribution
  - In practice integral is performed using MC integration, so often described as a 'sampling method'

$$L_m(\mu) = \frac{1}{N} \sum_{MC} L(\mu, \vec{\theta}_i) \pi(\vec{\theta}_i)$$

  - Method has been shown to have bad coverage
- Ad-hoc sampling methods of various types.
  - Usually amount to either MC integration or fancy error propagation

Note that sampling the conditional estimator $\left.\hat{\hat{\mu}}\right|_{\theta}$ over sample of θ values obtained from π(θ) is just glorified error propagation!

* But are known to have problems

How much do answers differ between methods?

A Prototype Problem

BROOKHAVEN NATIONAL LABORATORY

What is significance Z of an observation x =*178* events in a signal like region, if my expected background b =*100* with a 10% uncertainty?

- if you use the ATLAS TDR formula $Z_{5'}$=5.5
- if you use Cousins-Highland Z_N=5.0

The question seems simple enough, but it is not actually well-posed

- what do I mean by 10% background uncertainty?

Typically, we consider an auxiliary measurement y used to estimate background (Type I systematic)

- eg: a sideband counting experiment where background in sideband is a factor τ bigger than in signal region

$$L_P(x, y|\mu, b) = Pois(x|\mu + b) \cdot Pois(y|\tau b).$$

These slide discuss a 'prototype' likelihood that statisticians like:

Poisson(N_{sig}|s+b) · Poisson(N_{ctl}|τ·b)

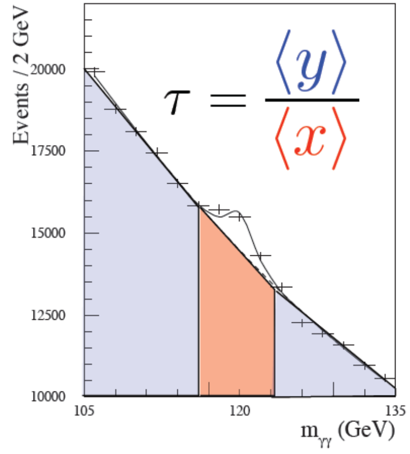
NB: This is one of the very few problems with nuisance parameters with can be exactly calculation

Example Sideband Measurement

BROOKHAVEN NATIONAL LABORATORY

Sideband measurement used to extrapolate / interpolate the background rate in signal-like region

For now ignore uncertainty in extrapolation.



$$L_P(x, y|\mu, b) = Pois(x|\mu + b) \cdot Pois(y|\tau b).$$

Recent comparisons results from PhyStat 2007

Comparison of Methods for Prototype Problem

BROOKHAVEN NATIONAL LABORATORY

In my contribution to PhyStat2005, I considered this problem and compared the coverage for several methods

- See Linnemann's PhyStat03 paper

Major results:

- Cousins-Highland result (Z_N) badly under-covers (only 4.2σ)!
  - rate of Type I error is 110 times higher than stated!
  - much less luminosity required
- Profile Likelihood Ratio (MINUIT/MINOS) works great out to 5σ!

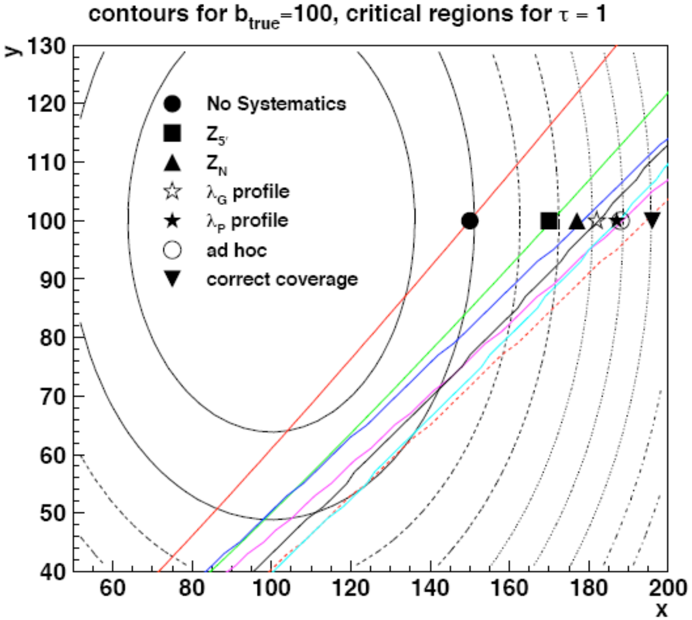


Figure 7. A comparison of the various methods critical boundary $x_{crit}(y)$ (see text). The concentric ovals represent contours of L_G from Eq. 15.

Method	L_G $(Z\sigma)$	L_P $(Z\sigma)$	$x_{crit}(y=100)$
No Syst	3.0	3.1	150
$Z_{5'}$	4.1	4.1	171
Z_N (Sec. 4.1)	4.2	4.2	178
ad hoc	4.6	4.7	188
$Z_\Gamma = Z_{Bi}$	4.9	5.0	185
profile λ_P	5.0	5.0	185
profile λ_G	4.7	4.7	~182

Exact solution

Summary of statistical treatment of nuisance parameters

- Each statistical method has an associated technique to propagate the effect of uncertain NPs on the estimate of the POI
  - Parameter estimation → Joint unconditional estimation
  - Variance estimation → Replace d^2L/dp^2 with Hessian matrix
  - Hypothesis tests & confidence intervals → Use profile likelihood ratio
  - Bayesian credible intervals → Integration ('Marginalization')

- Be sure to use the right procedure with the right method
  - Anytime you integrate a Likelihood you are a Bayesian
  - If you are minimizing the likelihood you are usually a Frequentist
  - If you sample something chances are you performing either a (Bayesian) Monte Carlo integral, or are doing glorified error propagation
- Answers can differ substantially between methods!
  - This is not always a problem, but can also be a consequence of a difference in the problem statement

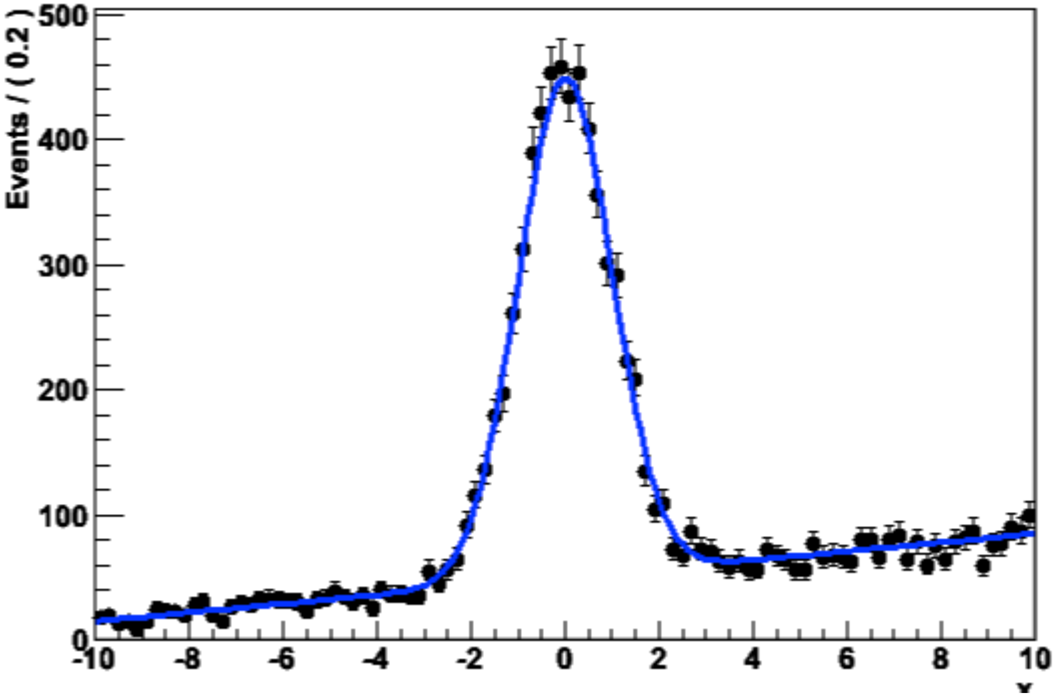
Overview

- Start with basics, gradually build up to complexity of

Statistical tests with simple hypotheses for counting data	*"What do we mean with probabilities"*
Statistical tests with simple hypotheses for distributions	*"p-values"*
Hypothesis testing as basis for event selection	*"Optimal event selection & machine learning"*
Composite hypotheses (with parameters) for distributions	*"Confidence intervals, Maximum Likelihood"*
Statistical inference with nuisance parameters	*"Fitting the background"*
Response functions and subsidiary measurements	*"Profile Likelihood fits"*

RooFit – Focus: coding likelihood functions

- Focus on one practical aspect of many data analysis in HEP: How do you formulate your likelihood functions in ROOT
  - For 'simple' problems (gauss, polynomial) this is easy



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

RooFit core design philosophy

- Mathematical objects are represented as C++ objects

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

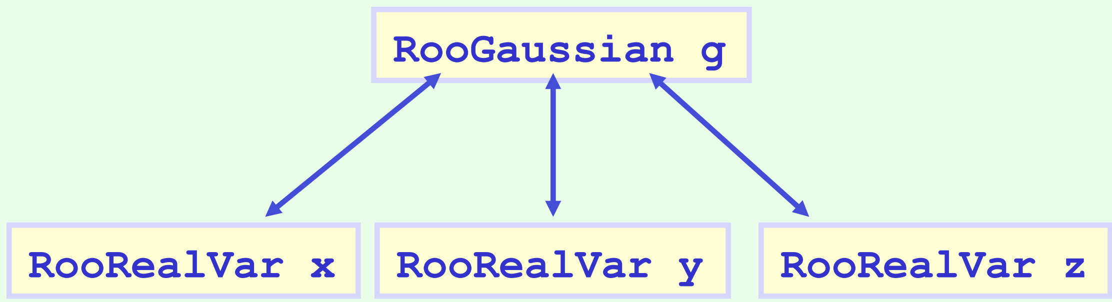
RooFit core design philosophy - Workspace

- Instead of **`'double Likelihood(double paramVec[])'`**, a flexible modular structure of 'programmed' functions

Math

Gauss(x,μ,σ)

RooFit diagram



RooFit code

```
RooRealVar x("x","x",-10,10) ;
RooRealVar m("m","y",0,-10,10) ;
RooRealVar s("s","z",3,0.1,10) ;
RooGaussian g("g","g",x,m,s) ;
```

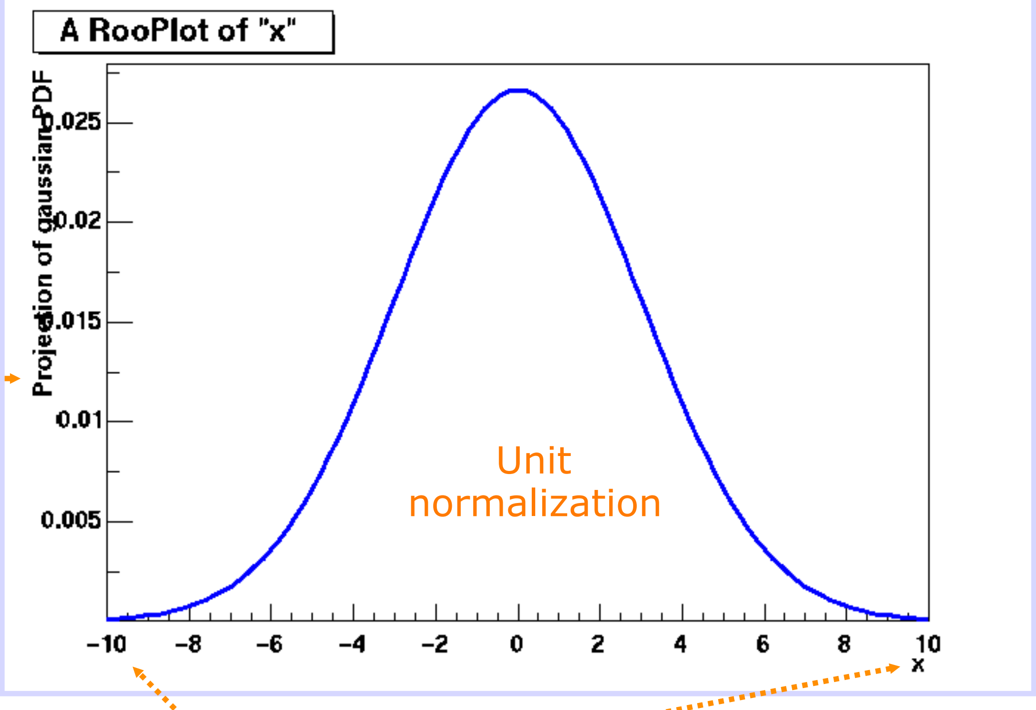
Basics – Creating and plotting a Gaussian p.d.f

Setup gaussian PDF and plot

```
// Create an empty plot frame
RooPlot* xframe = w::x.frame() ;

// Plot model on frame
model.plotOn(xframe) ;

// Draw frame on canvas
xframe->Draw() ;
```

Axis label from `gauss` title

A `RooPlot` is an empty frame capable of holding anything plotted versus it variable

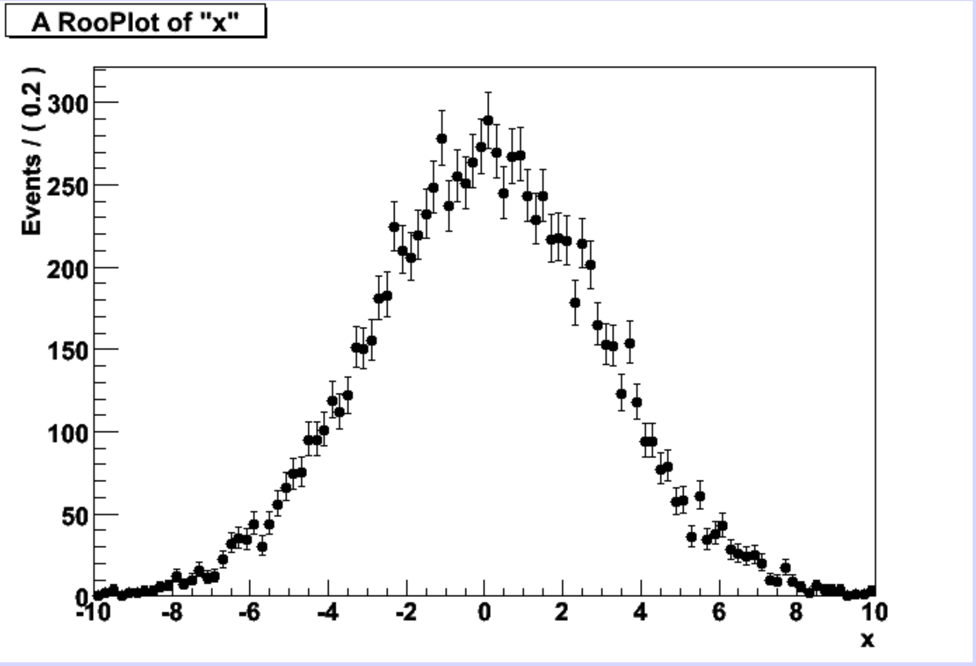


Plot range taken from limits of `x`

Basics – Generating toy MC events

Generate 10000 events from Gaussian p.d.f and show distribution

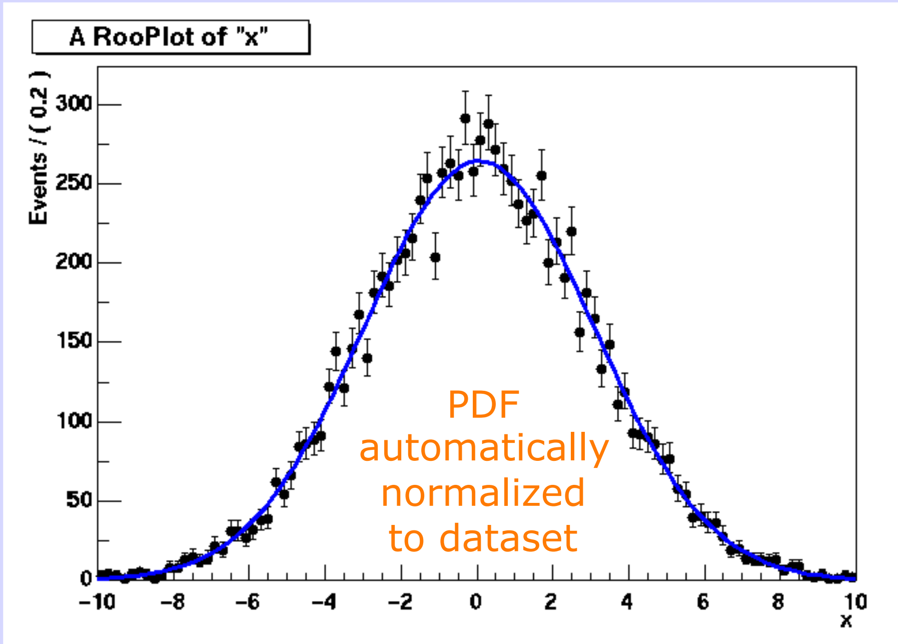
```
// Generate an unbinned toy MC set
RooDataSet* data = w::gauss.generate(w::x,10000) ;

// Generate an binned toy MC set
RooDataHist* data = w::gauss.generateBinned(w::x,10000) ;

// Plot PDF
RooPlot* xframe = w::x.frame()
data->plotOn(xframe) ;
xframe->Draw() ;
```



Can generate both binned and unbinned datasets

Basics – ML fit of p.d.f to *unbinned* data

```
// ML fit of gauss to data
w::gauss.fitTo(*data) ;
(MINUIT printout omitted)

// Parameters if gauss now
// reflect fitted values
w::mean.Print()
RooRealVar::mean = 0.0172335 +/- 0.0299542
w::sigma.Print()
RooRealVar::sigma = 2.98094  +/- 0.0217306

// Plot fitted PDF and toy data overlaid
RooPlot* xframe = w::x.frame() ;
data->plotOn(xframe) ;
w::gauss.plotOn(xframe) ;
```



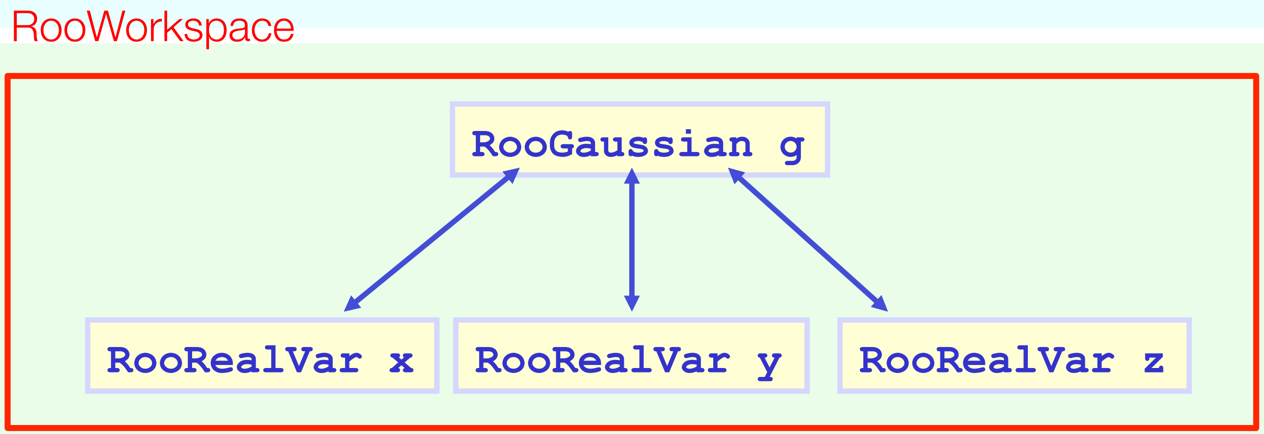
RooFit core design philosophy - Workspace

- The workspace serves a container class for all objects created

Math

Gauss(x,μ,σ)

RooFit diagram

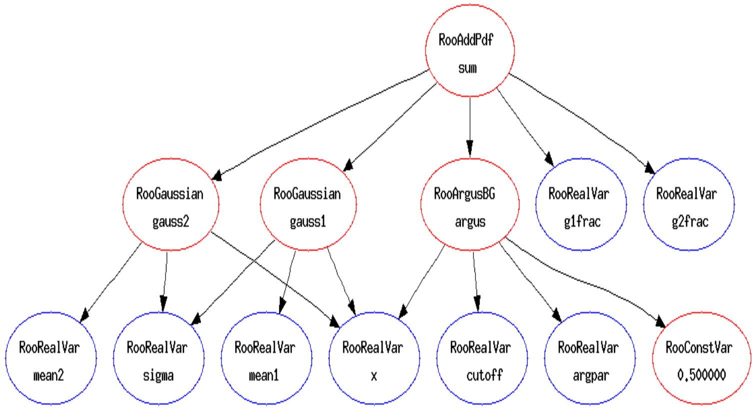


RooFit code

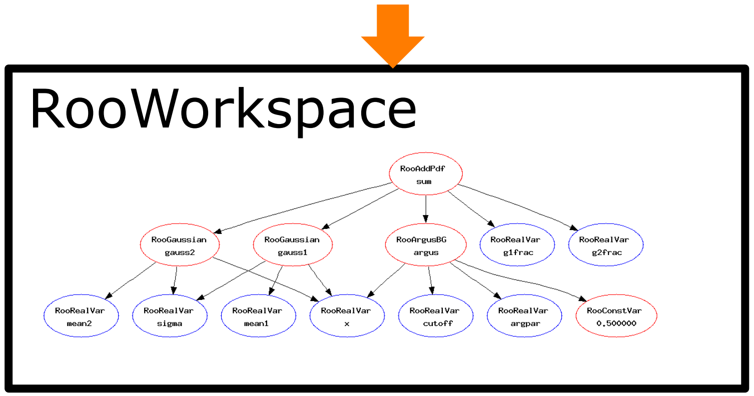
```
RooRealVar x(“x”,”x”,-10,10) ;
RooRealVar m(“m”,”y”,0,-10,10) ;
RooRealVar s(“s”,”z”,3,0.1,10) ;
RooGaussian g(“g”,”g”,x,m,s) ;
RooWorkspace w(“w”) ;
w.import(g) ;
```

The workspace

- The workspace concept has revolutionized the way people share and combine analysis
  - Completely factorizes process of building and using likelihood functions
  - You can give somebody an analytical likelihood of a (potentially very complex) physics analysis in a way to the easy-to-use, provides introspection, and is easy to modify.

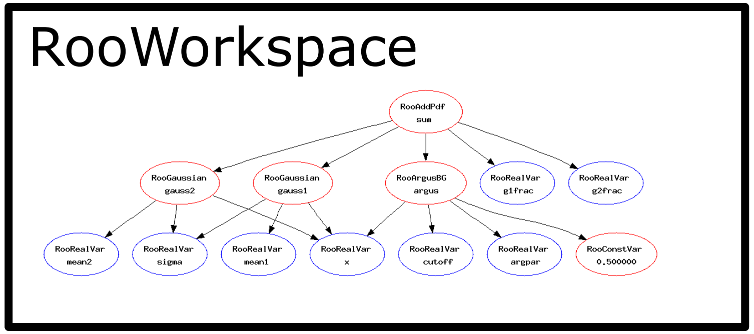


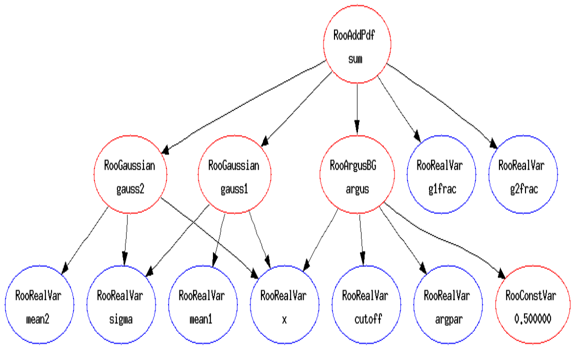

```
RooWorkspace w("w") ;
w.import(sum) ;
w.writeToFile("model.root") ;
```



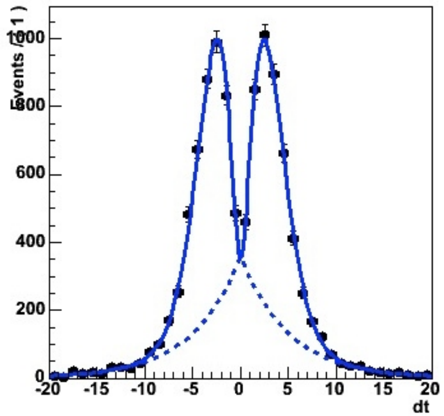
model.root

Using a workspace






```
// Resurrect model and data
TFile f("model.root") ;
RooWorkspace* w = f.Get("w") ;
RooAbsPdf* model = w->pdf("sum") ;
RooAbsData* data = w->data("xxx") ;

// Use model and data
model->fitTo(*data) ;

RooPlot* frame =
         w->var("dt")->frame() ;
data->plotOn(frame) ;
model->plotOn(frame) ;
```



Factory and Workspace

- *One C++ object per math symbol* provides ultimate level of control over each objects functionality, but results in lengthy user code for even simple macros
- Solution: add factory that auto-generates objects from a math-like language. Accessed through factory() method of workspace
- Example: reduce construction of Gaussian pdf and its parameters from 4 to 1 line of code

```
RooRealVar x("x","x",-10,10) ;
RooRealVar mean("mean","mean",5) ;
RooRealVar sigma("sigma","sigma",3)  ;
RooGaussian f("f","f",x,mean,sigma) ;
w.import(f) ;
```

```
w.factory("Gaussian::f(x[-10,10],mean[5],sigma[3])") ;
```

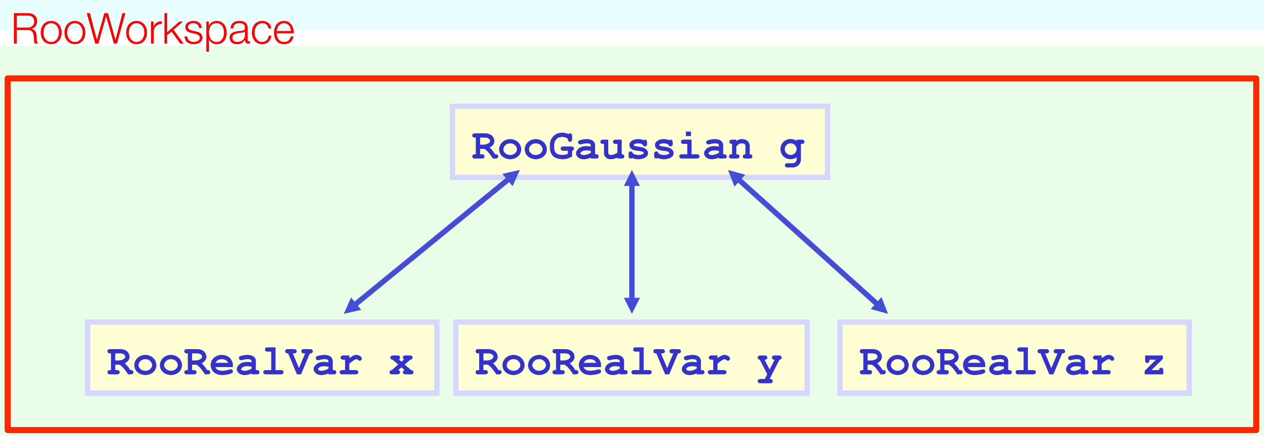
RooFit core design philosophy - Workspace

- The workspace serves a container class for all objects created

Math

$$\mathrm{Gauss}(x,\mu,\sigma)$$

RooFit diagram



RooFit code

```
RooRealVar x("x","x",-10,10) ;
RooRealVar m("m","y",0,-10,10) ;
RooRealVar s("s","z",3,0.1,10) ;
RooGaussian g("g","g",x,m,s) ;
RooWorkspace w("w") ;
w.import(g) ;
```

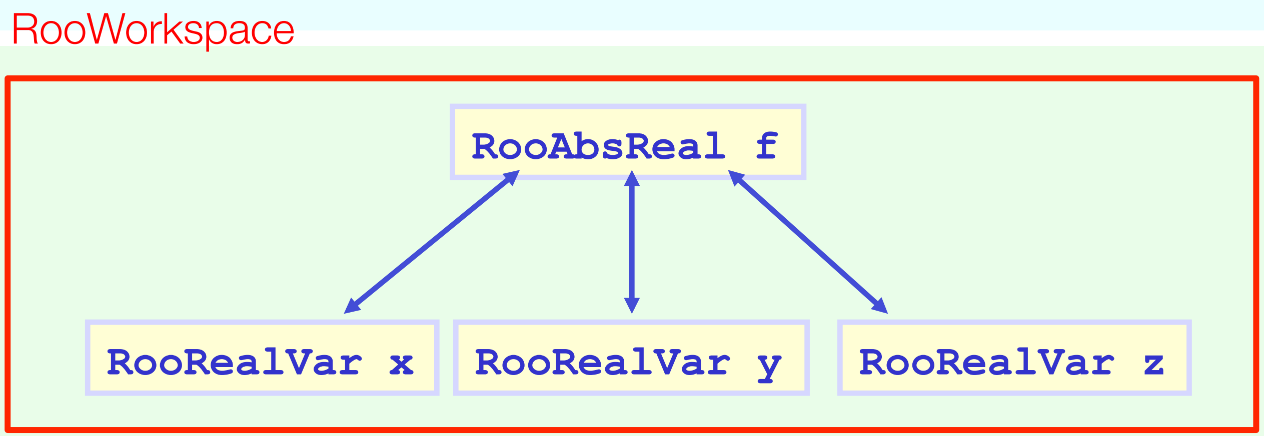
Populating a workspace the easy way – “the factory”

- The factory allows to fill a workspace with pdfs and variables using a simplified scripting language

Math

Gauss(x,μ,σ)

RooFit diagram

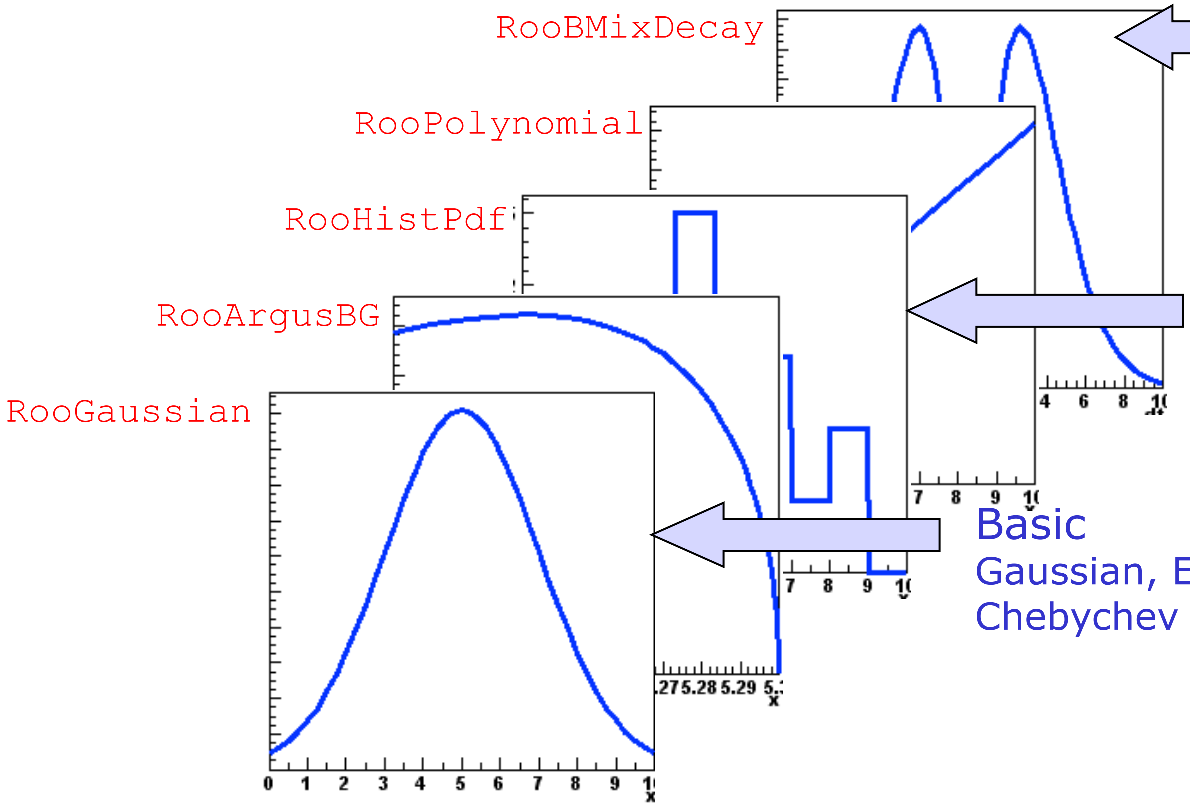


RooFit code

```
RooWorkspace w(“w”) ;
w.factory(“RooGaussian::g(x[-10,10],m[-10,10],z[3,0.1,10])”);
```

Model building – (Re)using standard components

- RooFit provides a collection of compiled standard PDF classes



Physics inspired
ARGUS,Crystal Ball,
Breit-Wigner, Voigtian,
B/D-Decay,….

Non-parametric
Histogram, KEYS

Basic
Gaussian, Exponential, Polynomial,…
Chebychev polynomial

Easy to extend the library: each p.d.f. is a separate C++ class

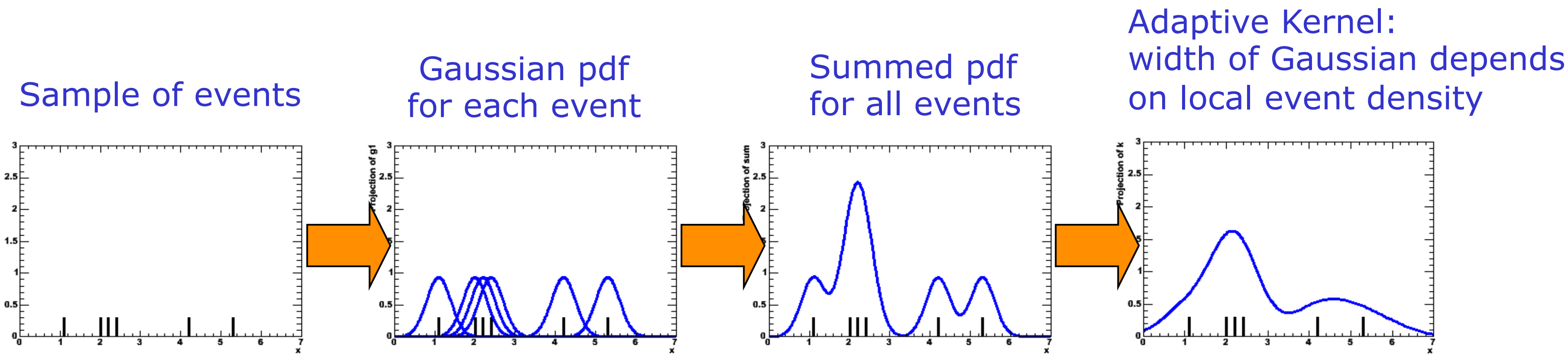
Model building – (Re)using standard components

- List of most frequently used pdfs and their factory spec

Gaussian	`Gaussian::g(x,mean,sigma)`
Breit-Wigner	`BreitWigner::bw(x,mean,gamma)`
Landau	`Landau::l(x,mean,sigma)`
Exponential	`Exponental::e(x,alpha)`
Polynomial	`Polynomial::p(x,{a0,a1,a2})`
Chebychev	`Chebychev::p(x,{a0,a1,a2})`
Kernel Estimation	`KeysPdf::k(x,dataSet)`
Poisson	`Poisson::p(x,mu)`
Voigtian (=BW⊗G)	`Voigtian::v(x,mean,gamma,sigma)`

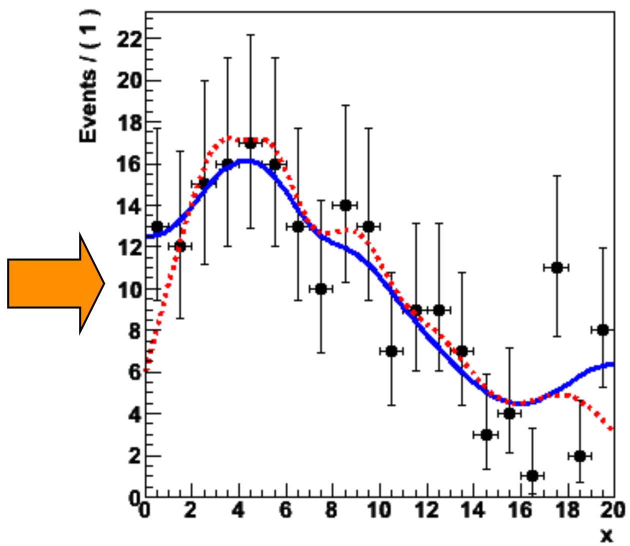
The power of pdf as building blocks – Advanced algorithms

- Example: a 'kernel estimation probability model'
  - Construct smooth pdf from unbinned data, using kernel estimation technique



- Example

```
w.import(myData,Rename("myData")) ;
w.factory("KeysPdf::k(x,myData)") ;
```



- Also available for n-D data

The power of pdf as building blocks – adaptability

- RooFit pdf classes do not require their parameter arguments to be variables, one can plug in functions as well
- Allows trivial customization, extension of probability models

class RooGaussian

$$Gauss(x \mid \mu, \sigma)$$

also class RooGaussian!

$$Gauss(x \mid \underbrace{\mu \cdot (1+2\alpha)}, \sigma)$$

Introduce a response function for a systematic uncertainty

```
// Original Gaussian
w.factory("Gaussian::g1(x[80,100],m[91,80,100],s[1])")


// Gaussian with response model in mean
w.factory("expr::m_response("m*(1+2alpha)",m,alpha[-5,5])") ;
w.factory("Gaussian::g1(x,m_response,s[1])")
```

NB: "expr" operates builds an intepreted function expression on the fly

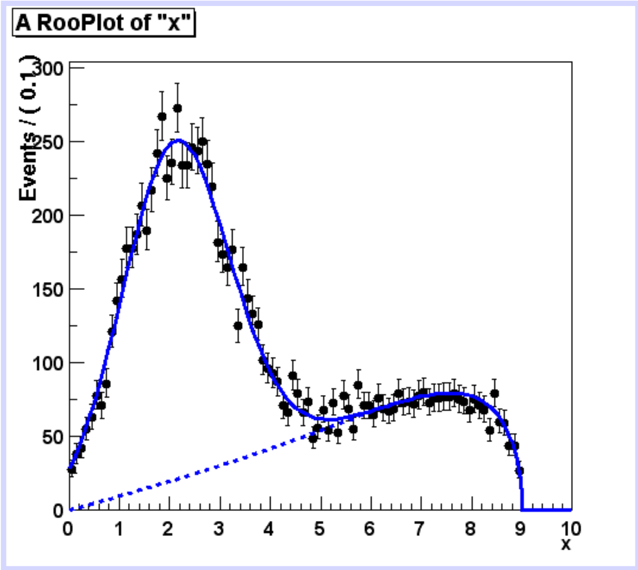
The power of building blocks – operator expressions

- Create a SUM expression to represent a sum of probability models

```
w.factory("Gaussian::gauss1(x[0,10],mean1[2],sigma[1]") ;
w.factory("Gaussian::gauss2(x,mean2[3],sigma)") ;
w.factory("ArgusBG::argus(x,k[-1],9.0)") ;

w.factory("SUM::sum(g1frac[0.5]*gauss1, g2frac[0.1]*gauss2, argus)")
```

- In composite model visualization components can be accessed by name

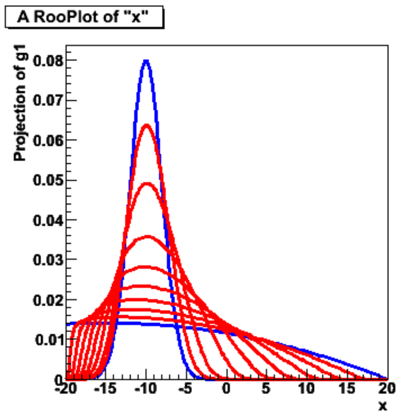
```
// Plot only argus components
w::sum.plotOn(frame,Components("argus"),
              LineStyle(kDashed)) ;
```

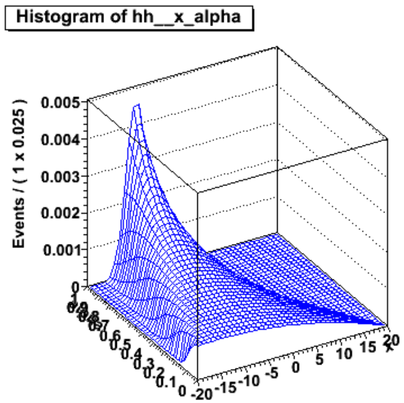


Powerful operators – Morphing interpolation

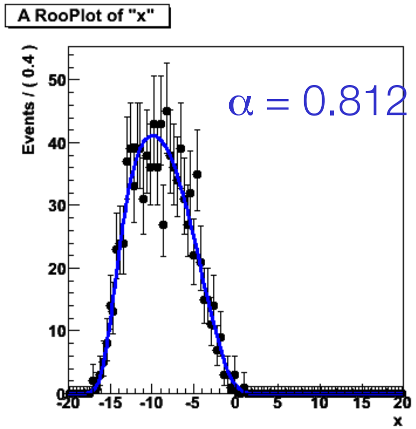
- Special operator pdfs can interpolate existing pdf shapes
  - Ex: interpolation between Gaussian and Polynomial

```
w.factory("Gaussian::g(x[-20,20],-10,2)") ;
w.factory("Polynomial::p(x,{-0.03,-0.001})") ;
w.factory("IntegralMorph::gp(g,p,x,alpha[0,1])") ;
```





Fit to data



$\alpha = 0.812 \pm 0.008$

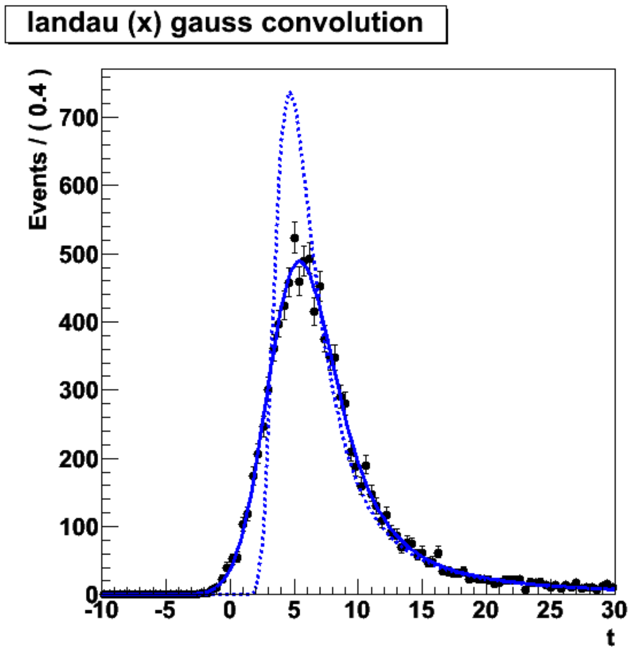
- Three morphing operator classes available
  - IntegralMorph (Alex Read).
  - MomentMorph (Max Baak).
  - PiecewiseInterpolation (via HistFactory)

Powerful operators – Fourier convolution

- Convolve any two arbitrary pdfs with a 1-line expression

```
w.factory("Landau::L(x[-10,30],5,1)") :
w.factory("Gaussian::G(x,0,2)") ;

w::x.setBins("cache",10000) ; // FFT sampling density
w.factory("FCONV::LGf(x,L,G)") ; // FFT convolution
```

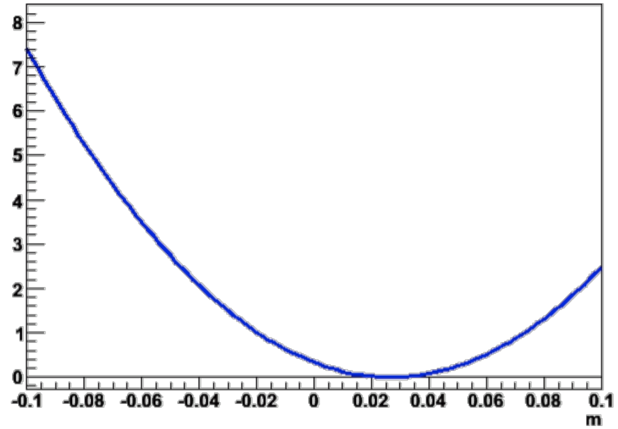
- Exploits power of FFTW package available via ROOT
  - Hand-tuned assembler code for time-critical parts
  - Amazingly fast: unbinned ML fit to 10.000 events take ~5 seconds!



Working with the likelihood function

- Plot the likelihood function versus a parameter

```
RooAbsReal* nll = w::model.createNLL(data) ;

RooPlot* frame = w::param.frame() ;
nll->plotOn(frame,ShiftToZero()) ;
```

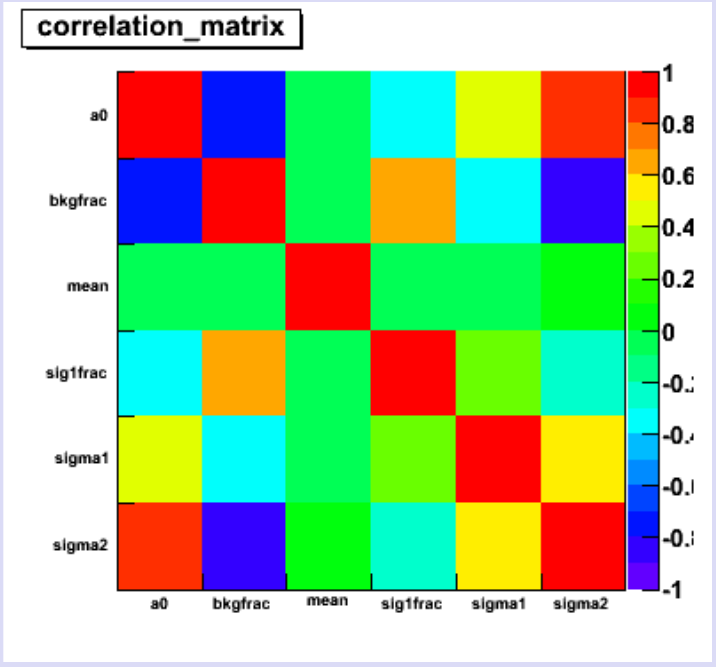


- Maximum Likelihood estimation of parameters and variance

```
RooMinimizer m(*nll) ;

// ML Parameter estimation
m.minimize("Minuit2","migrad") ;

// Variance estimation
m.hesse() ;

// Alternatively - all this in one line
pdf->fitTo(*data) ;
```

Working with covariance and correlation matrices

- Detailed information on parameter and covariance estimates can be saved for detailed information



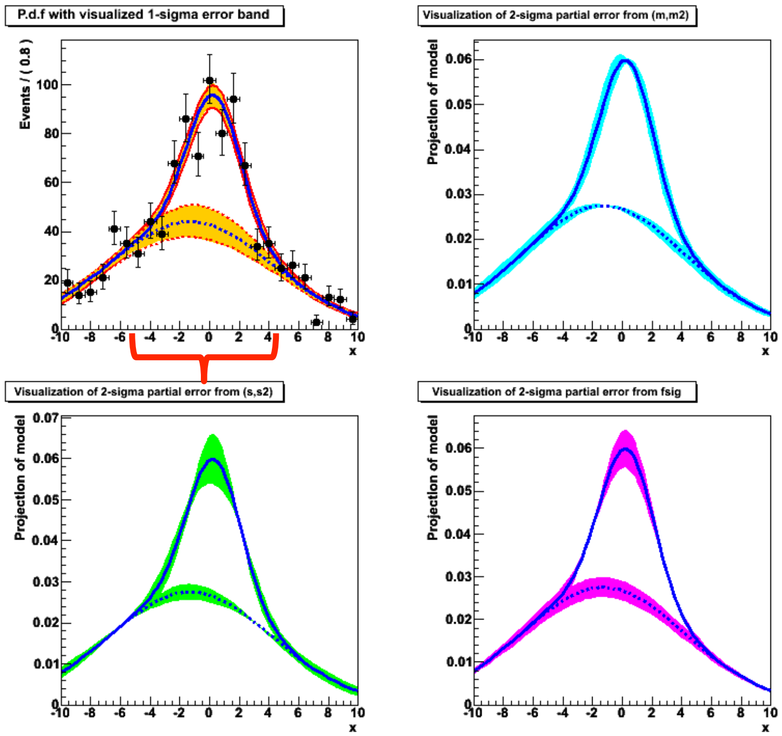

```
RooMinimizer m(*nll) ;
m.minimize("Minuit2","migrad") ;
m.hesse() ;
RooFitResult* r = m.save() ;

// Visualize correlation matrix
r->correlationHist->Draw("colz") ;

// Extract correlation,covariance matrix
TMatrixDSym cov = fr->covarianceMatrix() ;
TMatrixDSym cov = fr->covarianceMatrix(a,b) ;
```

Use covariance matrices for correlated error propagation

- Can (as visual aid) propagate errors in covariance matrix of a fit result to a pdf projection

```
w::model.plotOn(frame,VisualizeError(*fitresult)) ;
w::model.plotOn(frame,VisualizeError(*fitresult,fsig)) ;
```

  – Linear propagation on pdf projection $\Delta = \vec{E} V^{-1} \vec{E}$



- Propagated error can be calculated on arbitrary function
  – E.g fraction of events in signal range

```
RooAbsReal* fracSigRange =
     w::model.createIntegral(x,x,"sig") ;

Double_t err =
    fracSigRange.getPropagatedError(*fr);
```

Overview

- Start with basics, gradually build up to complexity of

Statistical tests with simple hypotheses for counting data	*“What do we mean with probabilities”*
Statistical tests with simple hypotheses for distributions	*“p-values”*
Hypothesis testing as basis for event selection	*“Optimal event selection & machine learning”*
Composite hypotheses (with parameters) for distributions	*“Confidence intervals, Maximum Likelihood”*
Statistical inference with nuisance parameters	*“Fitting the background”*
Response functions and subsidiary measurements	*“Profile Likelihood fits”*