RooStats for Searches

Grégory Schott

Karlsruhe Institute of Technology

Institute for Experimental Particle Physics
of the Karlsruhe Institute of Technology

CMS

Phystat 2011, CERN, January 17th 2011

Introduction 1/2

What is RooStats ?

- a collaborative project with contributors from ATLAS, CMS and ROOT aimed to provide & consolidate statistical tools needed by LHC
  - part of ROOT since Summer 2008
  - based on previous code in ATLAS & CMS and original contributions:
    - Cranmer (ATLAS), Moneta (ROOT), Schott (CMS), Verkerke (RooFit) and other contributors: Belasco, Kreiss, Lazzaro (ATLAS); Pelliccioni, Piparo, Ruthmann, Schmitz, Wolf (CMS)
  - oversight from the statistics committees of both experiments

What is the aim ?

- to cover the most popular statistical approaches used in HEP
  - it becomes possible to easily compare different statistical approaches
- using same tools: compare easily results across experiments
  - not only desirable but necessary for combinations
- to have quite flexible and well tested tools

Introduction 2/2

RooStats is built on top of the RooFit toolkit :

- data modelling language (for PDFs, likelihoods, ...)
  - very flexible & fits our needs

Common statistical applications:

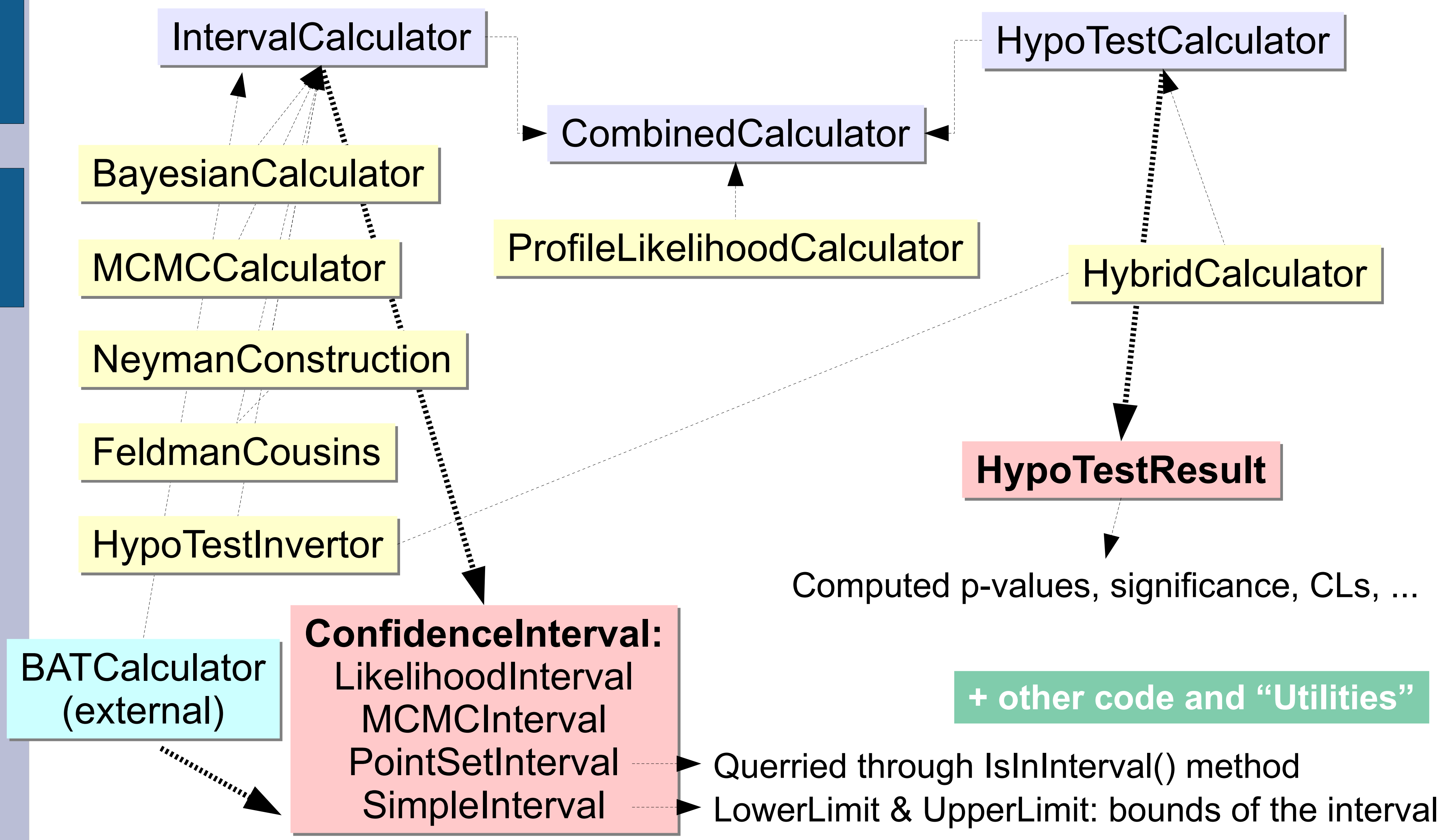
- Point estimation: determine the best estimate of a parameter
- Estimation of confidence/credible intervals: regions representing the range of a parameter of interest compatible with the data
- Hypothesis tests: evaluation of p-value for one or multiple hypotheses (significance)
- Goodness of fit: quantify how well a model describes the data

→ for these things in particular, RooStats can help your analysis

Outline

- RooFit
  - Likelihood function
  - The workspace
- Profiled likelihood
- Bayesian methods (analytical / MC-based)
- Frequentist approaches
  - Neyman construction / Feldman-Cousins
  - Hybrid Frequentist-Bayesian
  - Modified frequentist
- Advanced tools / latest developments
- Validations
- Conclusion

Overview of classes in RooStats



Likelihood function

- All statistical methods start from the description of a likelihood function
  - A rather general likelihood function with multiple observables, extended and unbinned can be written:

$$L(\vec{x}|r,s,b,\vec{\theta}_s,\vec{\theta}_b) = e^{-rs-b} \prod_{j=1}^{n} (rsf_s(\vec{x}_j|\vec{\theta}_s) + bf_b(\vec{x}_j|\vec{\theta}_b))$$

- f_s and f_b signal and background distributions from MC simulations or constrained with control samples (PDF)
- weighted by signal and background yields
  - r can also be called signal strength and written μ or μ_S

    For Higgs searches it corresponds to $r = \sigma(H) / \sigma(H_{exp/SM})$
- Having a 95% CL UL for r=1 means the Standard Model can be excluded at 95% CL

RooFit

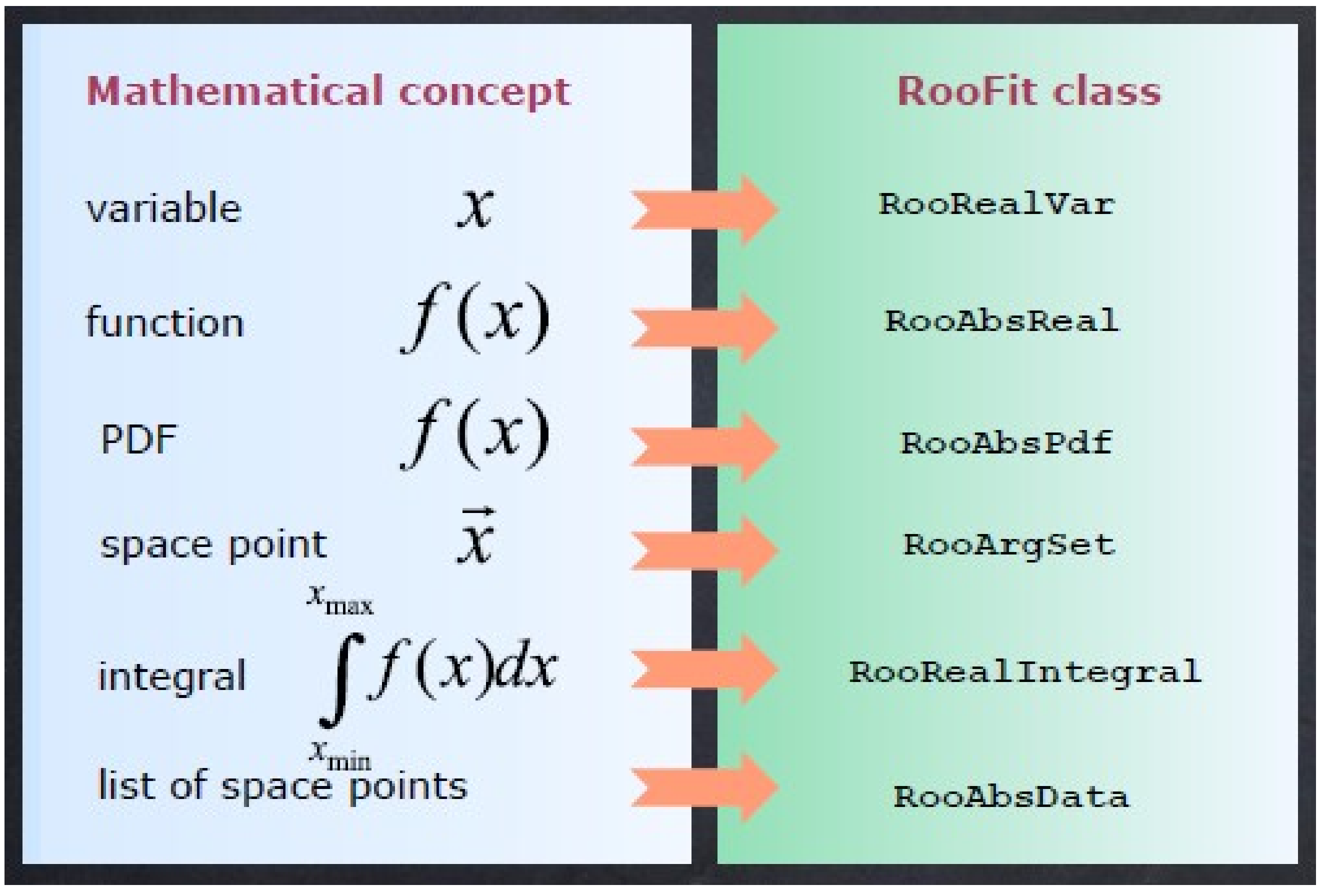
- Toolkit for data modelling (PDF, likelihood, variables, data)
  - developped by Kirby & Verkerke and used since > 10 years by the BaBar collaboration and others
- Large collection of models available
  - Composition to build complex models
    - Addition, product, convolution, ...
    - Simutaneous fits
  - Scales to arbitrary complexity
- Handles binned and unbinned model and data
- All models support for integration, maximum likelihood fitting, toy-MC generation, visualization, ...

→ Usable for complex problems

- *Modularity allows to work on arbitrary data and model and can handle many observables, parameters of interest and nuisance parameters*

RooFit design

- Relies on ROOT for core functionalities (with little redudancies)
- Mathematical concepts are mapped to C++ classes

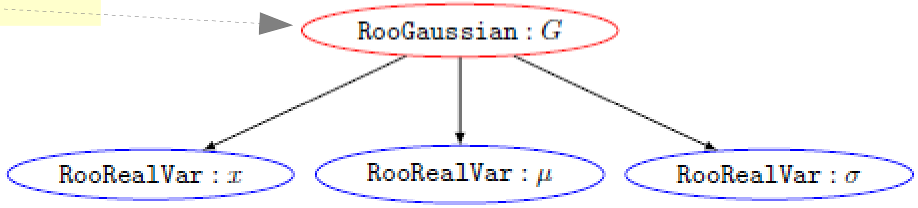


RooFit PDFs

- Example of PDF definition in RooFit:
  *Gaussian distribution of the random variable **x** with parameters **μ** and **σ***

$$G(x|\mu,\sigma)$$

```
//define observables and parameters
RooRealVar x("x","x",100,200);
RooRealVar mu("mu","#mu",150);
RooRealVar sigma("sigma","#sigma",5,0,20);
// make a simple model
RooGaussian G("G","gaussian",x,mu,sigma);
G.graphVizTree("GaussianModel.dot");
```



- Provides a factory to auto-generates objects from a math-like language

```
// shortcut factory definition of the model
RooWorkspace w;
w.factory("Gaussian::G(x[100,200],mu[150],sigma[5,0,20]");
w.Print();
```

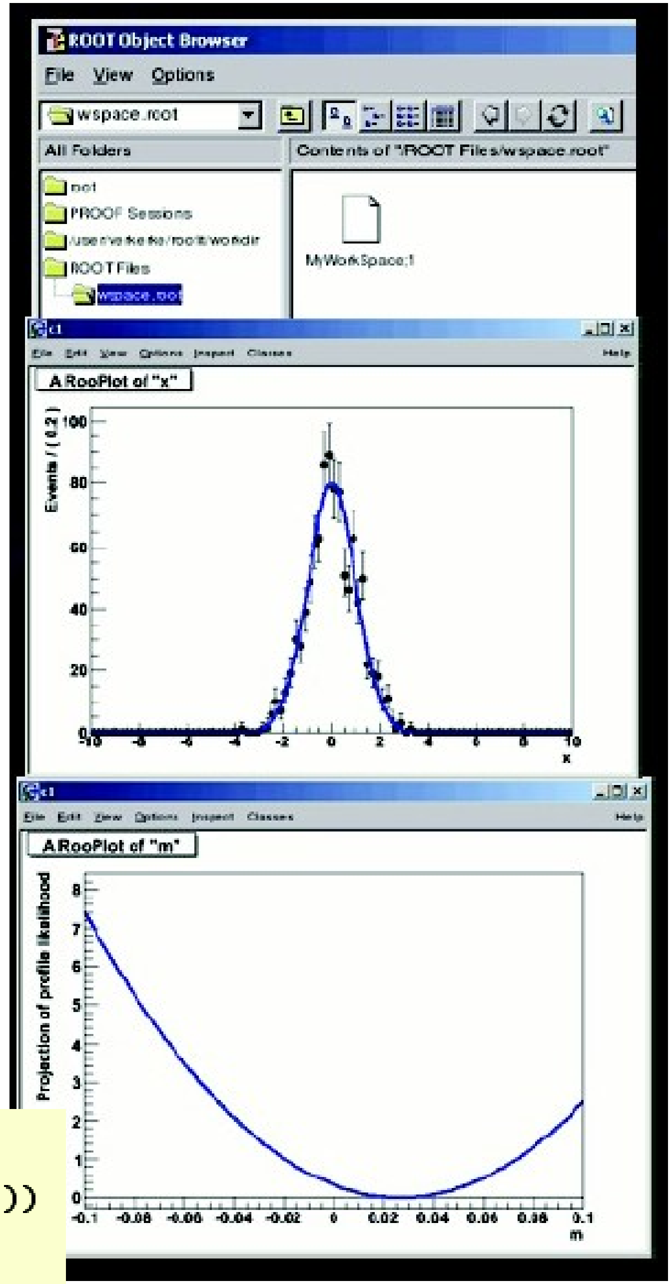
```
RooWorkspace()  contents
variables
    (mu,sigma,x)
p.d.f.s
    RooGaussian::G[ x=x mean=mu sigma=sigma ] = 1
```

RooWorkspace

RooWorkspace class of RooFit:

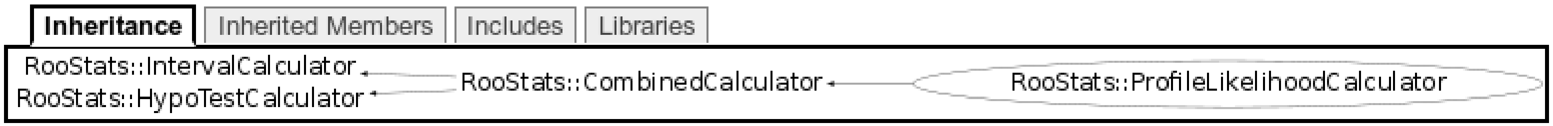
- can hold almost any RooFit object: variables, distributions, data, user-defined classes, ...
- maintain complete description of whole model
- possibility to save it to a ROOT file
  - very good for electronic publication of data and likelihood function
  - and greatly help for combination (that's the format agreed to share between Atlas & CMS)
- still leaves a lot of freedom on how to perform the analysis

- allows introspection and to do adjustments
- includes helper tools for combination




```
RooWorkspace w("w","joint workspace") ;
// Import top-level pdfs and all their components, variables
w.import("channelA.root:w:pdfA",RenameAllVariablesExcept("A","mhiggs"))
w.import("channelB.root:w:pdfB",RenameVariable("mH","mhiggs")) ;
w.import("channelC.root:w:pdfC") ;
// Construct joint pdf
w.factory("SIMUL::joint(chan[A,B,C],A=pdfA,B=pdfB,C=pdfC)") ;
```

ProfileLikelihoodCalculator



- Wilks' theorem: Log-likelihood ratio follow asymptotically χ^2 distribution (under regularity conditions)

- Defining likelihood ratio: $Q = L(r=0)/L(\hat{r})$ $\hat{r}$: best fit value

- Significance estimator: $S_L = \sqrt{-2\ln Q}$

  – Significance: represents probability for background to have fluctuated to the level actually observed; usually given in units of "sigma": *5σ for a p-value of 2.87×10 -7*

- Simplified formula for counting analysis: $L(n) = Poisson(n; \hat{r}\cdot s, b)$

$$S_{cL} = \sqrt{2\hat{r}s - 2n\ln(1+\hat{r}s/b)}$$

Inclusion of systematics

- Inclusion of systematic uncertainties:
  - Add a multiplicative term to the likelihood function:
    - for example: Correlated gaussian: where C is the covariance matrix $e^{-0.5\ \vec{\theta}^{\,T} C\ \vec{\theta}}$
  - Then, minimize against nuisance parameters too:

$$Q = L\left(r = 0|\hat{\hat{\vec{\theta}}}\right) / L\left(\hat{r}|\hat{\vec{\theta}}\right)$$

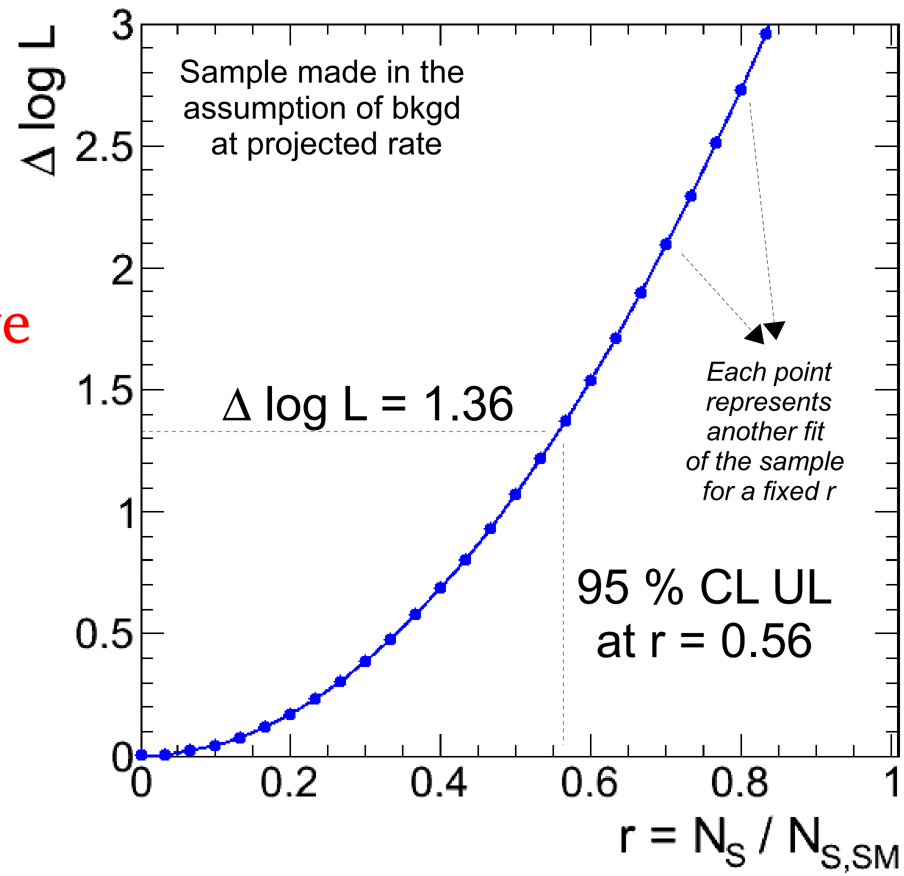
- for some specific cases there also exists simple formulae for significance but most generally one should use S_L with systematics taken into account in the likelihood function

ProfileLikelihood for intervals estimation

Still assuming Wilks' theorem holds for the analysis, PL can sometimes be used for estimating confidence intervals:

- construct the profile likelihood curve
- and look for r such as

Δ log L =
0.50 → 2-sided 68 % confidence interval
1.00 → 2-sided 95 % confidence interval
1.36 → 1-sided 95 % upper/lower limit



- 68% CL (1σ) interval estimation

```
ProfileLikelihoodCalculator plc(data,model,POI);
plc.SetTestSize(0.32);  // configure for 68% CL
LikelihoodInterval* interval = plc.GetInterval();
double lowerLimit = interval->LowerLimit(r);
double upperLimit = interval->UpperLimit(r);
LikelihoodIntervalPlot plot(interval);
plot.Draw();
```

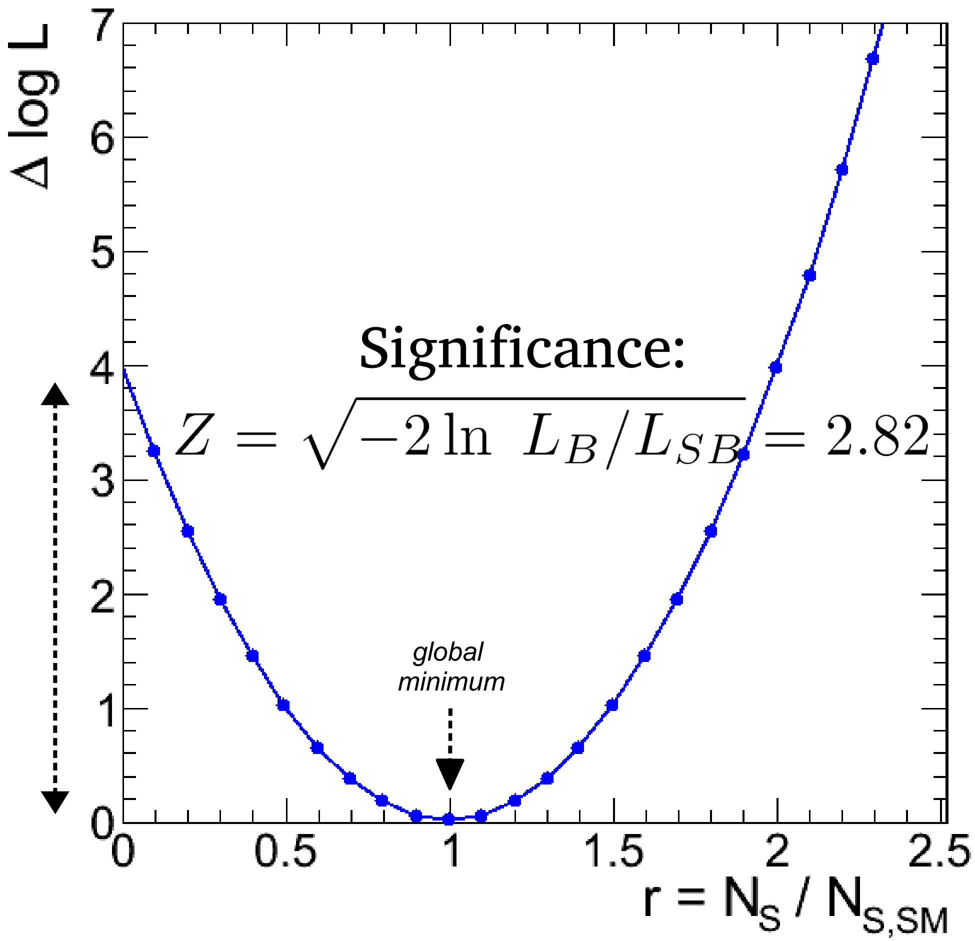
Minuit/Minos

Back on significance from PLC

Expected significance:

```
ProfileLikelihoodCalculator plc(data,model,POI);
r->setVal(0); // set value of r to zero
plc.SetNullParameters(RooArgSet(r));
HypoTestResult* hypotest = plc.GetHypoTest();
double significance = hypotest->Significance();
```

- data generated as expected (aka. "Asimov" dataset): i.e. the measurements are exactly at the values expected from the background-only model



New ProfileInspector class allow to also inspect the values of the nuisance parameters vs POI (see backup slide)

ModelConfig (1)

- ModelConfig let you specify in one block additional information needed to know how to use the PDF
  - what are the observables (esp. for toy-MC)
  - what is/are the parameter(s) of interest
  - is there additional prior on the parameters (Bayesian analysis)
  - is model conditional on other observables (eg. evt-by-evt errors)

An example with ModelConfig

Here we show use of the Workspace factory to create a model, and use of ModelConfig to specify what we will need for the statistical tools

Create a new workspace

Create a the pdf G(x|mu,1) and the variables x, mu, sigma using the factory syntax

Create a new ModelConfig

Define parameter sets for observables and parameters of interest

```
// make a simple model via the workspace factory
RooWorkspace* wspace = new RooWorkspace();
wspace->factory("Gaussian::normal(x[-10,10],mu[-1,1],sigma[1])");
wspace->defineSet("poi","mu");
wspace->defineSet("obs","x");

// specify components of model for statistical tools
ModelConfig* modelConfig = new ModelConfig("G(x|mu,1)");
modelConfig->SetWorkspace(*wspace);
modelConfig->SetPdf( *wspace->pdf("normal") );
modelConfig->SetParametersOfInterest( *wspace->set("poi") );
modelConfig->SetObservables( *wspace->set("obs") );

// create a toy dataset
RooDataSet* data = wspace->pdf("normal")->generate(*wspace->set("obs"),100);
```

Specify workspace that holds pdf, parameters of interest, observables, ...

... and we generate a toy dataset with 100 measurements of the observables (x)
(note, the data is NOT part of the ModelConfig)

ModelConfig (3)

```
RooStats::ProfileLikelihoodCalculator    ProfileLikelihoodCalculator(RooAbsData& data,
RooStats::ModelConfig& model, Double_t size = 0.05)
```

```
RooStats::ProfileLikelihoodCalculator    ProfileLikelihoodCalculator(RooAbsData& data,
RooAbsPdf& pdf, const RooArgSet& paramsOfInterest, Double_t size = 0.05, const
RooArgSet* nullParams = 0)
```

- This makes the interface more standard across the different calculator classes ... but also it becomes less obvious what of the elements in the ModelConfig are used/necessary for a given calculator
  - hopefully we will keep both in the constructors of our classes in the future

Bayesian analysis

- Bayesian theorem: $P(r|data,\theta) = pdf(data|r,\theta)\ \pi(r) / \int P(data|r,\theta)\ \pi(r)\ dr$

  probability density function

  posterior probability

  other parameters of the model

  prior probability

  normalisation term

  – Test the compatibility of the model against a given data sample

- Common treatment of systematics with an integration over a Bayesian prior

$$P(r|data,\theta) \propto \int pdf(data|r,\theta)\ \pi(r)\ \pi'(\theta)\ d\theta$$

- Perform the Bayesian integration via analytically, numerically or Markov Chain Monte-Carlo → 3 RooStats calculators:
  – BayesianCalculator: simple numerical
  – MCMCCalculator: Metropolis-Hastings MC used for computing the posterior probability
  – BATCalculator: also MCMC; external to RooStats but can be used with the same interface (Schmitz et Schott, see backup slide)

Bayesian RooStats usage

BayesianCalculator:

- Posterior and interval estimation with simple numerical integration
  - current implementation works only for one parameter of interest
  - very fast... but... beyond 5-8 nuisance parameters will suffer in numerical precision and will stop working properly

```
BayesianCalculator bc(*data,*modelPdf,*POI);
bc.SetTestSize(0.10);
SimpleInterval* interval = bc.GetInterval();
double lowerLimit = interval->LowerLimit(r);
double upperLimit = interval->UpperLimit(r);
```

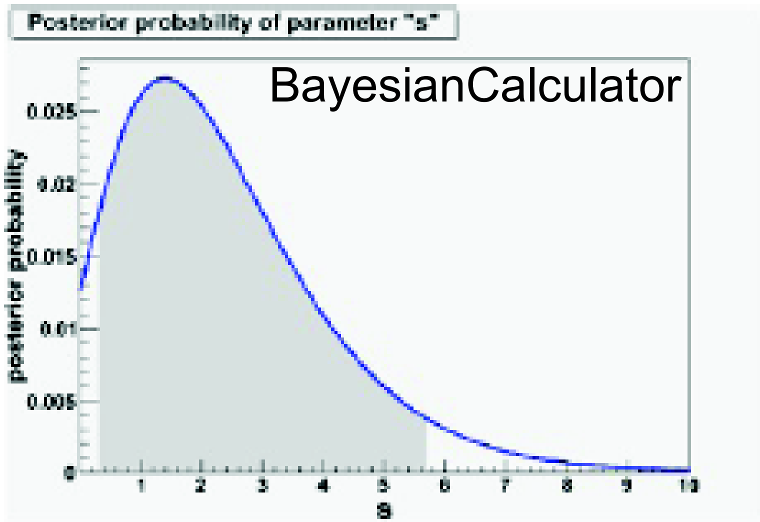
MCMCCalculator:

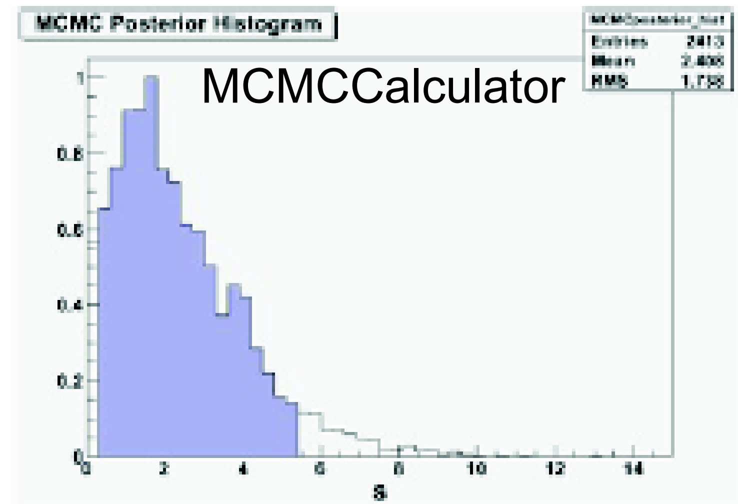
- Integration using Markov-Chain MC
  - Possible to specity ProposalFunction
  - Can vizualize posterior or the chain

```
MCMCCalculator mccalc(*data,*model,POI,*priorPOI);
mccalc.SetConfidenceLevel(0.90);
MCMCInterval * mcInterval = mcCalc.GetInterval();
```

Credible intervals

- But the way those integrations were performed should not impact the credible intervals computed from the posterior distribution
  - what varies is the speed, stability, precision of those calculations





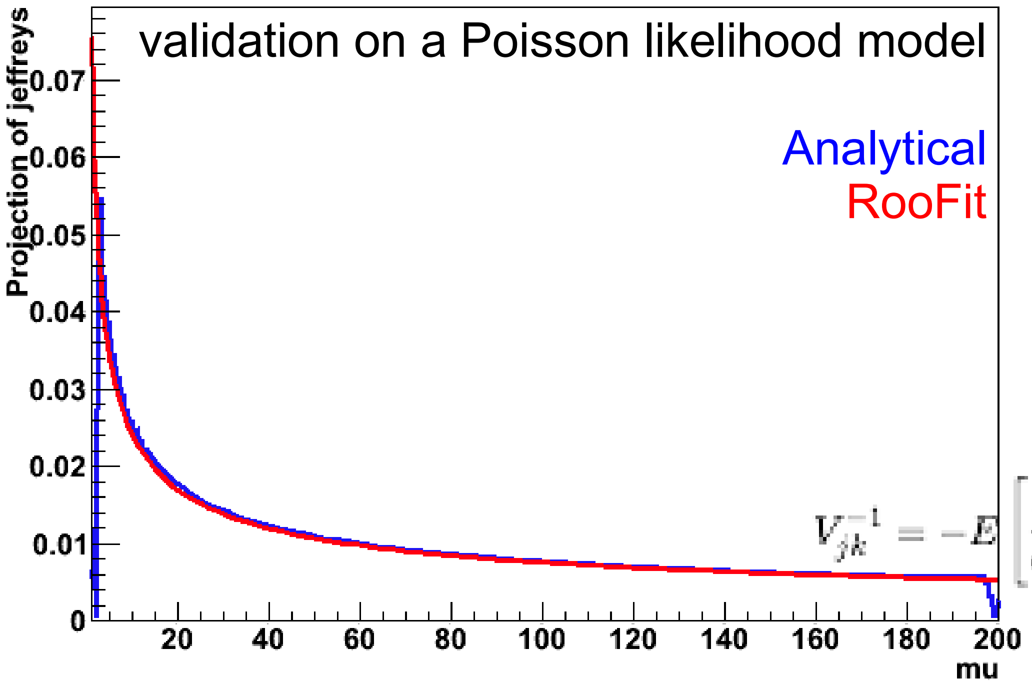
- What will however have an impact is the:
  - choice of priors on the nuisance parameters and on the parameter(s) of interest (this later: often taken flat → pseudo-Bayesian)
  - choice of ordering rule: central, shortest, one-sided interval (this is done by setting the left-side tail fraction to zero), ...
- those choices can be configured by the user in the classes!

Bayesian priors

- Added Lognormal and Gamma functions to RooFit:
  - those are better behaved prior for nuisance param than Gaussians

$$p(b|m0,k) = \frac{1}{\sqrt{2\pi} b \ln k} exp^{-0.5 \frac{\ln^2(b/m0)}{\ln^2 k}} \qquad p(x|\gamma,\beta,\mu) = \frac{(x-\mu)^{\gamma-1} \cdot \exp^{-(x-\mu)/\beta}}{\Gamma(\gamma) \cdot \beta^{\gamma}}$$

- New RooJeffreys class: "objective" prior based on formal rules (related to Fisher information and the Cramér-Rao bound)



- implemented for arbitrary PDF using "Asimov" dataset to help calculate the Fisher information [arXiv:1007:1727]

$$\pi(\vec{\theta}) \propto \sqrt{\det \mathcal{I}\left(\vec{\theta}\right)}.$$

$$\left(\mathcal{I}\left(\theta\right)\right)_{i,j} = -\operatorname{E}\left[\left.\frac{\partial^2}{\partial\theta_i \partial\theta_j} \ln f(X;\theta)\right|\theta\right].$$

$$V_{jk}^{-1} = -E\left[\frac{\partial^2 \ln L}{\partial\theta_j \partial\theta_k}\right] = -\frac{\partial^2 \ln L_A}{\partial\theta_j \partial\theta_k} = \sum_{i=1}^{N} \frac{\partial\nu_i}{\partial\theta_j}\frac{\partial\nu_i}{\partial\theta_k}\frac{1}{\nu_i} + \sum_{i=1}^{M} \frac{\partial u_i}{\partial\theta_j}\frac{\partial u_i}{\partial\theta_k}\frac{1}{u_i}$$

- Missing (but I heared some people are working on) a RooStats implementation of reference priors ...

Elements entering frequentist analyses 1

- What is the test statistics to use (5 currently in RooStats):

  (it is a numerical summary of a set of data that reduces the data to one value that can be used to perform a hypothesis test)

  - Specifying "I'm using the likelihood ratio" in not enough!

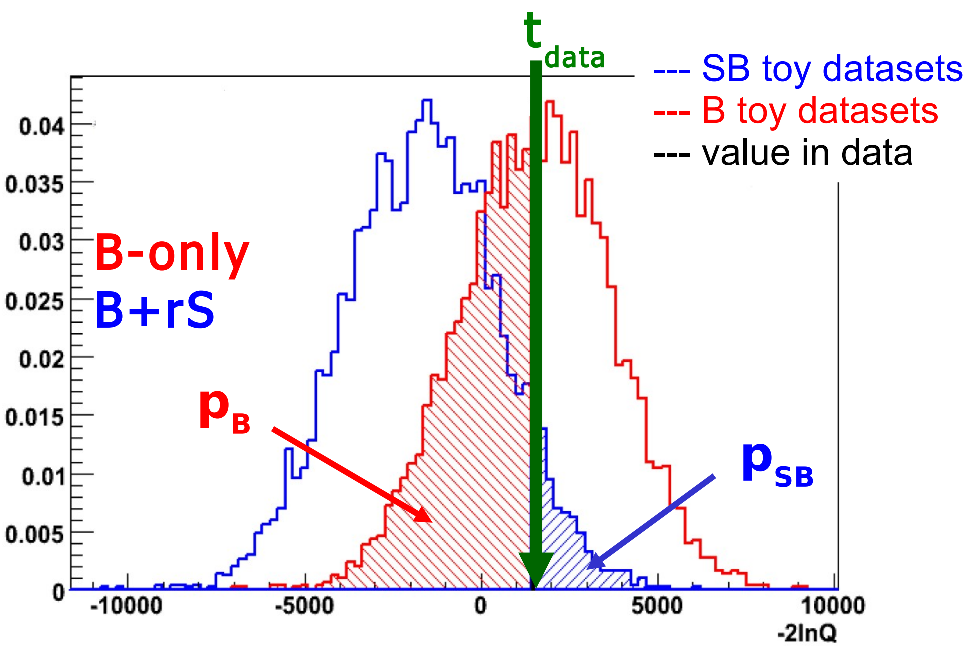
  – Simple likelihood ratio: $Q_{LEP} = L_{s+b}(\mu = 1)/L_b(\mu = 0)$
  – Ratio of profiled likelihoods: $Q_{TEV} = L_{s+b}(\mu = 1, \hat{\hat{\nu}})/L_b(\mu = 0, \hat{\hat{\nu}}')$
  – Profile likelihood ratio: $\lambda(\mu) = L_{s+b}(\mu, \hat{\hat{\nu}})/L_{s+b}(\hat{\mu}, \hat{\nu})$

- How to sample it:

  – Assuming asymptotic distribution (Wilks & Wald)?
  – Toy MC with nuisance parameters fixed (Neyman construction)?
  – Toy MC randomizing nuisance parameters according to a prior π(ν) in a Bayes-Frequentist Hybrid (prior-predictive)?
  – ...

HybridCalculator

- Perform hypothesis tests (for a given value of r)
- Evaluate frequentist compability of data (in toy-MC experiments)
  - Method based on a test statistics t
  - Play out toy experiments in the background-only hypothesis
    - Systematic uncertainties taken into account by Bayesian marginalization if requested, then: In generation of each pseudo-experiment vary value of nuisance parameters (Cousins-Highland integration)
  - Play out toy experiments again, now assuming $r \times S + B_i$



Distribution of the test statistics

Take the observation t_{data}

p-values definition:
$p_{SB} = CL_{SB} = \text{Prob}\ (\mathbf{t} > \mathbf{t}_{data})$
$p_B = 1\text{-}CL_B = \text{Prob}\ (\mathbf{t} < \mathbf{t}_{data})$

The background p-values give the significance of the signal observed

The class returns a HypoTestResult with p_{SB}, p_B, CL_S, significance

The issue with small p-values

Testing 5σ requires ~10^7 pseudo-experiments

- Monte Carlo methods are easy to parallelize
- ROOT provides a facility for parallel processing called PROOF

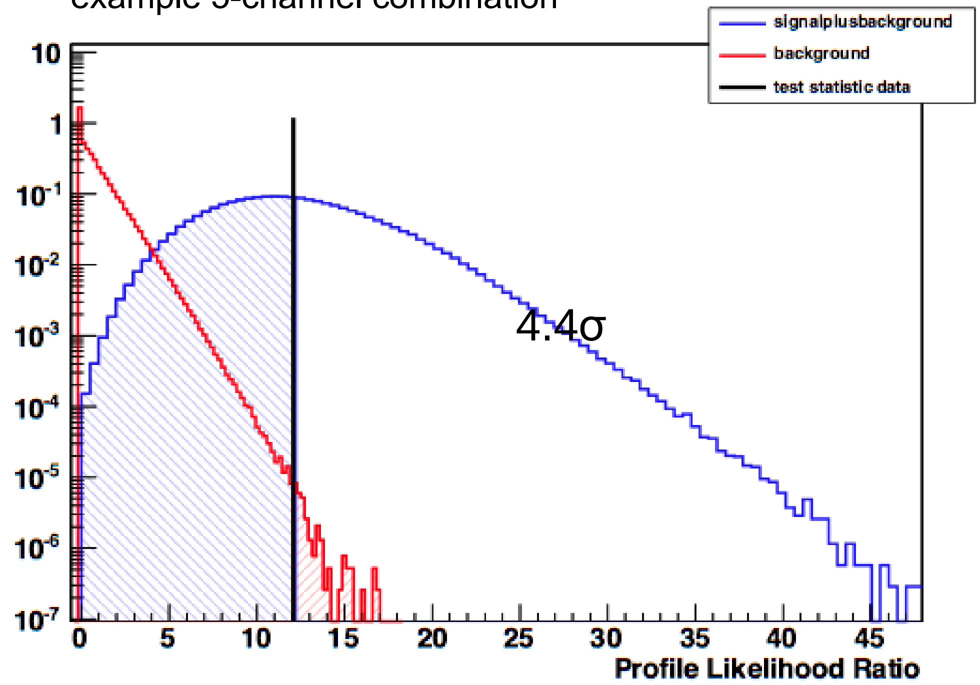
This example has 10 million pseudo-experiments, on a model with combining 5 searches with 50 nuisance parameters using profile likelihood ratio as test statistic

- (~1day on 30 computers)

Now importance sampling is also implemented,

- allows for 1000x speed increase! Still being tested in detail

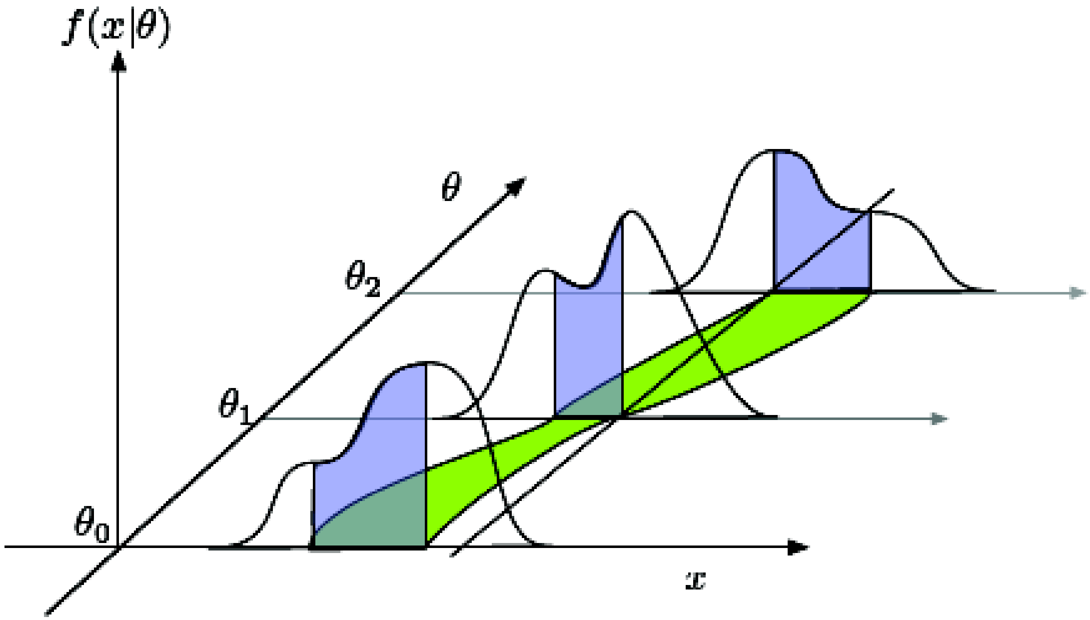
example 5-channel combination



We have also still an older version of HybridCalculator that provides tools to parallelize on batch farms and store/merge contribution to the calculation

NeymanConstruction

- Classical / Frequentist approach to interval estimation
- In order to construct a 1-α CL confidence region on the parameter θ
  - Determine the distribution of the test statistic x for many values of θ
  - Determine each time the 1-α CL region on x
  - Look at the value of x obtained in data
  - The intersection defines the confidence region on parameter θ [θ_1 ; θ_2]



- Different ordering rules possible (shortest interval, central interval, ...)
  - In FeldmanCousins, the ordering rule is a likelihood ratio
    - this sometimes define a 2-sided interval and sometimes a 1-sided limit: no problem of flip-flopping → the method is well adapted to cases close to a physical boundary

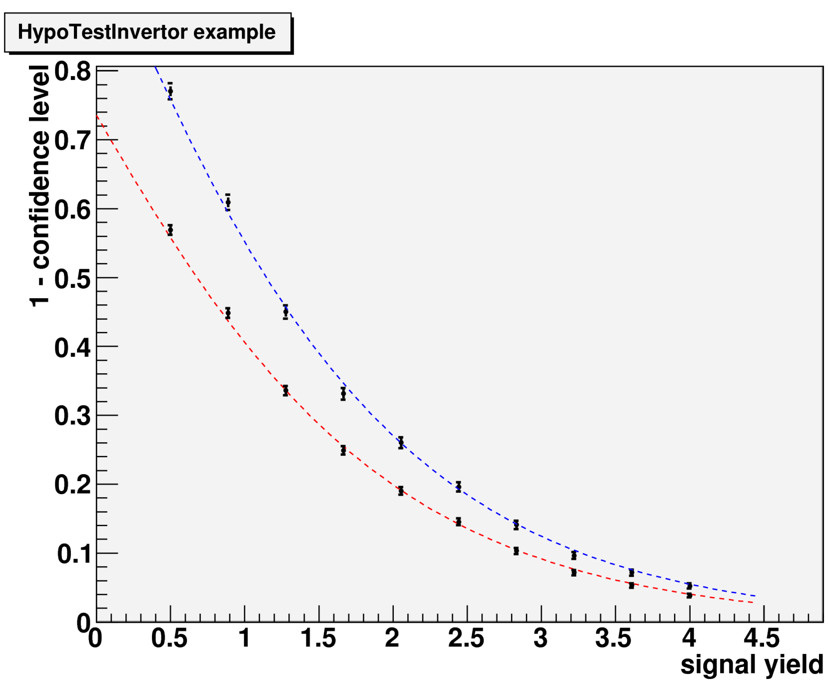
$$R_\mu = \left(x \middle| \frac{L(x|\mu)}{L(x|\hat{\mu})} > k_\alpha \right)$$

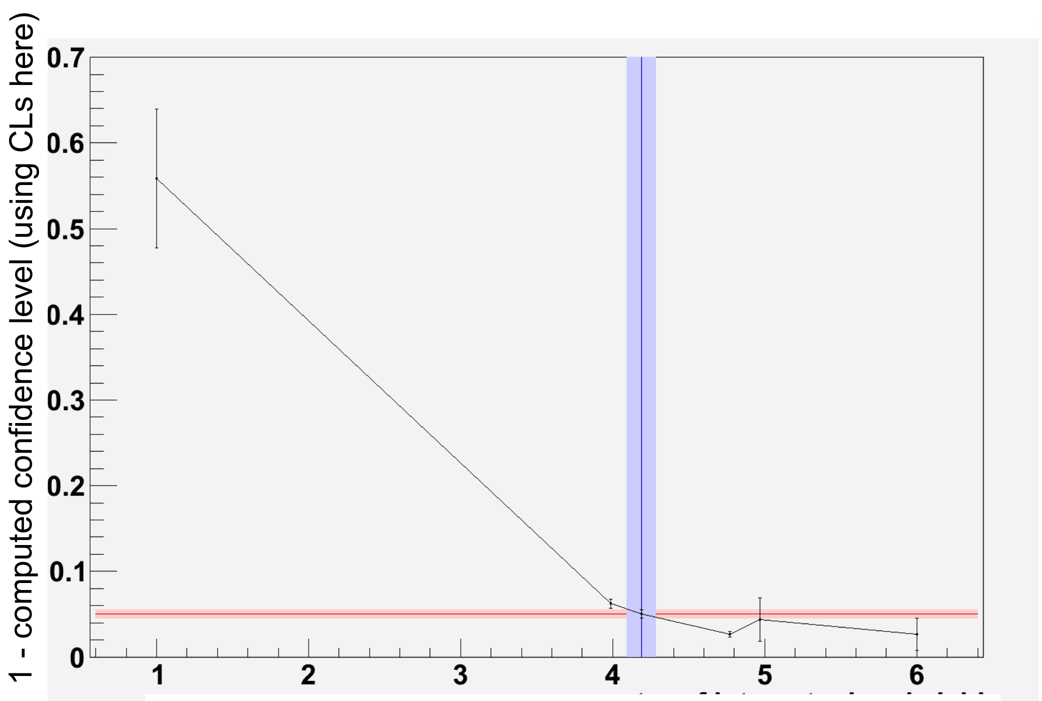
Elements entering frequentist analyses 2

- What is the test statistics?
- How to sample it?
- For intervals: what is the ordering rule?
- For limits, what is the condition that defines the upper bound?
  - p_{SB}: in the pure-frequentist approach
  - $CL_S = p_{SB}/(1-p_B)$: modified-frequentist that some people like because:
    - cures background downward fluctuations (more conservative)
    - in some cases, the same limits are expected as with Bayesian methods using a flat prior on the parameter of interest
    - it is the approach used for Higgs searches at LEP and Tevatron
  - power-constrained: don't quote limits below some threshold defined by a N-σ downward fluctuation of B-only experiments
  - in RooStats the first 2 can be specified as options while the 3rd is easily feasible by hand

HypoTestInverter

- Invert result from the hypothesis test (e.g. from HybridCalculatorOriginal) to compute a limit
  → run many times the calculator scanning the POI
  - can run using a fixed grid or automatically to minimize toy runs
  - obtain an upper limit from the CL_{SB} or CL_{S} curve
  - can estimate also the expected numerical errors





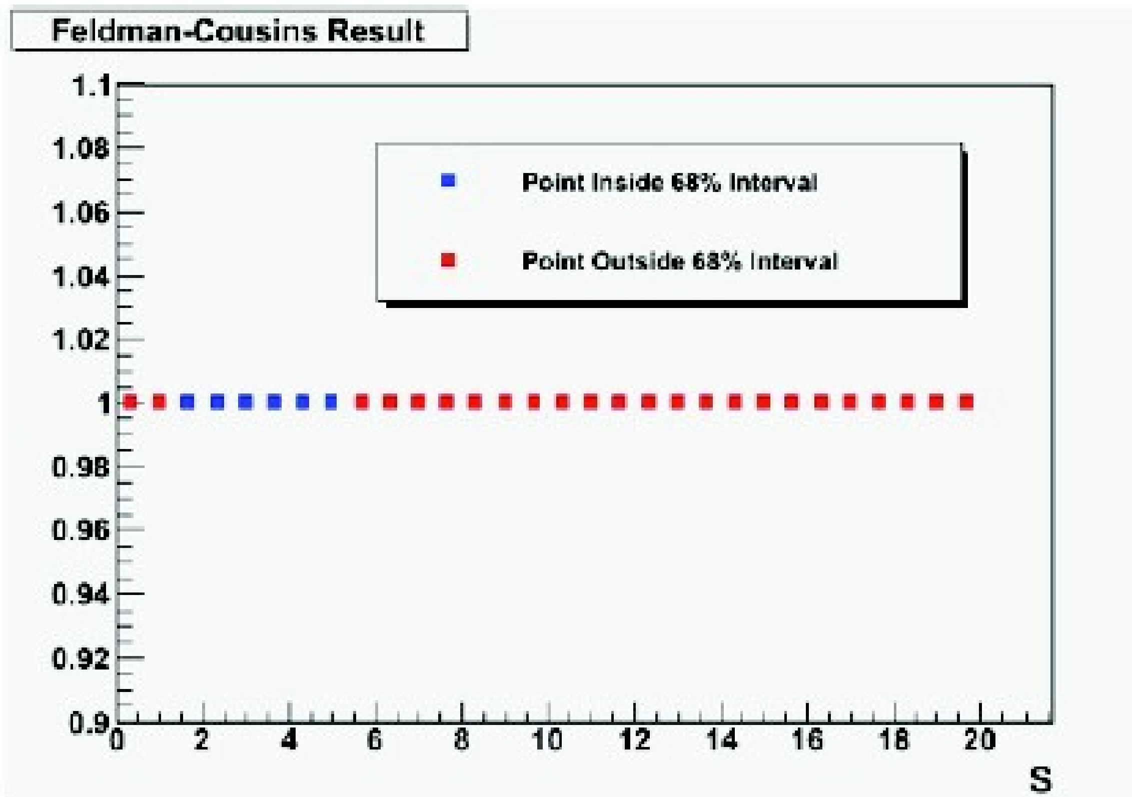
parameter of interest: tested signal yield in SB hypothesis

FeldmanCousins usage

```
RooStats::FeldmanCousins fc;
// set the distribution creator, which encodes the test statistic
fc.SetPdf(model);
fc.SetParameters(parameters);
fc.SetTestSize(0.05); // set size of test
fc.SetData(*data);
fc.UseAdaptiveSampling(true);
// number counting analysis: dataset always has 1 entry with N events observed
fc.FluctuateNumDataEntries(false);
fc.SetNBins(30); // number of points to test per parameter
```

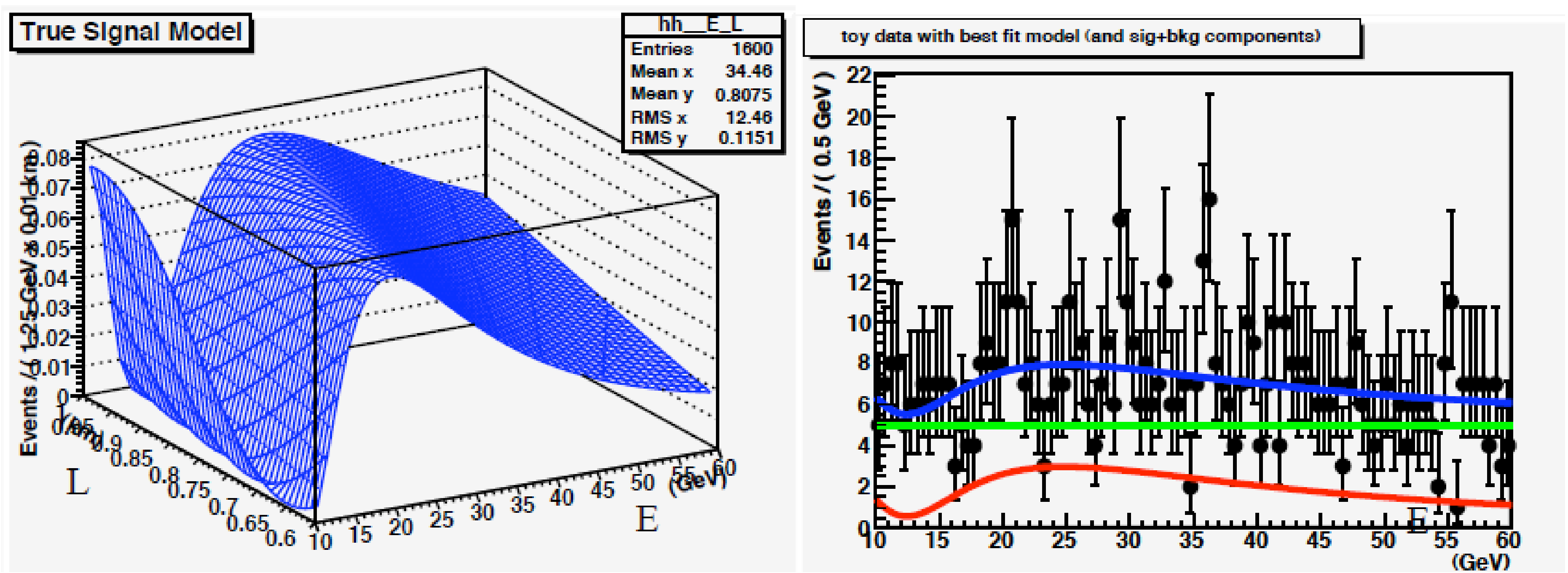
- SetNBins() specifies the number of points to test on the parameter of interest

- Returns a ConfidenceInterval as a PointSetInterval: you can test if a point is inside or outside that interval



Neutrino oscillation example

- Kyle coded up neutrino oscillation experiment based on description in Feldman-Cousins original paper
- Generate toy data at same true parameters and compare RooStats with results in paper



see:

http://root.cern.ch/root/html/tutorials/roostats/rs401d_FeldmanCousins.C.html

- Has also been validated against ROOT's TFeldmanCousins class

Other classes and utilities

- **New: NonCentralChiSquare:** arXiv:1007.1727 outlines use of generalization of Wilks's Thm. called Wald's Thm., which states asymptotic distribution of $\lambda(\mu)$ for $\mu \neq \mu_{true}$ is a non-central χ^2
- **HLFactory**: Simple wrapper around the RooWorkspace
  - Allows to define the likelihood model in a simple text file
    - Separation of physics model and C++ RooStats instructions
    - Added python-like instructions such as "comments" or "include"
  - Simple combination of multiple channels

```
HLFactory myHLF( "myHLF", "models.txt", false );
myHLF.AddChannel("Analysis1","sbModel1","bModel1","Data1");
myHLF.AddChannel("Analysis2","sbModel2","bModel2","Data2");
RooDataSet* combinedData = myHLF.GetTotDataSet();
RooAbsPdf* combinedPDF = myHLF.GetTotSigBkgPdf();
RooCategory* categoryVariable = myHLF.GetTotCategory();
```

- **SPlot**: a technique used to produce weighted plots of an observable distribution
- **BernsteinCorrection (utility to add polynomial corrections)**, utilities specific to number counting analyses, statistical definitions,

HistFactory

Many analyses are based on template histograms (ROOT TH1)

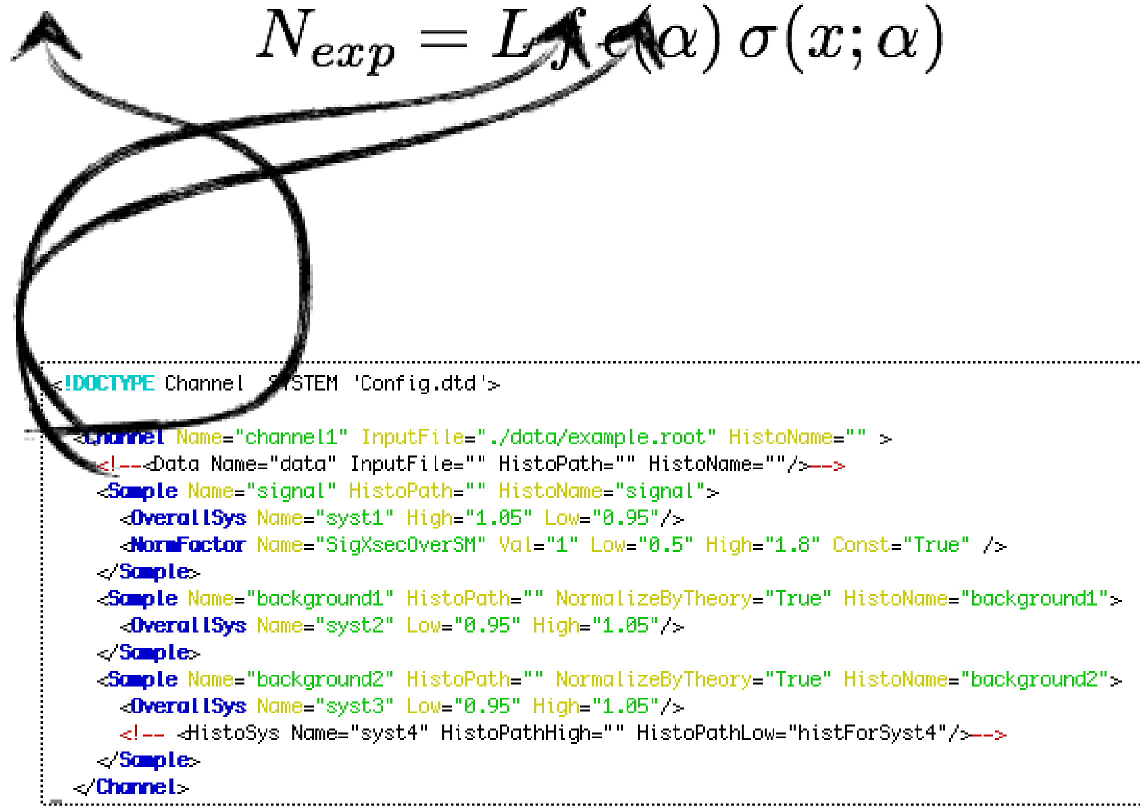
- provide a tool that allows one to use RooStats statistical tools without knowing RooFit’s data modeling language

N = luminosity x f x efficiency x cross-section

$$N_{exp} = L\, f\, e(\alpha)\, \sigma(x;\alpha)$$

In this approach, user provides other templates corresponding to variations of individual systematics

- this is done for each source of systematic and for each signal and background individually
- It is straightforward to provide a combined model for several channels and to identify the same systematic in each channel
- supports Gaussian, Gamma, Lognormal distributions on nuisance parameters



```
<!DOCTYPE Channel  SYSTEM 'Config.dtd'>

 <Channel Name="channel1" InputFile="./data/example.root" HistoName="" >
  <!--<Data Name="data" InputFile="" HistoPath="" HistoName=""/>-->
  <Sample Name="signal" HistoPath="" HistoName="signal">
    <OverallSys Name="syst1" High="1.05" Low="0.95"/>
    <NormFactor Name="SigXsecOverSM" Val="1" Low="0.5" High="1.8" Const="True" />
  </Sample>
  <Sample Name="background1" HistoPath="" NormalizeByTheory="True" HistoName="background1">
    <OverallSys Name="syst2" Low="0.95" High="1.05"/>
  </Sample>
  <Sample Name="background2" HistoPath="" NormalizeByTheory="True" HistoName="background2">
    <OverallSys Name="syst3" Low="0.95" High="1.05"/>
    <!-- <HistoSys Name="syst4" HistoPathHigh="" HistoPathLow="histForSyst4"/>-->
  </Sample>
 </Channel>
```

The user specifies all of these systematics via an XML file and a compiled command line executable parses the XML file to produce the combined model

LHC-HCG

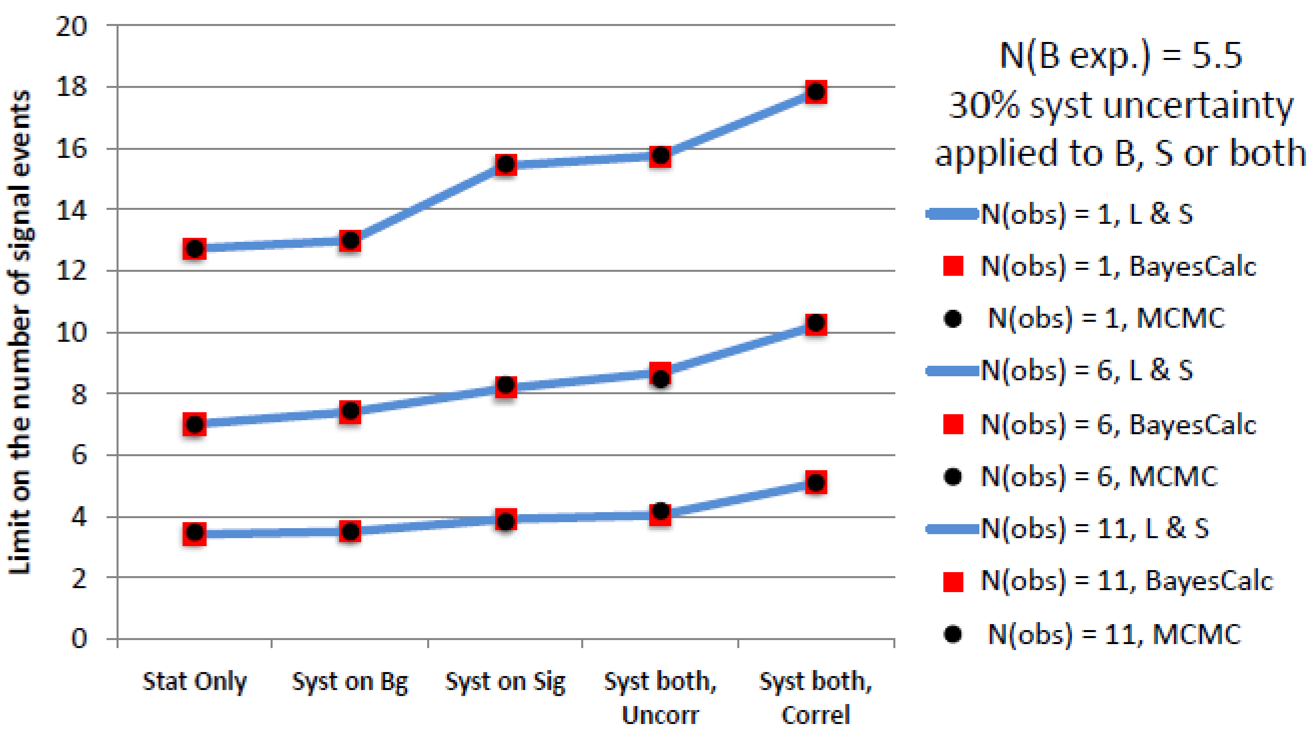
- Newly formed group (12/2010), mandated to prepare and produce a combined Higgs result from LHC
- Initial composition:

role	**ATLAS**	**CMS**
Convenor	B. Murray	V. Sharma
Overal contact	K. Assamangan	A. Korytov
Stat. Committee rep.	E. Gross	G. Schott
Higgs X-section rep.	R. Tanaka	C. Mariotti
+ ATLAS & CMS spokespeople and physics coordinators		
+ participation of experts as and when needed		

- Agressive schedule aiming at a result this summer
- 1^{st} working meeting was devoted to validate RooStats which is the tool designated for the combination

Validations

- There has been validations of RooStats in the past but we are now complementing those with systematic validations based on a realistic combined Higgs analysis case (see also talk from K. Cranmer and http://indico.cern.ch/conferenceDisplay.py?confId=120429)
  - comparision with many external packages (LEP/TEV/private/BAT)



Validations (and beyond)

Initial results do no reveal any indication of bias of the RooStats results for the methods checked so far: GOOD!

- we try to look at all "methods" we might use on the final Higgs analyses

Getting the result right is great but it's also important to:

- get a better control of numerical precision on limits and significances
- identify performance issues (memory leaks, speed bottelnecks)
- consolidate and make RooStats more fail-proof
- complement the document and educate the new users
- possibly develop new tools as needed: reference priors, goodness of fit estimators, tools for MC and coverage studies, look elsewhere effect, ...
  - Open project, new contributors are welcome

Comments adapted from talks of R. Cousins and G. Cowan

- Once the statistics problem is described, various methods can be easily applied and compared
  - Bayesian, Frequentist, Likelihood ratio, “CLs”, ...
- It is recommended / the community can ask the result be shown with one or another method and to study sampling properties
  - if methods agree → important check of robustness
  - if they disagree → we learn something
- Needless to say: Except for particular cases, we don't expect same results since the results are answers to different questions
  - Bayesian methods can have poor frequentist properties
  - Frequentist methods can badly violate likelihood principle
- Speaking at least for CMS, we have definite decision on what method and assumptions within to use:
  - that's one of the items to discuss/agree next for the LHC-HCG

Conclusion

- Code of Atlas and CMS combined and improved to form RooStats
  - RooStats available from ROOT since December 2008 (recommend latest release 5.28, December 2010)
- Allows statistical studies for LHC (and other) analyses
  - Use same implementation of methods
  - Speak common language for comparisons and combination
  - Flexible enough to accommodate all/most cases
  - Most statistical methods one would need are there
- The more users use RooStats the better it will be tested for all sorts of use-cases; user feedback is very valuable: Thank you!

Conclusion (announcements)

- Citation: "The RooStats project", http://arxiv.org/abs/1009.1003 Proceedings of the ACAT2010 Conference
- A RooStats tutorial will take place at CERN this Friday 13-18h:
  - see announcement in the roostats development mailing list and register yourself
- Will be able to support an expert (post-doc or advanced graduate student) to be at CERN in exchange for fraction of time spent in RooFit/RooStats development, validation, maintenance and support within ATLAS/CMS. Let us know if you are interested

Backup slides

RooStats Calculator classes

- **ProfileLikelihoodCalculator**: interval estimation and hypothesis testing
- **BayesianCalculator**: adaptive numerical integration
- **MCMCCalculator**: Bayesian with Markov-Chain Monte Carlo
- **NeymanConstruction**: classical/frequentist interval calculator
- **FeldmanCousins**: Neyman construction with likelihood ratio ordering rule
- **HybridCalculator** and HybridCalculatorOriginal: frequentist hypothesis testing with Bayesian integration of nuisance parameters
- **HypoTestInverter**: inversion of hypothesis tests into a confidence interval

- **BATCalculator**: Bayesian with Markov-Chain Monte Carlo (external but usable as a RooStats class)

Terminology

- **Observables** (or random variables): quantities that are directly measured by an experiment (eg. candidates mass, helicity angle, NNet output) – they form a dataset
- **Model**: based on probability density functions (PDF) that describe one or multiples observables – parametric or non-parametric. PDF are normalized such that their integral over any observable is 1
- **Parameters of interest**: parameters of the model that one wishes to estimate or constrain (eg. particle mass, cross-section)
- **Nuisance parameters**: parameters of the model that are uncertain but not "of interest" (systematics-associated normalization or shape parameters) – treatment of systematic uncertainties varies in different statistical approaches

Documentation and user support

Core developers: K. Cranmer (*Atlas*), L. Moneta (*ROOT*), G. Schott (*CMS*), W. Verkerke (*RooFit*)

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

Documentation:

- RooFit's user's guide: http://root.cern.ch/drupal/content/users-guide (*to be completed*)
- RooStats manual http://root.cern.ch/viewcvs/branches/dev/roostats/roofit/roostats/doc/usersguide/RooStats_UsersGuide.pdf (*in preparation*)
- ROOT reference guide: http://root.cern.ch/root/html/ClassIndex.html
- RooFit and RooStats tutorial macros: http://root.cern.ch/root/html/tutorials
- RooFit interface to the Bayesian Analysis Toolkit (BAT): http://cern.ch/schott/public/BCRooInterface

November tutorials:

- Lecture of L. Lista on statistics: http://indico.cern.ch/conferenceDisplay.py?confId=73545
- Tutorial contents: http://indico.cern.ch/conferenceDisplay.py?confId=72320

RooStats user support:

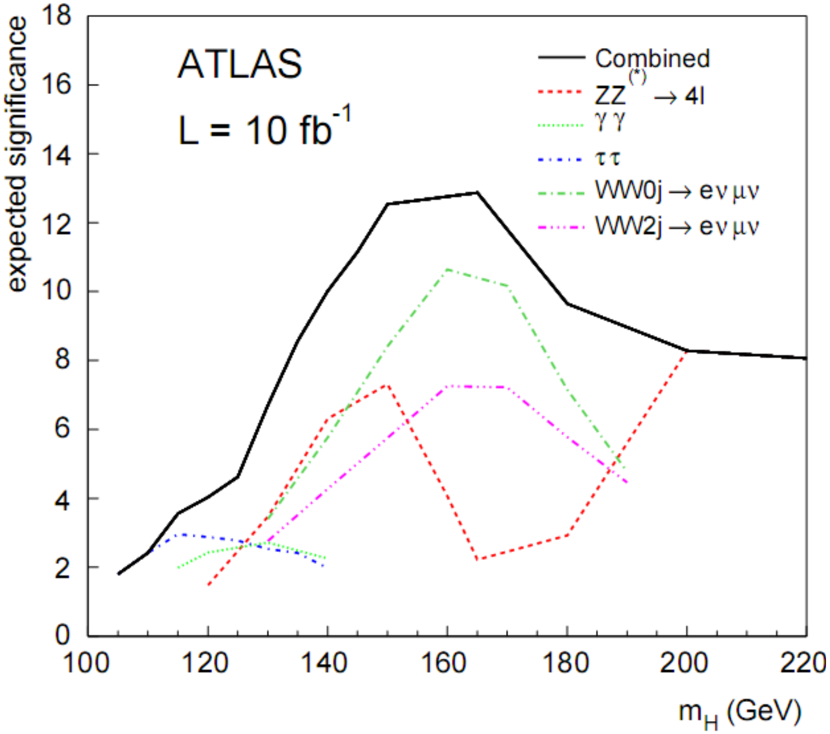
- Request support via ROOT talk forum: http://root.cern.ch/phpBB2/viewforum.php?f=15
  (questions on statistical concepts accepted)
- Submit bugs to ROOT Savannah: https://savannah.cern.ch/bugs/?func=additem&group=savroot
- *Often, posting also a simple self-contained macro reproducing the problem helps a lot*

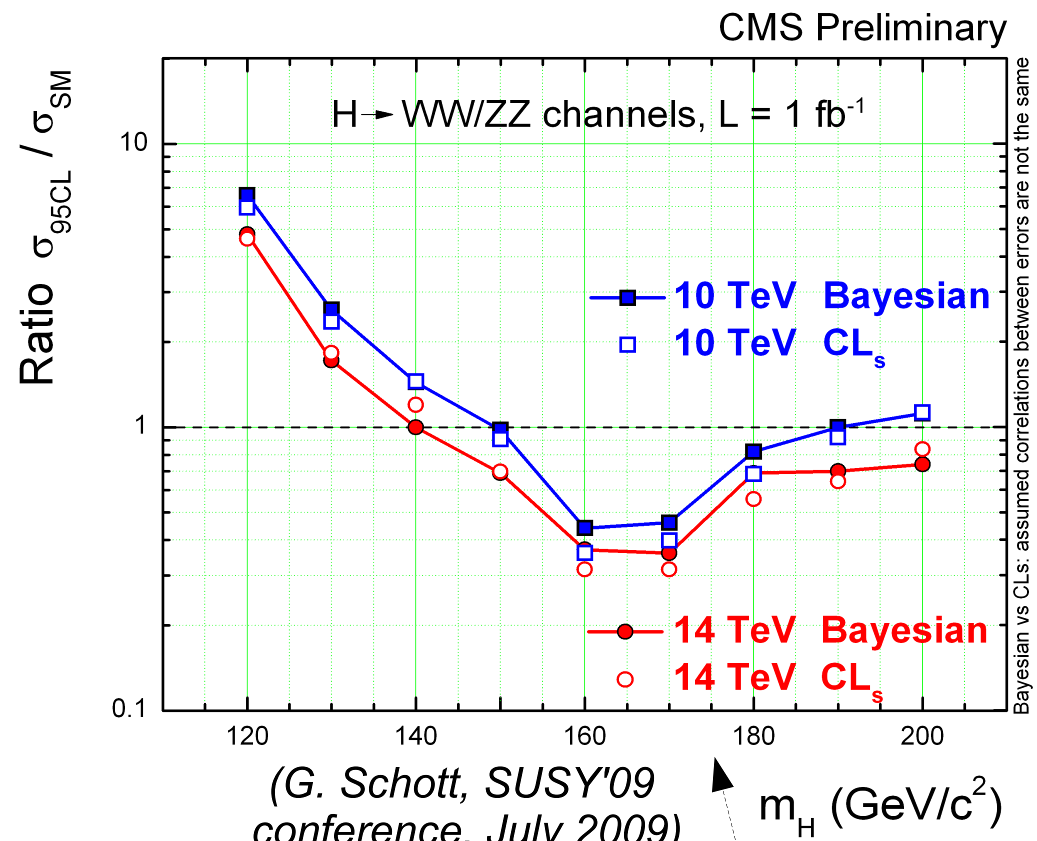
Contacts for statistical questions:

- ATLAS statistics forum: hn-atlas-physics-Statistics@cern.ch (Cowan, Gross *et al*)
  - TWiki: https://twiki.cern.ch/twiki/bin/view/AtlasProtected/StatisticsTools
- CMS statistics committee: (Cousins, Lyons *et al*)
  - via hypernews: hn-cms-statistics@cern.ch or directly: cms-statistics-committee@cern.ch

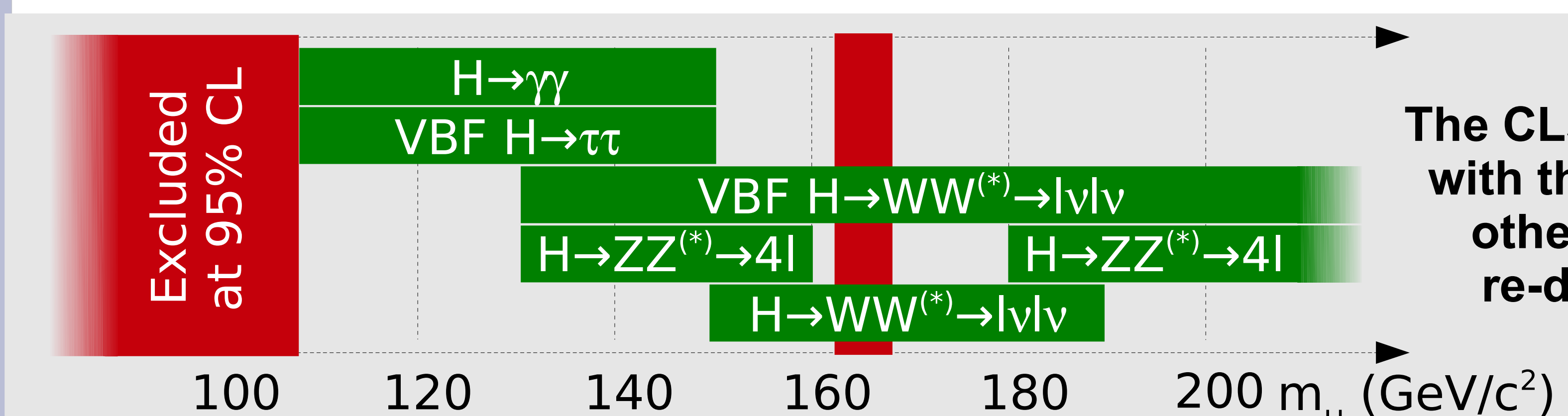
Example application: Higgs at LHC

Median expected exclusion
(CSC report arXiv:0901.0512)





(G. Schott, SUSY'09 conference, July 2009)



The CLs result was obtained with the code of RooStats, other results are being re-done with RooStats

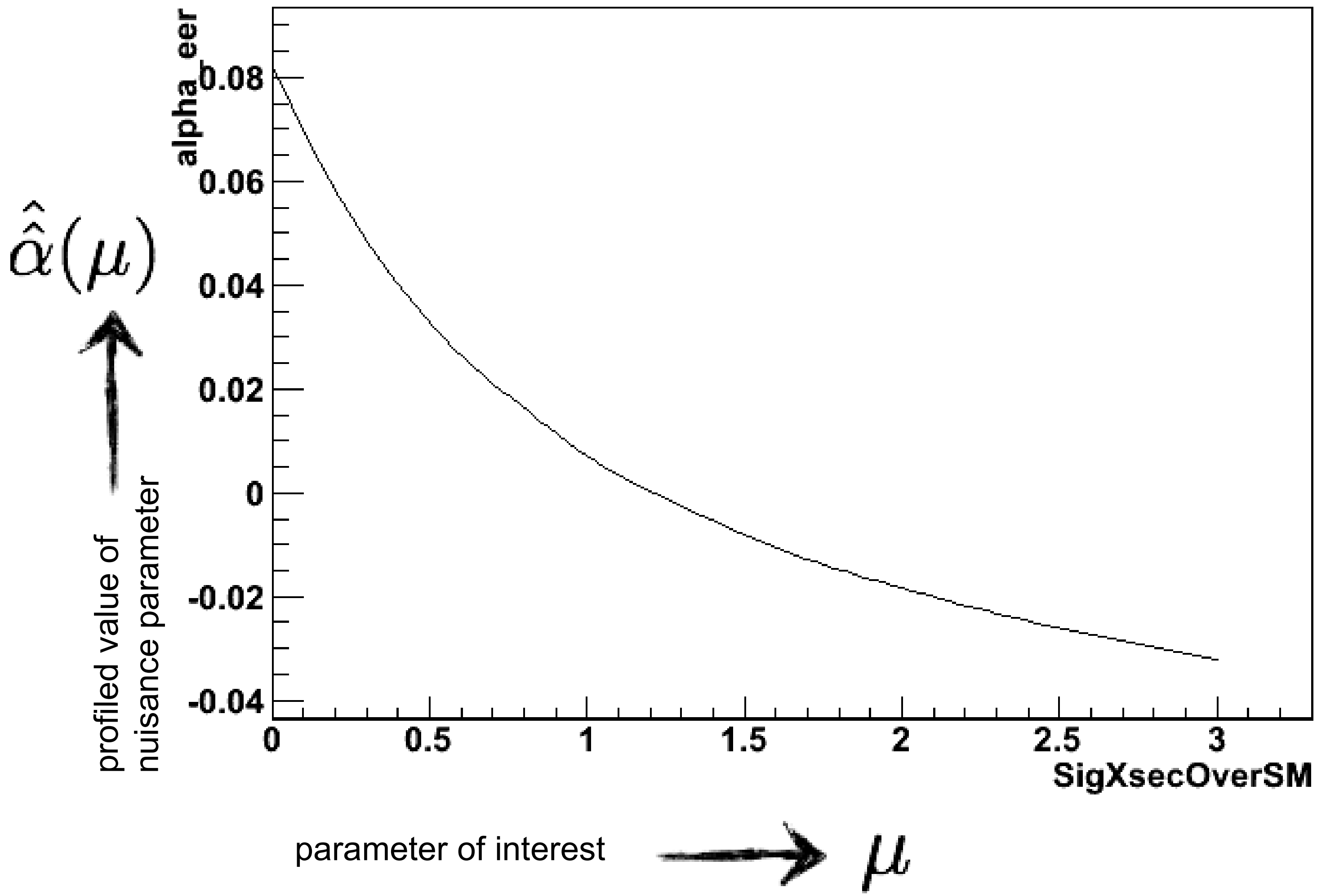
Recipe to run BATCalculator

```
source root_5.28.00/bin/thisroot.sh
wget http://www.mppmu.mpg.de/bat/source/BAT-0.4.1.tar.gz ; tar xvfz BAT-0.4.1.tar.gz
cd BAT-0.4.1 ; ./configure --prefix=${PWD} –with-roostats ; make ; make install
export BATINSTALLDIR=BAT-0.4.1
export LD_LIBRARY_PATH=${LD_LIBRARY_PATH}:${BATINSTALLDIR}/lib
export CPATH=${BATINSTALLDIR}/include:${BATINSTALLDIR}/models
```

```
{
  using namespace RooStats;
  gSystem->Load("libBAT.so"); gSystem->Load("libBATmodels.so");
  gSystem->Load("libBAT.rootmap"); gSystem->Load("libBATmodels.rootmap");
  HLFactory hlf("hlf","hww4ch-1fb-B-mH160.txt.hlf",false);
  RooWorkspace* wks = hlf.GetWs();
  RooArgSet* obsSet = wks->set("observables");
  RooDataSet* data = new RooDataSet("data","",*obsSet);
  data->add(*obsSet); wks->import(*data);
  RooUniform poiPrior("poiPrior","",*wks->set("POI")->first());
  RooProdPdf prior("prior","",poiPrior,*wks->pdf("nuisancePdf"));
  RooArgSet* nuisSet = wks->set("nuisances");
  RooAbsPdf* model = wks->pdf("model_s");
  BATCalculator batc(*data,*model,wks->set("POI"),prior,wks->set("nuisances"));
  batc->SetnMCMC(4000);
  batc->SetTestSize(0.20);  // bug to fix in BATCalculator!!
  RooPlot* plot = batc->GetPosteriorPlot1D(); plot->Draw();
  double upLim = batc.GetOneSidedUperLim();
}
```

ProfileInspector

- New in ROOT 5.28: The **ProfileInspector** is a tool for examining the value of the nuisance parameters (here α) of the minimized -log(likelihood) function as function of the parameter of interest (μ)



NonCentralChiSquare

arXiv:1007.1727, outlines use of generalization of Wilks's Thm. called Wald's Thm., which states asymptotic distribution of $\lambda(\mu)$ for $\mu \neq \mu_{true}$ is a non-central χ^2 with parameter

$$\Lambda_r = \sum_{i,j=1}^{r} (\theta_i - \theta_i') \tilde{V}_{ij}^{-1} (\theta_j - \theta_j')$$

Three forms:

without MathMore, sum of χ2

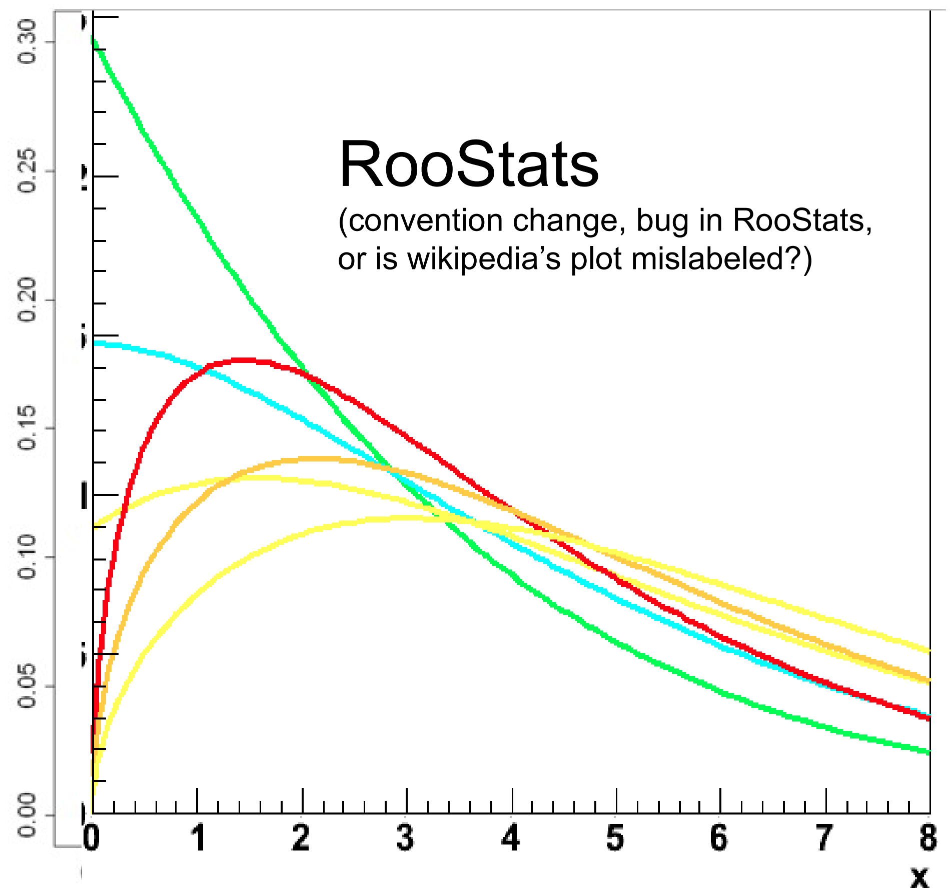
$$f_X(x;k,\lambda) = \sum_{i=0}^{\infty} \frac{e^{-\lambda/2}(\lambda/2)^i}{i!} f_{Y_{k+2i}}(x),$$

with MathMore: k≥2, use Incomplete Bessel

$$f_X(x;k,\lambda) = \frac{1}{2} e^{-(x+\lambda)/2} \left(\frac{x}{\lambda}\right)^{k/4-1/2} I_{k/2-1}(\sqrt{\lambda x})$$

for k<2 confluent hypergeometic functions

$$f_X(x;k,\lambda) = \mathrm{e}^{-\lambda/2}\,{}_0F_1(;k/2;\lambda x/4)\frac{1}{2^{k/2}\Gamma(k/2)}\mathrm{e}^{-x/2}x^{k/2-1}.$$



Test x=5,k=3, Λ=1.5:
RooStats 0.0972573
Matlab, R 0.097257

Markov Chain Monte Carlo

Markov Chain Monte Carlo (MCMC) is a nice technique which will produce a sampling of a parameter space which is proportional to a posterior

- it works well in high dimensional problems
- Metropolis-Hastings Algorithm: generates a sequence of points $\{\vec{\alpha}^{(t)}\}$
  - Given the likelihood function $L(\vec{\alpha})$ & prior $P(\vec{\alpha})$, the posterior is proportional to $L(\vec{\alpha}) \cdot P(\vec{\alpha})$
  - propose a point $\vec{\alpha}'$ to be added to the chain according to a proposal density $Q(\vec{\alpha}'|\vec{\alpha})$ that depends only on current point $\vec{\alpha}$
  - if posterior is higher at $\vec{\alpha}'$ than at $\vec{\alpha}$, then add new point to chain
  - else: add $\vec{\alpha}'$ to the chain with probability

$$\rho = \frac{L(\vec{\alpha}') \cdot P(\vec{\alpha}')}{L(\vec{\alpha}) \cdot P(\vec{\alpha})} \cdot \frac{Q(\vec{\alpha}|\vec{\alpha}')}{Q(\vec{\alpha}'|\vec{\alpha})}$$

  - (appending original point $\vec{\alpha}$ with complementary probability)
- RooStats works with any $L(\vec{\alpha})$ $P(\vec{\alpha})$
- Can use any RooFit PDF as proposal function $Q(\vec{\alpha}'|\vec{\alpha})$

Work done primarily by Kevin Belasco (Princeton)

Kyle Cranmer (NYU) RooStats Pilot Tutorial Oct 15-16, 2009