NEW YORK UNIVERSITY

RooFit/RooStats Tutorial

Statistics School

Hamburg

April 2-5, 2012

Sven Kreiss (NYU)

Lorenzo Moneta (CERN)

Introduction

Shared Folder: Machine → Settings → Shared Folders

- ➡ https://indico.desy.de/internalPage.py?pageId=1&confId=5065
- ➡ Linux: run inside VM: sudo mount -t vboxsf -o uid=1001,gid=1001 hosthome /mnt
- ➡ Mac: use automount, type “mount” and search for your folder to see where it is mounted, run as root

login/pwd: school / School12

Announcements

Collaborative project to provide and consolidate advanced statistical tools needed by LHC experiments.

Joint contribution from ATLAS, CMS, ROOT and RooFit: developments over sighted by ATLAS and CMS statistics committees.

Current Developers: K. Cranmer, G. Lewis, S. Kreiss (ATLAS), G. Schott, G. Kukartsev (CMS), Lorenzo Moneta (ROOT & CMS), Wouter Verkerke (RooFit & ATLAS), A. Lazzaro (OpenLab)
Contributions from: K. Belasco, A. De Cosa, M. Pellicioni, D. Piparo, G. Petrucciani, S. Schmitz, M. Wolf, M. Baak

Included since ROOT v5.22; RooStats is developing fast and the latest stable version of ROOT is recommended: currently v5.32.02

Example macros in **$ROOTSYS/tutorials/roostats**

ATLAS-only email list: **atlas-phys-stat-root@cern.ch**

CMS-only email list: **hn-cms-statistics@cern.ch**

Citation: "The RooStats project", http://arxiv.org/abs/1009.1003 Proceedings of the ACAT2010 Conference

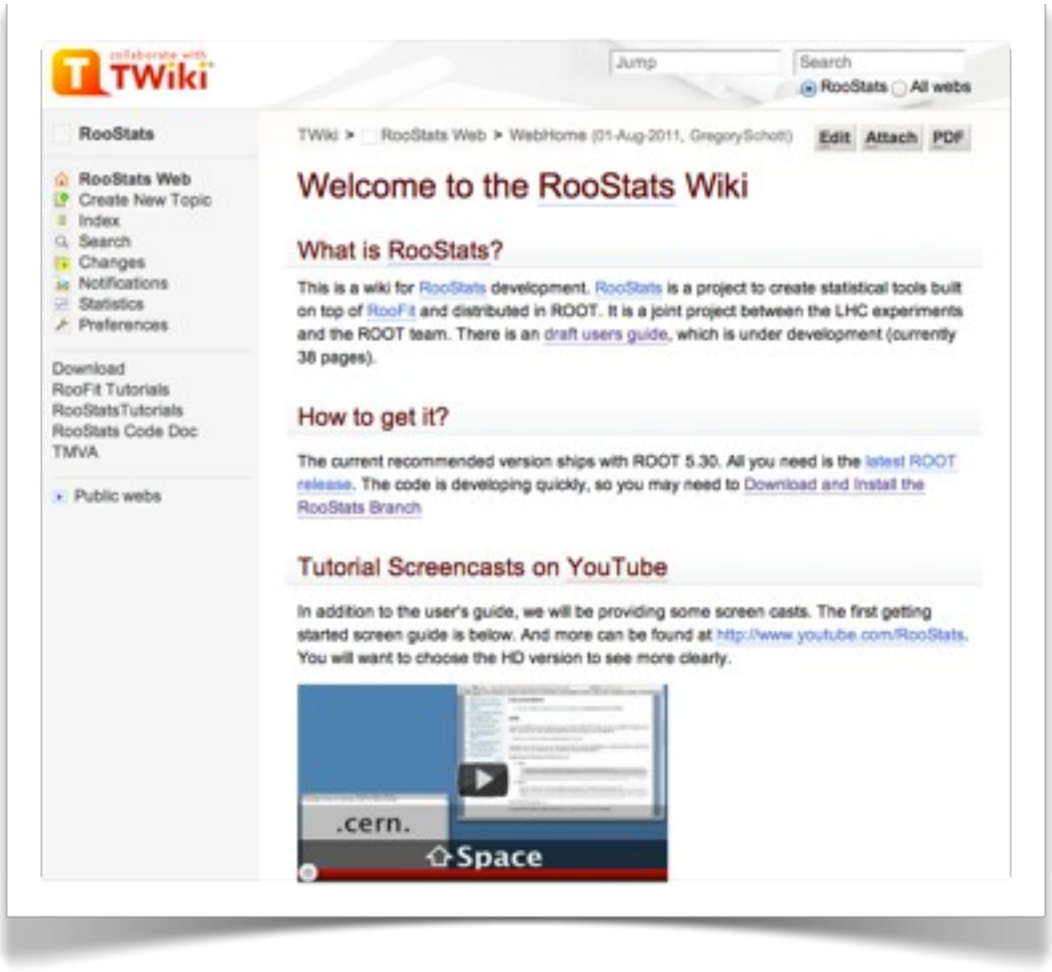
Sources

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

- ➡ your primary source and repository of links to other sources!

ATLAS statistics recommendations:
https://twiki.cern.ch/twiki/bin/view/AtlasProtected/ATLASStatisticsFAQ

CMS statistics recommendations:
https://twiki.cern.ch/twiki/bin/viewauth/CMS/StatisticsCommittee



Welcome to the RooStats Wiki

What is RooStats?

This is a wiki for RooStats development. RooStats is a project to create statistical tools built on top of RooFit and distributed in ROOT. It is a joint project between the LHC experiments and the ROOT team. There is an draft users guide, which is under development (currently 38 pages).

How to get it?

The current recommended version ships with ROOT 5.30. All you need is the latest ROOT release. The code is developing quickly, so you may need to Download and Install the RooStats Branch

Tutorial Screencasts on YouTube

In addition to the user's guide, we will be providing some screen casts. The first getting started screen guide is below. And more can be found at http://www.youtube.com/RooStats. You will want to choose the HD version to see more clearly.

Screencast tutorials: http://www.youtube.com/RooStats

User’s Guide: http://root.cern.ch/viewcvs/branches/dev/roostats/roofit/roostats/doc/usersguide/RooStats_UsersGuide.pdf

Code documentation via ROOT: http://root.cern.ch/root/html/ClassIndex.html#idx17

RooStats Goals

- Provide a common framework for statistical calculations
  - work on arbitrary models and datasets
  - implement most accepted techniques
    - frequentists, Bayesian and likelihood based tools
  - possible to easy compare different statistical methods
  - provide utility for combinations of results
  - using same tools across experiments facilities combinations of results

Statistical Applications

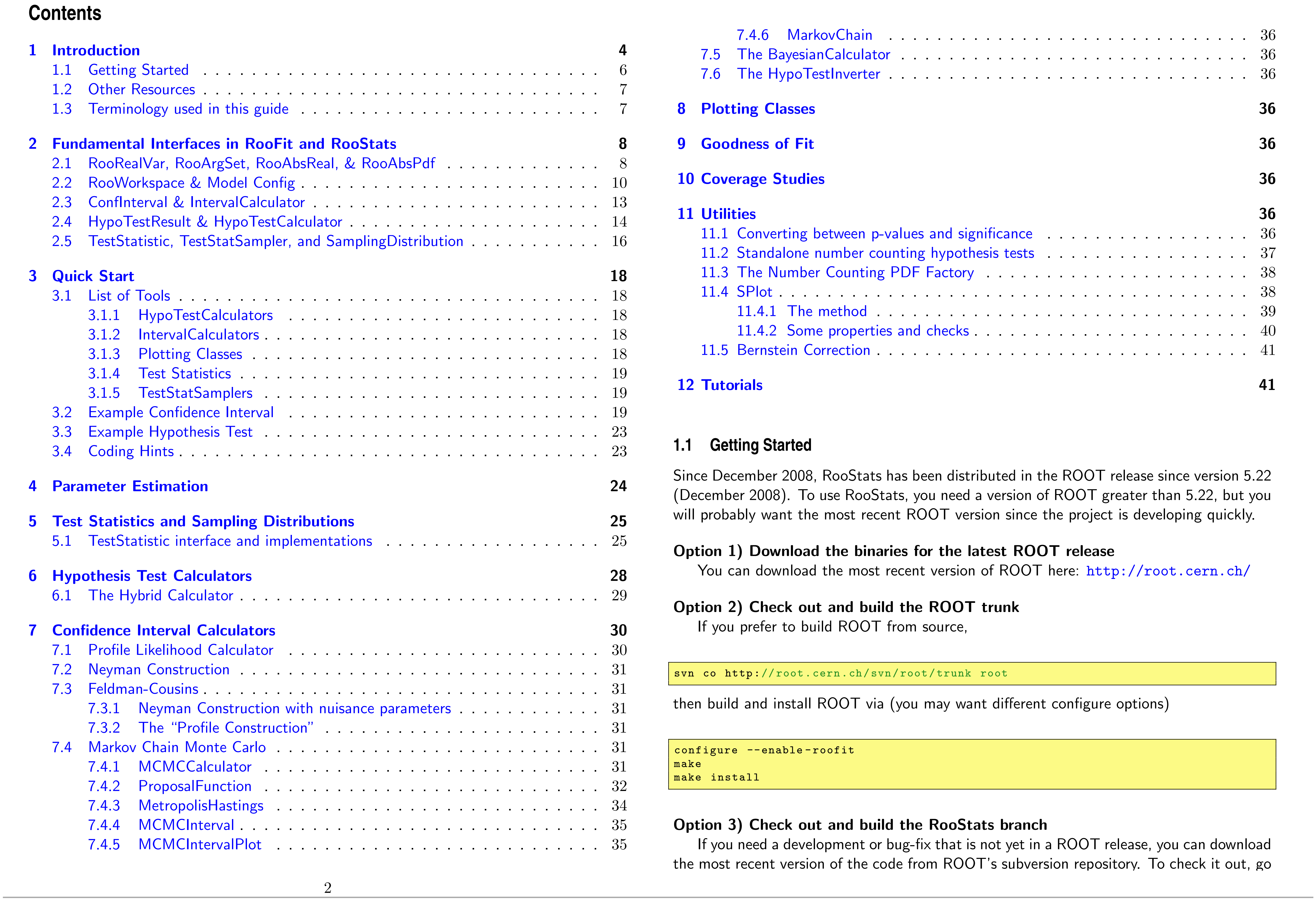
- Common purposes:
  - point estimation: determine the best estimate of a parameter
  - estimation of confidence (credible) intervals
    - multi-dimensional contours or just a lower/higher limit
  - hypothesis tests: evaluation of p-value for one or multiple hypotheses (significance)
  - goodness-of-fit: how well a model describes the data
- Analysis combination:
  - Performed at analysis level: full information available to treat correlations
- For these things RooStats can help you

RooStats Design

- Built on top of RooFit
  - generic and convenient description of models
    - probability density function or likelihood functions
  - easily generation of models (workspace factory)
  - tools for model combinations (e.g. simultaneous pdf)
  - possibility to persistify models in files using the RooFit RooWorkspace class
    - sharing and digital publishing of results

- workspace models are the inputs to all RooStats statistical tools

User’s Guide



Contents

1.1 Getting Started

Since December 2008, RooStats has been distributed in the ROOT release since version 5.22 (December 2008). To use RooStats, you need a version of ROOT greater than 5.22, but you will probably want the most recent ROOT version since the project is developing quickly.

Option 1) Download the binaries for the latest ROOT release

You can download the most recent version of ROOT here: http://root.cern.ch/

Option 2) Check out and build the ROOT trunk

If you prefer to build ROOT from source,

```
svn co http://root.cern.ch/svn/root/trunk root
```

then build and install ROOT via (you may want different configure options)

```
configure --enable-roofit
make
make install
```

Option 3) Check out and build the RooStats branch

If you need a development or bug-fix that is not yet in a ROOT release, you can download the most recent version of the code from ROOT’s subversion repository. To check it out, go

Current list of Calculators

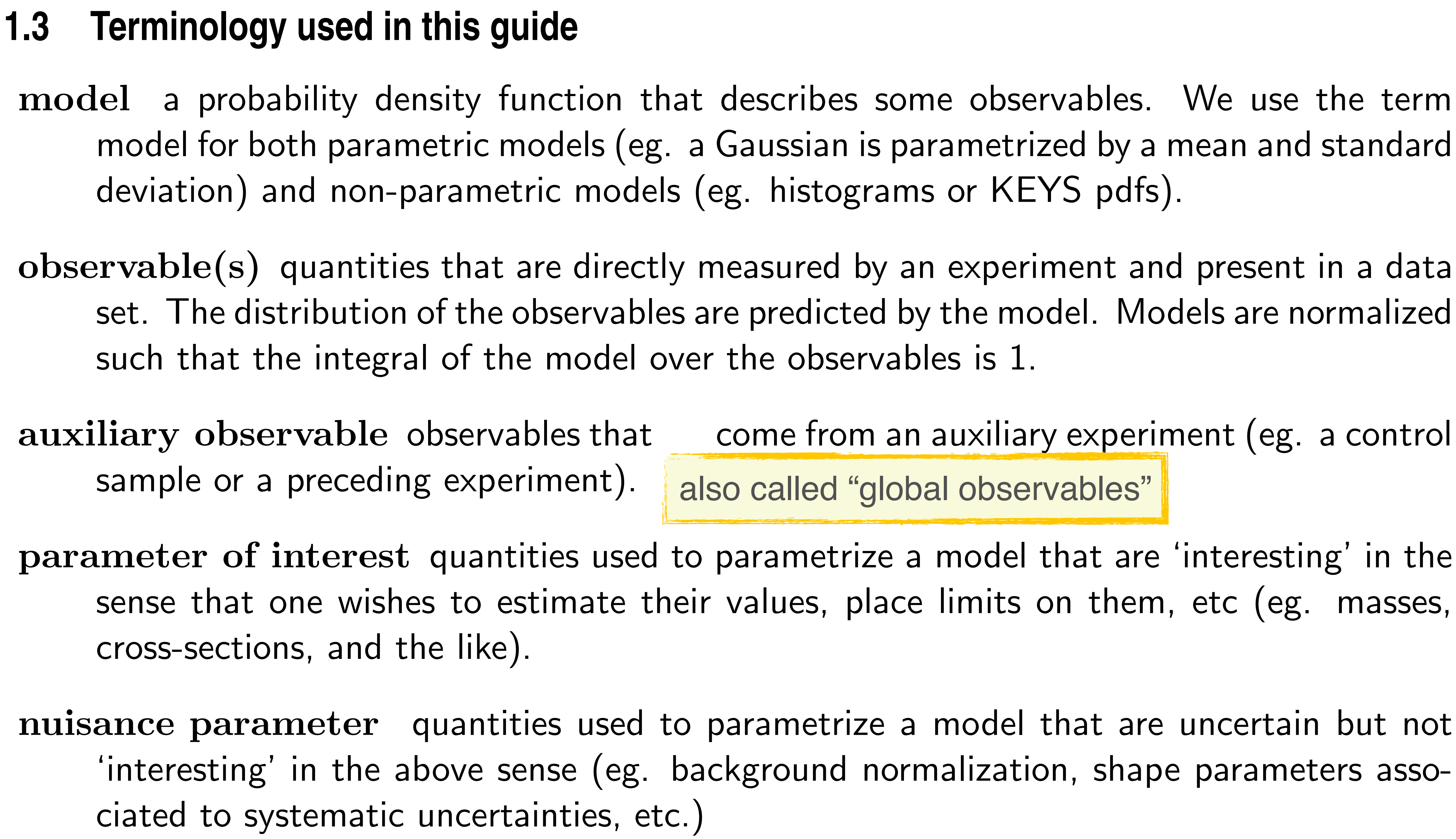
HypoTestCalculators

- FrequentistCalculator
  - frequentist calculation (profile nuisance parameters)
- HybridCalculator
  - hybrid Bayes-Frequentist calculation (marginalize nuisance parameters)
- ProfileLikelihoodCalculator
  - the method of MINUIT/MINOS, based on Wilks' theorem

IntervalCalculators

- HypoTestInverter
  - takes a HypoTestCalculator and forms an IntervalCalculator
- ProfileLikelihoodCalculator
  - method of MINUIT/MINOS, based on Wilks' theorem
- NeymanConstruction
  - general purpose Neyman Construction class, highly configurable: choice of TestStatistic, TestStatSampler (defines ensemble/conditioning), integration boundary (upper, lower, central limits), and parameter points to scan
- FeldmanCousins
  - specific configuration of NeymanConstruction for Feldman-Cousins (generalized for nuisance parameters)
- MCMCCalculator
  - Bayesian Markov Chain Monte Carlo (Metropolis Hastings), proposal function is highly customizable
- BayesianCalculator
  - Bayesian posterior calculated via numeric integration routines, currently only supports one parameter

Terminology (from Kyle Cranmer)



1.3 Terminology used in this guide

model a probability density function that describes some observables. We use the term model for both parametric models (eg. a Gaussian is parametrized by a mean and standard deviation) and non-parametric models (eg. histograms or KEYS pdfs).

observable(s) quantities that are directly measured by an experiment and present in a data set. The distribution of the observables are predicted by the model. Models are normalized such that the integral of the model over the observables is 1.

auxiliary observable observables that come from an auxiliary experiment (eg. a control sample or a preceding experiment).

also called "global observables"

parameter of interest quantities used to parametrize a model that are 'interesting' in the sense that one wishes to estimate their values, place limits on them, etc (eg. masses, cross-sections, and the like).

nuisance parameter quantities used to parametrize a model that are uncertain but not 'interesting' in the above sense (eg. background normalization, shape parameters associated to systematic uncertainties, etc.)

ModelConfig Class

- `ModelConfig` class input to all Roostats calculators
  - contains a reference to the RooFit workspace class
  - provides the workspace meta information needed to run RooStats calculators
    - pdf of the model stored in the workspace
    - what are observables (needed for toy generations)
    - what are the parameters of interest and the nuisance parameters
    - global observables (from auxiliary measurements) for frequentist calculators
    - prior pdf for the Bayesian tools
  - ModelConfig can be imported in workspace for storage and later retrieval

Building ModelConfig Class

- ModelConfig must be built after having the workspace
- Specifies names for all the components which are present in the workspace

```
//specify components of model for statistical tools
ModelConfig  modelConfig("G(x|mu,1)");
modelConfig.SetWorkspace(workspace);
//set components using the name of ws objects
modelConfig.SetPdf( "normal");
modelConfig.SetParameterOfInterest("poi");
modelConfig.SetObservables("obs");
```

- Alternatively ModelConfig can be used to import the components directly into the workspace

```
// set and to import into workspace
modelConfig.SetPdf( *pdf);
```

- Some tools (Bayesian) require to specify prior pdf

```
//Bayesian tools would also need a prior
modelConfig.SetPriorPdf( "prior");
```

- ModelConfig can be imported in workspace to be then stored in a file

```
//can import modelConfig into workspace too
workspace.import(*modelConfig);
```

Profile Likelihood Calculator

- Method based on properties of the likelihood function
- Profile likelihood function:

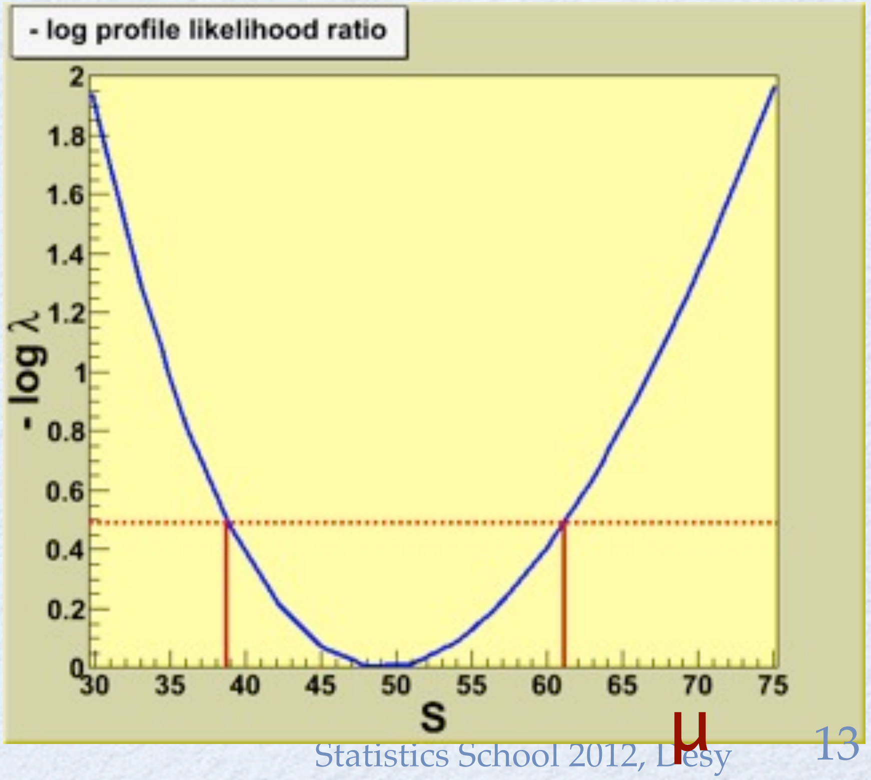
$$\lambda(\mu) = \frac{L(x|\mu, \hat{\hat{\nu}})}{L(x|\hat{\mu}, \hat{\nu})}$$

→ maximize w.r.t nuisance parameters ν and fix POI μ

→ maximize w.r.t. all parameters

λ is a function of only the parameter of interest μ

- Uses asymptotic propertis of eλ based on Wilks' theorem:
  - Taylor expansion of logλ around the minimum:
  - ➔ -2logλ is a parabola (λ is a gaussian function)
  - ➔ interval on μ from logλ values
- Method of MINUIT/MINOS
  - lower/upper limits for 1D
  - contours for 2 parameters



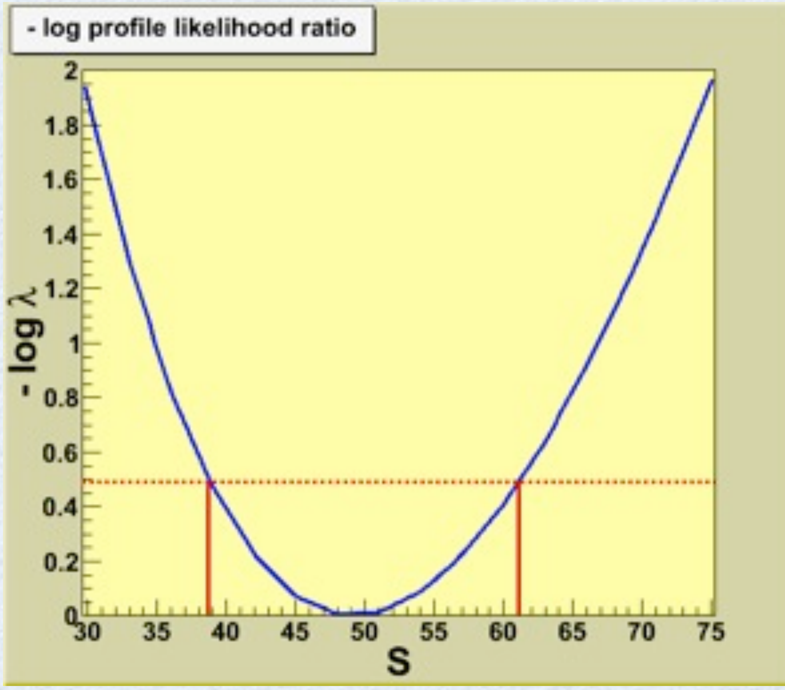
Usage of Profile Likelihood Calculator

```
// create the class using data and model
ProfileLikelihoodCalculator plc(*data, *model, *POI);

// set the confidence level
plc.SetConfidenceLevel(0.683);

// compute the interval
LikelihoodInterval* interval = plc.GetInterval();
double lowerLimit = interval->LowerLimit(*S);
double upperLimit = interval->UpperLimit(*S);

// plot the interval
LikelihoodIntervalPlot plot(interval);
plot.Draw();
```



- For one-dimensional intervals:
  - 68% CL (1 σ) interval : $\Delta\log\boldsymbol{\lambda} = \mathbf{0.5}$
  - 95% CL interval : $\Delta\log\boldsymbol{\lambda} = 1.96$
- **`LikelihoodIntervalPlot`** can plot the 2D contours

Example: Bayesian Analysis

- **`RooStats`** provides classes for
  - marginalize posterior and estimate credible interval

posterior probability

likelihood function

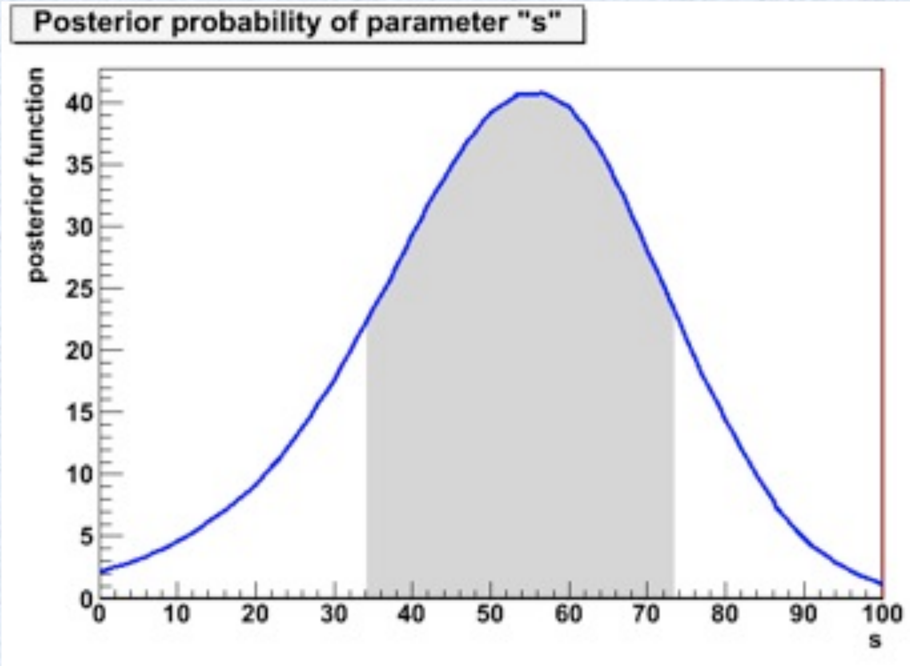
prior probability

nuisance parameters marginalization

$$\underset{\text{POI}}{P(\mu}|\underset{\text{data}}{x)} = \frac{\int L(x|\mu,\nu)\Pi(\mu,\nu)d\nu}{\underbrace{\int\int L(x|\mu,\nu)\Pi(\mu,\nu)d\mu d\nu}_{\text{normalisation term}}}$$

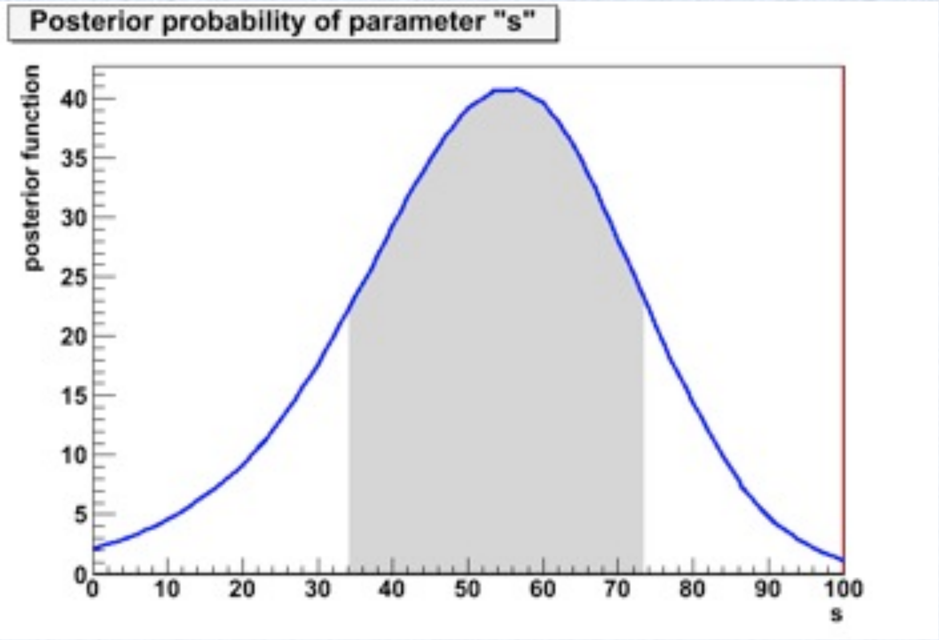
Bayesian Theorem

  - support for different integration algorithms:
    - adaptive (numerical)
    - MC integration
    - Markov-Chain
  - can work with models with many parameters (e.g few hundreds)



Bayesian Classes

- **`BayesianCalculator`** class
  - posterior and interval estimation using numerical integration
  - working only for one parameter of interest but can integrate many nuisance parameters
  - support for different integration algorithms, using **`BayesianCalculator::SetIntegrationType`**
    - adaptive numerical (default type), working only for few nuisances (< 10)
    - Monte Carlo integration (PLAIN, MISER, VEGAS)
    - TOYMC : sampling toys from nuisance pdf's (requires not-uniform nuisance pdf but can work with many parameters)
  - can compute central interval or one-sided interval (upper limit) or a shortest interval (SetCentralInterval)
  - provide plot of posterior and interval



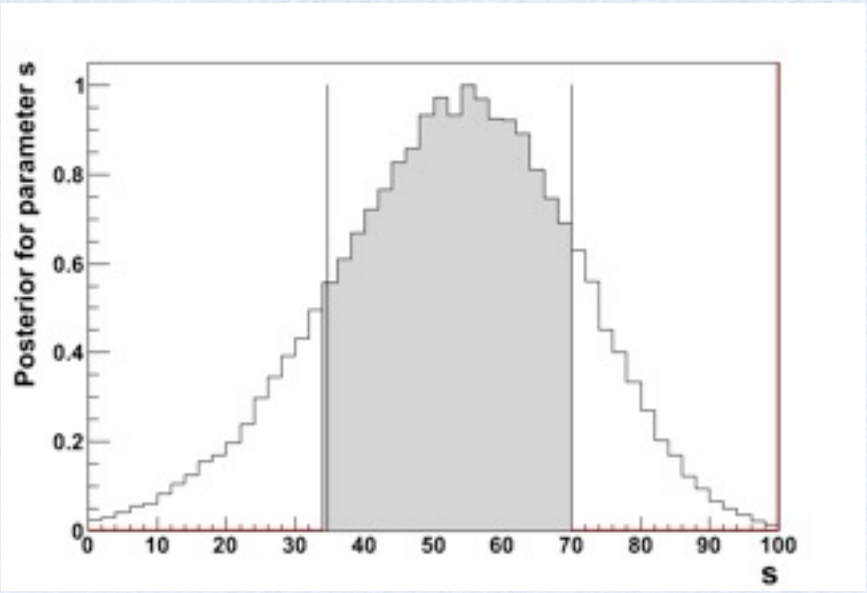
Example: 68% CL central interval

```
BayesianCalculator bc(data, model);
bc.SetConfidenceLevel(0.683);
bc.SetLeftSideTailFraction(0.5);
bc.SetIntegrationType("ADAPTIVE");
SimpleInterval* interval = bc.GetInterval();
double lowerLimit = interval->LowerLimit();
double upperLimit = interval->UpperLimit();
RooPlot * plot = bc.GetPosteriorPlot();
plot->Draw();
```

MCMC Calculator

- **`MCMCCalculator`** class
  - integration using Markov-Chain Monte Carlo (Metropolis Hastings algorithm)
  - can deal with more than one parameter of interest
  - can work with many nuisance parameters
    - e.g. used in Higgs combination with more than 300 nuisances
  - possible to specify ProposalFunction
    - multivariate Gaussian from fit result
    - Sequential proposal
  - can visualize posterior and also the chain result

MCMCCalculator



```
MCMCCalculator mc(data, model);
mc.SetConfidenceLevel(0.683);
mc.SetLeftSideTailFraction(0.5);
SequentialProposal sp(0.1);
mc.SetProposalFunction(sp);
mc.SetNumIters(1000000);
mc.SetNumBurnInSteps(50);
MCInterval* interval = bc.GetInterval();
RooRealVar * s = (RooRealVar*)
model.GetParametersOfInterest()->find(“s”);
double lowerLimit = interval->LowerLimit(*s);
double upperLimit = interval->UpperLimit(*s);
MCMCIntervalPlot plot(*interval);
plot.Draw();
```

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

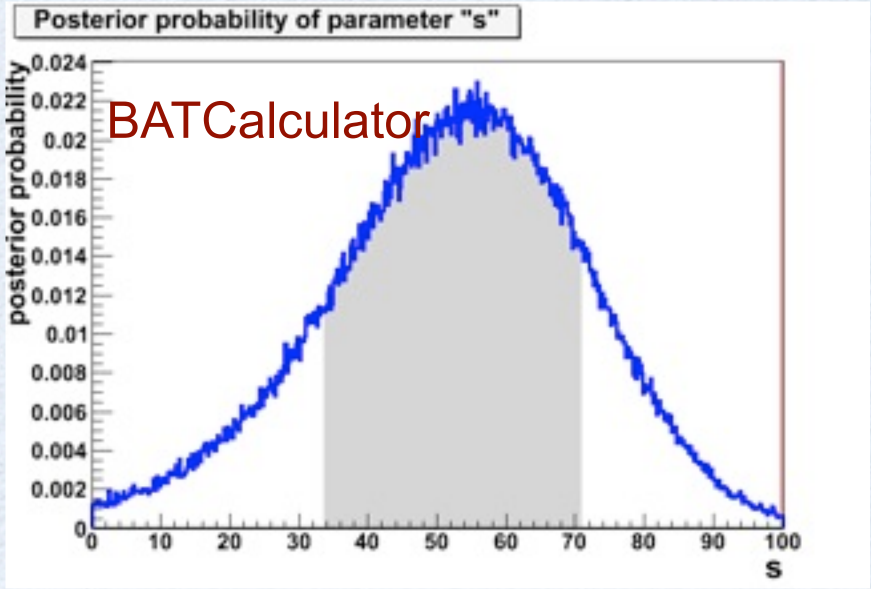
$$\rho = \frac{L(\vec{\alpha}') \cdot P(\vec{\alpha}')}{L(\vec{\alpha}) \cdot P(\vec{\alpha})} \cdot \frac{Q(\vec{\alpha}|\vec{\alpha}')}{Q(\vec{\alpha}'|\vec{\alpha})}$$

  - (appending original point $\vec{\alpha}$ with complementary probability)
- RooStats works with any $L(\vec{\alpha}), P(\vec{\alpha})$
- ~~Since last week~~: can use any RooFit PDF as proposal function $Q(\vec{\alpha}'|\vec{\alpha})$

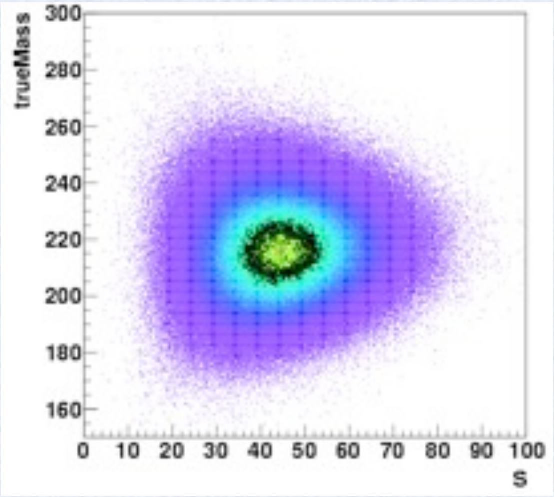
Work done primarily by Kevin Belasco, a Princeton undergraduate I'm working with.

BAT Calculator

- **BATCalculator** class
  - developed by S. Schmitz & G. Schott
  - provided by the BAT package (not part of Roostats)
    A. Caldwell, D. Kollar, K. Kröninger, Comp. Physics Comm. 180 (2009) 2197
    see also http://www.mppmu.mpg.de/bat/
  - valuable alternative for cross-checks
  - various options for controlling the Markov chain
  - similar interface as other RooStats Bayesian calculator
    - but requires to load first libBAT to use it



```
gSystem->Load(“libBAT”);
BatCalculator bc(data, model);
batc->SetnMCMC(500000);
MCInterval* interval = bc.GetInterval();
```



BATCalculator for a 2-dim problem