GEORG-AUGUST-UNIVERSITÄT
GÖTTINGEN

The Bayesian Analysis Toolkit (BAT) – a complex MCMC application

MC Methods in Advanced Statistics Applications and Data Analysis
Munich, Nov. 18th – 22nd 2013

Kevin Kröninger

The BAT men: Frederik Beaujean,
Allen Caldwell, Daniel Greenwald,
Daniel Kollar, Kevin Kröninger

Introduction to BAT

- **Aims**
  - Provide a flexible and modular **framework for statistical models** in context of Bayesian interpretation
  - Provide a **set of (mostly numerical) methods** to solve data-analysis problems
    (parameter estimation, limit setting, model comparison, goodness-of-fit tests, etc.)

- **Scope**
  - Developed in experimental particle-physics community
    (explains choice of C++ and ROOT-dependence)
  - Extended to other fields of research
    (phenomenology, medicine, astroparticle physics, etc.)

- **Requirements and solutions:**
  - Requirement: phrase arbitrary models and use data sets
    - **C++ library** based on ROOT
    - Models inherit from **base classes**
    - **Easy to interface** to any existing code
      (interesting for complex fitting, e.g., fits of CKM matrix, cosmological parameters)
  - Requirement: perform data analysis tasks
    - Graphical output via **ROOT** core functionality
    - Point estimation done using **Minuit** and **Simulated Annealing**
    - Interval estimation and uncertainty propagation done using **MCMC**
    - Model comparison via Bayes factors or evidence calculation using interface to **Cuba**
      (Cuba is a collection of integration methods, e.g., **VEGAS**)

USER DEFINED

- define model
- read data

COMMON METHODS

- normalize
- find mode / fit
- test the fit
- **marginalize wrt. several parameters**
- compare models
- provide nice output

Define MODEL

- define parameters $\vec{\lambda}$
- define likelihood $p(\vec{D}\,|\,\vec{\lambda})$
- define priors $p_0(\vec{\lambda})$

Read DATA

- from text file, ROOT tree, user-defined (anything)
- interface to user-defined software

$$p(\vec{\lambda}\,|\,\vec{D}) = \frac{p(\vec{D}\,|\,\vec{\lambda})\; p_0(\vec{\lambda})}{\int p(\vec{D}\,|\,\vec{\lambda})\; p_0(\vec{\lambda})\; d\vec{\lambda}}$$

Focus today:
Usage of MCMC in Bayesian inference

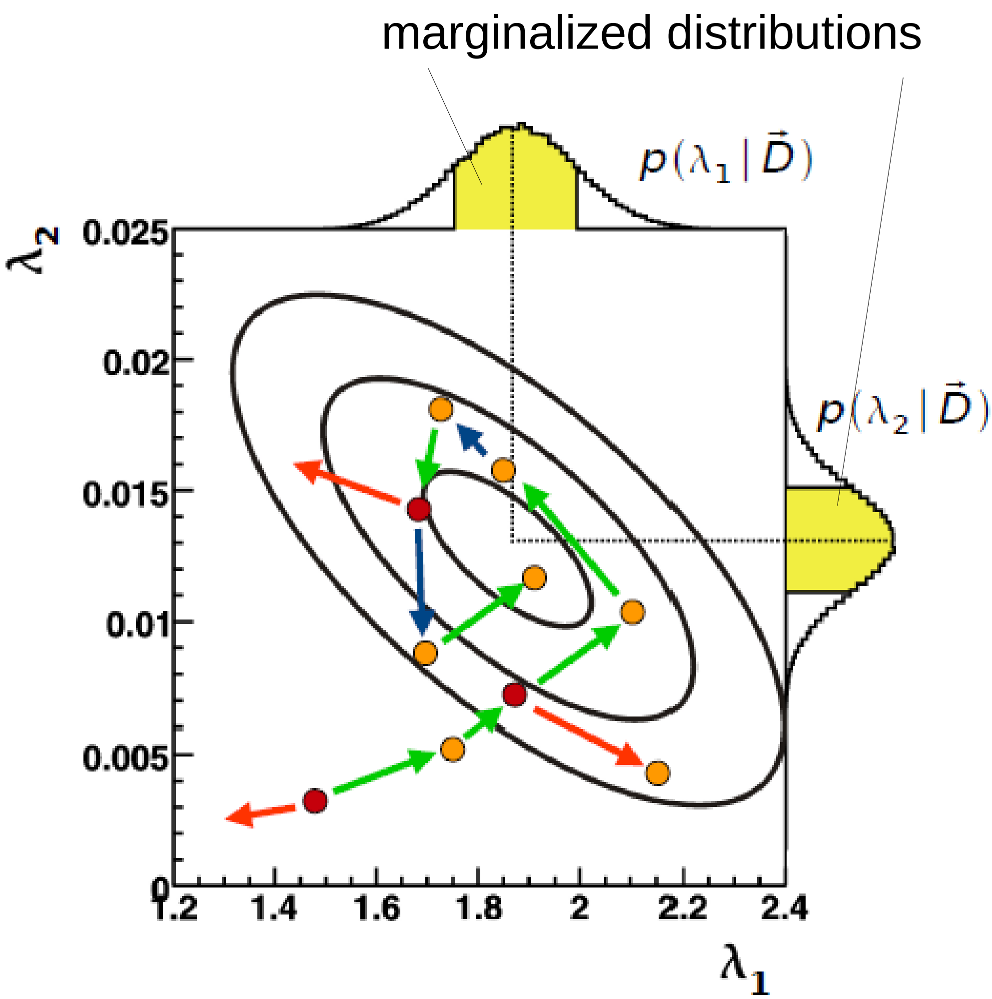
- **Usage of MCMC in Bayesian inference**
  - Use MCMC to **sample the posterior probability**, i.e.

    $$f(\vec{\lambda}) = p(\vec{D} \mid \vec{\lambda})\, p_0(\vec{\lambda})$$

  - Marginalization of posterior:

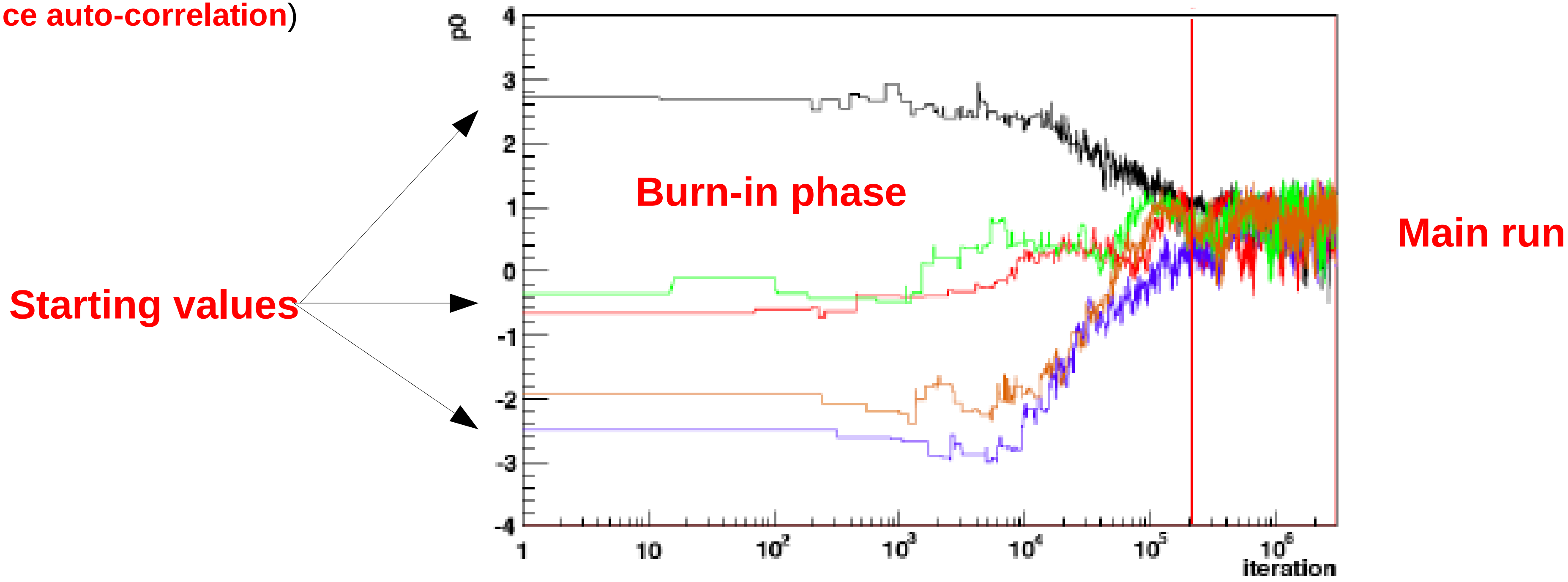
    $$p(\lambda_i \mid \vec{D}) = \int p(\vec{D} \mid \vec{\lambda})\, p_0(\vec{\lambda})\, d\vec{\lambda}_{j \neq i}$$

  - Fill a histogram with just one coordinate while sampling
  - Uncertainty propagation: calculate any function of the parameters while sampling
  - Point estimate: find mode while sampling

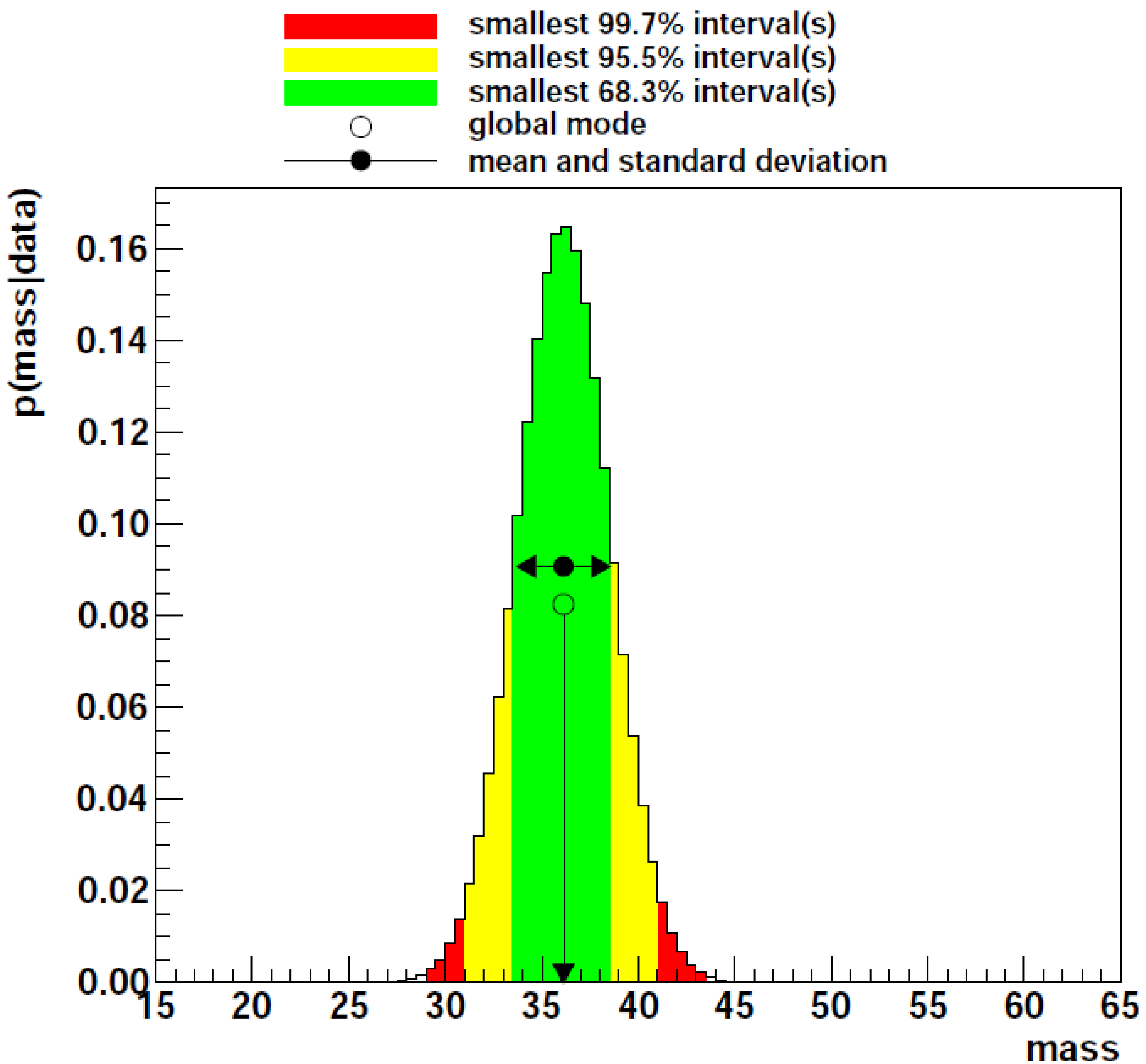


- **Step 1: Starting values**
  - Either **random** within parameter space (default)
  - or **center** of each dimension
  - or **user-defined**
- **Step 2: Burn-in phase**
  - Use multiple chains (default: 5)
  - Run until **convergence** is reached and chains are **efficient**
  - **Convergence** is reached if inter- and intra-chain variance are equal (Gelman and Rubin criterion)
    (Gelman & Rubin, StatSci 7, 1992)
  - Chains are **efficient** if the efficiency is between 15% and 50%
    - Run in sequences to adjust the width of the proposal functions:
    - If efficiency > 50%: increase the width
    - If efficiency < 15%: decrease the width

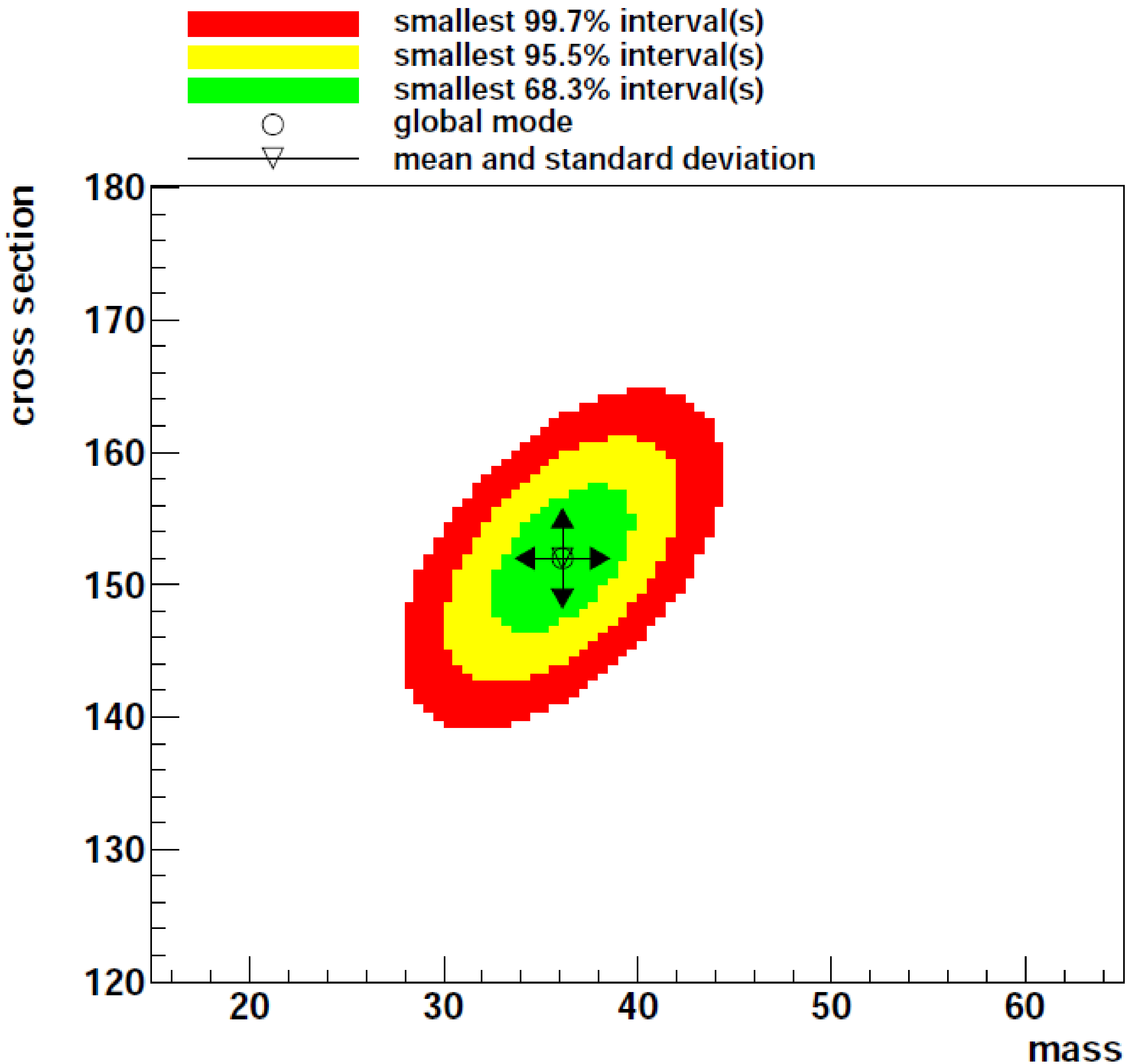
- **Step 3: Main run**
  - **Fix width of proposal function** to that obtained from efficiency optimization and convergence tests
    (always fixed during the main run)
  - Run for a specified number of iterations
  - Perform analysis-specific calculations
    (fill marginalized histograms, uncertainty propagation, fill ROOT tree, etc.)
  - Store information of every n^{th} iteration
    (**reduce auto-correlation**)



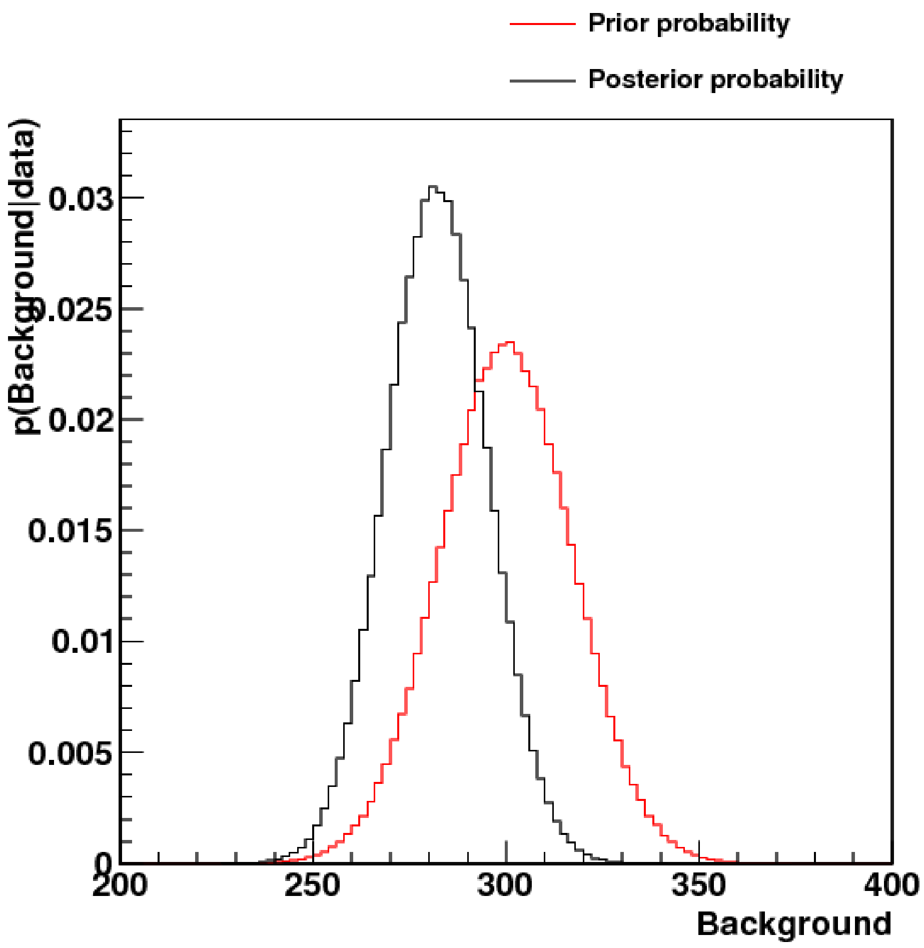
- **Output**
  - **Marginal distributions:** projection of posterior onto one or two parameter axes
  - Full (correlated) information in Markov Chain written as **ROOT tree**
  - Default **text output**:
    - Mean ± std. deviation
    - Median and central interval
    - Mode and smallest intervals(s)
    - Important quantiles
    - Global mode

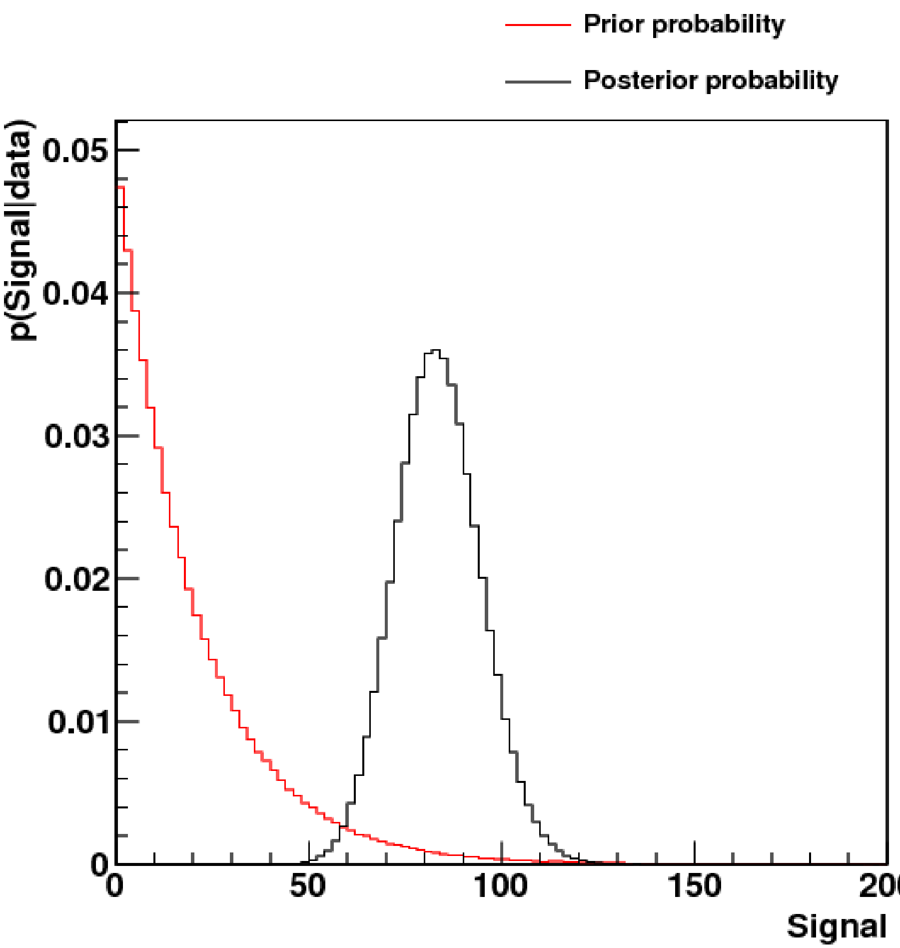


Output – marginalized 2D distributions

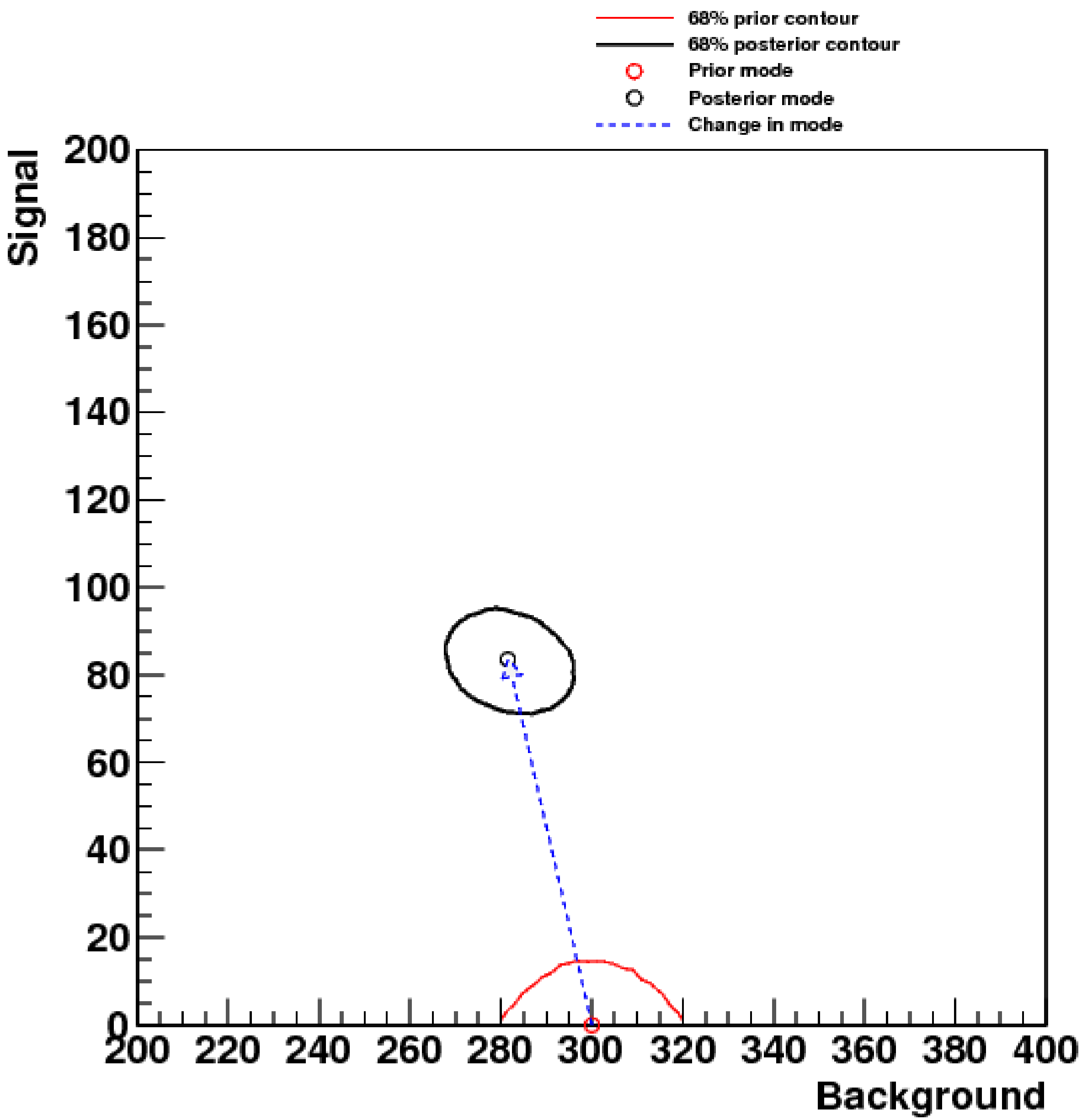


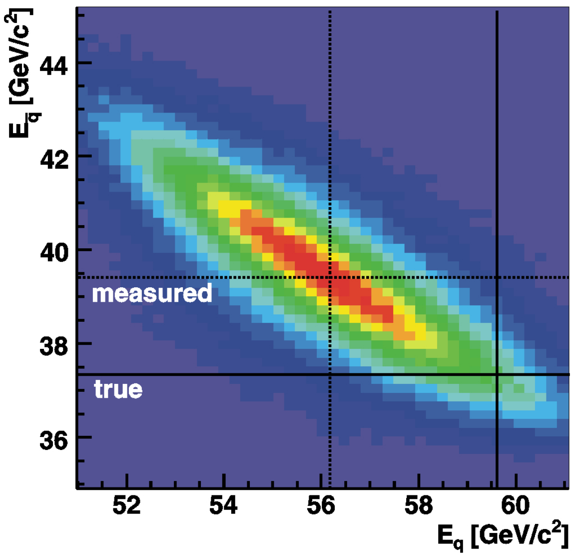
Output – update of knowledge

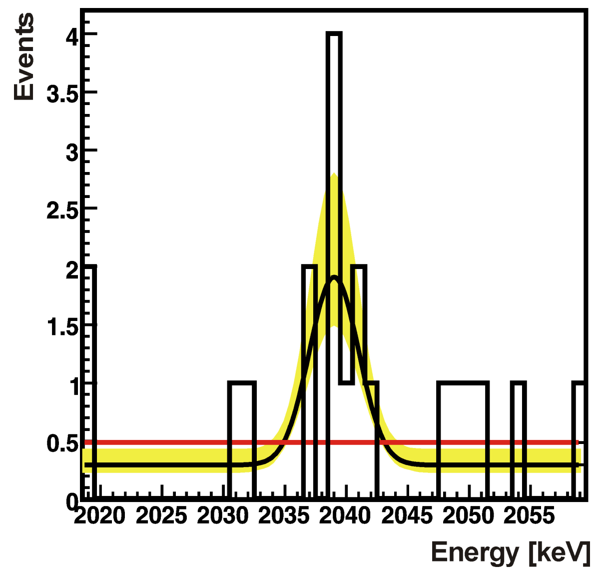


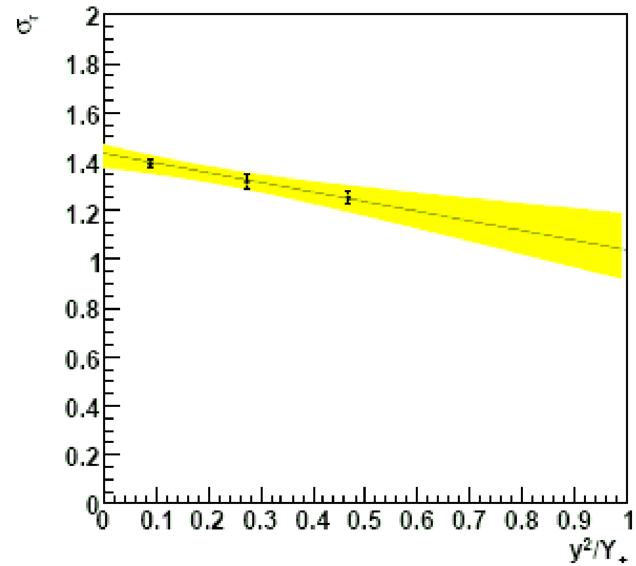


Output – update of knowledge

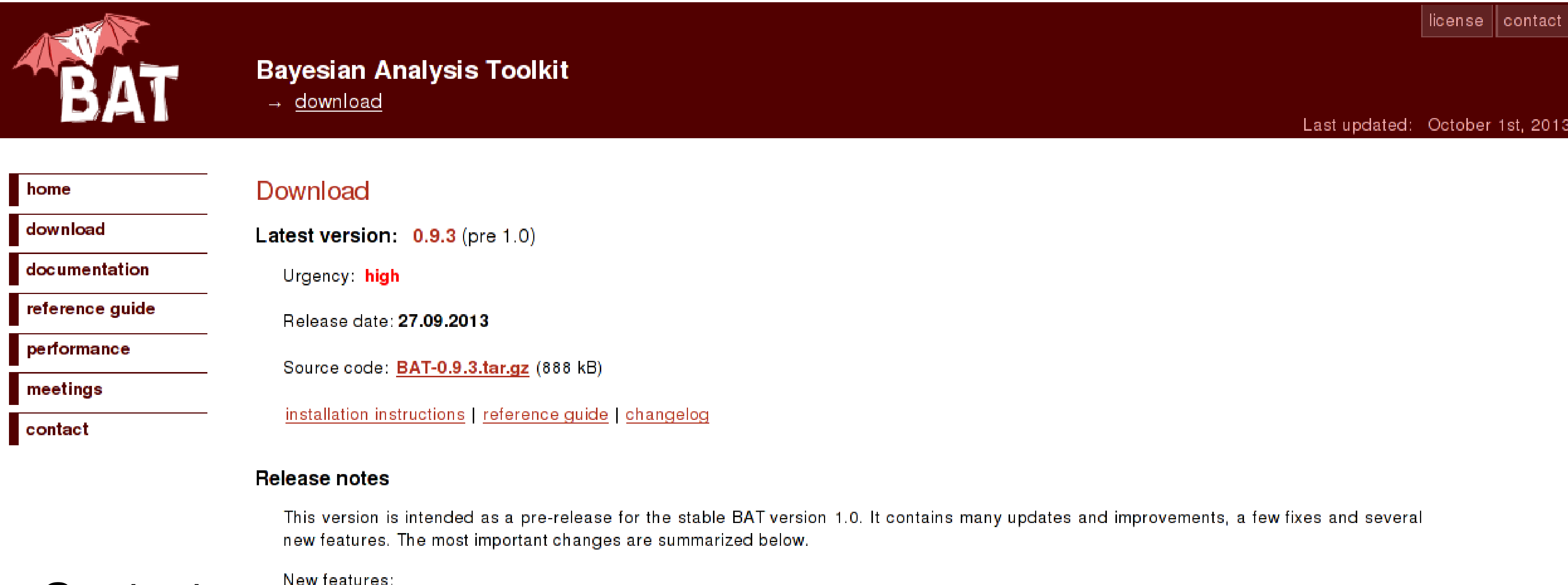








- Quentin Buat, *Search for extra dimensions in the diphoton final state with ATLAS* [arXiv:1201.4748]
- ATLAS collaboration, *Search for excited leptons in proton-proton collisions at sqrt(s) = 7 TeV with the ATLAS detector* [arXiv:1201.3293]
- I. Abt *et al.*, *Measurement of the temperature dependence of pulse lengths in an n-type germanium detector,* Eur. Phys. J. Appl. Phys.56:10104,2011 [arXiv:1112.5033]
- ATLAS collaboration, *Search for Extra Dimensions using diphoton events in 7 TeV proton-proton collisions with the ATLAS detector* [arXiv:1112.2194]
- ATLAS collaboration, *A measurement of the ratio of the W and Z cross sections with exactly one associated jet in pp collisions at sqrt(s) = 7 TeV with ATLAS*, Phys.Lett.B708:221-240,2012 [arXiv:1108.4908]
- ZEUS collaboration, *Search for single-top production in ep collisions at HERA*, Phys.Lett.B708:27-36,2012 [arXiv:1111.3901]
- CMS collaboration, *Search for a W' boson decaying to a muon and a neutrino in pp collisions at sqrt(s) = 7 TeV*, Phys.Lett.B701:160-179,2011 [arXiv:1103.0030]
- ZEUS collaboration, *Measurement of the Longitudinal Proton Structure Function at HERA,* Phys.Lett.B682:8-22,2009 [arXiv:0904.1092]



- **Contact**
  - Web page: **http://www.mppmu.mpg.de/bat/**
  - Contact: **bat@mppmu.mpg.de**
  - Paper on BAT:
    - A. Caldwell, D. Kollar, K. Kröninger, *BAT - The Bayesian Analysis Toolkit* Comp. Phys. Comm. 180 (2009) 2197-2209 [arXiv:0808.2552]

The tutorial

- **Setup**
  - BAT is installed on the NAF, no need to install it locally
  - Use your account to ssh into the NAF

- **Help**
  - Ask us directly (Dan, Fred, Kevin)
  - Check the reference guide on the web page: https://www.mppmu.mpg.de/bat/docs/refman/html-0.9.3/
  - Check the examples in the BAT release

- **Setting up BAT**
  - ssh schoolNN@naf-school01.desy.de
  - cd /afs/desy.de/group/school/mc-school/bat/tutorial
  - source setup_bat.sh

- **Getting started with BAT**
  - Create your own working directory and cd into it
  - cp /afs/desy.de/group/school/mc-school/bat/BAT-0.9.3/tools/CreateProject.sh .
  - ./CreateProject.sh <project> [<model>]

- **BAT examples**
  - Examples can be found in the directory /afs/desy.de/group/school/mc-school/bat/BAT-0.9.3/examples

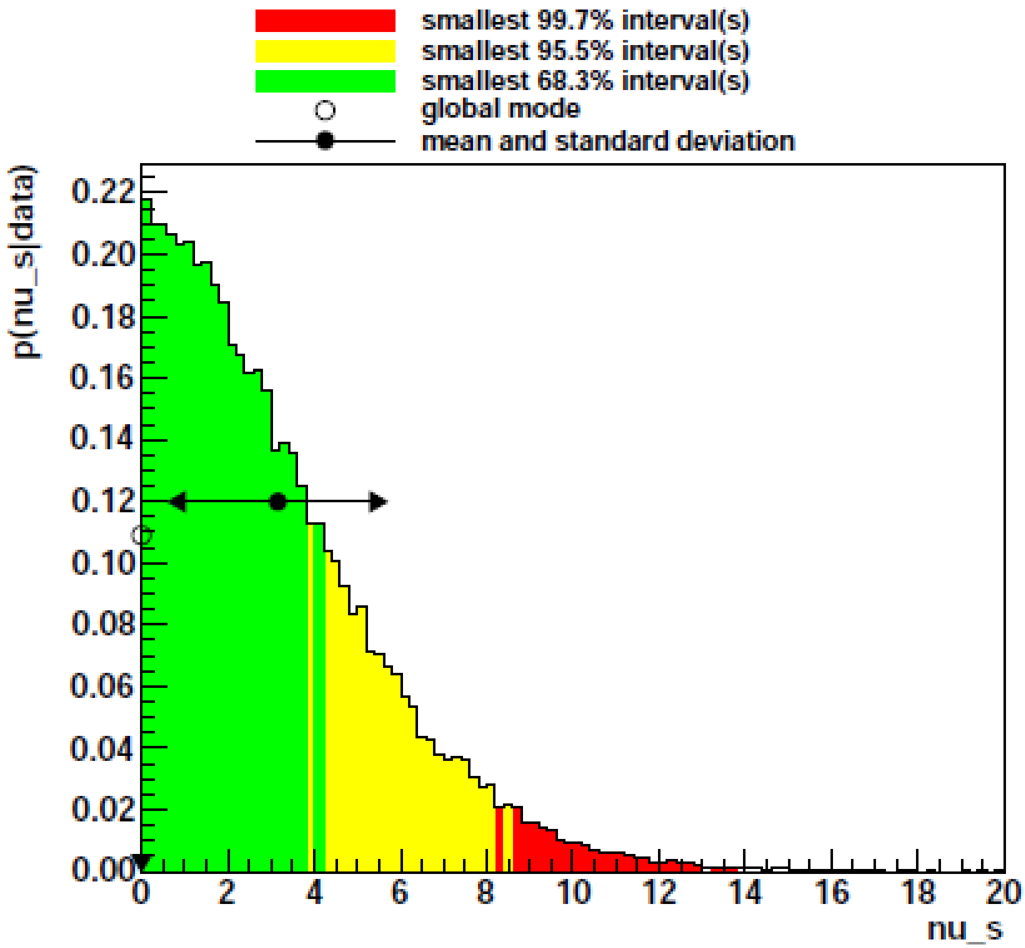
- **Physics case**
  - Counting experiment: searching for signal ν_s in presence of background
  - Expect ν_b = 10 ± 3 background events, observe n = 10 events
  - Later: limit on cross-section σ with efficiency of ε=0.1 ± 0.02 (assume luminosity to be L=1):

$$\sigma = \frac{\nu_s}{\epsilon \cdot L}$$

- **The tutorial**
  - Exercise 1: getting started; fix background to expected value
  - Exercise 2: assume not-so-well-known background
  - Exercise 3: update of knowledge
  - Exercise 4: propagation of uncertainty
  - Exercise 5: choice of priors (optional)
  - Exercise 6: evidence calculation and model comparison (optional)

- **Implementing a first model**
  - Run the `CreateProject.sh` script
    Takes name of project and name of model as arguments. The script generates the BAT model files (XXX.cxx and XXX.h), a run file (runXXX.cxx) and a Makefile
  - Modifications to your model file:
    - Add a signal parameter to the model
      Use `BCModel::AddParameter(...)`, in XXX::DefineParameters(). Consider an appropriate range.
    - Define the likelihood to be a Poisson, assume the number of background events to be fixed to 10, so $\nu_{exp} = \nu_s + 10$
      Use `BCMath::LogPoisson(double observed, double expected)`
    - Define a uniform prior for the signal parameter
  - Modifications to your run file:
    - Choose the Metropolis algorithm and marginalize:
      Use `BCModel::SetMarginalizationMethod(BCIntegrate::kMargMetropolis)`
      and `BCModel::MarginalizeAll()`
      Print using `BCModel::PrintAllMarginalized(...)` and `BCModel::PrintResults(...)`
  - Make and run the program. Investigate the plots and numbers.

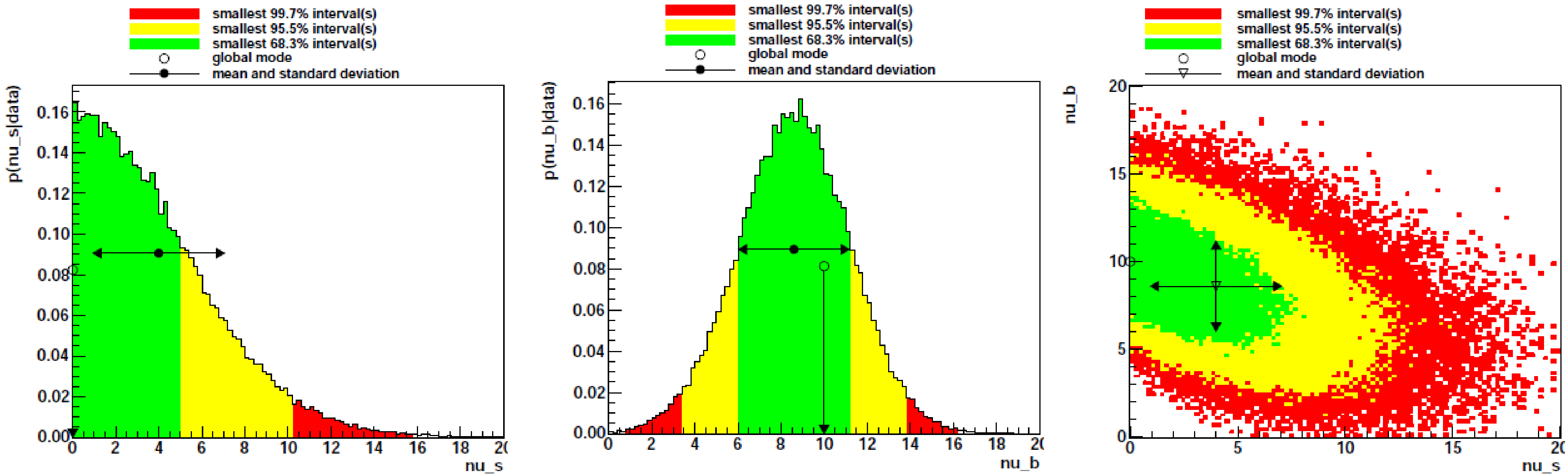
- **Plot:**



- **Numbers:**
  - 90% upper limit on signal: 6.59
  - 95% upper limit on signal: 8.02

- **Not-so-well-known background**
  - Modifications to your model file:
    - Add a background parameter to the model
      Use `BCModel::AddParameter(...)`, consider an appropriate choice of the range
    - Define the likelihood to be a Poisson, the number of expected events is now a function of the two parameters
    - Define a Gaussian prior for the background parameter with mean 10 and standard deviation 3.
  - Re-run the program and investigate the changes

- **Plots:**

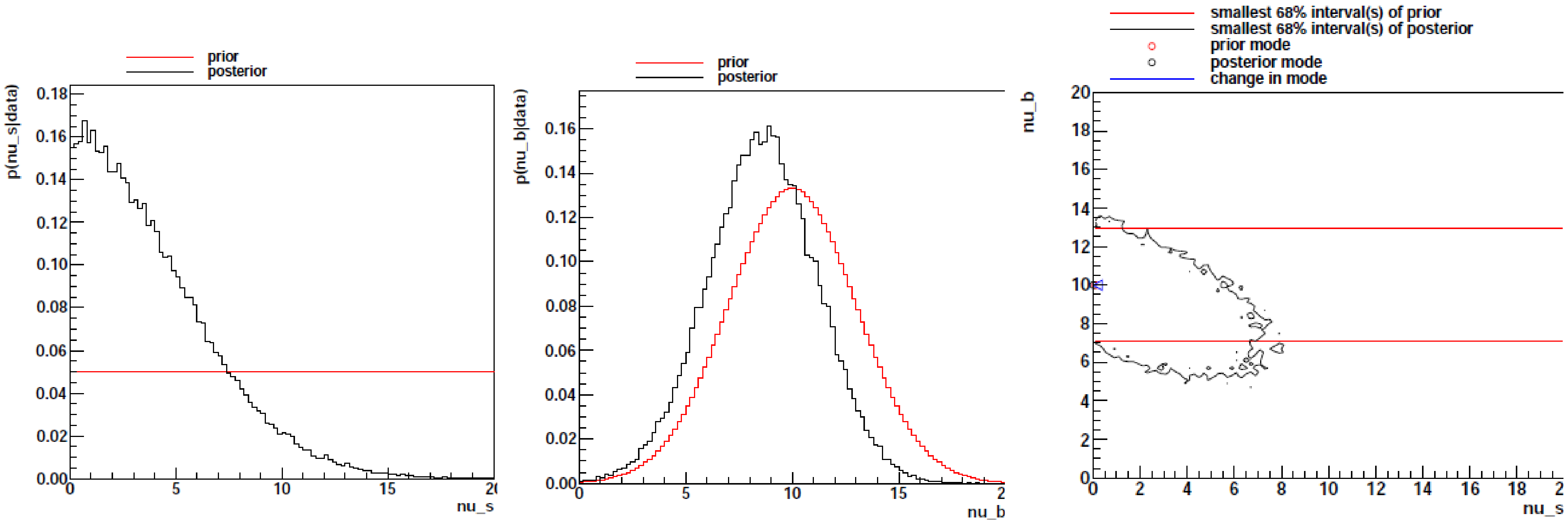


- **Numbers:**
  - 90% upper limit on signal: 8.26
  - 95% upper limit on signal: 9.90

Exercise 3

- **Update-of-knowledge**
  - Modifications to your run file:
    - Include an instance of the `BCSummaryTool`
      Check the reference guide for how to use the tool:
      https://www.mppmu.mpg.de/bat/docs/refman/html-0.9.3/
  - Print and study the knowledge update plot. How did your knowledge increase?

- **Plots:**


prior
posterior
p(nu_s|data)
nu_s
prior
posterior
p(nu_b|data)
nu_b
smallest 68% interval(s) of prior
smallest 68% interval(s) of posterior
prior mode
posterior mode
change in mode
nu_b
nu_s

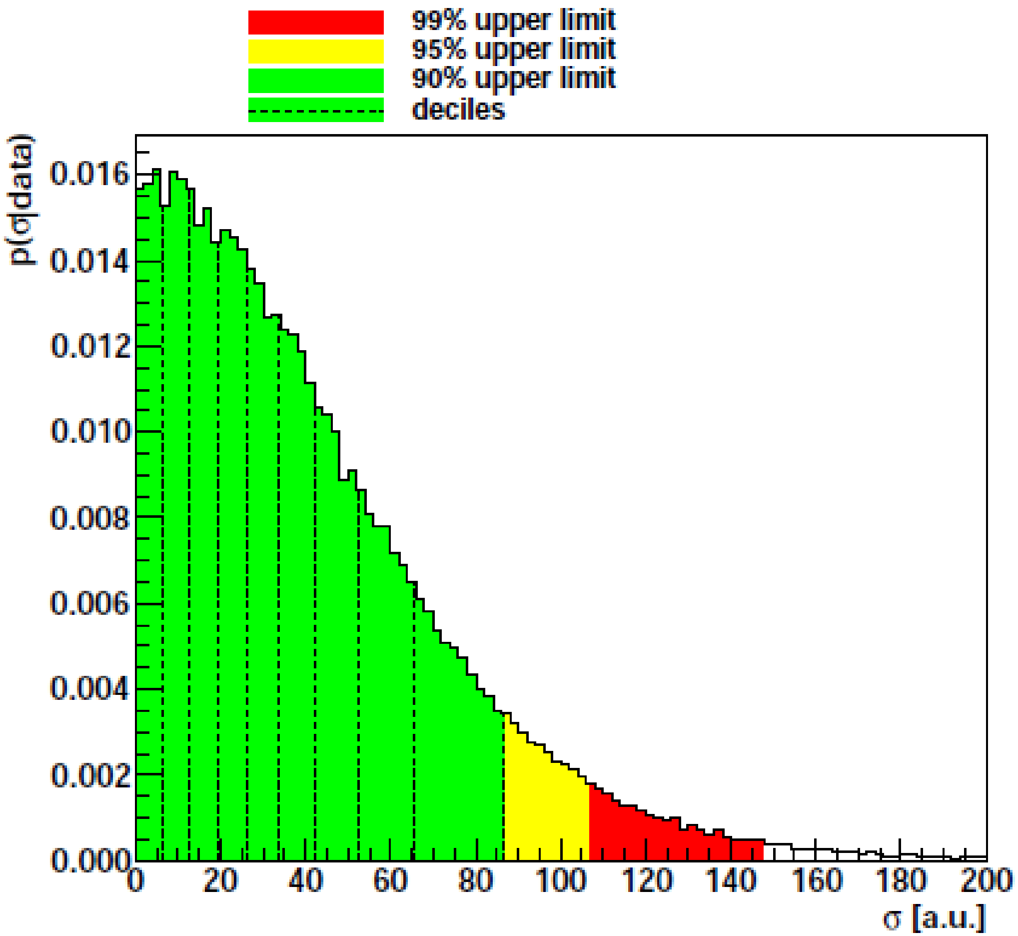
- **Propagation of uncertainty**
  - Modifications to your model file:
    - Add a method that is called for each sample:

```
void MyModel::MCMCUserIterationInterface() {
        int nchains = MCMCGetNChains();
        int npar = GetNParameters();
        for (int i = 0; i < nchains; ++i) {
                double x = fMCMCx.at(i * npar + 0);
                double y = fMCMCx.at(i * npar + 1);
                double z = fMCMCx.at(i * npar + 2);
                MyHistogram->Fill(x/z);
        }

}
```

    - Add a BCH1D Histogram to the .h file and fill it for each sample
    - Add a parameter for the efficiency with a Gaussian prior with mean 0.1 and standard deviation 0.02
  - Modifications to your run file
    - Get the histogram from the model and print the histograms
  - What is the 95% limit you can set on the cross-section?

- **Plots:**



- **Numbers:**
  - 90% upper limit on cross-section: 86.76
  - 95% upper limit on cross-section: 106.16

- **Priors, priors, priors**
  - Repeat your analysis with different priors, e.g. and expontential one, a Gaussian one or a Jeffreys prior
  - How does the limit on the signal and the cross-section change?

- **Model comparison and evidence calculation**
  - Modifications to your run file:
    - Choose an integration method
      `Use BCModel::SetIntegrationMethod(...)`
    - Run the integration
      `Use BCModel::Normalize()`
  - Repeat your studies for the signal fixed to 0 and compare the two evidences. Which model is more likely?

Backup material