BAT

A Bayesian Analysis Toolkit

A. Caldwell[1], D. Kollár[2], **K. Kröninger[3]**

[1]*Max-Planck Institut für Physik, München*

[2]*CERN*

[3]*II. Physikalisches Institut, Universität Göttingen*

Outline:
- Motivation
- Requirements/Specifications/Implementation
- Markov Chain Monte Carlo
- A simple example
- Applications and status

Motivation

Experimental data ↔ Data analysis ↔ Models and parameters

What does the data tell us about our model? **Parameter estimation**

Which model is favored by the data? **Model comparison**

Is the model in agreement with the data? **Goodness-of-fit test**

... from simple line fitting to cosmological model testing!

Need tools to extract information!

BAT
BAYESIAN ANALYSIS TOOLKIT

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

Requirements / Specifications

Should be able to ...

(1) phrase arbitrary problems,

(2) interface to HEP-software,

(3) estimate parameters,

(4) extract pdfs for single parameters and correlations; evaluate limits,

(5) propagate uncertainties,

(6) compare models,

(7) check validity of a models.

Implementation:

- **C++** framework based on ROOT core functionality. BAT comes as library.
- Define base class for models which then need to be specified by the user, or
- run ROOT interactively using pre-defined fitter methods.
- Use modules for numerical tasks (optimization, integration, etc.). Chose from a set of algorithms.

GEORG-AUGUST-UNIVERSITÄT GÖTTINGEN

Requirements / Specifications

Should be able to ...

(1) phrase arbitrary problems,

(2) interface to HEP-software,

(3) estimate parameters,

(4) extract pdfs for single parameters and correlations; evaluate limits,

(5) propagate uncertainties,

(6) compare models,

(7) check validity of a models.

Implementation:

- Use minimization methods, e.g. interface to TMinuit.
  Working on simulated annealing.
- Use **Markov Chain Monte Carlo (MCMC)** for marginalization and error estimation.
- Use MCMC for uncertainty propagation.
  (no Gaussian assumptions here)

Key element: MCMC

Requirements / Specifications

Should be able to ...

(1) phrase arbitrary problems,

(2) interface to HEP-software,

(3) estimate parameters,

(4) extract pdfs for single parameters and correlations; evaluate limits,

(5) propagate uncertainties,

(6) compare models,

(7) check validity of a models.

Implementation:

- Model comparison using common definitions (e.g. direct comparison of probabilities)
- Model validity is a difficult topic. Use **p-value** to judge. (generalization of χ^2-probability).

Implementation

User defined (Pseudo code)

```
class myModel:
    • Parameters
    • Conditional probability
    • Prior probabilities
```

```
int main(){
MyModel * mm;
mm → “read in data“();
mm → “perform analysis“(); }
```

Common tasks

Analysis

- Normalize (Integration)
- Maximize (Minimization)
- Marginalize/Sampling
- Goodness-of-fit
- Model comparison
- Error propagation
- Write output file
- Graphical output

Tools and algorithms

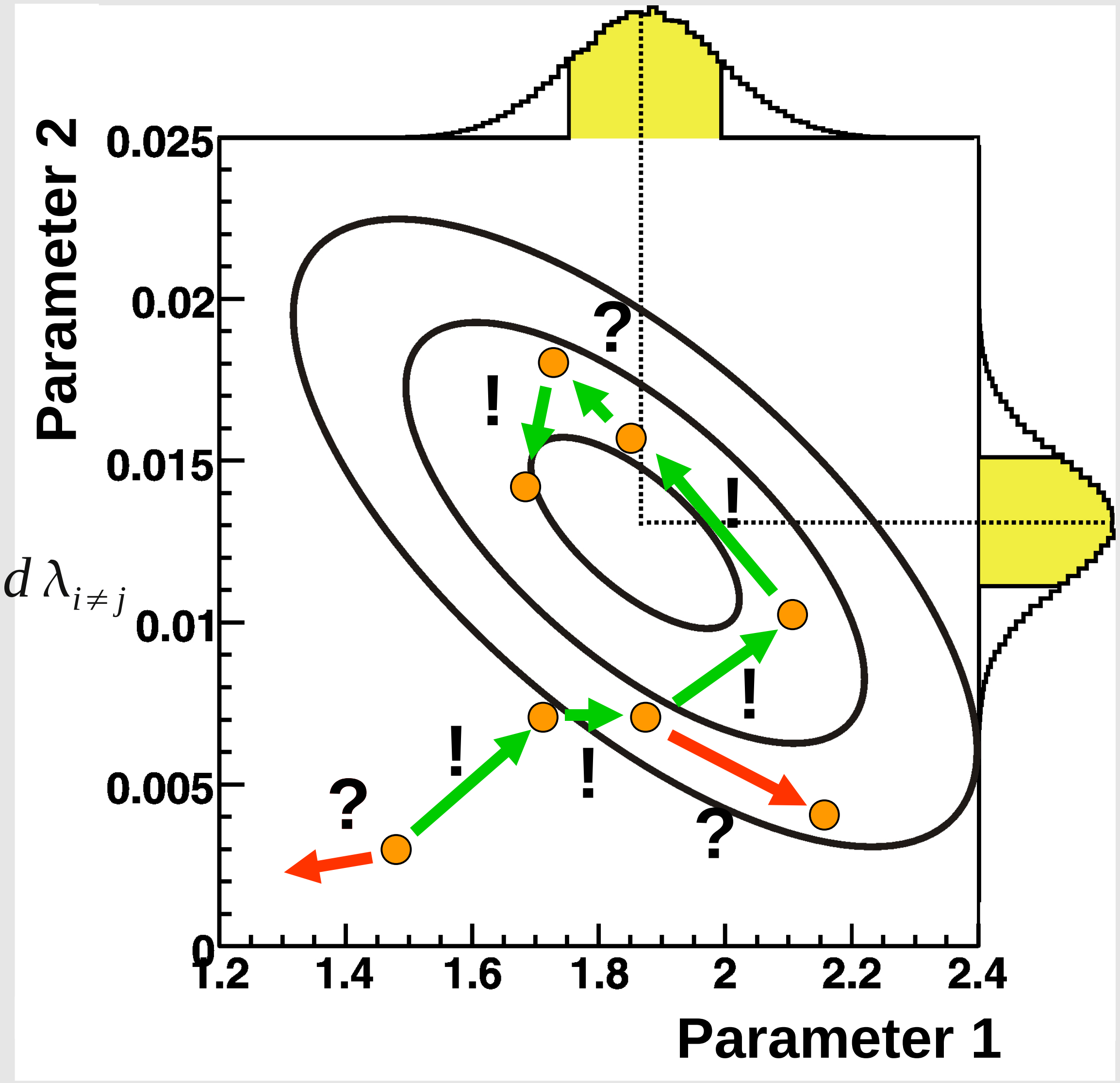
- **Integration:**
  - Simple Monte Carlo algorithms (sampled mean, importance sampling)
  - Interface to CUBA (VEGAS, ...)
- **Marginalization:**
  - **MCMC (Metropolis)**
- **Sampling:**
  - Monte Carlo sampling
  - **MCMC (Metropolis)**
- **Propagation of uncertainies:**
  - Calculate any function of the parameters during **MCMC**.
- **Minimization:**
  - Monte Carlo (hit & miss)
  - **MCMC** (during random walk)
  - Interface to Minuit
  - Simulated Annealing planned
- **Goodness-of-fit:**
  - Ensemble tests and p-value

Key: Markov Chain Monte Carlo (MCMC)

MCMC and Bayesian Inference

MCMC in BAT:

- Sample parameter space using MCMC
- Function:
  $$f(\vec{\lambda}) = p(D|\vec{\lambda}) \cdot p_0(\vec{\lambda})$$
- Marginalize pdf while walking:
  $$p(D|\vec{\lambda}) = \int p(D|\vec{\lambda}) \cdot p_0(\vec{\lambda})\, d\lambda_{i \neq j}$$
- Calculate any function of parameters while walking (uncertainty propagation)
- Find global maximum
- **Step size?**



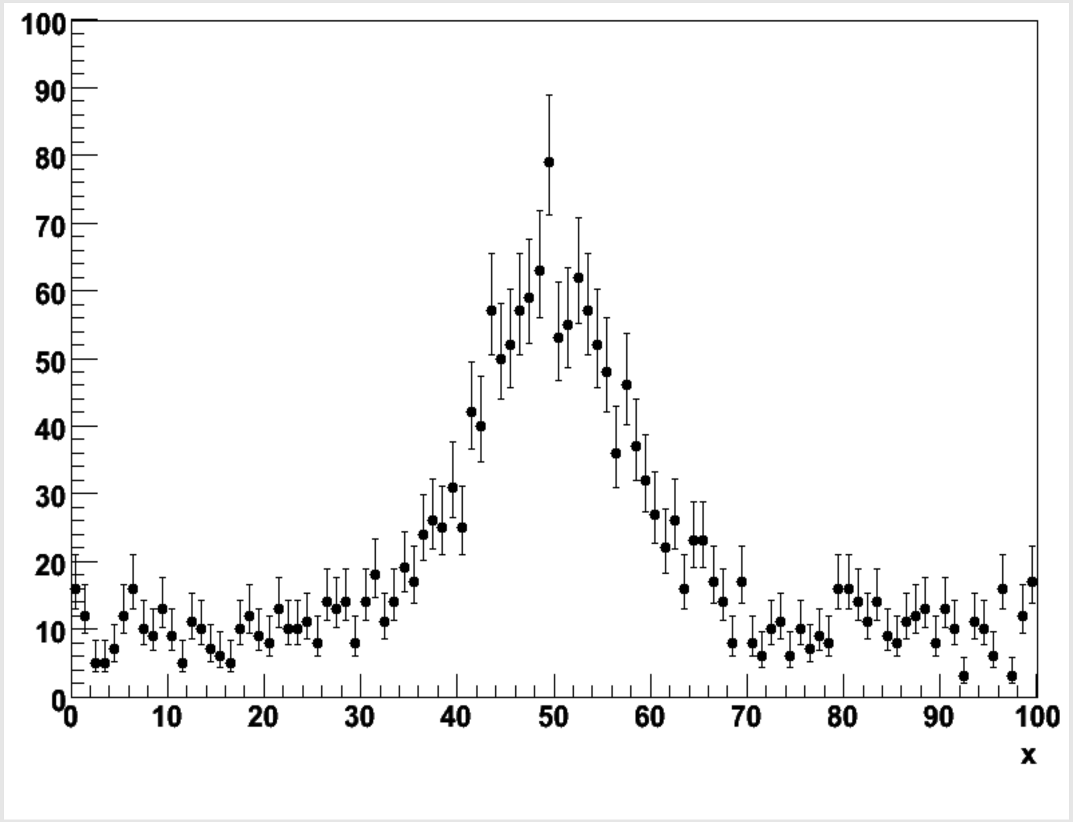
Example: Fit to spectrum

Simple spectrum:

- Gaussian signal:
  position μ = 50
  width σ = 7.5
  strength <S> = 1,000
- Flat background:
  strength <B> = 10/bin
- Number of events per bin fluctuate with Poisson distribution

$$p(D|S,\mu,\sigma,B)=\prod_{i=1}^{N_{bins}}\frac{\lambda_i^{n_i}}{n_i!}e^{-\lambda_i}$$

$$\lambda_i=S\int_{\Delta_i}\frac{1}{\sqrt{2\pi\sigma}}e^{\frac{-(x-\mu)^2}{2\sigma^2}}dx+\frac{B}{\Delta_i}$$



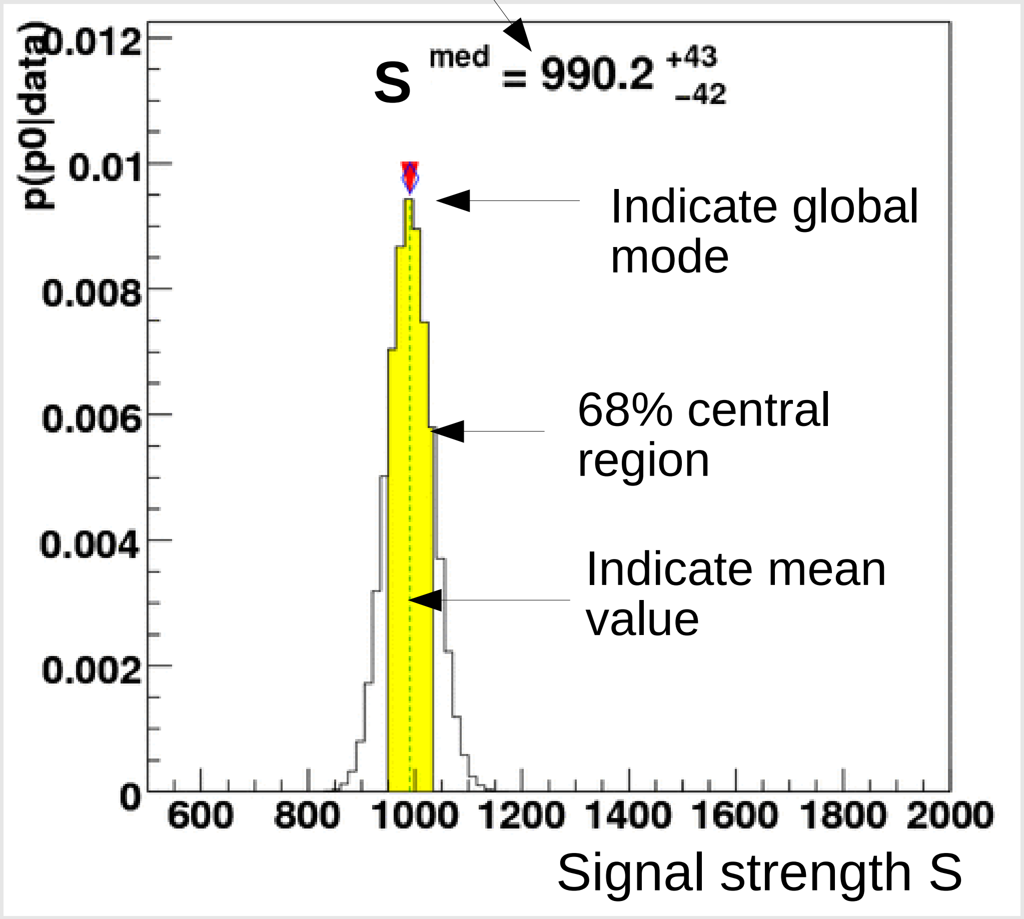
GEORG-AUGUST-UNIVERSITÄT GÖTTINGEN

Marginalized Distributions 1-D

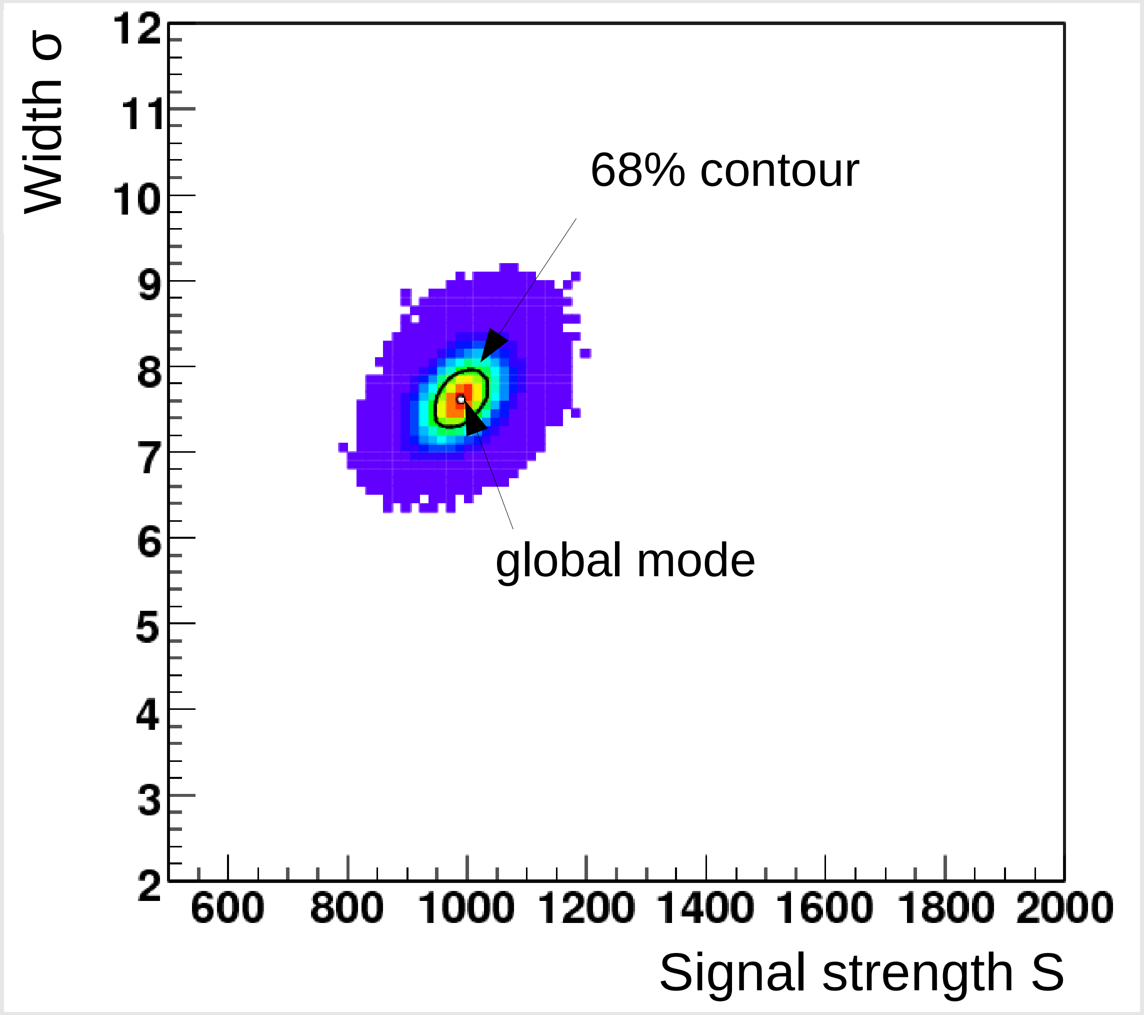
Marginalized distributions:

- Project sampled distribution onto one (or two) parameters. Integrate over all others.
- Mode of marginalized distribution is in general not equal global mode. User's decision.
- **Full information in Markov chain.**
- Default output:
    - Global mode
    - Mean value
    - Central and smallest 68% interval

Quote median value and (68% central interval) uncertainty

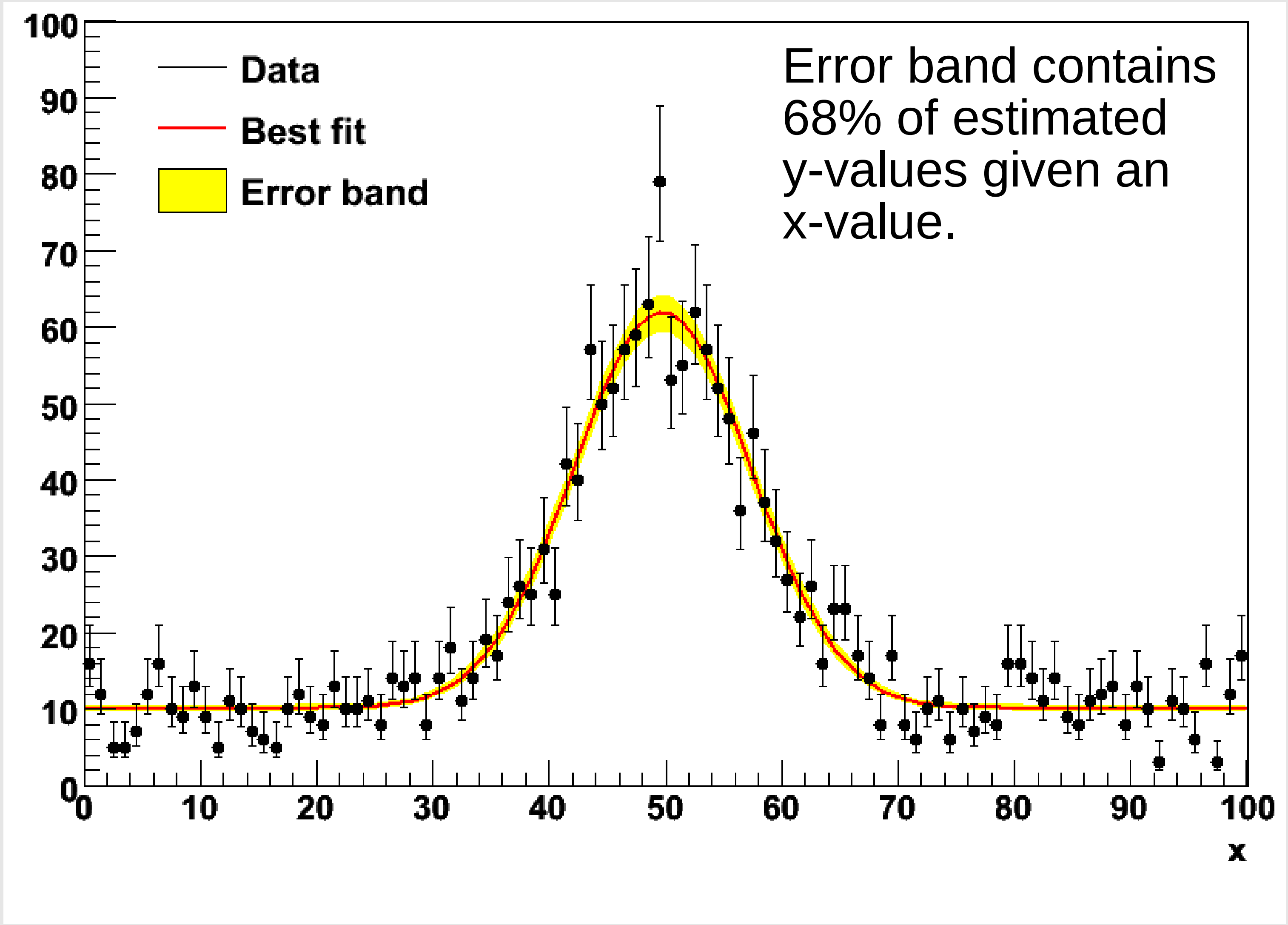


Marginalized Distributions 2-D



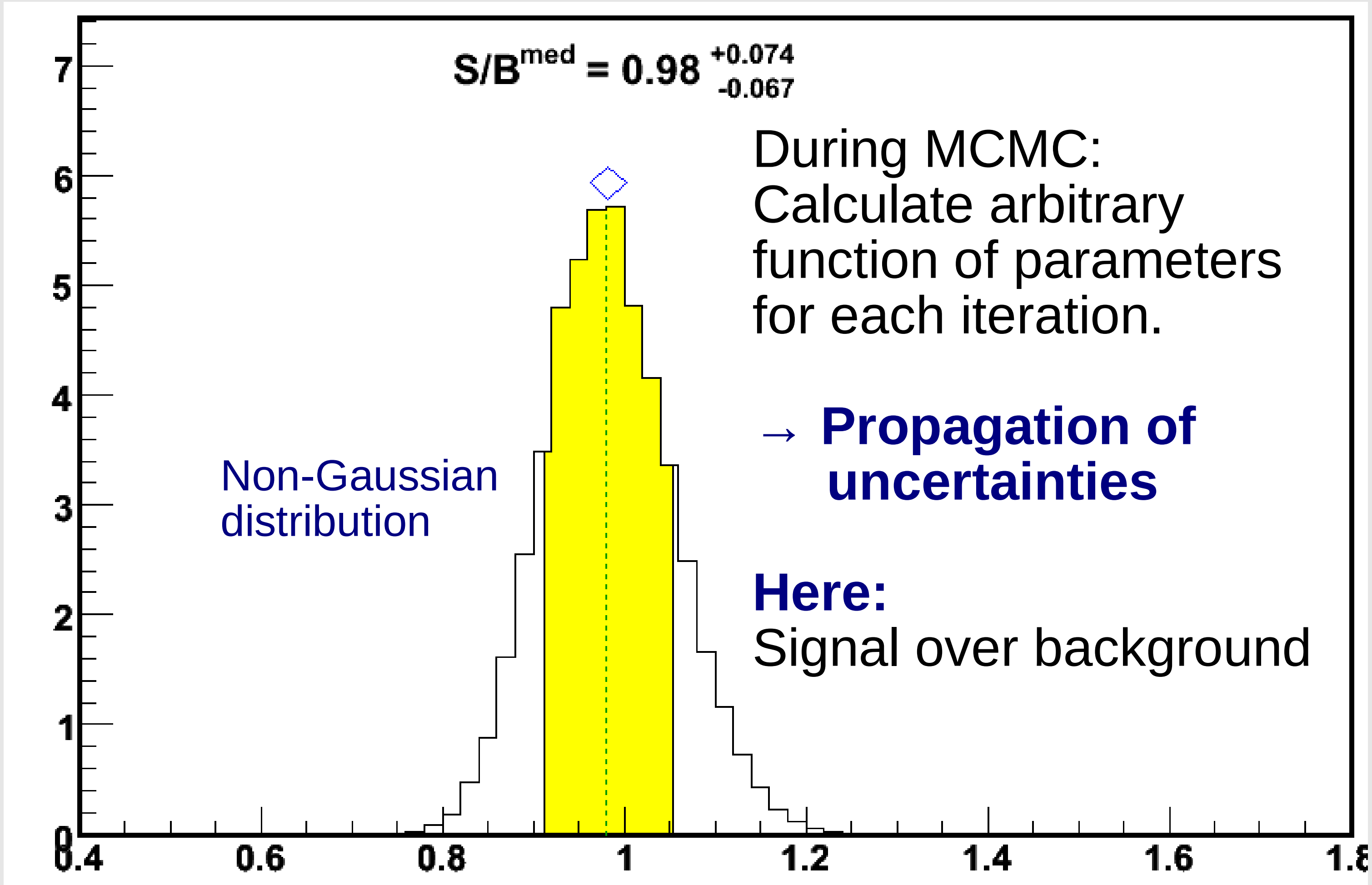
All distributions (1-d and 2-d) are stored during single run

Best Fit and Error Band



Error band contains 68% of estimated y-values given an x-value.

Propagation of Uncertainties



During MCMC:
Calculate arbitrary function of parameters for each iteration.

→ **Propagation of uncertainties**

Here:
Signal over background

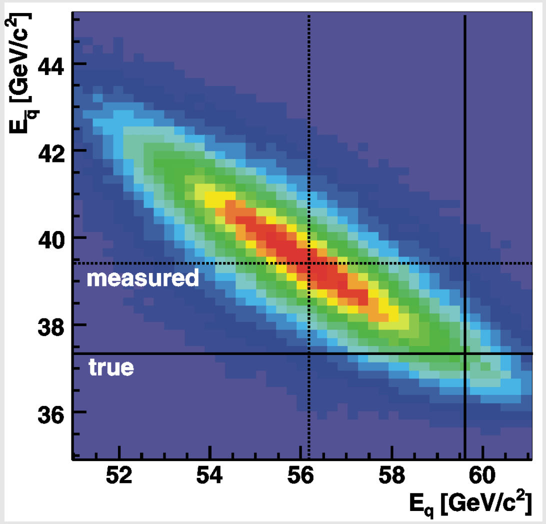
Generated S~1000 and B~10/bin in 100 bins, thus expect S/B ~ 1

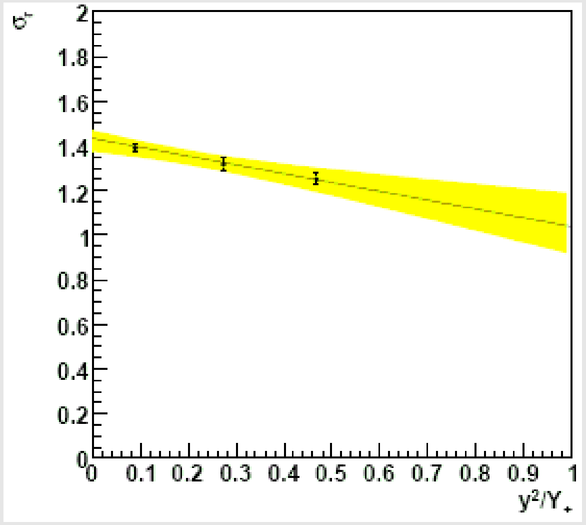
Applications

ATLAS

Kinematic fitting in top events

→ O. Nackenhorst (T 34.4)
→ A. Knue (T 36.2)



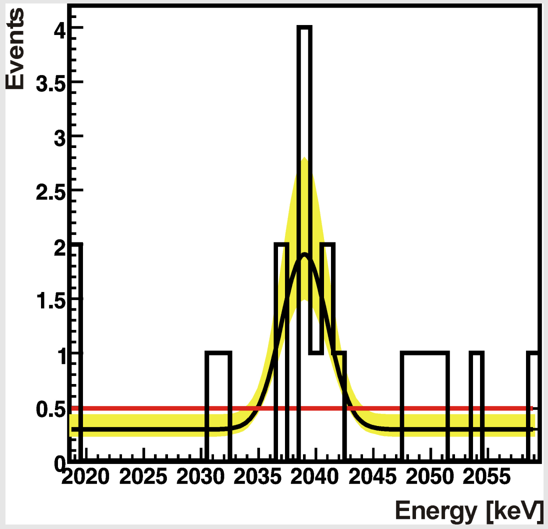


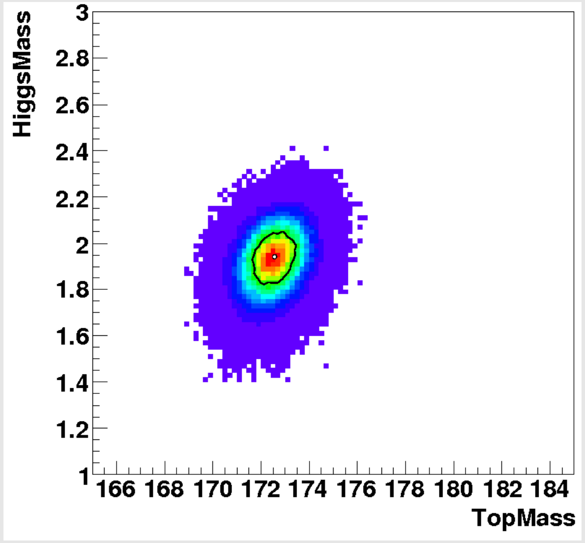
HERA

Structure function parameters

GERDA

Small statistics fitting





„Theory“

SM fitter

Status / Outlook

- Current software release available on web-site: www.mppmu.mpg.de/bat
- Paper on arxiv: http://arxiv.org/abs/0808.2552 (submitted to *Computer Physics Communications*)
- Examples and use-cases available
- Interface to RooStat (part of next BAT release)
- Implementation (of parts) into ROOT planned (with ROOT-team)
- More algorithms are currently being implemented
- **Developers and beta-testers welcome!**

Backup

MCMC: Metropolis Algorithm

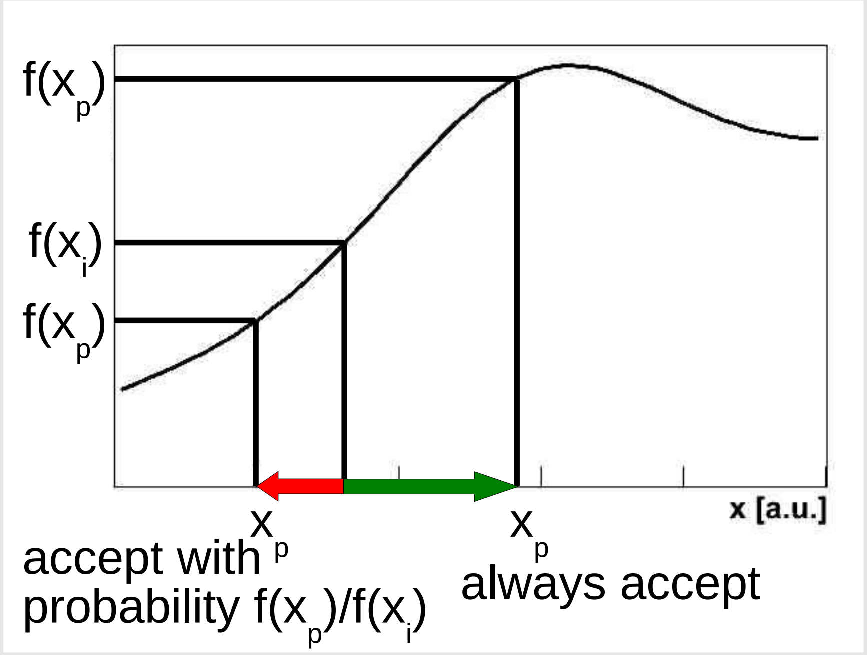
Sampling parameter space can efficiently be done using MCMC

Idea: random walk heading towards regions of larger probability

Metropolis algorithm:

1. Start at random point x_i.
2. Generate proposal point x_p vicinity of x_i.
3. If $f(x_p) > f(x_i)$ chose $x_{i+1} = x_p$, Else, accept proposal only with probability $p=f(x_p)/f(x_i)$.
4. Goto 2.

Causes migration towards region of large function values (probability)



Simple MCMC Examples

Test function:

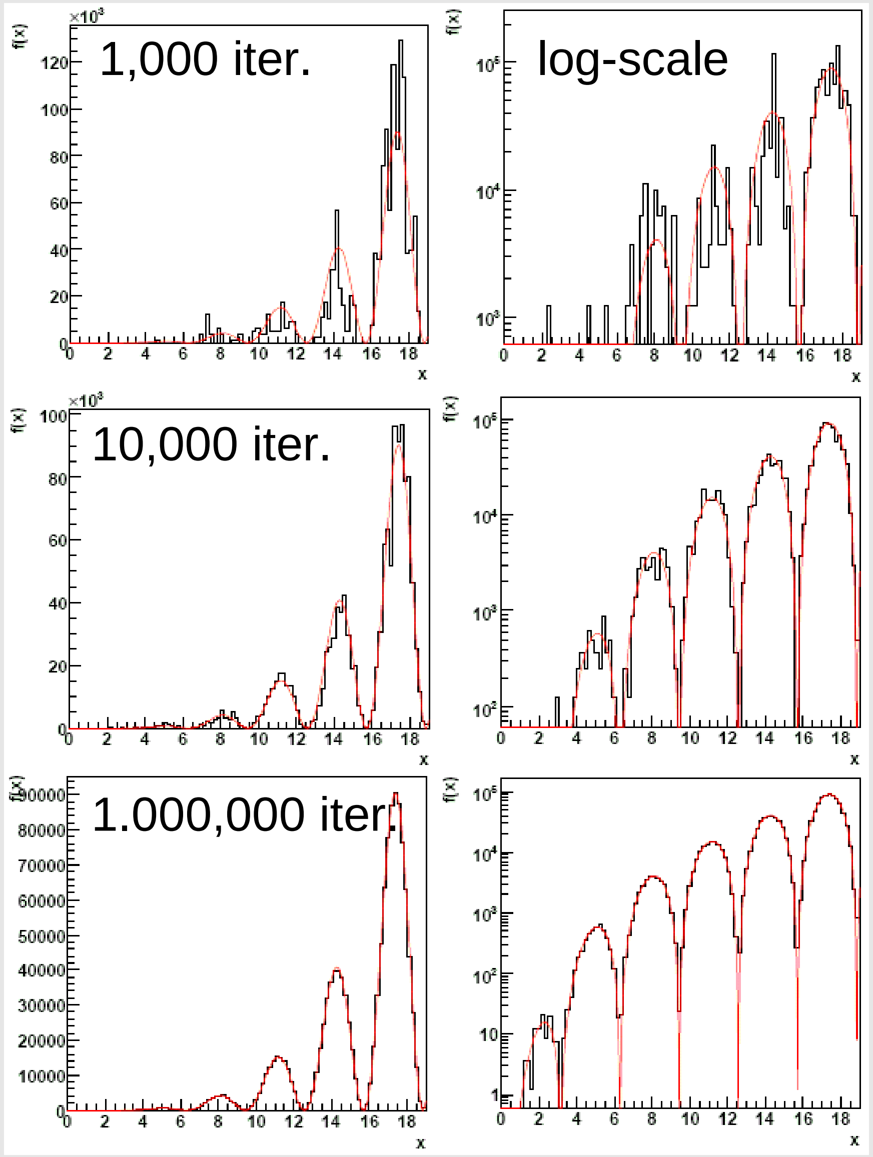
$$f(x) = x^4 \cdot \sin^2 x$$

Here:

- chains converge quickly towards underlying function.
- Several maxima/minima are no problem. Can sample from complicated functions.

But:
when is convergence reached?

→ **Convergence criteria** have to be defined.



MCMC in BAT

Flow of algorithm for MCMC:

- Run several chains in parallel (default: 5). ★
- Start a **pre-run**:
  - forget about initial position
  - sample through parameter space:
    - adjust proposal function
    - monitor convergence criterion
  - stop, if **(adjusted and converged)** or **(max. number of iterations)**
  - summarize MCMC-parameters
- Start **main run**:
  - sample through parameter space:
    - **update marginalized distributions** (1-D and 2-D)
    - **update global mode** (if necessary)
    - calculate any function of parameters for **uncertainty propagation**
  - stop, if maximum number of iterations reached (default: 100,000)

GEORG-AUGUST-UNIVERSITÄT GÖTTINGEN

MCMC in BAT: Proposal function

Proposal function:

- Proposal function is flat in a box around current point
- Start with width = interval of parameter
- Adjust width during MCMC-prerun.
- **Efficiency ε** is defined as the fraction of accepted proposals:

$\varepsilon \rightarrow 1$:	random sampling (no migration towards maxima)
$\varepsilon \rightarrow 0$:	no sampling (no walk at all)
$\varepsilon \sim 0.25$:	(empirical) optimal value (trade-off)

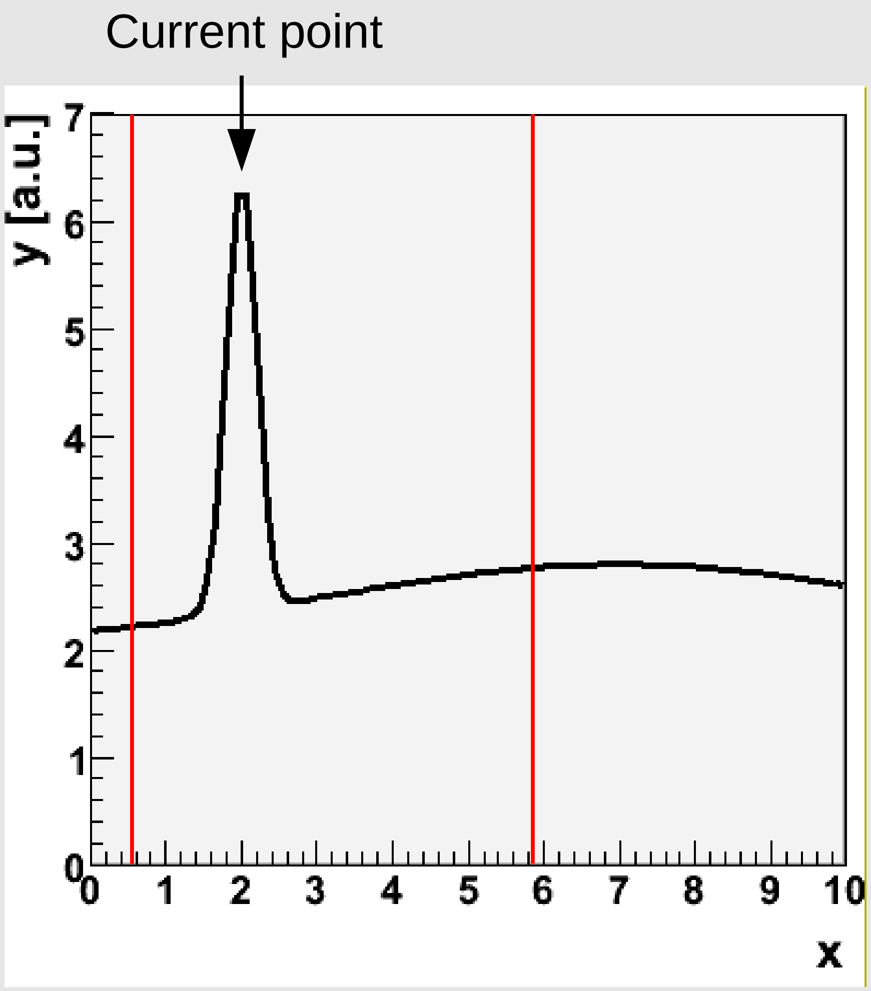
- Calculate efficiency for a consecutive sub-chains.

If $\varepsilon < 0.1$:	reduce width by factor ½ („curve“ becomes flatter)
if $\varepsilon > 0.5$:	increase width by factor 2 („jump around“ more)

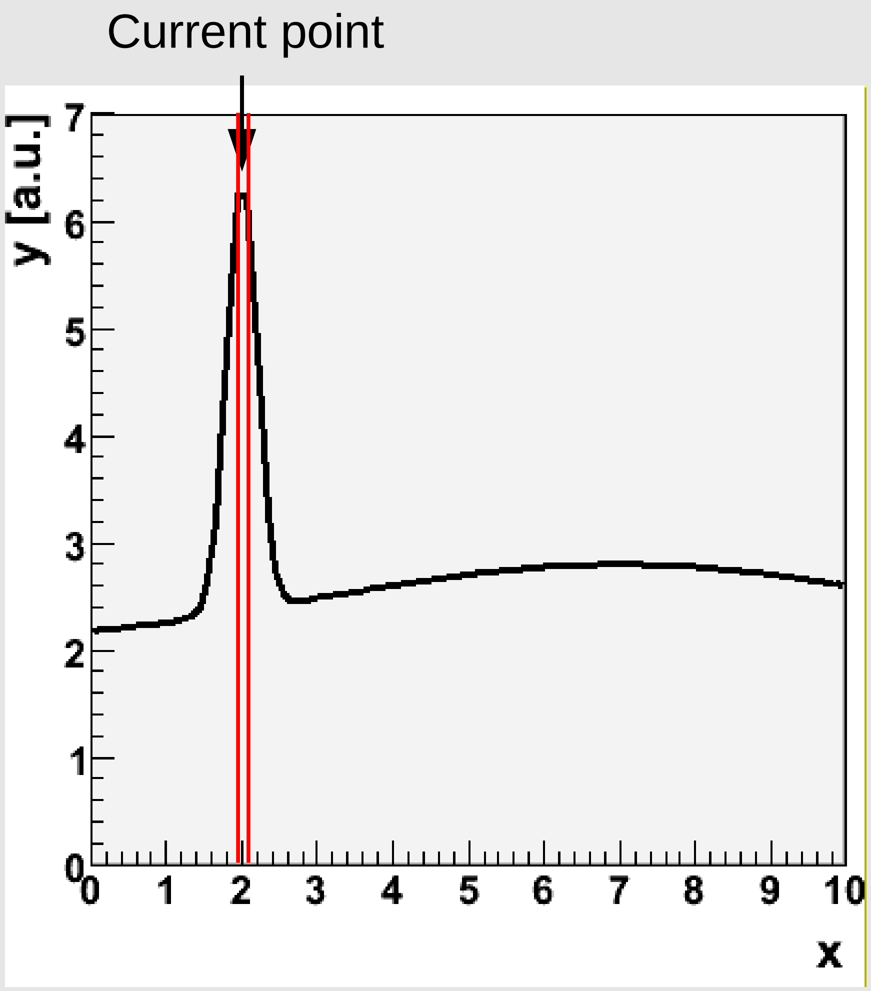
- Do not change efficiency during main run → bias.
- Form of proposal function can be changed. ★

GEORG-AUGUST-UNIVERSITÄT GÖTTINGEN

MCMC in BAT: Proposal function



Large sampling range
Small efficiency



Small sampling range
Large efficiency

MCMC in BAT: Convergence Criteria

Convergence:

- How do you know the chain converged towards the underlying distribution?
- Can build criteria which give hints
- Two types of convergence:
  single chain: compare chain and distributions or chain with itself
  several chains: compare chains to each other
- **Convergence criteria for single chains:**
  - could use, e.g., auto-correlation function, etc.
  - can be CPU-intensive
  - can be tested offline if chain is stored to file
  - not yet implemented. ★

MCMC in BAT: Convergence Criteria

Convergence:

- **Convergence criteria for several chains:**
  - can be tested offline if chains are stored to file
  - use approach from *Gelman & Rubin, StatSci 7, 1992*:
    - compare mean of variances of single chains with variance of target distribution.
    - define **r-value criterion:** $r = \sqrt{\frac{\hat{V}}{W}}$

$$W = \frac{1}{m}\frac{1}{n-1}\sum_{j=1}^{m}\sum_{i=1}^{n}(x_i - \bar{x}_j)^2$$

Mean of variances of all chains

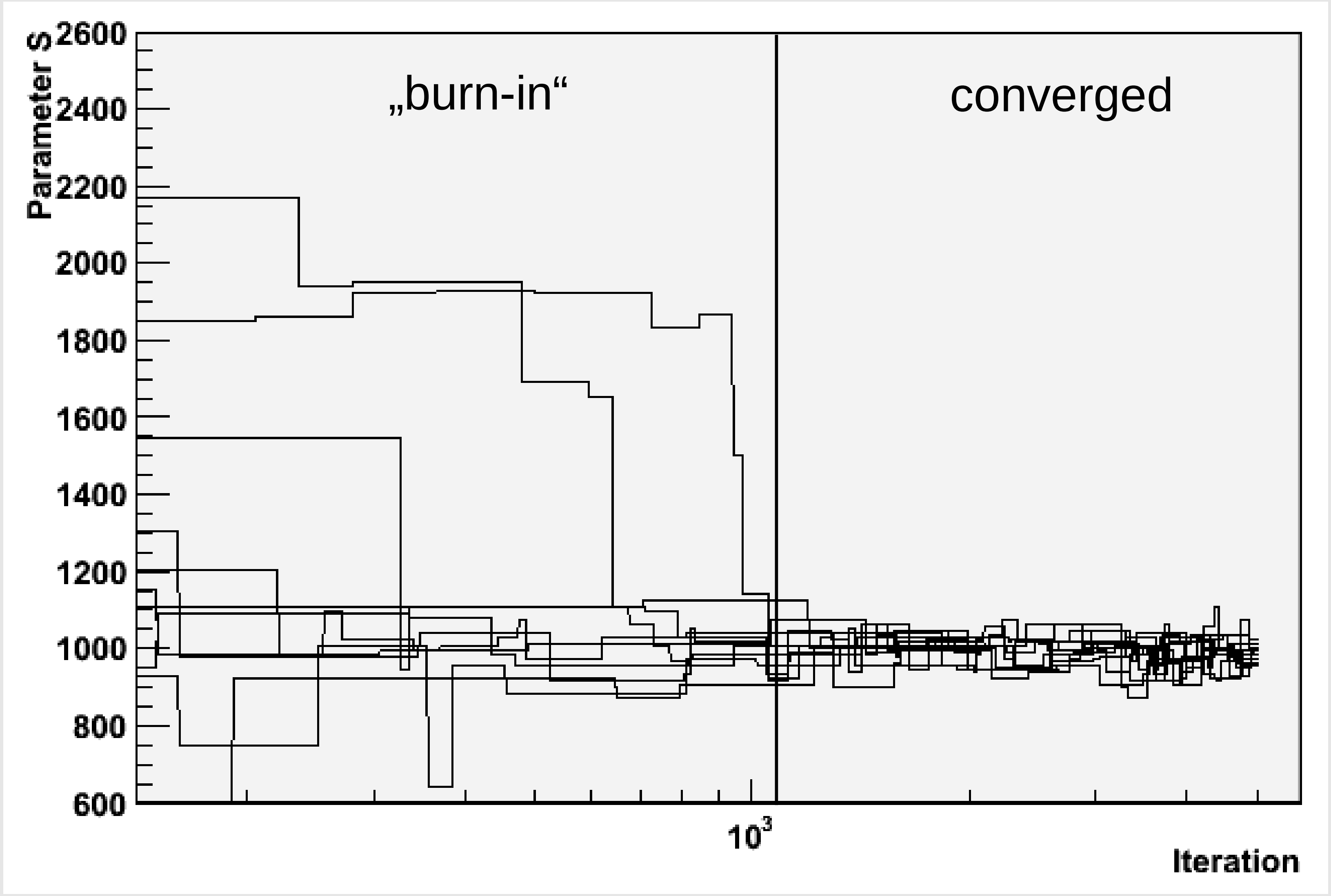
$$\hat{V} = (1-\frac{1}{n})W + \frac{1}{m-1}\sum_{j=1}^{m}(\bar{x}_j - \bar{x})^2$$

Estimate variance of target distribution

- require: (r-1) < 0.1 for each parameter and log prob.

- **General question:** Use one long chain or several shorter ones?

MCMC in BAT: Convergence Criteria



MCMC in BAT: Parameters of Running Chains

Option 1: „fast mode“:

- Single chain ran for a fixed number of iterations.
- Short pre-run to forget initial position.
- No convergence test
- No adjustment of proposal function

Option 2: „precision mode“:

- 5 chains ran in parallel
- Pre-run including convergence test and adjustment of proposal function
- Default number of iterations: 100,000

In general:
User can set all parameters of running individually:

- Iterations, convergence criteria, pre-run, ...
- Requires some knowledge or intuition on the problem studied.

Input/Output

Input to BAT:

- Possible input formats:
  - User defined (on the fly)
  - ROOT trees
  - ROOT histograms
  - ASCII files
- Data-classes for easy handling of data.
- Depending on model classes, input can be made available by user.
- For fitter classes:

- TH1D	(Poisson)
- 2 x TH1D	(Binomial)
- TGraphErrors	(Gauss)

Output from BAT:

- Log file with processing details
- ASCII summary to screen and to file:
  - Model summary
  - Mean values, modes, etc.
  - Posterior probabilities
- Plots:
  - Marginalized distributions (1-D)
  - Correlation plots (2-D)
  - Fit function and error band
- ROOT-file:
  - Summary data
  - add distributions
  - **Markov chain (all points)**

Example: Likelihood

Fit a function to spectrum:

- Fit function *f* is Gaussian (3) + const. (1)
- 4 parameters
- No prior knowledge on the parameters, i.e., p_0=const.
- Assume that Poissonian fluctuation per bin are independent, i.e., conditional probability is product of Poisson terms:

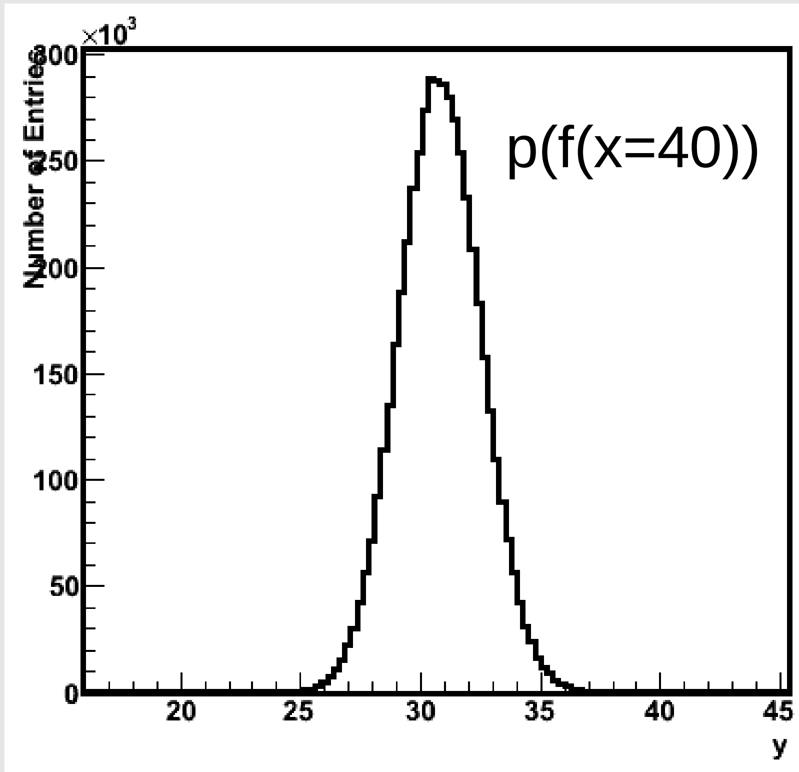
$$p(D|S,\mu,\sigma,B)=\prod_{i=1}^{N_{bins}}\frac{\lambda_i^{n_i}}{n_i!}e^{-\lambda_i}$$

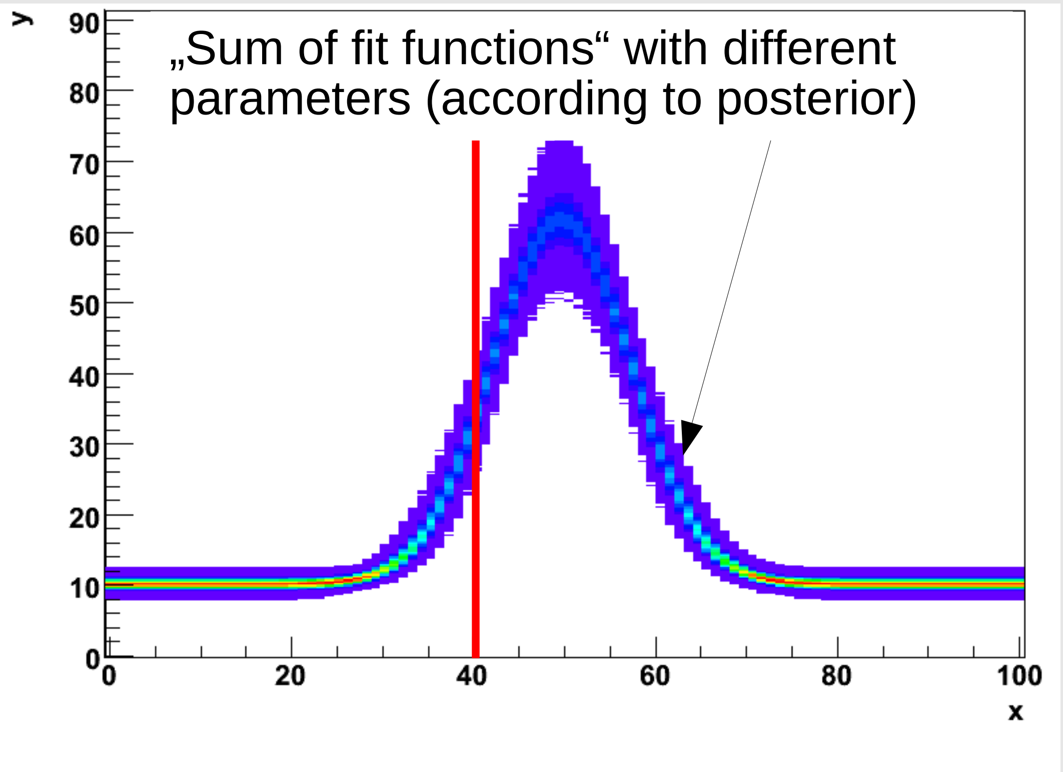
$$\lambda_i=\int_{\Delta_i}\frac{1}{\sqrt{2\pi}\sigma}e^{\frac{-(x-\mu)^2}{2\sigma^2}}dx+\frac{B}{\Delta_i}$$

GEORG-AUGUST-UNIVERSITÄT GÖTTINGEN

Extraction of Uncertainty Band

While sampling through parameter space:
calculate *f* at every x (at that point in parameter space)





At x = 40: probability (density) for function f to have value y
Use this to define uncertainty band (e.g., use 68% center interval)

Goodness-of-fit

How to judge if the model describes the data (at all)?

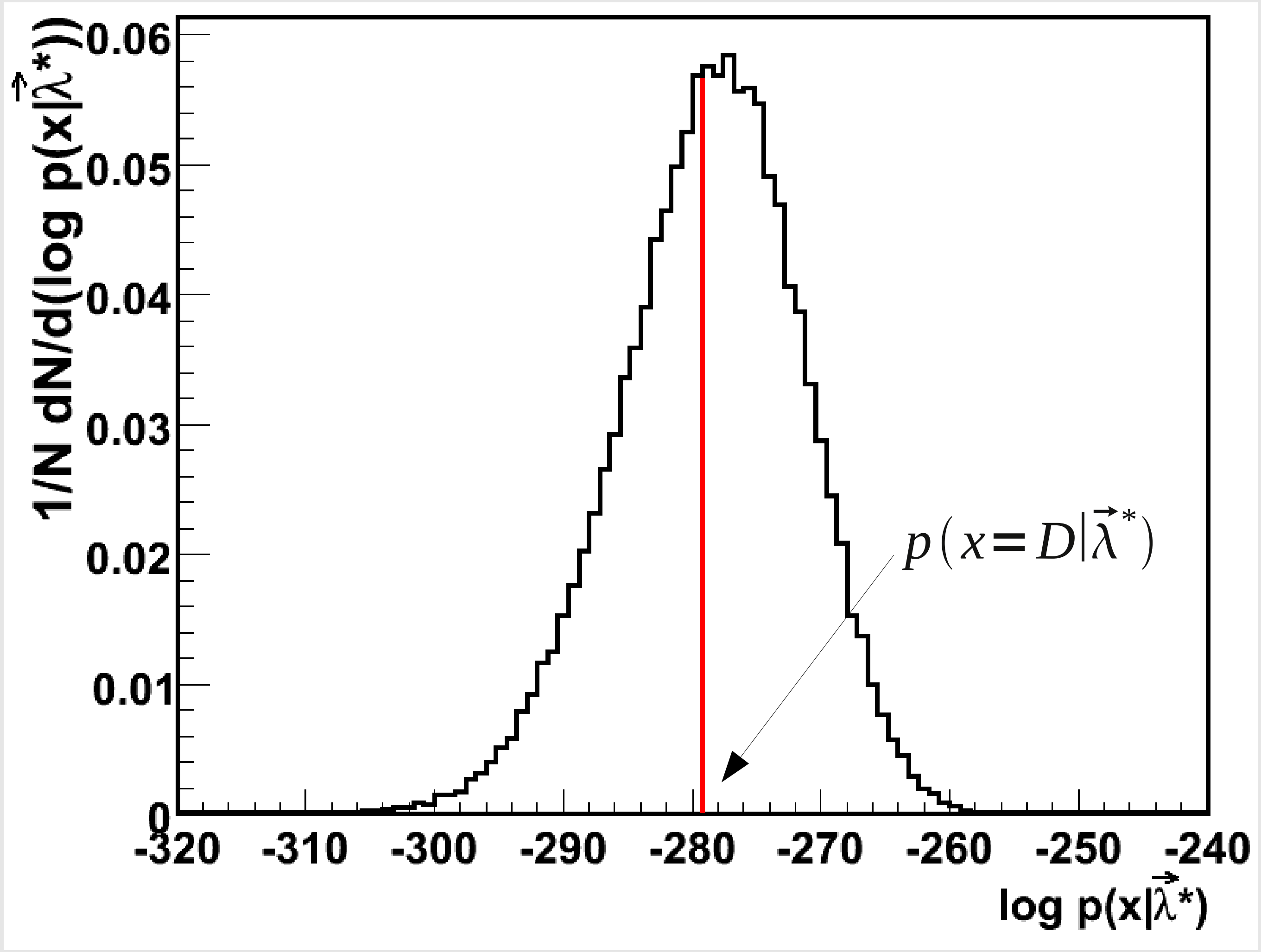
Strategy:

- Find best parameters from original data, $\vec{\lambda}^*$
- Generate data sets, *x*, using these parameters (ensemble tests)
- Calculate and histogram the probability $p(x|\vec{\lambda}^*)$
- Compare prob. distribution with prob. of original data, *D,* $p(x=D|\vec{\lambda}^*)$
- **Calculate the *p*-value:**
  - probability that $p(x|\vec{\lambda}^*)$ is equal to or less than $p(x=D|\vec{\lambda}^*)$
  - Returns value between 0 and 1
  - Large *p*-value means good agreement
- **Note:** for the Gaussian case, the *p*-value is equivalent to a X^2-probability.

Goodness-of-fit

For a good model, the probability to find a better matching between data and theory (larger likelihood) should be small.

Here:
p-value = 0.49



$1/N\ dN/d(\log p(x|\vec{\lambda}^*))$
0.06
0.05
0.04
0.03
0.02
0.01
0
-320
-310
-300
-290
-280
-270
-260
-250
-240
$\log p(x|\vec{\lambda}^*)$
$p(x=D|\vec{\lambda}^*)$