BAT — The Bayesian Analysis Toolkit

Allen Caldwell, Daniel Kollár, Kevin Kröninger

Cluster of Excellence for Fundamental Physics — Ringvorlesung

München, Garching, 14.5.2008

Aims of data analyses

- Compare data and models
- Judge validity of models
- Estimate model parameters

BAT → Software package to solve statistical problems using Bayesian approach

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

- Provide flexible environment to phrase arbitrary problems
- Provide set of numerical tools
- C++ based framework (flexible, modular)
- Interfaces to ROOT, Cuba, Minuit, user defined, ...

Building blocks / Implementation

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

Program flow:

USER DEFINED

- create model
- read-in data

MODEL INDEPENDENT (common tools)

- normalize
- find mode / fit
- test the fit
- marginalize wrt. one or two parameters
- compare models
- nice output

Define MODEL

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

Read DATA

- from text file, ROOT tree, user defined

Common tools

- Integration
  - Monte Carlo (sampled mean)
  - Importance sampling
  - CUBA (Vegas,...)
- Optimization
  - Monte Carlo (hit & miss)
  - Metropolis
  - Interface to Minuit
- Marginalization
  - Markov Chain Monte Carlo (MCMC)
- Validation
  - Ensemble testing and *p*-value
- Error propagation
  - Calculate any value of the parameters during the run

Key tool: Markov Chain Monte Carlo

MCMC — Metropolis algorithm

- In BAT implemented Metropolis algorithm
- Map function ***f(x)*** by random walk towards higher probabilities
- Algorithm:

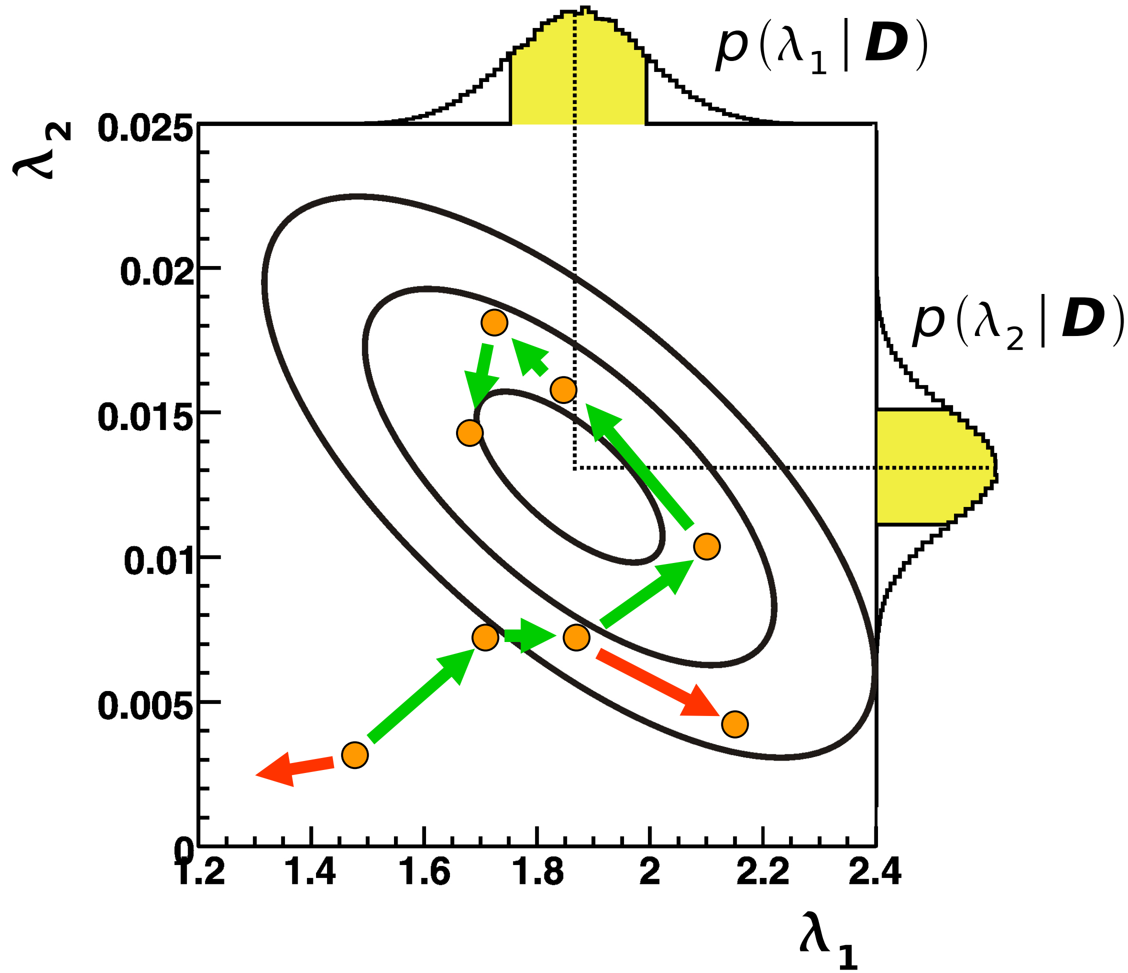
  - Start at some randomly chosen x_i
  - Randomly generate y
  - Set to x_{i+1} to y with probability $p = \min\left(\frac{f(y)}{f(x_i)}, 1\right)$
  - Otherwise $x_{i+1} = x_i$
  - Repeat

- Sampling is enhanced in regions with higher values of ***f(x)***

Scanning parameter space with MCMC

- In BAT, use MCMC to scan parameter space of $\vec{\lambda}$
- $f(\vec{\lambda}) = p(\boldsymbol{D} \mid \vec{\lambda})\, p_0(\vec{\lambda})$
- MCMC converges towards underlying distribution
  - Determining of the overall probability distribution of the parameters $p(\vec{\lambda} \mid \boldsymbol{D})$
- Marginalize wrt. Individual parameters while walking → obtain $p(\lambda_i|\boldsymbol{D})$
- Find maximum (mode)
- Error propagation

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

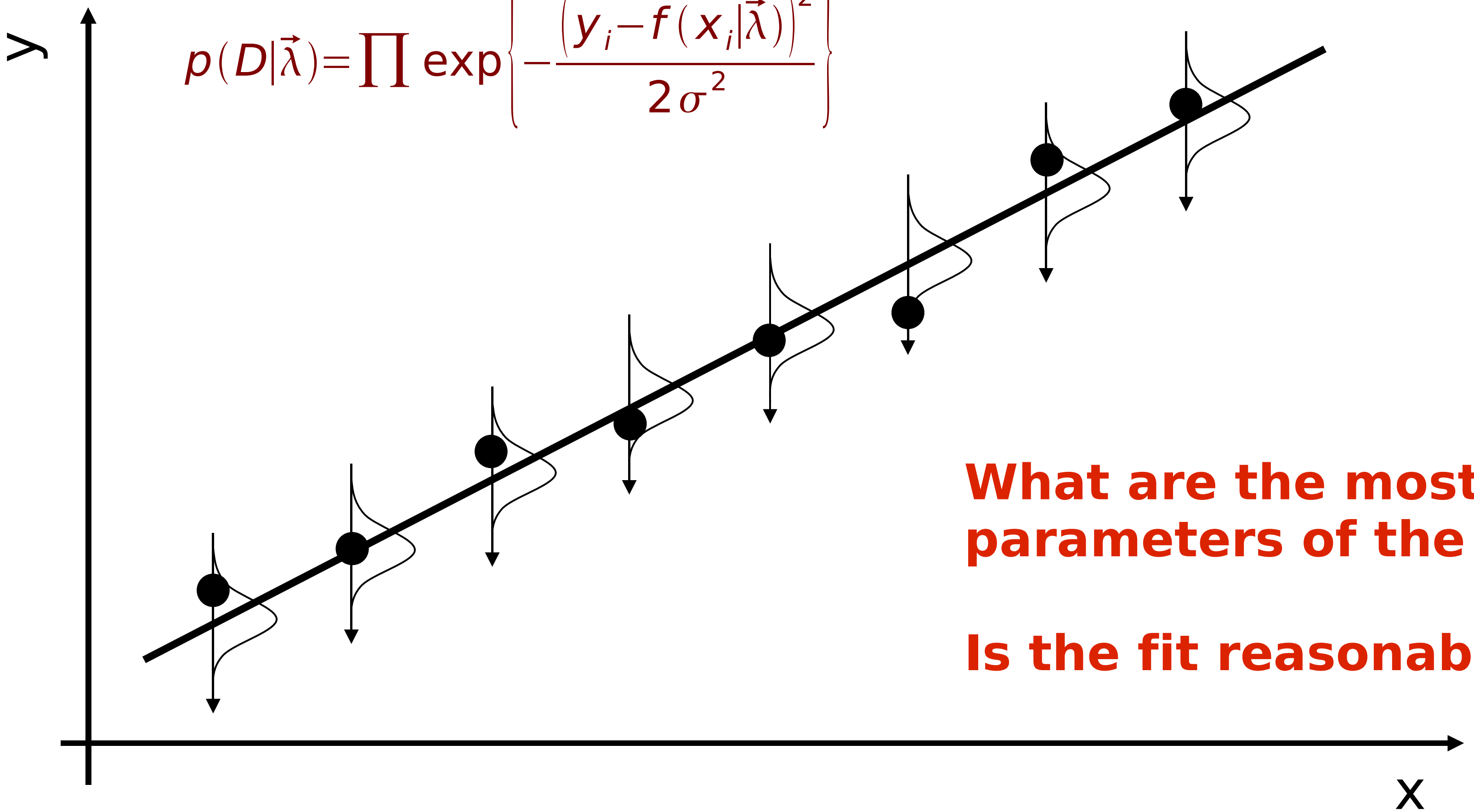


The concept

- Fit points (*x*,*y*) assuming Gaussian distribution in *y* around the function value at each *x*

Likelihood → Product of Gaussians:

$$p(D|\vec{\lambda}) = \prod \exp\left\{-\frac{\left(y_i - f(x_i|\vec{\lambda})\right)^2}{2\sigma^2}\right\}$$



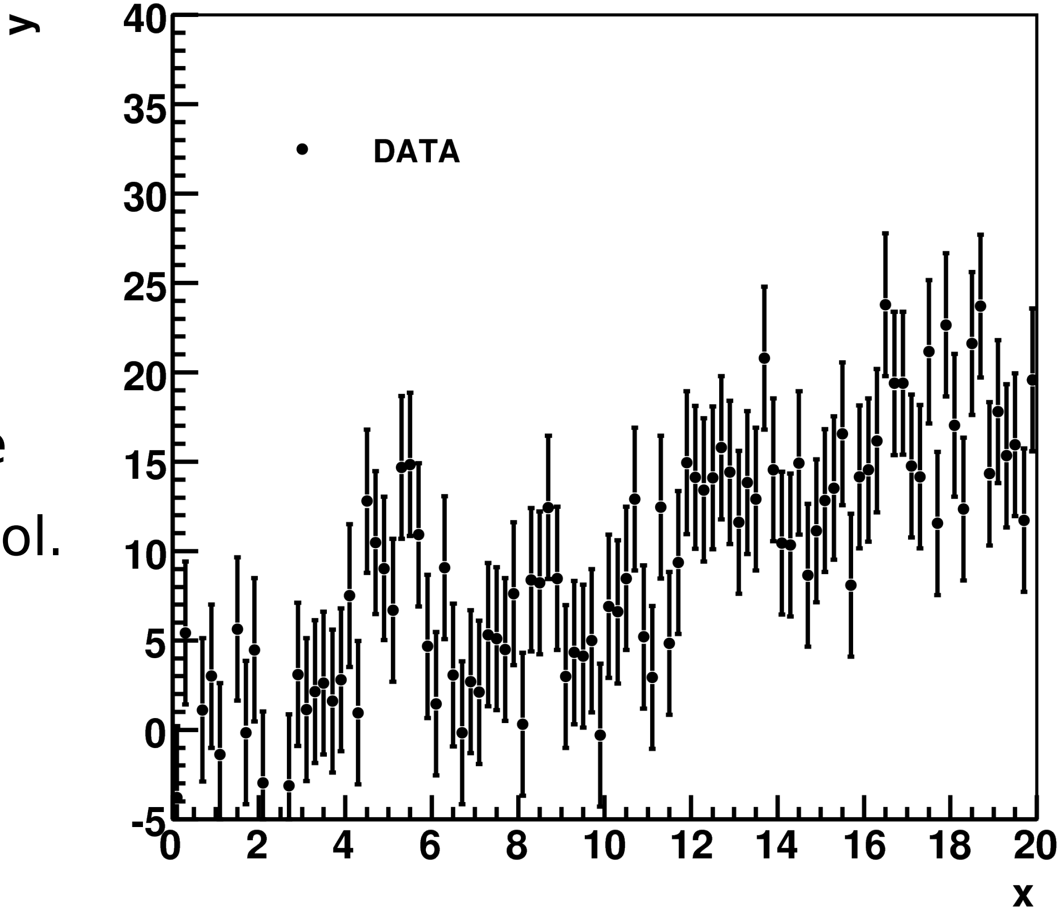
What are the most optimal parameters of the function?

Is the fit reasonable?

Peak on background

- Fit data set using:
  - 2nd order polynomial (no peak)
  - gaussian peak + constant
  - gaussian peak + straight line
  - gaussian peak + 2nd order pol.

- Assume flat a priori probabilities in certain ranges of parameters, i.e. $\boldsymbol{p_0(\lambda)}$ = const.



- Search for peak in range from 2. to 18. with maximum sigma of 4.

- Data were generated as gaussian peak + 2nd order polynomial (peak at x=5.)

Example code: Model definition

USER MODEL EXAMPLE – **2nd order polynomial** (model class)

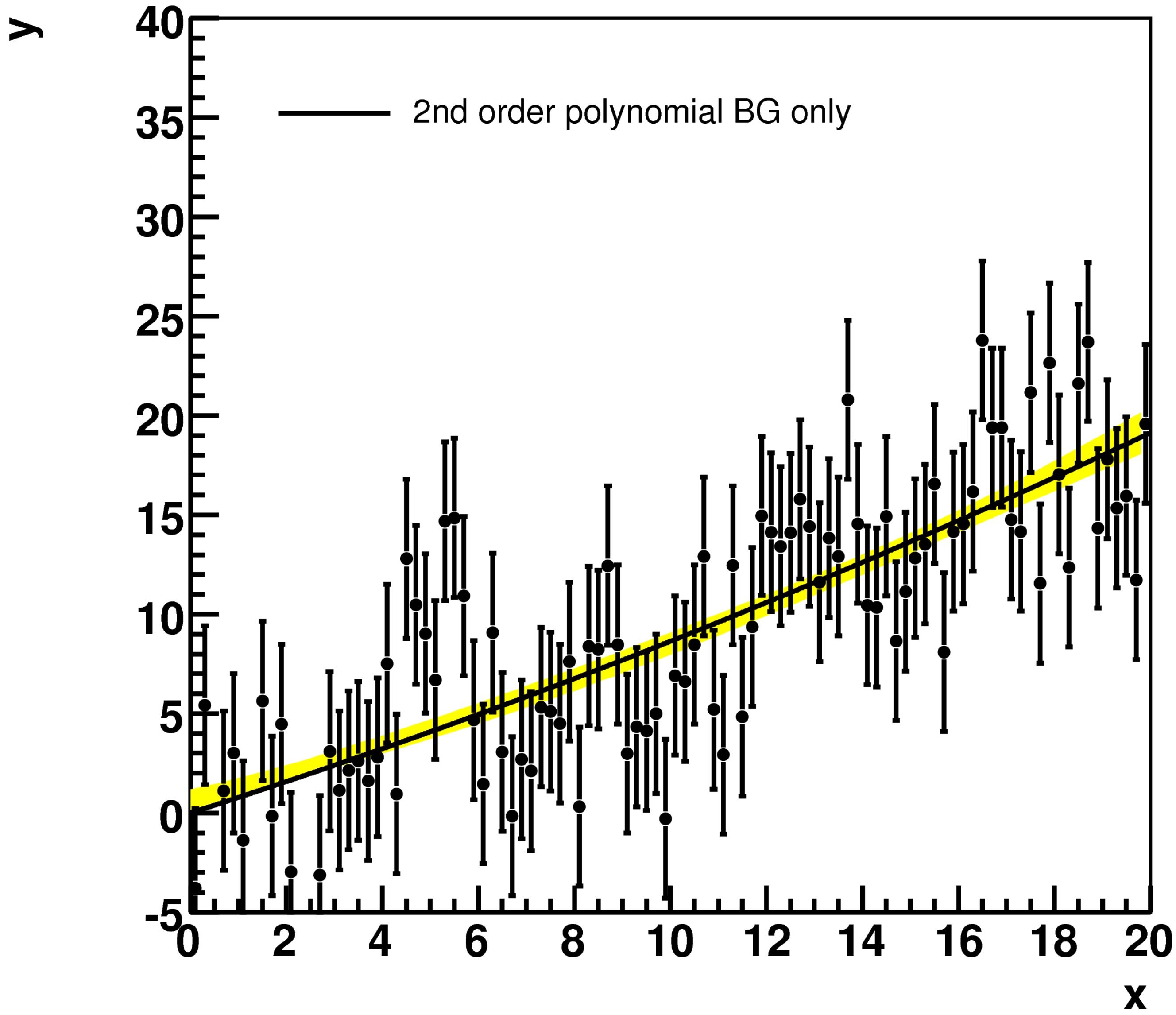
```
double BCModelPol2::DefineParameters() {  // define parameters of the model
     this->AddParameter("offset",  0.,   5.);   // index 0
     this->AddParameter("slope",  -0.,   1.2);  // index 1
     this->AddParameter("quad",   -0.1., 0.1);  // index 2
}

double BCModelPol2::Likelihood(vector <double> params) { // define likelihood
     double   prob = 1.;
     double offset = params[0];
     double  slope = params[1];
     double   quad = params[2];
     for(int i=0;i<this->GetNDataPoints();i++) {
          DataPoint * data = this->GetDataPoint(i);
          double    x = data[0];
          double    y = data[1];
          double yerr = data[2];
          prob *= TMath::Gaus(y, offset + x*slope + x*x*quad, yerr, true);
     }
     return prob;
}

double BCModelPol2::APrioriProbability(vector <double> params) { // define prior
     return 1.;  // flat prior probability for all parameters in their range
}
```

Example code: Main program

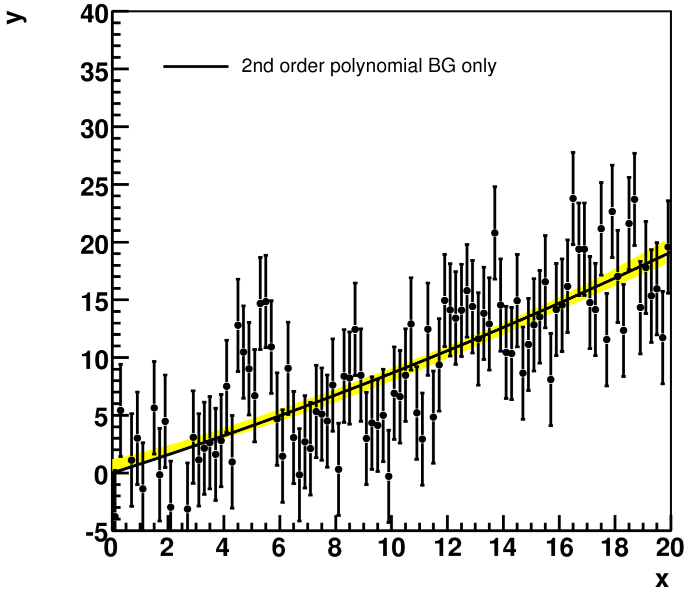
USER MODEL EXAMPLE – **2nd order polynomial** (simple main program)

```
int main()
{
      BCModelPol2 * mymodel = new BCModelPol2("2Dpol");  // create model object

      DataSet * mydata = new DataSet("measurement1");  // create data object
      mydata->ReadDataFromFileTxT("measurement1.dat",3); // read in data, 3 columns: x,y,yerr

      mymodel->SetDataSet(mydata);  // assign data to model

      mymodel->Normalize();  // integrate to get the normalization

      // marginalization
      mymodel->MarginalizeAll();
      mymodel->Marginalize("offset")->Print("mymodel_1D_offset.ps");
      mymodel->Marginalize("slope","quad")->Print("mymodel_2D_slope_quad.ps");

      mymodel->PrintSummary();

      // add more things to do

      return 0;
}
```

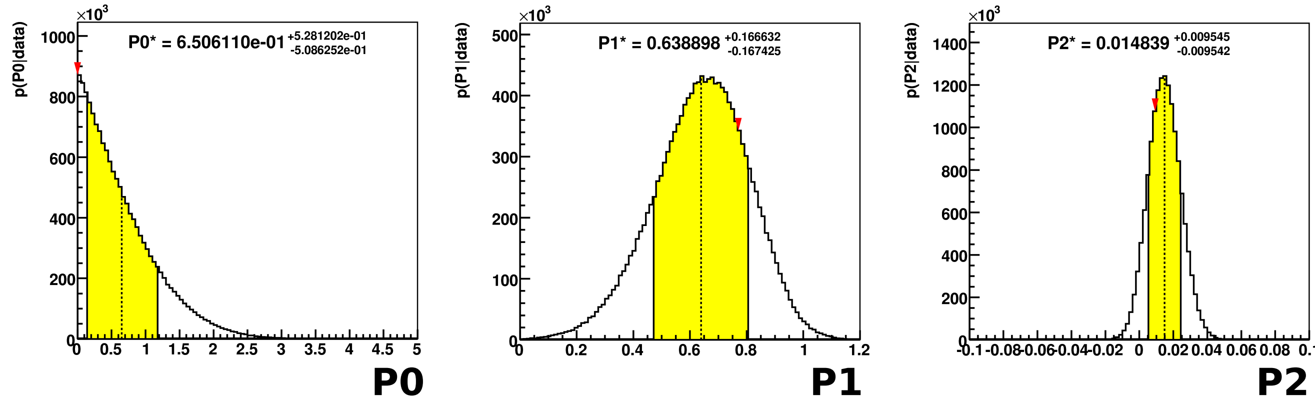
Fit for 2nd order polynomial



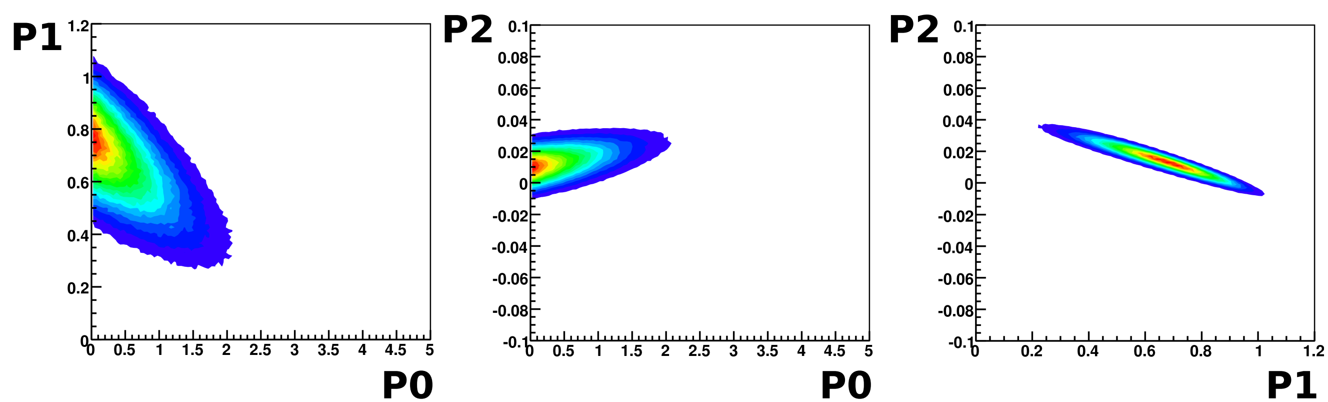
Extracted distributions



Marginalized probability distributions



Correlations

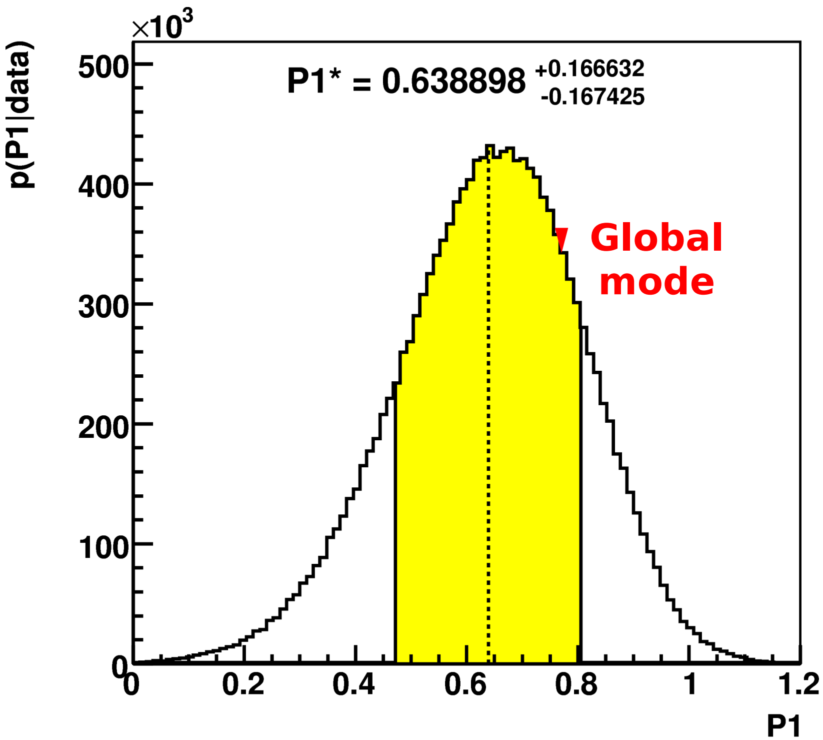


- All distributions including error band obtained during single MCMC run
- Distributions stored as 1D & 2D histograms
- Markov chain stored as ROOT tree

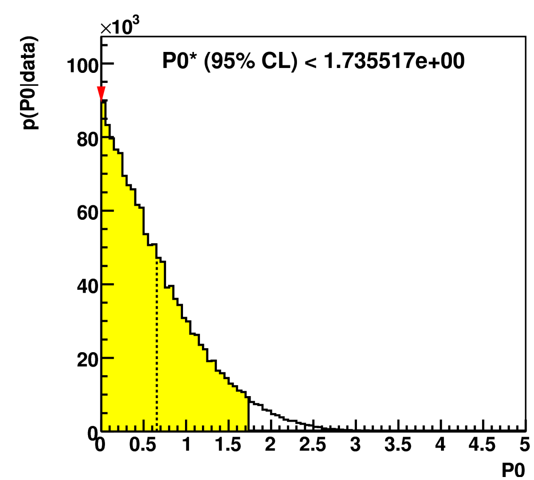
Probability distribution for single parameter

Marginalized probability wrt. one parameter

$p(P1 \mid \text{data})$



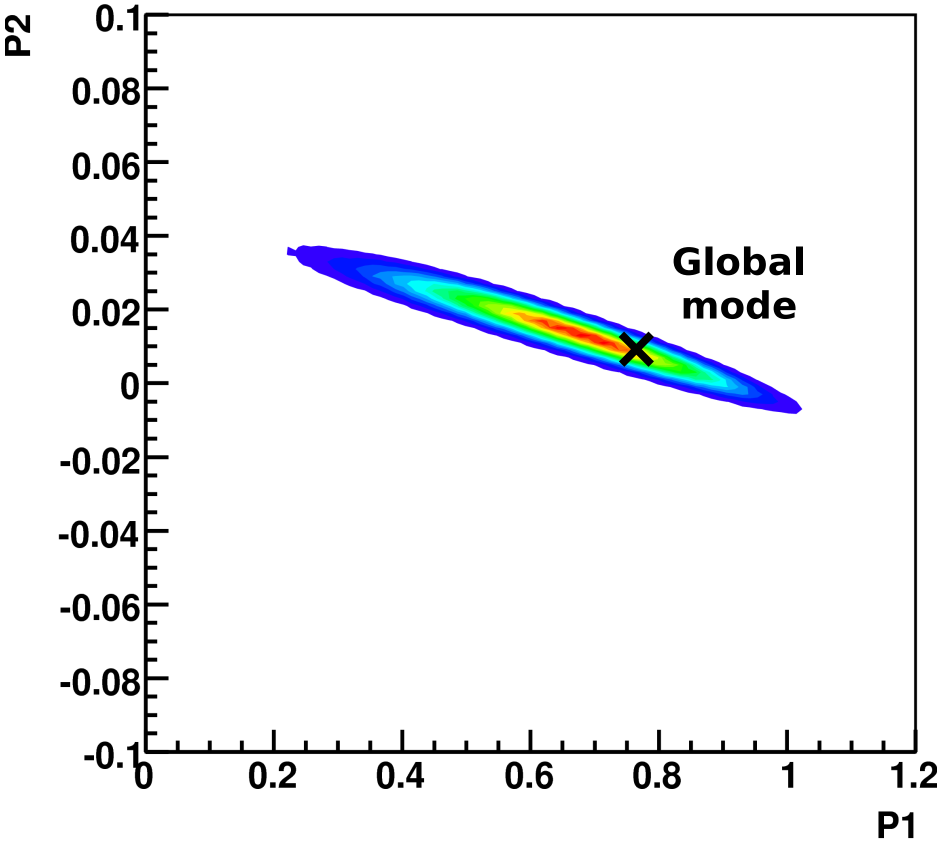
- Integrated over all other parameters (P0 and P2)
- In general, mode of the marginalized distribution not equal to global mode
- Extracted values left to the user
- Default output:
  - Mean
  - Central 68% interval
  - Confidence limits
  - Mode
  - Median



All information about the probability distribution is in the Markov chain

Probability distribution for two parameters

Marginalized probability wrt. two parameters — correlation



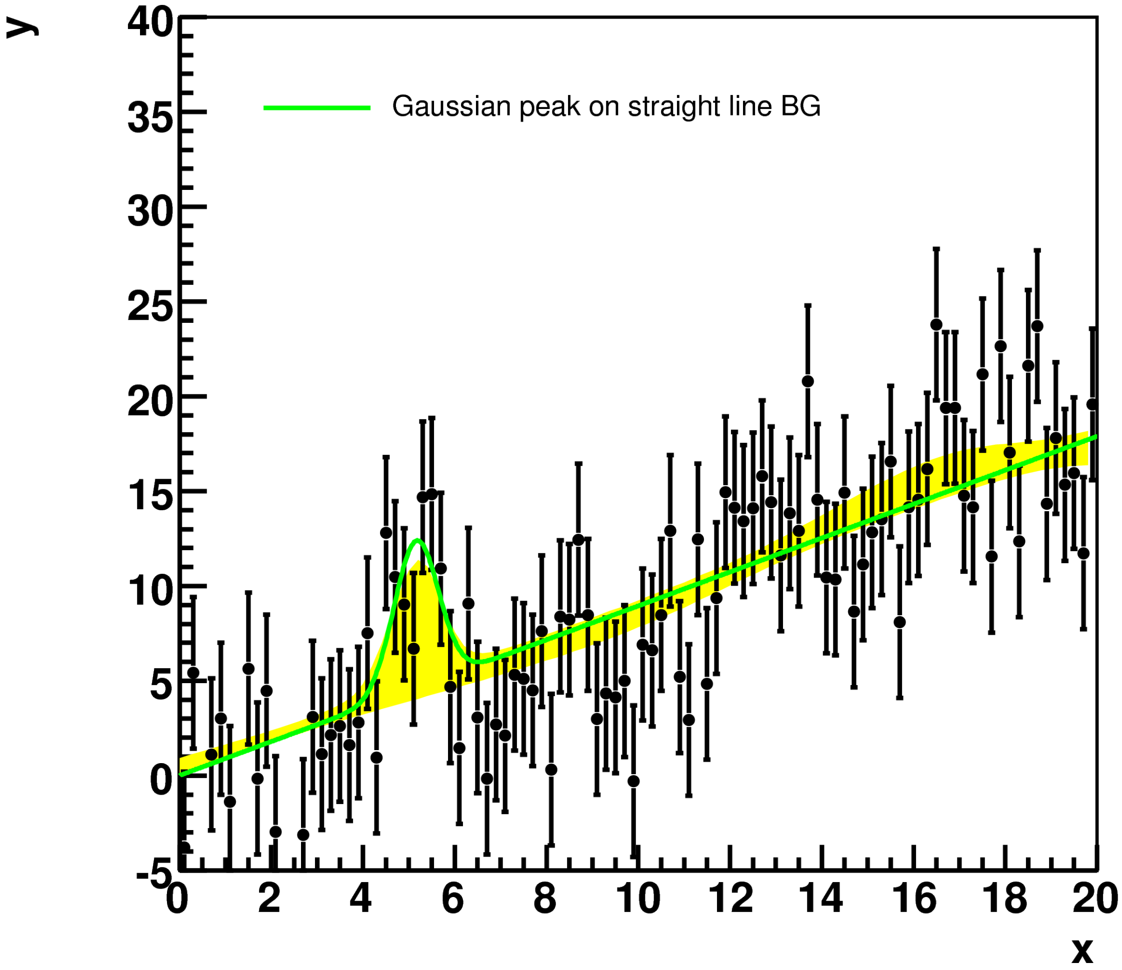
$$p(\mathrm{P1}, \mathrm{P2} \mid \mathrm{data})$$

- Integrated over all other parameters (P0)
- In general, mode of the marginalized distribution not equal to global mode
- Extracted values left to the user
- Default output:
  - Mean
  - 68% contour
  - Confidence limit contours
  - Mode

All information about the probability distribution is in the Markov chain

Fit for Peak + straight line 1

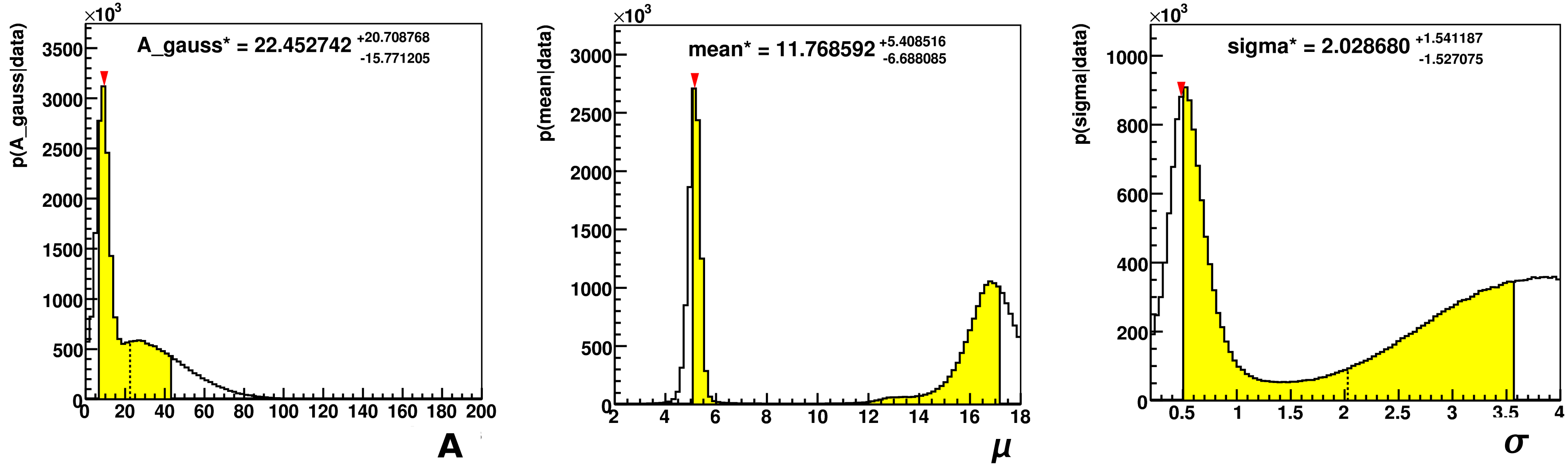
Total of 5 parameters — 1D marginalized distributions: 5
— 2D marginalized distributions: 10



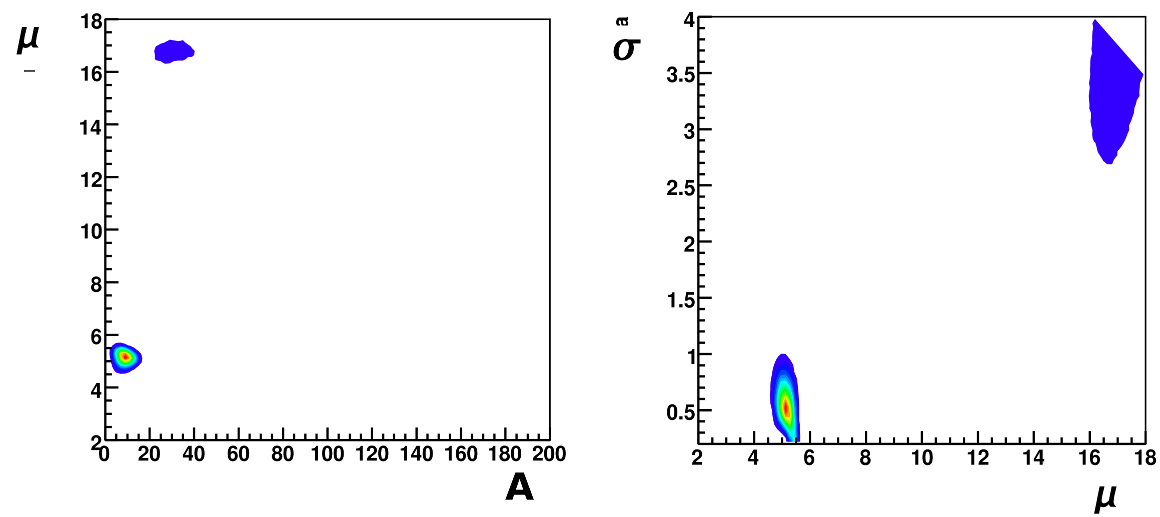
- Best fit (mode) is outside the 68% error band
- Error band has different shape

Fit for Peak + straight line 2

Marginalized probability distributions



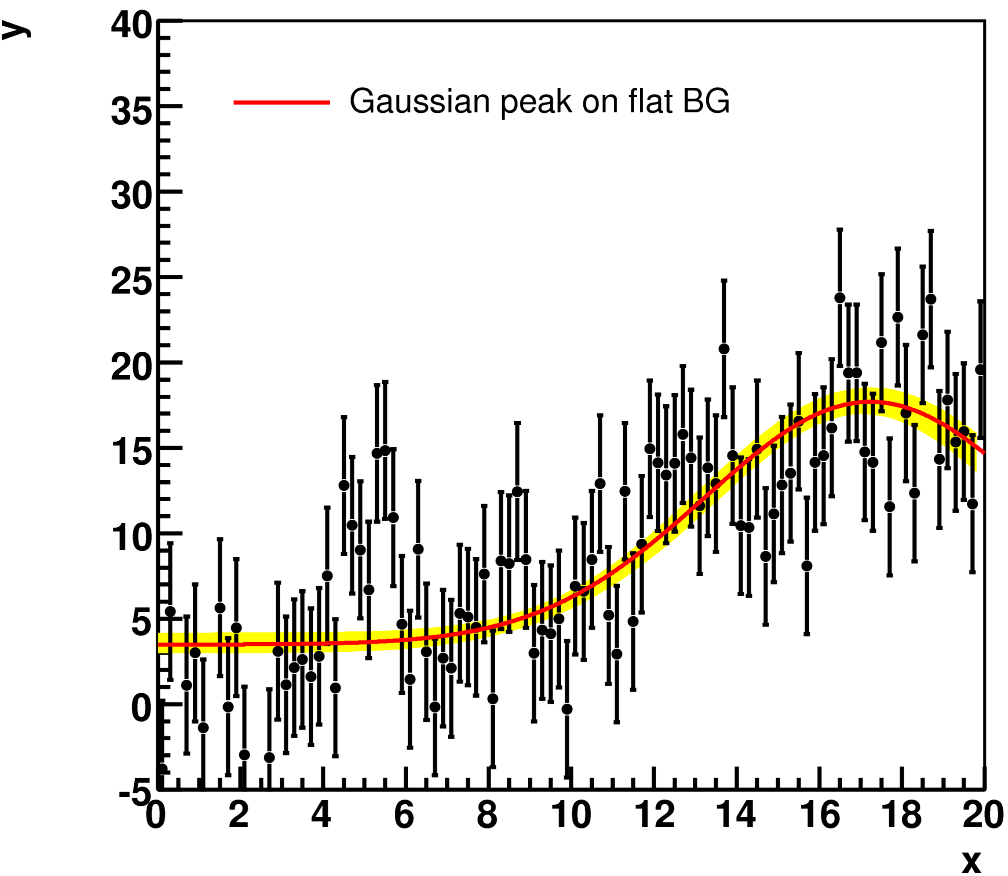
Correlations



- Double maximum in parameter space
- MCMC follows probability distributions with complicated shapes

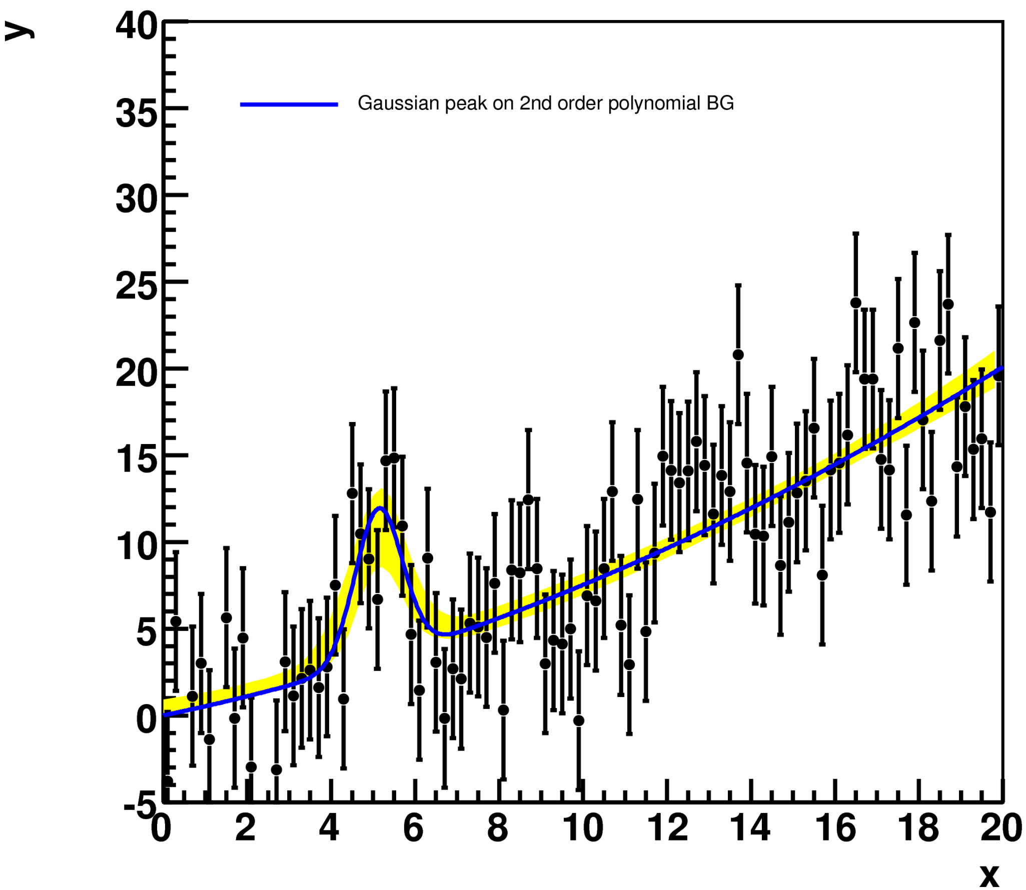
Remaining fits

Peak + const.



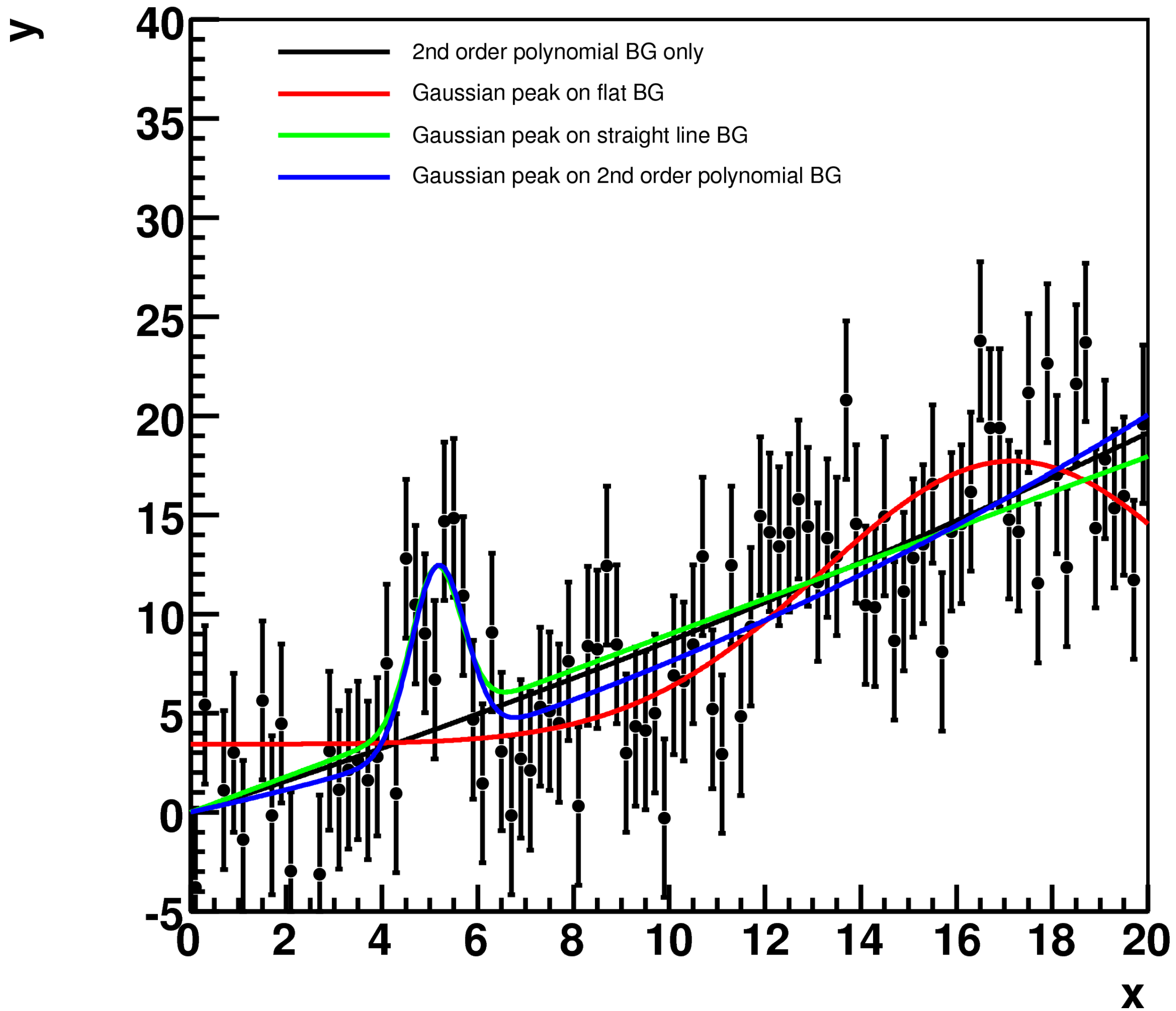
Total of 4 parameters
— 1D marginalized distributions: 4
— 2D marginalized distributions: 6

Peak + 2nd order polynomial



Total of 6 parameters
— 1D marginalized distributions: 6
— 2D marginalized distributions: 15

All models



Which model gives the best description of the data?

Do Goodness-of-fit test

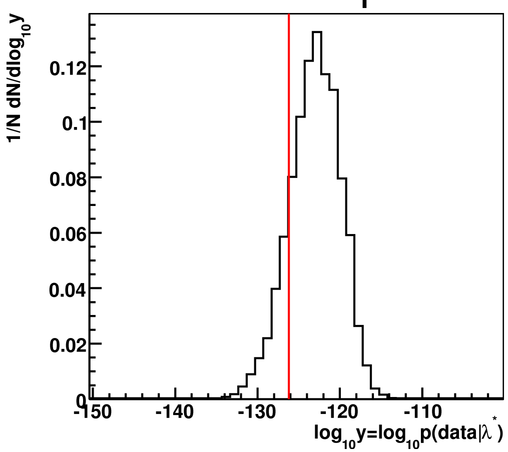
What is the probability to observe the data given the model and the best fit parameters?

Ensemble tests:

- Generate data sets given the model and the best fit parameters
- Calculate likelihood for each data set
- Compare the likelihood distribution to the likelihood of the original data
- Calculate p-value
  - Probability to find a dataset with likelihood less that the original data
  - Value between 0 and 1
  - High p-value means good description of the data by the model

For each model generated 5000 ensembles assuming best fit values

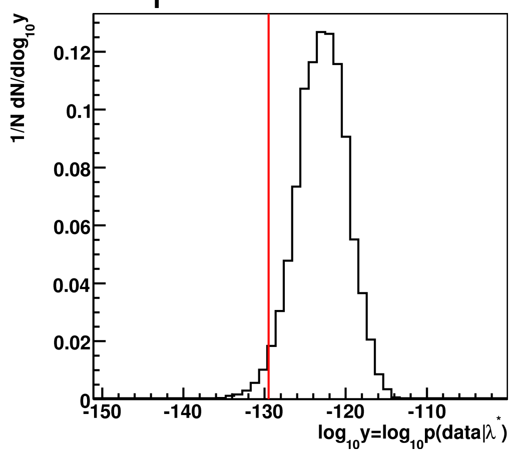
2nd order pol.



p-value = 0.232

Good fit

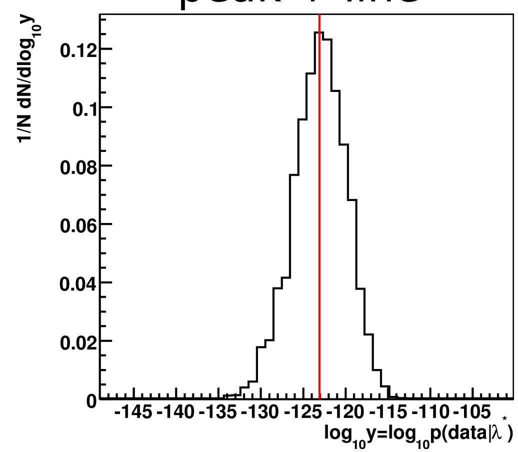
peak + const



p-value = 0.0406

Not good fit

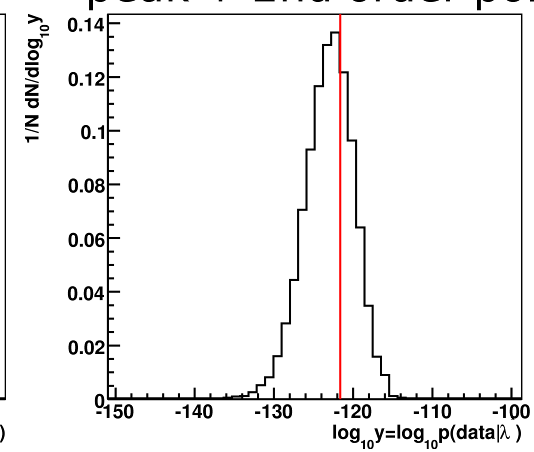
peak + line



p-value = 0.540

Good fit

peak + 2nd order pol.



p-value = 0.778

Good fit

Occam's razor: Use the simplest model/theory describing your data.

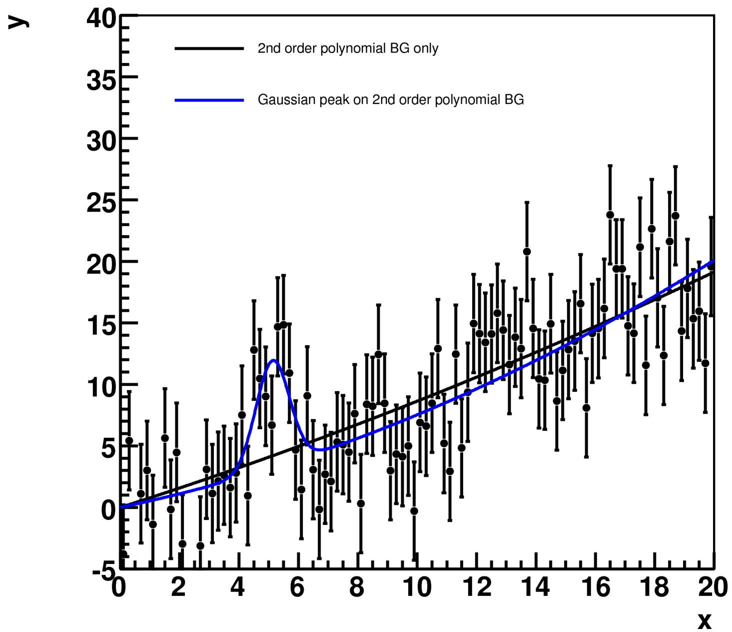
↳ Choose “2nd order polynomial” model

↳ If one knows that peak should be present, choose “peak+line” model

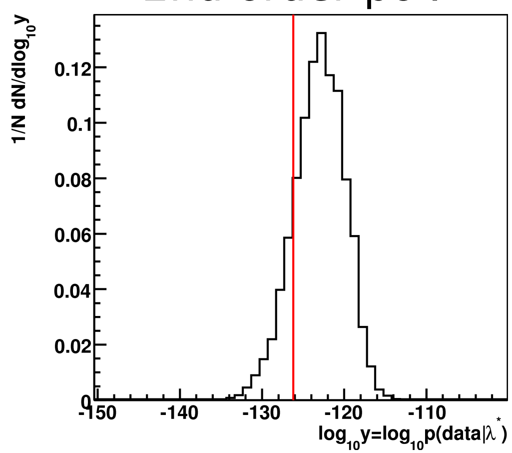
Signal of new physics on SM background

Now suppose that:

- the Standard Model (SM) background is quadratic
- New physics predicts signal peak in the range 2-18

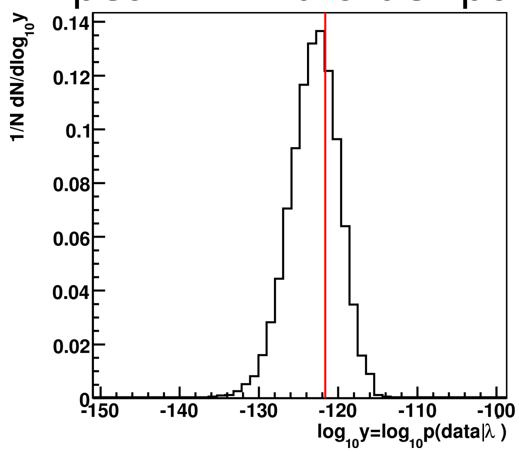


2nd order pol.



p-value = 0.232

peak + 2nd order pol.



p-value = 0.778

- SM gives good description of the data
- It is not possible to claim an evidence or discovery of new physics
  - More precise measurement is required

The BAT package

- Allows to solve simple statistical problems like function fitting as well as complex Data vs. Theory comparisons and parameter extractions
- Close to releasing 0th version to testers with good nerves
  - Hopefully sometimes this (or next) month
  - Bear with our programming skills, we're physicists ☺
- Publication on BAT in preparation
- ROOTified version being worked on

- Students (both Diploma and PhD) to work on BAT development are very welcome

BACKUP

Some details of MCMC implementation

- Running several chains in parallel (default is 5)
- Start at random locations in allowed parameter space
- Initialize chains by doing a pre-run to achieve convergence
  - Defined using r-value
    - Ratio of the mean of the RMS values of the probability and the RMS of the mean values
    - Convergence criterion r < 0.1
- Steps in parameter space done consecutively for each parameter and chain
- Proposal function for new steps is chosen flat with varying ranges
- The efficiency for accepting new point is evaluated for each parameter and chain over last 1000 iterations
  - If efficiency > 50%, decrease the step size
  - If efficiency < 15%, increase the step size