Journal of Physics: Conference Series **331** (2011) 072040 doi:10.1088/1742-6596/331/7/072040

BAT – The Bayesian Analysis Toolkit

Frederik Beaujean[1], Allen Caldwell[1], Daniel Kollár[2], Kevin Kröninger[3]

[1] Max-Planck-Institut für Physik, München, Germany
[2] CERN, Geneva, Switzerland
[3] II. Physikalisches Institut, Universität Göttingen, Germany

E-mail: beaujean@mpp.mpg.de

Abstract.

The main goals of data analysis are to infer the free parameters of models from data, to draw conclusions on the models' validity, and to compare their predictions allowing to select the most appropriate model.

The Bayesian Analysis Toolkit, BAT, is a tool developed to evaluate the posterior probability distribution for models and their parameters. It is centered around Bayes' Theorem and is realized with the use of Markov Chain Monte Carlo giving access to the full posterior probability distribution. This enables straightforward parameter estimation, limit setting and uncertainty propagation. Additional algorithms, such as Simulated Annealing, allow to evaluate the global mode of the posterior.

BAT is implemented in C++ and allows for a flexible definition of models. It is interfaced to software packages commonly used in high-energy physics: ROOT, Minuit, RooStats and CUBA. A set of predefined models exists to cover standard statistical problems.

1. Introduction

The main goals of a typical data analysis are to compare model predictions with data, to draw conclusions on the validity of the model as a representation of the data, and to extract the possible values of parameters within the context of a model. The Bayesian Analysis Toolkit [1] is a software package which aims to help the user to meet these goals. BAT is free software and can be obtained from `http://mpp.mpg.de/bat`. Its foundation is Bayes' Theorem which for a single model has the form

$$P(\vec{\lambda}|\vec{D}) = \frac{P(\vec{D}|\vec{\lambda})P_0(\vec{\lambda})}{\int P(\vec{D}|\vec{\lambda})P_0(\vec{\lambda})\,d\vec{\lambda}}\,, \tag{1}$$

i.e., the probability of parameter set $\vec{\lambda}$ given data $\vec{D}$, the *posterior* probability, is proportional to the probability of data given the parameters, also known as the *likelihood*, times the initial probability for the parameters, the *prior* probability[1]. The denominator on the right hand side of the equation, the *evidence*, is just the integral of the numerator over the allowed region of $\vec{\lambda}$ and it ensures that the posterior probability is normalized. The formula can also be interpreted as a learning rule: The knowledge about the parameters of the model (or the model itself) before the experiment, the prior, is updated using the probability of the new data for different values of the parameters, resulting in posterior knowledge.

[1] Throughout the paper the term probability is used for both probability and probability density.

Usage of Bayesian inference for data analysis is well established in the high energy physics (HEP) community. Even though the algorithms for performing calculations typical of Bayesian analyses are well known, for a long time there has not been a standard package written in C++. Instead, the most widely used package, WinBUGS [2], is written in the language R. Unfortunately, WinBUGS is not open source. Therefore, the replication of the non-trivial task of implementing standard algorithms in C++ was very common until recently.

BAT attempts to collect the standard algorithms and tools used in Bayesian analyses into one coherent framework. The technical implementation has been carried out to fulfill two main requirements; i.e. to provide:

i) a flexible framework which allows formulation of arbitrary models,

ii) reliable and fast code for numerical operations.

BAT is implemented in C++ and comes in a form of two libraries. It relies on ROOT [3] functionality for input/output and drawing. The core library consists of a set of classes which provide a general infrastructure, algorithms, output and logging. In the model library, extension classes to solve specific (fitting) problems are bundled together. The functionality covers the mapping of the posterior probability in multidimensional parameter space and the extraction of quantities of interest, such as: estimated values of parameters; uncertainty intervals; correlations between different parameters or parameter limits. It also facilitates the goodness-of-fit testing and model comparison. Emphasis is placed on Markov Chain Monte Carlo (MCMC), which is the key tool of BAT.

This paper provides an update to the exposition of BAT presented at CHEP 2009 [4]. A host of new features has been added to BAT since then. Therefore, after a brief summary of the basic features in Section 2, we focus on the recent updates and improvements in Section 3. Finally, we present an outlook for future releases of BAT.

2. Basics

In the Bayesian approach to data analysis, the state of knowledge about parameters $\vec{\lambda}$ is given by the posterior as defined in Eq. 1. The key task of BAT is therefore to evaluate the posterior. In order to allow for maximum flexibility and easy access to BAT's functionality, the user can define the posterior by inheriting from the class `BCModel` and overloading the two methods `LogLikelihood` and `LogAPrioriProbability`.

Once the data observed, $\vec{D}$, and the names of the parameters, $\vec{\lambda}$, are supplied, the posterior, $P(\vec{\lambda}|\vec{D})$, is specified. In order to calculate credibility regions and to visualize correlations between parameters, it is necessary to obtain the one dimensional (1-D) and two dimensional (2-D) marginal posterior probability distributions. For instance, in the 1-D case, the marginal distribution for parameter λ_i is given by integrating out all other parameters

$$P(\lambda_i|\vec{D}) = \int P(\vec{\lambda}|\vec{D}) d\vec{\lambda}_{j\neq i} \ .$$

Typically this integration cannot be done analytically for real life problems. It is therefore necessary to find an efficient numerical approximation. The solution is MCMC (see e.g [5, 6]). The basic idea is to generate random numbers according to the posterior, $\vec{\lambda} \sim P(\vec{\lambda}|\vec{D})$ and to fill 1-D [2-D] histograms with these posterior samples. In BAT, by default there is a 1-D histogram for each single parameter and a 2-D histogram for each unique combination of any two parameters. The histograms provide good approximations to the desired marginal distributions if a large number of samples is available and if these samples have the correct distribution. For the details of the MCMC implementation in BAT, we refer to our previous paper [4, Sec. 2]. Examples of marginal distributions are shown on the right panel in Fig. 1.

To sample from the posterior, it is not necessary to know the normalization constant (evidence) in the denominator of Eq. 1. However for model comparison, the evidence of a particular model is required. BAT provides a built-in sampled mean algorithm to estimate the evidence. Additionally, an interface to the external library CUBA [7] provides access to a set of highly optimized integration routines. In recent versions of BAT, the option to easily tune CUBA settings has been added.

3. New features

3.1. Simulated Annealing

In complicated problems it is often difficult to find the global mode of the posterior as several distinct local modes may exist. The mode found by gradient-based algorithms like ROOT's MINUIT therefore depends crucially on the starting position. During marginalization the Markov chain explores the parameter space and keeps track of the maximum of the posterior found. After sufficiently many iterations, this provides a good starting position to MINUIT. If, however, the user wants to get a quick estimate of the global mode, it may be much faster to run the Simulated Annealing algorithm [8]. It enjoys the same property as the Markov chain random walk, i.e. one does not get trapped in the local mode nearest to the starting position. In BAT's implementation of Simulated annealing, the user can pick among a Cauchy, Boltzmann or user defined annealing schedule. In addition, starting and threshold temperatures can be defined easily.

3.2. Easy setup of priors

In typical data analysis tasks, the prior distribution, $P_0(\vec{\lambda})$, is often chosen to be of a simple mathematical form. In many cases, it is assumed to factorize; i.e. $P_0(\vec{\lambda}) = \prod_i P_0(\lambda_i)$. In order to accomodate this formulation, it is now possible to set the 1-D priors $P_0(\lambda_i)$ individually using the function objects of class `TF1` provided by ROOT. In addition, the two most commonly chosen forms of prior, the constant, $P_0(\lambda_i) = \text{const}$, and the Gaussian, $P_0(\lambda_i) \propto \exp\left(-(\lambda_i - \mu_i)^2/\sigma_i^2\right)$, are directly available. In order to speed up the Markov chain, constant priors are calculated only once, and cached for future reuse. For maximum flexibility, it is of course still possible to overload the method `BCModel::LogAPrioriProbability` directly to use an arbitrarily complex prior distribution. The new features are illustrated in the following example:

```
int main()
{
   //...

   // create instance of the model and set its name
   MyModel * model = new MyModel("StandardModel");

   // set individual priors
   TF1 * prior0 = new TF1(...);
   model -> SetPrior("p0", prior0);
   model -> SetPriorConstant("p1");
   model -> SetPriorGauss("par2", mean, sigma);

   //...
}
```

3.3. Summary tool

The purpose of the class `BCSummaryTool` is to present the posterior distribution in an easy-to-understand manner. It is particularly useful for problems involving high dimensional parameter

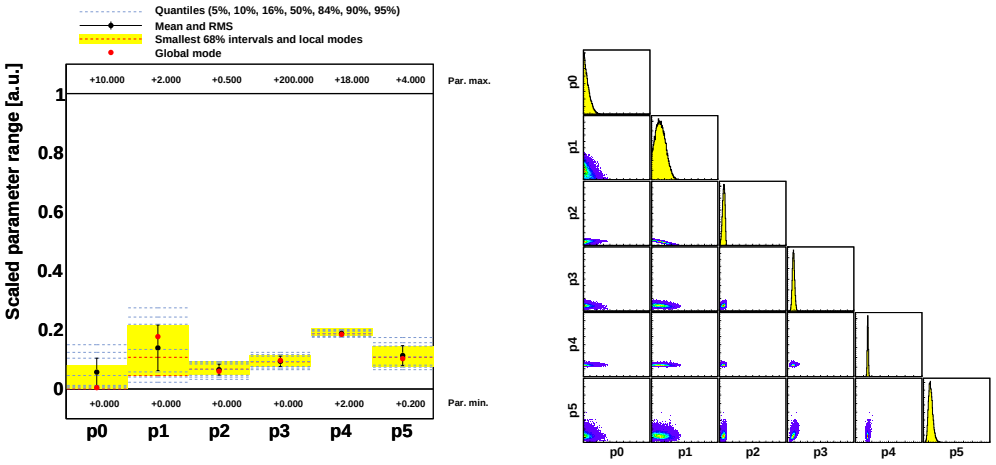

Figure 1. Example output of `BCSummaryTool` for a model with six parameters: (left) box plot summarizing 1-D marginal distributions, (right) all 1-D (diagonal) and 2-D (off-diagonal) distributions in matrix form.

spaces, but `BCSummaryTool` also provides utility methods for low dimensional problems. After the posterior samples have been collected, the following features are available:

- 1-D box plot: all 1-D marginalized posterior distributions are represented side-by-side by the median, mean, variance and the most important quantiles. In addition, the value of each parameter at the global mode of the posterior is shown. The plot is similar in spirit to the Tukey box plot [9]. For an example, see the left panel of Fig. 1.
- 1-D + 2-D marginalized plot: The histograms of all marginal distributions are shown in matrix format. On the diagonal, the 1-D distributions are plotted, while the correlation between parameter i and parameter j can be found from their posterior distribution in the matrix element (i, j), see right panel of Fig. 1.
- correlation coefficients plot: The square matrix of correlation coefficients ρ_{ij} between the individual parameters visualizes the correlation by displaying the coefficients both in text form and using a color palette.
- knowledge update plot: By analyzing the data, the knowledge is updated from the prior to the posterior. In displaying the 1-D [2-D] prior *and* posterior distribution together for each parameter [combination], it is easy to ascertain what effect the data have on the knowledge about that parameter [combination].
- parameter table: For easy inclusion in scientific articles, the main numbers about the parameters like mean, mode etc. can be output conveniently in latex format.

3.4. Template fitter

In an effort to supply not only the basic tools like the MC implementation, but also useful models, the class `BCTemplateFitter` is introduced. The physics motivation is as follows. In particle physics, one often searches for a signal in the presence of a background. As an output of detector simulations, often the signal and the background shape is known in the form of histograms, or templates. The analysis task is then to establish whether a signal contribution exists, and to estimate the signal contribution from the observed spectrum. In the fitting, the templates are assumed to have to statistical uncertainty, while Poissonian fluctuations are assumed for the data bins. Several goodness of fit statistics are implemented to judge the

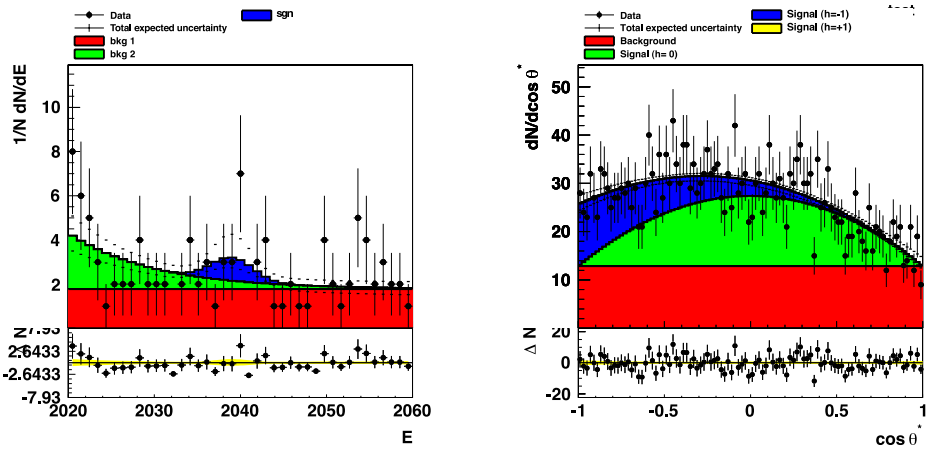

Figure 2. Example `BCTemplateFitter` output of (upper panel) the data and individual template contributions and (lower panel) the residuals and uncertainty band: (left) neutrinoless double beta decay of Germanium 76, (right) spin helicity of the W boson in top decays.

validity of the models quantitatively. Methods to combine uncertainties and impose limits on physical quantities are included. In addition, several plotting routines are available to display the individual template contributions, examples are displayed in Fig. 2.

3.5. RooStats interface

Starting with version 0.4, BAT has a new interface to the ROOT package RooStats [10]. It is now possible to call BAT to calculate limits, credibility intervals etc. for a statistical model defined within RooStats. This is achieved by a new class, `RooStats::BATCalculator`, inheriting from `RooStats::IntervalCalculator`. BAT supplies the desired results in RooStats' native format allowing for easy comparison with the results obtained from using other statistical methods.

3.6. Webpage

BAT is available from the official webpage, `http://mpp.mpg.de/bat`. A lot of documentation and extra information has been added in 2010, to allow new users to quickly get started without a steep learning curve. The main news are:

- Performance: a test suite of sample problems is run for each new release of BAT to check that the MCMC works correctly. To this end, simple models with known, analytical solutions are analyzed and several quantities such as the mean, mode etc. of the posterior distribution are compared. In addition, goodness of fit statistics like χ^2 and the Kolmogorov-Smirnov distance are calculated where applicable. All test results are publicly accessible.
- Tutorials: there are five detailed, step-by-step tutorials ranging from beginner to intermediate level in order to quickly introduce BAT to new users. All tutorials feature plots and the necessary source code to follow the tutorial.
- FAQ: a page collecting frequently answered questions has been set up.

4. Outlook

At present, BAT is used in the analysis of a variety of high energy physics experiments. For example, BAT is part of ATLAS experiment's analysis software Athena [11] and BAT is also a core part in the analysis of the neutrinoless double beta decay GERDA [12]. From release

to release the number of downloads increases, demonstrating the usefulness of BAT in analysis tasks. One might therefore expect that BAT converge to a stable release with only minor modifications. This, however, is not the case.

Currently, work has started on restructuring the code base in order to accomodate a number of new algorithms. In order to deal with fit problems involving $\mathcal{O}(50)$ parameters it is advisable to make use of parallel computing. Since BAT already is a useful tool, the plan is to retain all the existing features described above, but to rewrite BAT in such a modular way that the same algorithm (e.g. sampling from the posterior or calculating the evidence) could be implemented in different ways, making optimal use of the computing architecture available now and in the future. At the same time, this type of modularity will also allow to easily replace different algorithms to select the best one for the particular problem under consideration. This is already possible now; for example to maximize the posterior, the user can choose from MCMC, MINUIT or Simulated Annealing. Certainly there is room to include other search strategies such as particle swarm optimization or genetic algorithms. Another benefit of the new structure would be to allow interfacing with optimized, external libraries more naturally.

In retrospect, it is very promising that most of the planned improvements announced at CHEP09 [4] have been implemented, and, in addition, even more features have found their way into the release. The BAT developers welcome any contribution with the tasks ahead, in particular, comments and feature requests can be sent via email to `bat@mpp.mpg.de`.

References

[1] Caldwell A, Kollár D and Kröninger K 2009 *Comp. Phys. Comm.*
[2] Lunn D J, Thomas A, Best N and Spiegelhalter D 2000 *Statistics and Computing* **10** 325–337 URL `http://www.mrc-bsu.cam.ac.uk/bugs/winbugs/contents.shtml`
[3] Brun R and Rademakers F 1997 *Nuclear Instruments and Methods in Physics Research A* **389** 81–86 URL `http://root.cern.ch`
[4] Caldwell A C, Kollár D and Kröninger K 2010 *Journal of Physics: Conference Series* **219** 032013
[5] Karlin S and Taylor H 1975 *A First Course in Stochastic Processes* (Academic Press)
[6] Gilks W R, Richardson S and Spiegelhalter D J 1996 *Markov Chain Monte Carlo in Practice* (Chapman)
[7] Hahn T 2005 *Computer Physics Communications* **168** 78–95
[8] Kirkpatrick S 1984 *Journal of Statistical Physics* **34** 975–986
[9] McGill R, Tukey J W and Larsen W A 1978 *The American Statistician* **32** 12–16
[10] Verkerke W and Kirkby D 2003 (*Preprint* `physics/0306116`) URL `https://twiki.cern.ch/twiki/bin/view/RooStats/WebHome`
[11] Calafiura P, Lavrijsen W, Leggett C, Marino M and Quarrie D *Interlaken 2004, Computing in high energy physics and nuclear physics* 456–458
[12] Caldwell A and Kröninger K 2006 *Physical Review D* **74**