PROCEEDINGS OF SCIENCE

HistFitter: a flexible framework for statistical data analysis

Jeanette Miriam Lorenz*
Fakultät für Physik, LMU München, Am Coulombwall 1, 85748 Garching, Germany,
Excellence Cluster Universe, Boltzmannstr. 2, 85748 Garching, Germany
E-mail: Jeanette.Lorenz@physik.uni-muenchen.de

Max Baak
CERN, Route de Meyrin 385, 1217 Meyrin, Switzerland

Geert-Jan Besjes
Niels Bohr Institute, University of Copenhagen, Blegdamsvej 17, 2100 Copenhagen, Denmark
E-mail: geert-jan.besjes@cern.ch

David Côté
The University of Texas at Arlington, Department of Physics, Box 19059, Arlington, Texas 76019, United States of America

Aleksej Koutsman
TRIUMF, 4004 Wesbrook Mall, Vancouver, BC V6T 2A3, Canada

We present a software framework for statistical data analysis, called *HistFitter*, that has been used extensively by the ATLAS Collaboration to analyze big datasets originating from proton-proton collisions at the Large Hadron Collider at CERN. Since 2012 HistFitter has been the standard statistical tool in searches for supersymmetric particles performed by ATLAS.

HistFitter is a programmable and flexible framework to build, book-keep, fit, interpret and present results of data models of nearly arbitrary complexity. Starting from an object-oriented configuration, defined by users, the framework builds probability density functions that are automatically fit to data and interpreted with statistical tests.

Internally HistFitter uses the statistics packages RooStats and HistFactory.

A key innovation of HistFitter is its design, which is rooted in analysis strategies of particle physics. The concepts of control, signal and validation regions are woven into its fabric. These are progressively treated with statistically rigorous built-in methods. Being capable of working with multiple models at once that describe the data, HistFitter introduces an additional level of abstraction that allows for easy bookkeeping, manipulation and testing of large collections of signal hypotheses. Finally, HistFitter provides a collection of tools to present results with publication quality style through a simple command-line interface.

The European Physical Society Conference on High Energy Physics
22-29 July 2015
Vienna, Austria

*Speaker.

PoS(EPS-HEP2015)223

1. What is HistFitter and why using it?

HistFitter[1, 2] is a software framework for statistical data analysis, developed in the context of searches for new physics, analyzing proton-proton collisions at the Large Hadron Collider (LHC) at CERN. HistFitter has been used in numerous publications by the ATLAS Collaboration [3], mostly in searches for supersymmetry.

The framework consists of a C++ part dealing with CPU-intensive calculations and a Python part for configuration tasks. It is built on top of the packages ROOT [7, 8], HistFactory[4], RooFit [6] and RooStats [5]. HistFactory and RooFit are used in the construction of parametric models, RooFit for fitting and RooStats for statistical tests.

The innovation of HistFitter is to extend HistFactory, RooFit and RooStats in four key areas, easing the typical statistical analysis in high energy particle physics:

- HistFitter offers a *programmable framework* performing a complete statistical analysis starting from a user-defined configuration file.
- The typical *analysis strategy* in experimental particle physics using control, validation and signal regions is deeply woven into the design of HistFitter.
- In many searches for new physics hundreds to thousands of different models of new physics are tested. HistFitter provides the *bookkeeping* to keep track of all different data models, starting with their construction and concluding with statistical tests in an organized way.
- Many easy-to-use tools allow the *presentation and interpretation* of results in a publication-close quality, as e.g. tables and plots summarizing the analysis results.

2. Data analysis strategy with HistFitter

In particle physics experiments, large samples of data, recorded in e.g. proton-proton collisions at the LHC, are analyzed to measure properties of fundamental particles and to discover new physical processes. In doing so, the observed data is interpreted using external predictions for background and signal processes, e.g. coming from Monte Carlo simulations, which are summarized in statistical models aiming to describe the data. HistFitter configures and builds these statistical models and provides various tools to interpret the data.

The construction and handling of these models in HistFitter is based on the concept of statistically orthogonal control, validation and signal regions - a common concept in particle physics. As shown in Figure 1, signal regions (SRs) are regions in a phase-space, defined by cutting on kinematic variables, that are enriched by a potential signal process while suppressing background processes. The background processes in these signal regions are estimated in a semi-data-driven way using control regions (CRs) enriched in the background processes while being signal-free. As shown in Figure 2, the background predictions (which may come e.g. from an external Monte Carlo simulation) in the control regions are normalized in a simultaneous fit to data. Using transfer factors (the ratio of the expected event counts of the background in control and signal regions), the normalized background model is extrapolated to the signal regions. This extrapolation and the

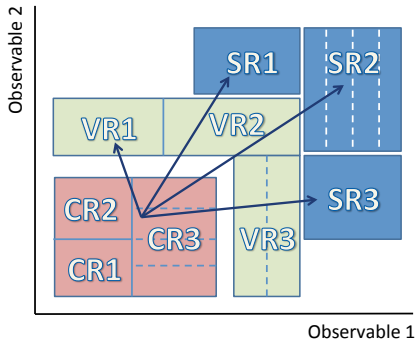

Figure 1: Schematic cartoon of an analysis strategy using one-binned or multi-binned control (CR1-3), validation (VR1-3) and signal regions (SR1-3).

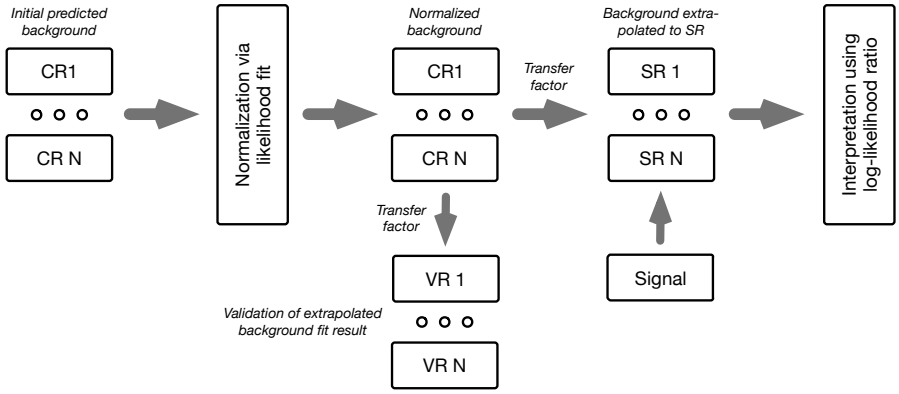

Figure 2: A typical analysis flow with HistFitter with normalizing a background model in control regions (CR1-N) via a simultaneous fit, extrapolating this normalized background model to validation (VR1-N) and signal regions (SR1-N) and performing interpretations concerning a potential signal.

validity of the background estimates found are cross-checked in validation regions (VRs) located in between control and signal regions.

The statistical models describing the data are represented by Probability Density Functions (PDFs), of which the parameters are adjusted in a likelihood fit. Typical parameters include normalization parameters and nuisance parameters to reflect systematic uncertainties. Different PDFs are built for the CRs, VRs and SRs but can be combined in a simultaneous fit as the regions are statistically orthogonal. An important point in the analysis strategy of HistFitter is the sharing of parameters between different regions. This allows the consistent treatment of background and signal models in all regions.

The HistFitter framework, illustrated in Figure 3, is designed based on the analysis strategy described. Starting from a user-defined configuration file and some raw data as input, HistFitter builds binned histograms for the background and signal processes (and also for the observed data) in a first processing step. Using HistFactory, PDFs are constructed from these histograms in a second processing step. In the following steps, the resulting statistical models are analyzed by

performing fits, by calculating limits or p-values in statistical tests and by presenting the results in plots and tables. These steps uses own tools of HistFitter as well as tools of RooFit and RooStats.

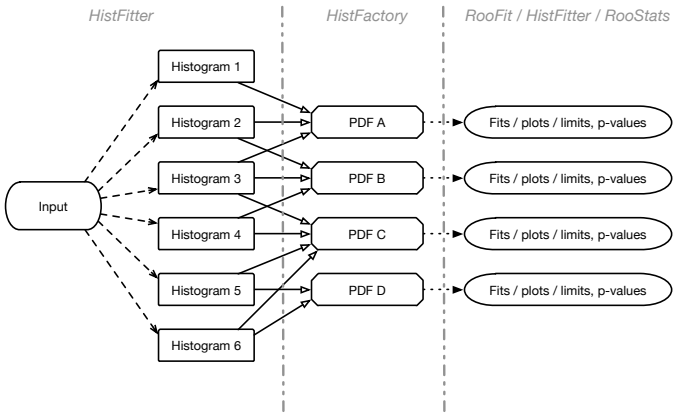

Figure 3: The HistFitter framework starting from the construction of histograms in the first step, building PDFs using HistFactory in the second step and analyzing the resulting statistical models in further steps.

3. Configuration and construction of statistical models in HistFitter

A typical analysis in particle physics searching for new physics quickly needs to deal with about 10 signal regions, has even more control regions and has about 1000 different models of new physics to test. To manage this correctly and efficiently, following the analysis flow described in Section 2, a substantial bookkeeping and configuration machinery is required.

Using a user-specific configuration file HistFitter manages the configuration of an analysis through a configuration manager with two singletons in C++ and Python. The configuration manager manages one or multiple fit configurations (`fitConfig` objects). Each of the fit configurations describes a certain statistical model and contains its PDF along with meta-data providing e.g. information on plotting. If different fit configurations contain shared shared data, this data is reused to save time and memory.

The likelihoods connected to the PDFs have the following form:

$$L(n,\theta^0|\mu_{\rm sig},b,\theta) = P_{\rm SR} \times P_{\rm CR} \times C_{\rm syst}, \tag{3.1}$$

which is the product of Poisson distributions of event counts in signal (or also validation) and control regions ($P_{\rm SR}$ and $P_{\rm CR}$) and of additional constraint terms for systematic uncertainties ($C_{\rm syst}$). The likelihood depends on number of observed events in all regions (n), nuisance parameters parameterizing the impact of systematic uncertainties (θ) with their central values θ^0, signal strength $\mu_{\rm sig}$ and predictions b for various background processes.

A PDF thus has the following building blocks: `channels` (control, validation and signal regions), `samples` (background/signal and data) and `systematics` (systematic uncertainties). HistFactory provides classes for each of these building blocks. HistFitter mirrors and extends these C++ classes in Python.

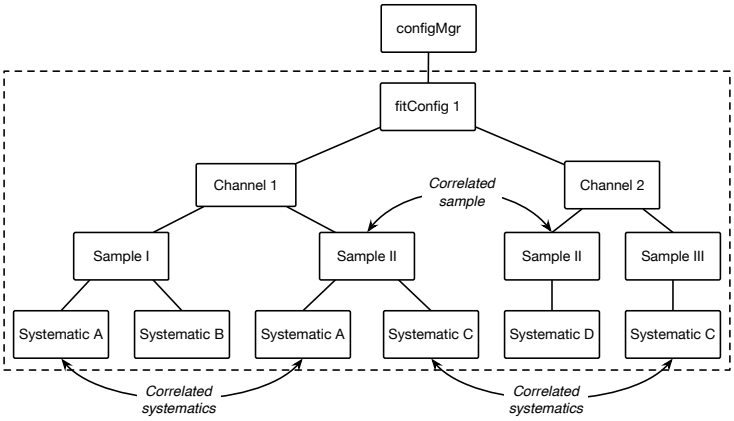

Figure 4: An example configuration with HistFitter showing the `fitConfig` object with depending `channels`, `samples` and `systematics`.

These three building blocks are connected with a `fitConfig` object as shown in Figure 4 and link the PDF to the input data for the model. A `fitConfig` object contains one or multiple one-bin or multi-bin `channels` that are either control, validation or signal regions. The `channels` contain `samples`, corresponding to a component of a PDF decorated with metadata. The `samples` have `systematics` that correspond typically to the $\pm 1\sigma$ variations of a nominal histogram (the nominal histograms are contained in the `samples`). `Samples` and `systematics` may be correlated between channels. Input data is provided to the `fitConfig` object either as ROOT TTree, TH1 or as float number at `sample` and `systematics` level.

HistFitter uses a 'trickle-down-mechanism' which allows the description of a complicated PDF through few lines of code and thus eases the construction of PDFs: this mechanism ensures that `samples` added to `fitConfig` objects are also automatically added to the depending `channels`. Similarly, `systematics` added to the `fitConfig` object (`channels`) are also added to the depending `channels` (`samples`).

4. Fits of the statistical models and presentation of the results

Having constructed the PDFs, different fit strategies can be pursued as illustrated in Table 1. In the *background-only* fit background estimates in the signal or validation regions are evaluated through extrapolation from the control regions. Only the control regions are used in this fit to fix the parameters of the PDF and no signal `sample` is used or allowed.

The aim of the *model-dependent* signal fit is to set limits on a specific signal model or to measure the properties of an excess. Multiple signals regions (which may have multi-bin histograms) can be fitted simultaneously.

In contrast, in the *model-independent* signal fit model independent upper limits are derived on the number of (new physics) events present in one specific one-binned signal region. No assumption about the type of the signal is made which may only be present in the signal region, but not in the control regions.

HistFitter includes various tools to visualize the results of these fits in plots and tables, examples are shown in Figure 5. These include tools for the visualization of fit results before or after the fit (Figure 5 left) and pull plots showing the agreement of the fitted background estimates with the observed data (Figure 5 right).

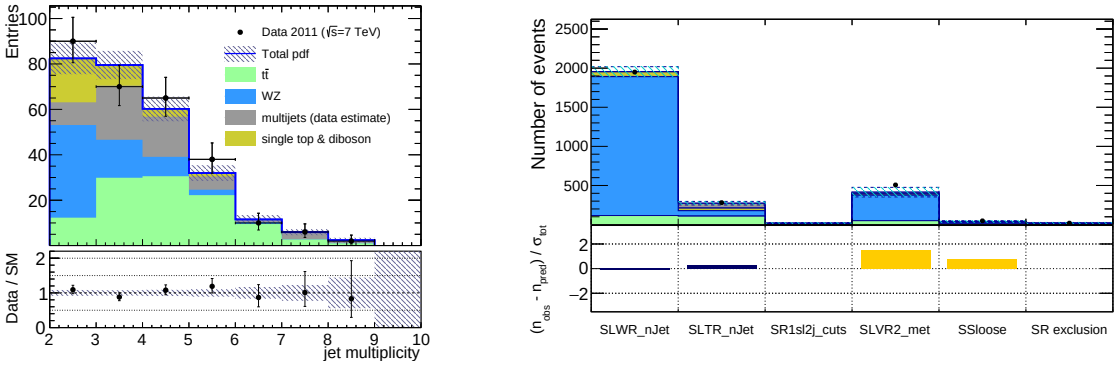

Figure 5: Background estimates shown together with observed data in an after-fit plot (left) and assessing the compatibility of the background estimates with observed data in a pull plot (right).

5. Analysis interpretations

HistFitter interprets the observed data through hypothesis tests comparing a background-only to a background+signal model by calling the appropriate functions in RooStats. Moreover, HistFitter provides tools to display the results of these interpretations in plots and tables.

In interpretations either the model-independent or model-dependent signal fit configurations are used. Using e.g. the model-dependent signal fit configuration, limits on a specific signal model can be derived as e.g. exclusion limits as shown in Figure 6, or as upper limits on the cross section.

Using the model-dependent signal fit configuration the agreement of observed data with the background-only hypothesis can be assessed. This allows the testing of the compatibility of the observed data with the Standard Model (background-only) assumption.

6. Summary

The software framework HistFitter is tailored for statistical analysis and has already been used in numerous publications. The framework can build and test data models of nearly arbitrary complexity. Starting from a user-defined configuration file and by using and *extending* functionalities

Fit setup	*Background-only fit*	*Model-dependent signal fit*	*Model-independent signal fit*
Samples used	backgrounds	backgrounds + signal	backgrounds + dummy signal
Fit regions	CR(s)	CR(s) + SR(s)	CR(s) + SR

Table 1: Summary of the different fit strategies used in HistFitter.

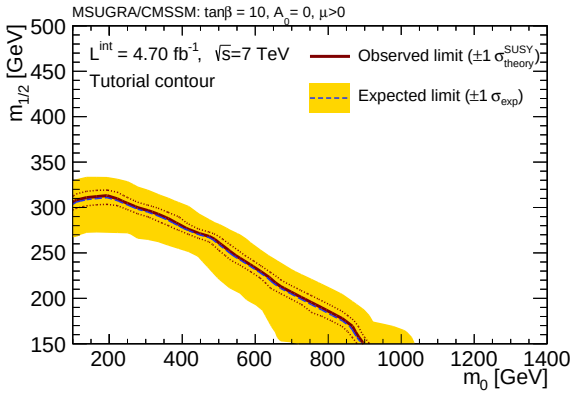

Figure 6: Exclusion limits in a class of supersymmetric models depending on the parameters on the *x*- and *y*-axes. Models below the red observed limit are excluded by hypothesis tests comparing a background-only with background+signal models.

provided by RooFit, HistFactory and RooStats, HistFitter constructs and fits PDFs and performs interpretations of observed data by statistical tests.

Innovative features of HistFitter include the efficient configuration and construction of PDFs using the modular configuration interface with the trickle-down mechanism. The built-in concepts of control, validation and signal regions allow a particularly rigorous treatment of the extrapolation from control to validation/signal regions. HistFitter provides an additional level of abstraction as being designed to provide the bookkeeping to work with multiple signal models at once which is very useful in searches for new physics. Apart from this, HistFitter provides a sizable collection of tools to present analysis end results in a publication-close quality.

References

[1] M. Baak, G.-J. Besjes, D. Côté, A. Koutsman, J. Lorenz and D. Short, *HistFitter software framework for statistical data analysis*, *Eur. Phys. J. C* **75** (2015) 4, [hep-ex/1410.1280], http://histfitter.web.cern.ch/histfitter/.

[2] J. Lorenz, M. Baak, G. J. Besjes, D. Côté, A. Koutsman and D. Short, *HistFitter - A flexible framework for statistical data analysis*, *J. Phys. Conf. Ser.* **608** (2015) 1, 012049.

[3] ATLAS Collaboration, *The ATLAS Experiment at the CERN Large Hadron Collider*, *JINST* **3** (2008) S08003, http://atlas.web.cern.ch/Atlas/Collaboration.

[4] ROOT Collaboration, K. Cranmer, G. Lewis, L. Moneta, A. Shibata and W. Verkerke, *HistFactory: A tool for creating statistical models for use with RooFit and RooStats*, CERN-OPEN-2012-016.

[5] L. Moneta *et al.*, *The RooStats Project*, in proceedings of ACAT 2010, PoS(ACAT2010)057.

[6] W. Verkerke and D. Kirkby, *The RooFit toolkit for data modeling*, *eConf C* **0303241** (2003) MOLT007, [physics/0306116].

[7] R. Brun and F. Rademakers, *ROOT: An object oriented data analysis framework*, *Nucl. Instrum. Meth. A* **389** (1997) 81.

[8] I. Antcheva *et al.*, *ROOT: A C++ framework for petabyte data storage, statistical analysis and visualization*, *Comput. Phys. Commun.* **182** (2011) 1384.