AFit

A Maximum Likelihood Fitting package based on RooFit and TMVA

Adrian Bevan

Queen Mary
University of London

THE ROYAL SOCIETY
CELEBRATING 350 YEARS

SuperB Workshop, LAL, Orsay Feb 2009

SuperB

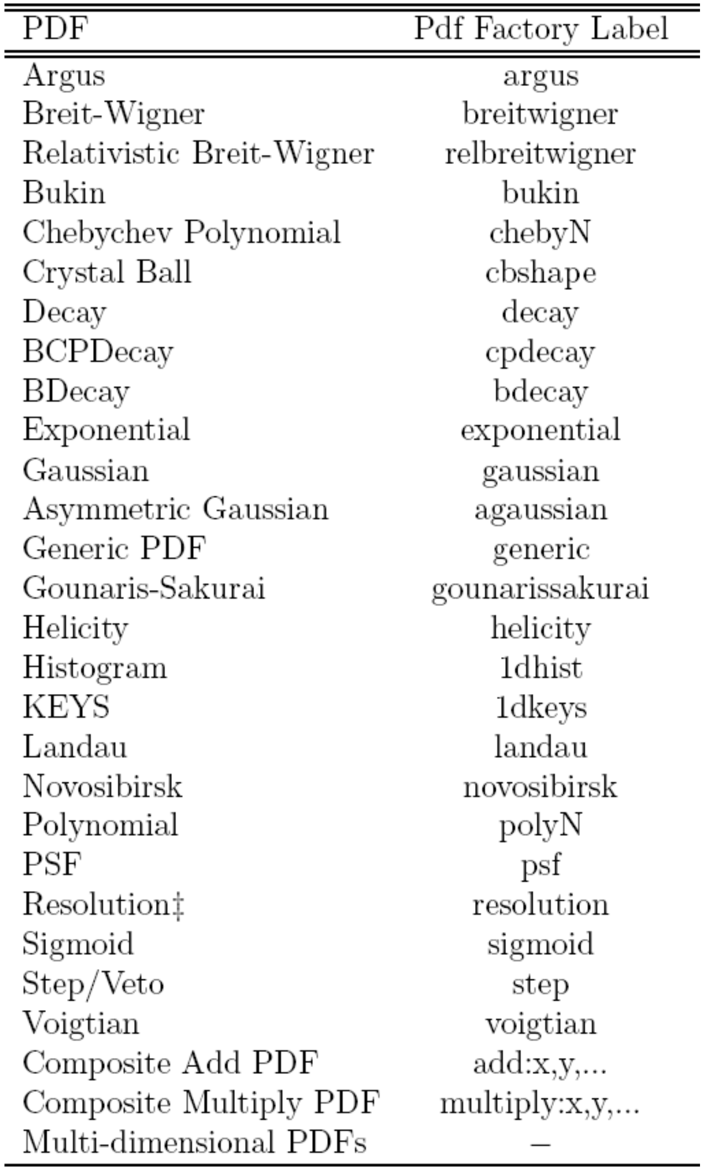
Overview

- Why do we need more than RooFit?
- Available PDFs
- Using AFit PDF builders
- Using AFitMaster to build the model
- Utilities
- Where to find out more information

Why do we need more than RooFit?

- RooFit is a flexible, reliable established fitting package.
- But…
  - Complicated fits take a lot of time to prepare and validate.
  - We lock into a PDF configuration when we write the fit code.
  - Need some impetus to change our analysis…
    - Takes time to modify macros.
    - Can be a real headache!
- The *AFit* extension uses an ASCII configuration file to define the fit model.
  - Quick and easy to modify a fit model.
  - A few more 1D PDFs available than RooFit.
  - Minimal coding required (undergraduates can learn how to use aspects of this tool quickly).
  - Utilities to help validate the fit.

Available PDFs

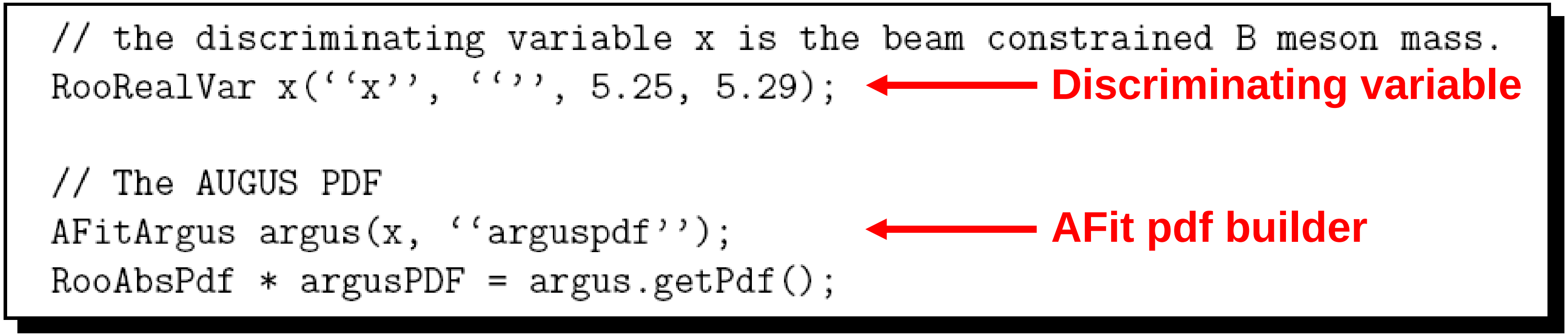


PDF	Pdf Factory Label
Argus	argus
Breit-Wigner	breitwigner
Relativistic Breit-Wigner	relbreitwigner
Bukin	bukin
Chebychev Polynomial	chebyN
Crystal Ball	cbshape
Decay	decay
BCPDecay	cpdecay
BDecay	bdecay
Exponential	exponential
Gaussian	gaussian
Asymmetric Gaussian	agaussian
Generic PDF	generic
Gounaris-Sakurai	gounarissakurai
Helicity	helicity
Histogram	1dhist
KEYS	1dkeys
Landau	landau
Novosibirsk	novosibirsk
Polynomial	polyN
PSF	psf
Resolution‡	resolution
Sigmoid	sigmoid
Step/Veto	step
Voigtian	voigtian
Composite Add PDF	add:x,y,...
Composite Multiply PDF	multiply:x,y,...
Multi-dimensional PDFs	–

- PDF library includes everything you'd expect
  - RooFit PDFs [only 2D missing]
  - Common line-shapes.
  - Sigmoid
  - Veto/step
  - Resolution models
  - Decay models for CP fitting
- Add PDFs together in 1D.
- Multiply PDF by an 'efficiency function' (e.g. helicity distribution).
- Multiply PDFs to make ND.

Using AFit PDF builders

- What is a builder?
  - Each AFit PDF is made using an AFitAbsPdfBuilder derived object.
  - Contains a set of variables: RooAbsReal and RooCategory types that define the shape of the PDF.
    - This is a RooArgSet called `‘varSet’`
  - Makes a RooAbsPdf by calling the `getPdf()` function.



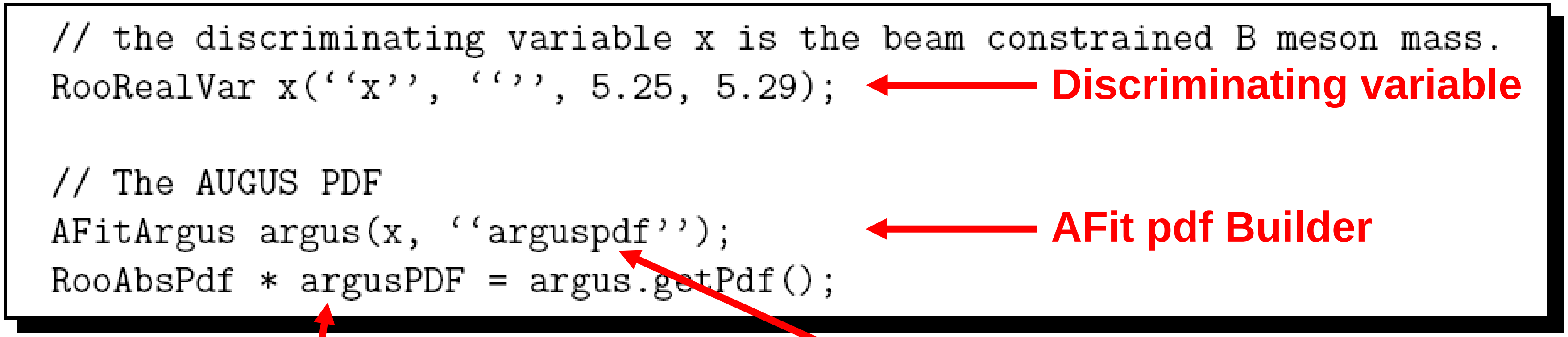
```
// the discriminating variable x is the beam constrained B meson mass.
RooRealVar x(‘‘x’’, ‘‘’’, 5.25, 5.29);

// The AUGUS PDF
AFitArgus argus(x, ‘‘arguspdf’’);
RooAbsPdf * argusPDF = argus.getPdf();
```

Discriminating variable

AFit pdf builder

Using AFit PDF builders

- What is a builder?
  - Each AFit PDF is made using an AFitAbsPdfBuilder derived object.
  - Contains a set of variables: RooAbsReal and RooCategory types that define the shape of the PDF.
    - This is a RooArgSet called `‘varSet’`
  - Makes a RooAbsPdf by calling the `getPdf()` function.



```
// the discriminating variable x is the beam constrained B meson mass.
RooRealVar x(‘‘x’’, ‘‘’’, 5.25, 5.29);

// The AUGUS PDF
AFitArgus argus(x, ‘‘arguspdf’’);
RooAbsPdf * argusPDF = argus.getPdf();
```

Discriminating variable

AFit pdf Builder

Unique name of PDF

PDF to use for fitting

Using AFit PDF builders

- The builder class has
  - Instances of fit parameters (and the `varSet`).
  - Interface to read parameters from a text file.

e.g. The `AFitArgus` instance has data members

```
        xi              =       ξ
        endpoint        =       m0
```

Where the PDF is:

$$\mathcal{P}(m\,;m_0,\xi) = \frac{1}{N}\cdot m\sqrt{1-(m/m_0)^2}\cdot\exp(\xi\,(1-(m/m_0)^2))\cdot\theta(m<m_0)$$

$$\text{where } \theta(m<m_0)=1 \text{ and } \theta(m>m_0)=0.$$

The configuration file to read is set using the

```
        AFitAbsPdfBuilder::setDataCard(const char *)
```

Function. When `getPdf()` is called the variables in `varSet` are read from the specified configuration file.

  - Can be accessed via a PDF Factory …

Using AFit PDF builders

- … the pdf factory: used to get any type of builder.

```
AFitAbsPdfBuilder * AFitPdfFactory::makePdf(TString name, TString type, RooAbsReal & var);
```

  - Make a 1D pdf with the specified name, of the specified type (see list on page 4) and the discriminating variable var.

```
AFitAbsPdfBuilder * AFitPdfFactory::makeConditionalPdf(TString name, TString type,
                                       RooAbsReal & var,  RooAbsReal & conditionalvar);
```

  - As above, but for a PDF with a conditional variable: e.g. $\mathcal{R}(\Delta t, \sigma(\Delta t))$.

```
AFitAbsPdfBuilder * AFitPdfFactory::makePdf(TString name, RooArgList &discVarList);
```

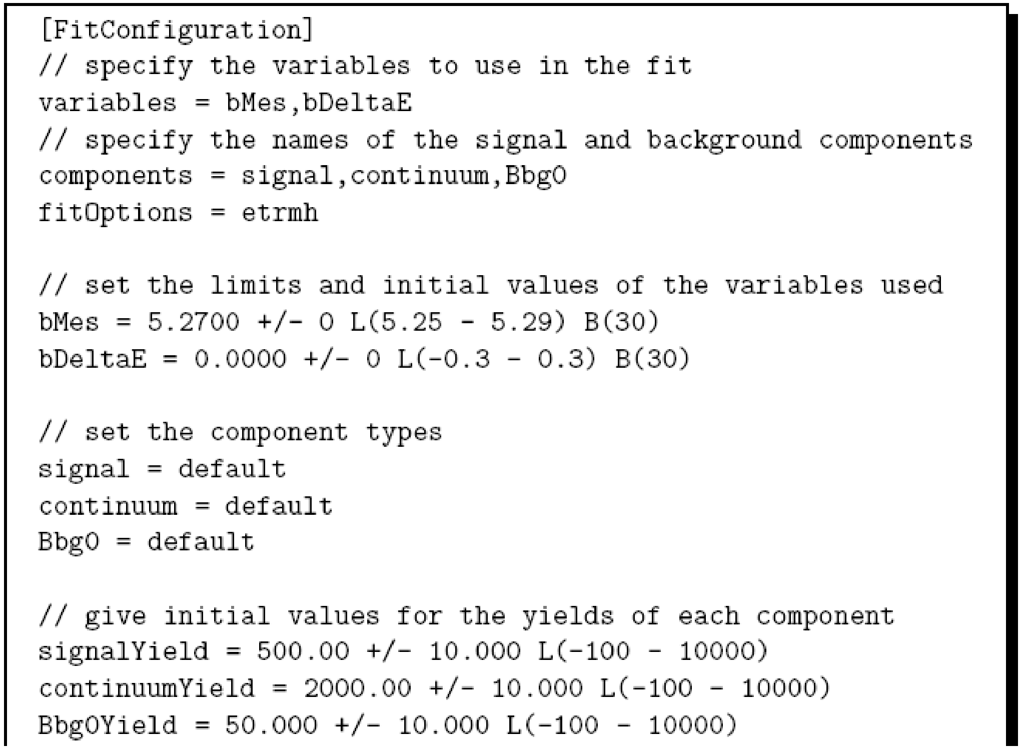
  - Make a multi-dimensional PDF.

Using AFitMaster to build the model

- Build a complicated model with a 2 line ROOT macro:

```
AFitMaster master(‘‘mydatacard.txt’’);
RooAbsPdf * pdf = master.getPdf();
```

- Have to specify fit configuration in text file:



```
[FitConfiguration]
// specify the variables to use in the fit
variables = bMes,bDeltaE
// specify the names of the signal and background components
components = signal,continuum,Bbg0
fitOptions = etrmh

// set the limits and initial values of the variables used
bMes = 5.2700 +/- 0 L(5.25 - 5.29) B(30)
bDeltaE = 0.0000 +/- 0 L(-0.3 - 0.3) B(30)

// set the component types
signal = default
continuum = default
Bbg0 = default

// give initial values for the yields of each component
signalYield = 500.00 +/- 10.000 L(-100 - 10000)
continuumYield = 2000.00 +/- 10.000 L(-100 - 10000)
Bbg0Yield = 50.000 +/- 10.000 L(-100 - 10000)
```

You define the:

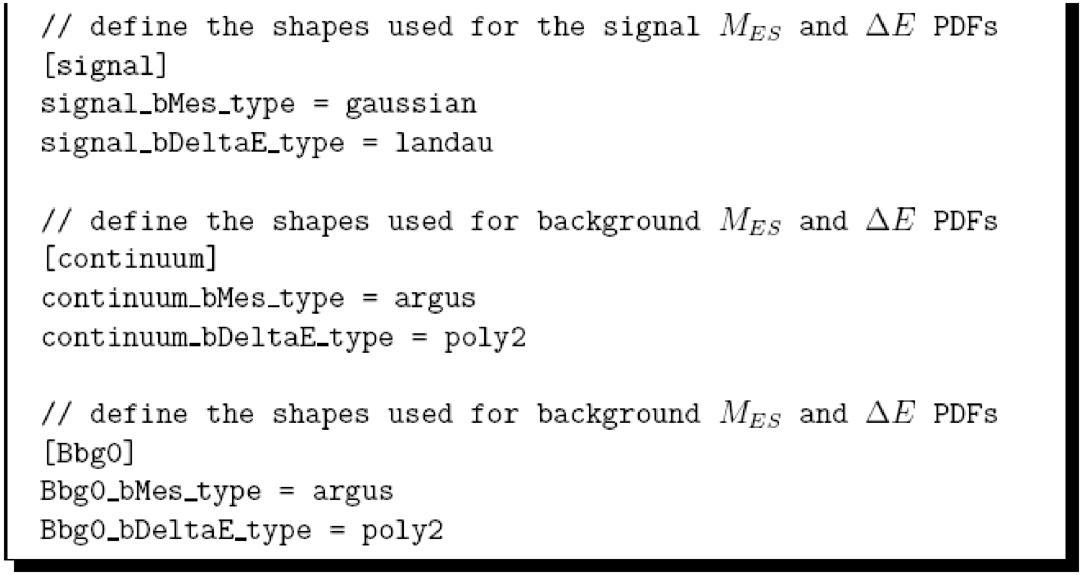
variables

fit components

component types

fit yields (assumes you want to do an extended-unbinned ML fit.

Using AFitMaster to build the model



```
// define the shapes used for the signal $M_{ES}$ and $\Delta E$ PDFs
[signal]
signal_bMes_type = gaussian
signal_bDeltaE_type = landau

// define the shapes used for background $M_{ES}$ and $\Delta E$ PDFs
[continuum]
continuum_bMes_type = argus
continuum_bDeltaE_type = poly2

// define the shapes used for background $M_{ES}$ and $\Delta E$ PDFs
[Bbg0]
Bbg0_bMes_type = argus
Bbg0_bDeltaE_type = poly2
```

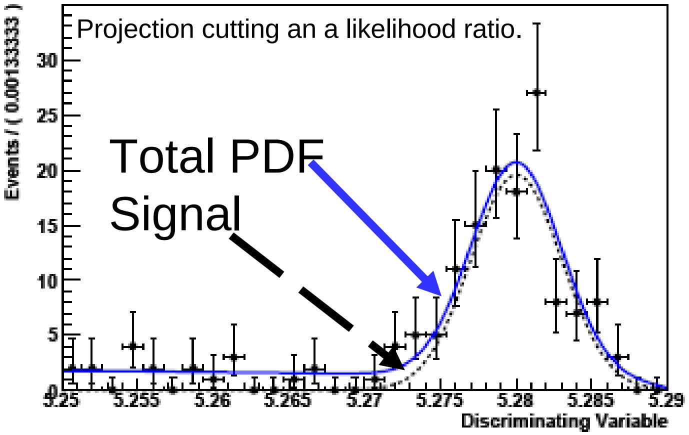
You define the:

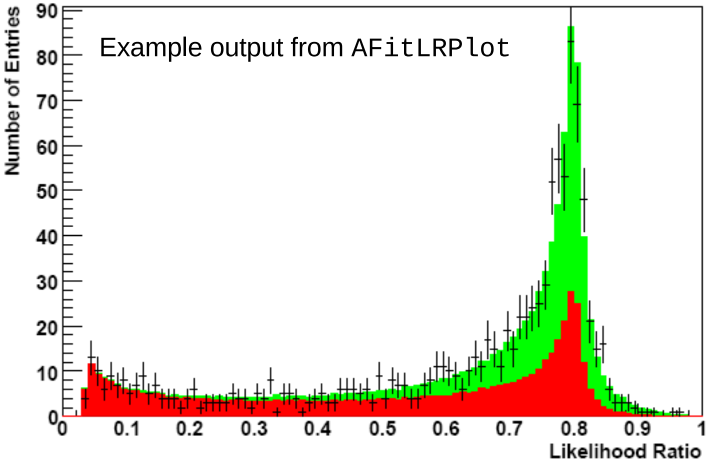
pdf types for each component

- The rest of the configuration file is used to specify the shape parameters.
  - By default all parameters are allowed to float, and have a dummy range.
  - Interface to fit to MC/data control samples (see user guide for details).
  - Can also build a RooSimultaneous using `getSimPdf()`.

Utilities

- When setting up your analysis you will think about
  - Checking for correlations between fit variables: `AFitStatTools`
  - Defining an MVA: `AFitTMVAInterface`
  - Running toys: `AFitToy`
  - Plotting: `AFitProjectionPlot`
    - Projections, (not)cutting on data, on likelihood ratio: S/(S+B)
  - Likelihood ratio plot to test global agreement between fit and MC: `AFitLRPlot`





Where to find out more information

- *AFit* is available from:

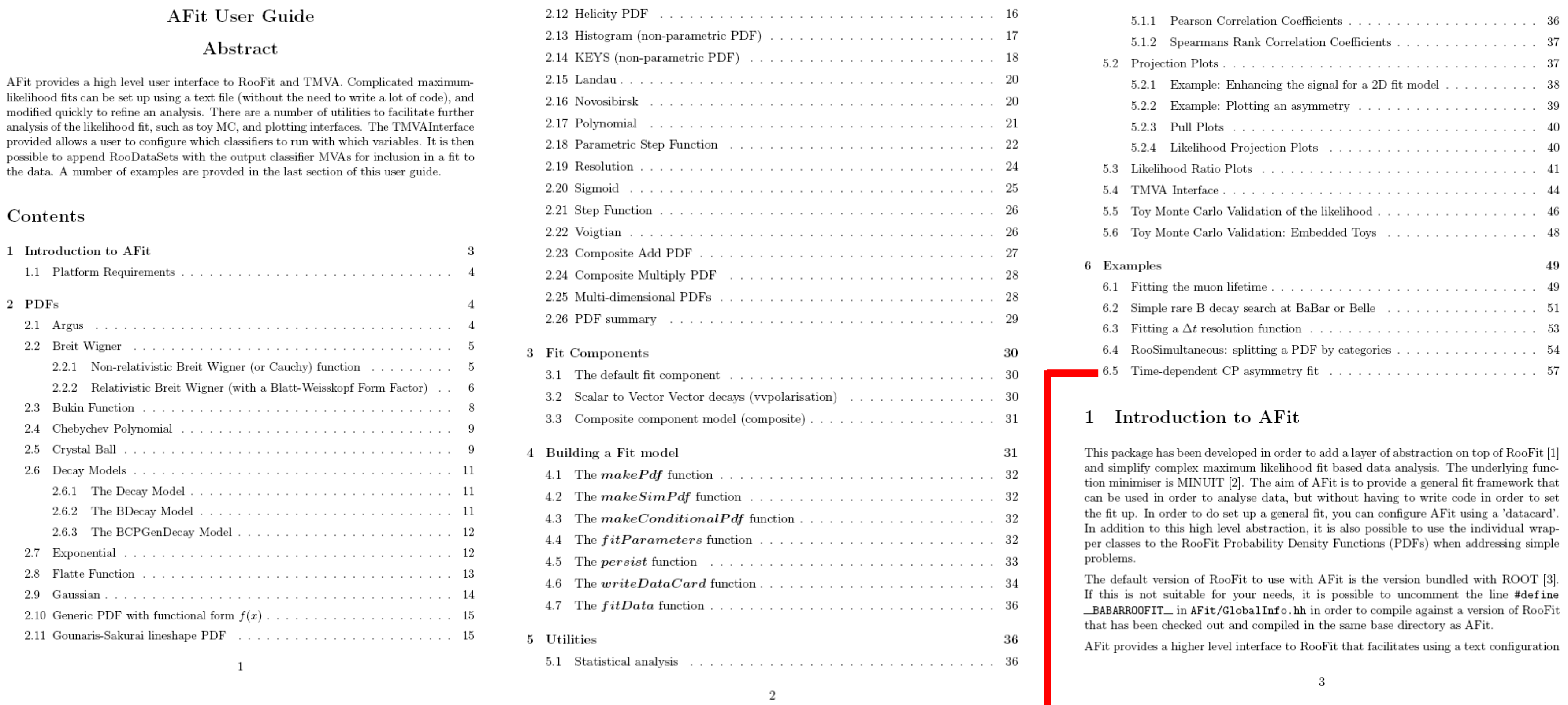
  http://pprc.qmul.ac.uk/~bevan/afit/

- Available for down load:
  - User guide
  - Source code
  - Examples available (sub-directory of source code):
    - Exponential decay fit for lifetime
    - M_{ES}-ΔE fit configuration for signal + continuum + B background
    - Simultaneous PDF
    - How to set up a simple TDCPV Δt fit.

- Web page also has quick start compile instructions.

- Requirements: ROOT v5.20 (compiled with RooFit).

Documentation available:



AFit User Guide

Abstract

AFit provides a high level user interface to RooFit and TMVA. Complicated maximum-likelihood fits can be set up using a text file (without the need to write a lot of code), and modified quickly to refine an analysis. There are a number of utilities to facilitate further analysis of the likelihood fit, such as toy MC, and plotting interfaces. The TMVAInterface provided allows a user to configure which classifiers to run with which variables. It is then possible to append RooDataSets with the output classifier MVAs for inclusion in a fit to the data. A number of examples are provded in the last section of this user guide.

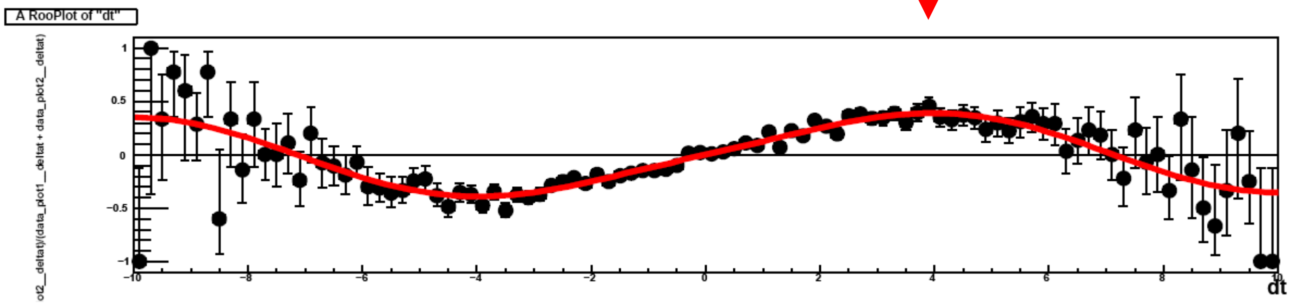
Contents

1 Introduction to AFit

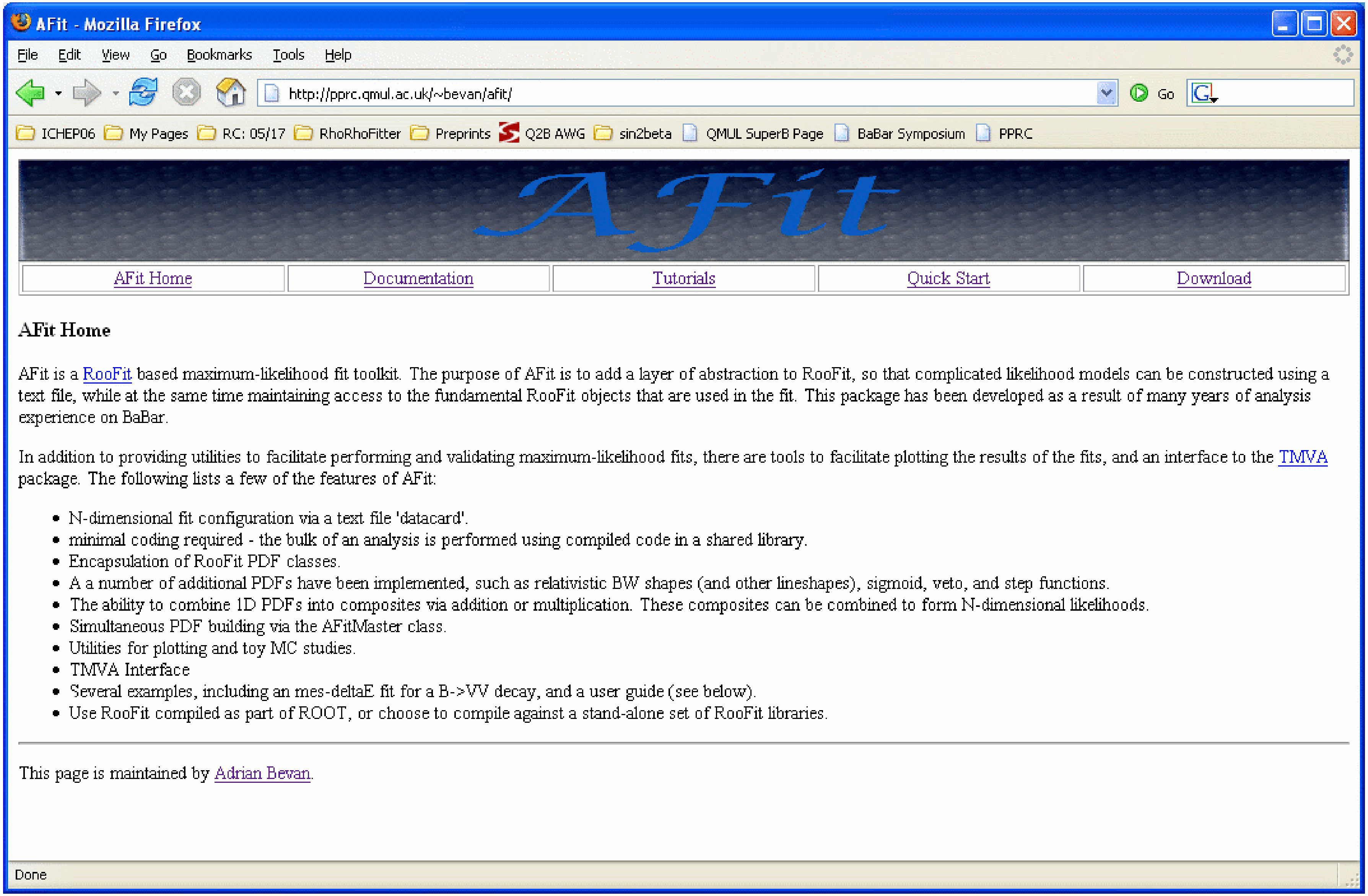
This package has been developed in order to add a layer of abstraction on top of RooFit [1] and simplify complex maximum likelihood fit based data analysis. The underlying function minimiser is MINUIT [2]. The aim of AFit is to provide a general fit framework that can be used in order to analyse data, but without having to write code in order to set the fit up. In order to do set up a general fit, you can configure AFit using a 'datacard'. In addition to this high level abstraction, it is also possible to use the individual wrapper classes to the RooFit Probability Density Functions (PDFs) when addressing simple problems.

The default version of RooFit to use with AFit is the version bundled with ROOT [3]. If this is not suitable for your needs, it is possible to uncomment the line `#define __BABARROOFIT__` in `AFit/GlobalInfo.hh` in order to compile against a version of RooFit that has been checked out and compiled in the same base directory as AFit.

AFit provides a higher level interface to RooFit that facilitates using a text configuration



Home Page



AFit - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://pprc.qmul.ac.uk/~bevan/afit/
Go
ICHEP06 My Pages RC: 05/17 RhoRhoFitter Preprints Q2B AWG sin2beta QMUL SuperB Page BaBar Symposium PPRC
AFit
AFit Home | Documentation | Tutorials | Quick Start | Download
Done

AFit Home

AFit is a RooFit based maximum-likelihood fit toolkit. The purpose of AFit is to add a layer of abstraction to RooFit, so that complicated likelihood models can be constructed using a text file, while at the same time maintaining access to the fundamental RooFit objects that are used in the fit. This package has been developed as a result of many years of analysis experience on BaBar.

In addition to providing utilities to facilitate performing and validating maximum-likelihood fits, there are tools to facilitate plotting the results of the fits, and an interface to the TMVA package. The following lists a few of the features of AFit:

- N-dimensional fit configuration via a text file 'datacard'.
- minimal coding required - the bulk of an analysis is performed using compiled code in a shared library.
- Encapsulation of RooFit PDF classes.
- A a number of additional PDFs have been implemented, such as relativistic BW shapes (and other lineshapes), sigmoid, veto, and step functions.
- The ability to combine 1D PDFs into composites via addition or multiplication. These composites can be combined to form N-dimensional likelihoods.
- Simultaneous PDF building via the AFitMaster class.
- Utilities for plotting and toy MC studies.
- TMVA Interface
- Several examples, including an mes-deltaE fit for a B->VV decay, and a user guide (see below).
- Use RooFit compiled as part of ROOT, or choose to compile against a stand-alone set of RooFit libraries.

This page is maintained by Adrian Bevan.