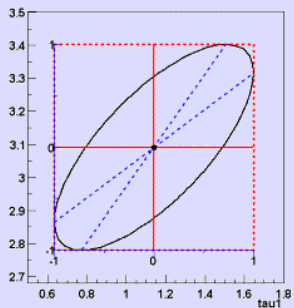
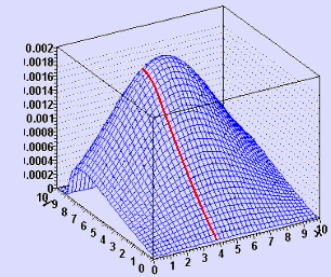
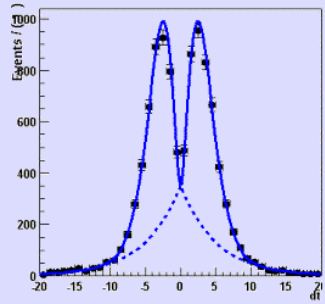


RooFit

A general purpose tool kit for data modeling

Wouter Verkerke (UC Santa Barbara)
David Kirkby (UC Irvine)





RooFit purpose - Data Modeling & Analysis

Distribution of observables \vec{x}

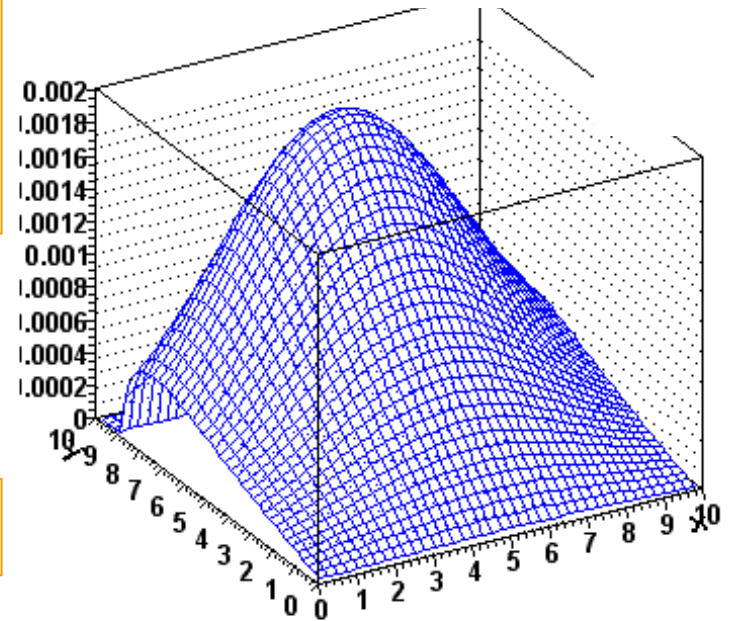
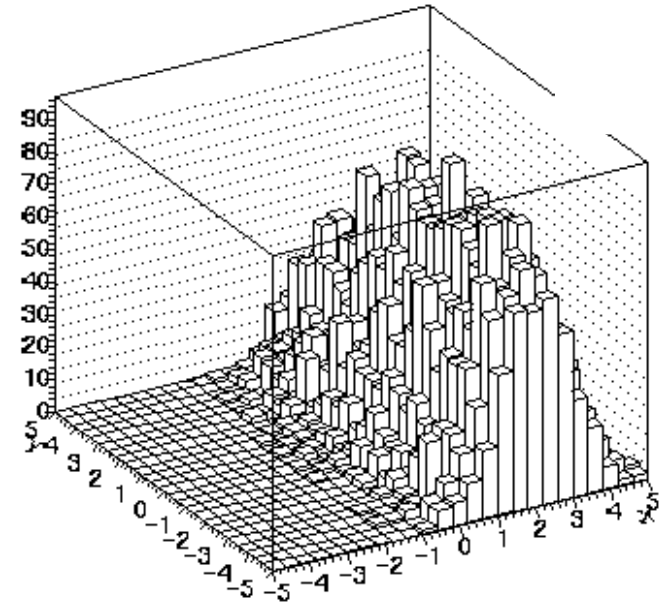
Define data model

Probability density function $F(\vec{x}; \vec{p}, \vec{q})$

- Physical parameters of interest \vec{p}
- Other parameters \vec{q} to describe detector effect (resolution, efficiency,...)
- Normalized over allowed range of the observables \vec{x} w.r.t the parameters \vec{p} and \vec{q}

Fit model to data

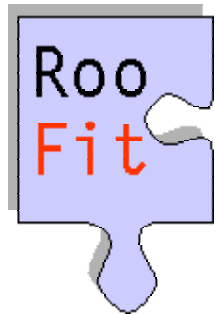
Determination of \vec{p}, \vec{q}





Implementation

- Add-on package to ROOT
 - C++ command line interface & macros
 - Data management and histogramming
 - Graphics interface
 - I/O support



- RooFit is a collection of classes that augment the ROOT environment
 - *Focus on data modeling*
 - Implemented as 2 shared libraries



Data modeling – Probability Density Functions

- RooFit models are Probability Density Functions (PDF)

observables: variables in the data

$$F(x, p)$$

parameters: all other variables

- Properties of PDFs:

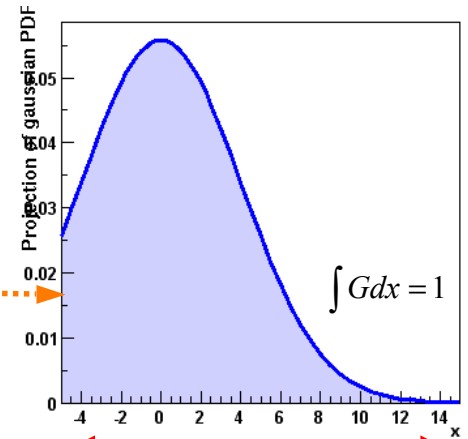
$$\int_{x_{\min}}^{x_{\max}} F(x, p) dx \equiv 1$$

Normalized to unity over all observables

$$F(x, p) \geq 0$$

Positive definite

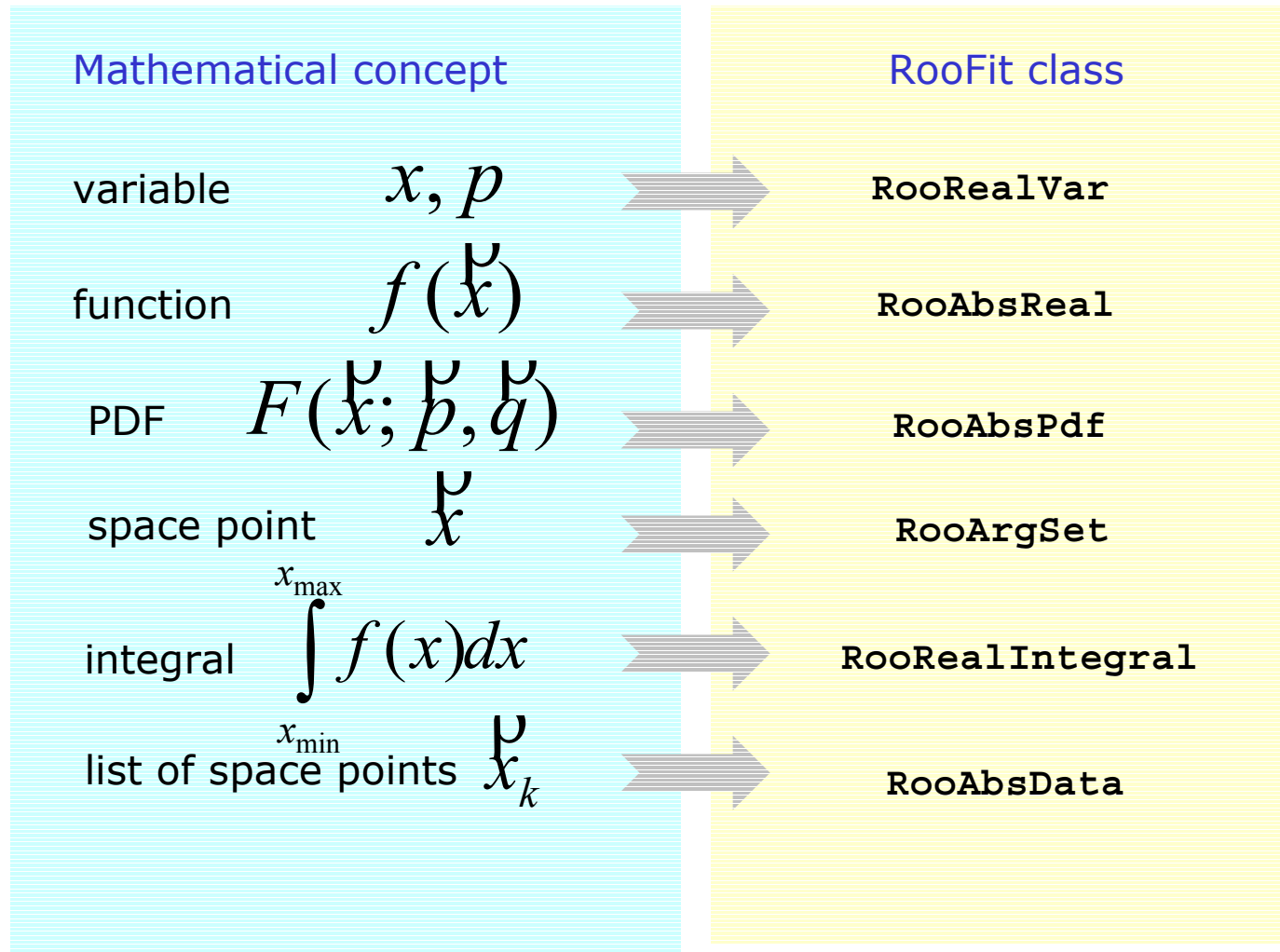
This essential property is traditionally the most difficult part of any implementation.





RooFit key features – OO Data/Model language

- Mathematical objects are represented as C++ objects





RooFit key features – OO Data/Model language

- Straightforward correlation between mathematical representation of formula and RooFit code

Math	$G(x, m, \sqrt{s})$
RooFit diagram	<pre>graph TD; x[RooRealVar x] <--> g[RooGaussian g]; m[RooRealVar m] <--> g; sqrts[RooFormulaVar sqrts] <--> g; sqrts <--> s[RooRealVar s];</pre>
RooFit code	<pre>RooRealVar x("x","x",-10,10) ; RooRealVar m("m","mean",0) ; RooRealVar s("s","sigma",2,0,10) ; RooFormulaVar sqrts("sqrts","sqrt(s)",s) ; RooGaussian g("g","gauss",x,m,sqrts) ;</pre>



RooFit key features – OO Data/Model language

- All objects are *self documenting*

Name – Unique identifier of object

Associated data
Limits, units, plot labels,... stored in object

Title – More elaborate description of object

Initial range

```
RooRealVar mass ("mass", "Invariant mass", 5.20, 5.30) ;  
RooRealVar width ("width", "B0 mass width", 0.00027, "GeV") ;  
RooRealVar mb0 ("mb0", "B0 mass", 5.2794, "GeV") ;
```

Objects representing a variable

Initial value Optional unit

Function object

```
RooGaussian b0sig ("b0sig", "B0 sig PDF", mass, mb0, width) ;
```

References to variables



RooFit key features – Integration & PDF Normalization

- Integrals are also represented as objects

- Easy to create:

```
// Create object representing  $\int(g) dx$   
RooRealIntegral* gi = g.createIntegral(x) ;
```

gi changes as parameters of g, or limits on x change.

- *Every function* can be integrated

- By default **numeric techniques** are used:
Adaptive trapezoid for 1-D, Monte Carlo (VEGAS) for >1-D
- Functions and PDFs that have known (partial) **analytical integrals** can advertise these (but it is not required)
- Final integral can be a **combination** of analytic and numeric methods.
Optimal combination automatically selected
- Integral values are automatically **cached** and recalculated only when necessary

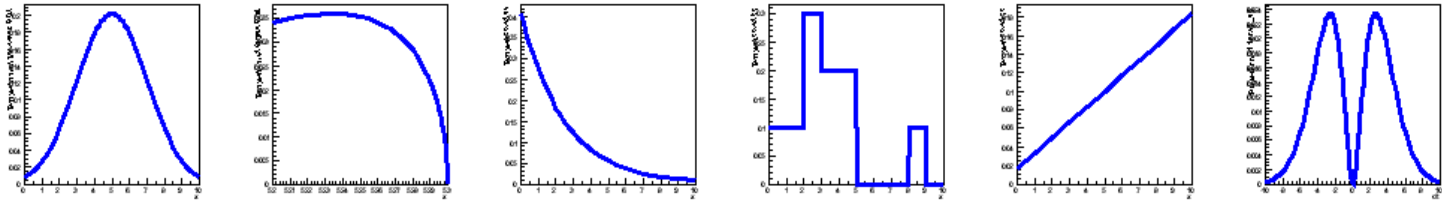
- Integral objects take care of **PDF normalization**

- Raw PDF value always **divided** by value of **integral object**
- Proper normalization guaranteed while support in PDF optional



RooFit key features - Modularity

- RooFit provides a **collection of standard PDF classes**, e.g.
 - Gaussian, Exponential, Polynomial (*basic*)
 - Argus, Crystal Ball, *B*-Decay, Breit-Wigner, Voigtian (*physics inspired*)
 - Histogram, KEYS (*non-parametric*)



- Generic **expression based PDF** also available:
 - Based on **Tformula**
 - Expression normalized via numeric integration

```
RooGenericPdf gp("gp", "Generic PDF", "exp(x*y+a) - b*x",  
                RooArgSet(x, y, a, b));
```

- **Easy** for users to **write/contribute new PDF/function classes**



Writing new PDF & function classes

- Base classes `RooAbsPdf`, `RooAbsReal`, ... hides all complex issues

```
class RooGaussian : public RooAbsPdf {
public:
    RooGaussian(const char *name, const char *title,
                RooAbsReal& x, RooAbsReal& mean, RooAbsReal& sigma);
    RooGaussian(const RooGaussian& other, const char* name=0) ;
    virtual TObject* clone(const char* newname) const
    inline virtual ~RooGaussian() { }

protected:
    Double_t evaluate() const ;

    RooRealProxy _x ;
    RooRealProxy _mean ;
    RooRealProxy _sigma ;

private:
    ClassDef(RooGaussian,0) Gaussian PDF
};
```

Minimal implementation:
ctor, ctor, dtor, clone() and evaluate()

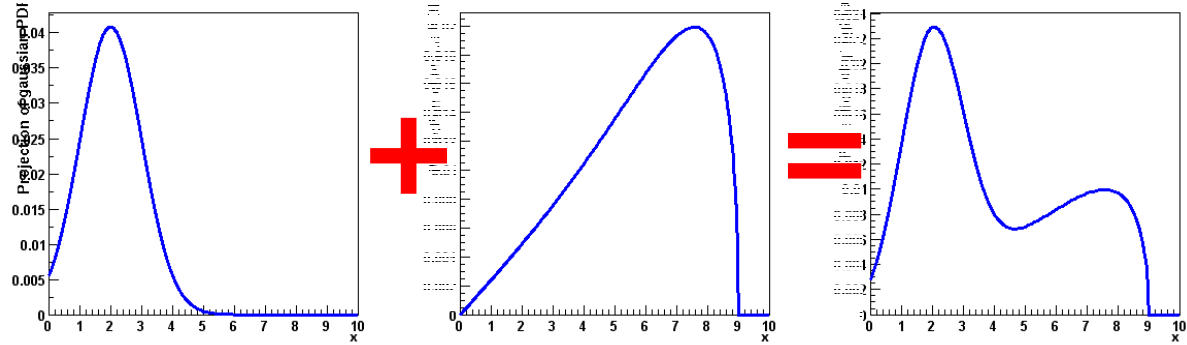
Proxy classes store references to value servers.
-All client/server link management taken care of
-**Easy to use**: behaves like `Double_t`



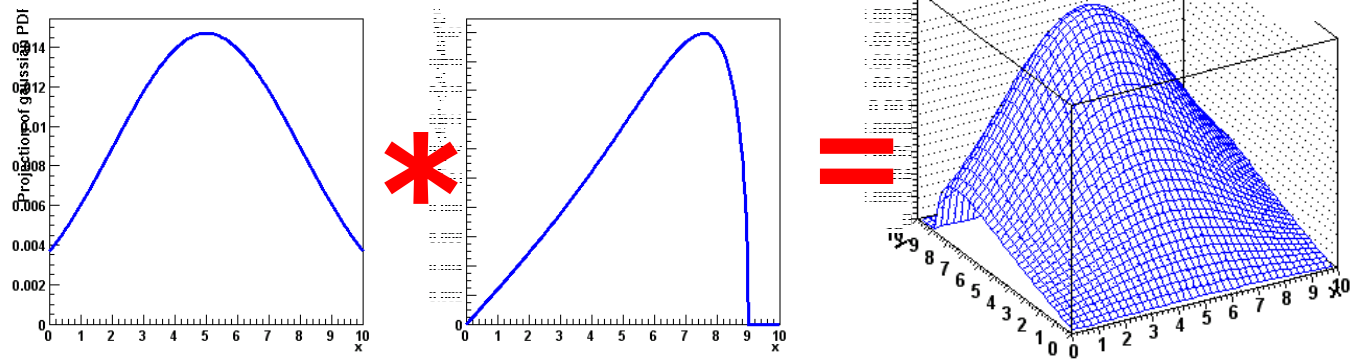
RooFit key features - Modularity

- Complex PDFs can be trivially composed from basic PDFs

- Addition



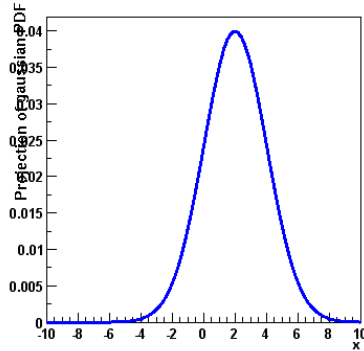
- Multiplication





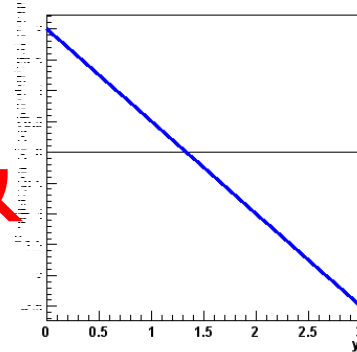
RooFit key features - modularity

- Composition ('plug & play')

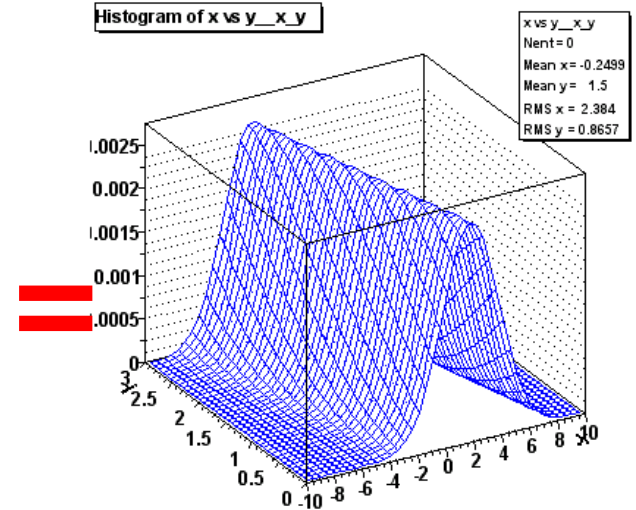


$$g(x; m, s)$$

&



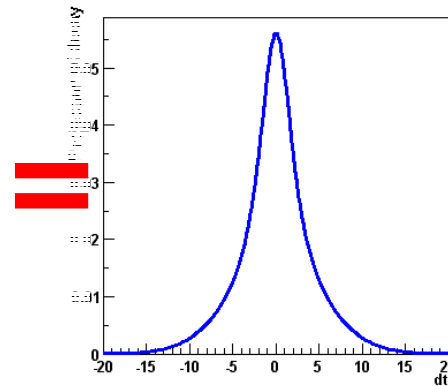
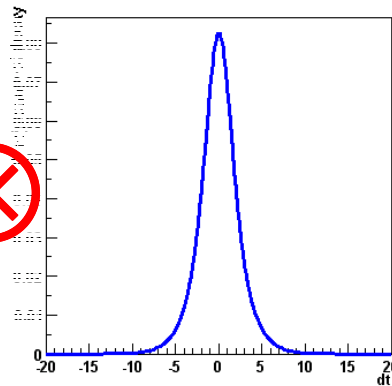
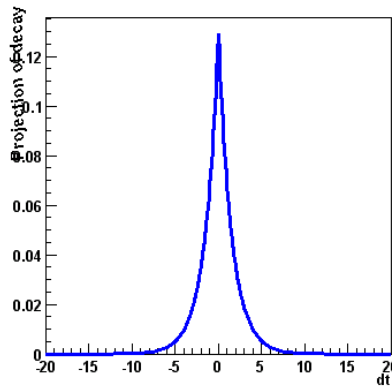
$$m(y; a_0, a_1)$$



$$g(x, y; a_0, a_1, s)$$

Possible in *any* PDF
No explicit support in PDF code needed

- Convolution





Datasets & Data types

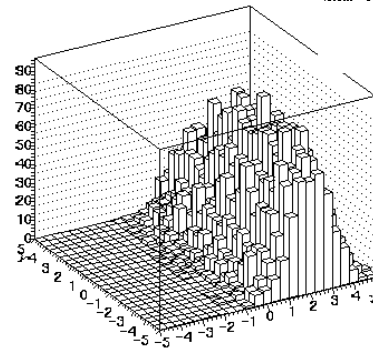
Unbinned

x	y	z
1	3	5
2	4	6
1	3	5
2	4	6

Weighted unbinned

x	y	z	wgt
1	3	5	0.11
2	4	6	0.75
1	3	5	0.30
2	4	6	0.92

Binned



RoodataSet

RoodataHist

Roodata

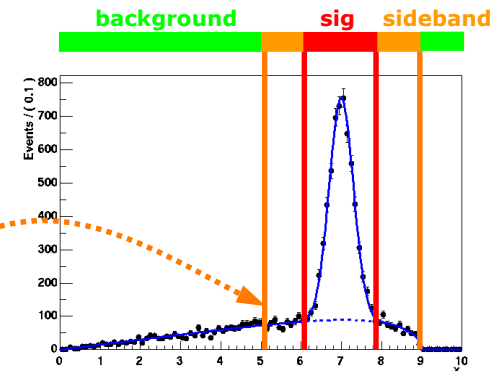
Many RooFit methods accept this abstract data type → Use data set types interchangeably

- Implemented with **Trees**
- Current row represented by set of RooFit value objects
- Unlimited number of dimensions (also binned)
- Dimensions may be mix of real, discrete, or string values
- Import data from existing ROOT types:
 - **TTree**, ASCII file → **RoodataSet**
 - **THx**, **RoodataSet** → **RoodataHist**



Discrete variables

- Many physics problems involve discrete variables
 - Decay mode, particle ID, flavor, charge, reconstruction technique
 - Traditionally represented by integers, but name→index mapping can be cumbersome
- RooFit has discrete variable base type with named states
 - Named states facilitate symbolic manipulation of data
 - Index→name mapping optional
- Discrete functions implement cuts
 - Real-to-discrete, Discrete-to-discrete
 - Multiplication, Mapping, Thresholds, ...
- Distribution of discrete variables in data can be tabulated

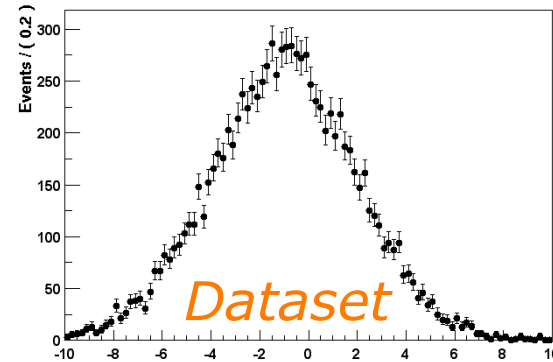
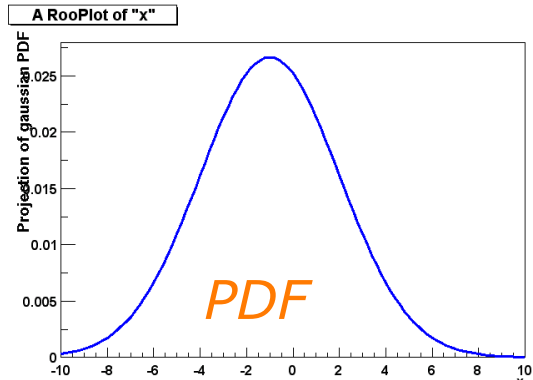


```
data->table(tagCat, "x>8.23")
```

Lepton	668
Kaon	717
NetTagger-1	632
NetTagger-2	616

RooFit key features – Fitting

Given **any** PDF, fitting is a 1-line operations



`fitResult`
stores
snapshot of
parameters,
correlation
matrix etc...

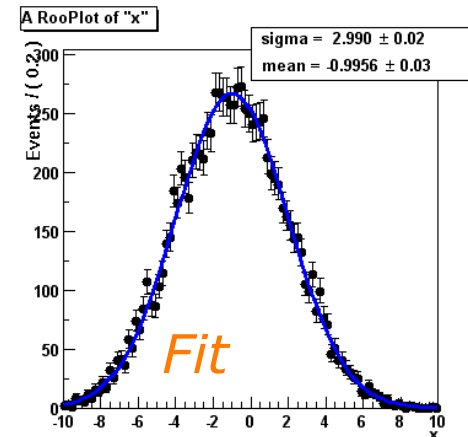
```
fitResult = gauss.fitTo(data)
```

Fit result reflected in
parameters of gauss

```
COVARIANCE MATRIX CALCULATED SUCCESSFULLY
FCN=25054.9 FROM HESSE      STATUS=OK          10 CALLS          69 TOTAL
                        EDM=3.65627e-06    STRATEGY= 1      ERROR MATRIX ACCURATE

EXT PARAMETER
NO.   NAME      VALUE          ERROR          INTERNAL          INTERNAL
1     mean     -9.95558e-01   3.01321e-02    6.59595e-04     -9.95558e-01
2     sigma     2.99001e+00   2.20203e-02    9.66748e-05     2.99001e+00

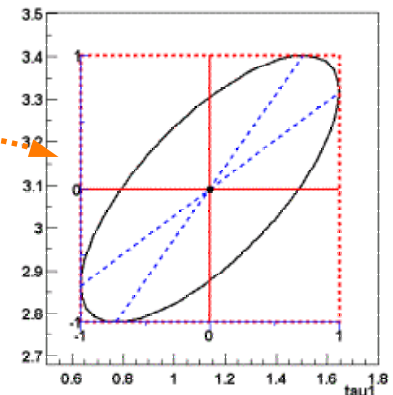
ERR DEF= 0.5
EXTERNAL ERROR MATRIX.  NDIM= 25  NPAR= 2  ERR DEF=0.5
 9.079e-04 -1.787e-05
-1.787e-05 4.849e-04
```





RooFit key features – Fitting

- RooFit provides out-of-the box support for
 - **Unbinned maximum likelihood fit** [PDF & unbinned dataset]
 - Unbinned **weighted** maximum likelihood fit [PDF & weighted unbinned dataset]
 - **Binned** maximum likelihood fit [PDF & binned dataset]
 - **Extended** maximum likelihood fit (binned, unbinned, unbinned weighted)
 - Binned **chi-square** fit [PDF & binned dataset]
 - Support for other user-defined goodness-of-fit quantities
- Easy to add additional terms to fit function
 - Penalty terms, Lagrange multiplier etc.
- Seamless integration of MINUIT with RooFit objects
 - **Interactive MINUIT** sessions possible from **ROOT command line (in C++)**
 - Full fit output, including correlation matrix, savable for easy access later
 - Graphical analysis of correlation matrix elements





RooFit key features – Fitting

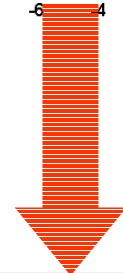
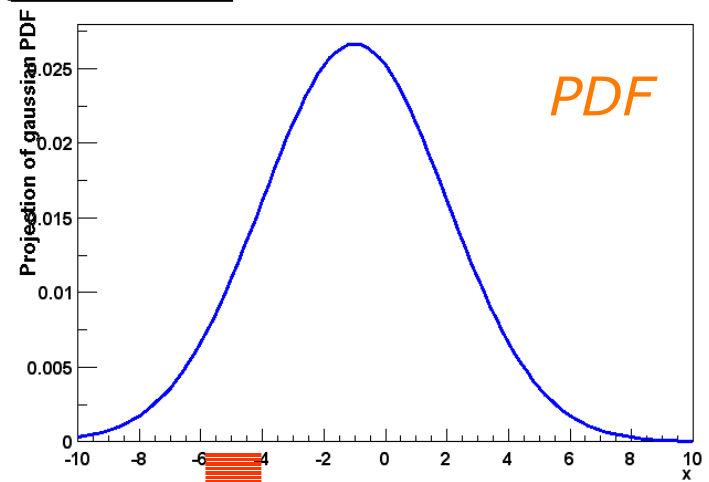
- Automatic pre-fit PDF **optimization**
 - Prior to each fit, the PDF is analyzed for possible optimizations
 - Optimization algorithms:
 - Detection and *precalculation of constant terms* in any PDF expression
 - Function *caching* and lazy evaluation
 - *Factorization* of multi-dimensional problems where ever possible
 - Optimizations are always tailored to the specific use in each fit.
- **No need for users to hard-code optimizations**
 - *Keeps your code understandable*, maintainable and flexible without sacrificing performance
 - Optimization concepts implemented by RooFit are applied consistently and completely to all PDFs
 - Speedup of factor 3-10 reported in realistic complex fits
- Fit **parallelization** on multi-CPU hosts
 - Option for *automatic parallelization* of fit function *on multi-CPU hosts* (no explicit or implicit support from user PDFs needed)



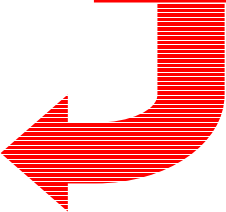
RooFit key features – Event Generation

Given **any** PDF, Toy Monte Carlo generation Is a 1-line operations

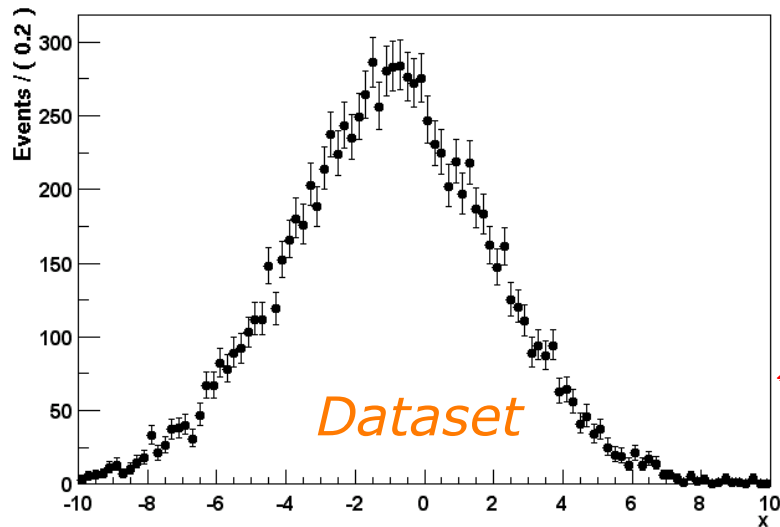
A RooPlot of "x"



```
data = gauss.generate(x, 1000)
```



A RooPlot of "x"

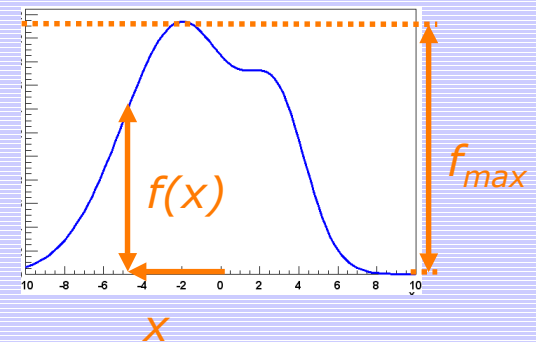


RooFit key features – Event generation

- **Generate** “toy Monte Carlo” samples from *any* PDF
 - **Accept/reject sampling** method used by **default**
 - PDF can advertise **internal generator** if more efficient methods exists (e.g. Gaussian)
 - Each generator request will use the **most efficient** accept-reject / internal generator **combination** available
 - Operator PDFs (sum,product,...) distribute generation over components whenever possible
- Subset of variables can be taken from a **prototype dataset**
 - E.g. to more accurately model the statistical fluctuations in a particular sample.
 - **Correlations** with prototype observables **correctly taken into account**
 - **Easy**: just supply prototype dataset

```
data = gauss.generate(x, protoData)
```

Accept/reject sampling



- 0) Estimate maximum PDF value f_{\max} by repeated sampling at random x
 - 1) Throw a uniform random value for x
 - 2) Throw uniform random number (ran) between 0 and f_{\max} .
 - 3) If $\text{ran} * f_{\max} < f(x)$, **accept** x as generated event
- Repeat 1)-3) as necessary



RooFit key features – Plotting support

- Plot container class holds all elements of 1D plot together
 - Histograms of datasets, PDF projections, text boxes, arrow,...

```
RooPlot* frame = dt.frame() ;
```

Created from variable object.
Remembers variable on x-axis

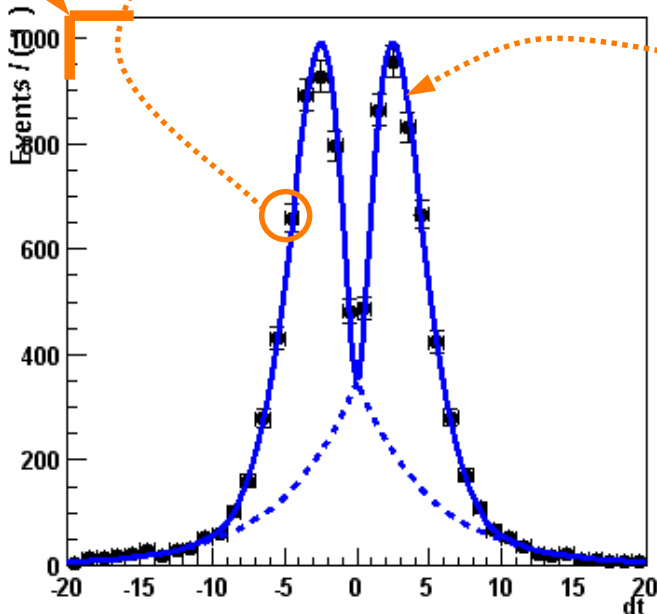
```
data->plotOn(frame) ;
```

Curve always **normalized** to last plotted dataset in **frame**

```
pdf->plotOn(frame) ;
```

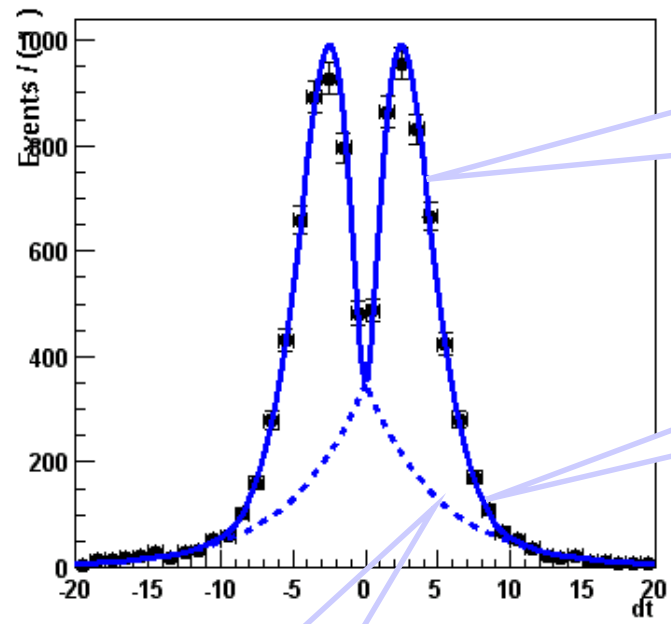
For multi-dimensional PDFs:
appropriate 1-dimensional projection is
automatically created:

$$\text{Projection}[F](x) = N \cdot \frac{\int F(x, y) dy}{\int F(x, y) dx dy}$$





RooFit key features – Plotting support



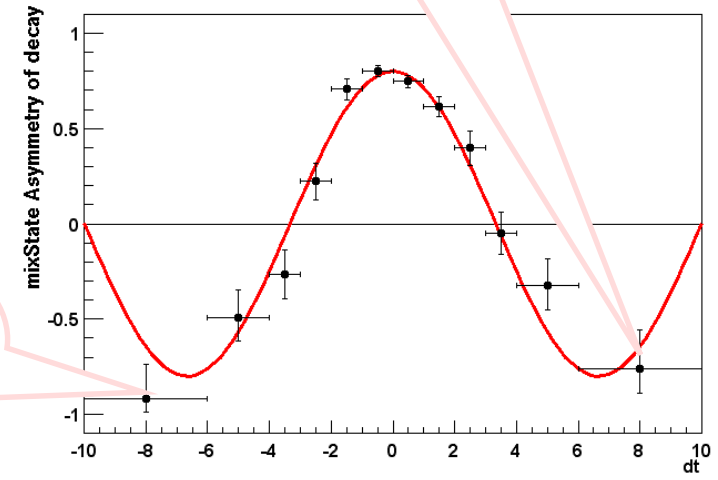
Adaptive spacing of curve points to achieve 1% precision

Poisson errors on histogram

Easy plotting of summed PDF components

Variable bin sizes OK (contents will be adjusted, if necessary)

Binomial errors on asymmetry histograms





RooFit key features – Plotting support

- Additional methods available to
 - Plot/project **slices** or arbitrarily shaped **regions** of PDFs
 - Plot PDF projections **averaged over observables** provided in a **dataset**
 - Plot generic **asymmetries** $(A-B)/(A+B)$
- Single method for all plot varieties: `plotOn()`
 - *Named argument* interface powerful yet easy to use

```
pdf->plotOn(frame) ;  
pdf->plotOn(frame, Slice(x)) ;  
pdf->plotOn(frame, Asymmetry(tag), LineColor(kRed)) ;  
pdf->plotOn(frame, Components("Bkg*"), ProjWData(dterr)) ;  
pdf->plotOn(frame, Normalization(0.5), DrawOption("F")) ;
```



Complete example: generating and fitting a decay PDF

```
// Decay PDF variable
RooRealVar dt("dt","dt",-20,20) ;
RooRealVar tau("tau","tau",1.548,-10,10) ;

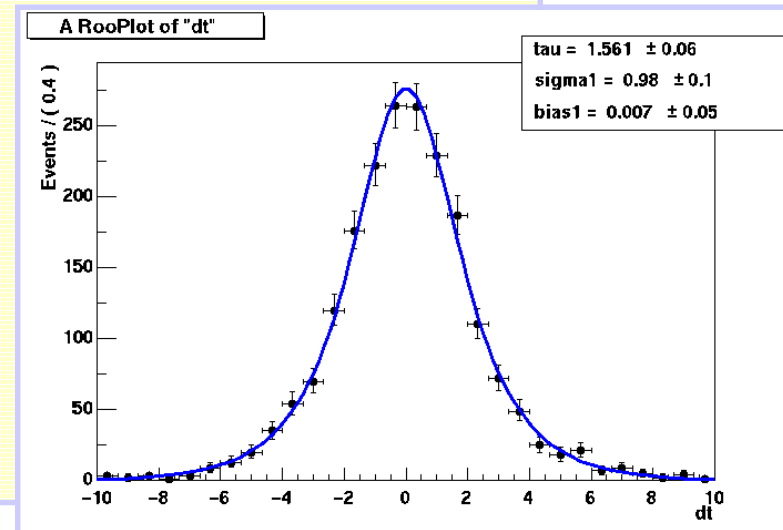
// Build a gaussian resolution model
RooRealVar bias("bias","bias",0,-5,5) ;
RooRealVar sigma("sigma","sigma",1,0.1,2.0) ;
RooGaussModel gm("gm","gauss model",dt,bias,sigma) ;

// Construct a decay (X) gm
RooDecay decay("decay","decay",dt,tau,gm,RooDecay::DoubleSided) ;

// Generate toy monte carlo dataset from the decay PDF
RooDataSet *data = decay.generate(dt,2000) ;

// Fit the generated data to the model
RooFitResult* r = decay.fitTo(*data) ;
r->correlation(sigma,tau) ;
-0.818443

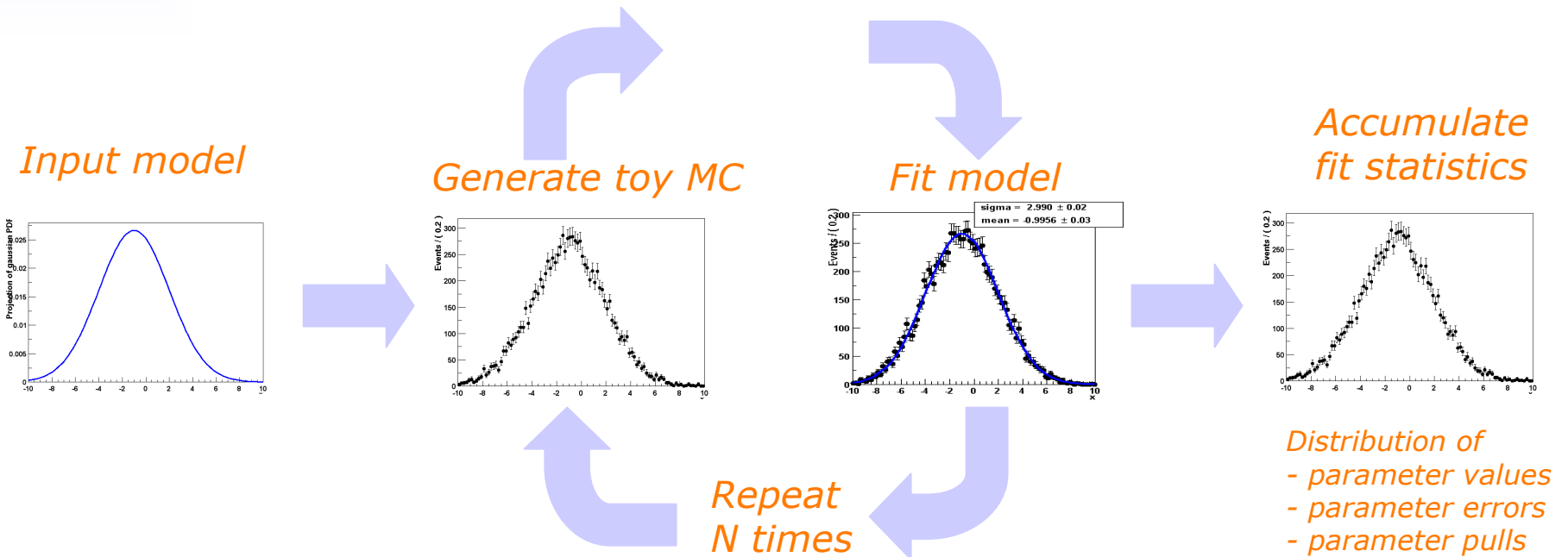
// Make a plot of the data and PDF
RooPlot* dtframe = dt.frame(-10,10,30) ;
data->plotOn(dtframe) ;
pdf.plotOn(dtframe) ;
pdf.paramOn(dtframe) ;
dtframe->Draw() ;
```





Advanced features – Task automation

- Support for routine task automation, e.g. goodness-of-fit study



```
// Instantiate MC study manager
RoomCStudy mgr(inputModel) ;

// Generate and fit 100 samples of 1000 events
mgr.generateAndFit(100,1000) ;

// Plot distribution of sigma parameter
mgr.plotParam(sigma) ->Draw()
```




Advanced features – Data sub-classing

Probability density function $F(\vec{x}; \vec{p}, \vec{q})$

- Physical parameters of interest \vec{p}
- Other parameters \vec{q} to describe detector effect (resolution, efficiency, ...)

- **Goal:** Exploit discrete information available in data sample to enhance statistical sensitivity of measurement of \vec{p}
 - For example: reconstruction technique, particle decay mode, ...
- **Method:** Use separate PDFs, with tailored set of parameters \vec{q} , for each data subset. Fit all sub-samples simultaneously.
- **Challenge:** Build PDFs $F_A(\vec{x}; \vec{p}, \vec{q}_A), F_B(\vec{x}, \vec{p}, \vec{q}_B), \dots$
 - Code replication and bookkeeping can be cumbersome
- **Solution:** Use RooFit replication & customization manager
 - User provides single 'prototype' $F(\vec{x}, \vec{p}, \vec{q})$
 - Automatic building of $F_A(\vec{x}; \vec{p}, \vec{q}_A), \dots$ from prototype and customization prescription



Advanced features – Automated PDF building

- RooFit PDF replication & customization manager
 - Takes 'prototype' PDF and dataset with discrete subset index variable
 - Makes customized copies following a customization prescription

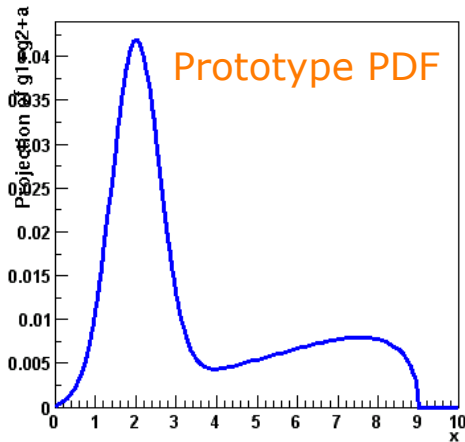
Replicated PDFs will have individually named copies of parameters f, s

Dataset ($x, \text{subsetID}$)

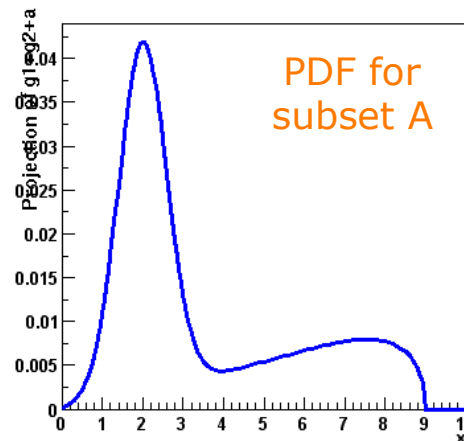
Customization prescription

$f \rightarrow f[\text{subsetID}]$
 $s \rightarrow s[\text{subsetID}]$

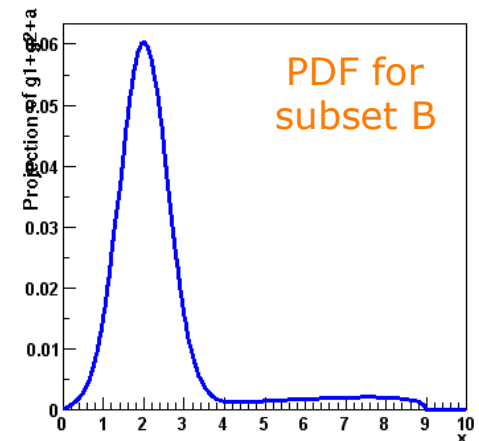
$$f * G(x; m, s) + (1 - f) * A(x; a, c)$$



$$f_A * G(x; m, s_A) + (1 - f_A) * A(x; a, c)$$



$$f_B * G(x; m, s_B) + (1 - f_B) * A(x; a, c)$$





Development and Use of RooFit in

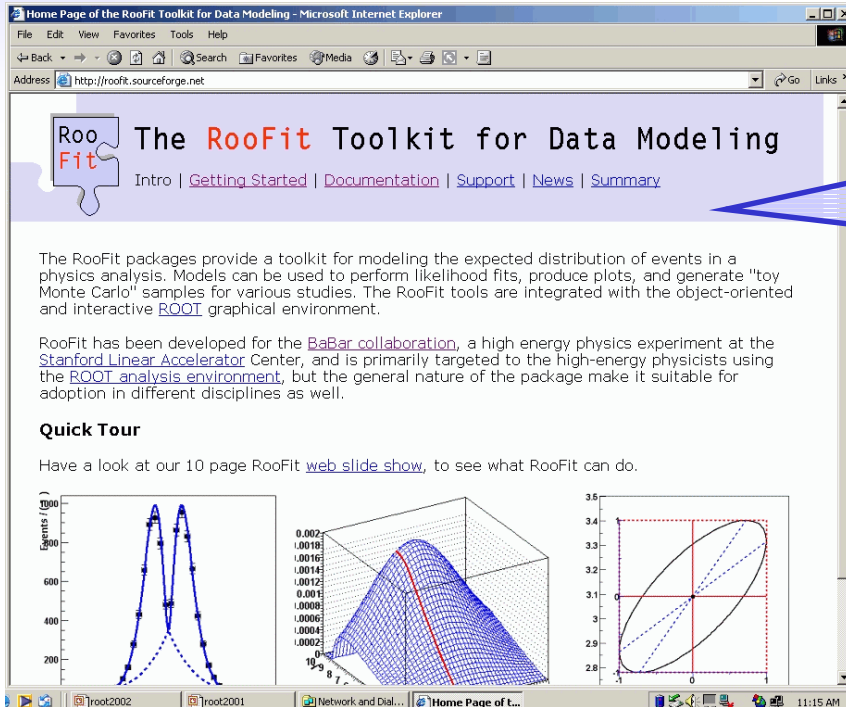
- Development

- RooFit started as RooFitTools (presented at ROOT2001) in late 1999
 - Original design was rapidly stretched to its limits
- Started comprehensive redesign early 2001
 - New design was released to BaBar users in Oct 2001 as RooFit
 - Extensive testing & tuning of user interface in the past year
- RooFit released on SourceForge in Sep 2002

- Current use

- Almost all BaBar analysis requiring a non-trivial fit now use RooFit or are in the process of switching to RooFit, e.g.
 - CP violation and mixing in hadronic decays ($\sin 2\beta'$)
 - B-Mixing in di-lepton events, $D^* \lambda \nu$ events
 - Measurement of $\sin 2\alpha_{(\text{eff})}$ from $B \rightarrow \rho \pi$, $B \rightarrow \pi \pi$
 - Searches for rare decays ($B \rightarrow \phi K_S$, $\eta' K_S$, ...)
- **Typical fit complexity**
 - **30 – 70 floating parameters**
 - **4-8 dimensions**
 - **PDF consists of 1000-10000 objects**
 - **Dataset of 500-100000 events**

Roofit at SourceForge - roofit.sourceforge.net



Home Page of the Roofit Toolkit for Data Modeling - Microsoft Internet Explorer

The Roofit Toolkit for Data Modeling

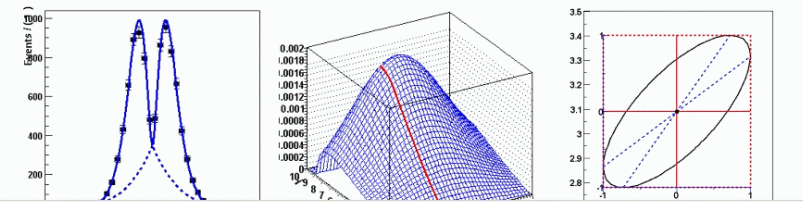
Intro | [Getting Started](#) | [Documentation](#) | [Support](#) | [News](#) | [Summary](#)

The Roofit packages provide a toolkit for modeling the expected distribution of events in a physics analysis. Models can be used to perform likelihood fits, produce plots, and generate "toy Monte Carlo" samples for various studies. The Roofit tools are integrated with the object-oriented and interactive ROOT graphical environment.

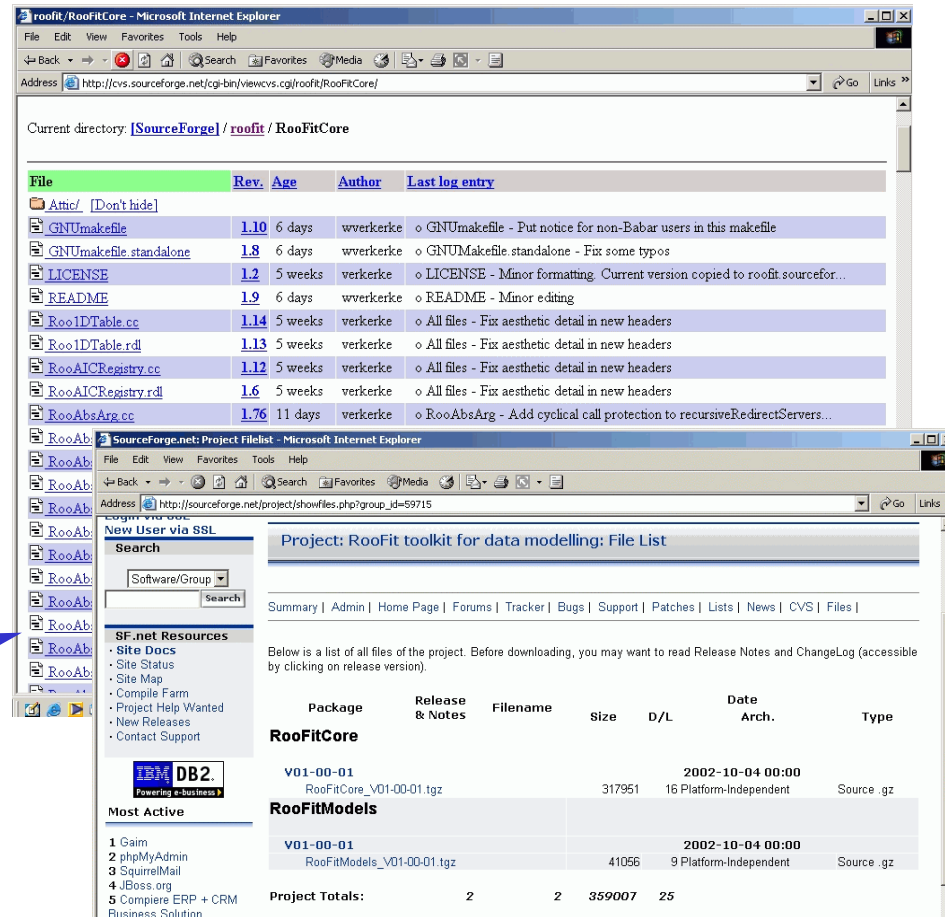
Roofit has been developed for the [BaBar collaboration](#), a high energy physics experiment at the [Stanford Linear Accelerator Center](#), and is primarily targeted to the high-energy physicists using the [ROOT analysis environment](#), but the general nature of the package make it suitable for adoption in different disciplines as well.

Quick Tour

Have a look at our 10 page Roofit [web slide show](#), to see what Roofit can do.



Roofit moved to SourceForge to facilitate access and communication with non-BaBar users



Current directory: [SourceForge](#) / [roofit](#) / [RoofitCore](#)

File	Rev.	Age	Author	Last log entry
Attic/ [Don't hide]				
GNUmakefile	1.10	6 days	wverkerke	o GNUmakefile - Put notice for non-Babar users in this makefile
GNUmakefile_standalone	1.8	6 days	wverkerke	o GNUmakefile_standalone - Fix some typos
LICENSE	1.2	5 weeks	verkerke	o LICENSE - Minor formatting. Current version copied to roofit sourcefor...
README	1.9	6 days	wverkerke	o README - Minor editing
RooidTable.cc	1.14	5 weeks	verkerke	o All files - Fix aesthetic detail in new headers
RooidTable.rdl	1.13	5 weeks	verkerke	o All files - Fix aesthetic detail in new headers
RooAICRegistry.cc	1.12	5 weeks	verkerke	o All files - Fix aesthetic detail in new headers
RooAICRegistry.rdl	1.6	5 weeks	verkerke	o All files - Fix aesthetic detail in new headers
RooAbsArg.cc	1.76	11 days	verkerke	o RooAbsArg - Add cyclical call protection to recursiveRedirectServers...

SourceForge.net: Project Filelist - Microsoft Internet Explorer

Project: Roofit toolkit for data modelling: File List

Summary | Admin | Home Page | Forums | Tracker | Bugs | Support | Patches | Lists | News | CVS | Files |

Below is a list of all files of the project. Before downloading, you may want to read Release Notes and ChangeLog (accessible by clicking on release version).

Package	Release & Notes	Filename	Size	D/L	Date Arch.	Type
RoofitCore						
V01-00-01		RoofitCore_V01-00-01.tgz	317951	16	2002-10-04 00:00	Platform-Independent Source .gz
RoofitModels						
V01-00-01		RoofitModels_V01-00-01.tgz	41056	9	2002-10-04 00:00	Platform-Independent Source .gz
Project Totals:	2	2	359007	25		

IBM DB2. Powering e-Business

Most Active

- 1 Gaim
- 2 phpMyAdmin
- 3 SquirrelMail
- 4 JBoss.org
- 5 Compiere ERP + CRM Business Solution

Code access

- CVS repository via pserver
- File distribution sets for production versions

RooFit at SourceForge - Documentation

Documentation

Comprehensive set of tutorials (PPT slide show + example macros)

Five separate tutorials

More than 250 slides and 20 macros in total

Slide 40 - Microsoft Internet Explorer

RooFit Tutorials

Slide 40

Discrete functions

- You can use discrete variables to describe cuts, e.g.
 - Signal, sideband mass windows
 - RooThresholdCategory
 - Defines regions of a real variable

Mass variable

```
RealVar m("m", "mass, 0, 10.");
```

Define threshold category

```
ThresholdCategory region("region", "Region of M", m, "Background");  
region.addThreshold(9.0, "Sideband");
```

Home Page of the RooFit Documentation

The two principal components of the RooFit documentation are

Tutorials

RooFit core design philosophy

- Composite functions → Composite objects

Here $f(w, x) = g(x) \rightarrow f(w, x) = f_1(x) \cdot f_2(x)$

If you are new to RooFit, start with the introductory tutorial to learn the basic interface.

Class Reference

```
class RooMinuit : public TObject
```

Class Description

RooMinuit is a wrapper class around TMinuit/THminuit that provides a seamless interface between the MINUIT functions and the native RooFit interface.

RooFit Class Index

RooFit Toolkit for Data Modeling

V01-00-01 Versi

Index

- Roo1DTable 1-dimensional table
- Roo2DKeysPdf Non-Parametric Multi Variate KEYS PDF
- RooAbsArg Abstract variable
- RooAbsBinning Abstract base class for binning specification
- RooAbsCategory Abstract index variable
- RooAbsCategoryValue Abstract modifiable index variable
- RooAbsCollection Collection of RooAbsArg objects
- RooAbsData Abstract data collection
- RooAbsFunc Abstract real-valued function interface
- RooAbsGenContext Abstract context for generating a dataset from a PDF
- RooAbsGenesPDF Abstract real-valued variable
- RooAbsHiddenReal Abstract hidden real-valued variable
- RooAbsIntegrator Abstract interface for real-valued function integrators
- RooAbsValue Abstract variable
- RooAbsPdf Abstract real-valued variable

Class reference in HTML style

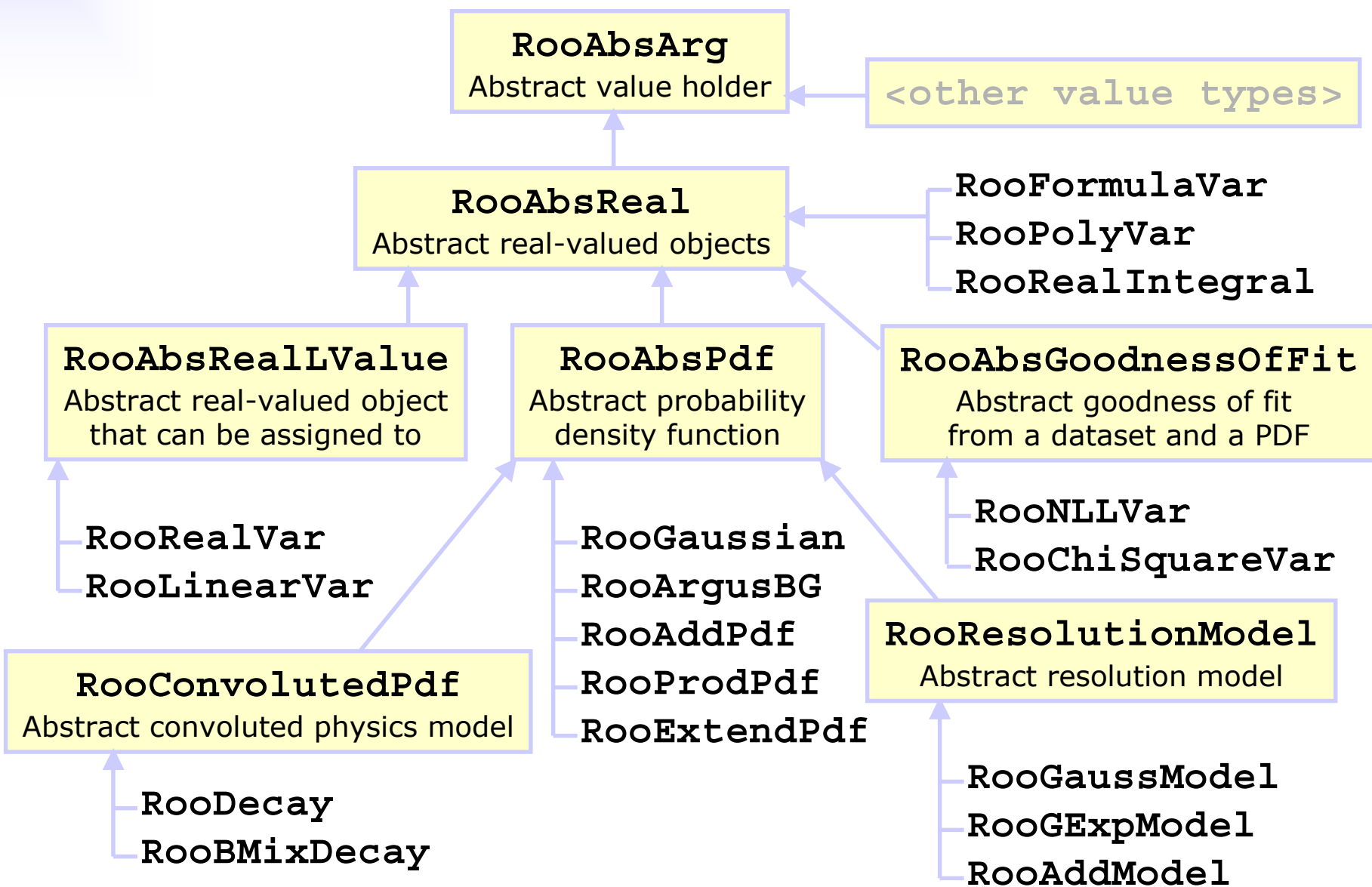


Summary / ROOT issues

- RooFit has been developed into a stable product in the course of $2\frac{1}{2}$ years
- Now available to all (HEP) users: `roofit.sourceforge.net`
 - Documentation in good shape
- ROOT issues
 - In general, ROOT works great
 - RooFit only uses a small set of available ROOT features
 - Mostly trees and graphics primitives, MINUIT
 - `TFitter/TMinuit` has no good interface to extract correlation matrix
 - Must dig into MINUIT memory space to retrieve info, not very robust
 - CINT is steadily improving, but still running into limitations once in a while.



Hierarchy of classes representing a value or function





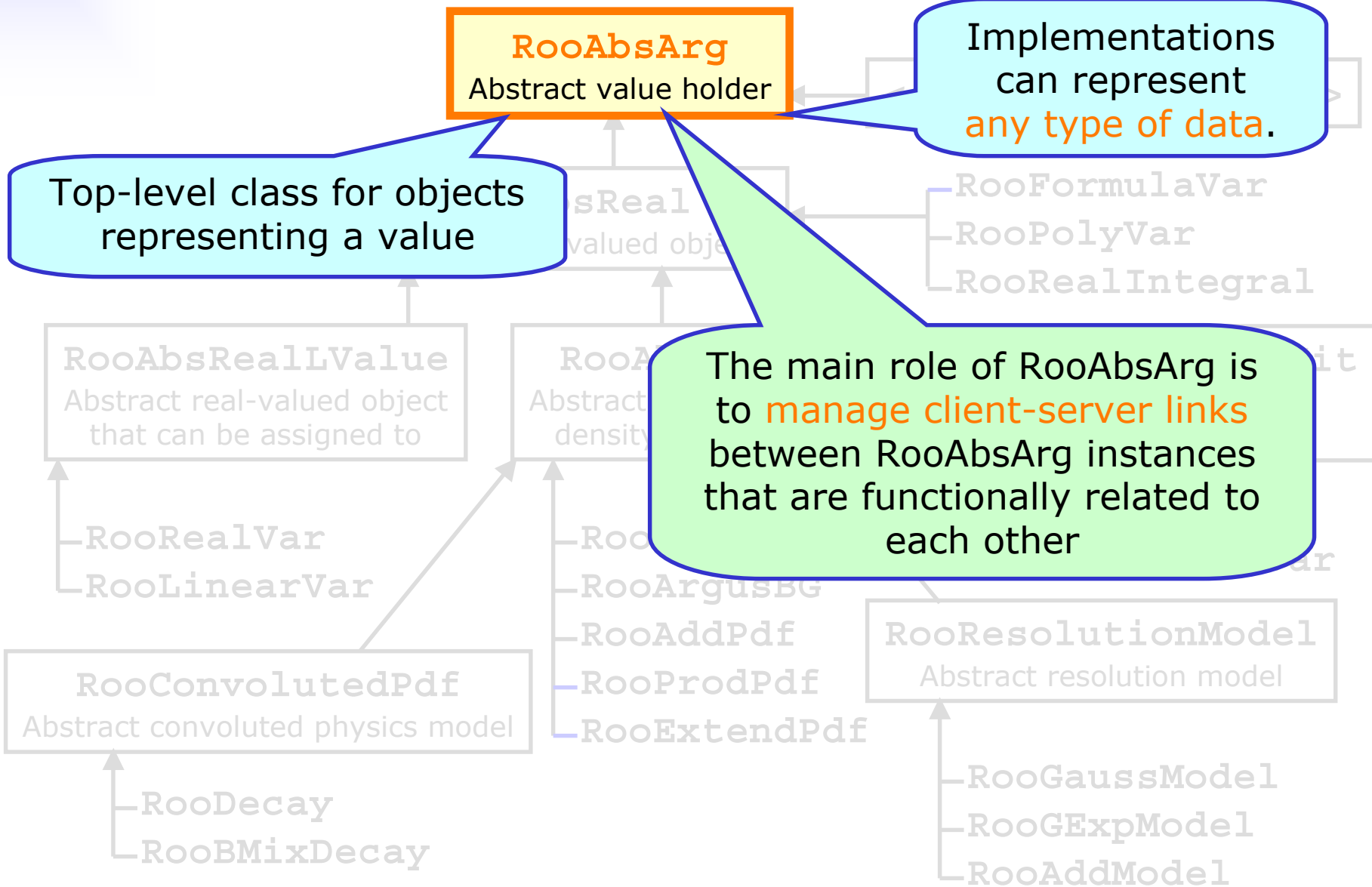
Class RooAbsArg

RooAbsArg
Abstract value holder

Implementations can represent **any type of data.**

Top-level class for objects representing a value

The main role of RooAbsArg is to **manage client-server links** between RooAbsArg instances that are functionally related to each other





Class RooAbsReal

Abstract base class for objects representing a real value

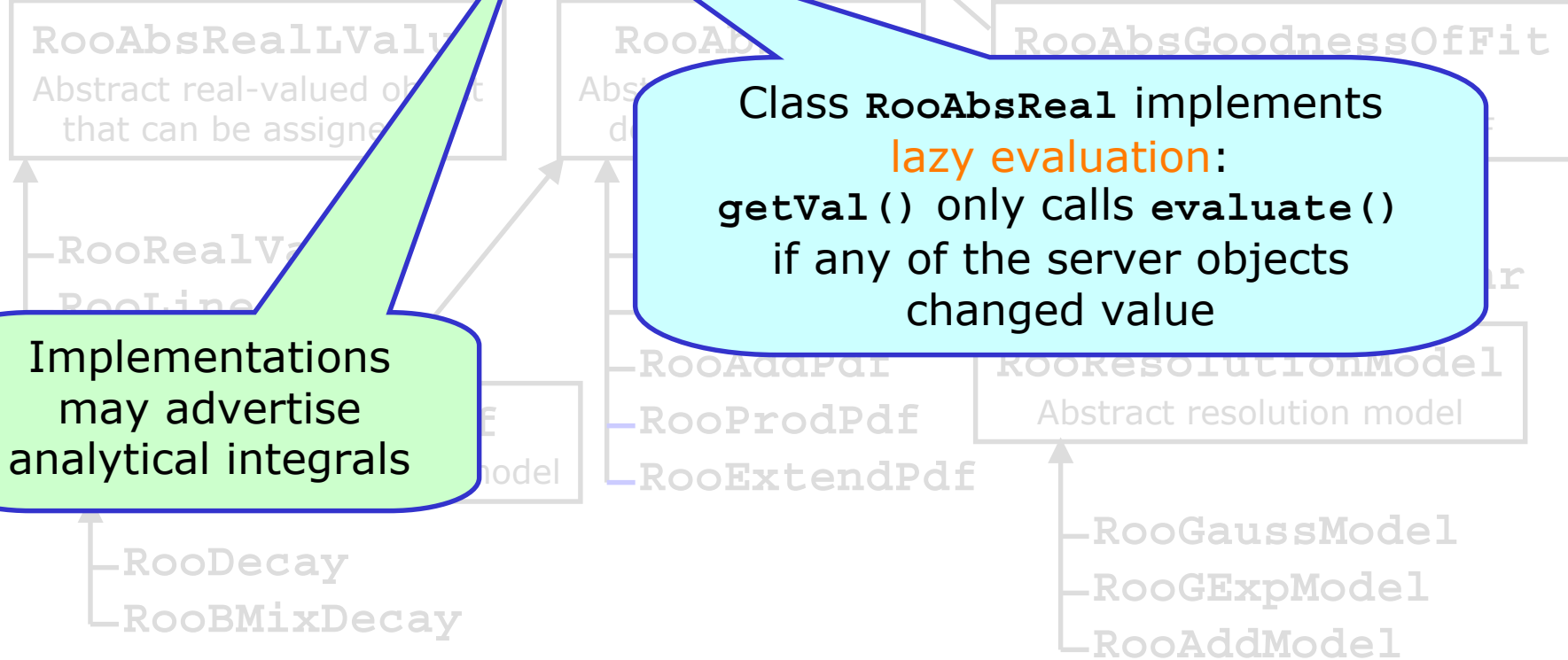
RooAbsReal
Abstract real-valued objects

<other value types>

- RooFormulaVar
- RooPolyVar
- RooRealIntegral

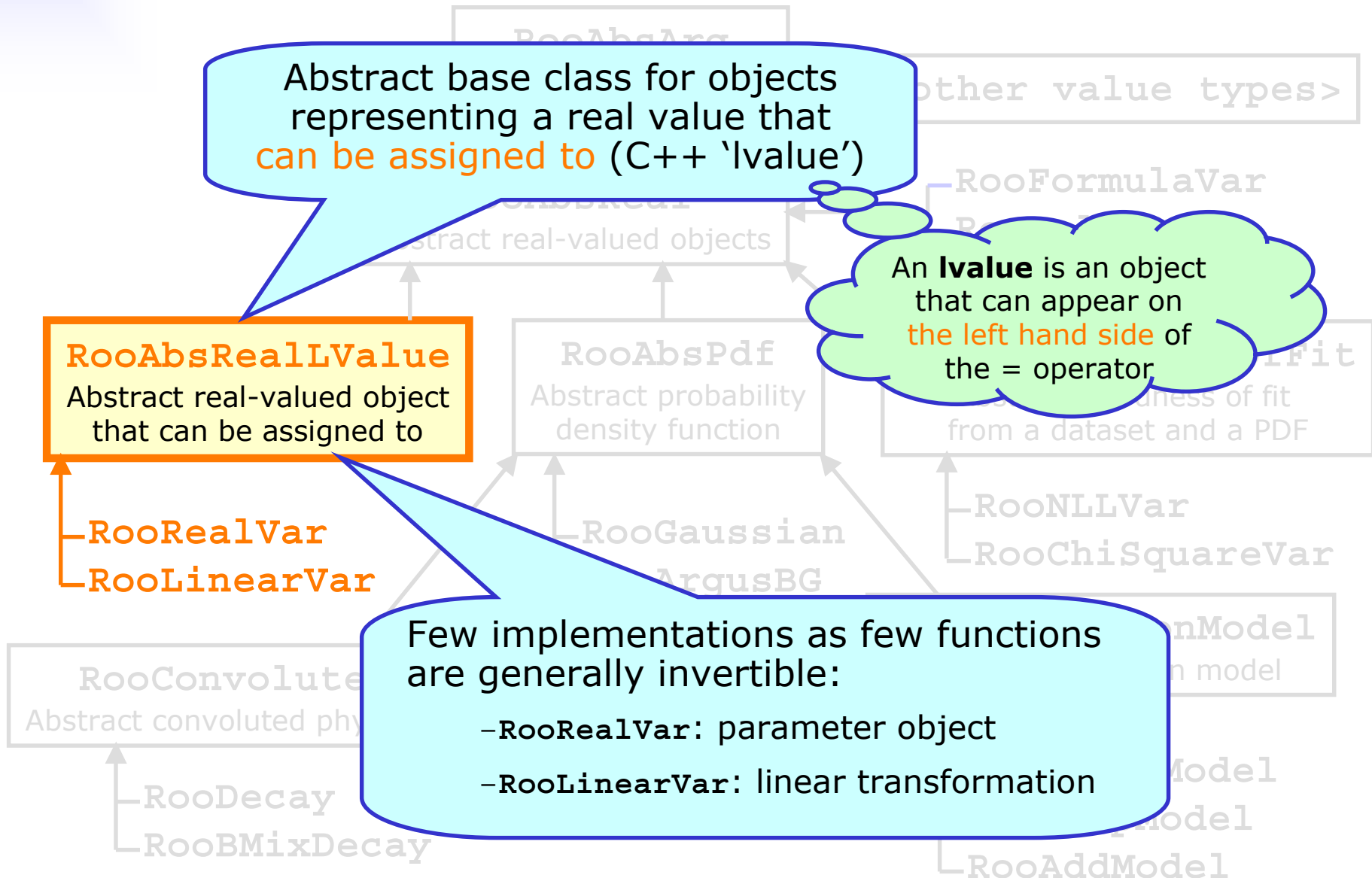
Class RooAbsReal implements **lazy evaluation**:
getVal() only calls evaluate() if any of the server objects changed value

Implementations may advertise analytical integrals





Class RooAbsRealLValue



Abstract base class for objects representing a real value that **can be assigned to** (C++ 'lvalue')

RooAbsRealLValue
Abstract real-valued object that can be assigned to

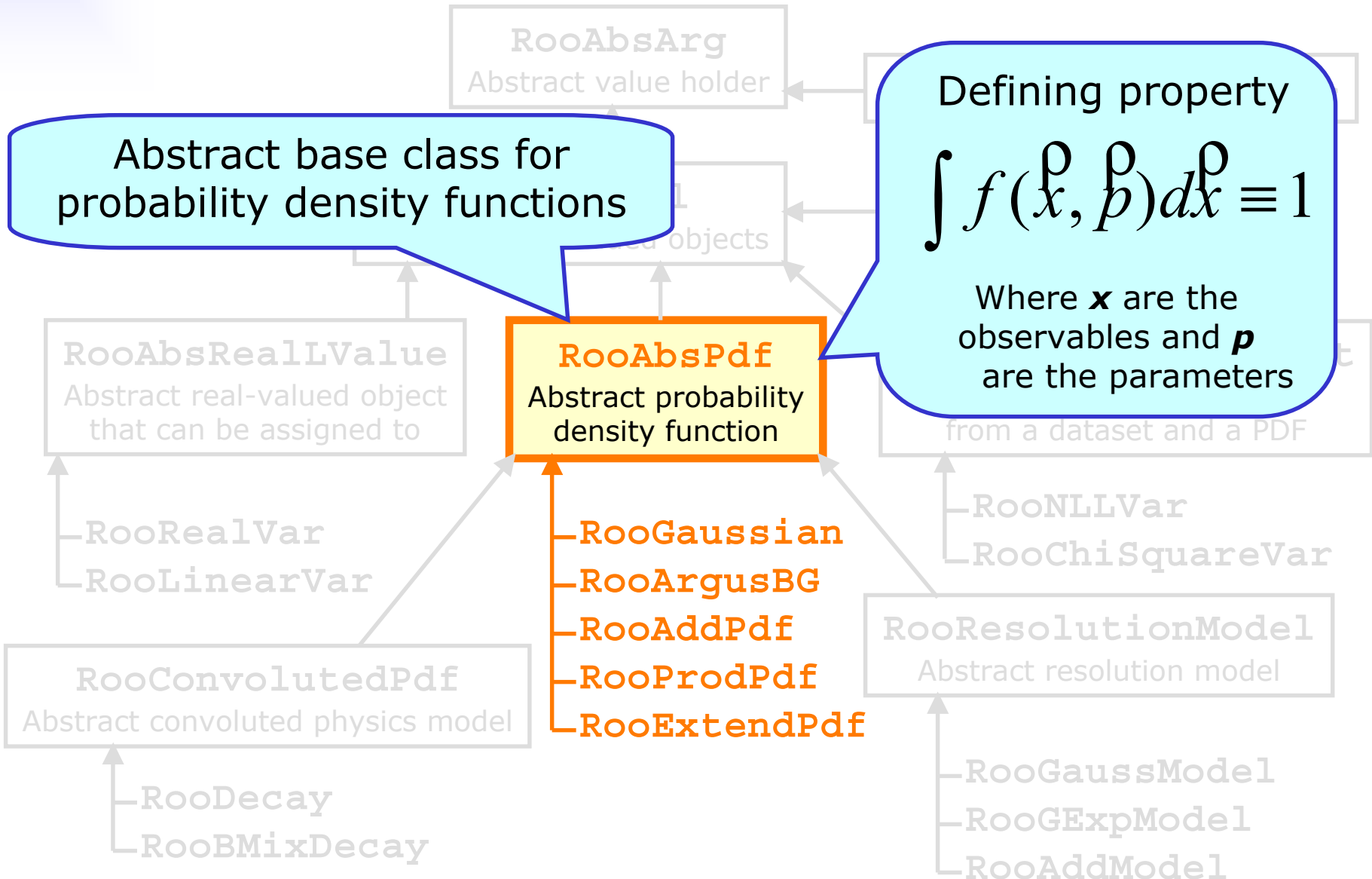
RooRealVar
RooLinearVar

An **lvalue** is an object that can appear on **the left hand side** of the = operator

Few implementations as few functions are generally invertible:

- RooRealVar: parameter object
- RooLinearVar: linear transformation

Class RooAbsPdf



Class RooConvolvedPdf

Implements $f_i(dt, \dots) \otimes R(dt, \dots)$
RooResolutionModel

$$P(dt, \dots) = \sum_k c_k(\dots) (f_k(dt, \dots) \otimes R(dt, \dots))$$

RooConvolvedPdf (physics model)

Implements c_k , declares list of f_k needed
No convolutions calculated in this class!

RooConvolvedPdf

Abstract convoluted physics model

— **RooDecay**

— **RooBMixDecay**

Abstract base class for
PDFs that can be convoluted
with a resolution model

RooResolutionModel
Abstract resolution model

— RooArgusBG

— RooAddPdf

— RooProdPdf

— RooExtendPdf

— RooChiSquareVar

> value types

FormulaVar

Var

Integral

GoodnessOfFit

Goodness of fit
set and a PDF

— RooLVar

— RooLVar



Class RooResolutionModel

Implementations of RooResolutionModel are **regular PDFs** with the **added capability** to calculate their function convolved with a series of 'basis' functions

Resolution model advertises which basis functions it can handle

To be used with a given RooConvolvedPdf implementation, a resolution model must **support all basis functions used by the RooConvolvedPdf**



- RooGaussModel
- RooGExpModel
- RooAddModel



Class RooAbsGoodnessOfFit

Provides the framework for efficient calculation of goodness-of-fit quantities.

A goodness-of-fit quantity is a function that is calculated from

- A dataset
- the PDF value for each point in that dataset

RooAbsGoodnessOfFit

Abstract goodness of fit from a dataset and a PDF

RooNLLVar

RooChiSquareVar

Built-in support for

- **Automatic constant-term optimization** activated when used by RooMinimizer(MINUIT)
- **Parallel execution on multi-CPU hosts**
- Efficient **calculation of RooSimultaneous** PDFs



Class tree for discrete-valued objects

