

RooRarFit

Lei Zhang

UC Riverside

ROOT/RooFit Workshop

12/06/2007

What is RooRarFit

- A general ML fitter based on ROOT/RooFit

Why use RooRarFit?

- A question asked by many people: Why so many people write their own fitters with ROOT/RooFit? What's wrong?
- ROOT/RooFit are toolkits, engines, so one needs to build a fitter, an application, to make use of them. So nothing wrong!
- RooRarFit is one of such fitters, applications, but as a **general** fitter, it can be used by many different analyses, with benefits:
- Access to the full power of RooFit
- Coding free to final users (no C++ experience needed)
- Driven by readable text configuration files
- Flexible design, 'programmable' config files
- Lots of fitting, plotting functions implemented
- Fully documented, lots of working examples from many analyses already

RooRarFit implementation

- Wrappers of RooRealVar, RooDataSet, RooPDFs, etc. implemented, so can be defined through ascii configs
 - Dataset classes
 - PDF classes
- Fitting actions implemented as functions driven by ascii configs
- Auxiliary classes, main program, scripts
 - String parser class
 - NLL class to deal with NLL curve
 - Minuit class derived from RooMinuit for contour plot
 - Version control header file
 - Main program
 - submitToy (toy study batch processor for [slac](#), [cu-boulder](#), [ed](#), [ral](#))
- Assume users have experience with ROOT/RooFit, so no ROOT/RooFit details here

Get and run RooRarFit

- Outside of BaBar: <http://rarfit.sourceforge.net/>
- With Babar SRT release version $\geq 16.0.3$ -physics-1
`addpkg RooRarFit V00-01-XX`
`gmake RooRarFit.all`
- In workdir, run the application (`rarFit`) without any args to get short help page:

```
rarFit [-options] <RooRarFit_Config_file>
  -h this help page
  -D <data input section> (default "Dataset Input")
  -C <mlFitter config section> (default "mlFitter Config")
  -A <fitter action section> (default "Fitter Action")
  -t <toy job id> (default 0)
  -n <toyNexp> (default 0, use config)
  -d <toy dir> (default .toyData)
```

e.g.

To run rarFit from config file demo.config

```
rarFit demo.config
```

and with mlfitter config from section [myMLFitter]

```
rarFit -C myMLFitter demo.config
```

and with mlfitter action from section [my Fit Action]

```
rarFit -C myMLFitter -A "my Fit Action" demo.config
```

RooRarFit config file

- Dataset definition section
 - Define dataset structures
- Dataset input section
 - Define, read in datasets
- PDF configuration section
 - PDFs are created from top to bottom with PDF config sections.
 - Simple, composite PDF
- Fit action section
 - pdfFit
 - toyStudy
 - mlFit
 - scanPlot
 - projPlot
 - contourPlot
 - sPlot

Dataset Definition & Input

Define the structure of dataset entry

&

Read in datasets

Dataset definition section

- Define dataset entry structure with dsd section
 - “Fields” declares dataset columns
 - Define all declared columns in this section
 - Observable definition syntax

```
columnName = <columnType> "<title>" <specifications>
```

Obs/Param types: `RoorealVar`, `Roocategory`, `RoostringVar`,
`RoocnstVar`, `RoounblindPrecision`, `RoomappedCategory`,
`RoorthresholdCategory`, `RooformulaVar`

```
[Dataset Definition]
Fields = runNum mes de fisher
runNum = RooRealVar "runNum" 0 1e10
mes = RooRealVar "M_{es}" 5.19 5.29 B(50) "GeV"
de = RooRealVar "#Delta E" -.2 .2 B(45) "GeV"
fisher = RooRealVar "Fisher" -4 5 B(45)
```

columnType

Title

RRV range Plot bins

Latex in RooFit

RRV unit

Dataset definition section (Cont.)

- "AddOns" declares derived columns
- Define all declared 'add-on' columns in this section as well
- columnType can be:

RoMappedCategory, RooThresholdCategory, RooFormulaVar

[Dataset Definition]

```
Fields = runNum mes de fisher
runNum = RooRealVar "runNum" 0 1e10
mes = RooRealVar "M_{es}" 5.19 5.29 B(50) "GeV"
de = RooRealVar "#Delta E" -.2 .2 B(45) "GeV"
fisher = RooRealVar "Fisher" -4 5 B(45)
AddOns = runBlock
runBlock = RoMappedCategory "Run Block" noIdx runRange \\
    Run1    "*Run1" "Run1"    "*Run2" "Run2"    \\
            "*Run3" "Run3"    "*Run4" "Run4"
runRange = RooThresholdCategory "Run Range" noIdx runNum \\
    "mcRun2" 18000 "daRun1" 30000 "daRun2" \\
            40000 "daRun3" 200000 "daRun4" \\
            770000 "mcRun1"
```

Dataset input section

- Input datasets with dsi section
 - “Datasets” declares all datasets to be input
 - Define all declared datasets in this section
 - Dataset definition syntax

```
datasetName = <inputType> "<title>" <specifications>
```

Dataset input types: **ascii**, **root**, **reduce**, **add**

```
[Dataset Input]
```

```
Datasets = onData sigMC bbMC gsbData desbData  
onData = ascii "on peak data" "mydats/on.text" Q  
sigMC = ascii "sig MC" "mydats/sigMC.txt" Q  
bbMC = ascii "peaking BB MC" "mydats/bbMC.txt" Q // Q for quiet mode  
gsbData = reduce "gsb Data" onData "mes<5.27"  
desbData = reduce "de sb Data" onData "(de<-.1) || (de>.1)"
```

Dataset file name

inputType

title

source dataset

reduce cuts

Dataset input section (Cont.)

- Other (optional) configs for dsi section
 - dsdSec = <dataset definition section name>
Default: **Dataset Definition**
 - setWeightVar = <obs used as weightVar>
Default: no weightVar for datasets
 - tabulateDatasets = yes
Tabulate datasets for each RooCategory observable.
Default: no tabulation
 - computeCorrelations = <yes|no|dataSetName1 ...>
Compute correlation matrices for (given) datasets.
Default: yes

PDF Configuration Sections

Define PDFs from top to bottom
with config sections

PDF config sections

- Master fitter config section
 - Starting point for all PDFs, also include:
 - Global configs
 - Simultaneous fit config for RooFit

RooSimPdfBuilder Configs

Configured in its own section, see next slide

```
[mlFitter Config]
Comps = myModel // can have multiple ML models for simFit
fitData = onData // Default dataset, also used as prototype dataset
simultaneousFit = yes // Build and use simPdf if set to yes
physModels = the_myModel // models for simPdf
splitCats = runBlock // List all splitting categories here
the_myModel = runBlock : mesSig_shift \
                runBlock : Frac_nSig_runBlock[Run4], \
                Frac_nBkg_runBlock[Run4], \
                Frac_nChls_runBlock[Run4]
```

PDF config sections

- ML model config section
 - Each model declared in ML fitter section has its own config section
 - `configStr` set to ``MLPdf'` for ML models
 - Declare as many species as wanted with config ``Comps'`
 - Each component has its own config section (see next slide)
 - Corresponding yields listed with config ``Coeffs'`
 - Declared yields defined within the section

Declared as full names

```
[myModel Config]
configStr = MLPdf "ml yield model"
Comps = Sig Chls Bkg // component PDFs for this model
Coeffs = nSig nChls nBkg // component yields for this model
nSig = nSig 100 L(-100 - 1000)
nChls = nChls 100 L(-100 - 1000)
nBkg = nBkg 2000 L(-100 - 10000)
postPdfFloat = nSig nChls nBkg
```

Optional title

RRV stream output

Float yields

PDF config sections

- Species config sections
 - Each species in ML model has its own config section
 - `configStr` can be set to any valid PDF type in RooRarFit, but usually `ProdPdf` for composite PDF as product of PDFs for observables included in the model

Signal config section

```
[Sig Config]
configStr = ProdPdf "Signal Pdf"
Comps = deSig mesSig fisSig // Product components
fitData = sigMC // Use sigMC to get pdf params
```

Chmls B Bkg config section

```
[Chls Config]
configStr = ProdPdf "Charmless B bkg"
Comps = deChls mesChls fisChls
fitData = bbMC // Use bbMC to get pdf params
```

Continuum config section

```
[Bkg Config]
configStr = ProdPdf "Continuum Bkg"
Comps = deBkg mesBkg fisBkg
fitData = gsbData // Use gsbData to get pdf params
```

Defined in their own sections

PDF config sections

- Signal PDF config sections

```
[deSig Config]
configStr = TwoGauss
x = de
meanC = 0.      L(-.1 - .1)
meanT = -0.04   L(-.1 - .1)
sigmaC = 0.025  L(0.0 - .15)
sigmaT = 0.09   L(0.0 - 0.3)
fracC = 0.75    L(0.5 - 1.0)
```

```
[mesSig Config]
configStr = TwoGauss
x = mes
meanC = 5.28     L(5.25 - 5.29)
meanT = 5.27     L(5.25 - 5.29)
sigmaC = 0.0028  L(0.0 - 0.01)
sigmaT = 0.008   L(0.0 - 0.10)
fracC = 0.95     L(0.5 - 1.0)
shift = 0.0      C L(-.1 - 0.1)
```

```
[fisSig Config]
configStr = BGGauss
x = fisher
mean  = -0.5 L(-2 - 2)
rms   = 0.6 L(0.0 - 1)
asym  = 0.1 L(-.5 - 0.5)
```

If full name is not specified for params, RRV name will be prefixed with PDF name, for example, `fisSig_mean`, `fisSig_rms`, and `fisSig_asym`.

In RooRarFit, params belong to RooRarFit PDF objects, so full names are needed to distinguish same param names from different PDFs.

PDF config sections

- Charmless B background PDF config sections

```
[deChls Config]
configStr = TwoGauss
x = de
meanC = 0.13 L(-.2 - .2)
meanT = -0.20 L(-.3 - .2)
sigmaC = 0.06 L(0.0 - .3)
sigmaT = 0.14 L(0.0 - 0.3)
fracC = 0.35 L(0.0 - 1.0)

[mesChls Config]
configStr = AddPdf
Comps = mesChlsG mesChlsA
Coeffs = fracG
fracG = T "f_{G}" 0.08 L(0 -1)
[mesChlsG Config]
configStr = Gaussian
x = mes
mean = 5.28 L(5.25 - 5.29)
sigma = 0.004 L(0.0 - 0.1)
```

```
[mesChlsA Config]
configStr = ArgusBG
x = mes
max = 5.29 C
c = -30 L(-80 - -.1)

[fisChls Config]
configStr = BGGauss
x = fisher
mean = -0.4 L(-2 - 2)
rms = 0.6 L(0.0 - 1)
asym = 0.1 L(-.5 - 0.5)
```

PDF config sections

- Continuum background PDF config sections

```
[deBkg Config]
configStr = Polynomial
x = de
nOrder = 2
P01 = -1.5 L(-100 - 100)
P02 = 2.0 L(-100 - 100)
postPdfFloat = P01
```

```
[mesBkg Config]
fitData = desbData
configStr = ArgusBG
x = mes
max = 5.29 C
c = -15 L(-80 - -.1)
postPdfFloat = c
```

```
[fisBkg Config]
configStr = AddPdf
Comps = fisBkgC fisBkgT
Coeffs = fracC
fracC = T "f_{C}" 0.97 L(0 -1)
[fisBkgC Config]
configStr = BGGauss
x = fisher
mean = T "#mu_{C}" -0.4 L(-2 - 2)
rms = T "#sigma_{C}" 0.6 L(0.0 - 1)
asym = T "A_{C}" 0.1 L(-.5 - 0.5)
[fisBkgT Config]
configStr = Gaussian
x = fisher
mean = T "#mu_{T}" 0.8 L(-1 - 2)
Sigma = T "#sigma_{T}" 2.0 L(0 - 5)
```

PDF config sections

- RooFit PDFs implemented:
 - RooProdPdf
 - RooAddPdf
 - RooAddModel
 - RooSimPdfBuilder
 - RooArgusBG
 - RooBDecay
 - RooDecay
 - RooBifurGauss
 - RooCBShape
 - RooExponential
 - RooGaussian
 - RooBreitWigner
 - RooLandau
 - RooGaussModel
 - RooGenericPdf
 - RooKeysPdf
 - Roo2DKeysPdf
 - RooPolynomial
 - RooChebychev
 - RooParametricStepFunction
 - Triple Gaussian
 - Triple Guassian Model
 - Gexp
 - Double Gaussian (w/ scale and shift)

Fit Action Sections

Guide fitter to finish its jobs

- pdfFit
- toyStudy
- mlFit
- scanPlot
- projPlot
- contourPlot
- sPlot

Fit action - pdfFit

- Get PDF params from fits of individual PDFs to datasets
 - Input: init values in PDF config sections
 - Output: fitted params in intermediate file; PDF plots in ROOT file

```
// Action section name, command line option -A PdfAct  
[PdfAct]  
// pdfFit options  
pdfFit = yes // enable pdfFit action  
postPdfMakePlot = yes // To output PDF plots into root file  
postPdfWriteParams = yes // To output params to text file
```

Other configs for pdfFit action

- **pdfToFit** = <pdf1> ...

Only those listed PDFs will be fitted if this config exists.

- **postPdfReadSecParams** = <no|yes|secName>

Override params from PDF fits with values listed in action section (and section if specified here).

Fit action - toyStudy

- toyStudy action is to validate the fitter
 - Input: param values from pdfFit action
 - Output: toyStudy results (pulls, etc) in ROOT file

```
// Action section name, command line option -A ToyAct
[ToyAct]
// toyStudy options
toyStudy = yes // enable toyStudy action
preToyReadParams = yes
preToyReadSecParams = yes
nSig = 93 +/- 10 L(-10 - 3000)
nChls = 200 +/- 50 L(-10 - 3000)
protDatasets = onData
toyNexp = 100
toyNevt = 0 floated // 0: nEvt from protDataset; and fluctuation.
toySrc_nSig = sigMC "@0 nSig"
toySrc_nChls = bbMC "@0 nChls"
toySrc_nBkg = gsbData 1 pdf "@0-1 nBkg"
```

Override params from pdfFit or init values

If not specified, `pure toy'.
a variety of `embd toys' if specified

Fit action - toyStudy advanced features

- Param scanning

Validate the fitter on all possible values for some params, for example, signal yields.

```
[ToyAct]
toyStudy = yes // enable toyStudy action
nSig = 93 +/- 10 L(-10 - 3000)
protDatasets = onData
toyNexp = 100
preToyRandParams = nSig "@0+@1 nSig nSigGen" \\
                  nBkg "@0-@1 nBkg nSigGen"
preToyRandGenerators = nSigGenPdf

// PDF config section for param randomizer
[nSigGenPdf Config]
configStr = Generic
formula = "1" nSigGen
nSigGen = nSigGen 0 L(-43 - 57) // nSig scan range (50, 150)
```

Fit action - toyStudy advanced features

- Prototype datasets
 - Default RooFit prototype dataset facilities supported, and
 - Can have prototype dataset for each species
 - Can have prototype obs generated from specified PDFs
- `toyEmbdUnCorrelate = <obsGroup1 >...`

Uncorrelate obs among groups for embedded events, for example

```
toyEmbdUnCorrelate = de mes "fisher resMass"
```

- `postEmbdRandObs`

Re-generate obs for embedded event, for example

```
postEmbdRandObs = sigMC tagFlav trueDtModel
```

Fit action - mlFit

- Fit full ML fitter on dataset to get yields, etc.
 - Input: param values from pdfFit action
 - Output: final params in intermediate file, mlFit results, including:
 - Yields, floating params
 - Significance of any floating param wrt any particular value
 - Systematic errors of floating params due to the uncertainty of fixed params

```
// Action section name, command line option -A MLAct
[MLAct]
// mlFit options
mlFit = yes // enable mlFit action
mlFitData = onData
postMLSignf = nSig // signal yield signif. (wrt 0 by default)
postMLSysVars = nSig // calculate syst. error for nSig by varying fixed params
postMLSysParams = deSig_meanC deSig_meanT \\  
                  deSig_sigmaC deSig_sigmaT \\  
                  deSig_fracC
```

↓
List all fixed params to vary for syst. errors

Fit action - scanPlot

- Scan NLL within interested param spaces
 - Input: params from mlFit action
 - Output: NLL points within param spaces, NLL plot (for 1D scanning), in ROOT file

```
// Action section name, command line option -A ScanAct  
[ScanAct]  
// scanPlot options  
scanPlot = yes // enable scanPlot action  
scanPlotData = onData  
nScanPoints = 100 // Number of NLL values  
scanVars = nSig 0 100  
scanUnCorrErr = 5.1 // nll plot with uncorrelated error included  
scanCorrErr = 2.2 // nll plot with correlated error (and uncorrelated  
// if any) included
```

- UL, signif., calculation based on NLL plot is provided as static functions within RooRarFit class, rarMLFitter
- NLL curve operations: combine, shift, etc.

Fit action - projPlot

- Get projection plots for given species
 - Input: params from mlFit action
 - Output: projection plots, LLR histograms, asym plots, in ROOT file

```
// Action section name, command line option -A ProjAct
[ProjAct]
// projPlot options
projPlot = yes // enable projPlot action
projPlotData = onData
projComps = Sig // signal PDF to be projected
projLLRPlots = yes // create LLR histograms
projVars = mes de fisher // obs to get projection plots
projLRatioCut = .810
projLRatioCut_mes = .835
projLRatioCut_de = .920
projLRatioCut_fisher = .890
projFindOptimCut = yes
projOptimRange_mes = "abs(mes-5.28)<0.006"
projOptimRange_de = "abs(de)<0.07"
projOptimRange_fisher = "fisher<0.0"
projPlotCat = runBlock // get projection plot for each cat type
```

Likelihood ratio cuts

Find optimal cuts

Fit action - contourPlot

- Get contour plots for two floating params
 - Input: params from mIFit action
 - Output: contour plots in ROOT file

```
// Action section name, command line option -A ContourAct  
[ContourAct]  
// contourPlot options  
contourPlot = yes // enable contourPlot action  
contourPlotData = onData  
nContours = 3 // Number of contours  
contourVars = nSig 0 100 fL .3 1.4
```

Fit action - sPlot

- Get sPlots
 - Input: params from mlFit action
 - Output: sPlots in ROOT file

```
// Action section name, command line option -A SPlotAct
[ProjAct]
// sPlot options
sPlot = yes // enable sPlot action
sPlotData = onData
sPlotComps = all // sPlots for all species
sPlotVars = mes de fisher // obs to get sPlots
sPlotPdfOverlay = direct // direct PDF used for PDF overlay
sPlotIgnoredVars_heli = resMass
sPlotNormIgnoredObs = tagFlav dtErr
```

correlated obs should be removed

removed from PDF normalization due to PDF inconsistency across species

- Extended sPlot with fixed species
 - To be implemented soon

Documentation & Current Status

Documentation

- HTML reference manual comes with the package

[RooRarFit/doc/RooRarFit.html](#)

[Preface](#)

[1. RooRarFit Implementation](#)

[2. RooRarFit Configuration](#)

[3. Questions and Answers](#)

[Index](#)

- Lots of working examples from many analyses

[RooRarFit/doc/Sample_configs](#)

- Sample scripts to deal with RooRarFit ROOT output

[RooRarFit/doc/Sample_scripts](#)

- Source code documented using [doxygen](#)

Future plan

- More RooFit PDFs will be `wrapped' (added) as needed
- More functionalities, like Dalitz analysis, might be added
- More independent of Babar SRT
- More active developers
- Any comments, contributions, etc. are welcome

<http://rarfit.sourceforge.net>

Get RooRarFit! Reclaim Your Fitter!