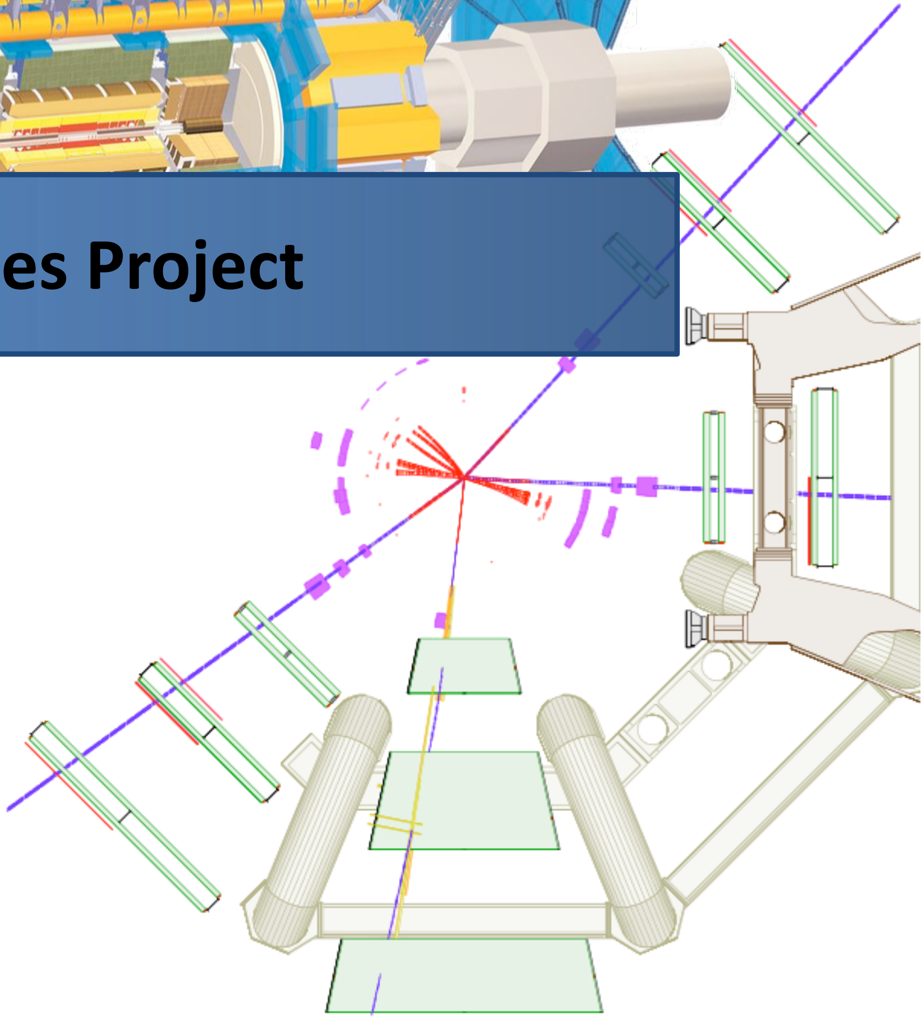
RooFitTrees Project

Will Buttinger

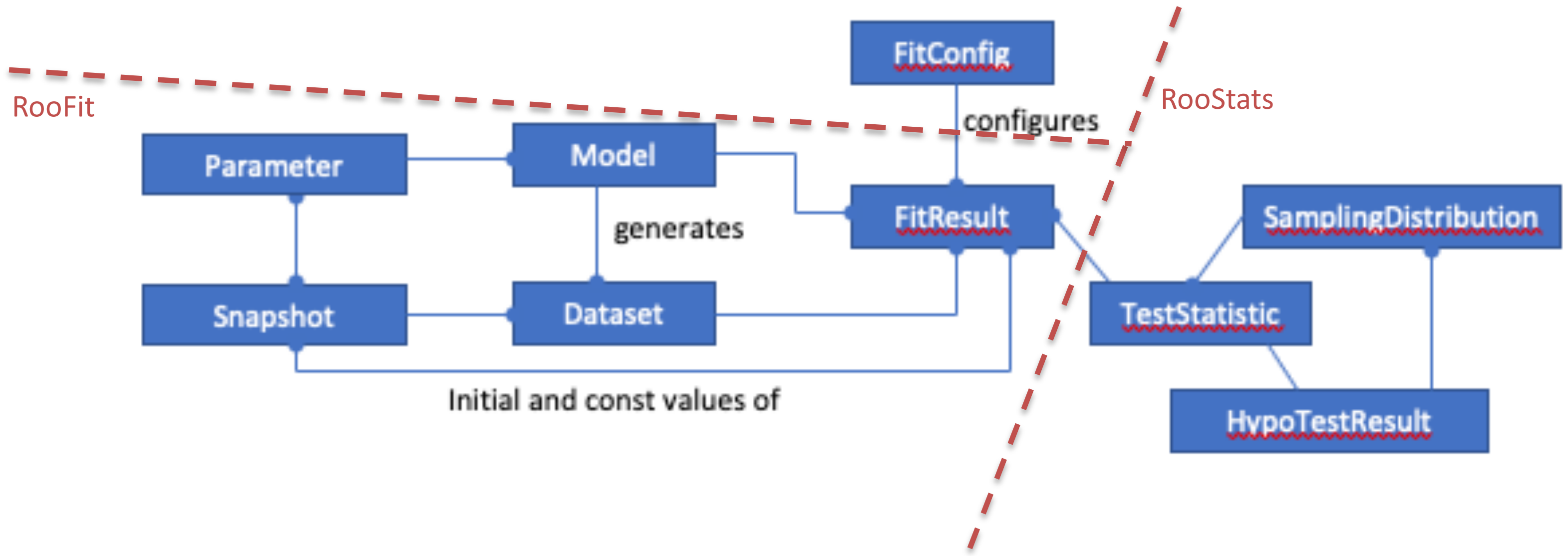
Tim Adye



What is (are?) RooFitTrees?

- RooFitTrees is a project that born out of two ideas:
  - Creation of a statistical analysis data model (Will)
  - Management of fitting procedure (Tim)

Original "SDM" entity relationship diagram



- RooFitTrees tries to manage efficiently creating, storing, and retrieving FitResults and Datasets, using TTree for backend storage
- The ROOT developers support this work and we plan to include it in ROOT when the project is mature enough

Working with RooFitTrees

- Two types of Tree: RooFitResultTree and RooDataTree
  - RooFitResultTree: entries (rows) of tree correspond to fit results
  - RooDataTree: entries correspond to generated datasets (toys or expected)
- Starting point assumes the user has the following:
  - A PDF (RooAbsPdf) - which is really just a function of a bunch of variables
    - Variables are either parameters or observables or global observables
  - Either an observed dataset or a list of the observables and global observables:
    - A dataset consists of both the *observed values* of the observables (in a RooAbsData) and the *observed values* of the global observables (in a RooArgSet)
    - Knowing the observables and global observables implicitly defines the parameters

Constructing Trees:

```
fits = ROOT.RooFitResultTree(“fits”,pdf);
toys = ROOT.RooDataTree(“toys”,pdf);
```

“Filling” the RooFitResultTree:

```
fitResult = fits.fitTo(data) # data is a std::pair<RooAbsData, RooArgSet>
```

“Filling” the RooDataTree:

```
data = toys.generate(fitResult, isExp) # fitResult is RooFitResult, isExp is bool
```

“Filling” the RooFitResultTree when you don’t have a dataset:

```
fitResult = fits.snapshotTo(obs,globs) # obs and globs are RooArgSet
```

More about RooFitTrees

- Information for fit results and toys stored in an underlying TTree
- Information such as fit configuration hyperparameters held in tree metadata
- Each RooFitTree can be explored like a regular TTree:
  - E.g. using Scan, Draw, etc

Drawing distribution of post-fit values of parName where that parameter was floating

```
fits.Draw("final.parName","const.parName!=const.parName")
```

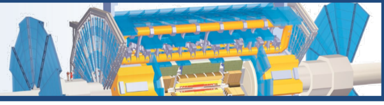
- Can also act like a TChain

Adding previously-saved fits to a FitResultTree

```
fits = ROOT.RooFitResultTree("fits",pdf);
fits.Add("previousFits/*.root") # load previous fit results
```

- RooFitTrees are great for distributed analysis workflows!

Ongoing R&D



- Ultimately want to be able to use RooFitTrees as underlying fitting and generating technology for statistical analysis in ROOT

- This means being able to e.g. "retrieve" a Profile Likelihood Ratio test statistic distribution from the trees

Two fits to a dataset (one floating POI, one with it fixed)

- Still working on the best way to achieve this. Two possible approaches so far:
  1. Advanced usage of 'friend' tree concept to connect RooDataTrees and RooFitResultTrees
  2. Methods for retrieving specific pre-existing fits

```
fitResult = fits.fitTo(data, checkExisting) # checkExisting is a bool
```

- Probably in end will be a combination of both