GooFit – sounds like RooFit

Michael D. Sokoloff, University of Cincinnati

GooFit is a GPU-friendly parallel function evaluation engine.

- The code is written in C++ and use the Thrust library with CUDA and OpenMP backends to run on nVidia GPUs and x86 servers;
- It is designed to calculate PDFs used in MINUIT fitting;
- The package is used primarily for unbinned amplitude analyses;
- GooFit is also highly performant for other unbinned likelihood fits;
- pybind11 bindings provide essentially all of its existing functionality in pure Python; extensions require writing C++ code.

Some Python code

meant to fit an invariant mass distribution of B_d and B_s candidates

```
import goofit as gf

xvar = gf.Observable("xvar", 5050., 5550.)
xvar.setNumBins(40)
myData = gf.UnbinnedDataSet(xvar)

nBd_fitted   = gf.Variable("nBd_fitted",8,0.1,0.1,500.)
nBs_fitted   = gf.Variable("nBs_fitted",2,0.1,-2.,500.)
nBkgd_fitted = gf.Variable("nBkgd_fitted",30,0.1,1.,1000.)

fittedBdMass = gf.Variable("fittedBdMass",5279.65)
fittedBsMass = gf.Variable("fittedBsMass",5366.88)
resolution   = gf.Variable("resolution",8.3,0.1,5.0,11.0)
expX0        = gf.Variable("expX0",5050.)
expAlpha     = gf.Variable("expAlpha",0.,0.0001,-0.01,0.01)

gaussBd       = gf.GaussianPdf('gaussBd', xvar, fittedBdMass,resolution)
gaussBs       = gf.GaussianPdf('gaussBs', xvar, fittedBsMass,resolution)
expBkgd       = gf.ExpPdf('expPdf',xvar,expAlpha,expX0)

totalFit      = gf.AddPdf('totalFit',[nBd_fitted,nBs_fitted,nBkgd_fitted],[gaussBd,gaussBs,expBkgd])

grid = totalFit.makeGrid()
totalFit.setData(grid)
binCenters = grid.to_matrix().flatten()

totalFit.setData(myData)
fitter = gf.FitManager(totalFit)
fitter.fit()
```

Amplitude Analyses

a common approach in LHCb

`GooFit` has `PDF` classes (really amplitude classes that return complex values) for many types of resonances;

- To build models, we generally start with decay descriptor files found in `AmpGen`. For example, see FitterExample.opt which describes amplitudes contributing to the decay $D^0 \to K^-\pi^-\pi^+\pi^+$.
- We then use scikit-hep/decaylanguage to write the `GooFit` code that implements the model

The user needs to write additional C++ code to account for backgrounds and efficiencies and to account for multiple data sets, etc. In general, Python bindings can be added so the fits can be executed from Jupyter notebooks or from Python scripts.