

# Modernizing GooFIT: A Case Study

---

Henry Schreiner

July 12, 2017

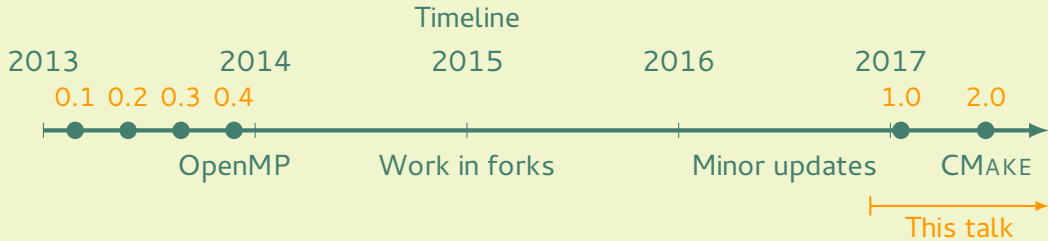



[goofit.github.io/pearc17.pdf](https://goofit.github.io/pearc17.pdf)

**PEARC**17

UNIVERSITY OF  
**Cincinnati**

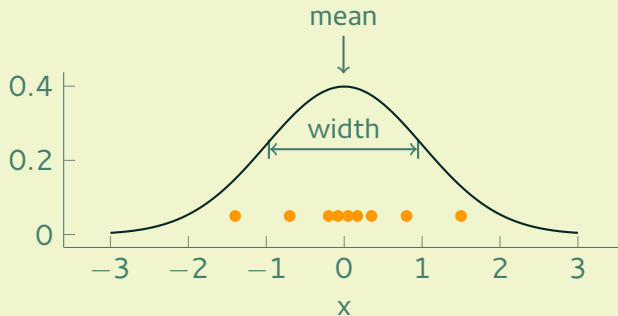
# History of GooFIT



/GooFit/GooFit

- Developed by Rolf Andreassen in 2013
- A lot of duplicated work in 2014-2016
- We will cover the 2016-2017 modernization

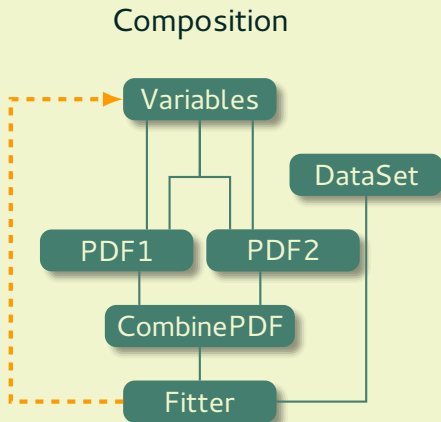
# GooFIT: Probability Density Function (PDF) Fitting



## Design

- Resembles the popular RooFIT package in ROOT
- Built with CUDA/OpenMP using THRUST
- Includes 30+ High Energy Physics (HEP) PDFs and examples

# Features of a Fit



## Composition

- Changed often
- 100+ Variables possible

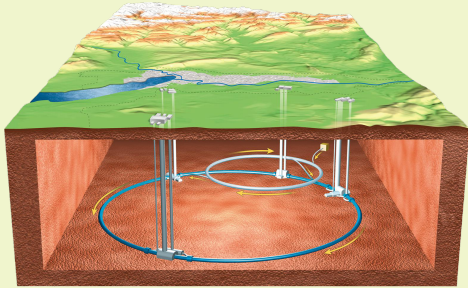
## PDFs

- Many provided
- Users may add more

## Backend

- Managed by core team

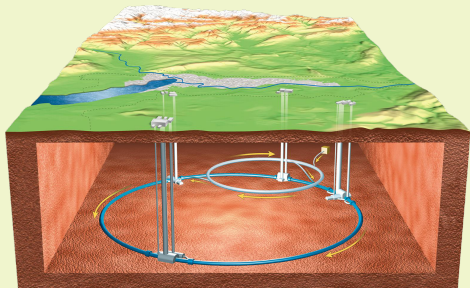
# Why GooFIT?



## Charm Physics at CERN's LHCb experiment

- 1,000,000+ events in dataset
- 5 or more independent variables common
- 20+ PDFs with complex coefficients
- Some analyses run 1,000+ fits

# Why GooFIT?

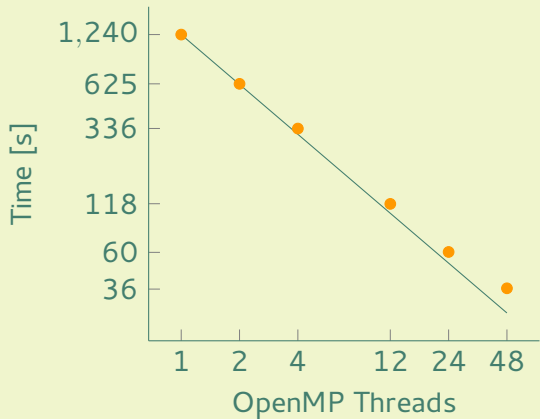


## Charm Physics at CERN's LHCb experiment

- 1,000,000+ events in dataset
- 5 or more independent variables common
- 20+ PDFs with complex coefficients
- Some analyses run 1,000+ fits

GooFIT transforms fitting with 950x speedup over single-core RooFIT

# Why GooFIT?

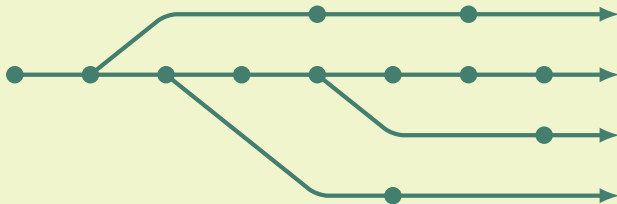


## GooFIT is fast

- 142576 events
- Unbinned fit
- 24 physical Xeon cores

K40	96.6 seconds
P100	23.5 seconds

# GooFIT 0.4: A State of Disrepair



## Research level code

- Written for CUDA 4.0 and Compute Architecture 2.0
- Hardcoded paths in Makefiles
- Little file organization
- Forked 10+ times, new features not in master
- Cludges: fake `nvcc`, globbing, fake `ROOT`, ...



# Build system updates

## Iterative Approach

1. Makefile cleanup and consolidation
2. Organization of file structure
  - ModernizeGooFit.py script
3. Adding CMAKE
  - Coexisted with makefiles for a while

```
#include "CompositePdf.hh"  
abortWithCudaPrintFlush(__FILE__, __LINE__, "Failed");
```

ModernizeGooFit.py



Uses regular expressions and Plumbum

```
#include <goofit/PDFs/combine/CompositePdf.h>  
GooFit::abort(__FILE__, __LINE__, "Failed");
```

# CMAKE


## CMAKE CUDA support

- Require CMAKE 3.4+
- Backported FindCUDA from CMAKE 3.7
- Keyword vs. standard targets
- CUDA in 3.8 massively improved

## Features

- IDE support (XCODE, QTcreator)
- Library discovery / configuration
- Multiple compiler support
- Integration with other tools
- Download datafiles from GitHub releases

## Git Submodules

- Libraries as submodules
- Automatic checkout by CMAKE build
- Separate CMAKE folder (/CLIUtils/cmake) and external libraries

# Automation and Testing

## Travis CI

- Verify OpenMP build
- Verify PRs
- Upload coverage reports
- Upload documentation

## Challenges

C++11 • ROOT • Docs

## Tests

- Run all examples with script
- Slowly added verification
- Unit-tests added with GOOGLETEST (evaluating CATCH)

## (NVIDIA-)Docker

- Added images with ROOT+Utils
- From scratch install

# Modernization



## C++11

- Limited to CUDA 7.0+
- Reduced # of lines / simplified
- Used CLANG-TIDY to convert (CMAKE 3.6+ integration)

## Major updates

nullptr • foreach • override  
Variadic templates • Initializer lists

## Cleanup

- Readability: CLANG-FORMAT
- Moved all code to namespace
- Compile-time logging choice  
 /fmtlib/fmt
- Smart color output  
 /agaunoyal/rang
- Removed custom classes and iterators (complex, etc)

# Command line parsing

```
./MyAnalysis generate_toy  
    --params=file.ini  
    --release_K892_mass  
    --A12=0.3  
    --plot
```

## Recurring theme

- Analyses require 40+ options and multiple procedures
- Found a lot of duplicated code for argument parsing
- Many bugs related to parsing (usually segfaults)
- Needed powerful solution, with direct access to values

## /CLIUtils/CLI11

- No dependencies
- Compiles to single header file

### Features

- Nested subcommands
- Configuration files
- 100% test coverage
- CI tests on macOS/Linux/Windows
- + GooFIT's features

## Use in GooFIT

- Testbed for new build features

### GooFit::Application

- Auto logging
- Optimization warnings
- GPU switches
- MPI support
- Completely optional

# Improvements

## Expanded physics tools


- Three body time-dependent amplitude analyses
- Four body time-integrated and time-dependent amplitude analyses
- Toy Monte Carlo generation using MCBOOSTER

## Caching: /bryancatanzaro/generics

- Support for LDG caching
- LDG generalized form
- Performance boost for mid-age cards

## MPI

- Available for Application
- Supports multiple GPUs

/MultithreadCorner/MCBooster is deprecated  
in favor of /MultithreadCorner/Hydra

# MINUIT 2

## MINUIT in HEP

- MINUIT is a standard HEP parameter search algorithm (CPU)
- MINUIT 1 and 2 available in ROOT
- Old copy of MINUIT 2 was available stand-alone

## Status in GooFit 0.4

- Internal MINUIT1 copy
- Required manual upkeep

## /GooFit/Minuit2

- Newly forked from ROOT 6.08
- CMAKE build, no other changes
- Already being used outside GooFit



# PYTHON Bindings (GooFIT 2.1 feature)

Pre-release syntax

```
from goofit import *
xvar = Variable("xvar", 0, 10)
xdata = UnbinnedDataSet(xvar)
xdata.from_numpy(np.random.exponential(size=100000))

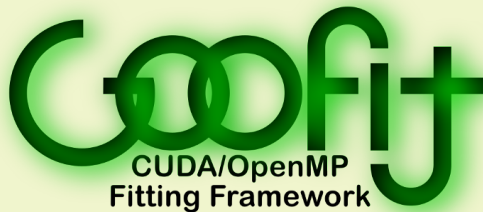
alpha = Variable("alpha", -2, 0.1, -10, 10)
exppdf = ExpPdf("exppdf", xvar, alpha)
exppdf.fitTo(data)
```


## /pybind/pybind11

- Only uses advanced C++11
- Full working example in GooFIT 2.0
- Expanding for GooFIT 2.1
- pip install with SCIKIT-BUILD

## Current Challenges

- CUDA vs. C++
- Pythonic syntax
- Example conversion



 /GooFit/GooFit

GoOFIT 2.0: Released

- Source code and docs on GitHub

GoOFIT 2.1: Coming soon

- Drastically expanded Python bindings by Himadri Pandey

Plans

- HYDRA integration
- Use in  $D^0 \rightarrow K^- \pi^- \pi^+ \pi^+$  amplitude analysis
- GoOFIT 2torial under development at [henryiii.gitbook.io/goofit](https://henryiii.gitbook.io/goofit)

# Build it yourself

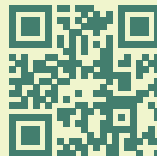
```
docker run -it alpine  
apk add --no-cache make cmake g++ git  
git clone --branch=stable https://github.com/GooFit/GooFit.git  
cd GooFit  
make
```

# Acknowledgments

- A. Augusto Alves Jr.
- Christoph Hasse
- Bradley Hittle
- Zachary Huard
- Brian Maddock
- Himadri Pandey
- Michael Sokoloff
- Karen Tomko

Development of GOOFIT 2.x is supported under NSF grant PHY-1414736. GOOFIT was originally developed under PHY-1005530. Any opinions, findings, and conclusions or recommendations expressed in this presentation are those of the developers and do not necessarily reflect the views of the National Science Foundation.

CERN image from project-hl-lhc-industry.web.cern.ch.



[goofit.github.io](https://goofit.github.io)

# Questions?

[henry.schreiner@uc.edu](mailto:henry.schreiner@uc.edu)