GooFit 2.0

Christoph Hasse\textsuperscript{2}  Bradley Hittle\textsuperscript{3}  Henry Schreiner\textsuperscript{1}  Himadri Pandey\textsuperscript{1}  Michael Sokoloff\textsuperscript{1}  Karen Tomko\textsuperscript{3}

August 22, 2017

\textsuperscript{1}University of Cincinnati  \textsuperscript{2}Technical University of Dortmund  \textsuperscript{3}Ohio Supercomputer Center
/GooFit/GooFit

- Resembles the RooFit package
- Built for CUDA GPUs
- Also supports OpenMP on CPUs

LGPLv3

Classic features

- PDF composition
- PDFs for amplitude analyses
- Binned and unbinned fits
Features of a fit

- Variables
- DataSet
- PDF1
- PDF2
- CombinePDF
- Fitter

Powerful composition of PDFs

- Long list of built-in PDFs
- Combination PDFs using addition, multiplication, convolution, composition, etc.
- 100+ Variables possible
- Advanced users can create new PDFs
- Expert users can extend GooFit core
Common GPUs

Classic CPUs

2.7 MHz × 2 cores × 16 FLOP/cyc = 86.4 MFLOP/s
2.4 MHz × 24 cores × 16 FLOP/cyc = 921.6 MFLOP/s

NVIDIA GPUs

- Programing language: CUDA
- Massively parallel identical operations (SIMD)
- Separate memory model (coprocessor)

<table>
<thead>
<tr>
<th>Name</th>
<th>Stream processors</th>
<th>Clock</th>
<th>TFLOPS SP</th>
<th>TFLOPS DP</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Gamer</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GTX 1050 Ti</td>
<td>768</td>
<td>1290 Mhz</td>
<td>1.98</td>
<td>0.062</td>
<td>$150</td>
</tr>
<tr>
<td>GTX 1080 Ti</td>
<td>3,584</td>
<td>1596 Mhz</td>
<td>11.3</td>
<td>0.33</td>
<td>$850</td>
</tr>
<tr>
<td><strong>Server</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Tesla K40</td>
<td>2,880</td>
<td>745 Mhz</td>
<td>4.29</td>
<td>1.43</td>
<td>$3,000</td>
</tr>
<tr>
<td>Tesla P100</td>
<td>3,584</td>
<td>1329 Mhz</td>
<td>9.3</td>
<td>4.7</td>
<td>$10,000</td>
</tr>
</tbody>
</table>
Why use GooFit?

### $\pi \pi \pi^0$, 16 time-dependent amplitudes

- Original RooFit code: 19,489 s single core
- 2 Cores: Core 2 Duo 1,159 s
- GPU: GeForce GTX 1050 Ti 86.4 s
- GPU: Tesla K40 64.0 s
- MPI: Tesla K40 $\times 2$ 39.3 s
- GPU: Tesla P100 20.3 s

### ZachFit: $M(D^{*-}) - M(D^0)$

- 142,576 events in unbinned fit
- 2 Cores: Core 2 Duo 738 s
- GPU: GeForce GTX 1050 Ti 60.3 s
- GPU: Tesla K40 96.6 s
- MPI: Tesla K40 $\times 2$ 54.3 s
- GPU: Tesla P100 23.5 s

[Phys.Rev. D93 (2016) no.11, 112014]
[CHEP 2013]
Why use GooFit? Reduce time to insight!

\[ \pi \pi \pi^0, \ 16 \text{ time-dependent amplitudes} \]
- Original RooFit code: 19,489 s single core
- 2 Cores:
  - Core 2 Duo: 1,159 s
  - GPU: GeForce GTX 1050 Ti: 86.4 s
  - GPU: Tesla K40: 64.0 s
  - MPI: Tesla K40 \times 2: 39.3 s
  - GPU: Tesla P100: 20.3 s

\[ \text{ZachFit: } M(D^{*+}) - M(D^0) \]
- 142,576 events in unbinned fit
- 2 Cores:
  - Core 2 Duo: 738 s
  - GPU: GeForce GTX 1050 Ti: 60.3 s
  - GPU: Tesla K40: 96.6 s
  - MPI: Tesla K40 \times 2: 54.3 s
  - GPU: Tesla P100: 23.5 s
New build features

CMake enabled features

- Supports IDEs like XCode and QtCreator
- Support for macOS, CPP backend
- Datafiles auto-download (New datasets!)
- Auto-library download and discovery
- Helpers at `CLIUtils/cmake`

Other build features

- Initial unit tests added
- Travis CI builds every push
- Travis powered documentation system
- Docker images available
- Example setup on new systems
It's easy to get started

docker run -it ubuntu
apt-get update

apt-get install -y git cmake make g++
git clone --branch=stable https://github.com/GooFit/GooFit.git
cd GooFit
make

Simple installation

- More systems available on

- Or use Docker images: goofit/goofit-omp and goofit/goofit-cuda
New design features

- C++11, extensive code cleanup
- Color logging, optimization warnings
- Simultaneous fit to disjoint datasets
- MPI support (GPUs or nodes)
- Optimizations for newer NVIDIA cards

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

CLIUtils/CLI11  BSD
- Command line parser using C++11
- Help generation, Subcommands, validation, config files, 100% test coverage
- Designed to support toolkit customization

/GooFit/Minuit2  LGPLv2.1
- Extracted from ROOT 6.08, no changes
- Full CMake build system

GooFit 2.0  H. Schreiner  August 22, 2017  7/14  ACAT 2017
New physics features

Three body time-dependent amplitude analyses (TDAA)

- Mixing in $D^0 \rightarrow \pi^+\pi^-\pi^0$ TDAA (BaBar)
  [Phys.Rev. D93 (2016) no.11, 112014]
- Mixing and CP violation search in $D^0 \rightarrow K_S^0\pi^-\pi^+$, TDAA (LHCb)

Four body time-integrated and time-dependent amplitude analyses

- Mixing parameters in $D^0 \rightarrow K^+\pi^-\pi^+\pi^-$ TDAA (LHCb)
  [CHEP 2016]

Toy Monte Carlo generation using /MultithreadCorner/MCBooster

- MIPWA in GooFit, such as $D^+ \rightarrow h^- h'^+ h'^+$ (LHCb)
  [CHEP 2016]
**GooFit 2.1: Python bindings**

- pip install scikit-build cmake
- pip install -v goofit

**Downloads and installs all of GooFit!**

**Prototype in GooFit 2.0**
- Intended to be interface to composition
- All backends supported
- Only one PDF in GooFit 2.0
- Using CMake and PyBind11

**Working on features**
- All PDFs added, including composition PDFs
- Install development version with `pip`
- Pythonization of objects ongoing
- Converting/adding examples/documentation

---

**Himadri Pandey**

- 2.1

**GooFit 2.0 H. Schreiner August 22, 2017 9/14**
```python
from goofit import *
import numpy as np

xvar = Variable("xvar", -10, 10)
xdata = UnbinnedDataSet(xvar)
npdata = np.random.normal(1, 2.5, 100000)
xdata.from_numpy([npdata], filter=True)

mean = Variable("mean", 0, -10, 10)
sigma = Variable("sigma", 1, 0, 5)
gauss = GaussPdf("gauss", xvar, mean, sigma)

exppdf.fitTo(data)

grid, values = gauss.evaluatePdf(xval)
```
from goofit import *
import numpy as np

xvar = Variable("xvar", -10, 10)
xdata = UnbinnedDataSet(xvar)
npdata = np.random.normal(1, 2.5, 100000)
xdata.from_numpy([npdata], filter=True)

mean = Variable("mean", 0, -10, 10)
sigma = Variable("sigma", 1, 0, 5)
gauss = GaussPdf("gauss", xvar, mean, sigma)

exppdf.fitTo(data)

grid, values = gauss.evaluatePdf(xval)

Data for red line PDF plot
GooFit future

New PDF indexing system
- Designed by Bradley Hittle at OSC
- Simpler PDF authoring
- We can alter core later

Other plans
- /multithreadcorner/Hydra (optional inclusion)
- GooFit 2torial: henryiii.gitbooks.io/GooFit
- Drop support for CMake < 3.8
## Summary

### GooFit 2.0 Available now
- Much easier to start using
  - Automated CMake builds
  - More supported platforms including macOS
  - Easy backend selection: OpenMP, CUDA, CPP, partial TBB
- More examples and PDFs
- Added example datasets
- Added MPI and performance enhancements

### GooFit 2.1 Coming soon
- Python bindings for composition
- Development releases on PyPI now

### GooFit 2.x series
- Simpler PDF authoring
- Extended documentation
- Hydra inclusion
- CMake 3.8+ requirement

---

GooFit 2.0  
H. Schreiner  
August 22, 2017  
12/14  
ACAT 2017
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.
Questions?

henry.schreiner@uc.edu

Give our projects a try and a star on 🌟!

/GooFit/GooFit
/GooFit/Minuit2
/CLIUtils/CLI11
/CLIUtils/cmake
/multithreadcorner/Hydra
Running Timing Examples

General notes
- You can pick cards with the prefix: CUDA_VISIBLE_DEVICES=0,1

$\pi^+\pi^-\pi^0$
- time ./pipipi0DPFit canonical dataFiles/cocktail_pp_0.txt --blindSeed=0
- time mpiexec -np 2 ./pipipi0DPFit canonical dataFiles/cocktail_pp_0.txt --blindSeed=0

ZachFit
- time ./zachFit 0 1
- time mpiexec -np 2 ./zachFit 0 1
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
CLI11

/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

BSD

Use in GooFit

- Testbed for new build features

GooFit::Application

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