GooStats,

a GPU-accelerated package for multi-variate spectral fitting analysis

Xuefeng Ding[1,2]

1. Gran Sasso Science Institute, L'Aquila, Italy
2. INFN Laboratori Nazionali del Gran Sasso, Assergi, Italy

10/05/2018, Wuhan, China (Remote)

What is GooStats?

GitHub, Inc. [US] | https://github.com/GooStats/GooStats

README.md

GooStats

GooStats is an open source statistical analysis framework using GPUs.

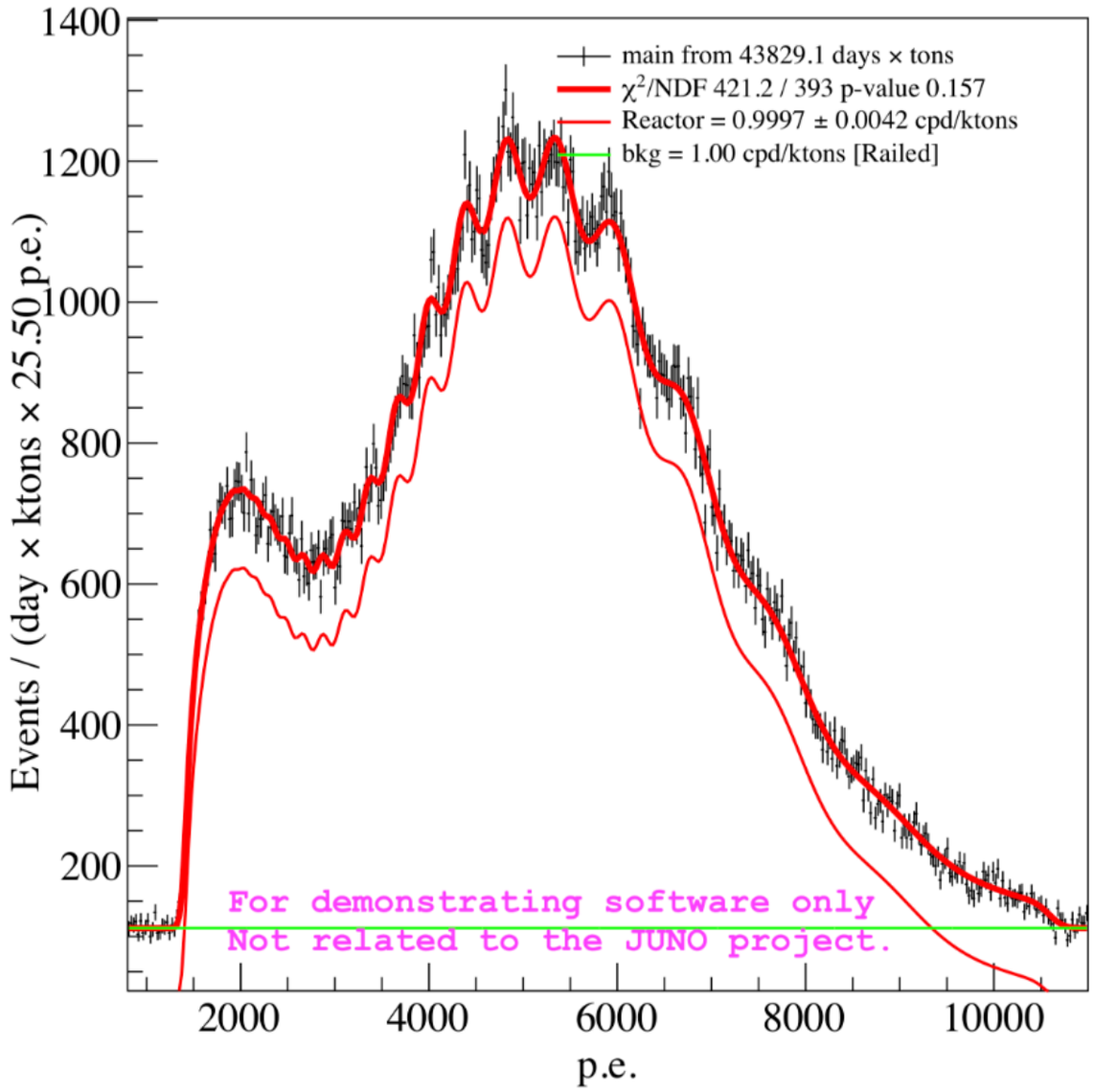
- It provide handful tools to configura input parametrs, datasets, spectrums, pdfs etc.
- It also provide flexible text/plot/TTree output class.
- The backend is GooFit on nVidia GPU, and the code is tuned and validated on GPU.

With a few lines of code, you will be able to use GooFit as low level engine and produce nice plots

- with a few more lines, you will be able to do joint analysis of multiple datasets.
- Look at Modules/naive-Reactor as an example.
- Here are some screen shots of the text/TTree output and plot produced, as well as user code.

Example output

pdf-format output




```
root [1] .ls
TFile**         test_tree.root
 TFile*         test_tree.root
  KEY: TTree    fit_result;1    Fit result of GooStats
root [2] fit_result->Show(0)
======> EVENT:0
 default.NReactor = 3.02987
 default.NReactor_err = 0.0159146
 default.Nbkg     = 1.02405
 default.Nbkg_err = 0.0162459
 default.U235     = 0.5
 default.U235_err = 0
 default.U238     = 0.2
 default.U238_err = 0
 default.Pu239    = 0.1
 default.Pu239_err = 0
 default.U241     = 0.2
 default.U241_err = 0
 default.LY       = 1300
 default.LY_err   = 0
 default.qc1      = 2.78788
 default.qc1_err = 0
 default.qc2      = -0.528003
 default.qc2_err = 0
 default.v1       = 0.3
 default.v1_err   = 0
 default.vT       = 5
 default.vT_err   = 0
 default.Reactor_dEvis = 1078.28
 default.Reactor_dEvis_err = 13.2291
 chi2             = 390.448
 NDF              = 397
 likelihood       = 1883.96
```

TTree-format output

Example output

```
FIT PARAMETERS
 NReactor = 3.030 #pm 0.016 cpd/ktons
 Nbkg = 1.024 #pm 0.016 cpd/ktons
 ^{235}U = 0.5  [Fixed]
 ^{238}U = 0.2  [Fixed]
 ^{239}Pu = 0.1  [Fixed]
 ^{241}U = 0.2  [Fixed]
 LY = 1300  [Fixed]
 ^{1}qc = 2.78788  [Fixed]
 ^{2}qc = -0.528003  [Fixed]
 ^{1}v = 0.3  [Fixed]
 vT = 5  [Fixed]
 Reactor dEvis = 1078.3 #pm 13.2

 chi^2                              = 390.4
 chi^2/N-DOF                        = 0.9835
 p-value                            = 0.583
 Minimized Likelihood Value         = 1883.96
```

text output on the screen

```
Checking GPU [0]/[2]
Running on [Tesla K20m] with compute capability of 3.5
Warning: ParSyncManager not set, default strategy are used.
Warning: RawSpectrumProvider not set, default strategy are used.
Loading from <IBD.cfg>
*********Dump options parsed*********************************
EvisVariable => <p.e.>
Evis_max => number: [11000]
Evis_min => number: [800]
Evis_nbins => number: [400]
Huber_Pu239_0 => number: [1.162]
Huber_Pu239_1 => number: [-0.392]
Huber_Pu239_2 => number: [-0.079]
Huber_Pu241_0 => number: [0.852]
Huber_Pu241_1 => number: [-0.126]
Huber_Pu241_2 => number: [-0.1037]
Huber_U235_0 => number: [0.904]
Huber_U235_1 => number: [-0.184]
Huber_U235_2 => number: [-0.0878]
Huber_U238_0 => number: [0.976]
Huber_U238_1 => number: [-0.162]
Huber_U238_2 => number: [-0.079]
LY_err => <0>
LY_init => number: [1300]
LY_max => number: [2000]
LY_min => number: [800]
NHatomPerkton => number: [7.75e+31]
NL_b_err => <0>
```

Log message

Example code

```
#include "ReactorAnalysisManager.h"
#include "InputManager.h"
#include "ReactorInputBuilder.h"
#include "ReactorSpectrumBuilder.h"
#include "OutputManager.h"
#include "SimpleOutputBuilder.h"
#include "SimplePlotManager.h"

int main (int argc, char** argv) {
  AnalysisManager *ana = new ReactorAnalysisManager();
  InputManager *inputManager = new InputManager(argc,argv);
  InputBuilder *builder = new ReactorInputBuilder();
  builder->installSpectrumBuilder(new ReactorSpectrumBuilder());
  inputManager->setInputBuilder(builder);
  ana->setInputManager(inputManager);
  OutputManager *outManager = new OutputManager();
  outManager->setOutputBuilder(new SimpleOutputBuilder());
  outManager->setPlotManager(new SimplePlotManager());
  ana->setOutputManager(outManager);

  ana->init();
  ana->run();
  ana->finish();

  return 0;
}
```

Object-orieneted
Fully written in C++11

Installation guide

GitHub, Inc. [US] | https://github.com/GooStats/GooStats/blob/master/docs/INSTALL.md

Installation guide for GooStats

1. First, you should install the `GooFit` shipped with this package. I have done some modification and it's not compatible with the original GooFit project

```
# suppose you create a folder GooStats-release
cd GooStats-release
git clone git@github.com:DingXuefeng/GooStats.git
mkdir build_GooFit
cd build_GooFit
cmake ../GooStats/GooFit -DCMAKE_INSTALL_PREFIX=../GooFit-install
make -j4
make -j4 install
export GOOFIT_DIR=$(readlink -f ../GooFit-install)
# you can put this sentence to your .bashrc
echo "export GOOFIT_DIR=$(readlink -f ../GooFit-install)" >> ~/.bashrc
```

Tips:

- for sure remember to compile it on a machine with GPU. if you work on a cluster, use qsub -I
- I have turned off OpenMP in GooFit. GooFit can run on GPU, OpenMP and MPI mode. However GooStats is only optimized for the GPU mode. Sometimes you work on a cluster with one GPU and only one CPU, if you turn on OpenMP, the code will be super slow. I might solve it in the future, currently I just turn off the OpenMP in GooFit.

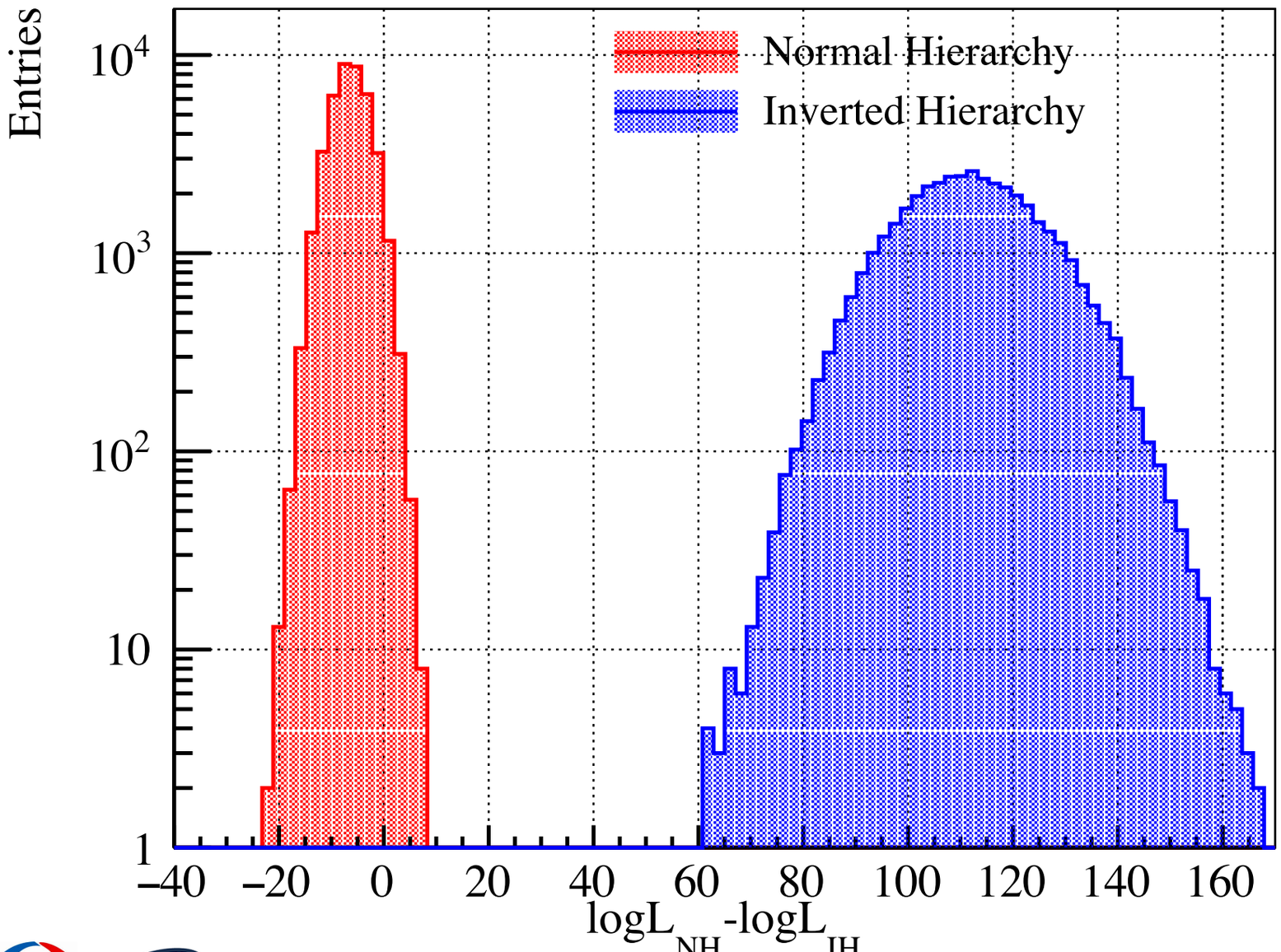
en, try to compile the test project

Response function

- Based on my thesis on Borexino analysis
  - **Use charge (or Evis) instead of non-linearity corrected energy to be able to use physics parameterization -> important for solar Be7 nu fitting (conclusion of my thesis)**
  - Generalized gamma instead of Gaussian
  - Resolution dependence with physics parameterization
    - $v1 = (1+v1) * \mu + \sigma_T^2 \mu^2$
    - v1: PMT single p.e. charge resolution
    - σ_T: residual non-uniformity

Performance and result

- 40000 fit x4 (NH-NH, NH-IH, IH-NH, IH-IH), on K20m GPU, **~10 hours**



- **Likelihood based fit.**
- 0.1% residual non-uniformity assumed
- No LY-related systematics
- No backgrounds
- 6 year x 20 ton, 36 GW, 52.5 km
- For 99.9975% cases the rejection power would be stronger than 1/40000.

Future plan

- Add background (make it open source, like Barcall's website? or use zentoro?)
- worse non-uniformity assumed, say 1%?
- Include systematics from LY/non-linearity? quite strait-forward.

Conclusion

- GooStats is a GPU fitter specifically for particle multi-variate spectral-fitting analysis. It's open source and welcome for co-operation.

- A hard-ware accelerated mass-hierarchy fitter based on GooStats is developed. throughout: 40000 x 4 fit per 10 hour, that is **0.23 s per fit**

- Using likelihood rather than chi^2, we got a different view of MH determination.

Thanks for you attention