RooUnfold

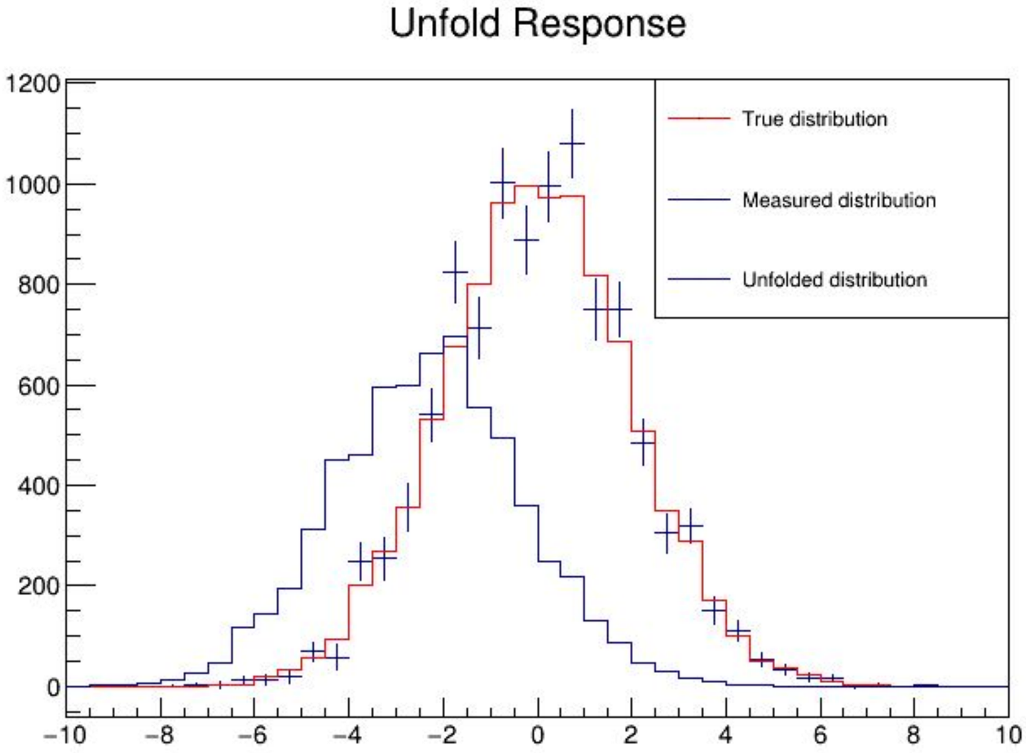
The ROOT Unfolding Framework

RooUnfold: Efficient deconvolution using state of the art algorithms

Archit Agrawal
archit18221@iiitd.ac.in

What is Unfolding?

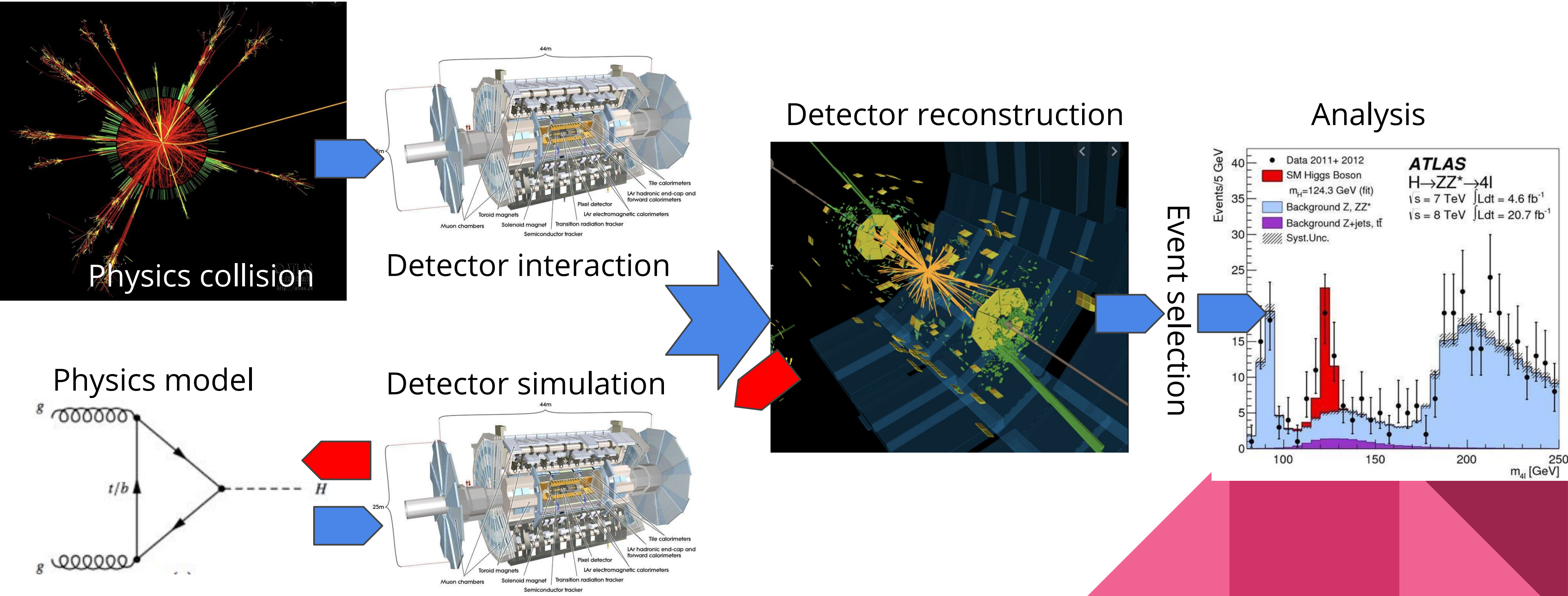
Statistical method to obtain 'truth' from both simulated and measured data



Unfolding at CERN

🟦 Standard Method

🟥 Unfolding Method



Physics collision

Detector interaction

Detector reconstruction

Event selection

Analysis

Physics model

Detector simulation

What is RooUnfold ?

RooUnfold is a framework for unfolding within the ROOT environment, implementing a number of specific algorithms while maintaining a consistent view of each method to the use

RooUnfold

The ROOT Unfolding Framework

ATLAS EXPERIMENT

Users of RooUnfold

CMS

180+ citations

HAWC
High Altitude Water Cherenkov
Gamma-Ray Observatory

CALIFORNIA INSTITUTE OF TECHNOLOGY 1891

ICECUBE
SOUTH POLE NEUTRINO OBSERVATORY

S••INE

Neutrino Factory
Muon Collider

ANTARES

ALICE

Daya Bay
θ_{13}

Fermilab

JAXA

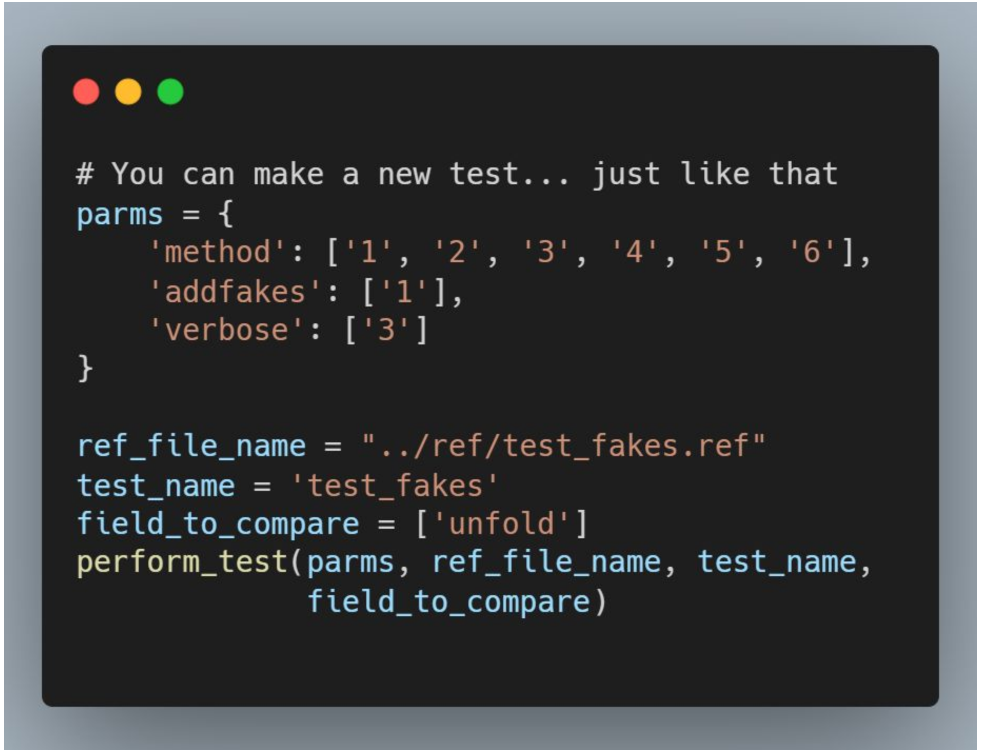
MINERvA

LHCb

My Work: GSoC ‘21

- ➔ Developing framework to define new tests with ease
- ➔ Develop new tests
- ➔ Set up changes to compute test code coverage
- ➔ Design and develop RooUnfoldAlgorithm

Development of new testing framework



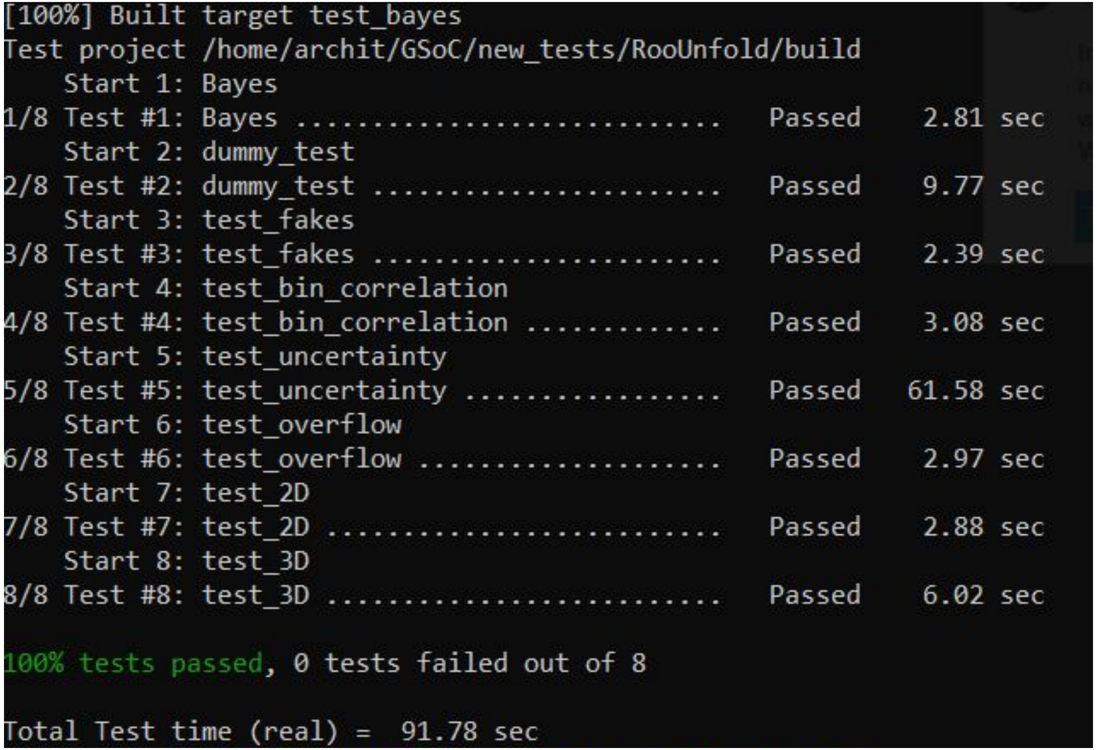
```python
# You can make a new test... just like that
parms = {
    'method': ['1', '2', '3', '4', '5', '6'],
    'addfakes': ['1'],
    'verbose': ['3']
}

ref_file_name = "../ref/test_fakes.ref"
test_name = 'test_fakes'
field_to_compare = ['unfold']
perform_test(parms, ref_file_name, test_name,
             field_to_compare)
```

- Framework to develop new tests with minimal code

- Just define what you need to test as JSON

- Reducing 100s of line of code to ~10

Revamped Unit Tests



```
[100%] Built target test_bayes
Test project /home/archit/GSoC/new_tests/RooUnfold/build
    Start 1: Bayes
1/8 Test #1: Bayes ............................   Passed    2.81 sec
    Start 2: dummy_test
2/8 Test #2: dummy_test .......................   Passed    9.77 sec
    Start 3: test_fakes
3/8 Test #3: test_fakes .......................   Passed    2.39 sec
    Start 4: test_bin_correlation
4/8 Test #4: test_bin_correlation .............   Passed    3.08 sec
    Start 5: test_uncertainty
5/8 Test #5: test_uncertainty .................   Passed   61.58 sec
    Start 6: test_overflow
6/8 Test #6: test_overflow ....................   Passed    2.97 sec
    Start 7: test_2D
7/8 Test #7: test_2D ..........................   Passed    2.88 sec
    Start 8: test_3D
8/8 Test #8: test_3D ..........................   Passed    6.02 sec

100% tests passed, 0 tests failed out of 8

Total Test time (real) =  91.78 sec
```

- Remove old tests
- Develop new unit tests with the framework developed
- Strategically choosing tests to maximise code coverage
- <10% to 77% Code Coverage

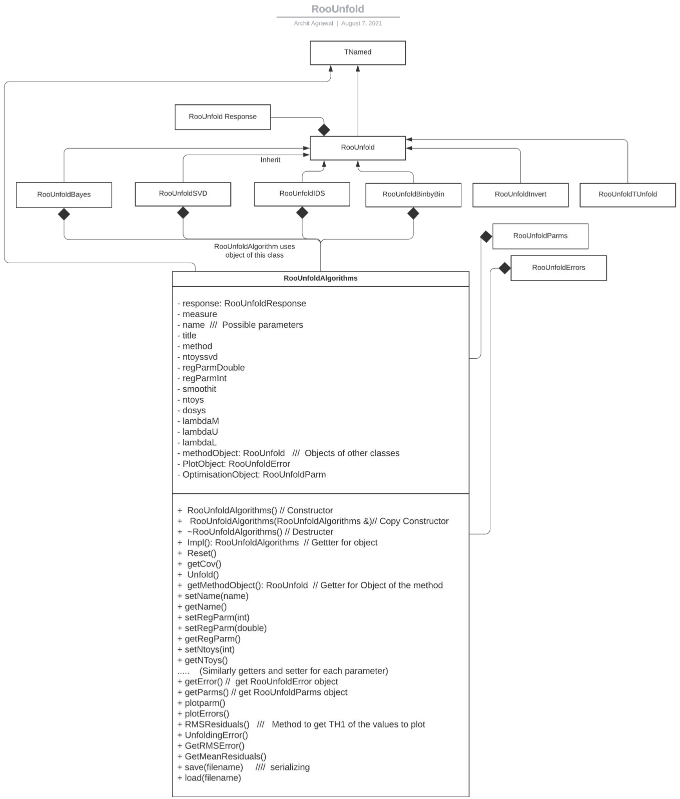
Cleaning up of dead code

- Clean up dead and unnecessary code

- Refactoring redundant code

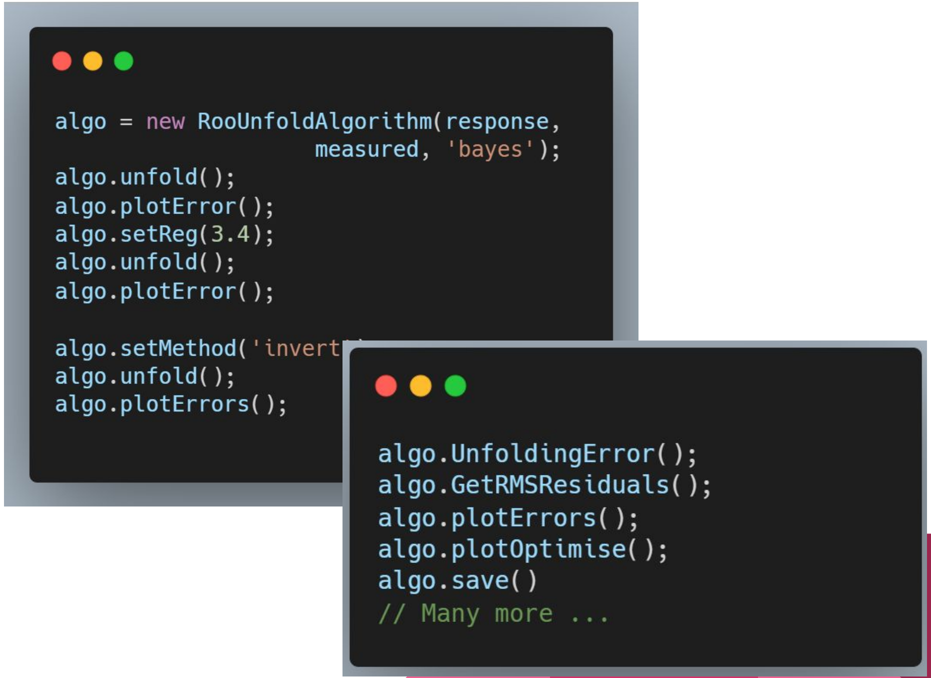
- Ensuring backwards compatibility due to changes in release

Design and Develop new Class (RooUnfoldAlgorithm)



- response: RooUnfoldResponse
- measure
- name /// Possible parameters
- title
- method
- ntoyssvd
- regParmDouble
- regParmInt
- smoothit
- ntoys
- dosys
- lambdaM
- lambdaU
- lambdaL
- methodObject: RooUnfold /// Objects of other classes
- PlotObject: RooUnfoldError
- OptimisationObject: RooUnfoldParm

```
+  RooUnfoldAlgorithms() // Constructor
+   RooUnfoldAlgorithms(RooUnfoldAlgorithms &)// Copy Constructor
+  ~RooUnfoldAlgorithms() // Destructer
+  Impl(): RooUnfoldAlgorithms  // Gettter for object
+  Reset()
+  getCov()
+  Unfold()
+  getMethodObject(): RooUnfold  // Getter for Object of the method
+ setName(name)
+ getName()
+ setRegParm(int)
+ setRegParm(double)
+ getRegParm()
+ setNtoys(int)
+ getNToys()
.....   (Similarly getters and setter for each parameter)
+ getError() //  get RooUnfoldError object
+ getParms() // get RooUnfoldParms object
+ plotparm()
+ plotErrors()
+ RMSResiduals()   ///   Method to get TH1 of the values to plot
+ UnfoldingError()
+ GetRMSError()
+ GetMeanResiduals()
+ save(filename)    ////  serializing
+ load(filename)
```

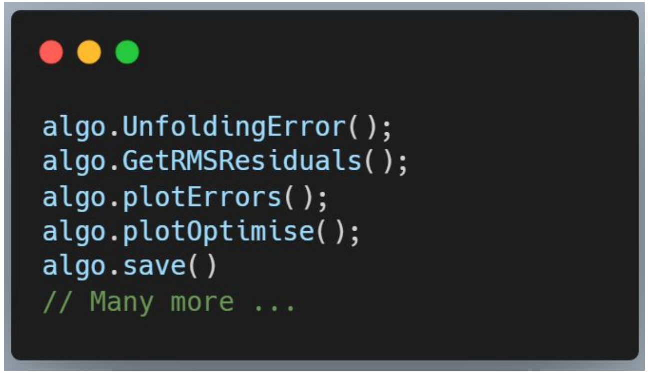


```
algo = new RooUnfoldAlgorithm(response,
                    measured, 'bayes');
algo.unfold();
algo.plotError();
algo.setReg(3.4);
algo.unfold();
algo.plotError();

algo.setMethod('invert
algo.unfold();
algo.plotErrors();
```

```
algo.UnfoldingError();
algo.GetRMSResiduals();
algo.plotErrors();
algo.plotOptimise();
algo.save()
// Many more ...
```

What is RooUnfoldAlgorithm?

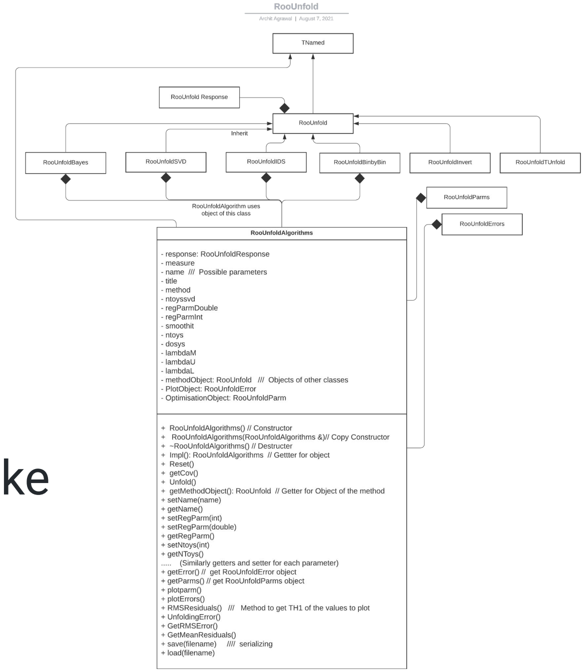
- A top-class implementation
- Provide a standardised interface to all algorithms
- Serialize the class object to a ROOT workspace



```
algo.UnfoldingError();
algo.GetRMSResiduals();
algo.plotErrors();
algo.plotOptimise();
algo.save()
// Many more ...
```

- Incorporating features like plotting errors and optimisation

Provide important analytics



RooUnfold

Archit Agrawal | August 7, 2021

TNamed

RooUnfold Response

RooUnfold

Inherit

RooUnfoldBayes

RooUnfoldSVD

RooUnfoldIDS

RooUnfoldBinbyBin

RooUnfoldInvert

RooUnfoldTUnfold

RooUnfoldParms

RooUnfoldErrors

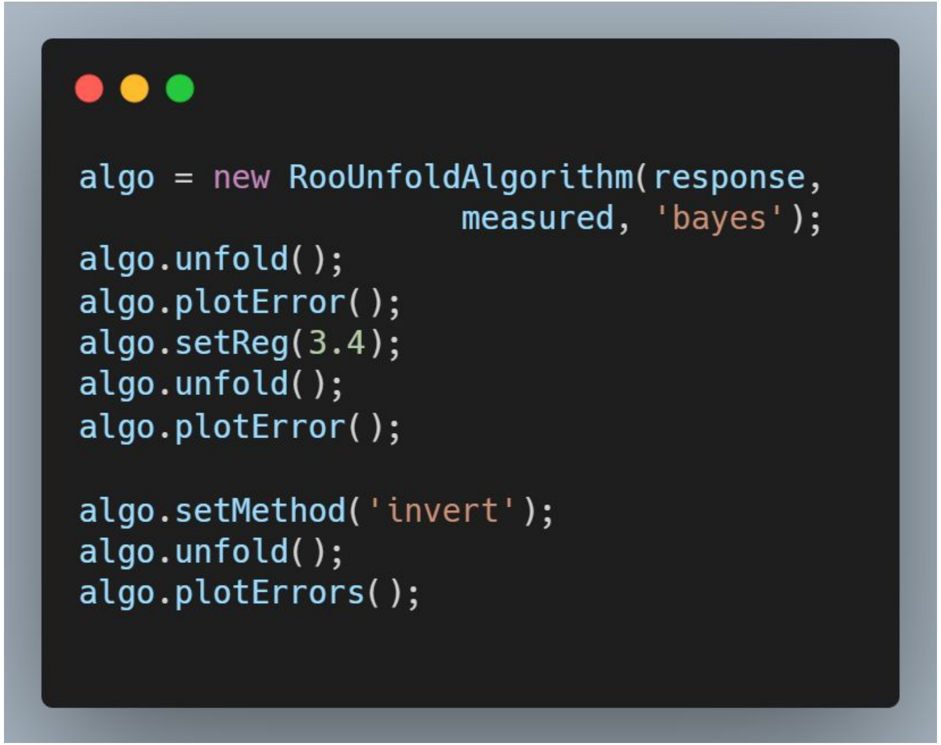
RooUnfoldAlgorithm uses object of this class

RooUnfoldAlgorithms

```
- response: RooUnfoldResponse
- measure
- name  ///  Possible parameters
- title
- method
- ntoyssvd
- regParmDouble
- regParmInt
- smoothit
- ntoys
- dosys
- lambdaM
- lambdaU
- lambdaL
- methodObject: RooUnfold   ///  Objects of other classes
- PlotObject: RooUnfoldError
- OptimisationObject: RooUnfoldParm
```

```
+  RooUnfoldAlgorithms() // Constructor
+   RooUnfoldAlgorithms(RooUnfoldAlgorithms &)// Copy Constructor
+  ~RooUnfoldAlgorithms() // Destructer
+  Impl(): RooUnfoldAlgorithms  // Gettter for object
+  Reset()
+  getCov()
+  Unfold()
+  getMethodObject(): RooUnfold  // Getter for Object of the method
+ setName(name)
+ getName()
+ setRegParm(int)
+ setRegParm(double)
+ getRegParm()
+ setNtoys(int)
+ getNToys()
.....    (Similarly getters and setter for each parameter)
+ getError() //  get RooUnfoldError object
+ getParms() // get RooUnfoldParms object
+ plotparm()
+ plotErrors()
+ RMSResiduals()   ///   Method to get TH1 of the values to plot
+ UnfoldingError()
+ GetRMSError()
+ GetMeanResiduals()
+ save(filename)     ////  serializing
+ load(filename)
```

Using RooUnfoldAlgorithm



```
algo = new RooUnfoldAlgorithm(response,
                    measured, 'bayes');
algo.unfold();
algo.plotError();
algo.setReg(3.4);
algo.unfold();
algo.plotError();

algo.setMethod('invert');
algo.unfold();
algo.plotErrors();
```

- Easing the process of reiterating unfolding analysis

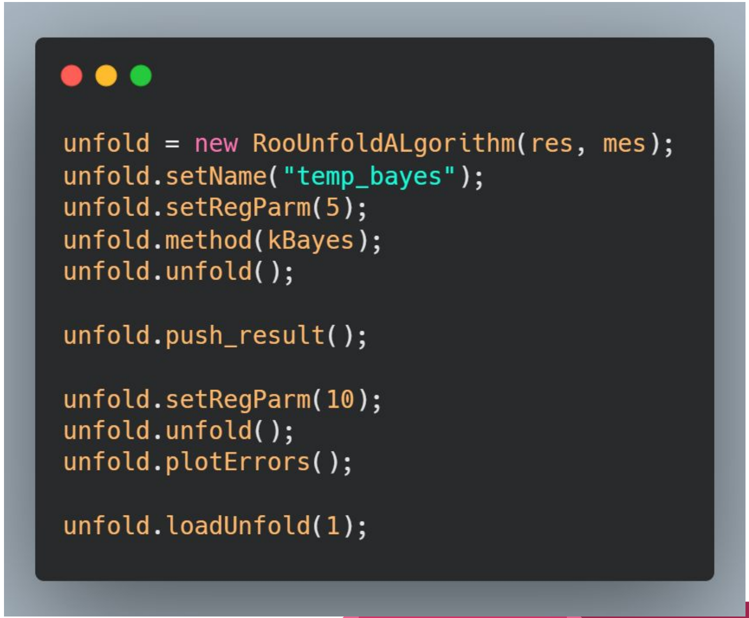
- Decoupling the implementation from the interface.

- Ensuring backward compatibility and easy to learn for users with optimized performance

What’s next !!

- Extend RooUnfoldAlgorithm
- Put development changes in production
- Ensure Backward compatibility changes in development
- Release version 3.0

Possible extensions



```
unfold = new RooUnfoldALgorithm(res, mes);
unfold.setName("temp_bayes");
unfold.setRegParm(5);
unfold.method(kBayes);
unfold.unfold();

unfold.push_result();

unfold.setRegParm(10);
unfold.unfold();
unfold.plotErrors();

unfold.loadUnfold(1);
```

Thank you

Any questions?

Acknowledgement

To my mentors for their constant support and guidance

Thanks Vince, Tim, Pim, Lydia, Carsten :)