RooFitUnfold

Lydia Brenner, Carsten Burgard,
Vincent Croft, Pim Verschuuren

Introduction

- Many frameworks / implementations for unfolding exist
  - Most use RooUnfold as a backend internally
  - Main focus of many of these frameworks: uncertainty handling
- Updates to RooUnfold itself
  - Integration with RooFit
    - Easier uncertainty handling
  - Workspace handling
    - Easy for combination
- Make RooUnfold “future proof”
  - Ready for possible unbinned unfolding methods in the future
  - Improved user friendliness
- End product
  - Saved in a way to allow changing of method at later time

Unfolding and fitting

- Unfold taking systematic variations into account
  - Also done by other advanced frameworks
- Can unfold after fitting signal and background contributions
  - Need to bring fit results into a format suitable as unfolding input
  - Non-trivial to propagate uncertainties
- Why not do both at the same time?
  - Ideal solution: RooFit implementation of unfolding!

Introduction to RooFitUnfold

Idea: Updated implementation of RooUnfold directly in RooFit

- Includes: Improved handling of uncertainties
  - Uses error propagation from any NPs to the unfolded distribution
  - Allows for inclusion of uncertainties coming from migration matrix
- Handels different input formats
  - Histograms (as RooUnfold did)
  - pdfs -> Means unbinned distributions can now be unfolded
    - Binned methods allow setting of internal binning
    - unbinned methods can technically be included in the future
- Lives in workspaces

Methods

All RooUnfold methods included

- Iterative Bayes
- IDS
- SVD
- TUnfold
- Gaussian Processes unfolding (NEW)
- Unregularised
  - Bin-by-bin
  - Matrix inversion
- Can easily include more methods

Documentation: https://gitlab.cern.ch/roofitunfold-tutorial-2019/RooUnfold/blob/master/README.md

https://arxiv.org/pdf/1105.1160.pdf

Implementation

- RooUnfold uses TH1 objects as basis
  - Very user-friendly, but internally not ideal with RooFit
- Templated to use RooAbsReal as a base object
  - Can easily be plugged on top of an existing workspace
- Created RooUnfoldFunc
  - a RooAbsReal wrapper around RooUnfold

Implementation: Inputs

- Truth distristributions
  - Histograms (TH1) or pdf (RooFit/Workspace)
- Reco distributions
  - Histograms (TH1) or pdf (RooFit/Workspace)
- Response matrix
  - 2D Histogram (TH2) or pdf (RooFit/Workspace)
- Data: background subtracted if needed
  - binned (TH1 or RooDataHist) or unbinned (TTree or RooDataSet)

Implementation: Features

- RooUnfoldFunc can be imported into workspace
  - Can use any existing workspace as an input
  - Can update reco level workspace after unfolding
  - Can unfold and fit (on reco-level) simultaneously
  - Easy persistence
  - Extremely useful for combinations
- RooUnfoldSpec can be used to construct RooUnfoldFunc
  - Helper class similar to HistFactory
- Unfolding result is only cached
  - Can switch to a different unfolding method a-posteriori

Workspace write out

Directly written out into a workspace

- At any level of the analysis
- Saves all information to be able to do a change of unfolding method on the fly
- Includes error propagation
- Writes out for ALL unfolding methods
    - So also for regularised methods

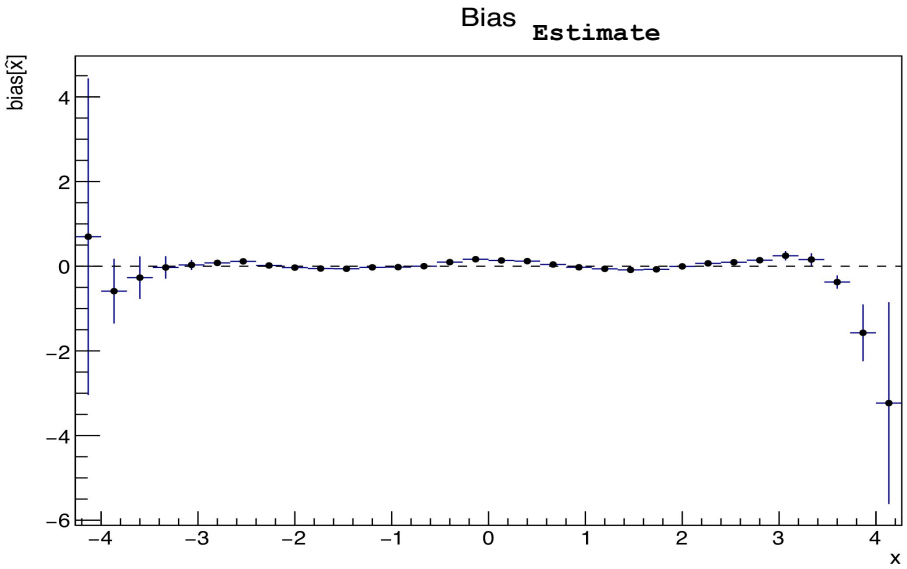
Error propagation

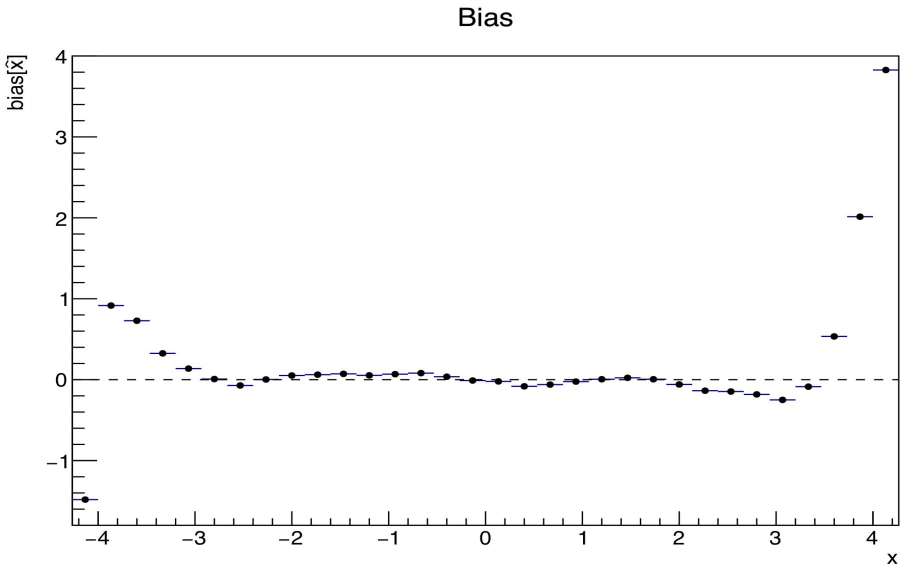
- Default RooUnfold can propagate simple uncertainties
  - Statistical uncertainties on Data
  - Bin-by-bin correlations
  - No handling of systematic uncertainties!
- RooFit functions (pdfs) can depend on arbitrarily many parameters
  - automatic error propagation from input parameters to all outputs by RooFit
  - ony requirement: the output needs to be a RooFit object
  - Nuisance parameter treatment comes “for free” with RooUnfold integration in RooFit
- No explicit handling of systematic uncertainties needed in RooUnfold
  - RooUnfold+RooFit handles uncertainties neatly :)
  - Some toy sampling methods required for bias calculation, but error bands on plots come directly from RooFit

Bias

Two bias calculations included

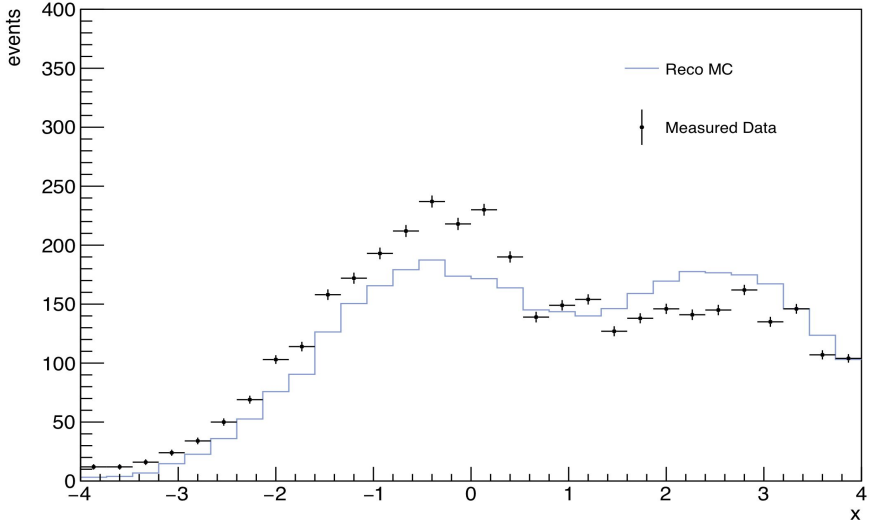
Bias estimate without toys and a full bias calculation

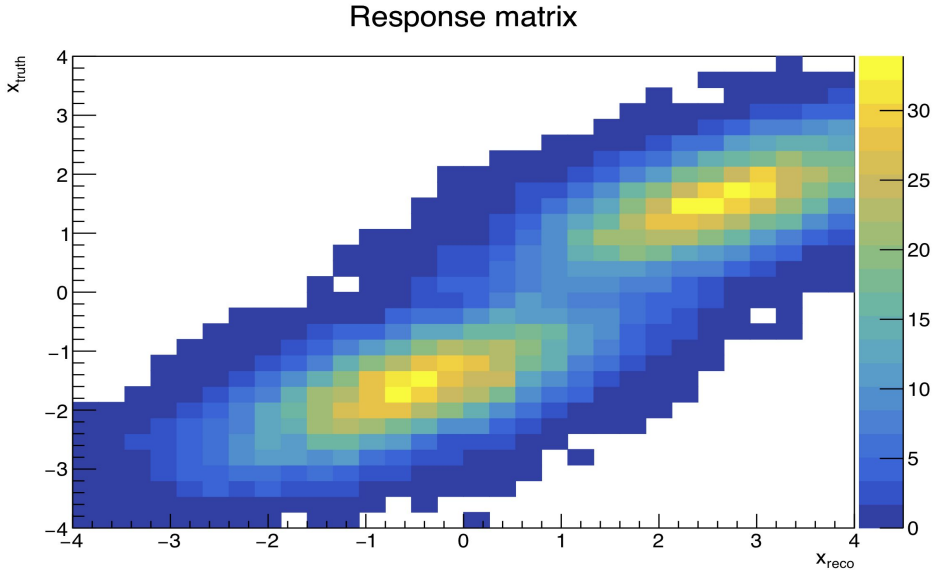




Example: Bimodal distribution

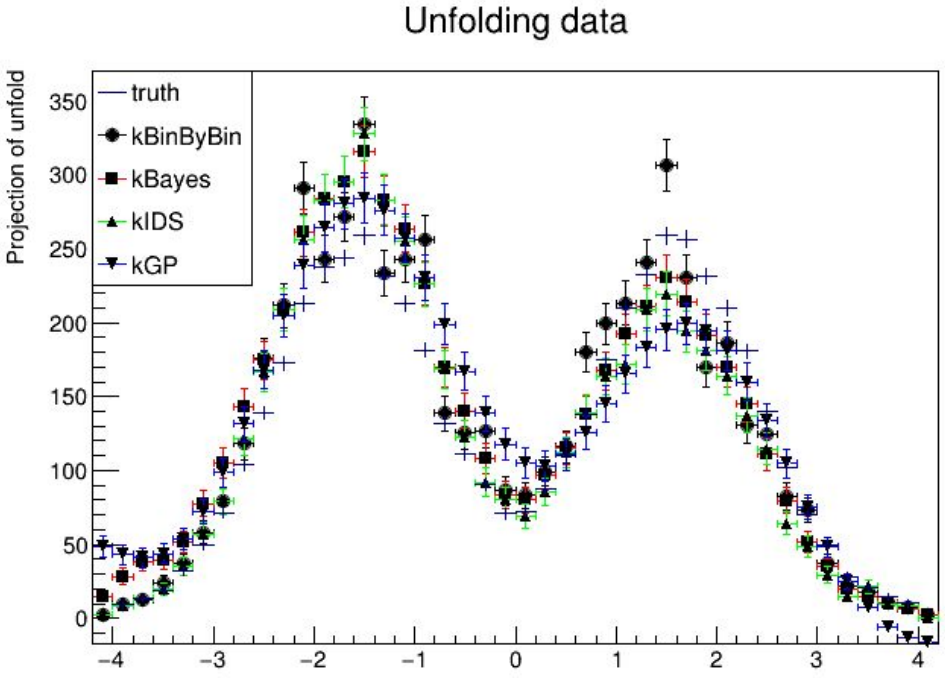
Reconstruction level plots





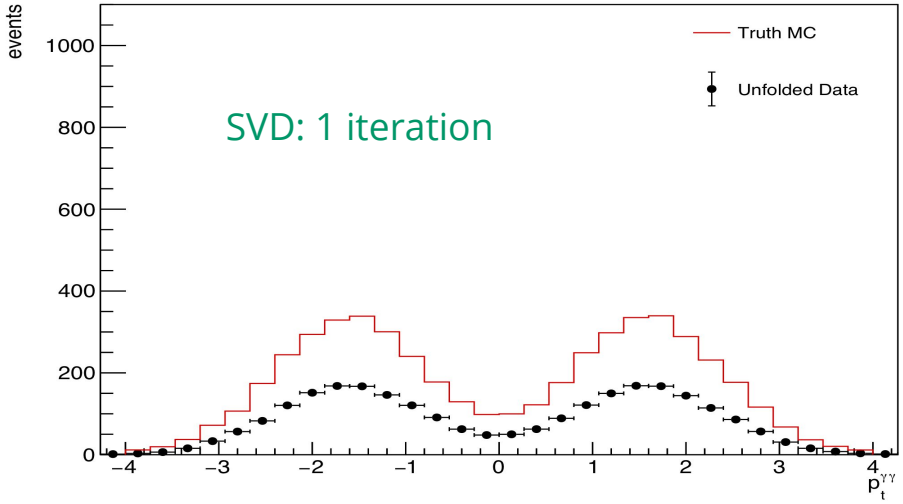
Example: Bimodal distribution

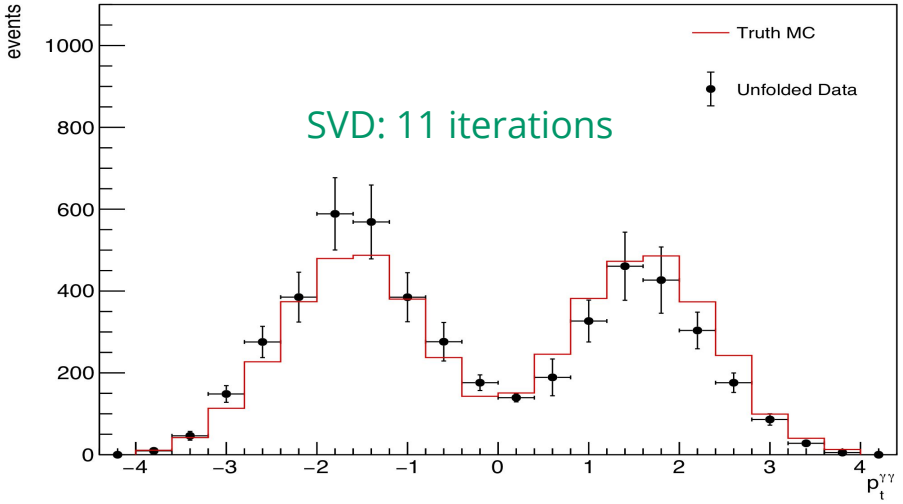
Compare different unfolding methods



Example: Bimodal distribution

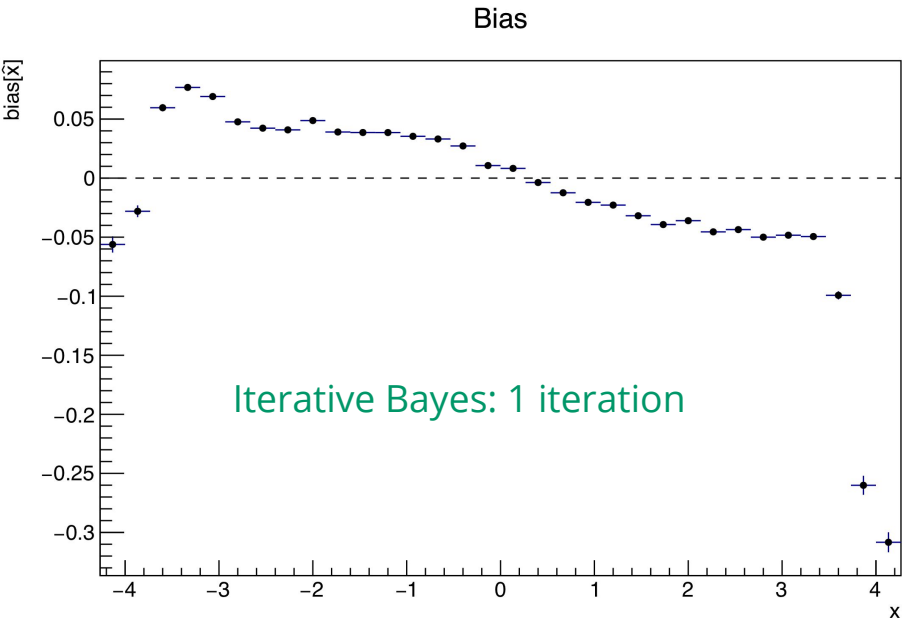
Compare different regularisation strengths

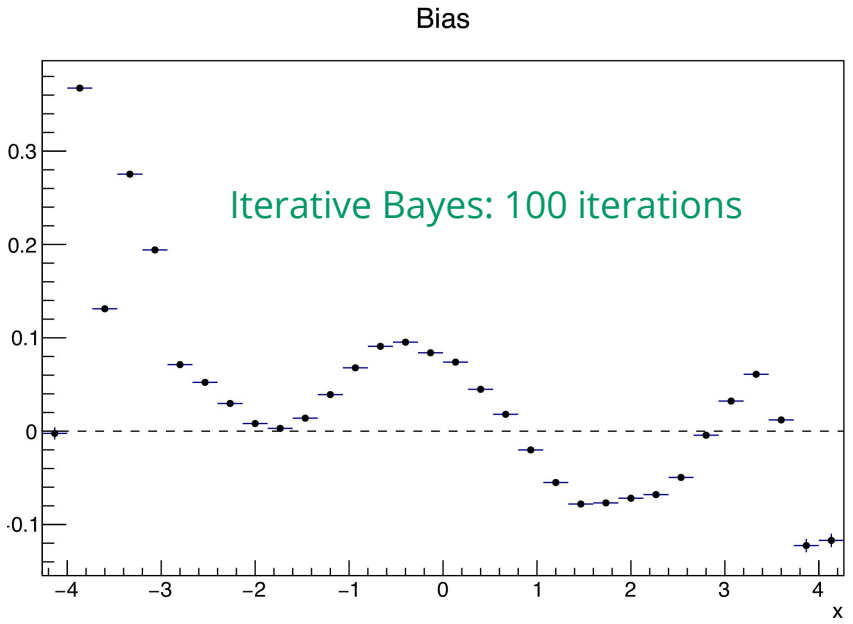




Example: Bimodal distribution

Compare different regularisation strengths: Don't forget the bias!





Planned updates

- Include more unfolding methods
  - RUN
  - Poisson based likelihood with regularization constraint term
  - Something unbinned?
- Improve regularisation optimisation algorithm
  - Improve the L curve scan with an eigenvalue analysis of the error matrix
  - Implement regularization strength or iterations choice based on the Effective Degrees of Freedom (EDF)
  - Other data driven choices of the regularization strength
- Incorporate the possibility to check coverage
  - Undersmoothing tool based on coverage
- Small improvements in current methods
  - Alter the SVD to handle asymmetrical cases
  - Show how the fakes are handled in the Iterative Bayes class

Summary

Updates with respect to RooUnfold

- Updated uncertainties handling
- Writes out workspace directly
  - And allows for on-the-fly change of unfolding method
- Automated bias calculations

RooFitUnfold only needs next ROOT release

Big thanks to Glen Cowan, Tim Adye, Wouter Verkerke and Mikael Kuusela

Back-up

Bias calculation

(Asimov) data-driven

First the uncertainties are taken truth level from the unfolded Asimov dataset. Toys are thrown in each bin around the Asimov truth values based on the full uncertainty. These toys are called level 1 toys. For each level 1 toy further toys are thrown, called level 2 toys. Each of the level 2 toys is folded and then unfolded with the chosen unfolding method. The bias for each level 2 toy is calculated as

$$\text{bias}_{l2} = (\sigma_{\text{refold}} - \sigma_{\text{truth}})/\sigma_{\text{truth}},$$

where the truth refers to the value of the level 1 toy at truth level from which the level 2 toy is thrown and refold refers to the value of the level 2 toy after folding and unfolding. The bias of each bin in the distribution that is being unfolding is the average over all bias_{l2}.