Unfolding with RooUnfold

Tim Adye
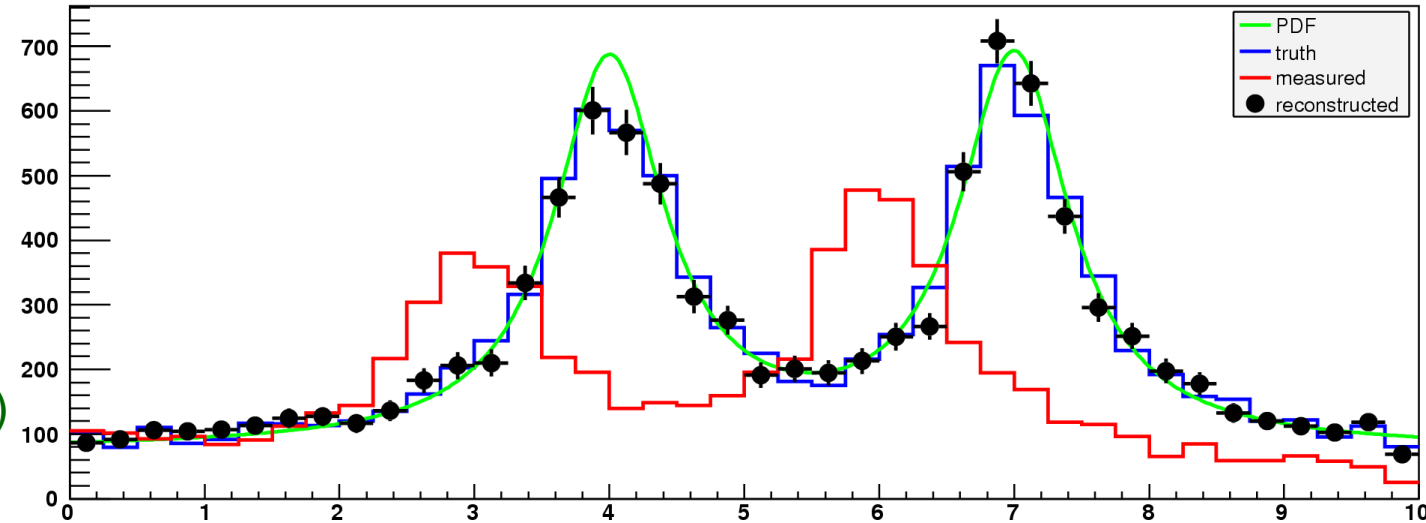
Rutherford Appleton Laboratory

PPD R&D Showcase Meeting

26th October 2022

- In other fields known as "deconvolution" or "unsmearing"
  - often applied to "correct" images; we normally use it for histograms
- Given a "true" distribution, μ_j, that is corrupted by measurement/detector effects, described by a response matrix, R_{ij}, and background, β_i, we measure

$$\nu_i = \sum_{j=1}^{N} R_{ij}\mu_j + \beta_i \qquad \text{or} \qquad \boldsymbol{\nu} = \boldsymbol{R}\boldsymbol{\mu} + \boldsymbol{\beta}$$

- This may involve
1. bin migration and smearing
   - events moving between bins (off-diagonal R_{ij})
   - if not this, then don't bother with unfolding
2. inefficiencies
   - undetected events ($\epsilon_i = \sum_{j=1}^{N} R_{ij} < 1$)
3. background / fake events
   - measured events not from true distribution ($\beta_i > 0$)

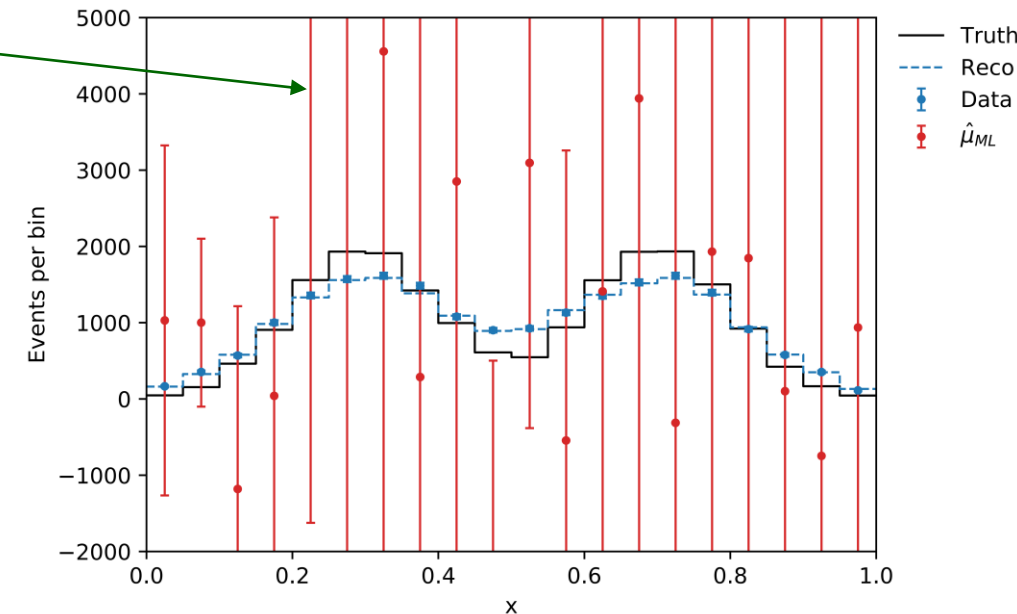- We use unfolding to try to recover the true distribution

Slide from PPD Graduate Lectures 2022

Regularisation

- We could unfold by inverting the response matrix
$$\boldsymbol{\nu} = \boldsymbol{R}^{-1}(\boldsymbol{\mu} - \boldsymbol{\beta})$$

- Statistical fluctuations mess this up: $\boldsymbol{R}^{-1}$ can't distinguish between a fluctuation and fine structure in the truth, $\boldsymbol{\nu}$
  - Results in large uncertainties in the unfolded result

Credit: Adam Bozson (RHUL, 2018)

- This can be resolved with some form of regularisation
  - reduce the statistical error by introducing a systematic bias
  - regularisation parameter controls degree of bias
  - many different methods available, e.g.
    - Simple correction factor method assumes similar bin-migrations as in MC
    - Tikhonov regularisation biases to a smooth distribution
    - Iterative Bayes Unfolding biases towards initial guess (MC truth)

A brief history of a long development – I

- 09/2004: **Fergus Wilson (RAL)** implements Iterative Bayes Unfolding (IBU) based on D'Agostini (1995)
- 12/2005: **TA (RAL)** developed RooUnfold within BaBar as a common tool for IBU and SVD unfolding from Kerstin Tackmann (LBNL)
- 04/2007: first public version on PPD web site: https://hepunx.rl.ac.uk/~adye/software/unfold/RooUnfold.html
  - includes bin-by-bin and unregularised matrix inversion reference algorithms
- 2010–11: Development/discussion as part of the Helmholtz Alliance Unfolding Project, including:
  - 05/2010: DESY unfolding workshop
  - 01/2011: PHYSTAT unfolding workshop at CERN
- 08/2010: Added interface to TUnfold after discussion with Stefan Schmitt (DESY) [Schmitt 2012]
- 08/2010: Error calculation using toys
  - These two improvements implemented by our summer student, **Richard Claridge (RAL)**
- 01/2011: Reimplement IBU unfolding using matrix methods; fix error propagation
- 10/2011: release version 1.1.1
- 10/2011: Add propagation of response matrix uncertainties
- 05/2015: RooUnfold included in ATLAS `RootCore` releases
- 07/2017: Chris Meyer (UPenn) adds Iterative, Dynamically Stabilized (IDS) unfolding algorithm [Malaescu 2011]

A brief history of a long development – II

- 09/2019: release version 2.0.0 and move to GitLab: https://gitlab.cern.ch/RooUnfold/RooUnfold
  - development and support shifted to a small team:
    Carsten Burgard (Nikhef), Lydia Brenner (DESY), Vincent Croft (Tufts), Pim Verschuuren (RHUL), **TA (RAL)**
- 12/2021: release version 3.0.0 incorporating RooFitUnfold
  - included new algorithms: Gaussian Processes and Poisson (Likelihood/Tikhinov) unfolding, both from Pim Verschuuren (RHUL)
  - migration to new version supported by GSoC 2021 student, Archit Agrawal
- 06/2022: RooUnfold included in ATLAS `StatAnalysis` releases by **Will Buttinger (RAL)**

INSPIRE lists 207 paper (+343 other) citations to RooUnfold PHYSTAT note
I've had email contact from 159 people from many different experiments in PP/PA/NP

Current status / work

- You can find RooUnfold here: https://gitlab.cern.ch/RooUnfold/RooUnfold
  - includes links to documentation and references

- most work now is support and bug fixes
  - work shared by the team – so not so much for me to do ☺
  - could add directly into ROOT, so don't have to compile separately
    - this has been a goal for decades (!)
    - would be excellent, we discuss with the ROOT team who are encouraging, but we never seem to have the time to implement
  - occasional new algorithms added
  - recent developments in optimisation and performance validation

- Separate to the direct development of RooUnfold, I've been involved with a small team that is planning an ERC grant application for statistical research, software implementation, and applications on unfolding.
  - Nikolay Gagunashvili and Helmut Neukirchen (Iceland), Michael Schmelling (MPI), Florin Maciuc (Bucharest)
  - Currently I'm listed as an "elder statesman/general advisor" for the project, but they'd like me to become an active participant (or PI) if I were available
    - I don't think I can devote the time to make this worthwhile

Backup

Hands-on Statistics

Tim Adye, Will Buttinger

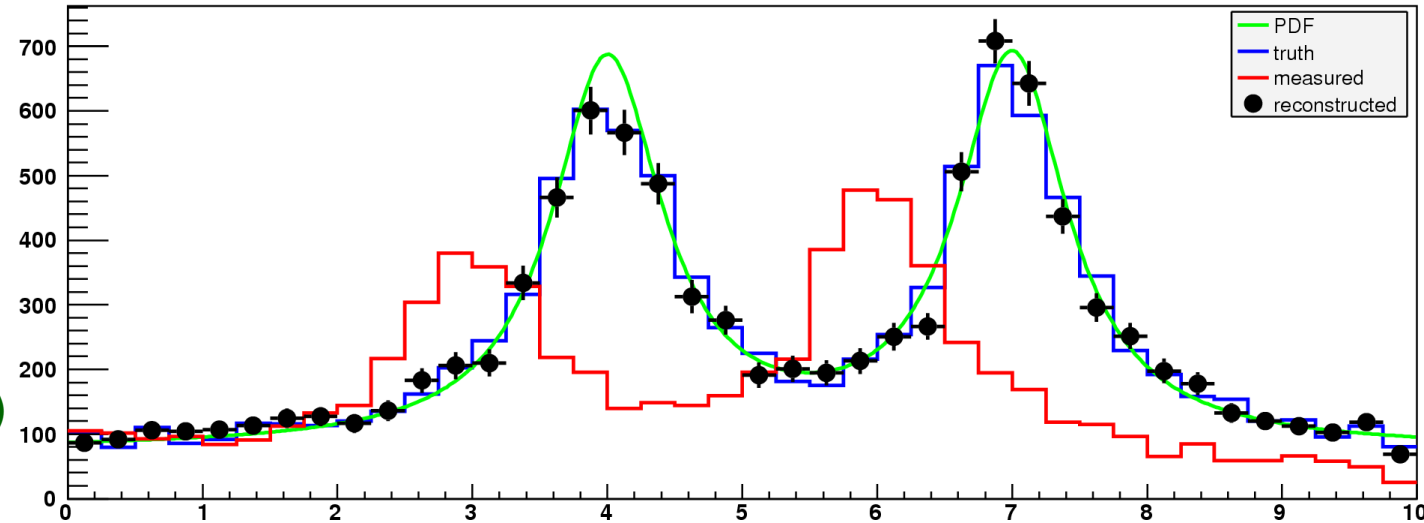Rutherford Appleton Laboratory

PPD Advanced Graduate Lectures
16th June 2022

- In other fields known as "deconvolution" or "unsmearing"
  - often applied to "correct" images; we normally use it for histograms
- Given a "true" distribution, μ_j, that is corrupted by measurement/detector effects, described by a response matrix, R_{ij}, and background, β_i, we measure

$$\nu_i = \sum_{j=1}^{N} R_{ij}\mu_j + \beta_i \qquad \text{or} \qquad \boldsymbol{\nu} = \boldsymbol{R}\boldsymbol{\mu} + \boldsymbol{\beta}$$

- This may involve
  1. bin migration and smearing
     - events moving between bins (off-diagonal R_{ij})
     - if not this, then don't bother with unfolding
  2. inefficiencies
     - undetected events ($\epsilon_i = \sum_{j=1}^{N} R_{ij} < 1$)
  3. background / fake events
     - measured events not from true distribution ($\beta_i > 0$)

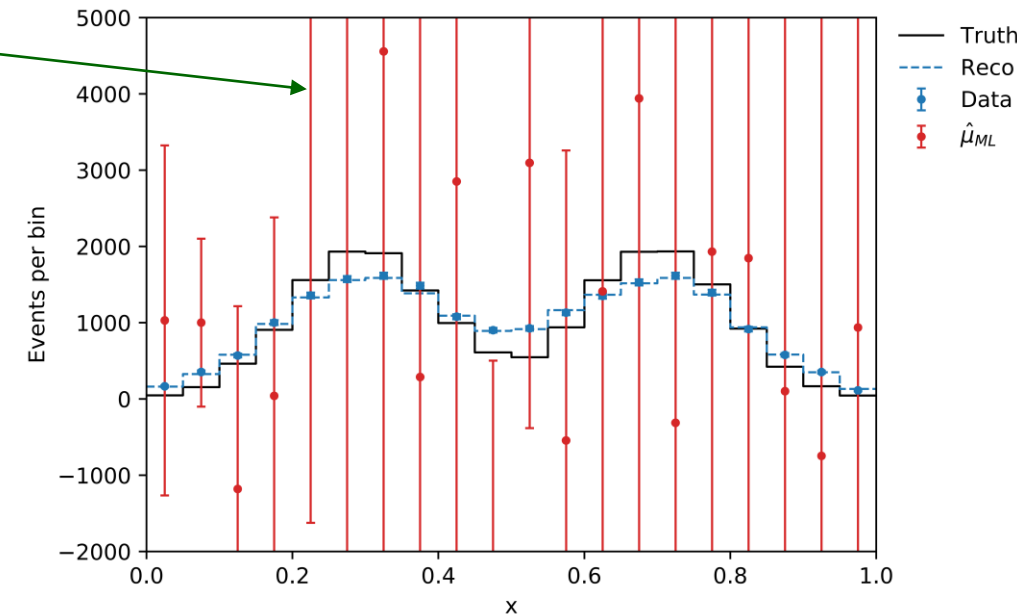- We use unfolding to try to recover the true distribution

- We could unfold by inverting the response matrix
  $$\boldsymbol{\nu} = \boldsymbol{R}^{-1}(\boldsymbol{\mu} - \boldsymbol{\beta})$$
- Statistical fluctuations mess this up: $\boldsymbol{R}^{-1}$ can't distinguish between a fluctuation and fine structure in the truth, $\boldsymbol{\nu}$
  - Results in large uncertainties in the unfolded result

Credit: Adam Bozson (RHUL, 2018)

- This can be resolved with some form of regularisation
  - reduce the statistical error by introducing a systematic bias
  - regularisation parameter controls degree of bias
  - many different methods available, e.g.
    - Simple correction factor method assumes similar bin-migrations as in MC
    - Tikhonov regularisation biases to a smooth distribution
    - Iterative Bayes unfolding biases towards initial guess (MC truth)

- Don't unfold! (unless you can't help it)
  - if you have the truth, apply ("fold") the detector/measurement response to the truth
    - much better than trying to unfold the measurements back to the truth, e.g.
    1. if you have a functional model for your truth distribution, fit the model parameters
    2. if you have a theory prediction, compare the corrected predictions to your measurements
- Unfolding is needed if your final result has to be the "true" distribution, for which there is no model
  - e.g. for comparison with another experiment

- Don't regularise if you don't have to
  - If the statistical fluctuations are small compared to the bin migrations, can just invert
    - check the statistical uncertainties (especially bin-bin correlations) are acceptable

- If you have to regularise:
  - optimise regularisation parameter for bias vs statistical uncertainties
  - check bias with systematically independent MC samples (e.g. different event generators)
    - bias should be included in systematic uncertainties of the result
  - unfolding introduces bin-bin correlations – in bias and statistical errors
    - need to be understood and reported
  - ideally cross-check with more than one unfolding method

- ROOT includes <u>TSVDUnfold</u> and <u>TUnfold</u> methods built-in

- <u>RooUnfold</u> package provides a common interface and tools for several unfolding methods:
    1. unregularised matrix inversion
    2. simple correction factors
    3. Iterative Bayes unfolding (IBU)
    4. SVD unfolding (via TSVDUnfold)
    5. Tikhonov regularization with least square fit (via TUnfold)
    6. unfolding with Gaussian Processes (GP)
    7. Iterative dynamically-stabilized unfolding (IDS)

Backup
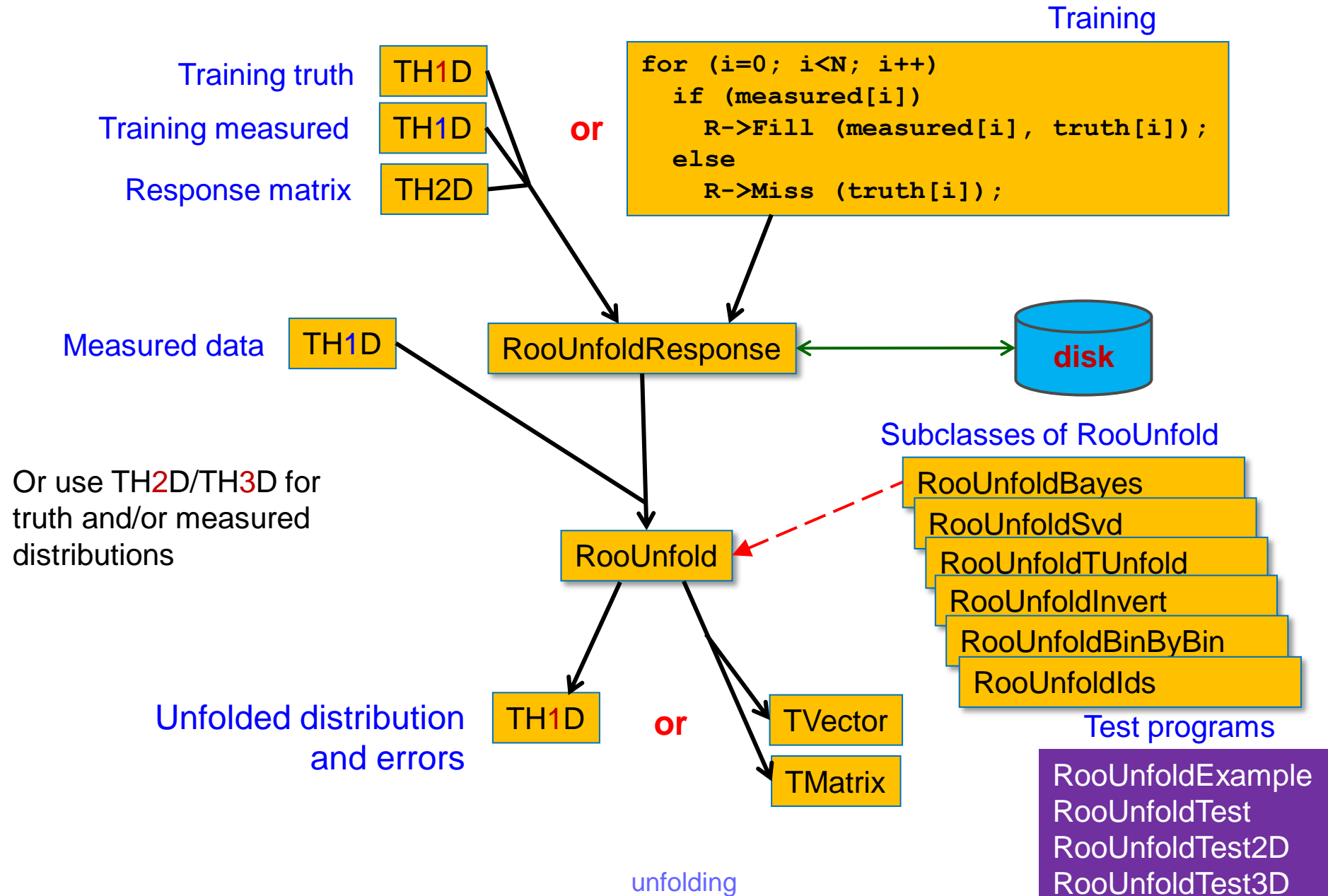
RooUnfold
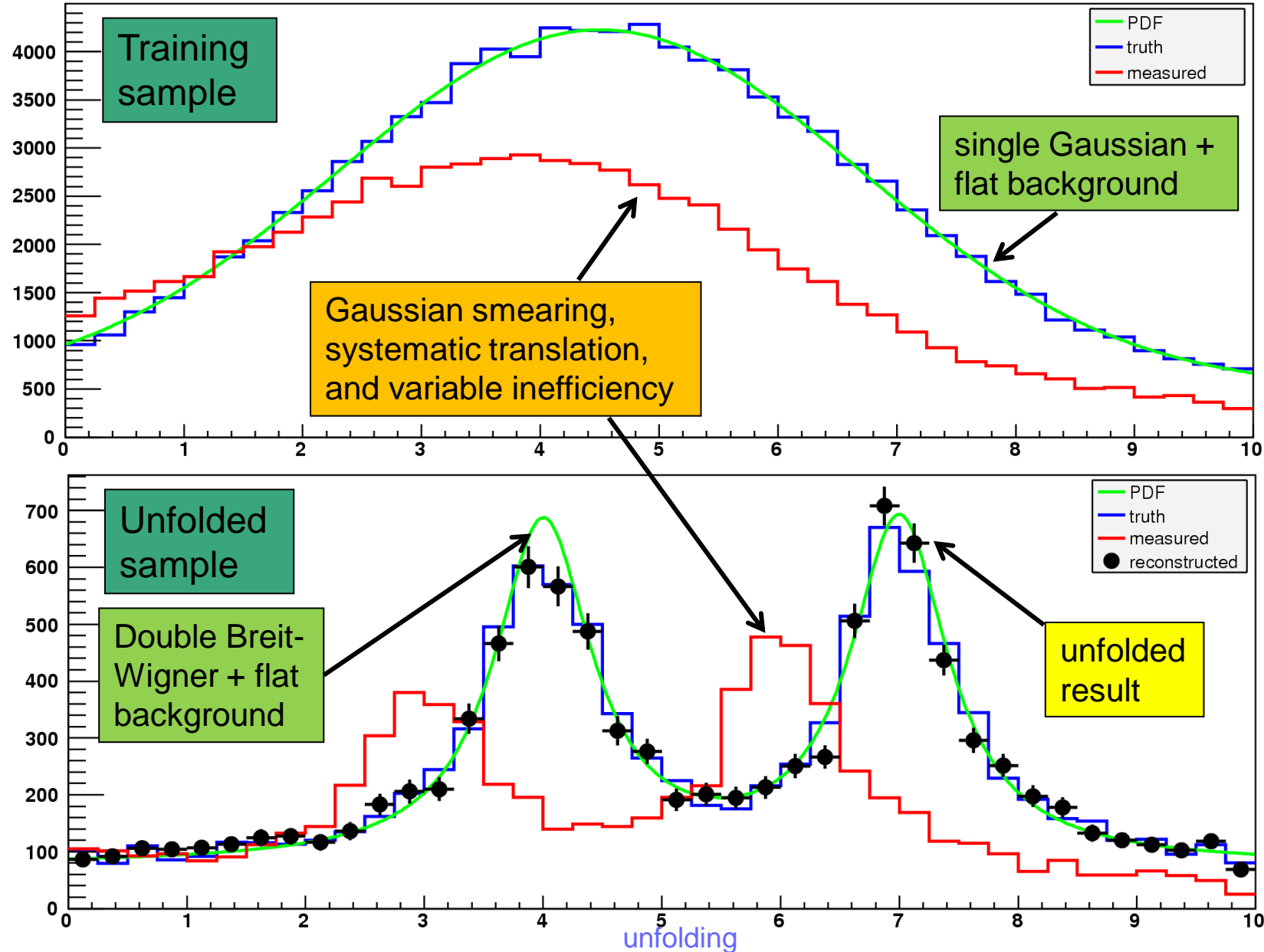**a history, scope, and status**

Tim Adye
Rutherford Appleton Laboratory

DIANA Meeting – Unfolding in HEP
13th November 2017

- RooUnfold's purpose is to provide a framework for testing and using different binned unfolding methods, as far as possible maintaining a consistent view of each method to the user
  1. Iterative Bayes method was implemented for RooUnfold
     - Based on D'Agostini (1995), with corrected error calculation
     - This method seems to be used by the majority of RooUnfold users
  2. Iterative, Dynamically Stabilized (IDS) unfolding (Malaescu 2011)
     - Recently added to RooUnfold by Chris Meyer
  3. Singular Value Decomposition (SVD) method (Hoecker, Kartvelishvili 1995)
     - RooUnfold interface to enhanced version of TSVDUnfold ROOT class by Kerstin Tackmann et al
  4. Regularised least square fit, implemented in TUnfold (Schmitt 2012)
     - RooUnfold interface to TUnfold class in ROOT
  5. unregularised matrix inversion with singular value removal (TDecompSVD)
  6. using bin-by-bin correction factors, with no inter-bin migration

- RooUnfold provides
  - Common features and interfaces to the different algorithms
  - Convenience methods for providing the inputs in different forms
    - response matrix can be given as histograms, matrices, or filled directly
    - 1D, 2D, 3D truth and/or measured distributions
    - Different binning in measured and truth
  - Support for inefficiency and fakes
    - even when not directly supported by underlying algorithm
  - Propagation of errors or automatic calculation using toys
    - Toys used to validate linear error propagation
  - Simple examples of use by C++ and Python
  - Extensive test suite for comparing methods using various simple toy MC distributions
    - 1D, 2D, 3D; bias and smearing; inefficiency and fakes; correlated inputs
    - Simple testing – all from the command-line

- Overview and class documentation, running instructions, download, and references:
  http://hepunx.rl.ac.uk/~adye/software/unfold/RooUnfold.html

RooUnfold classes

Training

Training truth — TH1D

Training measured — TH1D

Response matrix — TH2D

or

```
for (i=0; i<N; i++)
   if (measured[i])
      R->Fill (measured[i], truth[i]);
   else
      R->Miss (truth[i]);
```

Measured data — TH1D

RooUnfoldResponse ⟷ disk

Subclasses of RooUnfold

Or use TH2D/TH3D for truth and/or measured distributions

RooUnfold

RooUnfoldBayes
RooUnfoldSvd
RooUnfoldTUnfold
RooUnfoldInvert
RooUnfoldBinByBin
RooUnfoldIds

Unfolded distribution and errors — TH1D or TVector / TMatrix

Test programs

RooUnfoldExample
RooUnfoldTest
RooUnfoldTest2D
RooUnfoldTest3D

Training sample

single Gaussian + flat background

Gaussian smearing, systematic translation, and variable inefficiency

Unfolded sample

Double Breit-Wigner + flat background

unfolded result

- 1D, 2D, 3D truth and/or measured distributions
  - Internally unpacked into 1D distributions
    - Multi-dimensional smoothing automatically selected for TUnfold
    - Only 1D smoothing implemented for SVD method
    - Other methods don't require smoothing
  - Choose binning with care, eg. response matrix for 2D distributions has N^4 bins
    - can require lots of MC for reasonable statistics

- Different binning in truth and measured distributions
  - TUnfold method requires the number of bins, $N_{measured} \geq N_{true}$
    - TUnfold claims best results if $N_{measured} > N_{true}$, eg. $N_{measured} = 2N_{true}$
    - This is a common general recommendation from unfolding experts, but perhaps most relevant to these types of algorithms with explicit regularisation
    - This is an implicit additional regularisation, since we are "smoothing" two bins into one
  - SVD and bin-by-bin methods do not work well unless $N_{measured} = N_{true}$
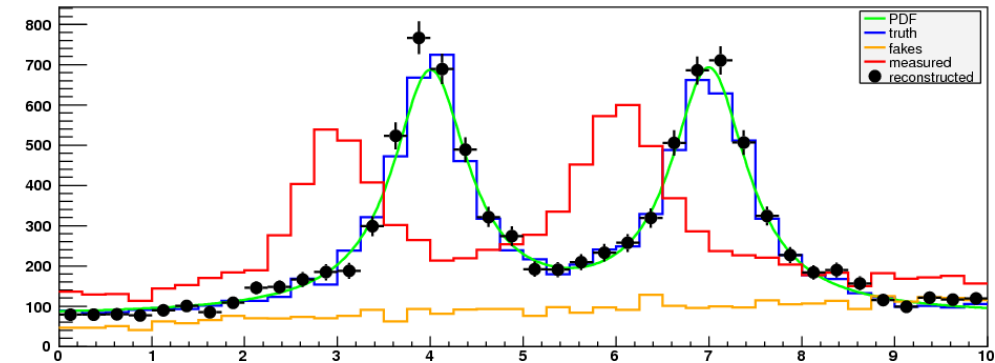
- As well as migration between bins, unfolding can also model
  - **misses, inefficiencies, effectless-causes:**
    - truth entries with no corresponding measurement
  - **fakes, backgrounds, causeless-effects:**
    - measurements with no corresponding truth entry

- RooUnfold allows misses and fakes to be specified when filling the response matrix
  - or inferred from the 1D truth/measured histograms given when creating the response matrix from a 2D histogram
  - Dealt with differently for each unfolding method, but generally misses are treated as a multiplicative efficiency and fakes as an additive background.
    eg. for the iterative Bayes method
    - Misses: The response matrix is scaled by the efficiency of each truth bin
    - Fakes: an extra truth bin for background is added to the response matrix
- Alternatively, can be handled outside RooUnfold by adjusting input and output
  - RooUnfoldResponse should then exclude misses and fakes
    - or in TH2 constructor, pass null histogram for TH1 measured/truth histograms

- RooUnfold can calculate the errors on the unfolded distribution as
  - histogram bin errors
    - no bin-bin correlations, returned in `unfold->Hreco()` histogram; or
  - full covariance matrix
    - `unfold->Ereco()`, returning a `TMatrixD`
    - Regularisation and bin-bin migrations normally introduce significant bin-bin correlations
- Unfolding errors can be based on
  - input measurement errors
    - passed with input histogram or default $\sqrt{n}$; or
  - full covariance matrix of bin-bin measurement errors
    - `unfold->SetMeasuredCov(TMatrixD)`; and/or
  - errors on the response matrix
    - eg. due to finite MC stats
    - enable with `unfold->IncludeSystematics()`
- The error propagation can be calculated by
  - propagation of errors through the unfolding transformations (default)
    - SVD (response matrix errors) and IDS (all errors) use toys internally; or
  - toys generated internally by RooUnfold (`RooUnfold::kCovToy` flag)
- All these possibilities should work with all the unfolding algorithms

unfolding errors for Bayes method

Legend:
- 9 iterations
- 3 iterations
- 2 iterations
- 1 iterations
- Toy MC RMS
- unfolding errors

histogram

points
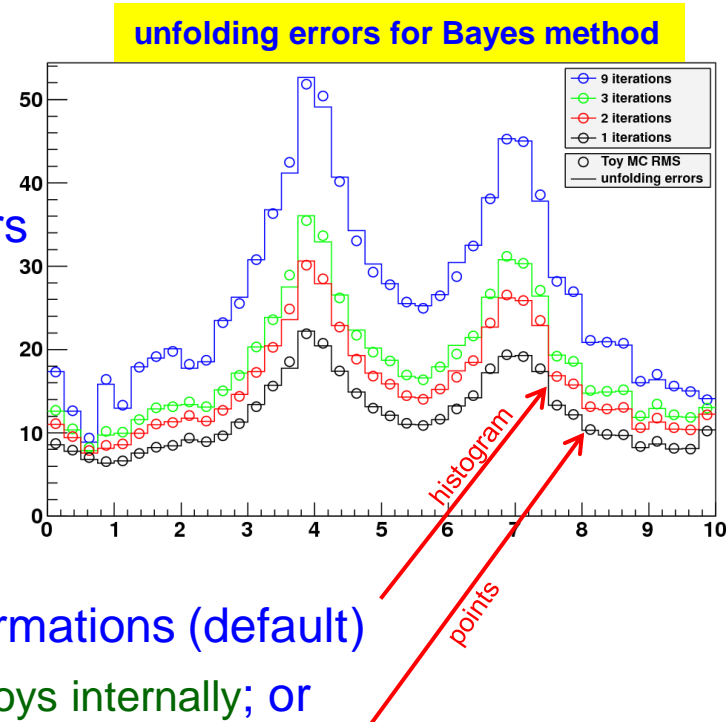
very brief summary of the <u>history</u>

- 10/2005: RooUnfold initially developed using an implementation of Bayes unfolding from Fergus Wilson (RAL) and then SVD unfolding from Kerstin Tackmann (LBNL)
- 2010–11: Development/discussion as part of the <u>Helmholtz Alliance Unfolding Project</u>, including:
  - 05/2010: <u>DESY unfolding workshop</u>
  - 01/2011: <u>PHYSTAT unfolding workshop at CERN</u>
- 08/2010: Added interface to TUnfold after discussion with Stefan Schmitt (DESY)
- 08/2010: Error calculation using toys
  - These two improvements implemented by RAL student, Richard Claridge
- 01/2011: Reimplement Bayes unfolding using matrix methods; fix error propagation
- 10/2011: last release version 1.1.1
  - Continuing development in SVN
- 10/2011: Add propagation of response matrix uncertainties
- 05/2015: RooUnfold included in ATLAS RootCore releases
- 07/2017: Chris Meyer (UPenn) adds IDS unfolding algorithm

- INSPIRE lists 110 paper (+158 other) citations to <u>RooUnfold PHYSTAT note</u>
- I've had email contact from 132 people from many different experiments in PP/PA/NP

- Last release 6 years ago: RooUnfold version 1.1.1

- There has been some continuing development and minor bug fixes that can be accessed from <u>DESY SVN</u>
  - Recommend that people use the development version until a new release is made
    - Main changes are now quite mature, and compatible with previous versions
    - Release version 1.1.1 no longer compiles with current versions of ROOT
      - Development version maintains compatibility with old and new versions of ROOT

- See the "Development" section of the <u>web page</u> to download, or browse the dev version's class documentation

- The main improvement in the dev version is to allow uncertainties on the response matrix (eg. due to finite MC stats) to be propagated to the unfolded errors
  - Can include the effect of measurement and response matrix errors together or separately
  - Can be calculated using error propagation or automatic toys

- The other main <u>changes</u> since version 1.1.1:
  - Addition of RooUnfoldIds, implementing IDS unfolding
  - Improvements in SVD method (and TSVDUnfold), especially the error propagation
    - requires updated TSVDUnfold not yet in ROOT
    - Local version included in RooUnfold package
  - Simplify interactive running with CINT, CLING, ACLiC, or PyROOT
 - Test alternatives to unfolding: parameterised fit with response matrix

- I have collected many interesting and useful suggestions for improvements
  - interface with new algorithms, new tests, convenience methods, API…
    - but had very little time to work on unfolding, which was always a spare-time job
    - see backup for unedited list

- The first priorities are:
  - Make a new release based on the dev version
    - Validate changes for all algorithms – so far concentrated on Bayes and SVD methods
  - Update and improve documentation, especially
    - Setup and examples for ROOT6 with CLING or PyROOT
    - Document different ways of creating and filling the response matrix
    - Errors: propagation of errors or toys, measurement or response matrix errors
  - Later, submit for inclusion in ROOT
    - agreed in principle, but discussion was a long time ago

- hope to get to it in the next few weeks/months – but I've said that before ☹

- RooUnfold is a widely-used framework for accessing different unfolding algorithms
  - in particular, implements D'Agostini's iterative Bayes method

- Implements many common requirements
  - multidimensional binning
  - inefficiencies, fakes
  - error calculations
  - algorithm tests using toy MC

- Missing a common framework for evaluating algorithms and their configuration for a specific physics distribution
  - See next talk   → Chris Meyer

Backup

Unordered log of tests and ideas that could be investigated. No plan yet to implement any of them.

- RooUnfoldBayes option to allow to use flat (or other) prior (Jan Therhaag, Kolja Kauder 2/10/2017)
- Iterative bin-by-bin method: V. B. Anykeyev, A. A. Spiridonov, V. P. Zhigunov, *Correcting factors method as an unfolding technique*, NIM A322(1992)280-285
- Better figure of merit, eg. alternate chi^2 definition (Vato Kartvelishvili)
- Low statistics tests. Does toy MC handle bins where the fluctuation may go negative? Bayes toy MC has some funny effects
- RooUnfoldResponse convenience methods to created from a TMatrix and to plot directly the "Miss" data. Presumably can be done by subtracting the projection from the truth. (Andrii Chyzh, 22/6/2011)
- Fully Bayesian Unfolding method, https://arxiv.org/abs/1201.4612
- Use THn class for response matrix?
- RooUnfoldResponse::RunToy should probably update the measured and truth normalisations. How to handle the uncertainty in the efficiency and fakes? (Katharina Bierwagen, 18/7/2012)
- Consistent handling of fakes: add bin or adjust input? Currently (since 27/1/2012) add bin for Bayes, and subtract fakes for the others. Adding bin may not work well for SVD or TUnfold since the extra bin isn't continuous with the others. (Kerstin Tackmann 19/12/2012)
- Add GetEfficiency() (or GetMiss()?) to see unfolded efficiencies
- RooUnfoldBayes with 0 iterations could return the initial distribution - useful for debugging? (Sercan Sen, 28/3/2013)
- When errors aren't required (especially for toys), skip error propagation in Bayes (and other?) unfolding. This can be really slow for large number of bins. (Elzbieta Richter-Was 11/11/2013)
- Better name for IncludeSystematics() (IncludeResponseErrors?) This might be confused with the systematic effect from the unfolding procedure, which can't be evaluated directly by RooUnfold.
- API improvement suggestions from Gero Flucke. Must be upward-compatible.

Unfolded x-projection

Unfolded y-projection

truth

measured

reco
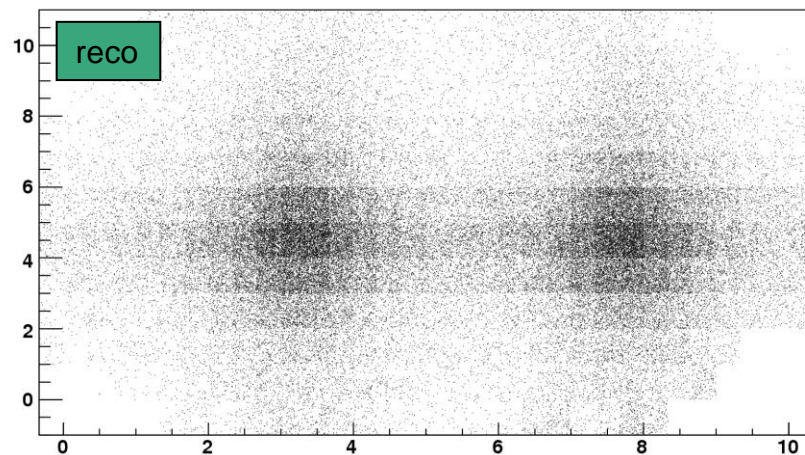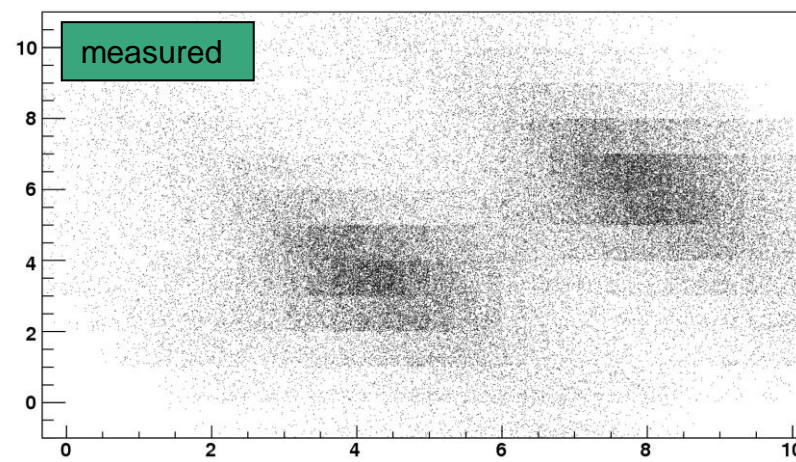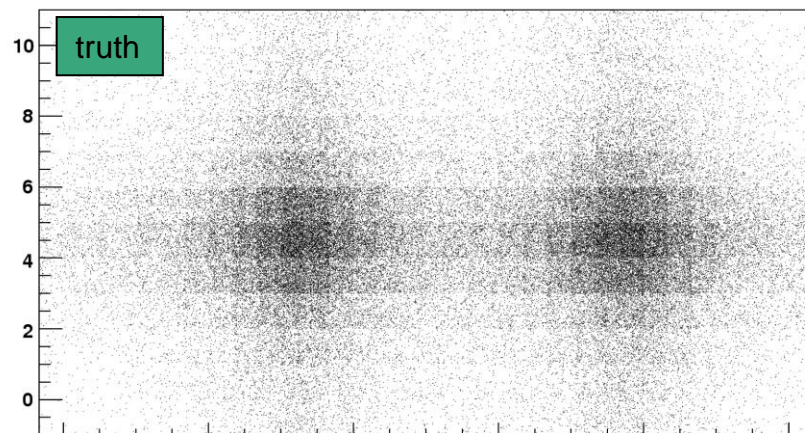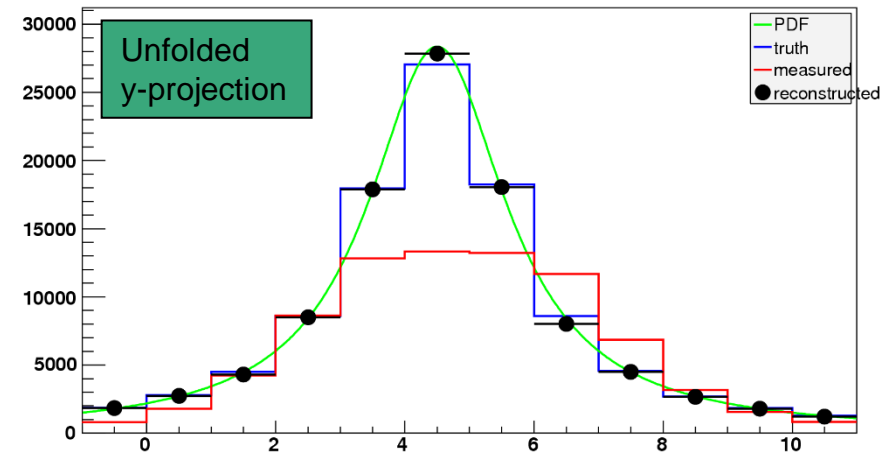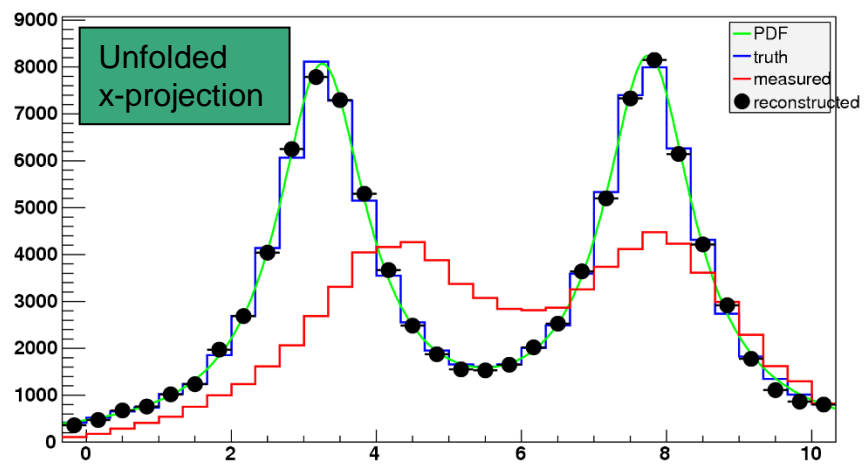
2D unfolding
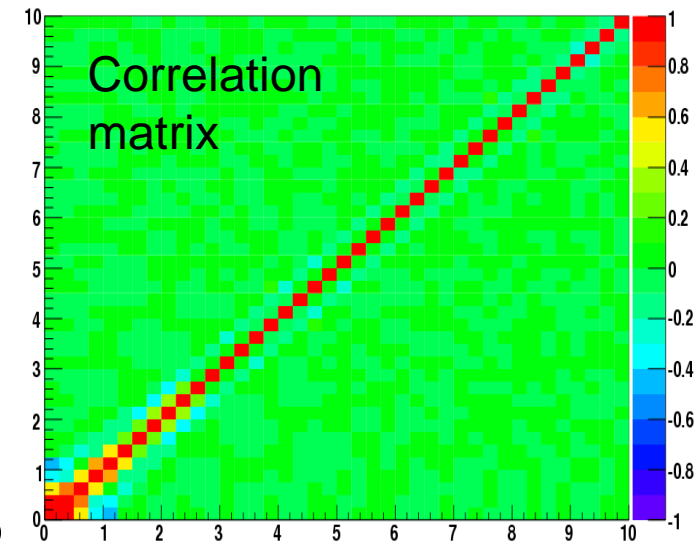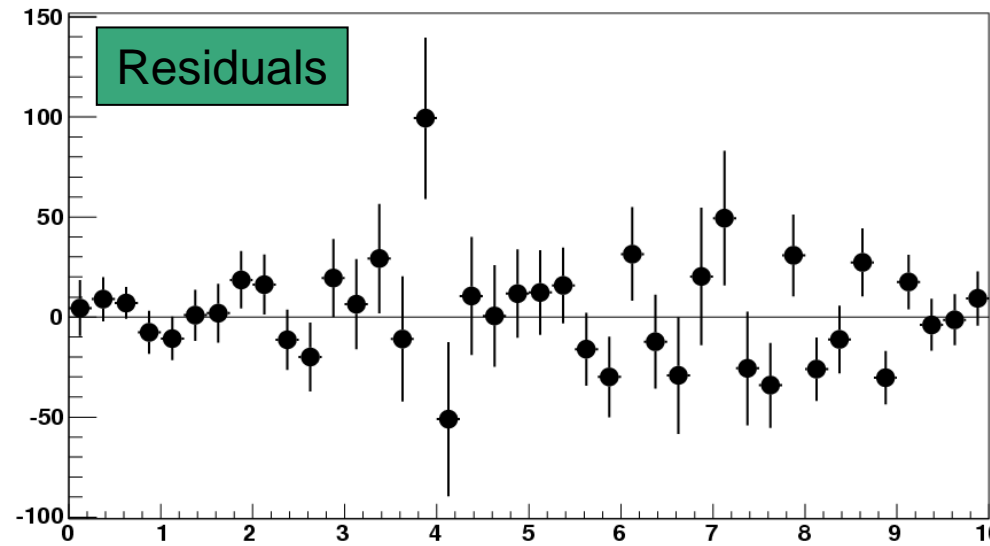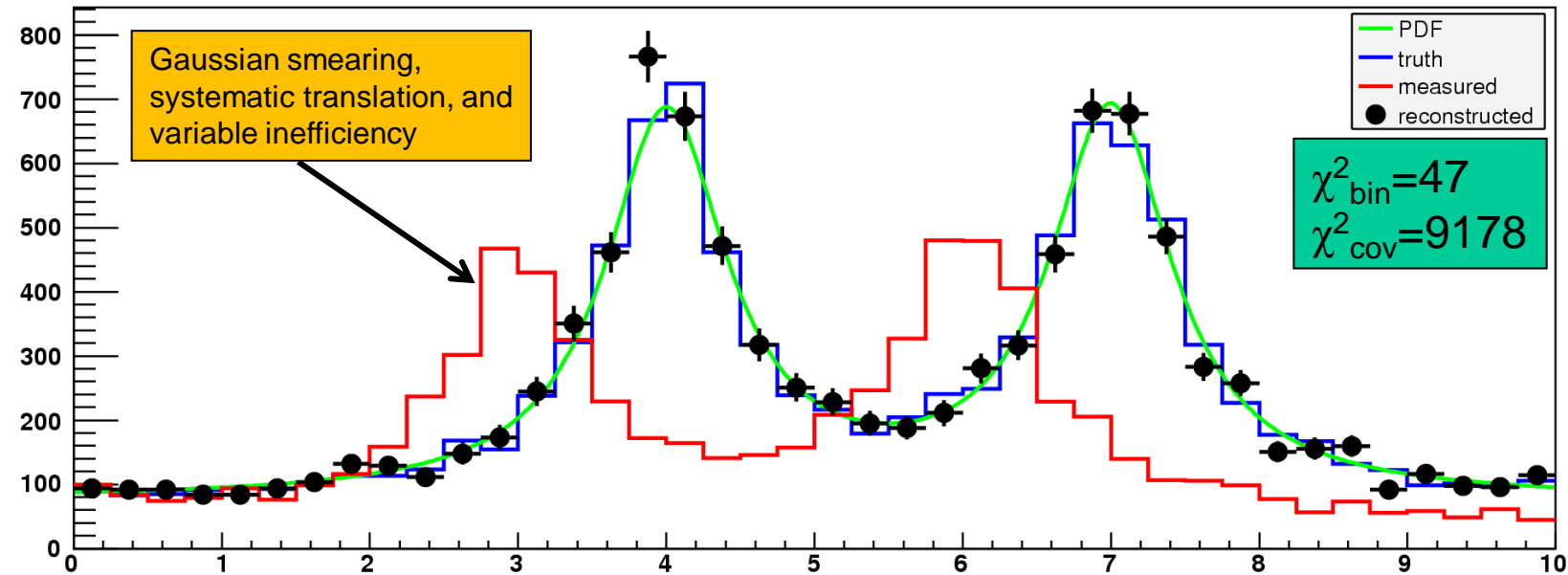
2D Smearing, bias, variable efficiency, and variable rotation

- Uses the method of Giulio D'Agostini (1995), implemented by Fergus Wilson and myself
  - Uses repeated application of Bayes' theorem to invert the response matrix
  - Regularisation by stopping iterations before reaching "true" (but wildly fluctuating) inverse
    - Regularisation parameters is the number of iterations, which in principle has to be tuned according to the statistics, number of bins, etc.
    - In practice, the results are fairly insensitive to the precise setting.
- Implementation details:
  - Initial prior is taken from training truth, rather than a flat distribution
    - Does not bias result once we have iterated, but perhaps reach optimum faster
  - Correctly account for propagation of errors through multiple iterations
    - D'Agostini's paper did not account for dependence on previous iterations in error propagation
  - Does not do smoothing
    - Did not find this helped, and can introduce additional bias

- Uses the method of Andreas Höcker and Vato Kartvelishvili (**GURU** Fortran program)
  - Implemented in C++/ROOT by Kerstin Tackmann and Heiko Lacker and subsequently included in ROOT as TSVDUnfold
    - RooUnfold includes an interface to this class
- Obtains inverse of response matrix using singular value decomposition
  - Use number-of-events matrix to keep track of MC uncertainties
- Regularisation with a smooth cut-off on small singular value contributions (these correspond to high-frequency fluctuations)
  - Replace $s_i^2 \rightarrow s_i^2 / (s_i^2 + s_k^2)$
  - k determines the relative contributions of MC truth and data
    - k too small → result dominated by MC truth
    - k too large → result dominated by statistical fluctuations
  - k needs to be tuned for the particular type of distribution, number of bins, and approximate sample size
- Unfolded error matrix includes effect of finite MC training statistics (usually small)
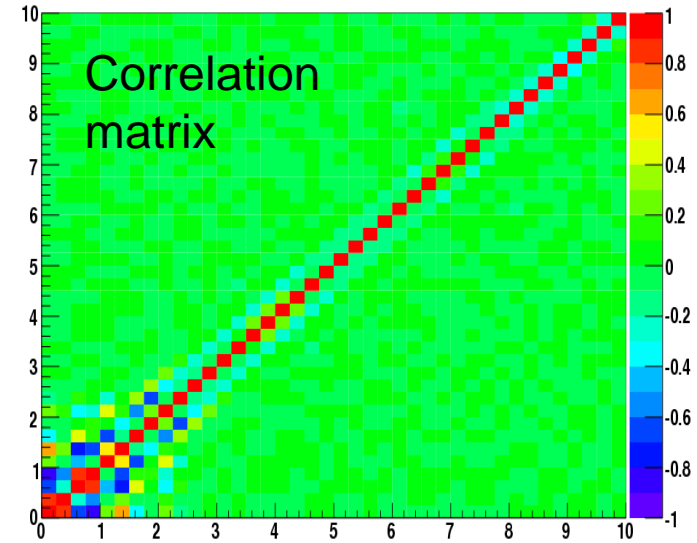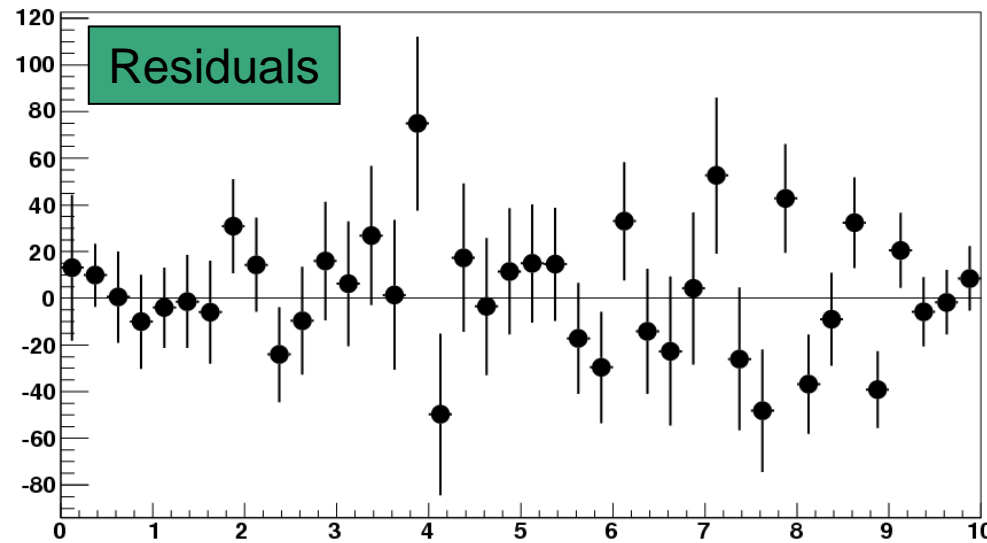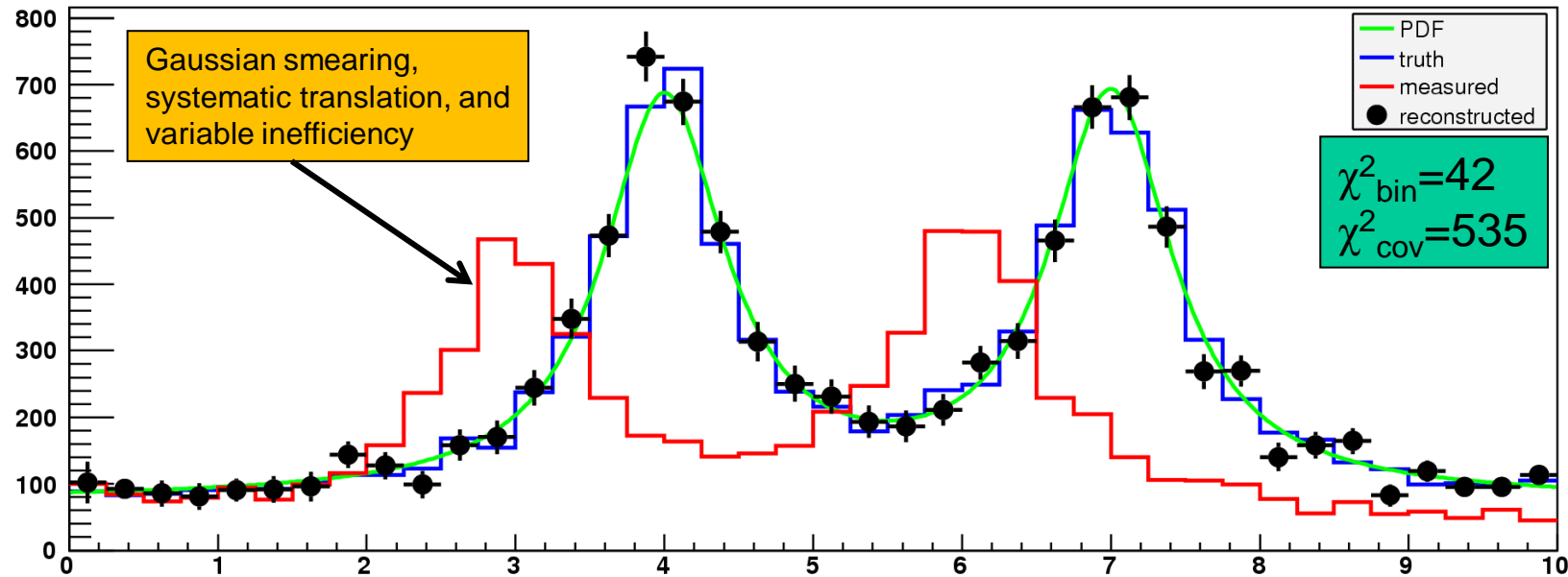
- Uses the TUnfold method implemented by Stefan Schmitt and included in ROOT
  - RooUnfold includes an interface to this class

- Performs a matrix inversion with 0-, 1-, or 2-order polynomial regularisation of neighbouring bins
  - RooUnfold automatically takes care of packing 2D and 3D distributions and creating the appropriate regularisation matrix required by TUnfold
- TUnfold can determine an optimal regularisation parameter (τ) by scanning the "L-curve" of $\log_{10}(\chi^2)$ vs $\log_{10}(\tau)$.
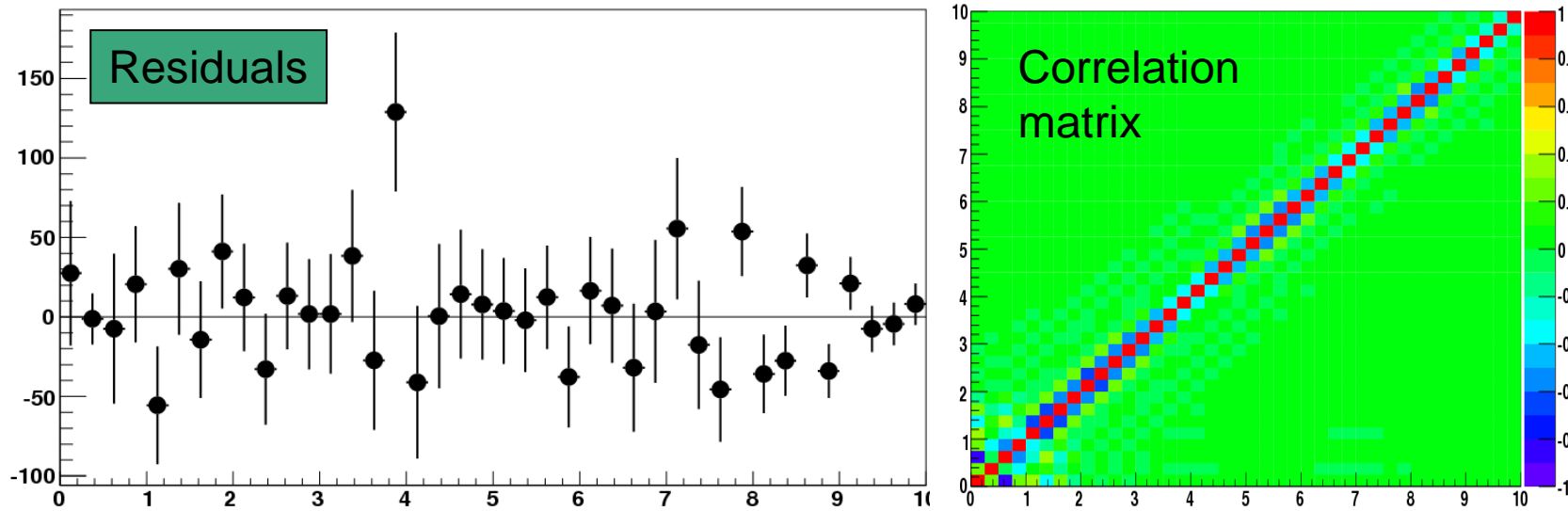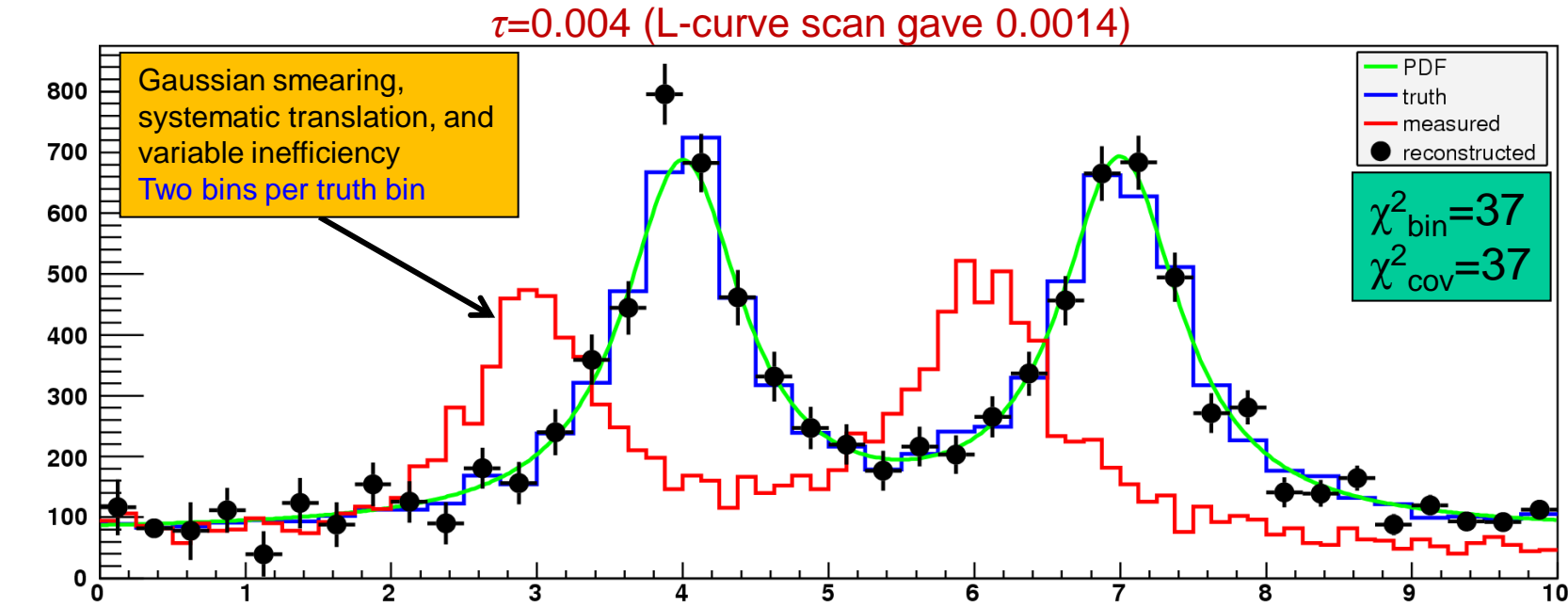
- Very simple algorithms
  - using bin-by-bin correction factors, with no inter-bin migration
  - using unregularised matrix inversion with singular value removal (TDecompSVD)

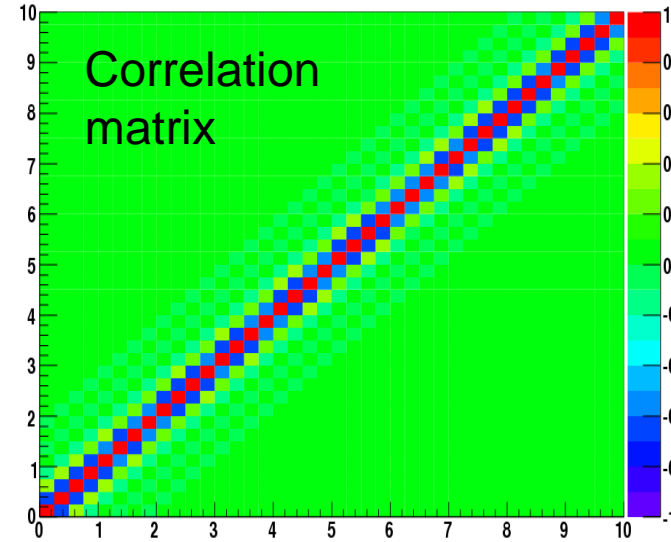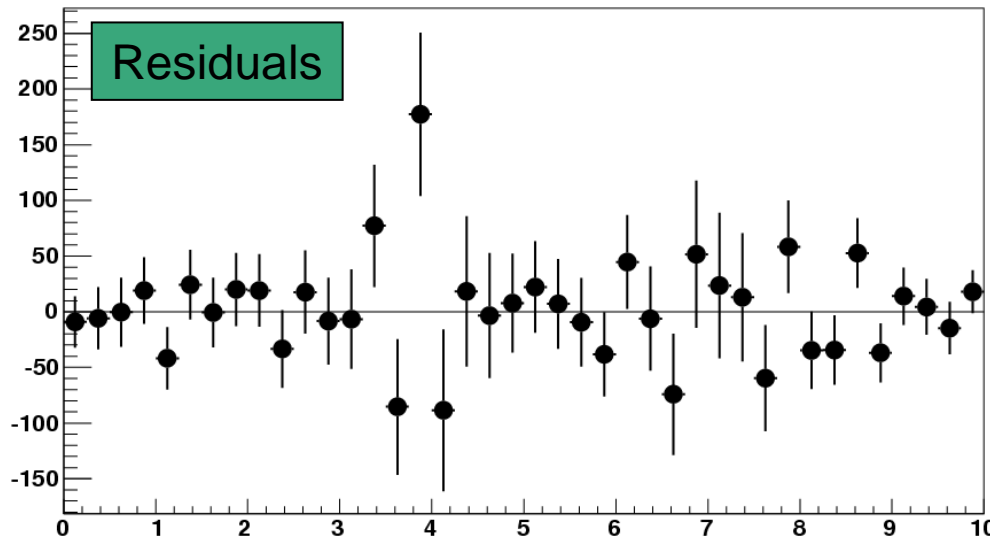 are included for comparison – and to demonstrate why they should not be used in most cases!

3 iterations

Gaussian smearing, systematic translation, and variable inefficiency

PDF
truth
measured
reconstructed

$\chi^2_{bin}=47$
$\chi^2_{cov}=9178$

Residuals

Correlation matrix

$k = 30$

Gaussian smearing, systematic translation, and variable inefficiency

PDF
truth
measured
● reconstructed

$\chi^2_{bin}=42$
$\chi^2_{cov}=535$

Residuals

Correlation matrix

τ=0.004 (L-curve scan gave 0.0014)

Gaussian smearing, systematic translation, and variable inefficiency
Two bins per truth bin

PDF
truth
measured
reconstructed

χ^2_{bin}=37
χ^2_{cov}=37

Residuals

Correlation matrix

Gaussian smearing, variable inefficiency, and **no** systematic translation

PDF
truth
measured
● reconstructed

$\chi^2_{bin}=30$
$\chi^2_{cov}=34$

Residuals

Correlation matrix

Gaussian smearing, systematic translation, and variable inefficiency
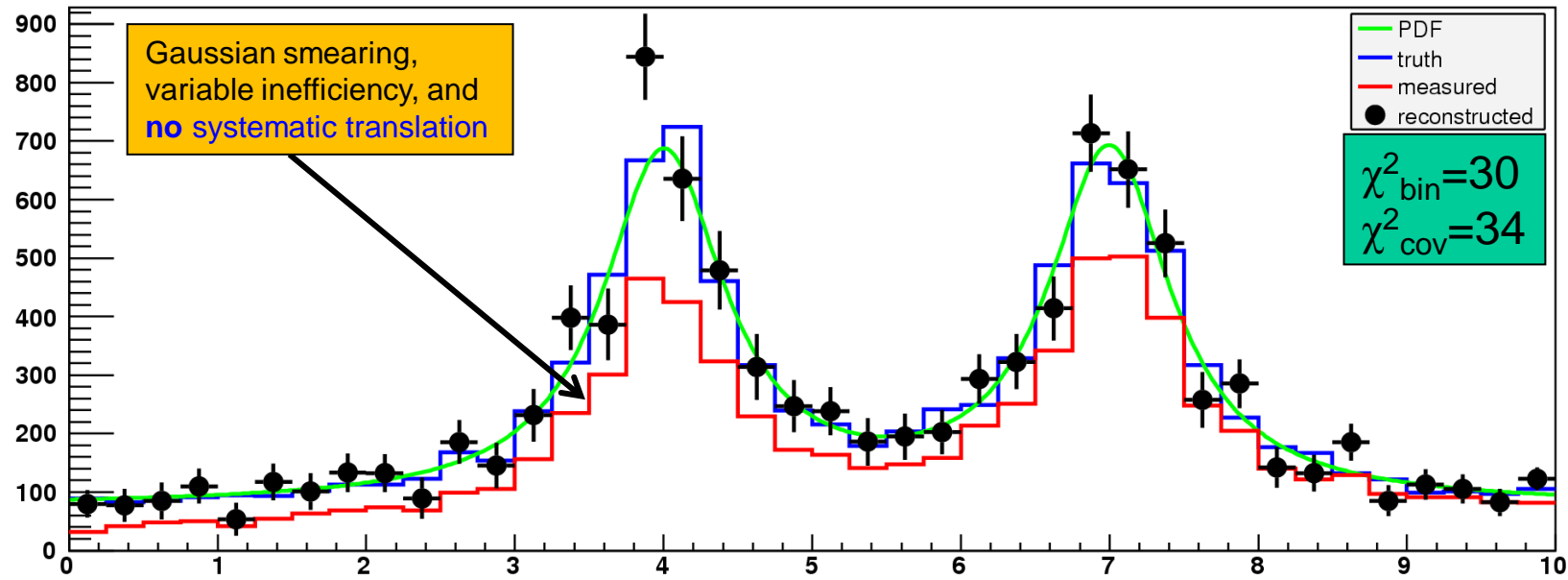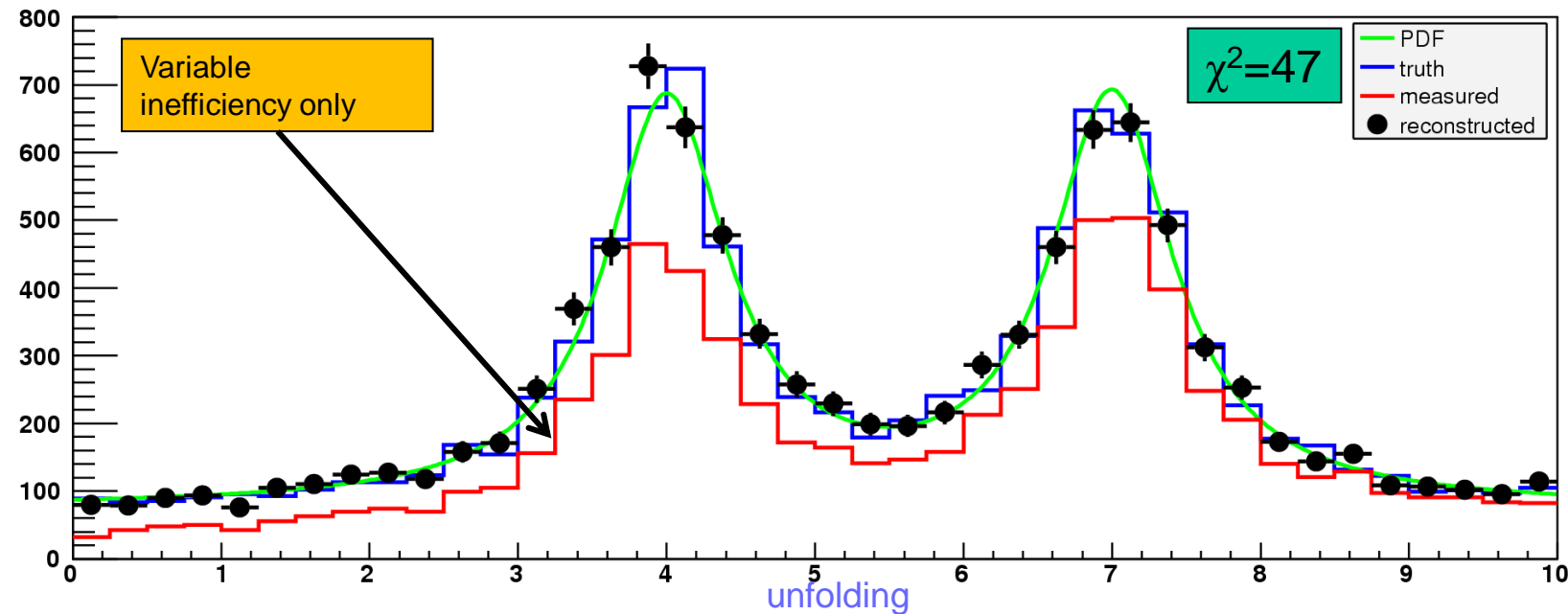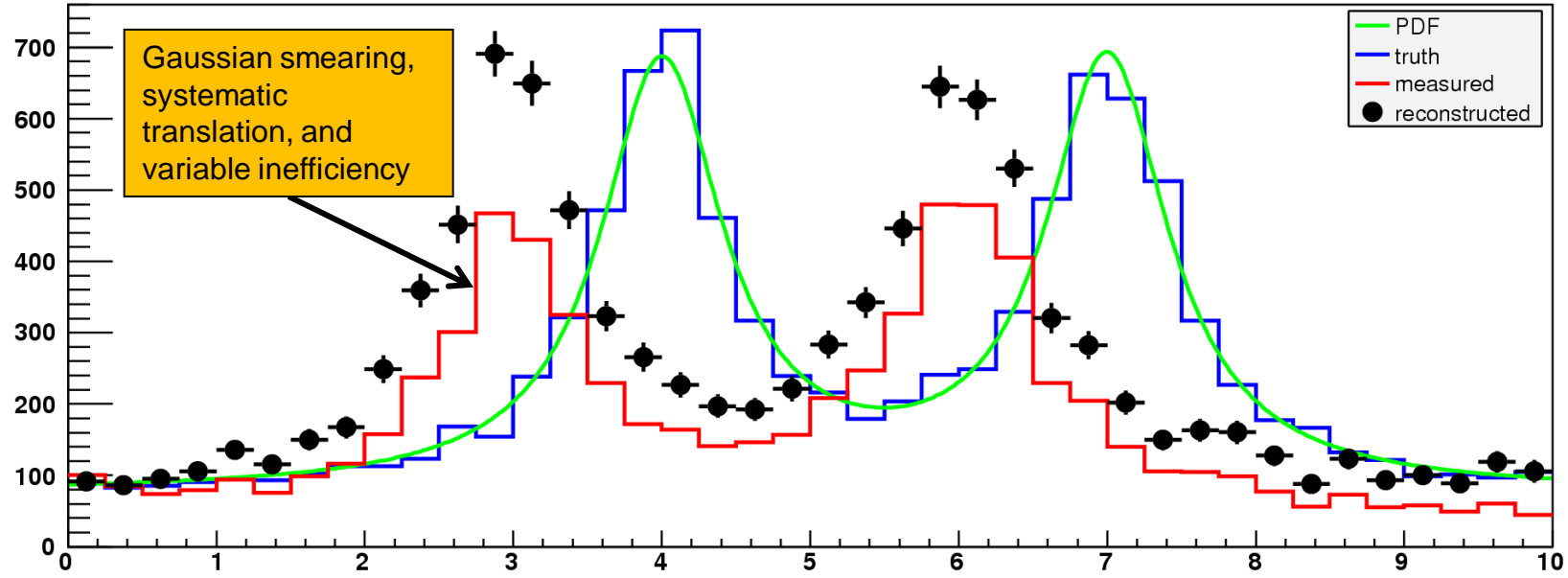
Variable inefficiency only

$\chi^2=47$

- Regularisation introduces inevitable correlations between bins in the unfolded distribution
  - To calculate a correct χ^2, one has to invert the covariance matrix:
    $$\chi^2 = (x_m - x_t)^T \, \mathbf{V}^{-1} \, (x_m - x_t)$$
- However, in many cases, the covariance matrix is poorly conditioned, which makes calculating the inverse problematic
  - Inverting a poorly conditioned matrix involves subtracting large, but very similar numbers, leading to significant effects due to the machine precision

- In any case, χ^2 may not be the best figure of merit
  - could improve χ^2 by relaxing regularisation $\rightarrow$ larger errors, but also larger residuals
  - Is there a better figure of merit?