Science & Technology Facilities Council
Rutherford Appleton Laboratory

Unfolding algorithms and tests using RooUnfold

Tim Adye
Rutherford Appleton Laboratory

PHYSTAT 2011
Workshop on Unfolding and Deconvolution
20th January 2011

Outline

1. RooUnfold aims, design, and features
2. Example of unfolding
3. Summary of the algorithms
4. Comparison of algorithms
5. Unfolding errors
6. Status and Plans
7. References

RooUnfold package aims

- Provide a framework for different algorithms
  - Can compare performance directly, with common user code
    - RooUnfold takes care of different binning, normalisation, efficiency conventions
  - Can use common RooUnfold utilities
    - Write once, use for all algorithms
  - Currently implement or interface to iterative Bayes, SVD, TUnfold, unregularised matrix inversion, and bin-by-bin correction factors algorithms
- Simple OO design
  - "response matrix" object can be filled separately from training sample
    - in a different routine, or a different program (**ROOT I/O** support)
- Simple interface for the user
  - From program, **ROOT/CINT** script, or interactive **ROOT** prompt
  - Fill with histograms, vectors/matrices,… or direct methods:
    - **`response->Fill(`**x_{measured}**`,`** x_{true}) and **`Miss(`**x_{true}**`)`** methods takes care of normalisation
  - Results as a histogram with errors, or vector and covariance matrix

RooUnfold features

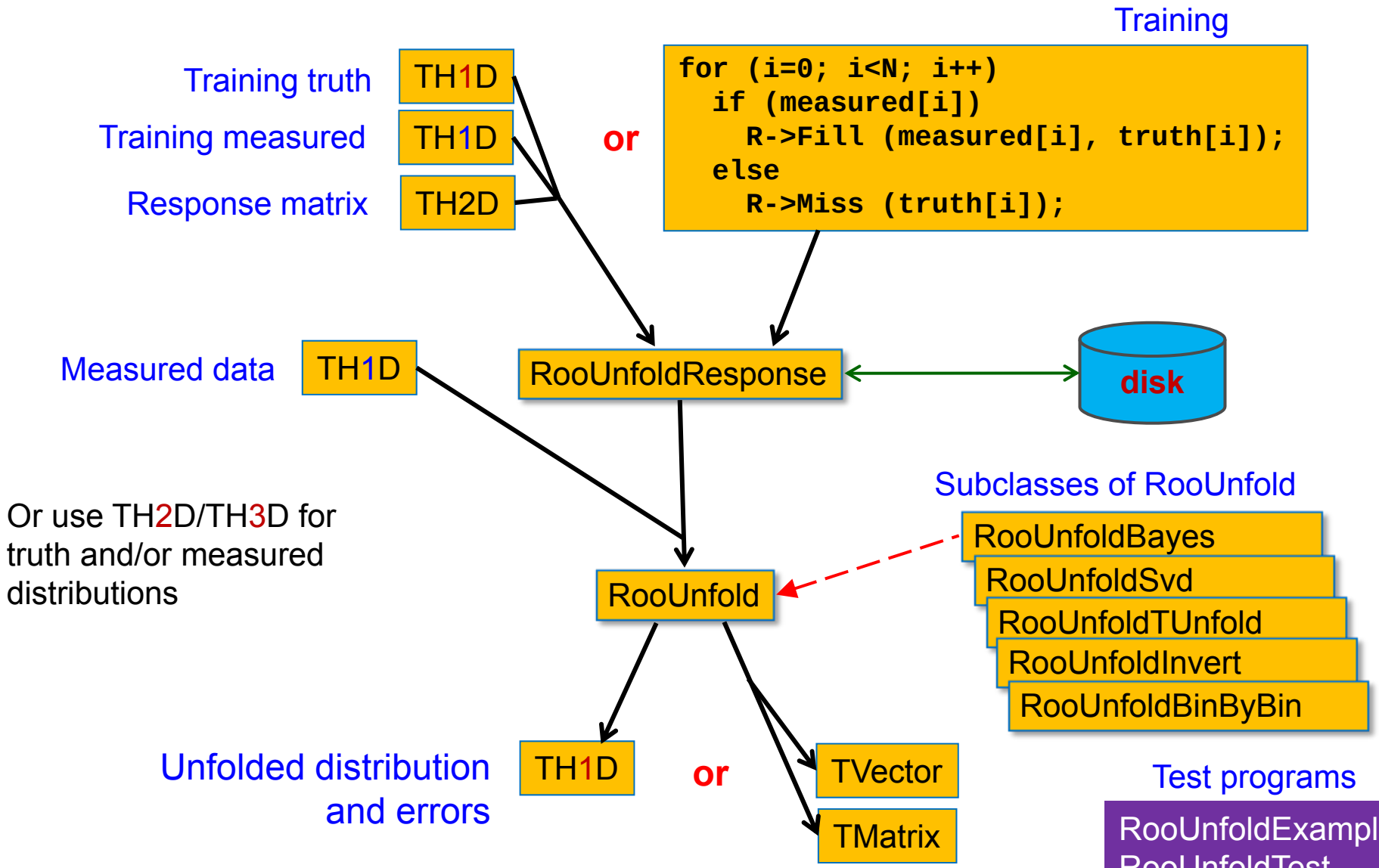
- Supports different binning scenarios
  - multi-dimensional distributions (1D, 2D, and 3D)
  - Different binning (or even dimensionality) for measured and truth
  - Option to include or exclude histogram under/overflow bins in the unfolding
- Supports different methods for error computation (simple switch). In order of increasing CPU time:
  - No error calculation (uses $\sqrt{N}$)
  - bin-by-bin errors (no correlations)
  - full covariance matrix from the propagation of measurement errors in the unfolding, or
  - covariance matrix from MC toys
    - useful to test error propagation and when it is inaccurate
- These details are handled by the framework, so don't need to be implemented for each algorithm

Testing

- Calculates resolutions, pulls, and χ^2
- Includes a toy MC test framework, allowing selection of different
  - PDFs and PDF parameters
  - binning
  - 1D, 2D, 3D tests
  - unfolding methods and parameters
  - Test procedures for the regularisation parameter and errors

  and plotting results from a single command

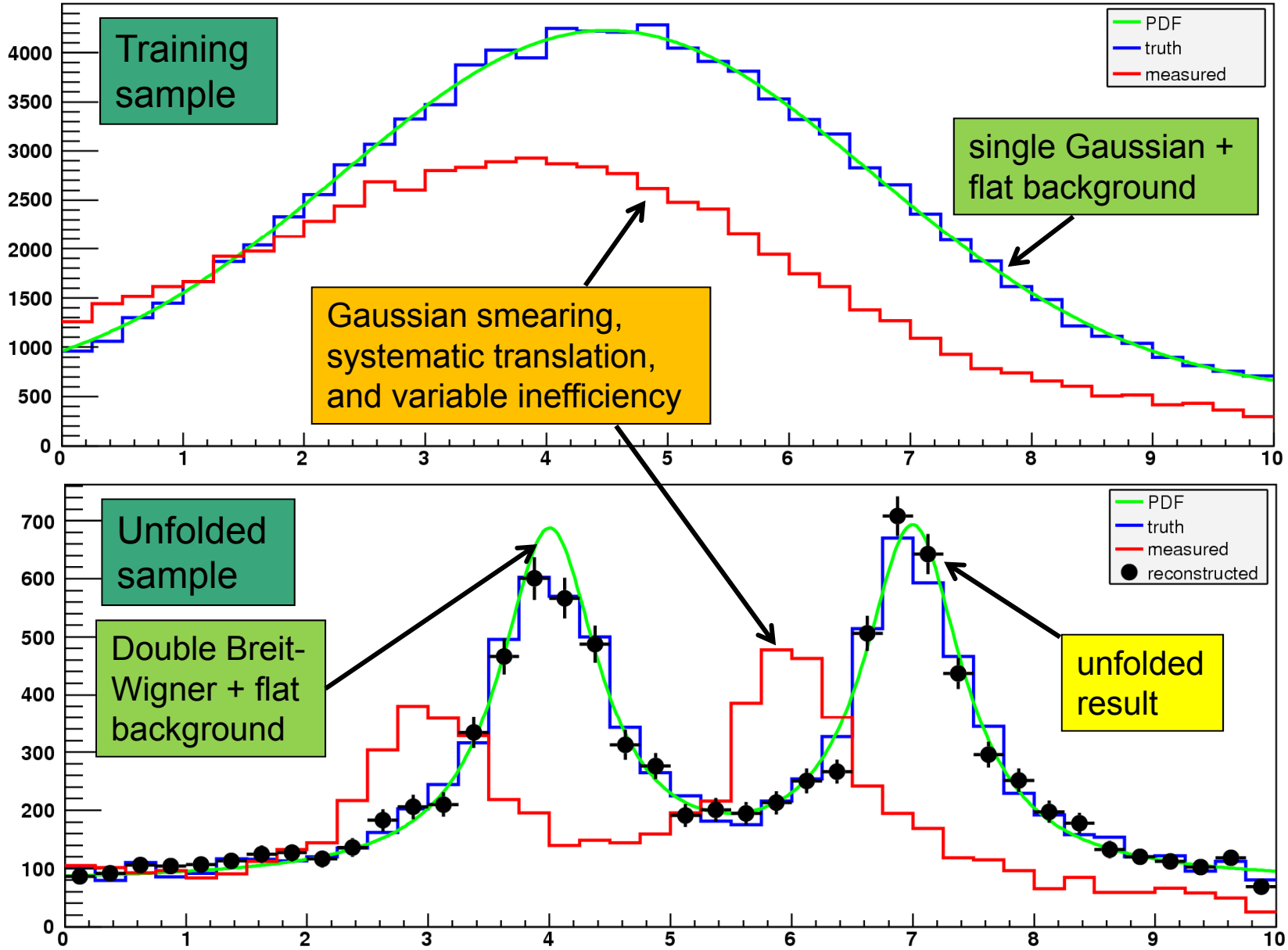
RooUnfold Classes

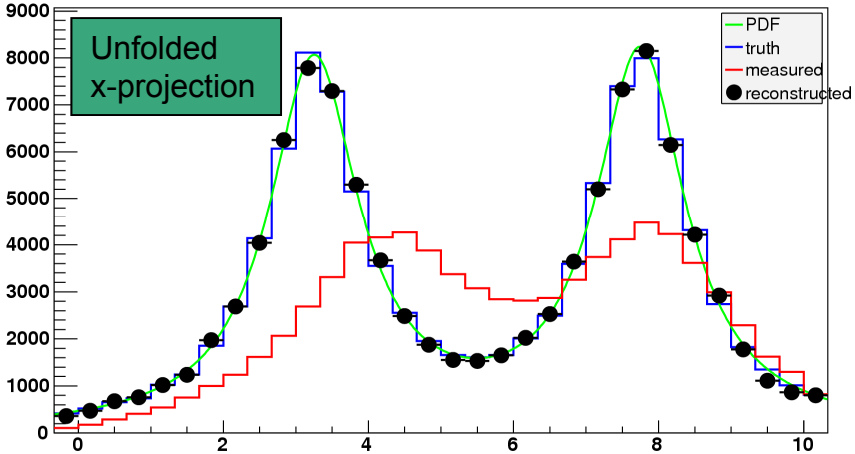


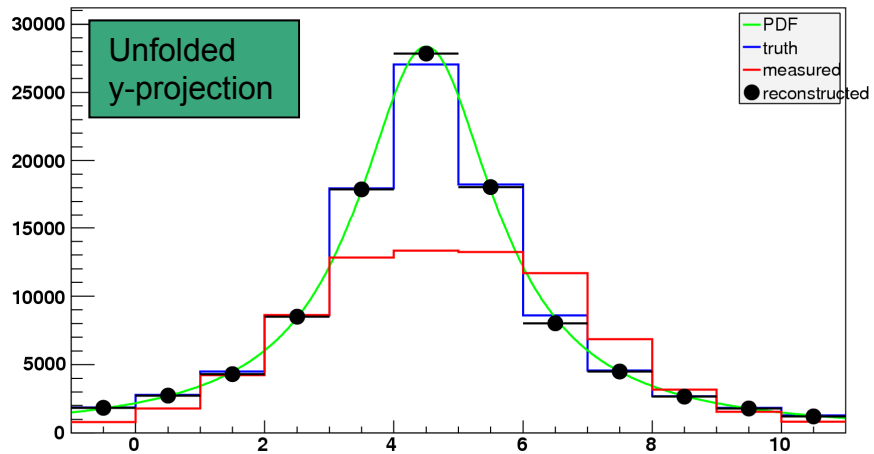
Test programs

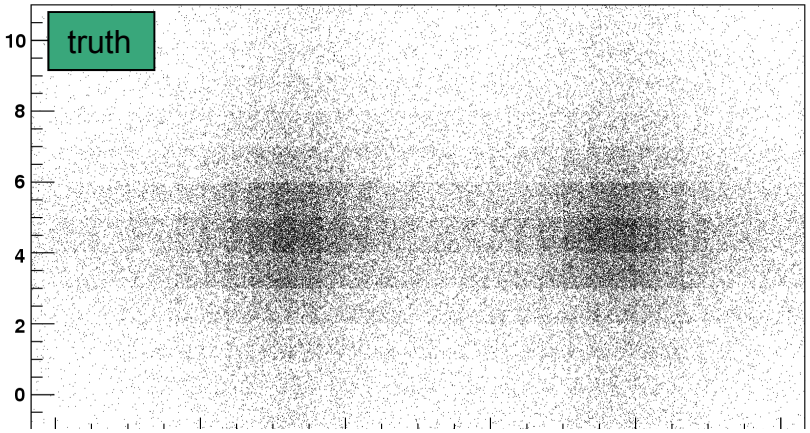
RooUnfoldExample
RooUnfoldTest
RooUnfoldTest2D
RooUnfoldTest3D

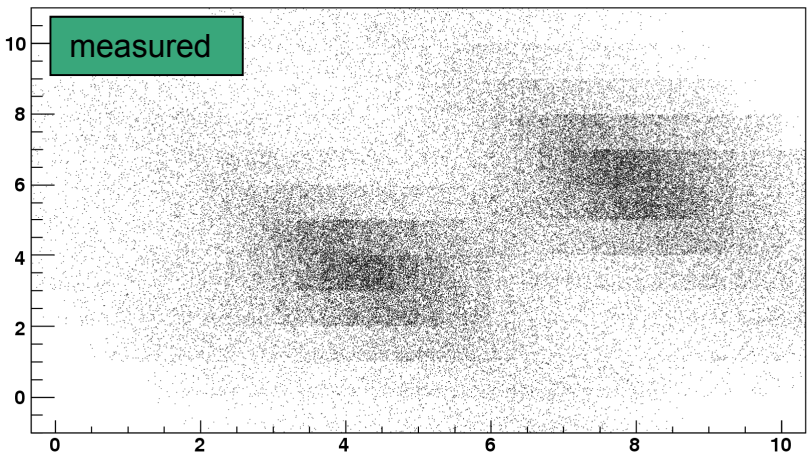
Unfolding example (Bayes)

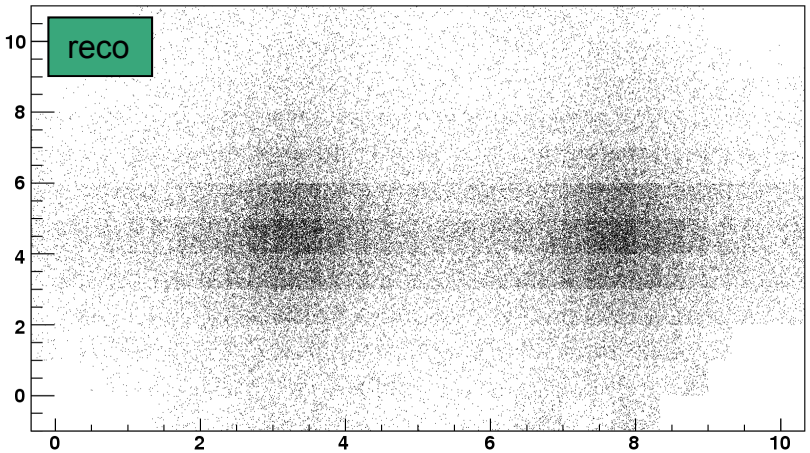












2D unfolding

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

Algorithms – Iterative Bayes' theorem

- Uses the method of Giulio D'Agostini (1995), implemented by Fergus Wilson and myself → *see talk by Katharina Bierwagen for details of the method*
  - 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
  - Takes account of multinomial errors on the data sample but not, by default, uncertainties in the response matrix (finite MC statistics), which is very slow
  - Does not normally do smoothing (can be enabled with an option)

Algorithms – SVD

- Uses the method of Andreas Höcker and Vato Kartvelishvili (**GURU** Fortran program) → *see Vato's talk for details of the method*
  - Implemented in C++/ROOT by Kerstin Tackmann and Heiko Lacker and included in the most recent version of ROOT (5.28) as TSVDUnfold → *see Kerstin's talk for details of the method*
    - 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)

Algorithms – TUnfold

- 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 $\log_{10}(\tau)$.

Unregularised Algorithms

- Very simple algorithms
  - using bin-by-bin correction factors, with no inter-bin migration
  - using unregularised matrix inversion with singular value removal (TDecompSVD)

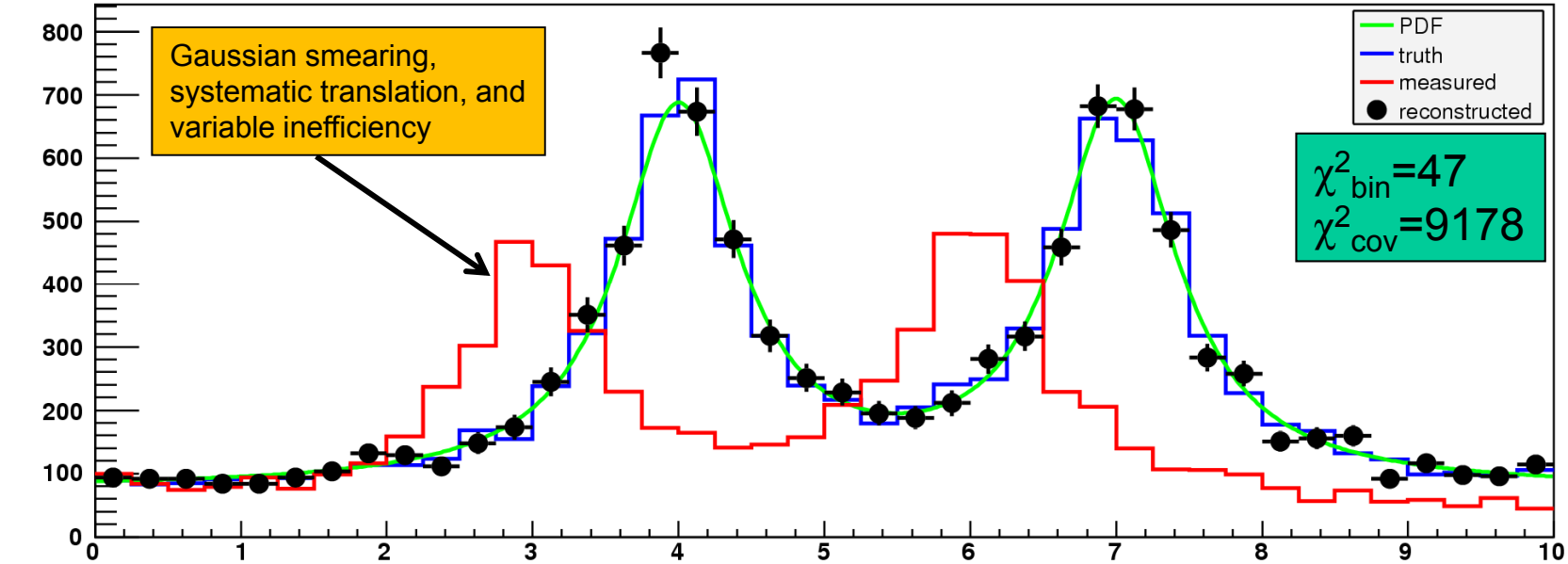
  are included for comparison – and to demonstrate why they should not be used in most cases!

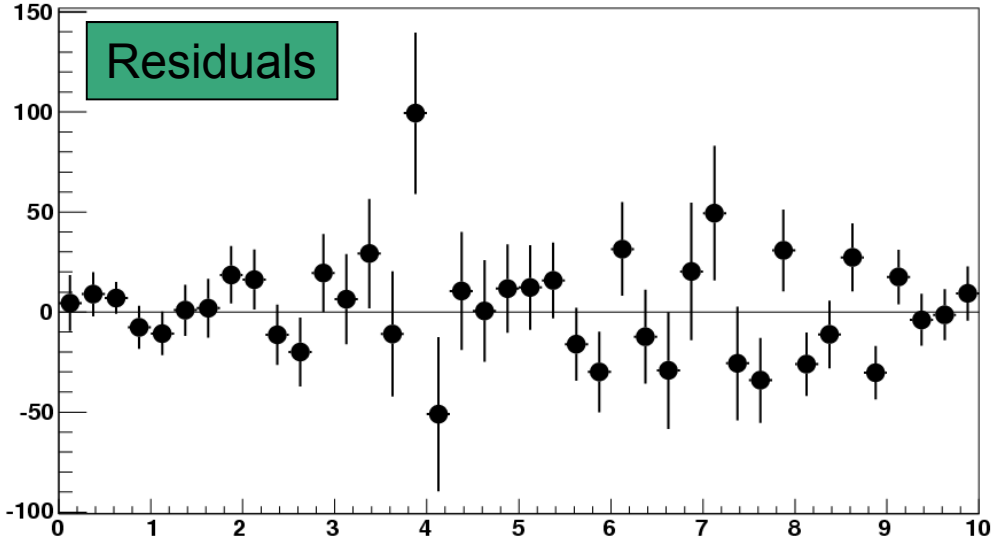
Comparison of algorithm features

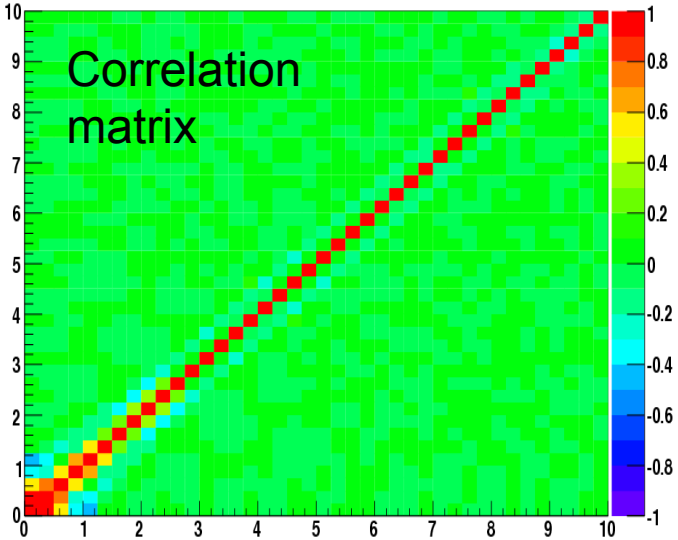
- TUnfold and unregularised matrix inversion require the number of bins, $N_{measured} \geq N_{true}$
  - TUnfold claims best results if $N_{measured} > N_{true}$, eg. $N_{measured} = 2N_{true}$
    - This is a common general recommendation from unfolding experts, but perhaps is most relevant to these types of algorithms with explicit regularisation
    - This is an implicit additional regularisation, since we are “smoothing” two bins into one
- SVD implementation and bin-by-bin methods only support $N_{measured} = N_{true}$
  - SVD implementation also only works well for 1D distributions
- The choice of the SVD regularisation parameter has to be done by the user
  - TUnfold can often do this automatically
    - Can we do something similar for the SVD method?
  - The performance of the Bayes method is relatively insensitive to the regularisation parameter (number of iterations)

RooUnfold with Bayes algorithm

3 iterations

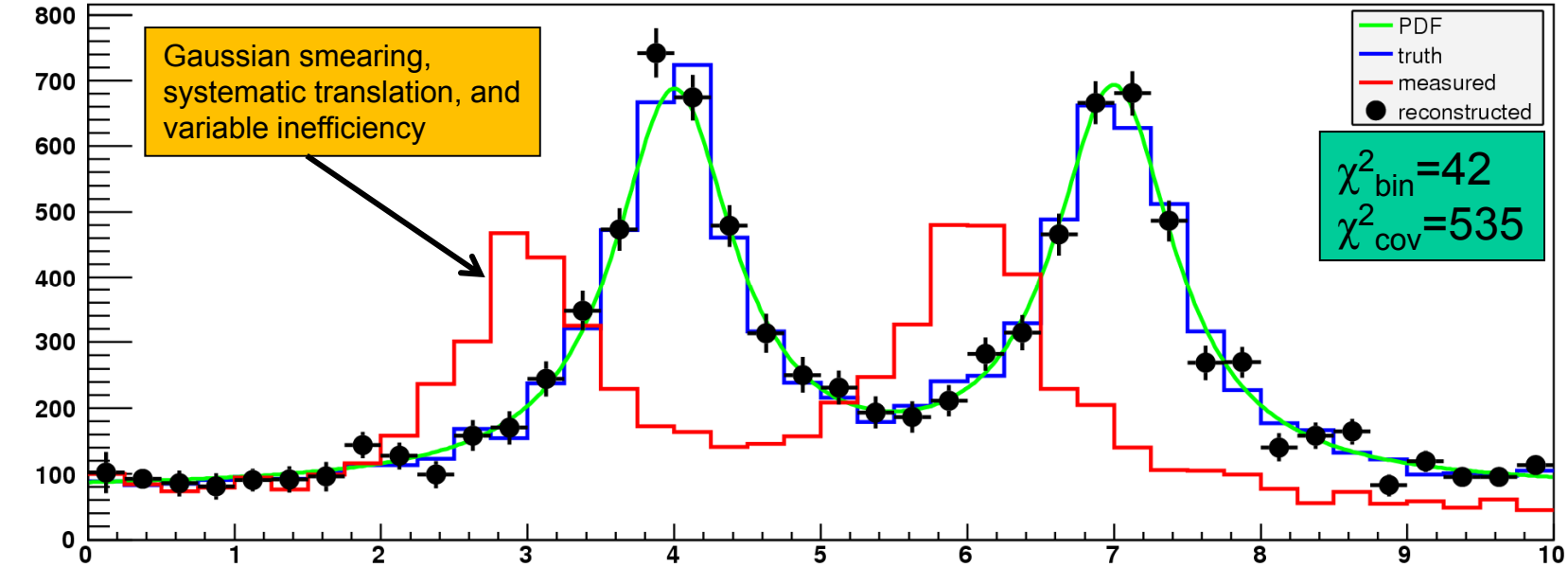


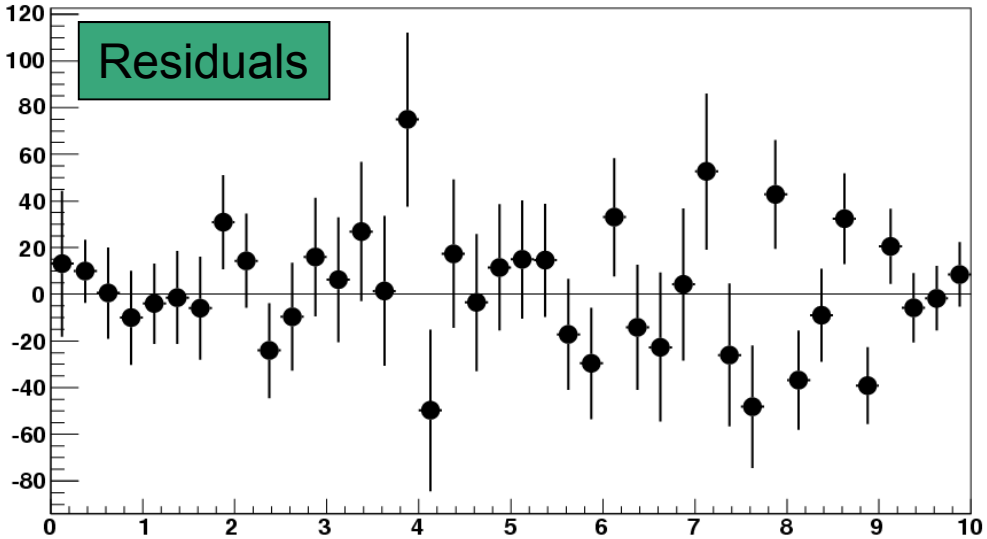


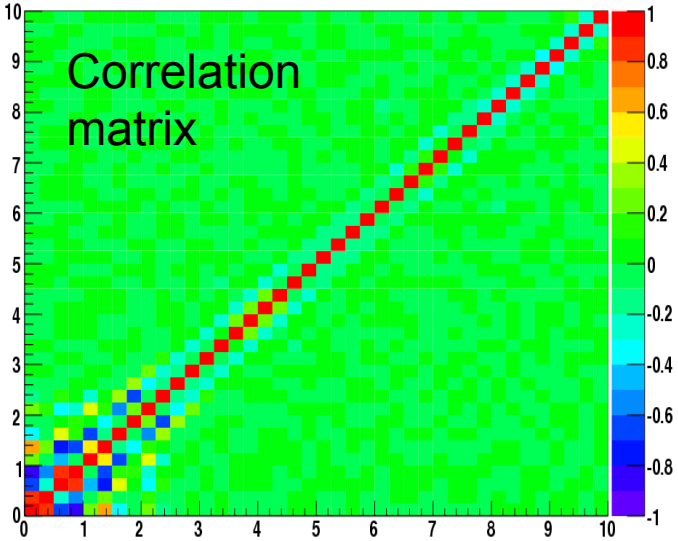


RooUnfold with SVD algorithm

k = 30

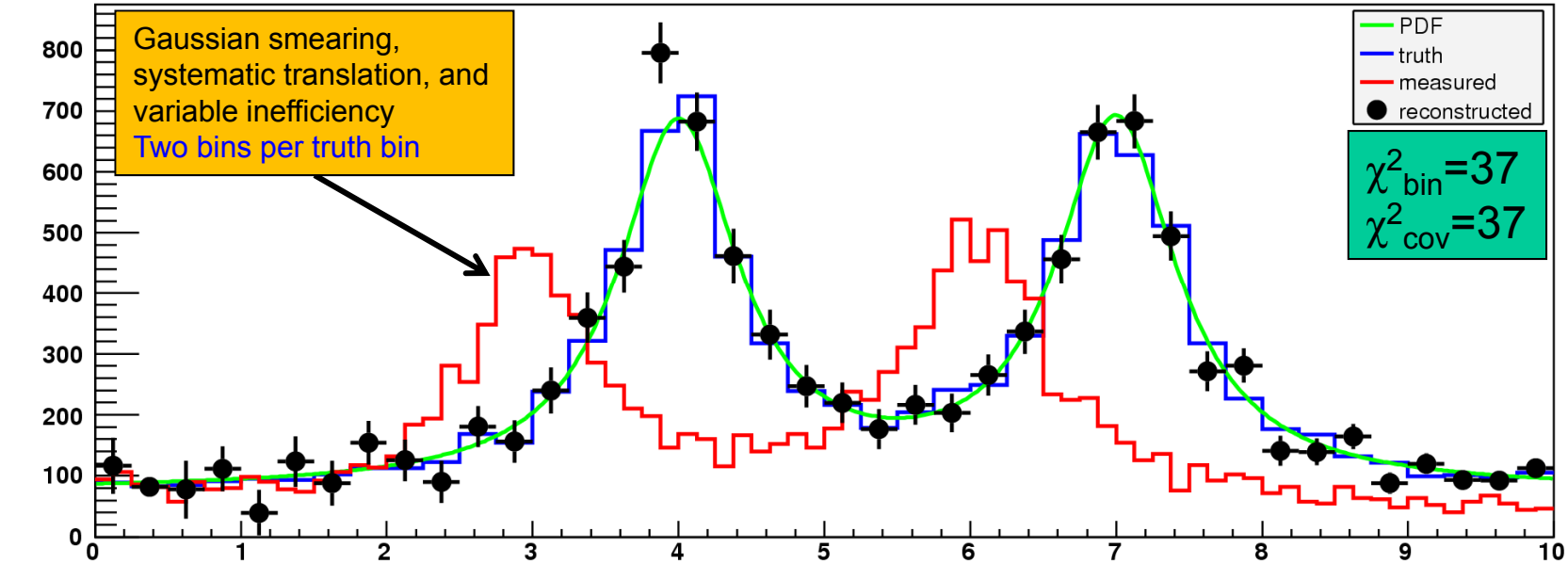


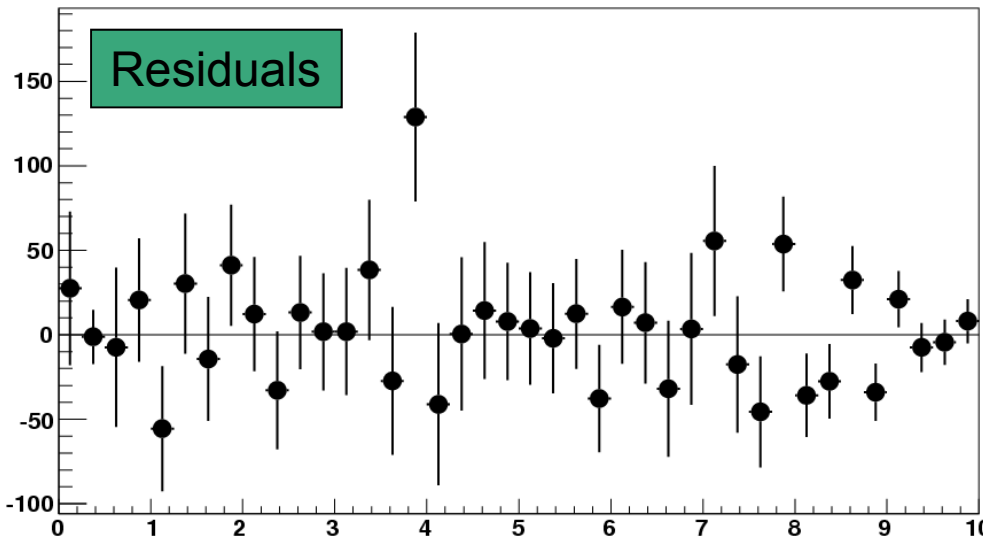


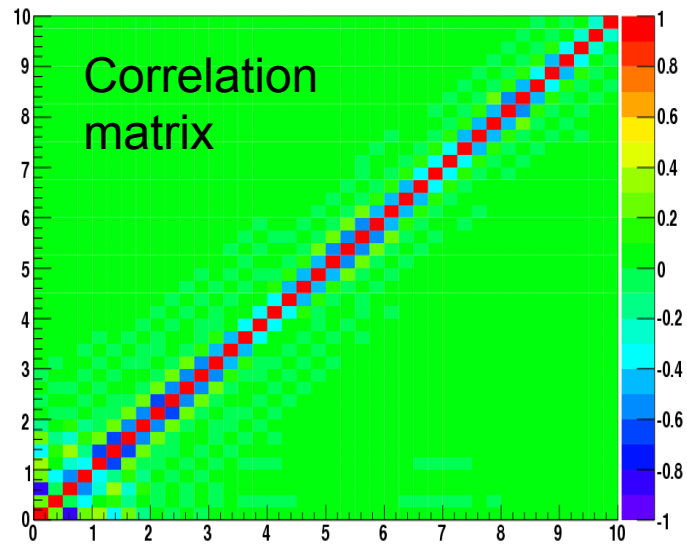


RooUnfold with TUnfold algorithm

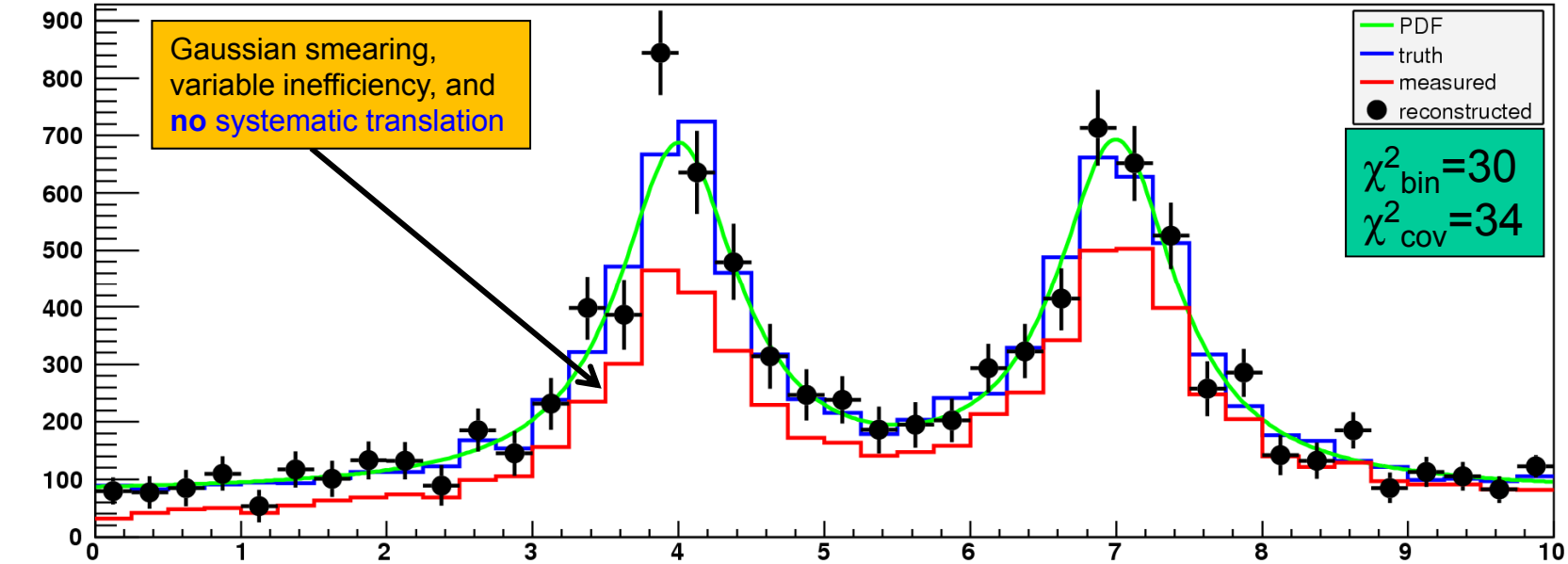
τ=0.004 (L-curve scan gave 0.0014)

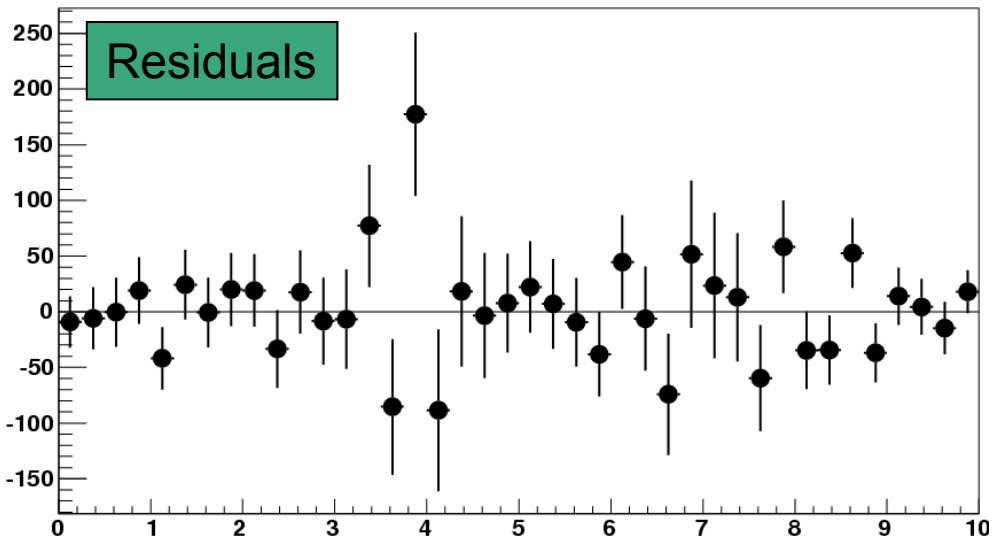


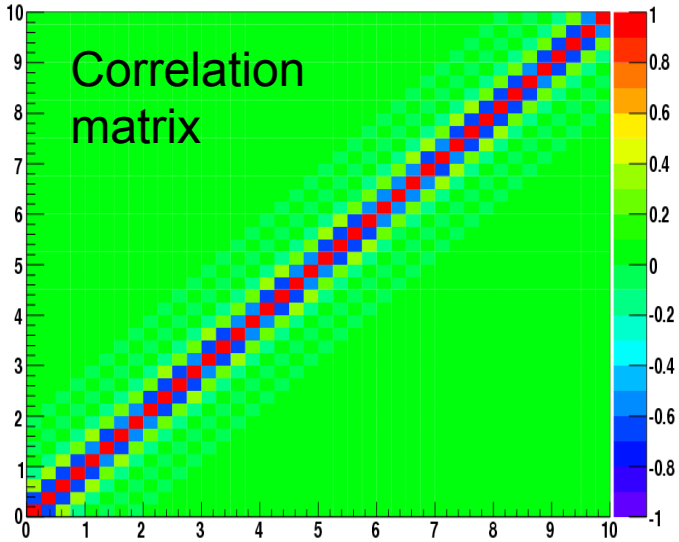




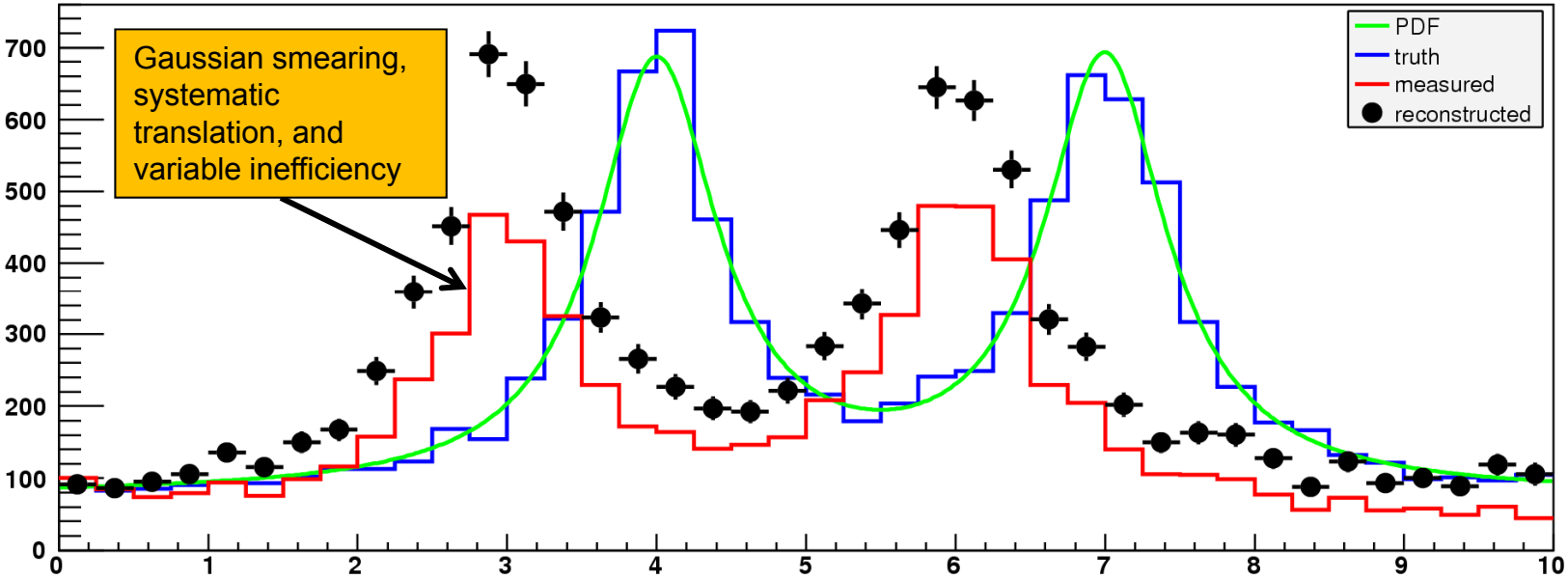
Unregularised matrix inversion

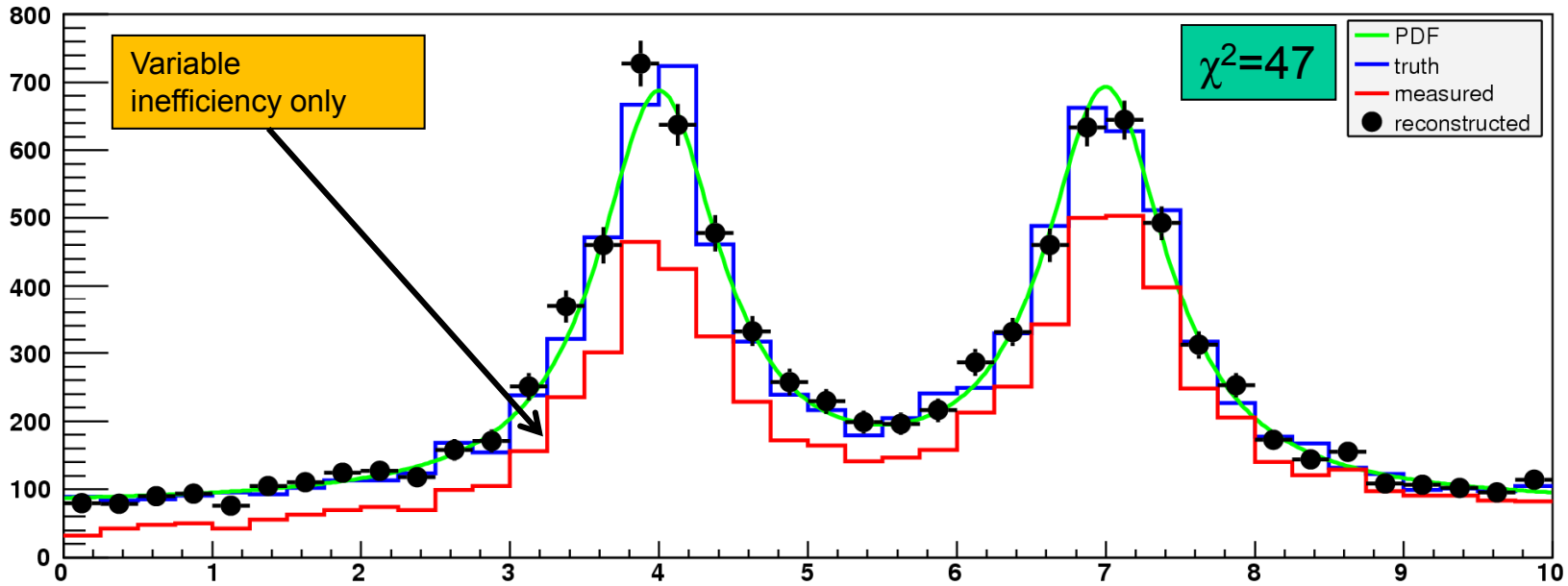






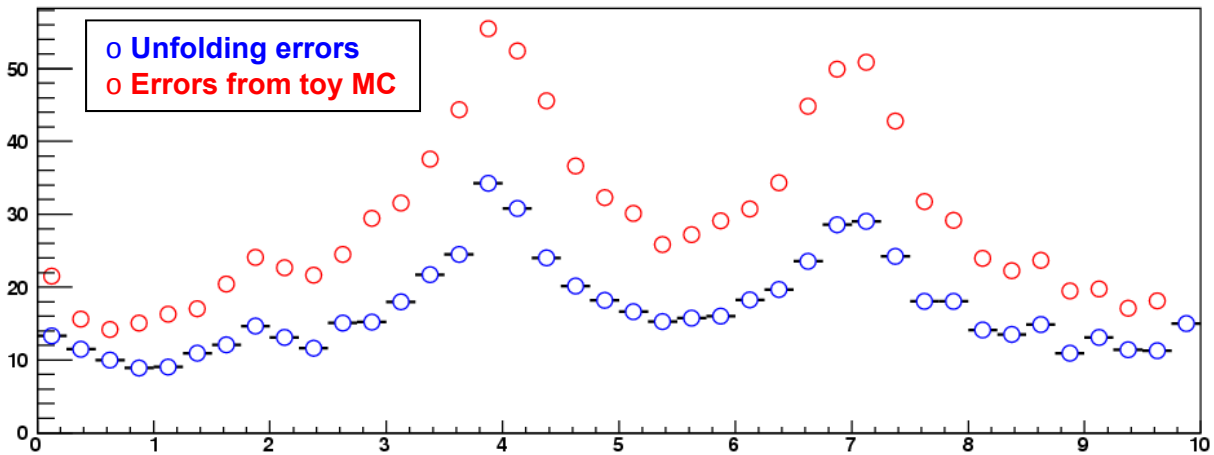
simple correction factors





Unfolding errors

- All methods return a full covariance matrix of the errors on the unfolded histogram due to uncertainties on the measured distribution.
  - This is often calculated by propagation of errors
    - but not always possible if there are non-linearities or other problems, eg. the iterations in the Bayes method are not handled in D'Agostini's formalism:



- RooUnfold allows the covariance matrix to be calculated from toy MC instead
  - provides a cross-check of the error propagation or replace it if there are problems

Bin-to-bin correlations

- 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?

RooUnfold Status

- RooUnfold was first released stand-alone (outside the BaBar framework) in 2007
  - Have received many questions, suggestions, and even a few bug reports
    - ~60 people working on many different experiments in PP, PA, and NP
    - Prompted new versions with fixes and improvements

- Last year I started working with a small group hosted by the Helmholtz Alliance, the Unfolding Framework Project
  - The project is developing unfolding experience, software, algorithms, and performance tests
    - Also developing and running a “Banff challenge”-esque example
  - It has adopted RooUnfold as a framework for development

Plans and possible improvements

1. Incorporate into the next version of ROOT
   - Encouraging discussions with Lorenzo Moneta from the ROOT team

2. More common tools, useful for all algorithms
   - Further automate validation and selection of regularisation parameter
   - Allow propagation of errors from the response matrix equally in all algorithms and with toy MC

3. More algorithms
   - Interface to TTRUE (Natalie Milke), which is a C++ implementation of Volker Blobel's RUN method
     - Requires unbinned MC responses
   - Interface to Iterative Dynamically Stabilized method of Bogdan Malaescu
   - ...?

References

RooUnfold code, documentation, and references to unfolding reviews and techniques can be found on this web page

http://hepunx.rl.ac.uk/~adye/software/unfold/RooUnfold.html