RooUnfold
unfolding framework
and algorithms

Tim Adye
Rutherford Appleton Laboratory

Alliance Workshop on
Unfolding and Data Correction
27-28th May 2010
Outline

1. RooUnfold aims and features
2. C++ Classes
3. Summary of the algorithms
4. Some toy MC examples
5. Status and Plans
6. References
RooUnfold package aims

- Provide a framework for different algorithms
  - Can compare performance directly
  - Common utilities can be (have been) implemented
    - Write once, use for all algorithms
  - Currently implement iterative Bayes, SVD, and simple correction factors
    - Interfaced to borrowed, adapted, and improved routines
    - Plan to add more
- 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)
  - Implementations inherit from RooUnfold base class
- 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 take 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 for measured and truth
  - Variable bins
  - Option to include or exclude under/overflow bins
These details are handled by the framework, so don’t need to be implemented for each algorithm
  - does not work well for algorithms that rely on global shape (SVD)
- Calculates resolutions, pulls, and $\chi^2$
- Includes a toy MC test framework, allowing selection of different
  - PDFs and PDF parameters
  - binning
  - 1D, 2D, 3D tests
  - unfolding methods and parameters
and plotting results from a single ROOT command
RooUnfold Classes

Training truth
Training measured
Response matrix

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

TH1D
TH2D
TH3D

Measured data

RooUnfoldResponse

Subclasses of RooUnfold
RooUnfoldBayes
RooUnfoldSVD
RooUnfoldBinByBin...

TH1D
TH2D
TH3D

Unfolded distribution and errors

Test programs
RooUnfoldExample
RooUnfoldTest
RooUnfoldTest2D
RooUnfoldTest3D

TVector
TMatrix

Tim Adye - RAL
Algorithms – Iterative Bayes’ theorem

• 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
  • 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 Vakhtang Kartvelishvili (GURU Fortran program)
  - Implemented in C++/ROOT by Kerstin Tackmann and Heiko Lacker (originally for BaBar $B \rightarrow Xu_\nu$ paper)

- 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 $\rightarrow$ result dominated by MC truth
      - $k$ too large $\rightarrow$ 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 – simple correction factors

• A very simple algorithm using bin-by-bin correction factors, with no inter-bin migration is included to aid comparison
RooUnfold with Bayes algorithm
3 iterations

Training sample

single Gaussian + flat background

Gaussian smearing, systematic translation, and variable inefficiency

Unfolded sample

Double Breit-Wigner + flat background

unfolded result

PDF
truth
measured
reconstructed
RooUnfold with SVD algorithm

$k = 20$

- **Training sample**
- Single Gaussian + flat background
- Gaussian smearing, systematic translation, and variable inefficiency
- **Unfolded sample**
- Double Breit-Wigner + flat background
- unfolded result
simple correction factors
2D unfolding with Bayes method

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

- RooUnfold’s authors are Kerstin Tackmann, Fergus Wilson, and myself.
  - I have continued most of the development, with help and advice from Kerstin and Fergus.

- RooUnfold was first released stand-alone (outside the BaBar framework) in 2007
  - Have received many questions, suggestions, and even a few bug reports
    - ~40 people from many different collaborations in PP, PA, and NP
    - Prompted new versions with fixes and improvements

- However it has mostly been a spare-time activity for me so development has been slow
  - I should be able to spend a little more time in future
    - if this is deemed a useful avenue to pursue (eg. in this workshop)
Plans and possible improvements

1. Additional input/output methods to simplify use, eg. RooFit datasets

2. Add common tools, useful for all algorithms
   - Automate validation and selection of regularisation parameter
   - Ensemble tests (follow work of Katharina Bierwagen)

3. More algorithms
   - Should be easy to provide an interface to another implementation

4. Publish as an official package
   - Incorporation into RooStats under discussion
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