LANCASTER UNIVERSITY

Unfolding with Singular Value Decomposition (SVD)

V. Kartvelishvili (Lancaster, UK)

PHYSTAT2011, Unfolding session, 20 January 2011

Outline

- Basic ideas
- Examples, old and new
- SVD view on fitting: folding vs unfolding
- Random bits of advice for unfolders

Basic notation

- Grateful to previous speakers, in general for their contributions to the field
- ...and today, for saving me from the need for a lengthy introduction

- **Just a few definitions:** An initial Monte carlo sample was used to create $\hat{A}_{ij}$, matrix simulating detector response: probability for an event generated in the true bin j to be found in measured bin i.

$$\hat{A}_{ij}\, {x_j}^{\rm ini} = {b_i}^{\rm ini} \qquad \text{exact}$$

b_i — vector (histogram) of measured values
x_j — vector (histogram) of true values

- Our problem is to determine x from a measured distribution b, given $\hat{A}_{ij}$ and some additional information on expected properties of x, e.i. find a meaningful way of solving the system

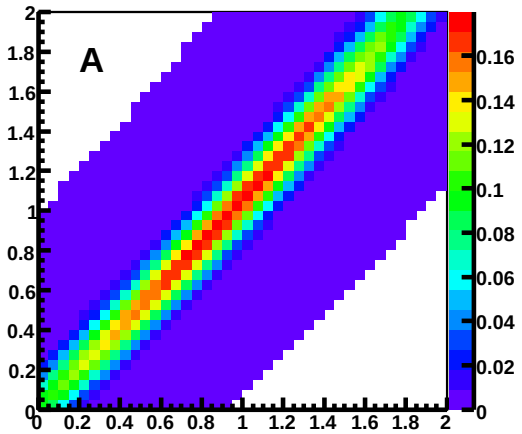
$$\hat{A}x = b$$

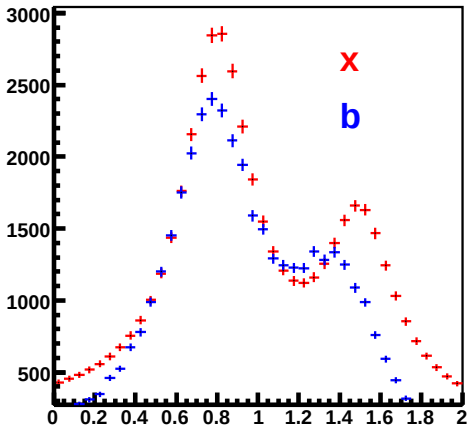
The problem

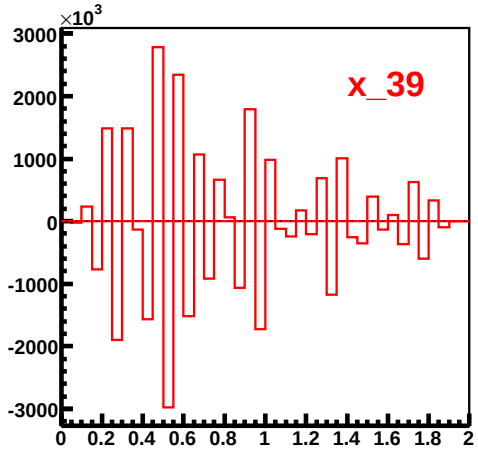
The problem is twofold:

- b is known with some precision
  In many cases covariance matrix $B = \mathrm{diag}\{b\}$
- Our knowledge of $\hat{A}$ is not perfect either, due to finite MC statistics, as well as imperfections in detector simulation. Even worse, $\hat{A}$ is almost always very close to being highly degenerate, so solving the system exactly does not make any sense.

An attempt to solve the problem directly and "exactly" will end up looking like this:







What is SVD?

A Singular Value Decomposition of a real $m \times n$ matrix A is its factorization of the form

$$A = U\,S\,V^T$$

U is an $m \times m$ orthogonal matrix, $U\,U^T = U^T U = I$.

V is an $n \times n$ orthogonal matrix, $V\,V^T = V^T V = I$.

S is an $m \times n$ diagonal matrix with non-negative diagonal elements:

$$S_{ij} = 0 \;\; \text{for} \;\; i \neq j, \quad S_{ii} \equiv s_i \geq 0.$$

s_i are called singular values of the matrix A. By swapping rows of U and V, s_i can be ordered from largest to smallest.

Columns of U and V are called the left and right singular vectors. They define convenient ortho-normal bases in their respective spaces.

Examples:

If A itself is orthogonal, all $s_i = 1$.

If A is degenerate, at least one $s_i = 0$.

Why is SVD useful?

With SVD, a solution of a linear system $Ax = b$ looks very simple:

$$U\,S\,V^T = b \quad \Rightarrow \quad z \equiv V^T x, \quad d \equiv U^T b$$

$$s_i z_i = d_i \quad \Rightarrow \quad z_i = \frac{d_i}{s_i}$$

There are two reasons why the last step can go horribly wrong:

- Due of the errors in b, some d_i may not be significant at all
- Some s_i may be small (or even zero), thus exaggerating errors in d_i

Orthogonal matrices are totally harmless, so SVD allows to narrow the problem down to individual s_i and/or d_i.

Rescaling variables and equations

Before going any further, let's do some rescaling.

If the Monte Carlo and the data are fast-varying functions ranging many orders of magnitude, it makes sense to try and find x relative to the true distribution $x^{\rm ini}$ of the MC used for creating the matrix A:

- Multiply each column of A_{ij} by $x_i^{\rm ini}$ and simultaneously define new unknowns $w_i = x_i/x_i^{\rm ini}$
- This transformation has a "side-effect" of changing A from "probalility" to "number-of-events" matrix.
- The latter is arguably more useful, as bins with higher statistics, and hence more significance, are now enhanced.
- In general, if one equation is multiplied by a factor, its SVD changes.
- Clearly, an equation is made more prominent if it is multiplied by a large factor.
- One of the ideas of the GURU algorithm was to make sure that all the equations are "made equal" by making sure that the error in the r.h.s. is always ± 1.
- This is acheved by dividing each equation (i.e. each row of A_{ij} as well as b_j) by the error Δb_j.

Down to business

After all these manipulations, the original system $Ax = b$ has transformed into

$$\sum_j \tilde{A}_{ij} w_j = \tilde{b}_i$$

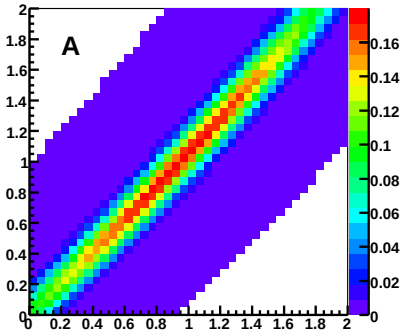
- Covariance matrix of the r.h.s. is equal to unit matrix
- Unknowns w_i are defined relative to the input MC distribution.

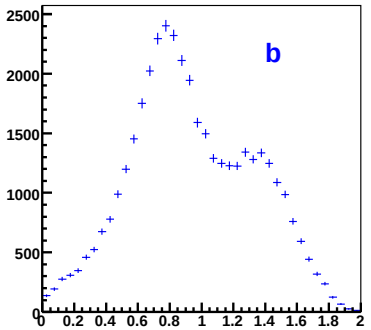
$w_i = 1$ would mean that the "unfolded" x is the same as the "truth" MC used to generate the response matrix.

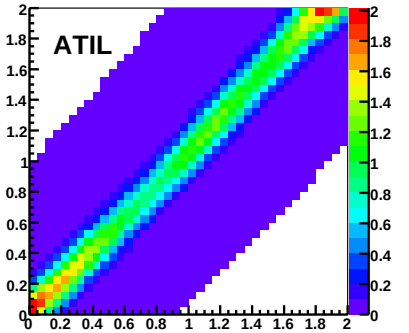
Just to give you some feeling, here are the original A, b and rescaled $\tilde{A}, \tilde{b}$ from the above example

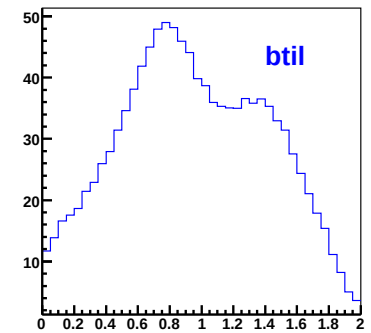
$x_i^{\rm ini} = {\rm const}$, so only equation rescaling matters.

By design, all $\tilde{b}_i$ are independent of each other, and have estimated errors of ± 1.









Solving equations

Now let's use SVD to solve the system (remember: S is diagonal!)

$$\tilde{A}w = \tilde{b} \quad \Rightarrow \quad \tilde{A} = U\,S\,V^T \quad \Rightarrow \quad U\,S\,V^T\,w = \tilde{b} \quad \Rightarrow \quad z \equiv V^T\,w, \quad d \equiv U^T\,\tilde{b}$$

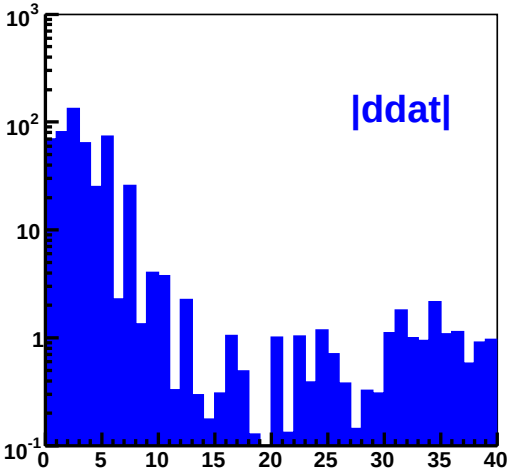
$$s_i z_i = d_i \quad \Rightarrow \quad z_i = \frac{d_i}{s_i} \quad \Rightarrow \quad w = V\,z$$

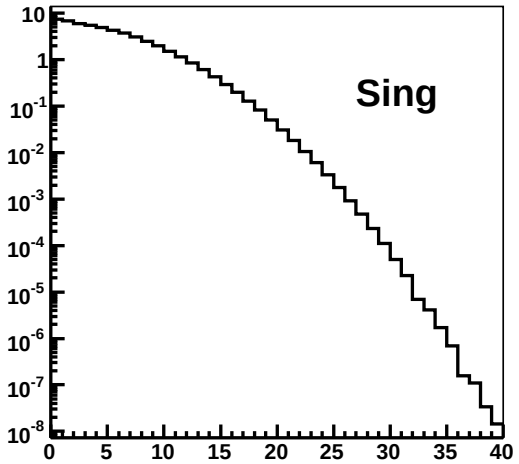
By construction, all d_i are independent and have errors ± 1.

Assuming all $s_i \neq 0$, z_i **form the exact solution** of the initial system.

Looking at the plots of $|d|$ and s, it's easy to understand what's happening:

- Many d_i are insignificant (compatible with zero)
- Corresponding s_i are small
- Respective z_i are **HUGE**, but nonetheless insignificant!





The way out

Exact solution of the system $\tilde{A}\,w = \tilde{b}$ is equivalent to minimisation of the residual χ^2:

$$\chi^2 \equiv (\tilde{A}w - \tilde{b})^T (\tilde{A}w - \tilde{b}) = \min.$$

where (ignoring machine precision and assuming all $s_i \neq 0$) $\chi^2_{\min} = 0$.

- Truncating the (diagonalised) system at some $i = k$ would make $\chi^2 = \sum_{i>k} d_i^2$.
- While simple truncation is better than keeping all i, this is not a good solution, as biases are hard to control.
- Regularisation by adding some external *a priori* information about the solution.
- Usual choice: the regularised solution has to be **smooth**.
- Techincally, an extra quadratic form is added to χ^2:

$$\chi^2 \equiv (\tilde{A}w - \tilde{b})^T (\tilde{A}w - \tilde{b}) + \tau (C\,w)^T C\,w = \min$$

- By choosing $C\,w$ to be the second finite difference of w

$$(C\,w)^T C\,w = \sum [(w_{i+1} - w_i) - (w_i - w_{i-1})]^2 \ .$$

  we find a minimum of χ^2 such that keeps the "integrated square of the second derivative" of w constrained.
- Essentially, τ is a Lagrange multiplier!

Minimising the new χ^2

A fairly straightforward method allows to reduce the new problem to the old one.

- The new minimisation problem gives rise to a new, overconstrained linear system:

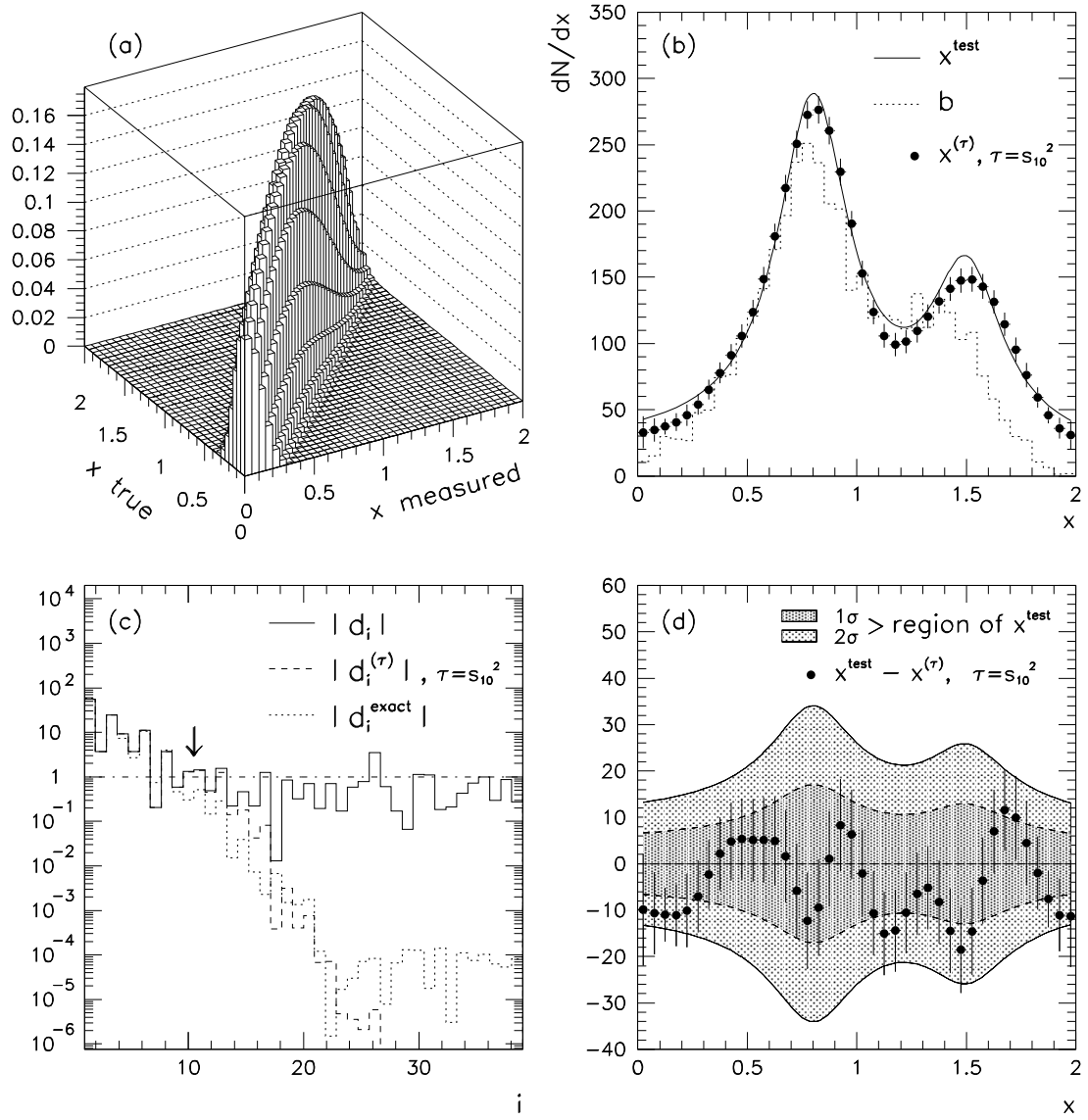
$$\begin{bmatrix} \tilde{A}\,C^{-1} \\ \sqrt{\tau}\cdot I \end{bmatrix} C\,w = \begin{bmatrix} \tilde{b} \\ 0 \end{bmatrix}.$$

- For $\tau = 0$ this is identical to the old system, and can be solved using SVD
- need to replace $\tilde{A}$ with $\tilde{A}C^{-1}$, and w with Cw.
- Once this solution is found, the solution for $\tau \neq 0$ is obtained immediately:

$$z_i^{(\tau)} = \frac{d_i}{s_i} \cdot \frac{s_i^2}{s_i^2 + \tau}$$

- For large $s_i \gg \tau$, the new suppression factor is close to 1, but for smaller s_i it works as a low-pass filter.
- Choose $\tau \simeq s_k^2$ where k is the index of the last significant d.

Back to the original example



Where are the errors?

One step back: the exact $z_i^{(\tau)} = \frac{d_i}{s_i}$ were independent, with unit covariance matrix.

The covariance matrix of regularied $z_i^{(\tau)}$ is no longer a unit matrix, although it is still diagonal:

$$Z_{ik}^{(\tau)} = \frac{s_i^2}{(s_i^2+\tau)^2} \cdot \delta_{ik}$$

Errors corresponding to insignificant d_i are suppressed.

Propagate these errors back to covariance matrices of w and x:

$$\begin{aligned} W^{(\tau)} &= C^{-1} V \, Z^{(\tau)} V^T \, C^{T-1} \\ X_{ik}^{(\tau)} &= x_i^{\mathrm{ini}} \, W_{ik}^{(\tau)} x_k^{\mathrm{ini}}. \end{aligned}$$

Inevitably, for any non-zero τ, the covariance matrices $W^{(\tau)}$ and hence $X^{(\tau)}$ are not diagonal, and the larger the value of τ, the larger are the bin-to-bin correlations in errors.

Bin-to-bin correlations of errors

This behaviour is not surprising: this type of unfolding is nothing else but a fit, and hence behaves as such.

In an extreme case of only one significant d_i, all one can say is that the data is **proprotional** to the Monte Carlo, with a fairly small error essentially reflecting the overall data statistics — but this error is fully correlated between all bins.

Unfolding cannot help here: you need more statistics (and/or a better detector, with a "more diagonal" response matrix) to make more d_i values significant!

However, I accept: a non-diagonal covariance matrix clearly creates a presentational problem: showing just $x_i \pm \sqrt{X_{ii}}$ is certainly misleading.

How about X^{-1}

In order to compare the measurement b with a theoretical prediction $f = f(\alpha)$, one could try **folding** the theory with the detector response matrix, and calculate

$$\chi^2 \quad = \quad (\mathbf{b} - A\mathbf{f})^T B^{-1} (\mathbf{b} - A\mathbf{f})$$

Alternatively, because $b = Ax$, one can replace b with Ax and have

$$\begin{aligned} \chi^2 &= (A\mathbf{x} - A\mathbf{f})^T B^{-1} (A\mathbf{x} - A\mathbf{f}) \\ &= (\mathbf{x} - \mathbf{f})^T A^T B^{-1} A \, (\mathbf{x} - \mathbf{f}) \\ &= (\mathbf{x} - \mathbf{f})^T X^{-1} (\mathbf{x} - \mathbf{f}) \end{aligned}$$

where $X^{-1} = A^T B^{-1} A$, or, in our notation,

$$X^{-1}_{jk} = \frac{1}{x^{\rm ini}_j x^{\rm ini}_k} \sum_i \tilde{A}_{ij} \tilde{A}_{ik}.$$

X^{-1} continued

The calculation of the inverse of the covariance matrix of the "true" distribution does not require any regularisation or unfolding.

By substituting $x^{(\tau)}$ for x, one only changes the χ^2 by a "small" amount of order of τ, which may or may not disturb the fit, depending on the model.

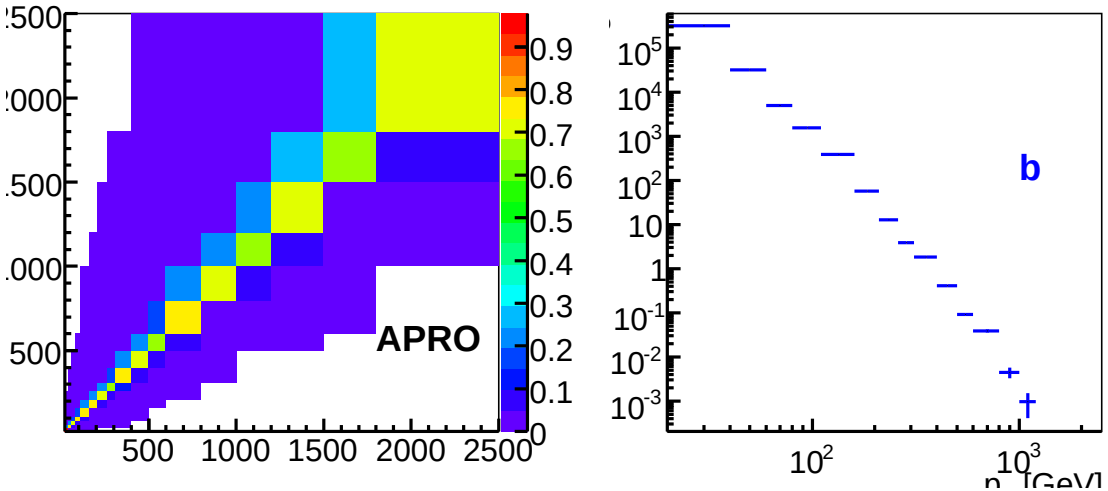
Hence, if the aim is to fit the unfolded distribution to theory, one is generally better off by folding theory, than by unfolding the measurement.

However if regularisation is "small", the results will be equivalent.

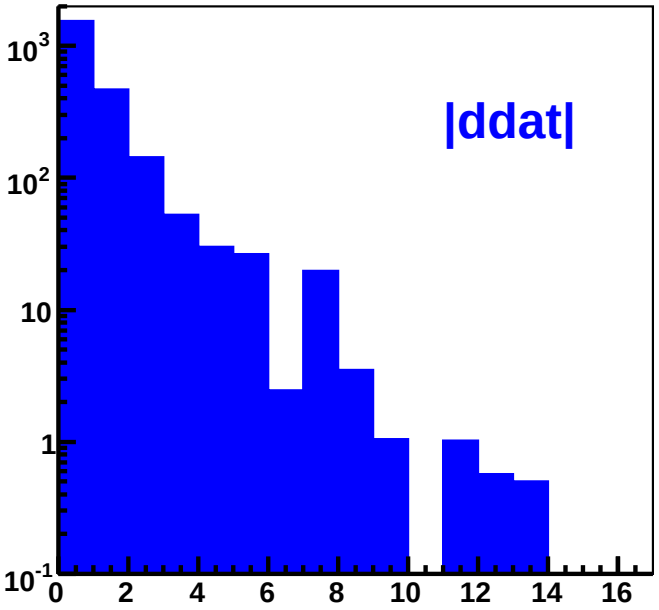
If a theoretical parameter happens to be sensitive to an insignificant element of the data, this will show up in the fit one way or another.

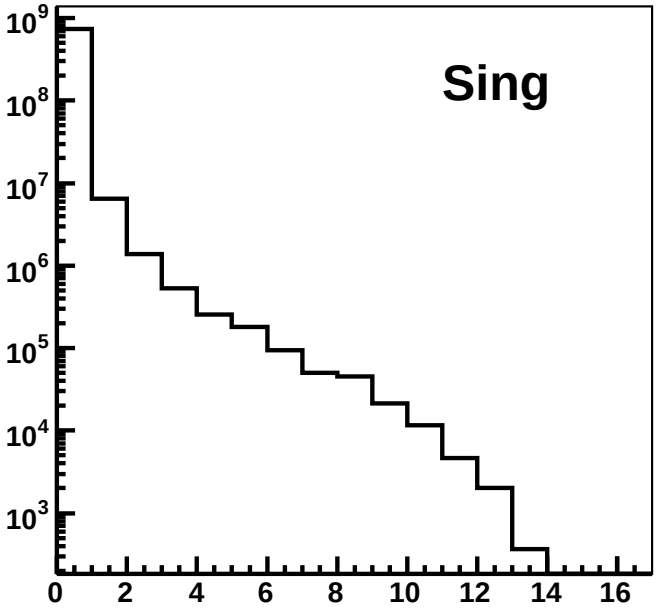
In any case, in my understanding, the regularised covariance matrix $X^{(\tau)}$ should only ne used for calculating errors on the unfolded distribution. Calculating its inverse is neither helpful, nor useful. Instead, if absolutely necessary, use X^{-1} above.

A real-life example — work in progress



Singular values and $|d|$

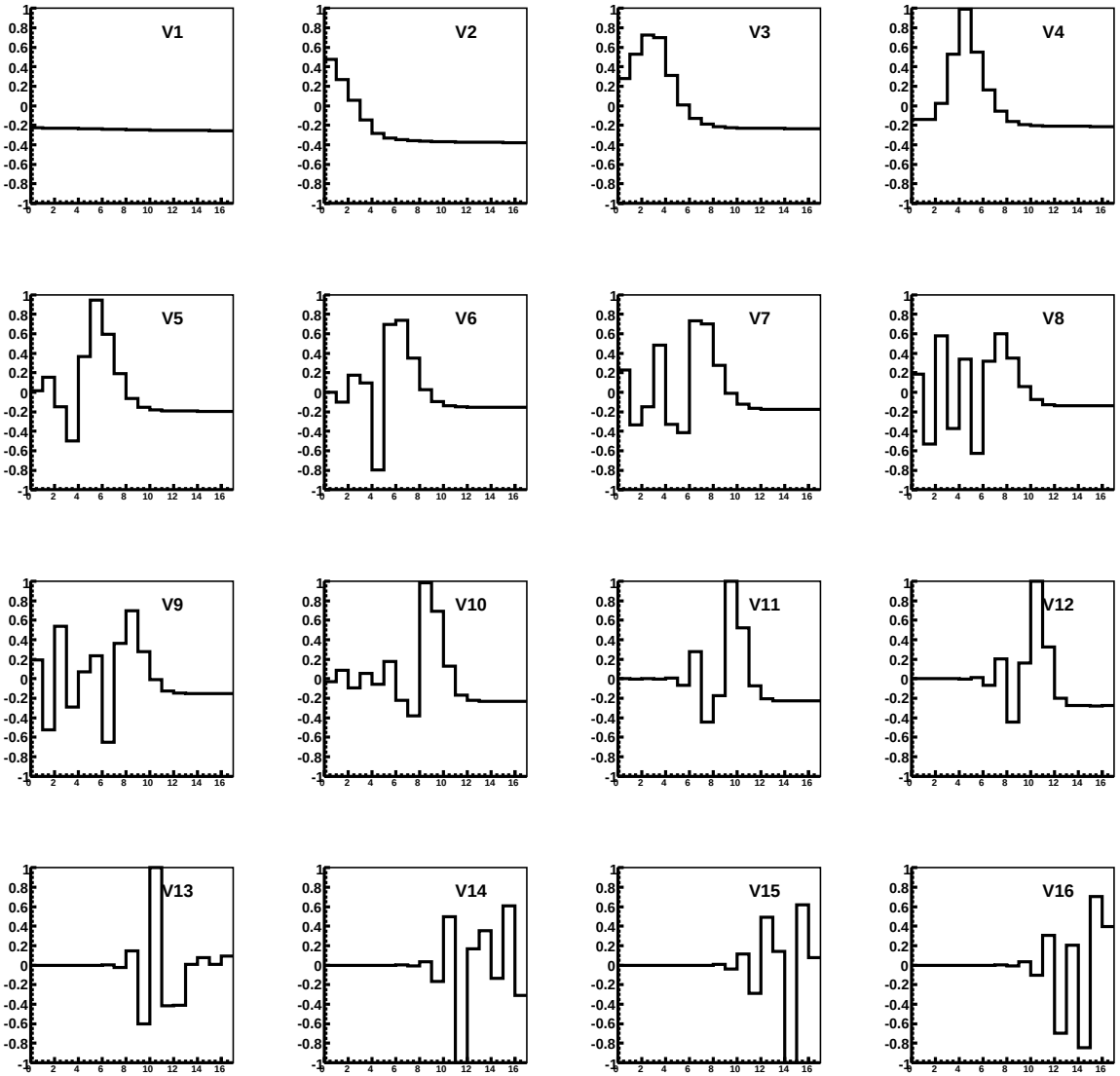




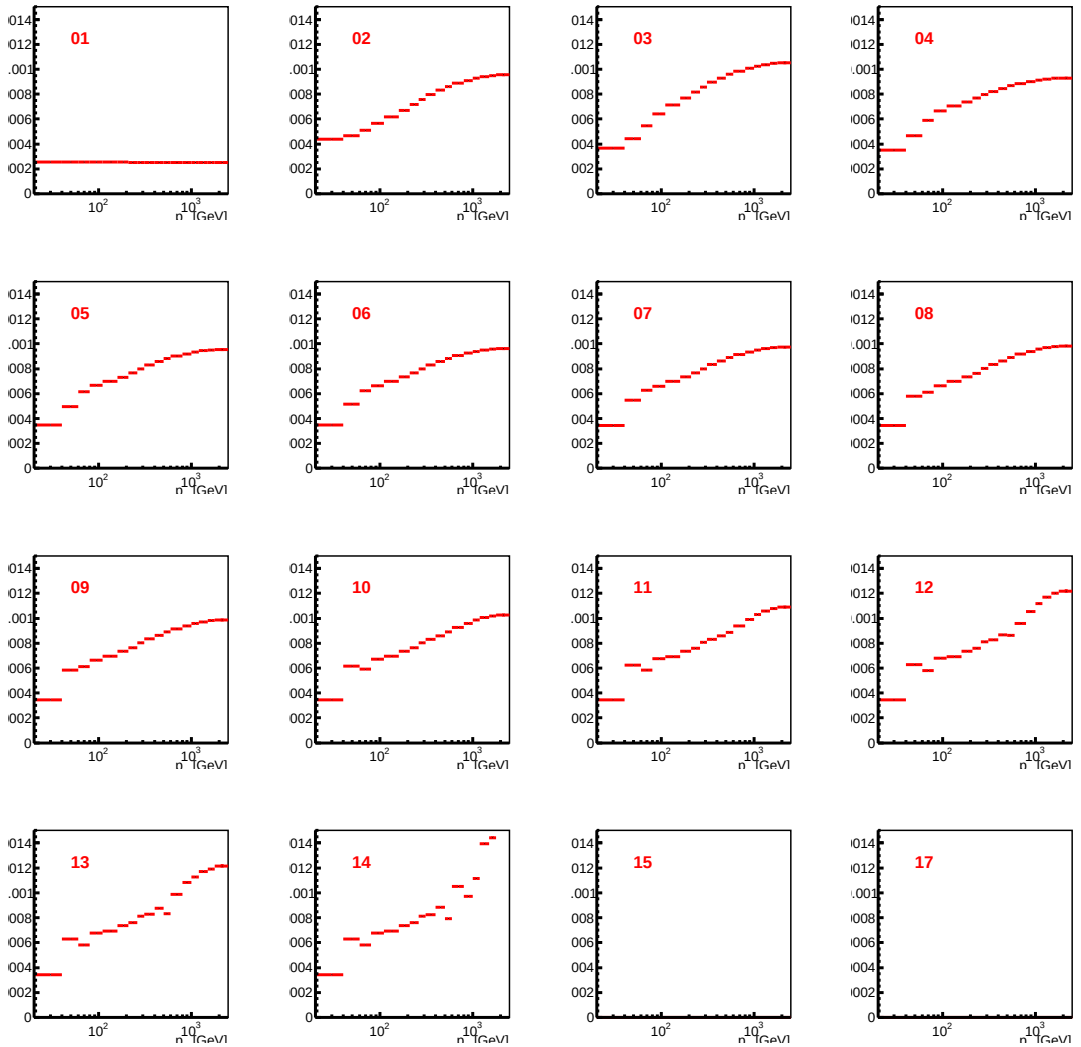
Looks like the last significant $|d|$ is number 9, so setting $\tau = s_{10}^2$ should be reasonable. Varying between 9 and 11 will dive some idea of relevant systematics.

The plot of singular values does not look too steep there, which is good.

The basis vectors: columns of V



Unfolded z for all τ choices



Based on information from previous page, number 10 would be my choice, but systematics needs to be studied. Stll much work to be done...

Some random pieces of advice

- Three things contribute to the solution: the detector response matrix, the data distribution, and the binning.
- Choose binning wisely: large variations in data may be avoided by using variable bin sizes. Size of the off-diagonal terms in the covariance matrix may also be affected.
- x and b need not have the same binning; in fact, these may be totally different variables!
- Rescaling is important: only unfold if equations have equal "weights".
- To avoid (some) surprises, use test MC samples with the same statistics as data.
- Vary the matrix within its tolerances, and study the effects on singular values.
- Don't expect any magic solutions!

Implementations

Several implementations of the above algorithm exist. Here are the ones I know:

- Fortran package called "GURU", still available from
  `http://www.hep.lancs.ac.uk/internal/guru.tar.gz`
- C++ wrapper for the above, written by Gavin Hesketh. See
  `http://www-d0.fnal.gov/~ghesketh/unfolding/`
- TSVDUnfold: Kerstin Tackmann will talk about it in the afternoon.
  `http://root.cern.ch/root/html/TSVDUnfold.html`

Even if you end up unfolding with some other algorithm, it's worth applying GURU/SVD first: it will help you identify the bottlenecks and assess any possible benefits.

Happy unfolding!