Unfolding

Unfolding

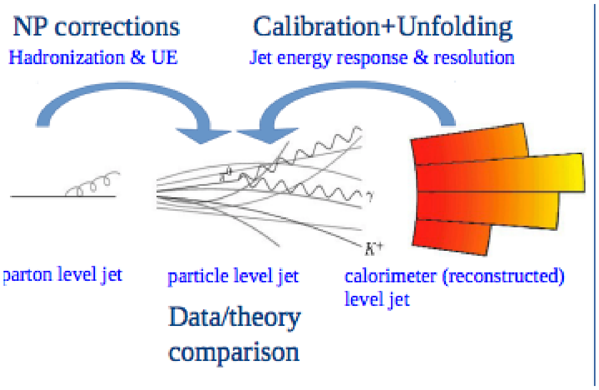
Introduction

We know that the detectors we measure physics in have a finite resolution, which *smears* the quantities we are interested in. Unfolding, or *deconvolution* is a technique to recover the "true information" we would measure with a perfect detector.

Unfolding is used after background subtraction in order to remove residual fluctuations, uncorrelated effects of the underlying event, and detector effects. The better our correction for the background is, the more mathematically stable the unfolding procedure is. The detector effects that unfolding corrects for are the following.

- Tracking inefficiency
- Missing long-lived neutral particles: n, K_L^0
- Track p_T resolution
- Gaps in acceptance
- Material loss in front of the EMCal
- Hadronic correction over- and under-subtraction
- EMCal energy spectrum "bump"
- Untracked charged particles outside of the EMCal depositing in EMCal
- Approximating the p_T of π^0, η by the E_T of their decay photons

Unfolding allows us to compare resulting distributions directly to theory which is particularly useful in isolating different jet energy loss mechanisms. Unfolding is additionally useful in comparing with other experimental measurements. A disadvantage of this method is that sometimes distinct features in data can be lost in the unfolding procedure. There are also a few ambiguities present in unfolding methods. The first is that unfolding makes an assumption as to the spectrum shape and the fragmentation pattern. We assume that observed jets have the same shape and fragmentation pattern as simulation, which is most certainly not true, especially in heavy-ion collisions. This is convoluted with the idea that the q/g jet fraction is different in data and simulation which also contributes to differing fragmentation. The second ambiguity is that there is not always a one-to-one correspondence between a truth jet and a detector level jet.



This comparison can also be made using a technique called *smearing* where we use a detector response (mapping from particle level MC → detector level MC) in order to create theory predictions at the reconstructed level which we compare directly to our measured distribution. The disadvantage of this method is that sometimes unfolding can reveal features that can tell us something about how the medium is modifying the partons.

General Procedure

We first form the response by using the mapping from true jets → reconstructed jets, which is achieved with MC by an embedding procedure (see corresponding section on embedding). Mathematically, this goes as

$$Rp_{\mathrm{T,MC}}^{\mathrm{true}} = p_{\mathrm{T,MC}}^{\mathrm{rec}}$$

Where mathematically the response matrix *R* represents

$$R_{\mathrm{ij}} = \mathrm{Prob}(\mathrm{observed\ in\ bin\ i} \mid \mathrm{true\ in\ bin\ j})$$

However, what we actually want is for the response to be the mapping from reconstructed jet → true jet. In order to get this mapping, we simply take the inverse of the response (R) and create the mapping to the true result as

$$p_{\mathrm{T,data}}^{\mathrm{true}} = R^{-1}p_{\mathrm{T,data}}^{\mathrm{rec}}$$

where the tag *data* here just means that this is the result we get from *data* (as opposed to MC).

> 💡 We see here that unfolding is at its core a matrix inversion. As we know from linear algebra, it is easier to take the inverse of a diagonal matrix. Hence, if there are strong non-diagonal components to the response matrix, then the matrix inversion, and therefore the unfolding, is made difficult. This is one reason why we always perform background subtraction before unfolding!

However, taking the inverse often results in an instability with respect to the statistical fluctuations. Within statistical uncertainties the smeared data can be explained by the actual physical solution, but also by a large family of unphysical solutions. We expect a degree of smoothness in physical solutions, so we impose this through a process called *regularization*.

Bayesian Unfolding

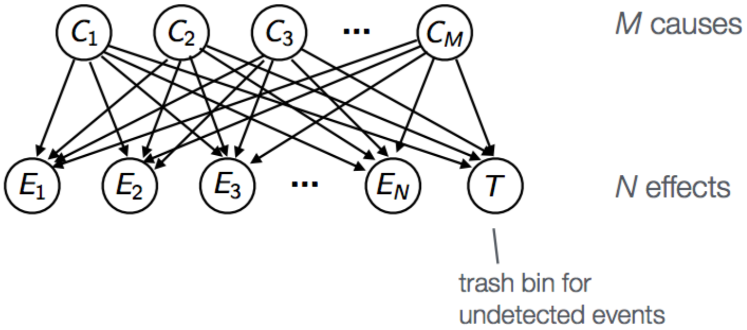
Paper: https://doi.org/10.1016/0168-9002(95)00274-X

Useful Slides: https://www.physi.uni-heidelberg.de/~reygers/lectures/2017/smipp/stat_methods_ss2017_09_unfolding.pdf

In physics, we are trying to derive causal relationships between different theoretical effects and observed data.

> 💡 Correlation is **not** causation!

This is where Bayesian causal networks come in. This is represented by a graph where nodes can be either causes or effects where the edges that connect them are the Bayesian probability for the given cause to produce the desired effect.



In this context, we can think of the response matrix as single representation of this network with the relation.

I: prior knowledge about probabilities of the causes C_i

$$R_{ji} = P(E_j | C_i, I)$$

Where here we are also building in prior knowledge. Remember from our statistics page

📈 Copy of Statistics

we said that Bayesian probabilities are determined as

$$P(\vec{\theta} \mid \vec{x}) = \frac{P(\vec{x} \mid \vec{\theta}) P(\vec{\theta})}{P(\vec{x})}$$

Bayes' theorem

However, here we also need to build in the prior knowledge, so this is now given by

$$P(C_i|E_j, I) = \frac{P(E_j|C_i, I) \cdot P(C_i|I)}{\sum_{k=1}^{M} P(E_j|C_k, I) \cdot P(C_k|I)}$$

and substituting in our relation for the response matrix we get

$$\theta_{ij} := P(C_i|E_j, I) = \frac{R_{ji} \cdot P(C_i|I)}{\sum_{k=1}^{M} R_{jk} \cdot P(C_k|I)}$$

so therefore, the estimation of the number of true events in bin i given we have n_j events in bin j is given as

$$\mu_i\big|_{n_j} = \frac{P(C_i|E_j, I) \cdot n_j}{\varepsilon_i} = \frac{\theta_{ij} \cdot n_j}{\varepsilon_i}$$

where ϵ_i is the efficiency defined mathematically as

$$\epsilon_i = \sum_{\mathrm{j}} R_{\mathrm{ji}}.$$

To reduce the dependence on our prior $P(C_{\mathrm{i}}, I)$, we need to apply this iteratively. The process roughly goes as follows

1. Choose a prior $P(C_{\mathrm{i}}, I)$

2. Make an estimate of the true distribution given the prior, response matrix and the measured distribution.
3. Update the Prior for the next iteration to be the true distribution from step 2.

where here the finite number of iterations provides implicit regularization. The central question here is after how many iterations do we stop? This choice will always introduce some sort of a bias. In order to minimize this bias, we stop iterating after the unfolding has become *stable* (i.e. does not change wildly from iteration to iteration). The RooUnfold implementation of bayesian unfolding additionally keeps track of error propagation from iteration to iteration and does not smooth (which will reduce the bias 🙂).

SVD Unfolding (With Regularization)

Paper: https://arxiv.org/pdf/hep-ph/9509307.pdf

Useful Resource: http://pillowlab.princeton.edu/teaching/statneuro2018/slides/notes02_SVD.pdf

RooUnfold Slides: https://indico.cern.ch/event/671301/contributions/2745801/attachments/1557488/2449991/171113-unfold.pdf

We saw from the introduction section that at its core, unfolding is a matrix inversion problem. In order to do this we will use a method analogous to the SVD (single-value decomposition) method from linear algebra. Before we embark, we will briefly review this.

SVD is a method for decomposing a single matrix *A*, into three matrices, *U, S, V* - each with special properties. This is written mathematically as

$$A = USV^t$$

where given *A* is an *n x m* matrix...

- *U* is a *n x m* matrix with n-element orthogonal columns.
- *S* is an *m x m* diagonal matrix (whose only non-zero entries), generally represented as an m-element vector. The diagonal entries $\{s_i\}$ are called *singular values*. The singular values are always $\geq$ 0.

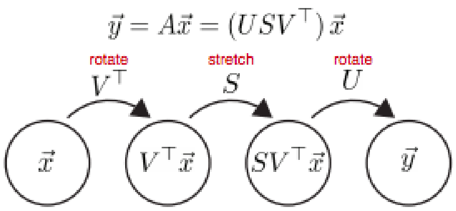
$$S = \begin{bmatrix} s_1 & & & \\ & s_2 & & \\ & & \ddots & \\ & & & s_n \end{bmatrix}$$

- *V* is an *m x m* matrix with m-element orthogonal columns

At its core, what SVD tells us is that we can think of the action of *A* upon any vector $\vec{x}$ in terms of three steps (one for each of the matrices in SVD)

1. rotation (multiplication by V^t which does not change the vector length of $\vec{x}$)

2. stretching along the cardinal axes (where the i'th component is stretched by s_i)
3. another rotation (multiplied by *U*)



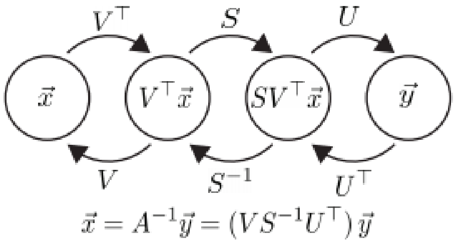
Now that we finished our brief review, how does any of this connect to unfolding? Well, as we said before, fundamentally unfolding is an inverse problem. SVD makes it easy to compute the inverse of a matrix. We go this by exploiting the fact that U and V are orthogonal, meaning that their transposes are their inverses. ($U^tU = UU^t = I$). From this we see that the inverse essentially reverses the above process, which is given mathematically as

$$A^{-1} = VS^{-1}U^t$$

where we know by the definition of inverse, S^{-1}must look like

$$S^{-1} = \begin{bmatrix} \frac{1}{s_1} & & & \\ & \frac{1}{s_2} & & \\ & & \ddots & \\ & & & \frac{1}{s_n} \end{bmatrix}$$

where visually this whole process can be represented as



Still not convinced? We can also mathematically prove that this is the correct inverse.

$$A^{-1}A = (VS^{-1}U^t)(USV^t) = VS^{-1}(U^tU)SV^t = V(S^{-1}S)V^t = VV^t = I$$

where we could also do something similar with AA^{-1}.

So for unfolding we can think of *A* as the response matrix *R* which we need to invert to create a linear mapping between measured and true. Of course I have just presented an idealized version of this process, instead of exactly determining the inverse of the experimental response matrix, we are really *numerically approximating* the inverse in an unfolding procedure. Numerically, this can actually be quite hard. To combat these difficulties, we use something called **regularization methods**. In SVD unfolding, the purpose of regularization methods is to suppress the effect of small eigenvalues (~noise) and constrain the smoothness of the unfolded distribution.

To implement regularization, we will introduce a *regularization parameter*, *k*. The parameter *k* places a cutoff on small single value contributions which therefore forbids high-frequency variations in the unfolded result. This is done by replacing $s_{\mathrm{i}}^2 \to \frac{s_{\mathrm{i}}^2}{s_{\mathrm{i}}^2+s_{\mathrm{k}}^2}$. Where essentially here what k is doing is setting the relative contributions of MC and Data, when

- *k* is too small → result dominated by MC truth
- *k* is too large → result dominated by statistical fluctuations

therefore k needs to be tuned for type of distribution, sample size, number of bins etc.

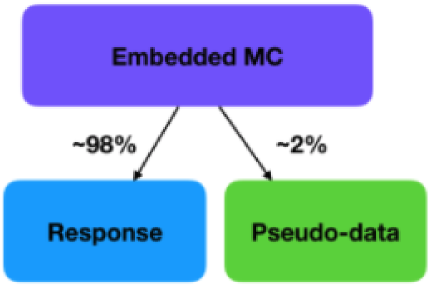
Tests of the Unfolding Procedure

Unfolding requires a few tests in order to determine that the procedure is stable and independent of choices of things like the number of iterations, regularization parameter etc. Most of these tests simply illustrate the validity of the unfolding procedure while others are taken as a systematic. We will discuss each of the various tests of the unfolding procedure as well as the variations which are taken as a systematic.

Trivial Closure Test: The trivial closure test is the most basic tests of mechanics of the unfolding itself. This test works by essentially unfolding the *exact same* distribution that is used to fill the response. This means that, almost trivially, the unfolded result should exactly agree with the true distribution that fills the response.

> 💡 If there is an issue with this test, be careful with the variable types that fill it. If the variables are not consistent (i.e. not all floats or not all doubles etc.) this has been know to cause issues.

Split MC Test: The Split MC test is similar to the trivial closure test in that embedded MC is being unfolded, but now we want to see that the unfolding is stable when we have realistic statistics in the distribution that we unfold and the response. To complete this test, we split the embedded MC with a certain percentage filling the response as usual and the remaining fraction filling a **"pseudo-data"** distribution that we unfold. What percentage is in the split will depend on the MC and data sample. The pseudo-data is then unfolded/refolded and checked for consistency with the true and the pseudo-data distribution.



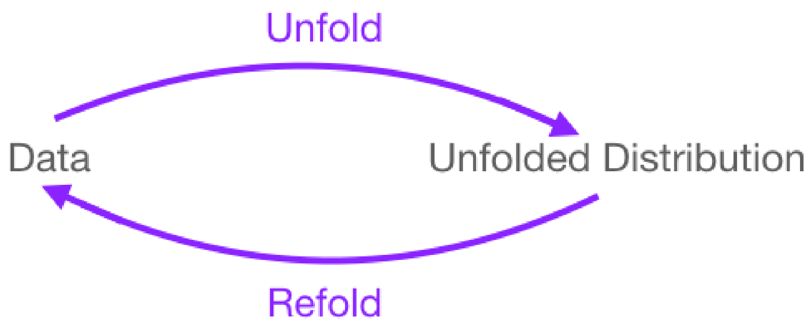
If the split MC test isn't working a good first thing to check is to vary the split. It very well could be that there is not enough statistics in the pseudo-data or the response. If it is not an issue in the split, another good thing to check on is the binning which is also impacted by statistics.

Stability with the Number of Iterations: A key feature of an unfolding procedure that is operating normally is that the resulting distribution is independent of the number of iterations used (when you unfold with SVD we want the result to be stable with respect to the regularization parameter). Generally the metric for stable unfolding is that the deviations between iterations remain 5% or less, however any sort of pattern or other sort of structure can be cause for concern.

There can be a number of different reasons why the unfolding may be unstable. Here are four (non-exhaustive) possibilities.

1. Outliers are not properly being removed in the response.
2. A mismatch in the jet population being unfolded and the jet population in the response.
3. Bumps in the measured spectrum from data (could be a binning issue).
4. Any kind of incorrect scalings.

Refolded to Raw Ratio: The refolded to raw ratio test basically ensured that the unfolding procedure has cyclic closure. This is represented graphically in the figure below.



> It's pretty rare that something goes wrong with the refolded test, but looks fine in other tests. If this test looks wrong, go back and check other tests! If the error still persists, a good thing to check is varying the measured binning.