Unfolding exercise

Beatriz Ribeiro Lopes (CERN), Jan Kieseler (KIT)

Plans

- Short presentation (<= 20 minutes) on unfolding
  - Reminder of (or first contact with) the idea behind unfolding and the pitfalls
  - Will introduce matrix inversion, regularisation, MLE unfolding

- Hands on exercises (notebooks or shell)
  - Low-level unfolding on simplified example
    - Implement yourself to understand the basics
  - High-level unfolding with combine
    - Hides the details but practical in analysis context

- We only need a few kB to MB of storage per person (all histograms)

Simplified unfolding example

Jupyter notebook with simple non-CMS example (particle decays) where the unfolding is done by hand with python

In almost any experiment, the measured variable is not directly the physics quantity of interest. Instead, the answer of the experimental apparatus has to be translated by the experimentalist, e.g., the signal height might be related to the energy of a particle.

In real experiments, it is impossible to always find the true value of the physics quantity, since the translation suffers from various uncertainties. E.g., the measured signal might be distorted due to noise and systematic limitations of the experiment.

Moreover, the *response function* of the experiment is convoluted in the measured signal. In most cases, an analytical deconvolution of the signal is impossible or at least not practical. Instead, numerical methods are applied to find the distribution of true values and their uncertainties for a given distribution of measured values. This process is called unfolding.

In this exercise, we consider only discrete distributions with N bins, corresponding either to histograms of continuous variables (like energy) or naturally discrete variables (like the mass number of atomic nuclei). Provided that the true and measured distributions have the same number of bins, the experimental response can be described by an $N \times N$ migration (or transfer) matrix R, where each element R_{ij} describes the probability of a true value in bin i to be classified or measured, respectively, in bin j.

This is reflected in the relation

$$\vec{g} = R \cdot \vec{f},$$

where $\vec{f}$ is a vector with N elements containing the true values for each bin, R is the migration matrix corresponding to the experimental response, and $\vec{g}$ is a vector containing the measured / observed values for each bin. This means that an ideal experiment would have the unit matrix as migration matrix.

The first step of the unfolding procedure is the determination of the migration matrix by calibration measurements or simulations. We assume that this step has been done already, and the exercises focus on the second step: unfolding of a given measured distribution provided that the migration matrix is known.

Hints:

Numpy offers several matrix operations, for example:

`linalg.inv(R)` returns the inverse of the a (migration) matrix `R`. See also `np.matrix.I` which performs the same operation.

`lambda, U = linalg.eig(R)` returns the eigenvalues and the matrix `U` of eigenvecotrs you need to diagonalize `R`.

Regularisation

Such unphysical *oscillations* in the unfolded result are typical for many unfolding techniques. Suppressing them is a major challenge when unfolding real data. One way to achieve this is regularization. There are various sophisticated methods for regularization (cf. lecture). Here is a simple method which can be programmed easily in Python:

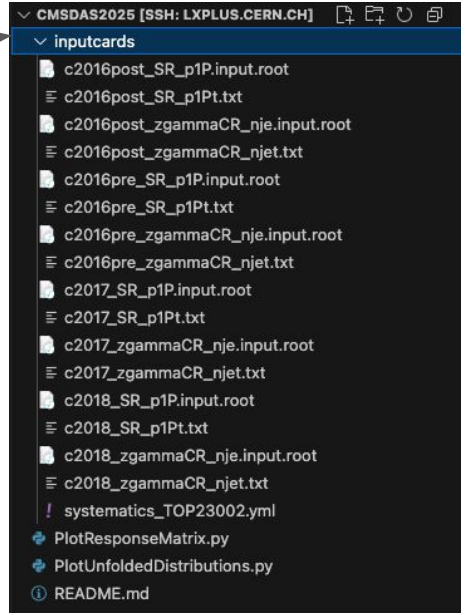
- Diagonalize the migration matrix R: $R_{\mathrm{diag}} = U^{-1}RU$ such that the eigenvalues λ_i of R form the diagonal of R_{diag}.
- Construct the observation vector $\vec{g}_{\mathrm{diag}} = U^{-1}\vec{g}_{\mathrm{obs}}$ and multiply each component i of the resulting vector with the corresponding component of $\vec{\lambda}^{-1}$, where $\vec{\lambda}^{-1}$ is a vector containing the reciprocals of the eigenvalues of R.
- The *regularization*: Set all elements i of $\vec{g}_{\mathrm{diag}}$ to 0, for which λ_i is smaller than a chosen threshold λ_{reg}. The choice of λ_{reg} is critical and a compromise between suppressing the unphysical oscillations, and keeping as much information as possible of the measurement.
- The unfolded result is $U \cdot \vec{g}_{\mathrm{diag}}$.

Apply the method with different thresholds λ_{reg} from -1 to 1. As a cross-check: if the algorithm is implemented correctly, the result should be identical to the one of the previous exercise, provided that λ_{reg} is smaller than the smallest eigenvalue of R. Discuss different choices for λ_{reg}: as measure for how similar the true and the unfolded distributions are, calculate the mean quadratic deviation (average over all bins). For which choice of λ_{reg} is the unfolded distribution most similar to the true distribution?

Unfolding with combine

- Based on TOP-23-002, we prepared a simplified ttgamma cross section measurement as a function of the photon pT

- We are collecting the datacards, shapes histograms, and skeleton scripts in a repository. The students have to:
  - Understand the datacard implementation (one process per generator-level bin to be measured, etc.)
  - Plot response matrix (reco vs gen photon pT)
  - Combine datacards and define the physics model with combine
  - Extract cross section modifiers from multidimensional fit to data
  - **Make the unfolded distribution**



- CMSSW_14_1_0_pre4 and combine v10 need to be installed before