Karlsruher Institut für Technologie

Computerpraktikum am 07.07. und 08.07.2022

**Verantwortlich für Vorlesung bzw. Computerpraktikum:**
Dr. P. Goldenzweig, Dr. R. Wolf, F. Metzner
**Tutoren:** M. Bauer, P. Ecker, M. Horzela, Dr. S. Stefkova, Dr. N. Trevisani, T. Voigtländer

# Computerpraktikum zur Vorlesung Moderne Methoden der Datenanalyse - Blatt 8

## Exercise 8: Unfolding

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 $j$ to be classified or measured, respectively, in bin $i$. 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, e.g., 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.

- **Exercise 8.1: Simple case with 2 categories** **voluntary**

  The simplest situation is an experiment with $N = 2$, for example the number of fouls per team in a soccer match between team A and B. Let us assume that the referee is biased and

favors team B. Whenever the referee detects a foul committed by team B, he gives a free kick to team A in only 70 % of the cases, but a free kick to team B in the remaining 30 % of the cases. Vice-versa, when team A commits a foul, team B gets the free kick in 90 % of the cases, but team A only in the remaining 10 % of the cases.

Thus, the diagonal entries of the migration matrix $R$ are 0.9 and 0.7, and the off-diagonal elements are 0.1 and 0.3 (think about which is the correct location of each element).

In the match there have been 22 free kicks for team A, and 24 for team B. Construct the migration matrix $R$, and estimate by inversion of $R$ how many fouls each team has committed. You can do this on a sheet of paper, or in Python with Numpy arrays.

- **Exercise 8.2: Regularization** **obligatory**

Now we consider a counting experiment with 7 categories (bins): each bin contains a number of observed events (e.g., particle decays, cars with different colors, classes of stars in an astronomical survey). We start from the following true distribution $\vec{f}$ of 3000 events in the 7 bins:

| | | | | | | |
|---|---|---|---|---|---|---|
| 35 | 218 | 814 | 1069 | 651 | 195 | 18 |

In our experiment some of the events are misclassified. For an event which truly belongs to bin $i$, there is a 30 % chance each that it is measured instead in bin $i-1$ or bin $i+1$ (except if the measured bin would be outside of the possible range from 1 to 7). Consequently, the migration matrix $R$ has 0.3 in all elements next to the diagonal, and 0.4 in the diagonal elements except for the corner elements which are 0.7.

First, let's consider an ideal experiment without any uncertainties:

a) Obtain the observed distribution of events $\vec{g}$. Then, unfold the observation by multiplication with $R^{-1}$, and compare the result to the true data. For this, plot the true distribution, the observed distribution, and the unfolded distribution.

Now we consider a more realistic case with uncertainties on the observed events (e.g., Poissonian uncertainties in the case of particle decays). Thus, the number of observed events deviates from the ideal case, and it is more difficult to reconstruct the original distribution. In our specific case, the experiment has observed the following distribution $\vec{g}_{\text{obs}}$ for the true distribution $\vec{f}$ given above:

| | | | | | | |
|---|---|---|---|---|---|---|
| 99 | 386 | 695 | 877 | 618 | 254 | 71 |

b) Unfold the observation by multiplication with $R^{-1}$, and compare the true, observed, and unfolded distribution by plotting them together. Which problems do you encounter?

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 $R$: $R_{\text{diag}} = U^{-1}RU$ such that the eigenvalues $\lambda_i$ of $R$ form the diagonal of $R_{\text{diag}}$.

- Construct the observation vector $\vec{g}_{\text{diag}} = U^{-1}\vec{g}_{\text{obs}}$ 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 $\vec{g}_{\text{diag}}$ to 0, for which $\lambda_i$ is smaller than a chosen threshold $\lambda_{\text{reg}}$. The choice of $\lambda_{\text{reg}}$ is critical and a compromise between suppressing the unphysical oscillations, and keeping as much information as possible of the measurement.
- The unfolded result is $U \cdot \vec{g}_{\text{diag}}$.

c) Apply the regularization method. Use different thresholds $\lambda_{\text{reg}}$ from $-1$ to 1. As a cross-check: if the algorithm is implemented correctly, the result should be identical to the one of exercise 8.2 b), provided that $\lambda_{\text{reg}}$ is smaller than the smallest eigenvalue of $R$. Discuss different choices for $\lambda_{\text{reg}}$: 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 $\lambda_{\text{reg}}$ is the unfolded distribution most similar to the true distribution?

Finally, we will apply an iterative method for unfolding. We start with a flat assumption for $\vec{f}_0$, i.e., each element is 3000/7 in our case, since we have 7 bins and 3000 events. Then, we fold $\vec{f}_0$ by multiplication with $R$ and compare the resulting $\vec{g}_0$ with the observation $g_{\text{obs}}$. Depending on the discrepancy between $g_0$ and $g_{\text{obs}}$, we *tune* our next guess for the true distribution $\vec{f}_1$ and repeat the procedure to obtain $\vec{g}_1$. If the *tuning* is reasonable, the guessed distribution will become closer to the the true distribution with each step. However, in the presence of uncertainties, too many iteration can lead to unreasonable results. Thus, choosing the number of iterations is again a challenge, similar to choosing the best threshold $\lambda_{reg}$ in the regularization method.

We will apply the following unfolding algorithm to improve our guess from iteration step $k$ to step $k+1$ (in a simplified version for symmetric response matrices):

1) $\vec{g}_{k+1} = R \cdot \vec{f}_k$
2) Tuning: calculate a vector $\vec{c}$ with weights to scale each bin $i$: $c^i = g^i_{\text{obs}}/g^i_{k+1}$. Consequently, the weight for bin $i$ is 1 if the observation is already reproduced by the result of step $k$.
3) Calculate $\vec{f}_{k+1}$: Multiply each element of $\vec{f}_k$ with the corresponding element of the vector $R \cdot \vec{c}$.

d) Apply the iterative method on the distribution you would observe without any experimental uncertainties. How many iterations do you need to achieve a maximum deviation of 0.1 % per bin between the true distribution and the unfolded observation?

e) Now, apply the method on the observation with uncertainties $\vec{g}_{\text{obs}}$. Try and discuss different choices for the number of iterations.

f) The choice of $\vec{f}_0$ is similar to the choice of a prior in a Bayesian analysis and influences the result. Try different choices for $\vec{f}_0$ and test the influence on the result for small and large number of iterations.

Hint:
Numpy offers several matrix operations, for example:

`linalg.inv(R)` returns the inverse of the migration matrix $R$.
`lambda, U = linalg.eig(R)` returns the eigenvalues and the matrix $U$ of eigenvectors you need to diagonalize $R$.