Third block exercises

Author: Aleix López Pascual
Statistics and Data Analysis

January 20, 2018

Hypothesis testing

Exercise 1

We are going to select signal-enriched subsamples out of the data sample (signal + background) and background control sample (only background).

(a) We are going to construct some test statistics, i.e. functions of the measured data T(x) used to investigate the level of agreement with the hypothesis. The idea is that instead of working directly with the data $\vec{x} = (x_1, x_2, ... \ x_n)$ and $\vec{y} = (y_1, y_2, ... \ y_n)$, we want to construct a one-dimensional test statistic with less information but equal discrimination power against the hypotheses. Once we have reduced the data to one value per each pair (x, y), we perform the hypothesis test. We construct the following test statistics:

- The radial distance $r = \sqrt{x^2 + y^2}$.
  It does not have free parameters to estimate, so we just construct a function of the data.
- The Fisher linear discriminant using polar coordinates (r, θ).
  It is a linear combination of the variables (r, θ):

$$T = \sum_{i=1}^{2} a_i x_i \qquad \text{where } \vec{x} = (r, \theta) \tag{1}$$

  In order to use $T(x)$, we need to find the parameters a_i. The goal is to determine the a_i so as to maximize the power against H_1 for a given significance level with respect to H_0. These a_i correspond to $\vec{a} \propto W^{-1}(\vec{\mu_0} - \vec{\mu_1})$ where $W = V_0 + V_1$. Then, it is clear that here we have the free parameters $\vec{\mu_0}$, $\vec{\mu_1}$, V_0 and V_1, which we will need to estimate from the train sample. However, our train sample is written in terms of x, y. So first we need to transform the variables: $r = \sqrt{x^2 + y^2}$ and $\theta = \tan^{-1}(y/x)$. Doing all this, we obtain:

$$\vec{\mu}_s = \begin{pmatrix} 0.535 \\ 0.469 \end{pmatrix} \qquad \vec{\mu}_b = \begin{pmatrix} 0.718 \\ 0.000 \end{pmatrix} \tag{2}$$

$$V_s = \begin{pmatrix} 0.092 & 0.072 \\ 0.072 & 0.559 \end{pmatrix} \qquad V_b = \begin{pmatrix} 0.062 & 0.001 \\ 0.001 & 0.802 \end{pmatrix} \tag{3}$$

  We note that V_s and V_b are covariance matrices. We compute the difference $(\vec{\mu_0} - \vec{\mu_1})$ and W^{-1} and we finally obtain the weight vector $\vec{a}$ up to some arbitrary scale factor. After analysing the results, we notice that we should compute the difference $(\vec{\mu_1} - \vec{\mu_0})$ instead of $(\vec{\mu_0} - \vec{\mu_1})$. Doing so, we finally obtain:

$$\vec{a} = \begin{pmatrix} 1.384 \\ -0.419 \end{pmatrix} \tag{4}$$

  Using these parameters $\vec{a}$, we expect to separate out the signal from the background so that we can further analyse a clean sample of the signal events.

Why using r and θ should be better than using x and y? Because if we compute the means and variances for the variables x and y we obtain:

$$\vec{\mu}_{s,xy} = \begin{pmatrix} -0.006 \\ -0.000 \end{pmatrix} \qquad \vec{\mu}_{b,xy} = \begin{pmatrix} -0.008 \\ 0.004 \end{pmatrix} \tag{5}$$

$$V_{s,xy} = \begin{pmatrix} 0.187 & 0.145 \\ 0.145 & 0.191 \end{pmatrix} \qquad V_{b,xy} = \begin{pmatrix} 0.291 & 0.002 \\ 0.002 & 0.287 \end{pmatrix} \tag{6}$$

We observe that the means are very close and the covariance matrices are not as small as in the case (r, θ). As a consequence, the separation power is weaker when we use the variables (x, y).

- The exact likelihood ratio $\lambda(x, y) = \frac{f_b(x,y)}{f_s(x,y)}$.
  Since we know the joint pdf's for the spatial coordinates for signal and background events, we can compute the exact likelihood. However, first we need to normalize such pdf's. In order to do so, we obtain the normalization constant computing the double integral of the function over all the domain $x \in [-1, 1]$ and $y \in [-1, 1]$. The normalization constants obtained are: $N_s = 0.868$ and $N_b = 3.014$.

- The likelihood ratio estimated from the train sample.
  In this case, we consider that we do not know the pdf's. Therefore, we need to use the train sample to estimate them. The simplest density estimators are the histograms. Since we are considering two variables (x, y), we estimate $f_s(x, y)$ and $f_b(x, y)$ as 2-dimensional histograms. Once we have the fitted model with the two pdf's estimated, we want to be able to use this model to predict the responses for the observations in a second dataset. What do we do? We define a function for each pdf that does the following: for each pair (x, y) of the data, find the corresponding bins in the histogram and return its entries. Considering this, we finally obtain the test statistic values dividing $f_b(x, y)/f_s(x, y)$.

- A neural network
  This test statistic is defined as

$$T(x) = s\left(a_0 + \sum_{i=1}^{n} a_i x_i\right) \tag{7}$$

  for the case of a single-layer perceptron, and can be generalized to N-layer perceptron adding $(N - 1)$ hidden layers. $s(\cdot)$ is the activation function and a_i are the weight parameters. In order to use $T(x)$, we must determine these parameters. This process is called the network training. We want to adjust the weights so that the resulting $T(x)$ gives an optimal separation between the hypotheses. This is not as straightforward as in the linear case (Fisher discriminant). Here, the optimization of the parameters is typically based on minimizing an error function, such as

$$\varepsilon = E_0[(T - T^{(0)})^2] + E_1[(T - T^{(1)})^2] \tag{8}$$

  where $T^{(0)}$ and $T^{(1)}$ represent the preassigned target values for the hypotheses H_0 and H_1, which we will take to be $T^{(0)} = 0$ and $T^{(1)} = 1$. The error assignment is based on the ability to correctly classify if an event is of the appropriate type. Signal events should be classified as signal (0) and background events as background (1). Then, we iterate numerically until we determine the parameters that minimize ε.

  In order to perform the network training, we are going to use the package sklearn (scikit-learn). We are going to define the test statistic using the train samples (signal and background). First, we start scaling the data using the class sklearn.preprocessing.StandardScaler. Standardization of datasets is a common requirement for many machine learning estimators implemented in scikit-learn; they might behave badly if the individual features do not more or less look

like standard normally distributed data: Gaussian with zero mean and unit variance. Once we have the datasets standardized, we define and train the neural network (NN). We use the class sklearn.neural_network.MLPRegressor, which implements a multi-layer perceptron (MLP) that trains using backpropagation with no activation function in the output layer.

When we use the MLPRegressor, we must indicate the architecture of the NN. There are some rules to determine the number of hidden layers and the number of neurons in each one. In our case, the training data is an array $(20000, 2)$, where we have appended signal and background data. In order to provide an arbitrary good parametrization of any function, we are going to consider two hidden-layers. The optimal choice of neurons per hidden layer would be 13333 neurons for the first one and 6667 neurons for the second one, where we have used $(\text{In/Out}) \cdot 2/3$ and $(\text{In/Out}) \cdot 1/3$. Using too few neurons in the hidden layers will result in underfitting (some parameters will be missing). Using too many neurons may result in overfitting (contains more parameters than can be justified by the data). However, such architecture takes a lot of time and memory to train the network. therefore, we will use less neurons (1000 neurons for the first hidden layer and 200 neurons for the second one). As a consequence, we will have underfitting.

In case there is overfitting or underfitting, we can also use a parameter alpha as regularization term, which helps in avoiding overfitting by penalizing weights with large magnitudes. Increasing alpha may fix high variance (overfitting). Decreasing alpha may fix high bias (underfitting). As we have underfitting, we fix $\alpha = 1 \times 10^{-5}$. Furthermore, since in our case we deal with a large dataset, we use the solver "adam".

Considering all this, we define our trained model which learns a function with its weights. Now, it only remains to enter some data and return the predicted output using this trained model.

(b) For each considered test, we are going to compute its values T for the train and test samples. Representing these values in histograms, we are able to show the distributions of T (Fig. 1-5).

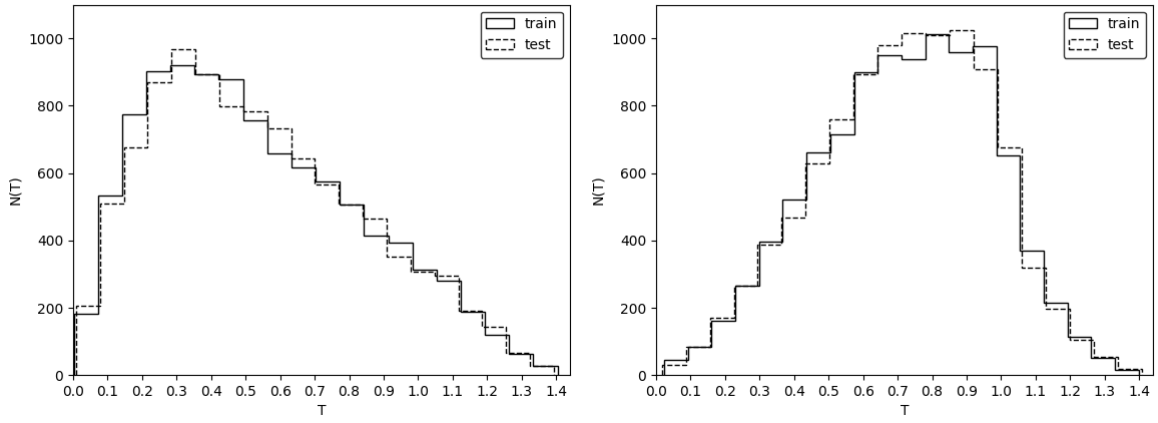

Figure 1: Distributions of T for the radial distance using the train and test samples. On the left, signal samples. On the right, background samples.

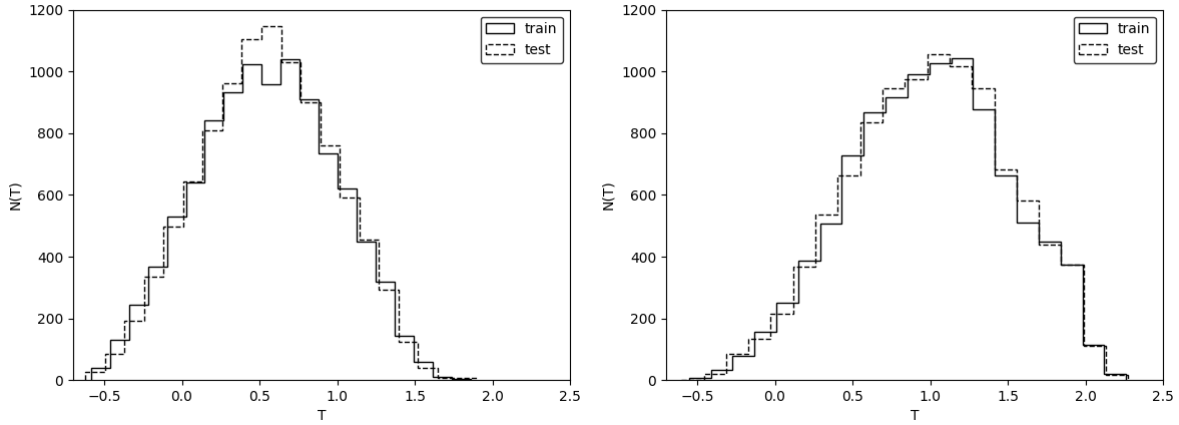

Figure 2: Distributions of T for the Fisher linear discriminant using the train and test samples. On the left, signal samples. On the right, background samples.

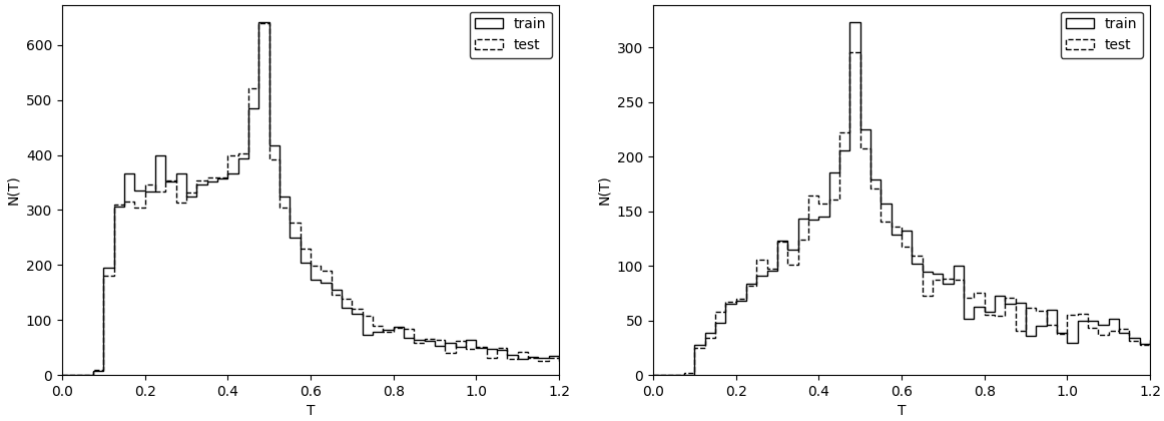

Figure 3: Distributions of T for the exact likelihood ratio using the train and test samples. On the left, signal samples. On the right, background samples.

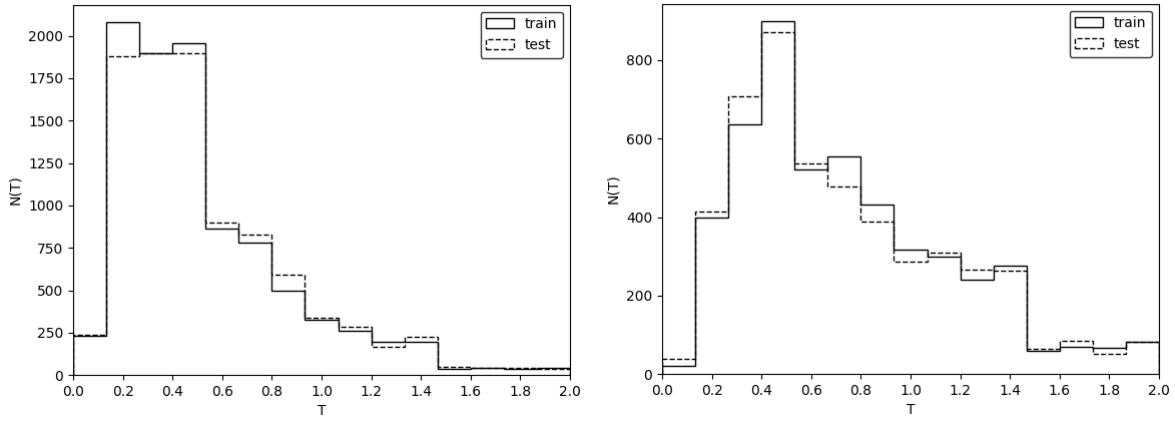

Figure 4: Distributions of T for the estimated likelihood ratio using the train and test samples. On the left, signal samples. On the right, background samples.

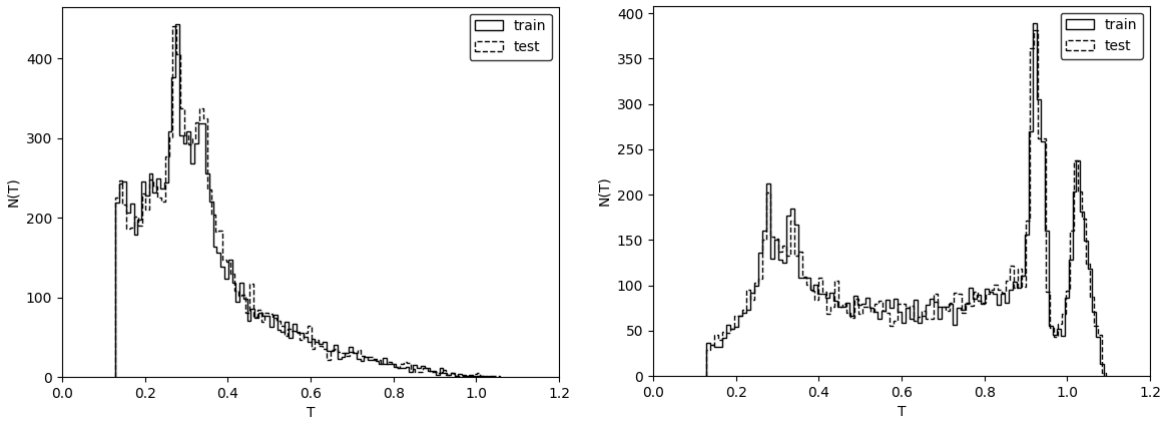

Figure 5: Distributions of T for the neural network using the train and test samples. On the left, signal samples. On the right, background samples.

The train dataset is the one used for learning, i.e. to fit the parameters. On the other hand, the test dataset is a dataset that is independent of the train dataset, but that follows the same probability distribution. Therefore, if a model fit to the train dataset, we should expect that it also fits the test dataset well. As we can observe (Fig. 1-5), this is in fact the case. The distributions obtained using the train sample and using the test sample are very similar for all the test statistic proposed.

In addition, we are going to compute the p-values using the Kolmogorov test for the compatibility of train and test T-distributions. In our case, we are interested in compare two datasets. Therefore, the Kolmogorov test check whether the 2 samples are drawn from the same distribution. There is a module scipy.stats.ks_2samp which computes this directly and returns the Kolmogorov statistic and the p-value. The Kolmogorov statistic is the maximum deviation between both measured cumulative distributions:

$$D_{NM} = \max |S_N(x) - S_M(x)| \tag{9}$$

If the Kolmogorov statistic is small or the p-value is high, there is a high agreement that the distributions of the two samples are the same.

Test statistic	Hypothesis	Kolmogorov statistic	p-value
Radial distance	Signal	0.0186	0.0621
	Background	0.0094	0.7671
Fisher discriminant	Signal	0.0161	0.1483
	Background	0.0123	0.4336
Exact likelihood ratio	Signal	0.0211	0.0230
	Background	0.0123	0.4336
Estimated likelihood ratio	Signal	0.0268	0.0015
	Background	0.0129	0.3740
Neural network	Signal	0.0235	0.0078
	Background	0.0121	0.4546

Table 1: Values of the Kolmogorov statistic and the p-value for the proposed test statistics. It is a two-sided test, so the p-values are two-tailed.

As we can observe (Table 1), we note a big difference between the p-values obtained from the signal samples and for the background samples for all the test statistics. For the background samples we obtain high p-values, but for the signal samples we obtain low p-values. Since this observation occurs for all the test statistics, even in those which did not require training, we conclude that the reason for these low p-values is that the train signal sample has important fluctuations with respect to the test signal sample. Instead, the train background sample does not have so many fluctuations.

Why is important the p-value? Given a significance level α of the test (traditionally 5% or 1%), if the p-value is less than the chosen significance level, that suggests that the observed data is sufficiently inconsistent with the null hypothesis that can be rejected (in our case the null hypothesis is that the two samples are drawn from the same distribution). Having said that, we are going to pay attention for the cases for which $p < 0.01$. From table 1, we identify that these cases are the estimated likelihood ratio and the neural network for the signal data. In such cases, we might consider rejecting the hypothesis that both samples are drawn from the same distribution.

(c) We are going to estimate and plot the values of $(1-\alpha)$, β and $(1-\alpha)/\sqrt{\beta}$ as a function of T_{cut} for each test. α is the significance level of the test and it is the probability of type 1 error (H_0 is rejected but H_0 is true). β is the probability of type 2 error (H_0 is accepted but H_0 is not true).

$$\alpha = \int_{T_{\text{cut}}}^{\infty} f_s(T)\, dT \qquad \beta = \int_{0}^{T_{\text{cut}}} f_b(T)\, dT \tag{10}$$

From now on, we will use the test samples. We could also use the train samples since we saw that both samples provide the same distributions of T. In order to compute Eq. (10), we need the pdf's $f_s(T)$ and $f_b(T)$. Since we do not know them, we estimate them using the histograms of the obtained T values for each test.

In order to compute the integrals, we recall that the integral is just the total area underneath the curve. Then, we identify the bins we want to integrate, we sum all the counts of them and we multiply by the bins width. Since α and β are probabilities, we want to receive values between 0 and 1, so we will have to use normalized histograms.

When we plot the values of interest as a function of T_{cut}, we select an appropriate range of T_{cut} in accordance to the values of T obtained. Otherwise α and β are directly 0 or 1. We also select a small step in T_{cut} to have more precision. The number of values of T_{cut} should be in accordance with the number of bins of the histograms.

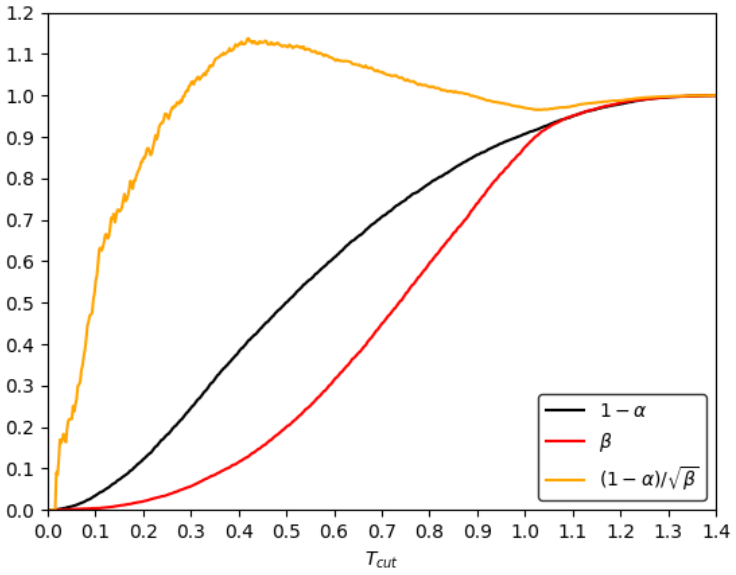

Figure 6: $(1-\alpha)$, β and $(1-\alpha)/\sqrt{\beta}$ as a function of T_{cut} for the radial distance

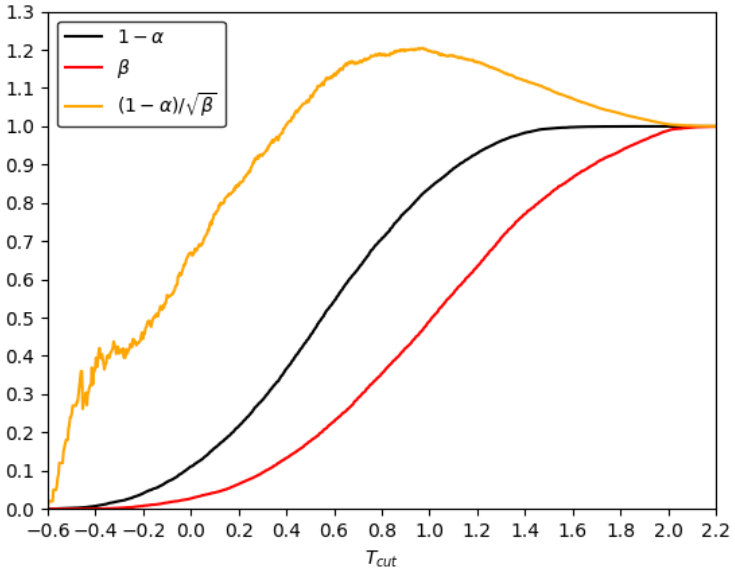

Figure 7: $(1-\alpha)$, β and $(1-\alpha)/\sqrt{\beta}$ as a function of T_{cut} for the Fisher linear discriminant

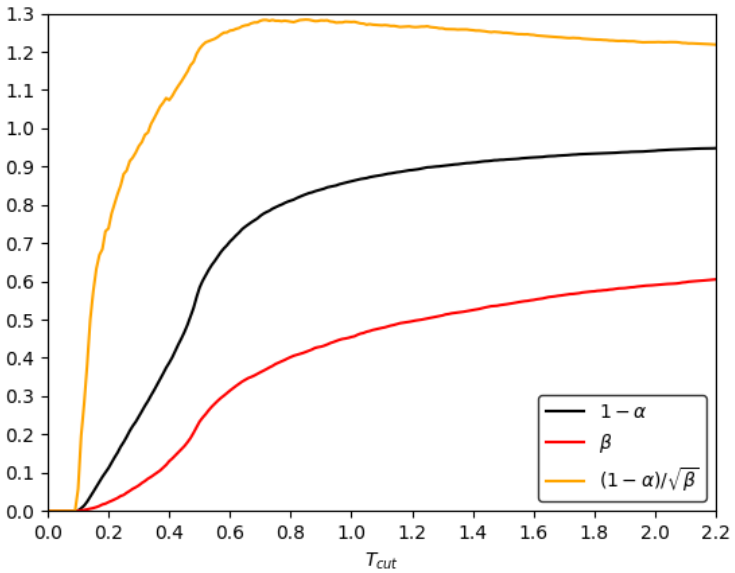

Figure 8: $(1-\alpha)$, β and $(1-\alpha)/\sqrt{\beta}$ as a function of T_{cut} for the exact likelihood ratio

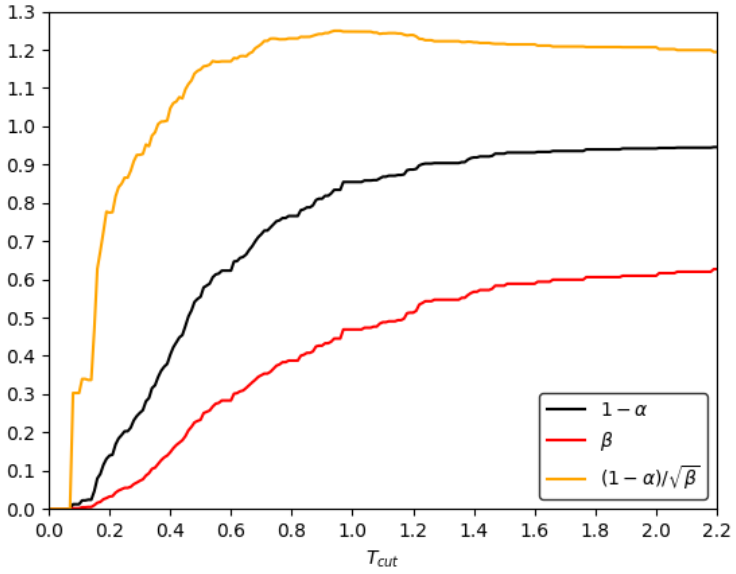

Figure 9: $(1-\alpha)$, β and $(1-\alpha)/\sqrt{\beta}$ as a function of T_{cut} for the estimated likelihood ratio

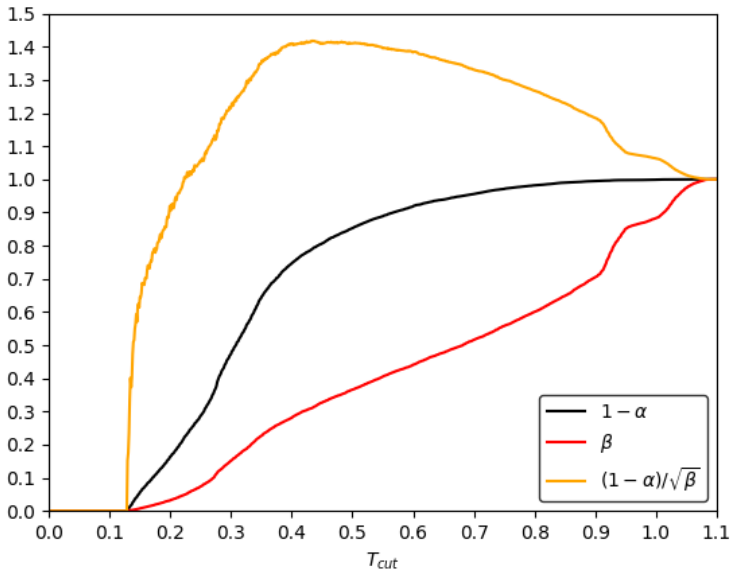

Figure 10: $(1-\alpha)$, β and $(1-\alpha)/\sqrt{\beta}$ as a function of T_{cut} for the neural network

(d) If we select events fulfilling $T < T_{\text{cut}}$, then the $(1-\alpha)/\sqrt{\beta}$ is the signal-to-noise ratio, which is approximately proportional to the statistical significance of the signal in the selected subsample. The statistical significance is used to determine whether the null hypothesis should be rejected or retained. For the null hypothesis to be rejected, an observed result has to be statistically significant, i.e. the observed p-value has to be less than the pre-specified significance level. The decision boundary T_{cut} is the value that defines the boundary between the acceptance region and the critical region for T. Therefore, we define the T_{cut} as the value that maximizes the statistical significance.

Test statistic	T_{cut}	$(1-\alpha)/\sqrt{\beta}$
Radial distance	0.42092	1.13782
Fisher discriminant	0.97247	1.20474
Exact likelihood ratio	0.85009	1.28411
Estimated likelihood ratio	0.94009	1.24970
Neural network	0.43363	1.41784

Table 2: T_{cut} values and its corresponding signal-to-noise ratio $(1-\alpha)/\sqrt{\beta}$ for each test statistic

(e) We are going to compare the performance of the different test statistic by plotting together their signal-to-noise ratio as a function of $(1-\alpha)$.

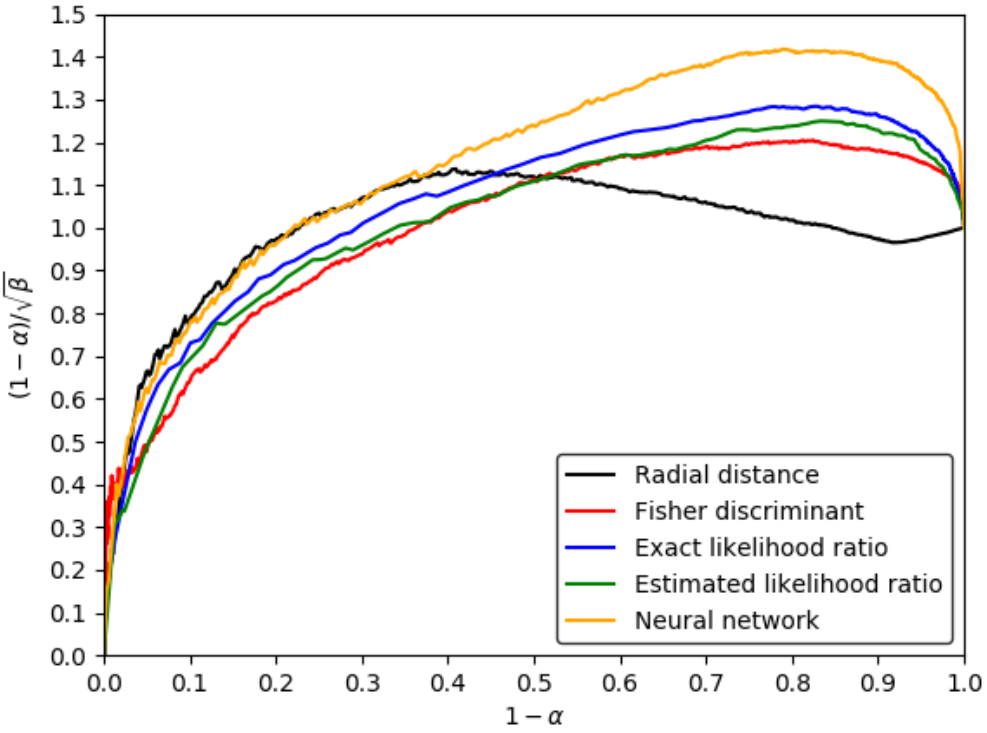

Figure 11: Signal-to-noise ratio $(1-\alpha)/\sqrt{\beta}$ as a function of $(1-\alpha)$ for the different proposed test statistics

A test statistic is used to investigate the level of agreement with the hypothesis. Therefore, the best test will be the one which yields the maximum power $(1-\beta)$ to discriminate H_0 against H_1 for a given significance level α, i.e. the one with less probability to make an error. In other words, the best test will be the one with higher signal-to-noise ratio $(1-\alpha)/\sqrt{\beta}$, which is proportional to the statistical significance.

Considering this, as we can observe from Fig. 11, we conclude that the best test statistic is the neural network. This is in fact, the result we were expecting after seeing the distributions of T for signal and background in part (b) (Fig. 5). However, it is not the result expected from the Neyman-Pearson lemma, which states that the best test statistic when performing a hypothesis test between two simple hypothesis should be the exact likelihood ratio. Why we have obtained a different result? Remember that when we constructed the neural network in part (a), we discussed that we had to estimate a large number of free parameters by performing a training. By including a larger number of parameters, we were able to better approximate the optimal test statistic given by the exact likelihood ratio. More parameters imply more neurons, and in part (a) we deduced the desired number of neurons, but we used less in order to reduce the time and memory to train the network. Having said that, we suspect that the reason of this different result is due to this underfitting.

The order of best test statistics is given in accordance to the number of parameters we train. Larger number of parameters imply a better approximate to the exact likelihood ratio. Obviously, the best test should be the exact likelihood ratio since it uses the pdf's $f(x|H_0)$ and $f(x|H_1)$ with no free parameters. Then, the order decrease up to the number of parameters we train. The last one is the radial distance as expected, since it does not have any free parameter.

(f) We are going to draw the boundaries of the critical region defined by T_{cut} in the (x, y) plane. In order to do so, first we generate values of x and y between the defined range and we make 2D coordinate arrays using np.meshgrid. Then we compute the value of the test statistic for each pair (x, y) of the array. We select the values above and below T_{cut} and we finally show the results in a contour filled plot.

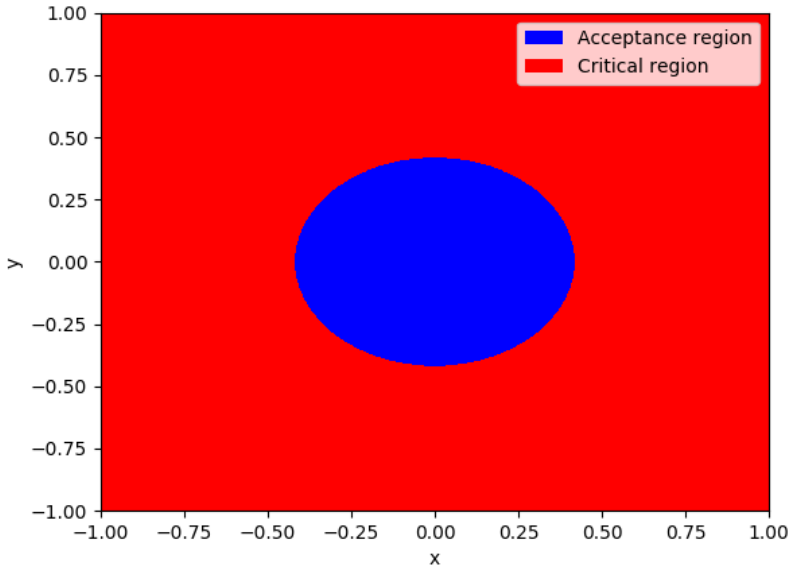

Figure 12: Acceptance region and critical region in the (x, y) plane for the radial distance

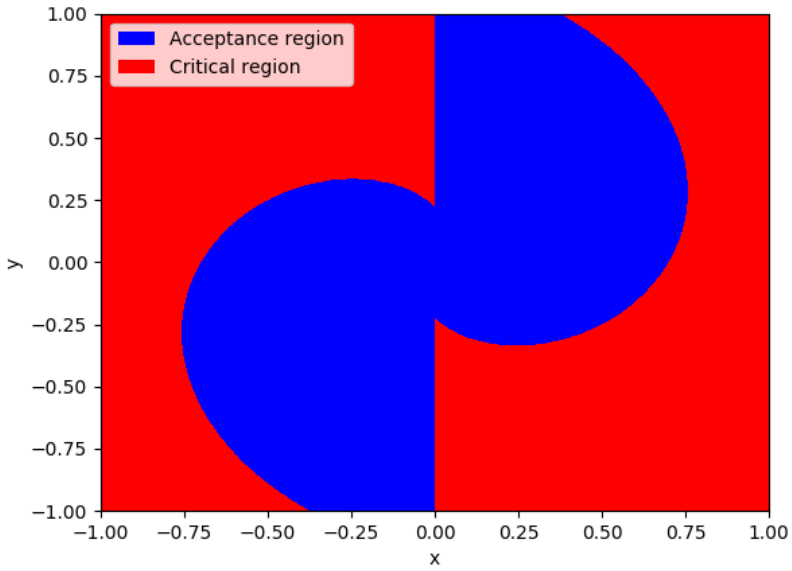

Figure 13: Acceptance region and critical region in the (x, y) plane for the Fisher linear discriminant

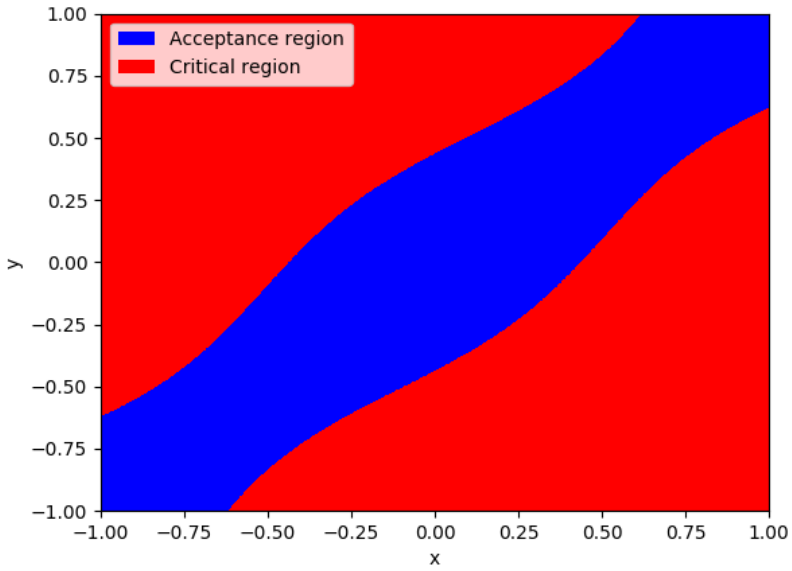

Figure 14: Acceptance region and critical region in the (x, y) plane for the exact likelihood ratio

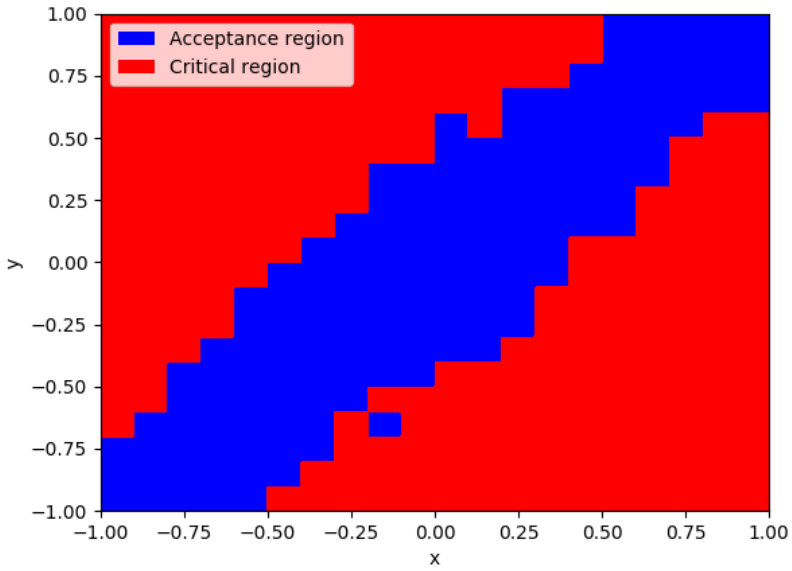

Figure 15: Acceptance region and critical region in the (x, y) plane for the estimated likelihood ratio

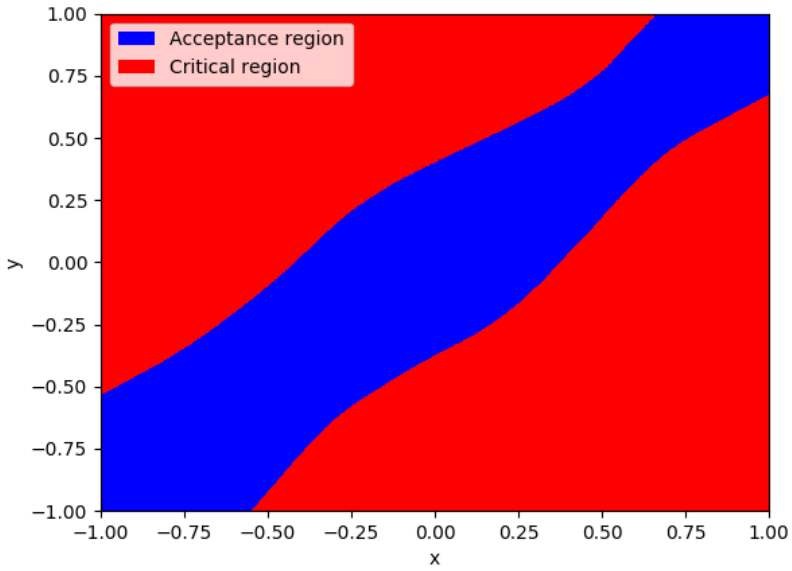

Figure 16: Acceptance region and critical region in the (x, y) plane for the neural network

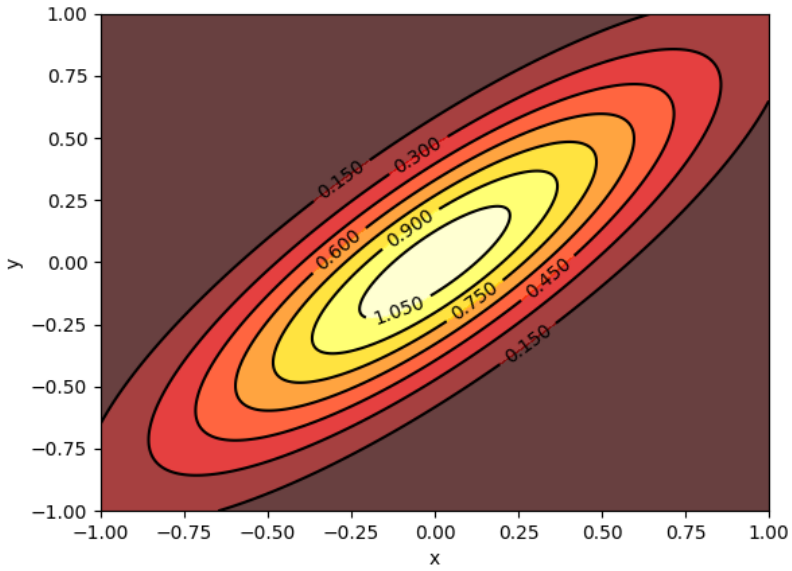

Figure 17: Joint pdf for the spatial coordinates for signal events $f_s(x, y)$ as a function of x and y

The critical region (red) is the region where the null hypothesis H_0 is rejected. The other region is the acceptance region (blue) where H_0 is accepted. We want to define these regions such as the probability to make an error is minimum.

Starting from the most optimal test, the exact likelihood ratio (Fig. 14), the shape of the acceptance region coincides approximately with the shape of the pdf $f(x|H_0)$, i.e. $f_s(x, y)$ (Fig. 17), since in this case the errors are minimal. It is clear from the definition of this test that if $T > T_{\text{cut}}$, this expresses that the data is more likely to be under the model $f_b(x.y)$ than the model $f_s(x, y)$. Therefore, this value is set in the critical region. Otherwise, if $T < T_{\text{cut}}$.

In the case of the neural network (Fig. 16), we assume a general parametrization (non-linear) with a great amount of parameters. Therefore, as we have said, we expect to obtain a shape very similar to the exact likelihood ratio.

In the case of the estimated likelihood ratio (Fig. 15), we also find a similar shape since we are performing a general parametrization using the histograms obtained from the train sample. Therefore, it is clear that the shape will depend on the number of bins considered in these histograms. More bins will imply more free parameters and then a best approximation to the exact likelihood ratio test. In our case, we have just considered 2D histograms of 20×20 bins, and that is why we present a shape not as smooth as the others.

In the case of the Fisher discriminant (Fig. 13), the test statistic is constructed as a linear function of the data. The good point of this test is that the only parameters to estimate are the means and covariances. However, it is clear that restricting the test to only have linear combinations causes, except some cases (multidimensional Gaussians with common covariances matrices) to not be a very optimal test statistic. As we can observe, the shape is not similar to the exact likelihood ratio and it shows a linear combination of r and θ (polar coordinates) with weight parameters $a_1 = 1.384$ and $a_2 = -0.419$ respectively (computed in part (a)). This implies an acceptance region with higher radius than the radial distance test, but modified by $a_2 \tan^{-1}(y/x)$. This second term has an important contribution in the second and fourth quadrants, which imply a higher value of T and then critical region.

In the case of the radial distance, the test is just constructed as the equation of the circle centred at $(0, 0)$. Since we do not use free parameters, it is obvious that the shape does not look like the shape of the exact likelihood ratio test.

(g) So far, we have been training, testing and evaluating the classifiers using data samples with known signal and background composition. Now we are going to use the optimal trained test statistic to classify unknown data samples, i.e. select signal-enriched subsamples out of the data.

As an optimal test, we choose the exact likelihood ratio. Then, given the data, we compute the values of $T(x)$. In order to select a signal-enriched subsample, we select those values for which $T(x) \leq T_{\text{cut}}$, since we know that this region corresponds to the acceptance region (H_0 is accepted). Considering $T_{\text{cut}} = 0.8501$, we select a subsample signal-enriched of 385 events out of a total group of 1047 events for the On data sample, and we select a subsample of 1003 events out of a total group of 2971 events for the Off data sample.

The resulting subsamples of events contain more signal events than background events, but still contain both classes. The reason of that is because we have a certain probability to make an error α or β. Therefore, it is clear that in these subsamples we will have background events coming from the type II error (H_0 is accepted but H_0 is not true). Then we can estimate the number of background events in the subsample as $N_{\text{select}}\,\beta$ and the number of signal events as $N_{\text{select}} - N_{\text{select}}\,\beta$.

In order to compute the probability of type II error, we need to use a sample under the hypothesis H_1 (background). We can use the test bkg sample as we did in part (c), or we can use the Off data sample, which is measured in conditions such that only background events are collected. As expected, we obtain very similar results in both cases: $\beta_{\text{testbkg}} = 0.4155$, $\beta_{\text{offdata}} = 0.4124$. Therefore, using β_{offdata} we obtain:

	signal-enriched subsample	signal events	background events
On data	385	226	159
Off data	1003	589	414

Table 3: Estimated number of signal and background events in the signal-enriched subsample

where the important result is for the On data sample. The Off data sample is the background control sample (only background events are collected). Instead, if we estimate the number of signal events ($\hat{s}$) and the number of background events ($\hat{b}$) in the measured On data applying the method of maximum likelihood to the joint likelihood obtained considering N_{on} and N_{off} as Poisson random variables, we get:

$$\hat{s} = N_{\text{on}} - \frac{1}{\tau} N_{\text{off}} = 51 \qquad\qquad \hat{b} = \frac{1}{\tau} N_{\text{off}} = 334 \tag{11}$$

which is quite different. Finally, we are going to compute the statistical significance for the presence of the signal, i.e. the significance for $s = 0$ (no signal events). We compute it in units of σ as

$$S = \sqrt{2}\left\{ N_{\text{on}} \ln\left[(\tau{+}1)\left(\frac{N_{\text{on}}}{N_{\text{on}} + N_{\text{off}}}\right)\right] + N_{\text{off}} \ln\left[\left(\frac{1+\tau}{\tau}\right)\left(\frac{N_{\text{off}}}{N_{\text{on}} + N_{\text{off}}}\right)\right]\right\}^{1/2} \tag{12}$$

where N_{on} is the number of events measured in the On data sample ($N_{\text{on}} = 385$), N_{off} is the number of events measured in the Off data sample ($N_{\text{off}} = 1003$) and τ is the ratio between Off and On exposures, i.e. how much more time we have been measuring the Off data with respect to the On data ($\tau = 3$). We obtain:

$$S = 2.3284\,\sigma \tag{13}$$

Exercise 2

In this second step, we are going to use the provided spectral information to try to increase the significance of signal detection. We know these energy spectrum for signal and background, however, they contain some undetermined parameters which are not of our interest but need to be included in our hypothesis (nuisance parameters). Therefore, we need to use a test that does not depend on these parameters: the profile likelihood ratio.

(a) In order to use the profile likelihood ratio, first we need to write the expression for the extended joint likelihood function. In our case, this is given by

$$\begin{aligned} &L(s, b, E_0, \gamma | E_{i=1}, \dots E_{i=N_{\text{on}}}, E_{j=1}, \dots E_{j=N_{\text{off}}}) \\ &= \frac{(s+b)^{N_{\text{on}}}}{N_{\text{on}}!} e^{-(s+b)} \frac{(\tau b)^{N_{\text{off}}}}{N_{\text{off}}!} e^{-(\tau b)} \prod_{i=1}^{N_{\text{on}}} f(E_i | s, b, E_0, \gamma) \prod_{j=1}^{N_{\text{off}}} k(E_j | \gamma) \end{aligned} \tag{14}$$

where

s = number of signal events in the On data

b = number of background events in the On data

$N_{\text{on}}, N_{\text{off}}$ = observed number of events in the On and Off datasets

$h(E)$ = pdf for signal events

$k(E)$ = pdf for background events

$f(E)$ = pdf for events in the On data (signal + background)

E_0, γ = undetermined parameters of $h(E)$ and $k(E)$

We know the pdfs $h(E)$ and $k(E)$:

$$h(E) = \frac{1}{N_h} \exp\left(-\frac{1}{2} \left(\frac{E - E_0}{\sigma_E} \right)^2 \right) \tag{15}$$

$$k(E) = \frac{1}{N_k}(2 + \gamma E) \tag{16}$$

where $E \in [0, 10]$, $E_0 \in [0, 10]$, $\sigma_E = 1$ and $\gamma > -1/5$. N_h and N_k are the normalization constants, however, we do not need to worry about them since they will cancel out in the likelihood profile ratio test. We can write the pdf $f(E)$ as

$$f(E) = \frac{1}{N_f} \frac{s\, h(E) + b\, k(E)}{s + b} \tag{17}$$

Replacing this into Eq. 14, we obtain

$$\begin{aligned} L(s, b, E_0, \gamma | E_{i=1}, \dots E_{i=N_{\text{on}}}, E_{j=1}, \dots E_{j=N_{\text{off}}}) &= \frac{(s+b)^{N_{\text{on}}}}{N_{\text{on}}!} e^{-(s+b)} \frac{(\tau b)^{N_{\text{off}}}}{N_{\text{off}}!} e^{-(\tau b)} \\ \cdot \prod_{i=1}^{N_{\text{on}}} \frac{1}{s+b} \left[s \exp\left(-\frac{1}{2} \left(\frac{E_i - E_0}{\sigma_E} \right)^2 \right) + b(2 + \gamma\, E_i) \right] &\prod_{j=1}^{N_{\text{off}}} (2 + \gamma\, E_j) \end{aligned} \tag{18}$$

where we have not considered the normalization constants.

(b) As we can observe (Eq. 18), our extended joint likelihood function has 4 parameters (s, b, E_0, γ). Our parameter of interest is s, so we are going to consider it as a free parameter. On the other hand, we consider (b, E_0, γ) as nuisance parameters. Having said that, we define the profile likelihood ratio test as

$$\lambda_p(s) = \frac{L(s, \hat{\hat{b}}, \hat{\hat{E}}_0, \hat{\hat{\gamma}})}{L(\hat{s}, \hat{b}, \hat{E}_0, \hat{\gamma})} \tag{19}$$

where $\hat{\nu}$ maximizes L and $\hat{\hat{\nu}}(s)$ maximizes L for a given s. Therefore, when computing $\lambda_p(s)$, we could think in deducing all the $\hat{\nu}$ and $\hat{\hat{\nu}}(s)$ applying the method of maximum likelihood. However, in our case we cannot do this analytically. In fact, there is no need to do it. We only need to compute the maximum value of $L(s, b, E_0, \gamma)$ given the four parameters (this is the denominator value), and the maximum value of $L(b, E_0, \gamma)$ for each value of s (this is the numerator value). Then, we perform the division and we obtain the value of $\lambda_p(s)$ for the given s. We repeat this process for a range of s values.

Before computing the maximum value of $L(s, b, E_0, \gamma)$, we need to make some manipulations: notice that in Eq. 18 we have the terms $\prod_{i=1}^{N_{\text{on}}} f(E_i)$ and $\prod_{j=1}^{N_{\text{off}}} k(E_j)$. Thus it is clear that we have to select the values of energy for which $T(x, y) < T_{\text{cut}}$ from the On data and the Off data respectively. Furthermore, we find numerical overflow problems maximizing the likelihood function. In order to solve this, we maximize the log-likelihood. We can do this because the parameters that maximize L will also maximize $\log L$. Note that there are no modules for maximize computationally. However, maximize a function is the same than minimize the -function. After simplifying the $\log L$, we obtain

$$\begin{aligned} \log L(s, b, E_0, \gamma) = &- \log(N_{\text{on}}!) - (s + b) + N_{\text{off}} \log(\tau\, b) - \log(N_{\text{off}}!) - \tau\, b \\ &+ \sum_{i=1}^{N_{\text{on}}} \log \left[s \, \exp\left(-\frac{1}{2}\left(\frac{E_i - E_0}{\sigma_E} \right)^2 \right) + b(2 + \gamma\, E_i) \right] + \sum_{j=1}^{N_{\text{off}}} \log(2 + \gamma\, E_j) \end{aligned} \tag{20}$$

We will minimize this expression multiplied by (-1). In this minimization, we will set the domains of s and b to $(0, \infty)$, since negative values are physically meaningless and they cause problems with the logarithms. Also, we will set the corresponding bounds to E_0 and γ.

Considering all this, we compute $-2 \log \lambda_p(s)$ for the different values of s. The distribution of $-2 \log \lambda_p(s)$ approaches (for large data samples) a χ^2-distribution of 1 degree of freedom, as stated by the Wilks' theorem.

Eventually (part (c)), we want to determine the value of s (parameter estimation), but first we want to make sure that our data are incompatible with the hypothesis H_1: there are no signal events in our On data (background only hypothesis). In order to do so, we compute the p-value or the significance for $s = 0$ (no signal events). Our significance in units of σ is

$$S = \sqrt{-2 \log \lambda_p(s = 0)} = 19.14\, \sigma \tag{21}$$

Comparing with our previous result $S = 2.33\, \sigma$, we note a great increment. The required significance for H_1 rejection is usually $5\, \sigma$. Therefore, now we are in conditions to reject H_1 and to claim for a new discovery (signal event detection). This significance for the signal means to set a threshold such that the probability to find greater values of $-2 \log \lambda_p(s)$ in the occurrence of absence of a signal is less than the corresponding p-value (in our case a very small value).

(c) Finally, we are going to determine the best fit values for the parameters. These values will be those for which we obtain the maximum value of $-2\log\lambda_p(s)$ (most likely observations). First we seek the value of s which maximizes the profile likelihood function. Then we look for the values of b, E_0 and γ that maximize the $\log L$ given this s. The results are shown in Table 4.

$\hat{s}$	$\hat{b}$	$\hat{E}_0$	$\hat{\gamma}$
38	347	10	1.1×10^8

Table 4: Parameter estimations using the profile likelihood ratio test

As we can observe, the values obtained are consistent with the observed number of events in the On dataset $N_{\text{on}} = 385$. If we compare these estimations with the previous ones ($\hat{s} = 51$, $\hat{b} = 334$), we note a difference. Therefore, we conclude that the introduction of the spectral information has been useful to achieve more accurate estimations.

Unfolding

Exercise 1

We are going to simulate and solve an unfolding problem. The unfolding is the procedure of correcting the distortions that appear in a distribution when the values of these variables are subject to additional random fluctuations due to the limited resolution of the measuring device, i.e. the measured values may differ in a random way from the values that were actually created (true values). In this first exercise we are going to generate the input elements for the unfolding and compute non-regularized solutions.

(a) The goal of the unfolding is to estimate the true distribution $f_{\text{true}}(y)$ of a given random variable y (true value, which we do not know) out of the measured one given by the observed values x. In general, one usually represents these two distributions as histograms of M and N bins respectively. We are going to consider $M = N = 20$ and a range of $x, y \in [0, 100]$ for each histogram.

In this first part, we are going to construct the response matrix R. The response matrix element R_{ij} is defined as the conditional probability that an event will be found with measured value x in bin i given that the true value y was in bin j. This is given by

$$R_{ij} = \frac{\int_{\text{bin i}} dx \int_{\text{bin j}} dy\, s(x|y)\, \varepsilon(y)\, f_{\text{true}}(y)}{\int_{\text{bin j}} dy\, f_{\text{true}}(y)} \tag{22}$$

where $s(x|y)$ is called the resolution function and $\varepsilon(y)$ is the detection efficiency (probability that an event leads to some measured value). Note that R_{ij} depends on $f_{\text{true}}(y)$, which is a priori unknown. However, if we consider the bins of the unfolded histogram small enough, $s(x|y)$ and $\varepsilon(y)$ are approximately constant over the bin of y, so the dependence on $f_{\text{true}}(y)$ cancels out. Then we obtain

$$R_{ij} = \int_{\text{bin i}} dx\, s(x|y)\, \varepsilon(y) \tag{23}$$

We are going to assume that $s(x|y)$ is a normalized Gaussian with no bias (centred at y) and with σ the resolution of our instrument, which we are going to set two times the width of the bins. Also, we are going to take all efficiencies $\varepsilon(y)$ to be unity. Considering this, we compute each element R_{ij} performing the integral over the bin i (variable x) and fixing the value of y as the center value of the corresponding bin j. The obtained response matrix is shown in Fig. 18.

(b) We are going to construct a histogram with $M = 20$ bins to represent the distribution $f_{\text{true}}(y)$. We assume that $f_{\text{true}}(y)$ is given by the sum of two normalized Gaussian functions with $\mu_1 = 30$, $\sigma_1 = 10$, $\mu_2 = 70$, $\sigma_2 = 10$. In order to construct the histogram, we compute the probabilities to find y in bin j given by

$$p_j = \int_{\text{bin j}} dy\, f_{\text{true}}(y) \tag{24}$$

We will assume that the expected value of the total number of events is $\mu_{tot} = 25000$. Then, the expected number of events in bin j is $\mu_j = \mu_{tot}\, p_j$.

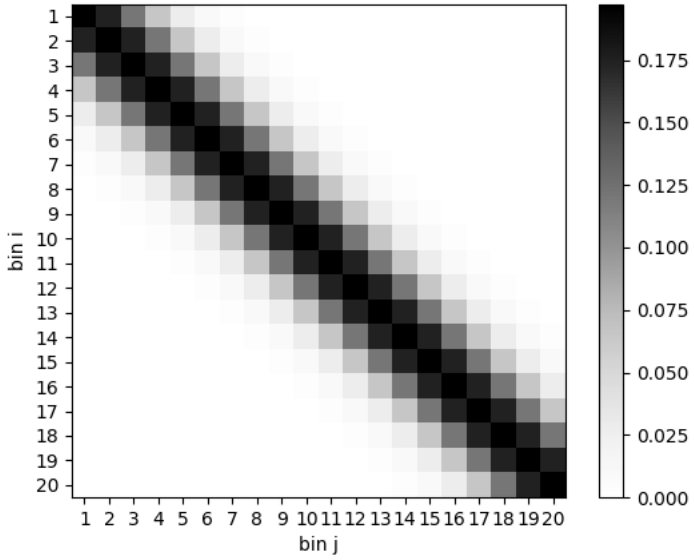

Figure 18: Response matrix R for $N = M = 20$ bins and range $x, y \in [0, 100]$

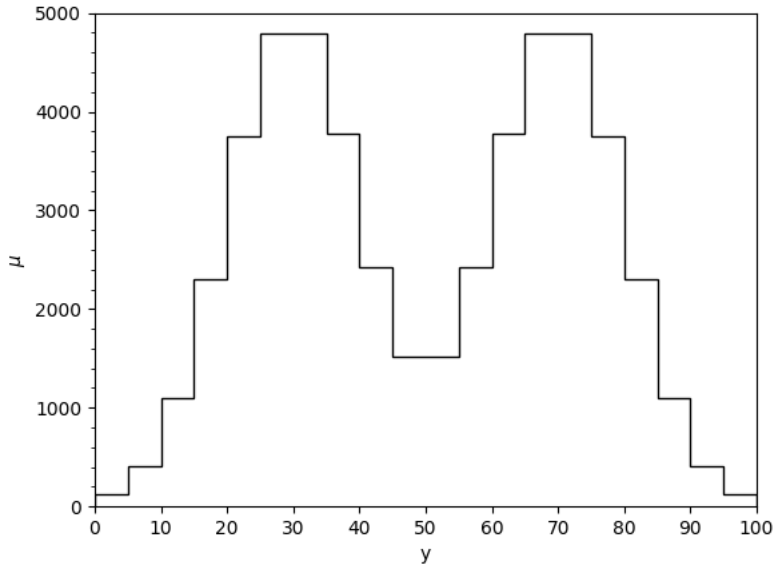

Figure 19: Histogram of the expectation values of the true numbers of entries, $\boldsymbol{\mu} = (\mu_1, \mu_2, \dots \mu_M)$ where $M = 20$

(c) We are going to construct the histogram of the expectation values of the measured number of entries, $\boldsymbol{\nu} = (\nu_1, \nu_2, \dots \nu_N)$. The vectors of expected measured and true values are related by the response matrix by the following relation (assuming no background):

$$\nu_i = \sum_{j=1}^{M} R_{ij}\, \mu_j \tag{25}$$

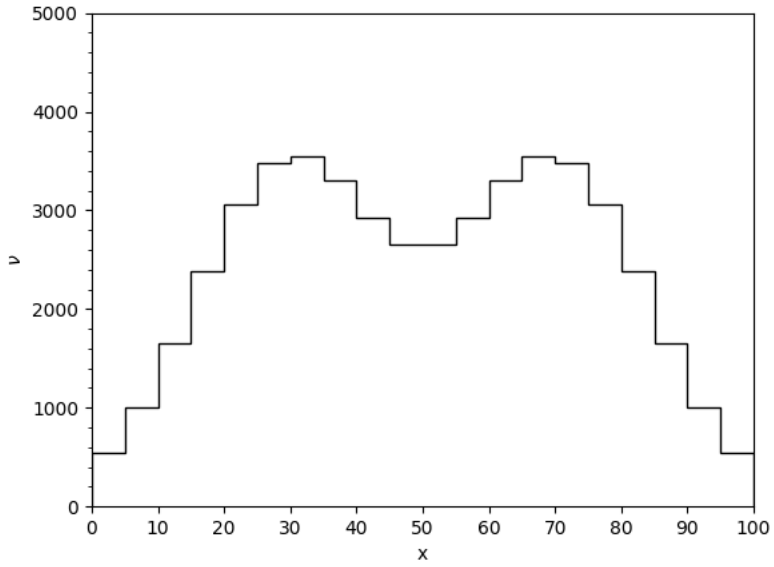

Figure 20: Histogram of the expectation values of the measured number of entries, $\boldsymbol{\nu} = (\nu_1, \nu_2, \ldots \nu_N)$ where $N = 20$

(d) We are going to generate the histogram of the actual number of entries observed (the data), $\boldsymbol{n} = (n_1, n_2, \ldots n_N)$. It is often possible to regard the variables n_i as independent Poisson variables with expectation values ν_i. Therefore, we generate randomly n_i using the module numpy.random.poisson with the expectation values ν_i computed in part (c).

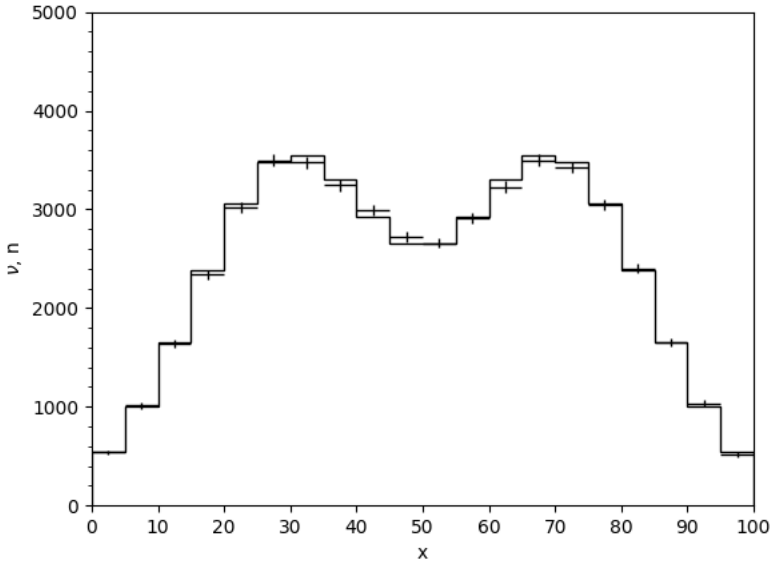

Figure 21: Histogram of $\boldsymbol{\nu}$ compared with the histogram of $\boldsymbol{n}$ represented as points with its error bars

As we can observe, the distribution $\boldsymbol{n}$ does not coincide exactly with the distribution $\boldsymbol{\nu}$ since there are statistical fluctuations due to the randomness. We have represented these statistical errors with error bars, which indicate the standard deviations for each bin. In order to compute these standard deviations, we have computed the covariance matrix $V_{ij} = \mathrm{cov}[n_i, n_j]$, which for independent Poisson variables is just $V_{ij} = \nu_i \, \delta_{ij}$. Then we calculate the square root of the diagonal terms.

(e) We are going to construct the exact solution $\boldsymbol{\mu_0}$ using the inverse of the response matrix. The method of inverting R is one of the methods for constructing estimators for the true histogram $\boldsymbol{\mu}$. Assuming no background, one gets $\hat{\boldsymbol{\mu}}$ as

$$\hat{\boldsymbol{\mu}} = R^{-1} \, \hat{\boldsymbol{\nu}} \tag{26}$$

In general, we do not know $\boldsymbol{\nu}$. Rather, we only have the data $\boldsymbol{n}$. So we have to estimate $\hat{\boldsymbol{\nu}} = \boldsymbol{n}$. Therefore, the exact solution given the corresponding data is $\hat{\boldsymbol{\mu_0}} = R^{-1} \, \boldsymbol{n}$. Notice that here we have only one exact solution for this equation because $N = M$.

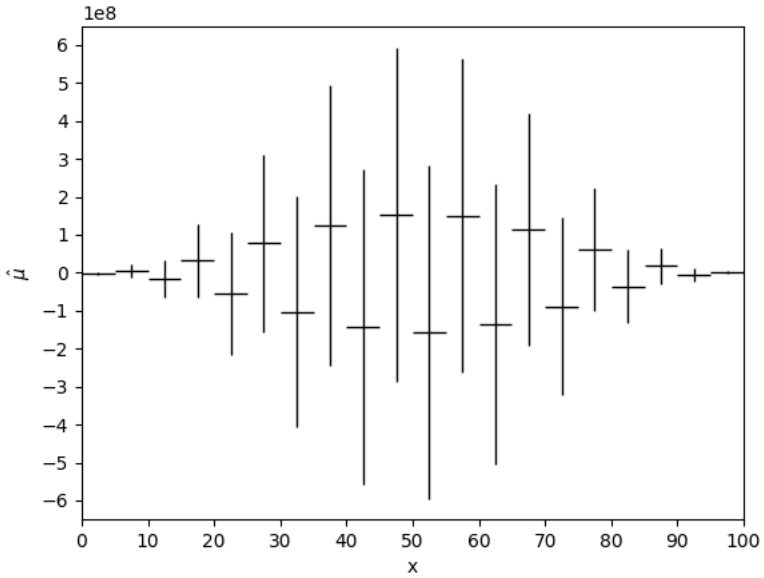

Figure 22: The estimators $\hat{\boldsymbol{\mu}}$ obtained from the inversion of the response matrix R given $\hat{\boldsymbol{\nu}} = \boldsymbol{n}$

As we can observe, far from achieving the precision that we had for the n_i, the $\hat{\mu}_j$ oscillate wildy from bin to bin and the error bars are as large as the estimated values themselves. In order to compute the standard deviations of $\hat{\boldsymbol{\mu}}$ (error bars), we have computed the covariance matrix $U_{ij} = \mathrm{cov}[\hat{\mu}_i, \hat{\mu}_j]$ given by $U = R^{-1} \, V \, (R^{-1})^T$ in matrix notation, where $V_{ij} = \mathrm{cov}[n_i, n_j]$. Then, we have selected the diagonal terms of U and calculated the square root.

Although the solution in Fig. 22 bears little resemblance to the true distribution, the estimators $\hat{\boldsymbol{\mu}}$ have zero bias and the smallest possible variance for any choice of estimator with zero bias. We will see that we will improve our estimators introducing some amount of bias so that the variance is reduced to an acceptable level. We will achieve this when we regularize the unfolded distribution (Exercise 2).

(f) We are going to compute numerically the maximum likelihood solution and compare it to the exact solution $\hat{\boldsymbol{\nu}} = \boldsymbol{n}$. In our case, the log-likelihood function is defined as

$$\log L(\boldsymbol{\mu}) = \sum_{i=1}^{N} \log p(n_i; \nu_i) = \sum_{i=1}^{N} \log \left(\frac{\nu_i^{n_i} \, e^{-\nu_i}}{n_i!} \right) \tag{27}$$

It is clear that in this case, one could compute the ML estimator analytically and obtain $\hat{\boldsymbol{\nu}} = \boldsymbol{n}$. We are going to prove this numerically: we define the function and we maximize it (minimize -function) over the 20 variables $\boldsymbol{\nu} = (\nu_1, \nu_2, \dots \nu_N)$.

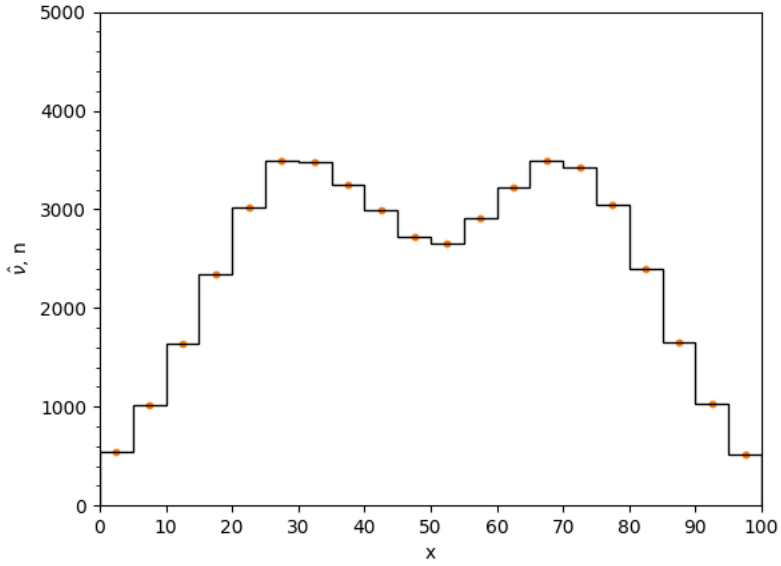

Figure 23: Histogram of $\boldsymbol{n}$ compared with the values of $\hat{\boldsymbol{\nu}}$ represented as points

(g) We are going to use the method of correction factors for constructing a estimator for $\boldsymbol{\mu}$. Assuming no background and $M = N$, one gets $\hat{\boldsymbol{\mu}}$ as

$$\hat{\mu}_i = C_i \, n_i \qquad \text{where } i = 1, 2, \dots M \tag{28}$$

C_i is the multiplicative correction factor and it is defined as $C_i = \mu_i^{MC} / \nu_i^{MC}$. This method is iterative, which means that for every iteration we obtain a different $\boldsymbol{C}$ and therefore a different $\hat{\boldsymbol{\mu}}$. In the first iteration, we use the $\boldsymbol{n}$ computed in part (d) as $\boldsymbol{\mu}^{MC}$ and $\boldsymbol{\nu}^{MC} = R\boldsymbol{\mu}^{MC}$. Then we compute $\boldsymbol{C}$ and $\hat{\boldsymbol{\mu}}$. In further iterations, we use the $\hat{\boldsymbol{\mu}}$ computed in the previous iteration as $\boldsymbol{\mu}^{MC}$ and then we proceed analogously.

As we can observe (Fig. 24), we have assumed that the statistical errors in the correction factors are negligible. Therefore, we do not have to incorporate the uncertainties of C_i into the uncertainties of the estimates $\hat{\mu}_i$. In order to compute the standard deviations of $\hat{\boldsymbol{\mu}}$, we have computed the covariance matrix $U_{ij} = \text{cov}[\hat{\mu}_i, \hat{\mu}_j] = C_i^2 \, \text{cov}[n_i, n_j] = C_i^2 \, \nu_i \, \delta_{ij}$.

It is clear that we have improved our results with respect to the method of inverting the response matrix (Fig. 22). We can observe a reduction of the variance. The price that one pays is a non-zero bias. However, convergence to the true distribution is not assured. For instance, in our case, if we apply the method for 20 iterations we do not obtain better results than the case of 5 iterations. Again, regularization is needed.

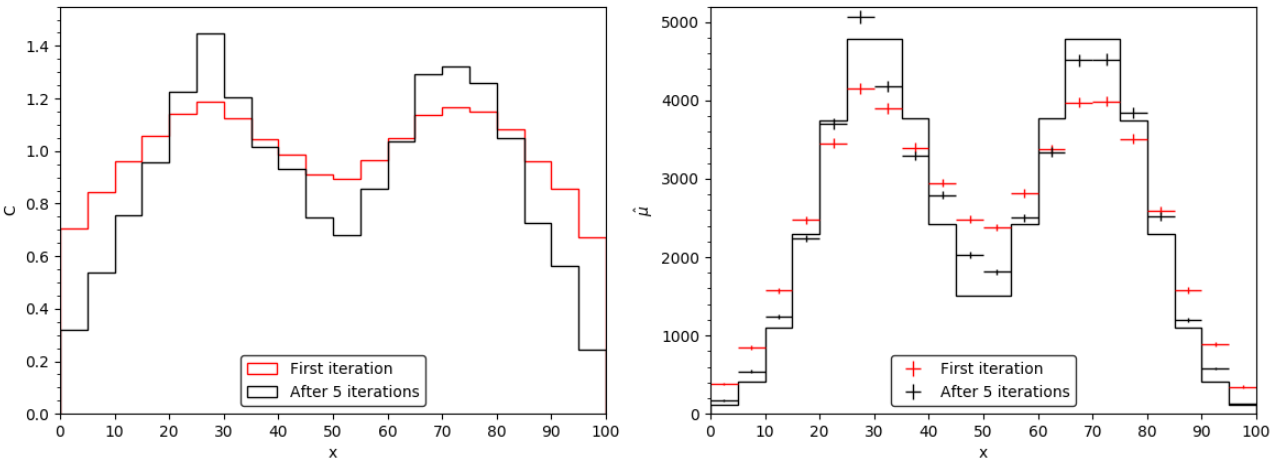

Figure 24: On the left, histogram of the correction factors $\boldsymbol{C}$ for different number of iterations. On the right, histogram of $\boldsymbol{\mu}$ (true distribution) compared with the histogram of $\hat{\boldsymbol{\mu}}$ represented as points with its error bars, for different number of iterations.

(h) We are going to compute the solution for $f_{\text{true}}(y)$ using the forward unfolding method. This method provides the most robust solution when the premise of knowing the shape of $f_{\text{true}}(y;\boldsymbol{\theta})$ is valid, where $\boldsymbol{\theta}$ are parameters. In our case, in part (b) we considered that $f_{\text{true}}(y;\boldsymbol{\theta})$ was the sum of two normalized Gaussian functions. Therefore we have 6 parameters: the means, variances and normalization constants of both Gaussians. This method consists in estimate $\boldsymbol{\theta}$ using the minimum of least-squares, i.e. minimizing

$$\chi^2(\boldsymbol{\theta}) = \sum_{ij=1}^{N} \left(\sum_{k=1}^{M} R_{ik}\, \mu_k(\boldsymbol{\theta}) - n_i \right) \left(V^{-1}\right)_{ij} \left(\sum_{l=1}^{M} R_{jl}\, \mu_l(\boldsymbol{\theta}) - n_j \right) \tag{29}$$

where $V_{ij} = \text{cov}[n_i, n_j] = \nu_i \delta_{ij}$ and

$$\mu_j(\boldsymbol{\theta}) = \mu_{tot}\, p_j(\boldsymbol{\theta}) = \mu_{tot} \int_{\text{bin j}} dy\, f_{\text{true}}(y;\boldsymbol{\theta}) \tag{30}$$

In order to compute this, we define a function for $\boldsymbol{\mu}(\boldsymbol{\theta})$ which computes the integral as we did in part (b). Then, we construct the function $\chi^2(\boldsymbol{\theta})$ which also calls the function $\boldsymbol{\mu}(\boldsymbol{\theta})$. Once we have this, we minimize $\chi^2(\boldsymbol{\theta})$ over the 6 variables $\boldsymbol{\theta}$. The obtained $\hat{\boldsymbol{\theta}}$ are shown in Table 5. Finally, we compute $\boldsymbol{\mu}(\hat{\boldsymbol{\theta}})$ using the function we created and we obtain Fig. 25 (left), or we compute $f_{\text{true}}(y;\hat{\boldsymbol{\theta}})$ and obtain Fig. 25 (right).

	True parameters	Estimated parameters
N_1	0.0399	0.0388
μ_1	30	30.3
σ_1	10	10.3
N_2	0.0399	0.0395
μ_2	70	70.3
σ_2	10	9.9

Table 5: Values of the true parameters $\boldsymbol{\theta}$ and the estimated parameters $\hat{\boldsymbol{\theta}}$ obtained using the forward unfolding method

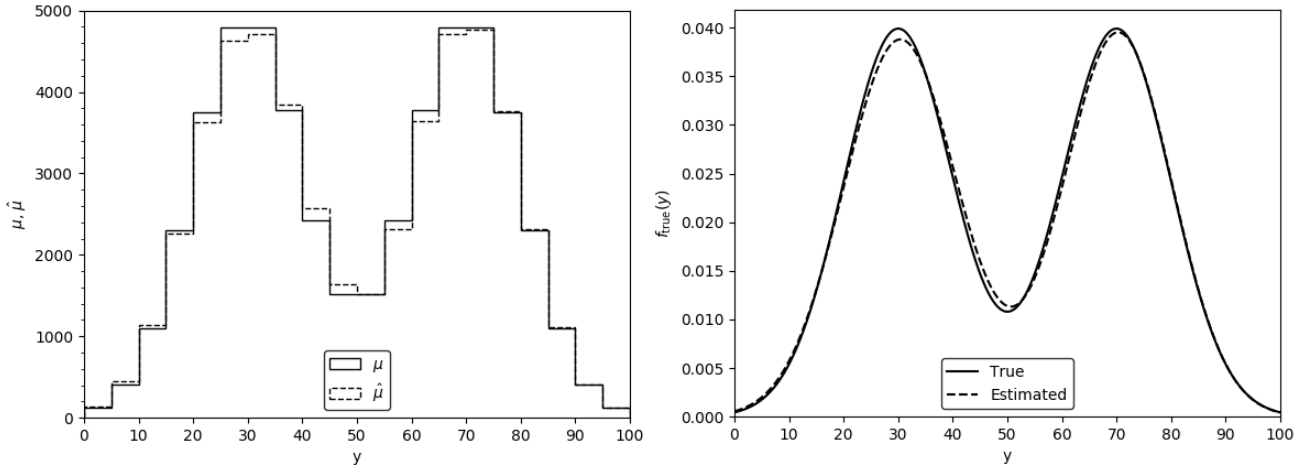

Figure 25: On the left, histogram of the true distribution $\boldsymbol{\mu}$ compared to the estimated one $\hat{\boldsymbol{\mu}}$. On the right, true distribution $f_{\text{true}}(y)$ compared to the estimated one $f_{\text{true}}(y; \hat{\boldsymbol{\theta}})$.

Exercise 2

In this second exercise, we are going to compute regularized solutions using the input elements for the unfolding generated in Exercise 1. The regularization consists in imposing a measure of smoothness on the estimators for the true distribution $\boldsymbol{\mu}$ in order to improve the solutions. This smoothness allows additional bias in a controlled way in exchange for a variance reduction.

(a) We are going to consider the migration matrix R and the distribution of measured values $\boldsymbol{n}$ produced in Exercise 1. On the other hand, we assume $\boldsymbol{\mu}$ and $\boldsymbol{\nu}$ are unknown.

(b) The measure of smoothness is given by the regularization function $S(\boldsymbol{\mu})$. There are several forms for this function. We are going to chose the Tykhonov regularization method, for which

$$S[f_{\text{true}}(y)] = -\int \left(\frac{d^k f_{\text{true}}(y)}{dy^k} \right)^2 dy \tag{31}$$

(c) We want to obtain the solution with the highest value of $S(\boldsymbol{\mu})$. However, not all the values are allowed, in the sense that the $\boldsymbol{\mu}$ must have an acceptance level of agreement between the predicted expectation values $\boldsymbol{\nu}$ and the data $\boldsymbol{n}$. This acceptance region is determined by

$$\log L(\boldsymbol{\mu}) \geq \log L_{\max} - \Delta \log L \tag{32}$$

Taking into account this constraint, we will find the solution $\hat{\boldsymbol{\mu}}_\alpha$ by finding the maximum of

$$\varphi(\boldsymbol{\mu}, \lambda) = \alpha \, \log L(\boldsymbol{\mu}) + S(\boldsymbol{\mu}) + \lambda \left[n_{\text{tot}} - \sum_{i=1}^{N} \nu_i \right] \tag{33}$$

where α is the regularization parameter and we have also imposed the condition $\hat{\nu}_{\text{tot}} = \sum_{i=1}^{N} \hat{\nu}_i = n_{\text{tot}}$. Note that we no longer have $\hat{\boldsymbol{\nu}} = \boldsymbol{n}$.

In our case, remember that n_i are independent Poisson variables, thus, dropping terms that do not depend on $\boldsymbol{\mu}$, we obtain

$$\log L(\boldsymbol{\mu}) = \sum_{i=1}^{N} (n_i \, \log \nu_i - \nu_i) \tag{34}$$

We also apply the relation $\nu_i = \sum_{j=1}^{M} R_{ij} \, \mu_j$ which is always true. We define the Tykhonov's regularisation function as

$$S(\boldsymbol{\mu}) = -\boldsymbol{\mu}^T \, G \, \boldsymbol{\mu} \tag{35}$$

where G is a known symmetric matrix of $M \times M$ constants. We consider the case for $k = 2$. The corresponding G matrix can be found in the literature.

Once we have $\varphi(\boldsymbol{\mu}, \lambda)$ defined, we maximize (minimize -$\varphi(\boldsymbol{\mu}, \lambda)$) with respect to $\boldsymbol{\mu}$ and λ (21 variables) for a set of 500 values $\alpha \in [7 \times 10^4, 3 \times 10^6]$. The obtained $\hat{\boldsymbol{\mu}}_\alpha$ are the estimators of the true distribution for a given α. In the minimization computation we had to use an initial guess $\lambda = 1$, 30000 maximum iterations and the method of Nelder-Mead in order to assure the complete convergence.

(d) So far, we have computed the estimator $\hat{\boldsymbol{\mu}}_\alpha$ of the true distribution for a set of values α. Now we are going to chose the optimal regularisation parameter α for our estimator. The idea is to chose α so that the value of $\Delta \log L$, which measures the trade-off between the variance and the bias of the solution, is the most optimal. We compute this value as $\Delta \log L = \log L_{\max} - \log L(\hat{\boldsymbol{\mu}}_\alpha)$ in accordance to Eq. 32, where $\log L_{\max} = \log L(\hat{\boldsymbol{\mu}}_0)$ with $\hat{\boldsymbol{\mu}}_0$ the solution computed by inversion fo the response matrix (Exercise 1 part (e)).

There are various definitions to chose the optimal bias/variance trade-off ($\Delta \log L$). However, they involve the covariance of the estimators $U_{ij} = \mathrm{cov}[\hat{\mu}_i, \hat{\mu}_j]$, their bias $\hat{b}_i$, and the variance of their biases $W_{ij} = \mathrm{cov}[\hat{b}_i, \hat{b}_j]$. Therefore, first we must construct these definitions. The covariance matrix $\mathrm{cov}[\hat{\mu}_i, \hat{\mu}_j]$ is obtained as

$$\mathrm{cov}[\hat{\mu}_i, \hat{\mu}_j] = \sum_{k,l=1}^{N} \frac{\partial \hat{\mu}_i}{\partial n_k} \frac{\partial \hat{\mu}_j}{\partial n_l} \mathrm{cov}[n_k, n_l] \tag{36}$$

where

$$\frac{\partial \hat{\mu}_i}{\partial n_j} = -(A^{-1}B)_{ij} \equiv C_{ij} \tag{37}$$

We compute the matrices A and B as shown in the literature, taking into account that in our case

$$\frac{\partial^2 \varphi}{\partial \mu_i \partial \mu_j} = -\alpha \sum_{k=1}^{N} \frac{n_k R_{ki} R_{kj}}{\nu_k^2} - 2G_{ij} \tag{38}$$

$$\frac{\partial^2 \varphi}{\partial \mu_i \partial n_j} = \alpha \frac{R_{ji}}{\nu_j} \tag{39}$$

We prepare a function that computes the matrix C for a given (μ, α). Once we have C, we compute U as

$$U = C\, V\, C^T \tag{40}$$

The bias of $\hat{\mu}_i$ is obtained as

$$\hat{b}_i = \sum_{j=1}^{N} C_{ij}\, (\hat{\nu}_j - n_j) \tag{41}$$

The covariance matrix for the bias $\hat{b}_i$ is

$$W = (CR - I)\, U\, (CR - I)^T \tag{42}$$

where I is the $M \times M$ unit matrix.

Once we have all this, we compute the quantities that allow us to chose the optimal $\Delta \log L$ and hence α. We are going to perform four methods: minimize the mean squared error (MSE), minimize the weighted MSE, increase the χ^2_{eff} in one unit and increase the χ^2_{b} in one unit. The definitions of those quantities are found in the literature. We construct a function of (μ, α) for each one. The results are shown in Fig. 26.

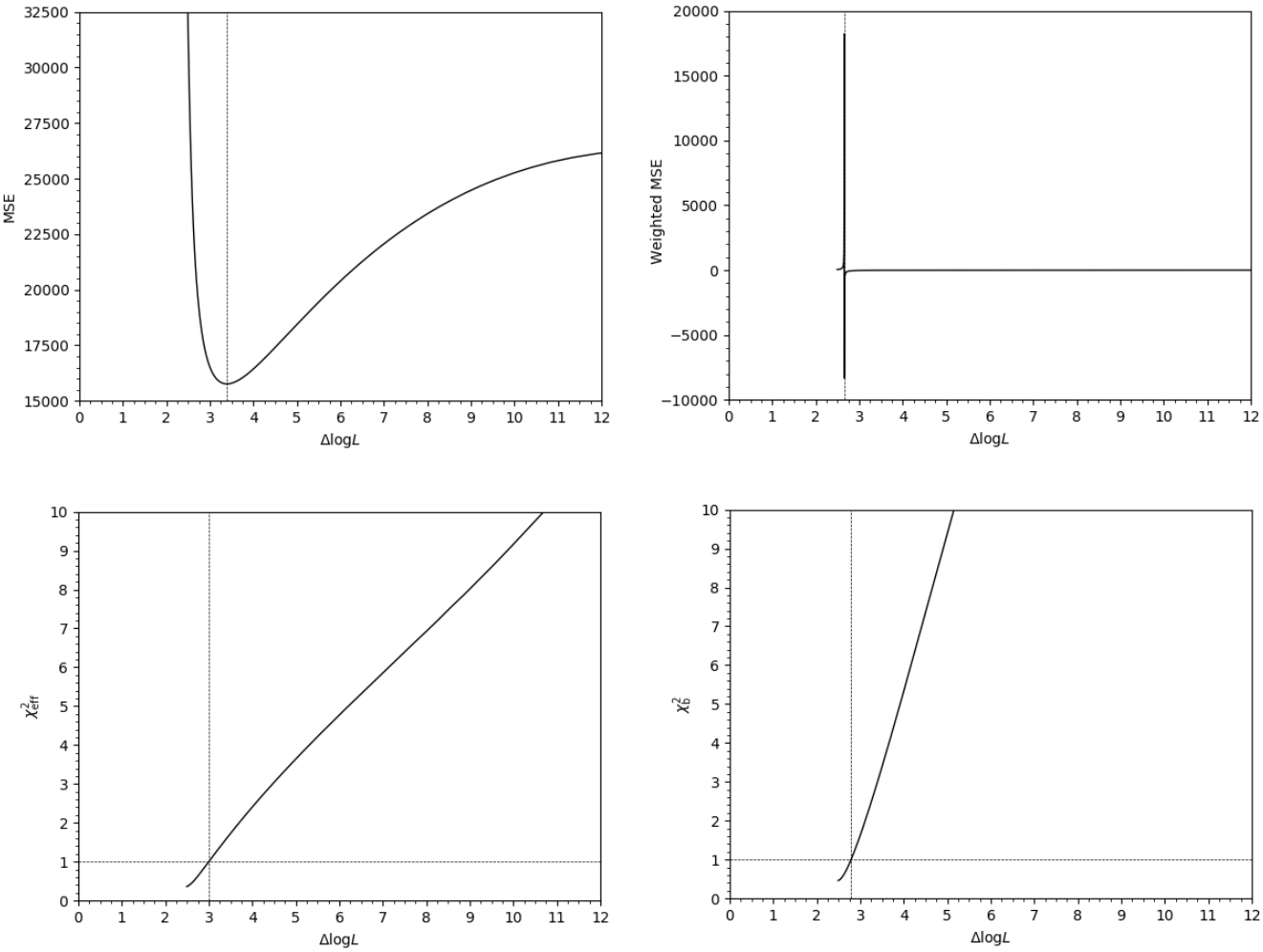

Figure 26: Different methods to chose the optimal $\Delta \log L$. The figures correspond to (a) MSE, (b) weighted MSE, (c) χ^2_{eff}, and (d) χ^2_{b}. The vertical lines show the chosen $\Delta \log L$.

As we can observe, we obtain values for the optimal $\Delta \log L$ very similar for the four methods, as expected. In practice, one only needs to select one of these methods. We can notice, that the shape of the weighted MSE (Fig. 26 (b)) looks a bit different than expected. This is because in the Tikhonov regularization, the estimates can become negative. In fact, it happens to our first and last bin of the obtained histograms $\hat{\boldsymbol{\mu}}_\alpha$. Since this bins also correspond to the small values of μ, then they have the bigger contribution to the sum that appear in the definition of the weighted MSE. As a consequence, the sum is negative for most of the values of α. However, although the shape is a bit different, the method is still functional and it gives the correct optimal $\Delta \log L$.

(e) Once we have selected the optimal α for each method, we compute its corresponding estimated true distribution $\hat{\boldsymbol{\mu}}_\alpha$ with its standard deviations (error bars), given by $\sqrt{U_{ii}}$. The results are shown in Fig. 27 (left).

(f) Finally we compute the estimated biases $\hat{b}_i$ and their standard deviations (error bars), given by $\sqrt{W_{ii}}$. The results are shown in Fig. 27 (right).

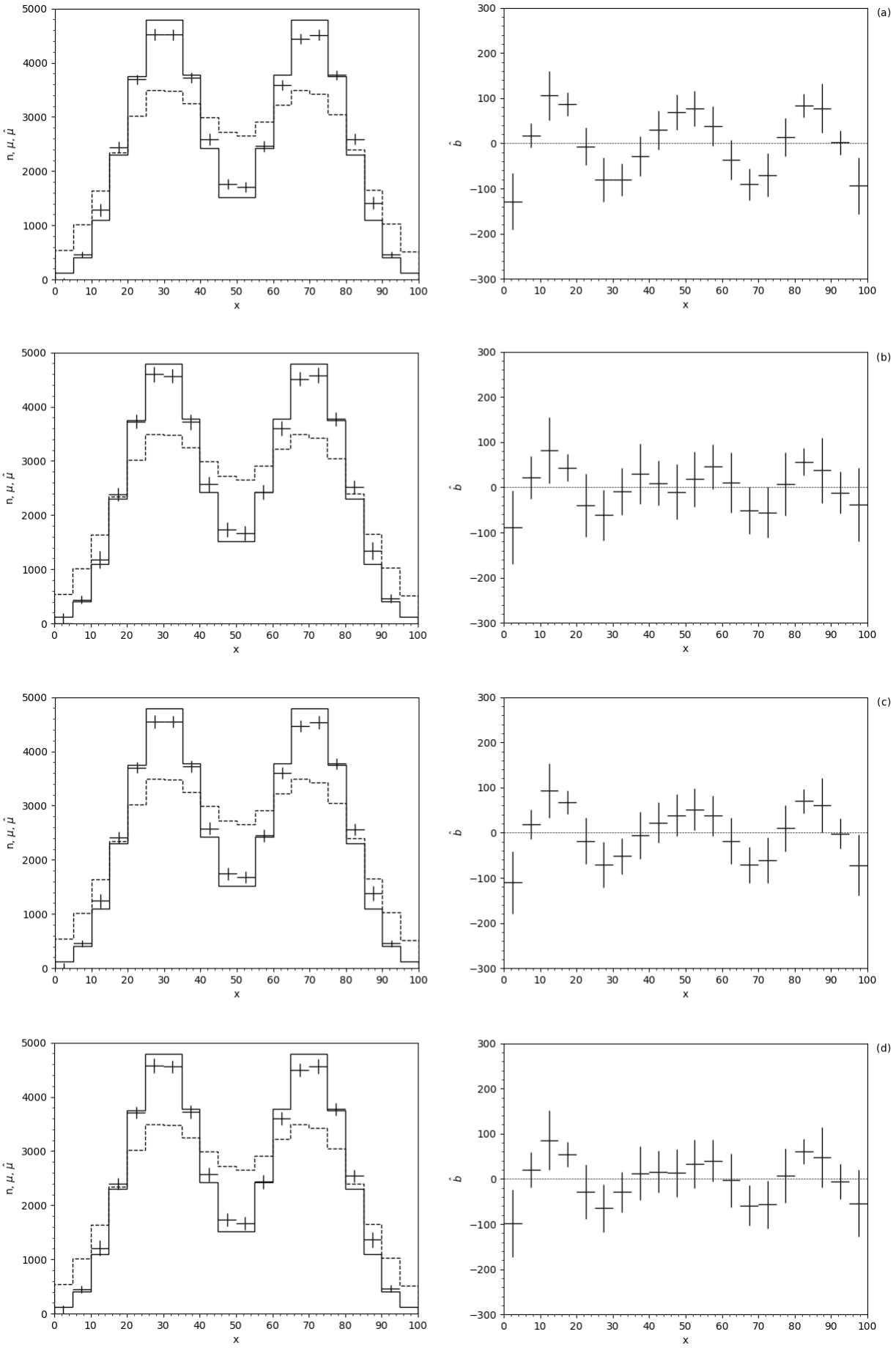

Figure 27: On the left, estimated true distribution $\hat{\boldsymbol{\mu}}$ shown as points with the true distribution $\boldsymbol{\mu}$ shown as a histogram and the measured distribution $\boldsymbol{n}$ shown as a dashed histogram. On the right, the estimated biases $\hat{\boldsymbol{b}}$. The figures correspond to (a) MSE, (b) weighted MSE, (c) χ^2_{eff}, and (d) χ^2_{b}.

As we can observe, the estimated true distribution $\hat{\boldsymbol{\mu}}$ fits approximately the true distribution $\boldsymbol{\mu}$ for every method. However, the solutions are not sufficiently good, since in some bins the $\hat{\mu}$ do not coincide with μ within its standard deviation. Also, note that we have obtained negative estimations for the first and last bin of the histogram, as we commented before.

In general, we have achieved an optimal bias/variance trade-off for every method. The biases are close to be consistent with zero, especially in the case of the weighted MSE, and the variances are not too large.

We have to consider that these estimations have been done considering that we knew nothing about $f_{\text{true}}(y)$ and with the approximation of cancelling out the $f_{\text{true}}(y)$ in the computation of R. We can observe that the solutions obtained are better than the solutions obtained with the inversion of the response matrix method (Fig. 22) and the correction factors method (Fig. 24), where no regularization was applied.