Statistical errors

Simone Albanesi

(Dated: June 3, 2022)

I. TASK

The goal of this notes is to show how we can estimate the error/confidence interval of a certain recovery/prediction. The problem can be formulated as follows. Consider a vector of variables y. Then we apply a certain transformation $\mathcal{N}$ so that we have a new array of variables $s = \mathcal{N}(y)$. Then we construct a new transformation $\mathcal{M}$ that satisfies $x = \mathcal{M}(s) = \mathcal{M}(\mathcal{N}(y))$ where x is closest as possible to y. If N is invertible, then the task is trivial since $\mathcal{M} = \mathcal{N}^{-1}$ and thus $x = y$. However in general $\mathcal{N}$ is not invertible and we have to rely on some heuristics to find $\mathcal{M}$, and the equality $x = y$ is not guaranteed. In other words, x is an estimate of y that we can obtain knowing $s = \mathcal{N}(y)$ and $\mathcal{M}$, and we want to estimate the error on x using a statistical approach. In our specific case, y are the injected value, N is the noise introduced by `GstLAL`, s are the recovered quantities, $\mathcal{M}$ is a ML algorithm and thus x are the predictions of our ML algorithm. The expression $x \simeq y$ can be formalized requiring that x minimize a loss function. To compute the error on the prediction x, we need to know the distribution of y values at fixed x, namely $f_x(y)$. Then, from $f_x(y)$ we can compute the confidence interval of the prediction x and we are done. Note that if $\mathcal{M}$ is a smooth 1-dimensional function, then $f_x(y)$ would be a Dirac-delta. However, even in the 1d case, it could be argued that if $\mathcal{M}$ is not smooth, then is possible that in an arbitrarily small neighborhood of a certain x we can have many different y values. The elegant example is a self-similar function, the more realistic case is a non-continuous function with a lot of discontinuities. In any case, if we are considering a ML algorithm, $\mathcal{M}$ should be smooth. Fortunately, we are not working on a 1d problem, i.e. x and y are actually vectors, so that if we consider one component at a time, for each x_i value we can have many y_i. More precisely,we project $\mathcal{M}^i(s_1, s_2, \dots, s_i, \dots) = (x_1, x_2, \dots, x_i, \dots)$ on the (x^i, y^i) plane, so that for each value of x_i we can have different values of y_i and thus $f_{x_i}(y_i)$ is not ill-posed. Treating each element of the arrays (features) separately means that we assume that the error associate to x^i can be determined knowing only the y^i distribution. Note that this is not equivalent to state that the error on x^i depends only on x^i itself, but it is quite the opposite, as discussed above. In the following we drop the i-index on the features.

II. APPROACH N.1: $\mu(x)$, $\sigma(x)$, $\gamma(x)$ FITS

We can then consider the (x, y) plane and define the sets $I_{\bar{x}} = \{x \in (\bar{x} - \frac{\Delta x}{2}, \bar{x} + \frac{\Delta x}{2})\}$ and $I_y^0 = \{y : \mathcal{M} \circ \mathcal{N}(y) \in I_{\bar{x}}\}$. Assuming N large enough, we can consider a small Δx and still have many elements in the sets $I_{\bar{x}}$ and I_y^0. We are interested in estimating the y-distribution given a certain x, but what we are doing here is to consider many y values that correspond to different x values. Therefore we introduce a linear transformation $\mathcal{P}_y : y \to y' = y + \bar{x} - x$ that defines a new set $I_y = \{\mathcal{P}_y(y), y \in I_y^0\}$. Note that the transformation $P : (x, y) \to (\bar{x}, y')$ is a translation of the (x, y) point along the identity line $y = x$. We thus find a set of y values that correspond to an unique x value, $\bar{x}$. The distribution of y values will give us information on the accuracy of our transformation $\mathcal{M} \circ \mathcal{N}$ for the considered feature, since they are all the values that can be predicted for a certain value x. This last sentence can sound strange since $\mathcal{M}$ is a function and thus should predict a single value for each x. However, as discussed above, consider that we are actually considering the projection of $\mathcal{M} \circ \mathcal{N}$ in the (x, y). We say that $I_{\bar{x}}$ and I_y define a *bin*. To cover all the possible values of x, we build many bins. For each bin we want to determine $f_{\bar{x}}(y)$.

We assume that $f_{\bar{x}}(y)$ can be written analytically as a skewed normal Gaussian distribution. This 1-parameter family of continuously-deformed Gaussian distributions depends on three parameters: the location ξ, the scale ω, and the shape α. If $\alpha = 0$ then the location is the mean value, the scale is the standard deviation and we recover the Gaussian distribution. These three parameters can reconstructed analytically for each bin computing the mean μ, variance σ^2, and skewness γ of the points in I_y. Consider that in order to do the transformation $(\mu, \sigma, \gamma) \to (\xi, \omega, \alpha)$, the skewness should be in the interval $(-1, 1)$, so that if we measure $|\gamma| > 0.99$, we set $\gamma = \pm 0.99$. After the computation of (μ, σ, γ) for each bin, we do a polynomial fit in order to obtain $(\mu(x), \sigma(x), \gamma(x))$. If the fit is good, then for each value of x we can obtain $f_{\bar{x}}(y)$ and we are done.

Summary:

1. Build the bins $(I_{\bar{x}}, I_y)$ for many $\bar{x}$ so that we cover the whole parameter space;

2. for each bin we compute the distribution moments (μ, σ, γ) on I_y;

3. we fit the moments found in order to obtain $(\mu(x), \sigma(x), \gamma(x))$;

4. from a value x^* we can obtain $(\mu(x^*), \sigma(x^*), \gamma(x^*))$ and compute (ξ, ω, α), i.e. $f_{x^*}(y)$.

The problem with this method is that the fits used to obtain the momenta must be good and this strongly depends on the distribution of data in the parameter space.

This method works for m_1 and m_2 on the dataset with only nonspinning BNS, see Fig. 1, but does not work for the complete dataset. One could try to improve the bin-sampling, but I have no practical ideas on how to do that.

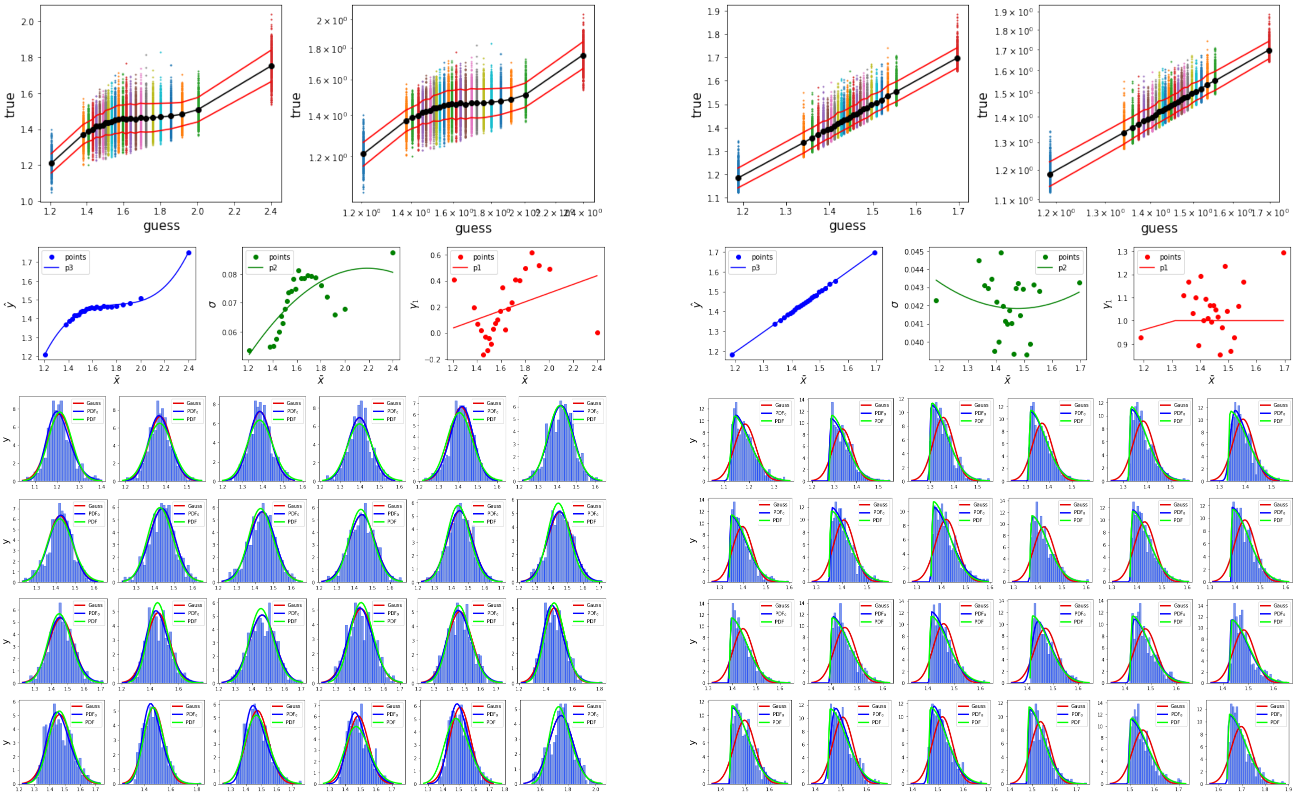

FIG. 1. Method 1 applied to recovered (left) and predicted (right) m_1 for the nonspinning BNS dataset. To build the histograms and thus the PDFs, we consider $n \simeq 565$ points and only the y values that are included in the 5 σ interval. The orders of polynomials used for (μ, σ, γ) are 3, 2, and 1, respectively (with $\gamma_{\max} = 0.99$). The green curve is the one computed from the fits, while the blue one is obtained directly from I_y. The red one is computed from I_y under Gaussian-hypothesis.