International Journal for Uncertainty Quantification, 11(4):81–94 (2021)

ACCRUE: ACCURATE AND RELIABLE UNCERTAINTY ESTIMATE IN DETERMINISTIC MODELS

***Enrico Camporeale*[1,*] *& Algo Carè*[2]**

[1]*University of Colorado, Boulder, Colorado, USA*

[2]*University of Brescia, Brescia, Italy*

*Address all correspondence to: Enrico Camporeale, University of Colorado, Boulder, Colorado, USA, E-mail: enrico.camporeale@colorado.gov

Original Manuscript Submitted: 4/30/2020; Final Draft Received: 12/13/2020

In this paper we focus on the problem of assigning uncertainties to single-point predictions generated by a deterministic model that outputs a continuous variable. This problem applies to any state-of-the-art physics or engineering models that have a computational cost that does not readily allow running ensembles and estimating the uncertainty associated to single-point predictions. Essentially, we devise a method to easily transform a deterministic prediction into a probabilistic one. We show that for doing so, one has to compromise between the accuracy and the reliability (calibration) of such a probabilistic model. Hence, we introduce a cost function that encodes their trade-off, and we call this new method ACCRUE (ACCurate and Reliable Uncertainty Estimate). We use the continuous rank probability score to measure accuracy and we derive an analytic formula for the reliability, in the case of forecasts of continuous scalar variables expressed in terms of Gaussian distributions. The new ACCRUE cost function is then used to estimate the input-dependent variance, given a black-box "oracle" mean function, by solving a two-objective optimization problem. The simple philosophy behind this strategy is that predictions based on the estimated variances should not only be accurate, but also reliable (i.e., statistically consistent with observations). Conversely, early works based on the minimization of classical cost functions, such as the negative log probability density, cannot simultaneously enforce both accuracy and reliability. We show several examples both with synthetic data, where the underlying hidden noise can accurately be recovered, and with large real-world datasets.

KEY WORDS: *calibration, deterministic models, machine learning*

1. INTRODUCTION

There is a growing consensus, across many fields and applications, that forecasts should have a probabilistic nature [1]. This is particularly true in decision-making scenarios where cost-loss analyses are designed to take into account the uncertainties associated to a given forecast [2,3]. Unfortunately, it is often the case that well established predictive models are completeley deterministic and thus provide single-point estimates only. For example, in engineering and applied physics, models often rely on computer simulations. A typical strategy to assign confidence intervals to deterministic predictions is to perform ensemble forecasting, that is, to repeat the same simulation with slightly different setup (i.e., free parameters, and initial or boundary conditions) [4,5]. However, this is rather expensive and it often requires a trade-off between computational cost and accuracy of the model, especially when there is a need for real-time predictions. Likewise, the most successful applications in machine learning techniques have focused on estimating target variables, with less emphasis on the estimation of the uncertainty of the prediction. Only recently, calibrated uncertainty for reliable predictions has become one of the central topics of machine learning research [6–11].

In this paper we focus on the problem of assigning uncertainties to single-point predictions, with a particular emphasis on the requirement of calibration. When dealing with a probabilistic forecast, calibration is as important as accuracy. Calibration, also known as reliability (for instance, in the meteorological literature), is the requirement that the probabilities should give an estimate of the expected frequencies of the event occurring, that is, a statistical consistency between probabilistic predictions and observations [12,13].

We restrict our attention to predictive models that output a scalar continuous variable, and whose uncertainties are in general input-dependent. For the sake of simplicity, and for its widespread use, we assume that the probabilistic forecast that we want to generate is in the form of a Gaussian distribution. Hence, the problem can be cast in terms of the estimation of the input-dependent variance associated to a normal distribution centered around forecasted values provided by an oracle.

In the machine learning community, elegant and practical ways of deriving uncertainties based on flexible probabilistic models are well established, either based on standard and Bayesian neural networks [6,8,14–17], or Gaussian Processes (GPs) [18]. However, it is important to emphasize that while in the classical heteroskedastic regression problem, one is interested in learning simultaneously the mean function $f(\mathbf{x})$ and the variance $\sigma^2(\mathbf{x})$, here we assume that the mean function is provided by a black-box model (for instance, a physics simulation) that cannot easily be improved; hence the whole attention is focused on the variance estimation. This is realistic in several applied fields, where decades of work have resulted in very accurate physics-based models that, however, suffer the drawback of being completely deterministic. Hence, we decouple the problem of learning mean function and variance, focusing solely on the latter. Also, it is important to keep in mind that we aim at estimating the variance using a single mean function (oracle prediction), and not an ensemble.

1.1 Summary of Contributions and Novelty

The task of generating uncertainties associated with black-box predictions, thus transforming a deterministic model into a probabilistic one, and simultaneously ensuring that such uncertainties are both accurate and calibrated is novel. The closest early works in the machine learning literature that are worth mentioning are concerned with post-processing calibration. In that case, a model outputs probabilistic predictions that are not well-calibrated and the task is to recalibrate these outputs by deriving a function $[0,1] \rightarrow [0,1]$ that maps the original probabilities to new, well-calibrated probabilities. Recalibration has been studied extensively in the context of classification, with methods such as Platt scaling [19], isotonic regression [20], and temperature scaling [7]. Applications to regression are less studied. A recent work is [21], where isotonic regression is used to map the predicted cumulative distribution function of a continuous target variable to the observed one, effectively recalibrating the prediction. This approach was later criticized for not being able to distinguish between informative and noninformative uncertainty predictions [22] and for not being able to ensure calibration for a specific prediction (but only in an average sense) [23]. Finally, a relevant approach has recently been proposed in [24], building on the original idea of [25] of designing a neural network that outputs simultaneously mean and variance of a Gaussian distribution, by minimizing a proper score, namely the negative log likelihood of the predictive distribution. Reference [24] points out the importance of calibration of probabilistic models, even though in that work calibration is not explicitly enforced.

Overall, it appears that none of the previous works has recognized that calibration is only one aspect of a two-objective optimization problem. In fact, we will demonstrate that post-processing calibration (reliability) is competing with accuracy (sharpness) and therefore one must seek for the optimal trade-off between these two equally important qualities of a probabilistic forecast.

Our method is very general and does not depend on any particular choice for the black-box model that predicts the output targets (which indeed is not even required; all that is needed are the errors between predictions and real targets). The philosophy is to introduce a cost function which encodes a trade-off between the accuracy and the reliability of a probabilistic forecast. Assessing the goodness of a forecast through proper scores, such as the negative log probability density, or the continuous rank probability score, is a common practice in many applications, such as weather predictions [26,27]. Also, the notion that a probabilistic forecast should be well-calibrated, or statistically consistent with observations, has been discussed at length in the atmospheric science literature [28,29]. However, the basic idea that these two metrics (accuracy and reliability) can be combined to estimate the empirical variance from a sample of observations, and possibly to reconstruct the underlying noise as a function of the inputs, has never been proposed. Moreover, as we will discuss, the two metrics are competing, when interpreted as functions of the variance only. Hence, this gives rise to a two-objective optimization problem, where one is interested in achieving a good trade-off between these two properties.

Our main contributions are the introduction of the reliability score (RS), that measures the discrepancy between empirical and ideal calibration, and the Accurate and Reliable Uncertainty Estimate (ACCRUE) cost function. We

show that for a Gaussian distribution the RS has a simple analytical formula. The accuracy part of the ACCRUE cost function is measured by means of the continuous rank probability score, that we argue has better properties than the more standard negative log probability density.

The paper is organized as follows. We first introduce the negative logarithm of the probability density and the continuous rank probability score as scores for accuracy. We then comment on the reliability and how to construct a reliability diagram for continuous probabilistic forecast, and we show that accuracy does not imply reliability and indeed the two metrics are competing. We then introduce a new score to measure reliability for Gaussian distributions and the ACCRUE score. Finally, we show how the new score can be used to estimate uncertainty both in toy and real-world examples.

2. SCORING RULES FOR PROBABILISTIC FORECAST

The problem of how to assess a probabilistic forecast has been studied at length in the literature (see, e.g., [1,30]). Here, we focus on the forecast of a continuous real variable and its associated uncertainty. In this section, we briefly revise two popular choices of so-called scoring rules for probabilistic predictions (albeit possibly popularized in different scientific communities): the negative logarithm of the probability density (NLPD), and the continuous rank probability score (CRPS). Although in the following we opt to use CRPS for our experiments, equal considerations apply for NLPD.

In the case of Gaussian distributions a forecast is simply given by the mean value μ and the variance σ^2. NLPD is defined as

$$\mathrm{NLPD}(\varepsilon, \sigma) = \frac{\log\sigma^2}{2} + \frac{\varepsilon^2}{2\sigma^2} + \frac{\log 2\pi}{2}, \tag{1}$$

where we define $\varepsilon = y^o - \mu$ as the error between a given observation y^o and the corresponding prediction μ. CRPS is a generalization of the well-known Brier score [31], used to assess the probabilistic forecast of continuous scalar variables, when the forecast is given in terms of a probability density function, or its cumulative distribution. CRPS is defined as

$$\mathrm{CRPS} = \int_{-\infty}^{\infty} [C(y) - H(y - y^o)]^2 dy, \tag{2}$$

where $C(y)$ is the cumulative distribution (cdf) of the forecast, $H(y)$ is the Heaviside function, and y^o is the true (observed) value of the forecasted variable. The CRPS for Gaussian forecasts can be calculated analytically [4] as

$$\mathrm{CRPS}(\varepsilon, \sigma) = \sigma\left[\frac{\varepsilon}{\sigma}\mathrm{erf}\left(\frac{\varepsilon}{\sqrt{2}\sigma}\right) + \sqrt{\frac{2}{\pi}}\exp\left(-\frac{\varepsilon^2}{2\sigma^2}\right) - \frac{1}{\sqrt{\pi}}\right]. \tag{3}$$

Several interesting properties of the CRPS have been studied in the literature. Its decomposition into reliability and resolution/uncertainty has been shown in [32]. There are a few reasons for preferring CRPS to NLPD. They are both negatively oriented, but CRPS is equal to zero for a perfect forecast with no uncertainty (deterministic). Indeed, the CRPS has the same unit as the variable of interest, and it collapses to the absolute error $|y^o - \mu|$ for $\sigma \to 0$, that is, when the forecast becomes deterministic. On the other hand, the limit $\sigma \to 0$ is problematic for NLPD. Figure 1 shows a graphical comparison between NLPD (left panel) and CRPS (right panel). Different curves show the isolines for the two scores, as a function of the error ε (vertical axis) and the standard deviation σ (horizontal axis). The black dashed line indicates the minimum value of the score, for a fixed value of ε. Because we are approaching the problem of variance estimation by assigning an empirical variance to single-point black-box predictions, it makes sense to minimize a score as a function of σ only, for a fixed value of the error ε. By differentiating Eq. (3) with respect to σ, one obtains

$$\frac{d\,\mathrm{CRPS}}{d\sigma} = \sqrt{\frac{2}{\pi}}\exp\left(-\frac{\varepsilon^2}{2\sigma^2}\right) - \frac{1}{\sqrt{\pi}}, \tag{4}$$

and the minimizer is found to be $\sigma^2_{\mathrm{min,CRPS}} = \varepsilon^2/\log 2$. Note that the minimizer for NLPD is $\sigma^2_{\mathrm{min,NLPD}} = \varepsilon^2$.

As it is evident from Fig. 1, CRPS penalizes under- and overconfident predictions in a much more symmetric way than NLPD. Both scores are defined for a single instance of forecast and observation; hence they are usually averaged over an ensemble of predictions, to obtain the score relative to a given model, for instance: $\overline{\text{CRPS}} = \sum_k \text{CRPS}(\varepsilon_k, \sigma_k)$.

3. RELIABILITY

Reliability is the property of a probabilistic model that measures its statistical consistency with observations. For forecasts of discrete events, it measures if an event predicted with probability p occurs, on average, with frequency p. This concept can be extended to forecasts of a continuous scalar quantity by examining the so-called reliability diagram [33–35]. Note that in this paper we use the terms "calibration" and "reliability" interchangeably. A reliability diagram for Gaussian forecasts is produced in the following way. Let us define the standardized errors associated to the ith observation/prediction in a set of size N as $\eta_i = \varepsilon_i/(\sqrt{2}\sigma_i)$. The value of the probability associated to a given Gaussian forecast is $\Phi_i = (1/2)(\text{erf}(\eta_i) + 1)$. The reliability diagram is then provided by the empirical cumulative distribution of the values Φ_i, defined as $C(\varphi) = (1/N)\sum_{i=1}^{N} H(\varphi - \Phi_i)$ (H is the Heaviside function). Its interpretation is of observed frequency as a function of the predicted probability (note that this method of producing a reliability diagram does not require binning). A perfect calibration shows in the reliability diagram as a straight diagonal line.

The motivating argument of this work is that two models with identical score (and we use here NLPD to illustrate the argument, but the same would be true for CRPS) can have remarkably different reliability diagrams. We show an example in Fig. 2. One thousand data points have been generated as $\mathcal{N}(0, \sigma(x)^2)$, with $x \in [0, 1]$ and $\sigma(x) = x + 1/2$, as in the synthetic dataset proposed in [36]. A model completely consistent with the data generation mechanism (i.e., with zero mean and variance σ^2) produces the blue line in the reliability diagram in the left panel, that is almost perfect calibration. However, one can generate a second model with a modified (wrong) variance $\tilde{\sigma}^2$ such that $\text{NLPD}(\varepsilon, \tilde{\sigma}) = \text{NLPD}(\varepsilon, \sigma)$, that is,

$$\frac{\log\tilde{\sigma}^2}{2} + \frac{\varepsilon^2}{2\tilde{\sigma}^2} = \frac{\log\sigma^2}{2} + \frac{\varepsilon^2}{2\sigma^2}. \tag{5}$$

Equation (5) always produces a solution $\tilde{\sigma} \neq \sigma$, as long as $\sigma^2 \neq \varepsilon^2$ (we recall that $\sigma^2 = \varepsilon^2$ is the global minimum of NLPD, for fixed ε). Graphically this can be seen in Fig. 1: for a constant ε value, there are two values of σ on the same NLPD contour. The red line in the left panel of Fig. 2 has been derived from such a modified model

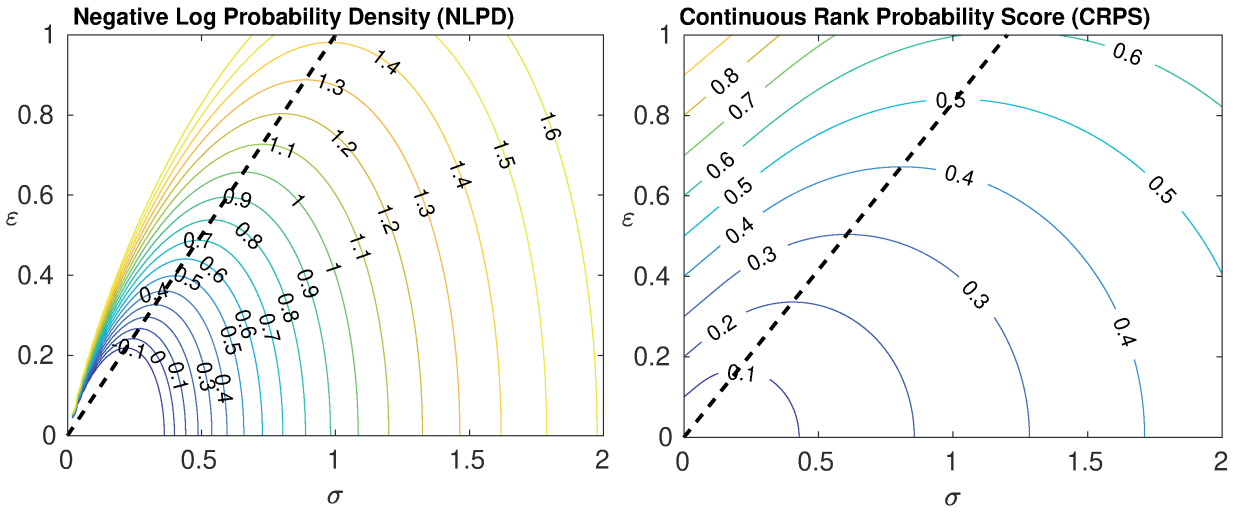

FIG. 1: Isoline of constant NLPD (left) and CRPS (right) as a function of standard deviation σ, and error ε. The black dashed line indicates the minimum, as function of ε.

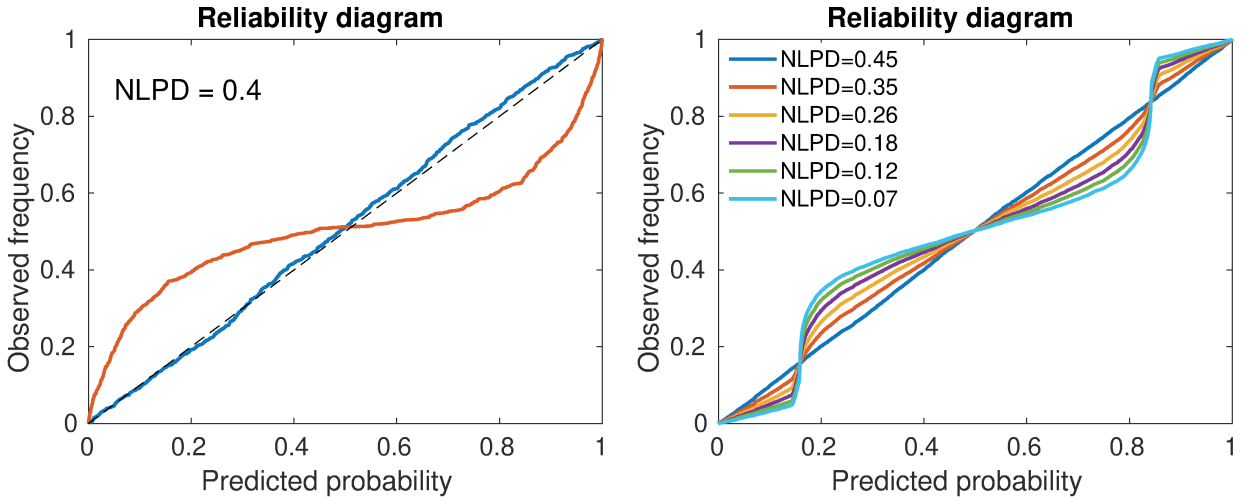

FIG. 2: Left: example of two models with identical value of NLPD $= 0.4$, and different reliability diagram. Right: Example of several models for which the NLPD decreases (from NLPD $= 0.45$ for the blue line to NLPD $= 0.07$ for the cyan line), at the expense of reliability. See text for details of how the synthetic data has been generated. Figures published online in color.

$\mathcal{N}(0, \tilde{\sigma}^2)$, which is obviously miscalibrated. For this example NLPD $= 0.4$ (equal for both cases). As a complementary argument, we show in the right panel of Fig. 2 the reliability diagram of several models, with decreasing values of NLPD. One can appreciate that progressively decreasing NLPD results in a worse and worse calibration (note that NLPD is negatively oriented, so smaller values mean better accuracy). These models have been generated again starting from the perfectly calibrated synthetic model, progressively shifting the values assigned to σ_i^2, towards the global minimum $\sigma_i^2 = \varepsilon_i^2$ (hence decreasing NLPD). Thus, minimizing a traditional cost function such as NLPD, for fixed errors, does not necessarily imply to achieve a well-calibrated model. Of course, we are not suggesting that any model generated by means of minimizing NLPD is inevitably miscalibrated. However, unless explicitly enforced, calibration will be a by-product of other properties. Once again, the same is true for CRPS.

3.1 Reliability Score for Gaussian Forecast

Reliability is a statistical property of a model, defined for a large enough ensemble of forecasts-observations. Here, we introduce the reliability score for normally distributed forecasts. In this case, we expect the standardized errors η calculated over a sample of N predictions-observations to have a standard normal distribution with cdf $\Phi(\eta) = (1/2)(\text{erf}(\eta) + 1)$. Hence we define the reliability score (RS) as:

$$\text{RS} = \int_{-\infty}^{\infty} [\Phi(\eta) - C(\eta)]^2 d\eta \tag{6}$$

where $C(\eta)$ is the empirical cumulative distribution of the standardized errors η, that is,

$$C(\eta) = \frac{1}{N} \sum_{i=1}^{N} H(\eta - \eta_i) \tag{7}$$

with $\eta_i = (y_i^o - \mu_i)/(\sqrt{2}\sigma_i)$. Note that each error $(y_i^o - \mu_i)$ is standardized with respect to a different (input-dependent) σ_i. RS measures the divergence of the empirical distribution of standardized errors η from a standard normal distribution. From now on we will use the convention that the set $\{\eta_1, \eta_2, \ldots, \eta_N\}$ is sorted ($\eta_i \leq \eta_{i+1}$). Obviously this does not imply that μ_i or σ_i are sorted as well. Interestingly, the integral in Eq. (6) can be calculated analytically, via expansion into a telescopic series, yielding

$$\mathrm{RS} = \sum_{i=1}^{N} \left[\frac{\eta_i}{N} (\mathrm{erf}(\eta_i) + 1) - \frac{\eta_i}{N^2}(2i-1) + \frac{\exp(-\eta_i^2)}{\sqrt{\pi} N} \right] - \frac{1}{2}\sqrt{\frac{2}{\pi}}. \tag{8}$$

Differentiating the ith term of the above summation, RS_i, with respect to σ_i (for fixed ε_i), one obtains

$$\frac{d\mathrm{RS}_i}{d\sigma_i} = \frac{\eta_i}{N\sigma_i} \left(\frac{2i-1}{N} - \mathrm{erf}(\eta_i) - 1 \right). \tag{9}$$

Hence, RS_i is minimized when the values $\sigma_{\mathrm{min}}^{\mathrm{RS}}$ satisfy

$$\mathrm{erf}(\eta_i) = \mathrm{erf}\left(\frac{\varepsilon_i}{\sqrt{2}\sigma_{\mathrm{min}}^{\mathrm{RS}}} \right) = \frac{2i-1}{N} - 1. \tag{10}$$

This could have been trivially derived by realizing that the distribution of η_i that minimizes RS is the one such that the values $\Phi(\eta_i)$ are uniform in the interval $[0, 1]$.

4. THE ACCRUE COST FUNCTION

The ACCRUE cost function introduced here follows from the simple principle that the variances σ_i^2 estimated from an ensemble of errors ε_i should result in a model that is both accurate (with respect to the CRPS score), and reliable (with respect to the RS score). Clearly, this gives rise to a two-objective optimization problem. It is trivial to verify that CRPS and RS cannot simultaneously attain their minimum value (as was evident from Fig. 2). Indeed, by minimizing the former, $\eta_i = [\mathrm{sign}(\varepsilon_i)/2]\sqrt{\log 4}$ for any i, which cannot result in a minimum for RS, according to Eq. (10). This demonstrates that methods that focus solely on recalibration (any method of choice can be reformulated in terms of minimizing RS) can possibly result in the deterioration of accuracy. In passing, we note that any cost function that is minimized (for constant ε) by a value of the variance σ^2 that is linear in ε^2 suffers this problem (because η_i will be a constant). Finally, notice that trying to minimize RS as a function of σ_i (for fixed errors ε_i) results in an ill-posed problem, because RS is solely expressed in terms of the standardized errors η. Hence, there is no unique solution for the variances that minimize RS. Hence, RS can be more appropriately thought of as a regularization term in the ACCRUE cost function. The simplest strategy to deal with multiobjective optimization problems is to scalarize the cost function, which we define here as

$$\mathrm{ACCRUE} = \beta \cdot \overline{\mathrm{CRPS}} + (1-\beta)\mathrm{RS}. \tag{11}$$

We choose the scaling factor β as

$$\beta = \mathrm{RS}_{\mathrm{min}} / (\overline{\mathrm{CRPS}}_{\mathrm{min}} + \mathrm{RS}_{\mathrm{min}}). \tag{12}$$

The minimum of $\overline{\mathrm{CRPS}}$ is $\overline{\mathrm{CRPS}}_{\mathrm{min}} = (\sqrt{\log 4}/2N) \sum_{i=1}^{N} \varepsilon_i$, which is simply the mean of the errors, rescaled by a constant. The minimum of RS follows from Eqs. (8) and (10):

$$\mathrm{RS}_{\mathrm{min}} = \frac{1}{\sqrt{\pi} N} \sum_{i=1}^{N} \exp\left(-\left[\mathrm{erf}^{-1}\left(\frac{2i-1}{N} - 1 \right) \right]^2 \right) - \frac{1}{2}\sqrt{\frac{2}{\pi}}. \tag{13}$$

Notice that $\mathrm{RS}_{\mathrm{min}}$ is only a function of the size of the sample N, and it converges to zero for $N \to \infty$. The heuristic choice in Eq. (12) is justified by the fact that the two scores might have different orders of magnitude, and therefore we rescale them in such a way that they are comparable in our cost function (11). We believe this to be a sensible choice, although there might be applications where one would like to weigh the two scores differently. In future work, we will explore the possibility of optimizing β in a principled way, for instance, constraining the difference between empirical and ideal reliability score to be within limits given by the dataset size N, or by making β a learnable parameter. Finally, in our practical implementation, we neglect the last constant term in the definition (13) so that, for sufficiently large N, $\mathrm{RS}_{\mathrm{min}} \simeq (1/2)\sqrt{2/\pi} \simeq 0.4$.

5. RESULTS

In summary, we want to estimate the input-dependent values of the empirical variances σ_i^2 associated to a sample of N observations for which we know the errors ε_i. We do so by solving an optimization problem in which the set of estimated σ_i minimizes the ACCRUE cost function defined in Eq. (11). This newly introduced cost function has a straightforward interpretation as the trade-off between accuracy and reliability, which are two essential but conflicting properties of probabilistic models. In practice, because we want to generate a model that is able to return σ^2 as a function of the inputs $\mathbf{x}$ on any point of a domain, we introduce a structure that enforces a certain degree of smoothness of the unknown variance, in the form of a regression model. In the following we show some experiments on toy problems and on multidimensional real datasets to demonstrate the easiness, robustness, and accuracy of the method.

5.1 Toy Problems

In order to facilitate comparison with previous works, we choose some of the datasets used in [37], although for simplicity of implementation we rescale the standard deviation so to be always smaller or equal to 1. Since in our method we assume that a mean function is provided, for the toy problems we use the result of a standard (homoskedastic) Gaussian process regression as $f(x)$.

For all datasets the targets y_i are sampled from a Gaussian distribution $\mathcal{N}(f(x), \sigma(x)^2)$. The first three datasets are one-dimensional in x, while in the fourth we will test the method on a five-dimensional space, thus showing the robustness of the proposed strategy.

G dataset: $x \in [0,1]$, $f(x) = 2\sin(2\pi x)$, $\sigma(x) = (1/2)x + 1/2$ [36].

Y dataset: $x \in [0,1]$, $f(x) = 2(\exp(-30(x-0.25)^2) + \sin(\pi x^2)) - 2$, $\sigma(x) = \exp(\sin(2\pi x))/3$ [38].

W dataset: $x \in [0,\pi]$, $f(x) = \sin(2.5x)\sin(1.5x)$, $\sigma(x) = 0.01 + 0.25(1-\sin(2.5x))^2$ [25,39].

5D dataset: $\mathbf{x} \in [0,1]^5$, $f(\mathbf{x}) = 0$, $\sigma(\mathbf{x}) = 0.45(\cos(\pi + \sum_{i=1}^{5} 5x_i) + 1.2)$ [40].

Examples of 100 points sampled from the **G**, **Y**, and **W** dataset are shown in Fig. 3 (circles), along with the true mean function $f(x)$ (red), and the one predicted by a standard Gaussian process regression model (blue), used as an oracle. The bottom-right plot in Fig. 3 shows the distribution of σ, which ranges in the interval $[0.09, 0.99]$.

For the **G**, **Y**, and **W** we use 100 points uniformly sampled in the domain. The **5D** dataset is obviously more challenging; hence we use 10,000 points (note that this results in fewer points per dimension, compared to the one-dimensional tests). For all experiments we test 100 independent runs.

We have tested a neural network and a polynomial best fit as a regression model. For simplicity, we choose a single neural network architecture that we use for all the tests. We use a network with two hidden layers, respectively, with 50 and 10 neurons. The activation functions are rectified linear (ReLU) and a symmetric saturating linear function, respectively. The output is given in terms of $\log\sigma$, to enforce positivity of σ^2. For all experiments, the datasets are randomly divided into training (33%), validation (33%), and test (34%) sets. All the reported metrics are calculated on the test set only. The network is trained using a standard BFGQ quasi-Newton algorithm, and the iterations are forcefully stopped when the loss function does not decrease for ten successive iterations on the validation set. The only inputs needed are the inputs $\mathbf{x}_i$ and the corresponding errors ε_i. Finally, in order to avoid local minima due to the random initialization of the neural network weights, we train five independent networks and choose the one that yields the smallest cost function.

In the case of low-dimensional data one might want to try simpler and faster approaches than a neural network, especially if smoothness of the underlying function $\sigma(x)$ can be assumed. For the one-dimensional test cases (**G**, **Y**, **W**) we have devised a simple polynomial best fit strategy. We assume that $\sigma(x)$ can be approximated by a polynomial of unknown order, equal to or smaller than 10: $\sigma(x) = \sum_{l=0}^{10} \theta_l x^l$, where in principle one or more θ_l can be equal to

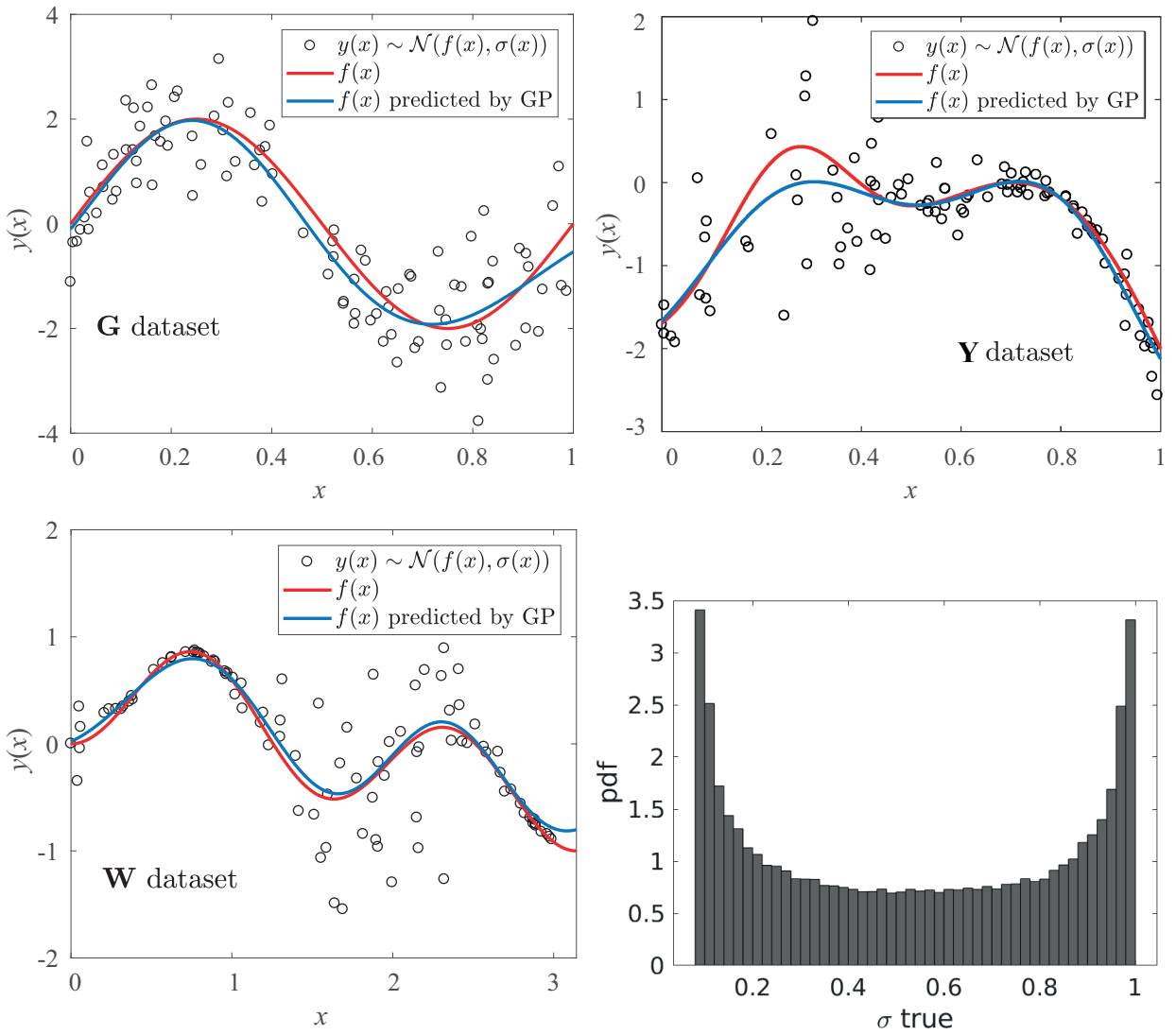

FIG. 3: Top: 100 points sampled from the **G** (left) and **Y** (right) dataset (circles). Bottom-left: 100 points sampled from the **W** dataset (circles). The red line shows the true mean function $f(x)$, while the blue line is the one predicted by the GP model. Bottom-right: Distribution of true values of standard deviation σ for the **5D** dataset. Figures published online in color.

zero. The vector $\Theta = \{\theta_0, \theta_1, \ldots, \theta_{10}\}$ is initialized with $\theta_0 =$ const and all the others are equal to zero. The constant can be chosen, for instance, as the standard deviation of the errors ε. The polynomial best fit is found by means of an iterative procedure (Algorithm 1). In other words, the algorithm finds the values of Θ for a given polynomial order that minimize the ACCRUE cost function. Then it tests the next higher order, by using the previous solution as the initial guess. Whenever the difference between the solutions obtained with two successive orders is below a certain tolerance, the algorithm stops. The multidimensional optimization problem is solved by a BFGQ quasi-Newton method with a cubic line search procedure. Note that whenever a given solution is found to yield a local minimum for the next polynomial order, the iterations are terminated.

The results for the 1D datasets **G**, **Y**, and **W** are shown in Figs. 4–6, in a way consistent with [37]. The red lines denote the true standard deviation $\sigma(x)$ used to generate the data. The black line indicates the values of the estimated σ averaged over 100 independent runs, and the gray areas represent one and two standard deviations from

Algorithm 1: Polynomial best fit

Input: data x_i, ε_i
Initialize $p = 0$, $\theta_0 =$ const, $P_{\max} = 10$, tol
while $p \leq P_{\max}$ & err $>$ tol **do**
  $p = p + 1$
  Initial guess for optimization $\Theta = \{\theta_0, \ldots, \theta_{p-1}, 0\}$
  $\Theta =$ argmin ACCRUE(σ_i) (with $\sigma_i = \sum_{l=0}^{p} \theta_l x_i^l$)
  err = $||\text{ACCRUE}(\sigma(\Theta_{\text{old}})) - \text{ACCRUE}(\sigma(\Theta_{\text{new}}))||_2$
end

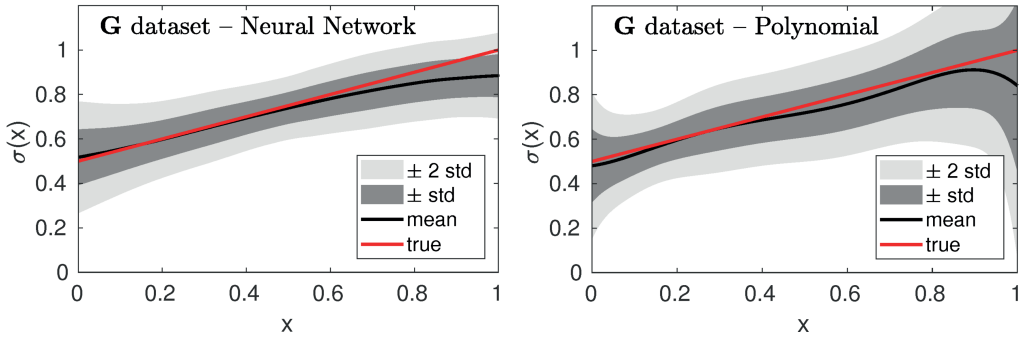

FIG. 4: G dataset: True value of the standard deviation σ (red line) and mean value obtained averaging over 100 independent runs (black line). The gray shaded areas denote the confidence interval of one and two standard deviations calculated from the same ensemble of runs. In the left panel σ is calculated through a neural network, while in the right panel as a polynomial function (see text). Figures published online in color.

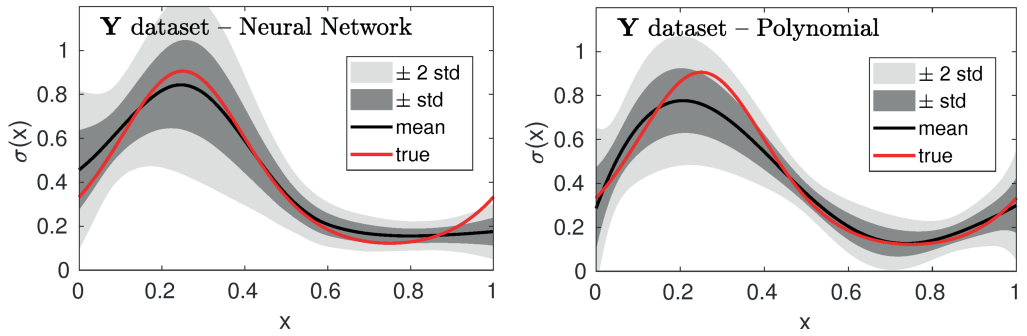

FIG. 5: Y dataset: True value of the standard deviation σ (red line) and mean value obtained averaging over 100 independent runs (black line). The gray shaded areas denote the confidence interval of one and two standard deviations calculated from the same ensemble of runs. In the left panel σ is calculated through a Neural Network, while in the right panel as a polynomial function (see text). Figures published online in color.

the mean. A certain spread in the results is due to different training sets (in each run the data points are sampled independently) and, for the neural network, to random initialization. The left panels show the results obtained with the neural network, while the right panels show the result obtained with the polynomial fit. In all cases, except for the **W** dataset (polynomial case, right panel), the results are very accurate.

For the **5D** dataset it is impractical to compare graphically the real and estimated $\sigma(\mathbf{x})$ in the five-dimensional domain. Instead, in Fig. 7 we show the probability density of the real versus predicted values of the standard deviation. Values are normalized such that the maximum value in the color map for any value of predicted σ is equal to 1 (i.e.,

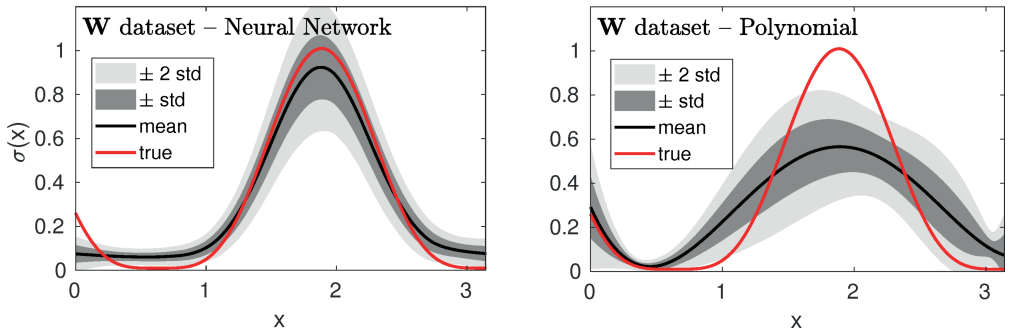

FIG. 6: W dataset: True value of the standard deviation σ (red line) and mean value obtained averaging over 100 independent runs (black line). The gray shaded areas denote the confidence interval of one and two standard deviations calculated from the same ensemble of runs. In the left panel σ is calculated through a Neural Network, while in the right panel as a polynomial function (see text). Figures published online in color.

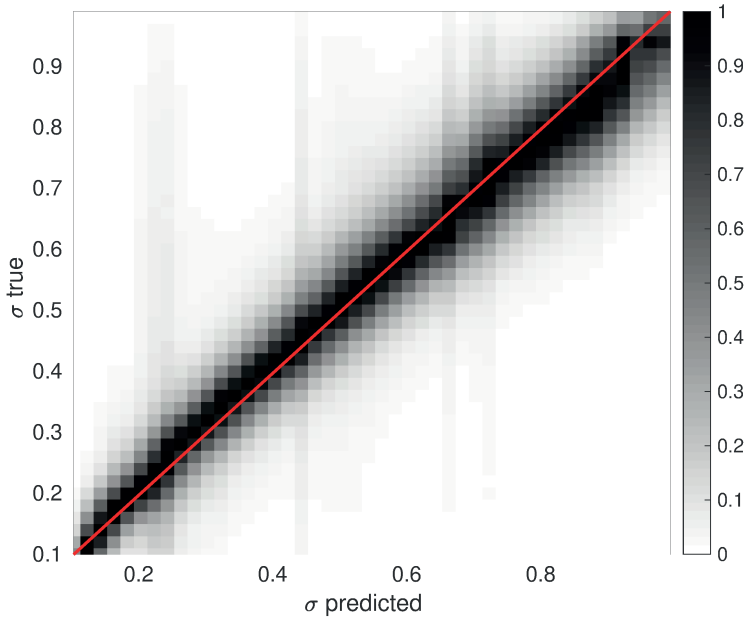

FIG. 7: Probability density of the prediction versus real values of σ for the **5D** dataset. The red line denotes perfect prediction. The densities are normalized to have maximum value along each column equal to one. 10,000,000 samples have been used to generate the plot (with a training set of 10,000 points). Figures published online in color.

along vertical lines). The red line shows a perfect prediction. The color map has been generated by 10,000,000 points, while the model has been trained with 3300 points only. For this case, we have used an exact mean function (equal to zero), in order to focus exclusively on the estimation of the variance. We believe that this is an excellent result for a very challenging task, given the sparsity of the training set, that shows the robustness of the method.

5.2 Real-World Dataset

We have tested our method on the same datasets used in [16]. The only difference with the toy problems is that we use 70% of the data for training, and we only use a neural network as regressor. The results reported in Table 1 are computed over 50 independent runs. For each run, we first train a standard neural network to provide the mean function $f(\mathbf{x})$, by minimizing the mean square errors with respect to the targets. We then compare our method against

TABLE 1: Comparison between different methods on several multidimensional datasets. Median values are reported, calculated over 50 runs. Confidence intervals represent one standard deviation. Best values are in bold

Method			**CRPS**	**RECAL**	**KM**	**ACCRUE**
Score			**CRPS**			
Dataset	**Size**	**Dim.**				
Boston Housing	506	13	0.25 ± 0.05	0.25 ± 0.04	0.25 ± 0.03	$\mathbf{0.23 \pm 0.04}$
Concrete	1030	8	0.22 ± 0.03	0.23 ± 0.13	0.26 ± 0.02	$\mathbf{0.21 \pm 0.03}$
Energy	768	8	0.059 ± 0.03	0.056 ± 0.03	0.087 ± 0.01	$\mathbf{0.052 \pm 0.01}$
Kin8nm	8192	8	0.17 ± 0.005	0.16 ± 0.01	0.24 ± 0.005	$\mathbf{0.16 \pm 0.005}$
Power plant	9568	4	0.13 ± 0.003	0.13 ± 0.05	0.15 ± 0.002	$\mathbf{0.12 \pm 0.01}$
Protein	45,730	9	0.38 ± 0.02	0.47 ± 0.13	0.40 ± 0.007	$\mathbf{0.37 \pm 0.02}$
Wine	1599	11	0.48 ± 0.03	0.50 ± 0.29	$\mathbf{0.46 \pm 0.02}$	0.48 ± 0.06
Yacht	308	6	0.06 ± 0.08	0.06 ± 0.02	0.19 ± 0.02	$\mathbf{0.06 \pm 0.02}$
Score			**Cal. err. (%)**			
Dataset	**Size**	**Dim.**				
Boston Housing	506	13	26.2 ± 7.9	20.6 ± 5.5	17.5 ± 3.7	$\mathbf{16.7 \pm 5.9}$
Concrete	1030	8	22.6 ± 5.8	14.4 ± 3.8	22.1 ± 3.0	$\mathbf{11.5 \pm 3.9}$
Energy	768	8	29.3 ± 8.9	29.2 ± 8.0	28.3 ± 2.8	$\mathbf{13.0 \pm 6.5}$
Kin8nm	8192	8	15.9 ± 1.28	8.3 ± 1.30	25.5 ± 0.5	$\mathbf{5.8 \pm 1.28}$
Power plant	9568	4	12.5 ± 1.4	3.4 ± 0.9	16.1 ± 0.8	$\mathbf{2.6 \pm 0.8}$
Protein	45,730	9	13.1 ± 0.8	$\mathbf{5.0 \pm 0.9}$	10.6 ± 0.9	5.4 ± 0.88
Wine	1599	11	16.0 ± 3.7	$\mathbf{7.9 \pm 2.0}$	8.0 ± 2.4	8.3 ± 2.4
Yacht	308	6	26.0 ± 9.4	24.3 ± 13.5	36.6 ± 3.0	$\mathbf{19.5 \pm 8.5}$

three different models (first row in Table 1): CRPS means that the variance is estimated by minimizing CRPS only, KM denotes a K-means method [41], RECAL indicates the isotonic regression method of [21], and ACCRUE denotes our method. The scores reported (second row) are the median values (calculated on the test set only) of CRPS and of the calibration error. To estimate the latter we derive the reliability diagram (in the way described in section 2), and we compute the maximum distance to the optimal reliability (straight diagonal line). This is denoted, in Table 1, as Cal. err. (in percentage). All quantities are reported along with their confidence interval, calculated as one standard deviation. For the K-means method (which is possibly the simplest baseline method) we have clustered the training data in k groups, calculated the standard deviation σ for each cluster, and assigned the same value of σ for all test points belonging to a given cluster. We have run experiments with k ranging from 1 to 10, and we report the minimum values obtained for CRPS, and Cal. err. for the model that yields the best calibration (hence, purposely "leaking" the test dataset to choose the optimal k). The RECAL method takes the σ estimated by the ACCRUE method and applies the recalibration algorithm of [21]. Hence, in a sense both RECAL and KM are run in a setting that gives them, in principle, an unfair advantage. This is done to emphasize the goodness of our method. Finally, the training sets used for all methods are the same.

The results obtained by using the ACCRUE cost function are always better calibrated than minimizing CRPS only and the KM method. In two cases only (Protein and Wine datasets) RECAL yields a slightly better calibration error. However, in both cases, the accuracy (CRPS) of the RECAL method is penalized and we believe that the best trade-off is still achieved by ACCRUE. In fact, ACCRUE offers the best trade-off between accuracy and calibration across all datasets, as expected.

6. DISCUSSION AND FUTURE WORK

We have presented a simple parametric model for estimating the input-dependent variance of probabilistic forecasts. We assume that the data are distributed as $\mathcal{N}(f(\mathbf{x}), \sigma(\mathbf{x})^2)$, and that an approximation of the mean function $f(\mathbf{x})$

is available (the details of the model that approximates the mean function are not important). In order to generate the variance $\sigma^2(\mathbf{x})$, we propose to minimize the ACCRUE cost function, which depends only on σ, on the errors ε, and on the size of the training set N. We have shown that the classical method of minimizing the negative log probability density (NLPD) does not guarantee that the result will be well-calibrated. On the other hand, methods that exclusively focus on the post-process calibration tend to spoil their accuracy. Indeed, we have discussed how accuracy and reliability are two conflicting metrics for a probabilistic forecast and how the latter can serve as a regularization term for the former. We have shown that by using the new ACCRUE cost function, one is able to accurately discover the hidden noise function. Several tests for synthetic and real-world (large) datasets have been shown.

An important point to notice is that the method will inherently attempt to correct any inaccuracy in $f(\mathbf{x})$ by assigning larger variances. For instance, the agreement between predicted and true values of the standard deviation σ presented in Figs. 4–6 must be understood within the limits of the approximation of the mean function (provided by a Gaussian process regression in those toy examples).

By decoupling the prediction of the mean function from the estimation of the variance, this method is not very expensive and is suitable for large datasets. Moreover, for the same reason this method is very appealing in all applications where the mean function is necessarily computed via an expensive black-box, such as computer simulations, for which the *de facto* standard of uncertainty quantification is based on running a large (time-consuming and expensive) ensemble, and for which large datasets of archived runs are often available. Finally, the formulation is well suited for high-dimensional problems, since the cost function is calculated pointwise for any instance of prediction and observation.

Although very simple and highly efficient the method is still fully parametric, and hence it bears the usual drawback of possibly dealing with a large number of choices for the model selection. Interesting future directions will be to incorporate the ACCRUE cost function in a nonparametric Bayesian method for heteroskedastic regression and to generalize the constraint of Gaussian residuals.

ACKNOWLEDGMENT

Enrico Camporeale is partially funded by NASA under Grant No. 80NSSC20K1580 (Ensemble Learning for Accurate and Reliable Uncertainty Quantification).

REFERENCES

1. Gneiting, T. and Katzfuss, M., Probabilistic Forecasting, *Ann. Rev. Stat. Its Appl.*, **1**:125–151, 2014.
2. Murphy, A.H., The Value of Climatological, Categorical and Probabilistic Forecasts in the Cost-Loss Ratio Situation, *Mon. Weather Rev.*, **105**(7):803–816, 1977.
3. Owens, M.J., Horbury, T., Wicks, R., McGregor, S., Savani, N., and Xiong, M., Ensemble Downscaling in Coupled Solar Wind-Magnetosphere Modeling for Space Weather Forecasting, *Space Weather*, **12**(6):395–405, 2014.
4. Gneiting, T., Raftery, A.E., Westveld III, A.H., and Goldman, T., Calibrated Probabilistic Forecasting Using Ensemble Model Output Statistics and Minimum CRPS Estimation, *Mon. Weather Rev.*, **133**(5):1098–1118, 2005.
5. Leutbecher, M. and Palmer, T.N., Ensemble Forecasting, *J. Comput. Phys.*, **227**(7):3515–3539, 2008.
6. Gal, Y. and Ghahramani, Z., Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning, in *Proc. of the 33rd Int. Conf. Machine Learning*, pp. 1050–1059, 2016.
7. Guo, C., Pleiss, G., Sun, Y., and Weinberger, K.Q., On Calibration of Modern Neural Networks, in *Proc. the 34th Int. Conf. on Machine Learning*, Vol. 70, pp. 1321–1330, Sydney, Australia, 2017.
8. Kendall, A. and Gal, Y., What Uncertainties Do We Need in Bayesian Deep Learning for Computer Vision, in *Proc. of Advances in Neural Information Processing Systems*, pp. 5574–5584, 2017.
9. Lakshminarayanan, B., Pritzel, A., and Blundell, C., Simple and Scalable Predictive Uncertainty Estimation Using Deep Ensembles, in *Proc. of Advances in Neural Information Processing Systems*, pp. 6402–6413, 2017.
10. Ovadia, Y., Fertig, E., Ren, J., Nado, Z., Sculley, D., Nowozin, S., Dillon, J., Lakshminarayanan, B., and Snoek, J., Can You Trust Your Model's Uncertainty? Evaluating Predictive Uncertainty under Dataset Shift, in *Proc. of Advances in Neural Information Processing Systems*, pp. 13991–14002, 2019.

11. Osband, I., Risk versus Uncertainty in Deep Learning: Bayes, Bootstrap and the Dangers of Dropout, in *NIPS Workshop on Bayesian Deep Learning*, Vol. 192, 2016.
12. Gneiting, T., Balabdaoui, F., and Raftery, A.E., Probabilistic Forecasts, Calibration and Sharpness, *J. R. Stat. Soc., Ser. B*, **69**(2):243–268, 2007.
13. Johnson, C. and Bowler, N., On the Reliability and Calibration of Ensemble Forecasts, *Mon. Weather Rev.*, **137**(5):1717–1720, 2009.
14. MacKay, D.J., A Practical Bayesian Framework for Backpropagation Networks, *Neural Comput.*, **4**(3):448–472, 1992.
15. Neal, R.M., *Bayesian Learning for Neural Networks*, Vol. 118, Berlin: Springer Science & Business Media, 2012.
16. Hernández-Lobato, J.M. and Adams, R., Probabilistic Backpropagation for Scalable Learning of Bayesian Neural Networks, in *Proc. of Int. Conf. on Machine Learning*, pp. 1861–1869, 2015.
17. Gal, Y. and Ghahramani, Z., Bayesian Convolutional Neural Networks with Bernoulli Approximate Variational Inference, *Stat. Mach. Learn.*, arXiv:1506.02158, 2015.
18. Rasmussen, C.E. and Williams, C.K., *Gaussian Process for Machine Learning*, Cambridge, MA: The MIT Press, 2006.
19. Platt, J., Probabilistic Outputs for Support Vector Machines and Comparisons to Regularized Likelihood Methods, *Adv. Large Margin Clas.*, **10**(3):61–74, 1999.
20. Zadrozny, B. and Elkan, C., Obtaining Calibrated Probability Estimates from Decision Trees and Naive Bayesian Classifiers, in *Proc. of 18th Int. Conf. on Machine Learning*, pp. 609–616, 2001.
21. Kuleshov, V., Fenner, N., and Ermon, S., Accurate Uncertainties for Deep Learning Using Calibrated Regression, *Comput. Sci. Mach. Learn.*, arXiv:1807.00263, 2018.
22. Levi, D., Gispan, L., Giladi, N., and Fetaya, E., Evaluating and Calibrating Uncertainty Prediction in Regression Tasks, *Comput. Sci. Mach. Learn.*, arXiv:1905.11659, 2019.
23. Song, H., Diethe, T., Kull, M., and Flach, P., Distribution Calibration for Regression, *Stat. Mach. Learn.*, arXiv:1905.06023, 2019.
24. Lakshminarayanan, B., Pritzel, A., and Blundell, C., Simple and Scalable Predictive Uncertainty Estimation Using Deep Ensembles, in *Proc. of Advances in Neural Information Processing Systems*, pp. 6405–6416, 2017.
25. Weigend, A.S. and Nix, D.A., Predictions with Confidence Intervals (Local Error Bars), in *Proc. of the Int. Conf. on Neural Information Processing*, pp. 847–852, 1994.
26. Matheson, J.E. and Winkler, R.L., Scoring Rules for Continuous Probability Distributions, *Manag. Sci.*, **22**(10):1087–1096, 1976.
27. Bröcker, J. and Smith, L.A., Scoring Probabilistic Forecasts: The Importance of Being Proper, *Weather and Forecast.*, **22**(2):382–388, 2007.
28. Murphy, A.H. and Winkler, R.L., Diagnostic Verification of Probability Forecasts, *Int. J. Forecast.*, **7**(4):435–455, 1992.
29. Toth, Z., Talagrand, O., Candille, G., and Zhu, Y., Probability and Ensemble Forecasts, in *Forecast Verification: A Practitioner's Guide in Atmospheric Science*, I.T. Jolliffe and D.B. Stephenson, Eds., New York: Wiley, pp. 137–163, 2003.
30. Winkler, R.L., Munoz, J., Cervera, J.L., Bernardo, J.M., Blattenberger, G., Kadane, J.B., Lindley, D.V., Murphy, A.H., Oliver, R.M., and Ríos-Insua, D., Scoring Rules and the Evaluation of Probabilities, *Test*, **5**(1):1–60, 1996.
31. Wilks, D.S., *Statistical Methods in the Atmospheric Sciences*, Vol. 100, Cambridge, MA: Academic Press, 2011.
32. Hersbach, H., Decomposition of the Continuous Ranked Probability Score for Ensemble Prediction Systems, *Weather Forecast.*, **15**(5):559–570, 2000.
33. Anderson, J.L., A Method for Producing and Evaluating Probabilistic Forecasts from Ensemble Model Integrations, *J. Climate*, **9**(7):1518–1530, 1996.
34. Hamill, T.M., Reliability Diagrams for Multicategory Probabilistic Forecasts, *Weather Forecast.*, **12**(4):736–741, 1997.
35. Hamill, T.M., Interpretation of Rank Histograms for Verifying Ensemble Forecasts, *Mon. Weather Rev.*, **129**(3):550–560, 2001.
36. Goldberg, P.W., Williams, C.K., and Bishop, C.M., Regression with Input-Dependent Noise: A Gaussian Process Treatment, in *Proc. of Advances in Neural Information Processing Systems*, pp. 493–499, 1998.
37. Kersting, K., Plagemann, C., Pfaff, P., and Burgard, W., Most Likely Heteroscedastic Gaussian Process Regression, in *Proc. of the 24th Int. Conf. on Machine Learning*, pp. 393–400, 2007.

38. Yuan, M. and Wahba, G., Doubly Penalized Likelihood Estimator in Heteroscedastic Regression, *Stat. Probab. Lett.*, **69**(1):11–20, 2004.
39. Williams, P.M., Using Neural Networks to Model Conditional Multivariate Densities, *Neural Comput.*, **8**(4):843–854, 1996.
40. Genz, A., Testing Multidimensional Integration Routines, in *Proc. of Int. Conf. on Tools, Methods and Languages for Scientific and Engineering Computation*, pp. 81–94, New York, 1984.
41. Jain, A.K., Data Clustering: 50 Years Beyond K-Means, *Pattern Recognit. Lett.*, **31**(8):651–666, 2010.