NPL
National Physical Laboratory

NPL REPORT MS 34

UNCERTAINTY EVALUATION FOR MACHINE LEARNING

ANDREW THOMPSON, KAVYA JAGAN, ASHISH SUNDAR, RAHUL KHATRY, JAMES DONLEVY, SPENCER THOMAS AND PETER HARRIS

NOVEMBER 2021

Department for
Business, Energy
& Industrial Strategy

FUNDED BY BEIS

NPL Report MS 34

Uncertainty evaluation for machine learning

Andrew Thompson, Kavya Jagan, Ashish Sundar,
Rahul Khatry, James Donlevy, Spencer Thomas and Peter Harris
Data Science

ABSTRACT

We explore the challenge of uncertainty evaluation in machine learning regression problems. We begin by outlining how regression problems typically arise in a metrology setting. We then perform a numerical investigation of three methods for performing uncertainty-aware machine learning regression, namely Gaussian Processes (GPs), Monte-Carlo Dropout and Deep Ensembles. Based upon our investigations, we identify a list of metrology requirements that methods for uncertainty evaluation in machine learning regression must satisfy, and we evaluate the three methods against these requirements. We conclude by describing how the requirements give rise naturally to several open challenges which constitute an agenda for future work.

NPL Report MS 34

https://doi.org/10.47120/npl.MS34

ISSN 1754-2960

National Physical Laboratory
Hampton Road, Teddington, Middlesex, TW11 0LW

Approved on behalf of NPLML by
Louise Wright, Head of Science (Data Science)

CONTENTS

1 INTRODUCTION

Accurate, standardised and quality-assured measurement brings numerous benefits to society. This goal is achieved through metrology infrastructures which define primary measurement standards and ensure that all measurements are traceable to these standards through an unbroken chain of calibrations, each contributing to the measurement uncertainty (BIPM et al.).

Machine learning (ML) and artificial intelligence (AI) has the potential to extend the reach of metrology to signal processing applications such as advanced manufacturing, autonomous transport, network communications and medical imaging, in which the underlying physical model is either not well understood or inefficient to compute. However, the adoption of ML with all its benefits is hindered by the perceived untrustworthiness of its outputs (Floridi, 2019). More specifically, it is fundamental to the traceability of ML predictions that they are accompanied by reliable quantitative assessment of uncertainty.

While recent advances in ML have focused to a large extent upon classification tasks, a typical use of ML in metrology is in a regression task in which a continuous quantity is to be estimated. Although the goal is to use information about a continuous (univariate) quantity to make decisions or to assess whether a system meets specific requirements, the fundamental challenge is to measure the quantity to provide that information, and so we will focus in this report on regression problems.

How do regression problems arise in metrology? They arise because it is often not possible or desirable to measure a quantity directly, but rather it is necessary or desirable to infer information about a quantity from a number of contributions which are easier to measure or for which information is available. These contributions could relate to measurements made by some instrument or system, to applied corrections, or to information obtained from sources such as manufacturer's specifications or calibration certificates. A model is then needed to infer information about the quantity of interest from these contributions. The framework of evaluating uncertainties by means of measurement models was standardised for the metrology community in the influential "Guide to the Expression of Uncertainty in Measurement" (GUM) (BIPM et al., 2008a).

There are many scenarios in which it is not possible to build or perform computations on an analytical model based upon physical understanding, and instead a data-driven approach must be taken. Take three example measurement problems where ML is either currently used or where it offers potential.

- In environmental monitoring, the data provided by sensors is used for forecasting to understand how the state of the environment is changing with time and the impacts of different sources, including anthropogenic sources, on the environment. An example is the use of underwater hydroacoustic sensors (hydrophones) to provide data about levels of sound in the oceans to understand how human activity is affecting those levels. One approach to forecasting is to use a model having a parametric form derived from an understanding of the physics underlying the deterministic and statistical processes that give rise to the data Harris et al. (2019).

However, for many applications, such an approach is not practical because the physical processes are far too complicated to be described in this way and because the environmental quantity of interest is far removed, through a series of data reduction steps, from the quantity that is actually recorded by the sensor. In these cases, methods based on ML, including Gaussian Processes and recurrent neural networks, may provide a viable solution Robinson et al. (2021).

- Mass spectrometry (MS) is a suite of surface analysis techniques that measure the chemical composition of a sample. This is achieved through removal of material from the surface (e.g desorption) and recording the intensity of a particular mass to charge (m/z) ratio of the removed material. The resulting data are high dimensional and heterogeneous intensity spectra containing peaks that characterise particular compounds of the analysed material Thomas et al. (2016). The spectra obtained are contingent on the physico-chemical properties of the sample, process of material removal, and the instrumentation measuring the intensities, hence analysis is difficult.
- In earth observation and remote sensing, inter-instrument bias is a ubiquitous problem, but a complete theoretical understanding of the causes of such biases is rarely available. Machine learning has therefore used for bias correction and cross-calibration of sensors in the context of various types of measurement including vegetation indices and ozone depletion. Machine learning has also been used to build data-driven models which infer measurements of airborne particulates and pollen estimation. We refer the interested reader to the review article Lary et al. (2018) for more details.

The solution, then, in metrology applications such as these, is to learn a measurement model from calibration data.

Suppose then that we are interested in measuring some quantity Y indirectly; we will refer to this quantity as the *measurand* or the output quantity. We have access to a set of *calibration data* comprising multiple values of the measurand Y and corresponding values of the contributions $X_1, \ldots, X_N$ in the measurement model. The regression problem is to learn the measurement model $Y \approx f(X_1, \ldots, X_N)$. The typical approach in ML is to restrict to some parametrised family of models and then learn its parameters. This process is usually described in ML as training the model, and for this reason the calibration data in an ML context is also referred to as the *training data*.

Having learned a model f, we then wish to use it to obtain an estimate along with an assessment of uncertainty for Y given similar information for the input quantities $X_1, \ldots, X_N$.

One of the foundational principles of uncertainty evaluation in metrology, systematized by the GUM framework (BIPM et al., 2008a,b, 2011), is that uncertainty is not an intrinsic property of a measurand, but rather a *probabilistic statement of belief* based upon our perceived knowledge. For example, uncertainty is defined in the GUM as "a parameter associated with the result of a measurement, that characterizes the dispersion of the values that could reasonably be attributed to the measurand". The most

comprehensive form that such a statement might take is the probability density function (pdf) of the measurand Y. In an ML context, only a numerical approximation to the pdf is realistic, which might be captured by an evaluation of certain quantiles of the distribution.

Alternatively, if it is legitimate to make assumptions concerning the distribution, for example that it is a Gaussian, information might be captured by summary information such as the mean (often referred to as the *best estimate* y) and standard deviation (often referred to as the *standard uncertainty* $u(y)$) of Y. Coverage intervals which contain Y with a stated probability p might also be obtained. Symmetric coverage intervals of the form $[y - U_p, y + U_p]$, where $U_p = k_p u(y)$ is an *expanded uncertainty* and where k_p is a *coverage factor*, might be obtained. Alternatively, a coverage interval of shortest length for a given probability p might be chosen (the two approaches are equivalent if the distribution is unimodal and symmetric about the mean).

It is realistic to expect that similar statements of belief about each of the input quantities $X_1, \ldots, X_N$ are also available.

Since uncertainty is conceived of as an expression of belief, a Bayesian framework is considered by many to be preferable in metrology, and there have been proposals to revise the GUM entirely along Bayesian lines (Elster and Toman, 2011; Bich et al., 2012; Bich, 2014).

Traditional uncertainty evaluation of the kind systematized by the GUM framework (BIPM et al., 2008a,b, 2011) makes the assumption that the measurement model is known. Uncertainty evaluation in ML deviates from this paradigm in one key respect. An ML model is not deduced from physical understanding, but is learned from data. It follows that, even after an ML model has been learned from data, the absence of physical insight leads to uncertainty in the model itself.

More recent work in metrology has focused on the use of Bayesian inference methods for uncertainty evaluation in parametrised models (Elster, 2007; Klauenberg et al., 2015). These approaches learn by combining data with prior knowledge, and replace the measurement model with a model of the observation (Forbes and Sousa, 2011). Learning measurement models in ML is closely related conceptually, but the main difference is that in an ML context little can be assumed about the form of the model, and heavily overparametrised models must be learned based on large volumes of data.

In this report, we describe three of the most popular approaches to uncertainty evaluation for ML regression problems and how they capture uncertainty (Section 2). We then perform numerical experiments on a synthetic test problem in order to investigate the performance of the three algorithms (Section 3). Armed with these observations, we then propose a framework of seven requirements that methods for machine learning uncertainty evaluation should meet in a metrology setting (Section 4). We then return to the three algorithms and evaluate them through the lens of these seven requirements (Section 5) in order to assess their suitability for use in a metrology setting. Finally, we summarize some of the open challenges in the area which represents a roadmap for future work (Section 6).

2 METHODS FOR UNCERTAINTY EVALUATION

2.1 ALEATORIC AND EPISTEMIC UNCERTAINTY

Before introducing three popular methods for uncertainty evaluation with ML, we introduce two notions which arise in the discussion of all three methods. In the ML literature, a distinction is often made between *aleatoric* uncertainty and *epistemic* uncertainty. Aleatoric uncertainty, often equated with random effects, refers to uncertainty that is inherent to the problem. Epistemic uncertainty, on the other hand, refers to uncertainty in the model. Aleatoric uncertainty is a property of the data and it is in this sense irreducible, and it is often equated with random effects. Epistemic uncertainty, on the other hand, is a property of the model, and it can be reduced either by increasing the volume of training data or specifying a better model. All three methods we describe allow for a decomposition of uncertainty into aleatoric and epistemic components. See Section 4.2 for a more in-depth discussion of these two notions, including how they relate to metrology.

2.2 GAUSSIAN PROCESSES

Gaussian Process (GP) regression (Williams and Rasmussen, 1996; Rasmussen, 2003) is a flexible way of modelling data that is not limited by a functional form. GPs are typically used to model a variable that has spatial (and/or temporal) dependence. They do so by regarding the values of the variable at different spatial locations as being correlated and describing the strength of the correlation in a flexible way as a function of spatial separation. The idea of GP regression is to learn the correlation behaviour from measured data collected at known, fixed locations and this knowledge can be used for prediction/interpolation within an area of interest. In effect, the interpolated value of the variable at a given location is evaluated as a weighted average of the measured data where the weights depend on the spatial correlation behaviour and hence the separations of the measured locations from the given point.

Just as a Gaussian distribution is a probability distribution over a finite set of random variables y, $y \sim \mathrm{N}(\mu, V)$, defined by a mean vector μ and covariance matrix V, a GP is a probability distribution over functions (or an infinite set of random variables) $y(x)$,

$$y(x) \sim \mathrm{GP}(\mu(x), V(x, x')),$$

defined by a mean function $\mu(x)$ and a covariance kernel $V(x, x')$, and has the property that any finite subset of function values has a Gaussian distribution. One commonly used covariance kernel, the radial basis function (RBF) kernel is expressed as

$$V(x_i, x_j) = \sigma_1^2 \exp\left\{-\frac{(x_i - x_j)^2}{\sigma_2^2}\right\},$$

where σ_1^2 governs the (prior) variance of variable y at any point x and σ_2^2 governs the (prior) correlation of the variable y at points x_i and x_j and consequently, controls the

"smoothness" of the interpolated function. These σ parameters (and hence the covariance kernel) are learnt from the data. Once these parameters are learnt, they can be used to make predictions at arbitrary locations.

The GP regression model can be defined as

$$\begin{bmatrix}\mathbf{y}_1\\ \mathbf{y}_2\end{bmatrix} \sim \mathrm{N}\left(\begin{bmatrix}\mathbf{C}_{\mathbf{x}_1}\\ \mathbf{C}_{\mathbf{x}_2}\end{bmatrix}\boldsymbol{\alpha}, \begin{bmatrix}\mathbf{V}_{11} & \mathbf{V}_{21}^\top\\ \mathbf{V}_{21} & \mathbf{V}_{22}\end{bmatrix}\right), \tag{1}$$

where $\mathbf{x}_1$ and $\mathbf{y}_1$ represent the training data, and $\mathbf{y}_2$ represent the predictions at test locations, $\boldsymbol{\alpha}$ are the parameters of the mean function, considered here to depend linearly on the parameters α, $\mathbf{V}_{ij}$ are the covariance matrices obtained by evaluating the covariance kernel for the locations of the training data and test data, and $\mathbf{C}_{\mathbf{x}_1}$ and $\mathbf{C}_{\mathbf{x}_2}$ are matrices whose rows are the training and test input vectors respectively.

When there is noise associated with the training data, this can be included in the model by adding a variance term to the diagonals of $\mathbf{V}_{11}$ i.e., $\mathbf{V}_{11} + \sigma_n^2\mathbf{I}$, where $\mathbf{I}$ is an identity matrix the same size as the training data. The GP regression model with noise can be defined as

$$\begin{bmatrix}\mathbf{y}_1\\ \mathbf{y}_2\end{bmatrix} \sim \mathrm{N}\left(\begin{bmatrix}\mathbf{C}_{\mathbf{x}_1}\\ \mathbf{C}_{\mathbf{x}_2}\end{bmatrix}\boldsymbol{\alpha}, \begin{bmatrix}\mathbf{V}_{11} + \sigma_n^2 I & \mathbf{V}_{21}^\top\\ \mathbf{V}_{21} & \mathbf{V}_{22}\end{bmatrix}\right).$$

In the rest of this section, we use the model defined in Equation (1). But the following equations can be easily modified to include measurement noise by adding the noise variance term to the diagonal elements of $\mathbf{V}_{11}$.

In a Bayesian paradigm, prior knowledge of model parameters can be captured through a prior distribution on the parameters $p(\boldsymbol{\sigma})$ (Forbes, 2015). The training data $\mathbf{x}_1$ and $\mathbf{y}_1$ can be used to learn about the parameters of the covariance function by maximising the log-marginalised posterior distribution

$$\log p(\boldsymbol{\sigma}|\mathbf{x}_1, \mathbf{y}_1) \propto \frac{1}{2}F(\boldsymbol{\sigma}) + \log|\mathbf{V}_{11}| + \log|\mathbf{C}_{\mathbf{x}_1}^\top\mathbf{V}_{11}^{-1}\mathbf{C}_{\mathbf{x}_1}| - \log p(\boldsymbol{\sigma}), \tag{2}$$

where

$$F(\boldsymbol{\sigma}) = (\mathbf{y}_1 - \mathbf{C}_{\mathbf{x}_1}\hat{\boldsymbol{\alpha}})^\top\mathbf{V}_{11}^{-1}(\mathbf{y}_1 - \mathbf{C}_{\mathbf{x}_1}\hat{\boldsymbol{\alpha}}),$$

and $\hat{\boldsymbol{\alpha}}$ is the best estimate of $\boldsymbol{\alpha}$, given by

$$\hat{\boldsymbol{\alpha}} = (\mathbf{C}_{\mathbf{x}_1}^\top\mathbf{V}_{11}^{-1}\mathbf{C}_{\mathbf{x}_1})^{-1}\mathbf{C}_{\mathbf{x}_1}^\top\mathbf{V}_{11}^{-1}\mathbf{y}_1,$$

with an associated covariance matrix

$$\mathbf{V}_{\boldsymbol{\alpha}} = (\mathbf{C}_{\mathbf{x}_1}^\top\mathbf{V}_{11}^{-1}\mathbf{C}_{\mathbf{x}_1})^{-1}.$$

Hence, the best estimate of the prediction $\mathbf{y}_2$ is given by

$$\boldsymbol{\mu}_{\mathbf{y}_2} = \mathbf{V}_{21}\mathbf{V}_{11}^{-1}\mathbf{y}_1 + \mathbf{E}\hat{\boldsymbol{\alpha}},$$

where $\mathbf{E} = \mathbf{C}_{\mathbf{x}_2} - \mathbf{V}_{21}\mathbf{V}_{11}^{-1}\mathbf{C}_{\mathbf{x}_1}$.

A particular advantage of GP regression is that the uncertainties associated with predicted values are automatically provided. Uncertainty here refers to the margin of doubt (or conversely the level of confidence) associated with predictions. The uncertainties for points close to the training data generally tend to be smaller than for points far from them. The covariance matrix associated with the model predictions is given by

$$\mathbf{V}_{\mathbf{y}_2} = \mathbf{V}_{22} - \mathbf{V}_{21}\mathbf{V}_{11}^{-1}\mathbf{V}_{21}^{\top} + \mathbf{E}\mathbf{V}_{\boldsymbol{\alpha}}\mathbf{E}^{\top}.$$

In a Bayesian setting, assuming a non-informative prior for $\boldsymbol{\alpha}$, it follows that the posterior predictive distribution of $\mathbf{y}_2$ given $\mathbf{y}_1$ is

$$\mathbf{y}_2|\mathbf{y}_1 \sim \mathrm{N}(\boldsymbol{\mu}_{\mathbf{y}_2}, \mathbf{V}_{\mathbf{y}_2}). \tag{3}$$

The diagonal entries of the covariance matrix $\mathbf{V}_{\mathbf{y}_2}$ of the predictions can be used as a measure of overall predictive uncertainty. We then subtract the aleatoric contribution to the variance from the overall predictive variance to obtain the epistemic variance (and hence uncertainties).

2.2.1 Taking into account uncertainty on hyperparameters

The correlation structure in data is captured by GPs through $\boldsymbol{\sigma}$, the hyperparameters that define the covariance kernel. In order to make predictions from GPs, these hyperparameters are learnt from the training data by minimising the log of the marginalised posterior (2). They are then treated as known and predictions are made based on these constants. Uncertainties associated with the hyperparameter estimates are not usually propagated through to predictions. In some practical situations, for instance when there is a high degree of correlation between the hyperparameters, accounting for hyperparameter uncertainty may lead to more reliable predictions.

Bayesian hierarchical models can be used to capture the lack of perfect knowledge of the hyperparameters. The exponent of Equation (2) is the marginalised posterior $p(\boldsymbol{\sigma}|\mathbf{y}_1)$ of the hyperparameters $\boldsymbol{\sigma}$ given the training data $(\mathbf{x}_1, \mathbf{y}_1)$. The posterior predictive distribution for $\mathbf{y}_2$ given $\mathbf{y}_1$ is given by

$$p(\mathbf{y}_2|\mathbf{y}_1) = \int_{\boldsymbol{\sigma}} p(\mathbf{y}_2|\mathbf{y}_1, \boldsymbol{\sigma}) p(\boldsymbol{\sigma}|\mathbf{y}_1) \mathrm{d}\boldsymbol{\sigma}.$$

This posterior predictive distribution is a weighted average of Gaussian distributions since we know from Equation (3) that $p(\mathbf{y}_2|\mathbf{y}_1, \boldsymbol{\sigma})$ is a multivariate Gaussian distribution. The predictive distribution cannot be defined analytically, and hence a sampling scheme is used to generate samples from this distribution. One sampling strategy is to use the Metropolis-Hastings Markov Chain Monte Carlo (MCMC) method (Gelman et al., 2013) to generate samples of $\boldsymbol{\sigma}$ from $p(\boldsymbol{\sigma}|\mathbf{y}_1)$. This sample of $\boldsymbol{\sigma}$ can then be used to generate a corresponding sample of $\mathbf{y}_2$ from the Gaussian distribution $p(\mathbf{y}_2|\mathbf{y}_1, \boldsymbol{\sigma})$. However, this scheme could be quite computationally expensive. An alternative approach is to approximate the distribution $\log(p(\boldsymbol{\sigma}|\mathbf{y}_1))$ by a Gaussian distribution centered at the maximum

a posteriori (MAP) estimates of $\log(\boldsymbol{\sigma})$ with the variance matrix approximated by the inverse of the Hessian matrix evaluated at the MAP estimate. Generating samples from this Gaussian distribution is far cheaper than MCMC but there will be an associated approximation error. As before, the corresponding sample of $\mathbf{y}_2$ can be generated by sampling from $p(\mathbf{y}_2|\mathbf{y}_1, \boldsymbol{\sigma})$ evaluated at sampled values of $\boldsymbol{\sigma}$ (note that the sample of $\boldsymbol{\sigma}$ can be obtained by exponentiating the sample of $\log(\boldsymbol{\sigma})$).

2.3 MONTE CARLO DROPOUT

Dropout is a popular regularisation technique, first proposed in (Srivastava et al., 2014) as a means of preventing neural networks from overfitting. The essential idea of dropout is to randomly drop nodes of the network (along with its connections) during training. The proposal in (Srivastava et al., 2014) is, in each epoch, to drop each node with some fixed probability.

Dropout was repurposed in (Gal, 2016; Gal and Ghahramani, 2016) as a method for obtaining uncertainty evaluations for neural networks. In this approach, an ensemble of models is learned, each of which drops out nodes over the entire training with some fixed probability (the dropout rate). Then this ensemble is used at test time to obtain an empirical distribution for the predictions, a method referred to as Monte Carlo Dropout (or MC-Dropout for short). The empirical distribution obtained in this way can be viewed as performing approximate Bayesian inference, and we will return at the end of this section to discussing this interpretation.

First, we describe the MC-Dropout method more precisely for the simplified case of a neural network with a single hidden layer. It is then intuitive to extend to deeper networks. Let us assume then a fully-connected neural network with a single hidden layer consisting of K nodes. The model can be written as

$$f(\mathbf{x}|\mathbf{W}_1, \mathbf{w}_2, \mathbf{c}) = \mathbf{w}_2^T \left[\rho(\mathbf{W}_1\mathbf{x} + \mathbf{c})\right],$$

where $\mathbf{W}_1 \in \mathbb{R}^{K\times N}$ and $\mathbf{w}_2 \in \mathbb{R}^K$ are the convolution weights, $\rho : \mathbb{R}^K \to \mathbb{R}^K$ denotes a componentwise nonlinearity such as a rectified linear unit (ReLU), and where $\mathbf{c} \in \mathbb{R}^K$ is the bias by which the input of the nonlinearity $\rho(\cdot)$ is shifted.

To randomly drop out nodes, we introduce two binary vectors $\mathbf{b}_1 \in \mathbb{R}^N$ and $\mathbf{b}_2 \in \mathbb{R}^K$, the elements of which take the value 1 with probability $0 \leq p \leq 1$ and are otherwise zero. Writing $\mathbf{B}_1 \in \mathbb{R}^{N\times N}$ and $\mathbf{B}_2 \in \mathbb{R}^{K\times K}$ for the diagonal matrices whose diagonals are $\mathbf{b}_1$ and $\mathbf{b}_2$ respectively, we can write the model as

$$f(\mathbf{x}|\mathbf{W}_1, \mathbf{w}_2, \mathbf{c}) = \mathbf{w}_2^T\mathbf{B}_2 \left[\rho(\mathbf{W}_1\mathbf{B}_1\mathbf{x} + \mathbf{c})\right].$$

For networks with more hidden layers, binary variables are similarly introduced before each convolution.

We train an ensemble of E such dropout networks. As in (Gal, 2016), Euclidean loss with l_2 regularisation on the weights is used, giving for each dropout network the

weight optimisation problem

$$\min_{\mathbf{W}_1,\mathbf{w}_2,\mathbf{c}} \sum_{i=1}^{N} [y_i - f(\mathbf{x}_i)]^2 + \lambda(\|\mathbf{W}_1\|_F^2 + \|\mathbf{w}_2\|_2^2 + \|\mathbf{c}\|_2^2).$$

We note the presence of two parameters p and λ for which there is no principled a priori choice and so which need to be tuned. The Adam optimiser, which requires the tuning of two further parameters, the learning rate and the momentum, is then used to fit the dropout models.

At test time, all E models are evaluated on each sample, and a sample mean and standard deviation are calculated. These statistics can then be used to obtain credible intervals by assuming that the samples are drawn from a Gaussian distribution.

It has been shown that MC-Dropout can be viewed as an approximate Bayesian inference over the model weights in two ways.

- It was shown in (Gal, 2016) that MC-Dropout can be viewed as performing approximate variational inference (VI) (Graves, 2011). VI aims to make Bayesian inference more computationally tractable by restricting the posterior distribution to some parameterised class of distributions and then optimising for these parameters by minimising the Kullback-Leibler (KL) divergence to the posterior. In the case of MC-Dropout, and indeed other VI approaches to Bayesian neural networks (Graves, 2011; Blundell et al., 2015; Kingma, 2017), the posterior is restricted to be multivariate Gaussian, with independence except for between weights emanating from the same node. It is shown in (Gal, 2016) that MC-Dropout approximates VI for the case where independent slab-and-spike priors are placed on all weights, namely

$$w \begin{cases} \sim & \mathcal{N}(0, l^2) & \text{with probability } p, \\ = & 0 & \text{with probability } 1-p. \end{cases}$$

  We emphasise, however, that a number of approximations are made in order to make this connection, and in particular that the approximation improves as the number of layers of the network increases. Furthermore, it is not transparent exactly what prior distribution (the parameter l) is being assumed.

- It was shown in (Gal and Ghahramani, 2016) that MC-Dropout can be viewed as an approximation to the probabilistic deep Gaussian process (Damianou and Lawrence, 2013) (marginalised over its covariance parameters). This approximation improves as the width of each layer of the network increases.

It may also be observed that there are connections between ensembling methods and Bayesian model averaging which could be explored further.

2.4 DEEP ENSEMBLES

Ensembling is a popular approach in supervised learning in which the predictions of multiple models are averaged, and it has also been used historically to capture the

variability in the parameters of a model. Deep ensembles use ensembles of so-called 'double-headed' neural networks to quantify uncertainty. Each model in the ensemble outputs both a mean $\hat{\mu}_i(\boldsymbol{x_i}; \mathbf{w})$ and variance $\hat{\sigma}_i^2(\boldsymbol{x_i}; \mathbf{w}) > 0$ for each prediction ($i \in \{1, \ldots, N\}$), where here $\mathbf{w}$ denotes the weights of the network. Treating the observed values as samples from a Gaussian distribution with the predicted mean and variance leads to minimising the negative log likelihood function.

$$\mathcal{L}(\mathbf{w}) \propto \sum_{i=1}^{N} \frac{\log \hat{\sigma}_i^2(\boldsymbol{x}_i; \mathbf{w})}{2} + \frac{(y_i - \hat{\mu}_i(\boldsymbol{x}_i; \mathbf{w}))^2}{2\hat{\sigma}_i^2(\boldsymbol{x}_i; \mathbf{w})}.$$

The variance associated with each data sample is a measure of the level of uncertainty associated with that data sample.

The variation in the output mean values provided by the models in the ensemble, measured by the standard deviation of those values, is used to describe the epistemic uncertainty (or model uncertainty). The epistemic uncertainty refers to the uncertainty that can be improved upon by supplying more data.

As was explained above, each of the the models themselves outputs a mean and a variance associated with each data sample. These variances are used to estimate the uncertainty associated with each prediction, and averaging these variances across the E models gives us an estimate of the aleatoric uncertainty - the irreducible uncertainty associated with the data itself. We note that the assumption is made that the variances associated with each model are the same, and we also note that the estimate of aleatoric uncertainty improves by adding more training data.

The networks themselves, as in the case of MC-Dropout considered in Section 2.3, are taken to be fully-connected with an arbitrary number of layers and nodes in each hidden layer. The positivity constraint on the variance is enforced by passing the second output (variance) through the softplus function $\log(1 + \exp(\cdot))$ and adding a minimum variance of 10^{-6} for numerical stability. We note that this *ad hoc* strategy might be fooled by a poor scaling of the data, and improvements could be explored.

The models are different due to the randomness introduced by random initialization of the weights and random shuffling of the data points (leading to random batching strategies). In (Lakshminarayanan et al., 2017), the use of adversarial training (Goodfellow et al., 2014) is also advocated, a technique in which perturbations of the existing training input data in directions likely to increase the loss are added to the training dataset. However, for the sake of simplicity we did not use adversarial training in our implementation of the method.

As in the case of MC-Dropout, a variant of batch gradient descent (such as the Adam optimiser (Kingma and Ba, 2014)) is used to optimise the loss function, which requires tuning of the learning rate and momentum parameters.

Deep ensembles were conceived of as a non-Bayesian or frequentist approach to uncertainty evaluation. Given the connections between ensembling and model averaging, it would be interesting to explore the possibility of a Bayesian interpretation of deep ensembles.

3 NUMERICAL EXPERIMENTS

Validating uncertainty evaluation methods for machine learning is in general complicated by the fact that ground truth information concerning uncertainty is not usually available.

One way to mitigate this difficulty is to focus upon a synthetic test problem for which the true model and the uncertainties in the data are known. It is also helpful to restrict attention to a low-dimensional setting to enable visualisation, and so with this in mind we consider regression with respect to univariate input data. The specific test problems we investigate are all inspired by a single test problem first considered in a Master's thesis by Bachstein (Bachstein, 2019), in which each of the three methods for uncertainty evaluation considered in this report (and some others) were also tested. More specifically, we work with synthetic data generated by the function

$$f(x) = 2 \cdot \left[\frac{x}{10} + \sin\left(\frac{4x}{10}\right) + \sin\left(\frac{13x}{10}\right) \right],$$

restricted to the interval $[-5, 5]$. We consider a *homoscedastic* noise model in which we add independent Gaussian noise, such that our observational data is given by

$$y_i = f(x_i) + \epsilon_i, \quad \epsilon_i \sim \mathcal{N}(0, \sigma^2)$$

for some noise variance σ^2.

We follow (Bachstein, 2019) in differentiating two main types of uncertainty: *aleatoric* and *epistemic* uncertainty (Kendall and Gal, 2017; Hllermeier and Waegeman, 2019). Aleatoric uncertainty refers to uncertainty which is intrinsic to the problem, and by which is usually meant random effects. Epistemic uncertainty, on the other hand, refers to uncertainty concerning the model. We refer the reader to Section 4.2 for more details and a metrology perspective on this distinction. In order to explore how each method captures random (aleatoric) uncertainty, we experiment with two choices for the noise level: $\sigma = 0.5$ and $\sigma = 1$. In order to explore how each method captures uncertainty due to model misspecification (epistemic uncertainty), we draw training samples uniformly at random, but we experiment with different numbers m of training data samples. We note that the choices of m are not intended to assess scaleability, and the focus is upon comparing the uncertainty evaluation performance. We also explore epistemic uncertainty in the context of two types of missing data scenario: an interpolation and an extrapolation problem.

Interpolation. The interpolation task is to 'fill in' the measurement function between values for which data exists. To study the behaviour of the algorithms in this setting, we draw training samples uniformly at random from the interval $[-5, -\nu] \cup [\nu, 5]$, for $\nu \in \{0.5, 1, 1.5, 2\}$.

Extrapolation. The extrapolation task is to project out the measurement function to areas of data space for which no data is available. To study this, we draw training samples from $[-5, 5]$ but examine predictions over the expanded range $[-7, 7]$. It is widely acknowledged that building models which extrapolate accurately is beyond the ability of machine learning, unless expert prior knowledge is incorporated. However, one may still ask whether a given method for uncertainty evaluation is able to indicate that little is known by returning a large evaluation of uncertainty.

For a test set, we take n equispaced points on the interval $[-5, 5]$ (or the expanded range $[-7, 7]$). Choosing equispaced test samples in contrast to the randomly drawn training samples ensures that the training and test sets are distinct. However, for such a simple univariate problem there is still clearly a great deal of similarity between the training and test sets, and so this feature is somewhat cosmetic.

Following (Bachstein, 2019), we use a visualisation tool to assess the results. We present plots in which several pieces of information are superimposed. We plot the training samples and the best estimate prediction of the model. We also plot uncertainty in the form of a 95% symmetric uncertainty interval, which may also be viewed as an expanded uncertainty corresponding to approximately two standard deviations either side of the best estimate assuming a Gaussian distribution. We plot three such intervals:

1. The original data uncertainty (fixed by the choice of σ)
2. The returned estimate of aleatoric uncertainty
3. The returned estimate of epistemic uncertainty.

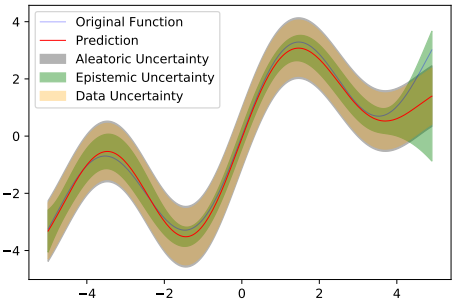
3.1 GAUSSIAN PROCESSES

We implemented GP regression as described in Section 2.2 with an RBF covariance kernel. Maximum likelihood estimation was used to learn the hyperparameters, using the LBFGS optimizer, with initializations $\sigma_1 = \sigma_2 = 3$ and $\sigma = 5$. We note that uncertainty on the hyperparameters themeselves (as described in Section 2.2.1) was not taken into account in these experiments.

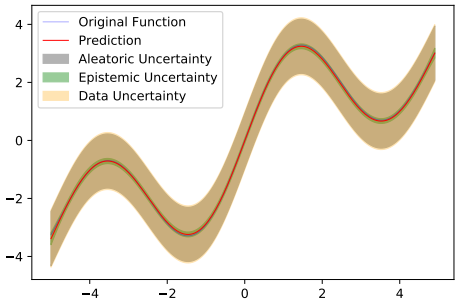
Figure 1 shows the aleatoric and epistemic uncertainty obtained upon the test problem with two choices of the noise level σ (0.5 and 1) and for two different numbers of training samples m (40 and 1000).

We make the following observations.

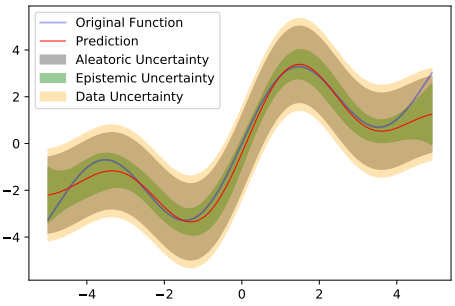
- We observe a decrease in the epistemic uncertainty with no noticeable change to the aleatoric uncertainty when more training samples are used.
- Both types of uncertainty increase as the noise level increases.
- We observe that aleatoric uncertainty is estimated accurately, though less so for larger noise and smaller number of training data samples.



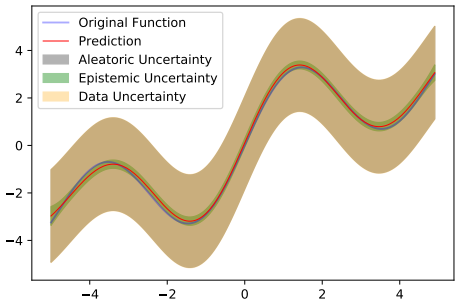
(a) $\sigma = 0.5$ **and** $m = 40$**.**



(b) $\sigma = 0.5$ **and** $m = 1000$**.**



(c) $\sigma = 1$ **and** $m = 40$**.**



(d) $\sigma = 1$ **and** $m = 1000$**.**

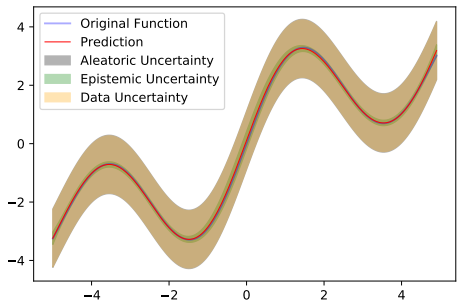
Figure 1 Uncertainty plots for different noise levels and numbers of training samples.

Figure 2 shows the results on the interpolation problem with different width intervals excluded from the training samples, and for the two noise levels $\sigma = 0.5$ and $\sigma = 1$. We fix the number of training samples m to be 1000 in these experiments.

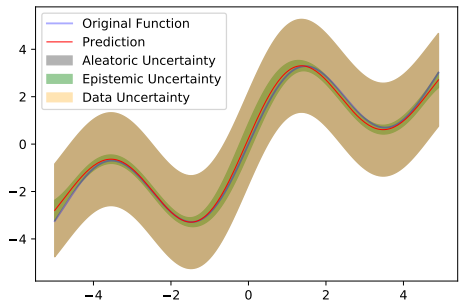
We make the following observations.

- We observe that the epistemic uncertainty increases within the interval for which training samples are excluded, reflecting the increased uncertainty concerning the model.
- The increase in epistemic uncertainty is especially great as the interval expands to encompass turning points in the original function. This makes sense as it is intuitively easier to interpolate over an interval in which the function is monotonic than over an interval in which the function has a turning point in an unknown location.
- The original function is contained in the 95% credible interval for the epistemic uncertainty as one might reasonably expect.

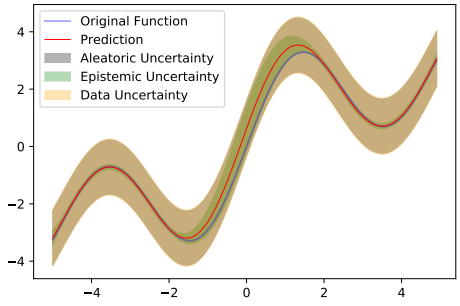
Figure 3 shows the results on the extrapolation problem for the same two noise levels σ. We observe that the epistemic uncertainty increases dramatically in the intervals that



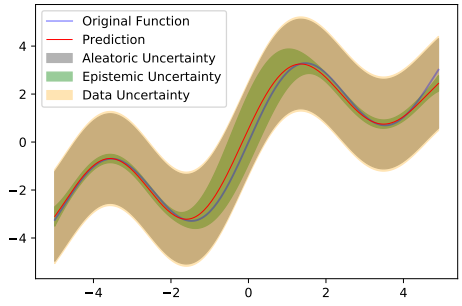
(a) $\sigma = 0.5$ **and** $\nu = 1$**.**



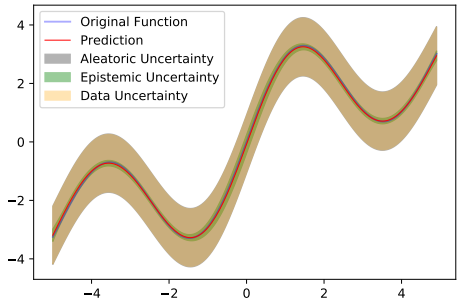
(b) $\sigma = 1$ **and** $\nu = 1$**.**



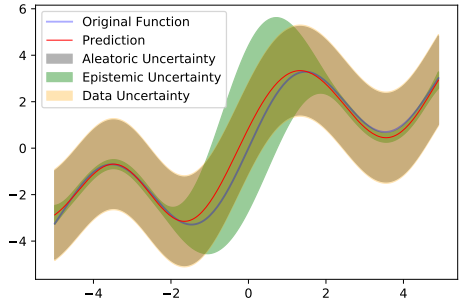
(c) $\sigma = 0.5$ **and** $\nu = 1.5$**.**



(d) $\sigma = 1$ **and** $\nu = 1.5$**.**



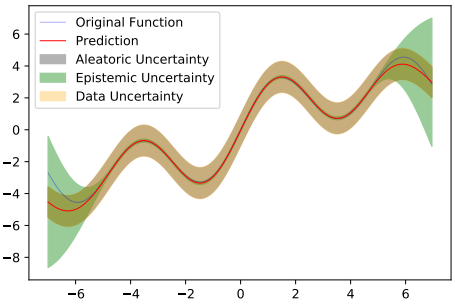
(e) $\sigma = 0.5$ **and** $\nu = 2$**.**



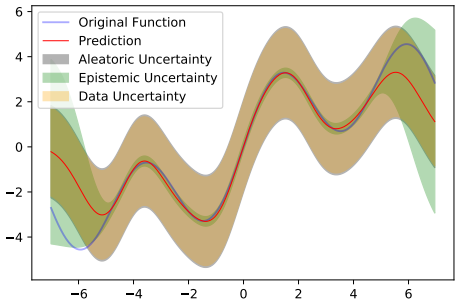
(f) $\sigma = 1$ **and** $\nu = 2$**.**

Figure 2 Uncertainty plots for GPs on interpolation problems with different interval widths ν and different noise levels σ.

have not been sampled, intuitively reflecting the lack of belief the GP has concerning function values in these intervals.



(a) $\sigma = 0.5$**.**



(b) $\sigma = 1$**.**

Figure 3 Uncertainty plots for the extrapolation problem with different noise levels σ**.**

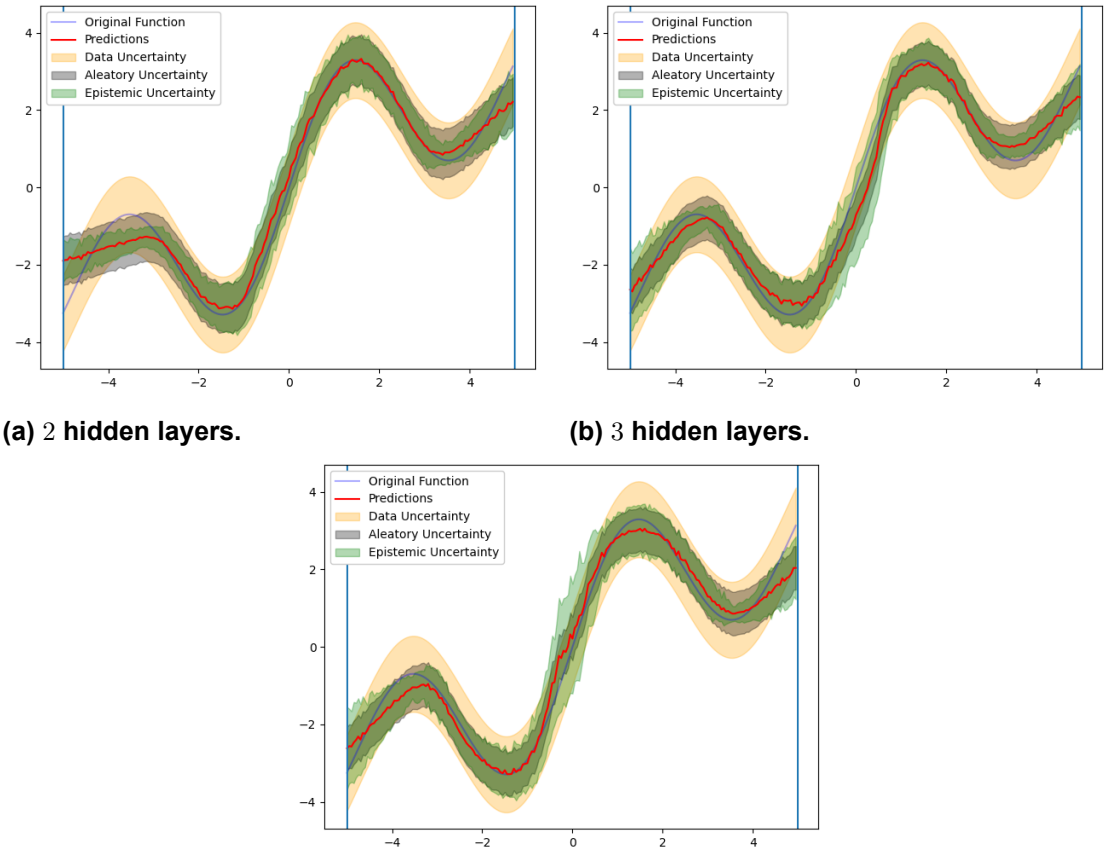
3.2 MONTE-CARLO DROPOUT

Throughout this section we use the same test function as previously described with the number of training samples fixed to be $m = 80$ and the number of test samples fixed to be $n = 1000$. In our first experiment, we highlight how sensitive the MC-Dropout method is to the choice of neural network architecture. Figure 4 shows uncertainty plots for MC-Dropout with two, three and four hidden layers respectively, where each hidden layer has 50 nodes and where ReLU activation functions are used throughout. We recall from Section 2.3 that MC-Dropout has two parameters that must be tuned: the dropout rate p and the regularisation parameter λ. For this experiment, we set $p = 0.95$ and $\lambda = 0.3$. We use the RMS Prop algorithm (Hinton et al., 2012) with batch size 50 and number of epochs 1000 to optimise the loss function. We use an ensemble size of $E = 80$ to perform dropout sampling.

We make the following observations.

- We observe that, in all cases, aleatoric uncertainty is underestimated using the method of residual analysis on the validation set.
- We observe underfitting in the case of two hidden layers, evidenced by a poor linear approximation at either end of the interval. This results in an unreasonable uncertainty evaluation in which the 95% credible intervals for epistemic uncertainty do not include the original function.
- We observe overfitting in the case of four hidden layers (and to a lesser extent for three hidden layers), evidenced by a noisy fit to the training samples. This leads

to poor model fit on the training data in places, which in turn leads to distortions of the credible intervals away from the original function.



(a) 2 **hidden layers.**

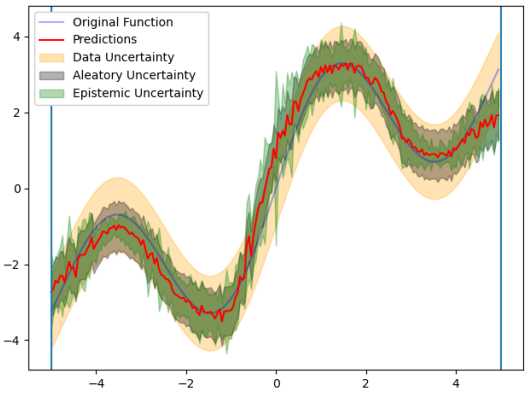
(b) 3 **hidden layers.**

(c) 4 **hidden layers.**

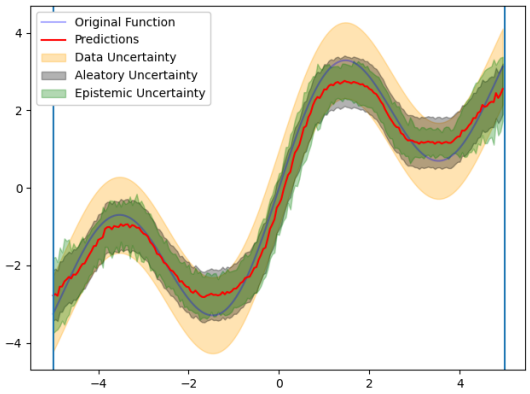
Figure 4 Uncertainty plots for MC-Dropout using different numbers of hidden layers.

We next investigate the impact of the parameters p and λ on the performance of the method. Figure 5(a) shows the effect of decreasing the dropout rate to 0.75, using three hidden layers and keeping all other specifications as in Figure 4. Figure 5(b) shows the effect of decreasing the regularisation parameter λ to 0.05, keeping all other specifications as in Figure 4.

We observe that the epistemic uncertainty is significantly increased for $p = 0.75$, an example of how sensitive the method is to the choice of dropout rate p. Moreover, it is not clear how this parameter should be chosen, revealing a lack of robustness in this respect. The choice of regularisation parameter appears to have a relatively small impact on the behaviour of the algorithm, but increasing the regularisation parameter does give slightly smoother predictions and uncertainty intervals, thereby reducing overfitting.



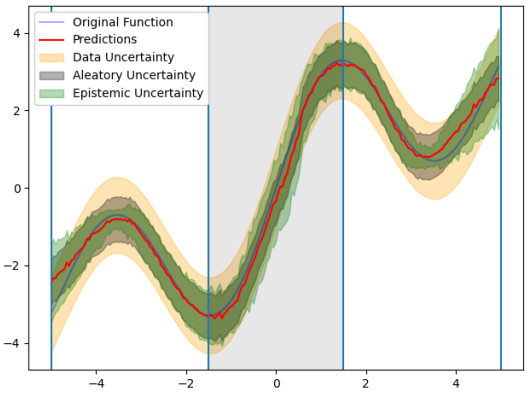
(a) With $p = 0.75$.



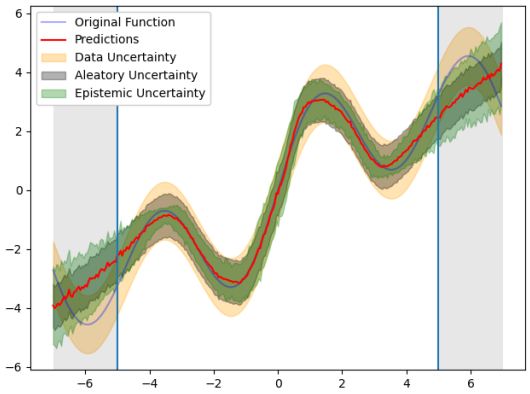
(b) With $\lambda = 0.05$.

Figure 5 Uncertainty plots exploring the effect of changing the parameters p and λ.

Finally we investigate interpolation and extrapolation. Figure 6(a) shows results on the interpolation task with $\nu = 1.5$ and Figure 6(b) shows results on the extrapolation task.



(a) The interpolation task with $\nu = 1.5$.



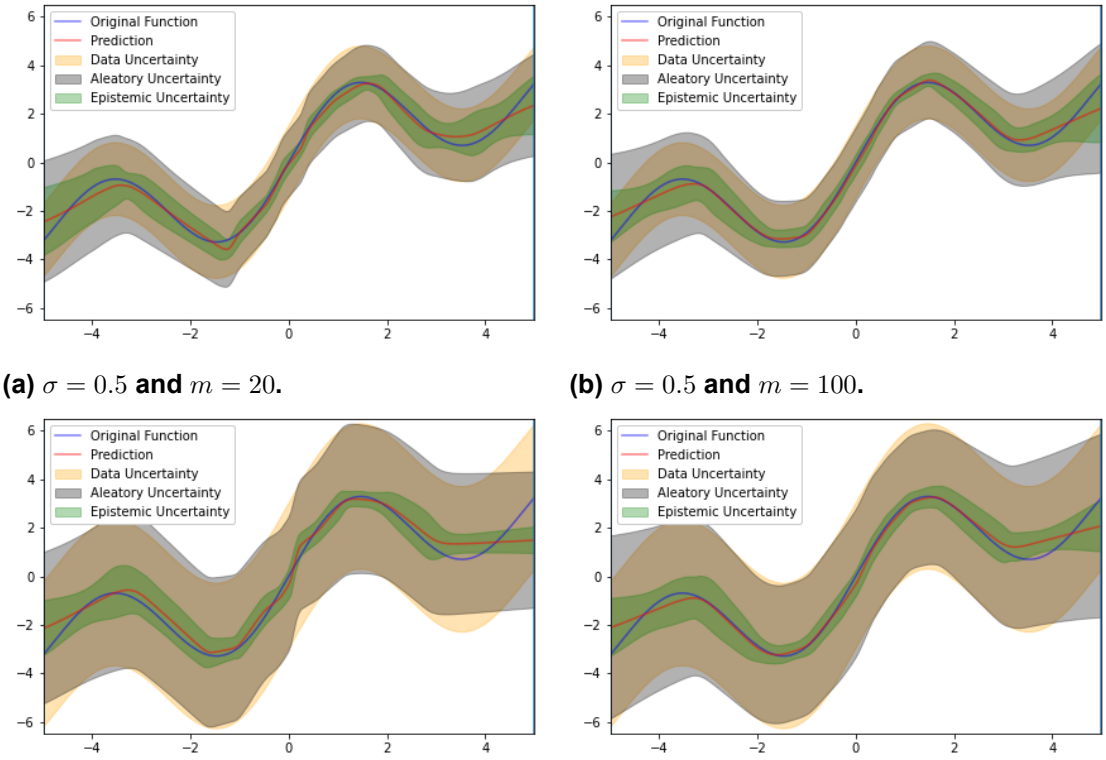
(b) The extrapolation task.

Figure 6 Uncertainty plots exploring the performance of MC-Dropout on interpolation and extrapolation tasks.

We observe an increase in epistemic uncertainty within the intervals where no training data has been sampled, capturing to some extent the intuitive lack of knowledge about the original function in these intervals. We observe that the epistemic uncertainty is only slightly increased for the extrapolation task, and it is much less than for the case of Gaussian Process regression, for example.

3.3 DEEP ENSEMBLES

Throughout this section we use the same test function and experimental methodology as previously described. We fix a network architecture of four hidden layers with 50 nodes in each layer, and with ReLU activation functions throughout. We fix the number of ensembles to be $E = 20$, and we use the uncentered RMS Prop optimisation algorithm (Hinton et al., 2012) with learning rate 0.001, moving average decay rate 0.9 and zero momentum, and we run it with batch size 50 for 1000 epochs in all cases. Figure 7 shows uncertainty plots for Deep Ensembles for two different choices of number of training samples $m = 20$ and $m = 100$ and two different choices for the noise level ($\sigma = 0.5$ and $\sigma = 1$).



(a) $\sigma = 0.5$ **and** $m = 20$**.**

(b) $\sigma = 0.5$ **and** $m = 100$**.**

(c) $\sigma = 1$ **and** $m = 20$**.**

(d) $\sigma = 1$ **and** $m = 100$

Figure 7 Uncertainty plots for different noise levels and numbers of training samples.

We make the following observations.

- By comparing with the underlying data uncertainty, we observe that the aleatoric uncertainty obtained from the double-headed neural networks is accurately captured in all experiments except near the boundaries of the interval. A uniform

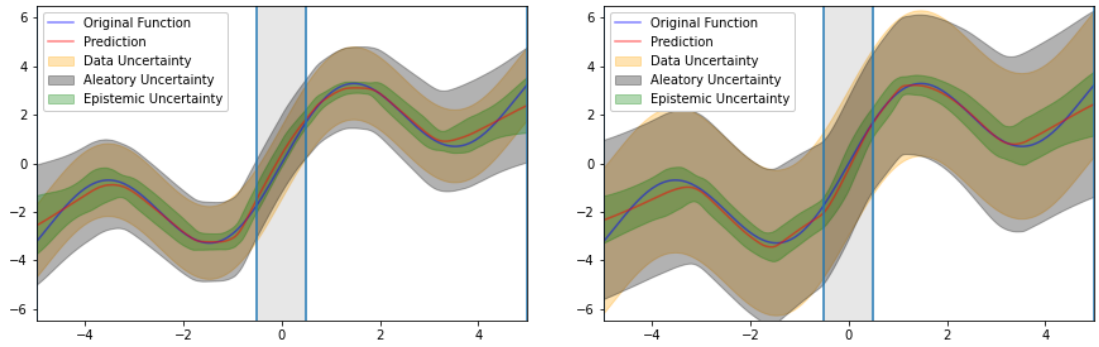
(homoscedastic) uncertainty is detected even though the method allows for more general heteroscedastic behaviour.

- All experiments show poor model fit in places, especially at the boundaries of the interval, suggesting that the network architecture is too constraining to eliminate all model bias. Epistemic uncertainty is largest in these regions, which should be the case. However, for $\sigma = 1$ and $m = 20$, the poor model fit at the upper boundary of the interval leads to an unreasonable 95% credible interval for epistemic uncertainty which does not include the original function.

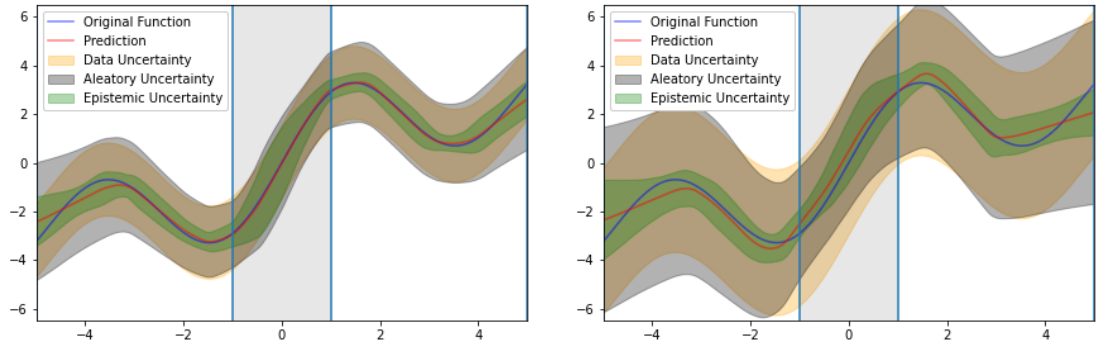
Figure 8 shows results for the interpolation task with $\nu \in \{0.5, 1, 1.5\}$ and for σ equal to both 0.5 and 1. We take $m = 100$ in this experiment.

We make the following observations.

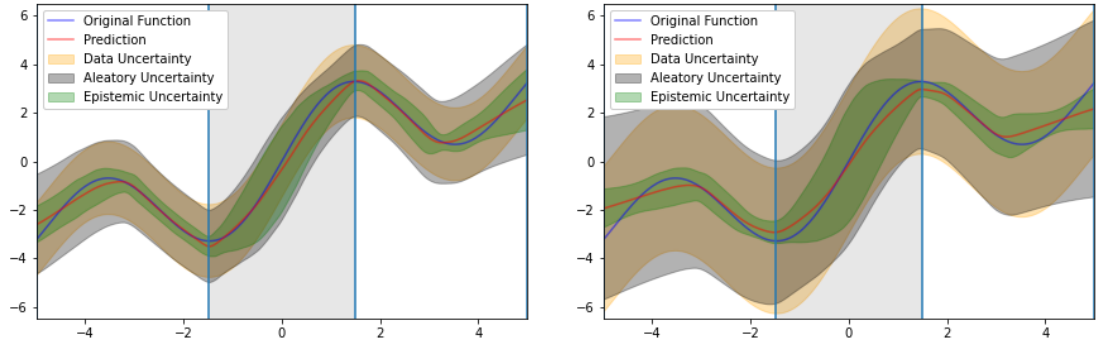
- We observe that the epistemic uncertainty increases within the interval for which training samples are excluded, reflecting the increased uncertainty concerning the model.
- The increase in epistemic uncertainty is especially great as the interval expands to encompass turning points in the original function. This makes sense as it is intuitively easier to interpolate over an interval in which the function is monotonic than over an interval in which the function has a turning point in an unknown location.
- Within the interpolation region, the original function is contained in the 95% credible interval for the epistemic uncertainty in all cases as one might reasonably expect, but only barely for $\nu = 1.5$.



(a) $\sigma = 0.5$ **and** $\nu = 0.5$**.** **(b)** $\sigma = 1$ **and** $\nu = 0.5$**.**



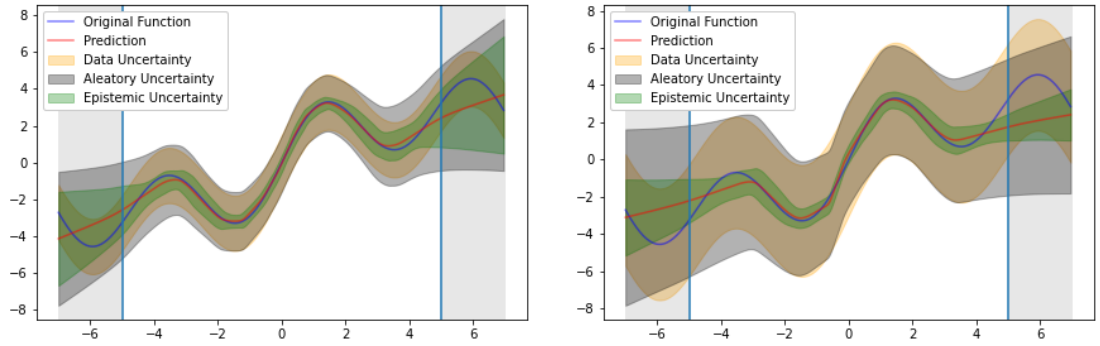
(c) $\sigma = 0.5$ **and** $\nu = 1$**.** **(d)** $\sigma = 1$ **and** $\nu = 1$**.**



(e) $\sigma = 0.5$ **and** $\nu = 1.5$**.** **(f)** $\sigma = 1$ **and** $\nu = 1.5$**.**

Figure 8 Uncertainty plots for Deep Ensembles on interpolation problems with different interval widths ν and different noise levels σ.

Finally, we show results for the extrapolation problem in Figure 9 with two different noise levels ($\sigma = 0.5$ and $\sigma = 1$) and number of training samples $m = 100$.



(a) $\sigma = 0.5$. **(b)** $\sigma = 1$.

Figure 9 Uncertainty plots for Deep Ensembles on the extrapolation problem with different noise levels σ.

We observe that, for both values of σ, epistemic uncertainty increases significantly in the extrapolation interval as expected. However, it is surprising to observe that the epistemic uncertainty is lower for $\sigma = 1$ than for $\sigma = 0.5$, which seems unreasonable. The resulting 95% credible interval for $\sigma = 1$ does not contain the original function, suggesting an unreasonable underestimate of uncertainty as was observed for MC-Dropout. For $\sigma = 0.5$, however, the credible interval expands much more significantly and contains the original function. In this case, then, Deep Ensembles is better able than MC-Dropout to capture epistemic uncertainty in the region of data space which is totally unobserved.

4 METROLOGY REQUIREMENTS CAPTURE

Informed by the experiments in Section 3, we now draw out some general requirements which methods for uncertainty evaluation for ML in metrology must satisfy. These requirements are illustrated in the form of a pyramid in Figure 10. We believe that no current algorithm in the literature meets all seven requirements. In the rest of the section, we describe each of these requirements in detail.

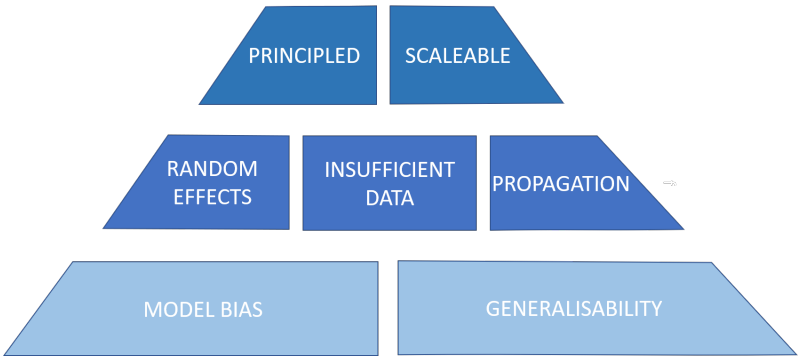


Figure 10 Metrology requirements for machine learning uncertainty evaluation.

4.1 ESTABLISHING CONFIDENCE IN THE MODEL

The first two requirements arise from a simple observation: *no uncertainty evaluation of an ML model can be trusted if the ML model itself cannot be trusted*. One way to express the two aspects of this requirement is that the ML model must neither underfit nor overfit the training data.

4.1.1 Model bias

If a family of models is chosen which is not sufficiently expressive to capture the relationships between the input and output quantities, underfitting will occur. A simple example would be performing linear regression when the relationship between the data is nonlinear.

Another way to describe this effect is as one of unwanted model bias. There is an inescapable tradeoff between bias and variance in ML models, and fitting a model that in any way generalises is only possible by introducing some degree of bias, either by restricting to a certain model class or through regularisation techniques in learning the parameters of the model. However, for uncertainty evaluation there is an imperative to eliminate model bias as far as possible. Underfitting leads to inaccurate uncertainty evaluation because, if relationships in the data have not been fully discovered, the model is ignorant about what it doesn't know.

This principle can be traced in the GUM framework, where care is taken to distinguish between random and systematic effects contributing to measurement error, and where, “It is assumed that the result of a measurement has been corrected for all recognized significant systematic effects and that every effort has been made to identify such effects” (BIPM et al., 2008a, 3.2.4). The problematic nature of undetected systematic effects holds equally in an ML context: it is possible to capture random fluctuations from a ML model, but not systematic departures from it; rather such systematic departures need to be captured by the model in the first place.

One implication of this requirement is that the two challenges of model accuracy (relative to the unknown, true behaviour) and uncertainty evaluation in ML are connected. It is not possible to reliably evaluate the uncertainty of a prediction made using a systematically inaccurate model. One observation made often in recent literature is that expressive models such as deep neural networks are able to largely eliminate model bias (underfitting), capturing all systematic relationships in the data and leaving residual errors consisting only of random fluctuations. It follows that, while more classical approaches to regression may appear to offer more mathematically tractable approaches to uncertainty evaluation, the greater expressivity of modern ML approaches such as deep neural networks gives them a significant advantage when it comes to avoiding model bias.

Model classes should be chosen which are not likely to underfit available measurement data, but we argue that the requirement needs to go further: evidence should be provided that a given model has successfully eliminated systematic errors. One possible approach to obtaining such evidence is to perform statistical testing on the residuals of a model resulting from a held-out validation dataset. There is a large body of work on statistical tests for randomness, independence and Gaussianity (or indeed fit to other distributions); see for example (Yazici and Yolacan, 2007; Jarque and Bera, 1980) .

4.1.2 Generalisability

The requirement for generalisability is the flipside to the previous requirement. While it is important that a model is sufficiently expressive to capture the systematic relationships between the measurement data, it is important that the model doesn’t overfit the training data in a way which doesn’t generalise. The danger of overfitting is well known in ML, and it is also well known that it is a particular danger for very expressive model classes such as deep neural networks. Overfitting occurs either when random effects are included in the model, or when an overly expressive model is trained with insufficient data.

The impact of overfitting on uncertainty evaluation is clear: if a model does not generalise to new measurements for which uncertainty evaluations are required, the results of the evaluations are invalid. Or to put it more positively, the validity of uncertainty evaluation rests upon the generalisability of the ML model.

Algorithmic approaches for avoiding overfitting in ML are well-developed, and there has been a particular focus upon the issue in relation to deep neural networks. These

techniques include early stopping and regularisation (Hastie et al., 2009; Yao et al., 2007; Ioffe and Szegedy, 2015; Srivastava et al., 2014) .

As for underfitting, we would argue that evidence that overfitting will not occur should be provided. The fundamental challenge here is that it is impossible to know in advance how well a model generalises to new data. The best that can be done to address this issue is to make use of validation strategies in which the loss function or prediction accuracy is evaluated on a held back proportion of the training data. Overfitting appears here in the form of superior performance upon the training set compared to the validation set. Another approach is to perform cross-validation in which the investigation is performed upon multiple training/validation splits and the results averaged. If the same level of predictive performance is carried over to the validation set, there is evidence that the model generalises to the validation set, and by implication to any similar unseen data.

4.2 CAPTURING THE DIFFERENT SOURCES OF UNCERTAINTY

Applying an ML model can be viewed as a two-stage process: a training phase followed by an evaluation phase. In the training phase, a measurement model is learned from training data, and then in the evaluation phase this measurement model is applied to new input data. Uncertainties are introduced in both stages of this process. During the training phase, uncertainty concerning the model itself is introduced. This can be of two kinds: uncertainty concerning the correct choice of model family (often referred to as *model uncertainty*) and uncertainty concerning the parameters of the model (often referred to as *parameter uncertainty*). In Section 4.1.1, we argued that, in order to have confidence in a model and its uncertainty evaluations, evidence should be provided that unwanted model bias has been removed. If this requirement is met, model uncertainty is removed and it remains to evaluate parameter uncertainty. Parameter uncertainty arises due to either insufficient coverage of or uncertainty in the values of training data.

Having learned the model, uncertainty also arises in evaluating the model on new input data. In addition to the uncertainty concerning the model's parameters, uncertainty in the input quantities is also propagated through the model. Provided systematic effects have been removed, this uncertainty can be equated with random effects.

The two main sources of uncertainty identified here correspond to a distinction that is often made in recent literature between *aleatoric* and *epistemic* uncertainty (Kendall and Gal, 2017; Hllermeier and Waegeman, 2019) (see also Section 2.1) . Aleatoric uncertainty refers to uncertainty which is intrinsic to the problem, and by which is usually meant random effects. Epistemic uncertainty, on the other hand, refers to uncertainty concerning the model. In this section we argue that capturing both types of uncertainty is crucial from a metrology standpoint.

The point is often made that aleatoric uncertainty is irreducible and that epistemic uncertainty can be reduced by providing more data or specifying a better model. However, we note that random effects in data can often in practice be reduced by providing more measurement data (for example repeated measurements). A further caveat is in order: the distinction between parameter uncertainty (epistemic) and random uncertainty

(aleatoric) is not absolute, but rather dependent upon the choice of input quantities. For example, effects may appear random and irreducible if important input quantities are excluded from the model, but if they are included they may become systematic and hence resolvable by specifying a better model; see (Hüllermeier and Waegeman, 2019, Section 2.3).

4.2.1 Random effects

Random effects must be taken into account, because otherwise incorrect assumptions are being made about the model, namely that there exists a correct model, uncorrupted by random fluctuations, thereby invalidating the uncertainty evaluation carried out upon the model. The requirement here, then, is precisely that aleatoric uncertainty be properly captured. We can expand on this requirement as follows.

1. Unwarranted assumptions cannot be made about the size of random effects. To take one example, Bayesian inference approaches to ML will often capture aleatoric uncertainty in terms of a single hyperparameter (for example the standard deviation of the noise). It is not in general possible in ML applications to make an *a priori* choice for this hyperparameter, but rather it also must be inferred from data. Various hyperparameter optimisation techniques exist for Bayesian inference; see for example (Hastie et al., 2009).

2. Unwarranted assumptions cannot be made about the nature of random effects. Typical assumptions about random effects are that they are *homoscedastic* (identically distributed across the data space) and that they are Gaussian. Neither assumption can be taken for granted. If, on the other hand, random effects are data-dependent, *heteroscedastic* noise assumptions should be made. Meanwhile, if random effects are assumed to arise from a non-Gaussian distribution, different likelihoods or loss functions are needed. There also exist more complex nonlinear effects in data, such as multifractality and other types of nonlinearity, for example, those that can be captured only by Fourier phases of a time series; see for example (Kalisky et al., 2005).

   Confidence in the approach taken is again key, and a natural way to assess noise assumptions is hypothesis testing on the residuals of a held back validation dataset. Various statistical tests exist for testing randomness, Gaussianity and homoscedasticity assumptions; see for example (Yazici and Yolacan, 2007; Jarque and Bera, 1980).

4.2.2 Insufficient data

It is widely acknowledged that it is dangerous to blindly extrapolate an ML model to regions of data space where no data has been seen. Intuitively, given some input data, our uncertainty in the output of a model should be related to the extent to which we have observed similar input data in training. It follows that, if epistemic uncertainty is

not captured and uncertainty evaluation is restricted to estimating noise distributions, our uncertainty evaluations will be overly optimistic and will not capture uncertainty due to insufficient data.

Many methods for uncertainty evaluation in ML focus upon epistemic uncertainty. A common approach is Bayesian inference with respect to the parameters of the model. The Bayesian framework is attractive because lack of knowledge in the absence of data can be captured by a suitably non-informative prior distribution. For complex models such as deep neural networks, much effort has focused upon computationally tractable approximations to Bayesian inference, including variational inference (Graves, 2011; Hernndez-Lobato and Adams, 2015; Kingma, 2017) , Monte Carlo Dropout (Gal and Ghahramani, 2016) and Hamiltonian dynamics (Welling and Teh, 2011) . Non-Bayesian approaches to epistemic uncertainty also exist, for example ensembling (Lakshminarayanan et al., 2017) , density estimation using generative models (Goodfellow et al., 2016) , and likelihood estimation (Hllermeier and Waegeman, 2019) .

4.3 PROPAGATION OF INPUT UNCERTAINTIES

We recall that central to the GUM framework is the concept of a measurement model, and that uncertainties are evaluated by propagating input uncertainties through the model. The approach to propagation espoused by the GUM is the so-called *law of propagation of uncertainty* (BIPM et al., 2008a, 5.1.2,5.2.2), which relies on Taylor approximations. It is valid provided the measurement model can be assumed to be locally linear (or locally low-order polynomial) and providing the output quantity can be assumed to be either Gaussian or t-distributed. An alternative Monte Carlo approach is recommended in Supplement 1 to the GUM (BIPM et al., 2008b; van der Veen and Cox, 2021) , which is useful when the above assumptions are not valid.

Many current approaches to uncertainty evaluation in ML only explicitly assume random uncertainties in the output quantity and do not explicitly model random uncertainties in input quantities. The problem is typically posed as one of standard regression, whereas errors-in-variables regression is commonly used to assess the impact of input uncertainty. For some simple models, such as linear regression models, the errors-in-variables problem is analytically tractable (Klepper and Leamer, 1984; Carroll et al., 2006) . In general, and for more complex models, an attractive approach is to use the approach of GUM Supplement 1 to analyse the propagation of uncertainty on a fixed 'best model'; see for example (Arduino et al., 2017) .

However, the situation is more complicated in an ML context, because the model itself must be learned from data. It follows that simply propagating input uncertainties through a fixed model ignores parameter uncertainty, and what is needed instead is to combine uncertainties in data with parameter uncertainty (which itself also partially arises from uncertainty in training data). A recent proposal which combines Monte Carlo sampling of input quantities with parameter uncertainty is given in (Loquercio et al., 2020) . In a different direction, another recent paper proposes the capturing of input uncertainties using Deming regression Martin and Elster (2021) .

4.4 PRINCIPLED, SCALEABLE UNCERTAINTY EVALUATION

4.4.1 Scaleable approaches

Gaussian Processes (GPs) have become a popular approach to learning data-driven models in the metrology community (Rasmussen, 2003). GPs are attractive for a number of reasons: they are non-parametric models which learn over classes of functions, they are intuitive to understand, they fit into a Bayesian framework, and uncertainty evaluation comes for free. The outputted mean and covariance matrices, which constitute best estimates and uncertainty evaluations respectively, have simple closed-form expressions. Whilst the vast majority of the attention in uncertainty evaluation for ML is focused on deep neural networks, it is worth noting that methods already exist, namely GPs, which automatically satisfy many of the requirements outlined in the previous sections.

There are two reasons why GPs have generally speaking been sidelined in favour of deep neural networks. The first is the greater expressivity of deep neural networks, often leading to improved models with reduced bias. The second is perhaps the most crucial: the improved scaleability of deep neural networks to large data sets. The critical step in GPs is a matrix inversion which in general has complexity which is cubic in the size of the dataset. If the covariance matrices in a GP can be assumed to have special structure, there has been success in improving this complexity considerably, although it is not possible to make such assumptions in all metrology applications. There has also been recent work on combining aspects of GPs and deep neural networks with the aim of achieving high expressibility, scaleability and reliable uncertainty evaluation, for example deep GPs (Damianou and Lawrence, 2013).

Scaleability remains a considerable barrier to the usefulness of GPs in large-scale ML applications, but for regression tasks of more modest size GPs are an attractive method for uncertainty evaluation in metrology.

4.4.2 Principled approaches

The focus of much recent research into uncertainty evaluation for ML has focused upon the challenge of finding approaches which are scaleable to large problems. For example, classical approaches to Bayesian inference when applied to deep neural networks do not scale well, leading to a search for methods which evaluate uncertainty more cheaply.

However, such attempts to reduce computational expense usually mean that principles have to be sacrificed to some extent. We have observed that uncertainty has precise definitions in metrology, and that vague attempts to quantify uncertainty are unlikely to be acceptable in this context. We identify three issues around principled uncertainty evaluation in relation to recently proposed approaches.

1. Existing methods vary according to their notions of uncertainty. Bayesian approaches to uncertainty evaluation aim for a statement of belief about the val-

ues a measurand might take. Many of the recently proposed approaches to uncertainty evaluation are frequentist. Ensembling approaches obtain the empirical distribution of local minimisers with respect to certain subsamplings, perturbations or reorderings (Lakshminarayanan et al., 2017). Quantile regression approaches alter the loss function to learn separate models which estimate conditional quantiles (Romano et al., 2019). Calibration approaches, meanwhile, obtain uncertainties with are empirically accurate over certain subsets of the data (Kuleshov et al., 2018).

Furthermore, these approaches also vary according to the source of uncertainty that they seek to identify. Ensembling, for example, addresses epistemic uncertainty, while quantile regression addresses aleatoric uncertainty.

2. Many current approaches achieve scaleability by making assumptions. Where Bayesian inference is used on deep neural networks, a common form that such assumptions take is the variational inference paradigm, where assumptions are made about the posterior distributions of the parameters (Graves, 2011; Hernández-Lobato and Adams, 2015; Kingma, 2017). Moreover, simplifying (and unrealistic) assumptions are often made about the prior distributions of parameters. For example many variational inference approaches typically place independent priors on all model parameters, which is likely unrealistic in the case of heavily over-parametrised models. It is worth noting that the difficulty in assigning sensible priors is a major hindrance to using Bayesian inference on complex ML models (and more generally).

3. Scaleability is also often achieved by making approximations. For example, the popular MC-Dropout method (Gal and Ghahramani, 2016) performs Bayesian inference only in an approximate sense.

5 EVALUATION OF THE ALGORITHMS

We next highlight to what extent the three methods we have considered in this report meet the metrology requirements outlined in Section 4. As mentioned earlier, the first two requirements are not model-specific, and so we focus on the last five.

5.1 GAUSSIAN PROCESSES

- **Random effects.** The traditional GP approach assumes a homoscedastic noise model (Williams and Rasmussen, 1996). The noise level is given by a hyperparameter, and it is common to estimate this hyperparameter by maximising the marginal likelihood. It is good practice to perform this optimisation on a separate validation dataset to avoid introducing bias. Various optimization procedures exist to perform the optimisation; see (Williams and Rasmussen, 1996). The noise hyperparameter, viewed as a parameter of the model, is itself subject to uncertainty, and an area for future work is to incorporate this uncertainty into the overall assessment of predictive uncertainty. Incorporating heteroscedastic noise assumptions into GP regression has been explored (Goldberg et al., 1997; Le et al., 2005). These approaches can be more computationally expensive.

- **Insufficient data.** The covariance matrix associated with outputs provided by a GP captures both epistemic and aleatoric uncertainty. Due to their effective modelling of proximity in data space through the kernel, GPs are well known to assign high levels of uncertainty far from existing samples.

- **Propagation of uncertainty.** GPs in their standard forms do not take into account uncertainties in the input parameters. A method for extending GPs in this way, involving Monte Carlo sampling, is described in (Rasmussen, 2003, Section 9.5). An approximation to the predictive distribution for errors-in-variables problems and a Monte Carlo approach are both described in (Girard et al., 2003).

- **Scaleability.** Here is where the biggest challenge for GPs lies. Generally speaking, solving a GP problem involves inverting a matrix whose dimensions are equal to the number of data samples N, which is $\mathcal{O}(N^3)$ (Williams and Rasmussen, 1996). There has been work on making GPs more scaleable by exploiting structure in either the data or the kernels, but such structure is not always present in a typical ML problem. An interesting review article summarising state-of-the-art for scaleable GPs is (Liu et al., 2020).

- **Principled approaches.** GPs offer a principled approach to Bayesian inference, without resorting to approximations, and they give rise to transparent belief statements.

5.2 MONTE CARLO DROPOUT

- **Random effects.** MC-Dropout in the form originally proposed only considers parameter uncertainty and does not attempt to capture random effects. The method investigated in this report, using analysis of residuals (see Section 3.2), assumes a homoscedastic noise model. We also found it to be unreliable in our experiments. It was proposed in (Kendall and Gal, 2017) to obtain aleatoric uncertainty in the context of MC-Dropout by using the same kind of 'double-headed' neural network approach as is used in Deep Ensembles (see Section 2.4). This approach is suitable for heteroscedastic noise and, given how well we found this to perform in the context of Deep Ensembles, it appears to be a promising approach.
- **Insufficient data.** MC-Dropout addresses parameter uncertainty, and so should be able in theory to capture uncertainty due to missing data, and this is in agreement with our experiments in Section 3.2. However, we observe that, for the extrapolation problem, uncertainties are much smaller than for GPs in the extrapolation region, suggesting an underestimate.
- **Propagation of uncertainty.** Like other standard approaches to ML uncertainty evaluation, MC-Dropout makes the idealistic assumption that all inputs to the measurement model have zero uncertainty. A recent paper (Loquercio et al., 2020) proposes an approach for combining propagation of input uncertainties with parameter uncertainty evaluation using the MC-Dropout method. The approach could also be extended to other methods for parameter uncertainty evaluation.
- **Scaleability.** MC-Dropout involves training an ensemble of deep neural networks, and so in this sense the method is scaleable and computational time grows linearly with the ensemble size.
- **Principled approaches.** It is here that MC-Dropout has the biggest challenges. While the method is motivated from Bayesian principles, our finding is that the method does not reliably output a posterior distribution, due to numerous approximations and lack of transparency about the prior distribution. The method also suffers from the more general problem that placing an independent prior on the weights of an overparametrised model (such as a neural network) runs counter to intuition.

5.3 DEEP ENSEMBLES

- **Random effects.** Deep Ensembles are designed to capture random effects as well as parameter uncertainty. We found the 'double-headed neural network' method to be accurate in estimating known aleatoric uncertainty on our test problem, and the method is also suitable for heteroscedastic noise.
- **Insufficient data.** Like MC-Dropout, Deep Ensembles addresses parameter uncertainty, and so should be able in theory to capture uncertainty due to missing

data, and this is in agreement with our experiments in Section 3.3. However, we observe that, for the extrapolation problem, uncertainties are much smaller than for GPs in the extrapolation region, suggesting an underestimate.

- **Propagation of uncertainty.** We are not aware of any extensions of Deep Ensembles to take into account uncertainties in input data.
- **Scaleability.** Like MC-Dropout, the Deep Ensembles method involves training an ensemble of deep neural networks, and so in this sense the method is scaleable and computational time grows linearly with the ensemble size.
- **Principled approaches.** One downside of the Deep Ensembles method is the unprincipled nature of the parameter uncertainty evaluation. What is obtained is the distribution of the optimizers of an ensemble of neural networks which vary through the injection of randomness in adversarial training and initializations. To equate this distribution with the posterior distribution of the parameters has no firm justification as far as we can see. Closely related to Deep Ensembles is Bayesian ensemble learning, which offers a way to root ensembling methods for uncertainty evaluation in a Bayesian framework; see for example (Fersini et al., 2014).

6 FUTURE WORK

We next outline some of the challenges that need to be addressed by the metrology community around uncertainty evaluation for ML.

6.1 METRICS FOR ASSESSING OVERFITTING

Techniques for assessing and detecting overfitting are standard int the ML community, but there is a need for standardisation in the metrology community around good practice in providing evidence that overfitting has not occurred. An important aspect is likely to be establishing generic, easy to use metrics which capture the extent to which overfitting has occurred in validation tests.

6.2 METRICS FOR ASSESSING UNDERFITTING

We highlighted earlier that assessing underfitting involves probing the residuals resulting from ML models for systematic errors, and that statistical tests for independence, randomness and Gaussianity are a natural approach to this task. There is a need for standardisation in the metrology community around good practice in assessing the extent of underfitting using validation tests, which again will involve identifying suitable metrics.

6.3 INCORPORATING HETEROSCEDASTIC NOISE ASSUMPTIONS

We noted in Section 4.2.1 that the homoscedastic noise models often used in conjunction with ML models may not be appropriate in many metrology applications. There is a need for established good practice in noise modelling and assessment in the metrology community so that resulting uncertainty evaluations rest on realistic and tested assumptions. Assumptions concerning random effects also impact training algorithms through likelihood functions and corresponding loss functions. Rather than blindly using likelihood models and loss functions that are implicitly making simplistic assumptions concerning random effects, good practice concerning the selection of appropriate likelihoods and loss functions should be identified.

6.4 PRINCIPLED AND SCALEABLE UNCERTAINTY EVALUATION

Perhaps the greatest challenge confronting the metrology community is the need to identify methods for uncertainty evaluation in ML which are both principled and scaleable. There is a spectrum of methods, from GPs on one hand which offer principled uncertainty evaluation but which do not generally scale well, to deep neural networks on the other hand which scale well but which are built on looser principles. We suggest at least three directions of interest for the metrology community.

1. A clear understanding of the precise underlying definitions of uncertainty and the extent to which assumptions and approximations are made in popular approaches to uncertainty evaluation for deep neural networks, such as MC-Dropout (Gal and Ghahramani, 2016), SWAG (Maddox et al., 2019), Stochastic Langevin Dynamics (Welling and Teh, 2011), Deep Ensembles (Lakshminarayanan et al., 2017) and Quantile Regression (Romano et al., 2019). This is a rapidly moving research area and we can expect new methods to appear, and the metrology community needs to be agile to assess the range of approaches that are proposed. We note a recent review of uncertainty evaluation methods for deep learning (Abdar et al., 2021).

2. Hybrid methods (Damianou and Lawrence, 2013; Di Martino et al., 2018) in which the ability of deep neural networks to identify complex relationships in data is combined with more principled approaches to uncertainty evaluation. Methods which combine aspects of GPs and deep neural networks are promising in this respect, and the metrology community should become familiar with these methods, and gain an understanding of the definitions, assumptions and approximations underlying them. Another family of methods decouple the training process by using deep neural networks for feature extraction and then fitting a regression model in the new feature space using a model more amenable to uncertainty evaluation. The advantages and challenges associated with the decoupling approach also need to be understood.

3. The metrology community needs to become familiar with the various approaches for making GPs more scaleable, such as data reduction using coresets and sparse and local kernel approximations. Tradeoffs between expressibility and scaleability need to be understood for these methods.

6.5 COMBINING MODEL UNCERTAINTIES WITH PROPAGATED UNCERTAINTIES

Methods for bringing together the two approaches of assessing model uncertainty and propagating input uncertainties for ML models are only just starting to appear. Given the importance of the propagation of uncertainty in metrology, this represents a challenge of great interest. The fundamental challenge is one of combining the various uncertainties, which requires to go beyond the Monte Carlo approach for fixed models described in GUM Supplement 1 (BIPM et al., 2008b) . One intriguing aspect of the problem is whether it is realistic in the context of data-driven models to assume knowledge of all random effects *a priori*. If this is not realistic, methods need to take into account input uncertainties, whilst at the same time not being tied to assumptions that they account for all random effects.

7 CONCLUSION

Uncertainty evaluation is crucial in metrology. The adoption of ML in metrology applications is crucially constrained by the capability to perform reliable uncertainty evaluation for ML models, usually in the context of large-scale data. We have presented requirements that need to be satisfied to enable such reliability. Other important considerations could also be mentioned beyond those that we identified as requirements in this report. For example, the interpretability of models, whilst not a requirement for reliable uncertainty evaluation, can offer assistance. The insight gained from interpretation of a model can provide qualitative confidence in a model and can help to identify overfitting. Finally, the requirements identified highlight several open challenges of interest and represent a roadmap for future work in the area.

ACKNOWLEDGEMENT

This work was supported by the UK Government's Department for Business, Energy and Industrial Strategy (BEIS). The authors would like to thank Valerie Livina and Jenny Venton (both NPL) for reviewing the reports and providing helpful feedback.

ACRONYMS

AI	Artificial Intelligence
BEIS	Business, Energy and Industrial Strategy
GP	Gaussian Process
GUM	Guide to the Expression of Uncertainty in Measurement
KL	Kullback-Leibler
MAP	Maximum *a posteriori*
MC	Monte Carlo
MCMC	Markov Chain Monte Carlo
ML	Machine Learning
RBF	Radial Basis Function
ReLU	Rectified Linear Unit
VI	Variational Inference

REFERENCES

BIPM, IEC, IFCC, ILAC, ISO, IUPAC, IUPAP, and OIML. International vocabulary of metrology – Basic and general concepts and associated terms (vim). *Joint Committee for Guides in Metrology (JCGM)*. 3rd edition (2008 version with minor corrections).

Luciano Floridi. Establishing the rules for building trustworthy AI. *Nature Machine Intelligence*, 1(6):261–262, 2019.

BIPM, IEC, IFCC, ILAC, ISO, IUPAC, IUPAP, and OIML. Evaluation of measurement data – Guide to the expression of uncertainty in measurement (GUM 1995 with minor corrections). *Joint Committee for Guides in Metrology (JCGM)*, 100, 2008a.

Peter Harris, Kostas Sotirakopoulos, Stephen Robinson, Lian Wang, and Valerie Livina. A statistical method for the evaluation of long term trends in underwater noise measurements. *The Journal of the Acoustical Society of America*, 145(1):228–242, 2019.

Steven Robinson, Peter Harris, Sei-Him Cheong, Lian Wang, and Valerie Livina. Study of the impact of the covid-19 pandemic on levels of deep-ocean acoustic noise. preprint, 2021.

Spencer A Thomas, Alan M Race, Rory T Steven, Ian S Gilmore, and Josephine Bunch. Dimensionality reduction of mass spectrometry imaging data using autoencoders. In *2016 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 1–7, 2016.

David J Lary, Gebreab K Zewdie, Xun Liu, Daji Wu, Estelle Levetin, Rebecca J Allee, Nabin Malakar, Annette Walker, Hamse Mussa, Antonio Mannino, et al. Machine learning applications for earth observation. *Earth observation open science and innovation*, 165, 2018.

BIPM, IEC, IFCC, ILAC, ISO, IUPAC, IUPAP, and OIML. Evaluation of measurement data – Supplement 1 to the 'Guide to the expression of uncertainty in measurement'. *Joint Committee for Guides in Metrology (JCGM)*, 101, 2008b.

BIPM, IEC, IFCC, ILAC, ISO, IUPAC, IUPAP, and OIML. Evaluation of measurement data – Supplement 2 to the 'Guide to the expression of uncertainty in measurement' – Extension to any number of output quantities. *Joint Committee for Guides in Metrology (JCGM)*, 102, 2011.

Clemens Elster and Blaza Toman. Bayesian uncertainty analysis for a regression model versus application of GUM Supplement 1 to the least-squares estimate. *Metrologia*, 48(5), 2011.

Walter Bich, Maurice G Cox, Ren Dybkaer, Clemens Elster, W Tyler Estler, Brynn Hibbert, Hidetaka Imai, Willem Kool, Carine Michotte, Lars Nielsen, et al. Revision of the 'Guide to the expression of uncertainty in measurement'. *Metrologia*, 49(6), 2012.

Walter Bich. Revision of the 'Guide to the expression of uncertainty in measurement'. why and how. *Metrologia*, 51(4), 2014.

Clemens Elster. Calculation of uncertainty in the presence of prior knowledge. *Metrologia*, 44(2):111, 2007.

Katy Klauenberg, Gerd Wübbeler, Bodo Mickan, Peter Harris, and Clemens Elster. A tutorial on Bayesian normal linear regression. *Metrologia*, 52(6):878, 2015.

AB Forbes and JA Sousa. The GUM, Bayesian inference and the observation and measurement equations. *Measurement*, 44(8):1422–1435, 2011.

Christopher KI Williams and Carl Edward Rasmussen. *Gaussian Processes for regression*. MIT, 1996.

Carl Edward Rasmussen. Gaussian processes in machine learning. In *Summer school on machine learning*, pages 63–71. Springer, 2003.

Alistair Forbes. Traceable measurements using sensor networks. *Transactions on Machine Learning and Data Mining*, 8(2):77–100, 2015.

Andrew Gelman, John B Carlin, Hal S Stern, David B Dunson, Aki Vehtari, and Donald B Rubin. *Bayesian data analysis*. CRC press, 2013.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

Yarin Gal. Uncertainty in deep learning. *University of Cambridge*, 2016.

Yarin Gal and Zoubin Ghahramani. Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059, 2016.

Alex Graves. Practical variational inference for neural networks. In *Advances in neural information processing systems*, pages 2348–2356, 2011.

Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural networks. *arXiv:1505.05424*, 2015.

Diederick Pieter Kingma. Variational inference & deep learning: A new synthesis. intelligent sensory information systems (ivi, fnwi)(2017), 2017.

Andreas Damianou and Neil D Lawrence. Deep Gaussian Processes. In *Artificial intelligence and statistics*, pages 207–215, 2013.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in neural information processing systems*, pages 6402–6413, 2017.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Simon Bachstein. *Uncertainty Quantification in Deep Learning*. PhD thesis, MA thesis, 2019.

Alex Kendall and Yarin Gal. What uncertainties do we need in Bayesian deep learning for computer vision? In *Advances in neural information processing systems*, pages 5580–5590, 2017.

Eyke Hüllermeier and Willem Waegeman. Aleatoric and epistemic uncertainty in machine learning: An introduction to concepts and methods. *arXiv preprint arXiv:1910.09457*, 2019.

Geoffrey Hinton, Nitish Srivastava, and Kevin Swersky. Neural networks for machine learning lecture 6a overview of mini-batch gradient descent. *Cited on*, 14(8), 2012.

Berna Yazici and Senay Yolacan. A comparison of various tests of normality. *Journal of Statistical Computation and Simulation*, 77(2):175–183, 2007.

Carlos M Jarque and Anil K Bera. Efficient tests for normality, homoscedasticity and serial independence of regression residuals. *Economics letters*, 6(3):255–259, 1980.

Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The elements of statistical learning: data mining, inference, and prediction*. Springer Science & Business Media, 2009.

Yuan Yao, Lorenzo Rosasco, and Andrea Caponnetto. On early stopping in gradient descent learning. *Constructive Approximation*, 26(2):289–315, 2007.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning*, pages 448–456, 2015.

Tomer Kalisky, Yosef Ashkenazy, and Shlomo Havlin. Volatility of linear and nonlinear time series. *Physical Review E*, 72(1):011913, 2005.

José Miguel Hernández-Lobato and Ryan Adams. Probabilistic backpropagation for scalable learning of Bayesian neural networks. In *International Conference on Machine Learning*, pages 1861–1869, 2015.

Max Welling and Yee W Teh. Bayesian learning via stochastic gradient Langevin dynamics. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 681–688, 2011.

Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. *Deep learning*. MIT press Cambridge, 2016.

Adriaan MH van der Veen and Maurice G Cox. Getting started with uncertainty evaluation using the monte carlo method in r. *Accreditation and Quality Assurance*, pages 1–13, 2021.

Steven Klepper and Edward E Leamer. Consistent sets of estimates for regressions with errors in all variables. *Econometrica: Journal of the Econometric Society*, pages 163–183, 1984.

Raymond J Carroll, David Ruppert, Leonard A Stefanski, and Ciprian M Crainiceanu. *Measurement error in nonlinear models: a modern perspective*. CRC press, 2006.

Alessandro Arduino, Mario Chiampi, Francesca Pennecchi, Luca Zilberti, and Oriano Bottauscio. Monte Carlo method for uncertainty propagation in magnetic resonance-based electric properties tomography. *IEEE Transactions on Magnetics*, 53(11):1–4, 2017.

Antonio Loquercio, Mattia Segu, and Davide Scaramuzza. A general framework for uncertainty estimation in deep learning. *IEEE Robotics and Automation Letters*, 5(2): 3153–3160, 2020.

Jörg Martin and Clemens Elster. Errors-in-variables for deep learning: rethinking aleatoric uncertainty. *arXiv preprint arXiv:2105.09095*, 2021.

Yaniv Romano, Evan Patterson, and Emmanuel J Candès. Conformalized quantile regression. *arXiv preprint arXiv:1905.03222*, 2019.

Volodymyr Kuleshov, Nathan Fenner, and Stefano Ermon. Accurate uncertainties for deep learning using calibrated regression. In *International Conference on Machine Learning*, pages 2796–2804, 2018.

Paul W Goldberg, Christopher KI Williams, and Christopher M Bishop. Regression with input-dependent noise: A Gaussian process treatment. *Advances in neural information processing systems*, 10:493–499, 1997.

Quoc V Le, Alex J Smola, and Stéphane Canu. Heteroscedastic Gaussian process regression. In *Proceedings of the 22nd international conference on Machine learning*, pages 489–496, 2005.

Agathe Girard, Carl Edward Rasmussen, Joaquin Quinonero-Candela, and Roderick Murray-Smith. *Gaussian process priors with uncertain inputs? application to multiple-step ahead time series forecasting*. MIT Press, 2003.

Haitao Liu, Yew-Soon Ong, Xiaobo Shen, and Jianfei Cai. When Gaussian Process meets big data: A review of scalable gps. *IEEE Transactions on Neural Networks and Learning Systems*, 31(11):4405–4423, 2020.

Elisabetta Fersini, Enza Messina, and Federico Alberto Pozzi. Sentiment analysis: Bayesian ensemble learning. *Decision support systems*, 68:26–38, 2014.

Wesley Maddox, Timur Garipov, Pavel Izmailov, Dmitry Vetrov, and Andrew Gordon Wilson. A simple baseline for Bayesian uncertainty in deep learning. *arXiv preprint arXiv:1902.02476*, 2019.

Moloud Abdar, Farhad Pourpanah, Sadiq Hussain, Dana Rezazadegan, Li Liu, Mohammad Ghavamzadeh, Paul Fieguth, Xiaochun Cao, Abbas Khosravi, U Rajendra Acharya, et al. A review of uncertainty quantification in deep learning: Techniques, applications and challenges. *Information Fusion*, 2021.

Alessandro Di Martino, Erik Bodin, Carl Henrik Ek, and Neill DF Campbell. Gaussian process deep belief networks: A smooth generative model of shape with uncertainty propagation. In *Asian Conference on Computer Vision*, pages 3–20, 2018.