Inclusion of data uncertainty in machine learning and its application in geodetic data science, with case studies for the prediction of Earth orientation parameters and GNSS station coordinate time series

Author(s):
Kiani Shahvandi, Mostafa; Soja, Benedikt

Publication date:
2022-08-01

Permanent link:
https://doi.org/https://doi.org/10.3929/ethz-b-000550916

Originally published in:
Advances in Space Research 70(3), https://doi.org/10.1016/j.asr.2022.05.042

Available online at www.sciencedirect.com

ScienceDirect

Advances in Space Research 70 (2022) 563–575

ADVANCES IN SPACE RESEARCH (a COSPAR publication)

www.elsevier.com/locate/asr

Inclusion of data uncertainty in machine learning and its application in geodetic data science, with case studies for the prediction of Earth orientation parameters and GNSS station coordinate time series

M. Kiani Shahvandi *, Benedikt Soja

Institute of Geodesy and Photogrammetry, ETH Zurich, Robert-Gnehm-Weg 15, 8093 Zurich, Switzerland

Received 25 November 2021; received in revised form 25 November 2021; accepted 25 November 2021
Available online 28 May 2022

Abstract

Data uncertainty plays an important role in the field of geodesy. Even though deep learning is becoming increasingly important for geodetic applications due to its high accuracy, it typically does not consider the data uncertainty. As we demonstrate in this study, we propose to include the uncertainty of data in deep neural network architectures to achieve a better generalization. This is advantageous for big data applications as well as for small datasets. Inspired by weighted and total least squares, we formulate the problem for both input and target uncertainties, and combine it with the Bayesian learning method. This results in a new form of the loss function in machine learning. As an alternative approach, we consider data uncertainties by including them as additional features. For comparison purposes, we use models without the consideration of data uncertainty as a benchmark. To show the efficacy of the proposed method, we apply it to the prediction of Earth Orientation Parameters (EOPs, namely polar motion, dUT1, and LOD) and Global Navigation Satellite System (GNSS) station coordinate time series. We demonstrate that the least-squares-inspired method outperforms both the benchmark and the feature-inspired method for both the studies. In the EOPs study, the improvement can be more than 50% in the study interval. In the study of GNSS station coordinate time series, which is presented for 1000 stations across the globe, the improvement on an average basis is around 12%. The results demonstrate the advantage of using uncertainty information in the machine learning algorithms, when applied to geodetic time series.

Keywords: Machine learning; Bayesian learning; Data uncertainty; Weighted and total least squares; Earth orientation parameters; GNSS station coordinate time series

1. Introduction

Mathematical models in the field of geodesy strongly rely on the uncertainty of data (Koch, 1999). The most important estimation method in geodesy, Least Squares (LS), does in fact use the observation uncertainty as part of its stochastic model (Teunissen, 2000). The inclusion of observation uncertainty typically leads to a more reliable estimation, facilitates outlier removal, and allows for a more rigorous prior and posterior quality assessment (Amiri-Simkooei et al., 2007). Variants of LS, including Total Least Squares (TLS), also use the concept of data uncertainty (Amiri-Simkooei and Jazaeri, 2012; Amiri-Simkooei et al., 2016).

In geodesy, the least-squares based methods are extensively used for regression and prediction problems, including parameter estimation (Odijk et al., 2016) and time series analysis and prediction (Li et al., 2000). Over the last few decades, machine learning approaches have been estab-

* Corresponding author.
E-mail addresses: mkiani@ethz.ch (M. Kiani Shahvandi), soja@ethz.ch (B. Soja).

https://doi.org/10.1016/j.asr.2022.05.042
0273-1177/

lished as powerful estimation methods as well. They can model arbitrarily complex functions, be used for both classification and regression problems, and make inferences and generalizations (LeCun et al., 2015; Goodfellow et al., 2016). In particular, deep learning, which is a subset of machine learning using several layers of representation, is of special interest because of its strong performance mostly on big data problems.

Recently, there have been studies on the application of machine learning algorithms in the field of geodesy. Some examples include vertical displacement assessment (MrwczyÅska, 2017), fault rupture analysis (Ren et al., 2020), precipitable water vapor analysis using generalized regression neural networks (Zhang and Yao, 2021), GNSS station position time series prediction using modified Transformers (Kiani Shahvandi and Soja, 2021), prediction of length of day using a new deep learning algorithm (Kiani Shahvandi and Soja, 2022), and Ionospheric investigations (Navarro et al., 2021). The reader is referred to Reiterer et al. (2010) and Butt et al. (2021) for further resources on the applications of machine learning in geodesy.

In order to enhance the performance of deep learning approaches, both in small and large datasets, one can combine the concept of uncertainty and weights in LS with deep learning methods. The present paper is focused on this problem with the main purpose to introduce a new form of the so-called loss function in deep learning.

There have been some studies on the problem of inclusion of data uncertainty, including Kendall and Gal (2017) and Gast and Roth (2018). However, their approaches are not LS-inspired in the sense of using quantified data uncertainty. In Kendall and Gal (2017), the important concepts of aleatory and epistemic uncertainties (the former refers to the variability of the values due to the intrinsic properties of the time series; the latter refers to the finite precision of the instrument with which phenomenon is observed (Sullivan, 2015)) are used to present a model in which both types of uncertainties are treated simultaneously. However, the aleatory uncertainties are derived based on the data and by not knowing them or having an estimate of them beforehand. This has the drawback that large datasets are needed to estimate these uncertainties well. In Gast and Roth (2018), the so-called probabilistic output layer (replacing the point estimation with a probability distribution) and density filtering methods (dealing with activation uncertainties) are used. These are used to iteratively propagate errors through layers and activation functions. As the result, these methods are conceptually not LS-inspired.

In this paper, our focus is on the development of a new framework for the prediction of geodetic time series using neural networks. Our study makes the following contributions:

1. Formulation of an LS-inspired learning method in three variations, namely only the target uncertainty, input and target uncertainty, and input and target uncertainty in combination with Bayesian learning.

2. Consideration of observation uncertainty as additional information contributing to the learning procedure and combining it with the Bayesian learning approach.

To demonstrate the potential application of the method, we analyze prediction problems in the field of geodesy. Prediction of the Global Navigation Satellite System (GNSS) station coordinate and Earth Orientation Parameters (EOP) time series is important in various geodetic applications and investigated in several studies (Kalarus et al., 2010; Piccolomini et al., 2019; Modiri et al., 2020; Beutler et al., 2020; Kiani Shahvandi and Soja, 2021; Kiani Shahvandi and Soja, 2022). In this regard, we use GNSS, polar motion, Length of Day (LOD), and dUT1 time series to demonstrate the application of the new method in geodesy.

The rest of this paper is organized as follows. Section 2 discusses the background. Section 3 includes derivation of the method. Section 4 provides description of data used in the studies, while Section 5 presents the numerical results and their analyses.

2. Background of the methodology

In this section, we present the background and prerequisites of the proposed method.

2.1. Overview of weighted and total least squares

The Weighted Least Squares (WLS) approach (Koch, 1999) for linear(ized) systems of equations is defined based on the so-called design matrix A, a series of adjustable unknowns, which we refer to as W (corresponding to the parameters of a neural network), as well as the observations (or targets) Y and their corresponding covariance matrix Q (which represents the uncertainties in and correlations between the values of Y). The system of equations is defined by

$$Y + v = AW \tag{1}$$

in which v denotes the vector of differences between estimated and observed values. The vector W of unknowns is said to be estimated through WLS when it is the solution to the minimization problem

$$v^T \Sigma v \longrightarrow min \tag{2}$$

where Σ is the matrix defined as $\Sigma = Q^{-1}$ and called the precision matrix.

Remark 1. Note that the analysis of the so-called a priori and a posteriori variance factors (also referred to as the standard deviation of unit weight (Kermarrec and Schn, 2017)) is not done in the case of machine learning models, since these algorithms are mostly unbiased (Kim et al., 2019; Jiang and Nachum, 2020), and often have higher number of unknown weights W than number of observa-

tions (negative degree of freedom, which is not possible in LS, (Zhang et al., 2017)).

In the formulation of Eq. (1), only the effect of uncertainties in observations (in the form of covariance matrix Q) is considered. If, however, uncertainties also exist in the partial derivatives within the design matrix A, TLS should be used. According to Amiri-Simkooei and Jazaeri (2012), the formulation of the TLS problem, which is an extension of Eq. (1), is

$$Y + v = (A + \delta A)W \tag{3}$$

in which δA represents the uncertainty of values in the design matrix A,. The goal of TLS is then to simultaneously minimize the sum-squared-error of both v and δA and for this reason the following minimization problem is defined

$$v^T \Sigma v + vec(\delta A)^T \Sigma_A vec(\delta A) \longrightarrow min \tag{4}$$

where $vec(\delta A)$ represents the vectorized form of δA (Amiri-Simkooei et al., 2016), and Σ_A is the weight matrix of A, defined based on the uncertainties in the values of the arguments, which would correspond to the input in the case of neural networks (in the form of the input's covariance matrix).

Both WLS and TLS are powerful estimators and have applications in many problems in geodesy, including GNSS positioning (Truong and Ta, 2013) and network analysis (Fazilova, 2017).

2.2. Types of uncertainties to use in neural networks

There are two types of uncertainties that we use in the derivation of our method: aleatory and epistemic. These are defined in the following.

Definition 1. The aleatory uncertainty is the uncertainty that is associated with the nature of the phenomenon being observed. An instance would be the initial distribution that we have for the weights W of the deep learning function or the inherent observational variability.

Definition 2. The epistemic uncertainty is associated with the finite precision of the instrument used to gather data or the method used to derive the parameters. Examples include the uncertainties in the observations in Eq. (7), due to the use of different measuring devices, or the accuracy considered for the initial distribution.

2.3. Overview of the regression problem in machine learning

As mentioned in the Introduction, machine learning methods can be seen as an alternative estimation approach. Here, a brief overview of the regression problem in machine learning and, in particular, deep learning approaches is given.

Definition 3. Let the input to the machine learning models be denoted by $X = \{x_1, \dots, x_n\}$, where $x_i \in \mathbb{R}^{c_1 \times \dots \times c_p}$, $i = 1, \dots, n$. Also, consider the corresponding target values as $Y = \{y_1, \dots, y_n\}$, where $y_i \in \mathbb{R}^{d_1 \times \dots \times d_q}$. Here, the numbers $\{c_1, \dots, c_p\}$ and $\{d_1, \dots, d_q\}$ represent the dimensionalities of x_i and y_i, respectively. The machine learning function, denoted by f, maps the inputs to the targets through a series of adjustable weights W (which are considered as unknowns that must be derived from data), as follows

$$Y = f(X, W) \tag{5}$$

Definition 4. The uncertainties and correlations associated with input and their corresponding target values are stacked into matrices denoted respectively by Q_X and Q_Y. The inverse of these matrices are called the precision matrices and will be denoted by Σ_X and Σ_Y, thus

$$\begin{aligned} \Sigma_X &= Q_X^{-1} \\ \Sigma_Y &= Q_Y^{-1} \end{aligned} \tag{6}$$

Remark 2. Note that the matrices Σ_X and Σ_Y are tantamount to the weight matrix Σ in Eq. (2) in the case of WLS. This means that they are derived from the inverse of the covariance matrix of input and targets, respectively. Note also these matrices are often diagonal and consist of the inverse of the squared values of uncertainties typically found in data files for the observed values X and Y, as is the case with the applications presented in this paper. In case data are multidimensional the precision matrices of different features are stacked together, as mentioned above. If there are p different input features with $\Sigma_{X,1}, \dots \Sigma_{X,p}$ precision matrices, and q different output features with $\Sigma_{Y,1}, \dots \Sigma_{Y,q}$ precision matrices, the total precision matrices used for computations may be represented as follows

$$\begin{aligned} \Sigma_X &= \begin{bmatrix} \Sigma_{X,1} & \cdots & \cdots \\ \vdots & \ddots & \vdots \\ \cdots & \cdots & \Sigma_{X,p} \end{bmatrix} \\ \Sigma_Y &= \begin{bmatrix} \Sigma_{Y,1} & \cdots & \cdots \\ \vdots & \ddots & \vdots \\ \cdots & \cdots & \Sigma_{Y,q} \end{bmatrix} \end{aligned} \tag{7}$$

In Eq. (7), the off-diagonal elements can be filled with the covariance matrices. However, one usually does not have these matrices and therefore in most cases a diagonal matrix is used.

Note that the process during which the weights W in Eq. (5) are determined is called training of the algorithm.

Definition 5. The so-called loss function, denoted by ℓ, is a measure of how well the machine learning function f maps the input to the output, which is used in the training of

function f. Most frequently used loss functions are based on the L_k norm, in which case ℓ takes the following form

$$\ell(f, Y) = ||Y - f(X, W)||^k \tag{8}$$

Remark 3. The previously discussed minimization problems in Eq. (2) and Eq. (4) use in fact the squared-error loss, i.e. L_2. In the case of $k = 1$, the loss is based on the absolute error associated with linear programming (Dantzig, 2002). We use the L_2 loss in the derivation of the following formulas.

Definition 6. The Bayesian estimation approach (Bishop, 2009), based on the conditional distributions and prior distribution $p(W)$ for unknown parameters is represented by

$$p(W|X, Y) \propto p(Y|X, W)p(W) \tag{9}$$

The Bayesian approach is based on using prior knowledge on the estimable parameters. This includes the distribution of the parameters, which has a direct impact on the formulation of the method, since different distributions have different probability density functions $p(W)$. This approach has proven to be effective for many applications resulting in more accurate predictions (Neal, 1996; Wenzel et al., 2020).

The relation in Eq. (9) will be used for the case of combining the new method with the Bayesian approach.

2.4. Overview of long short-term memory in machine learning

Here, an example of the machine learning function in Eq. (5) is given. One of the most widely used machine learning algorithms is the so-called Long Short-Term Memory (LSTM; introduced in Hochreiter and Schmidhuber (1997) and expanded upon in Gers et al. (1999) and Greff et al. (2017)). LSTM is a variation of Recurrent Neural Networks (RNNs, which were introduced in Rumelhart et al. (1986)), which are used for modelling sequential data. RNNs can model the long-term dependencies in data (Benitez et al., 2021; Torres et al., 2021). However, they suffer from the so-called vanishing gradient problem, in which the gradient of the function with respect to the weights of the network approaches zero, thus causing instability in the computations. LSTM, on the other hand, tries to solve this problem by introducing the concept of a forget gate, a mathematical formulation based on contracting functions that discard information when necessary. Its importance lies in the fact that it enables us to efficiently model different dependencies in sequences (such as time series), and is thus used in the applications presented in this paper. When used without data uncertainty, this algorithm is the baseline against which we compare the accuracy of our algorithm considering data uncertainty. Note, however, the proposed method can be used with any other RNNs.

Remark 4. LSTM is usually followed by a Fully Connected Layer (FCL) to map the representation in the last hidden layer to the target values. In our case, FCL has a linear form.

2.5. Parallels and differences between LS and machine learning

The background given above is needed to understand the similarities and differences between LS and machine learning approaches. The most important similarity is that both approaches deal with finding the best-fitting parameters of a function based on a set of observations and a fitting measure. However, to determine the parameters in the LS approach, usually the model is either linear or linearized, whereas in machine learning and especially deep learning, the function is usually highly nonlinear. As a result, the closed form of the minimization problem is available in the case of LS, but not in machine learning. In the latter case, other iterative and more general optimization algorithms such as gradient descent (Nocedal and Wright, 1999) are thus used to estimate the parameters. Note that the LS procedure of linearization cannot be well exploited for the case of deep learning because of the high complexity, despite the fact that it can be used in somewhat shallow neural networks (Al-Batah et al., 2010).

3. Inclusion of uncertainties in machine learning

In this section, we use the concepts introduced in the previous section for the inclusion of uncertainties in machine learning. We first use aleatory uncertainties to derive a new form of loss function incorporating the data uncertainty. We then focus on modelling epistemic uncertainty in the parameters.

It is important to note that the benchmark machine learning approaches use the following loss function, which is the L_2 norm, as the following

$$\arg\min_W ||Y - f(X, W)||^2 \tag{10}$$

The norm in this case is called Sum Squared Error (SSE), which is different from Mean Squared Error (MSE) only by a coefficient that does not affect the optimization problem.

3.1. Target uncertainty

Here, we only consider the uncertainties in the target values Y. We consider the following optimization problem, based on Eq. (9)

$$\arg\max_W p(Y|X, W) \tag{11}$$

As it is well known (Bishop, 2009), in the case that we assume the data come from a Gaussian distribution, it can be shown that

$$p(Y|X,W) \propto \exp\left(-\frac{1}{2}(Y-f(X,W))^T\Sigma_Y(Y-f(X,W))\right) \tag{12}$$

Therefore by using Eq. (12) in Eq. (11), we arrive at the minimization problem

$$\arg\min_W (Y-f(X,W))^T\Sigma_Y(Y-f(X,W)) \tag{13}$$

which is equivalent to weighted norm of $Y-f(X,W)$, denoted as

$$\arg\min_W ||Y-f(X,W)||^2_{\Sigma_Y} \tag{14}$$

This is in fact the procedure from which the weighted LS is derived. However, the function f in Eq. (13) is usually a deep learning function that is highly nonlinear. With the observation uncertainties available, we can form Σ_Y and subsequently use numerical optimization schemes (Nocedal and Wright, 1999) to solve for W.

Remark 5. Note that we used the assumption of a Gaussian distribution for target values. Other distributions can also be considered and used to model the problem based on Eq. (11) with the target uncertainties modified for that model. However, here the Gaussian distribution is considered to be consistent with the LS procedure.

3.2. Input and target uncertainty

A more realistic approach in dealing with the uncertainty is to consider the uncertainties of both inputs X and targets Y. Note that in this part it is assumed that both the input and target have Gaussian distribution. In many sequential approaches such as LSTM, the targets are some shifted form of the inputs. Inspired by TLS (Jazaeri et al., 2014), in which the input uncertainties are also taken into account, we propose the following optimization problem for the adjustment of the weights of the machine learning function.

Proposition 1. *Suppose that the uncertainties in the input X cause a change in the model, which we denote by δf. Thus Eq. (5) can be rewritten as*

$$Y = f(X,W) + \delta f(X,W) \tag{15}$$

in which $f(X,W)$ is the unperturbed machine learning function. The optimization scheme in which the aleatory uncertainties of the input and target are taken into account has the following form

$$\arg\min_W \left(||Y-f(X,W)||^2_{\Sigma_Y} + ||\delta f(X,W)||^2_{\Sigma_{\delta f}}\right) \tag{16}$$

where $\Sigma_{\delta f}$ is the precision matrix (or equivalently weight matrix) of δf.

Remark 6. Transitioning from Eq. (15) to Eq. (16) eliminates cross terms between $Y-f(X,W)$ and $\delta f(X,W)$, which are in the form of inner products. This is because of the fact that in the optimization Y is projected to the span of design matrix formed from $f(X,W)$. The term $Y-f(X,W)$ is therefore orthogonal to the span of $f(X,W)$, in which $\delta f(X,W)$ lies as well. Thus, the inner product $Y-f(X,W)$ and $\delta f(X,W)$ is zero. The reader is referred to Fox (2008) and other textbooks on least squares.

To obtain the exact form of $\Sigma_{\delta f}$, we have the following proposition, which is inspired by Amiri-Simkooei et al. (2016) and applied to the deep learning function.

Proposition 2. *The precision matrix $\Sigma_{\delta f}$ is the inverse of $Q_{\delta f}$, the covariance matrix of δf defined as the following*

$$Q_{\delta f} = J_X f_W Q_X (J_X f_W)^T \tag{17}$$

where f_W denotes the differentiation of the deep learning function with respect to the weights and J_X the Jacobian of f_W with respect to input X. These are computed through an automatic differentiation approach (Bcker et al., 2006).

Remark 7. Note that the optimization algorithms that are used for machine learning problems are generally based on the computation of the gradient (Kingma and Ba, 2015), which is equivalent to the first order differentiation in higher dimensional spaces. These algorithms, which use iterative linearization through the gradient, are convergent. Therefore, unlike the nonlinear method presented in (Teunissen, 1989), it is sufficient to use the linear form given in Eq. (17).

The relation in Eq. (17) is a direct consequence of the application of the covariance propagation law (Tellinghuisen, 2001) to the function given in Eq. (5). Note also that $Q_{\delta f}$ changes in the iterative procedure to adjust network weights.

3.3. Input and target uncertainty: combination with Bayesian learning

In this part, we present a complete framework to deal with uncertainties in machine learning. The uncertainties in input and target are aleatory, whereas the Bayesian learning deals with epistemic uncertainties. Therefore, the method presented here deals with both types of uncertainty.

Proposition 3. *With regard to Eq. (9) and considering the weights as taken from a Gaussian distribution* (Bishop, 2009; Alpaydin, 2020) *with zero mean and variance λ^{-1}, the following model is both based on Bayesian learning and takes the aleatory uncertainties into account*

$$\arg\min_W ||Y-f(X,W)||^2_{\Sigma_Y} + ||\delta f||^2_{\Sigma_{\delta f}} + \lambda||W||^2 \tag{18}$$

The derivation of formula in Eq. (18) is simply based on the addition of the term $||\delta f||^2_{\Sigma_{\delta f}}$ to the well-known minimization problem of the so-called Bayesian regulation,

given as follows (Bishop, 2009), in which we have used the uncertainty matrix Σ_Y as well

$$\arg\min_W ||Y - f(X,W)||^2_{\Sigma_Y} + \lambda||W||^2 \tag{19}$$

This approach is very common in the application of least squares and it appears with names such as Tikhonov Regularization (Calvetti and Reichel, 2003). Note that even with the Bayesian regulation in Eq. (19) we are using the uncertainties (weighted loss by Σ_Y), which is different from the usual Bayesian regulation that is used in neural networks (Izmailov et al., 2021). The relation in Eq. (18) contains three elements of uncertainty. The first term is due to the target uncertainty. The second term arises as the result of the uncertainty in the input data. Finally, the third term is related to the prior distribution of the unknown parameters W. In Section 5, we investigate the contribution of these terms with respect to the performance of the benchmark. To this end, we focus on the prediction problems for geodetic applications.

Remark 8. Note that we have not used the so-called full Bayesian approach (Bishop, 2009) since we mostly focus on the improvement of the generalizability in comparatively small datasets. The full Bayesian approach requires a large amount of data and extensive computations.

3.4. Alternative approach: aleatory uncertainties as features

So far, we have discussed the case where the uncertainties are considered in the loss function. However, it is also possible to insert uncertainties as additional input features together with the original input values. Thus, in this case we have

$$Y = f\left(\widetilde{X}, W\right) \tag{20}$$

where $\widetilde{X}$ is the input to the network incorporating the input uncertainties with

$$\widetilde{X} = X \| \Sigma_X \tag{21}$$

where $\|$ is used as the concatenation symbol.

Remark 9. The method presented in Eq. (18) is based on the modification of the loss function of the neural network, whereas in Eq. (20) extra features are considered without the modification to the loss function given in Eq. (10). In fact, the loss function used for the Bayesian approach can also be used alongside Eq. (20) as in Eq. (22)

$$\arg\min_W ||Y - f\left(\widetilde{X}, W\right)||^2 + \lambda||W||^2 \tag{22}$$

4. Data used for demonstration

In order to assess the methods presented in Eq. (18) and Eq. (20), we present two studies in the field of geodesy. The first study is about the prediction of Earth Orientation Parameters (EOP, in practice x_p and y_p components of the polar motion, LOD, and dUT1), and the second one is focused on the prediction of GNSS station coordinate time series. EOP prediction remains one of the most important tasks of geodesy (Modiri et al., 2020) and is needed in different applications including satellite orbit determination (Ye et al., 2020). In addition, GNSS time series analysis is an important task in geodesy to investigate various physical phenomena (Montillet and Bos, 2020). The predicted values can be used for positioning purposes. Note that we focus on the one-day ahead prediction problem.

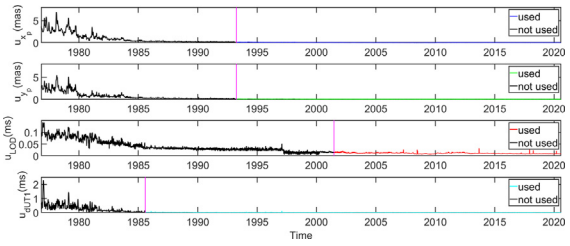

Fig. 1. Uncertainties in the EOP time series together with the interval used in the study. $u_{x_p}, u_{y_p}, u_{LOD}, u_{dUT1}$ denote, respectively, the uncertainties in x_p, y_p, LOD, and $dUT1$. The purple line marks the separation boundary between discarded and used data.

4.1. EOP data

The data for the EOP study are taken from NASA JPL (Chin et al., 2004; Chin et al., 2009) EOP noon files. The data include the values of EOP from September 1976 to June 2020, together with their uncertainties[1].

Remark 10. Since the data from earlier times have much higher uncertainty compared to the more recent data, we exclude the former part. This is to have a more homogeneous time series. Hence, in polar motion, LOD and dUT1 time series we use the time interval, respectively from March 1993, May 2001, July 1985 to June 2020, shown in Fig. 1. Apart from this choice of time interval, before feeding data to the deep learning algorithm they are standardized using 1σ standardization

$$X_S = \frac{X - \overline{X}}{\sigma_X} \tag{23}$$

where X_S is the standardized value of X, and $\overline{X}$ and σ_X represent the mean and standard deviation of X, respectively. When used as features, the uncertainties are also standardized. After predicting the time series, the predictions are scaled back to the original values by first multiplying them with σ_X and then adding $\overline{X}$. In the case of dUT1, we first remove the effect of leap-seconds observed as jumps.[2]

[1] The data can be accessed via https://keof.jpl.nasa.gov/combinations/2019/. Note that we have used the 2019 EOP noon, combined file (which is derived from the combination of different space geodetic techniques)

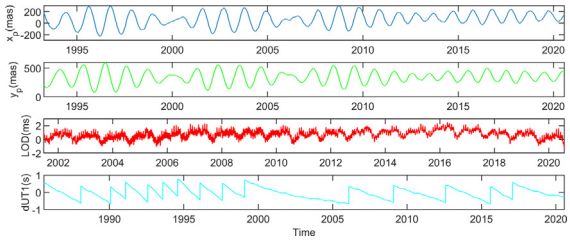

Fig. 2. EOP time series before normalization, with different time intervals that are used for the study. The dUT1 time series is shown before the removal of leap seconds.

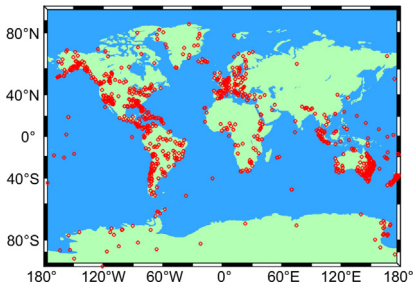

Fig. 3. Distribution of the selected GNSS stations across the globe.

Fig. 2 shows the values of these parameters, based on the time interval used for the study.

4.2. GNSS station coordinates data

The GNSS station coordinates time series are taken from the Nevada Geodetic Laboratory (Blewitt et al., 2018; Young et al., 2021)[3]. In order to show the efficiency of using uncertainties in deep learning for GNSS time series prediction, we have taken 1000 time series, which are located in different parts of the world. We divided the Earth's surface in $10^\circ \times 10^\circ$ cells and picked up the stations iteratively and uniformly, ensuring a uniform probability for each cell to contain stations.

Each station provides three time series, representing the three-dimensional change in the local coordinate system (e, n, u). We only use the u component, which represents the vertical direction and typically exhibits the strongest nonlinear signals. Fig. 3 shows the distribution of these stations over the globe.

Remark 11. We perform some preprocessing on all of these time series. First, we use median filtering (Dinc et al., 2015) to smooth the data and especially to remove outliers. This method is robust against outliers and also fills the missing points with a median estimate of the values before and after those points. To be a good compromise between filling small gaps and not smoothing too much, we choose the window length of 6 for the smoothing, as this has shown empirically to be the best value. In addition, like the EOP data, we perform the standardization in Eq. (23) for each GNSS station coordinate time series. However, unlike the EOP case, the whole available time series are used, since the time series are shorter and uncertainties do not change extensively in more recent times.

4.3. Use case demonstration

We present seven different analyses for each of the four EOP parameters and 1000 GNSS station position time series. These analyses are considered.

1. Benchmark
2. Target uncertainties
3. Target and input uncertainties
4. Target and input uncertainties combined with Bayesian learning
5. Target uncertainties combined with Bayesian learning
6. Input uncertainties as feature
7. Input uncertainties as feature combined with Bayesian learning

In our investigations, the term benchmark refers to the method without using either Bayesian or aleatory and epistemic uncertainties. The different cases of analyses are abbreviated in Table 1.

A neural network consisting of one LSTM layer is used, as shown in Fig. 4. The number of hidden neurons, denoted by H, is 1000. For all the analyses the same architecture is used. We have tried several other architectures and this one achieved the best performance in our tests. It is also simple and efficient such that it can cope with different types of input data. However, it can be shown that the method presented in this paper can perform well across a wide variety of network architectures, particularly for sequential neural networks.

The architecture consists of the layer of the inputs, recurrent model, fully connected linear mapping, and the layer of outputs. The dimension of the outputs Y depends on the number of values in the training and test sets. If there are n values in the training set, Y has the dimension $1 \times n$. The dimension of inputs X, however, is dependent on the number of features used, such that in case of using uncertainties as features (i.e, IF and IFB in Table 1) the dimension of the X reads as $2 \times n$, while in other cases it is $1 \times n$, the same as that of Y.

We have used the first 70 percent of data as training and the last 30 percent for the test of generalization (see Table 2). Adam optimization scheme is used (Kingma and Ba, 2015) for the optimization of the network. For the initial values of the weights $W \in \mathbb{R}^{H \times c_1 \times \ldots \times c_p}$ in the Adam algorithm we use the Glorot point initialization (Glorot and Bengio, 2010). The inputs are primarily the values of EOP and GNSS station coordinates time series and targets are the shifted values of inputs by one. Assuming there are n points in the input X, then the input and targets are, respectively, $[X_1, X_2, \ldots, X_{n-1}]$ and $[X_2, X_3, \ldots, X_n]$

[3] The data can be accessed via http://geodesy.unr.edu/NGLStationPages/GlobalStationList

Table 1
Analyses performed and their abbreviation. In the first four cases, the modification is done in the loss function, using Eq. (14), (16), (18), (19). In the last two cases, however, the uncertainties are used as features, using Eq. (20)–(22).

Analysis	Abbreviation
Target uncertainty	T
Target and input uncertainties	IT
Target and input uncertainties combined with Bayesian learning	ITB
Target uncertainties combined with Bayesian learning	TB
Input uncertainties as feature	IF
Input uncertainties as feature combined with Bayesian learning	IFB

for predicting one epoch to the future, see Fig. 5. In the case of our time series, we effectively predict a single epoch into the future from a set of values from the past. Additional epochs can be predicted by shifting the targets and inputs by a value bigger than one day.

Remark 12. In the EOP prediction study the number of samples is 9977 for polar motion components, 6978 for LOD, and 12778 for dUT1. The number of samples for training and testing are given in Table 5. In the case of GNSS station coordinates time series the number of training and testing for each time series is slightly different, but mostly around 2450 for training and 1050 for testing.

5. Results and discussions

5.1. EOP prediction

We predict the targets of the test samples using the sliding window approach mentioned in Fig. 5 and subtract their corresponding actual values in order to evaluate the performance. The results of the seven variations of the method are shown for each of the four selected EOP parameters in Figs. 6–9. Note that the curves in each figure are shifted so that the differences become clearer. The benchmark shows relatively lower performance compared to the other methods (for the numerical results refer to Table 3). The performance of the T, IT, TB, IF, IFB is similar, which shows they are not the most useful combination. In both parameters x_p and y_p using the complete loss function in Eq. (18) results in the best accuracy. From Figs. 6,7 one can see that there are periodic signals that are not modelled well with the recurrent model. A deeper network can, however, capture the periodicities, but the overall predictive performance is lower, meaning the current architecture works better than a model that captures the periodicities. In fact, we tested an architecture that has three layers of LSTM, with respectively 45, 30, and 7 for the number of hidden nodes H. Despite the fact that the periodicities are seemingly covered, the overall performance decreased. This could be explained by the fact that as the depth of the neural network increases, it could model the time series

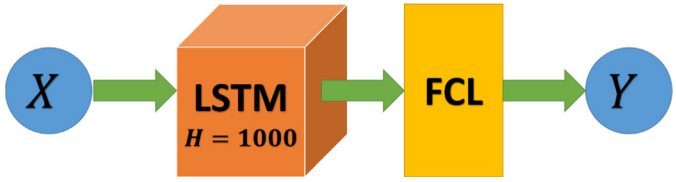

Fig. 4. LSTM deep learning architecture used in this study, together with the linear mapping FCL. H is the number of hidden nodes of the LSTM layer.

Table 2
Number of training and testing samples for EOP prediction based on a 70%-30% split.

EOP	Total	Training	Test
Polar motion	9978	6983	2994
LOD	6978	4884	2094
dUT1	12778	8944	3834

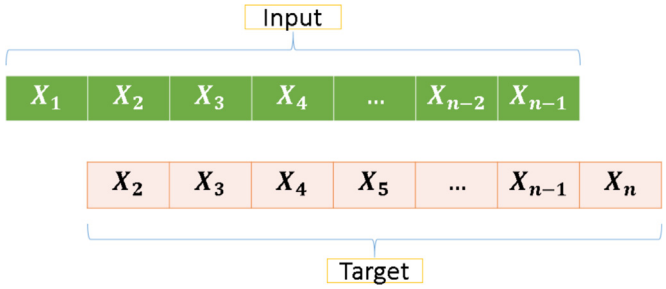

Fig. 5. Input X and target Y values with Y equal to X shifted by one epoch. This corresponds to the prediction of one epoch to the future.

better. On the other hand, however, the number of parameters to be estimated increases (it is $O(H^2)$) and thus there is the need for more data. But in this case the number of samples in data is the same for different architectures.

The y_p component shows better accuracy compared to the x_p component. This can be explained by the fact that the y_p component is determined more accurately than the x_p component. In fact, this is in line with the better prediction performance for y_p compared to x_p in different studies such as Modiri et al. (2018) and Jin et al. (2021).

From Fig. 8 it is observed that all the methods show similar performances and, unlike the case with components of polar motion, using uncertainties as features is advantageous. This could be referred to the fact that LOD represents a much more variable time series and more features help to understand the changes better, rather than constraining and modifying the loss function. This is in contrast to the much less variable dUT1 time series (even though its derivative is related to LOD, it is mostly dominated by secular trends), for which, like the polar motion components, using uncertainties in the loss function is more efficient than using them as features. As one can see from Fig. 9, using uncertainties as features is disadvantageous for dUT1. For time series like dUT1 with a strong

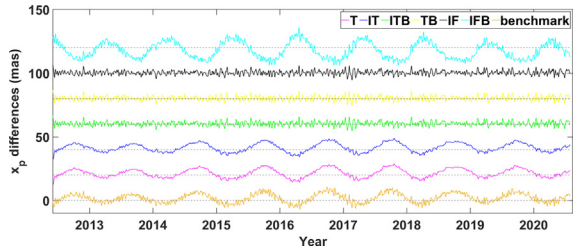

Fig. 6. Differences between the predicted and observed targets for the test set, x_p component of the polar motion. The curves are shifted by 20 mas each to make the the differences clearer.

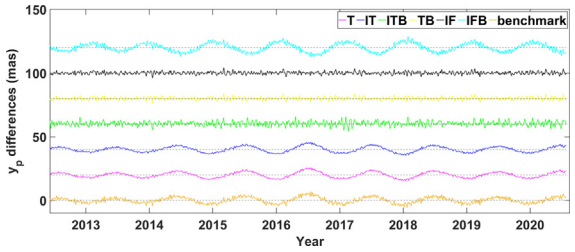

Fig. 7. Differences between the predicted and observed targets for the test set, y_p component of the polar motion. The curves are shifted by 20 mas each to make the the differences clearer.

trend, machine learning algorithms have difficulty learning (He et al., 2016), so adding more features would not help, but constraining the loss function and using Bayesian learning seems to help for a better generalization.

In order to quantitatively compare the prediction uncertainty of the seven variations of the proposed method, we use three measures of uncertainty, namely Root-Mean-Square Error (RMSE), Mean Absolute Error (MAE), and symmetric Mean Absolute Percentage Error (sMAPE, discussed in Makridakis et al. (2018)), defined based on the q observed target values $Y_i, i = 1, \ldots, q$ and their corresponding predicted values $\widehat{Y}_i$ as the following.

$$\begin{aligned} RMSE &= \sqrt{\frac{1}{q}\sum_{i=1}^{q}\left(Y_i - \widehat{Y}_i\right)^2} \\ MAE &= \frac{1}{q}\sum_{i=1}^{q}|Y_i - \widehat{Y}_i| \\ SMAPE &= \frac{200}{q}\sum_{i=1}^{q}\frac{|Y_i - \widehat{Y}_i|}{|Y_i| + |\widehat{Y}_i|} \end{aligned} \tag{24}$$

In Table 3, the values of these measures are included for each individual EOP and for each of the seven variations of the method. Note that the unit of RMSE and MAE for polar motion is milliarcseconds (mas), whereas for LOD and dUT1 are microseconds (μs) and milliseconds (ms), respectively. sMAPE is expressed in terms of percentages (%).

From Tables 3, it is evident that for polar motion and dUT1 the inclusion of target uncertainties in the loss function mentioned in Eq. (14) produces better results than considering them as features and even more so, when compared to the benchmark. The effect of the Bayesian learning is smaller than that of the target uncertainty. This implies that the most important contribution comes from the target uncertainties. For LOD, however, consideration of the uncertainties as features is more powerful than their inclusion in the loss function. The reason could be, as mentioned, due to the higher variability of LOD compared to other time series. In this case, using more features would help achieving better results compared to the benchmark. Across all parameters the effect of uncertainty in data–either as input or in the loss function–can be readily observed when comparing the prediction performance to the benchmark. This implies uncertainties contain useful source of information in machine learning framework.

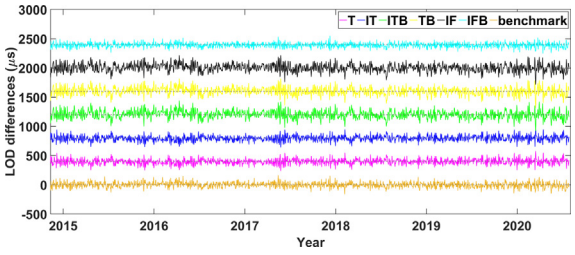

Fig. 8. Differences between the predicted and observed targets for the test set, LOD. The curves are shifted by 400 μs each to make the the differences clearer.

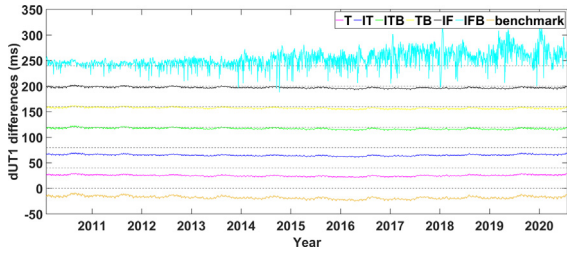

Fig. 9. Differences between the predicted and observed targets for the test set, dUT1. The curves are shifted by 40 ms each to make the the differences clearer.

5.2. GNSS time series prediction

We apply the same network architecture as in the case of EOP prediction. As an example, the differences between predictions and observed values of the u component (target) of the station 0FRL is shown in Fig. 10 for the same seven variations of the proposed method.

In Table 4 the mean of the three mentioned uncertainty measures over the 1000 stations of each of the seven analyses is given.

As one can observe from Table 4, based on MAE and sMAPE, addition of target uncertainties results in better

Table 3
RMSE, MAE, and sMAPE measures of the prediction of EOP, for seven different variations of the proposed method.

EOP	T	IT	ITB	TB	IF	IFB	benchmark
RMSE							
x(mas)	2.38	2.38	2.38	2.38	6.88	6.85	4.35
y(mas)	1.10	1.10	1.10	1.10	3.23	2.45	2.34
LOD(μs)	67.83	67.83	66.33	66.33	41.29	41.62	45.88
dUT1(ms)	3.16	3.16	2.81	3.16	20.44	20.44	17.85
MAE							
x(mas)	1.42	1.42	1.42	1.42	5.87	5.77	3.33
y(mas)	0.85	1.42	0.85	0.85	2.78	1.99	1.94
LOD(μs)	53.29	53.29	52.09	52.09	32.16	32.49	35.58
dUT1(ms)	2.81	2.81	2.81	2.81	15.64	15.63	17.61
sMAPE							
x(%)	6.06	6.06	6.06	6.06	19.53	18.04	7.47
y(%)	0.25	6.06	0.25	0.25	0.82	0.54	0.56
LOD(%)	14.90	14.90	14.90	14.90	7.77	7.60	13.64
dUT1(%)	0.02	0.02	0.02	0.02	0.12	0.12	0.14

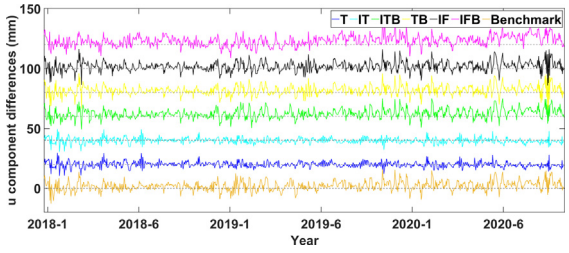

Fig. 10. Differences between the predicted and observed targets for the test set, 0FRL GNSS station coordinate time series, upward component (u). The curves are shifted make the the differences clearer: [20,40,60,80,100,120] mm.

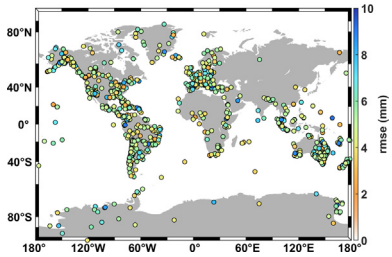

Fig. 11. Distribution of RMSE (mm) of the ITB configuration for all the 1000 GNSS stations shown in Fig. 3.

generalization when compared to the results obtained by the benchmark. In fact, the ITB variation works much better than the other varations, including TB and IT. Considering uncertainties as features is not performing as well, but still better than the benchmark. This means that it is usually better to constrain and modify the loss function for GNSS time series prediction, since the uncertainties are good indicators of the quality and usefulness of the samples.

In Fig. 11, the distribution of the RMSE (mm) of the ITB configuration which performed the best is shown for all the GNSS stations in Fig. 3. From Fig. 11, it can be seen that the general performance for different time series at different locations is approximately uniform. This negates the possibility of a large contribution from the length of the time series, when the length of time series are approximately in the same range. However, it reinforces the fact that the most important contribution comes from the use of uncertainties in the loss function.

Table 5
Time required to perform the training of algorithms and prediction of GNSS station coordinates.

Study	time required (days, hours)
T	1^d14^h
IT	1^d16^h
ITB	1^d17^h
TB	1^d15^h
IF	1^d13^h
IFB	1^d21^h
Benchmark	1^d12^h

Table 4
Mean of prediction accuracy measures of 1000 globally distributed GNSS stations for the different variations of the proposed method.

Uncertainty measure	T	IT	ITB	TB	IF	IFB	Benchmark
RMSE(mm)	14.74	15.78	5.21	15.75	15.93	15.80	16.06
MAE (mm)	11.35	12.30	3.90	12.26	12.40	12.26	12.83
sMAPE (%)	3.84	4.12	3.93	4.15	4.71	4.39	4.95

6. Conclusions

Inspired by weighted and total least squares, we developed a framework for incorporating data uncertainties in deep learning. We designed a specific loss function to include both input and target uncertainties, as well as to incorporate Bayesian learning. Through several tests related to geodetic parameter prediction, we show that the loss function that includes uncertainties in the input, output and Bayesian learning results in better generalizations. It is necessary to mention that the focus of the study is not on the prediction of a specific time series and to set a new baseline in terms of high accuracy compared to other methods. Rather, we show that the use of uncertainties in neural networks is advantageous for making better predictions compared to the case of not using them. This means that almost any regression problem for geodetic time series prediction can be analyzed in this framework. Using more specifically defined state-of-the-art algorithms in this framework could result in even better generalization and predictions. We specifically showed the improvement in the prediction performance by neural networks when applied to EOPs and GNSS station coordinate time series. The improvements can be more than 50% in the specific study interval of EOPs prediction. For the 1000, globally distributed GNSS station coordinate time series, the average improvement for all the time series is approximately 12%.

Although the method presented in this paper is fundamentally used for the regression problem, it can be applied to the classification problem as well. In the case of time series classification (Fawaz et al., 2019), the class uncertainty may not play role as important as it did for the regression problem. Therefore, it seems that the role of input uncertainties would be more evident there. In addition, the method can be applied to other fields. If the uncertainties in data are quantified, they can be used in deep learning approaches to achieve better generalization. For future works, we suggest to consider the uncertainties as the initial values of the uncertainties to be derived from data based on the method in Kendall and Gal (2017), provided that we have sufficient amount of data. This way, the uncertainties are re-estimated based on they behavior in neural networks. In addition, it would be an interesting problem to consider distributions other than Gaussian for datasets. The method in Kendall and Gal (2017) is especially useful when the size of the dataset is large and we have the uncertainties in data. This would help in deriving more reliable uncertainties.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Appendix A. Appendix

MATLAB R2021a is used for the experiments in the paper. Custom deep learning layers are implemented using the MATLAB deep learning toolbox. The code was run on the ETH Zurich Leonhard cluster, using 50 GB of RAM (since then, Leonhard has been integrated into the Euler cluster). The analysis of GNSS stations is more time consuming than the study of EOPs because of the sheer number of stations used in this study. The training of the algorithms and prediction of EOPs is a matter of a few minutes. The time required for each of the seven analyses in the GNSS station coordinate prediction study is mentioned in Table 5. As it can be seen, usually the most time consuming models are those that use the uncertainty in the input, i.e., IT and ITB. However, IFB, unlike IF, seems to be the most time consuming. The reason is the higher number of epochs used by the training procedure to learn meaningfully from data compared to the other methods. When the models do not improve after a certain number of epochs, the training process is stopped, referred to as early stopping in machine learning literature (Li et al., 2020).

References

Al-Batah, M.S., Isa, N.A.M., Zamli, K.Z., et al., 2010. Modified recursive least squares algorithm to train the hybrid multilayered perceptron (HMLP) network. Appl. Soft Comput. 10 (1), 236–244. https://doi.org/10.1016/j.asoc.2009.06.018.

Alpaydin, E., 2020. Introduction to machine learning, Fourth Edition. MIT Press.

Amiri-Simkooei, A.R., Tiberius, C.C.J.M., Teunissen, P.J.G., 2007. Assessment of noise in GPS coordinate time series: methodology and results. J. Geophys. Res.: Solid Earth 112 (B7). https://doi.org/10.1029/2006JB004913.

Amiri-Simkooei, A.R., Jazaeri, S., 2012. Weighted total least squares formulated by standard least squares theory. J. Geodetic Sci. 2 (2), 113–124. https://doi.org/10.2478/v10156-011-0036-5.

Amiri-Simkooei, A.R., Mortazavi, S., Asgari, J., 2016. Weighted total least squares applied to mixed observation model. Survey Rev. 48 (349), 278–286. https://doi.org/10.1179/1752270615Y.0000000031.

Amiri-Simkooei, A.R., Zangeneh-Nejad, F., Asgari, J., 2016. On the covariance matrix of weighted total least-squares estimates. J. Surv. Eng. 142 (3). https://doi.org/10.1061/(ASCE)SU.1943-5428.0000153.

Benitez, P.L., Garcia, M.C., Riquelme, J.C., 2021. An experimental review on deep learning architectures for time series forecasting. Int. J. Neural Syst. 31. https://doi.org/10.1142/S0129065721300011.

Beutler, G., Villiger, A., Dach, R., et al., 2020. Long polar motion series: Facts and insights. Adv. Space Res. 66 (11), 2487–2515. https://doi.org/10.1016/j.asr.2020.08.033.

Bishop, C.M., 2009. Pattern recognition and machine learning. Springer.

Bcker, M., Corliss, G., Hovland, P., et al., 2006. Automatic differentiation: Applications, theory, and implementations. Lecture Notes in Computational Science and Engineering. Springer. https://doi.org/10.1007/3-540-28438-9.

Butt, J., Wieser, A., Gojcic, Z., et al., 2021. Machine learning and geodesy: A survey. Appl. Geodesy. 15 (2), 117–133. https://doi.org/10.1515/jag-2020-0043.

Blewitt, G., Hammond, W.C., Kreemer, C., 2018. Harnessing the GPS data explosion for interdisciplinary science. Eos 99. https://doi.org/10.1029/2018EO104623.

Calvetti, D., Reichel, L., 2003. Tikhonov regularization of large linear problems. BIT Num. Mathe. 43, 263–283. https://doi.org/10.1023/A:1026083619097.

Chin, T.M., Gross, R.S., Boggs, D.H., et al., 2009. Dynamical and observation models in the Kalman Earth orientation filter. Interplanet. Network Progress Report 42 (176), 1–25.

Chin, T.M., Gross, R.S., Dickey, J., 2004. Modeling and forecast of the polar motion excitation functions for short-term polar motion prediction. J. Geodesy. 78, 343–353. https://doi.org/10.1007/s00190-004-0411-4.

Dantzig, G.B., 2002. Linear programming. Ope. Res. 50 (50), 42–47. https://doi.org/10.1007/b97672.

Dinc, I., Dinc, S., Sigdel, M., et al., 2015. Emerging trends in image processing, computer vision and pattern recognition. Elsevier, Chapter 12, pp. 183–199, https://doi.org/10.1016/C2014-0-01692-9.

Fawaz, H.I., Forestier, G., Weber, J., et al., 2019. Deep learning for time series classification: a review. Data Min. Knowl. Disc. 33, 917–963. https://doi.org/10.1007/s10618-019-00619-1.

Fazilova, D., 2017. The review and development of a modern GNSS network and datum in Uzbekistan. Geodesy Geodyn. 8 (3), 187–192. https://doi.org/10.1016/j.geog.2017.02.006.

Fox, J., 2008. Applied regression analysis and generalized linear models, (2nd ed.). Sage Publications Inc..

Gast, J., Roth, S., 2018. Deep learning. Lightweight Probabilistic deep networks. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition. https://doi.org/10.1109/CVPR.2018.00355.

Gers, F.A., Schmidhuber, J., Cummins, F., 1999. Learning to forget: continual prediction with LSTM. In: 9th International Conference on Artificial Neural Networks: ICANN, pp. 850–855. https://doi.org/10.1049/cp:19991218.

Glorot, X., Bengio, Y., 2010. Understanding the difficulty of training deep feedforward neural networks. In: Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, vol. 9, pp. 249–256, https://proceedings.mlr.press/v9/glorot10a.html.

Goodfellow, I., Bengio, Y., Courville, A., 2016. Deep learning. MIT Press, http://www.deeplearningbook.org.

Greff, K., Srivastava, R.K., Koutnk, J., et al., 2017. LSTM: A Search Space Odyssey. IEEE Trans. Neural Networks Learn. Syst. 28 (10), 2222–2232. https://doi.org/10.1109/TNNLS.2016.2582924.

He, K., Zhang, X., Ren, S., et al., 2016. Deep residual learning for image recognition. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2016), https://doi.org/10.1109/CVPR.2016.90.

Hochreiter, S., Schmidhuber, J., 1997. Long short-term memory. Neural Comput. 9 (8), 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735.

Izmailov, P., Vikram, S., Hoffman, M.D., et al., 2021. What are Bayesian neural network posteriors really like? In: Proceedings of the 38th International Conference on Machine Learning, PMLR, vol. 139, pp. 4629–4640, https://proceedings.mlr.press/v139/izmailov21a.html.

Jazaeri, S., Amiri-Simkooei, A.R., Sharifi, M.A., 2014. Iterative algorithm for weighted total least squares adjustment. Survey Rev. 46 (334), 19–27. https://doi.org/10.1179/1752270613Y.0000000052.

Jiang, H., Nachum, O., 2020. Identifying and correcting label bias in machine learning. In: Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics (AISTATS), vol. 108, pp. 702–712, http://proceedings.mlr.press/v108/jiang20a.html.

Jin, X., Liu, X., Guo, J., et al., 2021. Analysis and prediction of polar motion using MSSA method. Earth, Planets Space 73, 147. https://doi.org/10.1186/s40623-021-01477-2.

Kalarus, M., Schuh, H., Kosek, W., et al., 2010. Achievements of the Earth orientation parameters prediction comparison campaign. J. Geodesy. 84, 587–596. https://doi.org/10.1007/s00190-010-0387-1.

Kendall, A., Gal, Y., 2017. What uncertainties do we need in Bayesian deep learning for computer vision? In: Proceedings of the 31st International Conference on Neural Information Processing Systems, pp. 5580–5590.

Kermarrec, G., Schn, S., 2017. A priori fully populated covariance matrices in least-squares adjustment–case study: GPS relative positioning. J. Geodesy. 91, 465–484. https://doi.org/10.1007/s00190-016-0976-8.

Kim, B., Kim, H., Kim, K., et al., 2019. Learning not to learn: Training deep neural networks with biased data. In: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition. https://doi.org/10.1109/CVPR.2019.00922.

Kiani Shahvandi, M., Soja, B., 2021. Modified deep transformers for GNSS time series prediction. In: International Geoscience and Remote Sensing Symposium 2021, https://doi.org/10.1109/IGARSS47720.2021.9554764.

Kiani Shahvandi, M., Soja, B., 2022. Small Geodetic Datasets and Deep Networks: Attention-Based Residual LSTM Autoencoder Stacking for Geodetic Time Series. In: The 7th International Online & Onsite Conference on Machine Learning, Optimization, and Data Science. https://doi.org/10.1007/978-3-030-95467-322.

Kingma, D.P., Ba, J.L., 2015. Adam: A method for stochastic optimization. In: International Conference on Learning Representations (ICLR 2015).

Koch, K.R., 1999. Parameter estimation and hypothesis testing in linear models. Springer. https://doi.org/10.1007/978-3-662-03976-2.

LeCun, Y., Bengio, Y., Hinton, G., 2015. Deep learning. Nature 521, 436–444. https://doi.org/10.1038/nature14539.

Li, J., Miyashita, K., Kato, T., et al., 2000. GPS time series modeling by autoregressive moving average method: Application to the crustal deformation in central Japan. Earth, Planets, Space 52, 155–162. https://doi.org/10.1186/BF03351624.

Li, M., Soltanolkotabi, M., Oymak, S., 2020. Gradient descent with early stopping is provably robust to label noise for overparameterized neural networks. In: Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics (AISTATS), vol. 108, pp. 4313–4324.

Makridakis, S., Spiliotis, E., Assimakopoulos, V., 2018. Statistical and machine learning forecasting methods: Concerns and ways forward. PlosOne. https://doi.org/10.1371/journal.pone.0194889.

Modiri, S., Belda, S., Hoseini, M., et al., 2020. A new hybrid method to improve the ultra-short-term prediction of LOD. J. Geodesy. 94 (23). https://doi.org/10.1007/s00190-020-01354-y.

Modiri, S., Belda, S., Heinkelmann, et al., 2018. Polar motion prediction using the combination of SSA and Copulabased analysis. Earth, Planets Space 70 (115). https://doi.org/10.1186/s40623-018-0888-3.

Montillet, J.P., Bos, M.S., 2020. Geodetic time series analysis in Earth sciences. Springer. https://doi.org/10.1007/978-3-030-21718-1.

MrwczyÅska, M., 2017. Application of support vector machine in geodesy for the classification of vertical displacements. Comput. Assisted Methods Eng. Sci. 21 (1), 77–84.

Navarro, V., Grieco, R., Soja, B., et al., 2021. Data fusion and machine learning for innovative GNSS science use cases. In: Proceedings of the 34th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS + 2021), pp. 2656–2669, https://doi.org/10.33012/2021.18115.

Neal, R.M., 1996. Bayesian Learning for Neural Networks. Lecture Notes in Statistics. Springer. https://doi.org/10.1007/978-1-4612-0745-0.

Nocedal, J., Wright, S.J., 1999. Numerical optimization. Springer. https://doi.org/10.1007/978-0-387-40065-5.

Odijk, D., Zhang, B., Khodabandeh, A., et al., 2016. On the estimability of parameters in undifferenced, uncombined GNSS network and PPP-RTK user models by means of S-system theory. J. Geodesy. 90, 15–44. https://doi.org/10.1007/s00190-015-0854-9.

Piccolomini, E.L., Gandolfi, S., Poluzzi, L., et al., 2019. Recurrent neural networks applied to GNSS time series for denoising and prediction. In: 26th International Symposium on Temporal Representation and Reasoning, vol. 147, pp. 1–12, https://doi.org/10.4230/LIPIcs.TIME.2019.10.

Reiterer, A., Egly, U., Vicovac, T., et al., 2010. Application of artificial intelligence in Geodesy–A review of theoretical foundations and practical examples. J. Appl. Geodesy. 4, 201–217. https://doi.org/10.1515/jag.2010.020.

Ren, C.X., Hulbert, C., Johnson, P.A., et al., 2020. Machine learning and fault rupture: A review. Adv. Geophys. 61, 57–107. https://doi.org/10.1016/bs.agph.2020.08.003.

Rumelhart, D.E., Hinton, G.E., Williams, R.J., 1986. Learning internal representations by error propagation. Parallel distributed processing: explorations in the microstructure of cognition, 1, 318–362.

Sullivan, T.J., 2015. Introduction to Uncertainty Quantification. Springer. https://doi.org/10.1007/978-3-319-23395-6.

Tellinghuisen, J., 2001. Statistical error propagation. J. Phys. Chem., A 105, 3917–3921. https://doi.org/10.1021/jp003484u.

Teunissen, P.T.G., 2000. Adjustment Theory: an introduction. VSSD Press.

Teunissen, P.T.G., 1989. First and second moments of non-linear least-squares estimators. Bull. Godsique 63, 253–262. https://doi.org/10.1007/BF02520475.

Torres, J.F., Hadjout, D., Sebaa, A., et al., 2021. Deep learning for time series forecasting: A survey. Big Data 9. https://doi.org/10.1089/big.2020.0159.

Truong, D.M., Ta. T.H., 2013. Development of real multi-GNSS positioning solutions and performance analyses. In: International Conference on Advanced Technologies for Communications, https://doi.org/10.1109/ATC.2013.6698097.

Wenzel, F., Roth, K., Veeling, B.S., et al., 2020. How good is the Bayes posterior in deep neural networks really?. In: Proceedings of the 37th International Conference on Machine Learning (PMLR 2020), vol. 119, pp. 10248–10259, https://proceedings.mlr.press/v119/wenzel20a.html.

Ye, F., Yuan, Y., Deng, Z., 2020. Improved ultra-rapid UT1-UTC determination and its preliminary impact on GNSS satellite ultra-rapid orbit determination. Remote Sensing 12 (21), 3584. https://doi.org/10.3390/rs12213584.

Young, Z., Kreemer, C., Blewitt, G., 2021. GPS Constraints on Drought-Induced Groundwater Loss Around Great Salt Lake, Utah, with Implications for Seismicity Modulation. Journal of Geophysical Research: Solid. Earth 126. https://doi.org/10.1029/2021JB022020, e2021JB022020.

Zhang, B., Yao, Y., 2021. Precipitable water vapor fusion based on a generalized regression neural network. J. Geodesy. 95 (36). https://doi.org/10.1007/s00190-021-01482-z.

Zhang, C., Bengio, S., Hardt, M., et al., 2017. Understanding deep learning requires rethinking generalization. In: International Conference on Learning Representations (ICLR 2017), vol. 64, no. 3, pp. 107–115, https://doi.org/10.1145/3446776.