# Bayesian Neural Networks Uncertainty Quantification with Cubature Rules

Peng Wang
*Automatic Control & Systems Engineering Dept.*
*The University of Sheffield*
peng.wang@sheffield.ac.uk

Renke He
*Information and Navigation College*
*Air Force Engineering University*
lnzrds@163.com

Qibin Zhang
*The 32nd Research Institute of CETC*
zqb101@mail.ustc.edu.cn

Jikai Wang
*Department of Automation*
*University of Science and Technology of China*
wangjk@mail.ustc.edu.cn

Lyudmila Mihaylova
*Automatic Control & Systems Engineering Dept.*
*The University of Sheffield*
l.s.mihaylova@sheffield.ac.uk

Nidhal C. Bouaynaya
*Department of Electrical and Computer Engineering*
*Rowan University*
bouaynaya@rowan.edu

***Abstract*—Bayesian neural networks are powerful inference methods by accounting for randomness in the data and the network model. Uncertainty quantification at the output of neural networks is critical, especially for applications such as autonomous driving and hazardous weather forecasting. However, approaches for theoretical analysis of Bayesian neural networks remain limited. This paper makes a step forward towards mathematical quantification of uncertainty in neural network models and proposes a cubature-rule-based computationally-efficient uncertainty quantification approach that captures layer-wise uncertainties of Bayesian neural networks. The proposed approach approximates the first two moments of the posterior distribution of the parameters by propagating cubature points across the network nonlinearities. Simulation results show that the proposed approach can achieve more diverse layer-wise uncertainty quantification results of neural networks with a fast convergence rate.**

***Index Terms*—Bayesian neural networks, uncertainty quantification, cubature rules, variational inference, Bayesian rules**

## I. Introduction

During the past decades, Deep Neural Networks (DNNs) have achieved state-of-the-art results in a broad range of applications, including visual object recognition [1] and traffic forecasting [2], [3], [4]. To some extent, DNN methods revolutionised our way of coping with recognition and regression problems, and are holding the promise of emerging technologies like autonomous driving and hazardous weather forecasting. In all these applications, safety concerns are as relevant as accuracy to both researchers/developers and end-users. Most work in the literature focuses on accuracy improvement and network architecture adjustment [5], [6], [7], which indeed have pushed the cutting-edge DNN related research to a new era. However, we shall also point out that these typical networks lack the ability to quantify the uncertainty associated with the network prediction, which is critical in applications, such as autonomous driving and hazardous weather forecasting, where errors could cause severe consequences. Therefore, endowing networks with the ability to quantify their uncertainty is crucial to preventing undesirable and potentially dangerous behaviors in downstream decision making.

The most popular technique on neural network uncertainty quantification is a Bayesian treatment of the network weights and biases, which is well known as Bayesian Neural Networks (BNNs). In the BNN paradigm, network parameters, such as weights and biases are no longer deterministic; Instead, they are endowed with prior probabilistic distributions, e.g., Gaussian distributions. The random parameters then propagate forward and the model uncertainty is estimated by Markov Chain Monte Carlo (MCMC) sampling [8]. Neal introduced the Hamiltonian Monte Carlo, which falls into the MCMC paradigm, to learn BNN parameters by using the Hamiltonian dynamics-based sampling approach [9]. MCMC-based sampling methods approximate the posterior with a fairly high accuracy, but they also suffer from high computational complexity [10].

Variational Inference (VI) has been proved to be the most promising replacement of MCMC methods [11]. It explicitly transforms an inference problem to an optimisation counterpart. The philosophy of using a tractable distribution to approximate the posterior mitigates the issue of dealing with complex intractable integrals. Another benefit is that VI-based methods are much less dependent on computational resources compared with MCMC methods. Both advantages contributed to the wide use of VI approaches in BNNs. For instance, a closed-form solution for propagating moments through Rectified Linear Unit (ReLU) activation functions in a fully connected neural network is proposed in [12]. Gal [13] developed a theoretical framework casting dropout training as approximate Bayesian inference. Recently, Dera et al. [11] have developed the first statistical moments' propagation method for Convolutional Neural Networks (CNNs) with a general choice of activation functions. In [11], after

This work was supported by the National Science Foundation Awards NSF ECCS-EPSRC-1903466 and NSF CCF-1527822. We are also grateful to UK EPSRC support through EP/T013265/1 project NSF-EPSRC: ShiRAS. Towards Safe and Reliable Autonomy in Sensor Driven Systems. We also appreciate the support of NSFC (61703387).

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

the nonlinear activation functions, the mean and covariance of the network parameters are approximated by their first-order Taylor series. This approach simplifies the complexity of computing statistical moments through nonlinearities but the first-order approximation could jeopardize the accuracy.

In this paper, adopting the same framework of approximating Bayesian posterior of BNNs with VI, we employ the cubature rule to approximate the first and second moments of the weights and biases after nonlinear activation functions. With the cubature rule, instead of approximating the posterior by sampling using MCMC, or propagating the mean and covariance as in [11] followed by a linearisation approximation, we select a set of cubature points, which are then propagated layer by layer to quantify the statistical moments. The application of the cubature rule possesses the potential of achieving third-order Taylor series accuracy [14], which expands the spectrum of problems that can be dealt with using the method in [11]. After the cubature points are propagated, the obtained distribution is optimised by VI to enhance the (distribution) approximation accuracy. At the last layer, a regression unit is introduced to further improve the accuracy of the network outputs. In our approach, layer-wise uncertainty quantification becomes very simple since the first two moments are identified by a set of cubature points. The main contributions of this paper therefore include: *1)* the proposed approach for uncertainty quantification in BNNs with the cubature rule. *2)* Variational inference and the cubature rule combined lead to inference with fast convergence and computational efficiency.

The remainder of the paper is organised as follows. Section II introduces the cubature rule and its application in nonlinear integral approximation. Section III describes how variational inference can be used to approximate intractable distributions in Bayesian inference. The Bayesian neural network uncertainty quantification problem is formulated in Section IV. In Section V, we elaborate on how the cubature rule is applied to approximate statistical moments. Section VI presents the evaluation results and Section VII concludes this work.

## II. Cubature rule and Its Application in Nonlinear Approximations

High-dimensional nonlinear integrals frequently emerge in neural network research due to the complex nature of both the problem of interest, including data, models, and the structure of the network. In cases where the system/measurement noises are subject to Gaussian distributions, we often come across a Gaussian integral of the form

$$\mathbf{I}\left(f\right)=\int_{\mathbb{R}^D} f\left(\mathbf{x}\right)\exp\left(-\mathbf{x}^T\mathbf{x}\right)d\mathbf{x}, \tag{1}$$

defined in the Cartesian coordination system, where $D$ is the dimension of the input vector $\mathbf{x}$, and $f(\cdot)$ is an arbitrary function. The integral in (1) can become intractable as $D$ increases. To deal with it, a number of numerical approximations of (1) are proposed, such as particle filtering [15], [16] and Gaussian regression [17]. Recently, the cubature rule [18] was applied to numerically approximate the integral, which has achieved at least third-order Taylor series accuracy [14].

Just like the particle filter, the main idea of using the cubature rule to approximate the integral lies in transforming it into a weighted sum. The key step is a spherical-radial transformation [18] that maps a Cartesian vector $\mathbf{x}\in\mathbb{R}^D$ to a radius $r$ and a direction vector $\boldsymbol{\theta}$, i.e., $\mathbf{x}=r\boldsymbol{\theta}$ with $\boldsymbol{\theta}^T\boldsymbol{\theta}=\mathbf{1}$, so that $\mathbf{x}^T\mathbf{x}=r^2$ for $r\in[0,\infty)$. Thus, the Gaussian integral can be rewritten as

$$\mathbf{I}(f)=\int_0^\infty\int_{U_D} f\left(r\boldsymbol{\theta}\right)r^{D-1}\exp\left(-r^2\right)d\psi\left(\boldsymbol{\theta}\right)dr, \tag{2}$$

where $U_D$ is the surface of the sphere defined by $U_D=\left\{\boldsymbol{\theta}\in\mathbb{R}^D\,\middle|\,\boldsymbol{\theta}^T\boldsymbol{\theta}=1\right\}$ and $\psi\left(\cdot\right)$ is the spherical surface measure or the area element on $U_D$. Therefore, the radial integral in (2) can be rewritten as

$$\mathbf{I}(S)=\int_0^\infty S\left(r\right)r^{D-1}\exp\left(-r^2\right)dr, \tag{3}$$

where $S\left(r\right)$, given in (4), is defined by the spherical integral with the unit weighting function $w\left(\boldsymbol{\theta}\right)=1$,

$$S\left(r\right)=\int_{U_D} f\left(r\boldsymbol{\theta}\right)d\psi\left(\boldsymbol{\theta}\right). \tag{4}$$

The spherical-radial integral can be numerically computed by the spherical cubature rule and the Gaussian quadrature rule. In brief, the radial integral can be computed numerically by the $m_r$-point Gaussian quadrature rule

$$\int_0^\infty f\left(r\right)r^{D-1}\exp\left(-r^2\right)dr\approx\sum_{i=1}^{m_r}w_{r_i}f\left(r_i\right), \tag{5}$$

where $w_{r_i}$ is the radial weight. The spherical integral can be computed numerically by the $m_s$-point spherical rule

$$\int_{\mathbb{R}^D} f\left(r\boldsymbol{\theta}\right)d\psi\left(\boldsymbol{\theta}\right)\approx\sum_{j=1}^{m_s}w_{\theta_j}f\left(r_j\theta_j\right), \tag{6}$$

where $w_{\theta_j}$ is the spherical weight.

Hence, the third-degree spherical-radial rule entailing $2D$ cubature points with $m_r=1$ and $m_s=2D$ can be used to numerically computed the standard Gaussian weighted integral through

$$\mathbf{I}\left(f\right)=\int_{\mathbb{R}^D} f\left(\mathbf{x}\right)\mathcal{N}\left(\mathbf{x};\mathbf{0},\boldsymbol{\Sigma}_{\mathbf{I}}\right)d\left(\mathbf{x}\right)\approx\sum_{i=1}^{2D}w_i f\left(\boldsymbol{\xi}_i\right), \tag{7}$$

where the weight $w_i=\frac{1}{2D}$ and the cubature points $\boldsymbol{\xi}_i=\sqrt{D}\left[\mathbf{I}_i,-\mathbf{I}_i\right]$, with $\mathbf{I}_i$ the $D$-dimensional unit matrix. Note that when $\mathcal{N}(\cdot)$ is not a standard Gaussian distribution, the cubature points need to be transformed. We will give the transformed results in Section V. Please refer to [18] for more details.

The cubature rule is substantially the unscented transformation when the tune parameter $\kappa$ is set to zero [14]. Therefore, the cubature rule-based approximation can achieve the same third-order accuracy as the unscented transformation. Theoretically, this can obtain better approximation results than the method given in [11].

## III. Variational Inference

In Bayesian inference, given a set of observed variables $\mathbf{y} = \{y_{1:o}\}$, and a set of latent variables $\mathbf{z} = \{z_{1:k}\}$, we often need to deal with the following posterior to make inferences.

$$p(\mathbf{z} \mid \mathbf{y}) = \frac{p(\mathbf{y} \mid \mathbf{z})p(\mathbf{z})}{p(\mathbf{y})} \tag{8}$$

However, the denominator $p(\mathbf{y}) = \int p(\mathbf{z}, \mathbf{y}) d\mathbf{z}$ is unavailable in closed form or requires exponential time to compute [19]. Variational inference has been widely used to approximate this kind of intractable posterior distributions [20]. A detailed review of VI can be found in [19].

In VI, we propose a family $\Xi$ of densities over the latent variables, with each $q(\mathbf{z}; \boldsymbol{\lambda}) \in \Xi$ a candidate approximation of the conditional in (8), and $\boldsymbol{\lambda}$ is the parameter of the proposed distribution. The philosophy of VI is turning the approximation problem into an optimisation counterpart, where the best approximate $q^*(\cdot)$ is obtained by minimising the Kullback-Leibler divergence [21] as in

$$q^*(\cdot) = \arg\min_{\lambda} KL\big(q(\mathbf{z}; \boldsymbol{\lambda}) \| p(\mathbf{z}|\mathbf{y})\big), \tag{9}$$

which is substantially a distance measure to evaluate diffusion between $q(\mathbf{z}; \boldsymbol{\lambda})$ and $p(\mathbf{z} \mid \mathbf{y})$.

However, the Kullback-Leibler divergence is not computable because it requires computing $\log p(\mathbf{y})$ (the same reason why the posterior in (8) in not computable.). According to [19], we have

$$\begin{aligned} \log p(\mathbf{y}) &= \log \int_{\mathbf{z}} p(\mathbf{y}, \mathbf{z}) \\ &\geq \mathbb{E}_q \left[\log p(\mathbf{y}, \mathbf{z})\right] - \mathbb{E}_q[\log q(\mathbf{z}; \boldsymbol{\lambda})]. \end{aligned} \tag{10}$$

The difference between the left and the right parts is called Evidence Lower BOund (ELBO), which is denoted as

$$\text{ELBO} = \log p(\mathbf{y}) - KL\big(q(\mathbf{z}; \boldsymbol{\lambda}) \| p(\mathbf{z}|\mathbf{y})\big). \tag{11}$$

From which we see that minimising the Kullback-Leibler divergence is equivalent to maximising the ELBO.

Now the problem is reduced to determine the density family $\Xi$ to maximise the ELBO. The mean-field variational family [19] has been among the most popular ones due to the assumption that the latent variables are mutually independent and each governed by a distinct factor in the variational density as shown in

$$q(\mathbf{z}; \boldsymbol{\lambda}) = \prod_{j=1}^{k} q(z_j, ; \boldsymbol{\lambda}_j). \tag{12}$$

As long as $q(z_j, ; \boldsymbol{\lambda}_j), j = 1, \cdots, k$ are determined, we can maximise the ELBO given in (10) to get $q^*(\cdot)$, and make inferences thereafter.

## IV. Bayesian Neural Networks

It is widely agreed that the nonlinearity and high-dimensionality in neural network models have made them computationally demanding. In the BNN paradigm, the nonlinearity also makes the propagation of the moments challenging. Put simply, a neural network can be represented by a nonlinear function as

$$\mathbf{Y} = \mathbf{f}(\mathbf{X}, \mathbf{W}, \mathbf{b}), \tag{13}$$

where $\mathbf{f}$ is the neural network model, $\mathbf{X}$ is the input, $\mathbf{W}$ and $\mathbf{b}$ are the concatenated weights and biases, and $\mathbf{Y}$ is the output. In this paper, we assume $\mathbf{X} = \big[\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N\big]$ with the corresponding output $\mathbf{Y} = \big[\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_N\big]$, where $\mathbf{x}_i$ is of dimension $D \times 1$ and $\mathbf{y}_i$ is of dimension $D' \times 1$, with $i \in \{1, \cdots, N\}$. The dimension of $\mathbf{X}$ and $\mathbf{Y}$ are, therefore, $D \times N$ and $D' \times N$, respectively.

For clarity, we use $\mathbf{x}$ to denote an arbitrary input that is fed into a neural network $\mathbf{f}(\cdot)$. It is then passed through the network layer by layer. In this paper, a layer is defined by 'input $\rightarrow$ linear-combination $\rightarrow$ nonlinear-activation'. The output from the nonlinear-activation will be the 'input' to the next layer. Equivalently, the process can be represented as

$$\mathbf{a}_i^{(l+1)} = b_i^{(l+1)}\mathbf{1} + \sum_{j=1}^{C(l)} W_{i,j}^{(l+1)} \mathbf{x}_j^{(l)}, \tag{14}$$

and

$$\mathbf{x}_j^{(l)} = \phi\big(\mathbf{a}_j^{(l)}\big), \tag{15}$$

where $W_{i,j}^{(l+1)}$ indicates the weight that transforms the $j$-th element $\mathbf{x}_j^{(l)}$ from layer $l$ to the $i$-th element in layer $l+1$, resulting in $\mathbf{a}_i^{(l+1)}$ after linear combination. The nonlinear activation function is denoted by $\phi(\cdot)$. The bold $\mathbf{1}$ is a unit vector, $b_i^{(l+1)}$ denotes the bias, and $C(l)$ is the length of the column vector $\mathbf{x}_j^{(l)}$. Suppose that the $l = 1, \cdots, L-1$ layers are defined by (14) and (15), then the last layer $L$ is represented by

$$\mathbf{a}_i^{(L)} = b_i^{(L)}\mathbf{1} + \sum_{j=1}^{C(L-1)} W_{i,j}^{(L)} \mathbf{x}_j^{(L-1)}, \tag{16}$$

and

$$\mathbf{x}_j^{(L)} = \varphi\big(\mathbf{a}_j^{(L)}\big), \tag{17}$$

where $\varphi(\cdot)$ represents the activation function for layer $L$ as it is usually different from previous layers dependent on the task, e.g., classification or regression.

In traditional neural networks, both $W_{i,j}^{(l)}$ and $b_i^{(l)}$ are assumed to be deterministic. However, in the BNN paradigm, both $W_{i,j}^{(l)}$ and $b_i^{(l)}$ are allowed to be random variables, which enables Bayesian uncertainty quantification for neural networks. For brevity, we assume that $W_{i,j}^{(l)}$ is governed by a Gaussian distribution, which is shown in (18). The same rule applies to $b_i^{(l)}$ as well as given in (19).

$$W_{i,j}^{(l)} \sim \mathcal{N}\big(m_{w_{i,j}^{(l)}}, \sigma^2_{w_{i,j}^{(l)}}\big), \tag{18}$$

$$b_i^{(l)} \sim \mathcal{N}\big(m_{b_i^{(l)}}, \sigma^2_{b_i^{(l)}}\big), \tag{19}$$

where $m_{w_{i,j}^{(l)}}$ and $m_{b_i^{(l)}}$ are the respective means of $W_{i,j}^{(l)}$ and $b_i^{(l)}$. The corresponding variances are denoted by $\sigma^2_{w_{i,j}^{(l)}}$ and $\sigma^2_{b_i^{(l)}}$. The random weights and biases are then passed through the network, propagating uncertainty to the final outputs. The left side of Fig. 1 shows a general BNN, with a detailed substructure shown in the left side. Those curves are the bias and weight distributions.

## V. Cubature Approximation of Statistical Moments

### A. Cubature rules for Mean and Variance Approximation

Although we assume both the weights and the biases are subject to Gaussian distributions, it remains a challenge to capture the distributions after the nonlinear activation function. In this paper, we exploit the cubature rule to approximate the first and second moments of the distribution after nonlinear activation function [18].

Taking (14, 15, 18, and 19) and the Central Limit Theorem (CLT) into consideration, we know that $\mathbf{a}_i^{(l)}$ follows a Gaussian distribution as well when the network is rather wide [22] or deep [23]. Wu et al. [24] also empirically demonstrate that the claim is approximately valid even when (weak) correlations appear between the elements of $\phi(\cdot)$ during training. The CLT could be violated when a network is not wide(deep) enough, or the sample numbers are limited. VI becomes a powerful tool to approximation the distributions in such scenarios. In this paper, given the assumption that all the weights and biases are independent Gaussian random variables, we have the mean and variance of $\mathbf{a}_i^{(l)}$ as

$$\mathbb{E}\big(\mathbf{a}_i^{(l)}\big) = \langle \mathbf{a}_i^{(l)} \rangle = m_{b_i^{(l)}}\mathbf{1} + \sum_{j=1}^{C(l-1)} m_{w_{i,j}^{(l)}} \mathbf{x}_j^{(l-1)}, \tag{20}$$

$$\mathbb{V}\big(\mathbf{a}_i^{(l)}\big) = \langle \mathbf{a}_i^{(l)}, \mathbf{a}_i^{(l)} \rangle = \sigma^2_{b_i^{(l)}} + \sum_{j=1}^{C(l-1)} \Big[\mathbf{x}_j^{(l-1)} \sigma_{w_{i,j}^{(l)}}\Big]^2. \tag{21}$$

$$\begin{aligned}
&\mathbb{E}\big(\phi(\mathbf{a}_i^{(l)})\big) \\
&= \frac{1}{\sqrt{2\pi \det\Big(\mathbb{V}\big(\mathbf{a}_i^{(l)}\big)\Big)}} \int \phi(\alpha) \exp\Big[-\frac{\Big(\alpha - \mathbb{E}\big(\mathbf{a}_i^{(l)}\big)\Big)^2}{2\mathbb{V}\big(\mathbf{a}_i^{(l)}\big)}\Big] d\alpha \\
&= \int \sqrt{\frac{\mathbb{V}\big(\mathbf{a}_i^{(l)}\big)}{\pi \det\Big(\mathbb{V}\big(\mathbf{a}_i^{(l)}\big)\Big)}} \phi(\alpha) \exp(-\beta^2) d\beta \\
&= \int g(\beta) \exp(-\beta^2) d\beta.
\end{aligned} \tag{22}$$

The linear combination $\mathbf{a}_i^{(l)}$ is then passed to a generic activation function $\phi(\cdot)$ as shown in (15). The mean of $\phi(\mathbf{a}_i^{(l)})$ can be written as (22), where $g(\beta) = \sqrt{\frac{\mathbb{V}\big(\mathbf{a}_i^{(l)}\big)}{\pi \det\Big(\mathbb{V}\big(\mathbf{a}_i^{(l)}\big)\Big)}}\phi(\alpha)$, and $\alpha = \sqrt{2\mathbb{V}\big(\mathbf{a}_i^{(l)}\big)}\beta + \mathbb{E}\big(\mathbf{a}_i^{(l)}\big)$.

The integration shown in (22) can be rather complex or even intractable when the dimension of $\alpha$ or $\beta$ increases. In this paper, the cubature rule is adopted to approximate the integral. Without loss of generality, a set of cubature points are chosen and denoted as

$$\boldsymbol{\xi}_i = \sqrt{\frac{2D}{2}}\big[\mathbf{I}_i, -\mathbf{I}_i\big], \tag{23}$$

where $D$ is the dimension of $\alpha$ or $\beta$, $\mathbf{I}_i$ is the $D$-dimensional unit matrix. According to [18], the cubature points of $\alpha$ need to be transformed and the results are denoted as

$$\mathbf{A}_i^{(l)} = \sqrt{\mathbb{V}\big(\mathbf{a}_i^{(l)}\big)}\boldsymbol{\xi}_i + \mathbb{E}\big(\mathbf{a}_i^{(l)}\big), \tag{24}$$

After propagating the transformed cubature points from (24) through the nonlinear activation function $\phi(\cdot)$, we obtain the propagated cubature points as

$$\mathbf{A}_i^{*(l)} = \phi\big(\mathbf{A}_i^{(l)}\big). \tag{25}$$

The integral in (22) can therefore be simplified as

$$\mathbb{E}\big(\phi(\mathbf{a}_i^{(l)})\big) \approx \frac{1}{2D}\sum_{i=1}^{2D} \mathbf{A}_i^{*(l)} := \hat{\mathbf{h}}_i^{(l)}, \tag{26}$$

and the covariance after nonlinear activation function approximated as

$$\mathbb{V}\big(\phi(\mathbf{a}_i^{(l)})\big) \approx \frac{1}{2D}\sum_{i=1}^{2D} \mathbf{A}_i^{*T}\mathbf{A}_i^{*} - \hat{\mathbf{h}}_i^{(l)T}\hat{\mathbf{h}}_i^{(l)}. \tag{27}$$

Putting (15), (20), and (26) together, we know that $\mathbf{x}_i^{(l)}$ can be approximated by $\hat{\mathbf{h}}_i^{(l)}$ for $i = 2, \cdots, L$. Therefore, an approximation of (21) can also be obtained.

So far, we used the cubature rule to approximate the complex integral emerging in BNNs with summations involving a set of cubature points. Subsequently, the computational complexity reduces to $\mathcal{O}(D)$. In this paper, the hyperbolic function

$$\tanh x = \frac{e^{2x} - 1}{e^{2x} + 1} \tag{28}$$

is taken as an example nonlinear activation function to configure the uncertainties on the final results.

## VI. Experiments and Analysis

To assess the proposed approach in quantifying uncertainties in BNNs, a Bayesian neural network, as shown in Fig. 1, is employed to fulfill the regression task of a cosine function,

$$y = \cos(x) + v, \tag{29}$$

where $x$ is the input, $y$ is the output, and $v \sim \mathcal{N}(0, \sigma)$. As a simple two-layer network with nonlinear activation

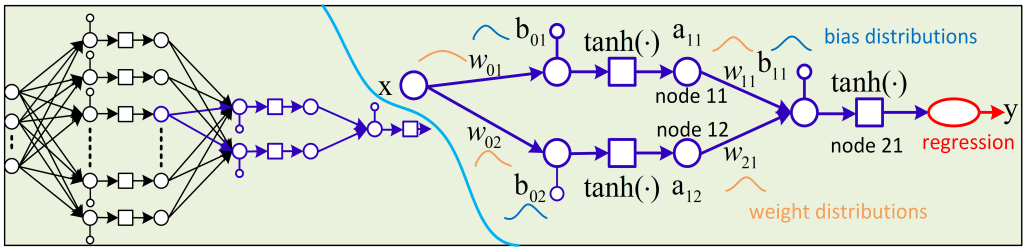


Fig. 1. The structure of a Bayesian neural network. The curves indicate bias and weight distributions.

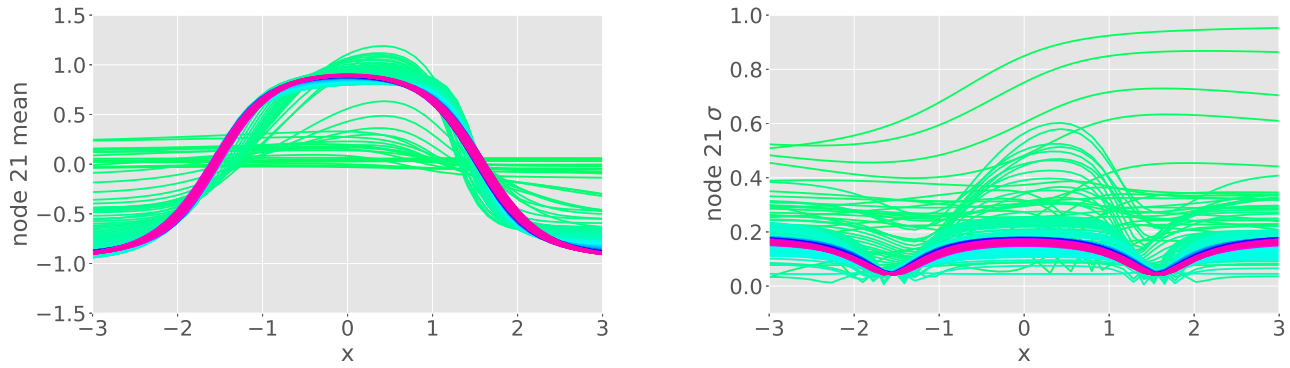


Fig. 2. Approximated mean and standard deviation of layer two of our approach: (a) approximated node 21 mean, (b) approximated node 21 standard deviation. The results from iteration 1 to iteration 500 are marked by the curves. The densely plotted curves show results when converged.

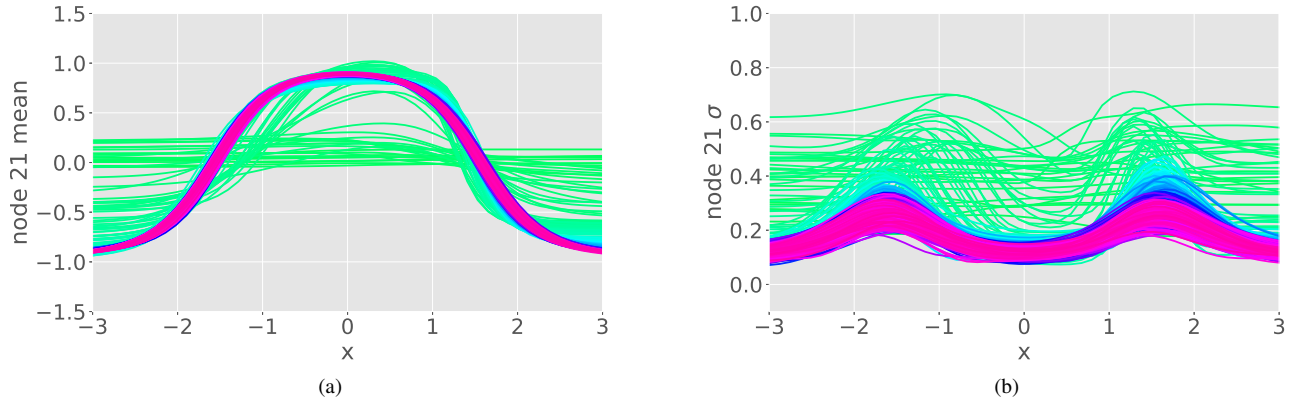


Fig. 3. Approximated mean and standard deviation of layer two of typical BNNs: (a) approximated node 21 mean, (b) approximated node 21 standard deviation. The results from iteration 1 to iteration 500 are marked by the curves. The densely plotted curves show results when converged.

function is considered a universal approximator [25], we use the network shown in the right side of Fig. 1 as an example. The experiments are implemented based on the probabilistic models research toolbox Edward, and $q(z_j, ; \boldsymbol{\lambda}_j)$ in (12) are set to be Gaussian.

In our experiment, as the input is one dimensional, cubature points become $\mathbf{I} = [-1, 1]$. These points are then transformed by (24) and propagated though the network. Mean and standard deviation of each node in each layer are computed using (26) and (27), respectively. In our case, two layers and three nodes are involved, as shown in Fig. 1. Weights and biases are initialised to follow Gaussian distributions with zero mean and variance 5.0 for all parameters. VI is employed to approximate these distributions. In our case, the network trained for 500 iterations so that observable convergence of all approaches is ensured. Within each iteration, VI is performed to optimise the network parameters. During training, fifty input-output pairs from (28) with $x$ in between $[-3.0, 3.0]$ are taken as samples. After each iteration, fifty means and deviations are recorded corresponding to each input-output pair. In our approach, mean and standard deviation are computed layer by layer following (23) to (27). We take results from a

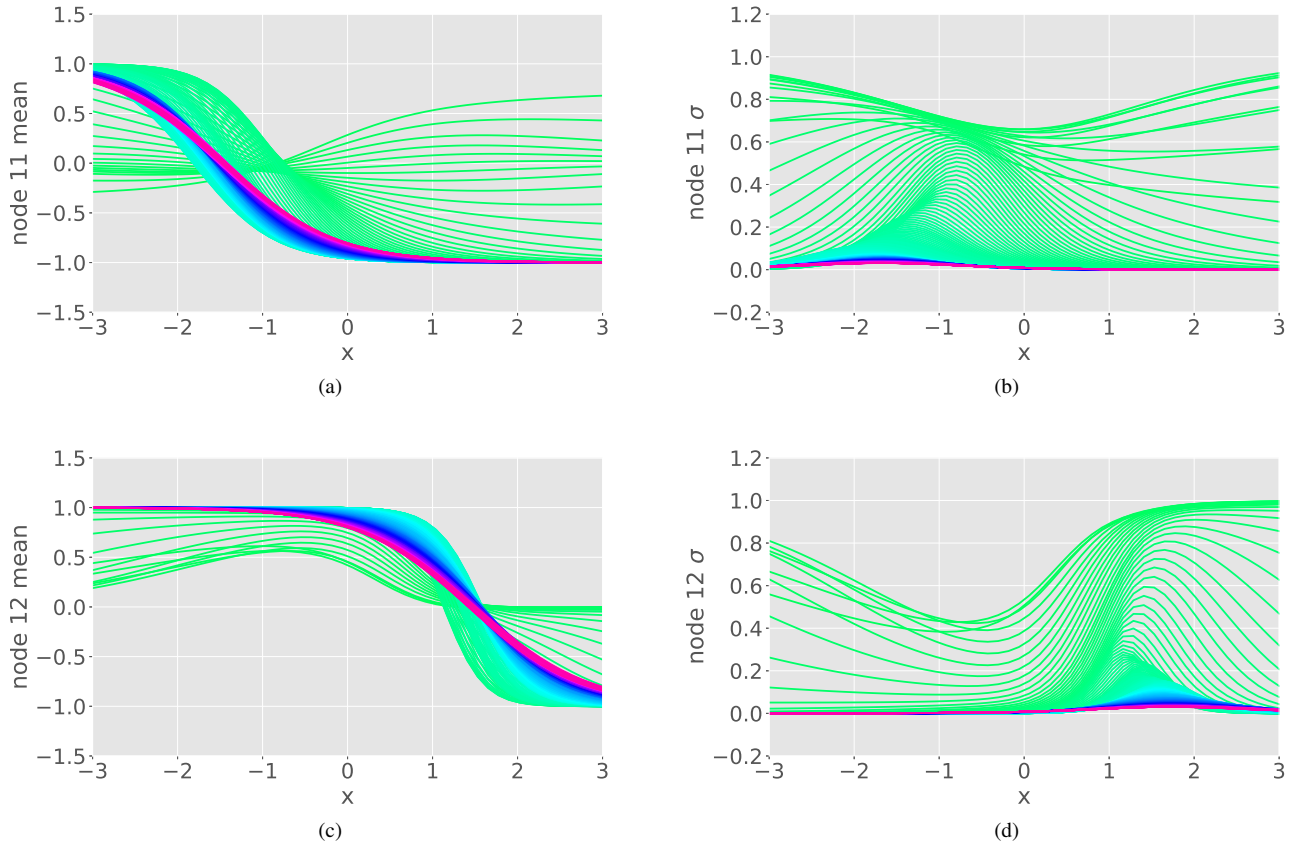


Fig. 4. Approximated mean and standard deviation of layer one with our approach: (a) approximated mean of node 11, (b) approximated variance of node 11, (c) approximated mean of node 12, (d) approximated variance of node 12. The results from iteration 1 to iteration 500 are marked by the curves. The densely plotted curves show results when converged.

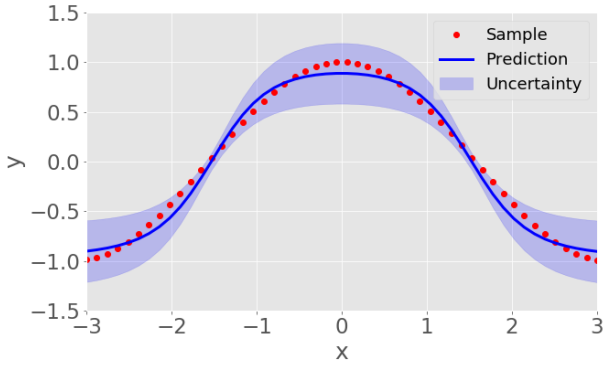


Fig. 5. One example of the approximated mean and the corresponding $2\sigma$ uncertainty of our approach

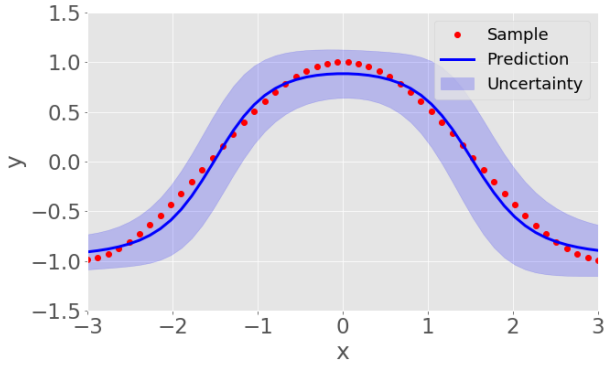


Fig. 6. One example of the approximated mean and the corresponding $2\sigma$ uncertainty of typical BNNs

typical BNN with the same structure for comparison [26]. The difference is that for the typical BNN, after each iteration, 20 samples of the weights and biases for each node are sampled to compute the layer-wise mean and standard deviation. In this paper, we also apply a regression unit (as shown in ellipse in Fig. 1) at the end of the network to improve the accuracy. In our case, a linear regression is applied.

Fig. 2 and Fig. 3 show the mean and standard deviation approximations of node 21 obtained from the 500 iterations, using the proposed approach and a typical BNN. We can see that both are able to approximate the cosine function by the statistical means in between $[-3.0, 3.0]$. The standard deviations that capture the uncertainties decrease as the iteration increases. We can see that both the mean and the

standard deviation from our approach, shown in Fig. 2(a) and 2(b), are 'narrower' than the results from a typical BNN as shown in Fig. 3(a) and 3(b) (the densely plotted curves (in red in electronic version) are actually results from the iterations after convergence). In particular, the standard deviation of the cubature-based approach is much narrower. This feature can be regarded as a 'metric' of convergence rate. As after a certain iteration, the proposed approach converges to a certain value (standard deviation), while these from the comparison approach still scatter in a wider range (the densely plotted curves (in red in electronic version)) than the proposed method. We have also shown the results of our approach from node 11 and node 12 in Fig. (4), to demonstrate that layer-wise uncertainty quantification can be achieved.

To make the results easier to understand, we have chosen the mean and the standard deviation of the two approaches from the last iteration for further demonstration. The samples, predictions (mean), and the $2\sigma$ uncertainty intervals from the two approaches are shown in Fig. 5 and Fig. 6, respectively. We can see that our approach shows more 'diverse' uncertainty quantification results, while the typical BNN shows more 'equally distributed' uncertainty quantification on all the predictions. The 'equally distributed' case is easy to understand because the mean and standard deviation are computed from the samples. We think the 'diverse' phenomenon is due to the fact that the expectations and variances used to propagate the cubature points in (24) and (25) vary, which actually is more informative than 'equally distributed' cases.

To conclude, we have approximated the nonlinear integrals involved in BNNs by propagating a set of cubature points, which achieves faster convergence rate and more 'diverse' uncertainty quantification.

## VII. Conclusion

We proposed to quantify uncertainties of Bayesian neural networks with an approach inspired by the cubature rule. With which, we can select a set of cubature points to propagate through the network and propagate the uncertainties of the network across the layers. As the number of cubature points is fairly small, it makes the approximation process very efficient when compared with sampling-based mean and standard deviation approximation approaches. A simulation is done to assess the performance of the proposed approach. The results show that cubature-based uncertainty quantification in BNNs converges faster while still achieving more 'diverse' uncertainty quantification results than a typical Bayesian neural network.

## References


[1] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[2] Y. Kim, P. Wang, and L. Mihaylova, "Scalable learning with a structural recurrent neural network for short-term traffic prediction," *IEEE Sensors Journal*, vol. 19, no. 23, pp. 11 359–11 366, 2019.

[3] Y. Kim, P. Wang, Y. Zhu, and L. Mihaylova, "A capsule network for traffic speed prediction in complex road networks," in *Proceedings of the 2018 Sensor Data Fusion: Trends, Solutions, Applications (SDF)*. IEEE, 2018, pp. 1–6.

[4] Y. Kim, P. Wang, and L. Mihaylova, "Structural recurrent neural network for traffic speed prediction," in *Proceedings of the ICASSP 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 5207–5211.

[5] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proceedings of the Advances in neural information processing systems*, 2012, pp. 1097–1105.

[6] Y. LeCun *et al.*, "Lenet-5, convolutional neural networks," *URL: http://yann. lecun. com/exdb/lenet*, vol. 20, p. 5, 2015.

[7] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1–9.

[8] P. Müller and D. R. Insua, "Issues in Bayesian analysis of neural network models," *Neural Computation*, vol. 10, no. 3, pp. 749–770, 1998.

[9] R. M. Neal, "Bayesian learning via stochastic dynamics," in *Proceedings of the Advances in neural information processing systems*, 1993, pp. 475–482.

[10] Y. Kwon, J. Won, B. J. Kim, and M. C. Paik, "Uncertainty quantification using Bayesian neural networks in classification: Application to ischemic stroke lesion segmentation," 2018.

[11] D. Dera, G. Rasool, and N. Bouaynaya, "Extended variational inference for propagating uncertainty in convolutional neural networks," in *Proceedings of the 2019 IEEE 29th International Workshop on Machine Learning for Signal Processing (MLSP)*. IEEE, 2019, pp. 1–6.

[12] W. Roth and F. Pernkopf, "Variational inference in neural networks using an approximate closed-form objective," in *Proceedings of the NIPS 2016 Workshop on Bayesian Deep Learning*, 2016.

[13] Y. Gal and Z. Ghahramani, "Dropout as a Bayesian approximation: Representing model uncertainty in deep learning," in *Proceedings of the International Conference on Machine Learning*, 2016, pp. 1050–1059.

[14] L. Chang, B. Hu, A. Li, and F. Qin, "Transformed unscented Kalman filter," *IEEE Transactions on Automatic Control*, vol. 58, no. 1, pp. 252–257, 2012.

[15] P. Wang, Q. Zhang, and Z. Chen, "A grey probability measure set based mobile robot position estimation algorithm," *International Journal of Control, Automation and Systems*, vol. 13, no. 4, pp. 978–985, 2015.

[16] Q. Zhang, P. Wang, and Z. Chen, "An improved particle filter for mobile robot localization based on particle swarm optimization," *Expert Systems with Applications*, vol. 135, pp. 181–193, 2019.

[17] P. Wang, Y. Kim, L. Vaci, H. Yang, and L. Mihaylova, "Short-term traffic prediction with vicinity Gaussian process in the presence of missing data," in *Proceedings of the 2018 Sensor Data Fusion: Trends, Solutions, Applications (SDF)*. IEEE, 2018, pp. 1–6.

[18] I. Arasaratnam and S. Haykin, "Cubature Kalman filters," *IEEE Transactions on automatic control*, vol. 54, no. 6, pp. 1254–1269, 2009.

[19] D. M. Blei, A. Kucukelbir, and J. D. McAuliffe, "Variational inference: A review for statisticians," *Journal of the American statistical Association*, vol. 112, no. 518, pp. 859–877, 2017.

[20] D. G. Tzikas, A. C. Likas, and N. P. Galatsanos, "The variational approximation for bayesian inference," *IEEE Signal Processing Magazine*, vol. 25, no. 6, pp. 131–146, 2008.

[21] R. A. Leibler and S. Kullback, "On information and sufficiency," *Annals of Mathematical Statistics*, vol. 22, no. 1, pp. 79–86, 1951.

[22] A. G. D. G. Matthews, M. Rowland, J. Hron, R. E. Turner, and Z. Ghahramani, "Gaussian process behaviour in wide deep neural networks," *arXiv preprint arXiv:1804.11271*, 2018.

[23] J. Lee, Y. Bahri, R. Novak, S. S. Schoenholz, J. Pennington, and J. Sohl-Dickstein, "Deep neural networks as Gaussian processes," *arXiv preprint arXiv:1711.00165*, 2017.

[24] A. Wu, S. Nowozin, E. Meeds, R. E. Turner, J. M. Hernández-Lobato, and A. L. Gaunt, "Deterministic variational inference for robust bayesian neural networks," *arXiv preprint arXiv:1810.03958*, 2018.

[25] M. Stinchcombe, K. Hornik, and H. White, "Universal approximation of an unknown mapping and its derivatives using multilayer feedforward networks," *Neural Networks*, vol. 3, no. 5, pp. 551–560, 1990.

[26] R. M. Neal, *Bayesian learning for neural networks*. Springer Science & Business Media, 2012, vol. 118.