14th International Conference on Applications of Statistics and Probability in Civil Engineering, ICASP14
Dublin, Ireland, July 9-13, 2023

Bayesian Neural Networks for Probabilistic Surrogate Models – Uncertainty Quantification, Propagation, and Sensitivity Analysis

James-A. Goulet
Professor, Dept. of Civil Engineering, Polytechnique Montréal, Montréal, Canada

Luong-Ha Nguyen
Research associate, Dept. of Civil Engineering, Polytechnique Montréal, Montréal, Canada

ABSTRACT: Neural networks are powerful function approximators which scale to problems having large input and output dimensionalities. Bayesian neural networks (BNN) are an interesting choice for surrogate models as they (1) natively enable performing sensitivity analyses by quantifying the derivative of the function's output with respect to its inputs, (2) are able to quantify heteroscesdastic model prediction uncertainties, (3) are able to quantify the epistemic uncertainties associated with parameters. However, a main limitation of typical BNN is that they do not allow propagating uncertainties analytically from the model inputs to its outputs. The tractable approximate Gaussian inference method (TAGI) solves this issue by enabling to propagate uncertainties analytically from the model inputs to its outputs, making it suited to be used as a surrogate model for probabilistic setups. One key limitation of TAGI is that, up to now, it relied on locally linearized activation functions. The result of that approximation is that the input uncertainties only affect the output variances without modifying the associated expected values, and these variances are only accurate for small magnitudes of input-uncertainties. The objectives of this paper are twofold: first it is to introduce the TAGI method for Bayesian neural networks to the surrogate modelling community and second, to present a new method based on a mixture of truncated Gaussians to replace the local linearization in order to accurately propagate uncertainties through neural networks.

Neural networks (NN) are powerful function approximators (Goodfellow et al., 2016). A key property is that unlike other surrogate modelling (SM) methods such as *polynomial chaos expansion* (PCE) (Sudret, 2008) and Gaussian process regression (GPR) or Kriging (Rasmussen and Williams, 2006), its computational complexity is primarily controlled by the number of hidden units employed, which in turns controls the model capacity. In typical neural networks, the computational complexity is linear with respect to the number of weight parameters in the network (Goulet et al., 2021). This enables scaling NN beyond the capacity of PCE and GPR, to problems having large dimensionalities as it easily extents from a handful to thousands of input as well as output values. Neural networks are an interesting choice for surrogate models as they natively enable performing sensitivity analyses by quantifying the derivative of the function's output with respect to its inputs (Margossian, 2019; Schröder et al., 2020). Backprop-based neural networks, have been shown to be able to quantify heteroscesdastic model prediction uncertainties (Izmailov et al., 2020; Maddox et al., 2019; Wu et al., 2019; Gupta and Nagar, 2018). This aspect is particularly important in order to quantify the surrogate model prediction accuracy as a function of the input values. One key limitation for the neural networks that are relying on the gradient backpropagation method (Rumelhart et al., 1986) for learning the network parameters is that they do not allow propagating uncertainties analytically from the model inputs to its outputs. Instead, they need to either rely on Monte-Carlo (MC) sampling (Lieu et al., 2022) to be trained while providing labelled examples for the output variance (Tripathy and Bilionis, 2018), or rely on linear approximations (Schröder et al., 2020).

In the general framing, Bayesian neural networks (BNN) have the particularity of considering the epistemic uncertainties by quantifying the posterior probability density function (PDF) for the weight and bias on a NN (Neal, 2012). Existing methods rely on approximate techniques such as MC sampling (Neal, 2012), Variational inference (Wu et al., 2019), and the Laplace approximation (Daxberger et al., 2021). Some alternative methods do not quantify the posterior over weights and instead only focus on capturing the predictive uncertainty by ensembling several models. Examples of such methods are MC dropout (Gal and Ghahramani, 2016) and ensembles (Barber and Bishop, 1998). Despite being able to better capture the predictive uncertainty than their non-Bayesian counterpart, they are also typically order of magnitudes slower and are still unable to propagate uncertainties analytically from the model inputs to its outputs.

A new method that tackle these limitations is the tractable approximate Gaussian inference method (TAGI) (Goulet et al., 2021) that enable performing closed-form analytical prediction and inference for BNN while being orders of magnitude faster than state-of-the-art BNN approaches (Deka, 2022). In addition, TAGI enables propagating uncertainties analytically from the model inputs to its outputs making it suited to be used as a SM for probabilistic setups. In addition to enabling analytical parameter inference, TAGI enables performing input-values inference such that for a trained network, we can analytically infer the input such they match specified output values (Nguyen and Goulet, 2022a). This aspect is relevant to the context of SM where one wants to optimize model parameters for reaching desired outputs. One key limitation of TAGI is that, up to now, it relied on locally linearized activation functions (Goulet et al., 2021). The result of that approximation is that (1) the input uncertainties only affect the output variances without modifying the output expected values, and (2) the output variances are only accurate for small magnitudes of input-uncertainties.

The objectives of this paper are twofold: first it is to introduce the TAGI method for Bayesian neural network to the surrogate modelling community, and second, to present a new method based on a mixture of truncated Gaussians to replace the local linearization and thus solve the two current limitations associated with it while keeping all other desirable properties allowing performing sensitivity analyses as well as quantifying heteroscedastic prediction uncertainties. The structure of the paper is the following: Section 2 presents a review of the TAGI method, Section 3 presents the mixture rectified linear activation unit (mReLU) proposed in this paper and section 4 presents experiments comparing the performance at propagating uncertainties for the existing and the new approach.

1. Tractable Approximate Gaussian Inference (TAGI)

Figure 1 presents a directed acyclic graph (DAG) describing the interconnectivity of a feedforward neural network architecture having $\mathtt{L}$ hidden layers $\boldsymbol{z}^{(j)}$ each consisting of $\mathtt{A}$ hidden units, $\mathtt{X}$ inputs and $\mathtt{Y}$ output variables. TAGI assumes that the joint distribution between the observations $\boldsymbol{y}$ and a neural network's parameters $\boldsymbol{\theta} = \{\boldsymbol{w}, \boldsymbol{b}\}$ is approximated

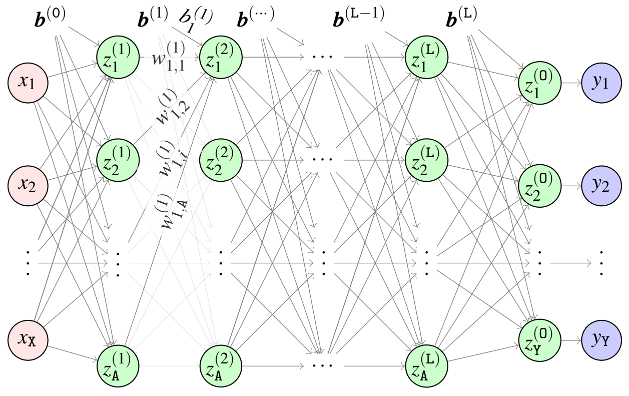


Figure 1: DAG representations of the variable nomenclature associated with feedforward neural networks.

by a multivariate Gaussian distribution,

$$f\begin{pmatrix}\boldsymbol{\theta}\\ \boldsymbol{y}\end{pmatrix} = \mathcal{N}\left(\begin{bmatrix}\boldsymbol{\theta}\\ \boldsymbol{y}\end{bmatrix};\begin{bmatrix}\boldsymbol{\mu}_\theta\\ \boldsymbol{\mu}_Y\end{bmatrix},\begin{bmatrix}\boldsymbol{\Sigma}_\theta & \boldsymbol{\Sigma}_{Y\theta}^\intercal\\ \boldsymbol{\Sigma}_{Y\theta} & \boldsymbol{\Sigma}_Y\end{bmatrix}\right),$$

so that the parameter inference can build upon the Gaussian conditional equation describing the probability density function (PDF) of $\boldsymbol{\theta}$ conditional on observations $\boldsymbol{y}$,

$$\begin{array}{rcl} f(\boldsymbol{\theta}|\boldsymbol{y}) & = & \mathcal{N}(\boldsymbol{\theta};\boldsymbol{\mu}_{\theta|y},\boldsymbol{\Sigma}_{\theta|y})\\ \boldsymbol{\mu}_{\boldsymbol{\theta}|y} & = & \boldsymbol{\mu}_{\boldsymbol{\theta}} + \boldsymbol{\Sigma}_{\boldsymbol{Y\theta}}^\intercal\boldsymbol{\Sigma}_{\boldsymbol{Y}}^{-1}(\boldsymbol{y}-\boldsymbol{\mu}_{\boldsymbol{Y}})\\ \boldsymbol{\Sigma}_{\boldsymbol{\theta}|y} & = & \boldsymbol{\Sigma}_{\boldsymbol{\theta}} - \boldsymbol{\Sigma}_{\boldsymbol{Y\theta}}^\intercal\boldsymbol{\Sigma}_{\boldsymbol{Y}}^{-1}\boldsymbol{\Sigma}_{\boldsymbol{Y\theta}}.\end{array}$$

The approach is inherently divided in two steps; first propagate uncertainties through the network in order to obtain the joint PDFs between the quantities to be updated (i.e., neural network's parameters and hidden state units) and the observations, and then update these quantities.

The first key operation to be considered is the propagation of uncertainty from the activation units $\boldsymbol{A}^{(j)} \sim \mathcal{N}(\boldsymbol{\mu}_{\boldsymbol{A}}^{(j)},\boldsymbol{\Sigma}_{\boldsymbol{A}}^{(j)})$ of a hidden layer j to a hidden unit $Z_i^{(j+1)}$ on the subsequent layer $j+1$

$$Z_i^{(j+1)} = \sum_{k=1}^{\mathtt{A}} W_{i,k}^{(j)} A_k^{(j)} + B_i^{(j)}, \qquad (1)$$

where $W_{i,k}^{(j)}$ are weights and $B_i^{(j)}$ the bias parameters that are modelled by Gaussian random variables. In order to maintain the analytical tractability of equation 1, TAGI approximates the product of any pair of weight and activation unit by a Gaussian random variable $WA \approx \mathcal{N}(\mu_{WA},\sigma_{WA}^2)$, for which the exact moments can be computed analytically using *Gaussian multiplicative approximation* (GMA) (Goulet et al., 2021; Deka et al., 2021). The second key operation is the propagation of uncertainty through non-linear activation functions

$$A_i^{(j+1)} = \psi\left(Z_i^{(j+1)}\right), \qquad (2)$$

where, in order to maintain the analytical tractability, TAGI locally linearize $\psi(\cdot)$ at the expected value of the hidden units $\mu_{Z_i}^{(j+1)}$. In practice it is most common to employ the rectified linear activation function (ReLU) (Goodfellow et al., 2016) which simply consists in

$$\psi_{\mathrm{R}}(z) = \max(0,z),$$

so that the local linearization results in

$$\begin{array}{lll} \mu_A = 0, & \sigma_A = 0 & \text{if } \mu_z \leq 0\\ \mu_A = \mu_Z, & \sigma_A = \sigma_Z & \text{if } \mu_z > 0.\end{array} \qquad (3)$$

Maintaining the computational tractability of equations 1 and 2 requires assuming diagonal covariance structures for the hidden units among a same layer $\boldsymbol{\Sigma}_{\boldsymbol{Z}}^{(j)}$, and for the parameters $\boldsymbol{\Sigma}_{\boldsymbol{\theta}}$, yet in the case where we want to propagate uncertainties from the input to the output layer, one needs to consider the full covariance $\boldsymbol{\Sigma}_{\boldsymbol{Z}}^{(j)}$ within each hidden layer j.

The update step, i.e., the Gaussian conditional inference step, is performed using a recursive layer-wise procedure; Using the shorthand notation $\{\boldsymbol{\theta}^+,\boldsymbol{Z}^+\} \equiv \{\boldsymbol{\theta}^{(j+1)},\boldsymbol{Z}^{(j+1)}\}$ and $\{\boldsymbol{\theta},\boldsymbol{Z}\} \equiv \{\boldsymbol{\theta}^{(j)},\boldsymbol{Z}^{(j)}\}$, the posteriors for the parameters and hidden states are computed following

$$\begin{array}{rcl} f(\boldsymbol{Z}|\boldsymbol{y}) & = & \mathcal{N}(z;\boldsymbol{\mu}_{\boldsymbol{Z}|\boldsymbol{y}},\boldsymbol{\Sigma}_{\boldsymbol{Z}|\boldsymbol{y}})\\ \boldsymbol{\mu}_{\boldsymbol{Z}|\boldsymbol{y}} & = & \boldsymbol{\mu}_{\boldsymbol{Z}} + \mathbf{J}_{\boldsymbol{Z}}\left(\boldsymbol{\mu}_{\boldsymbol{Z}^+|\boldsymbol{y}} - \boldsymbol{\mu}_{\boldsymbol{Z}^+}\right)\\ \boldsymbol{\Sigma}_{\boldsymbol{Z}|\boldsymbol{y}} & = & \boldsymbol{\Sigma}_{\boldsymbol{Z}} + \mathbf{J}_{\boldsymbol{Z}}\left(\boldsymbol{\Sigma}_{\boldsymbol{Z}^+|\boldsymbol{y}} - \boldsymbol{\Sigma}_{\boldsymbol{Z}^+}\right)\mathbf{J}_{\boldsymbol{Z}}^\intercal\\ \mathbf{J}_{\boldsymbol{Z}} & = & \boldsymbol{\Sigma}_{\boldsymbol{ZZ}^+}\boldsymbol{\Sigma}_{\boldsymbol{Z}^+}^{-1},\end{array} \qquad (4)$$

$$\begin{array}{rcl} f(\boldsymbol{\theta}|\boldsymbol{y}) & = & \mathcal{N}(\boldsymbol{\theta};\boldsymbol{\mu}_{\boldsymbol{\theta}|\boldsymbol{y}},\boldsymbol{\Sigma}_{\boldsymbol{\theta}|\boldsymbol{y}})\\ \boldsymbol{\mu}_{\boldsymbol{\theta}|\boldsymbol{y}} & = & \boldsymbol{\mu}_{\boldsymbol{\theta}} + \mathbf{J}_{\boldsymbol{\theta}}\left(\boldsymbol{\mu}_{\boldsymbol{Z}^+|\boldsymbol{y}} - \boldsymbol{\mu}_{\boldsymbol{Z}^+}\right)\\ \boldsymbol{\Sigma}_{\boldsymbol{\theta}|\boldsymbol{y}} & = & \boldsymbol{\Sigma}_{\boldsymbol{\theta}} + \mathbf{J}_{\boldsymbol{\theta}}\left(\boldsymbol{\Sigma}_{\boldsymbol{Z}^+|\boldsymbol{y}} - \boldsymbol{\Sigma}_{\boldsymbol{Z}^+}\right)\mathbf{J}_{\boldsymbol{\theta}}^\intercal\\ \mathbf{J}_{\boldsymbol{\theta}} & = & \boldsymbol{\Sigma}_{\boldsymbol{\theta}\boldsymbol{Z}^+}\boldsymbol{\Sigma}_{\boldsymbol{Z}^+}^{-1}.\end{array} \qquad (5)$$

Note that the layer-wise recursive procedure defined in equations 4 and 5 only requires the storage of the joint prior PDFs for pairs of subsequent hidden layers and pairs of hidden layers, along with the parameters directly connecting into them. This allows maintaining the computational tractability of the uncertainty propagation and inference steps which scale linearly with respect to the number of weight parameters.

Despite the simplifying assumptions mentioned above, TAGI was shown to match the performance of feedforward neural networks (FNN) trained with backpropagation for regression, and for classification with convolutional neural networks (Nguyen and Goulet, 2021b), for image generation with generative adversarial networks (Nguyen and Goulet, 2021b), and for reinforcement learning with discrete and continuous actions (Nguyen and Goulet, 2021a, 2022a). In the following section, we will show how we can replace the local linearization procedure from equation 2 in order to allow propagating uncertainties accurately from the input to the output of the network.

2. Mixture Rectified Linear Activation Unit

The idea behind the *mixture rectified linear activation unit* (mReLU) is to propagate the hidden unit's uncertainty $Z \sim \mathcal{N}(\mu_Z, \sigma_Z^2)$ through a ReLU $\psi_{\mathrm{R}}(z)$ by using a Gaussian mixture between a truncated Gaussian $\tilde{Z}$ and a zero-valued component.

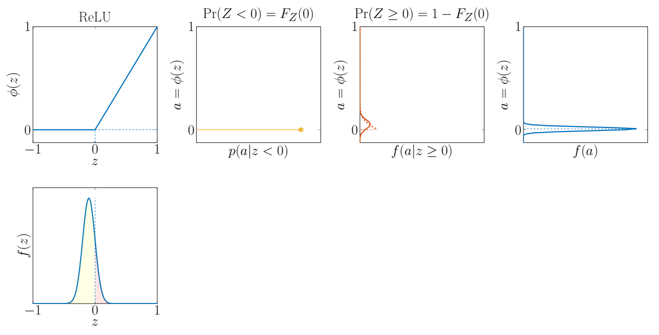
The expected value of a Gaussian truncated at $z \geq 0$ is

$$\begin{aligned}
\alpha &= -\frac{\mu_Z}{\sigma_Z} \\
\omega &= 1 - \Phi(\alpha) = \Pr(Z \geq 0) \\
\beta &= \frac{\phi(\alpha)}{\max(\delta, \omega)} \\
\tilde{\mu}_Z &= \mu_Z + \beta \sigma_Z,
\end{aligned}$$

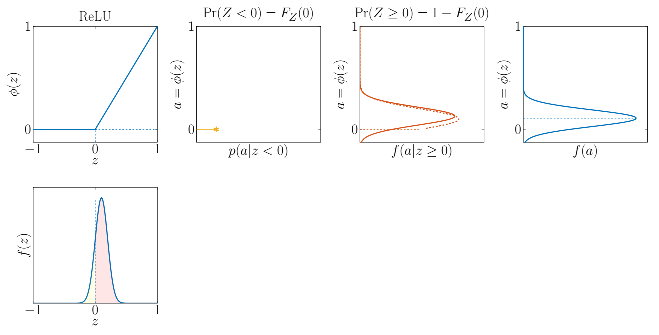
where, ϕ and Φ are the standard normal PDF and cumulative density function (CDF), and for practical cases, $\delta = 10^{-3}$ is a lower bound on the probability in order to avoid numerical errors associated with a possible division by 0 when calculating β. The variance of the truncated PDF is given by

$$\begin{aligned}
\kappa &= 1 + \alpha\beta - \beta^2 \\
\tilde{\sigma}_Z^2 &= \kappa \sigma_Z^2.
\end{aligned}$$

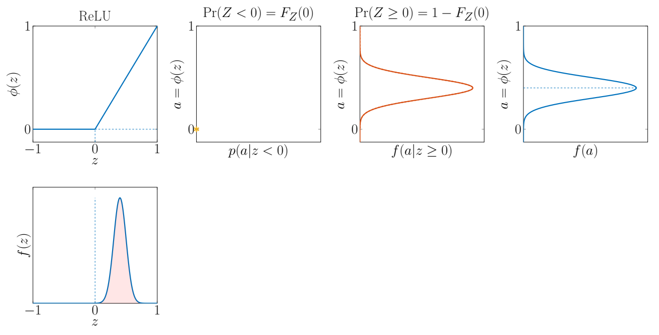
Figure 2 presents examples of such a truncation for different values of μ_Z. The first figure on the top rows presents the ReLU function itself, the second, the probability mass function (PMF) associated with $\Pr(Z < 0)$, the third presents the truncated PDFs for $z \geq 0$ as well as the Gaussian PDF whose moments match the truncated Gaussian, and the fourth presents the output Gaussian random variable obtained from the mixture (Runnalls, 2007) of the PMF for $\Pr(Z < 0)$ and the moment matching Gaussian for $\Pr(Z \geq 0)$. Figure 2a,b show how even for negative expected values, it results in non-zero mean and variance contrarily to the locally linearized ReLU presented in equation 3. Figure 2c shows that when the expected value is positive and much larger than the variance, the result is equivalent to the locally linearized ReLU.



(a) Example #1 – Small negative μ_Z



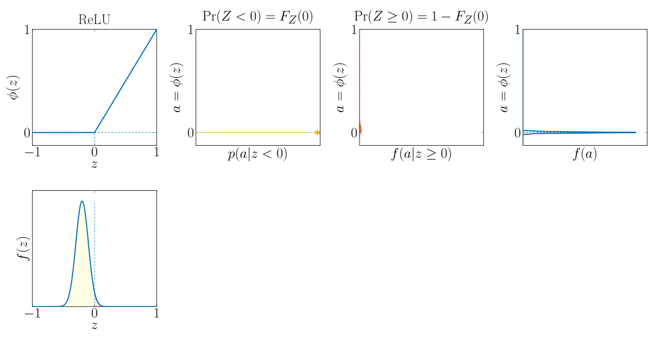
(b) Example #2 – Small positive μ_Z



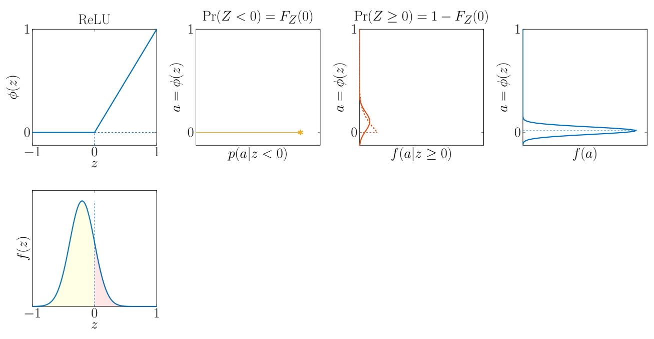
(c) Example #3 – Large positive μ_Z

Figure 2: Illustration of the effect of the hidden unit expected value μ_Z on the Mixture-ReLU.

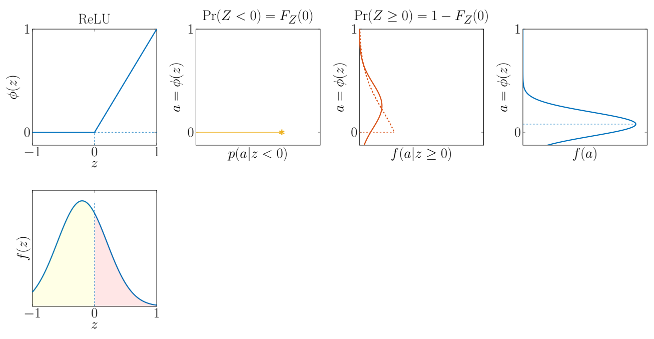
Figure 3 reproduces a similar experiment, this time while keeping μ_Z at a constant negative value while varying σ_Z. For all these cases, the locally linearized ReLU would have resulted in zero output mean and variance. With mReLU, we can see in (a) that as most of the probability content is in the negative domain, the output is highly concentrated around zero. Then for (b) and (c), as we increase σ_Z, the output mean get shifted towards positive values and the output variance increases.



(a) Example #1 – Small negative μ_Z & small σ_Z

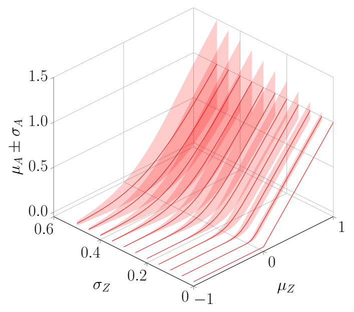


(b) Example #2 – Small negative μ_Z & moderate σ_Z

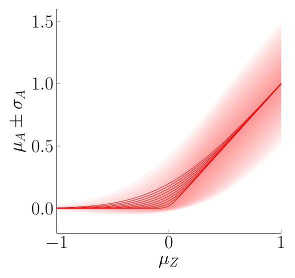


(c) Example #3 – Small negative μ_Z & large σ_Z

Figure 3: Illustration of the effect of the hidden unit variance σ_Z on the Mixture-ReLU



(a) 3D view



(b) 2D view

Figure 4: Illustration of the effect of the hidden unit expected value μ_Z and variance σ_Z on the Mixture-ReLU output moments $\{\mu_A, \sigma_A\}$.

The relationship between $\{\mu_Z, \sigma_Z\}$ and $\{\mu_A, \sigma_A\}$ is illustrated in Figure 4 where the special case where $\sigma_Z = 0$ is equivalent to the ReLU $\psi_{\mathrm{R}}(z)$. We can see how as σ_Z increase, the function value is smoothed and the output uncertainty increases.

The covariance between $\tilde{Z}$ and another variable H is given by

$$\mathrm{cov}(\tilde{Z}, H) = \kappa^{1/2}\mathrm{cov}(Z, H).$$

The probability $\Pr(Z > 0)$ is denoted by ω, so that the activation unit A is obtained from the mixture

$$A = \overline{\omega} \cdot 0 + \omega\tilde{Z},$$

for which the moment are

$$\begin{aligned} \mu_A &= \omega\tilde{\mu}_z \\ \sigma_A^2 &= \omega\tilde{\sigma}_Z^2 + \omega(1-\omega)\tilde{\mu}_Z^2 \\ \lambda &= (\omega\kappa)^{1/2} \\ \mathrm{cov}(A, H) &= \lambda\mathrm{cov}(Z, H). \end{aligned} \qquad (6)$$

The expected value and variance computed in Equation 6 are exact, however, the covariance is not because of the truncation. Figure 5 compares the covariance obtained from Equation 6 with the theoretical result obtained using MC simulations. We can see from Figure 5 that although the covariance is not computed exactly, the approximation closely mimic the theoretical values.

3. EXPERIMENTS

We explore the capacity of the locally linearized ReLU & mReLU functions to correctly propagate uncertainties through a neural network using a 1D

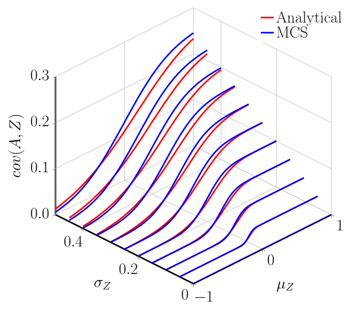


Figure 5: Comparison of the mReLU (red) output moments $\{\mu_A, \sigma_A, cov(A,Z)\}$ with the theoretical values (blue) as a function of μ_Z and σ_Z.

toy regression problem $y = g(x) = x^3 - x/2$. For that comparative setup, we use a feedforward fully-connected architecture with a single hidden layer having 50 hidden units. In each case, we first learn the function using synthetic observations so that the epistemic uncertainty associated with weights and biases become negligible. The reason behind this setup is that we want to look specifically at the effect of the input variable (x) uncertainty (σ_X) on the output without interference from the parameter's uncertainty. For that purpose, we consider $\mathbf{\Sigma_\theta} = \mathbf{0}$ and we query the trained network for different input values $X \sim \mathcal{N}(x, \sigma_X^2)$, where $\sigma_X \in \{0.05, 0.10, 0.25, 0.50\}$. The output expected value and variance are then compared with the theoretical values obtained using MC samples (10^5).

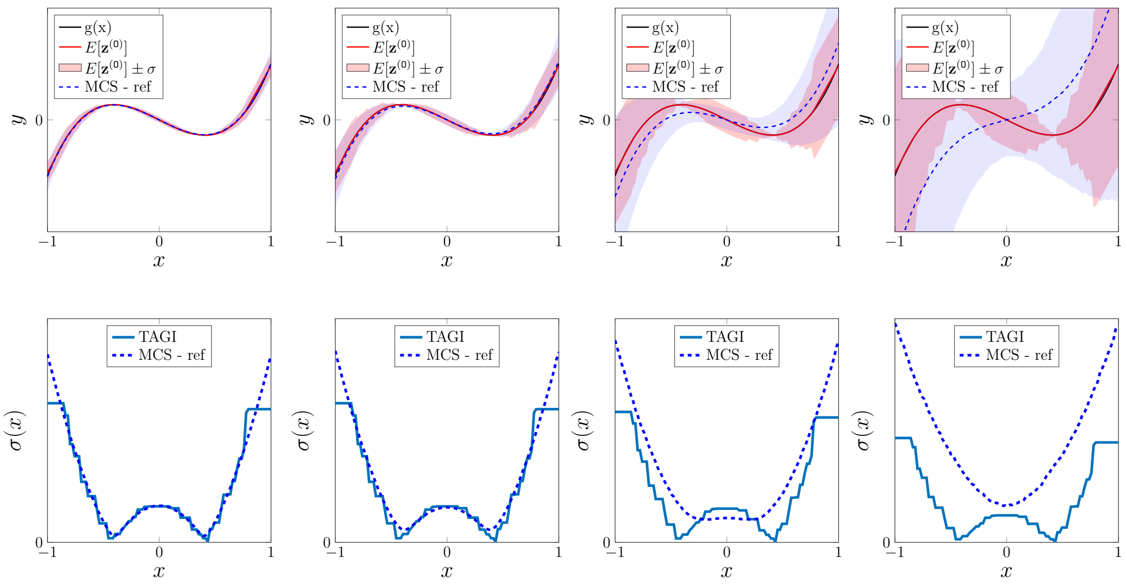
Figure 6 first present the results for the local linearized ReLU function. On the top row, we can directly see the predicted expected values in red with its $\pm\sigma$ coverage region, along with the theoretical values displayed in blue. Note that the NN predictions are almost coinciding with the true function $g(x)$ displayed in black. On the second row, we compare in solid blue the output standard deviation resulting from the input uncertainty associated with X, for TAGI in solid blue and MC simulations in dashed blue. We see that for small values of σ_X with respect to the changes in curvature of the function, the locally linearized ReLU matches the theoretical values. However, for larger σ_X values, the results do not match the theoretical values. Note how for $\sigma_X = 0.5$, the theoretical expected values depart from the function $g(x)$ whereas the output obtained from the locally linearized ReLU do not. The same thing is true for the output's standard deviation which goes to zero whenever the derivative of the function equals zero, which is not theoretically correct.

Figure 7 presents the results for the same experiment, this time for the mReLU. Again, for small values of σ_X, the mReLU matches the theoretical values. However, for larger σ_X values, we are now able to model the departure of the output expected value from the function $g(x)$. Although a discrepancy remains between predicted and theoretical values, the approximation is much better than with the locally linearized approximation. The same applies for the output standard deviation, which is highly accurate for small input uncertainties and then underestimate the output uncertainty as σ_X increases. Despite this approximation in the output uncertainty, note how contrarily to the locally linearized ReLU, the output's standard deviation does not go to zero whenever the derivative of the function equals zero.

The experiments conducted here show how the mReLU overcomes the limitations associated with the local linearization approach for propagating uncertainties through a neural network.

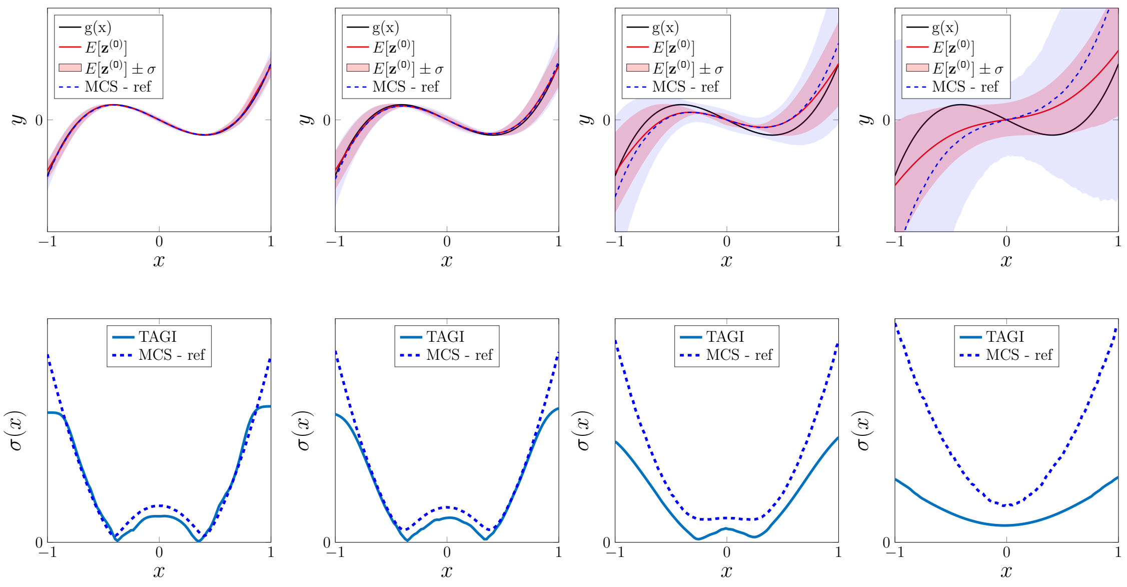
4. CONCLUSIONS

This paper introduces the possibilities that Bayesian neural networks can bring to the field of surrogate modelling. Up to now, surrogate models relying on neural networks had to rely on sampling methods or linear approximations in order to propagate input uncertainties through the network. This aspect has limited the potential of neural networks in comparison with other approaches such as polynomial chaos expansion. As presented in this paper, the tractable approximate Gaussian inference (TAGI) method using the mReLU proposed in this paper brings a paradigm shift by being the first method that allows propagating uncertainties analytically through a neural network. In addition to this unprecedented feature, TAGI preserves the capacity to quantify epistemic and heteroscedastic aleatory uncertainties, as well as to perform probabilistic sensitivity analysis through a closed-form formulation. Although much work remains to be done before BNN and TAGI can match the per-



(a) $\sigma_X = 0.05$ (b) $\sigma_X = 0.10$ (c) $\sigma_X = 0.25$ (d) $\sigma_X = 0.50$

Figure 6: Locally linearized ReLU modelling the effect of input-layer uncertainty (σ_X). The solid lines correspond to the TAGI method and the dashed lines are Monte-carlo analyses used as references.



(a) $\sigma_X = 0.05$ (b) $\sigma_X = 0.10$ (c) $\sigma_X = 0.25$ (d) $\sigma_X = 0.50$

Figure 7: mReLU modelling the effect of input-layer uncertainty (σ_X). The solid lines correspond to the TAGI method and the dashed lines are Monte-carlo analyses used as references.

formance of the existing state-of-the-art surrogate model methods, the scalability offered by BNN makes it an excellent candidate for the future investigations of complex SM problems. Users interested in experimenting with TAGI can rely on the Py/cuTAGI library (Nguyen and Goulet, 2022b) that include all the features introduced in this paper.

5. ACKNOWLEDGEMENTS

The second author is funded in part by a grant from Hydro-Quebec, IVADO, and the Natural Sciences and Engineering Research Council of

Canada.

6. References

Barber, D. and Bishop, C. M. (1998). "Ensemble learning in bayesian neural networks." *NATO ASI Series F Computer and Systems Sciences*, 168, 215–238.

Daxberger, E., Kristiadi, A., Immer, A., Eschenhagen, R., Bauer, M., and Hennig, P. (2021). "Laplace redux-effortless bayesian deep learning." *Advances in Neural Information Processing Systems*, 34, 20089–20103.

Deka, B. (2022). "Analytical bayesian parameter inference for probabilistic models with engineering applications." Ph.D. thesis, Polytechnique Montreal, Polytechnique Montreal.

Deka, B., Ha Nguyen, L., Amiri, S., and Goulet, J.-A. (2021). "The Gaussian multiplicative approximation for state-space models." *Structural Control and Health Monitoring*, e2904.

Gal, Y. and Ghahramani, Z. (2016). "Dropout as a Bayesian approximation: Representing model uncertainty in deep learning." *international Conference on Machine Learning*, PMLR, 1050–1059.

Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep learning*. MIT Press.

Goulet, J.-A., Nguyen, L. H., and Amiri, S. (2021). "Tractable approximate Gaussian inference for Bayesian neural networks." *Journal of Machine Learning Research*, 22(251), 1–23.

Gupta, A. K. and Nagar, D. K. (2018). *Matrix variate distributions*. Chapman and Hall/CRC.

Izmailov, P., Maddox, W. J., Kirichenko, P., Garipov, T., Vetrov, D., and Wilson, A. G. (2020). "Subspace inference for Bayesian deep learning." *Uncertainty in Artificial Intelligence*, PMLR, 1169–1179.

Lieu, Q., Nguyen, K., Dang, K., Lee, S., Kang, J., and Lee, J. (2022). "An adaptive surrogate model to structural reliability analysis using deep neural network." *Expert Systems with Applications*, 189, 116104.

Maddox, W. J., Izmailov, P., Garipov, T., Vetrov, D. P., and Wilson, A. G. (2019). "A simple baseline for Bayesian uncertainty in deep learning." *Advances in Neural Information Processing Systems*, 32.

Margossian, C. (2019). "A review of automatic differentiation and its efficient implementation." *Wiley interdisciplinary reviews: data mining and knowledge discovery*, 9(4), e1305.

Neal, R. M. (2012). *Bayesian learning for neural networks*, Vol. 118. Springer Science & Business Media.

Nguyen, L. H. and Goulet, J.-A. (2021a). "Analytically tractable Bayesian deep Q-learning." *arXiv preprint arXiv:2106.11086*.

Nguyen, L. H. and Goulet, J.-A. (2021b). "Analytically tractable inference in deep neural networks." *arXiv preprint arXiv:2103.05461*.

Nguyen, L.-H. and Goulet, J.-A. (2022a). "Analytically tractable hidden-states inference in Bayesian neural networks." *Journal of Machine Learning Research*, 23(50), 1–33.

Nguyen, L.-H. and Goulet, J.-A. (2022b). "cuTAGI: a CUDA library for Bayesian neural networks with tractable approximate Gaussian inference." *GitHub repository, https://github.com/lhnguyen102/cuTAGI*.

Rasmussen, C. E. and Williams, C. K. (2006). *Gaussian processes for machine learning*. the MIT Press.

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986). "Learning representations by back-propagating errors." *nature*, 323(6088), 533.

Runnalls, A. R. (2007). "Kullback-leibler approach to gaussian mixture reduction." *IEEE Transactions on Aerospace and Electronic Systems*, 43(3), 989–999.

Schröder, L., Dimitrov, N., and Sørensen, J. (2020). "Uncertainty propagation and sensitivity analysis of an artificial neural network used as wind turbine load surrogate model." *Journal of Physics: Conference Series*, Vol. 1618, IOP Publishing, 042040.

Sudret, B. (2008). "Global sensitivity analysis using polynomial chaos expansions." *Reliability engineering & system safety*, 93(7), 964–979.

Tripathy, R. and Bilionis, I. (2018). "Deep uq: Learning deep neural network surrogate models for high dimensional uncertainty quantification." *Journal of computational physics*, 375, 565–588.

Wu, A., Nowozin, S., Meeds, E., Turner, R. E., Hernandez-Lobato, J. M., and Gaunt, A. L. (2019). "Deterministic variational inference for robust Bayesian neural networks." *International Conference on Learning Representations*.