This preprint is accepted for publication as a full peer-reviewed conference proceedings paper at
The 3rd International Conference on Machine Learning and Intelligent Systems

A Pointwise Evaluation Metric to Visualize Errors in Machine Learning Surrogate Models

Seyed Shayan SAJJADINIA [a,1], Bruno CARPENTIERI [a] and Gerhard A. HOLZAPFEL [b,c]

[a] *Department of Computer Science, Free University of Bozen-Bolzano, 39100 Bozen-Bolzano, Italy*

[b] *Institute of Biomechanics, Graz University of Technology, 8010 Graz, Austria*

[c] *Department of Structural Engineering, Norwegian University of Science and Technology (NTNU), Trondheim, Norway*

Abstract. Numerical simulation is widely used to study physical systems, although it can be computationally too expensive. To counter this limitation, a surrogate may be used, which is a high-performance model that replace the main numerical model by using, e.g., a machine learning (ML) regressor that is trained on a previously generated subset of possible inputs and outputs of the numerical model. In this context, inspired by the definition of the mean squared error (MSE) metric, we introduce the pointwise MSE (PMSE) metric, which can give a better insight into the performance of such ML models over the test set, by focusing on every point that forms the physical system. To show the merits of the metric, we will create a dataset of a physics problem that will be used to train an ML surrogate, which will then be evaluated by the metrics. In our experiment, the PMSE contour demonstrates how the model learns the physics in different model regions and, in particular, the correlation between the characteristics of the numerical model and the learning progress can be observed. We therefore conclude that this simple and efficient metric can provide complementary and potentially interpretable information regarding the performance and functionality of the surrogate.

Keywords. Visualization, surrogate modeling, numerical analysis, machine learning metric, PMSE

1. Introduction

Numerical methods such as finite element (FE) methods [1] are powerful tools to simulate various physics behavior [2–6], but for some practical applications, they may be computationally prohibitive due to the high nonlinearity and large number of equations involved. This motivates the use of a surrogate model, which is an efficient model that can replace the main numerical model using machine learning (ML) or any other efficient data-driven method [7]. In this regard, a simple idea, among many others, is to first generate a subset of possible inputs and outputs of the numerical model in order to

[1] Corresponding author at ssajjadinia@unibz.it

train the ML surrogate model [8], which now enables real-time simulation for several scientific disciplines, e.g., in medicine [9–13] and in mechanical engineering [14–18]. Although this technique still requires implementation of an expensive numerical solver to create the dataset, over time, the highly efficient surrogate can substitute it for the next simulations.

The mean squared error (MSE) metric is often used to evaluate the performance of such ML methods. Metrics of this type find the average of errors, by some comparison between the predicted and target values, over all the samples' geometrical points that define the physical system [19, 20]. Despite their advantages, especially in ML training, they may not show the progress in learning at each point, or in other words, they are not pointwise in a sense that they cannot distinguish the importance of each point. If the average pointwise errors are not considered, the overall accuracy in different regions of the physical system cannot be shown, and then, we may lose some information about the importance of each point in ML evaluation. To the best of the authors' knowledge, the advantages and feasibility of a pointwise metric in this application area of ML have not yet been explored, and assessing them is our main contribution.

This study aims to present the pointwise MSE (PMSE) metric that efficiently evaluates the surrogates and may elucidate the role of numerical model definition. While related work is reviewed in Section 2, this metric is mathematically defined in Section 3. Next, we try to show its importance with a simple experiment worked out in Section 4, and then some final remarks are provided in Section 5. We shared our research data and code at github.com/shayansss/pmse.

2. Related Work

Global metrics, such as the mean absolute error [11], MSE [20], coefficient of determination [21], are widely used for ML training or model evaluation. Despite their benefits, especially in simple quantitative comparison of the ML and numerical models, they might be barely interpretable [22]. Therefore, it is a common practice to visualize the prediction, comparing to the corresponding testing target, typically by mapping the results onto the numerical model [23–25]. However, they may not replace the local metrics that can enable pointwise visualization of the errors associated with all the samples.

Several studies, e.g., [26–29], used local surrogates or pointwise metrics, e.g., using the local maximum absolute error [27]. Although, some of these local errors, such as the pointwise cross validation error [28], may not be necessarily reliable surrogates for the point errors of test samples, but they still can increase the overall performance, while with possibly higher computational costs [29]. Few studies [30–32] defined pointwise metrics to measure and precisely visualize errors of samples, but they have not been used to study the correlation between the numerical model and these errors. In this paper, we propose a similar pointwise metric, but it is used only for testing to keep the training fast while we use it to especially interpret the relationship between the numerical and ML models.

3. Evaluation Metrics of Geometrical Points

In a typical supervised surrogate model, the ML regressor can predict the outputs of the numerical model extracted at N assigned points evaluated on M test samples. Denoting by $\bar{y}_{m,n}$ and $y_{m,n}$ the prediction and target values of the n^{th} point of the m^{th} sample, respectively, the performance of the model can then be assessed as follows

$$\text{SE}_m = \frac{1}{N}\sum_{n=1}^{N}(\bar{y}_{m,n} - y_{m,n})^2. \tag{1}$$

Here, SE_m is the squared error of sample m, which is averaged over the whole points. This can again be averaged over all the test samples in order to find the MSE by

$$\text{MSE} = \frac{1}{M}\sum_{m=1}^{M}\text{SE}_m. \tag{2}$$

To propose the PMSE, we used an equation similar to Eq. (2), but this time, the average is only calculated over the constituent points using

$$\text{PMSE}_n = \frac{1}{M}\sum_{m=1}^{M}(\bar{y}_{m,n} - y_{m,n})^2, \tag{3}$$

where PMSE_n indicates the PMSE value of point n, which, together with the other errors of the other points, allow us to visualize a contour plot of them on the physical system. This can reveal some new interpretable information about the training performance, considering that we can then see the accuracy of all the geometrical points. Note that having two metrics simultaneously may not affect the computational cost of the surrogate significantly, since, here, we use the PMSE metric only for evaluation of the trained model, rather than training.

This metric may be applicable when the points are attached to the simulated materials in the Euclidean space that should have similar locations at some stage, regularly at the beginning of the numerical simulation, to which the contour can be mapped. In particular, this condition can be realized when the metric is applied to the mesh nodes of a typical FE simulation, where the physical variables are measured.

4. Experiment

4.1. Setup

Physics problem. We are seeking to simulate a group of contact problems on a rectangular 2D sample (with a dimension of $20 \times 50\,\text{mm}^2$) with a circular hole within 2.5 mm in radius (see Figure 1). We assign an incompressible neo-Hookean model to the sample because it is highly used for this type of simulations. Next, we assume that the model, fixed to the bottom, has a hard contact through a semi-circular rigid indenter (with a radius of 10 mm) moving in both x_1 and x_2 directions. See this review study [33] for more about the governing equations of these models.

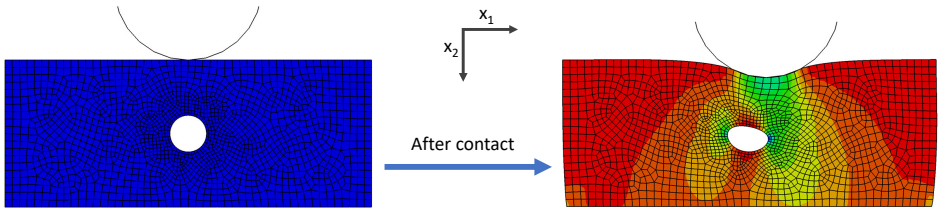

Figure 1. An example of the initial and final configurations of the numerical model. The maximum and minimum calculated values are indicated by a color range from red (165 MPa) to blue (−2750 MPa).

Numerical simulation. The physics problem has complex nonlinear properties, making its numerical simulation time-consuming enough to justify surrogate modeling. By FE numerical modeling, the complex physical system is discretized into a numerical model formed by a mesh of elements with simpler equations that subsequently approximates the physics involved for each point in each element. Since, here, a high-level FE solver is employed, we just need to assign different conditions of the physics problem, as stated in the former paragraph, and then, we run the solver to obtain the simulation data on each point, yielding contour plots of the Cauchy stress in the x_2 direction, see, e.g., Figure 1 (right). These values can roughly be interpreted as the distribution of the contact effect within the sample in the given direction.

Dataset. The input features include the movements of the indenter in the x_1 direction (l_1) and in the x_2 direction (l_2) as well as the material parameter (C). The outputs of the ML model are collected by rerunning the numerical simulation for a wide range of input values sampled by the uniform distribution (U), including $l_1 \sim U$ (−3 mm, 3 mm), $l_2 \sim U$ (0.1 mm, 3 mm), and $C \sim U$ (0.001 MPa, 1000 MPa). Once the data are normalized, 75 and 25 samples are randomly selected for the training and validation sets (used for the ML training), respectively, and 100 samples are inserted into the test set.

Training. A feed-forward neural network with the ReLU activation function is selected, as it is highly utilized for surrogate modeling of FE analysis [20]. While the learning rate is sufficiently small, with a value of 10^{-4}, to record the gradual progress of learning, in order to keep the training still fast, it is implemented by the Adam optimizer [34]. To speed up training even more, we reduce the number of parameters by applying only a single hidden layer that contains one-hundredth less than the number of neurons in the last layer. During training, the model is saved 8 times every 40 epochs for the subsequent evaluation.

Implementation. The numerical simulation is carried out by Abaqus [35], an FE solver, with its hybrid linear plane strain element formulation. This software is controlled by a Python script to generate the data set by assigning the input values to the numerical model and then iterating the simulation. As soon as the data set is created, the ML model is trained by the Keras [36] library with the TensorFlow [37] backend. This is implemented by Jupyter Notebook [38] on an Intel Core i5 CPU and 8 GB RAM. The evaluation metrics are calculated on the test set using the vectorized functions in NumPy [39], and ultimately, the visualization code is implemented using the Matplotlib [40] library along with another Python script to visualize the PMSE in Abaqus. More implementation details are available in the shared GitHub repository.

4.2. Results

The model was trained successfully, and the surrogate worked efficiently, considering that the numerical data generation took 55 min, whereas the training and evaluation of the ML surrogate took only about 2 min. Therefore, if the future repetition of the simulation is desired, the surrogate can guarantee real-time simulation.

The overall performance on the test set is shown in Figure 2 using the SE_m and MSE values during training. While the SE_m box plot provides some additional information about sample outliers, both estimate the overall performance without interpreting the role of the numerical model definition in training.

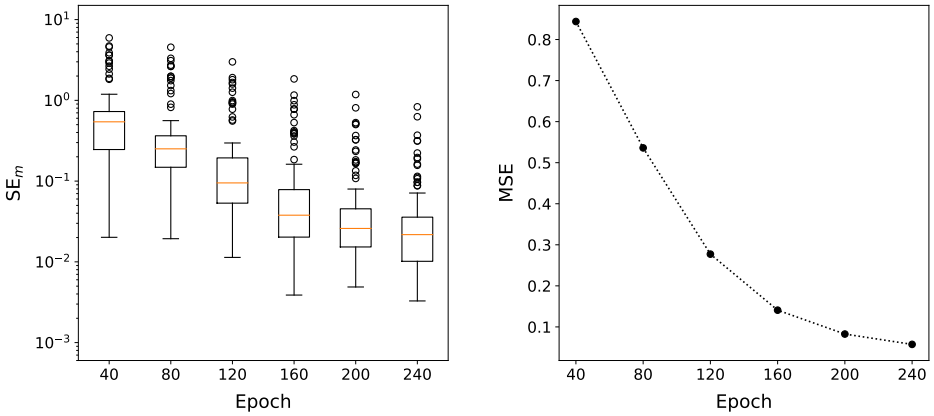

Figure 2. Squared errors of each test sample (SE_m in the left plot) and their mean values (MSE in the right plot) in relation to the number of epochs.

Figure 3 shows the corresponding PMSE contours, mapped to the initial configuration. In the first 40 epochs, the contour is mostly green, which means that most regions are still partially imprecise. These snapshots could indicate the learning progress across the points, which could be of high importance, e.g., while we needed more and more epochs to reach high accuracy (shown in blue), the area around the circular hole was well-trained after 160^{th} epoch. This could be an important basis for the analysis, because in a typical numerical simulation we can prioritize the accuracy in a certain region in order to reduce the computational time.

Other relevant results include: (i) by halving the model geometry into the right and left parts, due to the overall symmetry of the model, we expect similar complexity for each part, which can gradually lead to grossly symmetrical contour errors; (ii) some small zones near the top remained red or green in all snapshots, which could be explained by the fact that they underwent more deformation, caused by different movements of the contacting indenter, and consequently, the output data was more complex; (iii) looking at the last two snapshots, some areas where the elements change from irregular to regular rectangular shapes are not completely blue, suggesting that a more regular element shape can improve training. All of these results demonstrate the potential functionality of the pointwise metric to further interpret the association of the ML performance and the numerical model.

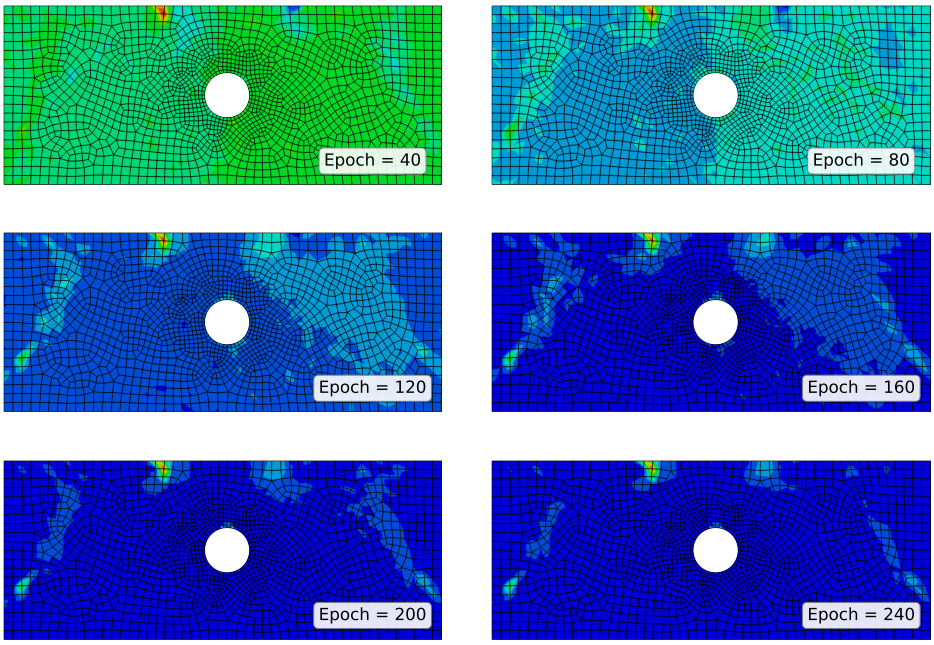

Figure 3. Contour plots of pointwise mean squared errors on the test set with respect to the trained model at different epochs. The contours are mapped onto the initial configuration of the model under contact before deformation. The color range from red to blue shows the values from 2 to 0, respectively.

For the sake of comparison, in Figure 4, we display the absolute error snapshots of a testing sample with the highest error in the first 40 epochs. We see that these errors are unsurprisingly larger than the PMSE values shown in Figure 3, especially on the right side of the plots. The contours do not show the small red zone that is observable in all PMSE snapshots, and they also show significantly less accuracy than the PMSE contours in the elements around the circular hole, particularly at late epochs, probably because of the large displacement boundary conditions contained in the input values. Therefore, the proposed metric proves to be clearly more generalizable.

5. Discussion and Conclusions

This research presented the PMSE, a simple and efficient metric for evaluating ML surrogate modeling by focusing on each point of the numerical model separately. Our experimental results demonstrate their important role by indicating more complex points for training that, along with our understanding of numerical simulation, could be used to interpret the possible correlation between the numerical model and learning progress. From this, we conclude that this metric can provide useful and complementary information about the performance of the model.

To thoroughly examine the benefits of the proposed metric, ideally it should have been tried on a variety of different numerical problems. However, we only experimented it on a simplified numerical problem, which is not a major limitation for this study, as the

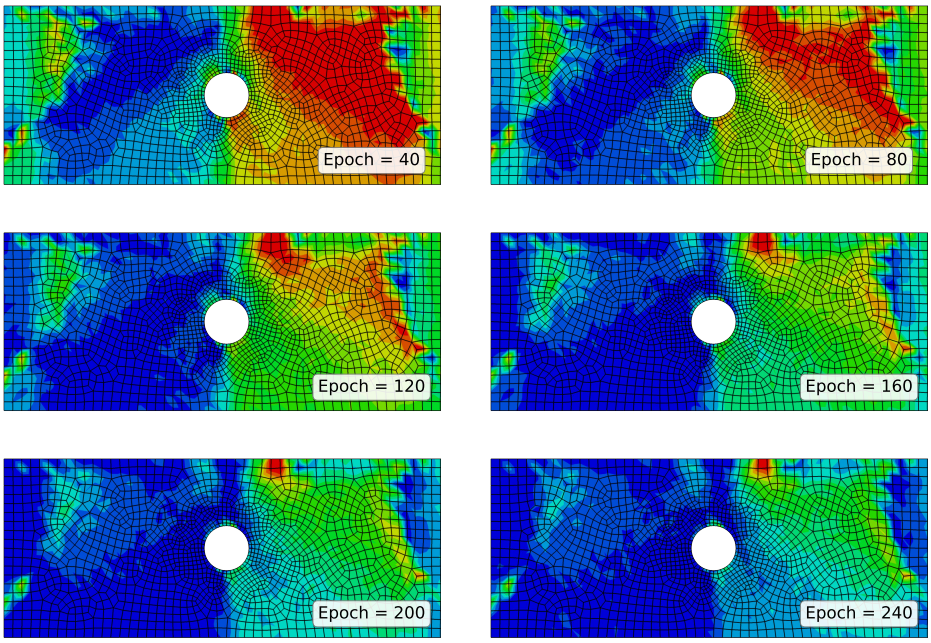

Figure 4. Contour plots of absolute errors corresponding to the sample with the highest squared error at the beginning, illustrated at different epochs. The contours are mapped onto the initial configuration of the model under contact before deformation. The color range from red to blue shows the values from above 4 to 0, respectively.

results obtained are sufficient to achieve our main objective, i.e., presenting preliminary evidence about the usefulness and feasibility of the pointwise metric.

Although the new metric is derived from the definition of MSE, similar pointwise metrics could be introduced using, e.g., the root mean squared error. We hope that such a pointwise metric will be applied to surrogate models in different (but applicable) domains in the future.

References

[1] Belytschko T, Liu WK, Moran B. Nonlinear finite elements for continua and structures. Chichester: John Wiley & Sons; 2000.

[2] Chen Q, Chen W, Wang G. Fully-coupled electro-magneto-elastic behavior of unidirectional multi-phased composites via finite-volume homogenization. Mechanics of Materials. 2021;154:103553.

[3] Pang L, Liu W, Zheng Q, Du Y, Meng X, Li X. Evaluation and analysis of metal mine filling based on numerical simulation and actual measurement. Environmental Earth Sciences. 2021;80(16):505.

[4] Torabi MR, Hosseini M, Akbari OA, Afrouzi HH, Toghraie D, Kashani A, et al. Investigation the performance of solar chimney power plant for improving the efficiency and increasing the outlet power of turbines using computational fluid dynamics. Energy Reports. 2021;7:4555-65.

[5] Sajjadinia SS, Haghpanahi M, Razi M. Computational simulation of the multiphasic degeneration of the bone-cartilage unit during osteoarthritis via indentation and unconfined compression tests. Proceedings of the Institution of Mechanical Engineers, Part H: Journal of Engineering in Medicine. 2019;233(9):871-82.

[6] Sajjadinia SS, Carpentieri B, Holzapfel GA. A backward pre-stressing algorithm for efficient finite element implementation of in vivo material and geometrical parameters into fibril-reinforced mixture models of articular cartilage. Journal of the Mechanical Behavior of Biomedical Materials. 2021;114:104203.

[7] Queipo NV, Haftka RT, Shyy W, Goel T, Vaidyanathan R, Kevin Tucker P. Surrogate-based analysis and optimization. Progress in Aerospace Sciences. 2005 Jan;41(1):1–28.

[8] Liang L, Liu M, Martin C, Sun W. A deep learning approach to estimate stress distribution: a fast and accurate surrogate of finite-element analysis. Journal of The Royal Society Interface. 2018;15(138):20170844.

[9] Holzapfel GA, Linka K, Sherifova S, Cyron CJ. Predictive constitutive modelling of arteries by deep learning. Journal of The Royal Society Interface. 2021;18(182):20210411.

[10] Shim VB, Holdsworth S, Champagne AA, Coverdale NS, Cook DJ, Lee TR, et al. Rapid Prediction of Brain Injury Pattern in mTBI by Combining FE Analysis With a Machine-Learning Based Approach. IEEE Access. 2020;8:179457-65.

[11] Liang L, Liu M, Martin C, Sun W. A machine learning approach as a surrogate of finite element analysis-based inverse method to estimate the zero-pressure geometry of human thoracic aorta. International Journal for Numerical Methods in Biomedical Engineering. 2018 Aug;34(8):e3103.

[12] Martin-Guerrero JD, Ruperez-Moreno MJ, Martinez-Martinez F, Lorente-Garrido D, Serrano-Lopez AJ, Monserrat C, et al. Machine learning for modeling the biomechanical behavior of human soft tissue. IEEE International Conference on Data Mining Workshops, ICDMW. 2016;0:247–253.

[13] Martínez-Martínez F, Rupérez-Moreno MJ, Martínez-Sober M, Solves-Llorens JA, Lorente D, Serrano-López AJ, et al. A finite element-based machine learning approach for modeling the mechanical behavior of the breast tissues under compression in real-time. Computers in Biology and Medicine. 2017 Nov;90:116–124.

[14] Gao W, Lu X, Peng Y, Wu L. A deep learning approach replacing the finite difference method for in situ stress prediction. IEEE Access. 2020;8:44063–44074.

[15] Kalyuzhnyuk AV, Lapin RL, Murachev AS, Osokina AE, Sevostianov AI, Tsvetkov DV. Neural networks and data-driven surrogate models for simulation of steady-state fracture growth. Materials Physics and Mechanics. 2019;42(3):351–358.

[16] Pfaff T, Fortunato M, Sanchez-Gonzalez A, Battaglia PW. Learning mesh-based simulation with graph networks. International Conference on Learning Representations. 2021:1–18.

[17] Xiao S, Hu R, Li Z, Attarian S, Björk KM, Lendasse A. A machine-learning-enhanced hierarchical multiscale method for bridging from molecular dynamics to continua. Neural Computing and Applications. 2020;32(18):14359-73.

[18] Ford E, Maneparambil K, Rajan S, Neithalath N. Machine learning-based accelerated property prediction of two-phase materials using microstructural descriptors and finite element analysis. Computational Materials Science. 2021;191:110328.

[19] Bhosekar A, Ierapetritou M. Advances in surrogate based modeling, feasibility analysis, and optimization: A review. Computers and Chemical Engineering. 2018 Jan;108:250–267.

[20] Phellan R, Hachem B, Clin J, Mac-Thiong JM, Duong L. Real-time biomechanics using the finite element method and machine learning: Review and perspective. Medical Physics. 2021;48(1):7–18.

[21] Cilla M, Martinez J, Pena E, Martínez Mn. Machine learning techniques as a helpful tool toward determination of plaque vulnerability. IEEE Transactions on Biomedical Engineering. 2012;59(4):1155-61.

[22] Molnar C. Interpretable machine learning; 2019.

[23] Calka M, Perrier P, Ohayon J, Grivot-Boichon C, Rochette M, Payan Y. Machine-Learning based model order reduction of a biomechanical model of the human tongue.

[24] Wang R, Kashinath K, Mustafa M, Albert A, Yu R. Towards physics-informed deep learning for turbulent flow prediction. In: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. KDD '20. New York, NY, USA: Association for Computing Machinery; 2020. p. 1457–1466.

[25] Sanchez-Gonzalez A, Godwin J, Pfaff T, Ying R, Leskovec J, Battaglia P. Learning to simulate complex physics with graph networks. In: III HD, Singh A, editors. Proceedings of the 37th International Conference on Machine Learning. vol. 119 of Proceedings of Machine Learning Research. PMLR; 2020. p. 8459-68.

[26] Ye Y, Wang Z, Zhang X. An optimal pointwise weighted ensemble of surrogates based on minimization

of local mean square error. Structural and Multidisciplinary Optimization. 2020 Aug;62(2):529-42.
[27] Xu H, Zhang X, Li H, Xiang G. An ensemble of adaptive surrogate models based on local error expectations. Mathematical Problems in Engineering. 2021;2021:1-14.
[28] Acar E. Various approaches for constructing an ensemble of metamodels using local measures. Structural and Multidisciplinary Optimization. 2010 Dec;42(6):879-96.
[29] Goel T, Hafkta RT, Shyy W. Comparing error estimation measures for polynomial and kriging approximation of noise-free functions. Structural and Multidisciplinary Optimization. 2008 Aug;38(5):429.
[30] Giselle Fernández-Godino M, Balachandar S, Haftka RT. On the use of symmetries in building surrogate models. Journal of Mechanical Design. 2019 01;141(6).
[31] Fuhg JN, Fau A. Surrogate model approach for investigating the stability of a friction-induced oscillator of Duffing's type. Nonlinear Dynamics. 2019;98(3):1709-29.
[32] Mo S, Lu D, Shi X, Zhang G, Ye M, Wu J, et al. A taylor expansion-based adaptive design strategy for global surrogate modeling with applications in groundwater modeling. Water Resources Research. 2017;53(12):10802-23.
[33] Freutel M, Schmidt H, Dürselen L, Ignatius A, Galbusera F. Finite element modeling of soft tissues: material models, tissue interaction and challenges. Clinical Biomechanics. 2014;29(4):363-72.
[34] Kingma DP, Ba JL. Adam: A method for stochastic optimization. In: Bengio Y, LeCun Y, editors. 3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings; 2015.
[35] Abaqus: Dassault Systèmes Simulia Corp., Providence, RI, USA; 2020. Available from: `www.simulia.com`.
[36] Chollet F, et al.. Keras. GitHub; 2015. Available from: `https://github.com/fchollet/keras`.
[37] Abadi M, Agarwal A, Barham P, Brevdo E, Chen Z, Citro C, et al. TensorFlow: Large-scale machine learning on heterogeneous systems. 2015. Software available from tensorflow.org.
[38] Jupyter Notebooks – a publishing format for reproducible computational workflows. In: Loizides F, Schmidt B, editors. Positioning and Power in Academic Publishing: Players, Agents and Agendas. IOS Press. IOS Press; 2016. p. 87-90.
[39] Harris CR, Millman KJ, van der Walt SJ, Gommers R, Virtanen P, Cournapeau D, et al. Array programming with NumPy. Nature. 2020 Sep;585(7825):357–362.
[40] Hunter JD. Matplotlib: A 2D graphics environment. Computing in Science and Engineering. 2007;9(3):90–95.