XLVI CILAMCE
VITÓRIA - ES - BRAZIL - 2025
XLVI IBERO-LATIN AMERICAN CONGRESS ON COMPUTIONAL METHODS IN ENGENEERING

An Artificial Neural Network Application for Efficient System Reliability Analysis

Bruno Gustavo dos Santos[1], Henrique Machado Kroetz[1]

[1]Center of Marine Studies, Federal University of Paraná
Av, Beira Mar, s/n, Pontal do Paraná, Paraná, Brazil
bruno.santos1@ufpr.br, henrique.kroetz@ufpr.br

Abstract. Artificial neural networks (ANNs) have emerged as a powerful tool for structural reliability analysis due to their capacity to effectively capture intricate, nonlinear relationships between input and output variables. In this domain, ANN models are typically trained on independent variables, such as material and geometric properties, to establish the connection between these inputs and the output variable, usually a probability of failure. However, despite their potential, ANNs are often disregarded in favor of well-established and computationally tractable methods such as Polynomial Chaos Expansions. Nevertheless, ANNs offer a significant advantage in addressing system reliability problems with multiple outputs. Their inherent ability to generate multiple outputs simultaneously eliminates the need for resorting to numerous surrogate models or intricate formulations, thereby leading to reduced computational demands. This paper proposes an efficient ANN-based surrogate modelling approach for system realibility analysis. A meta-heuristic optimization algorithm is employed to select the best architecture, considering three possible types of metamodels: multilayer perceptron (MLP), radial basis function (RBF) and long short-term memory networks (LSTM). A single optimal ANN is then used as a surrogate model for system reliability problems, whose several limit states are represented by the neurons on the last layer of the optimal network. Results are compared with a classical surrogate model. Examples are addressed, showing that multiple output ANNs can be a viable approach in this context.

Keywords: Artificial neural networks, structural reliability, metamodeling.

1 Introduction

Artificial neural networks (ANNs) are a mighty tool of meta-modeling that emerged through the years and achieved high levels of excellence, due to its capability to capture complex non-linear relationships between an amount of input and output variables. Inspired in the human neurons, this class of surrogate models are applied in reliability analysis to computationally describe a limit-state equation more easily, by reducing computational demand and speeding up the process. Such efficiency enables the testing of failure probability (P_f) through Monte Carlo Simulation (MCS), a method not always feasible with analytical equations [1,2]. A further advantage of ANNs is they doesn't require assumptions about smoothness nor distributions assumptions such as other surrogate models (all kinds of polynomial surface responses, for example). ANNs also have a substantial number of sub-models, providing different methodologies to achieve the desired results, one being more effective than the other for a certain application (some of these sub-models will be presented later). On the other hand, ANNs require a larger number of training data and more complex calibration procedures (related to its high capacities of modeling). Finally, ANNs have an underused capability of return multiple outputs in just one meta-model, something that can not be replicated by any other classic surrogate model. Thus, an entire system of functions can be described by only one metamodel. This work aims for comparing the effectiveness of different meta-modeling techniques (polynomial chaos expansion (PCE), single and multiple outputs ANN), comparing their precision, computational demand and overviewing the general results, providing new directions to the use of multiple output neural networks.

2 Surrogate models

Running real models might be hard and demand extreme long time. One alternative to reduce this cost is the use of a surrogate model. They are mathematical models made to achieve precise results, or in other words, analytical approximations for real models, which are more complex and hard to test. This happens by a previous sample set of inputs and outputs from the real problem (we can reefer to they as support points). The effective models are capable of provide similar responses to the real model, without costing as much as them. Let it be an effective model $\mathcal{M}$, of any type, considering only the relationship between the input variables and the output variables. Thus, the metamodel $\tilde{\mathcal{M}}$ is defined by the Equation 1:

$$\tilde{\mathcal{M}}(x) \approx \mathcal{M}(x) = y \tag{1}$$

where y is the high-fidelity results, which is a vector constructed by statistical tools, simplifying and capturing the real behavior of the real model. Equipped with such tools, structural reliability can run complex and multi variable problems. There are several surrogate models developed and wide applied for reliability. Some of this techniques, employed in this study, will be explained ahead.

2.1 Polynomial chaos expansion

Polynomial Chaos Expansion (PCE) has emerged as a classic surrogate model for structural reliability over the past decades, especially concerning stochastic finite element problems. In the context of PCE, $\tilde{Y}$ of a structural system's response is conceptualized as a random variable belonging to a specific vector space, suchating, for instance, the space of random variables with finite variance [3]. Thus, a common representation involves expressing it as a linear combination of the basis vectors of this particular space:

$$\mathcal{M}^{pce} = \tilde{Y} = \sum_{\alpha} \Psi_{\alpha} a_{\alpha} \tag{2}$$

where Ψ_{α} are the multivariate polynomial basis functions, also orthornormal with respect to the densities $f_X(x)$:

$$E(\Psi_i(X), \Psi_j(X)) = \delta_{ij} \tag{3}$$

The coefficients a_{α} are to be determined. To build an effective model, the series must be truncated. The precision of the truncated Polynomial Chaos Expansion (PCE) directly depends on the truncating order, which determines model complexity. A low-degree model might not capture system behavior adequately, while a high-degree model risks overfitting. A representative basis is formed by tensorizing the truncated univariate expansions of the input random variables. The number of basis functions selected n_b can be determined by a function of the number of random variables m and the maximum degree to be adopted p, as shown in Equation 4:

$$n_b = \frac{(m+b)!}{m!p!} \tag{4}$$

A practical algorithm to build the multivariate basis is presented in Sudret et al. (2006). The a may be obtained non-intrusively by least squares regression:

$$A = \left(\Psi^T \Psi\right)^{-1} \Psi^T Y \tag{5}$$

where A is a vector that gathers the coefficients a_{α} and is the data matrix where $\Psi_{ij} = \Psi_j(X^{(i)})$.

2.2 Artificial neural networks

This metamodeling class were inspired by human brain and its neuron's organization. They are computational models composed of interconnected neurons, designed to process data, learn variable relationships and generate predictions. Although their theoretical underpinnings were established earlier, ANNs gained significant practical use in the 1990s, exemplified by studies such as [4], thereby influencing future developments. Their robust predictive capability stems from adaptive weight adjustments and the application of non-linear transformations, allowing them to capture intricate dependencies and distinct operational behaviors. They do that without relying on the statistical simplifications often found in other meta-models. The field of ANNs boasts a vast array of subtypes, capable of performing in diverse applications. Given this briefing, several alternatives exist for the scope of this study, but only three specific types are proposed considering its behavior and applicability: the Multi-Layer Perceptron (MLP), Radial Basis Function (RBF) and Long Short-Term Memory (LSTM):

MLP: the most classic, popular and frequently used as meta model ANN. In its original form (which is used in the work), it is a feed-forward ANN, hence the information flow is one-directional. Its architecture is composed by a sequence of layers, and in these layers are interconnected artificial neural nods. The mandatory layers are input and output layers, but its greatest advantage comes from the hidden layers, providing to the MLP capturing non-linearity and abstractions in the data. MLP is very versatile, allowing to say it is a universal approximator [5]. The structure variates according to the optimizer and loss function. This work's parameters are loss function as mean squared error (MSE) and the optimizer Adam, because it isn't subject to problems in the gradient's diagonal rescale, also demanding low memory and useful for situations with lots of terms and data [6,7].

RBF: it is a different type of feed-forward ANN, with a different architecture: it is based in only three layers: input, hidden and output layer, only. There is no architecture variation besides the number of nodes. The hidden layer is the one responsible to analyses the data, where the radial basis function is the activation function, dictating its working. This function only depends on the euclidean distance between a reference point and the other data. There are three primary parameters for RBF ANN: the centers, which are the reference points mentioned before, the distance between the center and the data, called variance and finally the weights, pondering the influence of each analysis to built the result [8,9]. The outputs from this type of ANN are:

$$\tilde{\mathcal{M}} = \sum_{i=1}^{n} w_i^{RBF} \phi \left(||C_i^{RBF} - x|| \right) \tag{6}$$

where x is the inputs, n is the number of neuron of the hidden layer, w_i^{RBF} are the weights of each neuron $\| C_i^{RBF} - x \|$ is the euclidean distance and ϕ is the RBF function.

LSTM: recurrent ANN designed by [10] to deal with the devanescent gradient, a problem related to weights that explode or vanish by time steps. This methodology creates a constantly adaptative memory to base the fitting from each neuron, i.e, a database is kept to provide an adaptative memory to the nodes and according to time steps. Eventually, all the database will change, following the tendency of the fitting data. That is why is long (endure the whole process) and short (data of short time steps) name comes from. The classic LSTM model is fixed on three layers also, varying only the number of neurons. The standard node is composed by a cell and three doors: enter, exit and forgetting. The cell stores data in each time step, the enter and exit doors regulate the information that will come in and out at the current state and the forgetting door will choose which data from the previous cell's data is going to be forgotten, reducing memory expenses. This kind of ANN has been applied to emulate limit-state equations with high precision [11].

3 Surrogate models in system reliability problems

Let X be a vector that gathers the random variables of a problem, with joint probability density function $f_X(x)$. Let $\mathcal{F}$ be the event of system's failure (i.e. when the system can not hold its solicitation). The probability of failure P_f associated with this limit at the I state is given by:

$$P_f = \int_{D_f} f(x)\, dx \tag{7}$$

where D_f is the i-th failure domain, defined as the set of all points tested in the corresponding limit state function assumes negative values. Often, analytical solving this integral is challenging and demanding. That is why numerical methods such as MCS are employed for its solving. The computational cost of this approach is generally high, especially in reliability problems that involve multiple random variables, time dependency or numerical models with numerous degrees of freedom [12-14]. In engineering, the occurrence reliability system are very comum. These system are a composition of limit-state equations with different P_f resulting in a non-differentiable failure surface where occurs the intersection of the individual failures surfaces. This might represent challenging event to the use of metamodeling techniques.

4 Methodology and performance comparison

To ensure a fair comparison, this study maintained consistent parameters across all tests, prioritizing computational cost and the number of support points required for achieving desired precision. Training and testing data for all tests were generated using Latin Hypercube Sampling (LHS) to ensure well-distributed support points. Simple approaches were employed for each metamodel to ensure generalizable and replicable results. The proposed methodology generates a set number of support points (starting at 50 and limited at 2000) via LHS for training. Their precision was then evaluated by conducting ten Monte Carlo Simulations (MCSs) using metamodel predictions, rather than analytical functions. The objective was to achieve a mean error under 1% compared to the reference value. If this condition was not met, the process was restarted, incrementally increasing the number of support points until the criterion was satisfied. This criterion's fulfillment was linked to any of the multi-output networks – the primary focus of this paper – to effectively test the proposed technique. Once one of these networks achieved the goal, its class would be selected to continue the process (if necessary) and verify the differences between the classical and proposed approaches. The processing time was monitored throughout the simulations to quantify computational cost. The study also addresses the lack of established guidelines for Artificial Neural Network (ANN) architectures in structural reliability problems, particularly for multiple-output networks. The proposed solution for the internal optimization of these architectures (MLP and LSTM) involved defining parameter limits and treating the optimization as a trial-and-error problem due to the integer nature of parameters (e.g., number of neurons and layers). For RBF networks, the optimization of σ (a float variable) was handled differently. The optimization strategy employed was the meta-heuristic Particle Swarm Optimization (PSO), using swarm parameters as presented by [15]. The adopted metric was MSE, with the worst individual MSE from each limit-state equation being selected to represent the overall run, thereby ensuring safety. The "best" metamodel was defined as the one that fulfilled the precision condition using the smallest number of support points

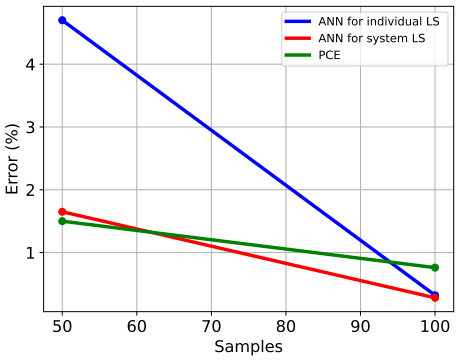
4.1 Serial system

This example was first presented by [16], and consists in the series system of nonlinear equations represented in Equation 8, where all random variables follow standard normal distributions (mean = 0 and standard deviation = 1).

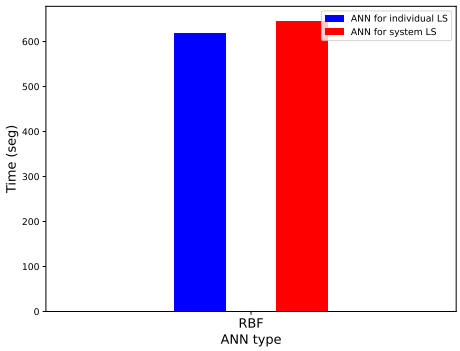
$$g_{sys}(X) = \min \begin{cases} g_1(x) = 2 - x_2 + e^{-0.1x_1^2} + (0.2x_1)^4 \\ g_2(x) = 4.5 - x_1 \cdot x_2 \end{cases} \tag{8}$$

The reference result $P_{f\ serial}$ = 3.47·10^{-3} was obtained via brute Monte Carlo simulation, using 10^7 samples. On side A of the grafics, the X-axis represents the number of samples for training, while the Y-axis indicates the mean MCS error ran with the surrogate models functions compared with the reference values of P_f. The B side of grafics represente the accumulated time of running, considering optimization, training and MCSs

simulations, The results of precision are presented below at Figures 1.a e 1.b:



a) Precision



b) Accumulated processing time

Figure 1. Serial example results

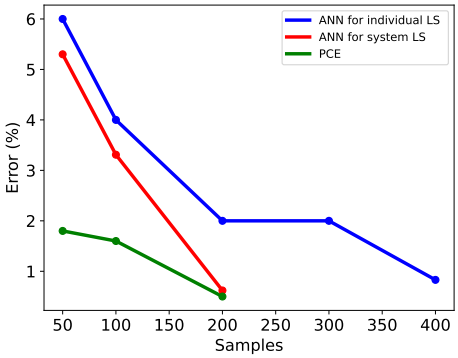
The best performance were a tie of PCE and the types of ANN RBF. In matter of time, the individual ANN obtained a small advantage over the multioutput.

4.2 Mixed system

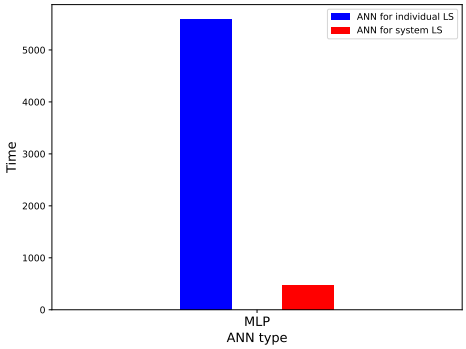
This example from [17], based in fragile chain system with high complexities levels and different from the others, the independent variables of this problems are log-normal, what may represent more complexity in its relationships to be described by the metamodels. Variables are exactly as shown at [17].

$$g_{sys}(X) = min \begin{cases} max \begin{cases} min \begin{cases} g_1(x) = x_1 - x_5 \\ g_2(x) = x_2 - \frac{x_5}{2} \\ g_3(x) = x_3 - \frac{x_5}{2} \\ g_4(x) = x_4 - \frac{x_5}{2} \end{cases} \\ max \begin{cases} min \begin{cases} g_5(x) = x_2 - x_5 \\ g_6(x) = x_3 - x_5 \end{cases} \\ g_7(x) = x_4 - x_5 \end{cases} \end{cases} \end{cases} \tag{9}$$

This problem works as seven limit-states, generated by five independent variables. The reference result $P_{f\ mixed} = 4.47 \cdot 10^{-2}$ was obtained via brute Monte Carlo simulation, using 10^7 samples. Results are presented below at Figure 2.a and 2.b:



a) Precision



b) Accumulated processing time

Figure 2. Mixed example results

The best performance was a tie of PCE and multioutput MLP. The advantage of time was considerable in favor of the proposed approach, demonstrating there are cases of the multioutput ANN can be very effective.

5 Conclusions

This paper studied the performance of multiple-output artificial neural networks on system-reliability problems, in comparison with classic surrogate models. Different situations were tested, to evaluate the global performance of the designed approach. Increasing number of points were considered for static support points, so that the performance between different surrogate models could be compared when the same amount of data is available, comparing their behaviors and the demanding time from each operation. On all cases, the optimal multiple output ANN showed different precision in comparison to other optimized surrogate models, proving its value as a high fidelity surrogate model. The PCE demonstrated to be a step a head as an ideal surrogate model to structural reliability, at general overview. Dealing with ANNs separately, multiple output and the classic approach were very similar in their performance, and with the increase of limit-states to evaluate, multiple ANN showed quicker at the total amount of time (only one model for a whole system against individual ones for each), which is very competitive about this new proposal. Paring results to previously works [18], it is notable the influence of optimization at the ANN architecture, where the results have been improved. Also, as the experiments were conducted, it was observed that despite evolve a more difficult problem, the complexity of the multiple output model didn't demand many layers or an abrupt increase of neurons, showing that ANN can deal with high complexity without becoming an over robust model, meaning no much more layers or nodes than the usual.

Acknowledgements. The authors thank the National Council for Scientific and Technological Development CNPq for the financial support.

Authorship statement. The authors hereby confirm that they are the sole liable persons responsible for the authorship of this work, and that all material that has been herein included as part of the present paper is either the property (and authorship) of the authors, or has the permission of the owners to be included here.

References

[1] Elhewy, A. H., Mesbahi, E., & Pu, Y. (2006). Reliability analysis of structures using neural network method. Probabilistic Engineering Mechanics, 21(1), 44–53.
[2] Kroetz, H. M., Tessari, R. K., & Beck, A. T. (2017). Performance of global metamodeling techniques in solution of structural reliability problems. Advances in Engineering Software, 114, 394–404.
[3] Panayirci, H. M., & Schu¨ eller, G. I. (2011). On the capabilities of the polynomial chaos expansion method within sfe analysis—an overview. Archives of Computational Methods in Engineering, 18(1), 43–55.
[4] Papadrakakis, M., Papadopoulos, V., & Lagaros, N. D. (1996). Structural reliability analyis of elastic-plastic structures using neural networks and Monte Carlo simulation. Comput Methods Appl Mech Eng, 136(1–2), 145–163.
[5] Marius-Constantin, P., Balas, V. E., Perescu-Popescu, L., & Mastorakis, N. (2009). Multilayer perceptron and neural networks. WSEAS Trans Circuits Syst, 8(7), 579–588.
[6] Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.
[7] Carlon, A. G., Kroetz, H. M., Torii, A. J., Lopez, R. H., & Miguel, L. F. F. (2022). Risk optimization using the chernoff bound and stochastic gradient descent. Reliability Engineering & System Safety, 223, 108512.
[8] de Souza, L. P., Batista, P. V. C., & Silva, P. C. L. (2021). A New Heuristic for Radius Estimation in RBF Neural Networks. ENCONTRO Nac. INTELIGÊNCIA Artif. E Comput. (ENIAC), 18, 189–196.
[9] Haykin, S. (2005). Neural Networks - A Comprehensive Foundation (2nd ed.). Pearson.
[10] Hochreiter, S., & Schmidhuber, J. (1997). Long Short-Term Memory. Neural Comput., 9(8), 1735–1780.
[11] Li, M., & Wang, Z. (2022). LSTM-augmented deep networks for time-variant reliability assessment of dynamic systems. Reliab. Eng. Syst. Saf., 217.
[12] Pellizzer, G. P., Kroetz, H. M., Leonel, E. D., & Beck, A. T. (2020). Time-dependent reliability of reinforced concrete considering chloride penetration via boundary element method. Latin American Journal of Solids and Structures, 17(8).
[13] Kroetz, H., Moustapha, M., Beck, A., & Sudret, B. (2020). A two-level kriging-based approach with active learning for solving time-variant risk optimization problems. Reliability Engineering & System Safety, 203, 107033.
[14] Tessari, R. K., Kroetz, H. M., & Beck, A. T. (2017). Performance-based design of steel towers subject to wind action. Engineering Structures, 143, 549–557.

[15] Bratton, D., & Kennedy, J. (2007). Defining a standard for particle swarm optimization. Proc. 2007 IEEE Swarm Intell. Symp. SIS 2007, 120–127.
[16] Grooteman, F. (2011). An adaptive directional importance sampling method for structural reliability. Probabilistic Engineering Mechanics, 26(2), 134–141.
[17] Zhao, Y.-G., & Ang, A. H.-S. (2003). System Reliability Assessment by Method of Moments. J. Struct. Eng., 129(10), 1341–1349.
[18] Kroetz, H. M., & Santos, B. G. dos. (2011). Performance Comparison between Multiple-Output Artificial Neural Networks and Classic Surrogate Models for System Reliability Problems. Proceedings of the joint XLIV Ibero-Latin-American Congress on Computational Methods in Engineering, 26(2), 134-141.