

Central Lancashire Online Knowledge (CLOK)

Title	A data-driven, machine learning scheme used to predict the structural response of masonry arches
Type	Article
URL	https://clock.uclan.ac.uk/id/eprint/49640/
DOI	https://doi.org/10.1016/j.engstruct.2023.116912
Date	2023
Citation	Motsa, Siphesihle Mpho, Stavroulakis, Georgios E. and Drosopoulos, Georgios (2023) A data-driven, machine learning scheme used to predict the structural response of masonry arches. Engineering Structures, 296. ISSN 0141-0296
Creators	Motsa, Siphesihle Mpho, Stavroulakis, Georgios E. and Drosopoulos, Georgios

It is advisable to refer to the publisher's version if you intend to cite from the work.
<https://doi.org/10.1016/j.engstruct.2023.116912>

For information about Research at UCLan please go to <http://www.uclan.ac.uk/research/>

All outputs in CLOK are protected by Intellectual Property Rights law, including Copyright law. Copyright, IPR and Moral Rights for the works on this site are retained by the individual authors and/or other copyright owners. Terms and conditions for use of this material are defined in the <http://clock.uclan.ac.uk/policies/>

A data-driven, machine learning scheme used to predict the structural response of masonry arches

Siphesihle Mpho Motsa^a, Georgios E. Stavroulakis^{b*}, Georgios A. Drosopoulos^{c,a}

^aDiscipline of Civil Engineering, University of Kwazulu-Natal, Durban, South Africa

^bDepartment of Production Engineering and Management, Technical University of Crete, Chania, 73100, Greece

^cDiscipline of Civil Engineering, University of Central Lancashire, Preston, United Kingdom

* Corresponding author.

E-mail address: gestavroulakis@tuc.gr

Abstract

A data-driven methodology is proposed, for the investigation of the ultimate response of masonry arches. Aiming to evaluate their structural response in a computationally efficient framework, machine learning metamodels, in the form of artificial neural networks, are adopted. Datasets are numerically built, integrating Matlab, Python and commercial finite element software. Heyman's assumptions are adopted within non-linear finite element analysis, incorporating contact-friction laws between adjacent stones, to capture failure in the arch. The artificial neural networks are trained, validated, and tested using the least square minimization technique. It is shown that the proposed scheme can be used to provide a fast and accurate prediction of the deformed geometry, the collapse mechanism and the ultimate load. Cases studies demonstrate the efficiency of the method in random, new arch geometries. Relevant Matlab/Python scripts and datasets are provided. The method can be extended towards structural health monitoring and the concept of digital twin.

Keywords: FEM, Machine Learning, Artificial Neural Network, Multi-hinge failure, Damage Prediction, Masonry Arches, Data-driven Mechanics, Digital Twin

1. Introduction

Masonry arches have been used widely during centuries to span and enclose space. The structural benefit of the arch shape is still under investigation even though the technique was first observed in Mesopotamian brick architecture, dating back to the 2nd Millennium BC (Anastasio, 2020). Arches are efficient load-bearing structures, which distribute applied loads through compression in adjacent masonry stones. The arch, as a method of construction, is directly relevant to the material behaviour of masonry. There exists a harmonic relation between the masonry stones and the shape of the arch, to ensure that the structure is mainly under compression. Compression failure of masonry arches is generally unlikely to take place, thus, the typical failure mode of arches is a tensile hinge mechanism (Heyman, 1966, Heyman, J. 1982, Drosopoulos et al., 2006, Grillanda et al., 2021).

In particular, the typical mode of failure for masonry arches is the formation of tension hinges in-between the masonry stones, activated when the thrust line is tangent to the masonry arch section edges. The change of the structural state, from equilibrium to mechanism, can be caused by settlement of supports due to earthquakes, vertical loads due to vehicles, erosion, or ground bearing failure. This hinge mechanism can result in damage and eventually partial or total collapse (Bergamo et. al., 2015, Cavalagli et. al., 2016, Portioli and Cascini, 2017, Sánchez-Aparicio et. al., 2019).

Several investigations have been conducted, to highlight the structural response of masonry arches. Hooke (1676) has been pioneer for first describing the compression behaviour of masonry arches under their self-weight and for proposing a rational rule to estimate the size of masonry stones and the geometry of masonry arches. This rule is based on the analogy of a hanging chain forming catenary in tension under its self-weight, and on a masonry arch (inverted chain), standing rigid in compression (Heyman, 1982, Heyman, 1998, O'Dwyer, 1999, Block et al., 2006). In (Poleni, 1748), Hooke's hanging chain principle was used to assess the safety of the cracked dome of St. Peter's in Rome. More literature is found in the same direction, providing numerical methods which can be used to determine the thrust line closest to the geometrical axis of a given arch (Moseley, 1833, Winkler, 1867, Heyman, 1969, Tempesta and Galassi, 2019). Recently, in (Gáspár et al., 2022) this principle was used in a study which relates the optimal geometry of a masonry arch and the number of concurrent hinges under self-weight, at a limit state quantified by minimum thickness.

Among the first approaches used to evaluate the response of masonry arches are those relying on limit analysis tools. Within these approaches, masonry blocks are simulated as rigid blocks and governing equations are often derived using static and kinematic theorems for limit analysis. Some relevant publications, evaluating the response of masonry arches under vertical and horizontal loads, as well as settlement of supports, can be found in (Ochsendorf, 2006, Milani and Lourenço, 2012, Cavalagli et. al. 2016, Portioli and Cascini, 2017, Cascini et. al., 2018, Galassi, 2023, Galassi and Zampieri, 2023). It is noted that in this work a different methodology, relying on finite element analysis, is adopted to capture the response of masonry arches. In addition, emphasis is mainly given in the data-driven nature of the proposed framework. Therefore, limit analysis techniques could also be adopted, in principle, to provide the structural response of arches under the suggested data-driven scheme.

In the later years, more advanced techniques have been developed, to evaluate the structural behaviour and ultimate, failure response of masonry arches. Often, the finite element method is the numerical tool used to simulate detailed two and three-dimensional geometries of masonry arches. To capture the failure response, different constitutive descriptions are introduced in these models. The arising computational cost is significant, in particular for bigger models with more structural parts, higher dimensions and complex non-linear material laws.

In (Özmen and Sayın, 2018), three-dimensional finite element models are used to assess the seismic response of an old masonry arch bridge in the framework of the macro modelling approach. In (Charalambidi et al., 2022), a finite element model introducing unilateral interfaces to capture failure between masonry stones, is proposed to identify and predict the cause of the existing structural damage of a masonry monument in Greece. In (Tapkın et al., 2022), various non-linear finite element models were used to simulate the structural response of a three-span masonry arch bridge located in Turkey. In (Drosopoulos and Stavroulakis, 2018), a computational homogenization method is proposed, to investigate localization of damage in masonry walls. Macroscopic, structural scale failure is represented by cohesive cracks in the framework of the extended finite element method, using the effective material properties obtained from microscopic simulations. More efforts on numerical modelling of masonry arches using the finite element method can be found in (Ferrero et al., 2023, Rahimi et al., 2022, Zampieri et al., 2021, Tubaldi et al., 2020, Stavroulaki et al., 2018, Conde et al., 2016, Sarhosis et al., 2016, Milani et al., 2006, Lourenço, 2002).

In the 4th industrial revolution era, machine learning elements, such as artificial intelligence (AI), have been adopted to solve complex non-linear engineering problems. In the recent years, even more machine learning algorithms have been developed to solve engineering problems. In structural engineering, artificial neural networks have been used to assess the strength and performance of concrete structures (Chang and Zheng, 2019, Prakash et al., 2019, Sadowski et al., 2018) and the structural response of steel (Beskopylny et al., 2020, Wołowiec and Kula, 2012). Other available machine learning approaches include non-destructive and vision-based measurement techniques, which are used as a method of structural health monitoring (Yuan et al., 2022, Bekas and Stavroulakis, 2017, Psychas et al., 2016, Cavaleri et. al., 2022, Grandio et al., 2022, Ashrafian et. al., 2023).

The main concept of introducing machine learning approaches in structural engineering, is to use existing data, carrying information for the structural response, in numerical simulations. Thus, databases are developed and used to train a machine learning algorithm. The trained algorithm is then used to assess the response of the structural system. A numerical metamodel is developed within this framework, able to potentially replace or complete existing structural evaluations, due to missing experiments or computationally expensive calculations.

In (Jing et al., 2022), an artificial neural network called *BridgeNet* is proposed, for automating the segmentation of masonry arch bridge elements obtained from large-scale point clouds. In (Melchiorre et al., 2021), machine learning algorithms are used to structurally optimize the cross-section of a circular arch by calculating the internal stresses and comparing them against the yield stresses of the material. In (Civera et al., 2022), artificial intelligence and machine learning algorithms are used to interpret

operational modal analysis mode shapes, which is normally a computationally expensive task, aiming in the structural health monitoring of masonry arches. In (Drosopoulos and Stavroulakis, 2020), machine learning is introduced in multi-scale computational homogenization to capture the non-linear response of masonry walls. Recent studies emphasize in using image recognition and deep machine learning tools, including for instance computer vision and convolutional neural networks (CNN), to generate geometric digital twins for masonry structures (Dais et al., 2021, Loverdos and Sarhosis, 2023, Loverdos and Sarhosis, 2023).

In this study, a data-driven numerical analysis of masonry arches is proposed, to evaluate their mechanical response considering different arch geometries (span and thickness of the masonry stones). The first step of this investigation is to conduct parametric simulations, in order to develop datasets, using as input and output values geometric and structural parameters of masonry arches. In a subsequent step, these datasets will be used to train artificial neural networks. Therefore, the article proposes a methodology for using machine learning, data-driven techniques, in order to achieve a fast and accurate prediction of the structural response of masonry arches. It is noted that to the authors' best knowledge, only limited works can be found, emphasizing in data-driven, machine learning approaches, for the structural evaluation of masonry arches. Also, the majority of the published research focuses more on the geometric aspects of the data-driven approaches, comparing to the structural response and the failure mode prediction which is the core outcome of this investigation.

In particular, two-dimensional, non-linear finite element models were developed to perform the parametric assessment, considering the following Heyman's assumptions: (i) masonry stones have no tensile strength and, (ii) the compressive strength of the stone is infinite (Heyman, 1966). The results obtained from the finite element models were used to train, validate, and test artificial neural networks. This procedure has been implemented using Python, Matlab and commercial finite element software. The trained neural networks can provide a fast structural evaluation of random masonry arches, with limited computational cost, emphasizing in critical and valid information for the ultimate structure response. Thus, the outputs of the trained neural networks, are, (a) the deformed geometry depicting potential damage under the self-weight, (b) the deformed geometry, also depicting potential damage, under self-weight and a vertical point load applied at $\frac{1}{4}$ of the span and (c) the ultimate load at collapse.

2. Ultimate (failure) response of masonry arches

Goal of this article is to propose a data-driven methodology for the structural evaluation of masonry arches. According to the overall concept, parametric structural simulations within non-linear finite element analysis are conducted to generate datasets providing the ultimate, failure response of masonry arches of various geometries. Machine learning elements in the form of artificial neural networks are then used to train the set of results obtained from the parametric finite element analysis. The trained neural networks are metamodels able to predict the failure response of randomly chosen masonry arch geometries. The parametric investigations are conducted on circular and parabolic

masonry arches, but any other geometry or even different structural type, could be adopted using the proposed concept.

In this section, principles of the mechanical response of masonry arches are provided, emphasizing in the ultimate, failure behaviour of these structures. The classical collapse mechanism theory as presented in Heyman (1982), has been widely used to determine the load-bearing capacity of masonry arches. This technique has been adopted by other analytical methods to estimate the thrust line passing through masonry arch stones. For arches to be fully under compression, the thrust line must lie within the core (middle third) of the section (Heyman, 1982). Unreinforced masonry arches form a plastic hinge when the thrust is tangent to the extrados and/or intrados of the arch. When the thrust line is tangent at three extrados and/or intrados points of the arch, three hinges are developed. The introduction of three hinges changes the determinacy of a fixed support arch from statically indeterminate to statically determinate. Then, the development of a fourth hinge triggers a kinematic collapse mechanism, widely known as the four-hinge collapse mechanism (Heyman, 1967).

According to this description, four-hinge collapse mechanism is generally the common cause of structural failure of masonry arches. In addition, this mechanism may not arise when a symmetrical arch is subjected to symmetric loading, like self-weight. Heyman (1967) demonstrated that this response may be obtained on semi-circular arches under their own weight. Under this condition, at least a 5th hinge must form to trigger a kinematic collapse.

To simulate the mentioned hinge mechanism between adjacent stones, principles taken from non-smooth mechanics have been adopted in this article within non-linear finite element analysis (Panagiotopoulos, 1985, Drosopoulos et al., 2006). In particular, a unilateral contact and friction law is used to describe the surface contact conditions. This law introduces a strong nonlinearity, even though the stones are assigned linear material properties. Therefore, nonlinearity is restricted to the interfaces between the stones, allowing for the opening and/or sliding along these interfaces. This opening and sliding that may appear in an interface, give rise to 8 configurations of deformation between two adjacent stones (Stockdale et al., 2022). These deformation modes include openings, rotations, slips, and slip/rotation combinations, as shown in figure 1. It is noted, that these deformation modes can be predicted by the machine learning, data-driven approach which is proposed in this study.

In this study, relations (1)-(3) are adopted to describe the contact conditions in the interface between two adjacent masonry blocks. Inequality (1) expresses the non-penetration condition and inequality (2) states that only compressive stresses can be developed in the interface, noticing that u is the single degree of freedom, g is the initial opening of the contacting bodies and t^n is the normal stress at the interface. Equality (3) expresses the complementarity relation, which states that either zero stresses arise and opening takes place or non-zero stresses appear, and contact is activated.

$$h = u - g \leq 0 \Rightarrow h \leq 0 \quad (1)$$

$$-t^n \geq 0 \quad (2)$$

$$t^n(u - g) = 0 \quad (3)$$

In the tangential direction, a static version of the Coulomb friction law is adopted, indicating that sliding takes place when the shear stress t^t in the interface reaches a critical value τ_{cr} , according to relation (4):

$$t^t = \tau_{cr} = \pm\mu|t^n| \quad (4)$$

where μ is the friction coefficient.

The coefficient of friction assigned to the unilateral contact-friction interfaces of this article is set equal to 0.5 (Melbourne and Gilbert, 1995). Loading of each arch involves the self-weight, applied at a first analysis step and a vertical point load applied at a second step, at the $\frac{1}{4}$ of the span. It is noted that this is the worst load position, since the ultimate load derived from this position is the lowest compared to those obtained from other load positions. This is discussed in several studies, such as in the classical work presented in (Heyman, 1982), as well as in subsequent studies (Drosopoulos et al., 2006). The arches are also assigned fixed boundary conditions. The assigned material properties on the finite element models are as follows: density is equal to 2300kg/m³, modulus of elasticity is 30GPa and Poisson's ratio is 0.2. For the implementation of the proposed data-driven scheme Abaqus commercial finite element software and two programming codes, namely, Matlab (Matlab, 2021) and Python, have been adopted.

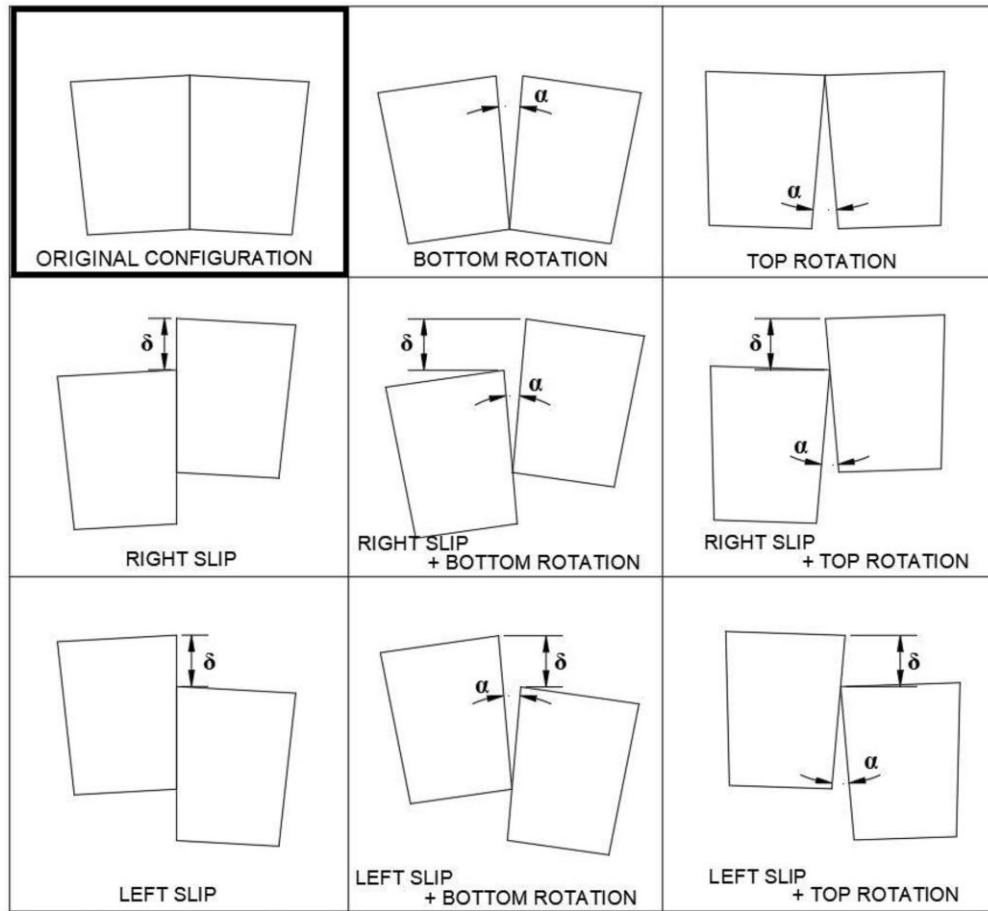


Figure 1: Potential deformation modes between two adjacent blocks of an arch (Stockdale et al., 2022).

3. Machine learning using artificial neural networks

Machine learning is a branch of artificial intelligence which focuses on training computer codes on how to make predictions of systems based on available datasets and algorithms. The ability of machine learning algorithms to recognise patterns from large datasets allows for their usage in various fields of study such as engineering, business, and science.

In (Reich, 1997) it was shown that machine learning was still in its infancy stage due to limitations on machine learning algorithms and computing power, as well as due to the lack of experimental databases to validate the machine learning models. Recently, it has been demonstrated a significant increase of using machine learning tools, to solve complex structural engineering problems (Thai, 2022). In addition, databases like *DataCenterHub*, *DesignSafe* and *Mendeley Data* can now be used to validate machine learning models. The number of machine learning algorithms has been significantly increased and tools like artificial neural networks, decision trees, regression analysis, support vector machine, random forest and boosting algorithms, have been adopted in structural engineering applications (Thai, 2022).

In this article, emphasis is given on using artificial neural networks as the numerical tool which implements data-driven structural assessment. An artificial neural network is developed by biomimicking the human brain structure, thus, how neurons are interconnected to imitate thinking, recognition and decision making (Simon, 1999, Nasrabadi, 2007). It was first invented by (Rosenblatt, 1958) in 1958 and called the *perceptron*. Due to improvements to computational power, various algorithms have been developed such as the feedforward neural network (Ivakhnenko, 1971), the radial basis function neural network (Broomhead and Lowe, 1988), the convolutional neural network (LeCun et al., 1998), the recurrent neural network (Elman, 1990) and the adaptive neuro-fuzzy inference system (Jang, 1993). The feedforward neural network is the most common system, due to its simplicity and robustness to solve multi-variate and nonlinear modelling problems (Mostafa et al., 2022, Thai, 2022).

In this study, a feedforward neural network is adopted to train the datasets which are built by finite element simulations. In figure 2 an example of a neural network represented by x-h-h-y is shown, where x is the number of inputs (variables), h is the number of neurons for one of the two hidden layers, and y is the number of outputs (prediction) (Mostafa et al., 2022). In the hidden layers, the input variables are assigned weights which need to be determined and then used to predict. Activation (sigmoid) functions, such as the nonlinear continuous sigmoid, the tangent sigmoid, and the logarithmic sigmoid, are also introduced (Haykin, 2009). The inputs are multiplied by weights to provide the values of the output layers, within acceptable accuracy (low error margins). The iterative process of assigning weights is called training. Equation (5) shows a generic neuron j in a hidden layer, where w_{ij}^h is the weight that connects the i^{th} neuron of the current layer to the j^{th} neuron of the following layer, x_i^h is the input variable, b is the bias associated with the j^{th} neuron to adjust the output along with the weighted sum, and f is the activation function (Mostafa et al., 2022). Equations (6) and (7) provide some of the commonly adopted activation functions, the tangent sigmoid and logarithmic sigmoid, respectively.

$$y_j^h = f(\sum_{i=1}^n w_{ij}^h x_i^h + b_j^k) \quad (5)$$

$$f(u) = -1 + \frac{2}{(1+e^{-2u})} \quad (6)$$

$$f(u) = \frac{1}{(1+e^{-u})} \quad (7)$$

In this study, the Levenberg–Marquardt backpropagation algorithm (Hagan and Menhaj, 1994, Marquardt, 1963) is adopted to perform the training. The algorithm consists of two steps: (a) feed-forward weight values are determined to calculate the error by minimizing the least squares error function, and (b) propagating back the error to previous layers and checking if the error value falls outside the acceptable error margin. This iterative process (epoch) of backpropagation is repeated until the errors from the interconnecting weights are within the acceptable error margin. The fixed interconnecting weights now form a neural network which can be used to predict complex problems with certain accuracy. A schematic diagram of the whole training process is shown in figure 3.

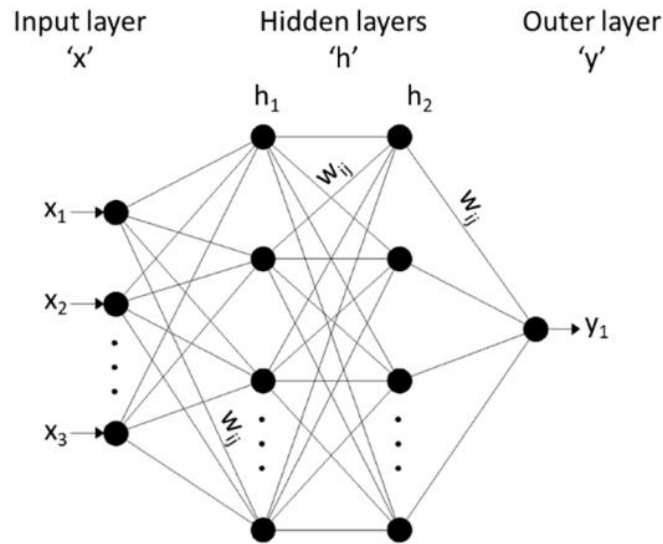


Figure 2: Feed-forward neural network architecture.

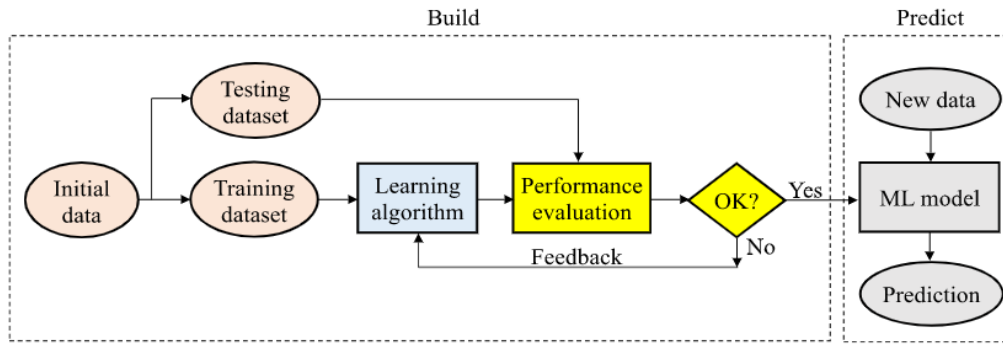


Figure 3: Typical workflow of machine learning (Thai, 2022).

4. The proposed data-driven scheme

The implementation in this article of data-driven structural analysis for masonry arches using machine learning principles, involves the use of a commercial finite element software (Abaqus) as well as Matlab (Matlab, 2021) and Python. A code developed in Matlab is the pillar of the procedure, since it provides the overall definition of the parameters used by the finite element models, it establishes the connection with the parametric finite element model, and it implements machine learning. In Appendix 9.1 of the article, descriptions for this Matlab code are given.

The finite element software is only used to conduct the parametric structural simulations of masonry arches. A Python script is also introduced within the Matlab code, to call the finite element model without opening the GUI of the finite element software. Descriptions for this Python script are provided in Appendix 9.2 of the article. The details of the proposed scheme are presented below:

- In the first step, a Matlab script (Appendix 9.1) is developed to define the geometry of the masonry arch and extract the (x) and (y) coordinates of the vertices of the individual stone blocks that make up the structure. The span of the arch and the thickness of the masonry blocks are the parameters introduced to define the geometry and used in the parametric investigation. The extracted coordinates are saved in a text file.
- In the second step, the Matlab script calls a Python script with the finite element model (Appendix 9.2), derived from the commercial finite element software, without the need to open the GUI of the software. The Python script initiates the solution for the discrete finite element model of the masonry arch, which is generated by reading the coordinates from the text file (previous step). All the details of the non-linear finite element model are included in the Python script, namely, the boundary conditions (fixed supports), the applied loads (step-1: self-weight and, step-2: a vertical point load applied at $\frac{1}{4}$ of span), the mesh (bilinear quadrilateral elements with size equal to 0.05m), and the unilateral contact/friction laws between the stones.
- In the third step, a second Matlab script (Appendix 9.3) calls the finite element software and runs a second Python script (Appendix 9.4) which extracts results from the finite element analysis solutions. The results extracted, include the (x)

and (y) displacements of the four vertices of each stone block after the completion of the first and second loading steps (self-weight only and self-weight plus vertical point load, respectively), and the ultimate load at collapse. It is noted that these (x) and (y) displacements are used to determine the deformed shape of the arch, after the end of each finite element analysis.

- In the fourth step, the results obtained from the finite element models are sorted and stored as mat files (Matlab) to form databases. These databases are trained using an artificial neural network which can then be used to predict the structural response of any masonry arch within the range of the database values.

A flowchart illustrating the steps of the whole process is shown in figure 4. The path marked by the dashed-line (red) arrows in figure 4 shows the workflow/application of the trained neural network.

All the simulations were run on a computer with quad-core Intel® Xeon E5520 at 2.266 GHz and 16 GB RAM. The computational time needed to predict the structural response using the trained neural networks is about 25.5s.

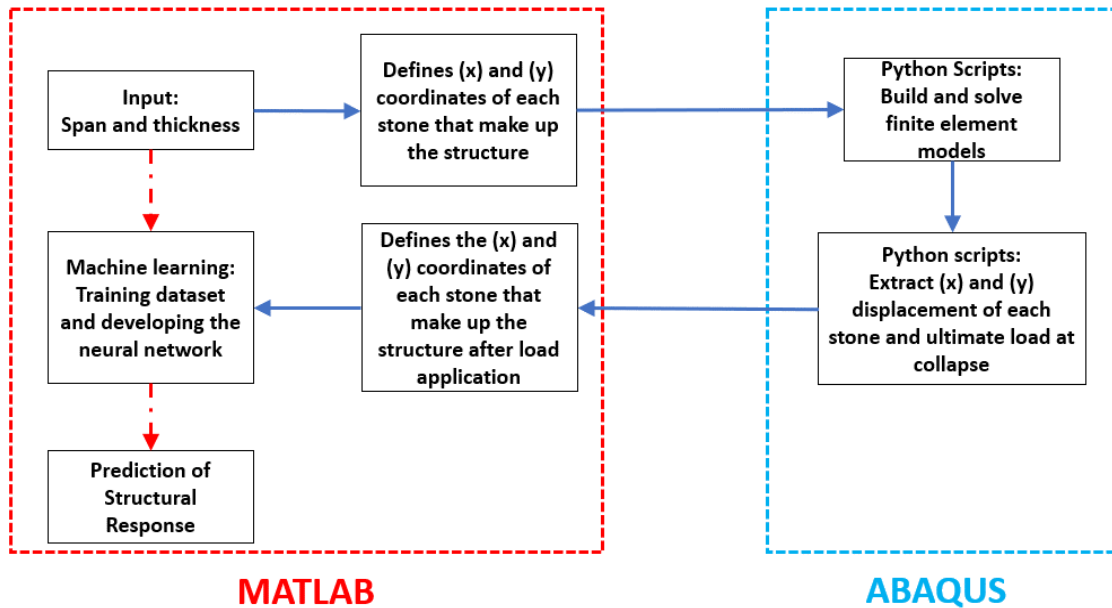


Figure 4: Flowchart of the proposed workflow.

5. Details of the parametric finite element analysis simulations

In this work, unilateral contact-friction interfaces are introduced between adjacent blocks to simulate the failure modes which are shown in figure 1. Sliding and/or opening of these interfaces, lead to the formation of hinges (the thrust line falls outside the section of the stone), depicting failure for two-dimensional masonry arches. To solve this unilateral contact-friction problem, the Lagrange multipliers method is adopted for simulating opening in the normal direction of the interfaces and the penalty method is used for

simulating sliding in the tangential direction. In addition, a surface-to-surface discretization is adopted for the masonry surfaces (edges) at each unilateral contact and friction interface. Due to the introduction of the unilateral-contact and friction interfaces the finite element model is non-linear, noticing that no material non-linearity and small displacement analysis are also considered. The Newton-Raphson incremental-iterative procedure is used to solve this problem.

To implement the parametric finite element simulations and generate the datasets, 1862 non-linear finite element models of circular arches and 550 models of parabolic arches have been developed to provide a holistic insight in the structural response of masonry arches, emphasizing in potential collapse mechanisms. Within the adopted discrete approach, 20 two-dimensional masonry blocks have been used to create each masonry arch. It is noted that in previous studies (Charalambidi et al., 2022, Tapkın et al., 2022), it has been shown that using more blocks than a chosen number, may not significantly affect the structural response, while it can increase the computational cost.

Due to its low tensile resistance, the mortar is neglected in the models developed for this study. Two steps are used to introduce static loads on the structure: the first step introduces a pure gravity load to simulate the state of inertia of the structure and the second step adds an incrementally applied point load at $\frac{1}{4}$ of the span.

Concerning the failure response of the masonry arches, as this arises from the used discrete finite element models, it is noticed that the ultimate strength is reached when parts of the structure lose contact and develop rigid body displacements. This happens due to the fact that the defined unilateral contact/friction boundary constraints (assigned between stones) become insufficient to equilibrate the loaded structure. On the numerical model, as collapse is being reached, at least one zero eigenvalue on the tangential stiffness matrix is introduced which makes the analysis unstable.

6. Building the artificial neural networks

In this study, three neural networks have been trained, validated and tested to predict the structural response of circular masonry arches. Each trained neural network will provide a different insight about the structural behaviour of the arch. The first neural network will be used to predict the deformed geometry of the structure when subjected to self-weight only. The second neural network will predict the deformed geometry when the structure is subjected to self-weight plus a vertical load applied at $\frac{1}{4}$ of span. The third neural network will be used to predict the ultimate (failure) load at collapse, when the structure is subjected to self-weight plus a vertical load applied at $\frac{1}{4}$ of span. The same process is repeated, and another three neural networks are also trained, to predict the response of parabolic arches. Then, a variable is introduced in a Matlab script, to establish the connection between the chosen shape, circular or parabolic, and the corresponding trained neural networks. For example, when the user selects this variable to be equal to “circular”, the trained neural networks which correspond to the circular arch datasets is called and predict the response of a random arch geometry. A similar process is followed for a parabolic arch shape or any other arch shape that may potentially be added to the dataset to widen the scope of the scheme.

To train the mentioned artificial neural networks, results derived from the finite element simulations, were extracted and used. In the input layer of each neural network are added

the span and thickness of the masonry blocks representing the initial geometry of each arch. Figure 5 shows Matlab plots of the vertices that make up the intrados and extrados of 5.0m span and 0.25m masonry ring thicknesses for a circular and a parabolic shape. In the output layer of the first two neural networks, are included the x - and y -displacements of each vertex on each of the 20 individual masonry stones, representing the deformed geometry at the end of each loading step of the finite element analysis. In particular, the x - and y - displacements values at the end of loading step-1 were used to build the neural network that predicts the deformation of the structure when subjected to self-weight only and the x - and y - displacements values at the end of loading step-2 were used to build the neural network that predicts the deformation of the structure when subjected to self-weight plus a vertical point load. In the third neural network, the output layer was defined by the ultimate load which is obtained at the end of the finite element analysis. Figure 6 shows the deformation of a 5.0m span circular arch with 0.25m thickness, subjected to self-weight and a vertical load applied at $\frac{1}{4}$ of the span. This figure is derived from one of the parametric finite element simulations, developed to create the databases that will be used to train the artificial neural networks. It is noted, that the opening and sliding between the masonry blocks as depicted in figure 6, can also be predicted and shown by the trained neural networks.

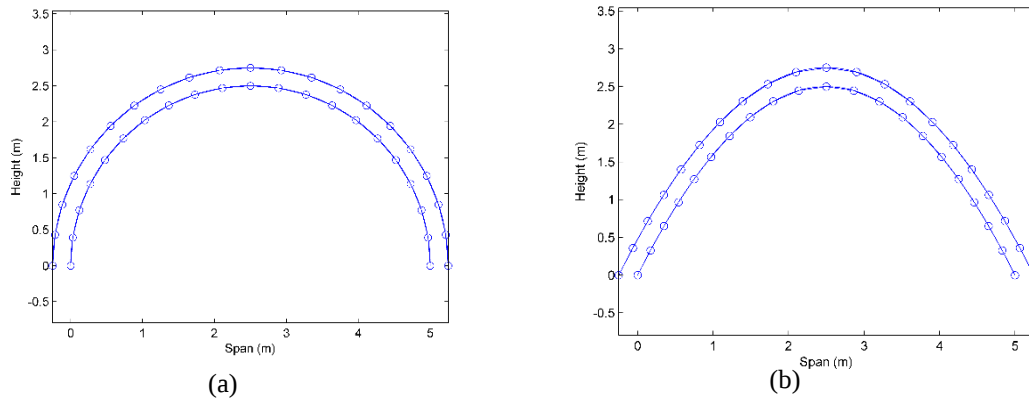


Figure 1: Matlab plot of the vertices that make up the intrados and extrados of a) a circular and b) a parabolic arch with 5m span and 0.25m ring thickness.

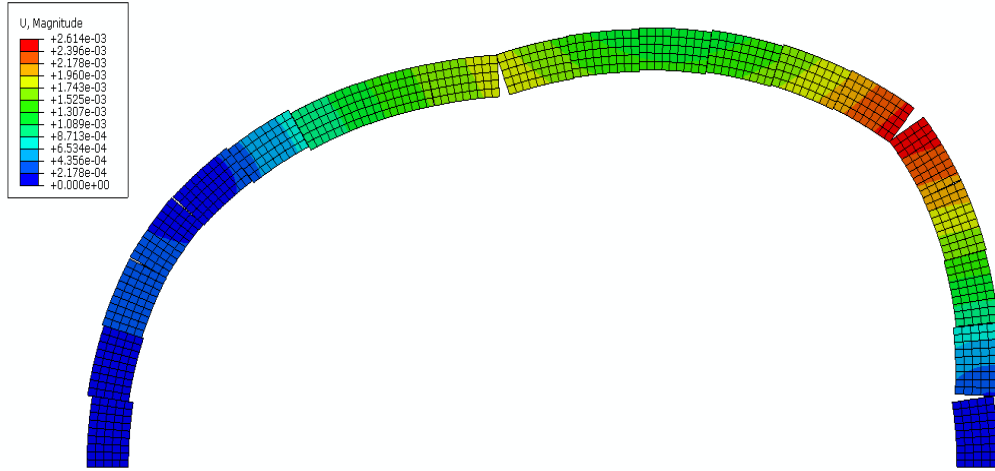


Figure 2: Deformation of 5m span arch with 0.25m thickness subjected to self-weight and a vertical load applied at 1/4 of the span.

6.1 Methodology and parameters adopted to train the artificial neural networks

In this study, the Levenberg-Marquardt algorithm is adopted to train the neural networks. This is considered as one of the fastest training algorithms (Matlab, 2021) but requires more memory than other techniques available. It uses Jacobian matrix to compute the solution and assumes that the performance function is the mean or sum of square errors. Like the quasi-Newton methods, second-order training speed can be achieved without solving the Hessian matrix (Liu et al., 2021). The Hessian matrix is approximated by equation (8) when the performance function is provided by the sum of squares errors and the gradient can be computed as Jacobian matrix multiplied by the vector of network errors, see equation (9).

Equation (10) shows how the Levenberg-Marquardt algorithms approximate the Hessian matrix (Hagan and Menhaj, 1994, Hagan et al., 1997) by combining the Gradient Descent and Newton-Raphson method. When μ is zero, equation (10) is transformed to Newton's method, using the approximate Hessian matrix. When μ is large, equation (10) forms Gradient Descent with a small step size. The algorithm is faster and more accurate when μ is small since Newton's method is quick when approaching the true value. With each successful iteration (epoch), the performance function is reduced unless the tentative step is not successful thus increasing the performance function. The aim to keep reducing μ makes the algorithm fast.

Table 1 shows the parameters used to train the neural networks. It should be noted that the neural networks were re-trained multiple times to improve the results, since during re-training different initial conditions and sampling were considered. The 70/15/15 rule was used during the training process, which states that 70% of the dataset is used for training, 15% is used for validating the neural network and the remaining 15% is reserved for testing the neural network.

$$H = J^T J \quad (8)$$

$$g = J^T e \quad (9)$$

$$x_{k+1} = x_k - [J^T J + \mu I]^{-1} J^T e \quad (10)$$

In equations (8)-(10), H is the approximated Hessian matrix, J is the Jacobian matrix, g is the gradient, I is the identity matrix, e is the vector of network errors, and μ is the adaptive value.

Table 1: Parameters used to train the neural networks.

Parameter name	Value
Number of neurons in hidden layer	40
Maximum number of epochs to train	1000
Performance goal	0
Maximum validation failures	100
Minimum performance gradient	1e-7
Maximum value for μ	1e10
Initial μ	0.001
Decrease factor for μ	0.1
Increase factor for μ	10

6.2 Using the artificial neural networks

After training, validating, and testing the neural networks, they can be used to predict the structural response. The final deformed geometry of the structure is practically determined using equation (11) where u_o is the vector of coordinates of the vertices of the masonry blocks depicting the undeformed geometry before any load is applied, u_i is the vector of the displacements of the vertices of the masonry blocks after the load application and C is a user defined scale factor to ensure the deformation of the structure is easily visible. The vector u_i is predicted by the neural network and is dependent on the geometry of the structure and the load application.

Figure 7 shows the deformation of a 5.0m circular span arch with 0.25m thickness, subjected to self-weight and a vertical load applied at $\frac{1}{4}$ of the span, as derived by using the equation (11). A scale factor of 100 is used in this example so that the hinge formation can easily be seen. It should be emphasized that figure 7 indicates the capacity of the proposed approach to predict the deformed shape and the collapse mechanism of a masonry arch for a random geometry, using the proposed data-driven scheme. The Matlab code which is used to generate the deformed geometry, with inputs the vectors of the initial coordinates of the vertices of the masonry blocks u_o and the vector u_i of the displacements of the vertices of the masonry blocks after the load application, is given in Appendix 9.5.

$$u_j = u_o + C \cdot u_i \quad (11)$$

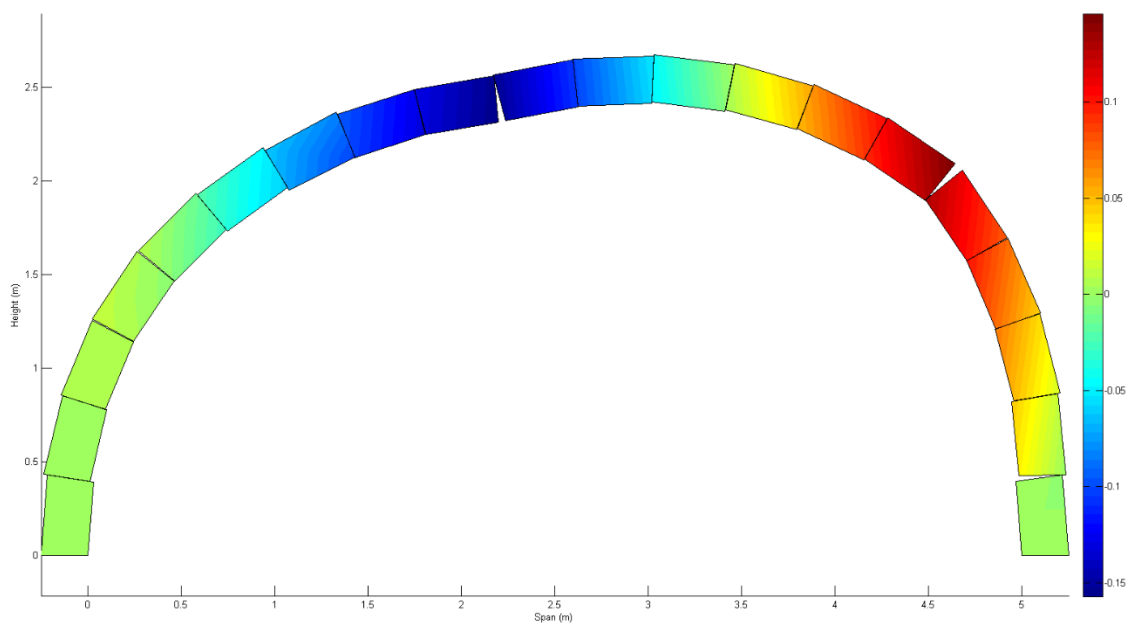


Figure 3: Deformation of 5m *circular* span arch with 0.25m thickness subjected to self-weight and a vertical load applied at 1/4 of the span, as derived by using equation (7).

7. Results and discussions

In this section, the performance and training accuracy of the developed neural networks are discussed. Then, it is shown how the trained neural networks can be used to predict the structural response of masonry arches of random geometry. Relevant results are provided and compared with existing output from literature.

7.1 Performance of the trained artificial neural networks

7.1.1 Circular arches

Three neural networks, namely, A, B and C, were trained by using 1304 data points, validated by 279, and tested by 279 data points respectively. In all neural networks, 2 input variables were used, namely, the span and the thickness of the masonry blocks that represent the arch geometry. In the first 2 neural networks (A and B) which are used to predict the deformed geometry under self-weight or self-weight and vertical loading, 160 output variables were used, namely, the displacements of the vertices of each of the 20 individual blocks making up the arches: 20 blocks x 4 vertices per block x 2 displacements per vertex. The deformed geometry of the arches can then be determined using theses 160 output variables, according to relation (11). In the third neural network (C), 1 output variable is considered, namely, the ultimate load.

In table 2, are provided details related to the training of the three neural networks. The neural network A in table 2 refers to the neural network that predicts the deformation due

to self-weight only, the neural network B refers to the one predicting the deformation due to self-weight and a vertical point load and the neural network C refers to the neural network that predicts the ultimate load at collapse.

Regarding the training times given in table 2, neural network C depicted a shorter training time as compared to the other two networks, since the output layer of network C had only one variable, the ultimate load at collapse, comparing to the 160 variables of the output layer of the neural networks A and B of table 2.

As shown in table 2, the training, validation, and testing of the neural networks are accurate, with neural networks A and C showing more than 98% accuracy and with neural network B showing more than 95% accuracy. The neural networks were trained four times to increase accuracy, with each proceeding training done from the previously trained neural network without reinitializing and starting weights from zero. In addition, the mean squared error obtained from the training of the networks is very small.

Table 2: Summary information from the training process of the three neural networks (circular arches).

	Neural Network A	Neural Network B	Neural Network C
Training time	1hr:10min:04sec	2hr:26min:41sec	15min:12sec
Iterations of train(epoch)	19	126	218
Training accuracy	98.63%	95.92%	99.23%
Validation accuracy	99.91%	96.14%	99.15%
Testing accuracy	99.86%	94.91%	99.26%
Mean Squared Error (MSE)	0.0001%	0.039%	0.0005%

Figures 8-10, show the regression plots for the training, validation and testing of the neural networks and how the trained neural network fit the dataset. From these figures, it is observed that the regression for training, validation and testing of the neural networks is almost 1, with 1 representing zero error in the trained neural network.

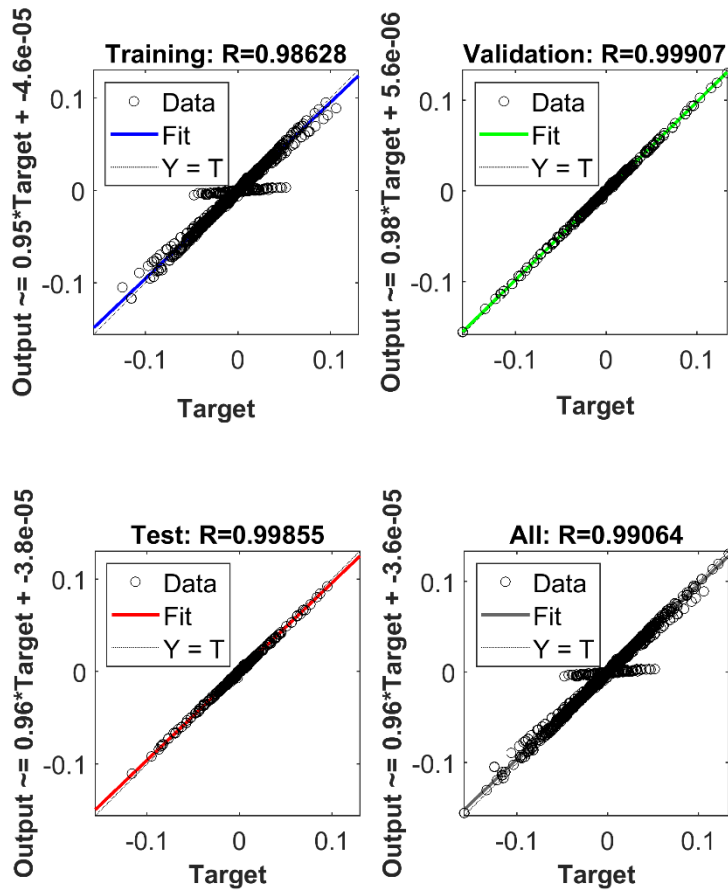


Figure 4: Regression plot for neural network A (circular arches).

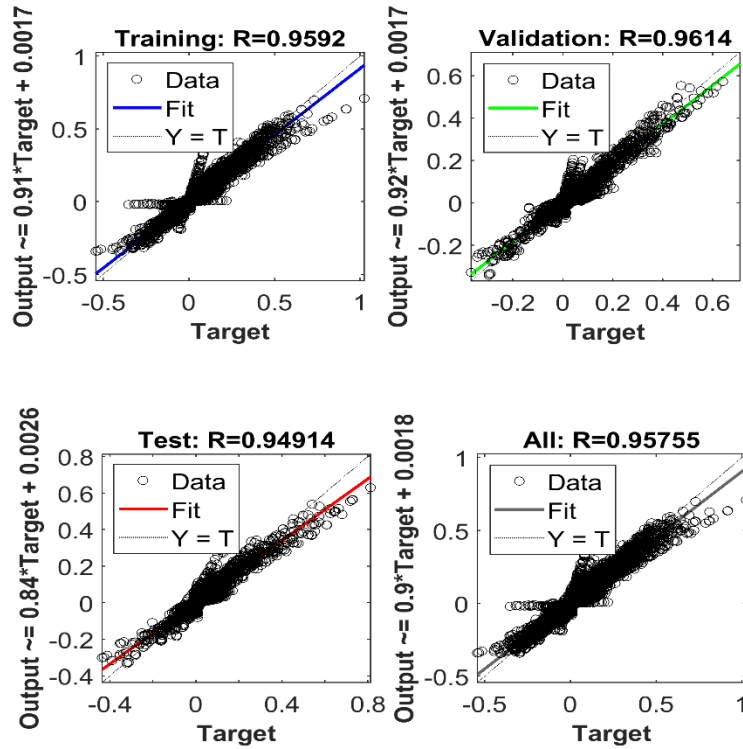


Figure 5: Regression plot for neural network B (circular arches).

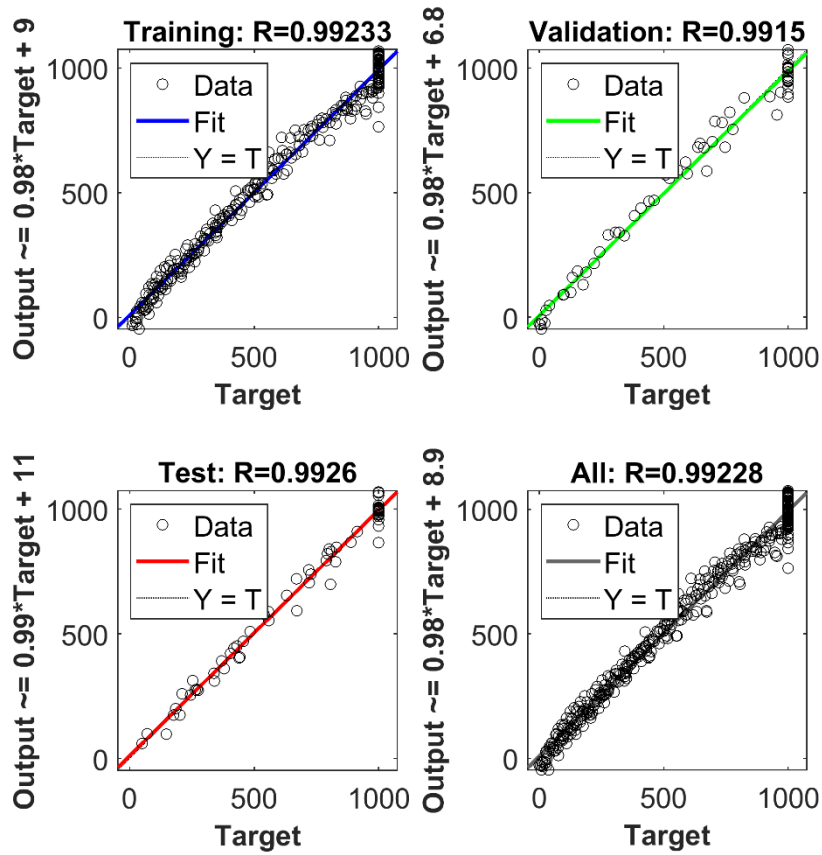


Figure 6: Regression plot for neural network C (circular arches).

7.1.2 Parabolic arches

In table 3 are provided details related to the training of the three neural networks. As previously mentioned, the neural network A in table 3 refers to the neural network that predicts the deformation due to self-weight only, the neural network B refers to the one predicting the deformation due to self-weight and a vertical point load and the neural network C refers to the neural network that predicts the ultimate load at collapse.

Regarding the training times given in table 3, neural network C depicted a shorter training time as compared to the other two networks, since the output layer of network C had only one variable, the ultimate load at collapse, comparing to the 160 variables of the output layer of the neural networks A and B of table 3.

As shown in table 3, the training, validation, and testing of the neural networks are accurate, with neural networks A and C showing more than 99% accuracy and with neural network B showing more than 98% accuracy. The neural networks were trained four times to increase accuracy, with each proceeding training done from the previously trained neural network without reinitializing and starting weights from zero. In addition, the mean squared error obtained from the training of the networks is very small.

Table 3: Summary information from the training process of the three neural networks (parabolic arches).

	Neural Network A	Neural Network B	Neural Network C
Training time	46min:30sec	4hr:29min:15sec	15min:12sec
Iterations of train(epoch)	9	54	471
Training accuracy	99.32%	98.79%	99.99%
Validation accuracy	99.34%	98.63%	99.96%
Testing accuracy	98.89%	98.63%	99.999%
Mean Squared Error (MSE)	0.0005%	0.0002%	0.00001%

Figures 11-13, show the regression plots for the training, validation and testing of the neural networks and how the trained neural network fit the dataset. From these figures, it is observed that the regression for training, validation and testing of the neural networks is almost 1, with 1 representing zero error in the trained neural network.

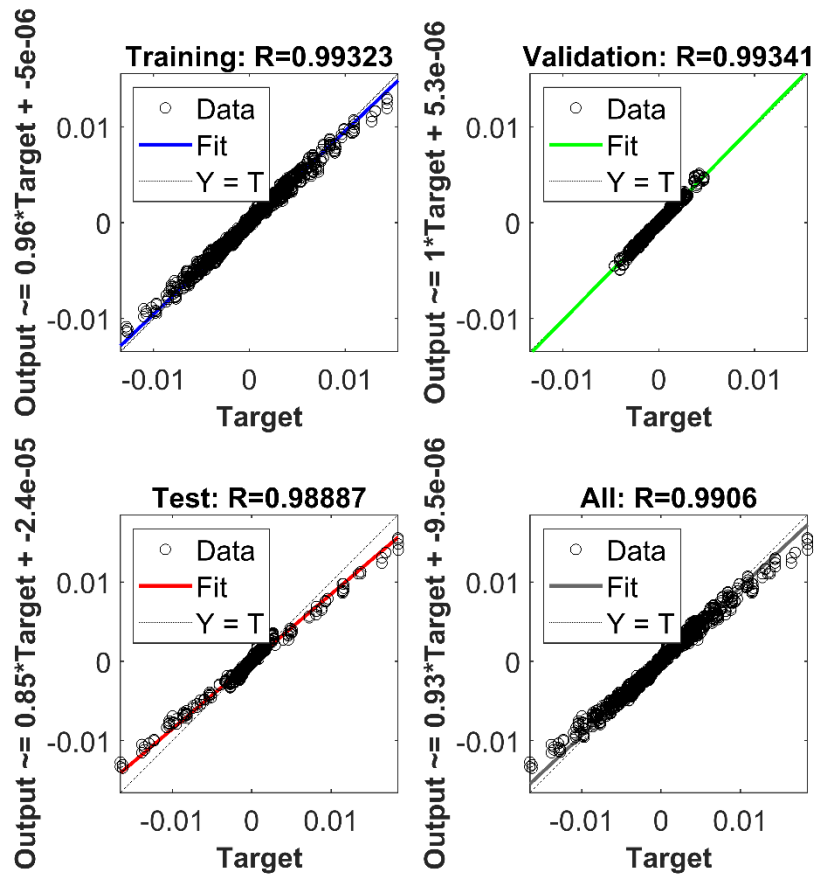


Figure 11: Regression plot for neural network A (parabolic arches).

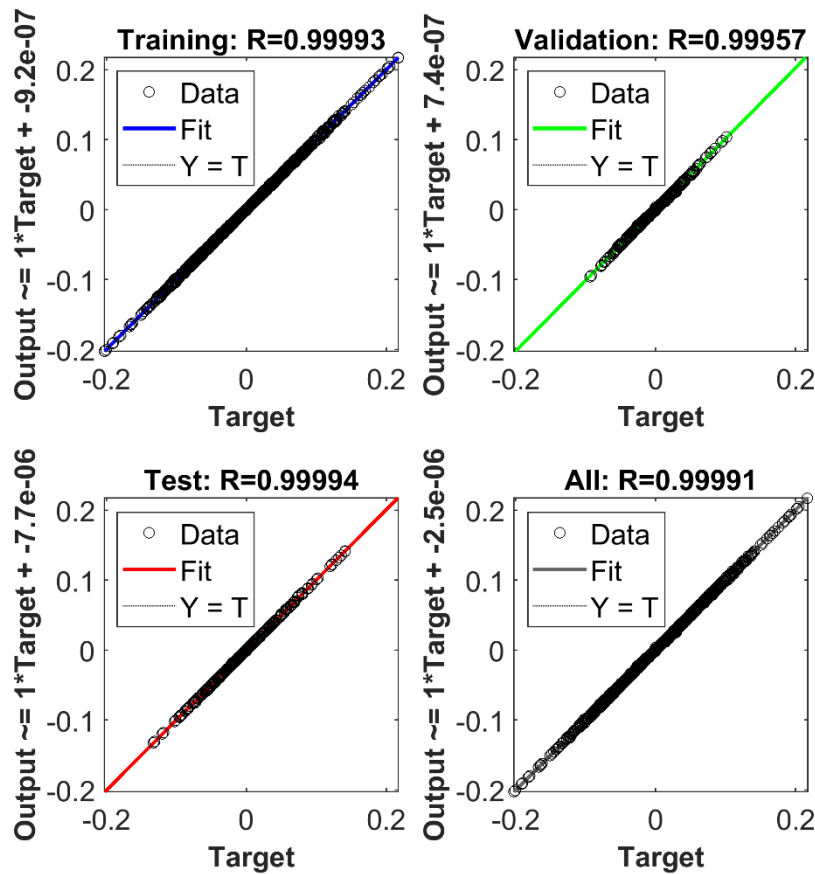


Figure 12: Regression plot for neural network B (parabolic arches).

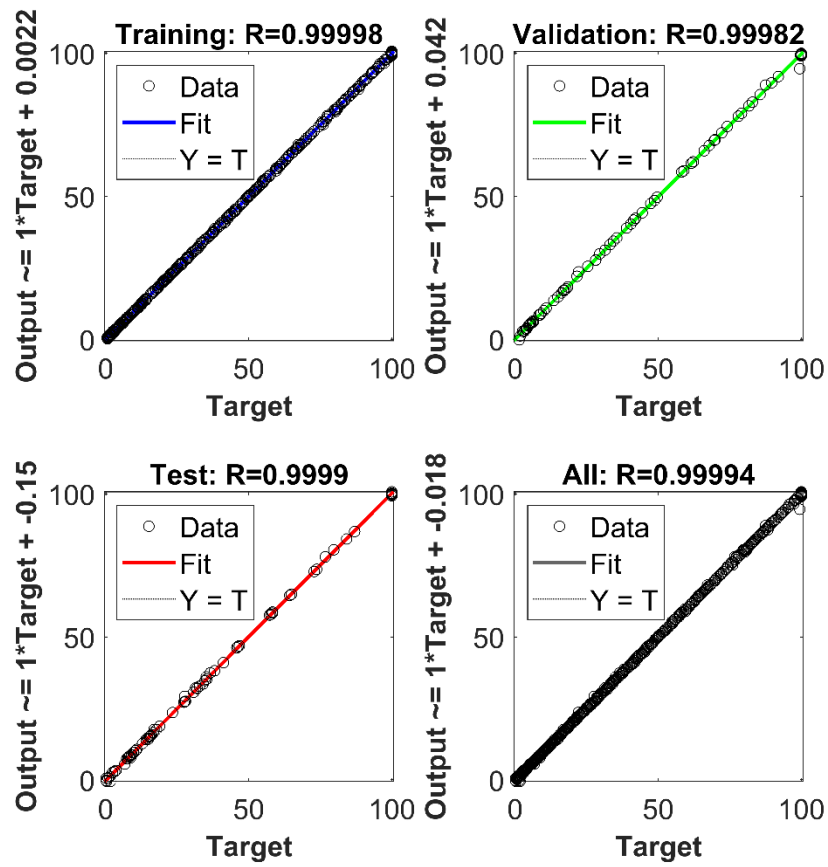


Figure 13: Regression plot for neural network C (parabolic arches).

7.2 Predicting the response of random masonry arches using the proposed data-driven scheme

This section investigates the structural behaviour of eight masonry arches, six circular and two parabolic arches, as predicted by the trained neural networks. The dimensions of the selected structures are within the range of the dataset values (1.5m-to-50m span, and 0.1m-to-1m masonry ring thickness). For the masonry arch geometries 5, 6 and 7 shown in table 4, the predicted by the proposed metamodel collapse mechanism and ultimate load, are compared with the results obtained from finite element analysis using commercial software.

In addition, arches 1 and 3 in table 4 are based on the minimum stone thickness for a circular arch to maintain stability under self-weight as proposed by (Couplet, 1729) and (Milankovitch, 1904, Milankovitch, 1907), respectively. (Couplet, 1729) proposed that the theoretical minimum thickness, t , of a circular masonry arch with radius, R , should be $t/R=0.1075$. Several years later, (Milankovitch, 1904, Milankovitch, 1907) proposed that the theoretical minimum thickness for a monolith arch should be $t/R=0.10748$.

In table 4 below, are provided the geometry of the selected arches as well as the ultimate load at collapse, which is predicted from the neural networks, when a vertical point load is applied at the quarter span.

Table 4: Geometry of masonry arches tested on neural networks and predicted ultimate load.

Name	Span (m)	Height (m)	Stone thickness (m)	Ultimate Load (kN)	Source
Arch 1 (circular)	2.3	1.15	0.12	0	(Couplet, 1729)
Arch 2 (circular)	16.0	8.0	1.0	100	-
Arch 3 (circular)	6.0	3.0	0.32	13.4	(Milankovitch, 1904, Milankovitch, 1907)
Arch 4 (circular)	12.0	6.0	0.5	9.1	-
Arch 5 (circular)	20.2	10.1	0.84	35.9	-
Arch 6 (circular)	10.4	5.2	0.45	8.8	-
Arch 7 (parabolic)	10.4	5.2	0.82	96.05	-
Arch 8 (parabolic)	15.25	7.63	0.45	20.7	-

The deformed shape of the arch 1 which is presented in figure 14, shows that the arch is highly unstable, since it collapses under its self-weight. It is noted that the deformation of

599 this arch is derived using the neural network which predicts the deformation due to the
 600 self-weight loading (neural network A of table 2). Then, once the neural network which
 601 predicts the ultimate load is used (neural network C of table 2), a zero load is obtained. A
 602 classical hinge failure mechanism of 5 hinges is obtained due to self-weight loading. This
 603 is a potential type of collapse, as found in literature when a symmetric, circular arch is
 604 subjected to symmetrical loading, e.g. self-weight (Cocchetti et al., 2012, Focé and
 605 Huerta, 2005, Heyman, 1995). A similar, five-hinge collapse mechanism is depicted in
 606 Figure 15 for a circular arch with the theoretical minimum thickness, as proposed in
 607 (Milankovitch, 1904, Milankovitch, 1907, Couplet, 1729).
 608
 609
 610

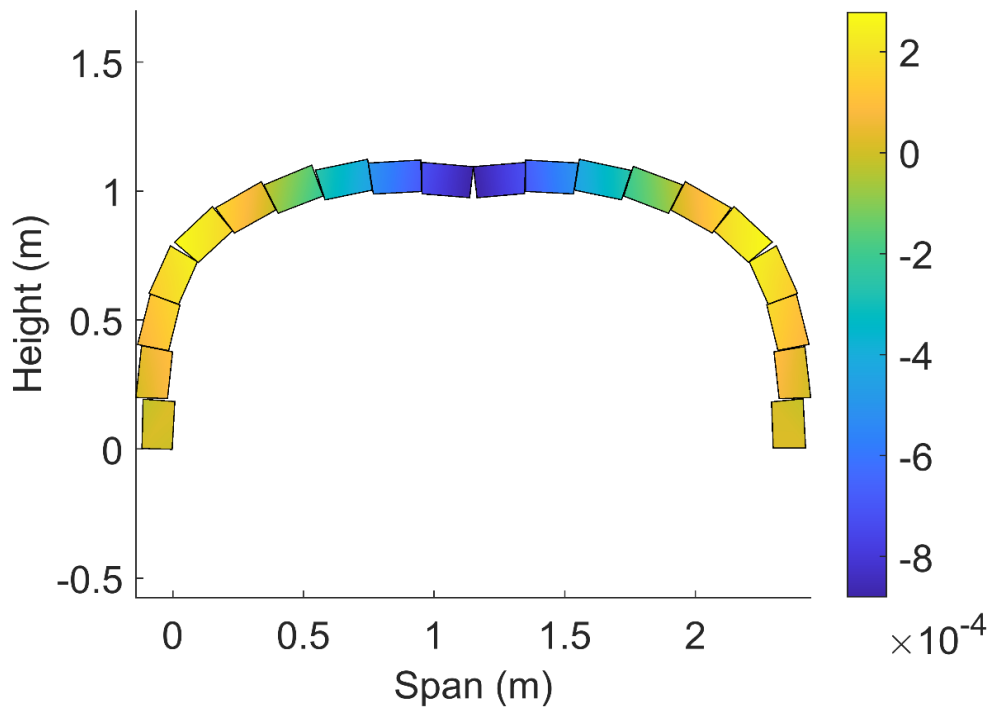


Figure 14: Deformation (m) of the arch 1 of table 4 (2.3m span, 0.12m thickness) due to self-weight only, when scale factor = 200.

613

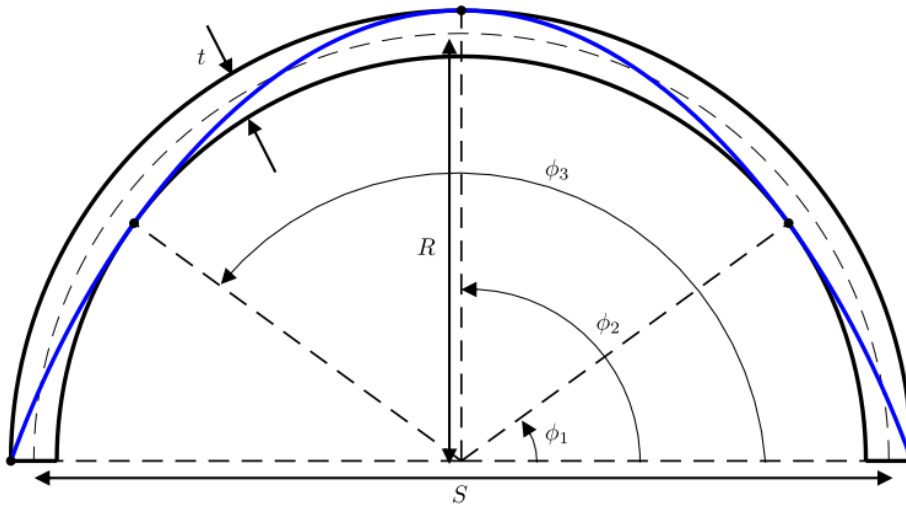


Figure 75: Five hinge mechanism for a circular masonry arch under self-weight based on literature. (McLean et al., 2021).

614

615

616 In figures 16-17, the deformed geometries of the arch 2 of table 4 due to self-weight only,
 617 as well as due to self-weight and a vertical point load, are shown. From the deformed
 618 shape due to self-weight (figure 16), is noticed that no hinge formation can be seen.
 619 According to figure 17, though some hinges have been developed in the arch, the four
 620 hinges mechanism is not fully developed at this load level, indicating that the arch is able
 621 to fully support the total applied load.

622

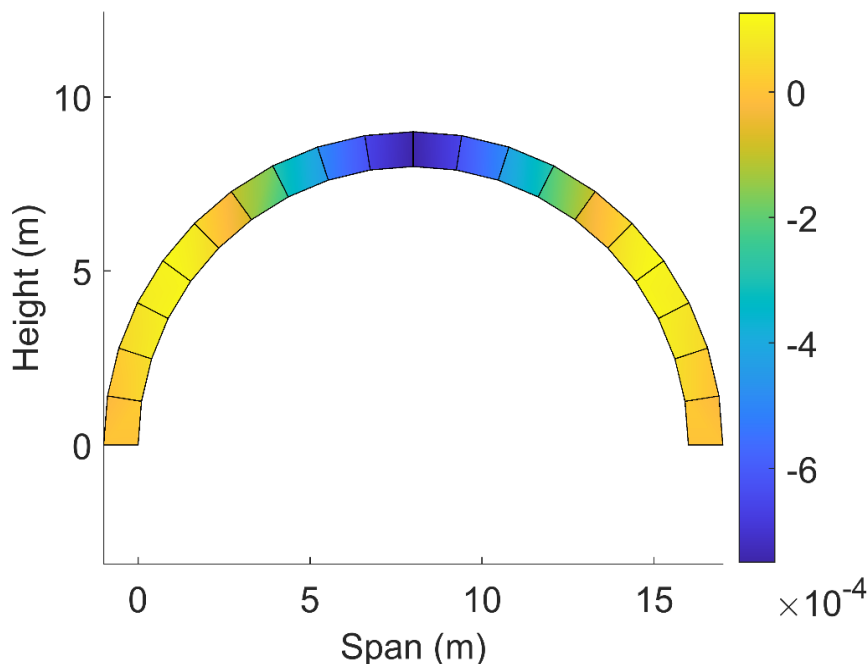
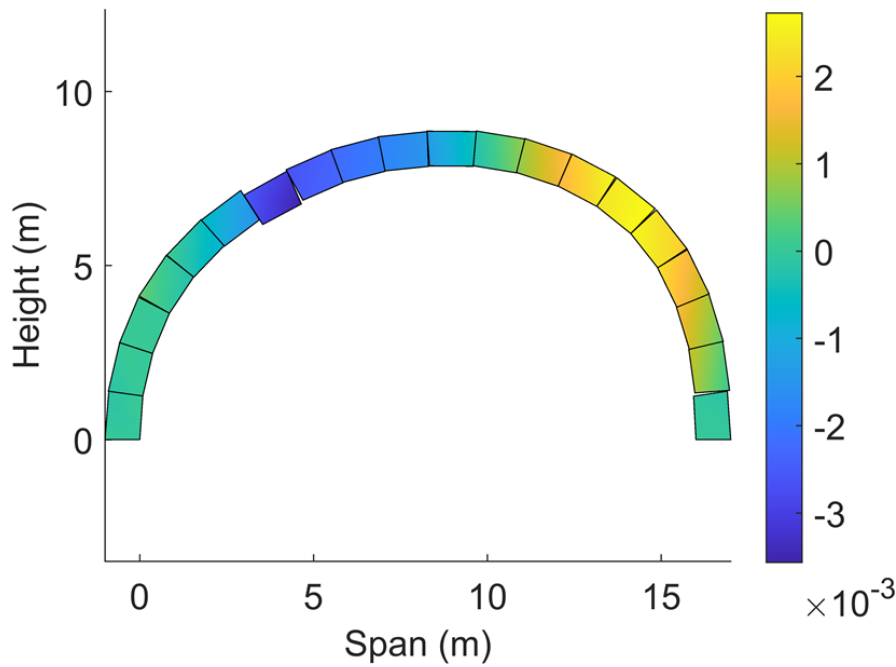


Figure 16: Deformation (m) of the arch 2 of table 4 (16m span, 1.0m thickness) due to self-weight only, when scale factor = 1.

623
624



625
626

Figure 17: Deformation (m) of the arch 2 of table 4 (16m span, 1.0m thickness) due to self-weight and vertical load, when scale factor = 100.

627

628 In figures 18 and 20, the deformed geometries of arches 3 and 4 due to self-weight only
629 are shown. These are followed by figures 19 and 21 depicting the deformed geometry of
630 the same arches, due to self-weight and the vertical point load. When arches 3 and 4 are
631 subjected to self-weight and a vertical point load, the deformed shapes of both arches as
632 shown in figures 19 and 21, indicate that the four-hinge mechanism is developed. Thus,
633 the arches fail to support the overall vertical load. The predicted ultimate loads at
634 collapse, as obtained by the neural network C of table 2, are 13.4kN and 9.1kN for arches
635 3 and 4, respectively.

636

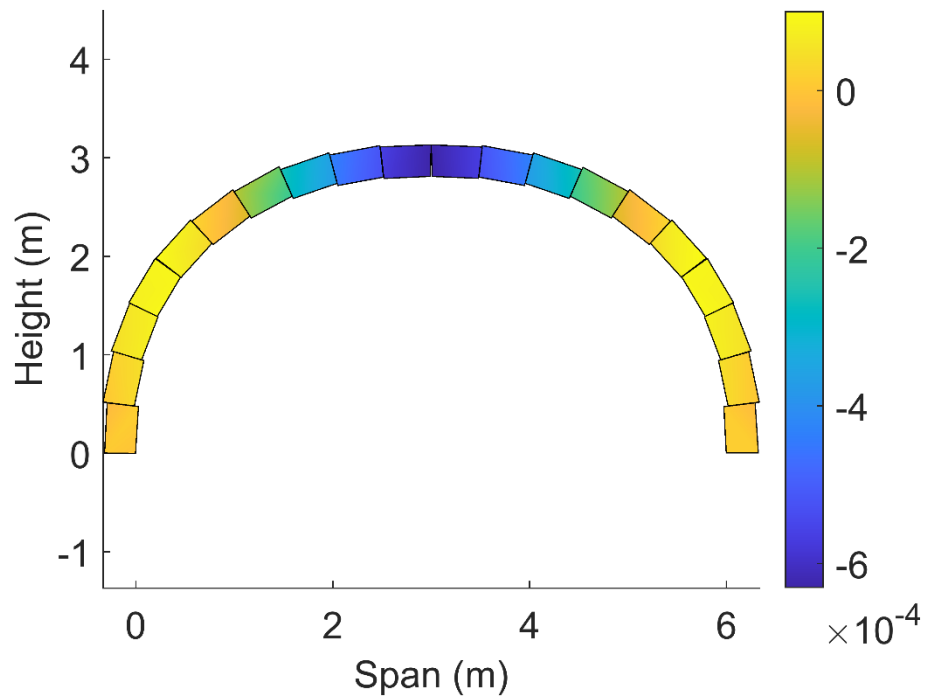


Figure 18: Deformation (m) of the arch 3 of table 4 (6m span, 0.32m thickness) due to self-weight only, when scale factor = 300.

637
638
639
640

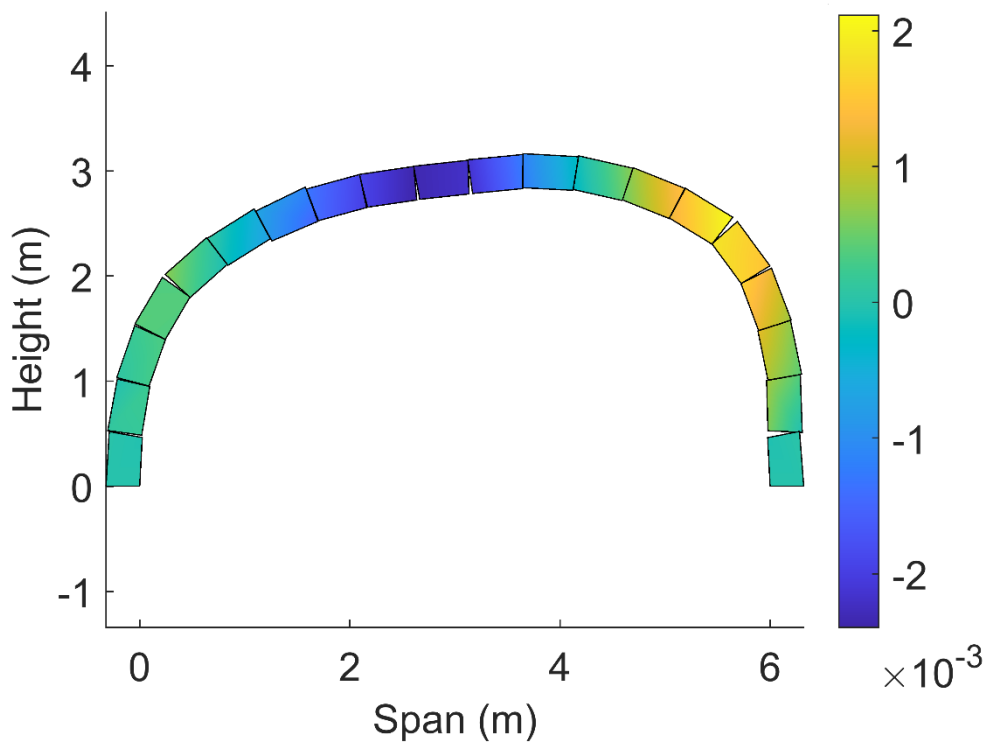


Figure 19: Deformation (m) of the arch 3 of table 4 (6m span, 0.32m thickness) due to self-weight and vertical load, when scale factor = 100.

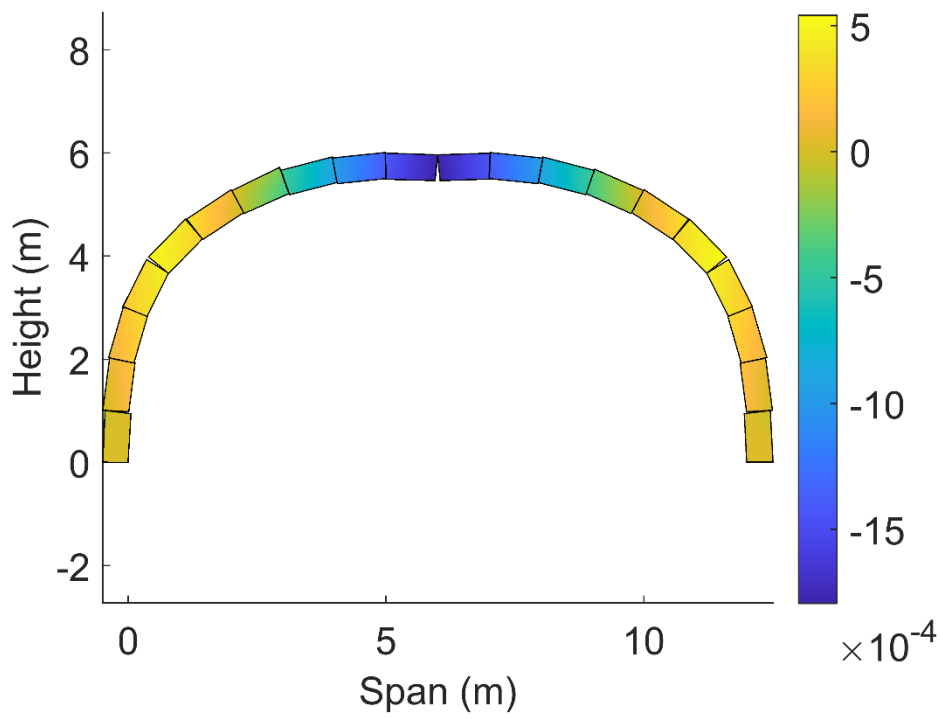


Figure 20: Deformation (m) of the arch 4 of table 4 (12.0m span, 0.5m thickness) due to self-weight only, when scale factor = 300.

641
642
643

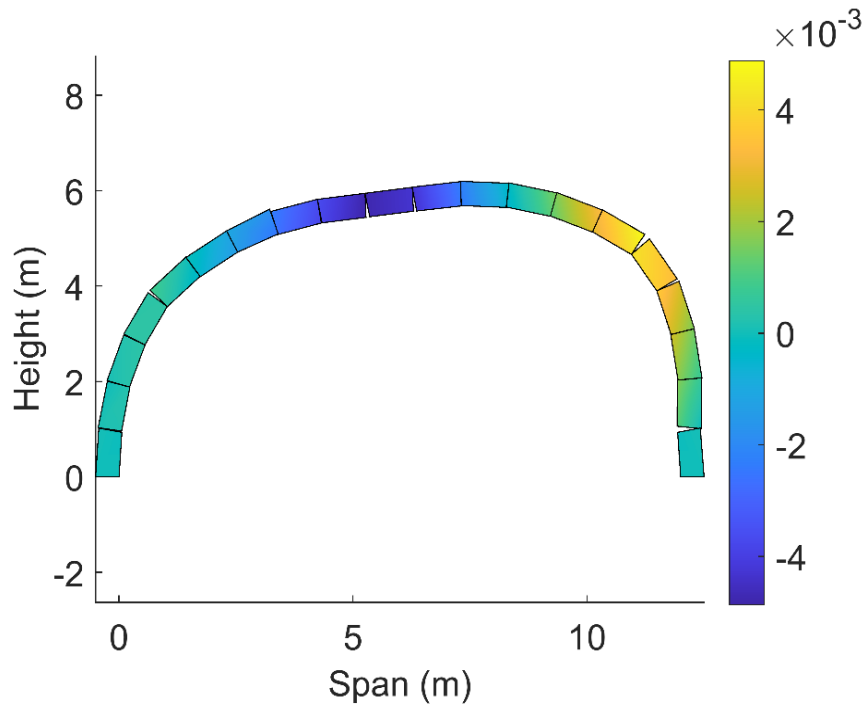


Figure 21: Deformation (m) of the arch 4 of table 4 (12.0m span, 0.5m thickness) due to self-weight and vertical load, when scale factor = 100.

644
645
646
647

In figures 22a and 23a, the deformed geometries of the arch 5 of table 4 due to self-weight only, as well as due to self-weight and the vertical point load, are shown, as obtained from the trained neural networks. To provide a comparison of the results which are derived by the predictions of the trained neural networks, the same arch has been simulated using finite element analysis implemented by the commercial software. Figures 22b and 23b show the deformed geometry of the arch 5 due to self-weight as well as due to self-weight and the point load, as obtained by finite element analysis. The geometry of the arch is stable under its self-weight as a symmetrical 3-hinge formation can be observed in figure 22. When the arch is subjected to self-weight and a vertical point load, the classical four-hinge collapse mechanism can be observed in figure 23. The predicted ultimate load (35.9kN, table 4) at collapse is comparable with the ultimate load (34.1kN) obtained from finite element analysis. It is noted that the ultimate load predicted by the neural network C of table 2 is slightly overestimated by 5.3%. The position of hinges, which are depicted in the neural network prediction and in the finite element analysis results, is also similar.

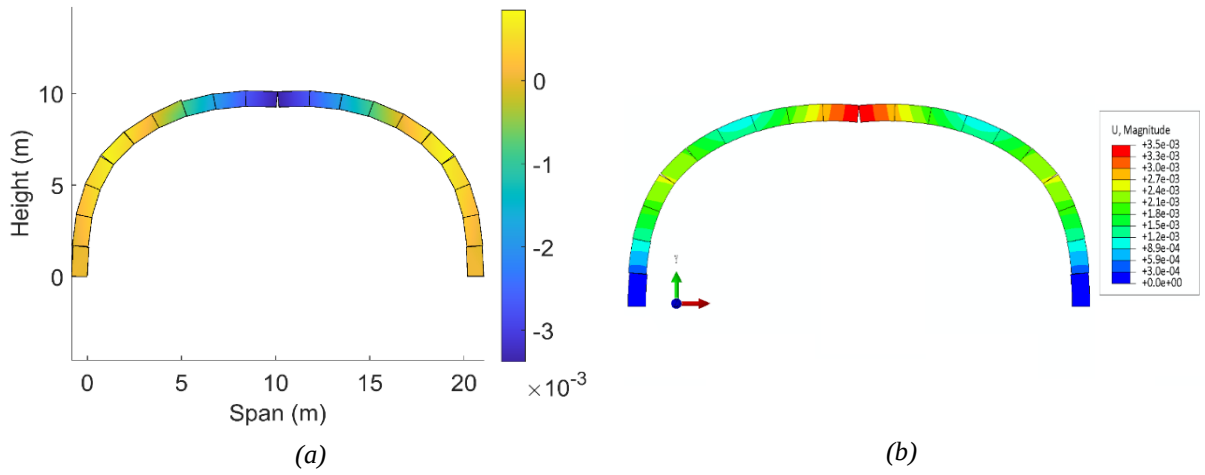


Figure 22: Deformation (m) of the arch 5 of table 4 (20.2m span, 0.84m thickness) due to self-weight only derived (a) from the trained artificial neural network (scale factor= 250), (b) from the finite element analysis (scale factor= 250).

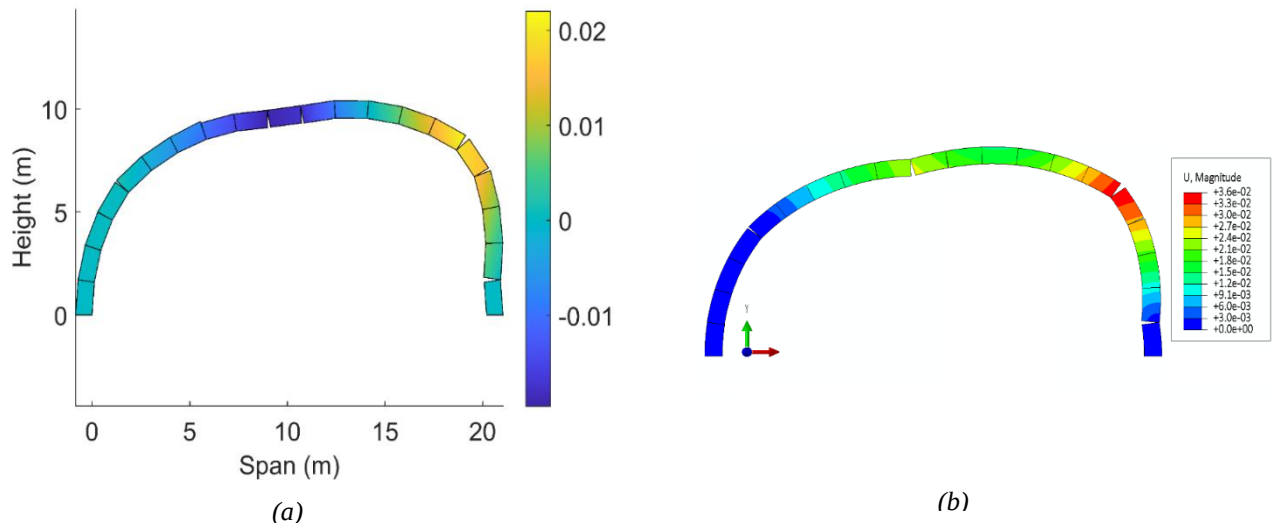


Figure 23: Deformation (m) of the arch 5 of table 4 (20.2m span, 0.84m thickness) due to self-weight and a vertical load derived (a) from the trained artificial neural network (scale factor= 45), (b) from the finite element analysis (scale factor= 45).

Another example is presented, providing a comparison of the ultimate failure load and collapse mechanism obtained from finite element analysis and the proposed metamodel. Figure 24 shows the deformed geometry of arch 6 due to self-weight and vertical point load as obtained from finite element analysis and as predicted by the neural network B of table 2, respectively. It can be noted that in both cases, the classical four-hinge collapse mechanism can be observed. The predicted ultimate load (8.8kN) at collapse is comparable with the ultimate load (10.4kN) obtained from the finite element model. It is noted that the ultimate load predicted by the neural network C of table 2 is conservative.

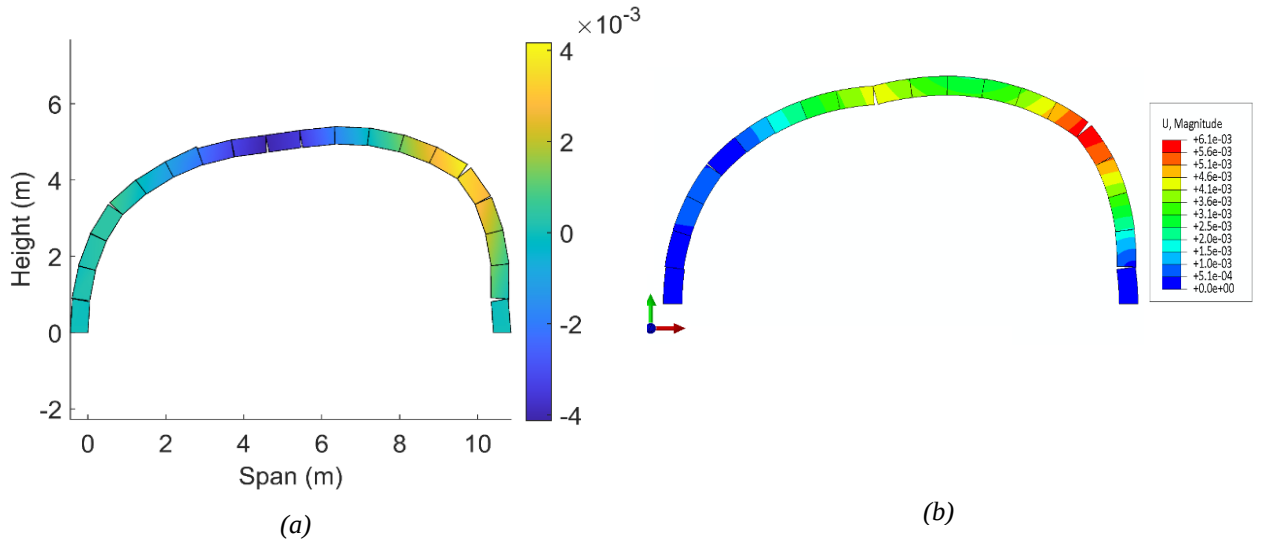


Figure 24: Deformation (m) of the arch 6 of table 4 (10.4m span, 0.45m thickness) due to self-weight and a vertical load derived (a) from the trained artificial neural network (scale factor=100), (b) from the finite element analysis (scale factor= 100).

To provide a holistic representation of the response of masonry arches obtained from different geometry shapes, results derived for parabolic arches, are presented next. In particular, the prediction of the response of the randomly chosen arch 7 of table 4 is given in figures 25 and 26, for self-weight and vertical point loading, respectively. As shown in these figures, the comparison between the machine learning prediction and finite element analysis is satisfactory in terms of the deformed shape, for both self-weight and vertical point loading, respectively. For the point load, a four-hinge mechanism arises as shown in figure 26. The ultimate load which is obtained from the machine learning scheme is equal to 96.05kN, that is close to the one derived from the finite element simulation (99.33kN).

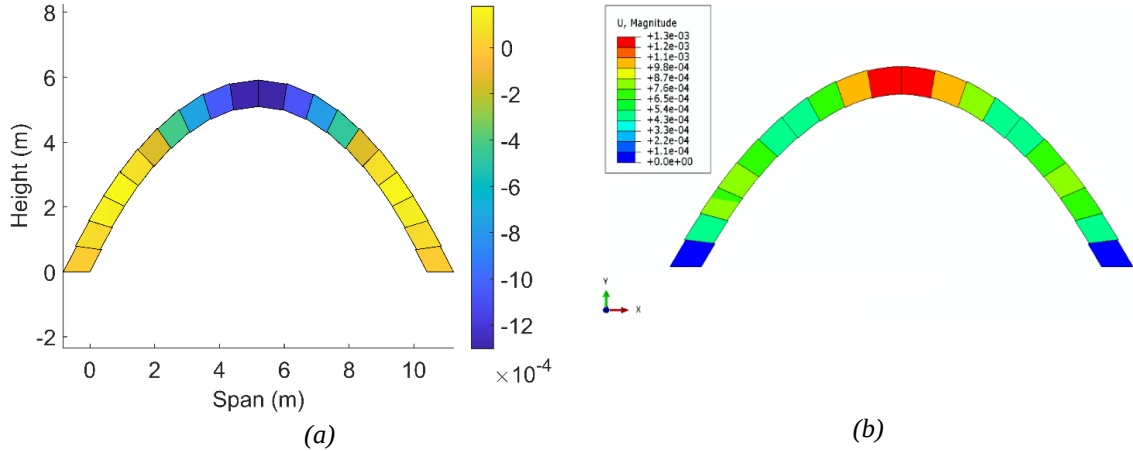


Figure 25: Deformation (m) of the arch 7 of table 4 (10.4m span, 0.82m thickness) due to self-weight derived (a) from the trained artificial neural network (scale factor=80), (b) from the finite element analysis (scale factor= 80).

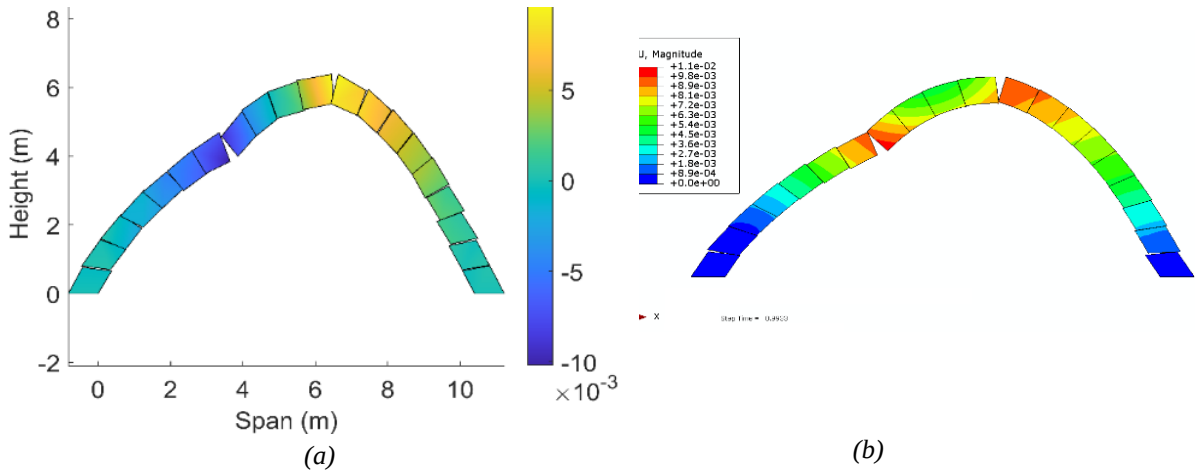


Figure 26: Deformation (m) of the arch 7 (10.4m span, 0.82m thickness) due to self-weight and a vertical load derived (a) from the trained artificial neural network (scale factor=80), (b) from the finite element analysis (scale factor= 50).

A last example is presented, providing the ultimate failure load and the collapse mechanism for the parabolic masonry arch 8 (table 4), using the proposed metamodel. Figure 27 shows the deformed geometry of arch 8 due to self-weight loading as well as due to self-weight and a vertical point load, as predicted by the neural networks A and B of table 3, respectively. It is noted that the classical four-hinge collapse mechanism can

be observed in this case too. The predicted ultimate load (20.7kN) is lower than the ultimate load derived for the parabolic arch 7.

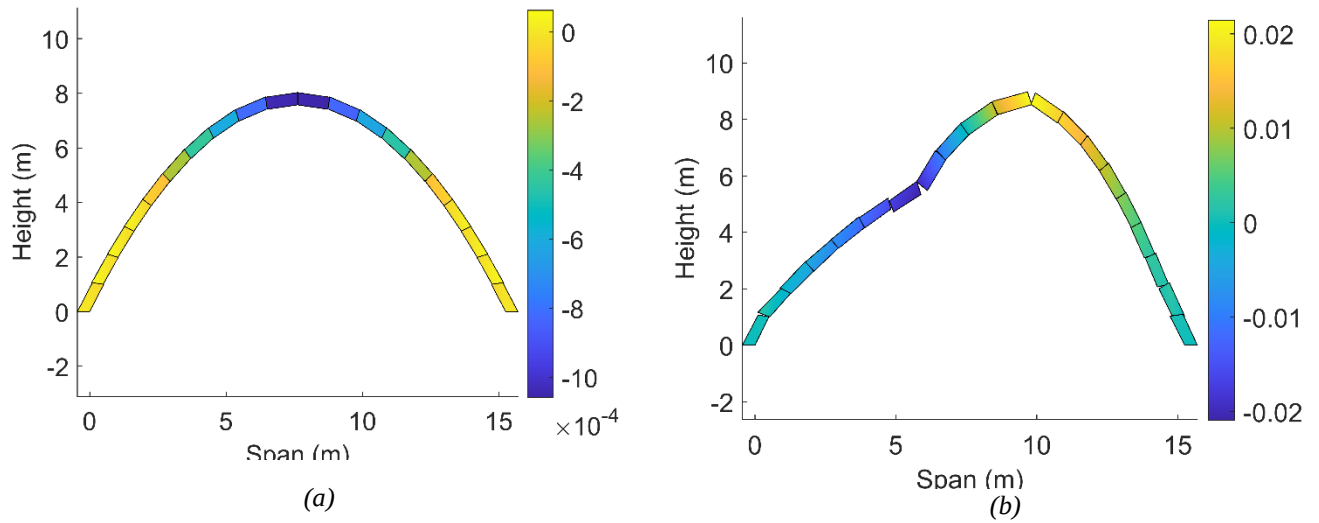


Figure 27: Deformation (m) of the arch 8 (15.25m span, 0.45m thickness) obtained from the trained artificial neural network (a) due to self-weight (scale factor=50), (b) due to self-weight and a vertical load (scale factor=50).

7.3 Summary of the results and datasets output

An effort to summarize the results provided in the datasets, reflecting holistically the structural response of masonry arches, is made in this section. In particular, it was observed that the structural response of a masonry arch varies with the span and masonry ring thickness. Therefore, both the span and the ring thickness values, which have been tested in the parametric simulations and included in the datasets for circular arches, are provided in figures 28 and 29. In both figures, unstable and stable masonry arch geometries are denoted. The unstable geometries correspond to arches which fail under their self-weight and thus, cannot support any vertical loading. Stable geometries are the ones which support their self-weight and potentially fail under the vertical loading.

In figure 28, masonry ring thickness versus span values are provided for unstable and stable geometries. It is shown that for higher spans, ring thicknesses significantly increase in order to provide a stable geometry. For example, for a span of 20m, ring thicknesses higher than 0.75m lead to stable arches.

In figure 29, thickness/span ratio versus the number of dataset points, called dataset node values in the graph, are provided for unstable and stable geometries. According to this graph, 400 dataset points from the parametric simulations (approximately) lead to stable masonry geometries, while more than 1300 dataset points lead to unstable geometries. In addition, for a masonry ring thickness to span ratio lower than 0.0383, as indicated by the average line in figure 29, unstable masonry arch geometries arise. For higher values of this ratio, depicting a dispersion of increased thicknesses (or reduced spans), stable masonry arches arise. It is noted that the datasets for stable and unstable masonry arch geometries, providing also the ultimate loads, accompany this article. Relevant descriptions can be found in Appendix 9.6.

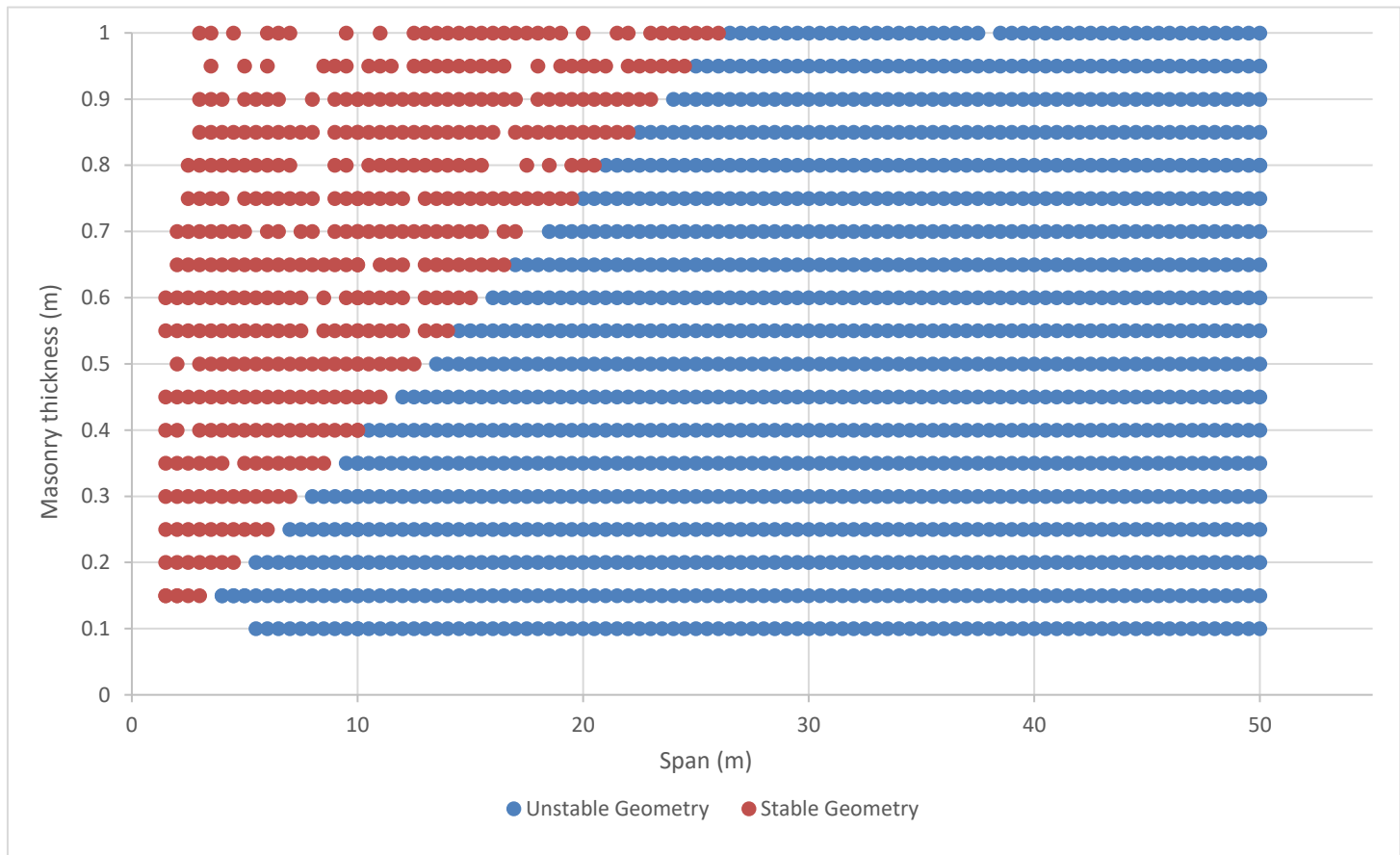


Figure 28: Plot of masonry thickness against span for stable and unstable *circular* masonry arch geometry.

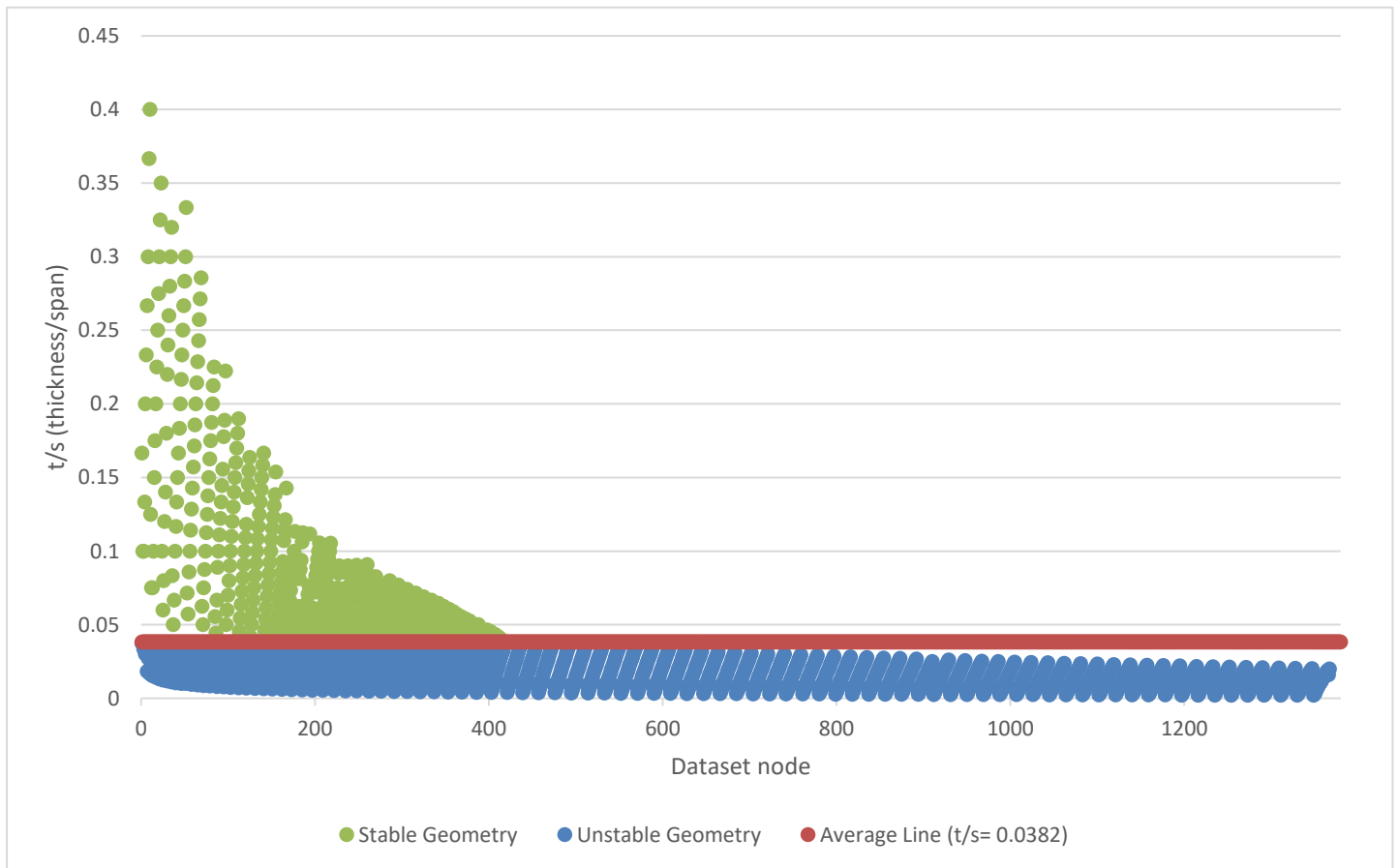


Figure 89: Plot of ratio of masonry thickness and span for stable and unstable *circular* masonry arch geometry.

In figure 30, masonry ring thickness versus span values are provided for unstable and stable parabolic arch geometries. In figure 31, thickness/span ratio versus the number of dataset points, called dataset node values in the graph, are provided for unstable and stable parabolic geometries. According to this graph, 500 dataset points from the parametric simulations (approximately) lead to stable masonry geometries, while more than 40 dataset points lead to unstable geometries. For a masonry ring thickness to span ratio lower than 0.01081, as indicated by the average line in figure 31, unstable masonry arch geometries arise. For higher values of this ratio, depicting a dispersion of increased thicknesses (or reduced spans), stable masonry arches are obtained.

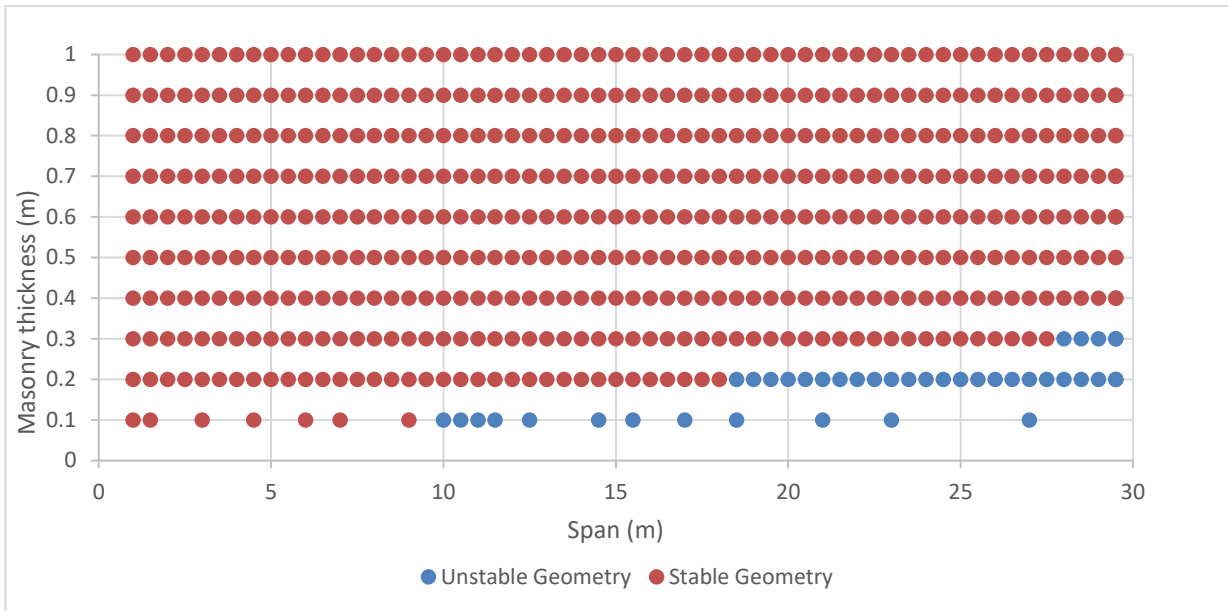


Figure 30: Plot of masonry thickness against span for stable and unstable parabolic masonry arch geometry.

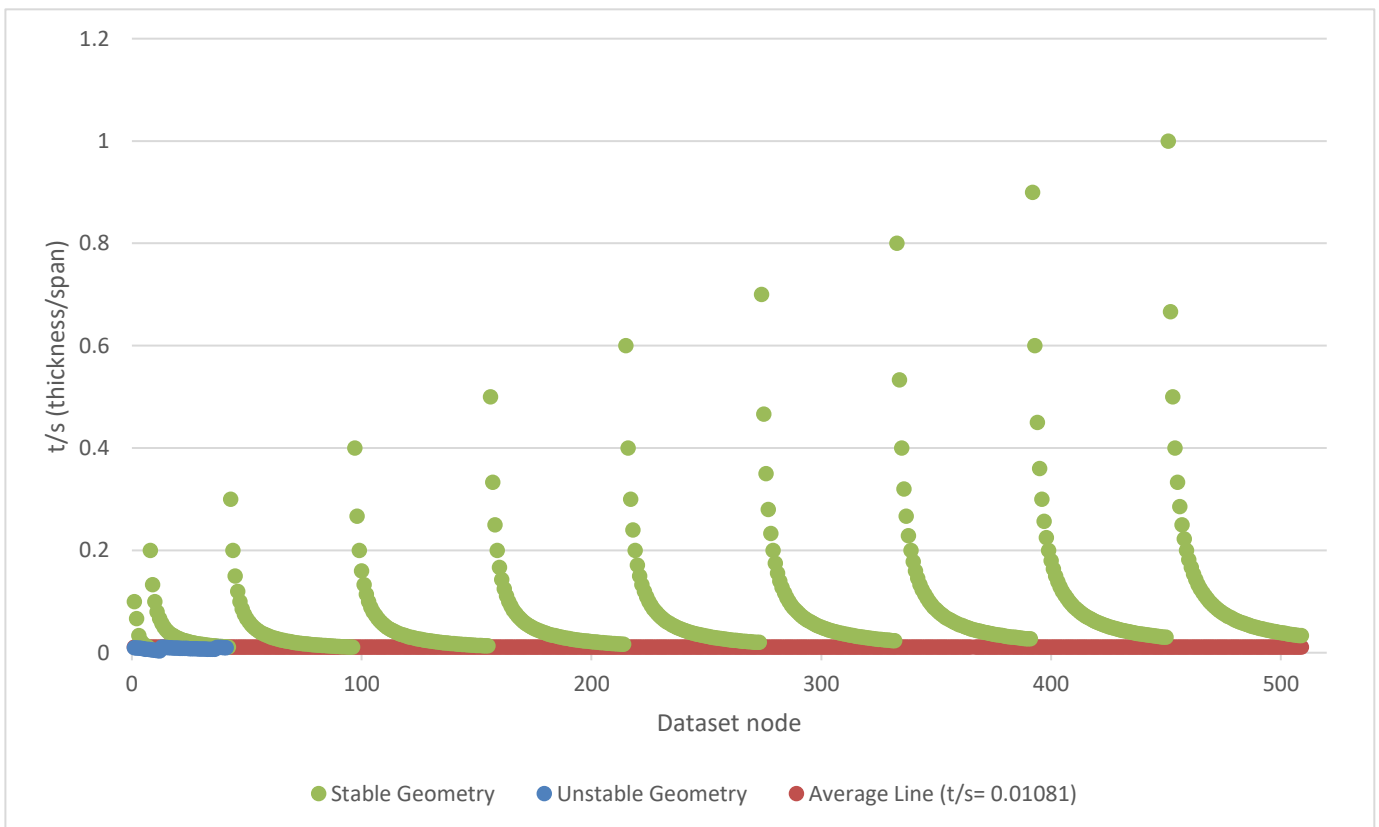


Figure 31: Plot of ratio of masonry thickness and span for stable and unstable parabolic masonry arch geometry.

Finally, comparisons of the displacements, ultimate loads and computational times, as obtained from the trained neural networks and finite element simulations, are provided. Table 5 shows the comparison of the ultimate loads and table 6 presents the comparison of the maximum displacement in the arch at an intermediate load level (self-weight) and at the ultimate state. It appears that results obtained from machine learning and finite element analysis are close.

Table 5: Comparison of ultimate loads obtained from the trained artificial neural networks (ANN) and finite element analysis (FEM).

Name	Ultimate load from FEM (kN)	Ultimate load predicted by ANN (kN)
Arch 5	34.10	35.9
Arch 6	10.40	8.80
Arch 7	99.33	96.05

Table 6: Comparison of displacements, intermediate at the end of self-weight and final displacements at the ultimate load, obtained from the trained artificial neural networks (ANN) and finite element analysis (FEM).

Name	Intermediate displacement at the end of self-weight (mm)		Displacement at the ultimate load (mm)	
	ANN	FEM	ANN	FEM
Arch 5	3	3.54	20	36.2
Arch 6	1	1.1	4	6
Arch 7	1.2	1.3	10	11

Data-driven analysis is also efficient in terms of the computation time compared to traditional finite element analysis, as given in table 7. Within traditional finite element analysis, it is estimated that an experienced user would need some hours to develop a model for one masonry arch like those investigated in this study. It is noted that setting up the model includes modelling the individual stones that make up the geometry of the arch (probably in CAD environment), assigning material properties to the stones, applying a surface-to-surface contact-law for each interface between adjacent stones, assigning boundary conditions, meshing the geometry and applying loads (gravity and point load). The same steps should be repeated for developing any other, randomly chosen arch geometry.

Concerning the proposed data-driven scheme, according to tables 2 and 3, some hours are needed to train the artificial neural networks. Some hours are also needed for the parametric finite element investigation of the different arch geometries, to create the datasets. However, this process takes place offline, and thus, it is implemented just once. When training of the neural networks is complete, the trained neural networks can be used as ready-to-use tools, in order to predict the response of random arches. According

to table 7, a trained neural network can make a prediction of the ultimate load and collapse mechanism in less than a minute.

Table 7: Comparison of the computation time which is needed to use an artificial neural networks (ANN) and run a finite element (FEM) simulation.

Name	Computational time for one FEM simulation (minutes)	Computational time for one ANN prediction (minutes)
Arch 5	11.3	0.4
Arch 6	2.1	0.4
Arch 7	0.5	0.4

8. Conclusions

A data-driven methodology, relying on machine learning and finite element analysis is proposed in this article, to investigate the structural behaviour of masonry arches. The structures are subjected to two loading steps, the self-weight and the self-weight plus a vertical point load applied at the quarter span. Parametric, non-linear finite element simulations were conducted to generate datasets providing the ultimate response. These datasets were then used to train artificial neural networks which stand as metamodels, providing the ultimate load and the collapse mechanism of random masonry arches.

Two-dimensional geometries of masonry arches were developed using a Matlab script, where the coordinates of each of the vertices of the masonry stones is extracted. The structural, finite element models were created using Python scripts called within Matlab to drive, a commercial finite-element software. The Python scripts provide the geometry of the structure by reading the extracted coordinates of the masonry stones vertices. The script also adds the mechanical boundary conditions, the subjected loads, and a unilateral law, used to simulate potential damage due to opening/sliding (contact-friction) between the masonry stones. Due to the nonlinearity of the models, the Newton–Raphson incremental–iterative process was used to solve the numerical problem. Python scripts within Matlab were also used to extract the results from the models. A total of 1862 dataset points for circular and 550 dataset points for parabolic arch shapes were used to train the neural networks. The training, validation and testing of the network neural networks were within acceptable tolerance.

The investigation shows that the proposed data-driven structural analysis of masonry arches can be used to provide accurate representation of the ultimate, failure response. The developed metamodel, can be used to predict the response of random masonry arches. The methodology can be extended to more complex three-dimensional geometries.

The article also proposes a numerical scheme to generate numerical datasets using Matlab and Python scripts as well as commercial finite element software. A complete set of relevant codes accompanies the article. Relevant descriptions can be found in the Appendix of the article.

The following conclusions can also be drawn:

- Machine learning can be a useful structural tool, in solving highly complex structural problems within a few seconds.

- The deformed geometries of the arches, which are predicted from the proposed process, are comparable with literature as well as with results obtained from finite element analysis.
- The developed structural tool can be used to investigate the structural behaviour of masonry arches without the need for extensive computational cost. Once the metamodel is built, predictions of the structural response can be provided in few seconds. Little or no structural knowledge is required since inputs of the metamodels are the span and the ring thickness of the structure. Thus, the proposed methodology can be extended and used for a first, fast and accurate representation of the ultimate response of similar structural systems.

Future work may involve the incorporation of image identification algorithms to the developed neural networks. With an image identification algorithm added, a photograph of a masonry arch can simply be supplied, read the geometry of the structure and feed it to the developed machine learning tool to predict the structural response. This would make the proposed methodology useful in structural health monitoring and site assessment for masonry arches. A system that quickly evaluates the remaining strength using these concepts could be helpful for the maintenance of these structures or during emergency situations after earthquakes or other disasters.

9. Appendices

In this section are provided descriptions of the source codes which have been developed to generate the dataset points for circular arch shapes. The interested reader can use, as well as extend the codes, for instance to generate more sophisticated (e.g. three-dimensional) geometries. All the source Matlab and Python files that have been used to create the parametric finite element simulations, as well as the datasets which have been used to train the artificial neural networks, accompany this article.

9.1 Central Matlab script

The central code in Matlab, which is used to create the parametric investigation of several geometries of masonry arches, is included in the Matlab script: *Appendix-1.m*.

Within this script, the commercial finite element software (Abaqus) is called, using a Python script (*Appendix-2.py*), to run a non-linear finite element simulation, for each arch geometry. The coordinates of the four vertices of each masonry block of each arch, are generated in *Appendix-1.m* script and saved in .txt files.

9.2 Python script implementing non-linear finite element analysis

The Python code, which is used to implement the non-linear finite element analysis of each parametric masonry arch geometry, is provided in the Python script: *Appendix-2.py*. Each parametric geometry, defined in *Appendix-1.m* script, is imported in *Appendix-2.py* script. In this script, all the steps of a finite element model can also be identified, including the material properties, the mesh, the loading and boundary conditions, as well as the unilateral contact-friction interfaces between the masonry blocks.

9.3 Matlab script generating results

The Matlab code which is used to generate results, is provided in the Matlab script: *Appendix-3.m*. Within this script, the commercial finite element software (Abaqus) is called via a Python script (*Appendix-4.py*), to provide the solution of the finite element analysis. *Appendix-3.m* script also runs a built-in code which reads the ultimate load obtained from the output files of the finite element simulations, generated in the previous steps.

9.4 Python script generating results

The Python code, which is used to extract the results from the finite element simulations, is included in the Python script: *Appendix-4.py*. The extracted results are the displacements at the four vertices of each masonry block.

9.5 Matlab script generating the deformed geometry of each masonry arch

The Matlab code, which is used to generate and visualize the deformed geometry of each masonry arch, is included in the Matlab script: *Appendix-5.m*. Inputs to generate one geometry, are the initial coordinates of the four vertices of each masonry block, as well as the displacements of the vertices of the masonry blocks at the end of each finite element simulation.

9.6 Datasets

The generated datasets, which have been used to train the artificial neural networks are also attached to this article in the form of an Excel spreadsheet. In particular, the datasets corresponding to circular arch shapes are included in *DataSet.xlsx* spreadsheet, while those corresponding to parabolic arch shapes are given in *DataSetParabolic.xlsx*. Within the datasets, stable and unstable geometries are identified.

At each column of the Excel spreadsheets, the following dataset points, derived from each parametric simulation, are provided: span of the masonry arch, thickness, ultimate load, thickness/span ratio, deformed geometry of the arch due to self-weight, deformed of the arch due to point load (for the stable geometries).

Acknowledgments

Siphesihle Mpho Motsa has been supported by **Erasmus+ Program** within the framework of action “**International Credit Mobility**” between the Technical University of Crete, School of Production Engineering and Management and the University of KwaZulu-Natal, department of Civil engineering under Structural Engineering & Computational Mechanics (SECM) Group.

References

- Anastasio, S. 2020. Building between the two rivers: an introduction to the building archaeology of ancient Mesopotamia. *Building between the Two Rivers*, 1-220.
- Ashrafian, A., Panahi, E., Salehi, S., Karoglou, M., Asteris, P.G. 2023. Mapping the strength of agro-ecological lightweight concrete containing oil palm by-product using artificial intelligence techniques. *Structures*, 48, 1209-1229.
- Bekas, G.K. & Stavroulakis, G.E. 2017. Machine learning and optimality in multi storey reinforced concrete frames. *Infrastructures*, 2, 6.
- Bergamo, O., Campione, G., Donadello, S., & Russo, G. 2015. In-situ NDT testing procedure as an integral part of failure analysis of historical masonry arch bridges. *Engineering Failure Analysis*, 57, 31-55.
- Beskopylny, A., Lyapin, A., Anysz, H., Meskhi, B., Veremeenko, A. & Mozgovoy, A. 2020. Artificial Neural Networks in Classification of Steel Grades Based on Non-Destructive Tests. *Materials (Basel)*, 13.
- Block, P., DeJong, M. & Ochsendorf, J. 2006. As Hangs the Flexible Line: Equilibrium of Masonry Arches. *Nexus Network Journal*, 8, 13-24.
- Broomhead, D. S. & Lowe, D. 1988. Radial basis functions, multi-variable functional interpolation and adaptive networks. Royal Signals and Radar Establishment Malvern (United Kingdom).
- Cascini, L., Gagliardo, R., & Portioli, F. 2018. LiABlock_3D: A Software tool for collapse mechanism analysis of historic masonry structures, *Int. J. Archit. Herit.* 14,1, 75-94.
- Cavalagli, N., Gusella, V., & Severini, L. 2016. Lateral loads carrying capacity and minimum thickness of circular and pointed masonry arches, *Int. J. Mech. Sci.* 115, 645-656.
- Cavaleri, L., Barkhordari, M.S., Repapis, C.C., Armaghani, D.J., Ulrikh, D.V., & Asteris, P.G. 2022. Convolution-based ensemble learning algorithms to estimate the bond strength of the corroded reinforced concrete. *Construction and Building Materials*, 359, 129504.
- Chang, W. & Zheng, W. 2019. Estimation of compressive strength of stirrup-confined circular columns using artificial neural networks. *Structural Concrete*, 20, 1328-1339.
- Charalambidi, B., Koutsianitis, P., Motsa, S. M., Tairidis, G., Kasampali, A., Drosopoulos, G., Stavroulaki, M. & Stavroulakis, G. 2022. Modelling, identification and structural damage investigation of the Neoria monument in Chania. *Developments in the Built Environment*, 10, 100069.
- Civera, M., Mugnaini, V. & Zanotti Fragonara, L. 2022. Machine learning-based automatic operational modal analysis: A structural health monitoring application to masonry arch bridges. *Structural Control and Health Monitoring*, 29, e3028.
- Cocchetti, G., Colasante, G. & Rizzi, E. 2012. On the Analysis of Minimum Thickness in Circular Masonry Arches. *Applied Mechanics Reviews*, 64.
- Conde, B., Drosopoulos, G., Stavroulakis, G., Riveiro, B. & Stavroulaki, M. 2016. Inverse analysis of masonry arch bridges for damaged condition investigation: Application on Kakodiki bridge. *Engineering Structures*, 127, 388-401.

988 Couplet, P. 1729. De la poussée des voûtes. *Histoire de l'Académie royale des sciences*,
 989 79, 117-141.
 990 Dais, D., Bal, I. E., Smyrou, E., & Sarhosis, V. 2021. Automatic crack classification and
 991 segmentation on masonry surfaces using convolutional neural networks and
 992 transfer learning. *Automation in Construction*, 125, 103606.
 993 Drosopoulos, G., Stavroulakis, G. & Massalas, C. 2006. Limit analysis of a single span
 994 masonry bridge with unilateral frictional contact interfaces. *Engineering*
 995 *Structures*, 28, 1864-1873.
 996 Drosopoulos, G. A. & Stavroulakis, G. E. 2018. A computational homogenization
 997 approach for the study of localization of masonry structures using the XFEM.
 998 *Archive of Applied Mechanics*, 88, 2135-2152.
 999 Drosopoulos, G. A. & Stavroulakis, G. E. 2020. Data-driven computational
 1000 homogenization using Neural Networks, *Journal on Computing and Cultural*
 1001 *Heritage*, 14, 1-19.
 1002 Elman, J. L. 1990. Finding structure in time. *Cognitive Science*, 14, 179-211.
 1003 Ferrero, C., Calderini, C. & Roca, P. 2023. Effect of joint deformability on the
 1004 experimental and numerical response of dry-joint masonry arches subjected to
 1005 large support displacements. *Engineering Structures*, 275, 115236.
 1006 Foce, F. & Huerta, S. 2005. On the safety of the masonry arch. Different formulations
 1007 from the history of structural mechanics. *Essays in the History of Theory of*
 1008 *Structures*, 117-142.
 1009 Galassi, S. 2023. An alternative approach for limit analysis of masonry arches on moving
 1010 supports in finite small displacements, *Eng. Fail. Anal.*, 145, 107004.
 1011 Galassi, S., Zampieri, P. 2023. A new automatic procedure for nonlinear analysis of
 1012 masonry arches subjected to large support movements, *Engineering Structures*,
 1013 276, 115359.
 1014 Gáspár, O., Sajtos, I. & Sipos, A. A. 2022. Multi-Hinge Failure Mechanisms of Masonry
 1015 Arches Subject to Self-Weight as Derived from Minimum Thickness Analysis.
 1016 *International Journal of Architectural Heritage*, 1-29.
 1017 Grandio, J., Riveiro, B., Soilán, M., Arias, P. 2022. Point cloud semantic segmentation of
 1018 complex railway environments using deep learning. *Automation in Construction*,
 1019 141, 104425.
 1020 Grillanda, N., Milani, G., Ghosh, S., Halani, B. & Varma, M. 2021. SHM of a severely
 1021 cracked masonry arch bridge in India: Experimental campaign and adaptive
 1022 NURBS limit analysis numerical investigation. *Construction and Building*
 1023 *Materials*, 280, 122490.
 1024 Hagan, M. T., Demuth, H. B. & Beale, M. 1997. *Neural network design*, PWS Publishing
 1025 Co.
 1026 Hagan, M. T. & Menhaj, M. B. 1994. Training feedforward networks with the Marquardt
 1027 algorithm. *IEEE transactions on Neural Networks*, 5, 989-993.
 1028 Haykin, S. 2009. *Neural networks and learning machines*, 3/E, Pearson Education India.
 1029 Heyman, J. 1966. The stone skeleton. *International Journal of solids and structures*, 2,
 1030 249-279.
 1031 Heyman, J. 1967. On shell solutions for masonry domes. *International Journal of Solids*
 1032 *and Structures*, 3, 227-241.
 1033 Heyman, J. 1969. The safety of masonry arches, *Int. J. Mech. Sci.*, 11,4, 363-85.

1034 Heyman, J. 1982. *The Masonry Arch*, Chichester, UK: Ellis Horwood.
 1035 Heyman, J. 1995. The stone skeleton: structural engineering of masonry structures.
 1036 Cambridge University Press, Cambridge, UK.
 1037 Heyman, J. 1998. *Structural analysis: a historical approach*, Cambridge University
 1038 Press.
 1039 Hooke, R. 1676. *A description of helioscopes, and some other instruments*, London,
 1040 printed by T.R. for John Martyn.
 1041 Ivakhnenko, A. G. 1971. Polynomial theory of complex systems. *IEEE transactions on*
 1042 *Systems, Man, and Cybernetics*, 364-378.
 1043 Jang, J.-S. 1993. ANFIS: adaptive-network-based fuzzy inference system. *IEEE*
 1044 *transactions on systems, man, and cybernetics*, 23, 665-685.
 1045 Jing, Y., Sheil, B. & Acikgoz, S. 2022. Segmentation of large-scale masonry arch bridge
 1046 point clouds with a synthetic simulator and the BridgeNet neural network.
 1047 *Automation in Construction*, 142, 104459.
 1048 LeCun, Y., Bottou, L., Bengio, Y. & Haffner, P. 1998. Gradient-based learning applied to
 1049 document recognition. *Proceedings of the IEEE*, 86, 2278-2324.
 1050 Liu, D., Tang, Z., Bao, Y. & Li, H. 2021. Machine-learning-based methods for output-
 1051 only structural modal identification. *Structural Control and Health Monitoring*,
 1052 28, e2843.
 1053 Lourenço, P. B. 2002. Computations on historic masonry structures. *Progress in*
 1054 *Structural Engineering and Materials*, 4, 301-319.
 1055 Loverdos, D., Sarhosis, V. 2022. Automatic image-based brick segmentation and crack
 1056 detection of masonry walls using machine learning. *Automation in Construction*,
 1057 140, 104389.
 1058 Loverdos, D., Sarhosis, V. 2023. Geometrical digital twins of masonry structures for
 1059 documentation and structural assessment using machine learning. *Engineering*
 1060 *Structures*, 275, Part A, 115256.
 1061 Marquardt, D. W. 1963. An algorithm for least-squares estimation of nonlinear
 1062 parameters. *Journal of the society for Industrial and Applied Mathematics*, 11,
 1063 431-441.
 1064 Matlab 2021. release R2021a. Natick, Massachusetts: The MathWorks Inc.
 1065 McLean, T., Málaga-Chuquitaype, C., Kalapodis, N. & Kampas, G. 2021. OpenArch: An
 1066 open-source package for determining the minimum-thickness of arches under
 1067 seismic loads. *SoftwareX*, 15, 100731.
 1068 Melbourne, C. & Gilbert, M. 1995. The behaviour of multiring brickwork arch bridges.
 1069 *Structural Engineer*, 73.
 1070 Melchiorre, J., Manuello, A. & Marano, G. 2021. Application of a Machine Learning
 1071 Algorithm for the Structural Optimization of Circular Arches with Different
 1072 Cross-Sections. *Journal of Applied Mathematics and Physics*, 09, 1159-1170.
 1073 Milani, G., Lourenco, P. & Tralli, A. 2006. Homogenization Approach for the Limit
 1074 Analysis of Out-of-Plane Loaded Masonry Walls. *Journal of Structural*
 1075 *Engineering-asce - J STRUCT ENG-ASCE*, 132.
 1076 Milani, G., Lourenço, P.B. 2012. 3D non-linear behavior of masonry arch bridges,
 1077 *Comput. Struct.* 110-111, 133-150.
 1078 Milankovitch, M. 1904. *Beitrag zur Theorie der Druckkurven*. Dissertation zur Erlangung
 1079 der Doktorwürde, KK Technische Hochschule Vienna

1080 Milankovitch, M. 1907. *Theorie der druckkurven*, éditeur inconnu.

1081 Moseley, H. 1833. On a new principle in statics called the principle of least pressure, The

1082 London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science, 3,16,

1083 285-88.

1084 Mostafa, K., Zisis, I. & Moustafa, M. A. 2022. Machine Learning Techniques in

1085 Structural Wind Engineering: A State-of-the-Art Review. *Applied Sciences*, 12,

1086 5232.

1087 Nasrabadi, N. M. 2007. Book Review: Pattern Recognition and Machine Learning. SPIE.

1088 Ochsendorf, J. 2006. The masonry arch on spreading supports, *Struct. Eng.* 84,2, 29-34.

1089 O'Dwyer, D. 1999. Funicular analysis of masonry vaults. *Computers & Structures*, 73,

1090 187-197.

1091 Özmen, A. & Sayın, E. 2018. *Linear dynamic analysis of a masonry arch bridge*.

1092 Panagiotopoulos, P. D. 1985. *Inequality Problems in Mechanics and Applications:*

1093 *Convex and nonconvex energy functions*, Springer Science & Business Media.

1094 Poleni, G. 1748. *Memorie storiche della Gran Cupola del Tempio Vaticano*, Padua:

1095 Nella Stamperia del seminario.

1096 Portioli, F., Cascini, L. 2017. Large displacement analysis of dry-jointed masonry

1097 structures subjected to settlements using rigid block modelling, *Eng. Struct.* 148,

1098 485-496.

1099 Prakash, M., Manikandan, S., Surether, I., Aswin Kumar, M., Ilakkiya, S. & Menaka, D.

1100 2019. Speculation of compressive strength of concrete in real-time. *International*

1101 *Journal of Recent Technology and Engineering*, 7, 988-992.

1102 Psychas, I.D., Schauer, M., Böhrnsen, J.U., Marinaki, M., Marinakis, Y., Langer, S.C.,

1103 Stavroulakis G.E. 2016. Detection of defective pile geometries using a coupled

1104 FEM/SBFEM approach and an ant colony classification algorithm. *Acta*

1105 *Mechanica* 227, 1279–1291.

1106 Rahimi, A., Aval, S. B. B., Noori, M., Sarhosis, V., Wu, Z., Nikkhoo, A. & Altabey, W.

1107 A. 2022. A simplified beam model for the numerical analysis of masonry arch

1108 bridges –A case study of the Veresk railway bridge. *Structures*, 45, 1253-1266.

1109 Reich, Y. 1997. Machine learning techniques for civil engineering problems. *Computer-*

1110 *Aided Civil and Infrastructure Engineering*, 12, 295-310.

1111 Rosenblatt, F. 1958. The perceptron: a probabilistic model for information storage and

1112 organization in the brain. *Psychological review*, 65, 386.

1113 Sadowski, Ł., Nikoo, M. & Nikoo, M. 2018. Concrete compressive strength prediction

1114 using the imperialist competitive algorithm. *Computers and Concrete, An*

1115 *International Journal*, 22, 355-363.

1116 Sánchez-Aparicio, L.J., Bautista-De Castro, Á., Conde, B., Carrasco, P., & Ramos, L.F.

1117 2019. Non-destructive means and methods for structural diagnosis of masonry

1118 arch bridges. *Automation in Construction*, 104, 360-382.

1119 Sarhosis, V., Bagi, K., Lemos, J. V. & Milani, G. Computational Modeling of Masonry

1120 Structures Using the Discrete Element Method. 2016.

1121 Simon, H. 1999. *Neural networks: a comprehensive foundation*, Prentice hall.

1122 Simulia, D. S. 2013. ABAQUS 6.13 User's manual. *Dassault Systems, Providence, RI*,

1123 305, 306.

1124 Stavroulaki, M. E., Drosopoulos, G. A., Tavlopoulou, E., Skoutelis, N. & Stavroulakis,

1125 G. E. 2018. Investigation of the structural behaviour of a masonry castle by

1126 considering the actual damage. *International Journal of Masonry Research and*
1127 *Innovation*, 3, 1-33.

1128 Stockdale, G., Yuan, Y. & Milani, G. 2022. The behavior mapping of masonry arches
1129 subjected to lumped deformations. *Construction and Building Materials*, 319,
1130 126069.

1131 Tapkın, S., Tercan, E., Motsa, S. M., Drosopoulos, G., Stavroulaki, M., Maravelakis, E.
1132 & Stavroulakis, G. 2022. Structural Investigation of Masonry Arch Bridges Using
1133 Various Nonlinear Finite-Element Models. *Journal of Bridge Engineering*, 27.

1134 Tempesta, G., Galassi, S. 2019. Safety evaluation of masonry arches. A numerical
1135 procedure based on the thrust line closest to the geometrical axis, *Int. J. Mech.*
1136 *Sci.*, 155, 206-21.

1137 Thai, H.-T. 2022. Machine learning for structural engineering: A state-of-the-art review.
1138 *Structures*, 38, 448-491.

1139 Tubaldi, E., Minga, E., Macorini, L. & Izzuddin, B. A. 2020. Mesoscale analysis of
1140 multi-span masonry arch bridges. *Engineering Structures*, 225, 111137.

1141 Winkler, E. 1867. *Die Lehre von der Elasticitat und Festigkeit*, Dominicus, Prague.

1142 Wołowiec, E. & Kula, P. Practical Application of Artificial Neural Networks in
1143 Designing Parameters of Steel Heat Treatment Processes. In: RUTKOWSKI, L.,
1144 KORYTKOWSKI, M., SCHERER, R., TADEUSIEWICZ, R., ZADEH, L. A. &
1145 ZURADA, J. M., eds. *Artificial Intelligence and Soft Computing*, 2012// 2012
1146 Berlin, Heidelberg. Springer Berlin Heidelberg, 196-203.

1147 Yuan, Y., Stockdale, G. & Milani, G. A novel fast and low-cost masonry monitoring
1148 strategy for masonry arches. 2022 IEEE International Workshop on Metrology
1149 for Living Environment (MetroLivEn), 2022. IEEE, 149-153.

1150 Zampieri, P., Tetougueni, C. D. & Pellegrino, C. 2021. Nonlinear seismic analysis of
1151 masonry bridges under multiple geometric and material considerations:
1152 Application to an existing seven-span arch bridge. *Structures*, 34, 78-94.