Parameterised non-intrusive reduced-order model for general unsteady flow problems using artificial neural networks

Oliviu Şugar-Gabor

Aeronautical Engineering, Engineering Directorate, University of Salford

Newton Building, 41-42 The Crescent, M5 4WT, Salford, United Kingdom

o.sugar-gabor@salford.ac.uk

Abstract

A non-intrusive reduced-order model for nonlinear parametric flow problems is developed. It is based on extracting a reduced-order basis from high-order snapshots via proper orthogonal decomposition and using multi-layered feedforward artificial neural networks to approximate the reduced-order coefficients. The model is a generic and efficient approach for the reduction of time-dependent parametric systems, including those described by partial differential equations. Since it is non-intrusive, it is independent of the high-order computational method and can be used together with black-box solvers. Numerical studies are presented for steady-state isentropic nozzle flow with geometric parameterisation and unsteady parameterised viscous Burgers equation. An adaptive sampling strategy is proposed to increase the quality of the neural network approximation while minimising the required number of parameter samples and, as a direct consequence, the number of high-order snapshots and the size of the network training set. Results confirm the accuracy of the non-intrusive approach as well as the speed-up achieved compared with intrusive hyper reduced-order approaches.

Keywords

Non-intrusive parameterised reduced-order model; proper orthogonal decomposition; artificial neural networks; viscous Burgers equation.

1. Introduction

Large-scale, high-fidelity numerical simulations are commonly used across a wide array of industrial applications. Increasingly, parametric studies and design optimisation studies are also being done using high-fidelity models, even if they are costly. For instance, aerodynamic shape optimisation based on the Reynolds-Averaged Navier-Stokes (RANS) equations is now common practice. However, some problems such as the accurate design of high-lift systems require the usage of more advanced turbulence modelling such as hybrid RANS-LES (Large Eddy Simulation) approaches. These models have an increased computational cost which makes them unfeasible for large-scale parametric studies. Surrogate models aimed at finding inexpensive to run approximations of a system's input-output relation can partly overcome this challenge [1]. Among them, reduced order models (ROMs) occupy a central place by generating a low-order representation of the system defined on a basis which optimally spans the space of the system's variables. ROMs have been successfully applied to a variety of problems requiring large-scale numerical simulations, including weather prediction, modelling of particle physics, data assimilation, flow through porous media, molecular dynamics or the study of advection-diffusion-reaction phenomena [5-10].

A popular technique for determining such a basis is Proper Orthogonal Decomposition (POD), originally introduced over a century ago [3]. POD uses a set of system outputs (commonly referred to as snapshots) to provide a set of orthonormal vectors which form the required basis. As for most systems the snapshots will be correlated, POD generally requires only a relatively small number of basis vectors to capture the significant information contained in the snapshots. The solution coordinates in the POD basis are usually known as the POD coefficients. Popular techniques to determine the coefficients are Petrov-Galerkin projection, least-squares residual (or another error) minimisation or data-fitting techniques [2].

In the Petrov-Galerkin projection approach, the original equations describing the system are projected onto the basis in order to determine the reduced-order system, which can be solved for the coefficients. However, this approach is code intrusive (requires extensive source code

modifications) and cannot be used with black-box solvers Intrusive ROM approaches can suffer from inefficiencies dealing with nonlinear problems and instability issues [11-13] remedies for these issues also being proposed in literature [14-17]. An alternative non-intrusive approach is to project only the snapshots onto the basis and determine a finite set of POD coefficients corresponding to those snapshots, and then use other surrogate techniques such as Kriging, artificial neural networks (ANNs) or radial basis functions (RBFs) as closures to model the full POD coefficient space based on the available set [18,19]. Regardless of whether the ROM is intrusive or non-intrusive, most approaches use an offline-online strategy. The offline stage refers to collecting the high-order model snapshots and generating the basis (plus any other fixed quantities involved in that specific ROM approach), while the online stage refers to using the ROM to get approximations for the system solution for varying situations of interest.

Non-intrusive ROMs using ANNs have been introduced only relatively recently [24,25], and the breadth of literature on the topic is increasing at a high rate. ANN-ROMs have been applied to modelling the steady incompressible Navier-Stokes equations [26] and the unsteady viscous Burgers equation [27,28] and show very good accuracy and considerable speed-up compared to intrusive ROMs in the online stage. A careful setup of the ANN such as in [26] can reduce the offline training time to values comparable to those required for the offline generation of some so-called hyper ROM approaches [14].

A recurrent neural network has been introduced and shown to be an efficient model order reduction technique for nonlinear dynamical systems [32]. Other neural network architectures, such as long short-term memory (LSTM) architectures have been utilised to complement an imperfect reduced-order model with live data streams [33]. Deep ANNs have also been used in conjunction with POD-based reduced order modelling for unsteady parametric fluid flow models but restricted to the Reynolds number as the only problem parameter, with a very limited number of samples [34]. This allowed for a single ANN to learn both the dynamic and the parameter space behaviour of the system. The accuracy achieved was shown to be

comparable with intrusive ROMs based on POD and a Galerkin projection. A very recent application of ANNs is presented in [35], where a parametric reduced-order model is generated for the steady-state inviscid transonic flow over a RAE 2822 aerofoil. The parametric space included eight independent parameters for modifying the aerofoil shape, the results showing that the neural network based model achieved accuracy comparable with intrusive ROM techniques. It was also highlighted that a reasonably good training of the ANN can be achieved even when the number of training samples is much lower compared to the typically-used training datasets numbering thousands of samples. A non-intrusive ROM strategy based on POD, physics-informed neural networks (PINNs) and physics-reinforced neural networks (PRNNs) was proposed in [39]. In training the networks, the residual error of the reduced-order equations is used alongside the error between the network output and the POD reduction. Results shown better ROM accuracy compared to other ANN-based approaches for parametric, time-dependent test problems, but the method was tested only for a low number of parameters as a single ANN is used to capture both the unsteady and the parameter space behaviour. Another ANN-ROM approach was developed in [43], formulated as a maximum-likelihood problem in which, out of a class on candidate architectures, the one which minimises the error on the available high-order solutions is selected. Very good accuracy and model reduction capabilities were achieved for two large-scale models, but without any parameterisation.

It is seen that the focus in most of these works has been either to capture the nonlinear parameter space variation of a steady parameter-dependent system, or to accurately capture the unsteady behaviour of a nonlinear system. Indeed, the most common usage for ANNs in ROM frameworks are related to time evolution or data compression, examples of the latter being found in [45] and [46], fewer works (such as [35]) focusing on the parametric space approximation. Work on ANN-ROMs for time-dependent phenomena involving arbitrarily large parameter spaces, as are common for many engineering design problems, is rarely covered in literature. It must be noted that approaches other than ANN-ROMs have been proposed to

tackle this challenging problem, with various degrees of success and flexibility, including POD-based models in which the ROM is complemented by an auxiliary linear PDE for enforcing the initial and boundary conditions [20,23], POD coupled with Taylor series expansions [21] and the widely-used POD-RBF models [19].

In recent years, researchers have shown increasing interest in using machine learning (ML) techniques not only to improve ROM prediction accuracy, but even to entirely bypass traditional numerical methods. A good review of the use of ML in model order reduction, together with representative examples from aerodynamics or structural mechanics can be found in [41]. In [36], a non-autoregressive time series approach is proposed and shown to have improved long-term forecasting capabilities for dynamical systems compared to LSTM network architectures. The evolution of time-dependent systems has also been captured using Gaussian Processes (GPs) to capture the reduced-space dynamics, together with convolutional encoders and decoders to map the high-order data into a low-order representation and back [37]. GPs have also been used in conjunction with POD for replacing a costly LES computation of airflow in an urban environment [40] and to model turbulent flow around transonic aerofoil section [44], while convolutional autoencoders have been proposed as alternatives for building suitable reduced subspaces for general dynamical problems [42]. An error-modelling technique for ROMs was developed in [38], aimed at correcting ROM outputs independent of the ROM generation strategy. Regression-based techniques are trained based on the error between the full and reduced-order models, with several approaches such as random-forest (RF) regression and least absolute shrinkage and selection operator (LASSO) regression being investigated.

To the authors best knowledge, the work presented here represents the first utilization of an ANN as a closure model in the POD-ROM modelling of parametric and time-dependent nonlinear phenomena in which the size and complexity of the parametric space can be arbitrarily high. The novelty and flexibility of the proposed approach consists of an efficient decoupling of the time-domain dynamics approximation from the parameter space

approximation. The most adequate ANN architecture can thus be selected for each approximation, based on the available number of POD modes, parameter samples, computational time constraints, etc. The paper is structured as follows: Section 2 provides a brief overview of ANNs; section 3 investigates the performance of ANN-ROM for parametric time-independent problems and presents an adaptive sampling method for improving the approximation; section 4 investigates the application to time-dependent non-parametric nonlinear problems; finally, section 5 explores parametric and time-dependent nonlinear problems and presents a novel extension to the adaptive sampling strategy to efficiently tackle problems with a high-dimensional parametric space while minimising the required number of sample points.

2. Brief overview of Artificial Neural Networks

An artificial neural network, often referred to as a neural network, is a computational model able to learn from a provided data set (for a thorough overview of ANNs see for example [29]). The architecture of ANNs was inspired by their biological counterparts and is represented by a collection of neurons (as fundamental data processing elements) and weighted, directed connections between neurons (the equivalent of synapses).

Assume an arbitrary neuron j in the network receives M input signals $\{x_1, x_2, \dots, x_M\}$ (either from M sending neurons or as the network input data) and produces a scalar output signal y_j (which can in turn be sent to N receiving neurons or represent the network output data).

The propagation function of neuron j converts the vector input into a scalar input, often referred to as the net input. The most common choice is a weighted sum taking the form:

$$u_j = f_{propagation}(x_1, x_2, \dots, x_M) = \sum_{k=1}^{M} w_{k,j} x_K \tag{1}$$

Here, u_j is the net input of neuron j and $w_{k,j}$ are the set of M weights for each of the neuron's vector input components. The activation function quantifies to what extent a given neuron is

active (generating an output value). The activation function combines the net input with some threshold $t_j \in \mathbb{R}$ and is responsible for creating the activation state of the neuron:

$$a_j = f_{activation}(u_j, t_j) = f_{activation}\left(\sum_{k=1}^{M} w_{k,j} x_K , t_j\right) \quad (2)$$

It is common practice to introduce a so-called bias neuron b into the network, a continuously active neuron having a constant output $y_b = 1$ which is connected with the arbitrary neuron j and whose output weight towards neuron j (called the bias weight $w_{b,j}$) is used as the threshold value $w_{b,j} = -t_j$. Using a bias neuron, the activation state becomes:

$$a_j = f_{activation}\left(\sum_{k=1}^{M} w_{k,j} x_K - t_j\right) \quad (3)$$

Various choices of activation function exist in literature, among which the most common are so-called sigmoid functions [29], the hyperbolic tangent being a widely used example:

$$a_j = \frac{e^{\sum_{k=1}^{M} w_{k,j} x_K - t_j} - e^{-\sum_{k=1}^{M} w_{k,j} x_K - t_j}}{e^{\sum_{k=1}^{M} w_{k,j} x_K - t_j} + e^{-\sum_{k=1}^{M} w_{k,j} x_K - t_j}} \quad (4)$$

The output function determines the neuron's scalar output y_j based on the activation state but is often taken as the identity function so that the output is directly given by a_j.

$$y_j = f_{output}(a_j) \quad (5)$$

The connection of individual neurons defines the topology of the ANN. Different topologies have been proposed for different applications, however for function approximation and regression the feedforward topology is preferred, first proposed in [30]. Neurons are arranged in layers, with one input layer, K hidden layers and one output layer. The neurons in the input layer to not perform any computation tasks, having an identity activation function, while the output layer neurons typically use simpler, linear activation functions. Neurons in any layer receive data only from the previous layer (towards the input) and output data only towards the next layer (towards the output).

The training of the neural network is the iterative process through which the neuron weights are determined. This is achieved by learning from a provided training set. Let $f: \mathbb{R}^I \rightarrow \mathbb{R}^O$ be a nonlinear function which the neural network must approximate. Let $\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_{NTR}$, with $\boldsymbol{x}_i \in \mathbb{R}^I$ be a set of NTR training points, and $\boldsymbol{y}_1, \boldsymbol{y}_2, \dots, \boldsymbol{y}_{NTR}$, with $\boldsymbol{y}_i = f(\boldsymbol{x}_i) \in \mathbb{R}^I$ be the function values corresponding to these points. The set $\{\boldsymbol{x}_i, \boldsymbol{y}_i\}, i = 1,2, \dots, NTR$ represents the training set which must be supplied to the neural network. The goal is to approximate f up to a certain tolerance ε. The performance of the network is determined based on a performance function, typically the mean squared error (MSE):

$$Performance = \sum_{i=1}^{NTR} (\|\boldsymbol{y}_i - F(\boldsymbol{x}_i)\|)^2 \leq \varepsilon \tag{6}$$

Where $F(\boldsymbol{x}_i)$ is the neural network prediction at training point $\boldsymbol{x}_i$. The training iterations, usually called epochs, are aimed at calculating the optimal neuron weights such that (6) is satisfied and can be conducted using nonlinear regression algorithms such as the Levenberg-Marquardt algorithm [31]. It must be noted that the Levenberg-Marquardt algorithm is the most efficient choice for relatively small networks, but not for larger ones due to its computational cost increasing nonlinearly with the number of hidden neurons.

3. Non-intrusive POD-based ROM for Parametric Time-Independent Systems

Let the high-order time-independent parametric model be represented by:

$$\boldsymbol{R}(\boldsymbol{u}(\boldsymbol{x}, \boldsymbol{\mu}), \boldsymbol{x}, \boldsymbol{\mu}) = \boldsymbol{0} \tag{7}$$

Here, $\boldsymbol{R}: \mathbb{R}^N \times \mathbb{R}^P \rightarrow \mathbb{R}^N$ with N typically being very large, $\boldsymbol{u}: \Omega \subset \mathbb{R}^N \times \mathbb{R}^P \rightarrow \mathbb{R}^N$ are the system variables defined on a subspace Ω of $\mathbb{R}^N$, $\boldsymbol{x} \in \mathbb{R}^N$ are spatial coordinates, and $\boldsymbol{\mu}: \mathrm{D} \subset \mathbb{R}^P \rightarrow \mathbb{R}^P$ are the problem parameters defined on a subspace D of $\mathbb{R}^P$. If the model is given by partial differential equations, such as Computational Fluid Dynamics (CFD) applications, it is assumed the equations are fully discretized and any boundary conditions are included in the algebraic form (7).

The generation of the POD basis uses the method of snapshots introduced in [4]. To build the POD basis, it is assumed evaluations of the high-order model $\boldsymbol{R}$ are available at a set of M points (or samples) in the parametric space $\{\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \dots, \boldsymbol{\mu}_M,\}$, with M typically being much smaller than N, since the evaluation of $\boldsymbol{R}$ can be very expensive. These evaluations, representing the snapshots, are arranged in the $N \times M$ snapshots matrix:

$$\boldsymbol{S} = [\boldsymbol{u}(\boldsymbol{x}, \boldsymbol{\mu}_1), \boldsymbol{u}(\boldsymbol{x}, \boldsymbol{\mu}_2), \dots, \boldsymbol{u}(\boldsymbol{x}, \boldsymbol{\mu}_M)] \tag{8}$$

The deviation matrix $\boldsymbol{D} = [\boldsymbol{u}(\boldsymbol{x}, \boldsymbol{\mu}_1) - \overline{\boldsymbol{u}}, \boldsymbol{u}(\boldsymbol{x}, \boldsymbol{\mu}_2) - \overline{\boldsymbol{u}}, \dots, \boldsymbol{u}(\boldsymbol{x}, \boldsymbol{\mu}_M) - \overline{\boldsymbol{u}}]$ is then built, where $\overline{\boldsymbol{u}}$ is the snapshots mean vector, whose components are evaluated as $\bar{u}_i = \frac{1}{M}\sum_{j=1}^{M} u_i(\boldsymbol{\mu}_j), i = 1,2, \dots, N$. A singular value decomposition (SVD) of $\boldsymbol{D}$ is computed, $\boldsymbol{D} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T$, where $\boldsymbol{U} \in \mathbb{R}^{N\times N}$, $\boldsymbol{V} \in \mathbb{R}^{M\times M}$ and $\boldsymbol{\Sigma} \in \mathbb{R}^{N\times M}$ is a diagonal matrix containing the ordered singular values $\sigma_i \in \mathbb{R},\ \sigma_1 > \sigma_2 > \cdots > \sigma_M$. The POD basis vectors (or modes) $\varphi_i \in \mathbb{R}^N$ are obtained by extracting the first K column vectors of $\boldsymbol{U}$. The problem solution can then be approximated as:

$$\boldsymbol{u} \cong \overline{\boldsymbol{u}} + \sum_{i=1}^{K} \alpha_i \varphi_i \tag{9}$$

Here, α_i are the POD coefficients. In a non-intrusive approach such as used in this paper, equation (9) is considered for each snapshot, $\boldsymbol{u}(\boldsymbol{\mu}_j) = \overline{\boldsymbol{u}} + \sum_{i=1}^{K} \alpha_i(\boldsymbol{\mu}_j)\varphi_i$, and the POD coefficients are determined by a Petrov-Galerkin projection of the snapshots onto the POD basis vectors:

$$\alpha_i(\boldsymbol{\mu}_j) = \left(\boldsymbol{u}(\boldsymbol{\mu}_j) - \overline{\boldsymbol{u}}\right)^T \varphi_i, i = 1,2, \dots, K, j = 1,2, \dots M \tag{10}$$

For many problems of interest, the most important part of the energy distribution is concentrated in the first few modes. The truncation of the basis to K modes is done by selecting the smallest possible $K < M$ such that a desired level of energy capture $\varepsilon \in [0,1]$ is achieved:

$$\frac{\sum_{i=1}^{K} \sigma_i^2}{\sum_{i=1}^{M} \sigma_i^2} \geq \varepsilon \tag{11}$$

In order to obtain a POD approximation of the solution $\boldsymbol{u}(\boldsymbol{\mu})$ at an arbitrary parameter value not included in the sampling $\{\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \dots, \boldsymbol{\mu}_M,\}$, the variation of the POD coefficients over the entire parametric space is captured using ANNs. The ANN is trained using the available parameter values as the input set, and the POD coefficient values as the target output, thus building the correspondence $\{\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \dots, \boldsymbol{\mu}_M,\} \xrightarrow{ANN} \{\boldsymbol{\alpha}(\boldsymbol{\mu}_1), \boldsymbol{\alpha}(\boldsymbol{\mu}_2), \dots, \boldsymbol{\alpha}(\boldsymbol{\mu}_M)\}$. The POD-based ANN surrogate then provides a solution approximation expressed as:

$$\tilde{\boldsymbol{u}}(\boldsymbol{\mu}) = \overline{\boldsymbol{u}} + \sum_{i=1}^{K} \tilde{\alpha}_i(\boldsymbol{\mu})\varphi_i \tag{12}$$

Here, $\tilde{\alpha}_i(\boldsymbol{\mu})$ are the coefficient values approximated by the ANN-POD surrogate at an arbitrary parameter value $\boldsymbol{\mu}$.

The approximate solution can be related to the exact solution as follows:

$$\begin{aligned} \boldsymbol{u}(\boldsymbol{\mu}) &= \overline{\boldsymbol{u}} + \sum_{i=1}^{M} \alpha_i(\boldsymbol{\mu})\varphi_i + D(\boldsymbol{\mu}) = \overline{\boldsymbol{u}} + \sum_{i=1}^{K} \alpha_i(\boldsymbol{\mu})\varphi_i + T(\boldsymbol{\mu}) + D(\boldsymbol{\mu}) = \\ &= \overline{\boldsymbol{u}} + \sum_{i=1}^{K} \tilde{\alpha}_i(\boldsymbol{\mu})\varphi_i + \sum_{i=1}^{K} \alpha_i(\boldsymbol{\mu})\varphi_i - \sum_{i=1}^{K} \tilde{\alpha}_i(\boldsymbol{\mu})\varphi_i + T(\boldsymbol{\mu}) + D(\boldsymbol{\mu}) = \\ &= \overline{\boldsymbol{u}} + \sum_{i=1}^{K} \tilde{\alpha}_i(\boldsymbol{\mu})\varphi_i + N(\boldsymbol{\mu}) + T(\boldsymbol{\mu}) + D(\boldsymbol{\mu}) = \tilde{\boldsymbol{u}}(\boldsymbol{\mu}) + N(\boldsymbol{\mu}) + T(\boldsymbol{\mu}) + D(\boldsymbol{\mu}) \end{aligned} \tag{13}$$

Here, $D(\boldsymbol{\mu})$ is the error arising when the sampling of the parameter space is not sufficiently refined, leading to a lack of enough richness of the POD basis to capture all relevant features appearing in the parameter space D. $T(\boldsymbol{\mu})$ is the error due to truncation of the POD basis to a rank $K < M$, and $N(\boldsymbol{\mu})$ is the error due to the ANN approximation of the POD coefficients at parameter value $\boldsymbol{\mu}$.

One effective strategy of reducing $D(\boldsymbol{\mu})$ is to improve the POD basis quality via an adequate adaptive sampling procedure in which parameter samples are sequentially added to the set while balancing adequate exploration (spanning of the entire solution space) and exploitation (additional refinement in regions of strong, nonlinear solution dependence on parameter

variations). $T(\boldsymbol{\mu})$ is usually small, provided that an adequate energy threshold is chosen ($\varepsilon >$ 0.990-0.995). The ANN approximation error $N(\boldsymbol{\mu})$ can be measured by the error at a finite number of points different from the set of samples used to build the POD basis. Adaptive sampling techniques can be extended to generate additional snapshots in the neighbourhood of parameter sample points where $\|\alpha(\boldsymbol{\mu}) - \tilde{\alpha}(\boldsymbol{\mu})\|$ is high, thus improving the quality of the ANN approximation. Another strategy is to use more complex ANN architectures, with an increased number of hidden layers and number of neurons per layer. Whatever strategy is used, building the surrogate ANN-POD model must remain computationally affordable, and number of snapshots (high-order solutions) added, or the complexity of the ANN (requiring an ever-increasing training effort) must be limited.

The adaptive sampling technique aimed to improve the quality of the POD basis is that developed in [1]. The method starts with a given set of parameter samples $\{\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \dots, \boldsymbol{\mu}_M, \}$ (selected a priori by methods such as Latin Hypercube Sampling (LHS)) and their corresponding snapshots $\boldsymbol{S} = [\boldsymbol{u}(\boldsymbol{x}, \boldsymbol{\mu}_1), \boldsymbol{u}(\boldsymbol{x}, \boldsymbol{\mu}_2), \dots, \boldsymbol{u}(\boldsymbol{x}, \boldsymbol{\mu}_M)]$. A POD basis is generated, and the relative influence of all snapshots on the basis is calculated as:

$$Infl_{Basis}^{Rel}(\boldsymbol{\mu}_j) = \frac{\sum_{i=1}^{K} \sigma_i \left(\frac{1}{\left\| \varphi_i^T \varphi_i^{-j} \right\|} - 1 \right)}{\sum_{k=1}^{M} \sum_{i=1}^{K} \sigma_i \left(\frac{1}{\left\| \varphi_i^T \varphi_i^{-k} \right\|} - 1 \right)}, j = 1,2, \dots, M \tag{14}$$

Here, φ_i^{-j} is the i-th basis vector of a POD basis constructed by leaving out the j-th snapshot from the $\boldsymbol{S}$ matrix, or $\boldsymbol{S}^{-j} = \left[\boldsymbol{u}(\boldsymbol{x}, \boldsymbol{\mu}_1), \boldsymbol{u}(\boldsymbol{x}, \boldsymbol{\mu}_2), \dots, \boldsymbol{u}\left(\boldsymbol{x}, \boldsymbol{\mu}_{j-1}\right), \boldsymbol{0}, \boldsymbol{u}\left(\boldsymbol{x}, \boldsymbol{\mu}_{j+1}\right), \dots, \boldsymbol{u}(\boldsymbol{x}, \boldsymbol{\mu}_M)\right]$. The quantity $Infl_{Basis}^{Rel}(\boldsymbol{\mu}_j)$ is a measure of the total sensitivity of all modes $\varphi_i, i = 1,2, \dots K$ with respect to the j-th snapshot. After equation (14) is evaluated for all parameters in the initial set $\{\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \dots, \boldsymbol{\mu}_M, \}$, the parametric space is heavily populated (using LHS, for example) with a set of candidate sample points $\{\boldsymbol{v}_1, \boldsymbol{v}_2, \dots, \boldsymbol{v}_Q, \}$, with $Q \gg M$. The potential of enrichment of each candidate sample is then evaluated:

$$Pot(\boldsymbol{v}_i) = Infl_{Basis}^{Rel}(\boldsymbol{\mu}_j) \cdot d(\boldsymbol{v}_i, \boldsymbol{\mu}_j), i = 1,2, \dots, Q$$
$$j = argmin_k d(\boldsymbol{v}_i, \boldsymbol{\mu}_k) \tag{15}$$

Here, $d(\boldsymbol{v}_i, \boldsymbol{\mu}_j)$ is the Euclidean distance between two sample points. Through (15), a balance is achieved between improvement of the POD basis and parameter space exploration (introducing the distance effects to a decrease in the potential of enrichment if the candidate sample is too close to an already existing sample). The candidate sample having the highest potential $\boldsymbol{\mu}_{new} = argmax_k Pot(\boldsymbol{v}_k)$ is added to the set, which becomes $\{\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \dots, \boldsymbol{\mu}_M, \boldsymbol{\mu}_{M+1} = \boldsymbol{\mu}_{new}\}$. The procedure continues as explained until the set of parameter sample points reaches a desired maximum size.

The technique outlined earlier can be extended and applied to selecting candidate sampling points which improve the quality of the ANN approximation. Starting with the given set of parameter samples $\{\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \dots, \boldsymbol{\mu}_M, \}$, the POD basis is generated, the ANN is trained and the influence of all snapshots on the ANN-determined coefficients is calculated as:

$$Infl_{Coeff}^{Rel}(\boldsymbol{\mu}_j) = \frac{\sum_{i=1}^{K} \sigma_i \left\| \alpha_i(\boldsymbol{\mu}_j) - \tilde{\alpha}_i{}^{-j}(\boldsymbol{\mu}_j) \right\|}{\sum_{k=1}^{M} \sum_{i=1}^{K} \sigma_i \left\| \alpha_i(\boldsymbol{\mu}_j) - \tilde{\alpha}_i{}^{-k}(\boldsymbol{\mu}_j) \right\|}, j = 1,2, \dots, M \tag{16}$$

Where $\tilde{\alpha}_i{}^{-j}(\boldsymbol{\mu}_j), i = 1,2, \dots, K$ is the set of coefficients obtained with the ANN for sampling point $\boldsymbol{\mu}_j$, but constructed starting from a snapshot matrix, a POD basis and a set of POD coefficients from which the j-th snapshot has been left out. Like the POD improvement strategy, the technique continues with generating a set of candidate sampling points $\{\boldsymbol{v}_1, \boldsymbol{v}_2, \dots, \boldsymbol{v}_Q, \}$, determining the potential of improving the ANN approximation using $Pot(\boldsymbol{v}_i) = Infl_{Coeff}^{Rel}(\boldsymbol{\mu}_j) \cdot d(\boldsymbol{v}_i, \boldsymbol{\mu}_j), i = 1,2, \dots, Q$ and finally adding the candidate sample having the highest potential $\boldsymbol{\mu}_{new} = argmax_k Pot(\boldsymbol{v}_k)$ to the set $\{\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \dots, \boldsymbol{\mu}_M, \boldsymbol{\mu}_{M+1} = \boldsymbol{\mu}_{new}\}$.

It must be noted that the adaptive sampling technique based only on improving the quality of the ANN-approximated coefficients is time-consuming, due to the need of training, for each added parameter sample, a number of neural networks equal to the number of parameters minus one. One potential way of mitigating this is to use a combined, POD basis improvement

and ANN approximation improvement sampling strategy, in which the newly added sampling points are alternatively selected by the two strategies.

In order to assess the ANN-POD method, an academic test case similar to the one used in [22] is chosen, namely the analysis of subsonic flow in a convergent-divergent nozzle of length L. The mathematical model is given by a nonlinear ordinary differential equation:

$$\frac{d}{dx}(\rho UA) = 0, x \in [0, L] \tag{17}$$

Which, after further manipulation, becomes:

$$\frac{d}{dx}\left(\rho_0 A(x)\sqrt{\gamma R T_0} M(x)\left(1 + \frac{\gamma - 1}{2} M^2(x)\right)^{-\frac{\gamma+1}{2(\gamma-1)}}\right) = 0, x \in [0, L] \tag{18}$$

Here, ρ_0 and T_0 are the stagnation density and temperature (which can be held constant and equal to the values at the inlet, assuming isentropic flow), $\gamma = 1.4$ is the ratio of specific heats, $R = 287$ is the specific gas constant, $A(x)$ is the nozzle area distribution and $M(x)$ is the Mach number for which the equation is solved. The nozzle area is parameterized as $A(x) = p_3(x - p_2)^2 + p_1$, using a parameter vector $\boldsymbol{\mu} = [p_1, p_2, p_3] \in \mathbb{R}^3$. The parameter values are limited to $0.015 \leq p_1 \leq 0.025$, $0.8 \leq p_2 \leq 1.2$ and $0.004 \leq p_3 \leq 0.007$.

Equation (18) is solved in a domain of length $L = 3$ on a uniformly spaced grid with $N = 1000$ points using an upwind-biased finite-difference scheme. The ANN-ROM results are compared not only with the high-order solution, but also with the solution given by an intrusive ROM based on the GNAT model [14]. GNAT is one of the most accurate and efficient intrusive ROMs, being successfully applied to complex, turbulent flow problems and nonlinear structural dynamics [14]. It is applicable to implicitly formulated models solved via Gauss-Newton iterations and involves a two-level approximation. In the first level, snapshots of the high-order residual are taken and used to generate a POD basis. The implicit equations are then projected onto the POD basis using a Petrov-Galerkin projection. In the second level, snapshots of the first-level ROM residual and Jacobian are taken and used to build two

iteration-dependent bases. These bases, together with a gappy-type least-squares data reconstruction of the ROM residual and Jacobian, are used to construct the second level hyper-reduced-order model (HROM), which achieves a significant speed-up in comparison with the first-level ROM.

Figure 1 shows the spectrum of singular values for the snapshots' matrix. The POD basis is truncated to $K = 25$ modes (which satisfies an error threshold of 99.99%), while the residual and Jacobian bases required for GNAT are truncated to $K_R = K_J = 100$ modes. In the first test, a number of 125 fixed, uniformly distributed sampling points are chosen in the parameter space. The ANN is configured as a feedforward network using a single hidden layer of neurons using the hyperbolic tangent sigmoid transfer function and a layer of output neurons using a linear transfer function and is trained using the Levenberg-Marquardt algorithm.

Table 1 contains the time required to perform the various stages of building the ROMs (the offline stage). All times are normalised with respect to t_{ref} the time required to collect the 125 high-order model (HOM) snapshots. It can be seen that for small basis size (small number of POD coefficients), increasing the number of hidden layer neurons only has a very small impact on the network training time. Overall, the HROM generation time is approximately 42% higher compared to the ANN-ROM due to the requirement of taking the second set of ROM snapshots, making the ANN-ROM more time efficient in the offline stage.

In order to verify the accuracy of both HROM and ANN-ROM, a set of 64 points uniformly distributed in the parameter space is chosen, all of which are checked to be different to the initial set of 125 used to build the ROMs. The mass flow rate through the nozzle as calculated with the local Mach number at the throat is compared (serving as variable of engineering interest), as well as the root mean square (serving as an indication of the overall ROM accuracy considering all points in the $[0, L]$ interval):

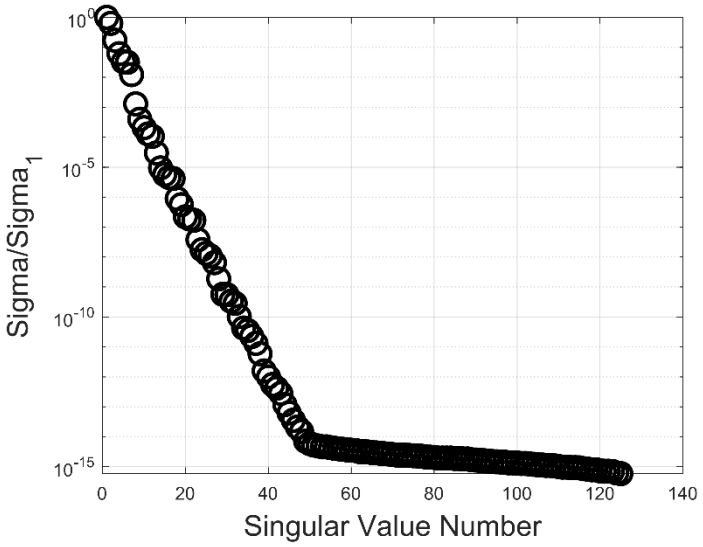


Figure 01. Singular values of the snapshots' matrix for the problem of subsonic flow in a convergent-divergent nozzle

Table 01. Comparison between time required to collect HOM snapshots, collect ROM snapshots required for building HROM and train ANN

	HOM Snapshots Collection	ROM Snapshots Collection	Train ANN with 40 hidden neurons	Train ANN with 60 hidden neurons	Train ANN with 80 hidden neurons
t/t_{ref}	1	0.423	0.0353	0.0366	0.0892

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(M_i^{ROM} - M_i^{HOM}\right)^2} \tag{19}$$

Table 2 shows the average runtime required for the HOM and two ROMs as calculated based on all 64 test runs, normalised with respect to t_{ref} the average runtime of the HOM. It can be observed the HROM achieves a runtime reduction of one order of magnitude, while for the non-intrusive ANN-ROM, the reduction is three orders of magnitude. This is expected to represent a significant advantage for more complex problems such as CFD or FEM analysis.

Table 02. Comparison between average runtimes over 64 test cases

	HOM	HROM	ANN-ROM
t/t_{ref}	1	0.1363	0.0031

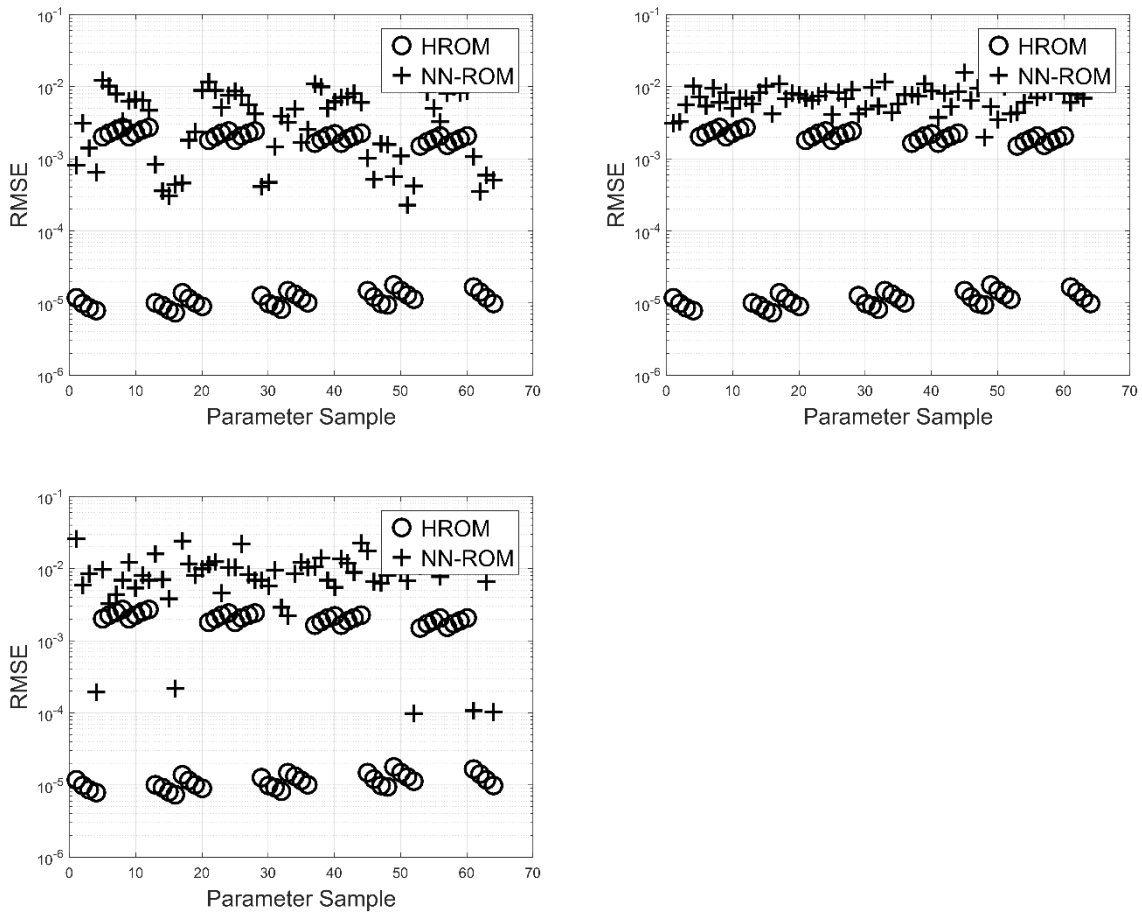


Figure 02. RMSE of HROM and ANN-ROM for all 64 test parameter samples, with a network hidden layer having 40 neurons (top left), 60 neurons (top right) and 80 neurons (bottom left)

Figure 2 depicts a comparison between the RMSE of the HROM and ANN-ROM with 40, 60 and 80 neurons in the hidden layer, for all 64 test parameter samples. Increasing the number of hidden layer neurons tends to level the RMSE values across all test points, but not to significantly increase the average accuracy. It must be kept in mind that training a neural network is not a deterministic process since the training samples are randomly chosen from the provided input set. In addition, overfitting in which weighting factors associated with

different neurons become correlated due to too high a number of neurons can decrease the predictive capabilities of ANNs. These factors can explain why the network with 80 neurons in the hidden layer does not provide any significant improvement compared to the network with 60 neurons. The RMSE values achieved by the HROM are highly dependent on the point in the parameter space. Figure 3 shows the error in mass flow rate calculated with both HROM and ANN-ROM for all 64 test parameter samples. It can be seen that the ANN-ROM achieves an error of the order of $O(10^{-2})$ across the entire parameter space, which is sufficient for many engineering design and optimisation problems, while the HROM achieves a much lower error of the order of $O(10^{-5})$. It must be noted that parameter p_1 directly controls the area at the throat section and has the highest effect on the mass flow rate, while parameters p_2 and p_3 (controlling the position of throat along the nozzle length and the magnitude of area change dA/dx) have a much more limited influence.

The adaptive sampling technique is considered next as a method of increasing the accuracy of the ANN-ROM approach. The total number of parameter samples (and thus HOM snapshots) remains 125. An initial number of 64 uniformly distributed points are taken, with the remaining 61 being sequentially added. The POD basis is truncated to $K = 25$ modes, while the residual and Jacobian bases required for GNAT are truncated to $K_R = K_J = 100$ modes. The ANN is generated using one hidden layer of 60 neurons. The set of candidate samples $\{\boldsymbol{v}_1, \boldsymbol{v}_2, \dots, \boldsymbol{v}_Q,\}$ is generated using LHS and has 625 samples. In the adaptive sampling aimed at improving the ANN coefficients approximation, all samples are added according to equation (10) in order to quantify both the method effectiveness and the time penalties it incurs. Table 3 shows a comparison between the time required to collect the 125 HOM snapshot with the three sampling techniques. All times are normalised with respect to t_{ref} the time required to collect the snapshots in the fixed a-priori sampling technique. It is seen that the adaptive sampling strategy aimed at improving the quality of the POD basis only introduces a 13% increase in the total snapshots collection time. However, the second

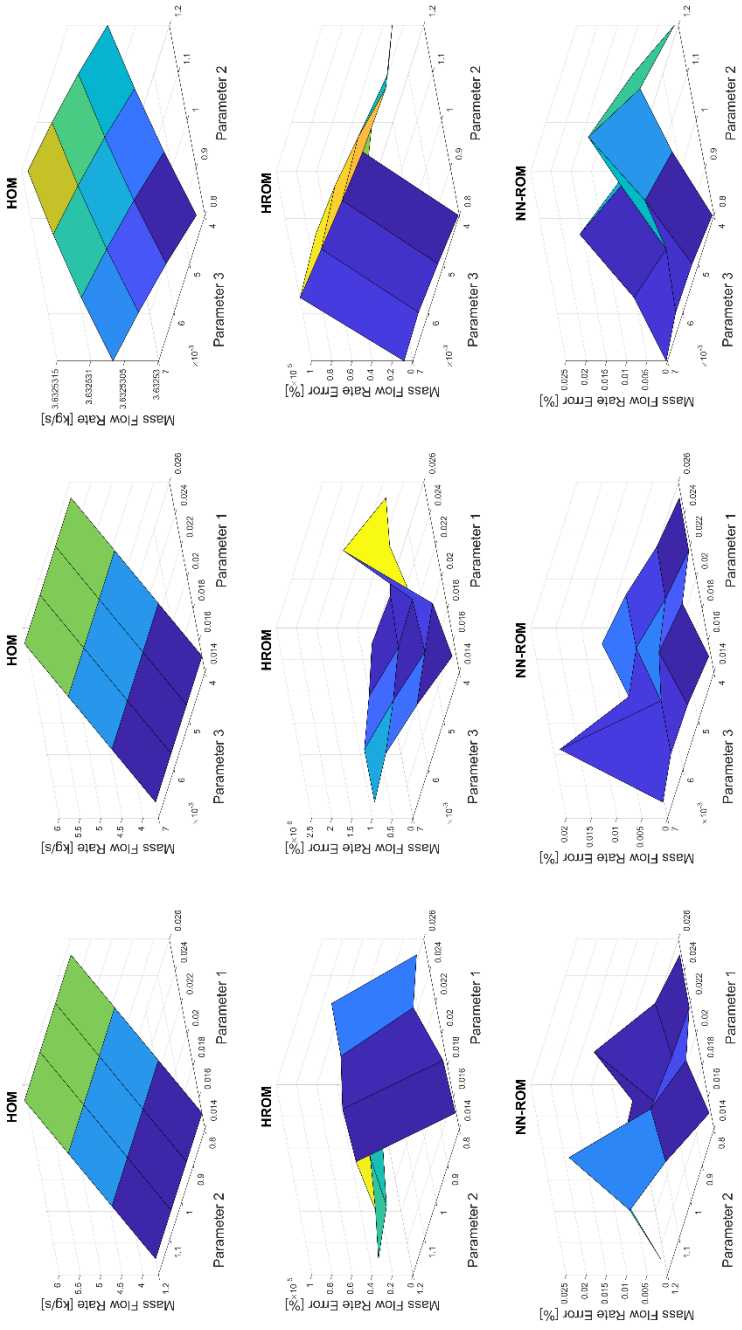


Figure 03. Variation of mass flow rate across parametric space calculated with the HOM, and error in mass flow rate calculated with both HROM and ANN-ROM for all 64 test parameter samples

Table 03. Comparison between HOM snapshots collection times for the three sampling techniques

	Fixed	Adaptive basis improvement	Adaptive ANN approximation improvement
t/t_{ref}	1	1.13	7.21

adaptive sampling technique requires a significant time increase due to continuous re-training of the neural network done as part of estimating the influence factors in equation (16).

Verification of the accuracy of both HROM and ANN-ROM is again made using a set of 64 points uniformly distributed in the parameter space, all of which are checked to be different to the initial set of 125 used to build the ROMs. Figure 4 depicts a comparison between the RMSE of the HROM and ANN-ROM constructed using the three sampling techniques, for all 64 test parameter samples. Both adaptive sampling strategies manage to reduce the RMSE of the ANN-ROM, indicating a better modelling of the system variables throughout the domain. The second adaptive strategy reduces the RMSE by one order or magnitude compared with the fixed sampling strategy, however it does incur a seven-fold increase in the time required to collect the snapshots. The HROM still achieves lower RMSE for a high proportion of the considered test points. Figure 5 shows the error in mass flow rate calculated with both HROM and ANN-ROM, for all 64 test parameter samples. Compared to the fixed sampling results of Figure 3, the adaptive sampling allows for an average one order of magnitude reduction in flow rate prediction error be achieved. The POD basis improvement adaptive sampling represents an effective strategy incurring only a small increase in the time required for snapshots collection while increasing the accuracy of the ANN-ROM approximation.

4. Non-intrusive POD-based ROM for Non-Parametric Time-Dependent Systems

Let the high-order time-dependent non-parametric model be represented by:

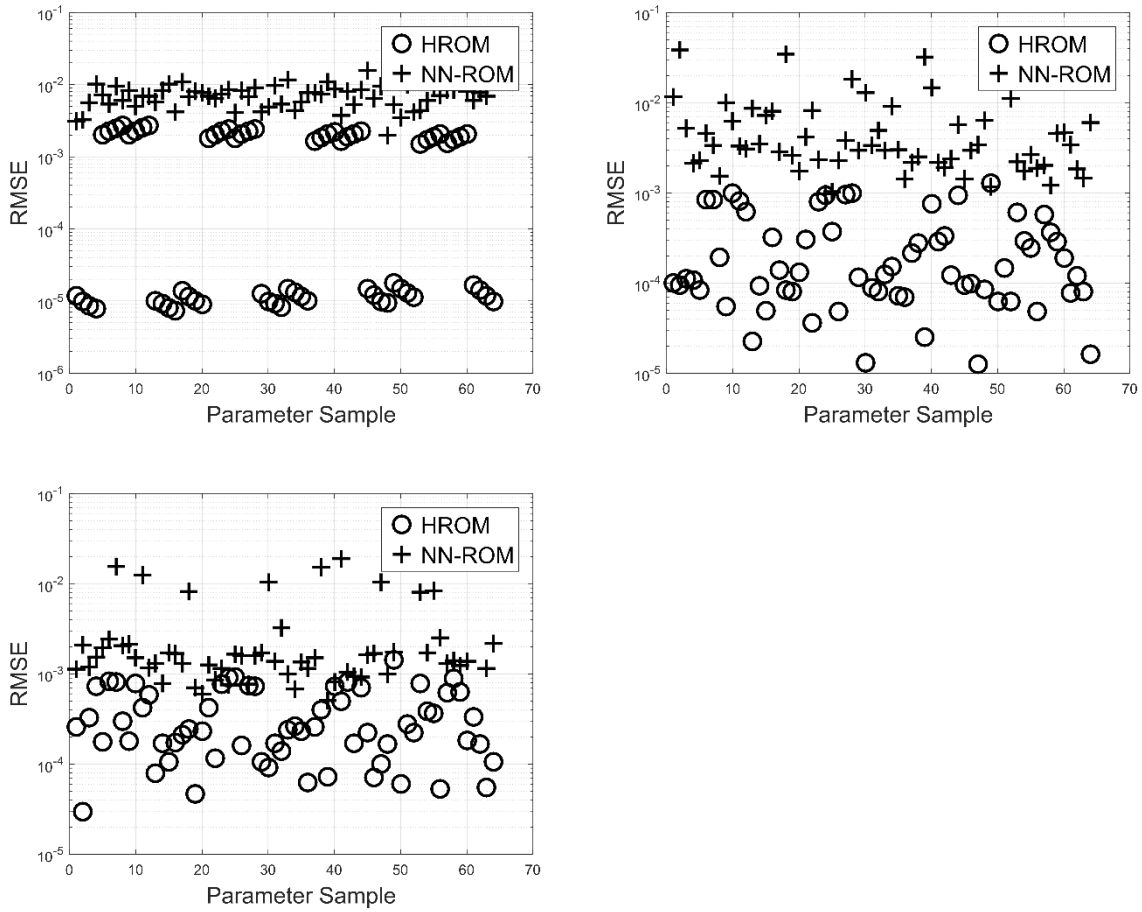


Figure 04. RMSE of HROM and ANN-ROM for all 64 test parameter samples, using fixed sampling (top left), basis improvement adaptive sampling (top right) and ANN approximation improvement adaptive sampling (bottom left)

$$\boldsymbol{R}(\boldsymbol{u}(\boldsymbol{x},t),\boldsymbol{x},t)=\boldsymbol{0} \tag{20}$$

Here, $\boldsymbol{R}\colon \mathbb{R}^N \times [0,\infty) \to \mathbb{R}^N$ with N typically being very large, $\boldsymbol{u}\colon \Omega \subset \mathbb{R}^N \times [0,\infty) \to \mathbb{R}^N$ are the system variables defined on a subspace Ω of $\mathbb{R}^N$, $\boldsymbol{x} \in \mathbb{R}^N$ are spatial coordinates, and t is the time, defined in the semi-infinite interval $[0,\infty)$.

The generation of the ANN-POD surrogate follows the same steps as in the time-independent parametric case. The high-order model $\boldsymbol{R}$ is solved (marched) in time and snapshots are collected at a set of M points (or samples) in time $\{t_1, t_2, \dots, t_M,\}$. Performing the SVD of the deviation matrix $\boldsymbol{D} = [\boldsymbol{u}(\boldsymbol{x},t_1) - \overline{\boldsymbol{u}}, \boldsymbol{u}(\boldsymbol{x},t_2) - \overline{\boldsymbol{u}}, \dots, \boldsymbol{u}(\boldsymbol{x},t_M) - \overline{\boldsymbol{u}}]$, with $\overline{u}_i = \frac{1}{M}\sum_{j=1}^{M} u_i(t_j)\,, i =$

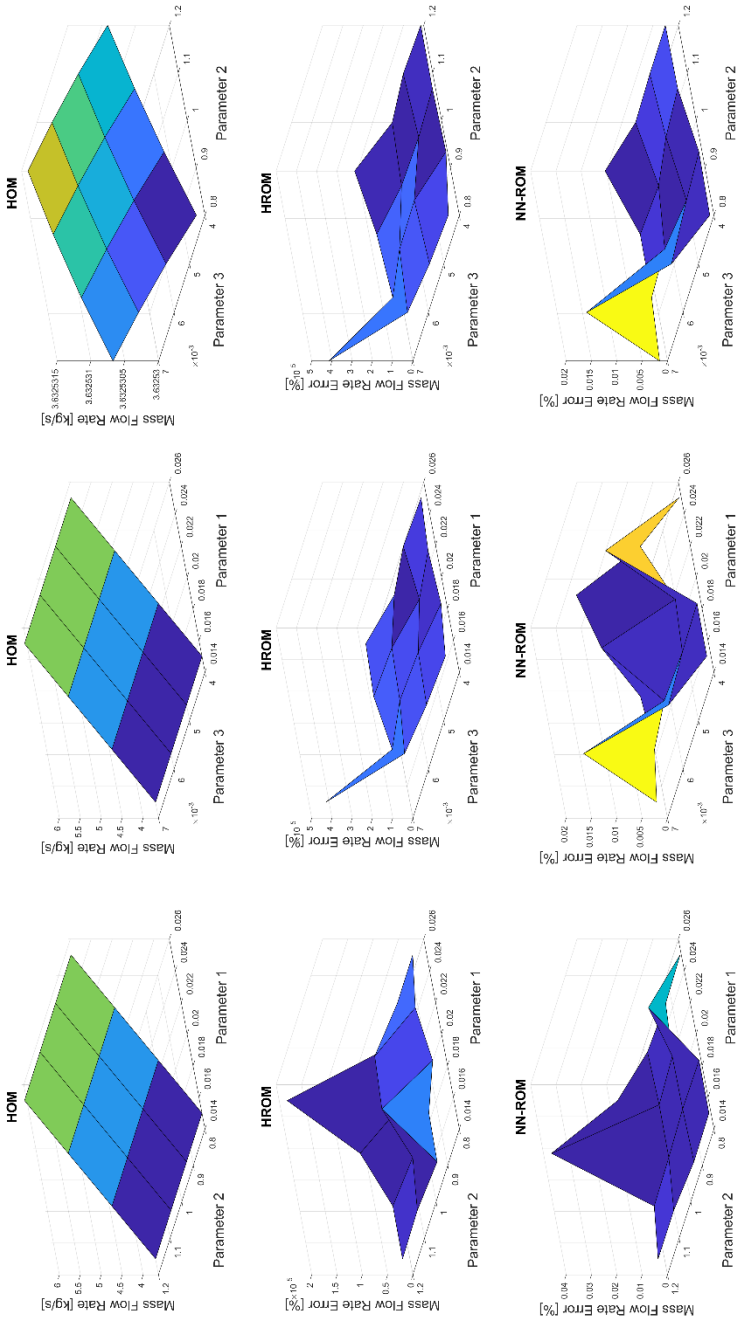


Figure 05. Variation of mass flow rate across parametric space calculated with the HOM, and error in mass flow rate calculated with both HROM and ANN-ROM for all 64 test parameter samples, ROMs build using adaptive sampling

$1,2,\dots,N$, and choosing the first K columns vectors of $\boldsymbol{U}$ provides the required POD basis. The POD coefficients at the set of discrete moments in time are determined by a Petrov-Galerkin projection of the snapshots onto the POD basis vectors, $\alpha_i(t_j) = \left(\boldsymbol{u}(t_j) - \overline{\boldsymbol{u}}\right)^T \varphi_i, i = 1,2,\dots,K, j = 1,2,\dots M$. The ANN is then trained using the available set of POD coefficient values to build the correspondence $\{t_1, t_2, \dots, t_M,\} \xrightarrow{ANN} \{\boldsymbol{\alpha}(t_1), \boldsymbol{\alpha}(t_2), \dots, \boldsymbol{\alpha}(t_M)\}$, and the solution at any moment in time is calculated as:

$$\tilde{\boldsymbol{u}}(t) = \overline{\boldsymbol{u}} + \sum_{i=1}^{K} \tilde{\alpha}_i(t)\varphi_i \tag{21}$$

Testing the ANN-POD for time-dependent problems is done using the inviscid Burgers equation. The mathematical problem is defined as:

$$\begin{gathered} \frac{\partial U}{\partial t} + \frac{1}{2}\frac{\partial U^2}{\partial x} = 0, x \in [0, L] \\ U(x, t = 0) = 0 \\ U(x = 0, t) = \sqrt{5}, t \geq 0 \\ \frac{\partial U}{\partial x}(x = L, t) = 0, t \geq 0 \end{gathered} \tag{22}$$

The inviscid Burgers equations is used a simplified test problem for the compressible Euler equations since it allows discontinuous solutions similar to shock waves in a fluid flow. Equation (16) is solved in a domain of length $L = 100$ on a uniformly spaced grid with $N = 1000$ points using an implicit second-order TVD Lax-Friedrichs scheme with an added artificial viscosity term. This choice ensures the HOM solution will be free of numerically induced oscillations around the shock wave. The equation is solved in the time interval $t \in [0,50]$, with a step $\Delta t = 0.1$.

The ANN-ROM results are compared not only with the high-order solution, but also with the solution given by the GNAT ROM. Snapshots of the HOM solution are taken every 5 time steps, giving a total of 100 snapshots for the chosen interval and Δt. The POD basis is truncated to $K = 25$ modes (which satisfies an error threshold of 99.90%), while the residual

and Jacobian bases required for GNAT are again truncated to $K_R = K_J = 100$ modes. The feedforward ANN is configured using a single hidden layer of neurons using the hyperbolic tangent sigmoid transfer function and a layer of output neurons using a linear transfer function and is trained using the Levenberg-Marquardt algorithm. Numerical tests indicate that 40 neurons in the hidden layer are sufficient to provide good accuracy while avoiding overfitting.

A comparison between the HOM, HROM and ANN-ROM at five selected moments in time is shown in Figure 6. It can be observed that both HROM and ANN-ROM approximations capture extremely well the HOM solution for all time instances within $[0,50]$, with only some very small oscillations around the advancing shock. Since these oscillations are present in the ANN-ROM as well, they are attributed to local projection errors in calculating the POD coefficients. The solution at time $t = 58$, being outside the initial sampling interval $[0,50]$, is not captured by either of the ROMs. This is due to lack of information in the POD basis related to any HOM snapshot for $t > 50$, and is an expected behaviour. ROMs can be very accurate at approximating the system variables in the subspace spanned by the POD basis vectors, but not at predicting values outside of range of HOM snapshots.

Table 4 contains the time required to capture the HOM snapshots and build the POD basis and generate the two ROMs (the offline stage for both approaches). All times are normalised with respect to t_{ref} the time required to build the POD basis. Collecting the ROM snapshots required for the HROM requires a time comparable to collecting the HOM snapshots due to the inefficiency of the simple ROM for this highly non-linear problem. Using the ANN-ROM allows for reductions in the time required for the offline stage, while keeping excellent accuracy (as seen in Figure 6).

Table 5 shows the runtime required for the HOM and two ROMs to numerically solve problem (16) in the time interval $t \in [0,50]$, normalised with respect to t_{ref} the runtime of the HOM. The HROM reduces the solution time to approximately one-third of the HOM solution time, while the ANN-ROM requires only 2% of the HOM solution time.

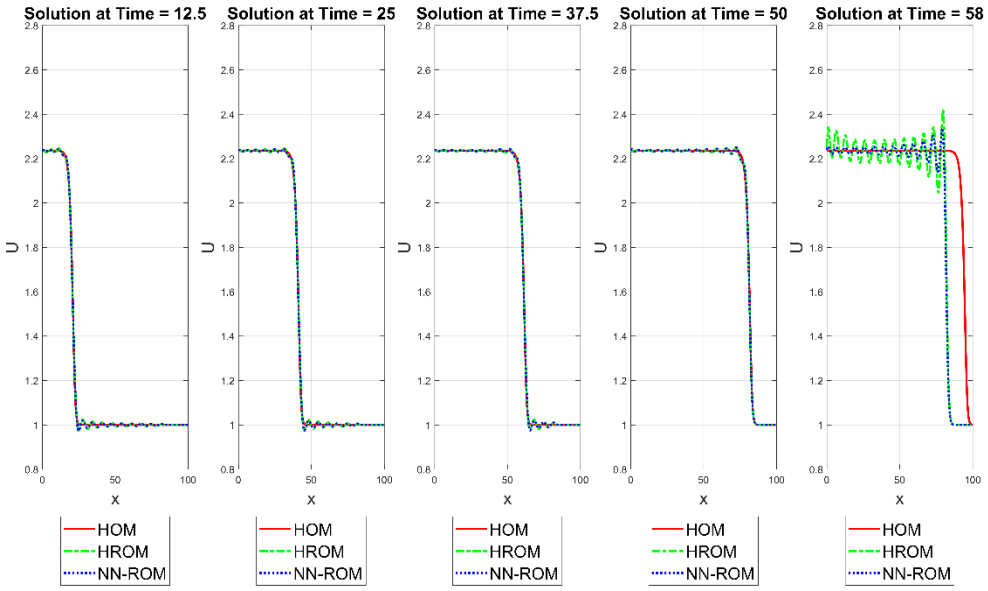


Figure 06. Solution of inviscid Burgers equation as obtained with HOM, HROM and ANN-ROM at five instances in time

Table 04. Comparison between HOM snapshot collection plus POD basis and ROM generation times

	HOM Snapshots Collection plus POD Basis	ROM Snapshots Collection plus HROM Generation	ANN-ROM Training
t/t_{ref}	1	0.897	0.1490

Table 05. Comparison between runtimes of HOM and the two ROMs

	HOM	HROM	ANN-ROM
t/t_{ref}	1	0.31	0.021

5. Non-intrusive POD-based ROM for Parametric Time-Dependent Systems

Let the high-order time-dependent parametric model be represented by:

$$\boldsymbol{R}(\boldsymbol{u}(\boldsymbol{x},t,\boldsymbol{\mu}),\boldsymbol{x},t,\boldsymbol{\mu})=\boldsymbol{0} \tag{23}$$

Here, $\boldsymbol{R}\colon \mathbb{R}^N \times \mathbb{R}^P \times [0,\infty) \to \mathbb{R}^N$ with N typically being very large, $\boldsymbol{u}\colon \Omega \subset \mathbb{R}^N \times \mathbb{R}^P \times [0,\infty) \to \mathbb{R}^N$ are the system variables defined on a subspace Ω of $\mathbb{R}^N$, $\boldsymbol{x} \in \mathbb{R}^N$ are spatial coordinates, t is the time, defined in the semi-infinite interval $[0,\infty)$ and $\boldsymbol{\mu}\colon \mathrm{D} \subset \mathbb{R}^P \to \mathbb{R}^P$ are the problem parameters defined on a subspace D of $\mathbb{R}^P$.

A set of M points (or samples) in the parametric space $\{\boldsymbol{\mu}_1,\boldsymbol{\mu}_2,\dots,\boldsymbol{\mu}_M,\}$, with M typically being much smaller than N, is initially chosen. The high-order model $\boldsymbol{R}$ from equation (23) is then solved (marched) in time for each parameter sample $\boldsymbol{\mu}_p$ and snapshots are collected at a set of MT points (or samples) in time $\{t_1,t_2,\dots,t_{MT},\}$ in order to generate the snapshots matrix $\boldsymbol{S}(\boldsymbol{\mu}_p)=\left[\boldsymbol{u}(\boldsymbol{x},t_1,\boldsymbol{\mu}_p),\boldsymbol{u}(\boldsymbol{x},t_2,\boldsymbol{\mu}_p),\dots,\boldsymbol{u}(\boldsymbol{x},t_{MT},\boldsymbol{\mu}_p)\right]$. The POD basis $\varphi_i(\boldsymbol{\mu}_p), i=1,2,\dots,K$ and the set of POD coefficient $\alpha_i(t_j,\boldsymbol{\mu}_p), i=1,2,\dots,K, j=1,2,\dots MT$ are determined using the steps described earlier in the paper.

Constructing the non-intrusive ROM requires a two-stage process. In the first stage, an ANN is trained for each parameter value, ANN which captures the unsteady behaviour of model (23) for that specific sample in the parametric space:

$$\{t_1,t_2,\dots,t_{MT},\}\xrightarrow{ANN(\boldsymbol{\mu}_p)}\{\boldsymbol{\alpha}(t_1,\boldsymbol{\mu}_p),\boldsymbol{\alpha}(t_2,\boldsymbol{\mu}_p),\dots,\boldsymbol{\alpha}(t_{MT},\boldsymbol{\mu}_p)\},\boldsymbol{\mu}_p=\boldsymbol{\mu}_1,\boldsymbol{\mu}_2,\dots,\boldsymbol{\mu}_M \tag{24}$$

With the ANNs of (18) generated and trained, the solution at any moment in time can be calculated as:

$$\tilde{\boldsymbol{u}}(t,\boldsymbol{\mu}_p)=\bar{\boldsymbol{u}}(,\boldsymbol{\mu}_p)+\sum_{i=1}^{K}\tilde{\alpha}_i(t,\boldsymbol{\mu}_p)\varphi_i(,\boldsymbol{\mu}_p),\boldsymbol{\mu}_p=\boldsymbol{\mu}_1,\boldsymbol{\mu}_2,\dots,\boldsymbol{\mu}_M \tag{25}$$

Where $\tilde{\alpha}_i(t,\boldsymbol{\mu}_p)$ are the POD coefficients approximated at time t by each of the ANNs trained in (24) and are given by $\tilde{\boldsymbol{\alpha}}(t,\boldsymbol{\mu}_p)=ANN(\boldsymbol{\mu}_p)(t)$. The second stage of the process is centred around constructing the approximation in the parameter space. Let t_D be the desired time for

which the solution of (23) must be determined. Using equation (25), the approximation at t_D of the high-order solution $\tilde{\boldsymbol{u}}(t_D, \boldsymbol{\mu}_p), p = 1,2,\dots,M$ is calculated at all available parameter samples. Next, an ANN is trained using the available set of $\tilde{\boldsymbol{u}}$ values to build the correspondence:

$$\{\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \dots, \boldsymbol{\mu}_M,\} \xrightarrow{ANN} \{\tilde{\boldsymbol{u}}(t_D, \boldsymbol{\mu}_1), \tilde{\boldsymbol{u}}(t_D, \boldsymbol{\mu}_2), \dots, \tilde{\boldsymbol{u}}(t_D, \boldsymbol{\mu}_M)\} \tag{26}$$

The total number of parameter samples varies significantly between problems, and might be, for some problems, much lower compared to training set size required for generating a typical accurate feedforward ANN. In addition, even when the number of sampling points is sufficient, the training required for (26) is performed during the online stage of the algorithm. This calls for the training to be completed in a matter of seconds and makes typical hyperparameter tuning impossible to perform. To alleviate these problems, the ANNs in (26) are configured in two layers, one using Radial Basis Functions (RBFs) and the second using a linear activation function. These networks are essentially configured as RBF-based nonlinear regression networks which require no hyperparameter tuning and the training only involves a single small linear system solution (whose size is driven by the total number of training samples) and matrix-vector and vector-vector multiplications (whose sizes are driven by the dimension of the training output, in this case by the size N of the discretised HOM) instead of iterative gradient-based backpropagation schemes. Thus, training is completed in mere seconds even when thousands of parameter sample points are provided. These features make such networks particularly suitable for usage during the online stage of the ROM deployment.

Finally, the solution $\hat{\boldsymbol{u}}$ of (23) at the desired time t_D and an arbitrary parameter value $\boldsymbol{\mu}_D$ is given by the ANN trained in (26) and is obtained at all grid points:

$$\hat{\boldsymbol{u}}(t_D, \boldsymbol{\mu}_D) = ANN(\boldsymbol{\mu}_D) \tag{27}$$

The first test is done using the viscous Burgers equation, a simplified test problem for the compressible Navier-Stokes equations. The mathematical problem is defined as:

$$\frac{\partial U}{\partial t}+\frac{1}{2}\frac{\partial U^2}{\partial x}=D\frac{\partial^2 U}{\partial x^2}, x\in[0,L]$$
$$U(x,t=0)=0$$
$$U(x=0,t)=\sqrt{5}, t\geq 0 \quad (28)$$
$$\frac{\partial U}{\partial x}(x=L,t)=0, t\geq 0$$

Here, D represent the diffusion coefficient and represents the parameter for the problem, being defined the interval $0.0001\leq D\leq 10$. Same as before, Equation (22) is solved in a domain of length $L=100$ on a uniformly spaced grid with $N=1000$ points using an implicit second-order TVD Lax-Friedrichs scheme. The equation is solved in the time interval $t\in[0,50]$, with a step $\Delta t=0.1$. A number of 10 fixed, uniformly distributed sampling points are chosen in the parameter space. For each parameter sample, snapshots of the HOM solution are taken every 5 time steps, giving a total of 100 snapshots for the chosen interval and Δt. The POD basis is truncated to $K=25$ modes, the spectrum of singular values for one of the snapshots' matrices being shown in Figure 7.

The feedforward ANN used for approximating the unsteady behaviour is configured using a single hidden layer of 40 neurons using the hyperbolic tangent sigmoid transfer function and a layer of output neurons using a linear transfer function. Since the number of parameter samples (and thus the size of the available training set) is relatively low, the ANN used for approximating the variation in the parametric space (20) is generated using radial basis functions (RBFs), being configured as an RBF network of 20 neurons.

In order to verify the accuracy of the ANN-ROM, a set of 5 random points in the parameter space is chosen, and for each of these points the RMSE between the HOM and the ANN-ROM at 5 instances in time is determined, the results being summarised in Table 6. Low RMSE values are obtained for all 25 instances investigated, of the order of $O(10^{-2})$ to $O(10^{-4})$ with lower values at high D values. This behaviour is expected due to the smoothing of the discontinuity by increasing viscosity values, thus eliminating the small oscillations from the

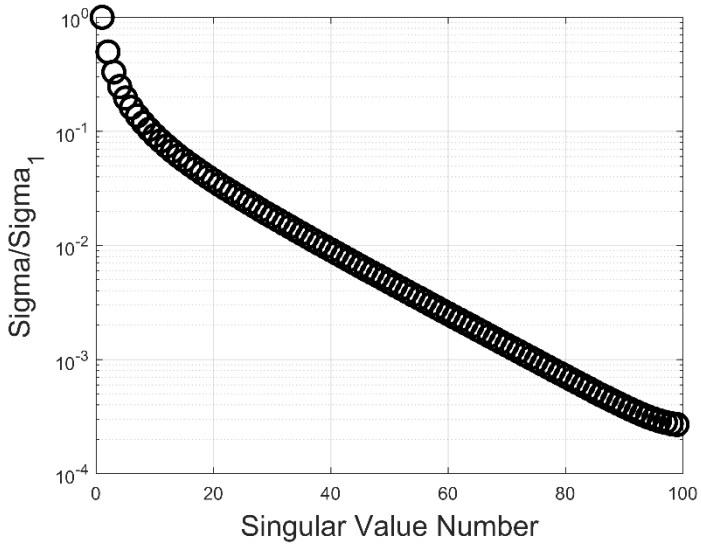


Figure 07. Singular values of the snapshots' matrix for the viscous Burgers equation

Table 06. RMSE at each 5 selected time instances for all 5 parameter test samples

	$t = 10$	$t = 20$	$t = 30$	$t = 40$	$t = 50$
$D = 0.00035$	0.0106	0.0102	0.0098	0.0090	0.0087
$D = 0.018$	0.0100	0.0097	0.0094	0.0086	0.0094
$D = 0.77$	0.0102	0.0117	0.0124	0.0129	0.0146
$D = 1.5$	0.0059	0.0068	0.0072	0.0075	0.0078
$D = 8.8$	0.00018	0.00031	0.00041	0.00035	0.00027

projection step required to determine the POD coefficients. In addition, Figure 8 shows a comparison between the HOM and ANN-ROM solution for $D = 1.5$.

A comparison between the CPU runtimes of HOM and the ROM (as shown earlier in the paper) is not presented in detail for either this test case or the next one. The justification for this omission lies in the relative simplicity of the HOM test cases, and the architecture and ultimate purpose of the ANN-ROM proposed, which is for HOMs involving a considerable number of degrees of freedom. It must be stressed that generating the ANN approximation in the parameter space (26) must be done during the online stage. Although RBF-based networks are used, which train in a matter of seconds for hundreds of sample points, for a test case in

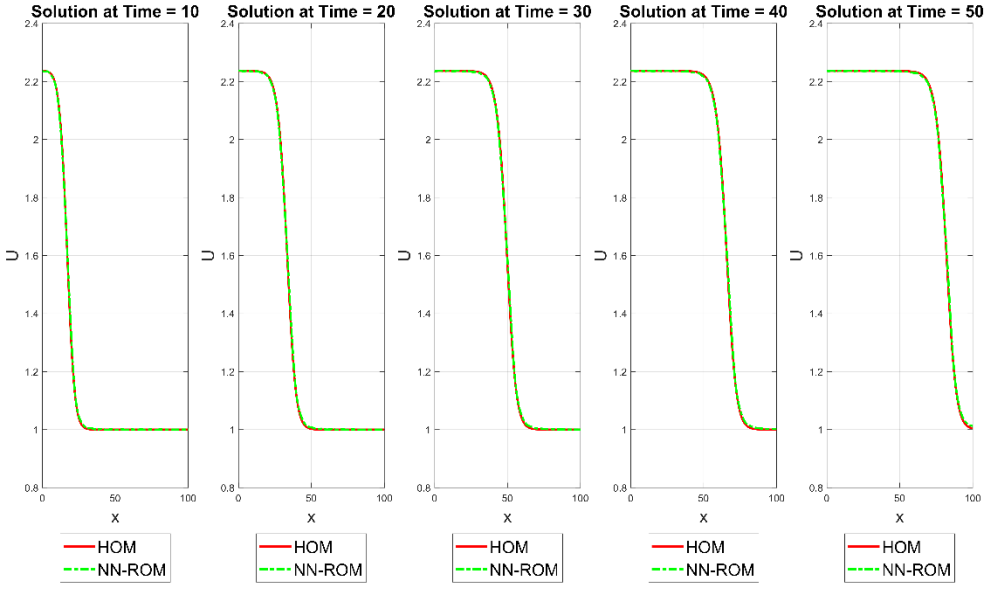


Figure 08. Solution of viscous Burgers equation as obtained with HOM and ANN-ROM at five instances in time

which the HOM has only $N = 1000$ degrees of freedom, the HOM runtime is shorter than the ROM online runtime. To provide some indication of the expected performance, Table 7 presents the runtime for the HOM and the ANN-ROM for problem (28) when N is progressively increased, the ROM runtime t being normalised with respect to t_{ref} the runtime of the HOM. No attempt has been made to optimise the HOM code for faster runtime and obtaining results for a grid containing 10^6 points was not possible. For the chosen test case having $N = 1000$, running the unsteady ROM up to a desired time is much faster than training the ANN approximation in the parameter space and obtaining the ROM results at a single instance in time. As N increases, the trend quickly reverses, even though the ROM runtime also increases due to working with arrays $\widetilde{\boldsymbol{u}}$ of increasing size while generating (25) and (26) online. Based on these observations, it is expected that for a CFD-based HOM containing millions of degrees of freedom, the ANN-ROM provides a significant speed-up.

Table 07. Comparison between runtimes of HOM and ROM

	$N = 10^3$	$N = 10^4$	$N = 10^5$	$N = 5 \times 10^5$	$N = 10^6$
t/t_{ref}	18.67	2.07	0.061	0.0083	Unsuccessful for HOM

A second, more complex but still one-dimensional test can be derived from the viscous Burgers equation by considering a parameterised boundary condition and source term. The equation is:

$$\begin{gathered} \frac{\partial U}{\partial t} + \frac{1}{2}\frac{\partial U^2}{\partial x} = D\frac{\partial^2 U}{\partial x^2} + S(x), x \in [0, L] \\ U(x, t = 0) = 0 \\ U(x = 0, t) = B, t \geq 0 \\ U(x = L, t) = 0, t \geq 0 \\ S(x) = C_1 e^{C_2 x} \end{gathered} \tag{29}$$

Compared to the previous case, equation is solved in the time interval $t \in [0,45]$, on the same domain of length $L = 100$ on a uniformly spaced grid with $N = 1000$. The time interval is reduced for the simple reason of accounting for the variations in the solution due to the introduction of the source term and the different intervals on which the parameters are defined. A small number of only 81 sampling points are chosen in the parameter space (3 uniformly distributed samples for each parameter). This small number of samples is justified by the requirement to keep the overall number of HOM solutions within reasonable limits.

The parameters are defined in the intervals $D \in [0.001,0.1]$, $B \in \left[\sqrt{4.5}, \sqrt{5.5}\right]$, $C_1 \in [0.01,0.03]$ and $C_2 \in [0.015,0.04]$. The POD basis is truncated to $K = 30$ modes. The ANN used for approximating the variation in the parametric space is configured as an RBF network of 20 neurons. Figure 9 presents a comparison between the HOM and ANN-ROM solutions at five instances in time, for $D = 0.05$, $B = \sqrt{4.8}$, $C_1 = 0.022$ and $C_2 = 0.019$. I can be clearly seen that the approximation is significantly affected by spurious oscillations, the largest amplitude

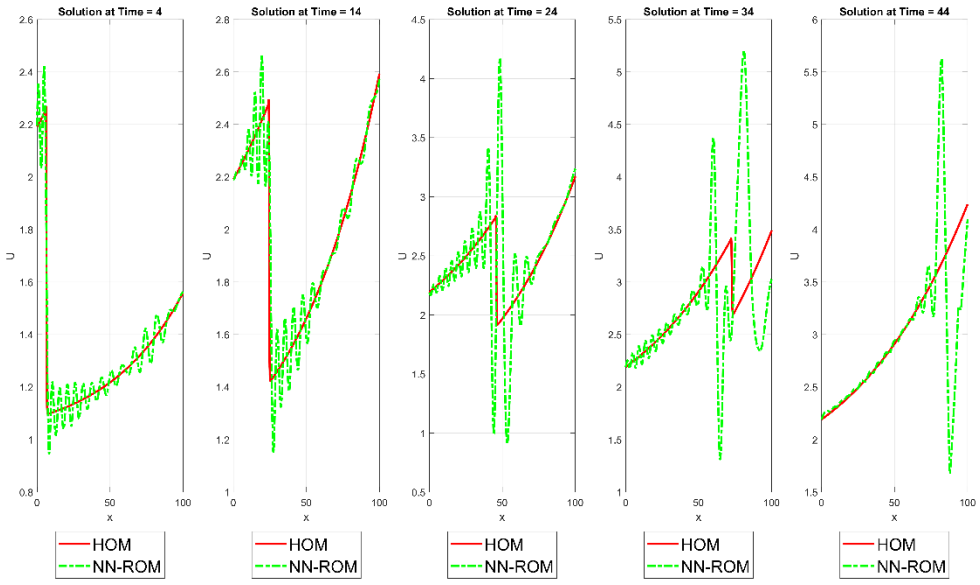


Figure 09. Solution of viscous Burgers equation with source term as obtained with HOM and ANN-ROM at five instances in time

being registered before and after the shock. These oscillations appear to grow in magnitude with time, leading to unphysical results and persisting even after the shock has travelled outside of the domain of interest. The source of these oscillations lies in the lack of richness of the POD basis due to small number of parameter samples chosen, leading to an inaccurate ANN over the parameter space.

An immediate approach to improve the quality of the approximation is to simply increase the number of parameter samples. However, this approach may prove feasible only for problems with a low-dimensional parametric space.

Problems with a high-dimensional parametric space are significantly affected by what is known as the curse of dimensionality. To illustrate, assuming $\boldsymbol{\mu} \in \mathbb{R}^4$ and 5 samples per dimension (two additional samples per parameter over the ones chosen), the total number of parameter combinations needing to be analysed increases to 81 from 625. For very intensive turbulent CFD or non-linear FEM simulations, analysing 625 unsteady cases in order to capture the

HOM snapshots required to build the ROMs may be a computational effort which is simply too expensive to undertake.

An alternative would be to improve the quality of ANN-ROM approximation not simply by increasing the number of samples taken, but rather by a more judicious choice of parameter samples. This is especially true in instances where the system response depends nonlinearly on the values of some or all of the parameters. The adaptive sampling strategy presented earlier can be extended to this effect.

Assume a small, initial set of parameter samples $\{\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \dots, \boldsymbol{\mu}_M, \}$ is chosen, the POD basis $\varphi_i(\boldsymbol{\mu}_p), i = 1,2, \dots, K$ and coefficients are determined and the ANNs which capture the unsteady behaviour of the system at each of these parameter samples are built as described earlier $\{t_1, t_2, \dots, t_{MT}, \} \xrightarrow{ANN(\boldsymbol{\mu}_p)} \{\boldsymbol{\alpha}(t_1, \boldsymbol{\mu}_p), \boldsymbol{\alpha}(t_2, \boldsymbol{\mu}_p), \dots, \boldsymbol{\alpha}(t_{MT}, \boldsymbol{\mu}_p)\}, \boldsymbol{\mu}_p = \boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \dots, \boldsymbol{\mu}_M$. Next, a small number NT of representative instances in time is chosen, and the second set of ANNs is built based on (26):

$$\{\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \dots, \boldsymbol{\mu}_M, \} \xrightarrow{ANN_i} \{\tilde{\boldsymbol{u}}(t_i, \boldsymbol{\mu}_1), \tilde{\boldsymbol{u}}(t_i, \boldsymbol{\mu}_2), \dots, \tilde{\boldsymbol{u}}(t_i, \boldsymbol{\mu}_M)\}, i = 1,2, \dots, NT \tag{30}$$

In order to maintain the training time within reasonable limits, RBF networks should be used for this and all subsequent steps due to the training time being orders of magnitude lower compared to typical feedforward neural networks. The relative influence of each parameter sample on the overall approximation is determined as:

$$Infl_{Coeff}^{Rel}(\boldsymbol{\mu}_j) = \frac{\sum_{i=1}^{NT} \left\| \tilde{\boldsymbol{u}}(t_i, \boldsymbol{\mu}_j) - \tilde{\boldsymbol{u}}^{-j}(t_i, \boldsymbol{\mu}_j) \right\|}{\sum_{k=1}^{M} \sum_{i=1}^{NT} \left\| \tilde{\boldsymbol{u}}(t_i, \boldsymbol{\mu}_j) - \tilde{\boldsymbol{u}}^{-j}(t_i, \boldsymbol{\mu}_j) \right\|}, j = 1,2, \dots, M \tag{31}$$

Here, $\tilde{\boldsymbol{u}}(t_i, \boldsymbol{\mu}_j) = ANN_i(\boldsymbol{\mu}_j)$ is the approximation of the solution at time instance t_i and parameter sample $\boldsymbol{\mu}_j$ using the ANNs constructed in (30), while $\tilde{\boldsymbol{u}}^{-j}(t_i, \boldsymbol{\mu}_j) = ANN_i^{-j}(\boldsymbol{\mu}_j)$ represents the same approximation, but as obtained with ANNs constructed by leaving out the j-th parameter sample from the training set:

$$\{\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \dots, \boldsymbol{\mu}_{j-1}, \boldsymbol{\mu}_{j+1}, \dots, \boldsymbol{\mu}_M,\}$$

$$\xrightarrow{ANN_i^{-j}} \{\tilde{\boldsymbol{u}}(t_i, \boldsymbol{\mu}_1), \tilde{\boldsymbol{u}}(t_i, \boldsymbol{\mu}_2), \dots, \tilde{\boldsymbol{u}}(t_i, \boldsymbol{\mu}_{j-1}), \tilde{\boldsymbol{u}}(t_i, \boldsymbol{\mu}_{j+1}), \dots, \tilde{\boldsymbol{u}}(t_i, \boldsymbol{\mu}_M)\}, i = 1,2, \dots, NT \tag{32}$$

The adaptive sampling procedure continues as was outline earlier in the paper. After equation (32) is evaluated for all parameters in the initial set $\{\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \dots, \boldsymbol{\mu}_M,\}$, the parametric space is heavily populated with a set of candidate sample points $\{\boldsymbol{v}_1, \boldsymbol{v}_2, \dots, \boldsymbol{v}_Q,\}$, with $Q \gg M$. The potential of enrichment of each candidate sample is then evaluated to balance local exploitation and global exploration of the parameter space:

$$Pot(\boldsymbol{v}_i) = Infl_{Coeff}^{Rel}(\boldsymbol{\mu}_j) \cdot d(\boldsymbol{v}_i, \boldsymbol{\mu}_j), i = 1,2, \dots, Q$$
$$j = argmin_k d(\boldsymbol{v}_i, \boldsymbol{\mu}_k) \tag{33}$$

Finally, the candidate sample having the highest potential $\boldsymbol{\mu}_{new} = argmax_k Pot(\boldsymbol{v}_k)$ is added to the set, which becomes $\{\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \dots, \boldsymbol{\mu}_M, \boldsymbol{\mu}_{M+1} = \boldsymbol{\mu}_{new}\}$, and the procedure continues until the desired number of samples have been added to the set.

Problem (29) is attempted a second time, with an initial set of 16 parameter sample points defined on the boundaries of each parameter, and 65 points added using the adaptive sampling procedure, for the same total of 81 samples. The results for $D = 0.05$, $B = \sqrt{4.8}$, $C_1 = 0.022$ and $C_2 = 0.019$, at the same five instances as shown previously, are illustrated in Figure 10. It is observed that the proposed adaptive sampling strategy ensures a choice of sampling points which eliminates the spurious oscillation present in the previous results, thus providing an overall better approximation to the solution. The RMSE values at the five selected instances in time for both fixed-sampling and adaptive sampling ANN-ROMs are provided in Table 8.

The moving shock is not approximated very well at the later instances in time, becoming diffused over an increasing number of points in the domain. A potential improvement could be achieved by a better selection of the NT instances at which the ANNs in (30) and (31) are constructed, but presently no clear method of selection has emerged as being more efficient and successful than a simple random selection.

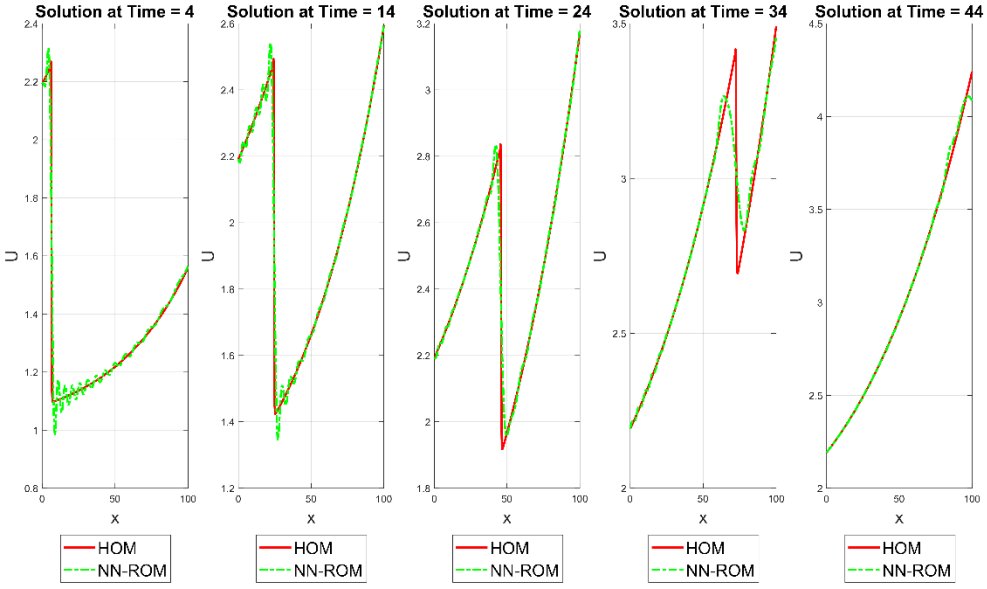


Figure 10. Solution of viscous Burgers equation with source term as obtained with HOM and ANN-ROM with adaptive sampling.

Table 08. RMSE at each 5 selected time instances for fixed and adaptive sampling strategies

	$t = 4$	$t = 14$	$t = 24$	$t = 34$	$t = 44$
Fixed sampling	0.0603	0.1056	0.5027	0.7210	0.5921
Adaptive sampling	0.0453	0.0454	0.0510	0.0671	0.0235

As expected, improving the quality of the ANN-ROM approximation via adaptive sampling comes with the cost of a higher time required to build the model. Table 9 shows a comparison between the total time required to build the POD basis, generate the $ANN(\boldsymbol{\mu}_p), p = 1,2,\dots,M$ for predicting the unsteady behaviour at all parameter samples and generate the networks to capture the behaviour across the parameter space in the two sampling approaches. All times required for HOM snapshots collection were eliminated from the total. The times are

Table 09. Comparison between ANN-ROM generation times for the two sampling techniques

	Fixed	Adaptive sampling
t/t_{ref}	1	2.65

normalised with respect to t_{ref} the time required in the fixed sampling approach. For 81 sampling points, the adaptive sampling strategy requires 2.65 times more time to build the ANN-ROM, and this value will increase as the total number of parameter samples chosen by the strategy increases. However, the relatively good results obtained with only 81 samples for a problem having $\boldsymbol{\mu} \in \mathbb{R}^4$ is indication that the method can perform well with a relatively low number of samples, and the increasing computational costs associated with evaluating (31) can thus be offset.

Finally, a set of 256 points uniformly distributed in the parameter space is chosen, all of which are checked to be different to the initial set of 81 points used to build the ROM with the adaptive sampling technique. For each point, both HOM and ANN-ROM are marched in time and the RMSE is calculated at 20 uniformly distributed time instances and summed. Figure 11 shows the total (summed up) RMSE values across the entire parametric space. In is observed that the ANN-ROM achieves a relatively uniform accuracy across the entire space, indicating good potential to be used in trade studies and ROM-based optimisation.

It is important to check the robustness of the method with respect to perturbations and noise in the ROM generation stage. For this purpose, the source term and parameterised boundary condition term in (29) are modified:

$$U(x=0,t) = B + \delta B(t), t \geq 0$$
$$S(x) = C_1 e^{C_2 x} + \delta S(t) \tag{34}$$

The perturbation terms $\delta B(t)$ and $\delta S(t)$ are randomly generated at each time step of the HOM solution during the ROM generation stage. Two scenarios are considered, in the first the

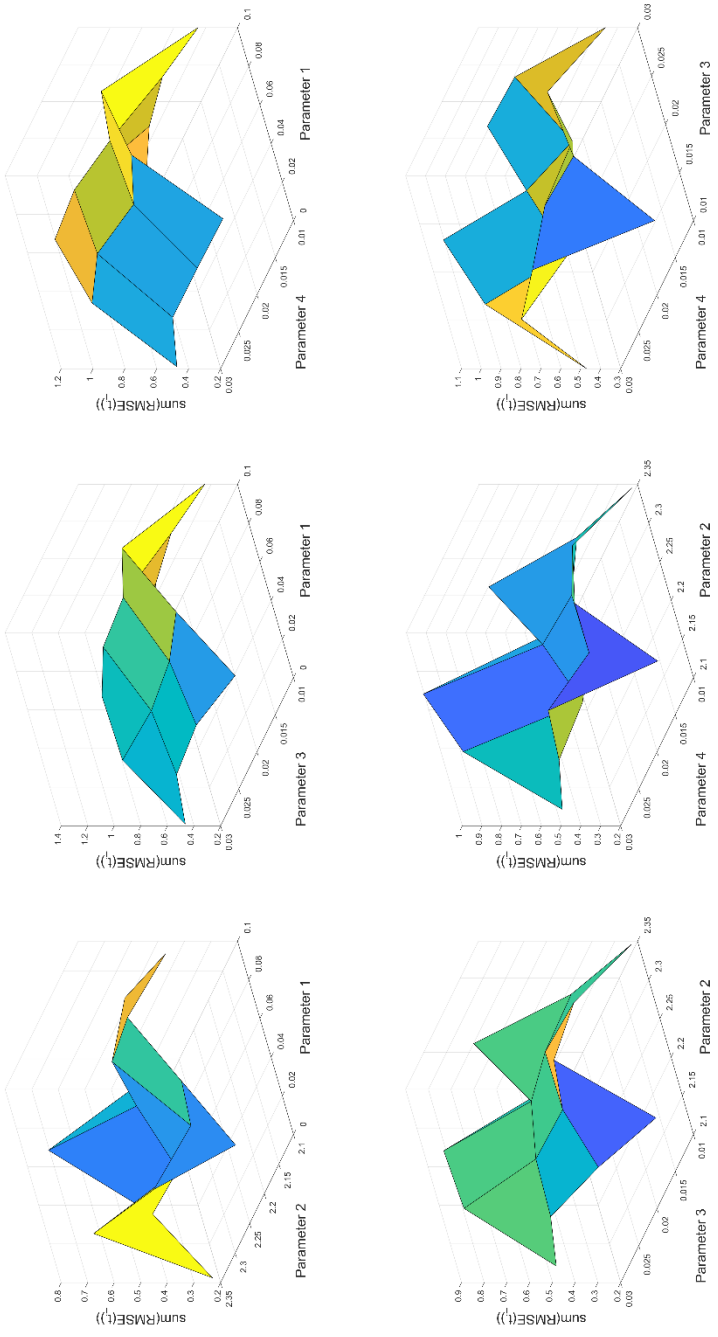


Figure 11. Variation of total RMSE across parametric space calculated with the ANN-ROM built using the adaptive sampling technique

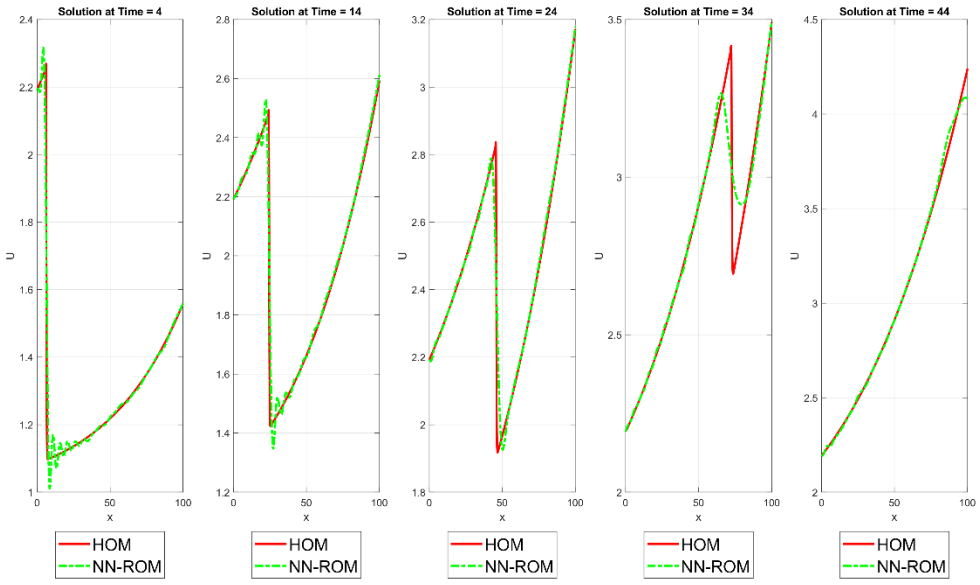
maximum amplitude of the perturbation being limited to 10% of the unperturbed value, and in the second to 30%.

The domains in which parameters are defined, the size of the POD basis and the configuration of the ANNs remain as defined for (29). The total number of parameter sample points is kept to 81, with 65 points added using the adaptive sampling procedure. The results obtained with the ANN-ROM are compared in Figure 12 against the unperturbed HOM (29) for $D = 0.05$, $B = \sqrt{4.8}$, $C_1 = 0.022$ and $C_2 = 0.019$, at the same five instances as shown previously.

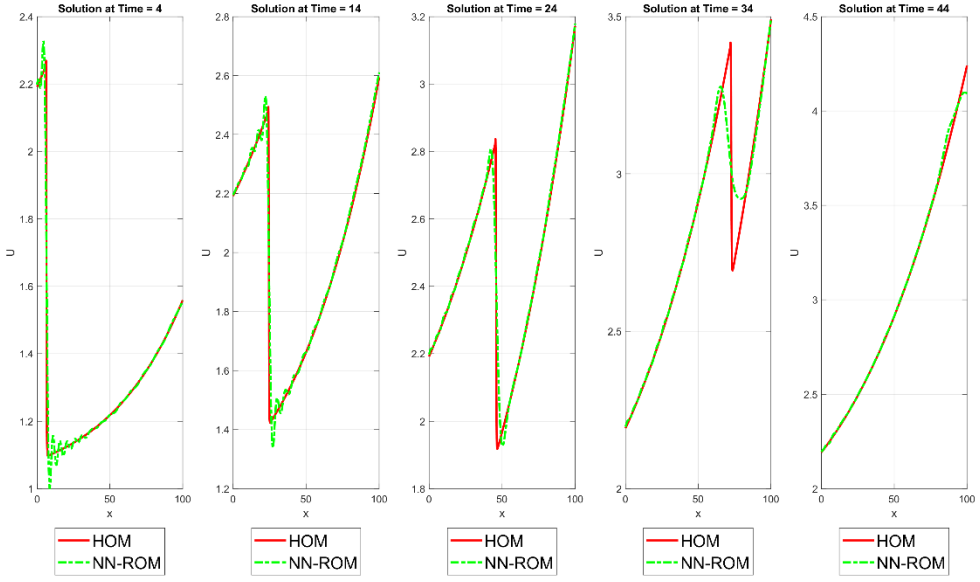
Figure 12 (a) shows a comparison between the unperturbed HOM solution and the unperturbed ANN-ROM, while (b) and (c) show the comparison for the ANN-ROM generated with perturbations of the terms $\delta B(t)$ and $\delta S(t)$. The differences in the ANN-ROM approximation are unnoticeable for the 10% perturbation amplitude scenario. For the 30% perturbation amplitude scenario, there is a noticeable offset to the HOM solution, but it remains small for all time instances considered. This shows the ANN-ROM has a relatively robust behaviour with respect to perturbations to the parameter values in the ROM generation stage. Of course, a much more in-depth analysis can be made, including perturbations and uncertainties in the HOM solution or mathematical model itself, not only in the parameter values, but this lies outside the scope of this paper.

6. Conclusions

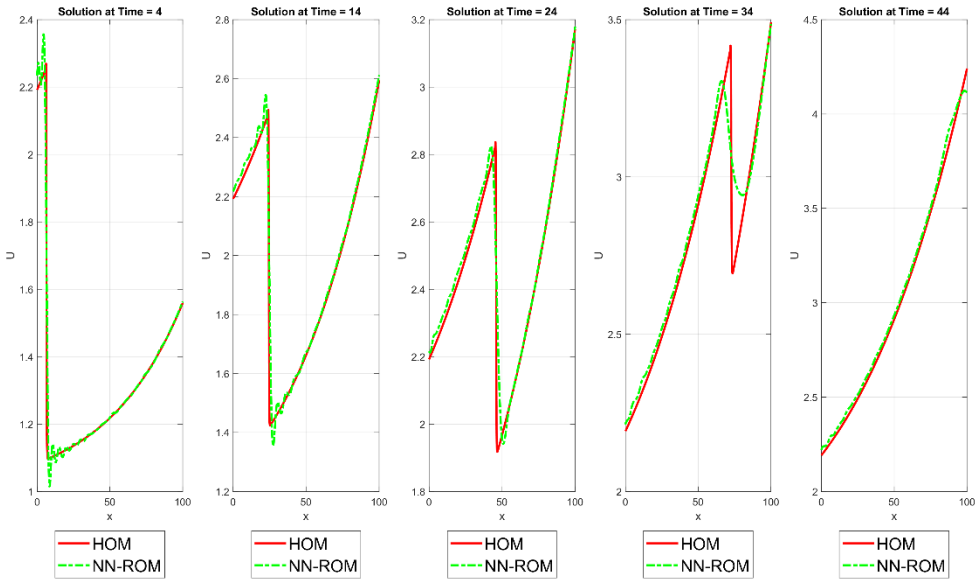
In this work, a non-intrusive reduced-order model is proposed, based on proper orthogonal decomposition and multi-layered feedforward artificial neural networks. The method is suitable for general, nonlinear, time-dependent and parametric problems, and works with black-box high-order solvers. Numerical studies are performed for several test problems. For the geometrically-parametrised steady-state flow through a nozzle, the ANN-ROM achieves speed-up of two order of magnitudes compared with intrusive HROM models, in addition to a



(a) ANN-ROM generated without perturbation



(b) ANN-ROM generated with perturbations limited at an amplitude of 10% of the unperturbed values



(c) ANN-ROM generated with perturbations limited at an amplitude of 30% of the unperturbed values

Figure 12. Solution of viscous Burgers equation with source term as obtained with HOM and ANN-ROM for various perturbation maximum amplitudes

reduction in the time required for the offline stage. Overall accuracy is not as good as the intrusive HROM but can be improved using adaptive sampling techniques aimed at improving the quality of the POD basis and/or the ANN approximation. For non-parametric time-dependent problems, the ANN-ROM achieves accuracy on-par with intrusive ROMs but with considerable speed-up and ease of implementation. The ANN-ROM is extended to parametric unsteady nonlinear problems, where limited work has been published in literature.

The viscous Burgers equation is used a test problem. When the parameter space is low-dimensional, the ANN-ROM achieves very good accuracy with a limited number of a priori samples. Problems with a high-dimensional parametric space require considerably more sampling points to achieve reasonable accuracy but suffer from the so-called curse of dimensionality. The adaptive sampling technique is extended and shown to significantly

improve the quality of the ANN approximation even when the number of sampling points is kept low.

The non-intrusive ANN-ROM shows great potential for modelling complex parametric unsteady phenomena, achieving good accuracy with execution times low enough to allow for online analysis and design. Future work includes applications to both incompressible and compressible Navier-Stokes equations, as well as ROM-based optimisation scenarios.

Declaration of Interest

Nothing to declare.

Funding Sources

This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

References

[1] Guenot, M, Lepot, I, Sainvitu, S, Goblet, J and Coelho, RF. Adaptive sampling strategies for non-intrusive POD-based surrogates. International Journal for Computer-Aided Engineering and Software, Vol. 30, No. 4, 2013, pp. 521-547.

[2] Breitkopf, P and Coelho, RF. Multidisciplinary Design Optimization in Computational Mechanics. Wiley, New York, NY, 2010.

[3] Pearson, K. On lines and planes of closest fit to systems of points in space. Philosophical Magazine, Vol. 2, No. 11, 1901, pp. 559-572.

[4] Sirovich, L. Turbulence and the dynamics of coherent structures. Quarterly of Applied Mathematics, Vol. 45, No. 3, 1987, pp. 561-571.

[5] Fang, F, Zhang, T, Pavlidis, D, Pain, C, Buchan, A and Navon, IM. Reduced order modelling of an unstructured mesh air pollution model and application in 2D/3D urban street canyons. Atmospheric Environment, Vol. 96, 2014, pp. 96-106.

[6] Buchan, A, Calloo, A, Goffin, M, Dargaville, S, Fang, F, Pain, C and Navon, IM. A POD reduced order model for resolving angular direction in neutron/photon transport problems. Journal of Computational Physics, Vol. 296, 2015, pp. 138-157.

[7] Cao, Y, Navon, IM and Luo, Z. A reduced order approach to four-dimensional variational data assimilation using proper orthogonal decomposition. International Journal for Numerical Methods in Fluids, Vol. 53, 2007, pp. 1571-1583.

[8] Alotaibi, M, Calo, VM, Efendiev, Y, Galvis, J and Ghommem, M. Global local nonlinear model reduction for flows in heterogeneous porous media. Computer Methods in Applied Mechanics and Engineering, Vol. 292, 2015, pp. 122-137. Special Issue on Advances in Simulations of Subsurface Flow and Transport (Honouring Professor Mary F. Wheeler).

[9] Hoang, K, Fu, Y and Song, J. An hp-proper orthogonal decomposition moving least squares approach for molecular dynamics simulation. Computer Methods in Applied Mechanics and Engineering, Vol. 298, 2016, pp. 548-575.

[10] Giere, S, Iliescu, T, John, V and Wells, D. SUPG reduced order models for convection-dominated convection diffusion reaction equations. Computer Methods in Applied Mechanics and Engineering, Vol. 289, 2015, pp. 454-474.

[11] Walton, S, Hassan, O and Morgan, K. Reduced order modelling for unsteady fluid flow using proper orthogonal decomposition and radial basis functions. Applied Mathematical Modelling, Vol. 37, No. 20, 2013, pp. 8930-8945.

[12] Schlegel, M and Noack, BR. On long-term boundedness of Galerkin models. Journal of Fluid Mechanics, Vol. 765, 2015, pp. 325-352.

[13] Osth, J, Noack, BR, Krajnovi, S, Barros, D and Bore, J. On the need for a nonlinear subscale turbulence term in POD models as exemplified for a high-Reynolds-number flow over an Ahmed body. Journal of Fluid Mechanics, Vol. 747, 2014, pp. 518-544.

[14] Carlberg, K and Farhat C. Efficient non-linear model reduction via a least-squares Petrov-Galerkin projection and compressive tensor approximations. International Journal for Numerical Methods in Engineering, Vol. 86, 2011, pp. 155-181.

[15] Chu, MSY and Hahn, J. State-preserving nonlinear model reduction procedure. Chemical Engineering Science, Vol. 66, 2011, pp. 3907-3913.

[16] Chaturantabut, S and Sorensen, D. Nonlinear model reduction via discrete empirical interpolation. SIAM Journal on Scientific Computing, Vol. 32, 2010, pp. 2737-2764.

[17] Barrault, M, Maday, Y, Nguyen, N and Patera, A. An empirical interpolation method: application to efficient reduced-basis discretization of partial differential equations. Comptes Rendus de l'Académie des Sciences Paris, Vol. 339, 2004, pp. 667-672.

[18] Noori, R, Karbassi, A, Mehdizadeh, H, Vesali-Naseh, M and Sabahi, M. A framework development for predicting the longitudinal dispersion coefficient in natural streams using an artificial neural network. Environmental Progress and Sustainable Energy, Vol. 30, No. 3, 2011, pp. 439-449.

[19] Xiao, D, Fang, F, Pain, C and Navon, IM. A parameterized non-intrusive reduced order model and error analysis for general time-dependent nonlinear partial differential equations and its applications. Computer Methods in Applied Mechanics and Engineering, Vol. 317, 2017, pp. 868-889.

[20] Audouze, C, De Vuyst, F and Nair, P. Reduced-order modelling of parameterized PDEs using time–space-parameter principal component analysis. International Journal for Numerical Methods in Engineering, Vol. 80, No. 8, 2009, pp. 1025-1057.

[21] Xiao, D, Fang, F, Buchan, A, Pain, C, Navon, IM and Muggeridge, A. Non-intrusive reduced order modelling of the Navier-Stokes equations. Computer Methods in Applied Mechanics and Engineering, Vol. 293, 2015, pp. 552-541.

[22] Amsallem, D, Zahr, M, Choi, Y and Farhat, C. Design optimization using hyper-reduced-order models. Structural and Multidisciplinary Optimization, Vol. 51, 2015, pp. 919–940.

[23] Audouze, C, De Vuyst, F and Nair, P. Nonintrusive Reduced-Order Modelling of Parametrized Time-Dependent Partial Differential Equations. Numerical Methods for Partial Differential Equations, Vol. 29, No. 5, 2013, pp. 1587-1628.

[24] Noack, BR, Morzynski, M and Tadmor, G. Reduced-order modelling for flow control. Vol. 528, Springer, 2011.

[25] Noori, R, Karbassi, AR, Ashrafi, KH., Ardestani, M and Mehrdadi, N. Development and application of reduced-order neural network model based on proper orthogonal decomposition for BOD5 monitoring: Active and online prediction. Environmental Progress and Sustainable Energy, Vol. 32, No. 1, 2013, pp. 120-127.

[26] Hesthaven, JS and Ubbiali, S. Non-intrusive reduced order modelling of nonlinear problems using neural networks. Journal of Computational Physics, Vol. 363, 2018, pp. 55-78.

[27] San, O and Maulik, R. Neural network closures for nonlinear model order reduction. Advances in Computational Mathematics, Vol. 44, 2018, pp. 1717-1750.

[28] Moosavi, A, Stefanescu, R and Sandu, A. Multivariate predictions of local reduced-order-model errors and dimensions. International Journal for Numerical Methods in Engineering, Vol. 113, No. 3, 2018, pp. 512-533.

[29] Haykin, S. Neural Networks: A Comprehensive Foundation. Prentice Hall, Upper Saddle River, NJ, 2004.

[30] Rosenblatt, F. The perceptron: a probabilistic model for information storage and organization in the brain. Psychological Review, Vol. 65, 1958, pp. 386-408.

[31] Marquardt, DW. An algorithm for least-squares estimation of nonlinear parameters. Journal of the Society for Industrial and Applied Mathematics, Vol. 11, No. 2, 1963, pp. 431-441.

[32] Kani, JN and Elsheikh, AH. DR-RNN: A deep residual recurrent neural network for model reduction. arXiv preprint, 2017, arXiv:1709.00939.

[33] Wan, ZY, Vlachas, P, Koumoutsakos, P and Sapsis, T. Data-assisted reduced-order modelling of extreme events in complex dynamical systems. PloS One, Vol. 13, No. 5, 2018.

[34] San, O, Maulik, R and Ahmed, M. An artificial neural network framework for reduced order modelling of transient flows. Communications in Nonlinear Science and Numerical Simulation, Vol. 77, 2019, pp.271-287.

[35] Renganathan, SA, Maulik, R and Rao, V. Machine learning for nonintrusive model order reduction of the parametric inviscid transonic flow past an airfoil. Physics of Fluids, Vol. 32, No. 4, 2020.

[36] Maulik R, Lusch B and Balaprakash P. Non-autoregressive time-series methods for stable parametric reduced-order models. Physics of Fluids, Vol. 32, No. 8, 2020.

[37] Maulik R, Botsas T, Ramachandra N, Mason LR and Pan I. Latent-space time evolution of non-intrusive reduced-order models using Gaussian process emulation. arXiv preprint, 2020, arXiv:2007.12167.

[38] Trehan S, Carlberg KT and Durlofsky LJ. Error modeling for surrogates of dynamical systems using machine learning. International Journal for Numerical Methods in Engineering, Vol. 112(12), 2017, pp. 1801-1827.

[39] Chen W, Wang Q, Hesthaven JS and Zhang C. Physics-informed machine learning for reduced-order modelling of nonlinear problems. Preprint. 2020.

[40] Xiao D, Heaney CE, Mottet L, Fang F, Lin W, Navon IM, Guo Y, Matar OK, Robins AG and Pain CC. A reduced order model for turbulent flows in the urban environment using machine learning. Building and Environment, Vol. 148, 2019, pp. 323-337.

[41] Swischuk R, Mainini L, Peherstorfer B and Willcox K. Projection-based model reduction: Formulations for physics-based machine learning. Computers & Fluids, Vol. 179, 2019, pp. 704-717.

[42] Lee K and Carlberg KT. Model reduction of dynamical systems on nonlinear manifolds using deep convolutional autoencoders. Journal of Computational Physics, Vol. 404, 2020, 404:108973.

[43] Regazzoni, F, Dede, L and Quarteroni, A. Machine learning for fast and reliable solution of time-dependent differential equations. Journal of Computational Physics, Vol. 397, 2019, 397: 108852.

[44] Dupuis R, Jouhaud JC and Sagaut P. Surrogate modelling of aerodynamic simulations for multiple operating conditions using machine learning. AIAA Journal, Vol. 56(9), 2018, pp. 3622-3635.

[45] Kim, B, Azevedo, VC, Thuerey, N, Kim, T, Gross, M and Solenthaler, B. Deep fluids: A generative network for parameterized fluid simulations. In: Computer Graphics Forum, volume 38, 2019, pp. 59–70, Wiley Online Library.

[46] Thuerey, N, Weißenow, K, Prantl, L and Hu, X. Deep learning methods for Reynolds-Averaged Navier–Stokes simulations of airfoil flows. AIAA Journal, Vol 58(1), 2020, pp. 25-36.