Mathematical Biosciences and Engineering

https://www.aimspress.com/journal/mbe

MBE, 22(7): 1825–1860.
DOI: 10.3934/mbe.2025066
Received: 16 April 2025
Revised: 22 May 2025
Accepted: 27 May 2025
Published: 05 June 2025

Research article

Paramatrized intrusive POD-based reduced-order models applied to advection–diffusion–reaction problems

P. Solán-Fustero[1,*], J. L. Gracia[2], A. Navas-Montilla[1] and Pilar García-Navarro[1]

[1] I3A and Fluid Mechanics Department, University of Zaragoza, Spain
[2] IUMA and Applied Mathematics Department, University of Zaragoza, Spain

* **Correspondence:** Email: psolfus@unizar.es.

Abstract: Parametrized problems involve high computational costs when looking for the proper values of their input parameters and solved with classical schemes. Reduced-order models (ROMs) based on the proper orthogonal decomposition act as alternative numerical schemes that speed up computational times while maintaining the accuracy of the solutions. They can be used to obtain solutions in a less expensive way for different values of the input parameters. The samples that compose the training set determine some computational limits on the solution that can be computed by the ROM. It is highly interesting to study what can be done to overcome these limits. In this article, the possibilities to obtain solutions to parametrized problems are explored and illustrated with several numerical cases using the two–dimensional (2D) advection–diffusion–reaction equation and the 2D wildfire propagation model.

Keywords: reduced-order modeling; POD methods; parametrized problems; computational resources; advection–diffusion–reaction equations

1. Introduction

Many phenomena in nature can be mathematically modeled by means of partial differential equations (PDEs) to approximate the underlying physical principles governing these phenomena. The advection–diffusion–reaction (ADR) equation can describe the spatio-temporal evolution of a physical quantity in a flowing medium, such as water or air [1]. This evolution expresses different physical phenomena: The transport of the physical quantity driven by the advection velocity field of the flowing medium, its diffusion from highly dense areas to less dense areas, and the reaction with other components, represented by source terms. The ADR equation plays an important role in many research fields, such as heat conduction [2], transport of particles [3], chemical reactions [4], wildfire propagation [5], etc.

In most cases, there are no analytical solutions for these PDEs, so they have to be solved by means of numerical methods. The finite volume (FV) method [6, 7] is based on the direct discretization of the integral form of the conservation laws, and this form does not require the fluxes to be continuous. The FV method being closer to the physical flow conservation laws is the reason why it is very useful when solving equations like the ADR equation. In this article, the FV method-based first-order upwind (FOU) scheme is used to discretize the information of the conservation laws by assuming a piecewise constant distribution of the conserved variables within computational cells [8, 9].

The large number and diversity of problems, together with the vast spatial domains considered in realistic scenarios requiring an improvement in computational cost, has led, in recent years, to the development of a wide range of mathematical strategies and tools in the scientific literature to facilitate, improve, and increase the calculation capacity of the classical methods used in the framework of fluid mechanics. Among many others, reduced-order modeling is one of the most popular in the field. It was originally developed as the reduced basis strategy for predicting the nonlinear static response of structures [10]. Intrusive reduced–order models (ROMs) based on proper orthogonal decomposition (POD) [11] are one of the most interesting tools in fluid mechanics [12]. Intrusive ROMs are intended to be alternative numerical schemes to replace the calculations performed by classical schemes, also called full-order models (FOMs), to save computational costs without losing accuracy in the solutions. These ROMs reside in a reduced dimensional space much smaller than the physical space, which is the reason why they are more efficient than FOMs [13].

The correct resolution of PDEs by means of numerical schemes requires a thorough calibration of the parameters on which the mathematical model depends. It is therefore of great interest to try to find alternatives that help in this calibration objective in parametrized problems [14]. In this sense, ROMs can be useful for performing multiple simulations in a less expensive way to map different values of the input parameter. ROMs are data-driven methods, which means that they need a training phase prior to their resolution. Training snapshots impose some computational limits on the parameters that define the problem (such as the final time [15] or the initial condition of the advection velocity) that ROMs cannot exceed when computing their solutions. However, it is highly interesting to study what can be done to overcome these limits by means of ROMs.

The aim of this article was to solve the advection–diffusion–reaction equation by means of parametrized ROMs to obtain solutions with parameter values that do not belong to the training set. There are different strategies available in the literature, such as those based on the reduced-basis methods [16, 17], interpolation in matrix manifolds [18], the shifted POD method [19], the extrapolation technique carried out in [20], the transformed modes used in [21] and proposed in [22, 23], or non-intrusive ROMs such as autoencoders and neural networks [24–26]. The technique used in this article is based on generating a multiple training sample from arbitrary values of the input parameters (training values), as done in [27]. The ROM is then solved for some values of interest of the input parameters (target values).

The novelty of this paper consists of the detailed analysis of the constitution of the training sample, taking into account the particular needs of each parameter studied. To do this, it is necessary to find out what size the training sample should be for the ROM to be sufficiently accurate, whether a minimum number of samples is required or even if only one sample is needed, and to check what the relationship is between the ROM's configuration parameters and the number of training samples. This part of the study is done by applying the parametrized ROM strategy to the two–dimensional (2D) ADR equation, whose advection velocity and the diffusion coefficient have been considered as input parameters, as well as parameters that define the initial condition (IC). The test cases solved are designed to evaluate the prediction of ROMs with all possible input parameters when considering the 2D ADR equation.

Within the framework of climate change, forest fires will become more frequent in the following years, with devastating impacts on society [28]. To minimize their risk, mathematical models of forest fire propagation have been developed to predict the evolution of a fire to adequately plan suppression actions, improve the safety of firefighting brigades, and reduce fire damage. Considering all the conclusions obtained via the 2D ADR equation, the parametrized ROM strategy is extended to the 2D wildfire propagation (WP) model [5, 29, 30], which governs the spatio-temporal evolution of fire in a wildfire and whose resolution in the reduced space is similar to that of the ADR equation.

The remainder of the article is organized as follows. Section 2 introduces the 2D ADR equation and the 2D WP model and their discretization using the FV method. Section 3 outlines the POD-based ROM applied to both equations and the modification of the ROM strategy to approach parametrized problems. Section 4 presents seven numerical cases in which the ROM is solved for different input parameters. Finally, the concluding remarks are given in Section 5.

2. The problem under study and the full-order model

2.1. The 2D advection–diffusion–reaction equation

The 2D advection–diffusion–reaction (ADR) equation is defined as follows:

$$\frac{\partial u}{\partial t} + \mathbf{a} \cdot \nabla u = \nu \nabla^2 u - ru, \tag{2.1}$$

where $u = u(x, y, t)$ is the conserved variable, $\mathbf{a} = \left(a_x, a_y\right)$ is the advection velocity, $\nu \geq 0$ is the diffusion coefficient, and r is the reaction coefficient. To find $u(x, y, t)$ in the domain $(x, y, t) \in [0, L_x] \times \left[0, L_y\right] \times [0, T]$, Equation (2.1) has to be solved, subject to appropiate initial conditions (ICs) and boundary conditions (BCs).

Equation (2.1) can be numerically approximated by means of the FV method [31, 32]. The computational domain is discretized using rectangular volume cells with a uniform width $\Delta x \times \Delta y$, where the position of the center of i-th cell is $\left(x_i, y_j\right)$, with $x_i = i\Delta x$ and $y_j = j\Delta y$, and $i = 1, ..., I_x$ and $j = 1, ..., I_y$, where I_x and I_y are the number of volume cells in the x- and the y-directions, respectively. The subsets J^B and J^I contain a pair of indices (i, j) indicating the cells that belong to the boundary and to the interior of the domain, respectively, with $J = J^B \cup J^I$, where $J = \left\{(i, j), 1 \leq i \leq I_x, 1 \leq j \leq I_y\right\}$ is the total set of indices.

Regarding the time discretization, the time step $\Delta t = t^{n+1} - t^n$ with $n = 0, ..., N_T - 1$, where N_T is the total number of time steps, is selected dynamically using the Courant–Friedrichs–Lewy (CFL) condition [33]

$$\Delta t = \frac{\text{CFL}}{\dfrac{|a_x|}{\Delta x} + \dfrac{|a_y|}{\Delta y} + 2\nu\left(\dfrac{1}{\Delta x^2} + \dfrac{1}{\Delta y^2}\right)}, \tag{2.2}$$

with $0 < \text{CFL} \leq 1$.

By means of the FV method, the discretised variables are integrated inside the volume cell $(x_{i-1/2}, x_{i+1/2}) \times \left(y_{j-1/2}, y_{j+1/2}\right)$ with $(i, j) \in J$ in the time interval $\left(t^n, t^{n+1}\right)$. The numerical fluxes are reconstructed using the first-order upwind method [34, 35], and the diffusion term is discretized using central differences, so that the final full-order model (FOM) of the 2D ADR equation is

$$\begin{aligned} u_{i,j}^{n+1} = u_{i,j}^{n} &- \frac{1}{2} a_x \frac{\Delta t}{\Delta x} \left(u_{i+1,j}^{n} - u_{i-1,j}^{n} \right) + \frac{1}{2} |a_x| \frac{\Delta t}{\Delta x} \left(u_{i+1,j}^{n} - 2u_{i,j}^{n} + u_{i-1,j}^{n} \right) \\ &- \frac{1}{2} a_y \frac{\Delta t}{\Delta y} \left(u_{i,j+1}^{n} - u_{i,j-1}^{n} \right) + \frac{1}{2} |a_y| \frac{\Delta t}{\Delta y} \left(u_{i,j+1}^{n} - 2u_{i,j}^{n} + u_{i,j-1}^{n} \right) \\ &+ \frac{\Delta t}{\Delta x^2} \nu \left(u_{i+1,j}^{n} - 2u_{i,j}^{n} + u_{i-1,j}^{n} \right) + \frac{\Delta t}{\Delta y^2} \nu \left(u_{i,j+1}^{n} - 2u_{i,j}^{n} + u_{i,j-1}^{n} \right) - r u_{i,j}^{n}, \end{aligned} \tag{2.3}$$

where $u_{i,j}^n \approx u(x_i, y_j, t^n)$ is the average value over the cell $(x_{i-1/2}, x_{i+1/2}) \times (y_{i-1/2}, y_{i+1/2})$ at t^n, with $(i, j) \in J^I$.

All the variables of interest belonging to the volume cells inside the domain, $(i, j) \in J^I$ are integrated in time, following the same updating relation indicated in (2.3). However, those on the boundary, $(i, j) \in J^B$, require a special treatment that takes into account the boundary conditions imposed on them. Two examples of free boundary conditions are given below for a point placed at the corner $(0, 0)$ and for a point placed at the boundary $y = 0$ of the rectangular domain, respectively

$$\begin{aligned} u_{1,1}^{n+1} = u_{1,1}^{n} &- \frac{1}{2} a_x \frac{\Delta t}{\Delta x} \left(u_{2,1}^{n} - u_{1,1}^{n} \right) + \frac{1}{2} |a_x| \frac{\Delta t}{\Delta x} \left(u_{2,1}^{n} - u_{1,1}^{n} \right) \\ &- \frac{1}{2} a_y \frac{\Delta t}{\Delta y} \left(u_{1,2}^{n} - u_{1,1}^{n} \right) + \frac{1}{2} |a_y| \frac{\Delta t}{\Delta y} \left(u_{1,2}^{n} - u_{1,1}^{n} \right) \\ &+ \nu_x \frac{\Delta t}{\Delta x^2} \left(u_{2,1}^{n} - u_{1,1}^{n} \right) + \nu_y \frac{\Delta t}{\Delta y^2} \left(u_{1,2}^{n} - u_{1,1}^{n} \right) - r u_{1,1}^{n}, \\ u_{i,1}^{n+1} = u_{i,1}^{n} &- \frac{1}{2} a_x \frac{\Delta t}{\Delta x} \left(u_{i+1,1}^{n} - u_{i-1,1}^{n} \right) + \frac{1}{2} |a_x| \frac{\Delta t}{\Delta x} \left(u_{i+1,1}^{n} - 2u_{i,1}^{n} + u_{i-1,1}^{n} \right) \\ &- \frac{1}{2} a_y \frac{\Delta t}{\Delta y} \left(u_{i,2}^{n} - u_{i,1}^{n} \right) + \frac{1}{2} |a_y| \frac{\Delta t}{\Delta y} \left(u_{i,2}^{n} - u_{i,1}^{n} \right) \\ &+ \nu_x \frac{\Delta t}{\Delta x^2} \left(u_{i+1,1}^{n} - 2u_{i,1}^{n} + u_{i-1,1}^{n} \right) + \nu_y \frac{\Delta t}{\Delta y^2} \left(u_{i,2}^{n} - u_{i,1}^{n} \right) - r u_{i,1}^{n}, \end{aligned} \tag{2.4}$$

with $2 \leq i \leq I_x - 1$.

2.2. The 2D wildfire propagation model

The wildfire propagation (WP) model is an application case of the ADR equation. Physics-based wildfire propagation models [5, 29, 30] are usually defined in two horizontal spatial dimensions, as an approach to predict the evolution of the fire on the Earth's surface and require the spatial distribution and characteristics of the biomass [28] as input parameters.

In the WP model, the energy within the fire is represented by the temperature $T = T(x, y, t) > 0$. According to this model, the two-dimensional finite layer of the ground is composed of fuel, represented by the presence of biomass $Y = Y(x, y, t) \in [0, 1]$, which sustains the evolution of the fire. The WP model, as proposed in [28], is

$$\frac{\partial T}{\partial t} + \mathbf{v} \cdot \nabla T = \left(\frac{k}{\rho c_p}\right) \nabla^2 T - \left(\frac{\alpha}{\rho c_p}\right)(T - T_\infty) + \Psi(T)\frac{H}{c_p} Y, \tag{2.5}$$

$$\frac{\partial Y}{\partial t} = -\Psi(T) Y, \tag{2.6}$$

where the rate of variation of the mass fraction of fuel is given by the Arrhenius law [36] as follows:

$$\Psi(T) = \begin{cases} 0, & \text{if } T < T_{pc}, \\ A e^{-\frac{T_{pc}}{T}}, & \text{if } T \geq T_{pc}. \end{cases} \tag{2.7}$$

The heat is generated by the burning reaction of the available fuel in the coupling term, it is transported by the advection term and diffused by the diffusion term, and it is lost to the atmosphere through the cooling (reaction) term.

The parameters of this model are $\mathbf{v} = \left(v_x, v_y\right)$ is the advection velocity, which is constant; k is the thermal conductivity; ρ is the density of the medium; c_p is the specific heat; T_∞ is the ambient temperature; H is the fuel combustion heat; A is the pre-exponential factor; and T_{pc} is the ignition temperature. It should be mentioned that all the physical variables that appear in the WP model are in units of the International System. They have been omitted in all cases for the sake of simplicity.

Following the FOU-based FV method, the variables are integrated into each of the volume cells, and the WP model is discretized as follows [37]:

$$\begin{aligned} T_{i,j}^{n+1} &= T_{i,j}^n - \frac{1}{2} v_x \frac{\Delta t}{\Delta x}\left(T_{i+1,j}^n - T_{i-1,j}^n\right) + \frac{1}{2}\left|v_x\right| \frac{\Delta t}{\Delta x}\left(T_{i+1,j}^n - 2T_{i,j}^n + T_{i-1,j}^n\right) \\ &\quad - \frac{1}{2} v_y \frac{\Delta t}{\Delta y}\left(T_{i,j+1}^n - T_{i,j-1}^n\right) + \frac{1}{2}\left|v_y\right| \frac{\Delta t}{\Delta y}\left(T_{i,j+1}^n - 2T_{i,j}^n + T_{i,j-1}^n\right) \\ &\quad + \left(\frac{k}{\rho c_p}\right) \frac{\Delta t}{\Delta x^2}\left(T_{i+1,j}^n - 2T_{i,j}^n + T_{i-1,j}^n\right) + \left(\frac{k}{\rho c_p}\right) \frac{\Delta t}{\Delta y^2}\left(T_{i,j+1}^n - 2T_{i,j}^n + T_{i,j-1}^n\right) \\ &\quad - \Delta t\left(\frac{\alpha}{\rho c_p}\right)\left(T_{i,j}^n - T_\infty\right) + \Delta t \frac{H}{c_p} Y_{i,j}^n \Psi(T_{i,j}^n) \\ Y_{i,j}^{n+1} &= Y_{i,j}^n - \Delta t Y_{i,j}^n \Psi(T_{i,j}^n), \end{aligned} \tag{2.8}$$

where $T_{ij}^n \approx T(x_i, y_j, t^n)$ and $Y_{ij}^n \approx Y(x_i, y_j, t^n)$ are the average values over the cell $(x_{i-1/2}, x_{i+1/2}) \times (y_{j-1/2}, y_{j+1/2})$ at t^n, with $(i, j) \in J^I$. The discretization of all the boundary conditions are similar to the linear case in (2.4). The time step is selected dynamically using the CFL condition in (2.2).

3. Reduced-order model

In this section, the parametrized training methodology used to predict solutions with the ROM for the values of the parameters of study that were different from those of the training set is presented, together with the standard strategy and the ROMs based on the 2D ADR equation (2.1) and the 2D WP model (2.5).

3.1. Standard ROM strategy

The reduced-order modeling strategy consists of two phases: (1) offline phase, in which the ROM is trained, and (2) online phase, in which the ROM is solved. This resolution in time justifies its use, since it accelerates the required computation time with respect to that of the FOM by several orders of magnitude.

A set of N_T-time numerical solutions computed by the FOM, or *training solutions*, are assembled in the so-called *snapshot matrix*

$$\mathbf{U} = \begin{pmatrix} U^1 & U^2 & \cdots & U^{I_y} \end{pmatrix}^T, \tag{3.1}$$

with

$$U^j = \begin{pmatrix} u^1_{1,j} & u^2_{1,j} & \cdots & u^{N_T}_{1,j} \\ \vdots & \vdots & \ddots & \vdots \\ u^1_{I_x,j} & u^2_{I_x,j} & \cdots & u^{N_T}_{I_x,j} \end{pmatrix} \in \mathbb{R}^{I_x \times N_T}. \tag{3.2}$$

The *proper orthogonal decomposition* (POD) [38] of $\mathbf{U}$ by means of the *singular value decomposition* (SVD) [39] decomposes the snapshot matrix into orthogonal components, also called *POD modes*

$$\mathbf{U} = \mathbf{\Phi}\mathbf{\Sigma}\mathbf{\Psi}^T,$$

where $\mathbf{\Sigma} \in \mathbb{R}^{I_x I_y \times N_T}$ is a diagonal matrix in which the entries of the main diagonal are the singular values of $\mathbf{U}$ and represent the magnitude of each POD mode; $\mathbf{\Phi} \in \mathbb{R}^{I_x I_y \times I_x I_y}$ and $\mathbf{\Psi} \in \mathbb{R}^{N_T \times N_T}$ are orthogonal matrices. The matrix $\mathbf{\Phi} = \left(\phi_1, \ldots, \phi_{I_x I_y}\right) \in \mathbb{R}^{I_x \times I_y}$ with $\phi_k = (\phi_{1,k}, \ldots, \phi_{I_x I_y,k})^T$ consists of the orthogonal eigenvectors of $\mathbf{U}\mathbf{U}^T$, which are used to define the reduced space.

Let M_{POD} be a positive integer such that $M_{\text{POD}} \ll \min\left(I_x I_y, N_T\right)$. It will be chosen to be as small as possible without significantly affecting the accuracy of the solution computed via with the reduced-order method [12]. The number of POD modes solved by the ROM in the test cases proposed in this paper has been obtained a posteriori by analyzing the efficiency obtained. In the following sections, Case 4 will illustrate this procedure.

3.2. Parametrized training methodology

The standard strategy used to train the ROM presented in the previous secion needs to be modified to handle the prediction of solutions for different values from those of the input parameters used to train the ROM, called the *parameters of study*, which are denoted by μ_m, with $m = 1, ..., M_{\text{train}}$.

For each solution that is calculated with the FOM for a set of training parameters, also called the *training sample*, a snapshot matrix is generated $\mathbf{U}(\mu_m)$. All the training sub-matrices are assembled together into one common snapshot matrix, also called the *training set*. This can be done by easily placing one sub-matrix after another

$$\mathbf{U} = \begin{pmatrix} \mathbf{U}(\mu_1) & \mathbf{U}(\mu_2) & \ldots & \mathbf{U}(\mu_{M_{\text{train}}}) \end{pmatrix},$$

where M_{train} is the number of snapshot sub-matrices of the training set. The SVD is applied without special treatment to obtain the reduced basis and obtains the basis functions that define the reduced space for a general value of the parameters. Once the reduced space has been defined, the ROM is solved for the new value of the parameters, namely the test value μ_T. In this paper, the benefits and limitations of this methodology are studied.

In order to extend the ROM strategy to parametrized nonlinear problems, such as the WP model (2.5), it is necessary to combine this modified method with the proper interval decomposition (PID) [20]. For this purpose, it is necessary to group all the snapshots generated in the different training samples according to the defined time windows. The SVD is applied to each time window to obtain the different reduced spaces. For the sake of simplicity, in this paper, both the training set samples and the solution predicted by the ROM at the same time instants will be calculated.

In the following subsections, the ROMs of the 2D ADR equation (2.1) and the 2D WP model (2.5) are presented.

3.3. The 2D ADR-based ROM

Intrusive ROMs based on the POD method are alternative numerical schemes that need to be developed from a standard numerical scheme by projecting it from the physical space to the reduced space. The Galerkin method [40] acts as the projection between these two spaces

$$u^n_{i,j} \approx \sum_{k=1}^{M_{\text{POD}}} \hat{u}^n_k \phi_{i+I_x(j-1),k}, \tag{3.3}$$

where $\hat{u}^n_k$ with $k = 1, ..., M_{\text{POD}}$ is the reduced variable that depend on time, with $(i, j) \in J$.

The 2D ADR-based ROM is developed by applying the Galerkin method (3.3) to the 2D ADR-based FOM (2.3) and projecting it to the reduced space

$$\hat{u}^{n+1}_p = \hat{u}^n_p + \Delta t \sum_{k=1}^{M_{\text{POD}}} \hat{u}^n_k A_{pk}, \tag{3.4}$$

where the coefficients are

$$\begin{aligned}
A_{pk} = &\sum_{(i,j)\in J^B} b_{i,j}\phi_{i+I_x(j-1),p} - \frac{1}{2}\frac{a_x}{\Delta x}\sum_{(i,j)\in J^I}\left(\phi_{i+1+I_x(j-1),j,k} - \phi_{i-1+I_x(j-1),k}\right)\phi_{i,j,p} \\
&+ \frac{1}{2}\frac{|a_x|}{\Delta x}\sum_{(i,j)\in J^I}\left(\phi_{i+1+I_x(j-1),k} - 2\phi_{i+I_x(j-1),k} + \phi_{i-1+I_x(j-1),k}\right)\phi_{i+I_x(j-1),p} \\
&- \frac{1}{2}\frac{a_y}{\Delta y}\sum_{i,j\in J^I}\left(\phi_{i+I_xj,k} - \phi_{i+I_x(j-2),k}\right)\phi_{i+I_x(j-1),p} \\
&+ \frac{1}{2}\frac{|a_y|}{\Delta y}\sum_{(i,j)\in J^I}\left(\phi_{i+I_xj,k} - 2\phi_{i+I_x(j-1),k} + \phi_{i+I_x(j-2),k}\right)\phi_{i+I_x(j-1),p} \\
&+ \frac{\nu}{\Delta x^2}\sum_{(i,j)\in J^I}\left(\phi_{i+1+I_x(j-1),k} - 2\phi_{i+I_x(j-1),k} + \phi_{i-1+I_x(j-1),k}\right)\phi_{i+I_x(j-1),p} \\
&+ \frac{\nu}{\Delta y^2}\sum_{(i,j)\in J^I}\left(\phi_{i+I_xj,k} - 2\phi_{i+I_x(j-1),k} + \phi_{i+I_x(j-2),k}\right)\phi_{i+I_x(j-1),p}
\end{aligned} \tag{3.5}$$

$$- r \sum_{(i,j)\in J} \phi_{i+I_x(j-1),k}\phi_{i+I_x(j-1),p};$$

where the boundary coefficients $b_{i,j}$ are given by the BCs considered. An example of free BCs is

$$b_{1,1} = \left(-\frac{1}{2}\frac{a_x}{\Delta x} + \frac{1}{2}\frac{|a_x|}{\Delta x} + \frac{\nu}{\Delta x^2}\right)(\phi_{2,k} - \phi_{1,k}) + \left(-\frac{1}{2}\frac{a_y}{\Delta y} + \frac{1}{2}\frac{|a_y|}{\Delta y} + \frac{\nu}{\Delta y^2}\right)(\phi_{1+I_x,k} - \phi_{1,k}). \qquad (3.6)$$

3.4. The 2D WP-based ROM

Regarding the WP model, the Galerkin method reduces both variables

$$T'^{n}_{ij} = \frac{T^n_{i,j} - T_\infty}{T_\infty} \approx \sum_{k=1}^{M_{\text{POD}}} \hat{T}^n_k \eta_{i+I_x(j-1),k}, \quad Y^n_{ij} \approx \sum_{k=1}^{M_{\text{POD}}} \hat{Y}^n_k \varphi_{i+I_x(j-1),k}, \quad (i,j) \in J. \qquad (3.7)$$

where the temperature needs to normalized to avoid loss of accuracy in the ROM calculation.

The ROM based on the 2D WP model (2.5) is obtained by applying the Galerkin method (3.7) to the FOM of Eq (2.8)

$$\begin{aligned} \hat{T}^{n+1}_p &= \hat{T}^n_p + \Delta t \sum_{k=1}^{M_{\text{POD}}} \hat{T}^n_k A_{pk} + \Delta t \sum_{k=1}^{M_{\text{POD}}} \hat{Y}^n_k B_{pk}, \\ \hat{Y}^{n+1}_p &= \hat{Y}^n_p + \Delta t \sum_{k=1}^{M_{\text{POD}}} \hat{Y}^n_{qk} C_{pk}, \end{aligned} \qquad (3.8)$$

where the coefficients are

$$\begin{aligned} A_{pk} = &-\frac{1}{2}\frac{v_x}{\Delta x}\sum_{(i,j)\in J^I}\left(\eta_{i+1+I_x(j-1),k} - \eta_{i-1+I_x(j-1),k}\right)\eta_{i+I_x(j-1),p} \\ &+ \frac{1}{2}\frac{|v_x|}{\Delta x}\sum_{(i,j)\in J^I}\left(\eta_{i+1+I_x(j-1),k} - 2\eta_{i+I_x(j-1),k} + \eta_{i-1+I_x(j-1),k}\right)\eta_{i+I_x(j-1),p} \\ &- \frac{1}{2}\frac{v_y}{\Delta y}\sum_{(i,j)\in J^I}\left(\eta_{i+I_xj,k} - \eta_{i+I_x(j-2),k}\right)\eta_{i+I_x(j-1),p} \\ &+ \frac{1}{2}\frac{|v_y|}{\Delta y}\sum_{(i,j)\in J^I}\left(\eta_{i+I_xj,k} - 2\eta_{i+I_x(j-1),k} + \eta_{i+I_x(j-2),k}\right)\eta_{i+I_x(j-1),p} \\ &+ \left(\frac{k}{\rho c_p}\right)\frac{1}{\Delta x^2}\sum_{(i,j)\in J^I}\left(\eta_{i+1+I_x(j-1),k} - 2\eta_{i+I_x(j-1),k} + \eta_{i-1+I_x(j-1),k}\right)\eta_{i+I_x(j-1),p} \\ &+ \left(\frac{k}{\rho c_p}\right)\frac{1}{\Delta y^2}\sum_{(i,j)\in J^I}\left(\eta_{i+I_xj,k} - 2\eta_{i+I_x(j-1),k} + \eta_{i+I_x(j-2),k}\right)\eta_{i+I_x(j-1),p} \\ &+ \sum_{(i,j)\in J^B} b_{i,j}\eta_{i+I_x(j-1),p} - \left(\frac{\alpha}{\rho c_p}\right)\sum_{(i,j)\in J}\eta_{i+I_x(j-1),k}\eta_{i+I_x(j-1),p}, \end{aligned}$$

$$B_{pk} = \frac{H}{T_\infty c_p} \sum_{(i,j)\in J} \Psi(\bar{T}'^{w}_{i,j}) \varphi_{i+I_x(j-1),k} \eta_{i+I_x(j-1),p},$$
$$C_{pk} = -\sum_{(i,j)\in J} \Psi(\bar{T}'^{w}_{i,j}) \varphi_{i+I_x(j-1),k} \varphi_{i+I_x(j-1),p}.$$

The boundary terms $b_{i,j}$ are computed in a similar manner to the linear case (3.6).

The rate variation of the mass fraction (2.7) is nonlinear, so its reduction is not straightforward. In this paper, the reduced version of the mass fraction $\Psi(T^n_{i,j})$ is carried out by defining different time windows to renew the POD basis

$$\Psi(\bar{T}'^{w}_{i,j}) = \begin{cases} 0, & \text{if } \bar{T}'^{w}_{i,j} T_\infty + T_\infty < T_{pc}, \\ Ae^{-\dfrac{T_{pc}}{\bar{T}'^{w}_{i,j} T_\infty + T_\infty}}, & \text{if } \bar{T}'^{w}_{i,j} T_\infty + T_\infty \geq T_{pc}. \end{cases} \tag{3.9}$$

The ROM is linearized inside each time window, because this function is computed in the off-line phase within each time window by using time-averaged temperatures as follows:

$$\bar{T}'^{w}_{i,j} = \frac{1}{M_{\text{STW}} M_{\text{train}}} \sum_{n=1}^{M_{\text{STW}}} \sum_{m=1}^{M_{\text{train}}} \left(T'^{n}_{i,j}\right)^{\text{train}}_{m},$$

where $\left(T'^{n}_{i,j}\right)^{\text{Train}}_{m}$ are the training samples, with M_{train} being the total number of samples computed by the the FOM that comprises the training set, and with M_{STW} being the number of snapshots included in each time window. The number of time windows is selected a posteriori in the numerical cases solved in this paper to improve efficiency.

Note that the reduced equation for updating $\hat{Y}$ does not depend on the values of $\hat{T}$ computed during the online phase and is therefore decoupled from the first equation. In the reduced temperature update equation, the biomass coupling term is precomputed in the coefficients. Because of this, a reduced ADR equation similar to (3.4) is solved at each time level.

4. Numerical results

This section presents the numerical results obtained by applying the parameterized ROM strategy to solve the 2D ADR equation and the 2D WP model.

4.1. Parametrized 2D ADR equation

First, the 2D ADR equation is used to study the limitations of the proposed methodology when the value of parameters such as the transport velocity, diffusion coefficient, initial conditions, or ROM the parameters are modified.

Case 1. Parameter: Diffusion coefficient

The parameter studied in this case is the diffusion coefficient $\mu = \nu$. Since this parameter is scalar, a one-dimensional problem will be worked with for the sake of simplicity. The time-space domain of the case is defined as $(x, y, t) \in [0, 20] \times [0, 1] \times [0, 40]$. The initial condition (IC) is defined as the following Gaussian profile

$$u(x, y, 0) = 1 + e^{-0.2(x-10)^2}, \ \forall y. \tag{4.1}$$

Free boundary conditions are considered. The physical domain is discretized using $I_x \times I_y = 200 \times 1$ volume cells and CFL $= 0.9$. The advection velocity is set to zero, $a_x = a_y = 0$, and the diffusion coefficient is given two different values to generate the training samples. These values, indicated by ν_1 and ν_2, are shown in Table 1, together with the target value ν_T.

Table 1. Case 1. Values of the diffusion coefficient ν for the training samples and the target value.

ν_1	ν_2	ν_T
0.01	0.2	0.1

All the settings of the problem are shown in Table 2.

Table 2. Case 1. Settings.

$L_X \times L_y$	T	CFL	$I_x \times I_y$	a_x	a_y	ν	BC	M_{POD}
20×1	40	0.9	200×1	0	0	Table 1	Free	5

Three different subcases have been solved with the ROM. For each of these, the training set has been constructed with different samples. As indicated in Table 3, in Subcase 1, the ROM has been trained with ν_1; in Subcase 2, with ν_2; and in Subcase 3, with both values ν_1 and ν_2. The ROM has been solved in all subcases using five POD modes, since this number achieves a good compromise between the accuracy of the solution obtained by the ROM and the central processing unit (CPU) time required to compute it. Later, in Case 4, it is analyzed how the solutions are modified if the value of this parameter is changed.

The accuracy obtained by the solutions u^{ROM} calculated by the ROM is measured by means of the differences with respect to the solutions u^{FOM} calculated by the FOM using the L_1 norm

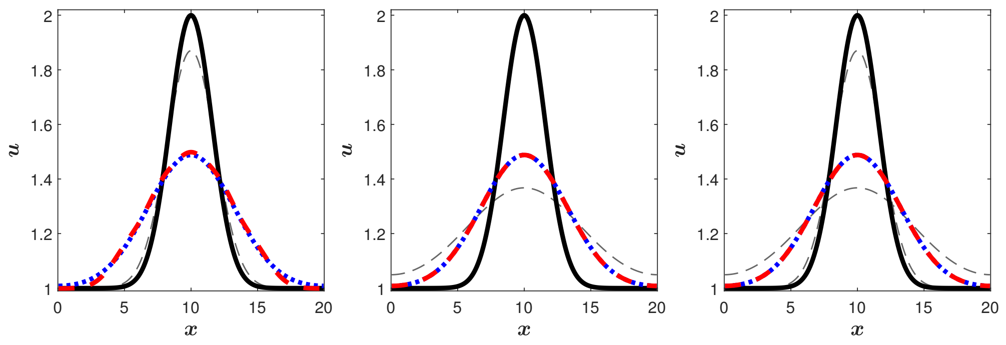
$$||d||_1 = \Delta x \sum_{i,j \in J} |u_{i,j}^{\text{FOM}} - u_{i,j}^{\text{ROM}}|. \tag{4.2}$$

These differences can be computed at each time step, so the time evolution of the errors can be visualized. The differences $||d||_1$ computed in this case are shown in Table 3. These results are shown in Figure 1, where the IC is represented by the black line, the ROM solution with the red line, the test solution with the blue line, and the training solution with the gray line. As can be seen from these representations, all three subcases show high accuracies. This means that the ROM is able to predict solutions for larger values of the diffusion coefficient than the one it has been trained with, as shown in Subcase 1. However, as indicated in Subcase 2, using larger values of this parameter in the training allows to improve the accuracy by three orders of magnitude. Conversely, increasing the training set with more samples does not improve the accuracy, as reported in Subcase 3.

It is also possible to study the *speed up* achieved by the ROM by means of the required CPU time, $\tau_{\text{CPU}}^{\text{ROM}}$, and that of the FOM, $\tau_{\text{CPU}}^{\text{FOM}}$, both measured in seconds. The CPU times required by the FOM to compute the test solution and the by ROM to compute the target solutions are $\tau^{\text{FOM}}_{\text{CPU}} = 1.00 \cdot 10^{-2}$ and $\tau^{\text{ROM}}_{\text{CPU}} = 2.61 \cdot 10^{-4}$, respectively, so the speed up achieved by the ROM is $\times 39$.

Table 3. Case 1. Training set for the different subcases and differences of the ROM solutions with respect to the test solutions.

Subcase	1	2	3
Training set	$\{\nu_1\}$	$\{\nu_2\}$	$\{\nu_1, \nu_2\}$
$\|\|d\|\|_1$	$1.96 \cdot 10^{-1}$	$5.90 \cdot 10^{-4}$	$6.79 \cdot 10^{-4}$



(a) Subcase 1. **(b)** Subcase 2. **(c)** Subcase 3.

Figure 1. Case 1. ROM solutions are shown in red, together with the test solutions in blue and the training solutions in gray at the final time. The IC is represented by the black solid line.

Case 2. Parameter: Advection velocity

The parameter studied in this case is the advection velocity in the x direction $\mu = a_x$ (with $a_y = 0$). In this case, only one component of the velocity is used for the sake of simplicity. Later, the two-dimensional character of the velocity field is studied. The time-space domain of the case is defined as $(x, t) \in [0, 20] \times [0, 10]$. The IC is defined as the following Gaussian profile:

$$u(x, y, 0) = 1 + e^{-0.2(x-6)^2}. \tag{4.3}$$

Free boundary conditions are considered. The physical domain is discretized using $I_x \times I_y = 200 \times 1$ volume cells and CFL = 0.9. The diffusion coefficient is set as $\nu = 0.01$, and the advection velocity in the x direction is given six different values to generate the training samples. These values are shown in Table 4, together with the target value $(a_x)_T$.

Table 4. Case 2. Advection velocity a_x values for the training samples and the target value.

$(a_x)_1$	$(a_x)_2$	$(a_x)_3$	$(a_x)_4$	$(a_x)_5$	$(a_x)_6$	$(a_x)_T$
0.01	0.2	0.4	0.6	0.8	1	0.5

All the settings of the problem are shown in Table 5.

Table 5. Case 2. Settings.

$L_x \times L_y$	T	CFL	$I_x \times I_y$	a_x	a_y	ν	BC	M_{POD}
20×1	10	0.9	200×1	Table 4	0	0.01	Free	5

The ROM has been solved 22 different times, corresponding to the 22 subcases indicated in Figure 2. In each of these subcases, the training set consists of different combinations of the six samples for different values of the parameter of interest a_x listed in Table 5. In the first six subcases, the ROM has been trained with a single sample, choosing a different one in each subcase; while in the rest, at least two samples are used. For example, in Subcase 5, the ROM has been trained using just Sample 5 ($(a_x)_5 = 0.8$); in Subcase 16, the ROM has been trained using Samples 2 and 5 ($(a_x)_2 = 0.2$ and $(a_x)_5 = 0.8$). Through this exploration of the number of samples and which ones in particular, the most appropriate composition of the parametric training set has been studied to predict the target solution with $(a_x)_T = 0.5$. In all subcases, the ROM has been solved using five POD modes.

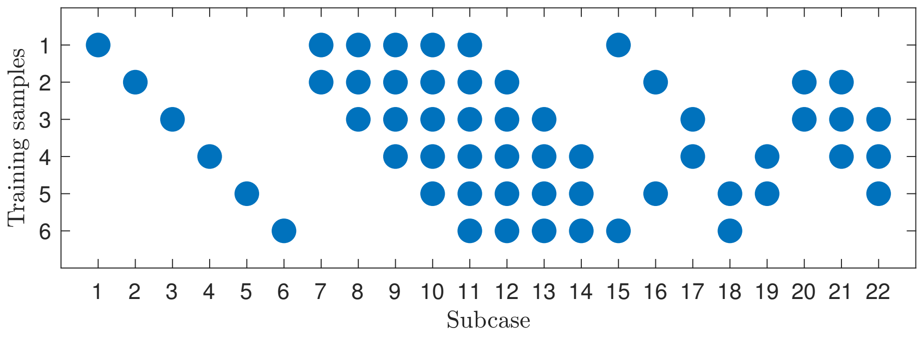


Figure 2. Case 2. Training set.

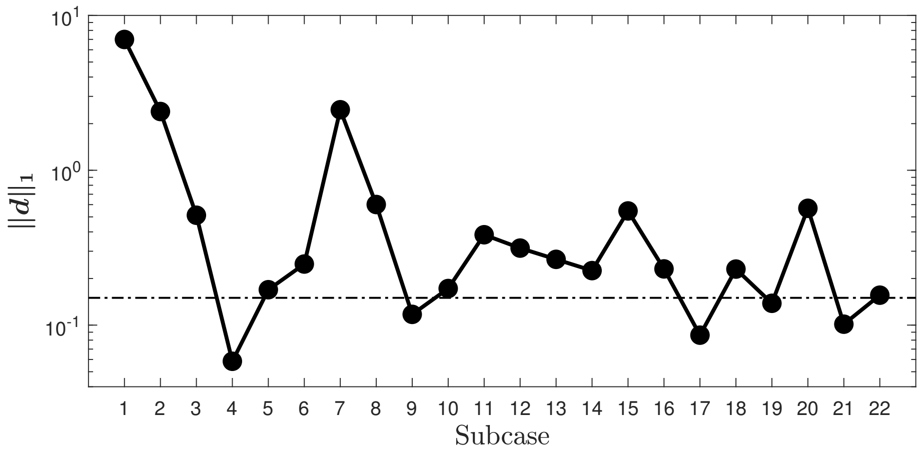
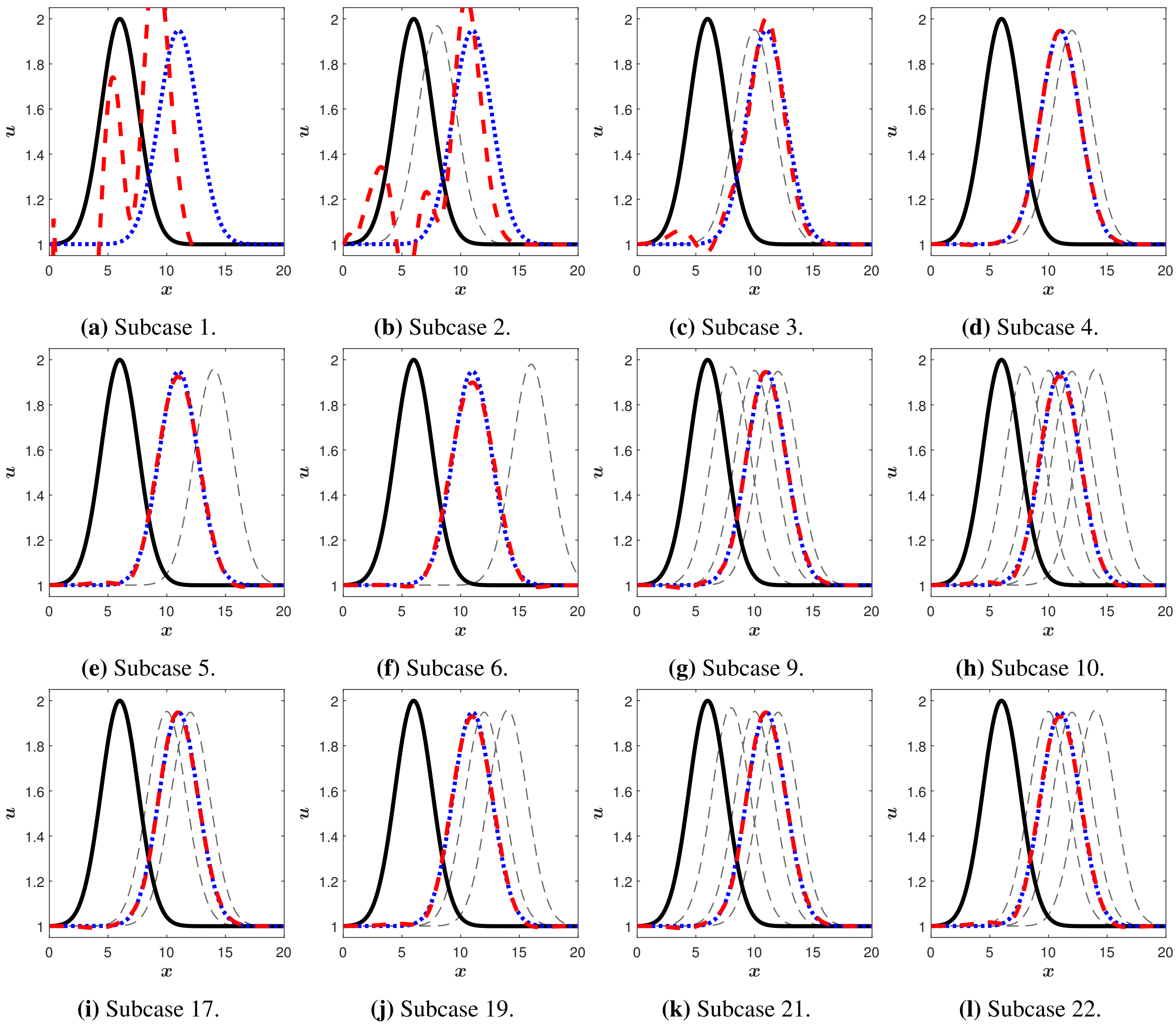


Figure 3. Case 2. Differences $||d||_1$ (black dots) between each subcase and the threshold difference $||d||_1^T$ (horizontal dashed line).

The differences between the ROM and FOM solutions $||d||_1$ are presented in Figure 3. The main conclusion that can be drawn from these results is that it is necessary to train the ROM with a solution obtained with a higher advection velocity than the target. In other words, it is necessary to use a solution that has gone further in space. In addition to this, the accuracy of the solution predicted by the ROM will be higher the closer the training value is to the target. If it gets further away towards higher values, the accuracy drops, although it still works. This is clearly shown in the single-trained Subcases 4–6, as Subcase 4 has the best accuracy, and Subcases 5 and 6 worsen with respect to Subcase 4, but remain below Subcases 1–3. The ROM is able to accurately predict the time evolution of the initial Gaussian profile in Subcases 4–6, as shown in Figure 4d–f, whereas in Subcases 1–3, the ROM changes the shape of the Gaussian profile, as shown in Figure 4a–c.



(a) Subcase 1. **(b)** Subcase 2. **(c)** Subcase 3. **(d)** Subcase 4.
(e) Subcase 5. **(f)** Subcase 6. **(g)** Subcase 9. **(h)** Subcase 10.
(i) Subcase 17. **(j)** Subcase 19. **(k)** Subcase 21. **(l)** Subcase 22.

Figure 4. Case 2. ROM solutions are shown in red, together with the test solutions in blue and the training solutions in gray at the final time. The IC is represented by the black solid line.

In Figure 3, of all the subcases, the one showing the smallest differences is Subcase 4, where the training set is composed of a single sample. However, the training sets are rarely composed of only one solution and, in addition, the possible lack of knowledge of their composition may mean that they are not sufficiently close. Thus, it is necessary to consider sets composed of several samples, as is done in Subcases 7–22. In general, it has been found that the subcases with the smallest differences are those containing Training Sample 4, such as Subcases 9, 10, 17, 19, 21, and 22. In these subcases, the differences obtained are above those of Subcase 4, but they are still good levels of error, as can be seen in the solutions shown in Figure 4g–l. In these subcases, the solutions predicted by the ROM are practically identical to the test solutions. From this, it can be deduced that as long as the differences are below a value of approximately 0.15, the ROM solution will be good. This value of $||d||_1^T = 1.5 \cdot 10^{-1}$ will be used as a threshold for selecting solutions with acceptable accuracy. This value has been included in the figure by means of a horizontal dashed line, according to which, Subcases 4, 9, 17, 19, and 21 would be accepted.

The CPU times required by the FOM and the ROM are $\tau_{\text{CPU}}^{\text{FOM}} = 1.08 \cdot 10^{-3}$ and $\tau_{\text{CPU}}^{\text{ROM}} = 1.93 \cdot 10^{-5}$, respectively, so the speed up is ×56.

Case 3. Parameter: Initial Gaussian profile

This test case is designed to study the ability of the ROM to predict solutions when the coefficients defining the IC act as input parameters. Since a detailed analysis of the elements that define the IC is carried out, in this case a one-dimensional case is used. The time-space domain of the case is defined as $(x, y, t) \in [0, 20] \times [0, 1] \times [0, 10]$. A Gaussian profile is defined as the IC

$$u(x, y, 0) = \bar{u} + u_0 e^{-c(x-x_0)^2}, \tag{4.4}$$

where the four coefficients $\bar{u}$, u_0, c, and x_0 are treated as input parameters. Each of these parameters is studied separately to see how they affect the ROM's predictions. They are given six different values structured around the target value, so that six training samples are considered to build the training set according to the distribution of subcases indicated in Figure 5.

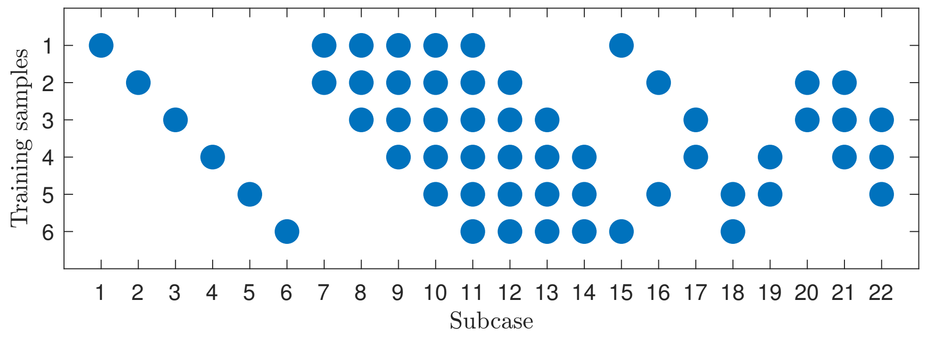


Figure 5. Case 3. Training set.

Free boundary conditions are considered. The physical domain is discretized using $I_x \times I_y = 200 \times 1$ volume cells and CFL = 0.9. The diffusion coefficient is set to $\nu = 0.01$, and the advection velocity is set to $\mathbf{a} = (a_x, a_y) = (0.5, 0)$. All the settings of the problem are kept fixed in all the following different cases and are shown in Table 6.

Table 6. Case 3. Settings.

$L_x \times L_y$	T	CFL	$I_x \times I_y$	a_x	a_y	ν	BC
20×1	10	0.9	200×1	0.5	0	0.01	Free

Parameter of study: $\bar{u}$

First, the offset of the initial Gaussian profile $\bar{u}$ is considered as the parameter of study and given the values indicated in Table 7. The target value is also included.

Table 7. Case 3.1. Training values of $\bar{u}$.

$\bar{u}_1$	$\bar{u}_2$	$\bar{u}_3$	$\bar{u}_4$	$\bar{u}_5$	$\bar{u}_6$	$\bar{u}_T$
0	0.2	0.4	0.6	0.8	1	0.5

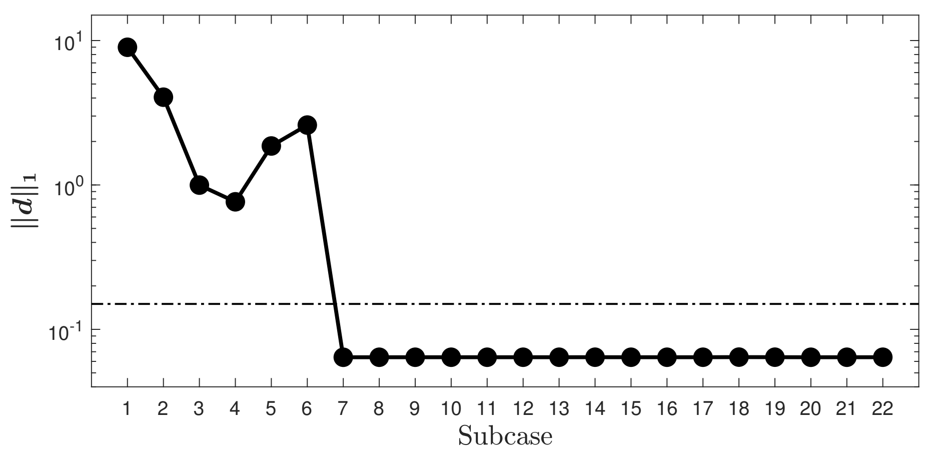
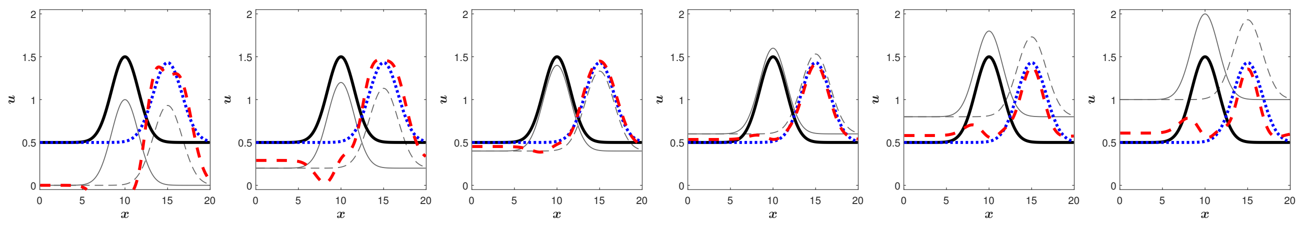
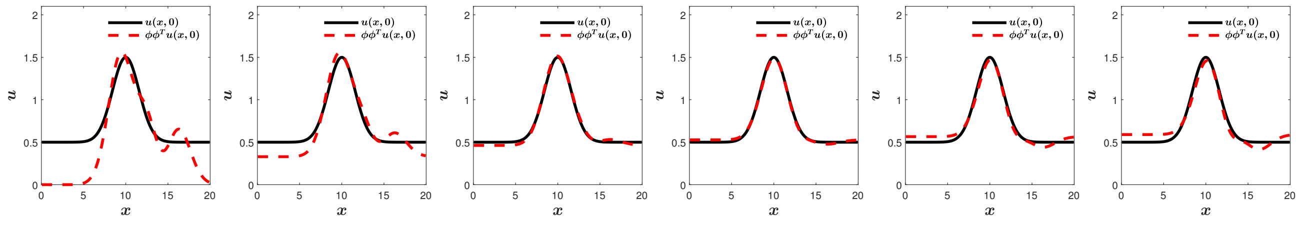


Figure 6. Case 3.1. Differences $||d||_1$ (black dots) between each subcase and the threshold difference $||d||_1^T$ (horizontal dashed line).



(a) Subcase 1. **(b)** Subcase 2. **(c)** Subcase 3. **(d)** Subcase 4. **(e)** Subcase 5. **(f)** Subcase 6.



(g) Subcase 1. **(h)** Subcase 2. **(i)** Subcase 3. **(j)** Subcase 4. **(k)** Subcase 5. **(l)** Subcase 6.

Figure 7. Case 3.1. ROM results and comparison between the IC and its reduced projection.

The rest of the coefficients defining the initial Gaussian profile are given the following values,

$$u_0 = 1, \quad c = 0.2, \quad x_0 = 10.$$

The ROM has been solved using $M_{\text{POD}} = 6$ POD modes in all subcases.

The differences $||d||_1$ of all subcases are shown in Figure 6. In all the single-trained subcases, the ROM obtains very high differences, even in Subcases 3 and 4, where the training values of $\bar{u}$ are the closest to the target value. The solutions computed by the ROM are not able to accurately predict the final Gaussian profile of the test solution, as indicated in Figure 7a–f. This is due to the fact that the initial Gaussian profile, when projected into the reduced space (using a small number of POD modes), is deformed and loses accuracy. Therefore, when it is transported in space, it maintains this deformation. This is illustrated by Figure 7g–l, in which the IC is represented together with a reprojection of space of its projection to the reduced space, i.e., $\phi\phi^T u(x, 0)$.

However, in subcases where the training set is composed of two or more samples, whatever they are, the ROM is able to predict solutions with high accuracy. Thus, Subcases 7–22 show similar differences that are below the acceptance threshold $||d||_1^T$, as shown in Figure 6. In short, the offset of the initial Gaussian profile $\bar{u}$ is a very malleable parameter that simply requires a training set with several samples and can be given arbitrary values.

The CPU times required by the FOM and the ROM are $\tau_{\text{CPU}}^{\text{FOM}} = 9.05 \cdot 10^{-4}$ and $\tau_{\text{CPU}}^{\text{ROM}} = 1.94 \cdot 10^{-5}$, respectively, so the speed up is $\times 47$.

Parameter of study: u_0

Second, the amplitude of the initial Gaussian profile u_0 is considered as the parameter of study and given the values indicated in Table 8.

Table 8. Case 3.2. Training values of u_0.

$(u_0)_1$	$(u_0)_2$	$(u_0)_3$	$(u_0)_4$	$(u_0)_5$	$(u_0)_6$	$(u_0)_T$
0.1	0.28	0.46	0.64	0.82	1	0.5

The rest of the coefficients defining the initial Gaussian profile are given the following values

$$\bar{u} = 1, \quad c = 0.2, \quad x_0 = 6.$$

The ROM has been solved using $M_{\text{POD}} = 5$ POD modes.

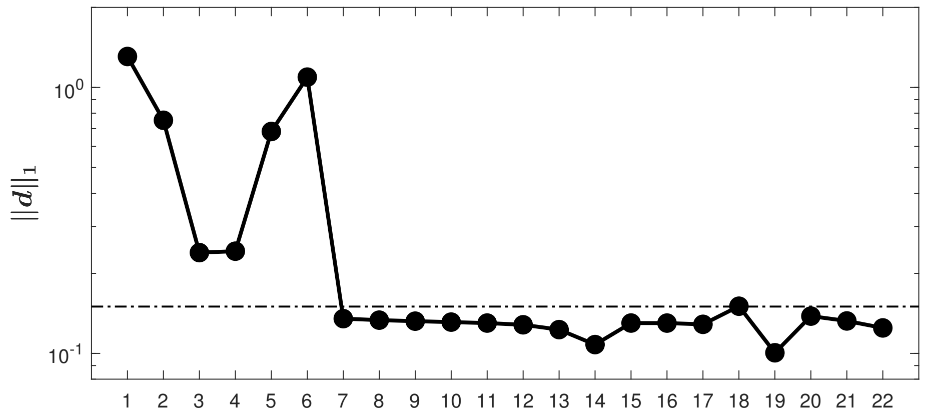


Figure 8. Case 3.2. Differences $||d||_1$ (black dots) between each subcase and the threshold difference $||d||_1^T$ (horizontal dashed line).

Similar to what has been observed with the offset coefficient $\bar{u}$, the ROM needs two or more samples in the training set when predicting solutions for different values of the amplitude of the initial Gaussian profile. This is shown in Figure 8, where Subcases 7–22 present similar differences below the threshold difference $||d||_1^T$.

The CPU times required by the FOM and the ROM are $\tau_{\text{CPU}}^{\text{FOM}} = 8.88 \cdot 10^{-4}$ and $\tau_{\text{CPU}}^{\text{ROM}} = 1.42 \cdot 10^{-5}$, respectively, so the speed up is $\times 63$.

Parameter of study: c

Third, the width of the initial Gaussian profile c is considered as the parameter of study and given the values indicated in Table 9.

Table 9. Case 3.3. Training values of c.

c_1	c_2	c_3	c_4	c_5	c_6	c_T
0.1	0.28	0.46	0.64	0.82	1	0.5

The rest of the coefficients defining the initial Gaussian profile are given the following values:

$$\bar{u} = 1, \quad u_0 = 1, \quad x_0 = 6.$$

The ROM has been solved using $M_{\text{POD}} = 8$ POD modes.

The width of the initial Gaussian profile is a more complicated parameter, since most of the subcases show higher differences than the threshold difference $||d||_1^T$, as shown in Figure 9. Only in Subcases 12, 17, 20, 21, and 22, the ROM is able to predict accurate solutions, as shown in Figure 10. What these subcases have in common is the presence of Sample 3 in their training set. This sample has been computed with the value of c that is the closest below the target value, which is the initial Gaussian profile of width that is the closest above the target. In the subcases in which the training set is composed of several samples (including Sample 3), the differences are also decreased, as indicated in Subcases 20–22. It can therefore be concluded that the ROM needs to be trained with several samples that include wider profiles.

In addition to this, with the previous parameters, only six and five POD modes have been enough to achieve good predictions, but when considering the width of the Gaussian profile, it is necessary to increase the number of POD modes to eight to obtain satisfactory differences.

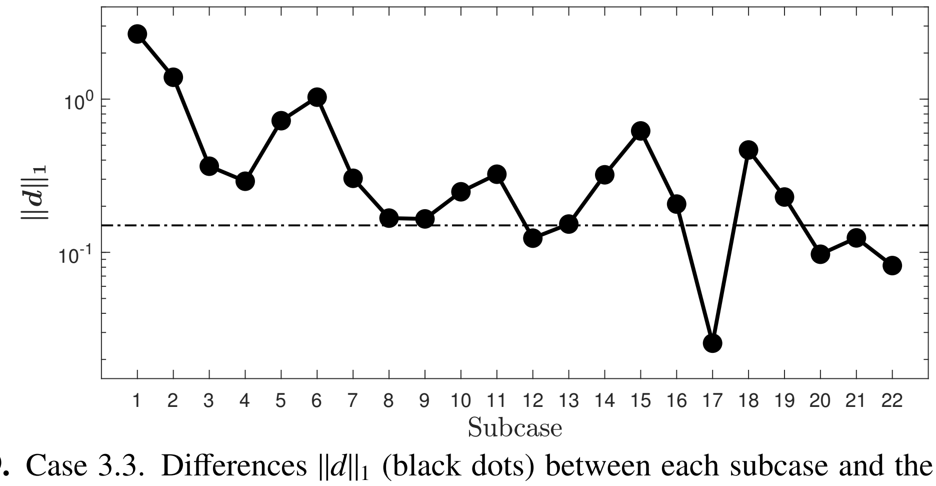
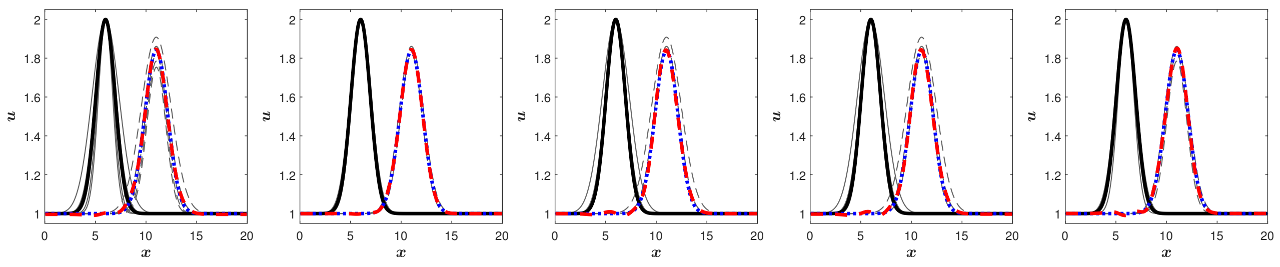
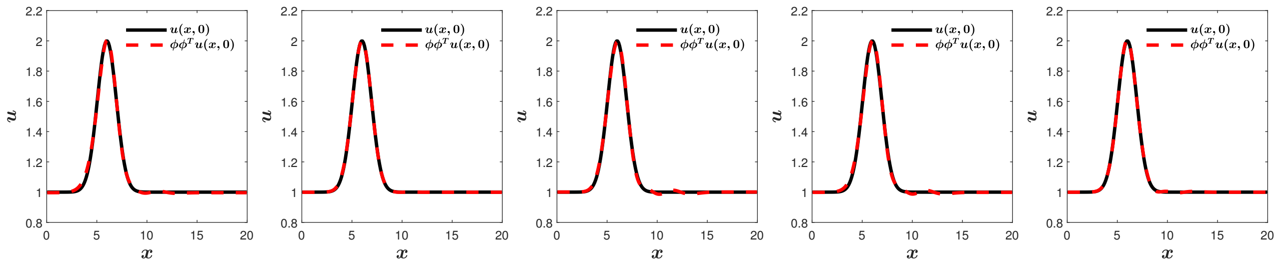


Figure 9. Case 3.3. Differences $||d||_1$ (black dots) between each subcase and the threshold difference $||d||_1^T$ (horizontal dashed line).

The CPU times required by the FOM and the ROM are $\tau_{\text{CPU}}^{\text{FOM}} = 8.86 \cdot 10^{-4}$ and $\tau_{\text{CPU}}^{\text{ROM}} = 3.09 \cdot 10^{-5}$, respectively, so the speed up is $\times 29$.



(a) Subcase 12. **(b)** Subcase 17. **(c)** Subcase 20. **(d)** Subcase 21. **(e)** Subcase 22.



(f) Subcase 12. **(g)** Subcase 17. **(h)** Subcase 20. **(i)** Subcase 21. **(j)** Subcase 22.

Figure 10. Case 3.3. ROM results and comparison between the IC and its reduced projection.

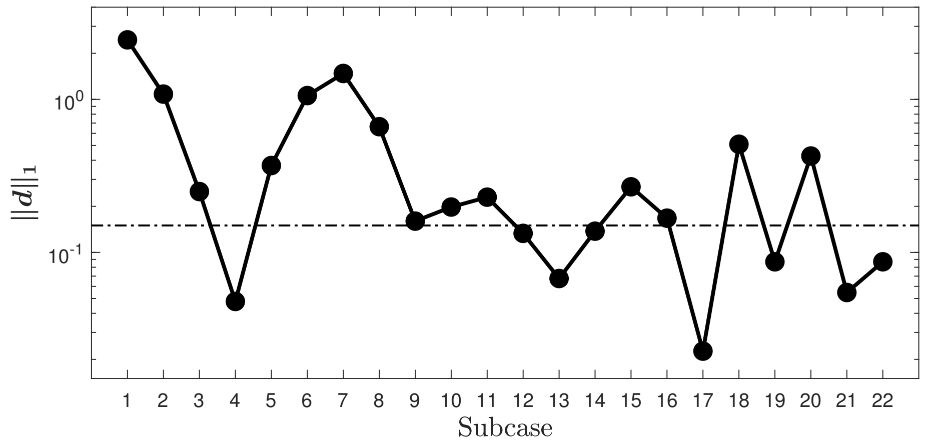


Figure 11. Case 3.4. Differences $||d||_1$ (black dots) between each subcase and the threshold difference $||d||_1^T$ (horizontal dashed line).

Parameter of study: x_0

Lastly, the position of the initial Gaussian profile x_0 is considered as the parameter of study and given the values indicated in Table 10.

Table 10. Case 3.4. Training values of x_0.

$(x_0)_1$	$(x_0)_2$	$(x_0)_3$	$(x_0)_4$	$(x_0)_5$	$(x_0)_6$	$(x_0)_T$
4	4.8	5.6	6.4	7.2	8	6

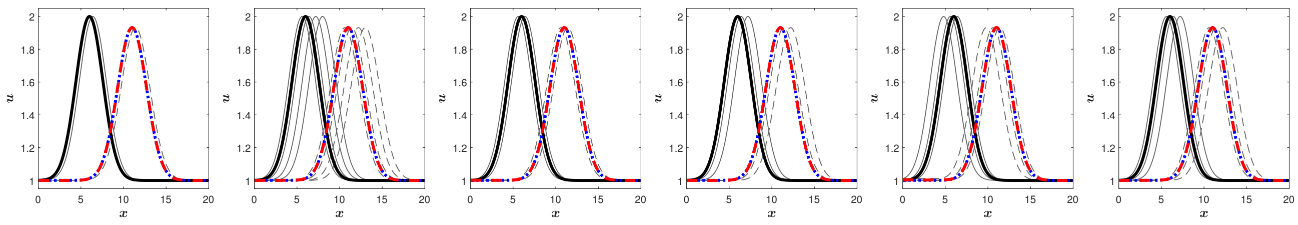
The rest of the coefficients defining the initial Gaussian profile are given the following values

$$\bar{u} = 1, \quad u_0 = 1, \quad c = 0.2.$$

The ROM has been solved using $M_{\text{POD}} = 5$ POD modes.

Subcases 4, 13, 17, 19, 21, and 22 show much smaller differences than the threshold difference $||d||_1^T$. In all these subcases, the training sets include Sample 4, which is computed with the closest value above the target value. The smallest differences are obtained in Subcase 17, where the training set includes just Samples 3 and 4. The differences are increased if more samples are added to the training set, although the ROM continues to obtain good solutions, as shown in Figure 12.

The CPU times required by the FOM and the ROM are $\tau_{\text{CPU}}^{\text{FOM}} = 8.96 \cdot 10^{-4}$ and $\tau_{\text{CPU}}^{\text{ROM}} = 1.44 \cdot 10^{-5}$, respectively, so the speed up is $\times 62$.



(a) Subcase 4. **(b)** Subcase 13. **(c)** Subcase 17. **(d)** Subcase 19. **(e)** Subcase 21. **(f)** Subcase 22.

Figure 12. Case 3.4. ROM results and comparison between the IC and its reduced projection.

Case 4. Parameters: M_{train} and M_{POD}

It is possible that the samples that compose the training set cannot be designed and that the training set is given in advance. In this case, it is necessary to study how to modify the parameters of the ROM which it is possible to access. With this objective, the training set has been built with ten samples in which the coefficients that define the initial Gaussian profile (4.4) have been given random values, as shown in Figure 13.

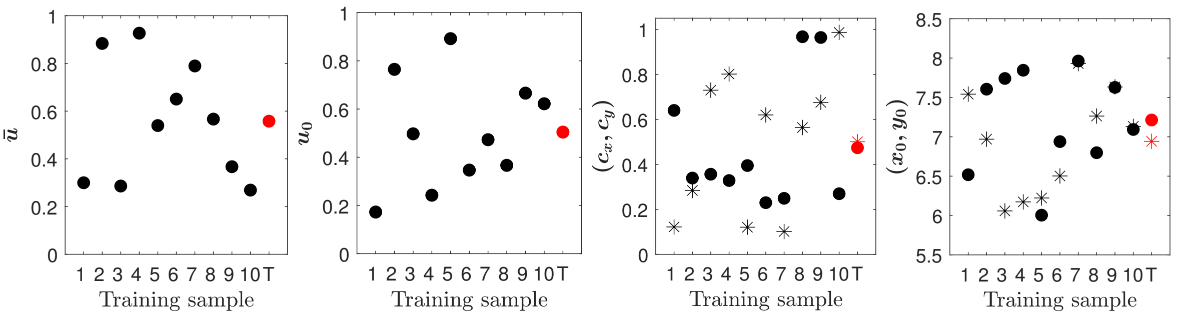
The time-space domain of the case is defined as $(x, y, t) \in [0, 20] \times [0, 20] \times [0, 10]$. A Gaussian profile is defined as the IC

$$u(x, 0) = \bar{u} + u_0 e^{-c_x(x-x_0)^2 - c_y(y-y_0)^2}, \tag{4.5}$$

where the six coefficients $\bar{u}$, u_0, c_x, c_y, x_0, and y_0 are treated as given random values, as shown in Figure 13. Free boundary conditions are considered. The physical domain is discretized using $I_x \times I_y = 100 \times 100$ volume cells and CFL = 0.4. The diffusion coefficient is set as $\nu = 0.01$, and the advection velocity is set as $\mathbf{a} = \left(a_x, a_y\right) = (0.5, 0.5)$. All the settings of the problem are kept fixed in all the following different cases and are shown in Table 11.

Table 11. Case 4. Settings.

$L_x \times L_y$	T	CFL	$I_x \times I_y$	a_x	a_y	ν	IC	BC
20×20	10	0.4	100×100	0.5	0.5	0.01	Eq (4.5)	Free



(a) Offset $\bar{u}$. **(b)** Amplitude u_0. **(c)** Width $\left(c_x, c_y\right)$. **(d)** Position (x_0, y_0).

Figure 13. Case 4. Values of the training set.

The training set has been designed to add samples one by one in increasing order, as follows:

$$M_{\text{train}} \in \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}\,.$$

In addition to this, the ROM has been solved using different values of the number of POD modes

$$M_{\text{POD}} \in \{3, 6, 9, 12, 15, 18, 21, 24, 27, 30\}\,.$$

Therefore, the subcases solved by ROM are the combination of all the values of these two parameters. That is, there are 100 subcases whose training sets are composed as shown in Figure 14. Each time the training set is returned to a training set consisting only of the first sample (every 10 subcases), three POD modes are added.

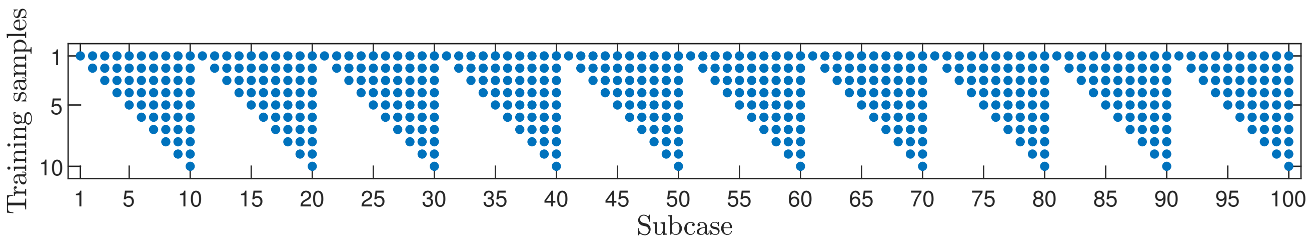


Figure 14. Case 4. Training set.

The differences in all the subcases can be seen in a single row in Figure 15, where the general tendency to decrease as the number of POD modes increases can be seen. In the last four data-sets (from Subcase 61 onwards), differences smaller than the threshold difference $||d||_1$ are reached. In other words, the ROM predicts good solutions whenever it uses at least 18 POD modes. This is best seen in the 2D representation shown in Figure 16a, where the red line indicates the threshold difference. It can be seen that just one training sample is never enough, since the error is always very big. However, at least four samples are needed in the training set when they have arbitrary values. From four samples onwards, and with a high number of POD modes, the differences remain at similar values (below the threshold error), for which the predictions are accurate.

As the number of POD modes used by the ROM increases, the computation times increase and, consequently, the speed up decreases, as shown in Figure 16b. Thus, it is desirable to use as few POD modes as possible to make the ROM faster than the FOM, but without losing accuracy in the predictions. Taking this into account, the optimal value of POD modes is around 18 POD modes, since it achieves three orders of magnitude of speed up and the accuracy of the solution is guaranteed.

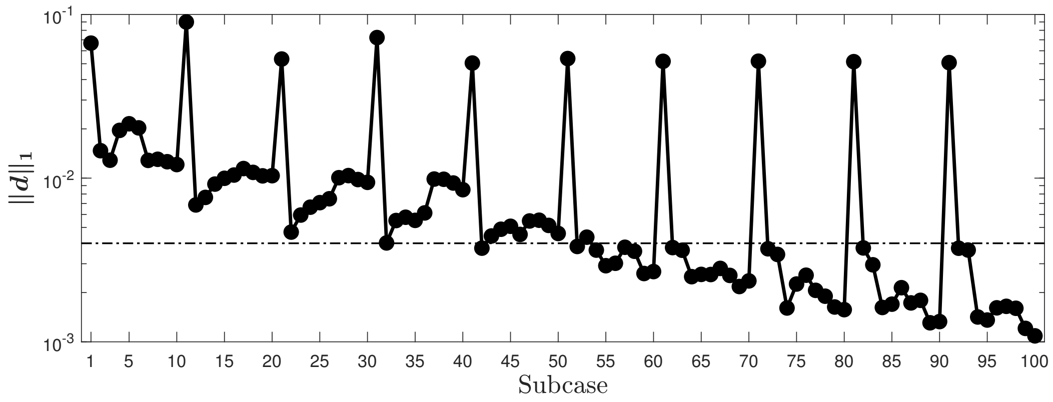
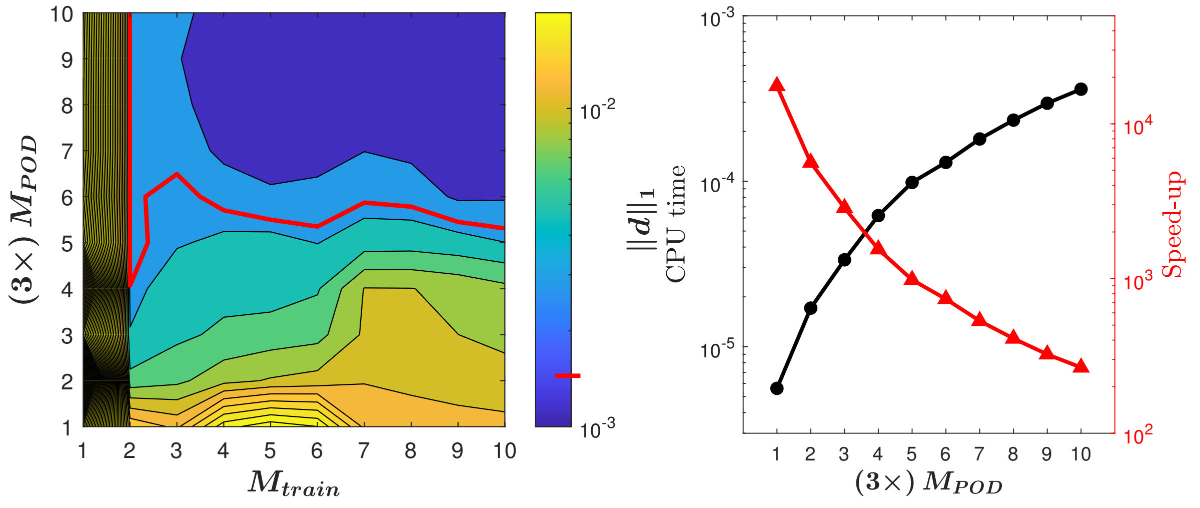


Figure 15. Case 4. Differences $\|d\|_1$ (black dots) between each subcase and the threshold difference $\|d\|_1^T$ (horizontal dashed line).



(a) Differences $\|d\|_1$ and threshold difference $\|d\|_1^T$ (red line). **(b)** CPU times and speed ups.

Figure 16. Case 4. Differences (left) and CPU times (right).

Case 5. Parameter: velocity

In Case 2, it was concluded that it is necessary for the training set to contain a sample with a higher advection velocity than the target for the ROM to predict accurate solutions. In the case of a two-dimensional velocity field, this conclusion could be extended to cases where the target solution is transported in the same direction as the training samples (and its magnitude may vary). That is, the ratio a_x/a_y is kept constant. However, it is highly interesting to study what happens if the angle of the transport direction of the target solution is modified.

The training set is composed of a single sample, as indicated in Figure 17a, and the ROM is solved for different target values by modifying the transport direction (and its magnitude) as indicated in the same table, which lists the values of the study parameters

$$\mu_1 = a_x, \quad \mu_2 = a_y.$$

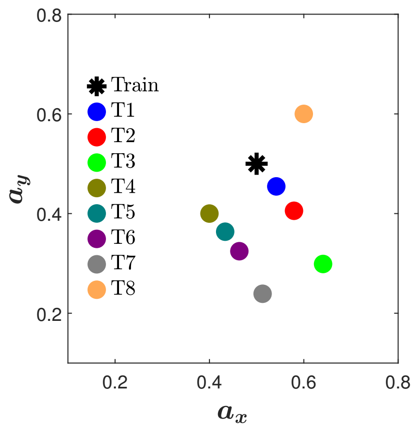
following the formulation indicated in 4.6. The diffusion parameter is given the following value $\nu = 0$.

$$a_x = \sqrt{a_0}\cos\alpha, \quad a_y = \sqrt{a_0}\sin\alpha. \tag{4.6}$$

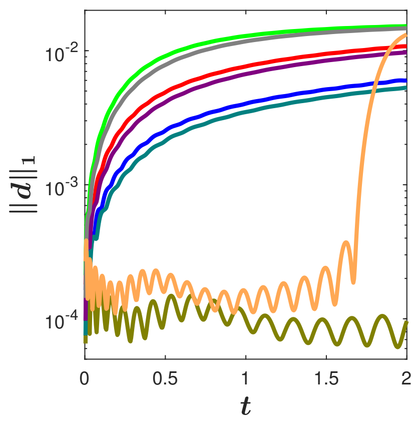
The training and target values are also shown in Figure 17a, where the circular symmetry of the field can be observed. The target velocities of T1, T2, and T3 are equal in magnitude to the training sample but move away from it by 5º, 10°, and 20° on the circumference with centre at (0, 0), respectively. The velocities of T5, T6, and T7 velocities are shifted by the same number of degrees but with a smaller magnitude. The velocities of T4 and T8 velocities are different in magnitude with respect to the training value, but they face in the same direction.

Table 12. Case 5. Model parameters.

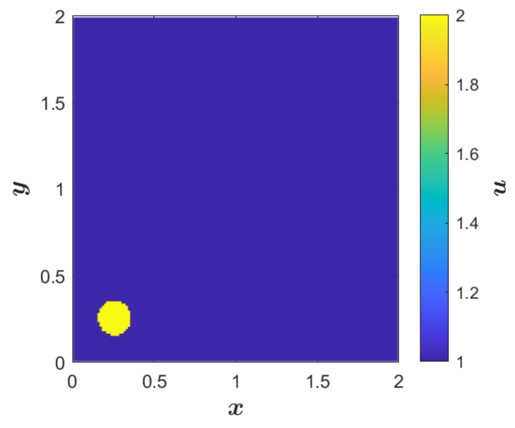
Sample	a_0	α	a_x	a_y
Train	0.5	45	0.5	0.5
Target 1	0.5	40	0.54167522	0.454519478
Target 2	0.5	35	0.57922797	0.405579788
Target 3	0.5	25	0.64085638	0.298836239
Target 4	0.4	45	0.4472136	0.447213595
Target 5	0.4	40	0.48448905	0.40653458
Target 6	0.4	35	0.51807724	0.36276159
Target 7	0.4	25	0.57319937	0.267287258
Target 8	0.6	45	0.54772256	0.547722558



(a) Training and target values.



(b) Differences.



(c) IC.

Figure 17. Case 5. Parameters values, final differences, and IC.

The time-space domain of the case is defined as $(x, y, t) \in [0, 2] \times [0, 2] \times [0, 2]$. The IC is the same for all target solutions and is given by

$$u(x,y,0) = \begin{cases} 2, & \text{if } (x-0.25)^2 + (y-0.25)^2 < 0.1, \\ 1, & \text{otherwise,} \end{cases} \tag{4.7}$$

and can be seen in Figure 17c. Free boundary conditions are considered.

The physical domain is discretized using $I_x \times I_y = 150 \times 150$ volume cells and CFL = 0.4. All the settings of the problem are shown in Table 13.

Table 13. Case 5. Problem settings.

$L_x \times L_y$	T	CFL	$I_x \times I_y$	a_x a_y	ν	BC	N_T	M_{POD}
2×2	2	0.4	150×150	Table 12	0	Free	467	20

Regarding the results, the target velocities of T4 and T8 agree with the results obtained in Case 2. As shown in Figure 17b, the differences computed for T4 stayed at a very low level for the whole time simulation, predicting highly accurate solutions, whereas the differences of T8 grow as its result exceeds the final position reached by the training solution (since its velocity is lower). As for the cases in which the direction of transport is modified, it can be seen in Figure 17b how their differences $||d||_1$ are grouped according to the angle at which they vary, being smaller for smaller angles. Accordingly, the change in magnitude has hardly any impact on these differences.

In addition to this, the representations shown in Figure 18 were proposed to further test the predictive capability of ROMs. In this figure, the time evolution of the solutions at three different points in the domain is shown. The dashed lines represent the target solution predicted by the ROM, and the solid lines represent the test solution computed by the FOM as reference solution. In this case, the following three points lie on the diagonal:

$$P_1 = (x_1, y_1) = (0.5, 0.5), \quad P_2 = (x_2, y_2) = (0.7, 0.7), \quad P_3 = (x_3, y_3) = (1, 1),$$

i.e., for values of $v_x = v_y$. Therefore, the figures show how the test solutions decrease in magnitude as their velocities move off the diagonal. In other words, at the point (x_3, y_3), the tests T1 and T5 present a magnitude that is greater than that of T2 and T6; at T3 and T7, it even disappears.

As it can be seen in the figures, the ROM predictions tend to maintain approximately constant magnitudes and do not decrease enough to resemble the tests. This is unacceptable for the targets of T2 and T6 (for 10° of separation from the diagonal) and even more so for T3 and T7 (for 10° of separation from the diagonal), where there should be no magnitude. It is therefore desirable that the target velocities are not modified beyond 5° with respect to the training velocities. With this in mind, the training sample must be carefully designed so as not to introduce an error that deviates from the expected physics, as shown in Case 6.

The ROM has been solved using 20 POD modes, requiring a CPU time of $\tau^{\text{ROM}}_{\text{CPU}} = 5.27 \cdot 10^{-4}$. Since the CPU time required by the FOM is $\tau^{\text{FOM}}_{\text{CPU}} = 6.64 \cdot 10^{-1}$, the speed up achieved is $\times 1268$.

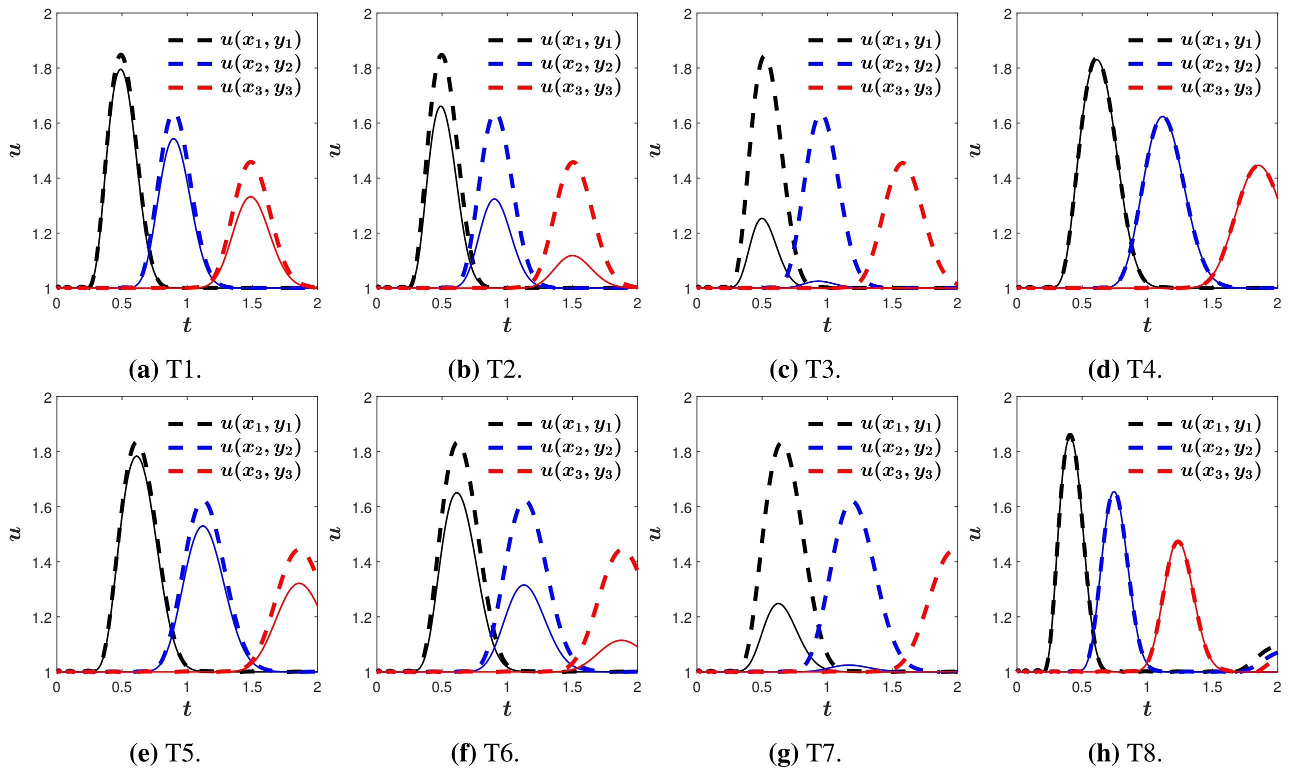


Figure 18. Case 5. Time evolution of u at different points.

4.2. Parametrized 2D WP model

A series of numerical results are presented below in which the parameterized fire model is solved taking into account the conclusions obtained from the results of the linear equation.

Case 6. Parameter: velocity

The parameter studied in this case is the wind velocity, as indicated in order in the following equations:

$$\mu_1 = v_x, \quad \mu_2 = v_y$$

These parameters are given the values shown in Figure 19, where the training velocities are grouped into two different magnitudes, i.e., 0.5 and 1. Within each group, the velocities are separated by 10° in circular symmetry, so that the four samples closest (in blue) to the test velocity (in red) are chosen to carry out the training. The rest of the parameters of the wildfire model (2.5) are given the following fixed values.

Table 14. Case 6. Model parameters.

v_x	v_y	k	ρ	c_p	α	H	T_∞	T_{ac}	T_{pc}	A
Figure 19	Figure 19	1	40	1	0.05	4000	300	400	400	0.05

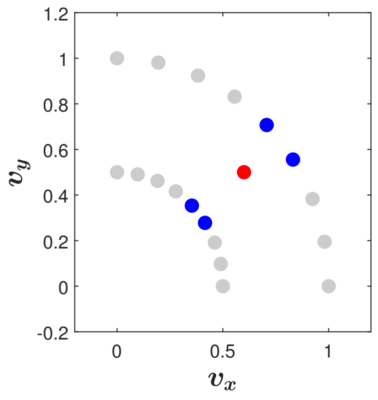


Figure 19. Case 6. Values of the training set (blue) and the target value (red).

The time-space domain of the case is defined as $(x, y, t) \in [0, 200] \times [0, 200] \times [0, 100]$. The IC for the temperature and the fuel mass fraction are given by

$$T(x, y, 0) = \begin{cases} 670, & \text{if } (x-25)^2 + (y-25)^2 < 10, \\ 300, & \text{otherwise,} \end{cases} \qquad Y(x, y, 0) = 1, \forall x, y. \tag{4.8}$$

Free boundary conditions are considered. The physical domain is discretized using $I_x \times I_y = 150 \times 150$ volume cells and CFL = 0.4. All the settings of the problem are shown in Table 15, including the number of POD modes M_{POD} and the number of snapshots per time window M_{STW}.

Table 15. Case 6. Problem settings.

$L_x \times L_y$	T	CFL	$I_x \times I_y$	BC	N_T	M_{POD}	M_{STW}
200×200	100	0.4	150×150	Free	939	30	100

Figure 20 shows the test solutions of T and Y at the final time, and Figure 21 shows the solutions predicted by the ROM at different time instants. In Figure 21c,f, it can be clearly seen that the solution predicted by the ROM is made up of the training solutions, because their shadows are superimposed. Because of this, the wavefront, which is very well delimited in the test case, as shown in the Figure 20, is very blurred in the ROM case.

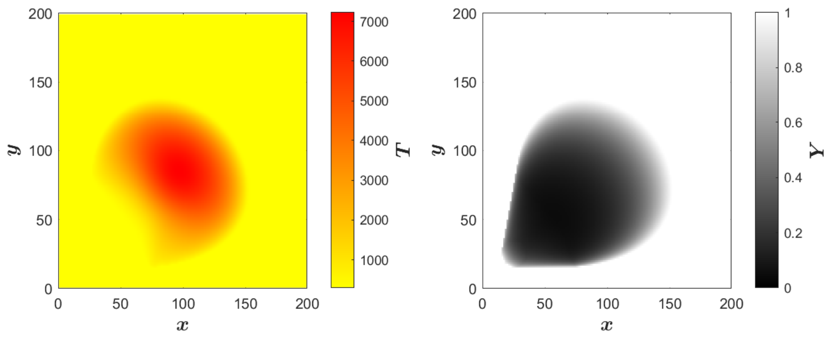
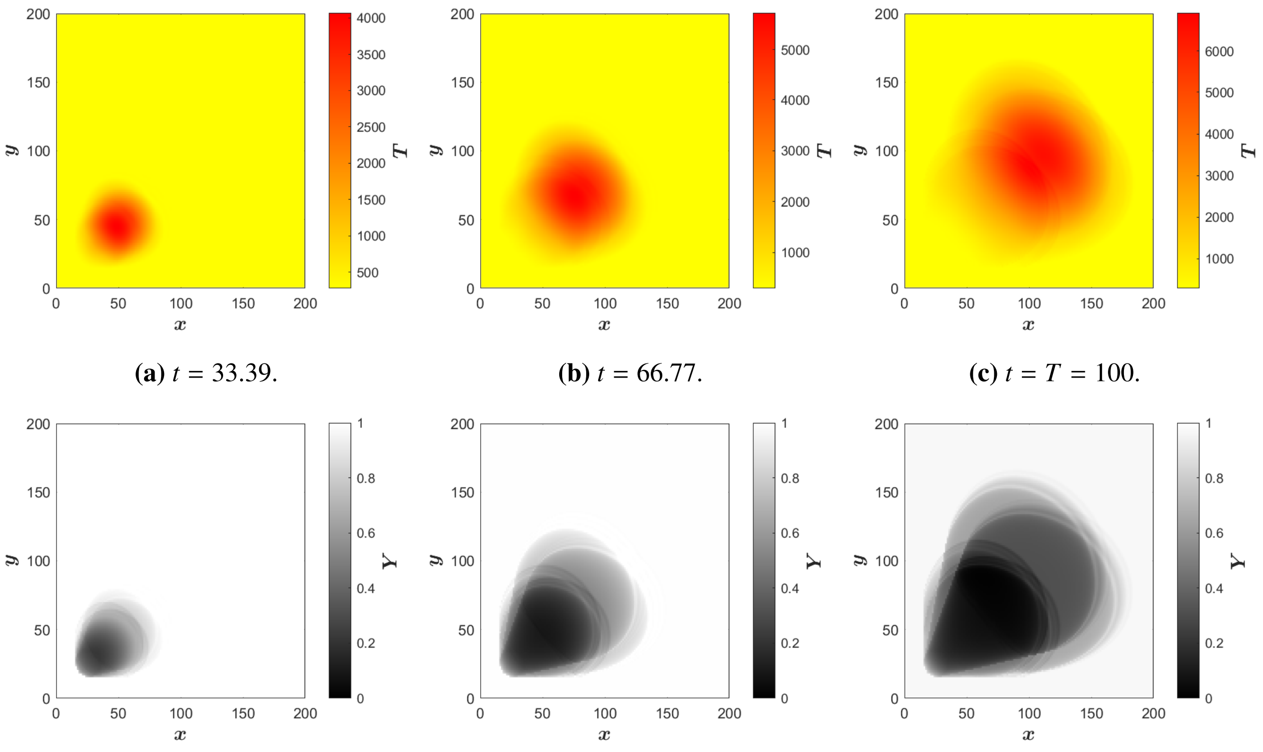


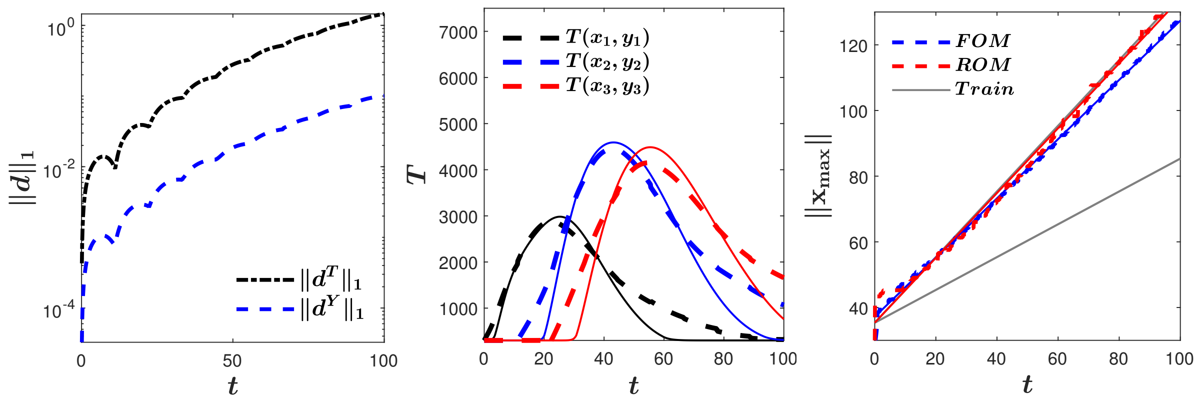
Figure 20. Case 6. Test solutions of T and Y at the final time.



(a) $t = 33.39$. **(b)** $t = 66.77$. **(c)** $t = T = 100$.

(d) $t = 33.39$. **(e)** $t = 66.77$. **(f)** $t = T = 100$.

Figure 21. Case 6. Test (left) and predicted (right) solutions of T and Y at the final time.



(a) Differences of T. **(b)** Solution of $T(150, 150, t)$. **(c)** Solution of $T(150, 150, t)$.

Figure 22. Case 6. Time evolution of the differences $\|d^T\|_1$ and $\|d^Y\|_1$, the time evolution of T at different points, and the speed-rate of the maximum value of T.

Nevertheless, the evolution of the solution at the three gauging points

$$P_1 = (x_1, y_1) = (35, 35), \quad P_2 = (x_2, y_2) = (50, 50), \quad P_3 = (x_3, y_3) = (65, 50), \tag{4.9}$$

shows an acceptable agreement between the two solutions, as can be seen in Figure 22b. The closer the predicted solution is to the origin of the IC, the more aligned it is with the test solution. However, as it evolves, it becomes out of phase. This can be seen in the Figure 22, which shows the time evolution of the absolute value of the position at which the maximum is located. In this figure, it can be seen that up to $t = 40$, the slope of the solution predicted by the ROM (red) overlaps with the test slope (blue); however, for longer times, it ends up tending to the closest training solution (gray).

The ROM has been solved using 30 POD modes with 100 snapshots per time window. The CPU time required by the FOM and the ROM are $\tau_{\text{CPU}}^{\text{ROM}} = 1.6$ and $\tau_{\text{CPU}}^{\text{ROM}} = 7.70 \cdot 10^{-2}$, respectively, so the speed up achieved is ×21.

Case 7. Parameter: Diffusion coefficient

The parameter studied in this case is the diffusion coefficient

$$\mu = k.$$

This parameter is given the values indicated in Table 16, where the two training values and the target value are found. The rest of the parameters of the model are given the fixed values indicated in Table 17.

Table 16. Case 7. Training and target values of the diffusion coefficient k.

k_1	k_2	k_T
0.01	10	1

Table 17. Case 7. Model parameters.

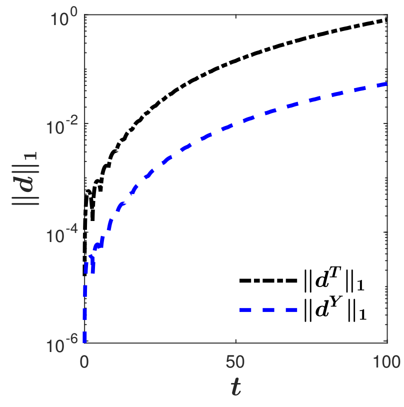
v_x	v_y	k	ρ	c_p	α	H	T_∞	T_{ac}	T_{pc}	A
0.5	0.5	Table 16	40	1	0.05	4000	300	400	400	0.05

The time-space domain of the case is defined as $(x, y, t) \in [0, 200] \times [0, 200] \times [0, 100]$. The IC for the temperature and the fuel mass fraction is given by (4.8). Free boundary conditions are considered. The physical domain is discretized using $I_x \times I_y = 150 \times 150$ volume cells and CFL = 0.4. All the settings of the problem are shown in Table 18, including the number of POD modes M_{POD} and the number of snapshots per time window M_{STW}.

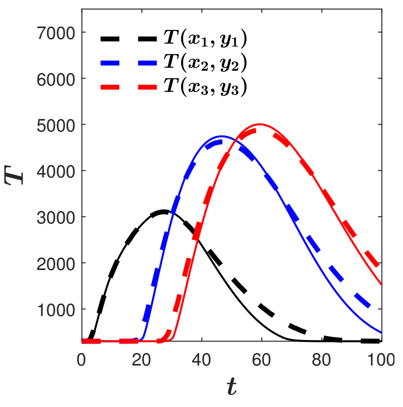
Table 18. Case 7. Problem settings.

$L_x \times L_y$	T	CFL	$I_x \times I_y$	BC	N_T	M_{POD}	M_{STW}
200×200	100	0.4	150×150	Free	5814	10	153

The time evolution of the solution compared at the three gauging points located in (4.9) reveals a good correspondence between the solution predicted by the ROM and the test solution, considering that the ROM takes a little longer to lower the temperature, as can be seen in the tails of each point. This confirms the conclusions obtained with the linear equation, since it is possible to train the ROM with only two values for the diffusion coefficient and predict accurate solutions, even using a training range as large as the one used in this case, ranging from 0.01 to 10. The ROM has been solved using 10 POD modes and 153 snapshots per time window, so the CPU time required by the FOM to compute the test solution and by the ROM to predict the target solution is $\tau_{\text{CPU}}^{\text{FOM}} = 9.71$ and $\tau_{\text{CPU}}^{\text{ROM}} = 9.30 \cdot 10^{-2}$, respectively, and the corresponding speed up is ×105.



(a) Differences of T.



(b) Solution of $T(150, 150, t)$.

Figure 23. Case 7. Time evolution of the differences $\|d^T\|_1$ and $\|d^Y\|_1$ and the time evolution of T at different points.

Case 8. Parameter: Pre-exponential factor

The parameter studied in this case is the pre-exponential factor

$$\mu = A.$$

This parameter is given the values indicated in Table 19, where the two training values and the target value are found. The rest of the parameters of the model are given the fixed values indicated in Table 20.

Table 19. Case 8. Training and target values of the pre-exponential factor A.

A_1	A_2	A_T
0.033	0.066	0.05

Table 20. Case 8. Model parameters.

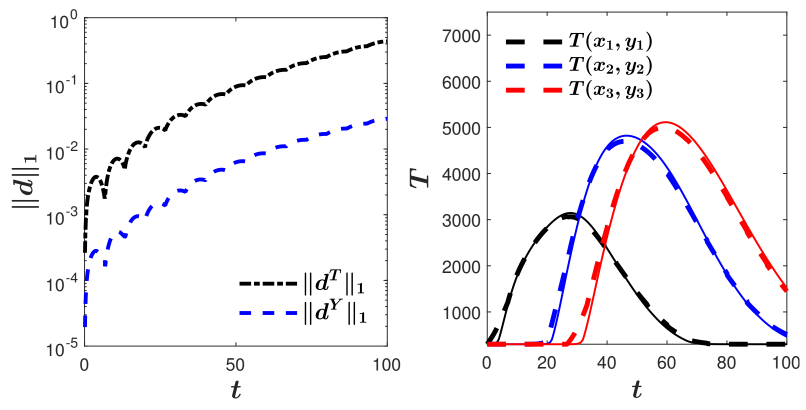
v_x	v_y	k	ρ	c_p	α	H	T_∞	T_{ac}	T_{pc}	A
0.5	0.5	2	40	1	0.05	4000	300	400	400	Table 19

The time-space domain of the case is defined as $(x, y, t) \in [0, 200] \times [0, 200] \times [0, 100]$. The IC for the temperature and the fuel mass fraction is given by (4.8). Free boundary conditions are considered. The physical domain is discretized using $I_x \times I_y = 150 \times 150$ volume cells and CFL $= 0.4$. All the settings of the problem are shown in Table 21.

Table 21. Case 8. Settings.

$L_x \times L_y$	T	CFL	$I_x \times I_y$	BC	N_T	M_{POD}	M_{STW}
200×200	100	0.4	150×150	Free	751	15	50

The results predicted by the ROM closely resemble the test results, as confirmed by the time evolution of the solution at the gauging points (4.9) in Figure 24b. The ROM is solved using 15 POD modes and 50 snapshots per time window. The CPU time required by the FOM to compute the test solution and by the ROM to predict the target solution is $\tau_{\text{CPU}}^{\text{FOM}} = 1.31$ and $\tau_{\text{CPU}}^{\text{ROM}} = 5.10 \cdot 10^{-2}$, respectively, and the corresponding speed up is $\times 26$.



(a) Differences of T. **(b)** Solution of $T(150, 150, t)$.

Figure 24. Case 8. Time evolution of the differences $||d^T||_1$ and $||d^Y||_1$ and time evolution of T at different points.

Case 9. Parameter: Reaction coefficient

The parameter studied in this case is the reaction coefficient

$$\mu = \alpha.$$

This parameter is given the values indicated in Table 22, where the two training values and the target value are found. The rest of the parameters of the model are given the fixed values indicated in Table 23.

Table 22. Case 9. Training and target values of the reaction coefficient α.

α_1	α_2	α_T
0.001	0.1	0.05

Table 23. Case 9. Model parameters.

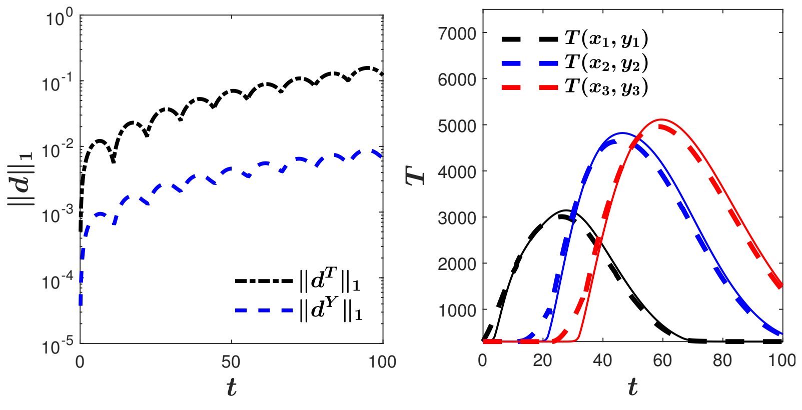
v_x	v_y	k	ρ	c_p	α	H	T_∞	T_{ac}	T_{pc}	A
0.5	0.5	1	40	1	Table 22	4000	300	400	400	0.05

The time-space domain of the case is defined as $(x, y, t) \in [0, 200] \times [0, 200] \times [0, 100]$. The IC for the temperature and the fuel mass fraction is given by (4.8). Free boundary conditions are considered. The physical domain is discretized using $I_x \times I_y = 150 \times 150$ volume cells and CFL $= 0.4$. All the settings of the problem are shown in Table 24.

Table 24. Case 9. Settings.

$L_x \times L_y$	T	CFL	$I_x \times I_y$	BC	N_T	M_{POD}	M_{STW}
200×200	100	0.4	150×150	Free	751	15	80

The results predicted by the ROM achieved high accuracy with respect to the test solution. The time evolution of the solution at the gauging points (4.9) confirms that only two training samples allow the ROM to predict accurate solutions when changing the reaction coefficient. The ROM has been solved using 15 POD modes and 80 snapshots per time window. The CPU time required by the FOM to compute the test solution and by the ROM to predict the target solution is $\tau_{\text{CPU}}^{\text{FOM}} = 1.31$ and $\tau_{\text{CPU}}^{\text{ROM}} = 2.90 \cdot 10^{-2}$, respectively, and the corresponding speed up is ×45.



(a) Differences of T. **(b)** Solution of $T(150, 150, t)$.

Figure 25. Case 9. Time evolution of the differences $\|d^T\|_1$ and $\|d^Y\|_1$ and time evolution of T at different points.

Case 10. Parameter: Shape of the IC

In this test case, the parameters that are studied belong to the IC. As concluded in the test cases solved with the linear equation, the shape of the IC can be arbitrarily modified, but its position presents more difficulties when it comes to being included in the ROM prediction. Because of this, in this case, we consider only the IC parameters related to its shape, which can be given by

$$T(x, y, 0) = \begin{cases} T_0, & \text{if } (x-20)^2 + (y-20)^2 < r^2, \\ 300, & \text{otherwise,} \end{cases} \qquad Y(x, y, 0) = 1, \forall x, y. \tag{4.10}$$

where

$$\mu_1 = r, \quad \mu_2 = T_0,$$

are the parameters of study. These are given the values contained in Table 25 together with the target values. The training ICs and the target IC are shown in Figure 26. The parameters that define the model (2.5) are given fixed values as shown in Table 26.

Table 25. Case 10. Training and target values of the IC given in (4.10).

Parameter	Train 1	Train 1	Train 1	Train 1	Target
r	20	14	7	2	10
T_0	400	550	750	900	670

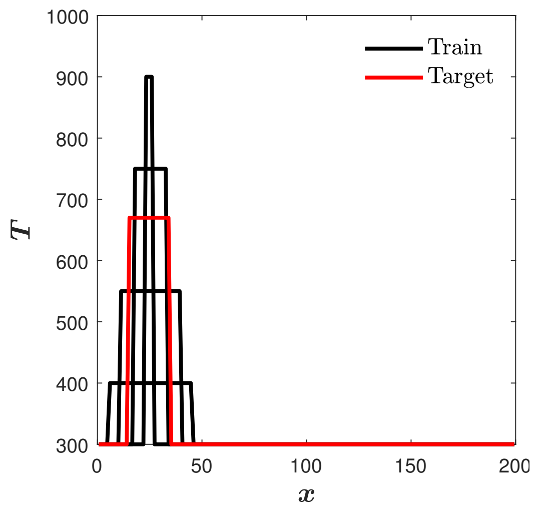


Figure 26. Case 10. Initial conditions of the training set (black) and the target solution (red).

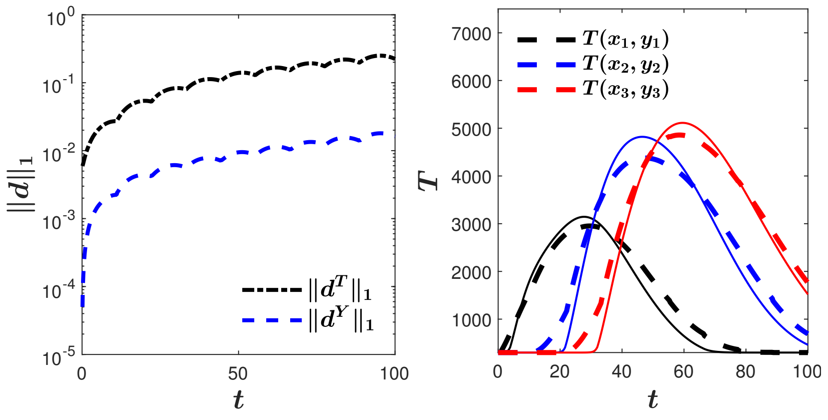
Table 26. Case 10. Model parameters.

v_x	v_y	k	ρ	c_p	α	H	T_∞	T_{ac}	T_{pc}	A
0.5	0.5	1	40	1	0.05	4000	300	400	400	0.05

The time-space domain of the case is defined as $(x, y, t) \in [0, 200] \times [0, 200] \times [0, 100]$. Free boundary conditions are considered. The physical domain is discretized using $I_x \times I_y = 150 \times 150$ volume cells and CFL = 0.4. All the settings of the problem are shown in Table 27.

Table 27. Case 10. Settings.

$L_x \times L_y$	T	CFL	$I_x \times I_y$	BC	N_T	M_{POD}	M_{STW}
200×200	50	0.3	150×150	Free	751	15	80



(a) Differences of T. **(b)** Solution of $T(150, 150, t)$.

Figure 27. Case 10. Time evolution of the differences $\|d^T\|_1$ and $\|d^Y\|_1$ and time evolution of T at different points.

The ROM is able to predict accurate solutions when the values that define the shape of the IC are changed. The time evolution of the solution at the gauging points (4.9) confirms this conclusion, since the ROM resembles the test even though the maximums are not as high as they should be (Figure 27). The ROM has been solved using 15 POD modes and 80 snapshots per time window. The CPU time required by the FOM to compute the test solution and by the ROM to predict the target solution is $\tau_{\mathrm{CPU}}^{\mathrm{FOM}} = 1.28$ and $\tau_{\mathrm{CPU}}^{\mathrm{ROM}} = 3.00 \cdot 10^{-2}$, respectively, and the corresponding speed up is $\times 43$.

5. Concluding remarks

A methodology has been proposed to solve the parametrized versions of the ADR equation (2.1) and the WP model (2.5) using POD-based ROMs that allows to predict solutions for values of the problem parameters other than the training ones.

Several numerical cases have been solved to perform a very detailed analysis of the ability of this parametrized ROM strategy, which are summarized in Table 28. In the first five cases, it has been applied to the 2D ADR equation (2.1), where solutions for different values of the advection velocity and the diffusion coefficient have been accurately predicted. Other input parameters succesfully considered are the coefficients defining the initial condition. The reduced version of the 2D WP model (2.5) needs definition of time windows to linearize the nonlinear coupling term. The parametrized ROM strategy is also applied to this model for certain physical parameters, such as the diffusion coefficient, the pre-exponential factor, and the reaction coefficient.

Table 28. Main conclusions of each test case.

Case	Equation	Parameter	$\|\|d\|\|_1$	speed up
1	2D ADR	ν	$\sim 10^{-4}$	$\sim 10^{1}$
2		a_x	$\sim 10^{-2}$	$\sim 10^{1}$
3.1		IC: $\bar{u}$	$\sim 10^{-2}$	$\sim 10^{1}$
3.2		IC: u_0	$\sim 10^{-1}$	$\sim 10^{1}$
3.3		IC: c	$\sim 10^{-1}$	$\sim 10^{1}$
3.4		IC: x_0	$\sim 10^{-2}$	$\sim 10^{1}$
4		$(M_{\mathrm{POD}}, M_{\mathrm{train}})$	$\sim 10^{-3}$	$\sim 10^{3}$
5		$\left(a_x, a_y\right)$	$\sim 10^{-2}$	$\sim 10^{3}$
6	2D WP	$\left(\nu_x, \nu_y\right)$	$\sim 10^{-1}$	$\sim 10^{1}$
7		k	$\sim 10^{-1}$	$\sim 10^{2}$
8		A	$\sim 10^{-1}$	$\sim 10^{1}$
9		α	$\sim 10^{-1}$	$\sim 10^{1}$
10		IC	$\sim 10^{-1}$	$\sim 10^{1}$

The particular conclusions obtained for the different parameters studied are summarized below. The ROM is able to predict solutions for new values of the diffusion and reaction coefficients if at least one training value is lower and other is higher than the target, as shown in Cases 1, 7, and 9. As for the advection velocity, it has been noted in Case 2 that when having an arbitrary set of training samples, it has to contain at least one with a velocity value as close as possible to the target and above it. In Cases 5 and 6, training samples with circularly distributed velocity values have been proposed, and it has been observed that a separation of 5° between the training and the target values allows the prediction of accurate solutions.

Given an arbitrary training set computed with random values of the parameters that define the IC, the ROM needs at least 18 POD modes (in 2D problems) and 4 training samples to predict accurate solutions with high speed up, as observed in Case 4. Regarding the values that define the IC, the ROM is able to predict solutions changing the offset or the amplitude of the initial profile when the training set is composed of at least two samples with arbitrary values, as shown in Case 3. Regarding the width of the initial profile, the ROM needs to be trained with samples computed for wider profiles than the target. Unlike the linear problem, the initial position is particularly difficult to predict when solving linearized ROMs, as shown in [41], such as the 2D WP model (3.8). It remains for future work to explore possibilities to overcome this limitation.

Use of AI tools declaration

The authors declare they have not used artificial intelligence (AI) tools in the creation of this article.

Acknowledgments

This work was supported by Project PID2022-137334NB-I00 funded by MCIN/AEI/10.13039/501100011033 and by European Regional Development Fund (ERDF) of the Spanish Ministry of Science and Innovation and the European Regional Development Fund. This work has also been funded by the Spanish Ministry of Science, Innovation and Universities – Agencia Estatal de Investigación (10.13039/501100011033) and Fondo Europeo de Desarrollo Regional (FEDER) under project-No. PID2022-141051NA-I00. It is also partially funded by the Government of Aragón, through the research grant T32_23R Fluid Dynamics Technologies. The authors belong and are supported by the Aragón Institute of Engineering Research (I3A) of the University of Zaragoza.

Conflict of interest

A. Navas-Montilla is a guest editor for this special issue and was not involved in the editorial review or the decision to publish this article. The authors declare there is no conflict of interest.

References

1. W. Hundsdorfer, J. G. Verwer, *Numerical Solution of Time-Dependent advection–diffusion–reaction Equations*, Springer Series in Computational Mathematics, Springer Berlin Heidelberg, 2013.
2. B. Q. Li, J. W. Evans, Boundary element solution of heat convection-diffusion problems, *J. Comput. Phys.*, **93** (1991), 255–272. https://doi.org/10.1016/0021-9991(91)90181-J
3. A. Boddy, E. Bentz, M. D. A. Thomas, R. D. Hooton, An overview and sensitivity study of a multimechanistic chloride transport model, *Cem. Concr. Res.*, **29** (1999), 827–837. https://doi.org/10.1016/S0008-8846(99)00045-9

4. F. Pigeonneau, L. Pereira, A. Laplace, Mass transfer around a rising bubble in a glass-forming liquid involving oxidation-reduction reaction: Numerical computation of the Sherwood number, *Chem. Eng. Sci.*, **232** (2021), 116382. https://doi.org/10.1016/j.ces.2020.116382

5. J. Mandel, L. S. Bennethum, J. D. Beezley, J. L. Coen, C. C. Douglas, M. Kim, et al., A wildland fire model with data assimilation, *Math. Comput. Simul.*, **79** (2008), 584–606. https://doi.org/10.1016/j.matcom.2008.03.015

6. P. W. McDonald, The computation of transonic flow through two-dimensional gas turbine cascades, in *Turbo Expo: Power for Land, Sea, and Air*, American Society of Mechanical Engineers, (1971), V001T01A089. https://doi.org/10.1115/71-GT-89

7. W. Maccormack, A. J. Paullay, *Computational Efficiency Achieved by Time Splitting of Finite Difference Operators*, AIAA, 1972. https://doi.org/10.2514/6.1972-154

8. S. K. Godunov, I. Bohachevsky, Finite difference method for numerical computation of discontinuous solutions of the equations of fluid dynamics, *Mat. Sbornik*, **47** (1959), 271–306.

9. R. J. LeVeque, *Finite Volume Methods for Hyperbolic Problems*, Cambridge Texts in Applied Mathematics, Cambridge University Press, 2002. https://doi.org/10.1017/CBO9780511791253

10. B. O. Almroth, P. Stern, F. A. Brogan, Automatic choice of global shape functions in structural analysis, *AIAA J.*, **16** (1978), 525–528. https://doi.org/10.2514/3.7539

11. J. L. Lumley, The structure of inhomogeneous turbulent flows, *Atmos. Turbul. Radio Wave Propag.*, (1967), 166–176.

12. I. Akhtar, Z. Wang, J. Borggaard, T. Iliescu, A new closure strategy for proper orthogonal decomposition reduced-order models, *J. Comput. Nonlinear Dyn.*, **7** (2012), 034503.

13. C. Prud'Homme, D. V. Rovas, K. Veroy, L. Machiels, Y. Maday, A. T. Patera, et al., Reliable real-time solution of parametrized partial differential equations: Reduced-basis output bound methods, *J. Fluids Eng.*, **124** (2001), 70–80. https://doi.org/10.1115/1.1448332

14. A. Quarteroni, A. Manzoni, F. Negri, *Reduced Basis Methods for Partial Differential Equations: An Introduction*, Springer, Cham, **92** (2016). https://doi.org/10.1007/978-3-319-15431-2

15. P. Solán-Fustero, J. L. Gracia, A. Navas-Montilla, P. García-Navarro, A POD-based ROM strategy for the prediction in time of advection-dominated problems, *J. Comput. Phys.*, **471** (2022), 111672. https://doi.org/10.1016/j.jcp.2022.111672

16. M. Ohlberger, S. Rave, Reduced basis methods: Success, limitations and future challenges, in *Proceedings of the Conference Algoritmy*, (2016), 1–12.

17. T. Taddei, A registration method for model order reduction: Data compression and geometry reduction, *SIAM J. Sci. Comput.*, **42** (2020), A997–A1027. https://doi.org/10.1137/19M1271270

18. R. R. Sánchez, M. Buchschmid, G. Müller, Model order reduction in design of parameterized structures under different load configurations, *Procedia Eng.*, **199** (2017), 378–383. https://doi.org/10.1016/j.proeng.2017.09.049

19. S. Burela, P. Krah, J. Reiss, Parametric model order reduction for a wildland fire model via the shifted pod-based deep learning method: Parametric model order reduction for a wildland fire model, *Adv. Comput. Math.*, **51** (2025), 9. https://doi.org/10.1007/s10444-025-10220-4

20. J. M. Zokagoa, A. Soulaïmani, A POD-based reduced-order model for uncertainty analyses in shallow water flows, *Int. J. Comput. Fluid Dyn.*, **32** (2018), 278–292. https://doi.org/10.1080/10618562.2018.1513496

21. F. Black, P. Schulze, B. Unger, Efficient wildland fire simulation via nonlinear model order reduction, *Fluids*, **6** (2021), 280. https://doi.org/10.3390/fluids6080280

22. F. Black, P. Schulze, B. Unger, Projection-based model reduction with dynamically transformed modes, *ESAIM Math. Model. Numer. Anal.*, **54** (2020), 2011–2043. https://doi.org/10.1051/m2an/2020046

23. K. Miller, R. N. Miller, Moving finite elements. I, *SIAM J. Numer. Anal.*, **18** (1981), 1019–1032. https://doi.org/10.1137/0718070

24. J. Fu, D. Xiao, R. Fu, C. Li, C. Zhu, R. Arcucci, et al., Physics-data combined machine learning for parametric reduced-order modelling of nonlinear dynamical systems in small-data regimes, *Comput. Methods Appl. Mech. Eng.*, **404** (2023), 115771. https://doi.org/10.1016/j.cma.2022.115771

25. R. Fu, D. Xiao, A. G. Buchan, X. Lin, Y. Feng, G. Dong, A parametric nonlinear non-intrusive reduce-order model using deep transfer learning, *Com. Met. App. Mec. Eng.Comput. Methods Appl. Mech. Eng.*, **438** (2025), 117807. https://doi.org/10.1016/j.cma.2025.117807

26. X. Pan, D. Xiao, Domain decomposition for physics-data combined neural network based parametric reduced order modelling, *J. Comput. Phys.*, **519** (2024), 113452. https://doi.org/10.1016/j.jcp.2024.113452

27. S. Georgaka, G. Stabile, G. Rozza, M. Bluck, Parametric pod-galerkin model order reduction for unsteady-state heat transfer problems, *Commun. Comput. Phys.*, **27** (2018), 1–32. https://doi.org/10.4208/cicp.OA-2018-0207

28. C. Reisch, A. Navas-Montilla, I. Özgen-Xian, Analytical and numerical insights into wildfire dynamics: Exploring the advection–diffusion–reaction model, *Comput. Math. Appl.*, **158** (2024), 179–198. https://doi.org/10.1016/j.camwa.2024.01.024

29. M. A. Finney, *FARSITE: Fire Area Simulator-model Development and Evaluation*, 1998. http://doi.org/10.2737/RMRS-RP-4.

30. G. L. W. Perry, Current approaches to modelling the spread of wildland fire: a review, *Prog. Phys. Geogr.*, **22** (1998), 222–245. https://doi.org/10.1177/030913339802200204

31. M. E. Hubbard, Multidimensional slope limiters for MUSCL-type finite volume schemes on unstructured grids, *J. Comput. Phys.*, **155** (1999), 54–74. https://doi.org/10.1006/jcph.1999.6329

32. R. J. LeVeque, A well-balanced path-integral f-wave method for hyperbolic problems with source terms, *J. Sci. Comput.*, **58** (2011), 209–226. https://doi.org/10.1007/s10915-010-9411-0

33. R. Courant, K. Friedrichs, H. Lewy, Über die partiellen Differenzengleichungen der mathematischen Physik, *Math. Ann.*, **100** (1928), 32–74. https://doi.org/10.1007/BF01448839

34. P. García-Navarro, M. E. Vázquez-Cendón, On numerical treatment of the source terms in the shallow water equations, *Comput. Fluids*, **29** (2000), 951–979. https://doi.org/10.1016/S0045-7930(99)00038-9

35. J. Murillo, P. García-Navarro, Weak solutions for partial differential equations with source terms: Application to the shallow water equations, *J. Comput. Phys.*, **229** (2010), 4327–4368. https://doi.org/10.1016/j.jcp.2010.02.016

36. H. Berestycki, B. Larrouturou, J. M. Roquejoffre, Mathematical investigation of the cold boundary difficulty in flame propagation theory, in *Dynamical Issues in Combustion Theory* (eds. P. C. Fife, A. Liñán, and F. Williams), Springer, New York, (1991), 37–61.

37. A. Navas-Montilla, J. Guallart, P. Solán-Fustero, P. García-Navarro, Exploring the potential of TENO and WENO schemes for simulating under-resolved turbulent flows in the atmosphere using Euler equations, *Comput. Fluids*, **280** (2024) 106349. https://doi.org/10.1016/j.compfluid.2024.106349

38. L. Sirovich, Turbulence and the dynamics of coherent structures. I - Coherent structures. II - Symmetries and transformations. III - Dynamics and scaling, *Q. Appl. Math.*, **45** (1987), 561–590. https://doi.org/10.1090/qam/910463

39. G. H. Golub, C. F. Van Loan, *Matrix Computations*, Johns Hopkins University Press, 2013.

40. B. G. Galerkin, Series occurring in various questions concerning the elastic equilibrium of rods and plates, *Eng. Bull.*, **19** (1915), 897–908.

41. P. Solán-Fustero, *Reduced-order Models Based on the Proper Orthogonal Decomposition Applied to Hyperbolic Problems*, Ph.D thesis, University of Zaragoza, 2024.