Received: 21 April 2017 | Revised: 9 August 2017 | Accepted: 26 August 2017

DOI: 10.1002/fld.4460

RESEARCH ARTICLE

WILEY

An indirect shooting method based on the POD/DEIM technique for distributed optimal control of the wave equation

Z. Sabeh[1] | Mostafa Shamsi[1] | Ionel Michael Navon[2]

[1]Department of Applied Mathematics, Faculty of Mathematics and Computer Science, Amirkabir University of Technology, Tehran, Iran

[2]Department of Scientific Computing, Florida State University, Tallahassee, FL 32306-4120, USA

Correspondence
Mostafa Shamsi, Department of Applied Mathematics, Faculty of Mathematics and Computer Science, Amirkabir University of Technology, Tehran, Iran.
Email: m_shamsi@aut.ac.ir

Summary

This paper presents a fast numerical method, based on the indirect shooting method and Proper Orthogonal Decomposition (POD) technique, for solving distributed optimal control of the wave equation. To solve this problem, we consider the first-order optimality conditions and then by using finite element spatial discretization and shooting strategy, the solution of the optimality conditions is reduced to the solution of a series of initial value problems (IVPs). Generally, these IVPs are high-order and thus their solution is time-consuming. To overcome this drawback, we present a POD indirect shooting method, which uses the POD technique to approximate IVPs with smaller ones and faster run times. Moreover, in the presence of the nonlinear term, to reduce the order of the nonlinear calculations, a discrete empirical interpolation method (DEIM) is applied and a POD/DEIM indirect shooting method is developed. We investigate the performance and accuracy of the proposed methods by means of 4 numerical experiments. We show that the presented POD and POD/DEIM indirect shooting methods dramatically reduce the CPU time compared to the full indirect shooting method, whereas there is no significant difference between the accuracy of the reduced and full indirect shooting methods.

KEYWORDS

discrete empirical interpolation method (DEIM), finite element method, indirect shooting method, necessary optimality conditions, optimal control of wave equation, proper orthogonal decomposition (POD)

1 | INTRODUCTION

In this paper, we consider the distributed optimal control problems governed by linear and nonlinear wave-type equations.[1-3] Let $\Omega \subseteq \mathbb{R}^d$, $d \in \{1, 2, 3\}$ be a bounded spatial domain with boundary $\partial\Omega$, $T \geq 0$ be the final time, $Q := (0, T) \times \Omega$ and $\Sigma := (0, T) \times \partial\Omega$. In the distributed optimal control problem for the wave equation, the aim is to find control $u(t, \mathbf{x})$ and state $y(t, \mathbf{x})$ such that the following wave equation:

$$y_{tt} - \Delta y + \mathcal{N}_1(y) = f + u, \qquad \text{in } Q \tag{1a}$$

with initial conditions

$$y(0,\mathbf{x}) = g(\mathbf{x}), \qquad \text{in } \Omega \tag{1b}$$

$$y_t(0,\mathbf{x}) = h(\mathbf{x}), \qquad \text{in } \Omega \tag{1c}$$

and boundary condition

$$y(t,\mathbf{x}) = 0, \qquad \text{on } \Sigma \tag{1d}$$

are satisfied and furthermore the following performance index is minimized:

$$\mathcal{J} = \frac{\kappa_1}{2}\iint_Q \left|y(t,\mathbf{x}) - z(t,\mathbf{x})\right|^2 d\mathbf{x}dt + \frac{\kappa_2}{2}\int_\Omega \left|y(T,\mathbf{x}) - w(\mathbf{x})\right|^2 d\mathbf{x} + \frac{1}{2}\iint_Q \left|u(t,\mathbf{x})\right|^2 d\mathbf{x}dt. \tag{1e}$$

In this problem, $z(t,\mathbf{x})$ and $w(\mathbf{x})$ are target functions, $g(\mathbf{x})$ and $h(\mathbf{x})$ are initial conditions, $f(t,\mathbf{x})$ is a given force function, $\kappa_1, \kappa_2 \geqslant 0$ are constants and $\mathcal{N}_1(y)$ is a generally nonlinear function of y.

Control problems for the wave equations arise in several applications, eg, medical applications,[4] acoustic problems as noise suppression,[5] and linear elasticity.[6] As a consequence, several papers have been presented in this area. In other works[7-9] and the references therein, the controllability of the wave equation is investigated. Optimal boundary control of the wave equation is considered in various publications. See for instance previous studies.[10-15] Recently, several papers have been presented for the numerical solution of the distributed control problem for the wave equation. In Luo et al,[16] the finite volume element method is applied to the hyperbolic optimal control problems and a priori error estimation is derived. A space-time finite element discretization, with a priori error estimation, is considered in Kröner,[17] for the optimal control of nonlinear wave equation. In Li et al,[3] a new central finite difference scheme and in Kunisch and Reiterer,[18] a Gautschi time-stepping scheme is proposed for optimal control problem of wave equations. The semi-smooth Newton methods are investigated in Kroner et al,[19] where the control is restricted by pointwise lower and upper bounds. For state constrained optimal distributed control of the wave equation, the Lavrentiev regularization is considered in Gugat et al.[20]

In this work, we propose another method on the basis of the shooting method and proper orthogonal decomposition method for solving the distributed optimal control problem (1). Our main aim is to develop an efficient algorithm with low computational complexity for solving (1).

Shooting methods are an important class of numerical methods for solving optimal control problems that are applicable in the framework of the direct and indirect approach. Direct shooting approach reformulates the optimal control problem to an optimization problem, which can be solved by well-developed solvers. In contrast, indirect shooting methods peruse first-optimize-then-shooting approach[21] and apply the shooting method to the optimality conditions of optimal control problem, which forms an initial terminal boundary value problem (ITBVP). Indirect shooting method converts the solution of ITBVP into an initial boundary value problem (IBVP) by considering guess values for some initial conditions. These guess values are iteratively updated until the terminal conditions of the original ITBVP are met.

In general, the solution found by indirect methods satisfies the optimality conditions and usually is more accurate than the solution of direct methods. On the other hand, the region of convergence for indirect methods may be smaller than the region of convergence for direct methods; therefore, they require better initial guess.[22] We note that a common feature of the two classes of shooting methods is that they are time-consuming and this is more significant for large-scale optimization problems. To overcome this drawback, it is necessary to use the techniques for reduction of computation time for these methods.

Proper Orthogonal Decomposition (POD) is a well-known model reduction technique for nonlinear PDEs. This method generates an optimal set of basis functions (so-called POD basis functions) where each of them has a global support and involves information about the system obtained out of a computational or experimental database (snapshots). After this, the solution is obtained as a linear combination of the POD basis functions by means of Galerkin projection method. We can refer to other works[23-26] for basic concepts in the POD method,[27-32] for using POD technique on numerical solution of PDEs,[33] for application of POD in PDE optimal control,[34-37] for error estimation of the POD method for optimal control problems and to previous studies[38-44] for other engineering applications.

In this paper, for solving the problem (1), the optimality conditions of this problem are considered. Then, using finite element (FE) spatial discretization and shooting strategy, the solution of the optimality conditions reduces to the solution of a series of initial value problems (IVPs). However, the dimension of these IVPs depends on the number of elements in the FE spatial discretization. On the other hand, to get a reasonable accuracy, the required size of IVPs sometimes is of the order of hundred thousands of variables. Consequently, the method is very time-consuming. To resolve this high dimensionality and CPU usage, we use the POD method for reducing the order of IVPs. To do this, the constructing of POD basis and applying the POD technique to the problem (1) are presented in details and an algorithm, for solving this problem, is developed.

In the absence of the nonlinear term $\mathcal{N}_1$, the POD shooting method is largely satisfactory. However, when the problem is nonlinear, in the evaluation of nonlinear term, the mentioned computational savings in the POD technique are not attained. As a result, the efficiency of POD strategy is reduced in the presence of the nonlinear term. Several approaches have been proposed to overcome this computational inefficiency.[45] One of these approaches is discrete empirical interpolation method (DEIM), which is used in this paper. DEIM technique approximates a nonlinear function by combining projection with interpolation.[45,46] DEIM is a discrete variant of empirical interpolation method that is based on the greedy algorithms.[24]

The structure of this paper is outlined as follows: In Section 2, the necessary optimality conditions of the distributed optimal control problem of the wave equation are introduced. A discussion of the shooting algorithm for solving the optimality system is also presented in this section. In Section 3, an algorithm based on the indirect shooting method and POD method is developed for solving the optimal control of the wave equation. In Section 4, an efficient algorithm, based on DEIM strategy, is presented for the optimal control of nonlinear wave equation. Numerical examples are given in Section 5 to demonstrate the effectiveness of the proposed method.

2 | INDIRECT SHOOTING METHOD AND FINITE ELEMENT SPATIAL DISCRETIZATION

The first-order necessary optimality conditions for optimal control problem (1) are well established in other studies.[47,48] We briefly recall them here. Suppose that $y(t,\mathbf{x})$ and $u(t,\mathbf{x})$ are the optimal state and control, then there exists an adjoint function $p(t,\mathbf{x})$ such that $(y(t,\mathbf{x}),u(t,\mathbf{x}),p(t,\mathbf{x}))$ satisfies the following first order necessary optimality conditions:

$$\begin{cases} y_{tt}-\Delta y+\mathcal{N}_1(y)=f+u, & \text{in } Q, \quad (2a)\\ y(0,\mathbf{x})=g(\mathbf{x}), & \text{in } \Omega, \quad (2b)\\ y_t(0,\mathbf{x})=h(\mathbf{x}), & \text{in } \Omega, \quad (2c)\\ y(t,\mathbf{x})=0, & \text{on } \Sigma, \quad (2d)\\ p_{tt}-\Delta p+\mathcal{N}_2(y,p)=\kappa_1\,(y-z)\,, & \text{in } Q, \quad (2e)\\ p(T,\mathbf{x})=0, & \text{in } \Omega, \quad (2f)\\ p_t(T,\mathbf{x})=-\kappa_2(y(T,\mathbf{x})-w(\mathbf{x})), & \text{in } \Omega, \quad (2g)\\ p(t,\mathbf{x})=0, & \text{on } \Sigma, \quad (2h)\\ p+u=0, & \text{in } Q, \quad (2i) \end{cases}$$

where $\mathcal{N}_2(y,p)=p\frac{\partial\mathcal{N}_1}{\partial y}(y)$. After eliminating u from (2a) by (2i), this system can be converted to a standard two-point boundary value problem with respect to t. We note that the resulted boundary value system contains both initial and terminal conditions in time. Accordingly, it cannot be solved by marching methods in time.[3,49] However, the well-known shooting method can be used for the numerical solution of it.

In the first step of shooting method for the numerical solution of (2), we consider a corresponding IBVP with unknown initial conditions for the adjoint variable. If the unknown initial conditions are denoted by $\eta,\gamma:\Omega\to\mathbb{R}$, then the corresponding IBVP is considered as follows:

$$\begin{cases} y_{tt}-\Delta y+\mathcal{N}_1(y)=f-p, & \text{in } Q, \quad (3a)\\ p_{tt}-\Delta p+\mathcal{N}_2(y,p)=\kappa_1\,(y-z)\,, & \text{in } Q, \quad (3b)\\ y(0,\mathbf{x})=g(\mathbf{x}), \quad y_t(0,\mathbf{x})=h(\mathbf{x}), & \text{in } \Omega, \quad (3c)\\ p(0,\mathbf{x})=\eta(\mathbf{x}), \quad p_t(0,\mathbf{x})=\gamma(\mathbf{x}), & \text{in } \Omega, \quad (3d)\\ y(t,\mathbf{x})=0, \qquad p(t,\mathbf{x})=0, & \text{on } \Sigma. \quad (3e) \end{cases}$$

It should be mentioned that the solution of IBVP (3) depends not only on t and $\mathbf{x}$ but also on the initial functions $\eta(\mathbf{x})$ and $\gamma(\mathbf{x})$. To emphasize this dependence, we use notations $y(t,\mathbf{x};\eta,\gamma)$ and $p(t,\mathbf{x};\eta,\gamma)$ for the solution of IBVP (3). In the second step of shooting method, we seek the initial functions $\eta(\mathbf{x})$ and $\gamma(\mathbf{x})$ such that the solutions $y(t,\mathbf{x};\eta,\gamma)$ and $p(t,\mathbf{x};\eta,\gamma)$ satisfy the following terminal conditions:

$$\mathrm{F}(\mathbf{x};\eta,\gamma)=\begin{bmatrix} p(T,\mathbf{x};\eta,\gamma) \\ p_t(T,\mathbf{x};\eta,\gamma)+\kappa_2\left(y\left(T,\mathbf{x};\eta,\gamma\right)-w(\mathbf{x})\right)\end{bmatrix}=0. \tag{4}$$

Therefore, solving the optimality system (2) is equivalent to finding initial functions $\eta(\mathbf{x})$ and $\gamma(\mathbf{x})$ such that the solution of IBVP (3) satisfies Equation 4. This strategy can be summarized as the following infinite shooting problem:

Infinite shooting problem (ISP): *Find $\eta,\gamma:\Omega\to\mathbb{R}$ such that the solutions $y\left(t,\mathbf{x};\eta,\gamma\right)$ and $p\left(t,\mathbf{x};\eta,\gamma\right)$ of Equation 3 fulfill the terminal condition* (4).

In the following, for the sake of simplicity in notation, we will omit η and γ from $y(t,\mathbf{x};\eta,\gamma)$ and $p(t,\mathbf{x};\eta,\gamma)$ and easily write $y(t,\mathbf{x})$ and $p(t,\mathbf{x})$.

We note that ISP is an infinite dimensional problem and at first, we need to make a spatial discretization for its numerical solution. In this paper, we consider the finite element method for spatial discretization of ISP. Suppose that functions $\varphi_1(\mathbf{x}),\ \dots\ ,\varphi_m(\mathbf{x})$ be linearly independent nodal basis functions and

$$X^h=\mathrm{span}\{\varphi_1(\mathbf{x}),\ \dots\ ,\varphi_m(\mathbf{x})\}\subset L^2(\Omega). \tag{5}$$

We approximate state $y(t,\mathbf{x})$ and adjoint functions $p(t,\mathbf{x})$ in the space X^h by $y^h(t,\mathbf{x})$ and $p^h(t,\mathbf{x})$ as follows:

$$y^h(t,\mathbf{x})=\sum_{j=1}^{m}\mathrm{y}_j(t)\varphi_j(\mathbf{x}),\qquad p^h(t,\mathbf{x})=\sum_{j=1}^{m}\mathrm{p}_j(t)\varphi_j(\mathbf{x}), \tag{6}$$

as well as the functions $\eta(\mathbf{x})$ and $\gamma(\mathbf{x})$ are approximated as follows:

$$\eta^h(\mathbf{x})=\sum_{j=1}^{m}\eta_j\varphi_j(\mathbf{x}),\qquad \gamma^h(\mathbf{x})=\sum_{j=1}^{m}\gamma_j\varphi_j(\mathbf{x}), \tag{7}$$

where $\mathrm{y}_j,\mathrm{p}_j,\eta_j,\gamma_j,\ j=1,\ \dots\ ,m$ are unknown finite element coefficients and we set the following:

$$\mathbf{y}(t)=\left[\mathrm{y}_1(t),\ \dots\ ,\mathrm{y}_m(t)\right]^T,\qquad \mathbf{p}(t)=\left[\mathrm{p}_1(t),\ \dots\ ,\mathrm{p}_m(t)\right]^T,$$
$$\boldsymbol{\eta}=[\eta_1,\ \dots\ ,\eta_m]^T,\qquad \boldsymbol{\gamma}=[\gamma_1,\ \dots\ ,\gamma_m]^T.$$

By substituting approximations (6) and (7) in the weak form of (3), finally, we obtain the following IVP:

$$\begin{cases} \mathbf{M}\ddot{\mathbf{y}}(t)+\mathbf{K}\mathbf{y}(t)+\mathbf{N}_1(t)=\mathbf{f}(t)-\mathbf{M}\mathbf{p}(t), & \text{(8a)}\\ \mathbf{M}\ddot{\mathbf{p}}(t)+\mathbf{K}\mathbf{p}(t)+\mathbf{N}_2(t)=\kappa_1\left[\mathbf{M}\mathbf{y}(t)-\mathbf{z}(t)\right], & \text{(8b)}\\ \mathbf{M}\mathbf{y}(0)=\mathbf{g}, \qquad \mathbf{M}\dot{\mathbf{y}}(0)=\mathbf{h}, & \text{(8c)}\\ \mathbf{p}(0)=\boldsymbol{\eta}, \qquad \dot{\mathbf{p}}(0)=\boldsymbol{\gamma}, & \text{(8d)}\end{cases}$$

where

$$[\mathbf{M}]_{ij}=\langle\varphi_j,\varphi_i\rangle,\quad [\mathbf{K}]_{ij}=\langle\nabla\varphi_j,\nabla\varphi_i\rangle,\quad \left[\mathbf{f}(t)\right]_i=\langle\varphi_i,f\rangle, \tag{9a}$$

$$[\mathbf{z}(t)]_i=\langle\varphi_i,z\rangle,\qquad \left[\mathbf{g}\right]_i=\langle\varphi_i,g\rangle,\qquad [\mathbf{h}]_i=\langle\varphi_i,h\rangle, \tag{9b}$$

$$[\mathbf{N}_1(t)]_i=\langle\varphi_i,\mathcal{N}_1(y^h)\rangle,\qquad [\mathbf{N}_2(t)]_i=\langle\varphi_i,\mathcal{N}_2(y^h,p^h)\rangle, \tag{9c}$$

and $\langle\cdot,\cdot\rangle$ stands for the L^2 inner product. In a similar manner, the shooting function (4) is discretized to

$$\mathrm{F}^h(\boldsymbol{\eta},\boldsymbol{\gamma})=\begin{bmatrix}\mathbf{M}\mathbf{p}(T)\\ \mathbf{M}\dot{\mathbf{p}}(T)+\kappa_2\left(\mathbf{M}\mathbf{y}(T)-\mathbf{w}\right)\end{bmatrix}=0, \tag{10}$$

where

$$[\mathbf{w}]_i=\langle\varphi_i,w\rangle. \tag{11}$$

In summary, by using finite element method, ISP can be discretized to the following finite dimensional shooting problem

Finite shooting problem (FSP): *Find $\boldsymbol{\eta},\boldsymbol{\gamma}\in\mathbb{R}^m$ such that the solution of IVP (8) fulfills the discrete terminal condition* (10).

On the basis of the above discussions, we summarize the indirect shooting method, for solving the problem (1), in Algorithm 1.

In Algorithm 1, for solving IVP (8), generally, an appropriate Runge-Kutta time marching method can be used. The dimension of IVP (8) depends on m, the number of considered basis functions in the finite element method. We note that m should be extremely large when high accuracy is required. Moreover, IVP (8) must be iteratively solved with different initial values during the solution of terminal Equation 10. Accordingly, the solution of the problem (1) by the indirect shooting algorithm is very time-consuming. This is the motivation for us to use a reduced-order approach on the basis of POD technique for improving the efficiency of the method.

Algorithm 1 : Indirect shooting (IS) algorithm

Inputs:

$m \in \mathbb{N}$ as the number of finite element basis.

1: **Begin**
2: Compute matrices $\mathbf{M}, \mathbf{K}$ and vectors $\mathbf{f}(t), \mathbf{z}(t), \mathbf{g}, \mathbf{h}, \mathbf{w}$ as (9) and (11).
3: Solve the system of equations $\mathrm{EQF}(\boldsymbol{\eta}, \boldsymbol{\gamma}) = \mathbf{0}$, by a solver and denote the solution by $(\boldsymbol{\eta}^*, \boldsymbol{\gamma}^*)$.
4: Solve IVP (8) with $(\boldsymbol{\eta}, \boldsymbol{\gamma}) = (\boldsymbol{\eta}^*, \boldsymbol{\gamma}^*)$ and denote the solutions by $\mathrm{y}_j^*(t)$ and $\mathrm{p}_j^*(t)$.
5: Use $\mathrm{y}_j^*(t)$ and $\mathrm{p}_j^*(t)$ in (6) to generate an approximation for the solution of the problem (1).
6: **End Algorithm**

Function $\mathbf{r} = \mathrm{EQF}(\mathbf{a}, \mathbf{b})$
Global: $\mathbf{M}, \mathbf{Q}, \mathbf{K}, \mathbf{f}(t), \mathbf{z}(t), \mathbf{g}, \mathbf{h}, \mathbf{w}$.

1: Solve IVP (8) to obtain the approximate values of $\mathbf{p}(T), \dot{\mathbf{p}}(T)$ and $\mathbf{y}(T)$, where $\mathbf{a}$ and $\mathbf{b}$ are considered as initial values for $\mathbf{p}(0)$ and $\dot{\mathbf{p}}(0)$, respectively.
2: **return** $\mathbf{r} = \begin{bmatrix} \mathbf{M}\mathbf{p}(T) \\ \mathbf{M}\dot{\mathbf{p}}(T) + \kappa_2 \left(\mathbf{M}\mathbf{y}(T) - \mathbf{w}\right) \end{bmatrix}$.

End Function

3 | PROPER ORTHOGONAL DECOMPOSITION(POD) METHOD FOR MODEL REDUCTION OF ISP

In this section, we first review POD technique and develop a POD Galerkin scheme for discretization of ISP. By this, we have a reduced-order initial value problem that must be iteratively solved in the shooting method.

3.1 | Computation of POD basis for discretizing ISP

As mentioned in the last section, by using the finite element method with basis functions $\{\varphi_1(\mathbf{x}), \ldots, \varphi_m(\mathbf{x})\}$, IBVP (3) is discretized to IVP (8). For a given $\boldsymbol{\eta}$ and $\boldsymbol{\gamma}$, let $\mathbf{y}(t) = \left[\mathrm{y}_1(t), \ldots, \mathrm{y}_m(t)\right]^T$ and $\mathbf{p}(t) = \left[\mathrm{p}_1(t), \ldots, \mathrm{p}_m(t)\right]^T$ be the obtained solutions of IVP (8). Thus, in view of (6) and (7), $y^h(t, \mathbf{x})$ and $p^h(t, \mathbf{x})$ are approximate solutions for IBVP (3) with $\eta(\mathbf{x}) = \eta^h(\mathbf{x})$ and $\gamma(\mathbf{x}) = \gamma^h(\mathbf{x})$. Indeed, $y^h(t, \mathbf{x})$ and $p^h(t, \mathbf{x})$ are approximate solutions of the problem (3). It is noted that by increasing m, the accuracy of the obtained $y^h(t, \mathbf{x})$ and $p^h(t, \mathbf{x})$ is improved but the computational time is increased. In the POD method by combining the basis functions $\{\varphi_1(\mathbf{x}), \ldots, \varphi_m(\mathbf{x})\}$, another orthonormal set of functions $\{\psi_1(\mathbf{x}), \ldots, \psi_\ell(\mathbf{x})\}$ are constructed that well express the main properties of the underlying problem and in addition, ℓ is small in comparison with m.

To construct $\{\psi_i\}_{i=1}^{\ell}$, we need information about the solution of the problem (3). For this purpose, at first $0 \le t_1 < t_2 < \cdots < t_n \le T$ are selected as a partition of $[0, T]$, and we set

$$y_j^h(\mathbf{x}) = y^h(t_j, \mathbf{x}), \qquad p_j^h(\mathbf{x}) = p^h(t_j, \mathbf{x}), \qquad j = 1, \ldots, n. \tag{12}$$

For $j = 1, \ldots, n$, $y_j^h(\mathbf{x})$ and $p_j^h(\mathbf{x})$ are called snapshots of the state and adjoint functions, respectively. Note that $y^h(t_j, \mathbf{x})$ and $p^h(t_j, \mathbf{x})$ are the solutions obtained by finite element method and from (6), we have

$$y_j^h(\mathbf{x}) = \sum_{i=1}^{m} \mathrm{y}_i(t_j)\varphi_i(\mathbf{x}), \qquad p_j^h(\mathbf{x}) = \sum_{i=1}^{m} \mathrm{p}_i(t_j)\varphi_i(\mathbf{x}). \tag{13}$$

Generally, the choice of snapshots is a critical question that arises in the POD methods. For example, in parametric problems with a moderate or large number of parameters, we require sophisticated approaches, such as greedy sampling technique, for generating snapshots.[25] However, the considered problem (1) is a nonparametric and time-dependent problem and according to Benner et al,[25] equidistant $t_i,\ i = 1, \dots, n$ can be simply selected, provided that a sufficiently high number of snapshots, ie, n, is choosen.

Let $\mathcal{V}$ be the spanned space by all snapshots, ie,

$$\mathcal{V} = \text{span}\{y_1^h, \dots, y_n^h, p_1^h, \dots, p_n^h\} \subseteq X^h \subset L^2(\Omega).$$

Clearly, $\mathcal{V}$ is a finite dimensional subspace of X^h and if $d = \dim(\mathcal{V})$ then $d \leq \min\{m, 2n\}$. Thus, there are orthonormal basis sets with cardinality d for space $\mathcal{V}$. In the POD technique, the aim is to find a complete orthonormal basis $\{\psi_i\}_{i=1}^d$, such that the mean square error between the snapshots and their orthonormal projection onto the first ℓ functions $\{\psi_i\}_{i=1}^\ell$ is minimized. Mathematically speaking, in the POD method, the POD basis functions $\{\psi_i\}_{i=1}^\ell$ are obtained from the following optimization problem

$$\begin{cases} \min\limits_{\psi_1,\dots,\psi_\ell} \sum\limits_{j=1}^{n} \alpha_j \left\| y_j^h - \sum\limits_{i=1}^{\ell} \langle y_j^h, \psi_i\rangle \psi_i \right\|_{L^2(\Omega)}^2 + \alpha_j \left\| p_j^h - \sum\limits_{i=1}^{\ell} \langle p_j^h, \psi_i\rangle \psi_i \right\|_{L^2(\Omega)}^2, \\ \text{s.t.} \quad \langle \psi_i, \psi_j\rangle = \delta_{ij}, \quad 1 \leq i, j \leq \ell, \end{cases} \tag{14}$$

where, $\alpha_j,\ j = 1, \dots, n$ are the trapezoidal weights,[24,33] ie,

$$\alpha_1 = \frac{t_2 - t_1}{2}, \quad \alpha_j = \frac{t_{j+1} - t_{j-1}}{2}, \text{ for } j = 2, \dots, n-1, \quad \alpha_n = \frac{t_n - t_{n-1}}{2}.$$

The solution set $\{\psi_i\}_{i=1}^\ell$ to (14) is called a POD basis of rank ℓ. We note that this optimization problem is not a classical optimization problem. However, in Studinger and Volkwein,[50] it is proved that the solution of the optimization problem (14) is obtained by solving the following eigenvalue problem:

$$\mathcal{R}^n \psi_i := \sum_{j=1}^{n} \alpha_j \langle y_j^h, \psi_i\rangle y_j^h + \alpha_j \langle p_j^h, \psi_i\rangle p_j^h = \lambda_i \psi_i, \qquad i = 1, \dots, \ell. \tag{15}$$

Note that $\mathcal{R}^n : X^h \to \mathcal{V}$ is a linear, bounded, nonnegative, compact, and selfadjoint operator.[50] Thus, there exist nonnegative eigenvalues $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_d > 0$ and an associated orthonormal set $\{\psi_i\}_{i=1}^d$ of eigenfunctions that solve (15). It is proved in previous works[24,33] that the set of first ℓ eigenfunctions $\psi_1, \dots, \psi_\ell$ is the solution of the problem (14).

For solving (15), by noting that $\psi_j \in \mathcal{V} = \text{span}\{\varphi_1, \dots, \varphi_m\}$, we can expand $\{\psi_j\}_{j=1}^\ell$ based on $\{\varphi_1, \dots, \varphi_m\}$ as follows:

$$\psi_j(\mathbf{x}) = \sum_{i=1}^{m} q_{ij} \varphi_i(\mathbf{x}), \qquad j = 1, \dots, \ell. \tag{16}$$

Let $\mathbf{q} \in \mathbb{R}^m$ and $\mathbf{Q} \in \mathbb{R}^{m\times\ell}$ be defined as follows:

$$\mathbf{q}_j = [q_{1j}, \dots, q_{mj}]^T, \qquad \mathbf{Q} = \begin{bmatrix} q_{11} & \cdots & q_{1\ell} \\ \vdots & & \vdots \\ q_{m1} & \cdots & q_{m\ell} \end{bmatrix}. \tag{17}$$

Note that, $\mathbf{q}_j$ is the jth column of $\mathbf{Q}$ and from (16), the components of $\mathbf{q}_j$ can be interpreted as the finite element coefficients of $\psi_j(\mathbf{x})$, for $j = 1, \dots, \ell$. By substituting (13) and (16) in the eigenvalue problem (15) and by using of delta Kronecker property of $\{\varphi_i(\mathbf{x})\}_{i=1}^m$ at the nodes of spatial discretization, finally, the eigenvalue problems (15) are converted to the following matrix eigenvalue problem:

$$\mathbf{Z}\mathbf{D}\mathbf{Z}^\mathbf{T}\mathbf{M}\mathbf{q}_j = \lambda_j \mathbf{q}_j, \qquad j = 1, \dots, \ell, \tag{18}$$

where

$$\mathbf{Z} = [\mathbf{y}(t_1), \dots, \mathbf{y}(t_n), \mathbf{p}(t_1), \dots, \mathbf{p}(t_n)], \tag{19a}$$

$$\mathbf{D} = \text{diag}(\alpha_1, \dots, \alpha_n, \alpha_1, \dots, \alpha_n), \tag{19b}$$

and $\mathbf{M}$ is introduced in (9). To efficiently solve the above eigenvalue problem and some numerical consideration, we refer to Studinger and Volkwein.[50]

By solving the eigenvalue problem (18), the coefficients q_{ij} are obtained and by using them in (16), the POD basis functions are obtained.

3.1.1 | On choosing the value of ℓ

In the POD method, to reduce the computational cost, the number of the POD basis functions must be chosen several orders lower than the number of finite element basis functions, ie, $\ell \ll m$. However, this decreases the accuracy. The crucial question is how to choose ℓ? Indeed, we need to strike a balance between accuracy and efficiency. For this purpose, following Xiao et al,[51] we consider the following ratio of the eigenvalues:

$$\varepsilon(\ell) = \frac{\Sigma_{i=1}^{\ell} \lambda_i}{\Sigma_{i=1}^{d} \lambda_i}. \tag{20}$$

By this ratio, we can measure the quality of the approximation. The value of $\varepsilon(\ell)$ will tend to 1 as ℓ increased to d. So, this value can be used to provide an appropriate choice of ℓ.[52] Indeed, we have to choose the smallest ℓ such that $\varepsilon(\ell)$ is still sufficiently close to 1 (eg, $\varepsilon(\ell) \simeq 0.99$).

3.2 | The presented POD Galerkin projection method

Suppose that $\{\psi_i\}_{i=1}^{\ell}$ be the POD basis functions that are obtained by the mentioned procedure. We use this basis to approximate the state y and adjoint p as follows:

$$y^{\ell}(t,\mathbf{x}) = \sum_{j=1}^{\ell} \mathrm{y}_j^{\ell}(t)\psi_j(\mathbf{x}), \qquad p^{\ell}(t,\mathbf{x}) = \sum_{j=1}^{\ell} \mathrm{p}_j^{\ell}(t)\psi_j(\mathbf{x}), \tag{21}$$

as well as the functions $\eta(\mathbf{x})$ and $\gamma(\mathbf{x})$ are approximated as follows:

$$\eta^{\ell}(\mathbf{x}) = \sum_{j=1}^{\ell} \eta_j^{\ell}\psi_j(\mathbf{x}), \qquad \gamma^{\ell}(\mathbf{x}) = \sum_{j=1}^{\ell} \gamma_j^{\ell}\psi_j(\mathbf{x}). \tag{22}$$

Moreover, we set

$$\mathbf{y}^{\ell}(t) = \left[\mathrm{y}_1^{\ell}(t), \ldots, \mathrm{y}_{\ell}^{\ell}(t)\right]^T, \qquad \mathbf{p}^{\ell}(t) = \left[\mathrm{p}_1^{\ell}(t), \ldots, \mathrm{p}_{\ell}^{\ell}(t)\right]^T, \tag{23a}$$

$$\boldsymbol{\eta}^{\ell} = \left[\eta_1^{\ell}, \ldots, \eta_{\ell}^{\ell}\right]^T, \qquad \boldsymbol{\gamma}^{\ell} = \left[\gamma_1^{\ell}, \ldots, \gamma_{\ell}^{\ell}\right]^T. \tag{23b}$$

By substituting (21) and (22) in the weak form of (3), finally, we obtain the following IVP:

$$\ddot{\mathbf{y}}^{\ell}(t) + \mathbf{K}^{\ell}\mathbf{y}^{\ell}(t) + \mathbf{N}_1^{\ell}(t) = \mathbf{f}^{\ell}(t) - \mathbf{p}^{\ell}(t), \tag{24a}$$

$$\ddot{\mathbf{p}}^{\ell}(t) + \mathbf{K}^{\ell}\mathbf{p}^{\ell}(t) + \mathbf{N}_2^{\ell}(t) = \kappa_1\left[\mathbf{y}^{\ell}(t) - \mathbf{z}^{\ell}(t)\right], \tag{24b}$$

$$\mathbf{y}^{\ell}(0) = \mathbf{g}^{\ell}, \qquad \dot{\mathbf{y}}^{\ell}(0) = \mathbf{h}^{\ell}, \tag{24c}$$

$$\mathbf{p}^{\ell}(0) = \boldsymbol{\eta}^{\ell}, \qquad \dot{\mathbf{p}}^{\ell}(0) = \boldsymbol{\gamma}^{\ell}, \tag{24d}$$

where

$$\left[\mathbf{K}^{\ell}\right]_{ij} = \langle\nabla\psi_j, \nabla\psi_i\rangle, \qquad \left[\mathbf{f}^{\ell}(t)\right]_i = \langle\psi_i, f\rangle, \tag{25a}$$

$$\left[\mathbf{z}^{\ell}(t)\right]_i = \langle\psi_i, z\rangle, \qquad [\mathbf{g}^{\ell}]_i = \langle\psi_i, g\rangle, \qquad [\mathbf{h}^{\ell}]_i = \langle\psi_i, h\rangle, \tag{25b}$$

$$\left[\mathbf{N}_1^{\ell}(t)\right]_i = \langle\psi_i, \mathcal{N}_1(y^{\ell})\rangle, \qquad \left[\mathbf{N}_2^{\ell}(t)\right]_i = \langle\psi_i, \mathcal{N}_2(y^{\ell}, p^{\ell})\rangle. \tag{25c}$$

In the same way, the terminal condition (4) is discretized to the following:

$$\mathrm{F}^{\ell}(\boldsymbol{\eta}^{\ell}, \boldsymbol{\gamma}^{\ell}) = \begin{bmatrix} \mathbf{p}^{\ell}(T) \\ \dot{\mathbf{p}}^{\ell}(T) + \kappa_2\left(\mathbf{y}^{\ell}(T) - \mathbf{w}^{\ell}\right) \end{bmatrix} = 0, \tag{26}$$

where

$$[\mathbf{w}^{\ell}]_i = \langle\psi_i, w\rangle. \tag{27}$$

In short, by the POD Galerkin projection method, ISP is discretized to the following finite shooting problem

POD-shooting problem (POD-SP): *Find* $\boldsymbol{\eta}^{\ell}, \boldsymbol{\gamma}^{\ell} \in \mathbb{R}^{\ell}$ *such that the solution of the IVP (24) satisfies the terminal condition* (26).

To assemble IVP (24), we need to compute the matrix and vectors (25) and (27). However, note that obtaining the matrix and vectors in (25) and (27) by the inner product is not straightforward. Since in contrast to the finite element basis functions, POD basis functions typically have global support and are not cardinal. However, it can be seen that there exist the following relation between the finite element matrices (9) and (11) and matrices in (25a), (25b), and (27):

$$\mathbf{K}^{\ell} = \mathbf{Q}^T\mathbf{K}\mathbf{Q}, \qquad \mathbf{f}^{\ell}(t) = \mathbf{Q}^T\mathbf{f}(t), \qquad \mathbf{z}^{\ell}(t) = \mathbf{Q}^T\mathbf{z}(t), \tag{28a}$$

$$\mathbf{g}^{\ell} = \mathbf{Q}^T\mathbf{g}, \qquad \mathbf{h}^{\ell} = \mathbf{Q}^T\mathbf{h}, \qquad \mathbf{w}^{\ell} = \mathbf{Q}^T\mathbf{w}, \tag{28b}$$

where $\mathbf{Q}$ is defined in (17). Thanks to the above relations, the matrix $\mathbf{K}^{\ell}$ and vectors $\mathbf{f}^{\ell}(t)$, $\mathbf{z}^{\ell}(t)$, $\mathbf{g}^{\ell}$, $\mathbf{h}^{\ell}$, and $\mathbf{w}^{\ell}$ can be obtained by simple matrix multiplication. However, for calculating $\mathbf{N}_1^{\ell}(t)$ and $\mathbf{N}_2^{\ell}(t)$, we need additional considerations. In general, the inner products (25c), for computing $\mathbf{N}_1^{\ell}(t)$ and $\mathbf{N}_2^{\ell}(t)$, cannot be expressed in terms of finite element matrices (9) and (11) and are commonly approximated by numerical quadratures such as Gauss quadrature. However, in Wang,[53] a more efficient approach for computation of the inner products in (25c) is proposed. This approach uses an interpolation operator $\mathcal{I}^h : C(\Omega) \to X^h$ relative to the nodes of finite element discretization and approximates $\mathcal{N}_1$ and $\mathcal{N}_2$ as follows:

$$\mathcal{I}^h\mathcal{N}_1\left(y\left(t,\mathbf{x}\right)\right) = \sum_{i=1}^{m} \mathcal{N}_1\left(y\left(t,\mathbf{x}_i\right)\right)\varphi_i(\mathbf{x}), \tag{29}$$

$$\mathcal{I}^h\mathcal{N}_2\left(y\left(t,\mathbf{x}\right),p\left(t,\mathbf{x}\right)\right) = \sum_{i=1}^{m} \mathcal{N}_2\left(y\left(t,\mathbf{x}_i\right),p\left(t,\mathbf{x}_i\right)\right)\varphi_i(\mathbf{x}). \tag{30}$$

This approximation also is known as the product approximation technique[54] or group finite element method.[55] For evaluation of the inner products in (25c), the nonlinear functions $\mathcal{N}_i$ are replaced by $\mathcal{I}^h\mathcal{N}_i$ for $i = 1, 2$. For instance, the ith row of nonlinear terms $\mathbf{N}_1^{\ell}(t)$ is approximated as follows:

$$\begin{aligned}
\left[\mathbf{N}_1^{\ell}(t)\right]_i &\approx \left\langle \psi_i(\mathbf{x}), \mathcal{I}^h\mathcal{N}_1(y^{\ell}(t,\mathbf{x}))\right\rangle \\
&= \left\langle \sum_{k=1}^{m} q_{ki}\varphi_k(\mathbf{x}), \sum_{j=1}^{m} \mathcal{N}_1(y^{\ell}(t,\mathbf{x}_j))\varphi_j(\mathbf{x}) \right\rangle \\
&= \sum_{k=1}^{m}\sum_{j=1}^{m} q_{ki}\left[\mathcal{N}_1(\mathbf{Q}\mathbf{y}^{\ell}(t))\right]_j \left\langle \varphi_k(\mathbf{x}), \varphi_j(\mathbf{x})\right\rangle \\
&= \mathbf{q}_i^T\mathbf{M}\mathcal{N}_1(\mathbf{Q}\mathbf{y}^{\ell}(t)),
\end{aligned} \tag{31}$$

so we have the following:

$$\mathbf{N}_1^{\ell}(t) \approx \mathbf{Q}^T\mathbf{M}\mathcal{N}_1(\mathbf{Q}\mathbf{y}^{\ell}(t)). \tag{32}$$

Similarly, we can get the following:

$$\mathbf{N}_2^{\ell}(t) \approx \mathbf{Q}^T\mathbf{M}\mathcal{N}_2(\mathbf{Q}\mathbf{y}^{\ell}(t), \mathbf{Q}\mathbf{p}^{\ell}(t)). \tag{33}$$

In summary, by using (28), (32), and (33), we can assemble IVP (24) to use it in the POD-SP. If $(\boldsymbol{\eta}^{*\ell}, \boldsymbol{\gamma}^{*\ell})$ be the solution of POD-SP, then let $\mathrm{y}_j^{*\ell}(t)$ and $\mathrm{p}_j^{*\ell}(t)$ be the obtained solutions of IVP (24) with $\boldsymbol{\eta}^{\ell} = \boldsymbol{\eta}^{*\ell}$ and $\boldsymbol{\gamma}^{\ell} = \boldsymbol{\gamma}^{*\ell}$. Now, from (21) and (2i), we can obtain the following approximations for the control and state functions

$$u(t,\mathbf{x}) \simeq -\sum_{j=1}^{\ell} \mathrm{p}_j^{*\ell}(t)\psi_j(\mathbf{x}), \tag{34}$$

$$y(t,\mathbf{x}) \simeq \sum_{j=1}^{\ell} \mathrm{y}_j^{*\ell}(t)\psi_j(\mathbf{x}). \tag{35}$$

Finally, the POD method presented in this section is summarized in Algorithm 2.

4 | DISCRETE EMPIRICAL INTERPOLATION METHOD(DEIM)

In Algorithm 2, a POD method for solving the problem (1) is presented. In line 8 of this algorithm, for solving $\mathrm{EQF}(\cdot,\cdot) = \mathbf{0}$, a nonlinear equation solver is used, which generally uses an iterative method. In each iteration of the solver, function EQF is evaluated at least one time, so the complexity of the evaluation of this function is crucial in our method.

In EQF, the IVP (24) is solved by an appropriate time stepping scheme and in the nonlinear cases, in each time step, we need to compute $\mathbf{N}_1^\ell(t)$ and $\mathbf{N}_2^\ell(t)$. For obtaining the complexity of the evaluation of $\mathbf{N}_1^\ell(t)$, let the complexity for evaluating the nonlinear function $\mathcal{N}_1$ with q components be $\mathcal{O}(\varrho(q))$, where ϱ is some function of q. Thus, in each time step, the complexity for computing $\mathbf{N}_1^\ell(t)$ by (32) is $\mathcal{O}(\varrho(m) + 4m\ell)$. This means that the complexity of evaluation $\mathbf{N}_1^\ell(t)$ depends on m. Moreover, the complexity of evaluation of $\mathbf{N}_2^\ell(t)$ depends on m too.

By noting that m is the number of the finite element basis functions and generally is a large number, thus in the presence of the nonlinear term $\mathcal{N}_1$, the evaluation of function EQF is very time-consuming. As a result, the proposed Algorithm 2 is less efficient in the presence of nonlinearity. To overcome this drawback, we use DEIM, which is an efficient technique, with low computational complexity, for approximating the nonlinear functions.[45,56]

Algorithm 2 : POD-indirect shooting (POD-IS) algorithm

Inputs:

$m \in \mathbb{N}$ as the number of finite element basis.
$n \in \mathbb{N}$ as the number of snapshots.
$\boldsymbol{\eta}^s, \boldsymbol{\gamma}^s \in \mathbb{R}^m$ as initial conditions for IVP (8) in order to generate snapshots.
$\epsilon > 0$ for estimating the appropriate number of the POD basis.

1: **Begin**
2: Compute matrices $\mathbf{M}, \mathbf{K}$ and vectors $\mathbf{f}(t), \mathbf{z}(t), \mathbf{g}, \mathbf{h}, \mathbf{w}$ as (9) and (11).
3: Solve IVP (8) with initial conditions $\boldsymbol{\eta}^s$, $\boldsymbol{\gamma}^s$ and denote the solutions by $\mathbf{y}(t)$ and $\mathbf{p}(t)$.
4: Construct the snapshot matrix $\mathbf{Z}$ and diagonal matrix $\mathbf{D}$ as (19).
5: Set $d = \text{rank}(\mathbf{Z})$, solve the matrix eigenvalue problem (18) and obtain $\mathbf{q}_1, \dots, \mathbf{q}_d$.
6: Use (20) to find the smallest integer number ℓ such that $\varepsilon(\ell) < 1 - \epsilon$.
7: Use (28) to construct matrix $\mathbf{K}^\ell$ and vectors $\mathbf{f}^\ell(t), \mathbf{z}^\ell(t), \mathbf{g}^\ell, \mathbf{h}^\ell, \mathbf{w}^\ell$.
8: Solve the system of equations $\text{EQF}(\boldsymbol{\eta}^\ell, \boldsymbol{\gamma}^\ell) = \mathbf{0}$, by a solver and denote the solution by $(\boldsymbol{\eta}^{*\ell}, \boldsymbol{\gamma}^{*\ell})$.
9: Solve IVP (24) with $(\boldsymbol{\eta}^\ell, \boldsymbol{\gamma}^\ell) = (\boldsymbol{\eta}^{*\ell}, \boldsymbol{\gamma}^{*\ell})$ and using (32) and (33). Denote the solutions by $\mathrm{y}_j^{*\ell}(t)$ and $\mathrm{p}_j^{*\ell}(t)$.
10: Use $\mathrm{y}_j^{*\ell}(t)$ and $\mathrm{p}_j^{*\ell}(t)$ in (34) and (35) to generate an approximation for the solution of the problem (1).
11: **End Algorithm**

Function $\mathbf{r} = \text{EQF}(\mathbf{a}, \mathbf{b})$
Global: $\mathbf{M}, \mathbf{Q}, \mathbf{K}^\ell, \mathbf{f}^\ell(t), \mathbf{z}^\ell(t), \mathbf{g}^\ell, \mathbf{h}^\ell, \mathbf{w}^\ell$.

1: Solve IVP (24) to obtain the approximate values of $\mathbf{p}^\ell(T)$, $\dot{\mathbf{p}}^\ell(T)$ and $\mathbf{y}^\ell(T)$, where
- ▶ **a** and **b** are considered as initial values for $\mathbf{p}^\ell(0)$ and $\dot{\mathbf{p}}^\ell(0)$,
- ▶ For evaluation of $\mathbf{N}_1^\ell(t)$ and $\mathbf{N}_2^\ell(t)$, Equations (32) and (33) are used.

2: **return** $\mathbf{r} = \begin{bmatrix} \mathbf{p}^\ell(T) \\ \dot{\mathbf{p}}^\ell(T) + \kappa_2 \left(\mathbf{y}^\ell(T) - \mathbf{w}^\ell\right) \end{bmatrix}$.

End Function

In the following, we use DEIM to present efficient formulas for approximating $\mathbf{N}_1^\ell(t)$ and $\mathbf{N}_2^\ell(\mathrm{t})$. For this purpose, we use the POD technique to construct a new basis for approximating the nonlinear functions $\mathcal{N}_1$ and $\mathcal{N}_2$. We define the following:

$$\mathbf{n}_1(t) = \mathcal{N}_1(\mathbf{Q}\mathbf{y}^\ell(t)). \tag{36}$$

For approximation $\mathbf{n}_1(t)$, at first, we construct the POD basis $\{\boldsymbol{\phi}_1^1, .., \boldsymbol{\phi}_{r_1}^1\} \subset \mathbb{R}^m$ corresponding to the following snapshot space:

$$\mathcal{V}_1 = \text{span}\{\mathcal{N}_1(\mathbf{y}(t_1)), \dots, \mathcal{N}_1(\mathbf{y}(t_n))\} \subset \mathbb{R}^m, \tag{37}$$

where $r_1 \ll m$ denotes the size of the reduced representation of $\mathbf{n}_1(t)$. For constructing the POD basis $\{\boldsymbol{\phi}_1^1, .., \boldsymbol{\phi}_{r_1}^1\}$, similar to Section 3.1, we consider the following optimization problem:

$$\begin{cases} \min\limits_{\boldsymbol{\phi}_1^1, \dots, \boldsymbol{\phi}_{r_1}^1} \sum\limits_{j=1}^{n} \alpha_j \left\| \mathcal{N}_1(\mathbf{y}(t_j)) - \sum\limits_{i=1}^{r_1} \langle \mathcal{N}_1(\mathbf{y}(t_j)), \boldsymbol{\phi}_i^1 \rangle_{\mathbf{M}} \boldsymbol{\phi}_i^1 \right\|_{\mathbf{M}}^2 \\ \text{s.t.} \qquad \langle \boldsymbol{\phi}_i^1, \boldsymbol{\phi}_j^1 \rangle_{\mathbf{M}} = \delta_{ij} \quad 1 \le i, j \le r_1 \end{cases} \tag{38}$$

in which the notations $\langle\cdot,\cdot\rangle_{\mathbf{M}}$ and $\|\cdot\|_{\mathbf{M}}$ are the weighted inner product and the corresponding weighted norm respectively, which are defined as:

$$\langle \mathbf{u}, \mathbf{v}\rangle_{\mathbf{M}} = \mathbf{u}^T\mathbf{M}\mathbf{v} \qquad \|\mathbf{u}\|_{\mathbf{M}} = (\langle \mathbf{u}, \mathbf{u}\rangle_{\mathbf{M}})^{1/2} \qquad \text{for } \mathbf{u}, \mathbf{v} \in \mathbb{R}^m. \tag{39}$$

The optimization problem (38) can be reformulated as the following eigenvalue problem[50,57]:

$$\mathbf{E}_1\hat{\mathbf{D}}\mathbf{E}_1^T\mathbf{M}\boldsymbol{\phi}_i^1 = \lambda_i\boldsymbol{\phi}_i^1, \tag{40}$$

where

$$\mathbf{E}_1 = \left[\mathcal{N}_1(\mathbf{y}(t_1)), \ \dots\ , \mathcal{N}_1(\mathbf{y}(t_n))\right], \tag{41a}$$

$$\hat{\mathbf{D}} = \mathrm{diag}(\alpha_1, \ \dots\ , \alpha_n), \tag{41b}$$

and $\mathbf{M}$ is defined in (9a). By using the constructed POD basis $\{\boldsymbol{\phi}_1^1, .., \boldsymbol{\phi}_{r_1}^1\}$, we can approximate $\mathbf{n}_1(t)$ by the following expansion:

$$\mathbf{n}_1(t) = \Phi_1\mathbf{c}_1(t), \tag{42}$$

where $\Phi_1 = \left[\boldsymbol{\phi}_1^1, .., \boldsymbol{\phi}_{r_1}^1\right] \in \mathbb{R}^{m\times r_1}$ and $\mathbf{c}_1(t) = \left[c_1^1(t), \ \dots\ , c_{r_1}^1(t)\right]^T \in \mathbb{R}^{r_1}$ is the unknown corresponding coefficient vector. To evaluate the approximation (42), we need to obtain $\mathbf{c}_1(t)$. DEIM algorithm selects r_1 rows from the overdetermined system (42) to specify $\mathbf{c}_1(t)$. The row indices $\{\rho_1^1, \ \dots\ , \rho_{r_1}^1\}$, which are used for determining the coefficient vector $\mathbf{c}_1(t)$, are selected inductively from the basis $\boldsymbol{\phi}_1^1, \ \dots\ , \boldsymbol{\phi}_{r_1}^1$ by the DEIM algorithm. For use of this algorithm, we refer to Lass and Volkwein[57] and assume that $\{\rho_1^1, \ \dots\ , \rho_{r_1}^1\}$ are the obtained row indices. We consider corresponding matrix $\mathbf{P}_1$ as follows:

$$\mathbf{P}_1 = [\mathbf{e}_{\rho_1^1}, \ \dots\ , \mathbf{e}_{\rho_{r_1}^1}] \in \mathbb{R}^{m\times r_1}, \tag{43}$$

where $\mathbf{e}_{\rho_i^1}$ is the ρ_i^1th column of the identity matrix $\mathbf{I}_m \in \mathbb{R}^{m\times m}$, $i = 1, \ \dots\ , r_1$. By multiplying (42) with $\mathbf{P}_1^T$, r_1 rows of it are selected and with assumption $\mathbf{P}_1^T\Phi_1$ is invertible, we have the following:

$$\mathbf{c}_1(t) = (\mathbf{P}_1^T\Phi_1)^{-1}\mathbf{P}_1^T\mathbf{n}_1(t) = (\mathbf{P}_1^T\Phi_1)^{-1}\mathbf{P_1}^T\mathcal{N}_1(\mathbf{Q}\mathbf{y}^\ell(t)). \tag{44}$$

We note that the nonlinear functions $\mathcal{N}_1$ and $\mathcal{N}_2$ are evaluated componentwise at their input vectors, thus the matrix $\mathbf{P}_1^T$ can be moved into the nonlinear function:

$$\mathbf{c}_1(t) = (\mathbf{P}_1^T\Phi_1)^{-1}\mathcal{N}_1(\mathbf{P}_1^T\mathbf{Q}\mathbf{y}^\ell(t)). \tag{45}$$

Now, we have everything to approximate $\mathbf{N}_1^\ell(t)$. By (32) and (42), we have

$$\mathbf{N}_1^\ell(t) = \mathbf{Q}^T\mathbf{M}\Phi_1\mathbf{c}_1(t), \tag{46}$$

and from (45), we get

$$\mathbf{N}_1^\ell(t) \approx \mathbf{N}_1^{r_1}(t) = \mathbf{A}_1\mathcal{N}_1\left(\mathbf{B}_1\mathbf{y}^\ell(t)\right), \tag{47}$$

where

$$\mathbf{A}_1 = \mathbf{Q}^T\mathbf{M}\Phi_1(\mathbf{P}_1^T\Phi_1)^{-1} \in \mathbb{R}^{\ell\times r_1} \quad \text{and} \quad \mathbf{B}_1 = \mathbf{P}_1^T\mathbf{Q} \in \mathbb{R}^{r_1\times\ell}. \tag{48}$$

We note that matrices $\mathbf{A}_1$ and $\mathbf{B}_1$ can be precomputed and the nonlinear function $\mathcal{N}_1$ at each time step has to be computed at a r_1-dimensional vector, that $r_1 \ll m$. More precisely, the corresponding computational complexity of the nonlinear term $\mathbf{N}_1^{r_1}(t)$ at each time step of IVP (24) is only $\mathcal{O}(\varrho(r_1) + 4r_1\ell)$. For approximation $\mathbf{N}_2^\ell(t)$ in (24), we follow a similar approach as that for $\mathbf{N}_1^\ell(t)$ and finally, we get:

$$\mathbf{N}_2^\ell(t) \approx \mathbf{N}_2^{r_2}(t) = \mathbf{A}_2\mathcal{N}_2\left(\mathbf{B}_2\mathbf{y}^\ell(t), \mathbf{B}_2\mathbf{p}^\ell(t)\right), \tag{49}$$

where

$$\mathbf{A}_2 = \mathbf{Q}^T\mathbf{M}\Phi_2(\mathbf{P}_2^T\Phi_2)^{-1} \in \mathbb{R}^{\ell\times r_2} \quad \text{and} \quad \mathbf{B}_2 = \mathbf{P}_2^T\mathbf{Q} \in \mathbb{R}^{r_2\times\ell}. \tag{50}$$

In summary, for using the DEIM in the POD indirect shooting method, we need to use the DEIM approximations (47) and (49) instead of approximations (32) and (33). Thus, the shooting problem with POD/DEIM discretization can be written as follows:

- **POD/DEIM-shooting problem (POD/DEIM-SP):** *Find* $\boldsymbol{\eta}^\ell, \boldsymbol{\gamma}^\ell \in \mathbb{R}^\ell$ *such that the solutions* $\mathbf{y}^\ell(t), \mathbf{p}^\ell(t) : [0, T] \to \mathbb{R}^\ell$ *of the IVP (24), by considering approximations* (47) *and* (49), *satisfy the terminal condition* (26).

As a result, we have a POD/DEIM indirect shooting method whose computational complexity is independent of the full dimension m. For the reader convenience, we summarize the POD/DEIM indirect shooting method in Algorithm 3.

Algorithm 3 : POD/DEIM-indirect shooting (POD/DEIM-IS) algorithm

Input:

$m \in \mathbb{N}$ as the number of finite element basis.
$n \in \mathbb{N}$ as the number of snapshots.
$\boldsymbol{\eta}^s, \boldsymbol{\gamma}^s \in \mathbb{R}^m$ as initial conditions for IVP (8) to generate snapshots.
$\epsilon_0 > 0$ for estimating the appropriate number of POD basis.
$\epsilon_1 > 0$ and $\epsilon_2 > 0$ for estimating the appropriate number of DEIM basis.

Begin
Compute matrices $\mathbf{M}, \mathbf{K}$ and vectors $\mathbf{f}(t), \mathbf{z}(t), \mathbf{g}, \mathbf{h}, \mathbf{w}$ as (9) and (11).
Solve IVP (8) by the given initial conditions $\boldsymbol{\eta}^s$, $\boldsymbol{\gamma}^s$ and call the solution as $\mathbf{y}(t)$ and $\mathbf{p}(t)$.
Construct the snapshot matrix $\mathbf{Z}$ and diagonal matrix $\mathbf{D}$ as (19).
Construct the snapshot matrix $\mathbf{E}_1$ and diagonal matrix $\hat{\mathbf{D}}$ as (41).
Construct the snapshot matrix $\mathbf{E}_2 = \left[\mathcal{N}_2(\mathbf{y}(t_1), \mathbf{p}(t_1)), \dots, \mathcal{N}_2(\mathbf{y}(t_n), \mathbf{p}(t_n))\right]$.
Set $d = \text{rank}(\mathbf{Z})$, solve the matrix eigenvalue problem (18) and obtain $\mathbf{q}_1, \dots, \mathbf{q}_d$.
Set $d_1 = \text{rank}(\mathbf{E}_1)$, solve the matrix eigenvalue problem (40) and obtain $\boldsymbol{\phi}_1^1, .., \boldsymbol{\phi}_{d_1}^1$.
Set $d_2 = \text{rank}(\mathbf{E}_2)$, solve the matrix eigenvalue problem $\mathbf{E}_2\hat{\mathbf{D}}\mathbf{E}_2^T\mathbf{M}\boldsymbol{\phi}_i^2 = \lambda_i\boldsymbol{\phi}_i^2$ and obtain $\boldsymbol{\phi}_1^2, .., \boldsymbol{\phi}_{d_2}^2$.
Use (20) to find the smallest integer numbers ℓ, r_1 and r_2 such that $\varepsilon(\ell) < 1 - \epsilon$, $\varepsilon(r_1) < 1 - \epsilon_1$ and $\varepsilon(r_2) < 1 - \epsilon_2$.
Set $\mathbf{Q} = [\mathbf{q}_1, \dots, \mathbf{q}_\ell]$, $\boldsymbol{\Phi}_1 = \left[\boldsymbol{\phi}_1^1, .., \boldsymbol{\phi}_{r_1}^1\right]$ and $\boldsymbol{\Phi}_2 = \left[\boldsymbol{\phi}_1^2, .., \boldsymbol{\phi}_{r_2}^2\right]$.
Determine row indices $\{\rho_1^1, \dots, \rho_{r_1}^1\}$ for $\{\phi_i^1\}_{i=1}^{r_1}$ and $\{\rho_1^2, \dots, \rho_{r_2}^2\}$ for $\{\phi_i^2\}_{i=1}^{r_2}$ by DEIM algorithm.[57]
Compute matrices $\mathbf{A}_1, \mathbf{B}_1$ by (48) and $\mathbf{A}_2, \mathbf{B}_2$ by (50).
Use (28) to construct matrix $\mathbf{K}^\ell$ and vectors $\mathbf{f}^\ell(t), \mathbf{z}^\ell(t), \mathbf{g}^\ell, \mathbf{h}^\ell, \mathbf{w}^\ell$.
Solve the system of equations $\text{EQF}(\boldsymbol{\eta}^\ell, \boldsymbol{\gamma}^\ell) = \mathbf{0}$, by a solver and denote the solution by $(\boldsymbol{\eta}^{*\ell}, \boldsymbol{\gamma}^{*\ell})$.
Solve IVP (24) with $(\boldsymbol{\eta}^\ell, \boldsymbol{\gamma}^\ell) = (\boldsymbol{\eta}^{*\ell}, \boldsymbol{\gamma}^{*\ell})$ and using (47) and (49). Denote the solutions by $\mathrm{y}_j^{*\ell}(t)$ and $\mathrm{p}_j^{*\ell}(t)$.
Use $\mathrm{y}_j^{*\ell}(t)$ and $\mathrm{p}_j^{*\ell}(t)$ in (34) and (35) to generate an approximation for the solution of the problem (1).
End Algorithm

Function $\mathbf{r} = \text{EQF}(\mathbf{a}, \mathbf{b})$
Global: $\mathbf{A}_1, \mathbf{B}_1, \mathbf{A}_2, \mathbf{B}_2, \mathbf{K}^\ell, \mathbf{f}^\ell(t), \mathbf{z}^\ell(t), \mathbf{g}^\ell, \mathbf{h}^\ell, \mathbf{w}^\ell$.
Solve IVP (24) to obtain the approximate values of $\mathbf{p}^\ell(T)$, $\dot{\mathbf{p}}^\ell(T)$ and $\mathbf{y}^\ell(T)$, where
- **a** and **b** are considered as initial values for $\mathbf{p}^\ell(0)$ and $\dot{\mathbf{p}}^\ell(0)$,
- For evaluation of $\mathbf{N}_1^\ell(t)$ and $\mathbf{N}_2^\ell(t)$, Equations 47 and 49 are used.

return $\mathbf{r} = \begin{bmatrix} \mathbf{p}^\ell(T) \\ \dot{\mathbf{p}}^\ell(T) + \kappa_2\left(\mathbf{y}^\ell(T) - \mathbf{w}^\ell\right) \end{bmatrix}$.
End Function

5 | NUMERICAL RESULTS

This section is devoted to illustrating the methods presented above using numerical experiments. We implemented the proposed Algorithms 1, 2, and 3 with MATLAB on a personal computer for four examples. In our implementations, we used the MATLAB PDE TOOLBOX, for the finite element spatial discretization with piecewise linear basis function.[58] Furthermore, the MATLAB function `ode45` was used for solving IVPs and for solving the system of equations, the MATLAB function `fsolve` was used.

The underlying algorithm of `ode45` makes use of 4/5 Runge-Kutta integration with a variable time step. Furthermore, `ode45` controls the error by two parameters `RelTol` and `AbsTol`. By these parameters, we can adjust the relative and absolute error tolerances.[59] In our simulations, the error tolerances are set to $RelTol = 10^{-6}$ and $AbsTol = 10^{-8}$. These considerations guarantee that the relative and absolute errors in satisfying IVPs are less than 10^{-6} and 10^{-8}, respectively.

As we said, we used `fsolve` to solve the nonlinear systems of equations. Using this solver, we can adjust the accuracy of the solution based on the two parameters `TolFun` and `TolX`, where the former specifies the termination tolerance on the function value and the later specifies the termination tolerance on the variables.[60] In our simulations, we continued the iteration of `fsolve` until `TolFun` and `TolX` become less than 10^{-12}.

As numerical examples, we considered 4 examples, where the first two examples are linear with exact solution[3] and the last two examples are nonlinear without exact solution.[61] In the cases that the exact solution is known, to assess the accuracy of the methods, the following relative errors are reported

$$E_m(y) = \frac{\|y^* - y_m\|_{L^2(Q)}}{\|y^*\|_{L^2(Q)}}, \quad E_m(p) = \frac{\|p^* - p_m\|_{L^2(Q)}}{\|p^*\|_{L^2(Q)}},$$

where y^* and p^* are the exact solutions and y_m and p_m are the approximated solutions corresponding to the discretization parameter m.

In addition, to further assess the quality of the obtained solutions, the root mean-square-error (RMSE) and the correlation coefficient (Corr)[62] are reported. For instance, let $\mathbf{x}_i, i = 1, \cdots, m$ be the spatial nodes, t_j be a time level and

$$\mathbf{y}^{*j} = \left[y^*(t_j, \mathbf{x}_1), \cdots, y^*(t_j, \mathbf{x}_m)\right], \quad \hat{\mathbf{y}}^j = \left[y^\ell(t_j, \mathbf{x}_1), \cdots, y^\ell(t_j, \mathbf{x}_m)\right],$$

where y^* is the exact solution and y^ℓ denotes the approximated solutions by the POD-IS method. The RMSE and Corr between the exact state and the obtained state by the POD-IS method at time level j is defined as follows:

$$\mathrm{RMSE}_y(\mathbf{y}^{*j}, \hat{\mathbf{y}}^j) = \frac{\|\mathbf{y}^{*j} - \hat{\mathbf{y}}^j\|_2}{\sqrt{m}},$$

$$\mathrm{Corr}_y\left(\mathbf{y}^{*j}, \hat{\mathbf{y}}^j\right) = \frac{\mathrm{cov}\left(\mathbf{y}^{*j}, \hat{\mathbf{y}}^{*j}\right)}{\sigma_{\mathbf{y}^{*j}} \sigma_{\hat{\mathbf{y}}^j}},$$

where, "cov" denotes covariance and σ denotes standard deviation.

In a similar manner, the RMSE and Corr between the obtained state or adjoint functions by the POD-IS and POD/DEIM-IS at time level j can be defined.

It is added that, in all examples, equidistant $t_i,\ i = 1, \ldots, n$ are considered for generating snapshots in POD technique and DEIM. Moreover, the zero initial conditions $\boldsymbol{\eta}^s = \boldsymbol{\gamma}^s = \mathbf{0}$ are considered for all examples. It is noted that other initial conditions may be considered.

5.1 | Example 1

Let $\Omega = (0, 1)$, $\kappa_2 = 0$, $T = 1$ and $\mathcal{N}_1(y) = 0$. Choose $g(x) = \sin(\pi x)$, $h(x) = 0$, $f = -\kappa_1 \sin(\pi x)(t - T)^2$ and $z = 2\sin(\pi x) + \pi^2 \sin(\pi x)(t - T)^2 + \sin(\pi x)\cos(\pi t)$. The exact solution is $y^*(t, x) = \sin(\pi x)\cos(\pi t)$ and $p^*(t, x) = -\kappa_1 \sin(\pi x)(t - T)^2$.

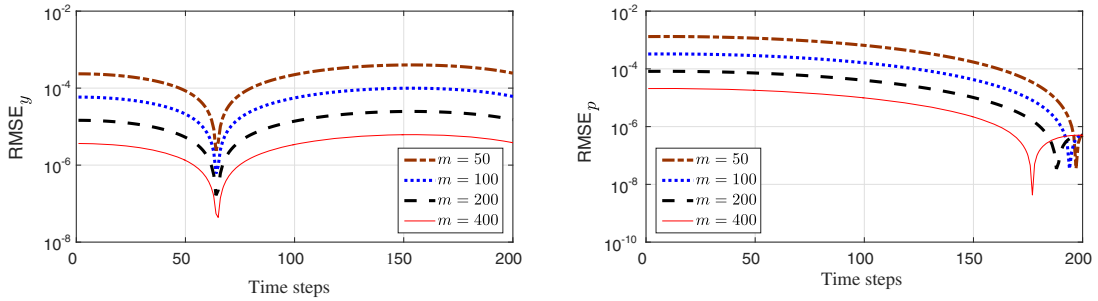
For solving this example, the IS and POD-IS methods are used. In the POD-IS method, the initial conditions $\boldsymbol{\eta}^s = \boldsymbol{\gamma}^s = \mathbf{0}$ are considered and 400 snapshots ($n = 200$) are generated from the obtained solution and POD basis is constructed by selecting $\epsilon = 0.001$.

In Table 1, we compare the relative errors and CPU time in the IS and POD-IS methods, for various values of κ_1 and m. In this table, "POD setup" refers to the computation time in the lines 2 to 7 of Algorithm 2 and "SUM" refers to the overall time of the algorithm.

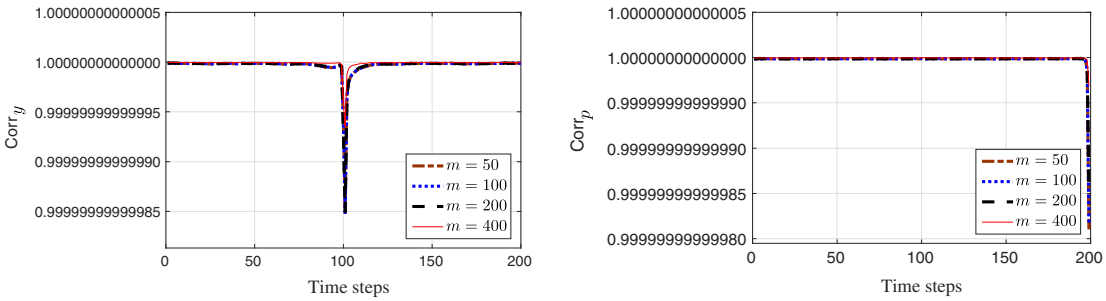
It is seen that the CPU time of the POD-IS method is much time smaller than the CPU time of the IS method, whereas there is no significant difference between the accuracy of two methods. Moreover, as m increases, the CPU time for solving the POD-SP algorithm is decreased. This is because that, by increasing m, the problem is solved more accurately and the

TABLE 1 (Example 1) Comparison of error and CPU time for IS method versus POD-IS method

Parameters of methods		Indirect shooting method			POD indirect shooting method				
		Error		CPU time	Error		CPU time		
κ_1	m	$E_m(y)$	$E_m(p)$	FSP	$E_m(y)$	$E_m(p)$	POD setup	POD-SP	SUM
10	50	5.3e-4	2.5e-4	476	5.3e-4	2.5e-4	0.47	2.9	3.4
10	100	1.3e-4	6.4e-5	9724	1.3e-4	6.4e-5	1.9	2.7	4.6
10	200	3.3e-5	1.6e-5	102897	3.3e-5	1.6e-5	10.6	2.5	13.1
10	400	-	-	-	8.3e-6	4.1e-6	53	2.5	55.5
100	50	6.8e-4	2.9e-4	4055	6.8e-4	2.9e-4	0.36	3.1	3.5
100	100	1.7e-4	7.3e-5	12342	1.7e-4	7.3e-5	1.3	3	4.3
100	200	4.2e-5	1.8e-5	120112	4.2e-5	1.8e-5	6.2	2.8	9
100	400	-	-	-	1.3e-5	7.1e-6	39	2.6	41.6



(A) The RMSE of (left) state y and (right) adjoint p between the exact solution and POD-IS solutions for various values of m (RMSE is plotted on logarithmic scale).



(B) Correlation of coefficients of (left) state y and (right) adjoint p between the exact solution and POD-IS solutions for various values of m.

FIGURE 1 (Example 1 with $\kappa_1 = 10$) The RMSE and Corr between the exact and POD-IS solutions [Colour figure can be viewed at wileyonlinelibrary.com]

resulted snapshots can more accurately capture the solution space of the Equation 3. Consequently, the more suitable POD basis is obtained, which help to speed up the solution time of the POD-SP algorithm.

We note that for $m = 400$, the IS method is not able to solve the problem within an acceptable computational time, whereas the POD-IS method yields an accurate solution in a reasonable CPU time.

For further analyzing, the obtained state and adjoint functions by the POD-IS method, in Figure 1, the RMSE and Corr between the exact and POD-IS solution are plotted for various values of m. It can be seen that by increasing m, the RMSE becomes sufficiently small and the correlation coefficient approaches to 1. This means that, when m increases, the obtained POD-IS solutions tend to closer agreement with the exact solutions.

5.2 | Example 2

Let $\Omega = (0,1)^2$, $\kappa_2 = 0$, $T = 1$, $\mathcal{N}_1(y) = 0$ and

$$g(x_1,x_2) = \sin(\pi x_1)\sin(\pi x_2), \qquad h(x_1,x_2) = \sin(\pi x_1)\sin(\pi x_2),$$
$$f = (1+2\pi^2)e^t \sin(\pi x_1)\sin(\pi x_2) - \kappa_1(t-T)^2 \sin(\pi x_1)\sin(\pi x_2),$$
$$z = (e^t + 2 + 2\pi^2(t-T)^2)\sin(\pi x_1)\sin(\pi x_2).$$

This problem has the following exact solution:

$$y^*(t,\mathbf{x}) = e^t \sin(\pi x_1)\sin(\pi x_2), \text{ and } p^*(t,\mathbf{x}) = -\kappa_1(t-T)^2 \sin(\pi x_1)\sin(\pi x_2).$$

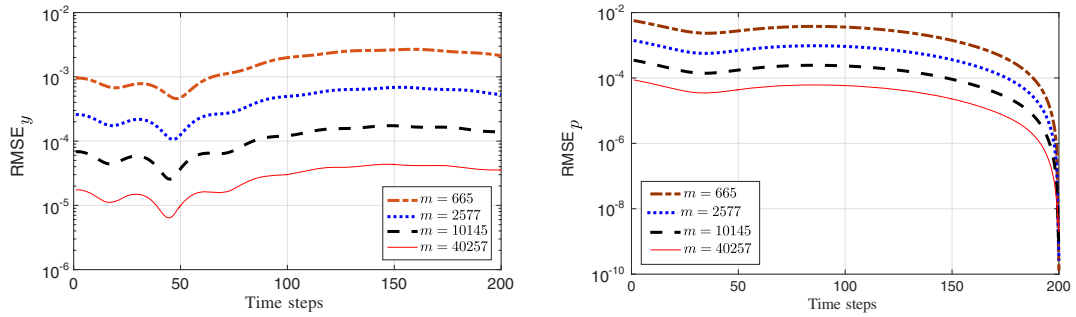
The results of applying the IS and POD-IS methods, with $\boldsymbol{\eta}^s = \boldsymbol{\gamma}^s = \mathbf{0}$, $n = 200$, and $\epsilon = 0.001$, are reported in Table 2.

As we see in this table, for $m = 177$, the POD technique significantly reduces the CPU time. Furthermore, for $m \geq 2577$, the POD-IS method get the solution in a reasonable time but the IS method is not able to solve the problem within an acceptable computational time.

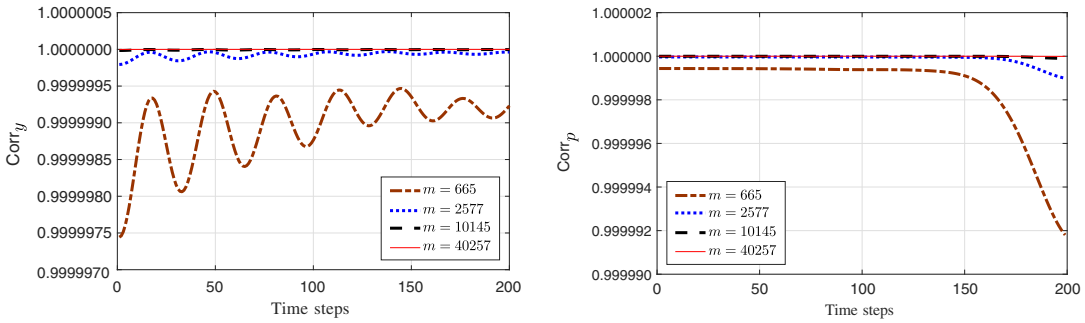
Figure 2 shows the RMSE and Corr between the exact solution and the obtained POD-IS solutions for various values of m. It can be seen that improving the quality of POD basis (by increasing m) leads to improvement in the accuracy of the POD-IS solutions.

TABLE 2 (Example 2) Comparison of error and CPU time for IS method versus POD-IS method

Parameters of Methods		Indirect shooting method			POD indirect shooting method				
		Error		CPU time	Error		CPU time		
κ_1	m	$E_m(y)$	$E_m(p)$	FSP	$E_m(y)$	$E_m(p)$	POD setup	POD-SP	SUM
10	177	8.3e-3	5e-3	1893	8.3e-3	5e-3	3.7	2.5	6.2
10	2577	-	-	-	5.2e-4	3.1e-4	20.4	1.8	22.2
10	10145	-	-	-	1.3e-4	7.7e-5	63	1.7	64.7
10	40257	-	-	-	3.3e-5	1.9e-5	286	1.6	287.6
100	177	1e-2	4.5e-3	2751	1.1e-2	4.6e-3	3.7	2.1	5.8
100	2577	-	-	-	6.8e-4	2.8e-4	20.7	2	22.7
100	10145	-	-	-	1.7e-4	7.1e-5	64	1.8	65.8
100	40257	-	-	-	4.2e-5	1.8e-5	283	1.7	284.7



(A) The RMSE of (left) state y and (right) adjoint p between the exact solution and POD-IS solution by various values of m. (RMSE is plotted on logarithmic scale)



(B) Correlation of coefficients of (left) state y and (right) adjoint p between the exact solution and POD-IS solution for various values of m.

FIGURE 2 (Example 2 with $\kappa_1 = 10$) RMSE and Corr between the exact solution and POD-IS solution [Colour figure can be viewed at wileyonlinelibrary.com]

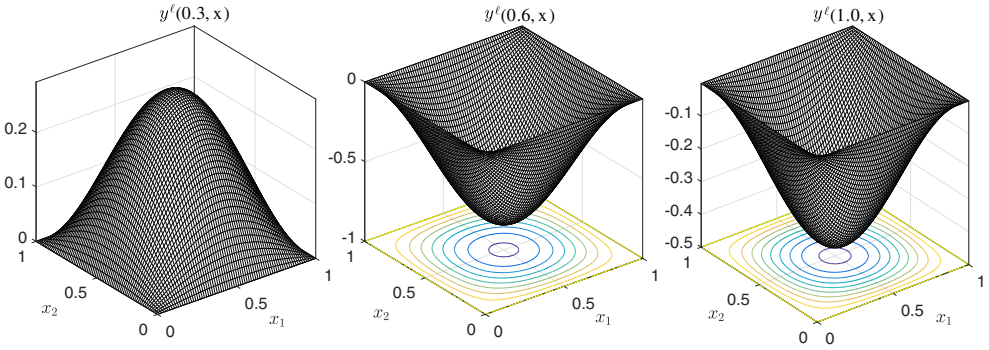
5.3 | Example 3

In the problem (1), let

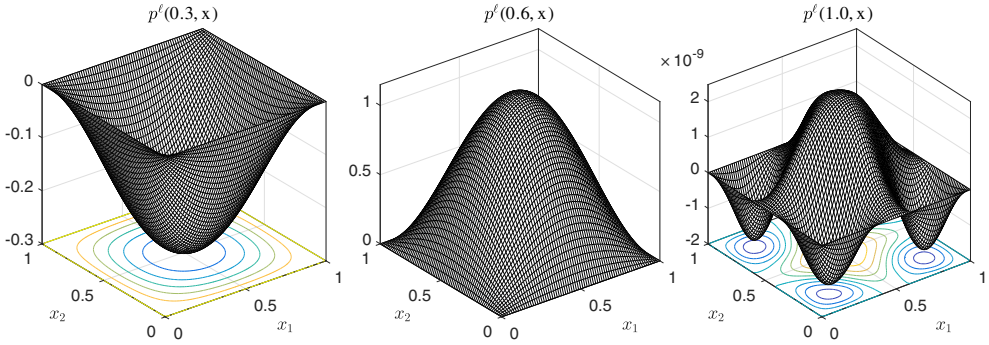
$$\Omega = (0,1)^2,\ T = 1,\ \ \mathcal{N}_1(y) = y^3 - y,\ \ g(x_1,x_2) = \sin(\pi x_1)\sin(\pi x_2),\ \ h = 0,$$
$$z(t,x_1,x_2) = \cos(\pi t)\sin(\pi x_1)\sin(\pi x_2),\ \ w(x_1,x_2) = \cos(\pi)\sin(\pi x_1)\sin(\pi x_2).$$

This problem is nonlinear, and we test the efficiency of the POD/DEIM-IS algorithm in comparison with the POD-IS algorithm. In all simulations, we select the following input parameters

$$n = 200,\ \boldsymbol{\eta}^s = \boldsymbol{\gamma}^s = \mathbf{0} \text{ and } \epsilon = \epsilon_1 = \epsilon_2 = 0.001.$$



(a) The POD-DEIM-IS solution of state variable at time instances $0.3s$, $0.6s$ and $1s$.



(b) The POD-DEIM-IS solution of adjoint variable at time instances $0.3s$, $0.6s$ and $1s$.

FIGURE 3 (Example 3 with $\kappa_1 = \kappa_2 = 10$) The POD/DEIM-IS solution with $m = 2577$ [Colour figure can be viewed at wileyonlinelibrary.com]

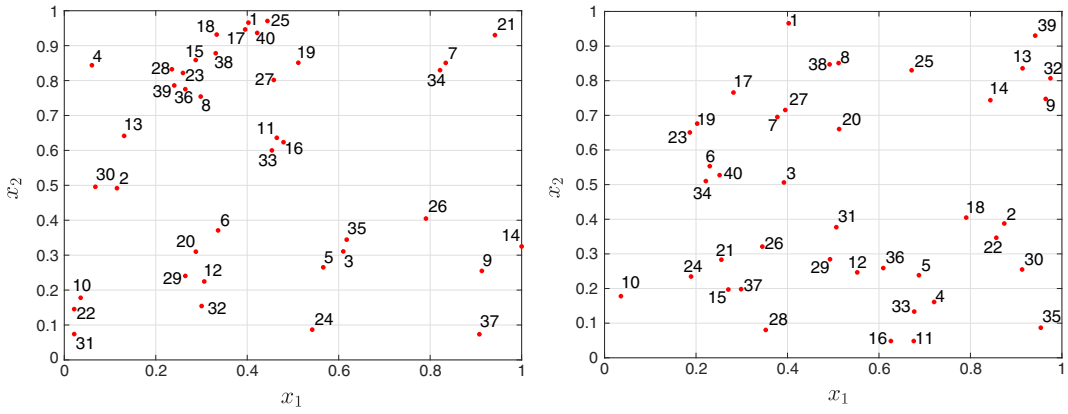
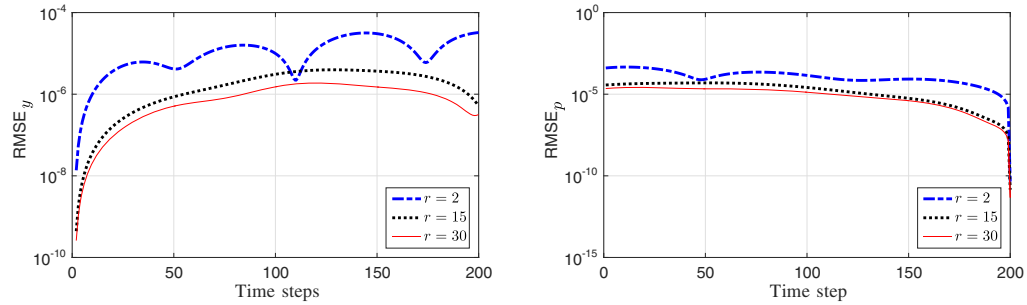


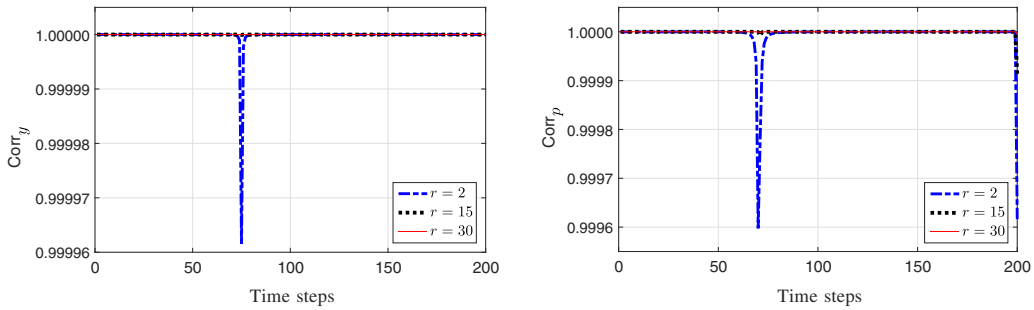
FIGURE 4 (Example 3 with $\kappa_1 = \kappa_2 = 10$) The first 40 points selected by DEIM for the nonlinear functions (left) $\mathcal{N}_1$ and (right) $\mathcal{N}_2$, with $m = 2577$ [Colour figure can be viewed at wileyonlinelibrary.com]

By applying Algorithms 3 with $m = 2577$ on this example with $\kappa_1 = \kappa_2 = 10$, we get $\ell = 5$ and $r_1 = r_2 = 15$. The obtained state and adjoint functions at time levels $t = 0.3$ second, $t = 0.6$ second, and $t = 1.0$ second are plotted in Figure 3. Moreover, the first 40 points selected by the DEIM for $\mathcal{N}_1$ and $\mathcal{N}_2$ are illustrated in Figure 4.

In this example, the exact solutions are unknown. So, for assessing the accuracy of POD/DEIM-IS algorithm, the RMSE and correlation coefficient between POD-IS solutions and POD/DEIM-IS solutions are reported in Figure 5. We see, by increasing the number of basis for approximation of the nonlinear functions $\mathcal{N}_1$ and $\mathcal{N}_2$, the RMSE is decreased. This point is also highlighted by the correlation coefficient.



(A) The RMSE of (left) state y and (right) adjoint p between POD-IS solution and POD/DEIM-IS solution by various values of $r_1 = r_2 = r$. (RMSE is plotted in logarithmic scale)



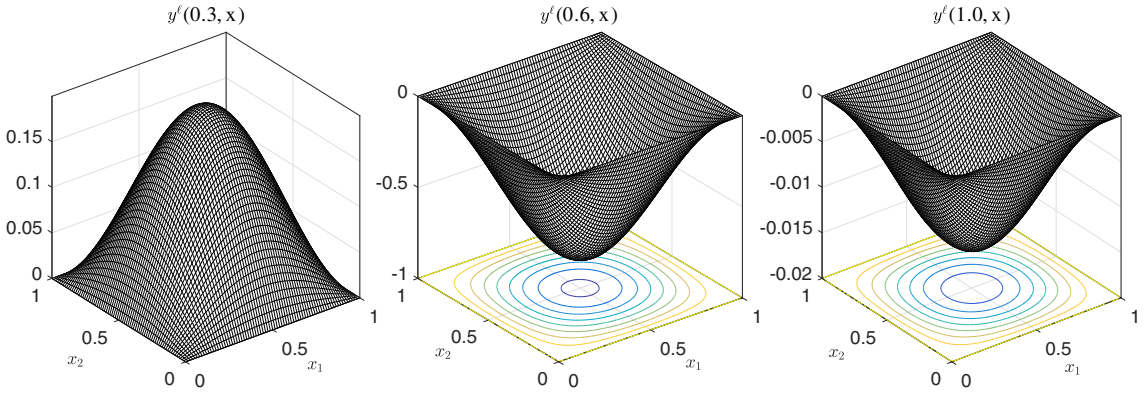
(B) Correlation of coefficients of (left) state y and (right) adjoint p between POD-IS solution and POD/DEIM-IS solution by various values of $r_1 = r_2 = r$.

FIGURE 5 (Example 3 with $\kappa_1 = \kappa_2 = 10$) RMSE and Corr between the POD-IS and POD-IS-DEIM solutions with $m = 2577$ [Colour figure can be viewed at wileyonlinelibrary.com]

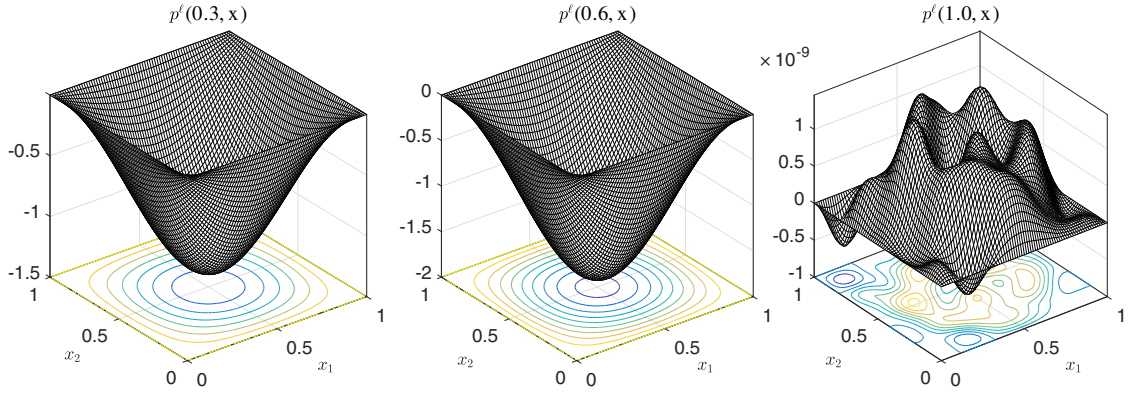
TABLE 3 (Example 3) Comparison between the obtained value of cost functional and CPU time, with POD-IS and POD/DEIM-IS methods, for various values of m and $\kappa_1 = \kappa_2 = \kappa$

Parameters of methods		POD-IS			POD-DEIM-IS		
		$\mathcal{J}(y,u)$	CPU time		$\mathcal{J}(y,u)$	CPU time	
κ	m		Setup time	POD-SP		Setup time	POD/DEIM-SP
10	177	0.615689	7.5	5.05	0.615693	7.6	4.3
10	665	0.601846	27.4	5.3	0.601852	27.6	3.5
10	2577	0.583256	77.5	7.4	0.583256	77.9	2.7
10	10145	0.581629	362.3	13.5	0.581630	363.5	2.63
10	40257	0.581222	1887	41.9	0.581222	1890	2.61
10^2	177	2.463484	7.5	6.7	2.463496	7.6	5.8
10^2	665	2.373453	27.4	7.3	2.373487	27.6	5.3
10^2	2577	2.350955	77.5	8.5	2.350957	77.9	3.7
10^2	10145	2.345315	362.3	19.3	2.345315	363.5	3.2
10^2	40257	2.343904	1887	61.2	2.343904	1890	3.2
10^3	177	3.995542	7.5	7	3.995553	7.6	6.1
10^3	665	3.863917	27.4	8	3.863959	27.6	5.8
10^3	2577	3.832058	77.5	11.7	3.832059	77.9	5.2
10^3	10145	3.824125	362.3	23.5	3.824128	363.5	3.8
10^3	40257	3.822150	1887	70.3	3.822151	1890	3.8
10^4	177	4.664583	7.5	9.3	4.664720	7.6	8.5
10^4	665	4.449323	27.4	10	4.449358	27.6	8
10^4	2577	4.408258	77.5	16.4	4.408278	77.9	7.4
10^4	10145	4.398990	362.3	38.6	4.398991	363.5	6.3
10^4	40257	4.396715	1887	116	4.396715	1890	5.7

In Table 3, for various values of m, κ_1, and κ_2, the obtained values of the cost functional and CPU times, with POD-IS and POD/DEIM-IS methods, are summarized for comparison. In this table, " Setup Time" for POD-IS (POD-DEIM-IS) refers to computation time for steps 1 to 7 of Algorithm 2 (steps 1 to 14 of Algorithm 3.)



(a) The POD-DEIM-IS solution of state variable at time instances $0.3s$, $0.6s$ and $1s$.



(b) The POD-DEIM-IS solution of adjoint variable at time instances $0.3s$, $0.6s$ and $1s$.

FIGURE 6 (Example 4 with $\kappa_1 = \kappa_2 = 10$) The POD/DEIM-IS solution with $m = 2577$ [Colour figure can be viewed at wileyonlinelibrary.com]

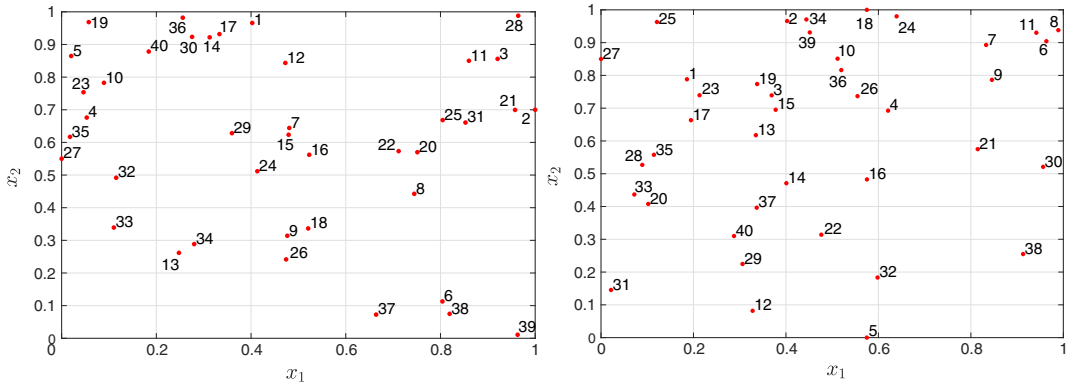


FIGURE 7 (Example 4 with $\kappa_1 = \kappa_2 = 10$) The first 40 points selected by DEIM for the nonlinear functions (left) $\mathcal{N}_1$ and (right) $\mathcal{N}_2$, with $m = 2577$ [Colour figure can be viewed at wileyonlinelibrary.com]

According to Table 3, we see that the obtained cost values by POD-IS and POD/DEIM-IS methods are very close to each other; moreover, in the two algorithms, the setup steps take almost the same CPU time. But the CPU time for POD/DEIM-IS algorithm is smaller than POD-SP algorithm and for larger m this difference is more significant.

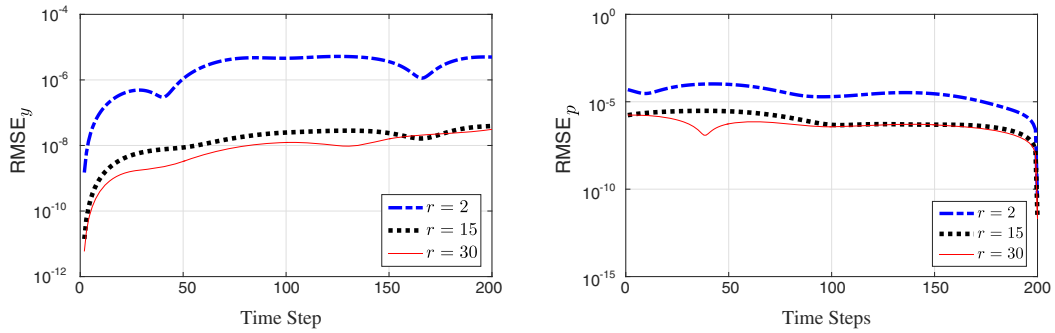
On the other hand, it can be observed that by increasing m, the CPU time of POD-SP is increased. This is because that the computational complexity of POD-SP is dependent to m. In contrast, by increasing m, the computational time of POD/DEIM-IS algorithm not only does not increase but actually decreases. The reason lies in the fact that, the computational complexity of POD/DEIM-IS algorithm is independent of m. Moreover, as m increases, we can obtain more suitable snapshots and thus more efficient POD and POD/DEIM bases are provided. Accordingly, the solution of POD/DEIM-IS can be obtained in less computational time.

5.4 | Example 4

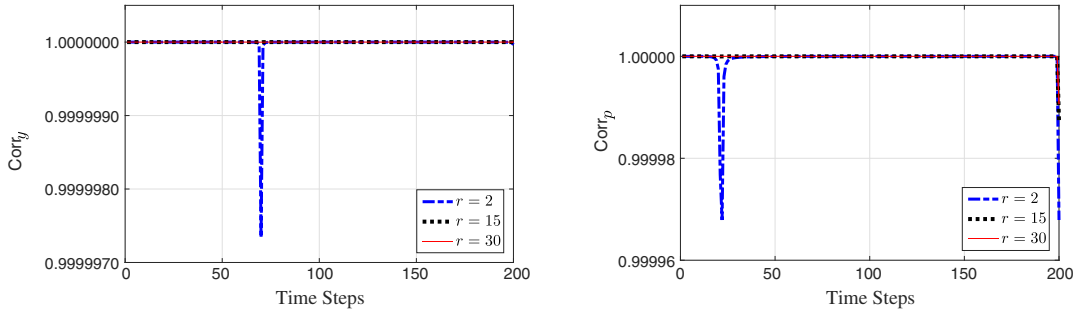
In the problem (1), let

$$\Omega = (0,1)^2,\ T = 1,\quad \mathcal{N}_1(y) = \sin(y),\quad g(x_1,x_2) = \sin(\pi x_1)\sin(\pi x_2),\quad h = 0,$$
$$z(t,x_1,x_2) = \cos(t)\sin(\pi x_1)\sin(\pi x_2),\quad w(x_1,x_2) = \cos(1)\sin(\pi x_1)\sin(\pi x_2).$$

We select $\kappa_1 = \kappa_2 = 10$ and apply Algorithm 3 with the input parameters $m = 2577$, $n = 200$, $\boldsymbol{\eta}^s = \boldsymbol{\gamma}^s = \mathbf{0}$, and $\epsilon_1 = \epsilon_2 = 0.001$. The obtained state and adjoint functions at time levels $t = 0.3$ second, $t = 0.6$ second, and $t = 1.0$ second are plotted in Figure 6. Moreover, the first 40 points selected by the DEIM for $\mathcal{N}_1$ and $\mathcal{N}_2$ are illustrated in Figure 7. In Figure 8, the RMSE and Corr between POD-IS and POD/DEIM-IS solutions are given. Figure 8 shows that the accuracy of DEIM is improved by increasing the number of basis for approximation of the nonlinear functions $\mathcal{N}_1$ and $\mathcal{N}_2$. Table 4 shows the results for various values of m, κ_1, and κ_2. The speedup when using DEIM approximation is significant.



(A) The RMSE of (left) state y and (right) adjoint p between POD-IS solution and POD/DEIM-IS solution by various values of $r_1 = r_2 = r$. (RMSE is plotted in logarithmic scale)



(B) Correlation of coefficients of (left) state y and (right) adjoint p between POD-IS solution and POD/DEIM-IS solution by various values of $r_1 = r_2 = r$.

FIGURE 8 (Example 4 with $\kappa_1 = \kappa_2 = 10$) RMSE and Corr between the POD-IS and POD-IS-DEIM solutions with $m = 2577$ [Colour figure can be viewed at wileyonlinelibrary.com]

TABLE 4 (Example 4) Comparison between the obtained value of cost functional and CPU time, with POD-IS and POD/DEIM-IS methods, for various values of m and $\kappa_1 = \kappa_2 = \kappa$

Parameters of methods		POD-IS			POD-DEIM-IS		
		$\mathcal{J}(y,u)$	CPU time		$\mathcal{J}(y,u)$	CPU time	
κ	m		Setup time	POD-SP		Setup time	POD/DEIM-SP
10	177	2.116025	8.2	4.4	2.116033	8.3	4.2
10	665	2.113724	23.2	4.6	2.113729	23.4	3.23
10	2577	2.113353	79.9	5.3	2.113355	80.1	2.35
10	10145	2.111856	362.7	7.9	2.111856	364	1.89
10	40257	2.083122	1833	21.2	2.083122	1836	1.88
10^2	177	12.167927	8.2	6.48	12.167935	8.3	6.33
10^2	665	12.166889	23.2	7.21	12.166896	23.4	4.25
10^2	2577	12.164032	79.9	9.46	12.164034	80.1	3.3
10^2	10145	12.163958	362.7	17.3	12.163959	364	2.99
10^2	40257	12.163841	1833	35.6	12.163841	1836	2.92
10^3	177	27.037811	8.2	8.08	27.037820	8.3	7.9
10^3	665	26.755052	23.2	8.17	26.755059	23.4	6.7
10^3	2577	26.683689	79.9	9.23	26.683694	80.1	5.7
10^3	10145	26.665662	362.7	25.2	26.665664	364	5.12
10^3	40257	26.665578	1833	52.9	26.665579	1836	4.68
10^4	177	32.560329	8.2	8.89	32.560335	8.3	8.84
10^4	665	32.069399	23.2	9.48	32.069412	23.4	7.66
10^4	2577	31.955331	79.9	12.92	31.955339	80.1	7.48
10^4	10145	31.954886	362.7	36.2	31.954887	364	6.32
10^4	40257	31.954873	1833	83.9	31.954873	1836	6.15

6 | CONCLUSION

In this paper, we used the POD technique in the indirect shooting method for solving the optimal control of wave equation. We showed that the POD technique greatly improves the ability of the indirect shooting method to obtain solutions quickly, without significant decrease in solution accuracy. Moreover, in the presence of the nonlinear term in the wave equation, to further speed up the solution time, a DEIM strategy is used for reducing the order of nonlinear calculations. We find that DEIM shows its impact on the problems, which require fine mesh discretization for obtaining reasonable accuracy. As a result, the reduced-order methods cause that the indirect shooting method becomes more applicable and attractive for the considered optimal control problems. Of course, investigating the sensitivity of the presented algorithms to the initial conditions and other input parameters would be desirable. This work is currently in progress. Another direction for further research would be to extend the presented method to the optimal control problems governed by PDE and inequality constraints.

ACKNOWLEDGEMENTS

The authors are very grateful to two referees for carefully reading this paper and for their comments and suggestions that have improved the paper.

ORCID

Mostafa Shamsi http://orcid.org/0000-0002-3806-9316
Ionel Michael Navon http://orcid.org/0000-0001-7830-7094

REFERENCES

1. Yang SD. Shooting methods for numerical solutions of exact controllability problems constrained by linear and semilinear 2-D wave equations. *Int J Numer Anal Model*. 2007;4(3-4):625-647.
2. Kroner A. Numerical methods for control of second order hyperbolic equations. *Ph.D. Thesis*: Fakultat fur Mathematik, Technische Universitat Munchen; 2011.
3. Li B, Liu J, Xiao M. A fast and stable preconditioned iterative method for optimal control problem of wave equations. *SIAM J Sci Comput*. 2015;37(6):A2508-A2534.

4. Clason C, Kaltenbacher B, Veljovic S. Boundary optimal control of the Westervelt and the Kuznetsov equations. *J Math Anal Appl*. 2009;356(2):738-751.
5. Banks HT, Keeling SL, Silcox RJ. Optimal control techniques for active noise suppression. In: Proceedings of the IEEE Conference on Decision and Control, Austin, Texas, USA, 1988:2006-2011.
6. Nestler P. Optimales design einer zylinderschale—eine problemstellung der optimalen steuerung in der linearen elastizitatstheorie. *Ph.D. Thesis*: Institut fur Angewandte Mathematik, Universitat Greifswald; 2010.
7. Lions JL. Exact controllability, stabilization and perturbations for distributed systems. *SIAM Rev*. 1988;30(1):1-68.
8. Zuazua E. Propagation, observation, and control of waves approximated by finite difference methods. *SIAM Rev*. 2005;47(2):197-243.
9. Ervedoza S, Zuazua E. *Numerical Approximation of Exact Controls for Waves*. New York: Springer; 2013.
10. Gerdts M, Greif G, Pesch HJ. Numerical optimal control of the wave equation: optimal boundary control of a string to rest in finite time. *Math Comput Simul*. 2008;79(4):1020-1032.
11. Gugat M. Penalty techniques for state constrained optimal control problems with the wave equation. *SIAM J Control Optim*. 2010;48(5):3026-3051.
12. Mordukhovich BS, Raymond JP. Neumann boundary control of hyperbolic equations with pointwise state constraints. *SIAM J Control Optim*. 2005;43(4):1354-1372.
13. Mordukhovich BS, Raymond JP. Optimal boundary control of hyperbolic equations with pointwise state constraints. *Nonlinear Anal Theory Methods Appl*. 2005;63(5-7):823-830.
14. Mordukhovich BS, Raymond JP. Dirichlet boundary control of hyperbolic equations in the presence of state constraints. *Appl Math Optim*. 2004;49(2):145-157.
15. Lagnese JE, Leugering G. Time-domain decomposition of optimal control problems for the wave equation. *Syst Control Lett*. 2003;48(3-4):229-242.
16. Luo XB, Chen YP, Huang YQ. A priori error estimates of finite volume element method for hyperbolic optimal control problems. *Sci China Math*. 2013;56(5):901-914.
17. Kröner A. Adaptive finite element methods for optimal control of second order hyperbolic equations. *Comput Methods Appl Math*. 2011;11(2):214-240.
18. Kunisch K, Reiterer SH. A Gautschi time-stepping approach to optimal control of the wave equation. *Appl Numer Math*. 2015;90:55-76.
19. Kroner A, Kunisch K, Vexler B. Semismooth Newton methods for optimal control of the wave equation with control constraints. *SIAM J Control Optim*. 2011;49(2):830-858.
20. Gugat M, Keimer A, Leugering G. Optimal distributed control of the wave equation subject to state constraints. *ZAMM Zeitschrift fur Angewandte Mathematik und Mechanik*. 2009;89(6):420-444.
21. Hesse HK. Multiple shooting and mesh adaptation for PDE constrained optimization problems. *Ph.D. Thesis*, Department of Applied Mathematics, University of Heidelberg; 2008.
22. Betts JT. Survey of numerical methods for trajectory optimization. *J Guidance, Control Dyn*. 1998;21(2):193-207.
23. Kunisch K, Volkwein S. Galerkin proper orthogonal decomposition methods for parabolic problems. *Numer Math*. 2001;90(1):117-148.
24. Volkwein S. Proper orthogonal decomposition: Theory and reduced-order modeling. Lecture Notes, University of Konstanz, Department of Mathematics and Statistics; 2013.
25. Benner P, Gugercin S, Willcox K. A survey of projection-based model reduction methods for parametric dynamical systems. *SIAM Rev*. 2015;57(4):483-531.
26. Quarteroni A, Manzoni A, Negri F. *Reduced Basis Methods for Partial Differential Equations: An Introduction*. UNITEXT, Milan: Springer International Publishing; 2015.
27. Ştefănescu R, Navon IM. POD/DEIM nonlinear model order reduction of an ADI implicit shallow water equations model. *J Comput Phys*. 2013;237:95-114.
28. Xiao D, Fang F, Buchan AG, Pain CC, Navon IM, Muggeridge A. Non-intrusive reduced order modelling of the Navier-Stokes equations. *Comput Methods Appl Mech Eng*. 2015;293:522-541.
29. Xiao D, Fang F, Pain C, Hu G. Non-intrusive reduced-order modelling of the Navier-Stokes equations based on RBF interpolation. *Int J Numer Methods Fluids*. 2015;79(11):580-595.
30. Wang Y, Navon IM, Wang X, Cheng Y. 2D Burgers equation with large Reynolds number using POD/DEIM and calibration. *Int J Numer Methods Fluids*. 2016;82(12):909-931.
31. Xiao D, Fang F, Pain C, Navon IM. A parameterized non-intrusive reduced order model and error analysis for general time-dependent nonlinear partial differential equations and its applications. *Comput Methods Appl Mech Eng*. 2017;317:868-889.
32. Wang Z, Xiao D, Fang F, Govindan R, Pain CC, Guo Y. Model identification of reduced order fluid dynamics systems using deep learning. *Int J Numer Methods Fluids*. 2017. https://doi.org/10.1002/fld.4416
33. Gubisch M, Volkwein S. Proper orthogonal decomposition for linear-quadratic optimal control. Technical Report, University of Konstanz; 2013. http://nbn-resolving.de/urn:nbn:de:bsz:352-250378
34. Troltzsch F, Volkwein S. POD a-posteriori error estimates for linear-quadratic optimal control problems. *Comput Optim Appl*. 2009;44(1):83-115.
35. Kammann E, Troltzsch F, Volkwein S. A posteriori error estimation for semilinear parabolic optimal control problems with application to model reduction by POD. *Math Model Numer Anal*. 2013;47(2):555-581.
36. Sachs EW, Schu M. A priori error estimates for reduced order models in finance. *ESAIM: Math Model Numer Anal*. 2013;47(2):449-469.

37. Chaturantabut S. Temporal localized nonlinear model reduction with a priori error estimate. *Appl Numer Math*. 2017;119:225-238.
38. Lucia DJ, King PI, Beran PS. Reduced order modeling of a two-dimensional flow with moving shocks. *Comput Fluids*. 2003;32(7):917-938.
39. Ştefănescu R, Sandu A, Navon IM. POD/DEIM reduced-order strategies for efficient four dimensional variational data assimilation. *J Comput Phys*. 2015;295:569-595.
40. Bistrian D, Susan-Resiga R. Weighted proper orthogonal decomposition of the swirling flow exiting the hydraulic turbine runner. *Appl Math Model*. 2016;40(5-6):4057-4078.
41. Ştefănescu R, Sandu A. Efficient approximation of sparse Jacobians for time-implicit reduced order models. *Int J Numer Methods Fluids*. 2017;83(2):175-204.
42. Bistrian D, Navon IM. The method of dynamic mode decomposition in shallow water and a swirling flow problem. *Int J Numer Methods Fluids*. 2017;83(1):73-89.
43. Lin Z, Xiao D, Fang F, Pain C, Navon IM. Non-intrusive reduced order modelling with least squares fitting on a sparse grid. *Int J Numer Methods Fluids*. 2017;83(3):291-306.
44. Attia A, Ştefănescu R, Sandu A. The reduced-order hybrid Monte Carlo sampling smoother. *Int J Numer Methods Fluids*. 2017;83(1):28-51.
45. Chaturantabut S, Sorensen DC. Nonlinear model reduction via discrete empirical interpolation. *SIAM J Sci Comput*. 2010;32(5):2737-2764.
46. Barrault M, Maday Y, Nguyen NC, Patera AT. An 'empirical interpolation' method: application to efficient reduced-basis discretization of partial differential equations. *Comptes Rendus Mathematique*. 2004;339(9):667-672.
47. Lions JL. *Optimal Control of Systems Governed by Partial Differential Equations*. Berlin: Springer-Verlag; 1971.
48. Lions JL. *Control of Distributed Singular Systems*: Paris: Gauthier-Villars; 1985.
49. Heinkenschloss M. A time-domain decomposition iterative method for the solution of distributed linear quadratic optimal control problems. *J Comput Appl Math*. 2005;173(1):169-198.
50. Studinger A, Volkwein S. Numerical analysis of pod a-posteriori error estimation for optimal control. In: Bredies K, Clason C, Kunisch K, Winckel Gv, eds. *Control and Optimization with PDE Constraints*. Basel: Springer; 2013:137-158.
51. Xiao D, Fang F, Buchan AG, et al. Non-linear model reduction for the Navier-Stokes equations using residual DEIM method. *J Comput Phys*. 2014;263:1-18.
52. Du J, Fang F, Pain CC, Navon IM, Zhu J, Ham DA. POD reduced-order unstructured mesh modeling applied to 2D and 3D fluid flow. *Comput Math Appl*. 2013;65(3):362-379.
53. Wang Z. Nonlinear model reduction based on the finite element method with interpolated coefficients: semilinear parabolic equations. *Numer Methods Partial Differ Equ*. 2015;31(6):1713-1741.
54. Christie I, Griffiths DF, Mitchell AR, Sanz-serna JM. Product approximation for non-linear problems in the finite element method. *IMA J Numer Anal*. 1981;1(3):253-266.
55. Fletcher CAJ. The group finite element formulation. *Comput Methods Appl Mech Eng*. 1983;37(2):225-244.
56. Chaturantabut S, Sorensen DC. Application of POD and DEIM on dimension reduction of non-linear miscible viscous fingering in porous media. *Math Comput Model Dyn Syst*. 2011;17(4):337-353.
57. Lass O, Volkwein S. POD Galerkin schemes for nonlinear elliptic-parabolic systems. *SIAM J Sci Comput*. 2013;35(3):A1271-A1298.
58. Langemyr L. *Partial Differential Equation Toolbox User's Guide: For use with MATLAB*. Math Works; 1996. http://www.mathworks.com/help/pdf_doc/pde/pde.pdf
59. Mehrpouya MA, Shamsi M, Razzaghi M. A combined adaptive control parametrization and homotopy continuation technique for the numerical solution of bang-bang optimal control problems. *ANZIAM J*. 2014;56(1):48-65.
60. Mehrpouya MA. An efficient pseudospectral method for numerical solution of nonlinear singular initial and boundary value problems arising in astrophysics. *Math Methods Appl Sci*. 2016;39(12):3204-3214.
61. Yang SD. Shooting methods for numerical solutions of control problems constrained by linear and nonlinear hyperbolic partial differential equations. *Ph.D. Thesis*: Department of Mathematics, Iowa State University; 2004.
62. Xiao D, Fang F, Buchan AG, Pain CC, Navon IM, Muggeridge A. Non-intrusive reduced order modelling of the Navier-Stokes equations. *Comput Methods Appl Mech Eng*. 2015;293:522-541.

How to cite this article: Sabeh Z, Shamsi M, Navon IM. An indirect shooting method based on the POD/DEIM technique for distributed optimal control of the wave equation. *Int J Numer Meth Fluids*. 2017;1–21. https://doi.org/10.1002/fld.4460