Model Order Reduction for High Dimensional Linear Systems based on Rank-1 Incremental Proper Orthogonal Decomposition

Chao Xu and Eugenio Schuster

***Abstract*— This work considers a modified incremental proper orthogonal decomposition (iPOD) method and applications to model order reduction (MOR) of linear evolutionary distributed parameter systems. A recursive matrix transformation approach is proposed to obtain reduced order models from high-dimensional systems with less computational cost than the non-recursive case. A detailed analysis of the computational complexity is carried out to compare different numerical procedures. Simulation results based on the heat conduction process in a two dimensional container validate the effectiveness of the proposed method.**

I. Introduction

The discretization of partial differential equations (PDEs) using various numerical methods usually results in large sets of ordinary differential equations (ODEs) or ordinary difference equations (OdEs). However, these high-dimensional dynamical models derived from the spatial, and eventually temporal, discretization procedure are often inappropriate for control synthesis because they are computationally costly in solving controller-synthesis-related equations (e.g., Riccati equation, Lyapunov equation, etc.). One may think to tackle the high-dimensionality challenge by using coarse meshes for the domain discretization. However, lower order models obtained by using coarse meshes may not be accurate enough to capture the essential dynamics. Therefore, model order reduction (MOR) using observational/simulation data becomes an alternative procedure to provide reduced-order models for controller synthesis. This is a typical data-driven model-reduction approach using both physical structures and observational data.

There are several MOR approaches, including the balanced truncation method, the Krylow subspace method and the proper orthogonal decomposition (POD) method. The balanced truncation is an important tool originated in the control systems community [1]. When this method is used to derive reduced-order models for high dimensional systems, Lyapunov equations with the same order of the original systems must be solved numerically. This is a computational challenge and it finally led to the birth of the balanced POD method which combines both the POD and the balanced truncation method but without solving high-dimensional Lyapunov equations [2], [3]. The Krylov subspace method has been used widely in fast simulations of large scale linear systems (e.g., discretized PDEs, large scale integrated circuits (IC), etc.). For a detailed introduction of both the balanced truncation method and the Krylov subspace method for model order reduction, the reader is referred to introductory books (e.g., [4], [5]).

The POD method is also known as the principal component analysis (PCA) or the Karhunen-Loeve decomposition. The original concept of the POD method goes back to Pearson's work published in 1833 [6]. The POD method has been widely used in understanding complex behaviors produced by many high dimensional dynamical systems, such as fluid flows [7], heat flow [8], microelectromechanical systems (MEMS) [9], aero-elasticity [10], etc. However, most work concerning POD focuses on what is called the *batch* POD method, where modes are extracted from given historic observations without having the capability to incorporate new observations when they become available. A new method, called *incremental* POD (iPOD) (e.g., [11]), has been proposed to enable dimension increase, and mode deformations if necessary, when new observations are available. The incremental POD method has been applied successfully to pattern recognition (e.g., [12]) and visual tracking (e.g., [13]).

The main contribution of this paper is the introduction of a modified incremental POD method for model order reduction of high-dimensional dynamical systems. Without using the mean value of the observations in this work, the proposed incremental POD problem has a simpler form than that in [11] and it can be extended to multi-snapshot-based incremental POD schemes (i.e., rank q ($q > 1$) incremental POD). To match the incremental POD method, an incremental Galerkin projection scheme is developed. This scheme computes system projection matrices recursively based on the existing matrices obtained in the previous mode extraction step. A detailed analysis of the computational complexity is carried out to illustrate the computational efficiency of the incremental POD-based MOR approach.

The paper is organized as follows. In Section II, the mathematical derivations of the POD method and the snapshots method are summarized. In Section III, the formulation and numerical schemes of the incremental POD method are discussed. In Section IV, an incremental Galerkin projection method for model order reduction is proposed. In Section V, the computational complexity of the incremental POD method is studied. In Section VI, numerical simulations that validate the effectiveness of the proposed method are provided. The paper is closed by stating conclusions and future research topics in Section VII.

This work was supported by the Fundamental Research Funds for the Central Universities (1A5000-172210101) and the National Science Foundation CAREER award program (ECCS-0645086).

C. Xu (cxu@csc.zju.edu.cn) is with the Department of Control Science and Engineering, Zhejiang University, 38 Zheda Road, Hangzhou, 310027, P. R. China. E. Schuster is with the Department of Mechanical Engineering and Mechanics, Lehigh University, 19 Memorial Drive West, Bethlehem, PA 18015, USA.

II. Proper Orthogonal Decomposition (POD)

We have N training samples (snapshots) $\mathbf{x}_i \in \mathbb{R}^n$ $(i = 1, 2, \ldots, N)$ and we use $\mathscr{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_N) \in \mathbb{R}^{n\times N}$ to denote the observations (snapshot matrix). Usually, n (the number of spatial discretization nodes) is much larger than N (the number of samples at different time instants). We introduce the covariance matrix of $\mathscr{X}$,

$$\mathscr{C} = \frac{1}{N}\sum_{i=1}^{N} \mathbf{x}_i \mathbf{x}_i^T. \tag{1}$$

The POD problem can be stated as the search of a sequence of orthogonal basis functions $\{\phi_k\}_{k=1}^{p}$ $(p \le N,\, \phi_i^T\phi_j = \delta_{ij})$ to represent each snapshot in the observation set $\mathscr{X}$, i.e.,

$$\mathbf{x}_i \approx \widetilde{\mathbf{x}}_i = \sum_{k=1}^{p} X_{ik}\phi_k, \quad \forall \mathbf{x}_i \in \mathscr{X},\, i = 1, 2, \ldots, N, \tag{2}$$

where the coefficient X_{ik} can be determined by $X_{ik} = \phi_k^T \mathbf{x}_i$. Then, the basis functions can be determined by the following approximation-error minimization problem:

$$\begin{aligned} &\min_{\phi_1,\ldots,\phi_p} \frac{1}{N}\sum_{i=1}^{N}\left\|\mathbf{x}_i - \sum_{k=1}^{p} X_{ik}\phi_k\right\|_2^2 \\ &\text{subject to: } \phi_k^T\phi_l = \delta_{kl} = \begin{cases} 1, & \text{if } k = l, \\ 0, & \text{otherwise,} \end{cases} \end{aligned} \tag{3}$$

where the constraint implies the orthonormality property of the basis vectors.

Lemma 1 ([14], [15]): Given the observation data set $\mathscr{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_N) \in \mathbb{R}^{n\times N}$ with rank $d \le \min\{n, N\}$, then the optimal solution to the minimization problem (3) is given by the first p $(p \le n)$ eigenvectors of the following eigenvalue decomposition problem

$$\frac{1}{N}\mathscr{X}\mathscr{X}^T\phi_l = \lambda_l\phi_l, \quad l = 1, \ldots, n. \tag{4}$$

The approximation error can be bounded by

$$\varepsilon^2(p) = \min_{\phi_1,\ldots,\phi_p} \frac{1}{N}\sum_{i=1}^{N}\left\|\mathbf{x}_i - \sum_{k=1}^{p} X_{ik}\phi_k\right\|_2^2 \le \sum_{k=p+1}^{n}\lambda_k. \tag{5}$$

Remark 1 (Method of snapshots [16]): If $n \gg N$, then it is impractical to solve the eigenvalue decomposition problem (4). However, one can solve the following symmetric eigenvalue decomposition $\frac{1}{N}\mathscr{X}^T\mathscr{X}\psi_l = \lambda_l\psi_l$, $l = 1, \ldots, n$. The POD modes are given by $\phi_l = \frac{1}{\sqrt{\lambda_l}}\mathscr{X}\psi_l$, $l = 1, 2, \ldots, p$.

III. Rank-1 Incremental POD

Given a new observation $\mathbf{x}_\# \in \mathbb{R}^n$, we can use the eigenspace model $\Phi = (\phi_1, \ldots, \phi_p)$ to give an approximation $\mathbf{x}_\# \approx \widetilde{\mathbf{x}}_\# = \sum_{i=1}^{p} g_i\phi_i$, $g_i = \phi_i^T\mathbf{x}_\#$, where the approximation error is given by

$$\mathbf{h} = \mathbf{x}_\# - \widetilde{\mathbf{x}}_\# = \mathbf{x}_\# - \Phi\mathbf{g}, \quad \mathbf{g} = (g_1, \ldots, g_p)^T. \tag{6}$$

If $\|\mathbf{h}\|$ is large, then the observation $\mathbf{x}_\#$ is not well represented by eigenspace model $(\phi_1, \ldots, \phi_p)$. We need to include this new observation $\mathbf{x}_\#$ and update the eigenspace model.

Remark 2 (Shift window POD): The most straightforward way to achieve this is by adding the latest sample to the end of the snapshot ensemble and dropping the sample at the beginning to retain a fixed length window. Then, the POD problem is solved again for the new data ensemble in order to update both the eigenvalues and eigenvectors.

It is not computationally efficient to carry out repeatedly POD computations at each ensemble update. We present an incremental POD method in this section. We consider the eigenvalue decomposition of the updated observation set $\mathscr{X}' \leftarrow (\mathscr{X}, \mathbf{x}_\#) \in \mathbb{R}^{n\times(N+1)}$. First, the covariance matrix becomes

$$\mathscr{C}' = \frac{N}{N+1}\mathscr{C} + \frac{1}{N+1}\mathbf{x}_\#\mathbf{x}_\#^T. \tag{7}$$

Remark 3: Using the residues $\mathbf{x}_i - \bar{\mathbf{x}}'$ $(i = 1, 2, \ldots, N+1)$, where $\bar{\mathbf{x}}' \triangleq \frac{1}{N+1}(N\bar{\mathbf{x}} + \mathbf{x}_\#)$, to replace the snapshots in (7), we have

$$\begin{aligned} \widetilde{\mathscr{C}}' &= \frac{1}{N+1}\left(\sum_{i=1}^{N}(\mathbf{x}_i - \bar{\mathbf{x}}')(\mathbf{x}_i - \bar{\mathbf{x}}')^T \right. \\ &\quad \left. + (\mathbf{x}_\# - \bar{\mathbf{x}}')(\mathbf{x}_\# - \bar{\mathbf{x}}')^T\right) (\textit{noting } \mathbf{x}_{N+1} = \mathbf{x}_\#) \\ &= \frac{N}{N+1}\widetilde{\mathscr{C}} + \frac{N}{(N+1)^2}(\mathbf{x}_\# - \bar{\mathbf{x}})(\mathbf{x}_\# - \bar{\mathbf{x}})^T, \end{aligned} \tag{8}$$

where $\widetilde{\mathscr{C}} \triangleq \frac{1}{N}\sum_{i=1}^{N}(\mathbf{x}_i - \bar{\mathbf{x}})(\mathbf{x}_i - \bar{\mathbf{x}})^T$ and the cross terms involving $\sum_{i=1}^{N}(\mathbf{x}_i - \bar{\mathbf{x}})(\bar{\mathbf{x}} - \mathbf{x}_\#)^T$ and its transpose are zero. We note that the recursive form (8) has the same structure of (7) where the average value is not used in computing the covariance matrix.

Remark 4: Given a data sequence $\mathbf{x}_i$ $(i = 1, 2, \ldots, N)$ and the newly collected data sequence $\mathbf{x}_{\#,j}$ $(j = 1, 2, \ldots, \Delta N)$, where the integer $\Delta N > 1$, we derive the covariance matrix using the following definition

$$\begin{aligned} \widetilde{\mathscr{C}}' &\triangleq \frac{1}{N+\Delta N}\sum_{i=1}^{N+\Delta N}(\mathbf{x}_i - \bar{\mathbf{x}}')(\mathbf{x}_i - \bar{\mathbf{x}}')^T \\ &= \frac{1}{N+\Delta N}\sum_{i=1}^{N}(\mathbf{x}_i - \bar{\mathbf{x}}')(\mathbf{x}_i - \bar{\mathbf{x}}')^T \\ &\quad + \frac{1}{N+\Delta N}\sum_{j=1}^{\Delta N}(\mathbf{x}_{\#,j} - \bar{\mathbf{x}}')(\mathbf{x}_{\#,j} - \bar{\mathbf{x}}')^T. \end{aligned} \tag{9}$$

Noting that $\bar{\mathbf{x}}' = \frac{1}{N+\Delta N}\left(N\bar{\mathbf{x}} + \sum_{k=1}^{\Delta N}\mathbf{x}_{\#,k}\right)$, then we have

$$\begin{aligned} \widetilde{\mathscr{C}}' &= \frac{1}{N+\Delta N}\sum_{i=1}^{N}(\mathbf{x}_i - \bar{\mathbf{x}})(\mathbf{x}_i - \bar{\mathbf{x}})^T + \frac{N}{(N+\Delta N)^3} \\ &\quad \times \left(\Delta N\bar{\mathbf{x}} - \sum_{k=1}^{\Delta N}\mathbf{x}_{\#,j}\right)\left(\Delta N\bar{\mathbf{x}} - \sum_{k=1}^{\Delta N}\mathbf{x}_{\#,j}\right)^T \\ &\quad + \frac{1}{N+\Delta N}\sum_{j=1}^{\Delta N}\left(\mathbf{x}_{\#,j} - \frac{N\bar{\mathbf{x}}}{N+\Delta N} - \frac{\sum_{k=1}^{\Delta N}\mathbf{x}_{\#,k}}{N+\Delta N}\right) \\ &\quad \times \left(\mathbf{x}_{\#,j} - \frac{N\bar{\mathbf{x}}}{N+\Delta N} - \frac{\sum_{k=1}^{\Delta N}\mathbf{x}_{\#,k}}{N+\Delta N}\right)^T, \end{aligned} \tag{10}$$

where we have noted that the terms involving the factor $\sum_{i=1}^{N}(\mathbf{x}_i-\bar{\mathbf{x}})\left(\Delta N\bar{\mathbf{x}}-\sum_{k=1}^{\Delta N}\mathbf{x}_{\#,k}\right)^T$ and its transpose are zero. We note that the recursive form in (10) cannot be formulated as the one in (7). However, by using the definition (1) proposed in this paper, the recursive form corresponding to (7) for block update ($\Delta N>1$) becomes

$$\mathscr{C}'=\frac{N}{N+\Delta N}\mathscr{C}+\frac{1}{N+\Delta N}\sum_{j=1}^{\Delta N}\mathbf{x}_{\#,j}\mathbf{x}_{\#,j}^T. \tag{11}$$

It is much more neat than the recursive form in (10) and still retain the same structure of (7) if we introduce a new update black matrix $\mathfrak{X}_{\#}=(\mathbf{x}_{\#,1},\ldots,\mathbf{x}_{\#,\Delta N})^T$.

The normalized residue vector $\mathbf{h}$ is a candidate to expand the original eigenspace generated from the observation set $\mathscr{X}\in\mathbb{R}^{n\times N}$:

$$\hat{\mathbf{h}}=\begin{cases}\dfrac{\mathbf{h}}{\|\mathbf{h}\|_2}, & \text{if } \|\mathbf{h}\|_2>\eta,\\ 0, & \text{otherwise,}\end{cases} \tag{12}$$

where η is a small threshold value. A rotation matrix R is used to change the subspace $(\phi_1,\ldots,\phi_p,\hat{\mathbf{h}})$ into a solution of the eigenvalue decomposition problem of $\mathscr{C}'$ based on the new observation set $\mathscr{X}'$:

$$\mathscr{C}'\left[(\Phi,\hat{\mathbf{h}})R\right]=\left[(\Phi,\hat{\mathbf{h}})R\right]\Lambda'. \tag{13}$$

We note that this is an $n\times n$ eigenvalue decomposition problem and we can multiply $(\Phi,\hat{\mathbf{h}})^T$ from the left to obtain $(\Phi,\hat{\mathbf{h}})^T\mathscr{C}'(\Phi,\hat{\mathbf{h}})R=R\Lambda'$, which is another eigenvalue decomposition problem but it is $(p+1)$-dimensional. By noting the expression of $\mathscr{C}'$ in (7), we have

$$(\Phi,\hat{\mathbf{h}})^T\left[\frac{N}{N+1}\mathscr{C}+\frac{1}{N+1}\mathbf{x}_{\#}\mathbf{x}_{\#}^T\right](\Phi,\hat{\mathbf{h}})R=R\Lambda'. \tag{14}$$

We define

$$(\Phi,\hat{\mathbf{h}})^T\mathscr{C}(\Phi,\hat{\mathbf{h}})=\begin{pmatrix}\Phi^T\mathscr{C}\Phi & \Phi^T\mathscr{C}\hat{\mathbf{h}}\\ \hat{\mathbf{h}}^T\mathscr{C}\Phi & \hat{\mathbf{h}}^T\mathscr{C}\hat{\mathbf{h}}\end{pmatrix}\approx\begin{pmatrix}\Lambda & 0\\ 0 & 0\end{pmatrix}$$

$$(\Phi,\hat{\mathbf{h}})^T\mathbf{x}_{\#}\mathbf{x}_{\#}^T(\Phi,\hat{\mathbf{h}})=\begin{pmatrix}\Phi^T\mathbf{x}_{\#}\mathbf{x}_{\#}^T\Phi & \Phi\mathbf{x}_{\#}\mathbf{x}_{\#}^T\hat{\mathbf{h}}\\ \hat{\mathbf{h}}^T\mathbf{x}_{\#}\mathbf{x}_{\#}^T\Phi & \hat{\mathbf{h}}^T\mathbf{x}_{\#}\mathbf{x}_{\#}^T\hat{\mathbf{h}}\end{pmatrix}$$
$$=\begin{pmatrix}\mathbf{g}\mathbf{g}^T & \gamma\mathbf{g}\\ \gamma\mathbf{g}^T & \gamma^2\end{pmatrix},\quad \gamma\triangleq\hat{\mathbf{h}}^T\mathbf{x}_{\#}.$$

to rewrite (14) as the following eigenvalue problem:

$$\left[\frac{N}{N+1}\begin{pmatrix}\Lambda & 0\\ 0 & 0\end{pmatrix}+\frac{1}{N+1}\begin{pmatrix}\mathbf{g}\mathbf{g}^T & \gamma\mathbf{g}\\ \gamma\mathbf{g}^T & \gamma^2\end{pmatrix}\right]R=R\Lambda'. \tag{15}$$

Remark 5: When $N\to\infty$, the incremental eigenvalue decomposition problem (15) becomes convergent and the newly added part can be considered as a small perturbation. Instead of introducing the error vector $\hat{\mathbf{h}}$ (i.e., $\hat{\mathbf{h}}=0$) to form the new subspace $\left(\Phi,\hat{\mathbf{h}}\right)$, a nonsingular rotation operation of the POD modes Φ, i.e., $\Phi\tilde{R}$ can give an alternative solution. The rotation transformation matrix $\tilde{R}$ can be provided by the eigenvalue problem of the matrix $\frac{N}{N+1}\Lambda+\frac{1}{N+1}\mathbf{g}\mathbf{g}^T$, which is a degenerate case of (15). Then the matrix perturbation theory [17] can be used to provide a fast (approximate) update.

IV. Model Order Reduction

Let us assume that a high-dimensional linear time-invariant dynamical model

$$\frac{d\mathbf{X}(t)}{dt}=\mathbf{A}\mathbf{X}(t)+\mathbf{B}\mathbf{U}(t),\quad \mathbf{Y}(t)=\mathbf{C}\mathbf{X}(t), \tag{16}$$

is given, where $\mathbf{X}\in\mathbb{R}^n$, $\mathbf{U}\in\mathbb{R}^m$, $\mathbf{Y}\in\mathbb{R}^r$ and the initial conditions is stated as $\mathbf{X}(t_0)=\mathbf{X}_0$. $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{C}$ are system matrices obtained from a standard spatial discretization procedure and $\mathbf{X}\in\mathbb{R}^n$ is the state vector representing the values at the discrete nodes. Usually, the number of discrete nodes (state dimension) is much larger than the number of sensors, i.e., $n\gg r$.

We generalize the snapshot expansion (2) to a time continuous time case $\mathbf{x}(t)\approx\widetilde{\mathbf{x}}(t)=\sum_{k=1}^{p}X_k(t)\phi_p$. The reduced order model is

$$\frac{d\mathcal{X}(t)}{dt}=\mathcal{A}\mathcal{X}(t)+\mathcal{B}\mathcal{U}(t),\quad \mathcal{Y}(t)=\mathcal{C}\mathcal{X}(t), \tag{17}$$

where $\mathcal{X}=(X_1(t),\ldots,X_p(t))^T\in\mathbb{R}^p$, $\mathbf{U}=\mathcal{U}\in\mathbb{R}^m$, $\mathcal{Y}\in\mathbb{R}^r$ and $\mathcal{A}=\Phi^T\mathbf{A}\Phi$, $\mathcal{B}=\Phi^T\mathbf{B}$ and $\mathcal{C}=\mathbf{C}\Phi$. Now we consider the transformation with $\Phi'=\left[\Phi,\hat{\mathbf{h}}\right]R$, then

$$\begin{aligned}\mathcal{A}'&=R^T\begin{pmatrix}\Phi\\ \hat{\mathbf{h}}\end{pmatrix}\mathbf{A}\begin{pmatrix}\Phi & \hat{\mathbf{h}}\end{pmatrix}R\\ &=R^T\begin{pmatrix}\Phi^T\mathbf{A}\Phi & \Phi^T\mathbf{A}\hat{\mathbf{h}}\\ \hat{\mathbf{h}}^T\mathbf{A}\Phi & \hat{\mathbf{h}}^T\mathbf{A}\hat{\mathbf{h}}\end{pmatrix}R\\ &=R^T\begin{pmatrix}\mathcal{A} & \Phi^T\mathbf{A}\hat{\mathbf{h}}\\ \hat{\mathbf{h}}^T\mathbf{A}\Phi & \hat{\mathbf{h}}^T\mathbf{A}\hat{\mathbf{h}}\end{pmatrix}R\end{aligned} \tag{18}$$

$$\mathcal{B}'=R^T\begin{pmatrix}\Phi^T\\ \hat{\mathbf{h}}^T\end{pmatrix}\mathbf{B}=R^T\begin{pmatrix}\Phi^T\mathbf{B}\\ \hat{\mathbf{h}}^T\mathbf{B}\end{pmatrix}=R^T\begin{pmatrix}\mathcal{B}\\ \hat{\mathbf{h}}^T\mathbf{B}\end{pmatrix} \tag{19}$$

$$\mathcal{C}'=\mathbf{C}\begin{pmatrix}\Phi & \hat{\mathbf{h}}\end{pmatrix}R=\begin{pmatrix}\mathbf{C}\Phi & \mathbf{C}\hat{\mathbf{h}}\end{pmatrix}R=\begin{pmatrix}\mathcal{C} & \mathbf{C}\hat{\mathbf{h}}\end{pmatrix}R \tag{20}$$

We summarize different POD-MOR procedures as the following algorithms:

Algorithm 1: `Shift window POD-MOR`

1) `New observation` $\mathbf{x}_{\#}$ `is available and compute the approximation error` $\mathbf{h}$ `based on (6);`
2) `If` $\|\mathbf{h}\|>\eta$, `the snapshot matrix` $\mathscr{X}=(\mathbf{x}_1,\ldots,\mathbf{x}_N)$ `becomes` $\mathscr{X}'=(\mathbf{x}_2,\ldots,\mathbf{x}_N,\mathbf{x}_{\#})$ `with the new covariance matrix denoted by` $\mathscr{C}'=\mathbf{Var}\{\mathscr{X}'\}$;
3) `We solve the` N`th-order eigenvalue problem` $\mathscr{C}'Q=Q\Lambda$, `where` $Q=(\mathbf{q}_1,\ldots,\mathbf{q}_N)$ `and` $\Lambda=\mathrm{diag}\,(\lambda_1,\ldots,\lambda_N)$;
4) `Extract the first` q `columns` $\mathbf{q}_1,\ldots,\mathbf{q}_q$ `(according to` $\lambda_1\geq\ldots\geq\lambda_q$`) to form` $\Phi'=\left(\frac{1}{\sqrt{\lambda_1}}\mathscr{X}'\mathbf{q}_1,\ldots,\frac{1}{\sqrt{\lambda_q}}\mathscr{X}'\mathbf{q}_q\right)$ `and` $\Lambda'=\mathrm{diag}\,(\lambda_1,\ldots,\lambda_q)$. `The truncation order` q `is chosen as` $q=\min\left\{q'\left|\frac{\sum_{k=1}^{q'}\lambda_k}{\sum_{k=1}^{N}\lambda_k}\geq 1-\epsilon\right.\right\}$;
5) `Compute the matrix transformations`

$$\mathcal{A}'=(\Phi')^T\mathbf{A}\Phi',\quad \mathcal{B}'=(\Phi')^T\mathbf{B},\quad \mathcal{C}'=\mathbf{C}\Phi'. \tag{21}$$

Algorithm 2: `Incremental POD-MOR`

1) `New observation` $\mathbf{x}_{\#}$ `is available and compute the approximation error` $\mathbf{h}$ `based on (6);`
2) `If` $\|\mathbf{h}\| > \eta$, `the snapshot matrix` $\mathscr{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_N)$ `becomes` $\mathscr{X}' = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N, \mathbf{x}_{\#})$ `with the covariance matrix` $\mathscr{C}'$ `determined by (7);`
3) `We solve the` $(p+1)$`-order eigenvalue problem (15);`
4) `Compute the matrix transformations based on (18)-(20) or` $\mathcal{A}' = \Phi^T\mathbf{A}\Phi'$, $\mathcal{B}' = (\Phi')^T\mathbf{B}$ `and` $\mathcal{C}' = \mathbf{C}\Phi'$.

V. Arithmetic Complexity Analysis

Now we study the arithmetic complexity of both the shift-window POD-MOR and the incremental POD-MOR approaches. We have to compare the arithmetic flops from two aspects: i- the flops in computing the transformed matrices; ii- the flops in computing the POD-modes.

Lemma 2 (Matrix multiplication flops): Given two matrices $\mathcal{M}_1 \in \mathbb{R}^{d_1 \times d_2}$ and $\mathcal{M}_2 \in \mathbb{R}^{d_2 \times d_3}$, then the complexity of the matrix multiplication $\mathcal{M}_1\mathcal{M}_2$ is $2d_1d_2d_3$ flops (a flop is a floating point operation [18]).

Proof: We first give the algorithm for the matrix multiplication $\mathcal{M}_1\mathcal{M}_2$:

(Matrix Multiplication)

(The procedure to compute $\mathcal{M} = \mathcal{M}_1\mathcal{M}_2$)

```
M = 0
for i = 1 : d1
    for j = 1 : d2
        for k = 1 : d3
            M(i,j) = M1(i,k)M2(k,j) + M(i,j)
        end
    end
end
```

For the most deeply nested statement of this algorithm, $\mathcal{M}(i,j) = \mathcal{M}_1(i,k)\mathcal{M}_2(k,j) + \mathcal{M}(i,j)$, there are two flops involved in the operation. In addition, the statement is executed $d_1d_2d_3$ times. Therefore, the matrix multiplication $\mathcal{M}_1\mathcal{M}_2$ requires $2d_1d_2d_3$ flops, which is denoted by $\mathscr{F}(\mathcal{M}) = \mathscr{F}(\mathcal{M}_1 * \mathcal{M}_2) = 2d_1d_2d_3$, where '$*$' is used to denote the operation whose computational-complexity flops are to be counted. ∎

We multiply matrices from left to right. For example, when we compute $\hat{\mathbf{h}}^T\mathbf{A}\Phi$, we first calculate $\mathbf{A}\Phi$ and later $\hat{\mathbf{h}}^T(\mathbf{A}\Phi)$. Now we estimate the computational complexity in calculating (18)-(20). Based on Lemma 2, we know that the computational-complexity flops of $\mathbf{A}\Phi$ are $2n \times n \times p$, i.e., $\mathscr{F}(\mathbf{A} * \Phi) = 2pn^2$, where $\mathbf{A} \in \mathbb{R}^{n\times}$ and $\Phi \in \mathbb{R}^{n\times p}$. Similarly, the computational-complexity flops of $\mathbf{A}\hat{\mathbf{h}}$ are $\mathscr{F}\left(\mathbf{A} * \hat{\mathbf{h}}\right) = 2n^2$. Thus, we can obtain the complexity of computing (18) as

$$\begin{aligned}
&\mathscr{F}\left(\mathcal{A}'|_{(18)}\right)\\
=&\mathscr{F}(\mathbf{A}*\Phi) + \mathscr{F}(\mathbf{A}*\hat{\mathbf{h}})\\
&+\mathscr{F}\left[\hat{\mathbf{h}}^T*(\mathbf{A}\Phi)\right] + \mathscr{F}\left[\hat{\mathbf{h}}^T*(\mathbf{A}\hat{\mathbf{h}})\right] + \mathscr{F}\left[\Phi^T*(\mathbf{A}\hat{\mathbf{h}})\right]\\
&+\mathscr{F}\left[\begin{pmatrix} \mathcal{A} & \Phi^T\mathbf{A}\hat{\mathbf{h}} \\ \hat{\mathbf{h}}^T\mathbf{A}\Phi & \hat{\mathbf{h}}^T\mathbf{A}\hat{\mathbf{h}} \end{pmatrix} * R\right]\\
&+\mathscr{F}\left\{R^T * \left[\begin{pmatrix} \mathcal{A} & \Phi^T\mathbf{A}\hat{\mathbf{h}} \\ \hat{\mathbf{h}}^T\mathbf{A}\Phi & \hat{h}^T\mathbf{A}\hat{\mathbf{h}} \end{pmatrix}\right]\right\}\\
=&2n^2p + 2n^2 + 2np + 2n + 2pn\\
&+2(p+1)^3 + 2(p+1)^3\\
=&2(p+1)n^2 + 2(2p+1)n + 4(p+1)^3.
\end{aligned} \tag{22}$$

Similarly, we have

$$\begin{aligned}
\mathscr{F}\left(\mathcal{B}'|_{(19)}\right) &= \mathscr{F}\left(\hat{\mathbf{h}}^T * \mathbf{B}\right) + \mathscr{F}\left[R^T * \begin{pmatrix} \mathcal{B} \\ \hat{h}^T\mathbf{B} \end{pmatrix}\right]\\
&= 2nm + 2(p+1)^2m,
\end{aligned} \tag{23}$$

and

$$\begin{aligned}
\mathscr{F}\left(\mathcal{C}'|_{(20)}\right) &= \mathscr{F}\left(\mathbf{C}*\hat{\mathbf{h}}\right) + \mathscr{F}\left[(\mathcal{C} \quad \mathbf{C}\hat{\mathbf{h}}) * R\right]\\
&= 2rn + 2r(p+1)^2.
\end{aligned} \tag{24}$$

Therefore, we have the order of the complexity

$$\mathscr{F}\left(\mathcal{A}'|_{(18)}\right) + \mathscr{F}\left(\mathcal{B}'|_{(19)}\right) + \mathscr{F}\left(\mathcal{C}'|_{(20)}\right) \sim O(n^2). \tag{25}$$

We can compare the complexity of (18)-(20) with that of (21):

$$\begin{aligned}
\mathscr{F}\left(\mathcal{A}'|_{(21)}\right) &= \mathscr{F}(\mathbf{A}*\Phi') + \mathscr{F}\left[(\Phi')^T * (\mathbf{A}\Phi')\right]\\
&= 2(p+1)n^2 + 2(p+1)^2n,
\end{aligned} \tag{26}$$

$$\mathscr{F}\left(\mathcal{B}'|_{(21)}\right) = \mathscr{F}\left((\Phi')^T * \mathbf{B}\right) = 2(p+1)nr, \tag{27}$$

$$\mathscr{F}\left(\mathcal{C}'|_{(21)}\right) = \mathscr{F}(\mathbf{C}*\Phi') = 2rn(p+1), \tag{28}$$

and

$$\mathscr{F}\left(\mathcal{A}'|_{(21)}\right) + \mathscr{F}\left(\mathcal{B}'|_{(21)}\right) + \mathscr{F}\left(\mathcal{A}'|_{(21)}\right) \sim O(n^2). \tag{29}$$

Therefore, we can see that the recursive transformations (18)-(20) can reduce computational complexity to some extent but there is no significant improvement over (21) in calculating the new system matrices.

Let us now focus on the computation of the POD-modes. Given a matrix $\mathcal{M} \in \mathbb{R}^{n\times n}$, the eigenvalue problem is $\mathcal{M}\mathbf{v} = \lambda\mathbf{v}$. There are various methods (e.g., Chapters 7 and 8 in [18]) that numerically provide the eigenvalue pair $(\lambda_i, \mathbf{v}_i)$, $i = 1, 2, \ldots, n$. The computational complexity of the Matlab function `eig` is $O(n^3)$. Thus, we can note that the eigenvalue decomposition (15) of the incremental method has much lower computational complexity $\left(O\left((p+1)^3\right)\right)$ than the eigenvalue decomposition (4) of the shift-window method $\left(O(N^3)\right)$.

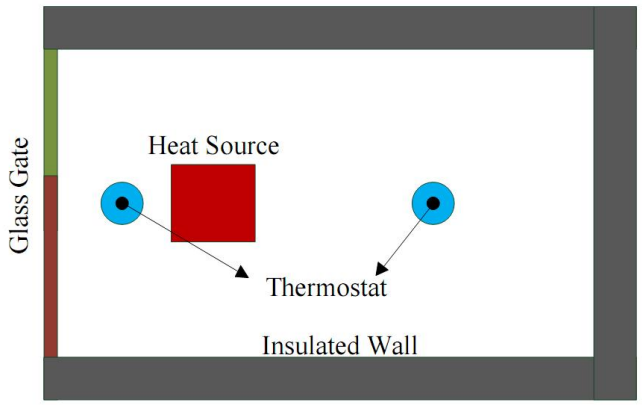

Fig. 1. Schematic of a container with a heat source and two thermostats.

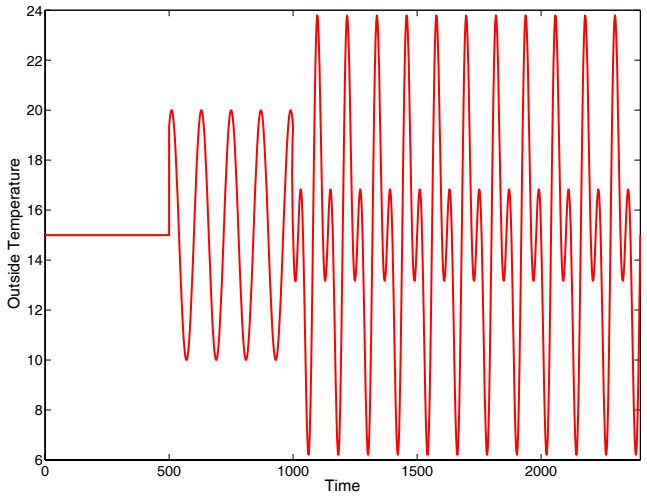

Fig. 2. Outside temperature evolution.

VI. Numerical Example

As shown in Section IV, we do not follow a classical integral-type Galerkin approach to derive finite-dimensional models from the projections of the PDEs onto chosen subspaces. We take advantage of the availability of high-dimensional discrete numerical models used for simulations and the generation of data employed for POD mode extraction. As shown above, we can obtain low-dimensional models by implementing matrix transformations directly on the discrete numerical models. The projection matrices carry the subspace information extracted from the simulation data. The computation of the integrals in the Galerkin projection have been embedded in various commercial software packages for the derivation of the simulation-oriented highly-dimensional discrete models. Instead of obtaining the discrete model directly from weak form integrals, the combination of computational software and model order reduction matrix operations can save much computational burden for control engineers while dealing with complex physical systems.

This idea motivates us to create a general MOR package to connect with various PDE numerical codes (e.g., Matlab, Fluent, COMSOL Multiphysics, etc.). For the development of such a package it is necessary to know how the various numerical codes save the discrete models after implementing the spatial domain decomposition/discretization. In this work, we choose the powerful COMSOL Multiphysics numerical software to connect with our POD-MOR algorithms.

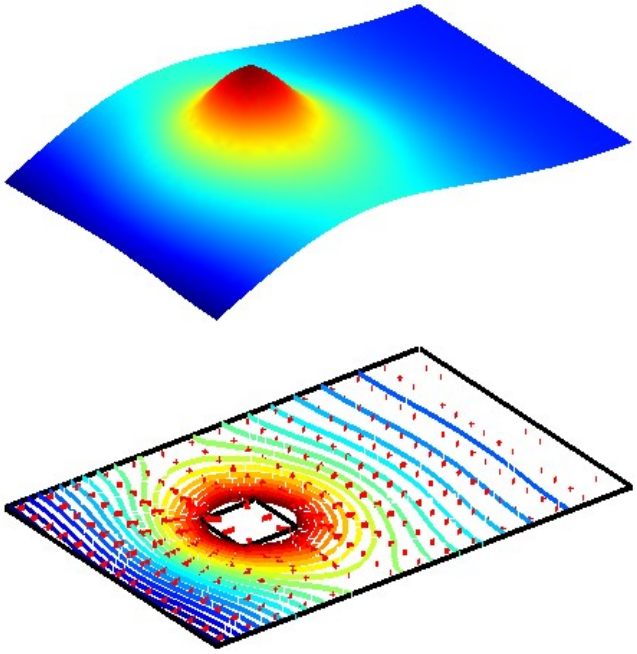

Fig. 3. Simulation snapshot at $T = 2400$s generated by COMSOL Multiphysics. The surface at the top is the 3D plot of the temperature field while the figure at the bottom is the temperature contours within the physical domain.

We consider a two dimensional heat transfer problem in a container with a glass gate on the left side (as shown in Fig. 1). Only heat conduction takes place. Air flows and ventilation are not considered. Then, we can use the heat equation to model the system dynamics, i.e.,

$$\rho C \frac{\partial T}{\partial t} - \nabla \cdot (k \nabla T) = Q. \tag{30}$$

The boundary conditions depend on the level of insulation around the system. On well-insulated sides the heat flux is zero, which gives the Neumann boundary condition $\mathbf{n} \cdot (k\nabla T) = 0$. On poorly insulated sides the Neumann condition is modified as $\mathbf{n} \cdot (k\nabla T) = \frac{k_g}{l_g}(T_{out} - T)$, where k_g and l_g are the thermal conductivity and the thickness of the glass sheet that separates the container and the exterior. We use the same parameters defined in the Model Library of COMSOL Multiphysics 3.5 (pages 348-359 of [19]). We assume that the heat source is kept open but the outside temperature is time varying. We consider an outside temperature evolution, shown in Fig. 2, of the following form

$$T_{out}(t) = \begin{cases} 15, & t \in [0, 500], \\ 15 + 5\sin(\frac{\pi}{60}t), & t \in [500, 1000], \\ 15 + 5\sin(\frac{\pi}{60}t) + 5\sin(\frac{\pi}{30}t), & t \geq 1000. \end{cases}$$

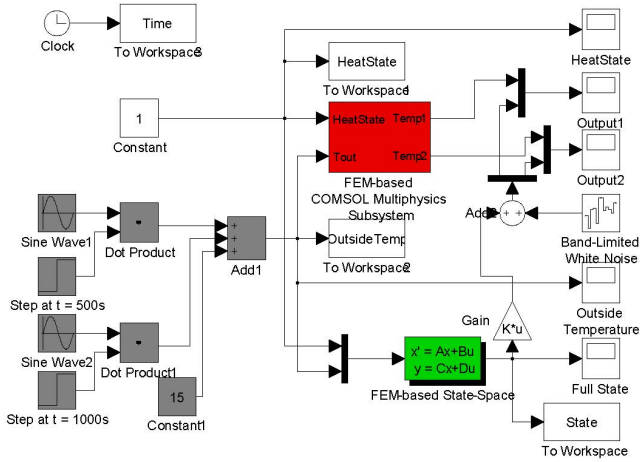

Fig. 4. Simulation of the ODEs extracted from COMSOL Multiphysics using FEM. The red-shaded FEM-based COMSOL Multiphysics Subsystem block is exported using the COMSOL Multiphysics Simulink Model Export while the green-shaded FEM-based State-space block is extracted by the COMSOL Multiphysics State-Space Model Export.

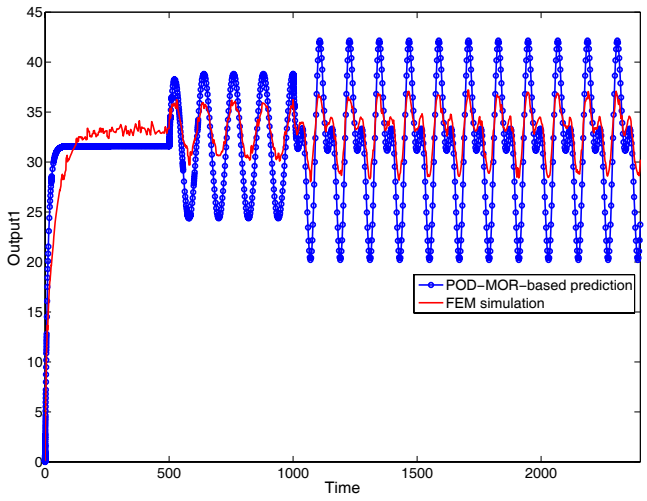

Fig. 5. Temperature measured at the left thermostat.

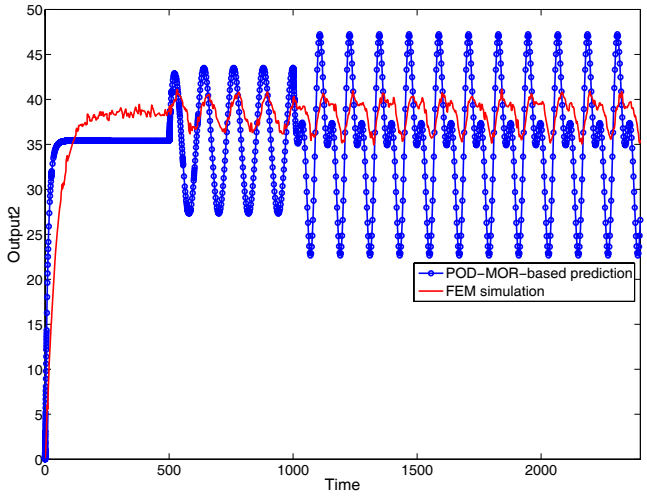

Fig. 6. Temperature measured at the right thermostat.

Using the finite element method (FEM) implemented by the numerical software package COMSOL Multiphysics, we can generate the temporal-spatial evolution of this process. The snapshot at $t = 2400$s is shown in Fig. 3. The high dimensional FEM-based state space representation (16) can be exported from COMSOL Multiphysics. Fig. 4 compares the FEM-based model and the state space model. The `FEM-based COMSOL Multiphysics Subsystem` block is extracted using **COMSOL Multiphysics Simulink Model Export**. The `FEM-based State-Space` block is made up of the matrices $\mathbf{A}, \mathbf{B}, \mathbf{C}$ which are obtained by using **COMSOL Multiphysics State-Space Export**.

For the POD mode extraction, we first use the historic data generated from the numerical simulation over the time interval $[0, 500]$. The first POD mode carries a dominant portion of the information in the generated simulation data. Then, this mode is used to generate a one dimensional model to approximate the heat conduction process. However, the error between the dynamics of both the original high-dimensional model and the reduced one-dimensional model, which is shown in Figs. 5-6 for both thermostat temperatures, is larger than what is desired.

Because of the unsatisfactory approximation accuracy (shown in Figs. 5-6), we need to use new simulation data to update the POD modes in an incremental way. By setting an appropriate error threshold value η, the proposed incremental POD scheme can pick the poorest approximated snapshots to supplement the snapshot ensemble collected over the time interval $[0, 500]$. Two poorly approximated snapshots are chosen and added one after another. In each update of the ensemble an eigenvalue decomposition problem is solved to provide a rotation matrix transformation. After two subsequent incremental POD computation procedures, the approximation accuracy is improved and the output signals at the two measurement points are shown in Figs. 7-8. The differences between the predictions by the FEM-based high-dimensional model and the iPOD-based low-dimensional model for the temperatures at the two measurement points are reduced to acceptable levels, which are much smaller that those in Figs. 5-6. This shows the effectiveness of the MOR approach proposed in this paper. The comparisons (Figs. 5-6 and Figs. 7-8) are carried out using the Matlab/Simulink block diagram shown in Fig. 9 where the `POD/MOR State-Space` block and the `iPOD/MOR State-Space` block represent low dimensional models obtained by the classic POD method and the incremental POD method, respectively.

VII. Conclusions

A incremental POD method is proposed in this paper to provide online POD mode subspace updates when new observation or simulation data becomes available. Instead of using the weak form of the PDE's, which requires spatial integrations over the whole physical domain, we use high-dimensional ODE models generated by simulation software through spatial discretization. Matrix transformations are

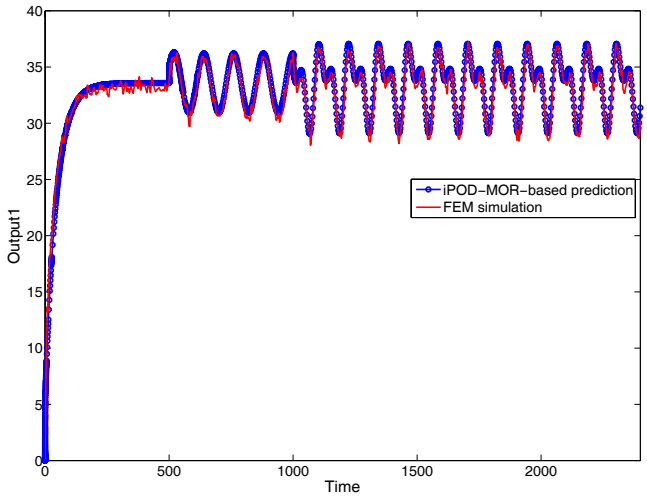

Fig. 7. Temperature measured at the left thermostat.

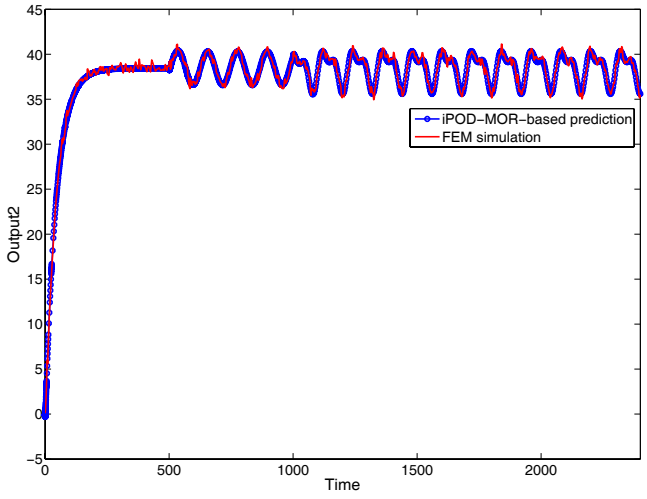

Fig. 8. Temperature measured at the right thermostat.

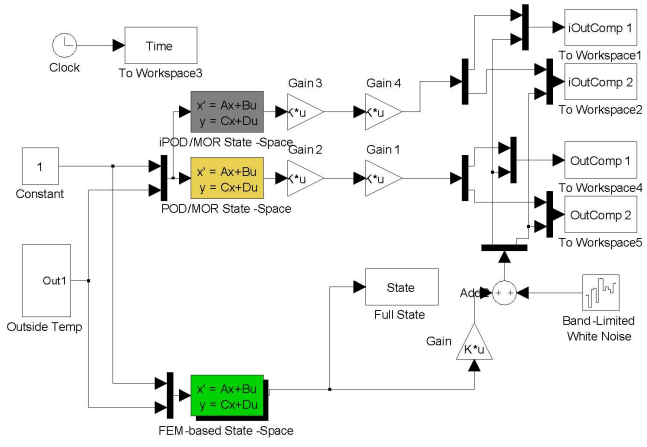

Fig. 9. Simulation comparisons of reduced order models using both the classic POD method and the incremental POD method.

used to obtain low-dimensional models. We propose a recursive matrix transformation computation method that can save significant computational effort. We provide a detailed computational analysis in this paper for the calculation of both the POD modes and the matrix transformations. A comparison is carried out in terms of computational complexity between the incremental POD and the shift-window POD methods. A numerical simulation study based on a 2D heat conduction process in a container has been used to illustrate the effectiveness of the proposed method.

REFERENCES

[1] B. Moore, "Principal component analysis in linear systems: Controllability, observability, and model reduction," *IEEE Transactions on Automatic Control*, vol. 17, pp. 17–32, 1981.

[2] K. Willcox and J. Peraire, "Balanced model reduction via the proper orthogonal decomposition," *AIAA Journal*, vol. 40, pp. 2323–2330, 2002.

[3] C. Rowley, "Model reduction for fluids using balanced proper orthogonal decomposition," *International Journal of Bifurcation & Chaos*, vol. 15, pp. 997–1013, 2005.

[4] A. Antoulas, *Approximation of Large-Scale Dynamical Systems*. Philadelphia: SIAM, 2005.

[5] Y. Jiang, *Model Order Reduction Techniques*. Beijing: Science Press, 2010.

[6] K. Pearson, "On lines and planes of closest fit to a system of points in space," *Philosophical Magzine*, vol. 2, pp. 609–629, 1833.

[7] P. Holmes, J. Lumley, and G. Berkooz, *Turbulence, Coherent Structures, Dynamical Systems and Symmetry*. Cambridge, UK: Cambridge University Press, 1996.

[8] M. Efe and H. Ozbay, "Proper orthogonal decomposition for reduced order modeling: 2D heat flow," *The Proceedings of the 2003 IEEE Conference on Control Applications*, pp. 1273–1277, 2008.

[9] Y. Liang, W. Lin, H. Lee, S. Lim, K. Lee, and H. Sun, "Proper orthogonal decomposition and its applications - Part II: model reduction for MEMS dynamical analysis," *Journal of Sound and Vibration*, vol. 256, pp. 515–532, 2002.

[10] D. Lucia, P. Beran, and W. Silva, "Aeroelastic system development using proper orthogonal decomposition and volterra theory," *AIAA Structures, Structural Dynamics, and Materials Conference, AIAA, Norfolk, VA*.

[11] P. Hall and R. Martin, "Incremental eigenanalysis for classification," *Proceedings of British Machine Vision Conference*, vol. 1, pp. 286–295, 1998.

[12] Y. Li, "On incremental and robust subspace learning," *Pattern Recognition*, vol. 37, pp. 1509–1518, 2004.

[13] D. Ross, J. Lim, R. Lin, and M. Yang, "Incremental learning for robust visual tracking," *International Journal of Computer Vision*, vol. 77, pp. 125–141, 2007.

[14] K. Kunisch and S. Volkwein, "Galerkin proper orthogonal decomposition methods for parabolic problems," *SIAM Journal on Numerical Analysis*, vol. 40, pp. 492–515, 2002.

[15] S. Volkwein, "Lecture notes on proper orthogonal decomposition: Applications in optimization and control," *CEA-EDF-INRIA Summer School Numerical Analysis Summer School Model Reduction and Reduced Basis methods: Application in Optimization, Saint-Lambert-des-Bois, Frankreich*.

[16] L. Sirovich, "Turbulence and the dynamics of coherent structures, Parts I-III," *Quarterly of Applied Mathematics, XLV*, vol. 42, pp. 561–590, 1987.

[17] C. Xu, L. Luo, and E. Schuster, "On the recursive proper orthogonal decomposition method and applications to distributed sensing in cyber-physical systems," *The Proceedings of the 2010 American Control Conference, 2010*, pp. 4905–4910, 2010.

[18] G. Golub and C. V. Loan, *Matrix Computation (2nd Edition)*. Baltimore: The John Hopkins University Press, 1989.

[19] www.comsol.com, *Comsol Multiphysics Model Library*. Comsol Version 3.5, 2008.