A Computationally Optimal Randomized Proper Orthogonal Decomposition Technique

Dan Yu, Suman Chakravorty

Abstract—In this paper, we consider the model reduction problem of large-scale systems, such as systems obtained through the discretization of partial differential equations. We propose a computationally optimal randomized proper orthogonal decomposition (RPOD∗) technique to obtain the reduced order model by perturbing the primal and adjoint system using Gaussian white noise. We show that the computations required by the RPOD∗ algorithm is orders of magnitude cheaper when compared to the balanced proper orthogonal decomposition (BPOD) algorithm and BPOD output projection algorithm while the performance of the RPOD∗ algorithm is much better than BPOD output projection algorithm. It is computationally optimal in the sense that a minimal number of snapshots is needed. We also relate the RPOD∗ algorithm to random projection algorithms. The method is tested on two advection-diffusion problems.

Keywords—Model Reduction, Proper Orthogonal Decomposition (POD), Randomization Algorithm.

I. INTRODUCTION

In this paper, we are interested in the model reduction of large scale systems such as those governed by partial differential equations (PDE). The dimension of the system is large due to the discretization of the PDEs. For instance, consider the atmospheric dispersion of an air pollutant [1]. The emission of the contaminants on the ground level is shown in Fig. 1 with four point sources labeled from S1 to S4.

This is a three dimensional problem, and after discretizing the PDE, the dimension of the system is $10^6$. Therefore, we are interested in constructing a reduced order model (ROM) that can capture the input/output characteristics of the large model such that this ROM can then be used by a filtering algorithm for updating the states of the field, such as the Kalman filter. Also, the actuators and sensors could be placed anywhere in this field, which leads to a model reduction problem of a large-scale system with a large number of inputs/outputs. There are two popular contemporary model reduction techniques that have been studied in the past few decades: Principal component analysis (PCA) and randomization algorithms.

Among the PCA algorithms, Balanced Proper Orthogonal Decomposition (BPOD) [2], [3] based on the balanced truncation [4] and the snapshot Proper Orthogonal Decomposition (POD) technique [5] has been widely used. Balancing transformations are constructed using the impulse responses of both the primal and adjoint system, and hence, the most controllable and observable modes can be kept in the ROM. In 1978, Kung [6] presented a new model reduction algorithm in conjunction with the singular value decomposition (SVD) technique, and the eigensystem realization algorithm (ERA) [7] was developed based on this technique. The BPOD is equivalent to the ERA procedure [8], and forms the Hankel matrix using the primal and adjoint system simulations as opposed to the input-output data as in ERA. More recently, there has been work on obtaining information regarding the dominant modes of the system based on the snapshot POD, followed by an eigendecomposition of an approximating linear operator, called the dynamic mode decomposition (DMD) [9], [10]. In [11], [12], an adaptive POD algorithm based on the snapshot POD algorithm is introduced to recursively revise the locally valid ROMs. When there are new snapshots collected, decisions are made if the basis functions need to be updated. Both the DMD and adaptive POD algorithm are applicable for the nonlinear systems.

The primary drawback of the BPOD and ERA is that for a large scale system, such as that obtained by discretizing a PDE, with a large number of inputs/outputs, the computational burden incurred is very high. There are two main parts to the computation: the first is to collect datasets from computationally expensive primal and adjoint simulation in order to generate the Hankel matrix. The second part is to solve the SVD problem for the resulting Hankel matrix.

To reduce the computational cost of BPOD, improved algorithms have been proposed. For example, [13] proposed an output projection method to address the problem when the number of outputs is large. The outputs are projected onto a small subspace via an orthogonal projection $P_r$ that minimizes the error between the full impulse response and the projected impulse response. However, the method cannot make any claim regarding the closeness of the solution to one that is obtained from the full Hankel matrix, and is still faced with a very high computational burden when both the numbers of inputs...
and outputs are large. There have also been methods proposed \([13]\) to reduce the number of snapshots, however, the primary problem regarding large number of inputs/outputs remains the same.

There are two major classes of randomization algorithms used for low-rank matrix approximations and factorizations: random sampling algorithms and random projection algorithms. For a large scale matrix \(H\), random sampling algorithms construct a rank \(k\) approximation matrix \(H\) by choosing and rescaling some columns of \(H\) according to certain sampling probabilities \([14]\), so the error satisfies \(\|H - Q \Omega' Q^* H\| \leq (1 + \epsilon)\|H - H(k)\|\) with high probability, where \(H(k)\) is a best rank \(k\) approximation of \(H\), \(\epsilon\) is a specified tolerance, and \(\|H\|_F\) denotes the Frobenius norm of \(H\). The improved algorithm proposed in \([15]\) is to sample some columns according to leverage scores, where the leverage scores are calculated by performing the SVD of \(H\), so that the error satisfies \(\|H - H\|_F \leq (1 + \epsilon)\|H - H(k)\|_F\), with high probability. A direct application of both algorithms would require the full Hankel matrix to be constructed, however, such a construction problem regarding large number of inputs/outputs remains the same.

In random projection method \([16]\), the large matrix \(H\) is projected on to an orthonormal basis \(Q\) such that the error satisfies \(\|H - QQ^* H\| \leq (1 + \epsilon)\|H - H(k)\|\) with high probability, where \(\|H\|\) denotes the spectral 2-norm of \(H\), \((.)^\dagger\) denotes the conjugate transpose of \((.)\). A Gaussian test matrix \(\Omega\) is generated, and the orthonormal basis \(Q\) is constructed by performing a QR factorization of the matrix product \(HQ\Omega\). The bottleneck of this algorithm remains, as above, the construction of the full Hankel matrix, which is computationally prohibitive owing to the size of the problem.

We had introduced an RPOD algorithm in \([17]\) that randomly chooses a subset of the input/output trajectories. A sub-Hankel matrix is constructed using these sampled trajectories, which is then used to form a ROM in the usual BPOD fashion. The Markov parameters of the ROM constructed using the sub-Hankel matrix were shown to be close to the Markov parameters of the full order system with high probability. We proved that a lower bound exists for the number of the input/output trajectories that need to be sampled. The major problem associated with this algorithm is that different choices of the sampling algorithms would lead to different lower bounds, and the choice of a good sampling algorithm other than the uniform distribution is unclear.

In this paper, we propose the RPOD* algorithm which is closely related to the random projection algorithm. In the RPOD* algorithm, we perturb the primal and adjoint system with Gaussian white noise, and we prove that similar to the BPOD algorithm, the controllable and observable modes are retained in the ROM. The Markov parameters of the ROM constructed using RPOD* are shown to be close to the Markov parameters of the full order system, while the error is bounded. The main contribution of the RPOD* method is that it is computational orders of magnitudes less expensive when compared to the BPOD/ERA and randomization algorithms for a large-scale system with a large number of inputs/outputs. The RPOD* algorithm can be viewed as applying the random projection on the full Hankel matrix \(H\) twice without constructing the full Hankel matrix \(H\), i.e., \(\hat{H} = \Omega_2 Z' X' \Omega_1 = \Omega_2 Z' \Omega_1\), where \(\Omega_1, \Omega_2\) are two random projection matrices, and \(Z, X\) are the usual impulse response matrices of the adjoint and primal system. However, we actually only generate \(Z\Omega_2\) and \(X\Omega_1\) which can be constructed from a single white noise perturbed trajectory each of the adjoint and primal system respectively, and thus, are orders of magnitude smaller in size than the impulse responses \(Z\) and \(X\). Thus, the computational cost to generate the Hankel matrix and to solve the SVD problem is saved by orders of magnitude. We believe that it is the most computational efficient POD algorithm. In practice, the RPOD* algorithm can be solved in real-time.

The rest of the paper is organized as follows. In Section \([11]\) the problem is formulated, and in Section \([11]\) we review the BPOD algorithm and illustrate in a simplified fashion how to relate the BPOD ROM to the controllable and observable modes of the system. The RPOD* algorithm is proposed in Section \([11]\) and the formal proofs and results are shown. Also, we discuss some implementation problems in this section. In Section \([11]\) we compare the RPOD* algorithm with BPOD, random projection and BPOD output projection algorithm. In Section \([11]\) we provide computational results comparing the RPOD* with the BPOD/BPOD output projection for a one dimensional heat transfer problem and a three dimensional atmospheric dispersion problem.

II. Problem Formulation

Consider a stable linear input-output system:

\[
x_k = Ax_{k-1} + Bu_k, \quad y_k = Cx_k,
\]

where \(x_k \in \mathbb{R}^N\), \(u_k \in \mathbb{R}^p\), \(y_k \in \mathbb{R}^q\) are the states, inputs, and outputs at discrete time instant \(t_k\) respectively. Assume that \(A, B, C\) matrices are real.

The adjoint system is defined as:

\[
z_k = A'z_{k-1} + C'v_k, \quad w_k = B'z_k,
\]

where \(z_k \in \mathbb{R}^N\), \(w_k \in \mathbb{R}^p\) is the state and output of the adjoint system at time \(t_k\) respectively, \(v_k \in \mathbb{R}^q\).

**Definition 1.** The Markov parameters of the system is defined as \(CA'B, i = 1, \ldots\).

**Assumption 1.** Assume that \(A\) is diagonalizable and stable. Under Assumption 1, let

\[
A = V \Lambda \Lambda',
\]

where \(\Lambda\) are the eigenvalues, \((V, U)\) are the corresponding right and left eigenvectors.

**Definition 2.** A mode \((A_i, U_i, V_i)\) is not controllable if \(U_i'B = 0\), and is not observable if \(CV_i = 0\), where \((A_i, V_i, U_i)\) is the \(i^{th}\) eigenvalue-eigenvector pair.

We partition the eigenvalues and eigenvectors \((\Lambda, V, U)\) into
four parts:

\[
A = \begin{pmatrix}
V_{co}^r & \Lambda_{co} & V_{co}^o \\
V_{es}^r & \Lambda_{es} & V_{es}^o \\
\end{pmatrix},
\]

where \((\Lambda_{co}, V_{co}, U_{co})\) are the controllable and observable modes, \((\Lambda_{es}, V_{es}, V_{es}^o)\) are the controllable but unobservable modes, \((\Lambda_{es}, V_{es}, U_{es})\) are the uncontrollable but observable modes, and \((\Lambda_{es}, V_{es}, U_{co})\) are the uncontrollable and unobservable modes.

In this paper, we consider the model reduction problem for large-scale systems with a large number of inputs/outputs. The goal is to construct an ROM such that the outputs of the ROM \(y_f\) are close to the outputs of the full order system \(y\), i.e., \(|y - y_f|\) is small.

Denote the number of the controllable and observable modes is exactly \(l\) throughout the paper. First, we summarize all the assumptions made in this paper.

- A1. \(A\) is stable and diagonalizable.
- A2. \(l \ll N\).
- A3. \(U_{co}^r B = 0, U_{co}^o B = 0, CV_{co} = 0, CV_{co} = 0\).
- A4. \(U_{es}^r B = \epsilon C_1, U_{es}^o B = \epsilon C_2, CV_{es} = \epsilon C_3, CV_{es} = \epsilon C_4\), where \(\epsilon\) is a small number. \(C_1, C_2, C_3, C_4\) are constant matrices. And if \(\|U_{es}^r B\| = O(||C_3||), \|C_{es}|| = O(||C_3||), \) then \(\|C_1||, \|C_2||, \|C_3||, \|C_4|| = O(||C_3||), \) where \(C_2\) is a constant matrix.

The structure of the paper is summarized in Figure 2.

**Discussion on Assumptions.** For most of the practical applications we consider, A1 is satisfied. A2 needs to be satisfied for the system to have an ROM, this assumption is typically satisfied for a large-scale system. It should be noticed that from Definition 2, assumption A3 is the statement of controllability/observability of the different modes of the system. However, in practice, \(U_{es}^r B, CV_{es}\) are never exactly zero, and hence, in assumption A4, we assume that \(\|U_{es}^r B\| \ll \epsilon, \|CV_{es}\| \ll \epsilon, \) where \(\epsilon\) is small.

### III. Simplified Analysis

In this section, first, we briefly review the BPOD algorithm, and then illustrate in a simplified fashion how the transformation bases and the Markov parameters of the ROM constructed using the BPOD algorithm can be related to the controllable and observable modes \((\Lambda_{co}, V_{co}, U_{co})\) of the system matrix \(A\). The simplified analysis is critical to understand the intuition behind the proposed RPOD* algorithm in Section [IV](#IV).

#### A. BPOD Algorithm

Consider the stable linear system (1)-(2), let \(B = [b_1, \cdots, b_p]\) where \(b_i, i = 1, \cdots, p\) is the \(i^{th}\) column of \(B\), and \(C' = [c_1', \cdots, c_q']\), where \(c_j, j = 1, \cdots, q\) is the \(j^{th}\) row of \(C\). The BPOD Algorithm [3] is summarized in Algorithm 1.

#### B. Simplified Analysis

First, we construct a modal BPOD ROM by projecting the BPOD bases \((T_b, S_b)\) onto the ROM eigenspace as in Algorithm [2].

Under assumptions A1, A2, and A3, we have the following result.

---

**Fig. 2. Structure of the Paper**
Algorithm 2 BPOD modal ROM Algorithm

1) Construct BPOD ROM \((A_b, B_b, C_b)\) and BPOD bases \((T_b, S_b)\) using BPOD Algorithm [1]
2) Solve the eigenvalue problem of \(A_b\):
\[
A_b = P_b \Lambda_b P_b^{-1},
\]
where \(\Lambda_b\) are the eigenvalues, and \(P_b\) are the corresponding eigenvectors.
3) Construct BPOD modal bases:
\[
\Phi_b = P_b^{-1} S_b, \Psi_b = T_b P_b,
\]
(12)
4) The modal ROM is:
\[
\hat{A}_b = \Phi_b A_b \Phi_b, \hat{B}_b = \Phi_b B, \hat{C}_b = C \Psi_b.
\]
(13)

**Proposition 1:** Denote \((\hat{A}_b, \hat{B}_b, \hat{C}_b)\) as the modal ROM constructed using Algorithm 2 \((\Phi_b, \Psi_b)\) are BPOD modal bases. Then \(A_b = \Phi_b A \Psi_b = \Lambda_{co}, B_b = \Phi_b B = U_c^o B, C_b = C \Psi_b = C V_{co}, \) where \((\Lambda_{co}, U_{co}, V_{co})\) are the controllable and observable modes of the system, and \(C_b \hat{A}_b \hat{B}_b = C A' B, i = 1, 2, \ldots .\)

**Proof:** Consider the snapshots in the primal snapshot ensemble \((5)\),
\[
\begin{align*}
x_i(t_k) &= A^{tk} b_i, \\
&= x_k(t_k) = A^{tk} B,
\end{align*}
\]
(14)
and the snapshot ensemble \(X_b\) can be written as:
\[
X_b = \begin{pmatrix} A^{k1} & A^{k2} & \cdots & A^{km} \end{pmatrix}.
\]
(16)
Under assumptions A1 and A3, and from \((4)\), we have:
\[
A^{tk} B = (V_{co}) \begin{pmatrix} \Lambda_{co} & \Lambda_{co} \\ \Lambda_{co} & \Lambda_{co} \end{pmatrix} \begin{pmatrix} U_{co}^t & U_{co}^t \end{pmatrix} B,
\]
(17)
where \(\alpha_k^{co}, \alpha_k^{co}\) are the coefficient matrices. Substitute \((17)\) into \((16)\), and \(X_b\) can be written as:
\[
X_b = \begin{pmatrix} V_{co} & V_{co} \end{pmatrix} \begin{pmatrix} \alpha_k^{co} & \alpha_k^{co} \\ \alpha_k^{co} & \alpha_k^{co} \end{pmatrix},
\]
(18)
where \(\alpha_k^{co}, \alpha_k^{co}\) the coefficient matrices. Similarly,
\[
Z_b = \begin{pmatrix} U_{co} & U_{co} \end{pmatrix} \begin{pmatrix} \beta_k^{co} & \beta_k^{co} \\ \beta_k^{co} & \beta_k^{co} \end{pmatrix},
\]
(19)
where \(\beta_k^{co}, \beta_k^{co}\) are some coefficient matrices.
Hence,
\[
Z_b X_b = ((\beta_k^{co}) U_{co}^t + (\beta_k^{co}) U_{co}^t) (V_{co} \alpha_k^{co} + V_{co} \alpha_k^{co}),
\]
(20)
where \(U_{co}^t V_{co} = 0, U_{co}^t V_{co} = 0, U_{co}^t V_{co} = 0, U_{co}^t V_{co} = 0, \ldots \) Under assumption A3, the \(l\) non-zero singular values in the resulting SVD problem, i.e.,
\[
Z_b X_b = L_b \Sigma_b R_b,
\]
(21)
where \(\Sigma_b \in \mathbb{R}^{l \times l}\) are the \(l\) non-zero singular values and \((L_b, R_b)\) are the corresponding left and right singular vectors. Moreover,
\[
Z_b A X_b = (\beta_k^{co}) \Lambda_{co} \alpha_k^{co}.
\]
(22)
Now, consider the BPOD ROM:
\[
A_b = S_b A T_b = \Sigma_b^{-1/2} L_b (Z_b A X_b) R_b \Sigma_b^{-1/2} = \Sigma_b^{-1/2} L_b \Phi_b \Lambda_{co} \alpha_k^{co} R_b \Sigma_b^{-1/2}.
\]
(23)
We show that \(\Lambda_{co}\) are the eigenvalues of \(A_b\) and \(P_b\) are the eigenvectors as follows:
\[
P_b \hat{P}_b = \Sigma_b^{-1/2} L_b \Phi_b \Lambda_{co} \alpha_k^{co} R_b \Sigma_b^{-1/2} = \Sigma_b^{-1/2} L_b \Sigma_b R_b \Sigma_b^{-1/2} = I.
\]
(24)
Also,
\[
\hat{P}_b P_b = \alpha_k^{co} R_b \Sigma_b^{-1/2} L_b \Phi_b \Lambda_{co} \alpha_k^{co} \alpha_k^{co} \alpha_k^{co} \alpha_k^{co} = I,
\]
(25)
where \((\cdot)^+\) denotes the pseudoinverse of \((\cdot)\). Hence, \(\hat{P}_b = P_b^{-1}\) and from \((23)\),
\[
\Lambda_{co} = (P_b^{-1} S_b) A (T_b P_b) \cdot \Phi_b \Psi_b,
\]
(26)
Also, we prove that \(\Psi_b, \Phi_b\) are biorthogonal as follows.
\[
\Psi_b \Phi_b = T_b P_b P_b^{-1} S_b = T_b S_b,
\]
(27)
where \((T_b, S_b)\) are BPOD bases, and are biorthogonal. Hence,
\[
\Psi_b = T_b P_b P_b^{-1} S_b = T_b S_b,
\]
(28)
\[
\Phi_b \Psi_b = P_b^{-1} S_b T_b P_b = P_b^{-1} P_b = I,
\]
(29)
where \(C_b = \alpha_k^{co} R_b \Sigma_b^{-1} L_b (\beta_k^{co})^t\). Similarly,
\[
\Phi_b = P_b^{-1} S_b = U_{co}^t + C T U_{co},
\]
(30)
where \(C_6 = \alpha_k^{co} R_b \Sigma_b^{-1} L_b (\beta_k^{co})^t\). Under assumption A3, the modal BPOD ROM constructed using \((\Psi_b, \Phi_b)\) is:
\[
\hat{A}_b = \Phi_b A \Phi_b = \Lambda_{co},
\]
\[
\hat{B}_b = \Phi_b \Psi_b = U_{co} B + C T U_{co} B = U_{co} B,
\]
\[
\hat{C}_b = C \Psi_b = C V_{co} + C V_{co} C_6 = C V_{co},
\]
(31)

**Discussion on Proposition [1]** Recall the impulse response
snapshot ensembles collected in the BPOD are:

\[
X_b = V_{co} \alpha_{co}^b + V_{co} \alpha_{co}^b Z_b = U_{co} \beta_{co} + U_{co} \beta_{co}^b
\]

and the Hankel matrix is:

\[
H_b = Z_b^* X_b = (z_{gb})^T \alpha_{co}^b \in \mathbb{R}^{qm \times pm}.
\]

There are two main parts to the computation:

1) The primal and adjoint snapshot ensembles \(X_b \in \mathbb{R}^{N \times pm}, Z_b \in \mathbb{R}^{N \times qn}\), and hence, the construction of \(H_b\) takes time \(O(pqmN)\).

2) The computational cost to solve the SVD of \(H_b\) is \(O(\min\{p^2m^2qn, pmq^2n^2\})\).

However, \(H_b\) is only rank “\(l\)”, where \(l \leq N, pm, qn\), and from the development of Proposition 1, we see that the modal ROM given in (33) is completely determined by the \(l\) controllable and observable modes and is invariant to the data \(X_b\) and \(Z_b\), i.e., as long as the snapshot ensembles can be written as:

\[
X^* = V_{co} \alpha^* + V_{co} \alpha^*,
\]

\[
Z^* = U_{co} \beta^* + U_{co} \beta^*,
\]

where \(\alpha^*, \beta^*\) are rank \(l\) constant matrices, and \(\alpha^*, \beta^*\) are some constant matrices of suitable dimensions, then under assumptions A1, A2, and A3, the following corollary holds.

Corollary 1: Denote \((\alpha^*, B^*, C^*)\) as the modal ROM constructed using Algorithm 2 with snapshot ensembles \(X^*, Z^*\) as in (34). Then \(A^* = \Lambda_{co}, B^* = U_{co} B, C^* = CV_{co}\), where \((\Lambda_{co}, U_{co}, V_{co})\) are the controllable and observable modes of the system, and \(C^* (A^*)^* B^* = CA^* B, i = 1, 2, \ldots\).

Corollary 1 can be proved by replacing \(X_b, Z_b\) in Proposition 1 with \(X^*, Z^*\), and the proof is omitted here. We make the following observations.

Observation: 1) The snapshot ensembles do not have to be collections of the impulse responses as in BPOD. 2) Only \(l\) snapshots may be enough to extract all the controllable and observable modes of the system.

Bearing this observation in mind, in the next section, we introduce the RPOD* algorithm which generates the snapshot ensembles using exactly “\(l\)” snapshots, such that the Corollary 1 holds.

IV. COMPUTATIONALLY OPTIMAL RANDOMIZED PROPER ORTHOGONAL DECOMPOSITION (RPOD*)

In this section, first, we define the computationally optimal snapshot ensemble, and prove that the snapshot ensembles generated by perturbing the primal/adjoint system with white noise are computationally optimal snapshot ensembles. Then we propose the RPOD* algorithm in Section IV-B. We discuss implementation issues of the algorithm in Section IV-C.

A. Computationally Optimal Snapshot Ensemble

Under assumptions A1, A2 and A3, we define a computationally optimal snapshot ensemble of the system as follows.

Definition 3: A computationally optimal primal snapshot ensemble \(X^*\) is an \(l\)-snapshot ensemble of rank \(l\) which can be written as in Eq. (34), i.e., \(m = l\).

A similar definition suffices for the computationally optimal adjoint snapshot ensemble \(Z^*\).

Consider the system (1), (2), under assumption A1 that \(A\) is stable, there exists a finite number \(t_{ss}\), such that \(\|A^{t_{ss}}\| \approx 0\). Under assumptions A1, A2 and A3, we have the following result.

Proposition 2: Perturb the primal system (1) with white noise \(u_0\), and collect \(m\) snapshots at time \(t_1, t_2, \ldots, t_m\), where \(t_m \geq t_{ss}\), and \(\|A^{t_{ss}}\| = 0\). Denote the snapshot ensemble as \(X_r = (x_1 x_2 \cdots x_m)\). If \(m = l\), where \(l\) is the number of the controllable and observable modes of the system, then \(X_r\) is a computationally optimal snapshot ensemble.

Proof: For the snapshots taken before \(t_{ss}\), the state snapshot \(x_k\) at time \(k\) is:

\[
x_k = \sum_{i=0}^{k-1} A^i B u(k-i), k \leq t_{ss}.
\]

Suppose there is a snapshot ensemble \(X_f\) which takes \(t_{ss}\) snapshots at time \(k = 1\) to \(k = t_{ss}\), then from (36), the snapshot ensemble \(X_f\) can be written as:

\[
X_f = (x_1 x_2 \cdots x_{t_{ss}}) = (B AB \cdots A^{t_{ss}-1} B) X_b
\]

\[
\left(\begin{array}{cccc}
(\omega(1) & u(2) & \cdots & u(t_{ss}-1) & u(t_{ss}) \\
0 & u(1) & \cdots & u(t_{ss}-2) & u(t_{ss}-1) \\
0 & 0 & \cdots & u(t_{ss}-3) & u(t_{ss}-2) \\
\vdots & \vdots & \ddots & \ddots & \vdots \\
0 & 0 & \cdots & 0 & u(1)
\end{array}\right),
\]

where \(X_b\) is the BPOD snapshot ensemble from time \(k = 0\) to \(k = t_{ss} - 1\). Denote \(\Omega = (\omega(1) \omega(2) \cdots \omega(t_{ss}))\), where \(\omega(i)\) is the \(i^{th}\) column of \(\Omega\). The \(\Omega\) matrix above has columns that are linearly independent since it is upper triangular.

Since \(X_r\) consists of \(m\) columns of \(X_f\), \(X_r\) can be written as:

\[
X_r = (x_1 \cdots x_m) = X_b (\omega(1) \cdots \omega(m)),
\]

where \((\omega(1) \cdots \omega(m))\) are the corresponding columns of \(\Omega\), and hence, \(\Omega_1\) has full column rank.

For the snapshot \(x_k\) taken after \(t_{ss}, x_k\) could also be written as: \(x_k = X_b \omega(k)\), where \(\omega(k)\) is a column vector whose entries are white noises. Therefore, \(\omega(k)\) is independent of \(\omega(1), \cdots, \omega(m)\) in (38), and hence, for all the snapshots collected in \(X_r, X_r = X_b \Omega_1\), where \(\Omega_1\) has full column rank.
Recall that from [13], $X_b$ can be written as:

$$X_b = V_{cb} \alpha_{cb}^{b} + V_{cb} \alpha_{c\theta}^{b} \Omega_{c\theta},$$

and $\alpha_{cb}^{b}$ has full row rank. Thus, when $m = l$,

$$X_r = X_b \Omega_1 = V_{cb} \alpha_{cb}^{b} \Omega_1 + V_{cb} \alpha_{c\theta}^{b} \Omega_1,$$

where rank $\alpha_{cb} = l$. Hence, $X_r$ is a computationally optimal snapshot ensemble.

Similarly, we take $n = l$ snapshots by perturbing the adjoint system (2) with white noise $v_k$, and the adjoint snapshot ensemble can be written as:

$$Z_r = U_{co} \beta_{co} + U_{co} \beta_{\theta},$$

where rank $\beta_{co} = l$, and $\beta_{\theta}$ is a constant matrix. Thus, $Z_r$ is a computationally optimal snapshot ensemble. In Section IV-C we discuss about how to choose $m, n$ and the snapshots in practice.

**B. RPOD* Algorithm**

The RPOD* algorithm is summarized in Algorithm 3.

Under assumptions A1, A2 and A4, the following result holds.

**Proposition 3:** Denote $(A_r, B_r, C_r)$ as the ROM constructed using RPOD* following Algorithm 3. If we keep the first $l$ non-zero singular values in (44), then $\|C_r A_r^{r} B_r - C A^{l} B\| \approx O(\epsilon)$, $i = 1, \ldots$, where $\epsilon$ is a small number defined in assumption A4.

The proof of Proposition 3 uses perturbation theory [18], [19] to extend the proof of the idealized Proposition 1 such that A4 holds instead of A3. Under assumption A4, the actual snapshot ensembles can be written as:

$$X_r = (V_{co} \quad V_{c\theta} \quad V_{cb} \quad V_{\theta}) \begin{pmatrix} \alpha_{cb}^{b} \\ \alpha_{c\theta}^{b} \\ \delta_{co} \\ \delta_{c\theta} \end{pmatrix},$$

where $\delta_{co} = co_{cb} = \epsilon \alpha_{cb}$ and $\epsilon$ is small. Therefore, $\|V_{co} \delta_{co} + V_{c\theta} \delta_{c\theta}\| = O(\epsilon)$ are small perturbations of the ideal snapshot ensemble, and we can expect the ideal result to be perturbed by a small amount as well.

The formal proof is shown in Appendix A.

**Corollary 2:** $\epsilon$ is assumed to be a small number in assumption A4, and $\epsilon$ can also be related to $\sigma_{l+1}$ as follows:

$$\|C_r A_r^{r} B_r - C A^{l} B\| \approx O(\epsilon) \approx O(\sigma_{l+1}).$$

The proof is shown in Appendix B.

**C. Implementation Issues**

Here we discuss some implementation problems in the RPOD* algorithm. We give the insight into how to collect

the snapshot ensembles, and how to select the ROM size.

**Snapshot selection** From the analysis in Section IV-A, we only need to collect $m = l$ snapshots from the primal simulations. However, $l$ is not known a priori, thus, in practice, we start with a random guess $m << N$, where $N$ is the dimension of the system, or we can choose $m$ from experience. For instance, in a fluid system with $10^6$ degrees of freedom, $m = O(10) \sim O(10^2)$. Similarly, we guess $n$, and then we check the rank of $Z_r X_r$. If $Z_r X_r$ has full rank, i.e., rank $(Z_r X_r) = \min(m, n)$, then it is possible that we did not take enough snapshots, and hence, we increase $m, n$, until rank $(Z_r X_r) < \min(m, n)$.

For the primal simulation, we take $m$ snapshots from one simulation with zero initial condition, and white noise perturbation $u(k)$. We assume that the snapshots are taken at $\Delta T, 2\Delta T, \ldots, m \Delta T$, WLOG. Here, $\Delta T$ is a small constant, and we require that $m \Delta T \geq t_{ss}$, where $\|A^{t_{ss}}\| \approx 0$. In Fig. 3 we show one simulation result comparing the accuracy of the ROMs using $\Delta T = 3, 5, 10, 20, 50$ for the atmospheric disper-
The zero eigenvalues. If eigenvalues of \( k \) are approximately zero, we know until 

Thus, 
eigenvalues of 

\( \text{inputs and needed for BPOD algorithm, where } p \) we stop and pick the number 

A. Comparison with BPOD algorithm.

BPOD, random projection and BPOD output projection algorithm.

First, we summarize the differences between the BPOD and \( \text{BPOD}^* \) algorithm. From (54), the adjoint snap-

\begin{equation}
X_r = X_b \Omega_1, Z_r = Z_b \Omega_2,
\end{equation}

where \( X_b \in \mathbb{R}^{n \times p_t \times s} \), \( Z_b \in \mathbb{R}^{n \times q_t \times s} \) are the impulse response snapshot ensembles that need to be collected in the BPOD algorithm from time step \( (0, \cdots, t_{ss} - 1) \), and \( \| A(t) \| \approx 0 \). \( \Omega_1 \in \mathbb{R}^{p_t \times m} \) and \( \Omega_2 \in \mathbb{R}^{q_t \times n} \) are full rank matrices. We have:

\begin{equation}
H_r = Z_r^* X_r = \Omega_2^* Z_b^* X_b \Omega_1 = \Omega_2^* H_b \Omega_1 \Omega_1 \Omega_1
\end{equation}

There is a significant difference between the proposed algorithm and a direct application of the random projection algorithm on BPOD. A direct application of the random projection would require to generate the Hankel matrix \( H_b \) (and \( X_b, Z_b \)) first. However, in practice, the construction and the storage of the Hankel matrix is computationally prohibitive when \( N \) is large and the number of inputs/outputs is large. Also, the bottleneck of the random projection algorithm is the projection of \( X_b, Z_b \) onto the Gaussian test matrices. In the proposed algorithm, the snapshot ensembles are constructed directly from the primal and adjoint simulations, and hence, the computational cost to generate the Hankel matrix and to project it onto the Gaussian test matrices is saved.

C. Comparison with BPOD output projection

As we mentioned in Section \( \text{III} \) when the number of the outputs \( q \) is large, the computation of the BPOD adjoint simulations may not be tractable. To reduce the number of the outputs, the output projection method in [3, 21] is proposed. In this section, we compare the \( \text{BPOD}^* \) algorithm with BPOD output projection algorithm. First, we briefly review the BPOD output projection method in Algorithm 4.

In the following, we compare the BPOD output projection method with \( \text{RPOD}^* \) algorithm. From (55), the adjoint snap-
Algorithm 4 BPOD output projection algorithm

1) Collect the primal snapshot ensemble $X_b$ in (5).
2) Compute the POD modes of the dataset $Y_b = C X_b$.

\[ Y_b^T Y_b \phi_k = \lambda_k \phi_k, \lambda_1 \geq \ldots \geq \lambda_n, \lambda_n \geq 0, \]  

where $\lambda_k$ are the eigenvalues, and $\phi_k$ are the corresponding eigenvectors. Thus, the POD modes $\Theta_s = [\phi_1, \ldots, \phi_s]$, where $s$ is the rank of the output projection, and $s \ll q$.

3) Collect the impulse responses of the adjoint system:

\[ z_{k+1} = A' z_k + C' \Theta_s v, w = B' z_k. \]  

The adjoint snapshot ensemble is denoted as $Z_s$.

4) Construct the Hankel matrix:

\[ H_s = Z_s' X_b. \]  

5) Solve the SVD problem of $H_s$, and construct the BPOD output projection bases as $(T_s, S_s)$ using equations (8) and (9). The ROM is:

\[ A_s = S_s A T_s, B_s = S_s B, C_s = C T_s. \]

The adjoint snapshot ensemble $Z_s$ can be written as:

\[ Z_s = (C' \Theta_s, A' C' \Theta_s, \ldots, (A')^s C' \Theta_s) \]

\[ = (C' A' C' \ldots (A')^s C') \Theta_s. \]

And hence, the projected Hankel matrix $H_s$ is:

\[ H_s = \Theta' Z_b' X_b = \Theta' H_b. \]

Recall that the adjoint snapshot ensembles collected in RPOD* can be written as $Z_r = Z_b \Theta$, and the projected Hankel matrix in RPOD* is $H_r = \Omega_2' H_b \Omega_1$. Thus, we make the following remark.

Remark 1: Both the BPOD output projection and RPOD* algorithms can be viewed as projecting the full order Hankel matrix onto a reduced order Hankel matrix with projection matrices $\Theta$ and $(\Omega_1, \Omega_2)$.

1) Differences between two algorithms: First, the information preserved in $H_r$, $H_r$ are not the same. The output projection $P_s = \Theta_s \Theta_s'$ minimizes the 2-norm of the difference between the original transfer function and the output-projected transfer function, i.e. $\| C AB - \Theta_s \Theta_s' C AB \|$, $i = 1, \ldots$, is minimized. Thus, the controllable and observable modes preserved in $H_s$ are an approximation of those in $H_b$, while in RPOD* algorithm, the exact controllable and observable modes are preserved using the Gaussian random projection matrix. As mentioned in [21], when $s < l$, where $l$ is the number of non-zero Hankel singular values (number of controllable and observable modes), then only the first $s$ Hankel singular values are the same as the full balanced truncations Hankel singular values.

Another difference between output projection algorithm and RPOD* algorithm is that RPOD* algorithm can be used when both the number of the inputs and outputs are large, while output projection can be used when the number of the inputs or the outputs is large. When the number of inputs is large, we can construct an input projection using the adjoint snapshot ensemble, but when both the number of inputs and outputs are large, the construction of the projection matrix is not helpful.

2) Comparison of computational cost: The comparison of the computational cost of the BPOD output projection algorithm with RPOD* algorithm is shown in Table I. We collect $m, n$ snapshots for RPOD* algorithm respectively, and $t_{ss}$ snapshots for BPOD output projection algorithm. $N$ is the dimension of the system, $p, q$ are the number of inputs and outputs respectively, the rank of the output projection is $s$, and without loss of generality, we assume $p \leq q$, $m \leq n$, and $p \leq s$.

Now we compare the computation time to generate the snapshot ensembles. If we denote $T$ as the time to propagate the primal/adjoint system once, then for BPOD output projection algorithm, $(p + s)$ simulations are needed, and for each simulation, we need to collect the snapshots up to $t = t_{ss}$. Thus, the total computation time to generate the snapshot ensembles is $(p+s)t_{ss}T$.

The RPOD* algorithm needs 1 primal and 1 adjoint simulation till time $m \Delta T, n \Delta T$ respectively, where $m \Delta T \geq t_{ss}, n \Delta T \geq t_{ss}$. When we choose $\Delta T$ such that $m \Delta T = t_{ss}$, $n \Delta T = t_{ss}$, the performance of the ROM constructed using RPOD* is better than those constructed using BPOD/BPOD output projection algorithm. While, the total computation time to generate the snapshot ensembles is $2t_{ss}T$, which means that RPOD* computational cost is the same as BPOD in a single input single output (SISO) system. The comparison of the computation time to generate the snapshot ensembles is shown in Table III for two examples.

VI. Computational Results

In this section, we show the comparison of RPOD* with BPOD for two examples: a one-dimensional heat transfer problem, and a three dimensional atmospheric dispersion problem to illustrate the proposed algorithm.

First, we define the output relative error:

\[ E_{output} = \frac{\| Y_{true} - Y_{rom} \|}{\| Y_{true} \|}. \]

where $Y_{true}$ are the outputs of the full order system, and $Y_{rom}$ are the outputs of the reduced order system.
The frequency response for multiple input multiple output systems can be represented by plotting the maximum singular value of the transfer function matrix \( \max(\sigma(H(j\omega))) \) as a function of frequency \( \omega \). We define the frequency responses error as:

\[
E_{fr}(j\omega) = \| \max(\sigma(H_{true}(j\omega))) - \max(\sigma(H_{rom}(j\omega))) \|, \quad (60)
\]

where \( H_{true}(j\omega) \) is the transfer function of the full order system, and \( H_{rom}(j\omega) \) is the transfer function of the ROM.

For the two experiments below, there are design parameters that need to be chosen manually, such as the rank of the output projection, when to take the snapshots, and the size of the ROM. We have shown the results of the best selections of these parameters for the BPOD/BPOD output projection algorithms and the RPOD* algorithm.

A. Heat Transfer

The equation for heat transfer by conduction along a slab is given by the partial differential equation:

\[
\frac{\partial T}{\partial t} = \alpha \frac{\partial^2 T}{\partial x^2} + f, \quad (61)
\]

with boundary conditions

\[
T|_{x=0} = 0, \quad \frac{\partial T}{\partial x}|_{x=L} = 0, \quad (62)
\]

where \( \alpha \) is the thermal diffusivity, and \( f \) is the forcing.

Two point sources are located at \( x = 0.15m \) and \( x = 0.45m \). The system is discretized using finite difference method, and there are 100 grids which are equally spaced. We take full field measurements, i.e., measurements at every node. The parameters of the system are summarized in Table III. In the following, we compare the ROM constructed using RPOD*, BPOD and BPOD output projection.

For RPOD*, the system is perturbed by the white noise with distribution \( N(0, I_{3x2}) \). At time \( t_{ss} = 3000s \), \( \|A_{new}\| \approx 0 \), thus, for the primal/adjoint simulation, one realization is needed, and we collect 80 equispaced snapshots during time \( t \in [0, 3200]s \). For BPOD, we need \( p = 2 \) realizations for the primal simulation, and \( q = 100 \) realizations for the adjoint simulations. In general, 3000 equispaced snapshots between \([0, 3000]s\) should be taken in each realization for BPOD/output projection. However, due to the memory limits on the platform, only 400 equispaced snapshots can be taken and for the optimal performance of BPOD/output projection, the 400 equispaced snapshots are taken from time \( t \in [0, 400]s \) (first 400 dominant impulse responses). The rank of the output projection is 40, and hence, for the BPOD output projection algorithm, 40 realizations are needed for the adjoint simulations. RPOD* algorithm and BPOD algorithm extract 70 modes, and BPOD output projection algorithm extract 50 modes. Here, we take 80 snapshots by trial and error following the procedure in Section IV-C.

In Fig. 4(a), we compare the norm of the Markov parameters of the ROM constructed using three algorithms with the true Markov parameters of the full order system. We perturb the system with random Gaussian noise, and compare the output relative errors in Fig. 4(b). In Fig. 5(a), we compare the frequency responses of the ROM constructed using three algorithms with the true frequency responses. In Fig. 5(b), we plot the error of the maximum singular value of the input-output model as a function of frequency.

For all three methods, we can see that the Markov parameters of the ROM are close to the true Markov parameters of the full order system. From Fig. 4(b), it can be seen that all three methods are accurate enough. The output relative error of BPOD output projection is less than 0.01%, and the performance of the RPOD* algorithm is much better than the BPOD output projection algorithm, the performance of RPOD* algorithm is better than BPOD algorithm because we do not take all the snapshots up to \( t_{ss} \) due to the memory limits. From Fig. 5 we can see that the frequency responses error of RPOD* algorithm is smaller than BPOD and BPOD output projection algorithm in low frequencies. With the increase of the frequency, the BPOD algorithm performs better than RPOD* algorithm, however, the errors are below \( 10^{-10} \), and hence, the difference is negligible. The comparison of computational time of RPOD* and BPOD output projection algorithm is shown in Section IV-C. We can see that the construction of the snapshot ensembles using RPOD* takes almost the same time as BPOD output projection, while the dominant computational cost is solving the SVD problem, and it can be seen that RPOD* is about 24 times faster than BPOD output projection.
TABLE II. PARAMETERS OF SYSTEM

|                | Heat                      | Atmospheric Dispersion                      |
|----------------|---------------------------|--------------------------------------------|
| **Domain**     | $x \in [0, 1](m)$         | $x \in [0, 2000](m), y \in [-100, 400](m), z \in [0, 50](m)$ |
| **Dimension of system** | $N = 100$              | $N = 10^5$                        |
| **Parameters** | $\alpha = 4.2 \times 10^{-4}(m^2/s)$ | Wind velocity $\vec{u} = (4m/s, 0, 0)$         |
| **Number of inputs** | 2                          | 10                                      |
| **Number of Outputs** | 100                        | 810                                     |
| **Primal Snapshots**   | 400 V.S. 80            | 200 V.S. 400                             |
| **Adjoint Snapshots**  | 400 V.S. 80             | 200 V.S. 400                             |
| **Snapshots taken during** | $t \in [0, 4000]s$ V.S. $t \in [0, 2000]s$ V.S. $t \in [0, 4000]s$ | 
| **ROM modes**        | 50 V.S. 70              | 200 V.S. 380                             |
| **Hankel matrix**    | $16000 \times 800$ V.S. $80 \times 80$ | $4000 \times 2000$ V.S. $400 \times 400$  |

B. Atmospheric Dispersion Problem

The three-dimensional advection-diffusion equation describing the contaminant transport in the atmosphere is:

$$ \frac{\partial c}{\partial t} + \nabla \cdot (\vec{u} c) = \nabla \cdot (K(\vec{X}) \nabla c) + Q \delta(\vec{X} - \vec{X}_s), \quad (63) $$

where

- $c(\vec{X}, t)$: mass concentration at location $\vec{X} = (x, y, z)$.
- $\vec{X}_s = (x_s, y_s, z_s)$: location of the point source.
- $\vec{u} = (ucos(\alpha), usin(\alpha), 0)$: wind velocity, $\alpha$ is the direction of the wind in the horizontal plane and the wind velocity is aligned with the positive $x$-axis when $\alpha = 0, u \geq 0$ is constant.
- $Q$: contaminant emitted rate.
- $\nabla$: gradient operator.
- $K(\vec{X}) = diag(K_x(x), K_y(x), K_z(x))$: diagonal matrix whose entries are the turbulent eddy diffusivities. In general $K(\vec{X})$ is a function of the downwind distance $x$ only.

In practice, the wind velocity is sufficiently large that the diffusion in the $x$-direction is much smaller than advection, and hence, assume that the term $K_x \frac{\partial^2 c}{\partial x^2}$ can be neglected.

Define $\sigma_x^2(x) = x \int_0^\infty K_y(\eta)d\eta$, $\sigma_y^2(x) = \frac{1}{2} \int_0^\infty K_z(\eta)d\eta$, where $\sigma_y(x) = ay(x+1) + by(x+1)^{0.5}$, $\sigma_z(x) = a_zx(1+b_zx)^{0.5}$, and $a_y = 0.008, b_y = 0.00001, a_z = 0.006, b_z = 0.00015$.

The boundary conditions are:

$$ c(0, y, z) = 0, c(\infty, y, z) = 0, c(x, \infty, z) = 0, $$
$$ c(x, y, \infty) = 0, K_z \frac{\partial c}{\partial z}(x, y, 0) = 0. \quad (64) $$

The system is discretized using finite difference method, and there are $100 \times 100 \times 10$ grids which are equally spaced. The parameters are summarized in Table II. There are 10 point sources which are shown in Fig. 5. We take the full field measurements (except the nodes on $x = 0, \infty$ and $y = \infty$).

In Fig. 6 we show the actual concentration of the field at time $t = 200s$ with $Q$ as Gaussian white noise where sources are the dotted points in the figure.

In this example, since the system dimension is $N = 10^5$, constructing the ROM with the full field measurements using BPOD is computationally impossible, and thus, we only compare the RPOD* algorithm with BPOD output projection algorithm.

For RPOD* algorithm, we collect the snapshots sequentially. The system is perturbed by white noise with distribution $N(0, I_{10\times10})$. One primal simulation and one adjoint simulation are needed. We collect 400 equispaced snapshots from time $t \in [0, 4000]s$, where at time $t_{ss} = 4000, \|A^t\| \approx 0$, and extract 380 modes. For BPOD output projection algorithm, we collect the impulse responses from the primal simulations, and $p = 10$ realizations are needed. Same as the heat example, we could not collect all the impulse responses from $t \in [0, 4000]s$ due to the memory limits on the platform. Hence, for the best performance available in this example, we collect 200 equispaced snapshots from $t \in [0, 200]s$ for each primal simulation. The rank of the output projection is 80, and hence, 80 adjoint simulations are needed, and in the adjoint simulations, we collect 50 equispaced snapshots from $t \in [0, 50]s$. We can extract 200 modes. The parameters are summarized in Table II.

In Fig. 7(a), we compare the Markov parameters of the ROM constructed using RPOD* and BPOD output projection with the full order system. Also, we perturb the system with random Gaussian noise, and compare the output relative errors in Fig. 7(b), The comparison of the computational time is shown in Section VI-C.

In Fig. 8(a), we compare the frequency responses of the ROM constructed using RPOD* and BPOD output projection with the full order system. We can see that the frequency responses of the ROMs are almost the same as the frequency responses of the full order system. In Fig. 8(b), we show the errors between the frequency responses. The comparison of the computational time is shown in Sec-
computational results show that the accuracy of the RPOD is much more accurate than the BPOD output projection algorithm.

APPENDIX A

PROOF OF PROPOSITION

Proof: The adjoint snapshot ensemble $Z_r$ can be written as:

$$Z_r = U_{co} \beta_{co} + U_{co} \delta \beta_{co} + U_{co} \beta_{co} + U_{co} \delta \beta_{co},$$

(65)

where $\delta \beta_{co} = \epsilon \beta_{co} - \epsilon \beta_{co}$, $\epsilon$ is defined in assumption A4, $\beta_{co}, \beta_{co}$ are coefficient matrices. From (49) and (65),

$$Z'_r X_r = \beta_{co}' \alpha_{co} + \delta \beta_{co}' \alpha_{co} + \beta_{co}' \delta \alpha_{co} + \delta \beta_{co}' \delta \alpha_{co} + O(\epsilon^2),$$

$$= \beta_{co}' \alpha_{co} + \epsilon \left( \beta_{co}' \alpha_{co} + \beta_{co}' \alpha_{co} \right) + O(\epsilon^2),$$

(66)

And similarly,

$$Z'_r A X_r = \beta_{co}' \Lambda_{co} \alpha_{co} + \epsilon \left( \beta_{co}' \Lambda_{co} \alpha_{co} + \beta_{co}' \Lambda_{co} \alpha_{co} \right) + O(\epsilon^2)$$

$$= \beta_{co}' \Lambda_{co} \alpha_{co} + \epsilon E_2 + O(\epsilon^2).$$

(67)

There are "l" controllable and observable modes, then rank $(Z'_r X_r) \geq l$ due to the small perturbations. Denote

$$H_r = \beta_{co}' \alpha_{co} = L_r \Sigma_r R'_r + L_r \Sigma_r R'_r,$$

$$H_r = Z'_r X_r = \beta_{co}' \alpha_{co} + \epsilon E_2 = L_r \Sigma_r R'_r + L_r \Sigma_r R'_r,$$

(68)

where $H_r, H_r \in \mathbb{R}^{n \times m}$, and WLOG, $n \leq m$.

Here, $H_r$ is the ideal Hankel matrix constructed with the simplifying assumption (assumption A3 is satisfied), and it can be seen that the true Hankel matrix $H_r$ (assumption A4 is satisfied) can be viewed as adding a small perturbation of $H_r$, i.e., $H_r = H_r + \epsilon E_1$.

$$\Sigma_r, \Sigma \in \mathbb{R}^{l \times 1}$$

contains the l non-zero singular values of $H_r$, and $(L_r, R_r)$ are the corresponding left and right singular vectors. $\Sigma_r \in \mathbb{R}^{(n-l) \times (n-l)}$ are the rest singular values, and $(L_r, R_r)$ are the corresponding left and right singular vectors. Similarly, $\Sigma \in \mathbb{R}^{l \times 1}$ contains the first l non-zeros singular values of $H_r$, and $\Sigma \in \mathbb{R}^{(n-l) \times (n-l)}$ contains the rest singular values. The left and right singular vectors are partitioned in the same way.

1. For the perturbation theory, the perturbed singular values and singular vectors $(\Sigma_r, L_r, R_r)$ are related to the singular values and singular vectors $(\Sigma, L, R_r)$ as:

$$\Sigma_r = \Sigma + \epsilon E_3 + O(\epsilon^2),$$

$$L_r = \bar{L}_r + \Delta L_r, R_r = \bar{R}_r + \Delta R_r,$$

(69)

where $E_3, \Delta L_r, \Delta R_r$ are some matrices, and $\|\Delta L_r\|, \|\Delta R_r\| \propto O(\epsilon)$. The expression of $E_3$ is given by the follows

$$\bar{\sigma}_i \in \Sigma_r, (\text{strictly positive singular values}) \text{ with multiplicity } t, \bar{L}_r, \bar{R}_r \text{ are the corresponding left and right singular
vectors, the perturbed singular values \( \sigma_i \in \Sigma_r \), and
\[
\sigma_i = \bar{\sigma}_i + \epsilon \frac{\lambda_i(\bar{R}_r^tE_i\tilde{L}_r + \tilde{L}_r^tE_i)}{2e_i} + O(\epsilon^2), \quad i = 1, \cdots, t \tag{70}
\]
where \( \lambda_i() \) denotes the \( i^{th} \) eigenvalue of (\). Thus,
\[
E_3 = \begin{pmatrix}
e_1 \\
\vdots \\
e_t
\end{pmatrix},
\tag{71}
\]
where \( e_1, \cdots, e_t \) are the coefficients computed from (70).

2. The ROM \( \tilde{A} = S_rAT_r \) can be proved to be a perturbation of \( A_b \) in (23).

The POD bases \( T_r, S_r \) are:
\[
T_r = X_rR_r\Sigma_r^{-1/2}, \quad S_r = \Sigma_r^{-1/2}L'_rZ'_r.
\tag{72}
\]
Therefore,
\[
\tilde{A} = S_rAT_r = \Sigma_r^{-1/2}L'_rS'_rAX_rR_r\Sigma_r^{-1/2} = \Sigma_r^{-1/2}L'_r(\beta'o\Lambda_\sigma o\epsilon + \epsilon E_2)R_r\Sigma_r^{-1/2}.
\tag{73}
\]
From (70), it can be proved that:
\[
\Sigma_r^{-1/2} = \bar{\Sigma}_r^{-1/2} + \epsilon C_r,
\tag{74}
\]
where \( C_r \) is a diagonal coefficient matrix. Therefore,
\[
\Sigma_r^{-1/2}L'_r = (\Sigma_r^{-1/2} + \epsilon C_r)(L'_r + \Delta L'_r) = \Sigma_r^{-1/2}L'_r + \Delta_1,
\tag{75}
\]
where \( \Delta_1 \) is some matrix, and \( \|\Delta_1\|_2 = k_1\epsilon \), \( k_1 \) is a constant. Similarly,
\[
R_r\Sigma_r^{-1/2} = \bar{R}_r\Sigma_r^{-1/2} + \Delta_2,
\tag{76}
\]
where \( \Delta_2 \) is some matrix, and \( \|\Delta_2\|_2 = k_2\epsilon \), \( k_2 \) is a constant. Thus,
\[
\tilde{A} = \Sigma_r^{-1/2}L'_r\beta'o\Lambda_\sigma o\epsilon \bar{R}_r\Sigma_r^{-1/2} + \Delta_3 + O(\epsilon^2),
\tag{77}
\]
where \( \Delta_3 \) is some matrix, and \( \|\Delta_3\|_2 = k_3\epsilon \), \( k_3 \) is a constant.

If we let
\[
\tilde{A} = \Sigma_r^{-1/2}L'_r\beta'o\Lambda_\sigma o\epsilon \bar{R}_r\Sigma_r^{-1/2},
\tag{78}
\]
then
\[
\tilde{A} = \tilde{A} + \Delta_3 + O(\epsilon^2) = P_r\Lambda_rP_r^{-1},
\tag{79}
\]
be proved that \( \Lambda_\sigma o \) are the eigenvalues of \( \tilde{A} \), and \( P_r \) are the corresponding eigenvectors.

3. From perturbation theory \([22]\), since
\[
\tilde{A} = \tilde{A} + \Delta_3 + O(\epsilon^2) = P_r\Lambda_rP_r^{-1},
\tag{80}
\]
it can be proved that \( \|P_r - \tilde{P}_r\| \leq \|\Delta_3\| = k_3\epsilon \), \( \|\Lambda_r - \Lambda_\sigma o\| \leq k_3\epsilon \). Thus,
\[
\Psi_r = T_rP_r = X_r\bar{R}_r\Sigma_r^{-1/2}\tilde{P}_r + \Delta_4,
\tag{81}
\]
\[
\Phi_r = P_r^{-1}S_r = \tilde{P}_r^{-1}\Sigma_r^{-1/2}L'_rZ'_r + \Delta_5,
\tag{82}
\]
where \( \Delta_4, \Delta_5 \) are some matrices and \( \|\Delta_4\|, \|\Delta_5\| \propto O(\epsilon) \).

Substitute (81) into the ROM Markov parameters, and collect the small perturbation terms, we have:
\[
C_rA'_rB_r = C\Psi_r\Lambda'_r\Phi_rB = CV_\sigma o\Lambda'_rU'_rB' + \Delta,
\tag{83}
\]
where \( \Delta \) is some matrix, and \( \|\Delta\| \propto O(\epsilon) \).

### APPENDIX B

#### Proof of Corollary

For \( \sigma_i \in \bar{\Sigma}_o \) (zero singular values) \([18]\) with multiplicity \( n - l \), the corresponding left and right singular vectors are \( \bar{L}_o, \bar{R}_o \). The perturbed singular values \( \sigma_i \in \Sigma_o \) and
\[
\sigma_i = \epsilon \sqrt{\lambda_i(\bar{R}_o^tE'_o\bar{L}_o + \bar{L}_o^tE'_o)} + \epsilon E_2R_o\Sigma_o^{-1/2}, \quad i = 1, \cdots, n - l.
\tag{83}
\]
From (70) and (83), we can see that:
\[
\sigma_i = \sigma_i + \epsilon_i\epsilon + O(\epsilon^2), \quad \sigma_i + 1 = \epsilon_{i+1}\epsilon,
\tag{84}
\]
Hence, we have:
\[
\sigma_{i+1} \propto O(\epsilon),
\tag{85}
\]
and
\[
\|C_rA'_rB_r - CA'B\| \propto O(\epsilon) \propto O(\sigma_{i+1}).
\tag{86}
\]

### REFERENCES

[1] J. M. Stockie, “The mathematics of atmospheric dispersion modeling,” *SIAM Review*, vol. 53, No.2, pp. 349–372, 2011.

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

[3] C. W. Rowley, “Model reduction for fluids using balanced proper orthogonal decomposition,” *International Journal of Bifurcation and Chaos*, vol. 15, pp. 997–1013, 2005.
[4] B. C. Moore, “Principal component analysis in linear systems: Controllability, observability and model reduction,” *IEEE Transactions on Automatic Control*, vol. 26, No.1, pp. 17–32, 1981.

[5] G. Berkooz, P. Holmes, and J. L. Lumley, “The proper orthogonal decomposition in the analysis of turbulent flows,” *Annual review of fluid mechanics*, vol. 25, pp. 539–575, 1993.

[6] S. Kung, “A new identification method and model reduction algorithm via singular value decomposition,” *12th Asilomar Conference on Circuits, Systems and Computers*, pp. 705–714, Nov. 1978.

[7] J.-N. Juang and R. S. Pappa, “An eigensystem realization algorithm for model parameter identification and model reduction,” *Journal of Guidance, Control, and Dynamics*, vol. 8, No.5, pp. 620–627, 1985.

[8] Z. Ma, S. Ahuja, and C. W. Rowley, “Reduced order models for control of fluids using the eigensystem realization algorithm,” *Theoret. Comput. Fluid Dyn.*, vol. 25, pp. 233–247, 2011.

[9] P. J. Schmid, “Dynamic mode decomposition of numerical and experimental data,” *Journal of Fluid Mechanics*, vol. 656, pp. 5–28, 2010.

[10] J. H. Tu, C. W. Rowley, D. M. Luchtenburg, S. L. Brunton, and J. N. Kutz, “On dynamic mode decomposition: Theory and applications,” *Journal of Computational Dynamics*, vol. 1, Issue 2, pp. 391–421, 2014.

[11] D. B. Pourkargar and A. Armaou, “Geometric output tracking of nonlinear distributed parameter systems via adaptive model reduction,” *Chemical Engineering Science*, vol. 116, pp. 418–427, 2014.

[12] ——, “Design of apod-based switching dynamic observers and output feedback control for a class of nonlinear distributed parameter systems,” *Chemical Engineering Science*, vol. 136, pp. 62–75, 2015.

[13] J. H. Tu and C. W. Rowley, “An improved algorithms for balanced pod through an analytic treatment of impulse response tails,” *Journal of Computational Physics*, vol. 231, pp. 5317–5333, June, 2012.

[14] P. Drineas et al., “Fast monte carlo algorithms for matrices ii: Computing a low rank approximation to a matrix,” *SIAM J. Computing*, vol. 36, pp. 158–183, 2006.

[15] M. W. Mahnoey, “Randomized algorithms for matrices and data,” arXiv: 1104.5557, Tech. Rep., 2011.

[16] N. Halko, P. Martinsson, and J. Tropp, “Finding structure with randomness: probabilistic algorithms for constructing approximate matrix decompositions,” *SIAM review*, vol. 52(2), pp. 217–288, 2011.

[17] D. Yu and S. Chakravorty, “A randomized proper orthogonal decomposition technique,” in *Proceedings of American Control Conference*, 2015, p. in press.

[18] T. Soderstrom, “Perturbation results for singular values,” Department of Information Technology at Uppsala University, Tech. Rep., 1999.

[19] B. Friedlander and B. Porat, “First-order perturbation analysis of singular vectors in singular value decomposition,” *IEEE Transactions on Signal Processing*, vol. 56, Issue 7, pp. 3044–3049, 2008.

[20] D. Yu and S. Chakravorty, “A computationally optimal randomized proper orthogonal decomposition technique,” arXiv: 1509.05705, Tech. Rep., 2015.

[21] M. Ilak and C. W. Rowley, “Modeling of transitional channel flow using balanced proper orthogonal decomposition,” *Physics of Fluids*, vol. 20, 2008.

[22] T. Kato, *Perturbation Theory for Linear Operators*. New York: Springer-Verlag, 1995.