algorithms

MDPI

Article

A Missing Data Reconstruction Method Using an Accelerated Least-Squares Approximation with Randomized SVD

Siriwan Intawichai and Saifon Chaturantabut *

Department of Mathematics and Statistics, Faculty of Science and Technology, Thammasat University, Pathum Thani 12120, Thailand; siriwan.int@dome.tu.ac.th
* Correspondence: saifon@mathstat.sci.tu.ac.th

Abstract: An accelerated least-squares approach is introduced in this work by incorporating a greedy point selection method with randomized singular value decomposition (rSVD) to reduce the computational complexity of missing data reconstruction. The rSVD is used to speed up the computation of a low-dimensional basis that is required for the least-squares projection by employing randomness to generate a small matrix instead of a large matrix from high-dimensional data. A greedy point selection algorithm, based on the discrete empirical interpolation method, is then used to speed up the reconstruction process in the least-squares approximation. The accuracy and computational time reduction of the proposed method are demonstrated through three numerical experiments. The first two experiments consider standard testing images with missing pixels uniformly distributed on them, and the last numerical experiment considers a sequence of many incomplete two-dimensional miscible flow images. The proposed method is shown to accelerate the reconstruction process while maintaining roughly the same order of accuracy when compared to the standard least-squares approach.

Keywords: singular value decomposition; missing data approximation; low-rank approximation; least-squares method; randomized algorithm; discrete empirical interpolation method

Citation: Intawichai, S.; Chaturantabut, S. A Missing Data Reconstruction Method Using an Accelerated Least-Squares Approximation with Randomized SVD. *Algorithms* **2022**, *15*, 190. https://doi.org/10.3390/a15060190

Academic Editor: Ulrich Pferschy

Received: 1 May 2022
Accepted: 26 May 2022
Published: 31 May 2022

1. Introduction

Missing data is a problematic issue in many applications. This problem may arise from incomplete data collection, unavailability of data, or corrupted data during the transmission process. Recovering missing data accurately is essential, especially for the purposes of data analysis and prediction. Various approaches have been proposed for reconstructing missing data. In [1], the deep convolution neural network was proposed to reconstruct missing data for a remote sensing image. A deep learning approach was used for irregularly and regularly missing data reconstruction in [2]. The concept of fuzzy similarity was used for recovering data in multidimensional time series in [3]. The long short-term memory(LSTM) approach was introduced in [4] for reconstructing time series Landsat images. A methodology based on principles of matrix theory and pseudo-inverses was proposed in [5] for data reconstruction in erasure codes. In [6], a method of tensor completion based on nuclear norm minimization was introduced for 5D seismic data reconstruction. Other approaches for approximating missing data include the regression method [7], K-nearest neighbors (KNN) [8,9], principal component analysis (PCA) [10], and least-squares approximation [11–13]. These approaches are mostly specific to certain problems. The goal of this paper is to improve a general framework based on a least-squares method for estimating missing data with an optimal low-rank basis in tje Euclidean norm, which is also called proper orthogonal decomposition (POD) basis.

POD is also known as, for example, Karhunen-Loève decomposition (KLD), principal component analysis (PCA), or singular value decomposition(SVD). POD has been successfully used in many applications, e.g., [14–18], since it can provide an approximation from the basis that extracts the dominant characteristic of the existing data. POD has been used in many applications, such as aerodynamic flow fields [19], chemical engineering [20,21],

mechanical engineering [22,23], image processing [24], and optimization of water flooding reservoir [25]. In general, the POD basis can be computed by using singular value decomposition (SVD). However, for high-dimensional data, obtaining SVD by a traditional approach may be computationally costly and memory intensive. Randomized SVD (rSVD) [26] is considered in this work to reduce this computational complexity.

The rSVD algorithm, introduced by Halko et al. [26], employs randomness to obtain a smaller matrix from a high-dimensional data matrix. The smaller matrix is then used to compute the low-dimensional basis. The rSVD algorithm is shown to be computationally efficient for approximating matrices with low-rank structures. The algorithm is divided into two stages. In the first stage, random sampling is used to obtain a reduced matrix whose range approximates the range of the original matrix. A small matrix can be obtained from the orthogonal projection onto the basis of the reduced matrix from the first stage. In the second stage, the smaller matrix is then inexpensively factorized. Randomized methods have been used in many applications, such as low-rank SVD of a large matrix [27], image compression [28,29], and image reconstruction [30]. The implementation and extension of the rSVD have been investigated in many works, e.g., [31–34]. This work applies rSVD to reconstruct missing data components with the notion of least-squares approximation.

The least-squares method has been used in missing data reconstruction in previous work, e.g., [11–13], which is mainly based on employing all remaining available data. In the case of high-dimensional data, the reconstruction process might require high-complexity computation. To accelerate this approximation process further, this work also employs certain important and relevant available data components, which are selected through a greedy iterative procedure called the discrete empirical interpolation method (DEIM). DEIM [35] was originally introduced to estimate nonlinear terms in dynamical systems by selecting the interpolation indices using a greedy algorithm. The index selection process in this algorithm is based on trying to capture most of the variation of the sample set heuristically. DEIM is often used with POD to provide interpolatory projection approximation. Theoretical works on this approximation and its extension can be found in [35–37]. DEIM has been used in many applications, such as two-dimensional shallow-water equations [38], four-dimensional variational data assimilation [39], three-dimensional nonlinear aeroelasticity model [40], and other problems [41,42].

This work is an extension of the result given in [43], which only considered one simple preliminary numerical test. This work provides substantially more materials with an additional improved method that incorporates a faster way to compute projection basis based on a randomized algorithm, as well as performs more numerical experiments with detailed analysis and comparisons to the standard approach. The main contributions of this work can be summarized as follows.

- We propose an algorithm that accelerates the missing data reconstruction procedure based on the least-squares method, which can reduce computational time in two aspects. The first one is based on applying the rSVD to obtain an optimal low-dimensional basis used in the least-squares projection. The second aspect arises from applying a greedy point selection procedure that can extract a small subset of important components to reduce the size of the reconstruction problem and, therefore, speed up the computational time.
- We provide numerical experiments that demonstrate the efficiency (in terms of accuracy and speedup) of the proposed method for some standard testing images with different amounts of missing data.
- We perform numerical tests on a sequence of two-dimensional miscible flow images. The proposed method was shown to give 10 to 100 times reduction in reconstruction time with the same order of accuracy. This illustrates the extensibility of the proposed method on other large-scale problems, as well as the applications of video processing.

This paper is organized as follow. Section 2 describes the approach for reconstructing missing components using the least-squares method. The connection between SVD and the projection basis, which is called POD basis, used in the approximation is discussed

in Section 3. The rSVD algorithm is explained in Section 4. Section 5 presents a greedy point selection process using DEIM as well as shows how it is used to reduce the computational complexity of missing data reconstruction. Section 6 discusses more details on the complexity reduction of the proposed method in terms of floating-point operations when compared to the standard least-squares approach. The numerical tests demonstrating the effectiveness of the proposed method are performed on three experiments in Section 7. Finally, the conclusion and some possible extensions are discussed in Section 8.

2. Standard Projection-Based Approach for Data Reconstruction

This section describes how SVD is used to reconstruct missing components of incomplete data. We will approximate an incomplete data vector by projecting on a subspace spanned by a basis that represents other related complete data vectors.

Let $\{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_{n_s}\}$ be a complete data set. Define $\mathcal{Y} := \{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_{n_s}\} \subset \mathbb{R}^n$ and $\mathbf{Y} = [\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_{n_s}] \in \mathbb{R}^{n \times n_s}$. Suppose $\widehat{\mathbf{y}}$ is an incomplete sample that has n_c known components and $n_g = n - n_c$ unknown components. Let $\mathcal{C} := \{c_1, c_2, \ldots, c_{n_c}\} \subset \{1, 2, \ldots, n\}$ be the indices of the *known* components in $\widehat{\mathbf{y}}$ and define $\mathbf{C} = [\mathbf{e}_{c_1}, \ldots, \mathbf{e}_{c_{n_c}}] \in \mathbb{R}^{n \times n_c}$, where $\mathbf{e}_{c_i} = [0, \ldots, 0, 1, 0, \ldots, 0]^T \in \mathbb{R}^n$ is the c_i-th column of the identity matrix $\mathbf{I}_n \in \mathbb{R}^{n \times n}$, for $i = 1, \ldots, m$. Note that pre-multiplying $\mathbf{C}^T$ is equivalent to extracting the n_c rows corresponding to the indices $c_1, \ldots, c_{n_c}$. Similarly, let $\mathcal{G} = \{g_1, g_2, \ldots, g_{n_g}\} \subset \{1, 2, \ldots, n\}$ be the indices of the *unknown* components in $\widehat{\mathbf{y}}$ and define $\mathbf{G} = [\mathbf{e}_{g_1}, \ldots, \mathbf{e}_{g_{n_g}}] \in \mathbb{R}^{n \times n_g}$. Let $\widehat{\mathbf{y}}_c := \mathbf{C}^T \widehat{\mathbf{y}} \in \mathbb{R}^{n_c}$ and $\widehat{\mathbf{y}}_g := \mathbf{G}^T \widehat{\mathbf{y}} \in \mathbb{R}^{n_g}$. Then, the known components and the unknown components are contained in the vectors $\widehat{\mathbf{y}}_c$ and $\widehat{\mathbf{y}}_g$, respectively. The missing components contained in $\widehat{\mathbf{y}}_g$ will be approximated by first projecting $\widehat{\mathbf{y}}$ onto the column span of a basis matrix $\mathbf{V}$ with rank k, where $k \leq r, r := \text{rank}(\mathbf{Y})$. i.e.,

$$\widehat{\mathbf{y}} \approx \mathbf{V}\mathbf{a}, \qquad \text{or} \qquad \widehat{\mathbf{y}}_c \approx \mathbf{V}_c\mathbf{a} \text{ and } \widehat{\mathbf{y}}_g \approx \mathbf{V}_g\mathbf{a},$$

for some coefficient vector $\mathbf{a} \in \mathbb{R}^k$, and where $\mathbf{V}_c := \mathbf{C}^T\mathbf{V} \in \mathbb{R}^{n_c \times k}$, $\mathbf{V}_g := \mathbf{G}^T\mathbf{V} \in \mathbb{R}^{n_g \times k}$.

The known components contained in $\widehat{\mathbf{y}}_c$ are then used to determine the coefficient vector $\mathbf{a}$ through the approximation $\widehat{\mathbf{y}}_c \approx \mathbf{V}_c\mathbf{a}$ from the following least-squares problem:

$$\min_{\mathbf{a} \in \mathbb{R}^k} \|\widehat{\mathbf{y}}_c - \mathbf{V}_c\mathbf{a}\|_2^2. \tag{1}$$

The solution obtained from the corresponding normal equation of the above problem is given by $\mathbf{a} = \mathbf{V}_c^+ \widehat{\mathbf{y}}_c$, where $\mathbf{V}_c^+ = (\mathbf{V}_c^T\mathbf{V}_c)^{-1}\mathbf{V}_c^T$ is the Moore–Penrose pseudo-inverse. That is,

$$\widehat{\mathbf{y}}_g \approx \mathbf{V}_g\mathbf{a} = \mathbf{V}_g\mathbf{V}_c^+\widehat{\mathbf{y}}_c. \tag{2}$$

The steps described above, which is called the POD-LS approach or standard LS approach, are summarized in Algorithm 1.

Note that the dominant computational work in Algorithm 1 generally comes from Step 2 for constructing a set of basis that can accurately approximate the incomplete data and Step 3 for solving the least-squares problem. In this work, Step 2 employs the basis obtained from the singular value decomposition (SVD) because it is optimal in the least-squares sense, as discussed next in Section 3. To reduce the computational cost for Step 2, we will apply a randomized SVD (rSVD), as discussed in Section 4. To reduce the computational cost in Step 3, we will consider a greedy procedure that can decrease the size of the least-squares problem efficiently in Section 5.1.

Algorithm 1 Standard POD-LS approach for approximating missing data

INPUT:

- Complete snapshot set $\{\mathbf{y}_j\}_{j=1}^{n_s} \subset \mathbb{R}^n$ and, dimension $k \leq rank(\{\mathbf{y}_j\}_{j=1}^{n_s})$
- Incomplete data $\widehat{\mathbf{y}} \in \mathbb{R}^n$ with known entries in $\widehat{\mathbf{y}}_c \in \mathbb{R}^{n_c}$ and unknown entries in $\widehat{\mathbf{y}}_g \in \mathbb{R}^{n_g}$, where $n_c + n_g = n$.

OUTPUT:

- Approximation of $\widehat{\mathbf{y}}_g$

1: Create snapshot matrix : $\mathbf{Y} = [\mathbf{y}_1, \ldots, \mathbf{y}_{n_s}] \in \mathbb{R}^{n \times n_s}$ and let $r = rank(\mathbf{Y})$.
2: Construct basis $\mathbf{V}$ of rank $k \leq r$ for $\mathbf{Y}$.
3: Find coefficient vector $\mathbf{a}$ from $\widehat{\mathbf{y}}_c$ using least-squares problem in (1): $\min_{\mathbf{a} \in \mathbb{R}^k} \|\widehat{\mathbf{y}}_c - \mathbf{V}_c \mathbf{a}\|_2^2$.
4: Compute the approximation $\widehat{\mathbf{y}}_g \approx \mathbf{V}_g \mathbf{a}$.

3. Optimal Basis for Data Set

Consider a set of snapshots $\{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_{n_s}\}$ where $\mathbf{y}_j \in \mathbb{R}^n$, $j = 1, 2, \ldots, n_s$. Suppose we want to approximate a snapshot $\mathbf{y}_j$ by using a set of orthonormal vectors $\{\phi_1, \phi_2, \ldots, \phi_k\} \subset \mathbb{R}^n$, where $k < n$. Then, the approximation is in the form $\mathbf{y}_j \approx \sum_{i=1}^{k} w_i \phi_i$, for some constant w_i, $i = 1, 2, \ldots, k$. Alternatively, we can write this approximation in matrix form as $\mathbf{y}_j \approx \mathbf{\Phi}_k \mathbf{w}$, where $\mathbf{\Phi}_k = [\phi_1, \phi_2, \ldots, \phi_k] \in \mathbb{R}^{n \times k}$ is a matrix of basis vectors and $\mathbf{w} = [w_1, w_2, \ldots, w_k]^T \in \mathbb{R}^k$ is the vector of unknown coefficients. To find $\mathbf{w}$, we use the fact that $\mathbf{\Phi}$ has orthonormal columns, i.e., $\mathbf{\Phi}^T \mathbf{\Phi} = \mathbf{I}$, and the minimum error occurs when the residual is orthogonal to the column span of $\mathbf{\Phi}_k$, i.e., $\mathbf{\Phi}_k^T (\mathbf{y}_j - \mathbf{\Phi}_k \mathbf{w}) = 0$, which implies that $\mathbf{w} = \mathbf{\Phi}^T \mathbf{y}_j$ and the approximation becomes

$$\mathbf{y}_j \approx \mathbf{\Phi}_k \mathbf{\Phi}_k^T \mathbf{y}_j. \tag{3}$$

The goal is to find the orthonormal basis that minimizes this approximation error in 2-norm for a given basis rank $k \leq \text{rank}(\{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_{n_s}\})$

$$\min_{\mathbf{\Phi}_k \in \mathbb{R}^{n \times k}} \sum_{j=1}^{n_s} \|\mathbf{y}_j - \mathbf{\Phi}_k \mathbf{\Phi}_k^T \mathbf{y}_j\|_2^2 \quad \text{such that} \quad \mathbf{\Phi}_k^T \mathbf{\Phi}_k = \mathbf{I}_k, \tag{4}$$

where $\mathbf{I}_k \in \mathbb{R}^{k \times k}$ is the identity matrix. This optimal basis is often called proper orthogonal decomposition (POD), which is also known by other names, for example, Karhunen-Loève decomposition (KLD), principal component analysis (PCA), or singular value decomposition (SVD).

It can be shown [44,45] that the POD basis defined above can be obtained from the left singular vector of the snapshot matrix $\mathbf{Y} = [\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_{n_s}]$. Let $\mathbf{Y} = \mathbf{U} \mathbf{\Sigma} \mathbf{V}^T$ be the singular value decomposition of $\mathbf{Y}$, where matrices $\mathbf{U} = [\mathbf{u}_1, \ldots, \mathbf{u}_r] \in \mathbb{R}^{n \times r}$ and $\mathbf{V} = [\mathbf{v}_1, \ldots, \mathbf{v}_r] \in \mathbb{R}^{n_s \times r}$ are matrices with orthonormal columns and $\mathbf{\Sigma} = diag(\sigma_1, \ldots, \sigma_r) \in \mathbb{R}^{r \times r}$ is a diagonal matrix with $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r > 0$. Then the POD basis matrix of dimension k is $\mathbf{U}_k = [\mathbf{u}_1, \ldots, \mathbf{u}_k] \in \mathbb{R}^{n \times k}$, $k \leq r$, i.e. $\mathbf{U}_k = \arg\min_{\mathbf{\Phi}_k \in \mathbb{R}^{n \times k}} \sum_{j=1}^{n_s} \|\mathbf{y}_j - \mathbf{\Phi}_k \mathbf{\Phi}_k^T \mathbf{y}_j\|_2^2$ with $\mathbf{U}_k^T \mathbf{U}_k = \mathbf{I}_k$. It is well known [44,45] that this minimum error is given by

$$\sum_{j=1}^{n_s} \|\mathbf{y}_j - \mathbf{U}_k \mathbf{U}_k^T \mathbf{y}_j\|_2^2 = \sum_{\ell=k+1}^{r} \sigma_\ell^2, \tag{5}$$

which is the sum of the neglected singular values $\sigma_{k+1}, \ldots, \sigma_r$ from the SVD of $\mathbf{Y}$.

When the dimension n of snapshots is not too large, we can directly obtain the POD basis from the SVD of the snapshot matrix. However, in practice, n can be extremely large and computing the POD basis through a standard SVD might not be efficient. In this work, we use the randomized SVD to handle this problem.

4. Randomized SVD (rSVD)

The randomized SVD (rSVD) was proposed in [26] to reduce the computational complexity of the standard approach for computing SVD. It consists of two stages. In the first stage, random sampling is used to obtain a reduced matrix whose range approximates the range of $\mathbf{Y}$; in the second stage, this reduced matrix is then factorized inexpensively.

In particular, suppose we want to construct a low-dimensional basis with rank k by using rSVD. A random matrix $\Omega \in \mathbb{R}^{n_s \times k}$ is first employed to generate the weights used in the linear combination of the data matrix $\mathbf{Y}$ to obtain a matrix with k columns, where $k < n_s$. Let

$$\mathbf{Z} = \mathbf{Y}\Omega \in \mathbb{R}^{n \times k}.$$

Then the matrix $\mathbf{Z}$ is used to find an orthonormal basis $\mathbf{Q} \in \mathbb{R}^{n \times k}$, which can be done through many algorithms, such as Gram–Schmidt process, Householder transformation, or the QR decomposition. At this point, we can use the basis matrix $\mathbf{Q}$ to approximate the column span of $\mathbf{Z}$, as well as the column span of the data matrix $\mathbf{Y}$, i.e.,

$$\mathbf{Y} \approx \mathbf{Q}\mathbf{B},$$

where $\mathbf{B} \in \mathbb{R}^{k \times n_s}$ can be obtained from the orthogonal projection of $\mathbf{Y}$ onto the low dimensional subspace spanned by the columns of $\mathbf{Q}$, which gives $\mathbf{B} = \mathbf{Q}^T\mathbf{Y}$. The second stage of rSVD then computes the SVD of this small matrix $\mathbf{B}$. Suppose $\mathbf{B} = \widehat{\mathbf{U}}\widehat{\Sigma}\widehat{\mathbf{V}}^T$ is the SVD of $\mathbf{B}$, where matrices $\widehat{\mathbf{U}} = [\widehat{\mathbf{u}}_1, \ldots, \widehat{\mathbf{u}}_{\hat{r}}] \in \mathbb{R}^{k \times \hat{r}}$ and $\widehat{\mathbf{V}} = [\widehat{\mathbf{v}}_1, \ldots, \widehat{\mathbf{v}}_{\hat{r}}] \in \mathbb{R}^{n_s \times \hat{r}}$ are matrices with orthonormal columns, $\widehat{\Sigma} = diag(\widehat{\sigma}_1, \ldots, \widehat{\sigma}_{\hat{r}}) \in \mathbb{R}^{\hat{r} \times \hat{r}}$ is a diagonal matrix with $\widehat{\sigma}_1 \geq \widehat{\sigma}_2 \geq \cdots \geq \widehat{\sigma}_{\hat{r}} > 0$ and $\hat{r} = \text{rank}(\mathbf{B})$. Then, we finally obtain the approximated POD basis or the left singular vectors of $\mathbf{Y}$ from the product $\mathbf{Q}\widehat{\mathbf{U}}$.

In practice, the matrix $\mathbf{Z}$ constructed from an n-by-k random matrix may not be a good representation of the original data matrix $\mathbf{Y}$. Consequently, the basis matrix $\mathbf{Q}$ may not give an accurate approximation for the column space of $\mathbf{Y}$. To solve this issue, we consider an improved algorithm for rSVD by using an oversampling strategy given in [26], which uses an n-by-$(k+p)$ random matrix instead of n-by-k, where p is a positive integer of additional random vectors. Algorithm 2 summarizes these two stages of the rSVD.

Algorithm 2 Randomized Singular Value Decomposition (rSVD)

INPUT: $\{\mathbf{y}_\ell\}_{\ell=1}^{n_s} \subset \mathbb{R}^n$, k where $k \leq rank(\{\mathbf{y}_\ell\}_{\ell=1}^{n_s})$
OUTPUT: POD basis: $\mathbf{U}_k$, Singular values: $\mathbf{S}_k$

1: Form $\mathbf{Y} = [\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_{n_s}]$
2: Construct random matrix $\Omega = rand(M, K_0)$, $K_0 = k + p$, where p is a positive integer
3: $\mathbf{Q} = orthog(\mathbf{Y}\Omega)$
4: Set $\mathbf{B} = \mathbf{Q}^T\mathbf{Y} \in \mathbb{R}^{K_0 \times n_s}$
5: Compute SVD of $\mathbf{B}$: $\mathbf{B} = \widehat{\mathbf{U}}\widehat{\Sigma}\widehat{\mathbf{V}}^T$
6: Basis: $\mathbf{U} = \mathbf{Q}\widehat{\mathbf{U}}$
7: Truncation: $\mathbf{U}_k = \mathbf{U}(:, 1:k)$, $\mathbf{S}_k = \Sigma(:, 1:k)$

5. Accelerated POD-LS Method (Gappy POD)

This section aims to discuss an efficient approach that can reduce the computational complexity for solving the standard LS approach (LS-POD approach) used in the data reconstruction technique described in Section 2. A greedy-based approach called the discrete empirical interpolation method (DEIM) is first discussed. Then, two variants of missing data approximations using DEIM are later introduced.

5.1. Discrete Empirical Interpolation Method (DEIM)

DEIM was first introduced for the purpose of approximating the nonlinear term in the differential equations [35]. In this work, the interpolation indices from this method will be used to reduce the computational complexity of solving the least-squares approximation in

Step 3 of Algorithm 1. This section describes DEIM in a general setting and provides the corresponding greedy algorithm for selecting *important* components that can be used in the approximation.

Consider a vector $\mathbf{f} \in \mathbb{R}^n$. For a given orthonormal matrix $\mathbf{U} \in \mathbb{R}^{n\times m}$ with $m \leq n$, an approximation of this vector $\mathbf{f}$ obtained from projecting onto the subspace span$\{\mathbf{U}\}$ is given by

$$\mathbf{f} \approx \mathbf{U}\mathbf{d}, \tag{6}$$

for some vector $\mathbf{d} \in \mathbb{R}^m$. The vector $\mathbf{d}$ can be obtained by using an interpolation method. That is, it can be solved from a square system that extracts only m rows of (6). In particular, suppose $\wp_1, \wp_2, \ldots, \wp_m$ are the indices of the selected components in $\mathbf{f}$ and define $\mathbf{P} = [\mathbf{e}_{\wp_1}, \ldots, \mathbf{e}_{\wp_m}] \in \mathbb{R}^{n\times m}$, where $\mathbf{e}_{\wp_i} = [0, \ldots, 0, 1, 0, \ldots, 0]^T \in \mathbb{R}^n$ is the $\wp_i$-th column of the identity matrix $\mathbf{I}_n \in \mathbb{R}^{n\times n}$, for $i = 1, \ldots, m$. Note that, as in the previous section, pre-multiplying $\mathbf{P}^T$ is equivalent to extracting the m rows corresponding to the interpolation indices $\wp_1, \ldots, \wp_m$. The coefficient vector $\mathbf{d}$ in the DEIM approximation solves the following minimization problem

$$\min_{\mathbf{d}\in\mathbb{R}^m} \|\mathbf{P}^T\mathbf{f} - \mathbf{P}^T\mathbf{U}\mathbf{d}\|_2^2,$$

which gives $\mathbf{d} = (\mathbf{P}^T\mathbf{U})^+\mathbf{P}^T\mathbf{f}$, where $(\mathbf{P}^T\mathbf{U})^+ = [(\mathbf{P}^T\mathbf{U})^T(\mathbf{P}\mathbf{U})]^{-1}(\mathbf{P}^T\mathbf{U})^T$ is the pseudo-inverse of $\mathbf{P}^T\mathbf{U}$. Notice that $\mathbf{P}^T\mathbf{U}$ is a square matrix of size $m \times m$. When the indices $\wp_1, \ldots, \wp_m$ are selected by the DEIM procedure given in Algorithm 3, it was shown in [35] that $\mathbf{P}^T\mathbf{U}$ is invertible. i.e., $(\mathbf{P}^T\mathbf{U})^+ = (\mathbf{P}^T\mathbf{U})^{-1}$, and the approximation in (6) for $\mathbf{f}$ becomes

$$\mathbf{f} \approx \mathbf{U}(\mathbf{P}^T\mathbf{U})^{-1}\mathbf{P}^T\mathbf{f}. \tag{7}$$

This approximation (7) is called the DEIM approximation. The sets of indices $\{\wp_1, \wp_2, \ldots, \wp_m\}$ are obtained by the DEIM index selection algorithm [35], shown in Algorithm 3, which is a greedy procedure that aims to capture the variation of the spatial behavior of the input basis using the infinity norm.

Algorithm 3 Algorithm to create for Interpolation Indices DEIM

INPUT: $\{\mathbf{u}_\ell\}_{\ell=1}^m \subset \mathbb{R}^n$ linearly independent
OUTPUT: $\vec{\wp} = [\wp_1, \ldots, \wp_m]^T \in \mathbb{R}^m$ and $\mathbf{P}$

1: $\wp_1 = \arg\max_{i=1,2,\ldots,n}\{|\mathbf{u}_{i1}|\}$
2: $\mathbf{U} = [\mathbf{u}_1], \mathbf{P} = [\mathbf{e}_{\wp_1}], \vec{\wp} = [\wp_1]$;
3: **for** $j = 2$ to m **do**
4: Solve $(\mathbf{P}^T\mathbf{U})\mathbf{z} = \mathbf{P}^T\mathbf{u}_j$;
5: $\mathbf{r} = \mathbf{u}_j - \mathbf{U}\mathbf{z}$
6: $\wp_j = \arg\max_{i=1,\ldots,n}\{|\mathbf{r}_i|\}$
7: $\mathbf{U} \leftarrow [\mathbf{U} \quad \mathbf{u}_j], \mathbf{P} \leftarrow [\mathbf{P} \quad \mathbf{e}_{\wp_j}], \vec{\wp} \leftarrow \begin{bmatrix}\vec{\wp}\\ \wp_j\end{bmatrix}$
8: **end for**

From Algorithm 3, DEIM selects the interpolation indices so that the approximation has the smallest error $\mathbf{r} = \mathbf{u}_j - \mathbf{U}\mathbf{z}$ in each iteration j. The procedure of DEIM Algorithm 3 can be described as follows. First, the input basis set $\{\mathbf{u}_\ell\}_{\ell=1}^m$ of rank m. The space spanned by this basis set, in general, is expected to approximately contain the vector $\mathbf{f}$. Then, it selects the first index of a component in the first basis vector $\mathbf{u}_1$ with the largest absolute value. Next, each of the other indices is selected from the component with the largest absolute residual error $\mathbf{r} = \mathbf{u}_\ell - \mathbf{U}\mathbf{z}$ in each step.

More details on DEIM approximation and its corresponding error bound can be found in [35]. Some extensions of this error bound to the state-space error estimate can be found in [36,37].

5.2. Accelerated POD-LS Method by Using DEIM

This work particularly focuses on the situation when the incomplete data $\widehat{\mathbf{y}}$ is in a high-dimensional space, i.e., the value of n is large, which may result in a large number of known components n_c, even though there are a lot of unknown components n_g. In this case, the main computational work in approximating missing components may occur while solving for the coefficient vector $\mathbf{a}$ in (1) or in Step 3 of Algorithm 1. One possible way to reduce this computational work is to use only a small number of known components to specify $\mathbf{a} \in \mathbb{R}^k$. These components have to be carefully selected so that they can represent all other n_c components and maintain the same accuracy in the approximation. For this purpose, the procedure shown in Algorithm 3 for selecting DEIM indices, which corresponde to *important* components, will be used as described.

Recall from Step 3 of Algorithm 1 that it has to solve for a vector $\mathbf{a}$ from the following minimization problem:

$$\min_{\mathbf{a}\in\mathbb{R}^k} \|\widehat{\mathbf{y}}_c - \mathbf{V}_c\mathbf{a}\|_2^2.$$

Suppose $\mathbf{V}_c \in \mathbb{R}^{n_c\times k}$ has linearly independent columns. Then, for a given DEIM dimension $m \leq \min\{n_c, k\}$, the columns of $\mathbf{V}_c$ can be used as an input of Algorithm 3 to select m DEIM indices that can cover the variations of n_c components.

In many applications, when n is much larger than the number of available complete data n_s , i.e., $n > n_s \geq k$, we have $\min\{n, k\} = k$. Notice that, based on the standard DEIM approximation, dimension m is limited to the truncated dimension k of $\mathbf{V}$. We can improve the accuracy of the reconstruction by further including more important components of the complete data, i.e., using dimension m that is larger than k. However, using $k > m$ is impossible if we follow the standard DEIM approximation and use $\mathbf{V}_c$ as an input to Algorithm 3, because $\mathbf{V}_c$ has only k columns, which implies Algorithm 3 can give, at most, k indices. To resolve this issue, instead of using $\mathbf{V}_c$ directly, we will consider a matrix constructed from the full set of the POD basis for the complete snapshot matrix $\mathbf{Y}$ with rows selected from the indices of the known components. In particular, if we define $\bar{\mathbf{V}} \in \mathbb{R}^{n\times r}$ to be the left singular matrix from SVD of the complete snapshot matrix $\mathbf{Y}$, where $r = rank(\mathbf{Y})$, the input to Algorithm 3 would be the matrix obtained from $\bar{\mathbf{V}}$ with rows corresponding to the complete components, i.e.,

$$\bar{\mathbf{V}}_c := \mathbf{C}^T\bar{\mathbf{V}} \in \mathbb{R}^{n_c\times r}, \tag{8}$$

which comes from $\bar{\mathbf{V}}$ with selected rows corresponding to the n_c known components in $\mathcal{C}$. In this case, we can use the DEIM algorithm to select up to r indices that correspond to the most (r) relevant components of the reconstruction problem in (1). Note that the columns in $\mathbf{V}$ are the first k columns in $\bar{\mathbf{V}}$.

Suppose $\bar{\mathbf{V}}_c \in \mathbb{R}^{n_c\times r}$ has r linearly independent columns with $r \leq n_c$. Then, the columns of $\bar{\mathbf{V}}_c$ can be used in Algorithm 3 to select m DEIM indices, where $m \leq r$. In this case, suppose $\wp_1, \wp_2, \ldots, \wp_m$ are the output indices of DEIM Algorithm 3 and let $\mathbf{P} = [\mathbf{e}_{\wp_1}, \ldots, \mathbf{e}_{\wp_m}] \in \mathbb{R}^{n_c\times m}$ as defined earlier in this section. Vector $\mathbf{a}$ in (1) can be computed from a smaller least-squares problem:

$$\min_{\mathbf{a}\in\mathbb{R}^k} \|\mathbf{P}^T\widehat{\mathbf{y}}_c - \mathbf{P}^T\mathbf{V}_c\mathbf{a}\|_2^2. \tag{9}$$

Note that we have assumed that $\bar{\mathbf{V}}_c \in \mathbb{R}^{n_c\times r}$ has linearly independent columns, which may not hold in general cases. To avoid this assumption, we can apply SVD to the matrix $\bar{\mathbf{V}}_c$ and use the set of first m corresponding left singular vectors as an input to the DEIM Algorithm 3. The accelerated POD-LS steps are summarized in Algorithm 4 for both cases of (i) using $\bar{\mathbf{V}}_c$ directly as an input and (ii) using the set of left singular vectors of $\bar{\mathbf{V}}_c$ as an input to Algorithm 3 for selecting the most (m) important components. Note that, as

later shown in the numerical tests, the reconstruction results using case (ii) may not be as accurate as the ones using case (i).

Algorithm 4 Accelerated POD-LS method for approximating missing data

INPUT:

- Complete snapshot set $\{\mathbf{y}_j\}_{j=1}^{n_s} \subset \mathbb{R}^n$ and, dimension $k \leq rank(\{\mathbf{y}_j\}_{j=1}^{n_s})$
- Incomplete data $\widehat{\mathbf{y}} \in \mathbb{R}^n$ with known entries $\widehat{y}_j$, $j \in \mathcal{C}$ and unknown entries $\widehat{y}_j$, $j \in \mathcal{G}$

OUTPUT:

- Approximation: $\widehat{\mathbf{y}}_g = [\widehat{y}_j]$, $j \in \mathcal{G} = \{g_1, g_2, \ldots, g_{n_g}\}$

1: Create snapshot matrix : $\mathbf{Y} = [\mathbf{y}_1, \ldots, \mathbf{y}_{n_s}] \in \mathbb{R}^{n \times n_s}$ and let $r = rank(\mathbf{Y})$.
2: Compute POD basis $\mathbf{V}$ of rank $k \leq r$ for $\mathbf{Y}$ from rSVD in Algorithm 2.
3: Find coefficient vector $\mathbf{a}$
   3.1 Find indices $\vec{\wp} = [\wp_1, \ldots, \wp_m]^T \in \mathbb{R}^m$ and $\mathbf{P}$ from Algorithm 3 by using input from either
   (i) $\bar{\mathbf{V}}_c$ defined in (8), or
   (ii) Left singular vectors of $\bar{\mathbf{V}}_c$ using SVD or rSVD
   3.2 Solve $\mathbf{a}$ from (9): $\min_{\mathbf{a} \in \mathbb{R}^k} \|\mathbf{P}^T \widehat{\mathbf{y}}_c - \mathbf{P}^T \mathbf{V}_c \mathbf{a}\|_2^2$.
4: Compute the approximation $\widehat{\mathbf{y}}_g \approx \mathbf{V}_g \mathbf{a}$.

Remarks: In practice, the computation for DEIM indices can be performed in advance and reused for many incomplete snapshots. Once the DEIM indices are found from Algorithm 3, the terms $\mathbf{P}^T \widehat{\mathbf{y}}_c$ and $\mathbf{P}^T \mathbf{V}_c$ can be computed without actually performing matrix multiplication since this can be performed through selected row indices.

The next section discusses the computational complexity reduction of the proposed approach.

6. Computational Complexity

In this work, as shown in Algorithm 4, we have reduced the computational complexity in two main parts. The first one is based on using the rSVD for constructing the projection basis, and the other one is based on applying the DEIM procedure to efficiently select a small subset of the most relevant components in the least-squares problem used in the reconstruction process.

In the first part, for a given complete data matrix $\mathbf{Y} \in \mathbb{R}^{n \times n_s}$, computing a projection basis of rank k for $\mathbf{Y}$ by using SVD can be performed efficiently via a rank-revealing QR factorization and then manipulating the factors to obtain the final SVD. The cost of this approach is typically $\mathcal{O}(nn_sk)$ floating-point operations (flops) [26,46]. The rSVD can approximate SVD with computational cost that is reduced to $\mathcal{O}(nn_s \log(k) + (n + n_s)k^2)$ flops. This computational cost can be reduced further when using a random matrix Ω that has some internal structure to accelerate the computation of $\mathbf{Y}\Omega$ in Step 3 of Algorithm 2.

In the second part of reducing the computational cost for solving the least-squares problem, we incorporated the DEIM procedure to reduce the size of the problem in the reconstruction process. In general, the well-known approaches for solving least-squares problems are based on using normal equations, QR factorization and SVD [47–49]. The last two approaches are efficient for large-scaled problems [48,49]. For the coefficient matrix $\mathbf{V}_c$ of size n_c-by-k with $n_c \gg k$, the computational cost of solving the least-squares problem in (1) using the QR factorization or the SVD approaches is in the of order $\mathcal{O}(n_c n_s^2)$ flops. Therefore, when the DEIM algorithm is used to reduce the size of the least-squares problem to a coefficient matrix of size $m \times k$ with $m \ll n_c$ as shown in (9), the computational cost is reduced to $\mathcal{O}(mn_s^2)$ flops. Note that the complexity of performing the DEIM procedure is $\mathcal{O}(m^3)$, which is negligible for $m < n_s$ and $m \ll n_c$ in most practical cases when compared to the overall complexity of the reconstruction process.

The next section investigates the effectiveness of the proposed techniques described in Algorithm 4 through different cases of numerical tests. The reduction in computational cost explained above is also reflected in these reconstruction results.

7. Numerical Results

The numerical experiments in this section compare the efficiency and accuracy of the standard least-squares approach using the standard SVD with the accelerated least-squares approach using rSVD with DEIM. These results are obtained from MATLAB 2021b running on a computer with the following specifications Intel Core i5-1035G4, CPU 1.10, 1.50 GHz, and 8 GB RAM.

We consider three numerical experiments. The first two numerical experiments use data from images with missing pixels. The last one uses data from the solution of 2D nonlinear flow in porous media with some missing components at certain time instances. This dataset can be considered as a sequence of images in the form of movie data. The accuracy of the reconstruction is measured by the average relative error of the reconstructed missing data. The CPU times used in the reconstruction process will be scaled with the ones used by the standard LS approach to see the speedup of the simulation time.

7.1. Numerical Test 1

This section considers a standard test image, called the *Lena* image, of size 200×200 pixels. The goal is mainly to investigate the accuracy of the reconstruction by the proposed accelerated LS approach for different amounts of missing pixels. The speedup of the reconstruction time will be considered later in the next two numerical tests. We consider the cases when 30%, 50%, and 65% pixels are missing uniformly on the original image, as shown in Figure 1. The reconstruction results for all these cases are shown in Figure 2 when using the standard POD-LS approach with $k = 30$ and the proposed accelerated POD-LS approach with $k = 30$ and $m = 50$. Figure 3 demonstrates the average relative errors of the reconstructed missing components using the accelerated POD-LS approach for $k = 30$ and different values of $m = 10, 20, 30, \ldots, 80$ for the cases of 30%, 50%, and 65% missing pixels. It shows the convergence of the accelerated POD-LS approach to the standard POD-LS approach as dimension m increases. When there are more missing pixels, this convergence seems to be slower, i.e., the error of the accelerated POD-LS method approaches the error of the standard method at a larger dimension m. Note that, in Figure 3, the accelerated POD-LS approach for both cases (i) and (ii) in Algorithm 4 are performed. The errors for case (ii) seem to be larger than the ones for case (i), and these errors for case (ii) seem to be oscillating with no trend when $m < 30$. This suggests that to select the DEIM components it is more appropriate to use the matrix constructed from the rows of POD basis matrix corresponding to the indices of the know components in case (i) than to use its orthogonalized basis in case (ii).

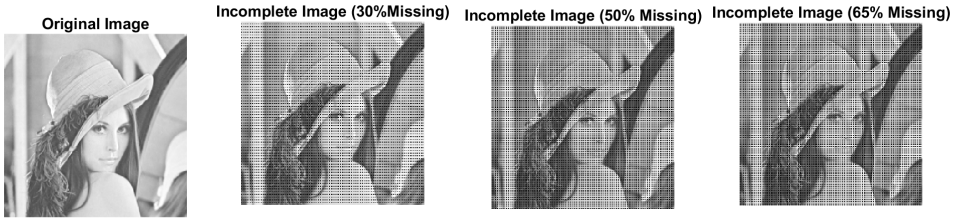


Figure 1. (Numerical Test 1) The original Lena image and incomplete images with 30%, 50%, and 65% missing pixels distributed uniformly over the image.

Figure 2. *Cont.*

Figure 2. (Numerical Test 1) Reconstructions of incomplete images with 30%, 50%, and 65% missing pixels by the standard POD-LS accelerated POD-LS approach using $k = 30$ (**Top**) and the proposed POD-LS approach using $k = 30$ and $m = 50$ (**Bottom**).

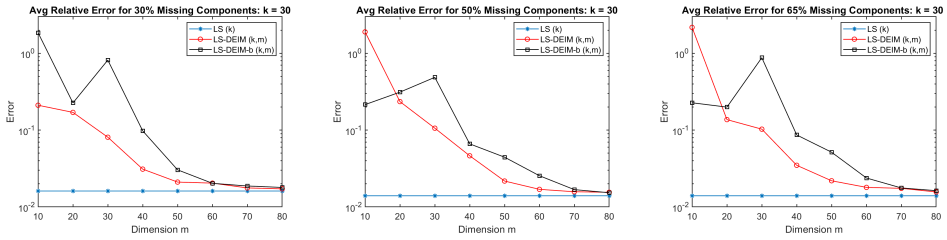


Figure 3. (Numerical Test 1) Average relative error of the reconstruction for 30%, 50%, and 65% missing components using the standard LS method and the accelerated LS with basis from rSVD, denoted by LS-DEIM for case (i) and LS-DEIM-b for case (ii) as described in Algorithm 4 for POD dimension $k = 30$ with DEIM dimension $m = 10, 20, 30, 40, 50, 60, 70, 80$.

7.2. Numerical Test 2

This section considers a test image of size 807×605 pixels. The corresponding incomplete image has uniform missing pixels equal to 50%, as shown in Figure 4. The goal of this numerical test is to investigate the accuracy and the speedup of the proposed accelerated POD-LS approach when compared to the standard POD-LS approach.

(**a**) (**b**)

Figure 4. (Numerical Test 2) (**a**) Original image and (**b**) an incomplete image with 50% missing pixels.

Figure 5 illustrates the reconstructed images by the standard POD-LS approach with basis from SVD using dimension $k = 10, 50$ and by the accelerated POD-LS approach with basis from rSVD as described in Algorithm 4 case (i) using $k = 10$ with $m = 10, 20, 50$ and $k = 50$ with $m = 50, 80, 150$. The corresponding values of the average relative errors and the scaled CPU time of the reconstructed images in Figure 5 are given in Tables 1 and 2, respectively, for the cases of $k = 10$ and $k = 50$. From these tables, the CPU time of the accelerated POD-LS approach is shown to be roughly reduced by 10–50 times when compared to the standard POD-LS approach for the same level of accuracy. Figure 6 demonstrates the trend of the average relative errors for these 2 approaches for different dimensions $k = 10, 30, 50$ with DEIM dimension m ranging from 10 to 210. Notice that, as the dimensions k and m get larger, the overall errors seem to decrease. Figure 6 also shows that as m increases, the errors of the accelerated POD-LS approach decrease and get closer to the errors of the standard POD-LS approach. As in the previous numerical test, in Figure 6, we consider the errors of the accelerated POD-LS approach for both cases (i)

and (ii) in Algorithm 4. The errors for case (ii) seem to be larger in all cases of k considered in Figure 6. This confirms the result in the previous numerical test that we should use the POD basis with rows extracted from corresponding to the known component directly, instead of using its orthogonalized basis, to select the DEIM points used in the accelerated least-squares approach.

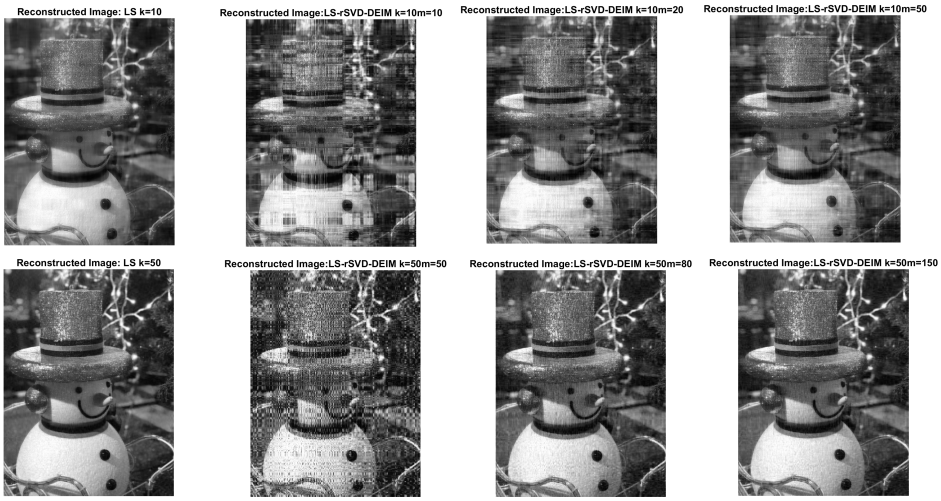


Figure 5. (Numerical Test 2) Reconstructed image from standard POD-LS with basis from SVD using dimension $k = 10, 50$ and from the accelerated POD-LS with basis from rSVD using $k = 10$ with $m = 10, 20, 50$ and using $k = 50$ with $m = 50, 80, 150$.

(**a**) (**b**) (**c**)

Figure 6. (Numerical Test 2) Average relative error of the reconstructed missing components using the standard POD-LS method and the accelerated POD-LS method, denoted by LS-DEIM for case (i) and LS-DEIM-b for case (ii) defined in Algorithm 4 for POD dimension $k = 10, 30, 50$ are shown in (**a**–**c**), respectively, with DEIM dimension $m = 10, 30, 50, \ldots, 210$.

Table 1. (Numerical Test 2) The average relative error of the reconstructed missing data and the corresponding scaled CPU times for the reconstruction when the standard POD-LS method and the accelerated POD-LS method with rSVD are used. Dimensions for POD: $k = 10$, DEIM: $m = 10, 20, 50$.

Method	dim POD	dim DEIM	Relative Error	CPU Time (Scaled)
Standard POD-LS	$k = 10$	-	0.0627	1
Accelerated POD-LS	$k = 10$	$m = 10$	0.8566	0.01382
	$k = 10$	$m = 20$	0.1239	0.02425
	$k = 10$	$m = 50$	0.0969	0.10185

Table 2. (Numerical Test 2) The average relative error of the reconstructed missing data and the corresponding scaled CPU times for the reconstruction when the standard POD-LS method and the accelerated POD-LS method with rSVD are used. Dimensions for POD: $k = 50$, DEIM: $m = 50, 80, 150$.

Method	dim POD	dim DEIM	Relative Error	CPU Time (Scaled)
Standard POD-LS	$k = 50$	-	0.01753	1
Accelerated POD-LS	$k = 50$	$m = 50$	9.20223	0.02120
	$k = 50$	$m = 80$	0.05181	0.03578
	$k = 50$	$m = 150$	0.02855	0.04114

7.3. Numerical Test 3

This section uses snapshots from the solutions of miscible flow, which changes continuously with no particular pattern for each time instance, as shown in Figure 7. Each snapshot has n =15,000 pixels. The POD basis is constructed from $n_s = 200$ complete snapshots and used to reconstruct the missing pixels of 50 incomplete snapshots. Each incomplete snapshot has 30% missing pixels, as demonstrated in Figure 8. The average relative error for the reconstruction and the corresponding CPU time are displayed in Figure 9 for different cases of POD dimensions $k = 10, 30, 50, 100$. For each of these cases, different values of DEIM dimensions $m = 10, 30, \ldots, 190$ are used in the accelerated POD-LS approach. Figure 9 demonstrates the convergence of the reconstruction error and the speed up of the CPU time of the accelerated POD-LS and the standard POD-LS approaches as in the previous numerical tests. In this numerical test, the accelerated POD-LS can be used to speed up the CPU time by roughly $\mathcal{O}(10)$ to $\mathcal{O}(100)$ times. Note that the dimension of m that is suitable in each case of k is generally selected from the *corner* of the error plot before the error becomes constant. Therefore, from Figure 9, the error plot from the accelerated least-squares approach using case (i) in Algorithm 4 suggests that we should use an m that is slightly larger than k, for each case of $k = 10, 30, 50, 100$.

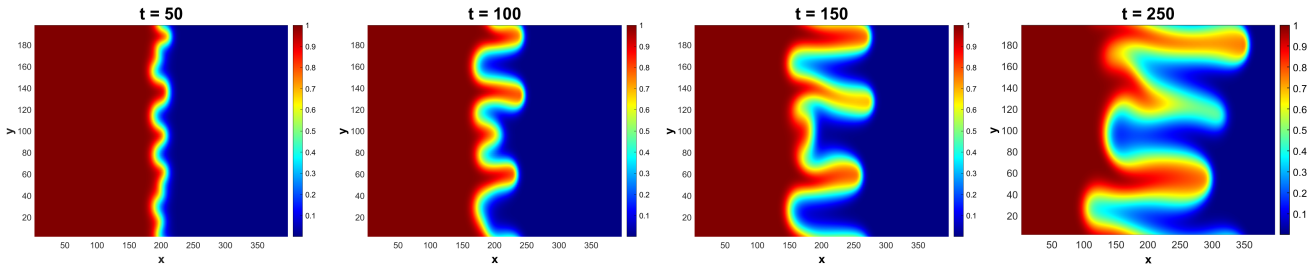
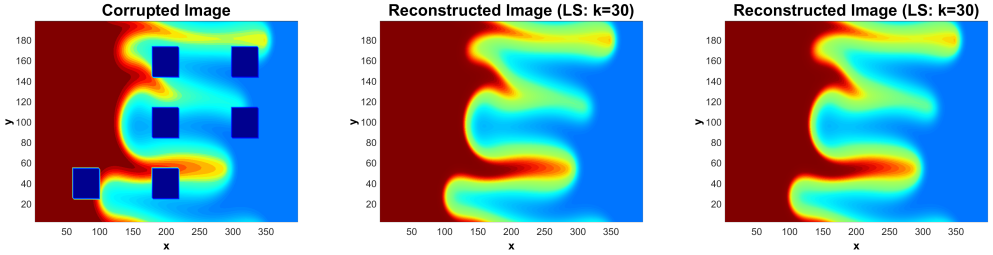


Figure 7. (Numerical Test 3) Snapshots of flow data at different four time instances.



(**a**) (**b**) (**c**)

Figure 8. (Numerical Test 3) Example of an incomplete image (**a**), with the reconstructions from the standard least-squares with SVD (**b**) and the accelerated least-squares with rSVD (**c**).

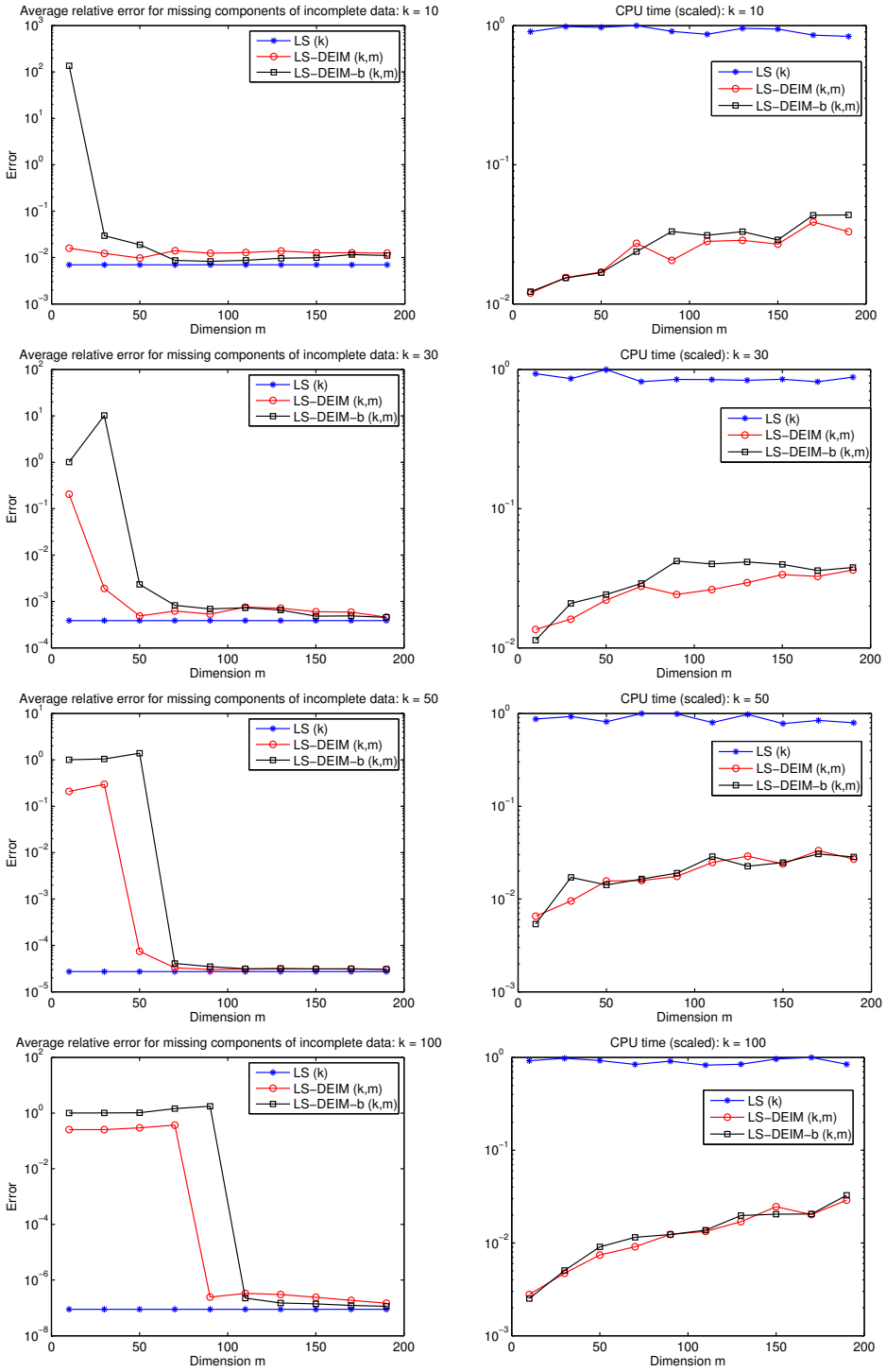
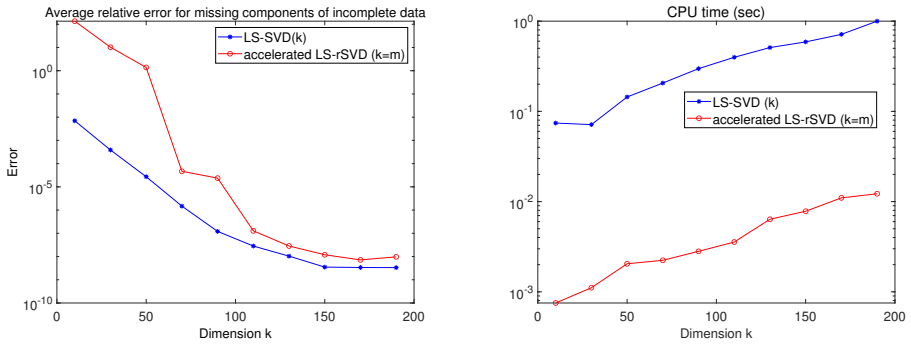


Figure 9. (Numerical Test 3) Average relative error of the reconstructed missing components using the standard LS method and the accelerated least-squares with basis from rSVD, denoted by LS-DEIM for case (i) and LS-DEIM-b for case (ii) defined in Algorithm 4 for POD dimensions $k = 10, 30, 50$ with DEIM dimensions $m = 10, 30, 50, 70, 90, 110, 130, 150, 170, 190$.

To clearly illustrate the effect of dimension k on the POD basis, Figure 10 compares the average relative error and the corresponding CPU time used in the reconstruction from the accelerated POD-LS approach using $m = k$ with the ones from the standard POD-LS approach for each k ranging from 10 to 190. Figure 10 shows that the accelerated POD-LS approach can provide almost the same accurate reconstruction as the standard POD-LS roughly 100 times faster. This numerical test, therefore, demonstrates the potential of the proposed approach for other large-scale problems.



(**a**) (**b**)

Figure 10. (Numerical Test 3) Reconstruction error (**a**) and CPU time (**b**) of the incomplete image from the standard least-squares approach and from the accelerated least-squares approach case (i) in Algorithm 4 using different dimensions of k and $m = k$.

8. Conclusions

This work presented an accelerated data reconstruction technique based on the least-squares method, which can reduce computational time in two aspects. The first one is based on using rSVD in computing a low-dimensional basis used in least-squares approximation. The second aspect that can speed up the reconstruction process is based on applying the DEIM greedy point selection procedure on the least-squares approximation. Accuracy and computational time reduction of the proposed method are demonstrated through three numerical experiments, which considered standard test images, as well as a set of sequential two-dimensional flow images in a similar form as video data. These numerical results show that the proposed method can speed up the reconstruction process up to 100 times with a small accuracy trade-off. This work can be extended by employing different approaches to construct the projection basis, e.g., [50–52]. It is also possible to adapt other approaches used for solving differential equations, e.g., in [53–56], to select the relevant components used in the least-squares approximation for data reconstruction. These extensions are left for future investigations.

Author Contributions: S.C. study planning, project administration, design, conceptualization, methodology, manuscript preparation, writing—review and editing, numerical analysis, funding acquisition; S.C. and S.I. background research, paper review, writing—original draft preparation, numerical experiments. All authors have read and agreed to the published version of the manuscript.

Funding: The authors gratefully acknowledge the financial support provided by the Faculty of Science and Technology, Contract No. SciGR 12/2565.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: The authors would like to thank the reviewers for many insightful comments to improve this paper.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

POD	Proper Orthogonal Decomposition
DEIM	Discrete Empirical Interpolation
LS	Least-squares
SVD	Singular Value Decomposition
rSVD	Randomized Singular Value Decomposition

References

1. Zhang, Q.; Yuan, Q.; Zeng, C.; Li, X.; Wei, Y. Missing data reconstruction in remote sensing image with a unified spatial–temporal–spectral deep convolutional neural network. *IEEE Trans. Geosci. Remote. Sens.* **2018**, *56*, 4274–4288. [CrossRef]
2. Chai, X.; Gu, H.; Li, F.; Duan, H.; Hu, X.; Lin, K. Deep learning for irregularly and regularly missing data reconstruction. *Sci. Rep.* **2020**, *10*, 3302.
3. Baraldi, P.; Di Maio, F.; Genini, D.; Zio, E. Reconstruction of missing data in multidimensional time series by fuzzy similarity. *Appl. Soft Comput.* **2015**, *26*, 1–9. [CrossRef]
4. Zhou, Y.; Wang, S.; Wu, T.; Feng, L.; Wu, W.; Luo, J.; Zhang, X.; Yan, N. For-backward LSTM-based missing data reconstruction for time-series Landsat images. *Gisci. Remote. Sens.* **2022**, *59*, 410–430. [CrossRef]
5. Hafner, J.L.; Deenadhayalan, V.; Rao, K.; Tomlin, J.A. *Matrix Methods for Lost Data Reconstruction in Erasure Codes;* USENIX Association: Berkeley, CA, USA, 2005; Volume 5, pp. 15–30.
6. Kreimer, N.; Stanton, A.; Sacchi, M.D. Tensor completion based on nuclear norm minimization for 5D seismic data reconstruction. *Geophysics* **2013**, *78*, V273–V284. [CrossRef]
7. Liu, T.; Wei, H.; Zhang, K. Wind power prediction with missing data using Gaussian process regression and multiple imputation. *Appl. Soft Comput.* **2018**, *71*, 905–916. [CrossRef]
8. Maillo, J.; Ramírez, S.; Triguero, I.; Herrera, F. kNN-IS: An Iterative Spark-based design of the k-Nearest Neighbors classifier for big data. *Knowl. Based Syst.* **2017**, *117*, 3–15. [CrossRef]
9. Liu, Z.-G.; Liu, Y.; Dezert, J.; Pan, Q. Classification of incomplete data based on belief functions and K-nearest neighbors. *Knowl.-Based Syst.* **2015**, *89*, 113–125. [CrossRef]
10. Scholz, M.; Kaplan, F.; Guy, C.L.; Kopka, J.; Selbig, J. Non-linear PCA: A missing data approach. *Bioinformatics* **2005**, *21*, 3887–3895. [CrossRef]
11. Bø, T.H.; Dysvik, B.; Jonassen, I. LSimpute: Accurate estimation of missing values in microarray data with least squares methods. *Nucleic Acids Res.* **2004**, *32*, e34. [CrossRef]
12. Ren, H.; Gao, F.; Jiang, X. Least-squares method for data reconstruction from gradient data in deflectometry. *Appl. Opt.* **2016**, *55*, 6052–6059. [CrossRef]
13. Kaplan, S.T.; Naghizadeh, M.; Sacchi, M.D. Data reconstruction with shot-profile least-squares migration. *Geophysics* **2010**, *75*, WB121–WB136. [CrossRef]
14. Berkooz, G.; Holmes, P.; Lumley, J.L. The proper orthogonal decomposition in the analysis of turbulent flows. *Annual Rev. Fluid Mech* **1993**, *25*, 539–575. [CrossRef]
15. Lanata, F.; Grosso, A.D. Damage detection and localization for continuous static monitoring of structures using a proper orthogonal decomposition of signals. *Smart Mater. Struct.* **2006**, *15*, 1811. [CrossRef]
16. Schenone, E. Reduced Order Models, Forward and Inverse Problems in Cardiac Electrophysiology. Ph.D. Thesis, Université Pierre et Marie Curie, Paris, France, 2014.
17. Gurka, R.; Liberzon, A.; Hetsroni, G. POD of vorticity fields: A method for spatial characterization of coherent structures. *Int. J. Heat Fluid Flow* **2006**, *27*, 416–423. [CrossRef]
18. Tsering-Xiao, B.; Xu, Q. Gappy POD-based reconstruction of the temperature field in Tibet. *Theor. Appl. Climatol.* **2019**, *138*, 1179–1188. [CrossRef]
19. Bui-Thanh, T.; Damodaran, M.; Willcox, K. Aerodynamic Data Reconstruction and Inverse Design Using Proper Orthogonal Decomposition. *AIAA* **2004**, *42*, 1505–1516. [CrossRef]
20. Bizon, K.; Continillo, G.; Merola, S.S.; Vaglieco, B.M. Reconstruction of flame kinematics and analysis of cycle variation in a Spark Ignition Engine by means of Proper Orthogonal Decomposition. *Comput. Aided Chem. Eng.* **2009**, *26*, 1039–1043. [CrossRef]
21. Choi, O.; Lee, M.C. Investigation into the combustion instability of synthetic natural gases using high speed flame images and their proper orthogonal decomposition. *Int. J. Hydrog. Energy* **2016**, *41*, 20731–20743. [CrossRef]
22. Bouhoubeiny, E.; Druault, P. Note on the POD-based time interpolation from successive PIV images. *Comptes Rendus MéCanique* **2009**, *337*, 776–780. [CrossRef]
23. Wang, M.; Dutta, D.; Kim, K.; Brigham, J.C. A computationally efficient approach for inverse material characterization combining Gappy {POD} with direct inversion. *Comput. Methods Appl. Mech. Eng.* **2015**, *286*, 373–393. [CrossRef]
24. Lei, J.; Qiu, J.; Liu, S. Dynamic reconstruction algorithm for electrical capacitance tomography based on the proper orthogonal decomposition. *Appl. Math. Model.* **2015**, *39*, 6925–6940. [CrossRef]
25. Sun, X.-H.; Xu, M.-H. Optimal control of water flooding reservoir using proper orthogonal decomposition. *J. Comput. Appl. Math.* **2017**, *320*, 120–137. [CrossRef]
26. Halko, N.; Martinsson, P.; Tropp, J. Finding Structure with Randomness: Probabilistic Algorithms for Constructing Approximate Matrix Decompositions. *SIAM Rev.* **2011**, *53*, 217–288. [CrossRef]
27. Ji, H.; Yu, W.; Li, Y. A Rank Revealing Randomized Singular Value Decomposition (R3SVD) Algorithm for Low-rank Matrix Approximations. *arXiv* **2016**, arXiv:1605.08134
28. Zhang, J.; Erway, J.; Hu, X.; Zhang, Q.; Plemmons, R. Randomized SVD Methods in Hyperspectral Imaging. *J. Electr. Comput. Eng.* **2012**, *2012*, 2737–2764. [CrossRef]
29. Ji, H.; Li, Y. GPU Accelerated Randomized Singular Value Decomposition and Its Application in Image Compression. In Proceedings of the MSVESCC, Suffolk, VA, USA, 17 April 2014.

30. Intawichai, S.; Chaturantabut, S. Missing Image Data Reconstruction Based on Least-Squares Approach with Randomized SVD. In Proceedings of the International Conference on Intelligent Computing & Optimization, Koh Samui, Thailand, 17–18 December 2020; Springer: Berlin/Heidelberg, Germany, 2020; pp. 1059–1070.
31. Yu, W.; Gu, Y.; Li, Y. Efficient randomized algorithms for the fixed-precision low-rank matrix approximation. *Siam J. Matrix Anal. Appl.* **2018**, *39*, 1339–1359. [CrossRef]
32. Voronin, S.; Martinsson, P.G. RSVDPACK: An implementation of randomized algorithms for computing the singular value, interpolative, and CUR decompositions of matrices on multi-core and GPU architectures. *arXiv* **2015**, arXiv:1502.05366
33. Musco, C.; Musco, C. Randomized Block Krylov Methods for Stronger and Faster Approximate Singular Value Decomposition. In Proceedings of the 28th International Conference on Neural Information Processing Systems, Montreal, QC, Canada, 7–12 December 2015; MIT Press: Cambridge, MA, USA, 2015; Volume 1, pp. 1396–1404.
34. Li, H.; Linderman, G.C.; Szlam, A.; Stanton, K.P.; Kluger, Y.; Tygert, M. Algorithm 971: An implementation of a randomized algorithm for principal component analysis. *ACM Trans. Math. Softw.* **2017**, *43*, 1–14. [CrossRef]
35. Chaturantabut, S.; Sorensen, D.C. Nonlinear Model Reduction via Discrete Empirical Interpolation. *SIAM J. Sci. Comput.* **2010**, *32*, 2737–2764. [CrossRef]
36. Wirtz, D.; Sorensen, D.C.; Haasdonk, B. A posteriori error estimation for DEIM reduced nonlinear dynamical systems. *Siam J. Sci. Comput.* **2014**, *36*, A311–A338. [CrossRef]
37. Chaturantabut, S. Stabilized model reduction for nonlinear dynamical systems through a contractivity-preserving framework. *Int. J. Appl. Math. Comput. Sci.* **2020**, *30*, 615–628.
38. Ştefănescu, R.; Navon, I.M. POD/DEIM nonlinear model order reduction of an ADI implicit shallow water equations model. *J. Comput. Phys.* **2013**, *237*, 95–114. [CrossRef]
39. Stefanescu, R.; Sandu, A.; Navon, I.M. POD/DEIM Strategies for reduced data assimilation systems. *J. Comput. Phys.* **2014**, *295*, 569–595.
40. Feng, Z.; Soulaimani, A. Reduced Order Modelling Based on POD Method for 3D Nonlinear Aeroelasticity. In Proceedings of the 18th IASTED International Conference on Modelling and Simulation, Montreal, QC, Canada, 30 May–1 June 2007; ACTA Press: Anaheim, CA, USA, 2007; pp. 489–494.
41. Ploymaklam, N.; Chaturantabut, S. Reduced-Order Modeling of a Local Discontinuous Galerkin Method for Burgers-Poisson Equations. *Thai J. Math.* **2020**, *18*, 2053–2069.
42. Xiao, D.; Fang, F.; Buchan, A.G.; Pain, C.C.; Navon, I.M.; Du, J.; Hu, G. Non-linear model reduction for the Navier–Stokes equations using residual DEIM method. *J. Comput. Phys.* **2014**, *263*, 1–18. [CrossRef]
43. Chaturantabut, S. Accelerated POD least-squares approach for missing data reconstruction. In Proceedings of the 17th International Conference on Mathematical Methods in Science and Engineering, Beijing, China, 21–22 May 2017; pp. 563–575.
44. Eckart, C.; Young, G. The approximation of one matrix by another of lower rank. *Psychometrika* **1936**, *1*, 211–218. [CrossRef]
45. Volkwein, S. *Proper Orthogonal Decomposition: Theory and Reduced-Order Modelling*; Lecture Notes; University of Konstanz: Konstanz, Germany, 2013
46. Gu, M.; Eisenstat, S.C. Efficient Algorithms for Computing a Strong Rank-Revealing QR Factorization. *Siam J. Sci. Comput.* **1996**, *17*, 848–869. [CrossRef]
47. Lawson, C.L.; Hanson, R.J. *Solving Least Squares Problems*; SIAM: Philadelphia, PA, USA, 1995.
48. Golub, G. Numerical methods for solving linear least squares problems. *Numer. Math.* **1965**, *7*, 206–216. [CrossRef]
49. Lee, D.Q. *Numerically Efficient Methods for Solving Least Squares Problems*; Pennsylvania State University: State College, PA, USA, 2012.
50. Gu, Z.; Lin, W.; Lee, B.S.; Lau, C.T.; Paul, M. Two dimensional singular value decomposition (2D-SVD) based video coding. In Proceedings of the 2010 IEEE International Conference on Image Processing, Haikou, China, 11–12 November 2010; pp. 181–184.
51. Gutiérrez, P.A.; Hervás, C.; Carbonero, M.; Fernández, J.C. Combined projection and kernel basis functions for classification in evolutionary neural networks. *Neurocomputing* **2009**, *72*, 2731–2742. [CrossRef]
52. Diener, J.; Rodriguez, M.; Baboud, L.; Reveret, L. Wind projection basis for real-time animation of trees. In *Computer Graphics Forum*; Wiley Online Library: Hoboken, NJ, USA, 2009; Volume 28, pp. 533–540.
53. Liu, X.; Liu, G.; Tai, K.; Lam, K. Radial point interpolation collocation method (RPICM) for the solution of nonlinear Poisson problems. *Comput. Mech.* **2005**, *36*, 298–306. [CrossRef]
54. Weickert, J.; Welk, M. Tensor field interpolation with PDEs. In *Visualization and Processing of Rensor Fields*; Springer: Berlin/Heidelberg, Germany, 2006; pp. 315–325.
55. Nguyen, N.C.; Patera, A.T.; Peraire, J. A 'best points' interpolation method for efficient approximation of parametrized functions. *Int. J. Numer. Methods Eng.* **2008**, *73*, 521–543. [CrossRef]
56. Groza, G.; Pop, N. Approximate solution of multipoint boundary value problems for linear differential equations by polynomial functions. *J. Differ. Equ. Appl.* **2008**, *14*, 1289–1309. [CrossRef]