Randomized Dynamic Mode Decomposition

N. Benjamin Erichson[†][¶], Lionel Mathelin[‡][†], J. Nathan Kutz[†], and Steven L. Brunton[§]

Abstract. This paper presents a randomized algorithm for computing the near-optimal low-rank dynamic mode decomposition (DMD). Randomized algorithms are emerging techniques to compute low-rank matrix approximations at a fraction of the cost of deterministic algorithms, easing the computational challenges arising in the area of 'big data'. The idea is to derive a small matrix from the high-dimensional data, which is then used to efficiently compute the dynamic modes and eigenvalues. The algorithm is presented in a modular probabilistic framework, and the approximation quality can be controlled via oversampling and power iterations. The effectiveness of the resulting randomized DMD algorithm is demonstrated on several benchmark examples of increasing complexity, providing an accurate and efficient approach to extract spatiotemporal coherent structures from big data in a framework that scales with the intrinsic rank of the data, rather than the ambient measurement dimension. For this work we assume that the dynamics of the problem under consideration is evolving on a low-dimensional subspace that is well characterized by a fast decaying singular value spectrum.

Key words. Dynamic mode decomposition, randomized algorithm, dimension reduction, dynamical systems.

1. Introduction.

Extracting dominant coherent structures and modal expansions from high-dimensional data is a cornerstone of computational science and engineering [62]. The dynamic mode decomposition (DMD) is a leading data-driven algorithm to extract spatiotemporal coherent structures from high-dimensional data sets [59, 68, 38]. DMD originated in the fluid dynamics community [59, 55], where the identification of coherent structures, or *modes*, is often an important step in building models for prediction, estimation, and control [9, 62]. In contrast to the classic proper orthogonal decomposition (POD) [4, 35], which orders modes based on how much energy or variance of the flow they capture, DMD identifies spatially correlated modes that oscillate at a fixed frequency in time, possibly with an exponentially growing or decaying envelope. Thus, DMD combines the advantageous features of POD in space and the Fourier transform in time [38].

With rapidly increasing volumes of measurement data from simulations and experiments, modal extraction algorithms such as DMD may become prohibitively expensive, especially for online or real-time analysis. Even though DMD is based on an efficient singular value decomposition (SVD), computations scale with the dimension of the measurements, rather than with the intrinsic dimension of the data. With increasingly vast measurements, it is often the case that the intrinsic rank of the data does not increase appreciably, even as the dimension of the ambient measurements grows. Indeed, many high-dimensional systems exhibit low-dimensional patterns and coherent structures, which is one of the driving perspectives in both POD and DMD analysis.

[†]Department of Applied Mathematics, University of Washington, Seattle, WA (kutz@uw.edu)
[¶]ICSI and Department of Statistics, University of California, Berkeley (erichson@berkeley.edu)
[‡]LIMSI–CNRS, Campus Universitaire d'Orsay, 91405 Orsay cedex, France (mathelin@limsi.fr)
[§]Department of Mechanical Engineering, University of Washington, Seattle, WA (sbrunton@uw.edu)

In this work, we develop a computationally effective strategy to compute the DMD using randomized linear algebra, which scales with the intrinsic rank of the dynamics, rather than with the measurement dimension. More concretely, we embed the DMD in a probabilistic framework by following the seminal work of Halko et al. [29]. The main concept is depicted in Figure 1. Numerical experiments show that our randomized algorithm achieves considerable speedups over previously proposed algorithms for computing the DMD such as the compressed DMD algorithm [10]. Further, the approximation error can be controlled via oversampling and additional power iterations. This allows the user to choose the optimal trade-off between computational time and accuracy. Importantly, we also demonstrate the ability to handle data which are too big to fit into fast memory by using a blocked matrix scheme.

1.1. Related Work. Efforts to address the computational challenges of DMD can be traced back to the original paper by Schmid [59]. It was recognized that DMD could be computed in a lower dimensional space and then used to reconstruct the dominant modes and dynamics in the original high-dimensional space. This philosophy has been adopted in several strategies to determine the low-dimensional subspace, as shown in Fig. 2. Since then, efficient parallelized algorithms have been proposed for computations at scale [57].

In [59], the high-dimensional data are projected onto the linear span of the associated POD modes, enabling an inexpensive low-dimensional DMD. The high-dimensional DMD modes have the same temporal dynamics as the low-dimensional modes and are obtained via lifting with the POD modes. However, computing the POD modes may be expensive and strategies have been developed to alleviate this cost. An algorithm utilizing the randomized singular value decomposition (rSVD) was presented in [22] and [5, 6]. While this approach is reliable and robust to noise, only the computation of the SVD is accelerated, so that subsequent computational steps involved in the DMD algorithm remain expensive. The paper by Bistrian

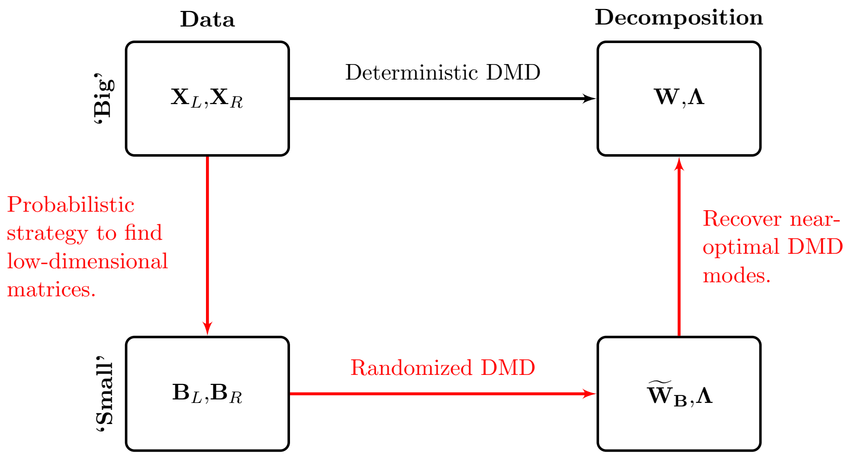

Data
Decomposition
'Big'
$\mathbf{X}_L$,$\mathbf{X}_R$
Deterministic DMD
$\mathbf{W}$,$\mathbf{\Lambda}$
Probabilistic strategy to find low-dimensional matrices.
Recover near-optimal DMD modes.
'Small'
$\mathbf{B}_L$,$\mathbf{B}_R$
Randomized DMD
$\widetilde{\mathbf{W}}_\mathbf{B}$,$\mathbf{\Lambda}$

Figure 1: Conceptual architecture of the randomized dynamic mode decomposition (rDMD). First, small matrices are derived from the high-dimensional input data. The low-dimensional snapshot matrices $\mathbf{B}_L$ and $\mathbf{B}_R$ are then used to compute the approximate dynamic modes $\widetilde{\mathbf{W}}_\mathbf{B}$, and eigenvalues $\mathbf{\Lambda}$. Finally, the near-optimal modes $\mathbf{W}$ may be recovered.

and Navon [5] has other advantages, including identifying the most influential DMD modes and guarantees that the low-order model satisfies the boundary conditions of the full model.

As an alternative to using POD modes, a low-dimensional approximation of the data can be obtained by random projections [10]. This is justified by the Johnson-Lindenstrauss lemma [36, 23] which provides statistical guarantees on the preservation of the distances between the data when projected into the low-dimensional space. This approach avoids the cost of computing POD modes and has favorable statistical error bounds.

In contrast with these projection-based approaches, alternative strategies for reducing the dimension of the data involve computations with subsampled snapshots. In this case, the reduction of the computational cost comes from the fact that only a subset of the data is used to derive the DMD and the full data is never processed by the algorithm. DMD is then determined from a *sketch-sampled* database. Variants of this approach differ in the sampling strategy; for example, [10] and [20] develop the compressed dynamic mode decomposition, which involves forming a small data matrix by randomly projecting the high-dimensional row-space of a larger data matrix. This approach has been successful in computational fluid dynamics and video processing applications, where the data have relatively low noise. However, this is a suboptimal strategy, which leads to a large variance in the performance due to poor statistical guarantees in the resulting sketched data matrix. Clustering techniques can be employed to select a relevant subset of the data [28]. Further alternatives derive from algebraic considerations, such as the maximization of the volume of the submatrix extracted from the data (e.g., Q-DEIM [18, 45]) or rely on energy-based arguments such as leverage-score and length-squared samplings [24, 44, 16].

1.2. Assumptions, Limitations, Applications, and Extensions. Within a short time, DMD has become a workhorse algorithm for extracting dominant low-dimensional oscillating patterns from high-dimensional data. One key benefit of DMD is that it is equally valid for experimental and numerical data, as it does not rely on knowledge of the governing equations. Although it may not be stated explicitly, it is often assumed that the data are

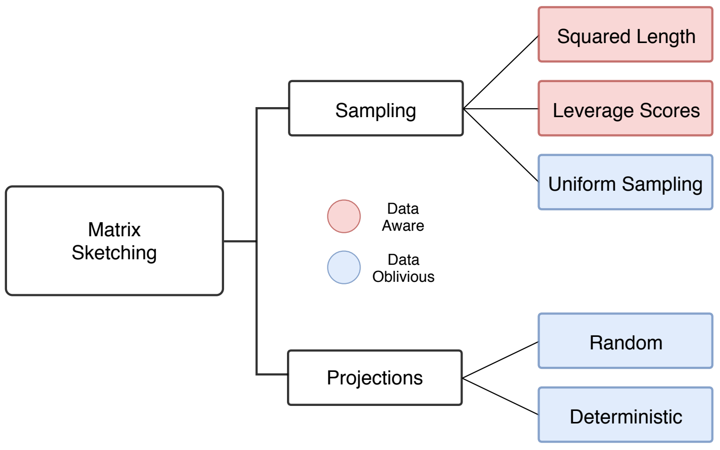


Figure 2: Sketching techniques for high-dimensional data; this work uses random projections.

periodic or quasi-periodic in nature. DMD typically fails for data that is non-stationary or that exhibits broadband or intermittent phenomena. DMD is also quite sensitive to noise [19, 14], although several algorithms exist to de-bias DMD results in the presence of noisy data [14, 34]. Although not a limitation, most applications of DMD involve high-dimensional systems that evolve on a low-dimensional attractor, in which case the singular value decomposition is used to determine the subspace. Despite the assumptions and limitations above, DMD has been widely applied on a variety of systems in fluid mechanics [55, 58, 42, 60, 2], epidemiology [51], neuroscience [7], video modeling [21, 22, 40], robotics [3], and plasma physics [65]. Much of the success of DMD relates to its simple formulation in terms of linear algebra. The simplicity of DMD has enabled several extensions, including for control [50], multiresolution analysis [39], recursive orthogonalization of modes for Galerkin projection [48], the use of time-delay coordinates [68, 1, 13], sparse identification of dynamic regimes [37], Bayesian formulations [64].

Many of the applications and extensions above fundamentally rely on the dynamics evolving on a low-dimensional subspace that is well characterized by a fast decaying singular value spectrum. Although this is not a fundamental limitation of DMD, it is often a basic assumption, and we rely on this low-rank structure here.

1.3. Contribution of the Present Work.

This work presents a randomized DMD algorithm that enables the accurate and efficient extraction of spatiotemporal coherent structures from high-dimensional data. The algorithm scales with the intrinsic rank of the dynamics, which are assumed to be low-dimensional, rather than the ambient measurement dimension. We demonstrate the effectiveness of this algorithm on two data sets of increasing complexity, namely the canonical fluid flow past a circular cylinder and the high-dimensional sea-surface temperature dataset. Moreover, we show that is possible to control the accuracy of the algorithm via oversampling and power iterations. In order to promote reproducible research, our open-source code is available at https://github.com/erichson/ristretto.

The remainder of the paper is organized as follows. Section 2 presents notation and background concepts used throughout the paper. Section 3 outlines the probabilistic framework used to compute the randomized DMD, which is presented in Sec. 4. Section 5 provides numerical results to demonstrate the performance of the randomized DMD algorithm. Final remarks and an outlook are given in Sec. 6.

2. Technical Preliminaries.

We now establish notation and provide a brief overview of the singular value decomposition (SVD) and the dynamic mode decomposition (DMD).

2.1. Notation.

Vectors in $\mathbb{R}^n$ and $\mathbb{C}^n$ are denoted as bold lowercase letters $\mathbf{x} = [x_1, x_2, ..., x_n]$. Both real $\mathbb{R}^{n\times m}$ and complex $\mathbb{C}^{n\times m}$ matrices are denoted by bold capitals $\mathbf{X}$, and its entry at the i-th row and j-th column is denoted as $\mathbf{X}(i, j)$. The Hermitian transpose of a matrix is denoted as $\mathbf{X}^*$. The spectral or operator norm of a matrix is defined as the largest singular value $\sigma_{\max}(\mathbf{X})$ of the matrix $\mathbf{X}$, i.e., the square root of the largest eigenvalue $\lambda_{\max}$ of the positive-semidefinite matrix $\mathbf{X}^*\mathbf{X}$:

$$\|\mathbf{X}\|_2 = \sqrt{\lambda_{\max}(\mathbf{X}^*\mathbf{X})} = \sigma_{\max}(\mathbf{X}).$$

The Frobenius norm of a matrix $\mathbf{X}$ is the positive square root of the sum of the absolute squares of its elements, which is equal to the positive square root of the trace of $\mathbf{X}^*\mathbf{X}$

$$\|\mathbf{X}\|_F = \sqrt{\sum_{i=1}^{n}\sum_{j=1}^{m} |\mathbf{X}(i,j)|^2} = \sqrt{\text{trace}(\mathbf{X}^*\mathbf{X})}.$$

The relative reconstruction error is $\|\mathbf{X} - \widehat{\mathbf{X}}\|_F / \|\mathbf{X}\|_F$, where $\widehat{\mathbf{X}}$ is an approximation to the matrix $\mathbf{X}$. The column space (range) of $\mathbf{X}$ is denoted $\text{col}(\mathbf{X})$ and the row space is $\text{row}(\mathbf{X})$.

2.2. The Singular Value Decomposition. Given an $n \times m$ matrix $\mathbf{X}$, the 'economic' singular value decomposition (SVD) produces the factorization

$$\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^* = \sum_{i}^{r} \mathbf{u}_i \sigma_i \mathbf{v}_i^*, \tag{2.1}$$

where $\mathbf{U} = [\mathbf{u}_1, ..., \mathbf{u}_r] \in \mathbb{R}^{n\times r}$ and $\mathbf{V} = [\mathbf{v}_1, ..., \mathbf{v}_r] \in \mathbb{R}^{m\times r}$ are orthonormal, $\mathbf{\Sigma} \in \mathbb{R}^{r\times r}$ is diagonal, and $r = \min(m, n)$. The left singular vectors in $\mathbf{U}$ provide a basis for the column space of $\mathbf{X}$, and the right singular vectors in $\mathbf{V}$ form a basis for the row space of $\mathbf{X}$. $\mathbf{\Sigma}$ contains the corresponding nonnegative singular values $\sigma_1 \geq ... \geq \sigma_r \geq 0$. Often, only the k dominant singular vectors and values are of interest, resulting in the low-rank SVD:

$$\mathbf{X}_k = \mathbf{U}_k\mathbf{\Sigma}_k\mathbf{V}_k^* = [\mathbf{u}_1, \ldots, \mathbf{u}_k]\text{diag}(\sigma_1, \ldots, \sigma_k)[\mathbf{v}_1, \ldots, \mathbf{v}_k]^* = \sum_{i}^{k} \mathbf{u}_i \sigma_i \mathbf{v}_i^*.$$

The Moore-Penrose Pseudoinverse. Given the singular value decomposition $\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^*$, the pseudoinverse $\mathbf{X}^\dagger$ is computed as

$$\mathbf{X}^\dagger := \mathbf{V}\mathbf{\Sigma}^\dagger\mathbf{U}^* = \sum_{i}^{r} \mathbf{v}_i \sigma_i^\dagger \mathbf{u}_i^*, \tag{2.2}$$

where $\mathbf{\Sigma}^\dagger$ is here understood as

$$\mathbf{\Sigma}^\dagger = \text{diag}\left(\sigma_i^\dagger\right), \qquad \sigma_i^\dagger = \begin{cases} \sigma_i^{-1} & \text{if } \sigma_i > 0, \\ 0 & \text{otherwise,} \end{cases} \qquad \forall i.$$

We use the Moore-Penrose pseudoinverse in the following to provide a least squares solution to a system of linear equations.

2.3. Dynamic Mode Decomposition. DMD is a dimensionality reduction technique, originally introduced in the field of fluid dynamics [59, 55, 38]. The method extracts spatiotemporal coherent structures from an ordered time series of snapshots $\mathbf{x}_0, \mathbf{x}_1, ..., \mathbf{x}_m \in \mathbb{R}^n$, separated in time by a constant step Δt. Specifically, the aim is to find the eigenvectors and eigenvalues of the time-independent linear operator $\mathbf{A} : \mathbb{R}^n \to \mathbb{R}^n$ that best approximates the map from a given snapshot $\mathbf{x}_j$ to the subsequent snapshot $\mathbf{x}_{j+1}$ as

$$\mathbf{x}_{j+1} \approx \mathbf{A}\mathbf{x}_j. \tag{2.3}$$

Following [68], the deterministic DMD algorithm proceeds by first separating the snapshot sequence $\mathbf{x}_0, \mathbf{x}_1, ..., \mathbf{x}_m$ into two overlapping sets of data

$$\mathbf{X}_L = \begin{bmatrix} | & | & & | \\ \mathbf{x}_0 & \mathbf{x}_1 & \cdots & \mathbf{x}_{m-1} \\ | & | & & | \end{bmatrix}, \quad \mathbf{X}_R = \begin{bmatrix} | & | & & | \\ \mathbf{x}_1 & \mathbf{x}_2 & \cdots & \mathbf{x}_m \\ | & | & & | \end{bmatrix}. \tag{2.4}$$

$\mathbf{X}_L \in \mathbb{R}^{n\times m}$ and $\mathbf{X}_R \in \mathbb{R}^{n\times m}$ are called the left and right snapshot sequences. Equation (2.3) can then be reformulated in matrix notation as

$$\mathbf{X}_R \approx \mathbf{A}\mathbf{X}_L. \tag{2.5}$$

Many variants of the DMD have been proposed since its introduction, as discussed in [38]. Here, we discuss the *exact DMD* formulation of Tu et al. [68].

2.3.1. Non-Projected DMD. In order to find an estimate for the linear map $\mathbf{A}$, the following least-squares problem can be formulated

$$\widehat{\mathbf{A}} = \arg\min_{\mathbf{A}} \|\mathbf{X}_R - \mathbf{A}\mathbf{X}_L\|_F^2. \tag{2.6}$$

The estimator for the best-fit linear map is given in closed form as

$$\widehat{\mathbf{A}} := \mathbf{X}_R\mathbf{X}_L^\dagger, \tag{2.7}$$

where $\mathbf{X}^\dagger$ denotes the pseudoinverse from Eq. (2.2). The DMD modes are the eigenvectors of $\widehat{\mathbf{A}} \in \mathbb{R}^{n\times n}$.

2.3.2. Projected DMD. If the data are high-dimensional (i.e., n is large), the linear map $\widehat{\mathbf{A}}$ in Eq. (2.7) may be intractable to evaluate and analyze directly. Instead, a rank-reduced approximation of the linear map is determined by projecting it onto a k-dimensional subspace, $k \leq \min(n, m)$. We denote $\mathbf{X}_L = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^*$ the singular value decomposition of $\mathbf{X}_L$. The projected map onto the class of rank-k linear operators can be obtained by pre- and postmultiplying Eq. (2.7) with $\mathbf{P}_k \in \mathbb{R}^{n\times k}$ and $\mathbf{T}_k \in \mathbb{R}^{n\times k}$:

$$\widetilde{\mathbf{A}} := \mathbf{P}_k^*\widehat{\mathbf{A}}\mathbf{T}_k = \mathbf{P}_k^*\mathbf{X}_R\mathbf{X}_L^\dagger\mathbf{T}_k = \mathbf{P}_k^*\mathbf{X}_R\mathbf{V}\mathbf{\Sigma}^\dagger\mathbf{U}^*\mathbf{T}_k, \tag{2.8}$$

where $\widetilde{\mathbf{A}} \in \mathbb{R}^{k\times k}$ is the projected map, i.e., a low-dimensional system matrix.

The *projected DMD* [59] relies on the hypothesis that the columns of $\mathbf{A}\mathbf{X}_L$ are in the linear span of $\mathbf{X}_L$. Within this approximation, projection matrices $\mathbf{P}$ and $\mathbf{T}$ are chosen such that they span a subspace of the column space of $\mathbf{X}_L$. A convenient choice is given by the dominant (i.e., associated with the largest singular values) left singular vectors $\mathbf{U}_k$ of $\mathbf{X}_L$ (i.e., POD modes) so that $\mathbf{P}_k = \mathbf{T}_k = \mathbf{U}_k$, $\mathbf{U}_k^*\mathbf{U}_k = \mathbf{I}_k$, with $k \leq \operatorname{rank}(\mathbf{X}_L)$, and

$$\widetilde{\mathbf{A}} = \mathbf{U}_k^*\mathbf{X}_R\mathbf{V}_k\mathbf{\Sigma}_k^{-1}, \tag{2.9}$$

since $\mathbf{U}^*\mathbf{U}_k = \begin{bmatrix}\mathbf{I}_k \\ \mathbf{0}\end{bmatrix}$.

Note that when we use the POD modes $\mathbf{U}$, one can interpret $\widehat{\mathbf{A}}\mathbf{U}_k$ as the shifted spatial POD modes over one time step Δt. Now, by premultiplying $\widehat{\mathbf{A}}\mathbf{U}_k$ with $\mathbf{U}_k^*$, one obtains a projected map $\widetilde{\mathbf{A}} = \mathbf{U}_k^*\widehat{\mathbf{A}}\mathbf{U}_k$ which encodes the cross-correlation of the POD modes $\mathbf{U}_k$ with the time-shifted spatial POD modes. It thus becomes obvious that the DMD encodes more information about the temporal evolution of the system under consideration than the time-averaged POD modes [59].

Low-dimensional projection is a form of spectral filtering which has the positive effect of dampening the influence of noise [32, 26]. This effect is illustrated in Figure 3, which shows the non-projected linear map in absence and presence of white noise. The projected linear map avoids the amplification of noise and acts as a hard-threshold regularizer. The difficulty is to choose the rank providing a good trade-off between suppressing the noise and retaining the useful information. In practice, the optimal hard-threshold [25] provides a good heuristic to determine the rank k.

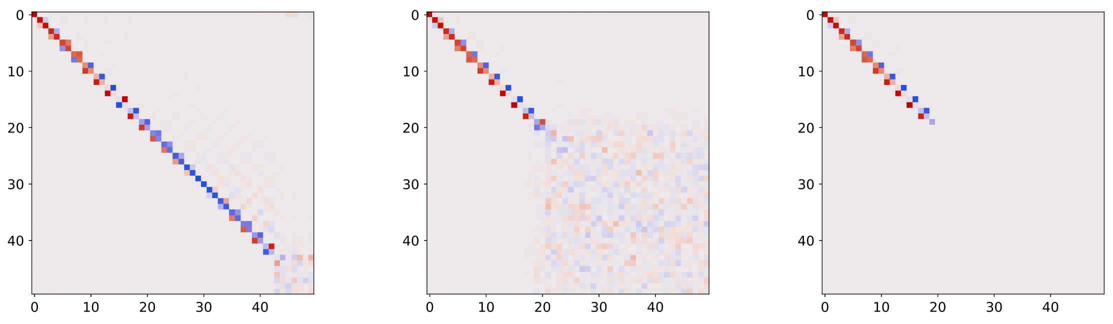
Once the linear map $\widetilde{\mathbf{A}}$ is approximated, its eigendecomposition is computed

$$\widetilde{\mathbf{A}}\widetilde{\mathbf{W}} = \widetilde{\mathbf{W}}\mathbf{\Lambda}, \tag{2.10}$$

where the columns of $\widetilde{\mathbf{W}} \in \mathbb{C}^{k\times k}$ are eigenvectors of $\widetilde{\mathbf{A}}$, and $\mathbf{\Lambda} \in \mathbb{C}^{k\times k}$ is a diagonal matrix containing the corresponding eigenvalues λ_j. Eigenvalues of $\widetilde{\mathbf{A}}$ are eigenvalues of $\widehat{\mathbf{A}}$ and one may recover DMD modes as the columns of $\mathbf{W}_k$, [68]:

$$\mathbf{W}_k := \mathbf{X}_R\mathbf{V}_k\mathbf{\Sigma}_k^{-1}\widetilde{\mathbf{W}}. \tag{2.11}$$

In both the projected and the non-projected formulation, a singular value decomposition of $\mathbf{X}_L$ is involved. This is computationally demanding, and the resources required to compute DMD can be tremendous for large data matrices.



(a) Non-projected linear map in the absence of white noise. (b) Non-projected linear map in the presence of white noise. (c) Projected linear map in the presence of white noise.

Figure 3: Illustration of the non-projected and projected map for toy data in the absence and presence of white noise. The regularization effect of the projected map reduces the influence of noise, while preserving the dominant information.

2.4. A Note on Regularization. The projected algorithm described above relies on the truncated singular value decomposition (TSVD), also referred to as pseudo-inverse filter, to solve the unconstrained least-squares problem in Eq. (2.6). Indeed, Figure 3 illustrates that the low-rank approximation introduces an effective regularization effect. This warrants an extended discussion on regularization, following the work by Hansen [30, 31, 32, 33].

In DMD, we often experience a linear system of equations $\mathbf{X}_R = \mathbf{A}\mathbf{X}_L$ where $\mathbf{X}_L$ or $\mathbf{X}_R$ is ill-conditioned, i.e., the ratio between the largest and smallest singular value is large, so that the pseudo-inverse may magnify small errors. In this situation, regularization becomes crucial for computing an accurate estimate of $\mathbf{A}$, since a small perturbation in $\mathbf{X}_R$ or $\mathbf{X}_L$ may result in a large perturbation in the solution. Tikhonov-Phillips regularization [66, 49], also known as ridge regression in the statistical literature, is one of the most popular regularization techniques. The regularized least squares problem becomes

$$\widehat{\mathbf{A}} = \arg\min_{\mathbf{A}} \|\mathbf{X}_R - \mathbf{A}\mathbf{X}_L\|_F^2 + \lambda^2 \|\mathbf{A}\|_F^2 . \tag{2.12}$$

More concisely, this can be expressed as the following augmented least-squares problem

$$\widehat{\mathbf{A}} = \arg\min_{\mathbf{A}} \left\| \begin{bmatrix} \mathbf{X}_R \\ \mathbf{0} \end{bmatrix} - \mathbf{A} \begin{bmatrix} \mathbf{X}_L \\ \lambda \mathbf{I} \end{bmatrix} \right\|_F^2 . \tag{2.13}$$

The estimator for the map $\mathbf{A}$ takes advantage of the regularized inverse

$$\mathbf{X}_\lambda^\dagger := [\mathbf{X}^*\mathbf{X} + \lambda^2\mathbf{I}]^{-1}\mathbf{X}^* = \mathbf{V}\boldsymbol{\Sigma}_\lambda^+\mathbf{U}^* \quad \text{with} \quad \boldsymbol{\Sigma}_\lambda^+ := \operatorname{diag}\left(\frac{\sigma_1}{\sigma_1^2 + \lambda^2}, ..., \frac{\sigma_r}{\sigma_r^2 + \lambda^2}\right) \tag{2.14}$$

where the additional regularization (controlled via the tuning parameter λ) improves the conditioning of the problem. This form of regularization can also be seen as a smooth filter that attenuates the parts of the solution corresponding to the small singular values. Specifically, the filter f takes the form [30]

$$f_i = \frac{\sigma_i^2}{\sigma_i^2 + \lambda^2}, \qquad i = 1, 2, ..., r. \tag{2.15}$$

The Tikhonov-Phillips regularization scheme is closely related to the TSVD and the Wiener filter. Indeed, the TSVD is known to be a method for regularizing ill-posed linear least-squares problems, which often produces very similar results [69]. More concretely, regularization via the TSVD can be seen as a hard-threshold filter, which takes the form

$$f_i = \begin{cases} 1, & \sigma_i \geq \sigma_k \\ 0, & \sigma_i < \sigma_k \end{cases} . \tag{2.16}$$

Here, k controls how much smoothing (or low-pass filtering) is introduced, i.e., increasing k includes more terms in the SVD expansion; thus components with higher frequencies are included. See [32] for further details. As a consequence of the regularization effect introduced by TSVD, the DMD algorithm requires a careful choice of the target-rank k to compute a meaningful low-rank DMD approximation. Indeed, the solutions (i.e., the computed DMD modes and eigenvalues) may be sensitive to the amount of regularization which is controlled via k. In practice, one can treat the parameter k as a tuning-parameter and find the optimal value via cross-validation.

3. Randomized Methods for Linear Algebra.

With rapidly increasing volumes of measurement data from simulations and experiments, deterministic modal extraction algorithms may become prohibitively expensive. Fortunately, a wide range of applications produce data which feature low-rank structure, i.e., the rank of the data matrix, given by the number of independent columns or rows, is much smaller than the ambient dimension of the data space. In other words, the data feature a large amount of redundant information. In this case, randomized methods allow one to efficiently produce familiar decompositions from linear algebra. These so-called *randomized* numerical methods have the potential to transform computational linear algebra, providing accurate matrix decompositions at a fraction of the cost of deterministic methods. Indeed, over the past two decades, probabilistic algorithms have been prominent for computing low-rank matrix approximations, as described in a number of excellent surveys [29, 44, 16, 8]. The idea is to use randomness as a computational strategy to find a smaller representation, often denoted as *sketch*. This sketch can be used to compute an approximate low-rank factorization for the high-dimensional data matrix $\mathbf{X} \in \mathbb{R}^{n\times m}$.

Broadly, these techniques can be divided into random sampling and random projection-based approaches. The former class of techniques carefully samples and rescales some 'interesting' rows or columns to form a sketch. For instance, we can sample rows by relying on energy-based arguments such as leverage-scores and length-squared samplings [24, 44, 16]. The second class of random projection techniques relies on favorable properties of random matrices and forms the sketch as a randomly weighted linear combination of the columns or rows. Computationally efficient strategies to form such a sketch include the subsampled randomized Hadamard transform [56, 67] and the CountSketch [72]. Choosing between the different sampling and random projection strategies depends on the particular application. In general, sampling is more computational efficient, while random projections preserve more of the information in the data.

3.1. Probabilistic Framework.

One of the most effective off-the-shelf methods to form a sketch is the probabilistic framework proposed in the seminal work by Halko et al. [29]:

- **Stage A:** Given a desired target-rank k, find a near-optimal orthonormal basis $\mathbf{Q} \in \mathbb{R}^{n\times k}$ for the range of the input matrix $\mathbf{X}$.
- **Stage B:** Given the near-optimal basis $\mathbf{Q}$, project the input matrix onto the low-dimensional space, resulting in $\mathbf{B} \in \mathbb{R}^{k\times m}$. This smaller matrix can then be used to compute a near-optimal low-rank approximation.

3.1.1. Stage A: Computing a Near-Optimal Basis.

The first stage is used to approximate the range of the input matrix. Given a target rank $k \ll \min(m, n)$, the aim is to compute a near-optimal basis $\mathbf{Q} \in \mathbb{R}^{n\times k}$ for the input matrix $\mathbf{X} \in \mathbb{R}^{n\times m}$ such that

$$\mathbf{X} \approx \mathbf{Q}\mathbf{Q}^*\mathbf{X}. \tag{3.1}$$

Specifically, the range of the high-dimensional input matrix is sampled using the concept of random projections. Thus a basis is efficiently computed as

$$\mathbf{Y} = \mathbf{X}\mathbf{\Omega}, \tag{3.2}$$

where $\mathbf{\Omega} \in \mathbb{R}^{m\times k}$ denotes a random test matrix drawn from the normal Gaussian distribution. However, the cost of dense matrix multiplications can be prohibitive. The time complexity is order $\mathcal{O}(nmk)$. To improve this scaling, more sophisticated random test matrices, such as the subsampled randomized Hadamard transform, have been proposed [56, 67]. Indeed, the time complexity can be reduced to $\mathcal{O}(nm \cdot \log(k))$ by using a structured random test matrix to sample the range of the input matrix.

The orthonormal basis $\mathbf{Q} \in \mathbb{R}^{n\times k}$ is obtained via the QR-decomposition $\mathbf{Y} = \mathbf{QR}$.

3.1.2. Stage B: Computing a Low-Dimensional Matrix. We now derive a smaller matrix $\mathbf{B}$ from the high-dimensional input matrix $\mathbf{X}$. Specifically, given the near-optimal basis $\mathbf{Q}$, the matrix $\mathbf{X}$ is projected onto the low-dimensional space

$$\mathbf{B} = \mathbf{Q}^*\mathbf{X}, \tag{3.3}$$

which yields the smaller matrix $\mathbf{B} \in \mathbb{R}^{k\times m}$. This process preserves the geometric structure in a Euclidean sense, so that angles between vectors and their lengths are preserved. It follows that

$$\mathbf{X} \approx \mathbf{QB}. \tag{3.4}$$

3.1.3. Oversampling. In theory, if the matrix $\mathbf{X}$ has exact rank k, the sampled matrix $\mathbf{Y}$ spans a basis for the column space, with high probability. In practice, however, it is common that the truncated singular values $\{\sigma_i\}_{i\geq k+1}$ are nonzero. Thus, it helps to construct a slightly larger test matrix in order to obtain an improved basis. Thus, we compute $\mathbf{Y}$ using an $\mathbf{\Omega} \in \mathbb{R}^{m\times l}$ test matrix instead, where $l = k + p$. Thus, the oversampling parameter p denotes the number of additional samples. In most situations small values $p = \{5, 10\}$ are sufficient to obtain a good basis that is comparable to the best possible basis [29].

3.1.4. Power Iteration Scheme. A second strategy to improve the performance is the concept of power iterations [54, 27]. In particular, a slowly decaying singular value spectrum of the input matrix can seriously affect the quality of the approximated basis matrix $\mathbf{Q}$. Thus, the method of power iterations is used to preprocess the input matrix in order to promote a faster decaying spectrum. The sampling matrix $\mathbf{Y}$ is obtained as

$$\mathbf{Y} = \big((\mathbf{XX}^*)^q\mathbf{X}\big)\mathbf{\Omega} \tag{3.5}$$

where q is an integer specifying the number of power iterations. With $\mathbf{X} = \mathbf{U\Sigma V}^*$, one has $\mathbf{X}^{(q)} := (\mathbf{XX}^*)^q\mathbf{X} = \mathbf{U\Sigma}^{2q+1}\mathbf{V}^*$. Hence, for $q > 0$, the preprocessed matrix $\mathbf{X}^{(q)}$ has a relatively fast decay of singular values compared to the input matrix $\mathbf{X}$. The drawback of this method is that additional passes over the input matrix are required. However, as few as $q = \{1, 2\}$ power iterations can considerably improve the approximation quality, even when the singular values of the input matrix decay slowly.

Algorithm 3.1 describes this overall procedure for the randomized QB decomposition.

Remark 3.1. A direct implementation of the power iteration scheme, as outlined in Section 3.1.4, is numerically unstable due to round-off errors. Instead, the sampling matrix $\mathbf{Y}$ is orthogonalized between each computational step to improve the stability [29].

Remark 3.2. As default values for the oversampling and power iteration parameter, we suggest $p = 10$, and $q = 2$.

Algorithm 3.1 Randomized QB decomposition.
Given a matrix $\mathbf{X} \in \mathbb{R}^{n\times m}$ and a target rank $k \ll \min(m,n)$, the basis matrix $\mathbf{Q} \in \mathbb{R}^{n\times k}$ with orthonormal columns and the smaller matrix $\mathbf{B} \in \mathbb{R}^{k\times m}$ are computed. The approximation quality can be controlled via oversampling p and the computation of q power iterations.

```
(1)   $l = k + p$                                   Slight oversampling.
(2)   $\mathbf{\Omega} = \mathtt{rand}(m, l)$        Generate random test matrix.
(3)   $\mathbf{Y} = \mathbf{X\Omega}$                Compute sampling matrix.
(4)   for $j = 1, \ldots, q$                        Power iterations (optional).
(5)       $[\mathbf{Q}, \sim] = \mathtt{qr}(\mathbf{Y})$
(6)       $[\mathbf{Z}, \sim] = \mathtt{qr}(\mathbf{X}^*\mathbf{Q})$
(7)       $\mathbf{Y} = \mathbf{XZ}$
(8)   end for
(9)   $[\mathbf{Q}, \sim] = \mathtt{qr}(\mathbf{Y})$  Orthonormalize sampling matrix.
(10)  $\mathbf{B} = \mathbf{Q}^*\mathbf{X}$          Project input matrix to smaller space.
```

3.1.5. Theoretical Performance. Both the concept of oversampling and the power iteration scheme provide control over the quality of the low-rank approximation. The average-case error behavior is described in the probabilistic framework as [46]:

$$\mathbb{E}_{\mu_\mathbf{Q}} \|\mathbf{X} - \mathbf{QQ}^*\mathbf{X}\|_F \leq \left[1 + \sqrt{\frac{k}{p-1}} + \frac{e\sqrt{l}}{p} \cdot \sqrt{\min(m,n) - k}\right]^{\frac{1}{2q+1}} \sigma_{k+1}(\mathbf{X}),$$

with $e \equiv \exp(1)$, $\mathbb{E}_{\mu_\mathbf{Q}}$ the expectation operator over the probability measure of $\mathbf{Q}$ and $\sigma_{k+1}(\mathbf{X})$ the $(k+1)$-th element of the decreasing sequence of singular values of $\mathbf{X}$. Here it is assumed that $p \geq 2$. Thus, both oversampling and the computation of additional power iterations drive the approximation error down.

3.1.6. Computational Considerations. The steps involved in computing the approximate basis $\mathbf{Q}$ and the low-dimensional matrix $\mathbf{B}$ are simple to implement, and embarrassingly parallelizable. Thus, randomized algorithms can benefit from modern computational architectures, and they are particularly suitable for GPU-accelerated computing. Another favorable computational aspect is that only two passes over the input matrix are required in order to obtain the low-dimensional matrix. Pass efficiency is a crucial aspect when dealing with massive data matrices which are too large to fit into fast memory, since reading data from the hard disk is prohibitively slow and often constitutes the actual bottleneck.

3.2. Blocked Randomized Algorithm.

When dealing with massive fluid flows that are too large to read into fast memory, the extension to sequential, distributed, and parallel computing might be inevitable [57]. In particular, it might be necessary to distribute the data across processors which have no access to a shared memory to exchange information. To address this limitation, Martinsson and Voronin [47] proposed a blocked scheme to compute the QB decomposition on smaller blocks of the data. The basic idea is that a given high-dimensional sequence of snapshots $\mathbf{x}_0, \mathbf{x}_1, ..., \mathbf{x}_m \in \mathbb{R}^n$ is subdivided into b smaller blocks along the rows.

The submatrices can then be processed in b independent streams of calculations. Here, b is assumed to be a power of two, and zero padding can be used in order to divide the data into blocks of the same size. This scheme constructs the smaller matrix $\mathbf{B}$ in a hierarchical fashion, for more details see [70].

In the following, we describe a modified and simple scheme that is well suited for practical applications such as computing the dynamic mode decomposition for large-scale fluid flows. Unlike [47, 70], which discuss a fixed-precision approximation, we focus on a fixed-rank approximation problem. Further, our motivation is that it is often unnecessary to use the full hierarchical scheme if the desired target rank k is relatively small. By small we mean that we can fit a matrix of dimension $(b \cdot k) \times m$ into fast memory. To be more precise, suppose again that we are given an input matrix $\mathbf{X}$ with n rows and m columns. We partition $\mathbf{X}$ along the rows into b blocks $\mathbf{X}_i$ of dimension $n/b \times m$. To illustrate the scheme we set $b = 4$ so that

$$\mathbf{X} = \begin{bmatrix} \mathbf{X}_1 \\ \mathbf{X}_2 \\ \mathbf{X}_3 \\ \mathbf{X}_4 \end{bmatrix}. \tag{3.6}$$

Next, we approximate each block $\mathbf{X}_i$ by a fix rank-k approximation, using the QB decomposition described in Algorithm 3.1, which yields

$$\mathbf{X} \approx \begin{bmatrix} \mathbf{Q}_1\mathbf{B}_1 \\ \mathbf{Q}_2\mathbf{B}_2 \\ \mathbf{Q}_3\mathbf{B}_3 \\ \mathbf{Q}_4\mathbf{B}_4 \end{bmatrix} = \operatorname{diag}\left(\mathbf{Q}_1, \mathbf{Q}_2, \mathbf{Q}_3, \mathbf{Q}_4\right) \begin{bmatrix} \mathbf{B}_1 \\ \mathbf{B}_2 \\ \mathbf{B}_3 \\ \mathbf{B}_4 \end{bmatrix}. \tag{3.7}$$

We can then collect all matrices $\mathbf{B}_i \in \mathbb{R}^{k \times m}$ and stack them together as

$$\mathbf{K} = \begin{bmatrix} \mathbf{B}_1^* & \mathbf{B}_2^* & \mathbf{B}_3^* & \mathbf{B}_4^* \end{bmatrix}^*. \tag{3.8}$$

Subsequently, we compute the QB decomposition of $\mathbf{K} \in \mathbb{R}^{(b \cdot k) \times m}$ and obtain

$$\mathbf{K} \approx \widehat{\mathbf{Q}}\mathbf{B}. \tag{3.9}$$

The small matrix $\mathbf{B} \in \mathbb{R}^{k \times m}$ can then be used to compute the randomized dynamic mode decomposition as described below. The basis matrix $\mathbf{Q} \in \mathbb{R}^{n \times k}$ can be formed as

$$\mathbf{Q} = \operatorname{diag}\left(\mathbf{Q}_1, \mathbf{Q}_2, \mathbf{Q}_3, \mathbf{Q}_4\right) \widehat{\mathbf{Q}}. \tag{3.10}$$

In practice, we choose the target-rank l for the approximation slightly larger than k, i.e., $l = k + p$.

4. Randomized Dynamic Mode Decomposition.

In many cases, even with increased measurement resolution, the data may have dominant coherent structures that define a low-dimensional attractor [35]; in fact, the presence of these structures was the original motivation for methods such as DMD. Hence, it seems natural to use randomized methods for linear algebra to accelerate the computation of the approximate low-rank DMD.

4.1. Problem Formulation Revisited. We start our discussion by formulating the following least-squares problem to find an estimate $\widehat{\mathbf{A}}_{\mathbf{B}}$ for the projected linear map $\widetilde{\mathbf{A}} \in \mathbb{R}^{l\times l}$ in terms of the projected snapshot sequences $\mathbf{B}_L := \mathbf{P}^*\mathbf{X}_L \in \mathbb{R}^{l\times m}$ and $\mathbf{B}_R := \mathbf{P}^*\mathbf{X}_R \in \mathbb{R}^{l\times m}$:

$$\widehat{\mathbf{A}}_{\mathbf{B}} := \arg\min_{\mathbf{A}_{\mathbf{B}}} \|\mathbf{B}_R - \mathbf{A}_{\mathbf{B}}\mathbf{B}_L\|_F^2, \tag{4.1}$$

where $\mathbf{P} \in \mathbb{R}^{n\times l}$ is a projection matrix discussed below. Recall that $l = k + p$, where k denotes the desired target rank, and p is the oversampling parameter. Here we make the assumption that $\mathrm{col}(\mathbf{P}) \approx \mathrm{col}(\mathbf{X}_L)$ as well as $\mathrm{col}(\mathbf{P}) \approx \mathrm{col}(\mathbf{X}_R)$.

The question remains how to construct $\mathbf{P}$ in order to quickly compute $\mathbf{B}_L$ and $\mathbf{B}_R$. The dominant left singular vectors of $\mathbf{X}_R$ provide a good choice; however, the computational demands to compute the deterministic SVD may be prohibitive for large data sets. Next, we discuss randomized methods to compute DMD.

4.2. Compressed Dynamic Mode Decomposition. Brunton et al. [10] proposed one of the first algorithms using the idea of matrix sketching to find small matrices $\mathbf{B}_L$ and $\mathbf{B}_R$. The algorithm proceeds by forming a small number of random linear combinations of the rows of the left and right snapshot matrices to form the representation

$$\mathbf{B}_L = \mathbf{S}\mathbf{X}_L, \quad \mathbf{B}_R = \mathbf{S}\mathbf{X}_R. \tag{4.2}$$

with $\mathbf{S} \in \mathbb{R}^{l\times n}$ a random test matrix. As discussed in Section 3, $\mathbf{S}$ can be constructed by drawing its entries from the standard normal distribution. While using a Gaussian random test matrix has beneficial theoretical properties, the dense matrix multiplication $\mathbf{S}\mathbf{X}_L$ can be expensive for large dense matrices. Alternatively, $\mathbf{S}$ can be chosen to be *a random and rescaled subset* of the identity matrix, i.e., $\mathbf{B}$ is formed by sampling and rescaling l rows from $\mathbf{X}$ with probability p_i. Thus, $\mathbf{S}$ is very sparse and is not required to be explicitly constructed and stored. More concretely, we sample the ith row of $\mathbf{X}$ with probability p_i and, if sampled, the row is rescaled by the factor $1/\sqrt{l\cdot p_i}$ in order to yield an unbiased estimator [17]. A naive approach is to sample rows with uniform probability $p_i = 1/n$. Uniform random sampling may work if the information is evenly distributed across the input data, so that dominant structures are not spatially localized. Indeed, it was demonstrated in [10] that the dominant DMD modes and eigenvalues for some low-rank fluid flows can be obtained from a massively undersampled sketch. However, the variance can be large and the performance may be poor in the presence of white noise. While not explored by [10], the performance can be improved using more sophisticated sampling techniques, such as leverage-score sampling [15, 43, 17]. Better performance is expected when using structured random test matrices such as the subsampled randomized Hadamard transform [56, 67] or the CountSketch [72]. Both of these methods enable efficient matrix multiplications, yet they are more computationally demanding than random sampling.

The shortcoming of the algorithm in [10] is that l depends on the ambient dimension of the measurement space of the input matrix. Thus, even if the data matrix is low rank, a large l may be required to compute the approximate DMD. Next, we discuss a randomized algorithm which depends on the intrinsic rank of the input matrix.

4.3. An Improved Randomized Scheme. In the following, we present a novel algorithm to compute the dynamic mode decomposition using the probabilistic framework above. The proposed algorithm is simple to implement and can fully benefit from modern computational architectures, e.g., parallelization and multithreading.

Given a sequence of snapshots $\mathbf{x}_0, \mathbf{x}_1, ..., \mathbf{x}_m \in \mathbb{R}^n$, we first compute the near-optimal basis $\mathbf{Q} \in \mathbb{R}^{n\times l}$ using the randomized methods which we have discussed in Section 3. Then, we project the data onto the low-dimensional space, so that we obtain the low-dimensional sequence of snapshots

$$\mathbf{b}_0, \mathbf{b}_1, ..., \mathbf{b}_m := \mathbf{Q}^*\mathbf{x}_0, \mathbf{Q}^*\mathbf{x}_1, ..., \mathbf{Q}^*\mathbf{x}_m \in \mathbb{R}^l.$$

More concisely, we can express this as

$$\mathbf{B} := \mathbf{Q}^*\mathbf{X}.$$

Next, we separate the sequence $\mathbf{b}_0, \mathbf{b}_1, ..., \mathbf{b}_m$ into two overlapping matrices $\mathbf{B}_L \in \mathbb{R}^{l\times m}$ and $\mathbf{B}_R \in \mathbb{R}^{l\times m}$

$$\mathbf{B}_L = \begin{bmatrix} | & | & & | \\ \mathbf{b}_0 & \mathbf{b}_1 & \cdots & \mathbf{b}_{m-1} \\ | & | & & | \end{bmatrix}, \quad \mathbf{B}_R = \begin{bmatrix} | & | & & | \\ \mathbf{b}_1 & \mathbf{b}_2 & \cdots & \mathbf{b}_m \\ | & | & & | \end{bmatrix}. \tag{4.3}$$

This leads to the following projected least-squares problem

$$\widehat{\mathbf{A}}_\mathbf{B} = \arg\min_{\mathbf{A}_\mathbf{B}} \|\mathbf{B}_R - \mathbf{A}_\mathbf{B}\mathbf{B}_L\|_F^2. \tag{4.4}$$

Using the pseudoinverse, the estimator for the linear map $\widehat{\mathbf{A}}_\mathbf{B} \in \mathbb{R}^{l\times l}$ is defined as

$$\widehat{\mathbf{A}}_\mathbf{B} := \mathbf{B}_R\mathbf{B}_L^\dagger = \mathbf{B}_R\mathbf{V}\mathbf{\Sigma}^{-1}\widetilde{\mathbf{U}}^*, \tag{4.5}$$

where $\widetilde{\mathbf{U}} \in \mathbb{R}^{l\times k}$ and $\mathbf{V} \in \mathbb{R}^{m\times k}$ are the truncated left and right singular vectors of $\mathbf{B}_L$. The diagonal matrix $\mathbf{\Sigma} \in \mathbb{R}^{k\times k}$ contains the corresponding singular values. If $p = 0$, then $l = k$, and no truncation is needed. Next, $\widehat{\mathbf{A}}_\mathbf{B}$ is projected onto the left singular vectors

$$\widetilde{\mathbf{A}}_\mathbf{B} = \widetilde{\mathbf{U}}^*\widehat{\mathbf{A}}_\mathbf{B}\widetilde{\mathbf{U}} \tag{4.6a}$$

$$= \widetilde{\mathbf{U}}^*\mathbf{B}_R\mathbf{V}\mathbf{\Sigma}^{-1}. \tag{4.6b}$$

The DMD modes, containing the spatial information, are then obtained by computing the eigendecomposition of $\widetilde{\mathbf{A}}_\mathbf{B} \in \mathbb{R}^{k\times k}$

$$\widetilde{\mathbf{A}}_\mathbf{B}\widetilde{\mathbf{W}}_\mathbf{B} = \widetilde{\mathbf{W}}_\mathbf{B}\mathbf{\Lambda}, \tag{4.7}$$

where the columns of $\widetilde{\mathbf{W}}_\mathbf{B} \in \mathbb{C}^{k\times k}$ are eigenvectors $\widetilde{\mathbf{w}}_{\mathbf{B},j}$, and $\mathbf{\Lambda} \in \mathbb{C}^{k\times k}$ is a diagonal matrix containing the corresponding eigenvalues λ_j. The high-dimensional DMD modes $\mathbf{W} \in \mathbb{C}^{n\times k}$ may be recovered as

$$\mathbf{W} = \mathbf{Q}\mathbf{B}_R\mathbf{V}\mathbf{\Sigma}^{-1}\widetilde{\mathbf{W}}_\mathbf{B}. \tag{4.8}$$

The computational steps for a practical implementation are sketched in Algorithm 4.1.

4.3.1. Derivation. In the following we outline the derivation of Eq. (4.8). Recalling Eq. (2.7), the estimated transfer operator $\widehat{\mathbf{A}}$ is defined as

$$\widehat{\mathbf{A}} := \mathbf{X}_R \mathbf{X}_L^\dagger. \tag{4.9}$$

Random projections of the data matrix $\mathbf{X}$ result in an approximate orthonormal basis $\mathbf{Q}$ for the range of $\widehat{\mathbf{A}}$, as described in Sec. 3. The sampling strategy is assumed to be efficient enough so that $\mathbf{X}_L \approx \mathbf{Q}\mathbf{Q}^*\mathbf{X}_L$ and $\mathbf{X}_R \approx \mathbf{Q}\mathbf{Q}^*\mathbf{X}_R$. Equation (4.9) then becomes

$$\widehat{\mathbf{A}} \approx (\mathbf{Q}\mathbf{Q}^*\mathbf{X}_R)\,(\mathbf{Q}\mathbf{Q}^*\mathbf{X}_L)^\dagger\,. \tag{4.10}$$

Letting $\mathbf{B}_L := \mathbf{Q}^*\mathbf{X}_L$ and $\mathbf{B}_R := \mathbf{Q}^*\mathbf{X}_R$, the projected transfer operator estimate is defined as in Eq. (4.5), $\widehat{\mathbf{A}}_\mathbf{B} := \mathbf{B}_R\mathbf{B}_L^\dagger$. Substituting this into Eq. (4.10) leads to

$$\widehat{\mathbf{A}} \approx \mathbf{Q}\widehat{\mathbf{A}}_\mathbf{B}\mathbf{Q}^*. \tag{4.11}$$

Letting $\mathbf{B}_L = \widetilde{\mathbf{U}}\mathbf{\Sigma}\mathbf{V}^*$ be the SVD of $\mathbf{B}_L$, and left- and right-projecting $\widehat{\mathbf{A}}_\mathbf{B}$ onto the dominant k left singular vectors $\widetilde{\mathbf{U}}$, yields

$$\widetilde{\mathbf{A}}_\mathbf{B} := \widetilde{\mathbf{U}}^*\widehat{\mathbf{A}}_\mathbf{B}\widetilde{\mathbf{U}}. \tag{4.12}$$

The eigendecomposition is given by $\widetilde{\mathbf{A}}_\mathbf{B}\widetilde{\mathbf{W}}_\mathbf{B} = \widetilde{\mathbf{W}}_\mathbf{B}\mathbf{\Lambda}$, as in Eq. (4.7).

Let $\widehat{\mathbf{W}}_\mathbf{B} := \mathbf{B}_R\,\mathbf{V}\mathbf{\Sigma}^{-1}\widetilde{\mathbf{W}}_\mathbf{B}$. It is simple to verify that this is an eigenvector of $\widehat{\mathbf{A}}_\mathbf{B}$:

$$\begin{aligned}
\widehat{\mathbf{A}}_\mathbf{B}\widehat{\mathbf{W}}_\mathbf{B} &= \mathbf{B}_R\,\mathbf{B}_L^\dagger\mathbf{B}_R\,\mathbf{V}\mathbf{\Sigma}^{-1}\widetilde{\mathbf{W}}_\mathbf{B},\\
&= \mathbf{B}_R\,\mathbf{V}\mathbf{\Sigma}^{-1}\widetilde{\mathbf{A}}_\mathbf{B}\widetilde{\mathbf{W}}_\mathbf{B},\\
&= \mathbf{B}_R\,\mathbf{V}\mathbf{\Sigma}^{-1}\widetilde{\mathbf{W}}_\mathbf{B}\mathbf{\Lambda},\\
&= \widehat{\mathbf{W}}_\mathbf{B}\mathbf{\Lambda}.
\end{aligned}$$

Substituting the eigendecomposition of $\widehat{\mathbf{A}}_\mathbf{B}$ in Eq. (4.11) and right-multiplying by $\mathbf{Q}\widehat{\mathbf{W}}_\mathbf{B}$ leads to

$$\widehat{\mathbf{A}}\mathbf{Q}\widehat{\mathbf{W}}_\mathbf{B} \approx \mathbf{Q}\widehat{\mathbf{W}}_\mathbf{B}\mathbf{\Lambda}, \tag{4.13}$$

Algorithm 4.1 Randomized Dynamic Mode Decomposition (rDMD).
Given a snapshot matrix $\mathbf{X}$ and a target rank k, the near-optimal dominant dynamic modes $\mathbf{W}$ and eigenvalues $\mathbf{\Lambda}$ are computed. The approximation quality can be controlled via oversampling p and the computation of power iterations q. An implementation in Python is available via the GIT repository https://github.com/erichson/ristretto.

```
(1)   $[\mathbf{Q}, \mathbf{B}] = \texttt{rqb}(\mathbf{X}, p, q)$                                  Randomized QB decomposition (Alg. 3.1).
(2)   $\mathbf{B} \to \{\mathbf{B}_L, \mathbf{B}_R\}$                                              Left/right low-dimensional snapshot matrix.
(3)   $[\widetilde{\mathbf{U}}, \mathbf{\Sigma}, \mathbf{V}] = \texttt{svd}(\mathbf{B}_L, k)$      Truncated SVD.
(4)   $\widetilde{\mathbf{A}}_\mathbf{B} = \widetilde{\mathbf{U}}^*\mathbf{B}_R\mathbf{V}\mathbf{\Sigma}^{-1}$      Least squares fit.
(5)   $[\widetilde{\mathbf{W}}_\mathbf{B}, \mathbf{\Lambda}] = \texttt{eig}(\widetilde{\mathbf{A}}_\mathbf{B})$    Eigenvalue decomposition.
(6)   $\mathbf{W} \leftarrow \mathbf{Q}\mathbf{B}_R\mathbf{V}\mathbf{\Sigma}^{-1}\widetilde{\mathbf{W}}_\mathbf{B}$    Recover high-dimensional DMD modes $\mathbf{W}$.
```

so that identification with $\widehat{\mathbf{A}}\mathbf{W} = \mathbf{W}\widetilde{\mathbf{\Lambda}}$ verifies the claim in Eq. (4.8) on the eigendecomposition of $\widehat{\mathbf{A}}$:

$$\mathbf{W} \approx \mathbf{Q}\mathbf{B}_R\mathbf{V}\mathbf{\Sigma}^{-1}\widetilde{\mathbf{W}}_{\mathbf{B}}, \qquad \widetilde{\mathbf{\Lambda}} \approx \mathbf{\Lambda}.$$

5. Numerical Results.

In the following we present numerical results demonstrating the performance of the randomized dynamic mode decomposition (rDMD). First, we provide some visual results for a flow behind a cylinder and climate data. Then, we evaluate and compare the computational time and accuracy of randomized algorithm to the deterministic algorithm.

All computations are performed using Amazon Web Services (AWS). We use a G3 instance with 16 Xeon E5-2686 CPUs and 1 NVIDIA Tesla M60 GPU. The underlying numerical linear algebra routines are accelerated using the Intel Math Kernel Library (MKL).

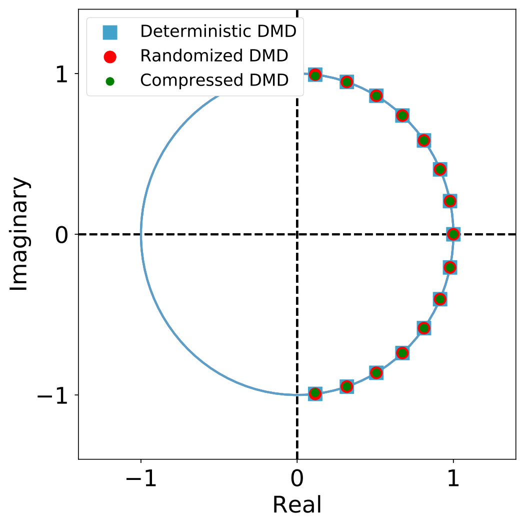
5.1. Fluid Flow Behind a Cylinder.

As a canonical example, we consider the fluid flow behind a cylinder at Reynolds number $Re = 100$. The data consist of a sequence of 151 snapshots of fluid vorticity fields on a 449×199 grid[1], computed using the immersed boundary projection method [63, 12]. The flow features a periodically shedding wake structure and the resulting dataset is low rank. While this data set poses no computational challenge, it demonstrates the accuracy and quality of the randomized approximation on an example that builds intuition. Flattening and concatenating the snapshots horizontally yields a matrix of dimension $\mathbf{X} \in \mathbb{R}^{89{,}351 \times 151}$, i.e., the columns are the flattened snapshots $\mathbf{x} \in \mathbb{R}^{449 \times 199}$.

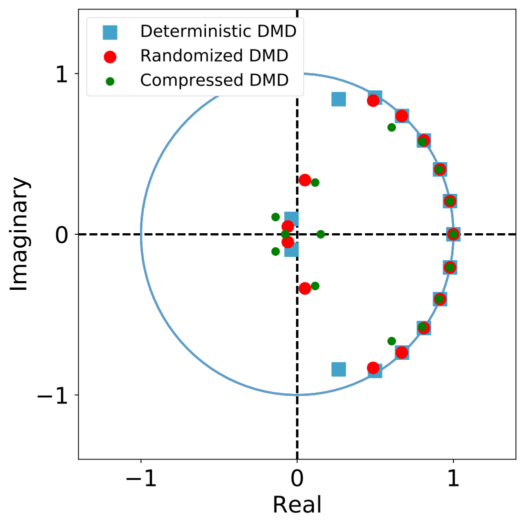
We compute the low-rank DMD approximation using $k = 15$ as the desired target rank. Figure 4a shows the DMD eigenvalues. The proposed randomized DMD algorithm (with $p = 10$ and $q = 0$), and the compressed DMD algorithm (with $l = 1000$) faithfully capture the eigenvalues. Overall, the randomized algorithm leads to a 6 fold speedup compared to the deterministic DMD algorithm. Further, if the singular value spectrum is slowly decaying, the approximation accuracy can be improved by computing additional power iterations q. To further contextualize the results, Fig. 5 shows the leading six DMD modes in absence of noise. The randomized algorithm faithfully reveals the coherent structures, while requiring considerably fewer computational resources.

Next, the analysis is repeated in presence of additive white noise with a signal-to-noise ratio (SNR) of 10. Figure 4b shows distinct performance of the different algorithms. The deterministic algorithm performs most accurately, capturing the first eleven eigenvalues. The randomized DMD algorithm reliably captures the first nine eigenvalues, while the compressed algorithm only accurately captures seven of the eigenvalues. This is, despite the fact that the compressed DMD algorithm uses a large sketched snapshot sequence of dimension 1000×151, i.e., $1{,}000$ randomly selected rows out of the $89{,}351$ rows of the flow data set are used. For comparison the randomized DMD algorithm provides a more satisfactory approximation using $l = 25$ and 2 additional power iterations, i.e., the sketched snapshot sequence is only of dimension 25×151. The results show that the randomized DMD algorithm yields a more accurate approximation, while allowing for higher compression rates. This is because the approximation quality of the randomized DMD algorithm depends on the intrinsic rank of the data and not on the ambient dimensions on the measurement space.

[1]Data for this example may be downloaded at dmdbook.com/DATA.zip.

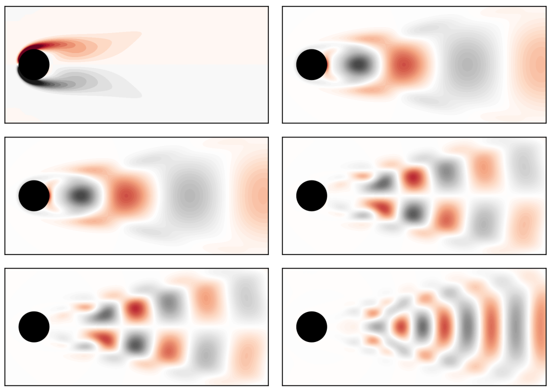


(a) No noise.

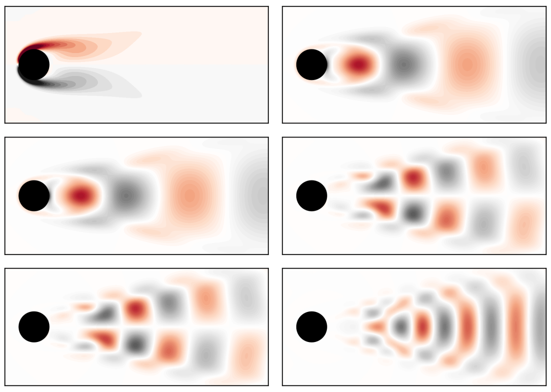


(b) Noisy with SNR of 10.

Figure 4: DMD eigenvalues for fluid flow behind a cylinder. Both the compressed and randomized DMD algorithms capture the eigenvalues in the absence of noise (a). In the presence of white noise with SNR of 10, rDMD performs better than sampling rows (b).



(a) Deterministic DMD modes.



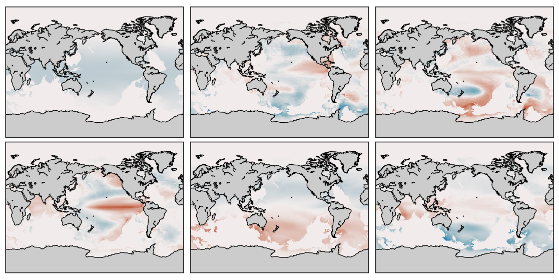
(b) Randomized DMD modes.

Figure 5: Leading dynamic modes extracted from the fluid flow behind a cylinder.

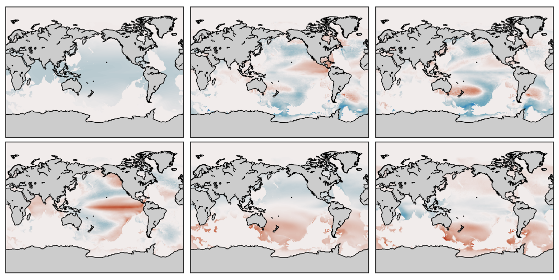
5.2. Sea Surface Data. We now compute the dynamic mode decomposition on the high-resolution sea surface temperature (SST) data. The SST data are widely studied in climate science for climate monitoring and prediction, providing an improved understanding of the interactions between the ocean and the atmosphere [53, 52, 61]. Specifically, the daily SST measurements are constructed by combining infrared satellite data with observations provided by ships and buoys. In order to account and compensate for platform differences and sensor errors, a bias-adjusted methodology is used to combine the measurements from the different sources. Finally, the spatially complete SST map is produced via interpolation. A comprehensive discussion of the data is provided in [52].

The data are provided by the National Oceanic and Atmospheric Administration (NOAA) via their web site at https://www.esrl.noaa.gov/psd/. Data are available for the years from 1981 to 2018 with a temporal resolution of 1 day and a spatial grid resolution of 0.25°. In total, the data consist of $m = 13,149$ temporal snapshots which measure the daily temperature at $1440 \times 720 = 1,036,800$ spatial grid points. Since we omit data over land, the ambient dimension reduces to $n = 691,150$ spatial measurements in our analysis. Concatenating the reduced data yield a 36GB data matrix of dimension $\mathbf{X} \in \mathbb{R}^{691,150\times 13,149}$, which is sufficiently large to test scaling.

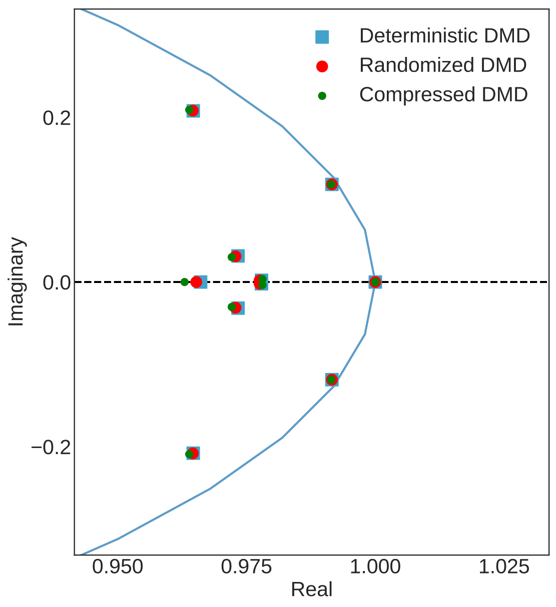
5.2.1. Aggregated (Weekly Mean) Data. The full data set outstrips the available fast memory required to compute the deterministic DMD. Thus, we perform the analysis on an aggregated data set first in order to compare the randomized and deterministic DMD algorithms. Therefore, we compute the weekly mean temperature, which reduces the number



(a) Deterministic DMD modes.



(b) Blocked Randomized DMD modes.



(c) DMD eigenvalues.

Figure 6: Leading dynamic modes extracted from the aggregated high-resolution sea surface dataset. No distinct differences can be obtained. Further, the randomized DMD algorithm captures faithfully the eigenvalues. In contrast, the compressed DMD algorithm provides only a crude approximation for the eigenvalues.

of temporal snapshots to $m = 1,878$. Figure 6c shows the corresponding eigenvalues. Unlike the eigenvalues obtained via the compressed DMD algorithm, the randomized DMD algorithm provides an accurate approximation for the dominant eigenvalues. Next, Fig. 6a and 6b show the extracted DMD modes. Indeed, the randomized modes faithfully capture the dominant coherent structure in the data.

5.2.2. Full Data. The blocked randomized DMD algorithm allows us to extract the DMD modes from the high-dimensional data set. While the full data set does not fit into fast memory, it is only required that we can access some of the rows in a sequential manner. Computing the $k = 15$ approximation using $b = 4$ blocks takes about 150 seconds. The resulting modes are shown in Fig. 7. Here, the leading modes are similar to the modes extracted from the aggregated data set. However, subsequent modes may provide further insights which are not revealed by the aggregated data set.

5.3. Computational Performance.

Next, we evaluate the computational performance of the randomized algorithms in terms of both time and accuracy. We measure the accuracy by computing the relative error of the randomized DMD compared to the approximation produced by the deterministic DMD algorithm

$$\rho(\widehat{\mathbf{X}}_{(\mathrm{DMD})}, \widehat{\mathbf{X}}_{(\mathrm{rDMD})}) = \frac{\|\widehat{\mathbf{X}}_{(\mathrm{DMD})} - \widehat{\mathbf{X}}_{(\mathrm{rDMD})}\|_F}{\|\widehat{\mathbf{X}}_{(\mathrm{DMD})}\|_F}, \tag{5.1}$$

where $\widehat{\mathbf{X}}$ denotes the reconstructed snapshot matrix using either the deterministic or randomized DMD algorithm. Here, we use a (high-performance) partial SVD algorithm to compute the k deterministic modes [41].

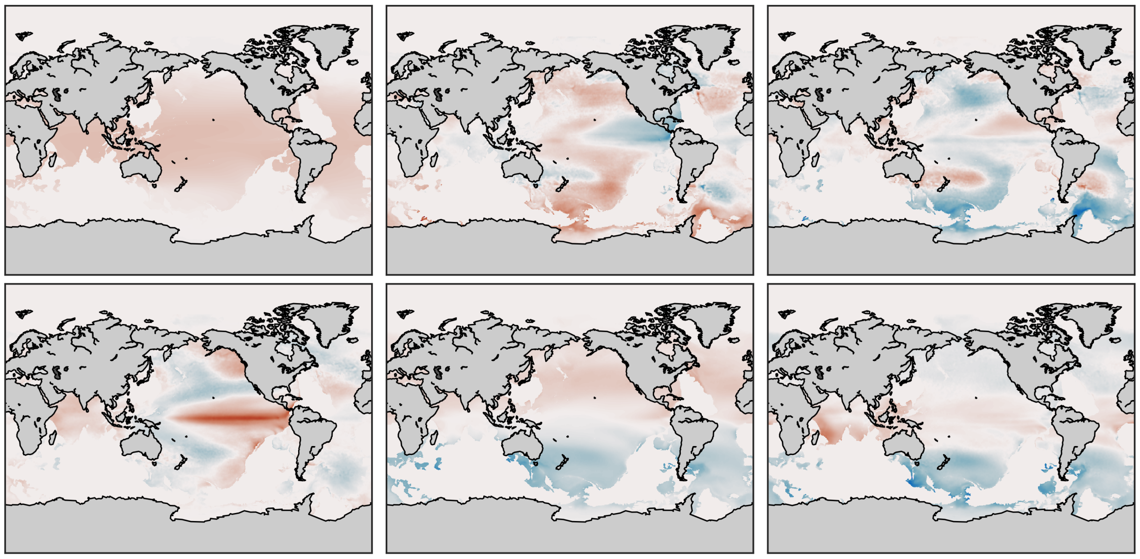
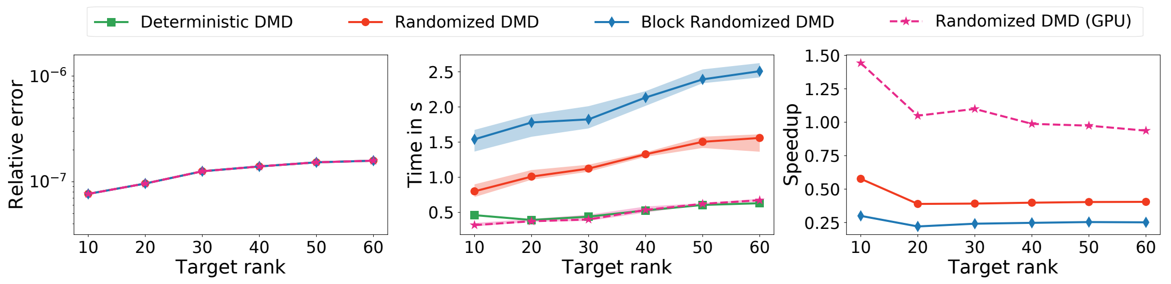
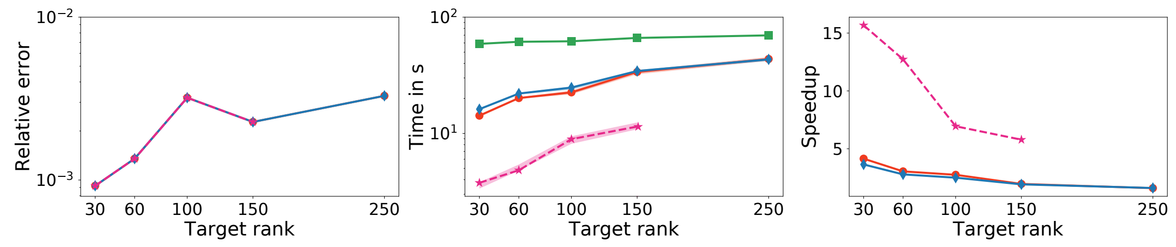


Figure 7: Leading dynamic modes of the full high-resolution SST dataset.

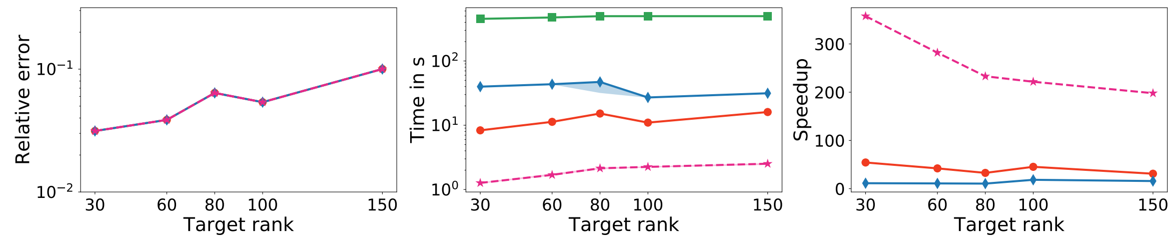
Figure 8 summarizes the computational performance, for three different examples, and for varying target ranks. First, Figure 8 (a) shows the performance for the flow past a cylinder. The snapshot matrix is tall and skinny. In this setting there is no computational advantage of



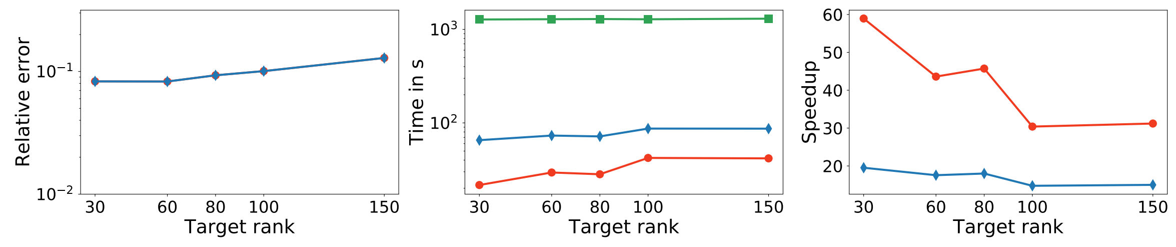
(a) Flow past cylinder (matrix of dimension $89,351 \times 151$).



(b) Aggregated sea surface temperature (matrix of dimension $691,150 \times 1,878$).



(c) Turbulent flow (matrix of dimension $65,536 \times 15,000$).



(d) Turbulent flow (matrix of dimension $131,072 \times 20,000$).

Figure 8: Average runtimes and errors, over 20 runs, for varying target ranks. Further, the gained speedup compared to the deterministic algorithm (baseline) is shown.

the randomized algorithm over the deterministic algorithm. This is because the data matrix is overall very small and the (deterministic) partial SVD is highly efficient for tall and skinny matrices. Importantly, we stress that the randomized DMD algorithm provides a very accurate approximation for the flow past cylinder. That is, because this fluid flow example has a fast decaying singular value spectrum.

Second, Figure 8 (b) shows the performance for the aggregated SST data. This snapshot matrix is also tall and skinny, yet its dimensions are substantially larger than those of the previous example. Here, we start to see the computational benefits of the randomized DMD algorithm. We gain about a speedup factor of 3-5 for computing the low-rank DMD, while maintaining a good accuracy. Note that this problem has a more slowly decaying singular value spectrum and thus the accuracy is poorer than in the previous examples. This example also demonstrates the performance boost by using a GPU accelerated implementation of the randomized DMD. Clearly, we can see a significant speedup by a factor of about 10-15 for computing the top 30 to 60 dynamic modes. Note that we run out of memory for target ranks larger than $k > 150$.

Third, Figure 8 (c) and (d) shows the performance for a turbulent flow data set. The fluid flow was obtained with the model implementation of [11], which is based on the algorithm presented by [71]. This data set shows the advantages and disadvantages of the randomized algorithm. Indeed, we see some substantial gains in terms of the computational time, i.e., the GPU-accelerated algorithms achieve speedups of factors > 300. However, the accuracy is less satisfactory due to the slowly decaying singular value spectrum of the data. This example illustrates the limits of the randomized algorithm, i.e., we need to assume that the data set features low-rank structure and has a reasonably fast decaying singular value spectrum.

The reader might have noticed that the blocked randomized scheme (using 2 blocks) requires more computational time to compute the approximate DMD. This is because the data matrices considered in the above examples fit into the fast memory. Hence, there is

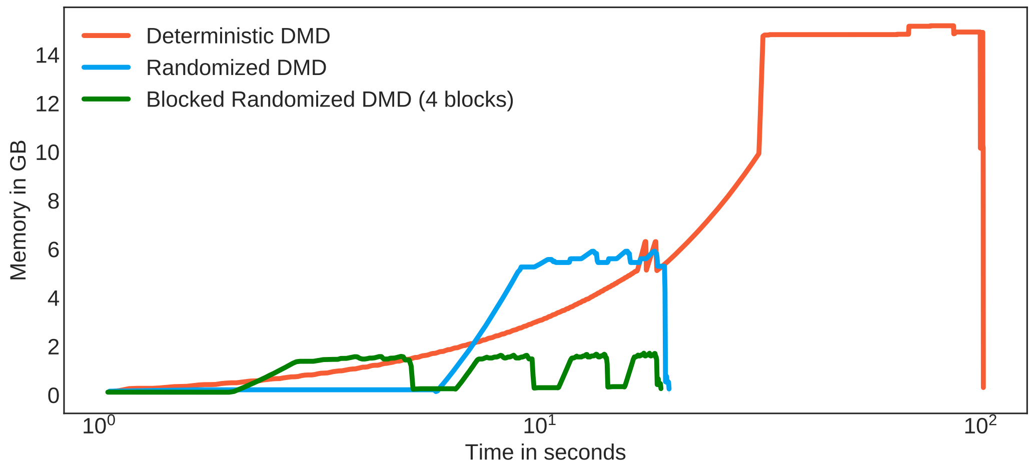


Figure 9: A profile of the memory usage vs runtime of the different DMD algorithms.

little advantage in using the blocked scheme here. However, the memory requirements are often more important than the absolute computational time. Figure 9 shows a profile of the memory usage vs runtime. The blocked randomized DMD algorithm requires only a fraction of the memory, compared to the deterministic algorithm, to compute the approximate low-rank dynamic mode decomposition. This can be crucial when carrying out the computations on mobile platforms, on GPUs (as seen in Figure 8 (c)), or when scaling the computations to very large applications.

6. Conclusion. Randomness as a computational strategy has recently been shown capable of efficiently solving many standard problems in linear algebra. The need for highly efficient algorithms becomes increasingly important in the area of 'big data'. Here, we have proposed a novel randomized algorithm for computing the low-rank dynamic mode decomposition. Specifically, we have shown that DMD can be embedded in the probabilistic framework formulated by [29]. This framework not only enables computations at scales far larger than what was previously possible, but it is also modular, flexible, and the error can be controlled. Hence, it can be also utilized as a framework to embed other innovations around the dynamic mode decomposition, for instance, see [38] for an overview.

The numerical results show that randomized dynamic mode decomposition (rDMD) has computational advantages over previously suggested probabilistic algorithms for computing the dominant dynamic modes and eigenvalues. More importantly, we showed that the algorithm can be executed using a blocked scheme which is memory efficient. This aspect is crucial in order to efficiently deal with massive data which are too big to fit into fast memory. Thus, we believe that the randomized DMD framework will provide a powerful and scalable architecture for extracting dominant spatiotemporal coherent structures and dynamics from increasingly large-scale data, for example from epidemiology, neuroscience, and fluid mechanics.

Acknowledgments. We would like to express our gratitude to the two anonymous reviewers. Their helpful feedback allowed us to greatly improve the manuscript. Further, we thank Diana A. Bistrian for many useful comments. NBE would like to acknowledge the generous funding support from the Defense Advanced Research Projects Agency (DARPA) and the Air Force Research Laboratory (FA8750-17-2-0122) as well as Amazon Web Services for supporting the project with EC2 credits. LM acknowledges the support of the French Agence Nationale pour la Recherche (ANR) and Direction Générale de l'Armement (DGA) via the *FlowCon* project (ANR-17-ASTR-0022). SLB and JNK would like to acknowledge generous funding support from the Army Research Office (ARO W911NF-17-1-0118), Air Force Office of Scientific Research (AFOSR FA9550-16-1-0650), and DARPA (PA-18-01-FP-125).

REFERENCES

[1] H. Arbabi and I. Mezić, *Ergodic theory, dynamic mode decomposition and computation of spectral properties of the koopman operator*, SIAM J. Appl. Dyn. Syst., 16 (2017), pp. 2096–2126.
[2] J. Basley, L. R. Pastur, N. Delprat, and F. Lusseyran, *Space-time aspects of a three-dimensional multi-modulated open cavity flow*, Physics of Fluids (1994-present), 25 (2013), p. 064105.
[3] E. Berger, M. Sastuba, D. Vogt, B. Jung, and H. B. Amor, *Estimation of perturbations in robotic behavior using dynamic mode decomposition*, Journal of Advanced Robotics, 29 (2015), pp. 331–343.

[4] G. Berkooz, P. Holmes, and J. L. Lumley, *The proper orthogonal decomposition in the analysis of turbulent flows*, Annual Review of Fluid Mechanics, 23 (1993), pp. 539–575.

[5] D. Bistrian and I. Navon, *Randomized dynamic mode decomposition for non-intrusive reduced order modelling*, International Journal for Numerical Methods in Engineering, (2016), pp. 1–22, https://doi.org/10.1002/nme.5499.

[6] D. Bistrian and I. Navon, *Efficiency of randomised dynamic mode decomposition for reduced order modelling*, International Journal of Computational Fluid Dynamics, 32 (2018), pp. 88–103.

[7] B. W. Brunton, L. A. Johnson, J. G. Ojemann, and J. N. Kutz, *Extracting spatial–temporal coherent patterns in large-scale neural recordings using dynamic mode decomposition*, Journal of Neuroscience Methods, 258 (2016), pp. 1–15.

[8] S. L. Brunton and J. N. Kutz, *Data-Driven Science and Engineering: Machine Learning, Dynamical Systems, and Control*, Cambridge University Press, 2018.

[9] S. L. Brunton and B. R. Noack, *Closed-loop turbulence control: Progress and challenges*, Applied Mechanics Reviews, 67 (2015), pp. 050801–1–050801–48.

[10] S. L. Brunton, J. L. Proctor, J. H. Tu, and J. N. Kutz, *Compressed sensing and dynamic mode decomposition*, Journal of Computational Dynamics, 2 (2015), pp. 165–191.

[11] D. B. Chirila, *Towards Lattice Boltzmann Models for Climate Sciences: The GeLB Programming Language with Applications*, PhD thesis, University of Bremen, 2018, https://elib.suub.uni-bremen.de/peid/D00106468.html.

[12] T. Colonius and K. Taira, *A fast immersed boundary method using a nullspace approach and multi-domain far-field boundary conditions*, Comp. Meth. App. Mech. Eng., 197 (2008), pp. 2131–2146.

[13] S. Das and D. Giannakis, *Delay-coordinate maps and the spectra of koopman operators*, arXiv preprint arXiv:1706.08544, (2017).

[14] S. T. Dawson, M. S. Hemati, M. O. Williams, and C. W. Rowley, *Characterizing and correcting for the effect of sensor noise in the dynamic mode decomposition*, Experiments in Fluids, 57 (2016), pp. 1–19.

[15] P. Drineas, M. Magdon-Ismail, M. W. Mahoney, and D. P. Woodruff, *Fast approximation of matrix coherence and statistical leverage*, Journal of Machine Learning Research, 13 (2012), pp. 3475–3506.

[16] P. Drineas and M. W. Mahoney, *Randnla: Randomized numerical linear algebra*, Communications of the ACM, 59 (2016), pp. 80–90, https://doi.org/10.1145/2842602.

[17] P. Drineas and M. W. Mahoney, *Lectures on randomized numerical linear algebra*, arXiv preprint arXiv:1712.08880, (2017).

[18] Z. Drmač and S. Gugercin, *A new selection operator for the discrete empirical interpolation method—improved a priori error bound and extensions*, SIAM J. Sci. Comput., 38 (2016), pp. A631–A648.

[19] D. Duke, J. Soria, and D. Honnery, *An error analysis of the dynamic mode decomposition*, Experiments in fluids, 52 (2012), pp. 529–542.

[20] N. B. Erichson, S. L. Brunton, and J. N. Kutz, *Compressed dynamic mode decomposition for background modeling*, Journal of Real-Time Image Processing, (2016), pp. 1–14, https://doi.org/10.1007/s11554-016-0655-2.

[21] N. B. Erichson, S. L. Brunton, and J. N. Kutz, *Compressed dynamic mode decomposition for real-time object detection*, Journal of Real-Time Image Processing, (2016).

[22] N. B. Erichson and C. Donovan, *Randomized low-rank dynamic mode decomposition for motion detection*, Computer Vision and Image Understanding, 146 (2016), pp. 40–50, https://doi.org/10.1016/j.cviu.2016.02.005.

[23] J. E. Fowler, *Compressive-projection principal component analysis*, IEEE Transactions on Image Processing, 18 (2009), pp. 2230–2242, https://doi.org/10.1109/TIP.2009.2025089.

[24] A. Frieze, R. Kannan, and S. Vempala, *Fast Monte-Carlo algorithms for finding low-rank approximations*, Journal of the ACM, 51 (2004), pp. 1025–1041.

[25] M. Gavish and D. L. Donoho, *The optimal hard threshold for singular values is* $4\sqrt{3}$, IEEE Transactions on Information Theory, 60 (2014), pp. 5040–5053.

[26] I. F. Gorodnitsky and B. D. Rao, *Analysis of error produced by truncated SVD and Tikhonov regularization methods*, in Proceedings of 1994 28th Asilomar Conference on Signals, Systems and Computers, IEEE, 1994, pp. 25–29.

[27] M. Gu, *Subspace iteration randomization and singular value problems*, SIAM Journal on Scientific

Computing, 37 (2015), pp. A1139–A1173, https://doi.org/10.1137/130938700.

[28] F. Guéniat, L. Mathelin, and L. Pastur, *A dynamic mode decomposition approach for large and arbitrarily sampled systems*, Physics of Fluids, 27 (2015), p. 025113.

[29] N. Halko, P. G. Martinsson, and J. A. Tropp, *Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions*, SIAM Review, 53 (2011), pp. 217–288, https://doi.org/10.1137/090771806.

[30] P. C. Hansen, *The truncatedsvd as a method for regularization*, BIT Numerical Mathematics, 27 (1987), pp. 534–553.

[31] P. C. Hansen, *Truncated singular value decomposition solutions to discrete ill-posed problems with ill-determined numerical rank*, SIAM Journal on Scientific and Statistical Computing, 11 (1990), pp. 503–518.

[32] P. C. Hansen, J. G. Nagy, and D. P. O'leary, *Deblurring images: matrices, spectra, and filtering*, vol. 3, Siam, 2006.

[33] P. C. Hansen, T. Sekii, and H. Shibahashi, *The modified truncated svd method for regularization in general form*, SIAM Journal on Scientific and Statistical Computing, 13 (1992), pp. 1142–1150.

[34] M. S. Hemati, C. W. Rowley, E. A. Deem, and L. N. Cattafesta, *De-biasing the dynamic mode decomposition for applied Koopman spectral analysis*, Theoretical and Computational Fluid Dynamics, 31 (2017), pp. 349–368.

[35] P. J. Holmes, J. L. Lumley, G. Berkooz, and C. W. Rowley, *Turbulence, coherent structures, dynamical systems and symmetry*, Cambridge Monographs in Mechanics, Cambridge University Press, Cambridge, England, 2nd ed., 2012.

[36] W. B. Johnson and J. Lindenstrauss, *Extensions of Lipschitz mappings into a Hilbert space*, Contemporary mathematics, 26 (1984), pp. 189–206, https://doi.org/10.1090/conm/026/737400.

[37] B. Kramer, P. Grover, P. Boufounos, M. Benosman, and S. Nabi, *Sparse sensing and dmd based identification of flow regimes and bifurcations in complex flows*, arXiv preprint arXiv:1510.02831, (2015).

[38] J. N. Kutz, S. L. Brunton, B. W. Brunton, and J. L. Proctor, *Dynamic Mode Decomposition: Data-Driven Modeling of Complex Systems*, SIAM, 2016.

[39] J. N. Kutz, X. Fu, and S. L. Brunton, *Multi-resolution dynamic mode decomposition*, SIAM Journal on Applied Dynamical Systems, 15 (2016), pp. 713–735. Preprint. Available: arXiv:1506.00564.

[40] J. N. Kutz, X. Fu, S. L. Brunton, and N. B. Erichson, *Multi-resolution dynamic mode decomposition for foreground/background separation and object tracking*, (2015), pp. 921–929.

[41] R. Lehoucq, D. Sorensen, and C. Yang, *Arpack users' guide: Solution of large scale eigenvalue problems with implicitly restarted arnoldi methods.*, Software Environ. Tools, 6 (1997).

[42] F. Lusseyran, F. Gueniat, J. Basley, C. L. Douay, L. R. Pastur, T. M. Faure, and P. J. Schmid, *Flow coherent structures and frequency signature: application of the dynamic modes decomposition to open cavity flow*, in Journal of Physics: Conference Series, vol. 318, IOP Publishing, 2011, p. 042036.

[43] P. Ma, M. W. Mahoney, and B. Yu, *A statistical perspective on algorithmic leveraging*, The Journal of Machine Learning Research, 16 (2015), pp. 861–911.

[44] M. W. Mahoney, *Randomized algorithms for matrices and data*, Foundations and Trends in Machine Learning, 3 (2011), pp. 123–224, https://doi.org/10.1561/2200000035.

[45] K. Manohar, B. W. Brunton, J. N. Kutz, and S. L. Brunton, *Data-driven sparse sensor placement*, IEEE Control Systems Magazine, 38 (2018), pp. 63–86.

[46] P.-G. Martinsson, *Randomized methods for matrix computations*, arXiv preprint arXiv:1607.01649, (2016).

[47] P.-G. Martinsson and S. Voronin, *A randomized blocked algorithm for efficiently computing rank-revealing factorizations of matrices*, SIAM Journal on Scientific Computing, 38 (2016), pp. S485–S507.

[48] B. R. Noack, W. Stankiewicz, M. Morzynski, and P. J. Schmid, *Recursive dynamic mode decomposition of a transient cylinder wake*, Journal of Fluid Mechanics, 809 (2016), pp. 843–872.

[49] D. L. Phillips, *A technique for the numerical solution of certain integral equations of the first kind*, Journal of the ACM (JACM), 9 (1962), pp. 84–97.

[50] J. L. Proctor, S. L. Brunton, and J. N. Kutz, *Dynamic mode decomposition with control*, SIAM Journal on Applied Dynamical Systems, 15 (2016), pp. 142–161.

[51] J. L. Proctor and P. A. Eckhoff, *Discovering dynamic patterns from infectious disease data using*

dynamic mode decomposition, International health, 7 (2015), pp. 139–145.

[52] R. W. Reynolds, N. A. Rayner, T. M. Smith, D. C. Stokes, and W. Wang, *An improved in situ and satellite SST analysis for climate*, Journal of climate, 15 (2002), pp. 1609–1625.

[53] R. W. Reynolds, T. M. Smith, C. Liu, D. B. Chelton, K. S. Casey, and M. G. Schlax, *Daily high-resolution-blended analyses for sea surface temperature*, Journal of Climate, 20 (2007), pp. 5473–5496.

[54] V. Rokhlin, A. Szlam, and M. Tygert, *A randomized algorithm for principal component analysis*, SIAM Journal on Matrix Analysis and Applications, 31 (2009), pp. 1100–1124.

[55] C. W. Rowley, I. Mezić, S. Bagheri, P. Schlatter, and D. Henningson, *Spectral analysis of nonlinear flows*, J. Fluid Mech., 645 (2009), pp. 115–127.

[56] T. Sarlos, *Improved approximation algorithms for large matrices via random projections*, in Foundations of Computer Science. 47th Annual IEEE Symposium on, 2006, pp. 143–152.

[57] T. Sayadi and P. J. Schmid, *Parallel data-driven decomposition algorithm for large-scale datasets: with application to transitional boundary layers*, Theoretical and Computational Fluid Dynamics, (2016), pp. 1–14.

[58] P. Schmid, L. Li, M. Juniper, and O. Pust, *Applications of the dynamic mode decomposition*, Theoretical and Computational Fluid Dynamics, 25 (2011), pp. 249–259.

[59] P. J. Schmid, *Dynamic mode decomposition of numerical and experimental data*, Journal of Fluid Mechanics, 656 (2010), pp. 5–28.

[60] A. Seena and H. J. Sung, *Dynamic mode decomposition of turbulent cavity flows for self-sustained oscillations*, International Journal of Heat and Fluid Flow, 32 (2011), pp. 1098–1110.

[61] T. M. Smith and R. W. Reynolds, *A global merged land–air–sea surface temperature reconstruction based on historical observations (1880–1997)*, Journal of Climate, 18 (2005), pp. 2021–2036.

[62] K. Taira, S. L. Brunton, S. Dawson, C. W. Rowley, T. Colonius, B. J. McKeon, O. T. Schmidt, S. Gordeyev, V. Theofilis, and L. S. Ukeiley, *Modal analysis of fluid flows: An overview*, AIAA Journal, 55 (2017), pp. 4013–4041.

[63] K. Taira and T. Colonius, *The immersed boundary method: A projection approach*, J. Comp. Phys., 225 (2007), pp. 2118–2137.

[64] N. Takeishi, Y. Kawahara, Y. Tabei, and T. Yairi, *Bayesian dynamic mode decomposition*, Twenty-Sixth International Joint Conference on Artificial Intelligence, (2017).

[65] R. Taylor, J. N. Kutz, K. Morgan, and B. Nelson, *Dynamic mode decomposition for plasma diagnostics and validation*, arXiv preprint arXiv:1702.06871, (2017).

[66] A. N. Tikhonov, *Solution of incorrectly formulated problems and the regularization method*, Soviet Math., 4 (1963), pp. 1035–1038.

[67] J. A. Tropp, *Improved analysis of the subsampled randomized hadamard transform*, Advances in Adaptive Data Analysis, 3 (2011), pp. 115–126.

[68] 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, 1 (2014), pp. 391–421.

[69] J. Varah, *A practical examination of some numerical methods for linear discrete ill-posed problems*, Siam Review, 21 (1979), pp. 100–111.

[70] S. Voronin and P.-G. Martinsson, *RSVDPACK: Subroutines for computing partial singular value decompositions via randomized sampling on single core, multi core, and GPU architectures*, 2015, https://arxiv.org/abs/1502.05366.

[71] J. Wang, D. Wang, P. Lallemand, and L.-S. S. Luo, *Lattice Boltzmann simulations of thermal convective flows in two dimensions*, Computers and Mathematics with Applications, 65 (2013), pp. 262–286, https://doi.org/10.1016/j.camwa.2012.07.001.

[72] D. P. Woodruff et al., *Sketching as a tool for numerical linear algebra*, Foundations and Trends in Theoretical Computer Science, 10 (2014), pp. 1–157.