APPLICATIONES MATHEMATICAE
50,1 (2023), pp. 55–65

G. H. Nedzhibov (Shumen)

DYNAMIC MODE DECOMPOSITION: AN ALTERNATIVE ALGORITHM FOR FULL-RANK DATASETS

Abstract. Dynamic mode decomposition (DMD) is a modal decomposition technique that describes high-dimensional dynamic data using coupled spatial-temporal modes. It combines the main features of performing principal component analysis (PCA) in space, and power spectral analysis in time. The method is equation-free in the sense that it does not require knowledge of the underlying governing equations and is entirely data-driven. The purpose of this paper is to introduce a new algorithm for computing the dynamic mode decomposition in the case of full rank data. The new approach is more economical from a computational point of view, which is an advantage when working with large datasets.

1. Introduction. The dynamic mode decomposition algorithm (*DMD method*) has been established as a leading technique for identifying spatiotemporal coherent structures from high-dimensional data. It can be considered to be a numerical approximation to Koopman spectral analysis, and in this sense it is applicable to nonlinear dynamical systems (see [R09, M05]). In recent years the popularity of the DMD method has grown, and it has been applied for a variety of dynamical systems in many different fields, such as video processing [GK14], epidemiology [PE15], neuroscience [BJOK16], financial trading [MK16, CL16, KG$^+$17], robotics [BS$^+$15], cavity flows [S10, SS11] and various jets [R09, S11]. For a review of the DMD literature, we refer the reader to [TR$^+$14, KB$^+$16, BK$^+$20]; see also [M13, CTR12, B13].

In the following, we begin by presenting a framework for the standard DMD method. We then introduce and analyze an efficient algorithm, an al-

2020 *Mathematics Subject Classification*: Primary 65P99; Secondary 37M10.
Key words and phrases: algorithms for dynamic mode decomposition, singular value decomposition, Koopman operator, equation-free.
Received 1 October 2022; revised 8 December 2022.
Published online 8 May 2023.

DOI: 10.4064/am2465-4-2023

ternative to the standard algorithm, in the case of a full-rank dataset. We give theoretical results proving that the DMD modes obtained with the new algorithm are the exact DMD modes of the corresponding Koopman operator. We analyze the complexity of the algorithm introduced compared to that of the standard DMD algorithm. We show that the new algorithm is more efficient as it requires fewer computational resources: it requires fewer matrix multiplications as well as a smaller amount of memory when calculating the corresponding DMD matrix. As a result, it has better speed and more economical memory usage. Then we discuss two numerical experiments that illustrate the proposed algorithm and provide a comparison between it and the standard algorithm.

The remainder of this work is organized as follows: in Section 2 we describe the DMD method. Section 3 proposes and discusses a novel approach to DMD computation. A comparative analysis of the computational efficiency of the proposed algorithm is discussed in Section 4. Section 5 contains examples demonstrating the new algorithm. The conclusion is in Section 6. Throughout the paper, we use the following notations: uppercase (Latin or Greek) letters for matrices, lowercase bold letters for vectors, and lowercase letters for scalars.

2. DMD method. Here we briefly consider the DMD method, which was introduced for the first time by Schmid [SS08] in fluid mechanics. The standard definition of DMD takes into account a sequential set of data $Z = \{\mathbf{z}_0, \ldots, \mathbf{z}_m\}$, where each $\mathbf{z}_k$ is in $\mathbb{R}^n$. The data $\mathbf{z}_i$ could be from measurements, experiments, or simulations collected from a given nonlinear system at time t_i. Assume that the data are evenly spaced in time, with a time step of $\triangle t$, and that the collection time begins at t_0 and ends at t_m. The method uses the arrangement of the dataset into two large data matrices

$$X = [\mathbf{z}_0, \ldots, \mathbf{z}_{m-1}] \quad \text{and} \quad Y = [\mathbf{z}_1, \ldots, \mathbf{z}_m]. \tag{2.1}$$

The method's main assumption is that there exists a linear (unknown) operator A that relates $\mathbf{z}_k$ to the subsequent $\mathbf{z}_{k+1}$,

$$\mathbf{z}_{k+1} = A\mathbf{z}_k. \tag{2.2}$$

Expression (2.2) is equivalent to

$$Y = AX. \tag{2.3}$$

Then the dynamic mode decomposition of the data matrix Z is given by the eigendecomposition of A. The *DMD modes* and *eigenvalues* are intended to approximate the eigenvectors and eigenvalues of A.

To approximate the operator A, one approach is to use the singular value decomposition (SVD) of the data matrix $X = U\Sigma V^*$ and the expression

$$A \approx YX^{\dagger} = YV\Sigma^{-1}U^*, \tag{2.4}$$

where $X^\dagger$ is the Moore–Penrose pseudoinverse of X. It should be noted that calculating the eigendecomposition of the $n \times n$ matrix A can be prohibitively expensive if n is large, i.e. $n \gg m$. The algorithm below computes DMD modes without directly calculating A by using reduced SVD of $X = U\Sigma V^*$, where U is $n \times r$, S is $r \times r$ diagonal, V is $m \times r$, and $r \leq m$.

Algorithm 1: Exact DMD Algorithm [TR$^+$14]

---

1. Compute the SVD of $X = U\Sigma V^*$ and substitute into (2.4):
   $A = YV\Sigma^{-1}U^*$.
2. Define the reduced-order matrix $\tilde{A} = U^*AU = U^*YV\Sigma^{-1}$.
3. Compute the eigendecomposition of $\tilde{A}$,
   $\tilde{A}W = W\Lambda$,
   where W is the eigenvector matrix and Λ is the diagonal matrix of eigenvalues, $\Lambda = \operatorname{diag}\{\lambda_i\}$. Each λ_i is a DMD eigenvalue.
4. Compute the DMD modes
   $\Phi = YV\Sigma^{-1}W$.
   Each column ϕ_i of Φ is a DMD mode corresponding to λ_i.

The approximate dynamics of the dataset Z can then be reconstructed as

$$\mathbf{z}_{\mathrm{DMD}}(k) = \Phi\Lambda^k\mathbf{b}\,, \tag{2.5}$$

where $\mathbf{b} = \Phi^\dagger\mathbf{z}_0$, and $\Phi^\dagger$ is the Moore–Penrose pseudoinverse of Φ.

In its original form [SS08], the algorithm of the DMD method differs slightly from the one described above. The only difference is that the DMD modes (at Step 4) are computed by the formula

$$\Phi = UW. \tag{2.6}$$

The DMD modes calculated by Algorithm 1 are frequently referred to as *exact DMD modes* because Tu et al. [TR$^+$14] proved that these are exact eigenvectors of the matrix A. The modes in (2.6) are referred to as *projected DMD modes*.

An alternative DMD algorithm. We recently published [N22] a new algorithm for computing the DMD decomposition of A. The new algorithm follows the same steps as in Algorithm 1. The distinctive feature is the following: we use the eigendecomposition of the matrix

$$\hat{A} = \Sigma^{-1}U^*YV, \tag{2.7}$$

rather than of the matrix $\tilde{A}$ in Algorithm 1, Step 2. Because they are similar with a transformation matrix Σ, the eigenvalues of the two matrices $\tilde{A}$ and $\hat{A}$ are the same. We use the formula

$$\Phi = YV\hat{W} \tag{2.8}$$

for the DMD modes, where $\hat{W}$ is the eigenvector matrix of $\hat{A}$, i.e.

$$\hat{A}\hat{W} = \hat{W}\Lambda. \tag{2.9}$$

It is seen that although the matrices $\tilde{A}$ and $\hat{A}$ have a similar representation, the expression (2.8) for calculating the DMD modes is simpler than the corresponding formula in Algorithm 1, Step 4. Therefore, the algorithm introduced in [N22] is more efficient in terms of computational cost.

3. A new DMD algorithm for full-rank datasets. Consider two data matrices X and Y defined by (2.1) with arbitrary but the same dimensions $n \times m$, where $n > m$. As mentioned in the previous section, the DMD method involves approximating the eigendecomposition of the best fit linear operator A, which refers to

$$Y = AX.$$

In fact, the algorithms of the DMD method allow the calculation of DMD modes and eigenvalues without direct calculation of A. The SVD of the matrix X is used to obtain a reduced-order approximation of the matrix A, such as

$$\tilde{A} = U^*AU = U^*YV\Sigma^{-1} \tag{3.1}$$

defined in Algorithm 1, or

$$\hat{A} = \Sigma^{-1}U^*AU\Sigma, \tag{3.2}$$

defined by (2.7) in equivalent form. The matrices $\tilde{A}$ and $\hat{A}$ have the same order r, and the number of non-zero singular values of X within numerical precision is $r \leq m$.

In the case that the matrix X has full rank, i.e. $\operatorname{rank}(X) = m$, both $\tilde{A}$ and $\hat{A}$ will be $m \times m$ matrices.

We will assume that X is a full-rank matrix for the rest of this paper. Our goal is to introduce a more computationally efficient algorithm for calculating DMD modes and eigenvalues in this particular case.

Let us consider the matrix

$$\bar{A} = V\hat{A}V^*, \tag{3.3}$$

where V is the $m \times m$ unitary matrix from the SVD of $X = U\Sigma V^*$. The matrices $\hat{A}$ and $\bar{A}$ are obviously unitarily similar, with V^* being the similarity transformation matrix.

From (2.7) and (3.3), we get

$$\bar{A} = V\Sigma^{-1}U^*Y, \tag{3.4}$$

which yields

$$\bar{A} = X^{\dagger}Y, \tag{3.5}$$

where $X^{\dagger}$ is the Moore–Penrose pseudoinverse of X.

Let

$$\bar{A}\bar{W} = \bar{W}\Lambda \tag{3.6}$$

represent the eigendecomposition of $\bar{A}$, with $\bar{W}$ columns representing eigenvectors and Λ a diagonal matrix containing the corresponding eigenvalues.

Relations (3.2), (3.3) and (3.6) yield

$$AU\Sigma V^*\bar{W} = U\Sigma V^*\bar{W}\Lambda \tag{3.7}$$

or equivalently

$$A(X\bar{W}) = (X\bar{W})\Lambda. \tag{3.8}$$

Thus, we showed that

$$\Phi = X\bar{W} \tag{3.9}$$

is the matrix of DMD modes.

The following theorem holds.

THEOREM 3.1. *Let* $(\lambda, \mathbf{w})$ *represent an eigenpair of* $\bar{A}$ *defined by* (3.5)*, with* $\lambda \neq 0$*. Then* (λ, φ) *is the corresponding eigenpair of* A*, where*

$$\varphi = Y\mathbf{w}.$$

Proof. Let us use (2.4) to express $A\varphi$:

$$A\varphi = YV\Sigma^{-1}U^*Y\mathbf{w}.$$

We get

$$A\varphi = Y\bar{A}\mathbf{w}$$

from the previous relation and (3.4), which implies that by using (3.6),

$$A\varphi = \lambda Y\mathbf{w} = \lambda\varphi.$$

Furthermore, $\varphi \neq 0$, because if $Y\mathbf{w} = 0$, then $V\Sigma^{-1}U^*Y\mathbf{w} = \bar{A}\mathbf{w} = 0$, which implies $\lambda = 0$. As a result, φ is an eigenvector of A with eigenvalue λ. The theorem is proved. ∎

Next, we resume the results above in the following algorithm.

Algorithm 2: DMD Algorithm for full-rank dataset

---

1. Define $\bar{A} = X^\dagger Y$, where $X^\dagger$ is the pseudoinverse of X.
2. Compute the eigendecomposition of $\bar{A}$
   $\bar{A}\bar{W} = \bar{W}\Lambda$, where $\bar{W}$ is the eigenvector matrix and Λ is the diagonal matrix of eigenvalues $\Lambda = \mathrm{diag}\{\lambda_i\}$. Each λ_i is a DMD eigenvalue.
3. Compute the DMD modes
   $\Phi = Y\bar{W}$.
   Each column ϕ_i of Φ is a DMD mode corresponding to λ_i.

According to Theorem 3.1, the modes generated by Algorithm 2 are the eigenvectors of the Koopman operator A.

Application of the new algorithm. DMD theory initially focused on full-rank, sequential time series with $n \gg m$. Since the method was introduced, it has found applications in many different fields. This presumes, in practice, that in some cases the inverse relationship is also valid, namely the number of measurements m taken in time may be greater than the dimension of spatial measurements n, i.e. $n < m$.

The following options are available:

- When the high-dimensional dynamics of the data have some underlying low-dimensional structure, it may be possible to capture the key dynamics of the data with relatively few DMD modes. In this case, the rank of the dataset is equal to the number of DMD modes.
- In some datasets, there are linear dependencies among the measurements (snapshots), i.e. the rank of the dataset is too low, and the DMD fails to fully capture the dynamics of the system. The solution of this rank mismatch is by rearranging the dataset in modified (augmented) data matrices inspired by the Hankel matrix constructed in the eigenvalue realization algorithm (ERA); see [TR$^+$14].

Algorithm 2 has the advantage of being more cost-effective than standard DMD algorithms when used on a full-range dataset. This type of data occurs in various fields, including neuroscience [BJOK16], finance [MK16, CL16] and others [PE15, BS$^+$15].

4. Computational cost and memory requirement. Table 1 gives a brief summary of the main matrices in the two algorithms considered. The representations of the corresponding reduced-order approximations of the Koopman operator are shown, as are the formulas for calculating the DMD modes in the two cases.

Table 1. Reduced matrices ($\tilde{A}$ and $\bar{A}$) and DMD modes (Φ and $\bar{\Phi}$)

	Algorithm 1	Algorithm 2
Reduced matrix	$\tilde{A} = U^* Y V \Sigma^{-1}$	$\bar{A} = X^\dagger Y$
DMD modes	$\Phi = Y V \Sigma^{-1} W$	$\bar{\Phi} = Y \bar{W}$

The reduced-order approximation matrices $\tilde{A}$ and $\bar{A}$ have similar structures, especially when the SVD of X is used to compute the pseudoinverse matrix $X^\dagger$ (in Algorithm 2). However, as shown in Table 1, the matrix $\bar{\Phi}$ has a simpler structure than Φ. Three matrices are required in Algorithm 2 to be

stored and three matrix multiplications performed, while in Algorithm 2, it is necessary to store only two matrices and perform one matrix multiplication.

To estimate the computational cost for the two algorithms considered, we will ignore the comparable computations and focus on the different ones. We can assume that in Algorithm 1, the SVD of X is equivalent to computing the pseudoinverse matrix of X in Algorithm 2. The computational costs for the reduced matrices and DMD modes for the two algorithms are shown in Table 2.

Table 2. Computational costs

Computation cost of	Algorithm 1	Algorithm 2
Reduced matrix ($\tilde{A}$ and $\bar{A}$)	$\mathcal{O}\left(m^3 + (n+1)m^2\right)$	$O\left(nm^2\right)$
DMD modes (Φ and $\bar{\Phi}$)	$\mathcal{O}\left(m^3 + (n+1)m^2\right)$	$\mathcal{O}\left(nm^2\right)$
Total cost	$\mathcal{O}\left(2m^3 + 2(n+1)m^2\right)$	$\mathcal{O}\left(2nm^2\right)$

While Algorithm 1 requires three matrix multiplications each for the calculation of the matrices $\tilde{A}$ and Φ, Algorithm 2 requires one matrix multiplication for the corresponding matrices $\bar{A}$ and $\bar{\Phi}$; see Golub and Van Loan [GL96, Chapter 1].

From the memory point of view, the following matrices require the same amount of memory for both algorithms: the data matrix Y, the reduced matrix ($\tilde{A}$ and $\bar{A}$), and the eigenvectors matrix (W and $\bar{W}$). The floating-point numbers that must be stored for both algorithms are shown in Table 3.

Table 3. Memory requirements for DMD mode matrices

Matrix	Algorithm 1	Algorithm 2
Y	nm	nm
V_r	nm	–
Σ_r^{-1}	m	–
Total memory	$(2n+1)m$	nm

The matrices $\tilde{A}, \bar{A}, W, \bar{W}$ require the storage of m^2 floating-point numbers. The difference in the required memory for the two algorithms is determined by the matrices needed to calculate the DMD modes.

5. Illustrative examples. To demonstrate the algorithm introduced in Section 3, let us look at a simple example of a standing wave signal. Also, our objective is to compare the results with the results of the standard DMD algorithm (Algorithm 1).

Example 5.1 (Standing waves). It is known that the standard DMD algorithm is not able to represent a standing wave in the data [TR$^+$14]. For

example, if only measurements of a single sine or cosine wave are collected, DMD fails to capture periodic oscillations in the data. To demonstrate this, we perform the DMD method on a single measurement

$$\mathbf{x}(t) = \cos(t).$$

In this case, the X matrix only contains a single row

$$X = [x_1 \quad x_2 \quad \dots \quad x_{n-1}]$$

and the DMD algorithm only returns a single eigenvalue, which is unable to capture the oscillation in the data; see Figure 1.

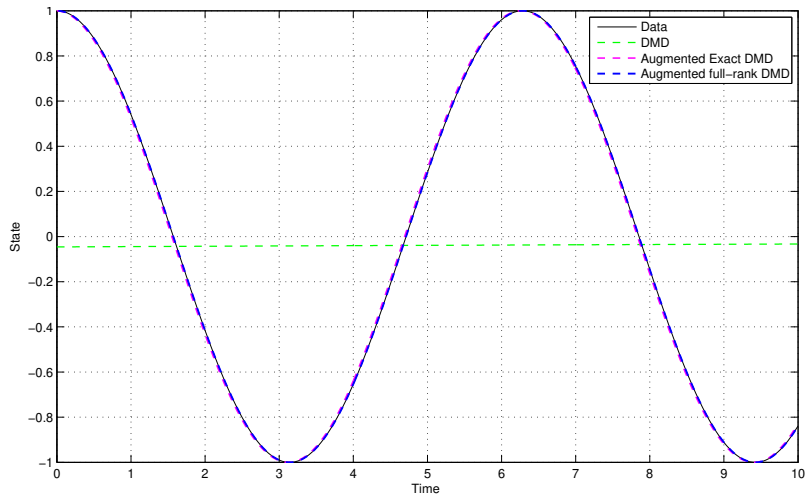

Fig. 1. Example of DMD, augmented exact DMD and augmented full-rank DMD on a standing wave example $\mathbf{x}(t) = \cos(t)$

In fact, for DMD to capture a standing wave, two complex conjugate eigenvalues corresponding to the sine and cosine pair are required. To solve this issue, we construct two augmented data matrices (shift-stacked data matrices):

$$X_{\text{aug}} = \begin{bmatrix} x_1 & x_2 & \dots & x_{n-2} \\ x_2 & x_3 & \dots & x_{n-1} \end{bmatrix} \text{ and } Y_{\text{aug}} = \begin{bmatrix} x_2 & x_3 & \dots & x_{n-1} \\ x_3 & x_4 & \dots & x_n \end{bmatrix}. \tag{5.1}$$

Since X_{aug} contains two linearly independent rows, i.e. it is a full-rank matrix, it has two (conjugate pair complex) DMD eigenvalues. We used both of the aforementioned algorithms to represent $x(t)$; see Figure 1. We can say that the new algorithm (Algorithm 2) produces the same result as the exact DMD procedure (Algorithm 1).

Note. For Algorithm 2, the corresponding augmented matrices $\bar{X}_{\text{aug}}$ and $\bar{Y}_{\text{aug}}$ are the transposed matrices of X_{aug} and Y_{aug} defined in (5.1), i.e.

$\bar{X}_{\text{aug}} = X_{\text{aug}}^T$ and $\bar{Y}_{\text{aug}} = Y_{\text{aug}}^T$. Thus, $\bar{A} = \bar{X}_{\text{aug}}^{\dagger}\bar{Y}_{\text{aug}}$ defined by (3.5) is a 2×2 matrix.

EXAMPLE 5.2 (Example case from finance). We can demonstrate the rank deficiency problem with an example from finance. We can use the DMD method to discover evolutionary patterns in the commodities market. In particular, if we consider the evolution in the price of only one type of commodity, we will get the rank related issue. In fact, this problem is quite similar to the standing wave problem.

Consider the price evolution of Brent crude oil for the period 1.2.2022 – 28.2.2022, which contains 20 trading days; see Fig 2.

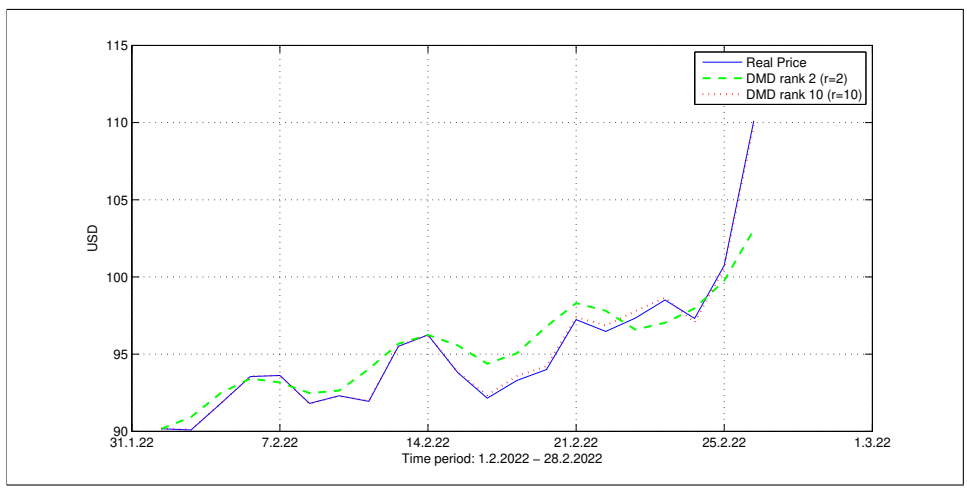

Fig. 2. Brent crude oil price for the period 1.2.2022 – 28.2.2022

Similarly to Example 5.1, the data matrix X has a single row

$$X = [x_1 \quad x_2 \quad \dots \quad x_{20}],$$

where each x_i represents the closing price on that day. In order to overcome the rank mismatch issue, we construct augmented data matrices

$$X_{\text{aug}} = \begin{bmatrix} x_1 & x_2 & \dots & \dots \\ x_2 & x_3 & \dots & \dots \\ \vdots & \vdots & \ddots & \vdots \\ x_s & x_{s+1} & \dots & x_{19} \end{bmatrix}, \quad Y_{\text{aug}} = \begin{bmatrix} x_2 & x_3 & \dots & \dots \\ x_3 & x_4 & \dots & \dots \\ \vdots & \vdots & \ddots & \vdots \\ x_{s+1} & x_{s+2} & \dots & x_{20} \end{bmatrix},$$

where we can choose s such that $10 \leq s \leq 18$, which ensures that the matrices X_{aug} and Y_{aug} will be such that the number of rows will be greater than the number of columns. For each value of s, the matrix X_{aug} has full rank.

We used Algorithm 2 to perform the DMD method on augmented data matrices X_{aug} and Y_{aug} for each s. The results show that the best approximation of the real data is obtained at X_{aug}'s highest rank, which is $\text{rank}(X_{\text{aug}}) = 10$ obtained for $s = 10$. The two approximations for $\text{rank}(X_{\text{aug}}) = 2$ and $\text{rank}(X_{\text{aug}}) = 10$ are shown in Figure 2.

Execution times for Algorithm 1 and Algorithm 2 are computed with the dataset of this example. Table 4 presents a comparison between the two algorithms.

Table 4. Execution time (in sec.)

Number of cycles (k)	Standard DMD (Algorithm 1)	Alternative DMD (Algorithm 2)
$k = 1000$	0.1933	0.1324
$k = 10000$	1.6319	0.9783

6. Conclusion. The goal of this study was to present a new approach to computing approximate DMD modes and eigenvalues. We have introduced and analyzed a new algorithm, an alternative procedure for executing the DMD decomposition in the case of a full-rank dataset. We have demonstrated the performance of the presented algorithms with numerical examples. Based on the results, we can conclude that the introduced approach gives identical results to those of the exact DMD method.

Acknowledgements. Paper written with financial support of Shumen University under Grant No. RD-08-35/18.01.2023.

References

[BK+20] Z. Bai, E. Kaiser, J. L. Proctor, J. N. Kutz and S. L. Brunton, *Dynamic mode decomposition for compressive system identification*, AIAA J. 58 (2020), 561–574.

[B13] S. Bagheri, *Koopman-mode decomposition of the cylinder wake*, J. Fluid Mech. 726 (2013), 596–623.

[BS+15] E. Berger, M. Sastuba, D. Vogt, B. Jung and H. B. Amor, *Estimation of perturbations in robotic behavior using dynamic mode decomposition*, J. Advanced Robotics 29 (2015), 331–343.

[BJOK16] 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*, J. Neuroscience Methods 258 (2016), 1–15.

[CTR12] K. K. Chen, J. H. Tu and C. W. Rowley, *Variants of dynamic mode decomposition: Boundary condition, Koopman, and Fourier analyses*, J. Nonlinear Sci. 22 (2012), 887–915.

[CL16] L. Cui and W. Long, *Trading strategy based on dynamic mode decomposition: Tested in Chinese stock market*, Phys. A: Statist. Mech. Appl. 461 (2016), 498–508.

[GL96] G. H. Golub and C. F. Van Loan, *Matrix Computations*, Johns Hopkins Univ. Press, Baltimore, MD, 1996.

[GK14] J. Grosek and J. Nathan Kutz, *Dynamic mode decomposition for real-time background/foreground separation in video*, arXiv:1404.7592 (2014).

[KG+17] D. P. Kuttichira, E. A. Gopalakrishnan, V. K. Menon, and K. P. Soman, *Stock price prediction using dynamic mode decomposition*, in: 2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI) (Udupi, 2017), 55–60.

[KB+16] J. N. Kutz, S. L. Brunton, B. W. Brunton and J. L. Proctor, *Dynamic Mode Decomposition: Data-Driven Modeling of Complex Systems*, SIAM, 2016.

[MK16] J. Mann and J. N. Kutz, *Dynamic mode decomposition for financial trading strategies*, Quant. Finance 16 (2016), 1643–1655.

[M13] I. Mezić, *Analysis of fluid flows via spectral properties of the Koopman operator*, Ann. Rev. Fluid Mech. 45 (2013), 357–378.

[M05] I. Mezić, *Spectral properties of dynamical systems, model reduction and decompositions*, Nonlinear Dynam. 41 (2005), 309–325.

[N22] G. Nedzhibov, *Dynamic mode decomposition: a new approach for computing the DMD modes and eigenvalues*, Ann. Acad. Rom. Sci. Ser. Math. Appl. 14 (2022), 5–16.

[PE15] J. L. Proctor and P. A. Eckhoff, *Discovering dynamic patterns from infectious disease data using dynamic mode decomposition*, Internat. Health 7 (2015), 139–145.

[R09] C. W. Rowley, I. Mezić, S. Bagheri, P. Schlatter and D. S. Henningson, *Spectral analysis of nonlinear flows*, J. Fluid Mech. 641 (2009), 115–127.

[S10] P. J. Schmid, *Dynamic mode decomposition of numerical and experimental data*, J. Fluid Mech. 656 (2010), 5–28.

[S11] P. J. Schmid, *Application of the dynamic mode decomposition to experimental data*, Experiments Fluids 50 (2011), 1123–1130.

[SS08] P. J. Schmid and J. Sesterhenn, *Dynamic mode decomposition of numerical and experimental data*, in: 61st Annual Meeting of the APS Division of Fluid Dynamics, Amer. Phys. Soc., 2008, 208–231.

[SS11] A. Seena and H. J. Sung, *Dynamic mode decomposition of turbulent cavity ows for selfsustained oscillations*, Int. J. Heat Fluid Flows 32 (2011), 1098–1110.

[TR+14] J. H. Tu, C. W. Rowley, D. M. Luchtenburg, S. L. Brunton and J. N. Kutz, *On dynamic mode decomposition: Theory and applications*, J. Comput. Dynam. 1 (2014), 391–421.

G. H. Nedzhibov
Faculty of Mathematics and Informatics
Shumen University
Shumen 9700, Bulgaria
ORCID: 0000-0002-7422-7369
E-mail: g.nedzhibov@shu.bg