TIME-RESOLVED DENOISING USING MODEL ORDER REDUCTION, DYNAMIC MODE DECOMPOSITION, AND KALMAN FILTER AND SMOOTHER

MOJTABA F. FATHI*
Department of Mechanical Engineering, University of Wisconsin-Milwaukee
Milwaukee, WI, USA

AHMADREZA BAGHAIE
Department of Electrical and Computer Engineering, New York Institute of Technology
Long Island, NY, USA

ALI BAKHSHINEJAD
Departments of Neurosurgery, Medical College of Wisconsin
Milwaukee, WI, USA

RAFAEL H. SACHO
Departments of Neurosurgery, Medical College of Wisconsin
Milwaukee, WI, USA

ROSHAN M. D’SOUZA
Department of Mechanical Engineering, University of Wisconsin-Milwaukee
Milwaukee, WI, USA

ABSTRACT. In this research, we investigate the application of Dynamic Mode Decomposition combined with Kalman Filtering, Smoothing, and Wavelet Denoising (DMD-KF-W) for denoising time-resolved data. We also compare the performance of this technique with state-of-the-art denoising methods such as Total Variation Diminishing (TV) and Divergence-Free Wavelets (DFW), when applicable. Dynamic Mode Decomposition (DMD) is a data-driven method for finding the spatio-temporal structures in time series data. In this research, we use an autoregressive linear model resulting from applying DMD to the time-resolved data as a predictor in a Kalman Filtering-Smoothing framework for the purpose of denoising. The DMD-KF-W method is parameter-free and runs autonomously. Tests on numerical phantoms show lower error metrics when compared to TV and DFW, when applicable. In addition, DMD-KF-W runs an order of magnitude faster than DFW and TV. In the case of synthetic datasets, where the noise-free datasets were available, our method was shown to perform better than TV and DFW methods (when applicable) in terms of the defined error metric.

2020 Mathematics Subject Classification. Primary: 58F15, 58F17; Secondary: 53C35.
Key words and phrases. Data driven, dynamic mode decomposition, model order reduction, Kalman Filter, denoising.

* Corresponding author: Mojtaba F. Fathi.
1. **Introduction.** The concept of Dynamic Mode Decomposition (DMD) was first introduced by [21] in 2008. It was initially aimed at studying the spatial dynamic modes of fluid flow [20, 2]. Given a time-varying dataset, DMD can approximate its underlying nonlinear dynamics as a linear auto-regressive model. DMD extracts a set of mode shapes and associates each one with an eigenvalue. Each mode shape represents the spatial spread of a dominant feature and the corresponding eigenvalue specifies how that feature evolves over time in terms of the frequency of oscillation and the rate of growth or decay. The DMD was initially developed for extracting the dominant dynamic features from flow fields [20]. Nonetheless, it shortly found applications in other areas of study as well. In 2009, [19] showed the connection between DMD and the Koopman operator. The Koopman operator is an infinite-dimensional linear representation of nonlinear finite-dimensional dynamics and the DMD was shown to approximate its modes [19, 2]. Even though the DMD was initially developed as a tool for analyzing the dynamic modes of a dataset of equally-spaced sequential snapshots, in 2014, its theory was expanded to handle the mapping between two different datasets of paired snapshots [22]. [26] further introduced the extended DMD (EDMD) in 2015 to approximate not only the modes of the Koopman operator, but also its leading eigenvalues and eigenfunctions [26]. The extended DMD is a computationally intensive algorithm. As a workaround, the kernel-based DMD (KDMD) was proposed in 2015 [25]. By applying DMD on a noisy dataset, many of the dynamic modes identified will be mostly due to noise [6]. As [6] showed, the effect of sensor noise is seen as a shift in the computed DMD eigenvalues such that they appear to be more stable than they are in reality. In order to compensate the effect of noise, they proposed three different variants of DMD among which forward-backward DMD (fbDMD) was shown to usually perform better than the other two. The fbDMD also has the advantage of not requiring any prior knowledge about the noise covariance matrix. Another attempt to deal with noise in a partially-sampled 2-D dataset was made by combining DMD and a set of discrete cosine transform (DCT) basis vectors [7].

The Kalman filter is a recursive filter that can efficiently estimate the internal state of a linear dynamic system by minimizing the mean of the squared error in the presence of modeling uncertainties, process noise, and measurement noise [13]. Since its first introduction, The Kalman filter has been used extensively in various applications. Also, many researchers have attempted to expand its applicability by proposing variants of the Kalman filter, e.g. the extended Kalman filter [10], the unscented Kalman filter [23], and the linear Kalman smoothing [18], to name a few. The linear Kalman filter may be applied to dynamic processes governed or approximated by a linear difference equation, as the name implies. Since the DMD and its variants can derive such approximation for a given dynamic dataset solely based on the data, the idea of integrating DMD and the Kalman filter sounds promising. In 2015, [11] proposed a similar idea for prediction of wind turbine wakes by embedding a reduced-order model (ROM) in a Kalman filter where the ROM was derived by using DMD. [15] proposed a novel DMD method based on a Kalman filter for parameter estimation [15]. They also introduced an online algorithm based on the extended Kalman filter and DMD for simultaneous system identification and denoising of datasets with a small number of degrees of freedom [16]. The Kalman filter is a real-time state predictor, i.e., the state is predicted based on observations until the current time. In case of post processing, the Kalman filter can be combined with a Kalman smoother to further refine the state estimation. The
Kalman smoother computes the optimal state prediction based on all observations. When the size of the observations is large, computing the Kalman filtered and smoothed state may be computationally expensive. In our case, assuming the input data vector is quite large (n > 1000 with n being the size of the input data vector), we employ a reduced-order model formulation to significantly reduce computation time.

In this paper, we present a novel purely data-driven, parameter-free, and autonomous method to denoise time-resolved data. It uses the Kalman Filtering-Smoothing (KFS) framework [18]. The forward model inside the KFS is derived from the input data using Dynamic Mode Decomposition (DMD)[14]. We have developed a novel method to estimate the measurement noise and process noise covariance matrices based on the input data. These matrices are needed for KFS to operate properly. To reduce computation time, a reduced order model implementation has been developed [24].

2. Method.

2.1. Dynamic mode decomposition (DMD). DMD is a data-driven method to find the the best fit linear dynamic representation from observations that are possibly coming from a non-linear system. Given a sequence of m time-indexed data vectors/snapshots $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m$, DMD describes the evolution of the state $\mathbf{x}$ of the system from time $k$ to $k+1$ in terms of a linear operator $A$ as

$$\mathbf{x}_{k+1} \approx A \mathbf{x}_k$$

with $\mathbf{x}_k \in \mathbb{R}^n$ and $A \in \mathbb{R}^{n \times n}$. While it may seem impossible to represent the dynamics of a nonlinear system using superposition of modes whose dynamics are governed by eigenvalues, it has been shown that DMD can be considered as the numerical approximation of the Koopman spectral analysis [19].

Arranging the snapshots $\mathbf{x}_k, 1 \leq k \leq m$ in matrices

$$\mathbf{X} = \begin{bmatrix} \mathbf{x}_1 & \mathbf{x}_2 & \cdots & \mathbf{x}_{m-1} \end{bmatrix}$$

$$\mathbf{X}' = \begin{bmatrix} \mathbf{x}_2 & \mathbf{x}_3 & \cdots & \mathbf{x}_m \end{bmatrix}$$

the relationship between $\mathbf{X}$ and $\mathbf{X}'$ is given by

$$\mathbf{X}' = A \mathbf{X}$$

The operator $A$ can be computed as

$$A = \mathbf{X}' \mathbf{X}^\dagger$$

where $\mathbf{X}^\dagger$ is the pseudo-inverse of $\mathbf{X}$. This formulation finds the operator $A$ which minimizes the Frobenius norm $\|\mathbf{X}' - A \mathbf{X}\|_F$. For typical dynamic systems described by partial differential equations (PDEs), $n >> 1$ and therefore, finding the operator $A$ directly is intractable. The DMD algorithm prescribed by [21] finds compact representation of $A$ in terms of the spatial DMD modes and the temporal eigenvalues. The state vector at time instant $k$ can be written as a weighted sum of the DMD modes as

$$\mathbf{x}_k = \sum_{i=1}^{r} b_i \lambda_i^k \phi_i$$
where $\phi_i \in \mathbb{C}^n$ is the $i^{th}$ DMD mode, $\lambda_i \in \mathbb{C}$ is the corresponding eigenvalue, and $b_i \in \mathbb{C}$ is the weight. The exact DMD algorithm based on singular value decomposition (SVD) proposed in [20] is used to compute the $r$-ranked decomposition $[\phi_i, \lambda_i]_{i=1}^r$ and the associated weights $b_i$ as

$$
\begin{bmatrix}
    b_1 \\
    b_2 \\
    \vdots \\
    b_m
\end{bmatrix} = \begin{bmatrix}
    \phi_1 \\
    \phi_2 \\
    \vdots \\
    \phi_m
\end{bmatrix}^\dagger x_0
$$

(7)

Data:
- $[x_1 \cdots x_m]$: the sequence of $m$ time-indexed data vectors each of size $N \times 1$

Result:
- $\Phi$: the matrix of DMD modes
- $\Lambda$: the diagonal matrix of DMD eigenvalues
- $U$: the matrix of POD basis vectors

\begin{algorithm}
\textbf{X} ← $[x_1 \ldots x_m-1]$
\textbf{X'} ← $[x_2 \ldots x_m-1]$
Find the rank $r$ by applying SVHT to the matrix $X$

\textbf{begin} the forward path
\begin{itemize}
    \item Find the SVD of $X$ such that $X = U_x \Sigma_x V_x^*$
    \item $U_{rx} \leftarrow U_x$ truncated to the first $r$ columns
    \item $\Sigma_{rx} \leftarrow \Sigma_x$ truncated to the upper-left $r \times r$ matrix
    \item $V_{rx}^* \leftarrow V_x^*$ truncated to the first $r$ rows
    \item $\hat{A}_m \leftarrow U_{rx}^* X' V_{rx} \Sigma_{rx}^{-1}$
\end{itemize}
\textbf{end}

\textbf{begin} the backward path
\begin{itemize}
    \item Find the SVD of $X'$ such that $X' = U_y \Sigma_y V_y^*$
    \item $U_{ry} \leftarrow U_y$ truncated to the first $r$ columns
    \item $\Sigma_{ry} \leftarrow \Sigma_y$ truncated to the upper-left $r \times r$ matrix
    \item $V_{ry}^* \leftarrow V_y^*$ truncated to the first $r$ rows
    \item $\hat{B}_m \leftarrow U_{ry}^* X V_{ry} \Sigma_{ry}^{-1}$
\end{itemize}
\textbf{end}

$\hat{A} \leftarrow (\hat{A}_m \hat{B}_m^{-1})^{1/2}$
Find the eigenvalues $\Lambda$ and eigenvectors $W$ of $\hat{A}$, i.e. $\hat{A}W = WA$
Compute the DMD modes $\Phi \triangleq X' V_{rx} \Sigma_{rx}^{-1} W$
\textbf{return} $\Phi, \Lambda, U_{rx}$ // $U_{rx}$ is returned as the matrix of POD basis vectors $U$
\end{algorithm}

Typically, the observables are contaminated by acquisition noise. This affects the computation of DMD modes and eigenvalues. [6] showed that noise in the observables causes a shift in the computed DMD eigenvalues such that they appear more stable than they are in reality. To overcome the effect of noise, methods such as noise-corrected DMD, total least-squares DMD, and forward-backward DMD
have been proposed [6]. Of these, the forward-backward DMD (fbDMD) is quite
effective since it works in cases of high dimensional systems and does not require the
estimate of noise covariance. Algorithm 1 illustrates the fbDMD process given a time
sequence of \( m \) possibly noisy data vectors. One important detail is the parameter
\( r \) that specifies the desired number of DMD modes. We use the method of Singular
Value Hard Thresholding (SVHT) [9] prior to fbDMD to find the optimal number
of DMD modes. Given the matrix of DMD modes \( \Phi \) and the diagonal matrix of
eigenvalues \( \Lambda \), we can write an approximation to the matrix \( A \) as
\[
A \approx \Phi \Lambda \Phi^\dagger
\] (8)
The matrix \( A \) is not explicitly computed as it can be too large to even store. Instead,
a reduced order version of \( A \) is precomputed to be used as the forward model in a
Kalman filter-smoother framework.

2.2. Kalman filter. Consider an autonomous discrete-time dynamic process gov-
erned by the linear stochastic equation
\[
\hat{x}_k = A \hat{x}_{k-1} + w^{(x)}_{k-1}
\] (9)
and the full state measurements
\[
\bar{x}_k = \hat{x}_k + u^{(x)}_k
\] (10)
where \( \hat{x} \in \mathbb{R}^n \) is the state vector, \( \bar{x} \in \mathbb{R}^n \) is the measurement, \( w^{(x)} \in \mathbb{R}^n \) is the
white normally-distributed process noise with covariance \( Q_x \), and \( u^{(x)} \in \mathbb{R}^n \) is the
white normally-distributed measurement noise with covariance \( R_x \). The Kalman
filter provides an estimate to the state vector \( \hat{x} \) given the noisy measurements \( \bar{x} \) and the noise covariance matrices \( Q_x \) and \( R_x \) through a sequence of prediction and
correction steps. The prediction step is performed as
\[
\begin{aligned}
\hat{x}_k^{(p)} &= A \hat{x}_{k-1} \quad \text{(forward model)} \\
P_{x_k}^{(p)} &= \begin{pmatrix} A & \Lambda \end{pmatrix} P_{x_{k-1}}^{(c)} \begin{pmatrix} A^T \ 0 \end{pmatrix} + Q_x \\
K_{x_k} &= P_{x_k}^{(p)} \left( P_{x_k}^{(p)} + R_x \right)^{-1} \\
\bar{x}_k^{(p)} &= \hat{x}_k^{(p)} + K_{x_k} \left( \bar{x}_k - \hat{x}_k^{(p)} \right) \\
P_{x_k}^{(c)} &= (I - K_{x_k}) P_{x_k}^{(p)}
\end{aligned}
\] (12)
where \( K_{x_k} \) denotes the Kalman gain.

2.3. Reduced-order model. Due to the matrix inversion operation in equation
(12), applying the Kalman filter to large-scale problems might not be feasible. As a
possible workaround, we project the snapshots onto the space spanned by the POD
basis vectors and take the projection coefficients as the new state variables. The
number of the new state variables will be the same as the number of POD basis
vectors. As seen in Algorithm 1, as many POD basis vectors as the DMD modes
are taken and therefore, the number of state variables will equal the rank of DMD.

A low-dimensional state vector \( \hat{z} \in \mathbb{R}^r \) is defined as the projection of the high-
dimensional state vector \( \hat{x} \) onto the POD basis vectors (columns of \( U \)) obtained
from fbDMD,
\[
\hat{z} \triangleq U^T \hat{x}
\] (13)
Conversely, the high-dimensional state vector $\hat{x}$ can be approximated as
\[
\hat{x} \approx U\hat{z}
\] (14)

The autonomous discrete-time dynamic process given as equation (9) becomes
\[
U^T\hat{x}_k = U^TA\hat{x}_{k-1} + U^T\hat{w}_k^{(x)} \\
\approx U^TA\hat{z}_{k-1} + U^T\hat{w}_k^{(x)} \\
\approx \left(U^T\Phi\Lambda\Phi^TU\right)\hat{z}_{k-1} + U^T\hat{w}_k^{(x)} \\
\therefore \hat{z}_k \approx A\hat{z}_{k-1} + \hat{w}_k^{(z)}
\] (15)

where $A = \left(U^T\Phi\Lambda\Phi^TU\right) \in \mathbb{R}^{r \times r}$ is the reduced-order forward model matrix and $\hat{w}_k^{(z)} \triangleq U^T\hat{w}_k^{(x)}$ is the reduced-order process noise. Given that $\hat{w}_k^{(x)}$ is a white normally-distributed variable with covariance $Q_z$, $\hat{w}_k^{(z)}$ will also be a white normally-distributed variable with covariance $Q_z$ such that
\[
Q_z = U^TQ_xU
\]

Similarly, the full state measurements given as equation (10) become
\[
U^T\bar{x}_k = U^T\hat{x}_k + U^T\bar{v}_k^{(x)} \\
\therefore \bar{z}_k = \hat{z}_k + \bar{v}_k^{(z)}
\] (19)

where $\bar{v}_k^{(z)} \triangleq U^T\bar{v}_k^{(x)}$ is the reduced-order measurement noise. Given $\bar{v}_k^{(x)}$ is a white normally-distributed variable with covariance $R_x$, $\bar{v}_k^{(z)}$ will also be a white normally-distributed variable with covariance $R_z$ such that
\[
R_z = U^TR_xU
\]

2.4. Reduced-order Kalman filter-smoother. Considering the reduced-order process and measurement equations (16) and (19), the prediction equations of the Kalman filter (equations (11)) become
\[
\begin{cases}
\hat{z}^{(p)}_k = A\hat{z}_{k-1}^{(c)} \\
P^{(p)}_k = A\hat{z}^{(c)}_k + Q_z
\end{cases}
\]
(reduced-order forward model) (21)

Likewise, the correction equations of the Kalman filter (equations (12)) become
\[
\begin{cases}
K_k = P_k^{(p)} \left(P_k^{(p)} + R_z\right)^{-1} \\
\hat{z}_k^{(c)} = \hat{z}^{(p)}_k + K_k \left(U^T\hat{x}_k - \hat{z}^{(p)}_k\right) \\
P_k^{(c)} = (I - K_k) P_k^{(p)}
\end{cases}
\]

The initial values of the predicted state $\hat{z}^{(p)}_1$ and the predicted estimate error covariance $P_1^{(p)}$ are taken as
\[
\begin{cases}
\hat{z}_1^{(p)} = U^T\bar{x}_1 \\
P_1^{(p)} = Q_z
\end{cases}
\]

(23)
After the prediction and correction steps are performed for all snapshots, a backward smoother [5] is applied to further smooth the data as given below

\[
\begin{align*}
K_k^{(s)} &= P_k^{(c)} A_z \left( P_{k+1}^{(p)} \right)^{-1} \\
P_k^{(s)} &= P_k^{(c)} - K_k^{(s)} \left( P_{k+1}^{(p)} - P_k^{(s)} \right) \left( K_k^{(s)} \right)^T \\
\hat{z}_k^{(s)} &= \hat{z}_k^{(c)} - K_k^{(s)} \left( \hat{z}_{k+1}^{(p)} - \hat{z}_{k+1}^{(s)} \right)
\end{align*}
\] (24)

where the values at \( k = m \) are given as

\[
\begin{align*}
\hat{z}_m^{(s)} &= \hat{z}_m^{(c)} \\
P_m^{(s)} &= P_m^{(c)}
\end{align*}
\] (25)

Note that equations (21) - (25) are in the reduced-order space with dimension \( r < n \) and therefore can be evaluated very fast.

2.5. Reconstruction. As seen in Algorithm 1, the matrix of POD basis vectors \( U \) is directly derived from the input data matrix. Since the input data is noisy, the derived POD vectors will capture some noise as well. In order to prevent the noise captured in the POD vectors from spreading into the reconstructed data, the POD vectors should be denoised. Here we employ the wavelet hard-thresholding method for denoising as explained in [27] combined with cycle spinning [4]. Denoising is performed for each POD vector \( \hat{u}_k \) individually as detailed in Algorithm 2.

Data:
- \( u_k \): the \( k \)-th POD vector

Result:
- \( \hat{u}_k \): the denoised POD vector

for \( i \leftarrow 1 \) to desired number of cycles do
  \( \hat{v} \leftarrow \) cyclically shift \( \hat{u}_k \) by \( i \)
  Do wavelet decomposition of \( \hat{v} \) to get the wavelet coefficients
  \( w \leftarrow \) the non-zero fine-scale wavelet coefficients
  \( \sigma_n \leftarrow \text{MAD}(w)/0.6745 \) // MAD represents Median Absolute Deviation
  \( T \leftarrow \sigma_n \sqrt{2 \log \text{(length of } w)} \) // the threshold
  Set to zero all wavelet coefficients which their absolute value is less than \( T \)
  \( \hat{v} \leftarrow \) the inverse wavelet transform using the modified wavelet coefficients
  \( \hat{u}_{ki} \leftarrow \) cyclically shift \( \hat{v} \) by \(-i\)
end
\( \hat{u}_k \leftarrow \) the average of all \( \hat{u}_{ki} \)
return \( \hat{u}_k \)

Algorithm 2: Wavelet hard-thresholding and cycle spinning for POD vector denoising [27, 4]

Algorithm 3 summarizes the proposed implementation of Kalman filter and smoother.
Data:
- $\mathbf{U}$: the $N \times r$ matrix of orthonormal POD basis vectors
- $\Phi$: the $N \times m$ matrix of DMD modes
- $\Lambda$: diagonal matrix of $r$ DMD eigenvalues
- $\bar{x}_k$: the $m$ vectorized noisy snapshots of length $N$ ($1 \leq k \leq m$)
- $\mathbf{Q}_z$: the $r \times r$ matrix of process noise covariance
- $\mathbf{R}_z$: the $r \times r$ matrix of measurement noise covariance

Result:
- $\tilde{x}_k$: the $m$ denoised snapshots ($1 \leq k \leq m$)

Algorithm 3: Kalman filtering and smoothing in the reduced-order space of POD basis vectors

2.6. The noise covariance matrices. A key part of the Kalman filter is estimating the covariance matrices of the process and measurement noise. Here, a scheme is proposed to find an estimate to those matrices. Using the reduced-order process and measurement equations (16) and (19)

$$\tilde{z}_k = \hat{z}_k + \tilde{u}_k^{(z)}$$
\[
A \tilde{z}_k - z_{k-1} + \tilde{w}_{k-1} + \tilde{v}_k
= A z_{k-1} - z_{k-1}^{(z)} + w_{k-1}^{(z)} + v_k^{(z)}
\]  

(26)

based on which, the error \(e_k\) between the measured snapshot \(k\) and its fbDMD-based prediction is defined as

\[
e_k \triangleq z_k - A z_{k-1} = -A z_{k-1}^{(z)} + w_{k-1}^{(z)} + v_k^{(z)}
\]  

(27)

Since the three random variables on the right hand side of equation (27) are white, normally-distributed, and independent from each other, the error \(e_k\) will also be a white normally-distributed random variable with the covariance \(S\) given as

\[
S = E[e_k e_k^T] = A z R z A^T z + Q z + R_z
\]  

(28)

The covariance matrix \(S\) may be calculated by finding the error between the projected snapshots and their corresponding projected fbDMD-based predictions and then, employing an Oracle Approximating Shrinkage (OAS) estimator [3] to estimate the covariance of the projected error. If the measurement noise covariance matrix \(R_x\) is known, the projected covariance matrix \(R_z\) can be calculated from equation (20) and so, equation (28) can be solved for \(Q_z\) easily.

### 2.6.1. Determining the measurement noise variance

In our research, we assume that the signal noise at each location is uncorrelated and the noise variance remains the same over the whole region. Therefore, the noise covariance matrix \(R_x\) can be given by a diagonal matrix. For example, for a 2D flow velocity field, it will be

\[
R_x = \text{diag} (\sigma_u^2, \sigma_v^2, \cdots, \sigma_u^2, \sigma_v^2)
\]  

(29)

where \(\sigma_u^2, \sigma_v^2\) are the measurement noise variances corresponding to the \(u\) and \(v\) velocity components, respectively. If the given dataset has a geometric mask which distinguishes the dynamic region from the surrounding static region, the noise covariance may be approximated by following the approach presented here. First, the static region mask is found by enlarging the geometric mask by 2 pixels in each direction and then, inverting it. Then, a region of interest (ROI) inside the static region is chosen to compute the variances. For the data point \(i\) in the ROI,

\[
s_{ui}^2 = \frac{1}{m-1} \sum_{k=1}^{m} (u_{ik} - \bar{u}_i)^2
\]  

(30)

where \(u_{ik}\) is the \(u\) velocity component at location \(i\) at the time instant \(k\), \(\bar{u}_i\) is the mean of all \(u_{ik}\) values, and \(m\) is the number of snapshots. The noise variance of the \(u\) component is estimated as

\[
\sigma_u^2 = \frac{1}{n} \sum_{i=1}^{n} s_{ui}^2
\]  

(31)

where \(n\) is the number of points in the ROI. The variance \(\sigma_v^2\) is estimated similarly.

### 2.7. Implementation

The proposed methods were implemented in the Python programming language. Numpy library was used for various matrix operations. For comparison, the proposed method was evaluated against the total variation method (TV) [1] and divergence-free wavelets with Sure-Shrink, median absolute deviation, and partial cycle spinning (DFW) [17] (where applicable). Both TV and DFW are the leading state-of-the-art denoising methods as shown in our previous study [8]. The DFW and TV implementations in MATLAB with C++ bindings developed by the respective authors were used. Both DFW and TV require manual selection of
parameters. In case of TV, it is the weighting parameter of the regularization terms and in the case of DFW, it is the minimum size of the wavelet (which has direct bearing on the number of levels of decomposition) and the number of cycles in partial cycle spinning. The latter accounts for issues with lack of shift invariance in the critically-sampled wavelet transform by averaging over denoised cyclically-shifted versions of the signal. There are no prescribed methods to select these parameters. On the other hand, the DMD-KF-W framework (Dynamic Mode Decomposition combined with Kalman Filtering, Smoothing, and Wavelet Denoising) presented in this work is entirely autonomous.

3. Results. The proposed method was evaluated in four different test cases against TV and DFW methods. For the first test, a synthetic dataset generated by using the unforced Duffing equation was used [7]. As the second test, a dataset representing the 2D velocity field for the wake behind a cylinder at Reynolds number Re = 100 taken from [14] was used. For the third test, a synthetic 4D-Flow MRI dataset generated based on the geometry of an actual aneurysm was used. Since the noise-free dataset was available in these three cases, the root mean square error (RMSE) defined below was used as the comparison metric

\[
\text{RMSE} \triangleq \frac{1}{\sqrt{n}} \|Z_{\text{rec}} - Z_{\text{ref}}\|_F
\]

where \(Z_{\text{rec}}\) is the reconstructed dataset, \(Z_{\text{ref}}\) is the reference noise-free dataset, and \(n\) is the total number of elements of the dataset. For the last test, two datasets acquired by in-vivo 4D-Flow MRI scans were used.

3.1. Test case 1: The unforced Duffing equation. The unforced Duffing equation taken from [7] was used to generate the test dataset. The governing differential equation is

\[
\ddot{x} = -\delta \dot{x} - x(\gamma + \alpha x^2)
\]

where \(\delta = 0.5\), \(\gamma = -1\), and \(\alpha = 1\). The region \(x, \dot{x} \in [-2, 2]\) was discretized as a \(41 \times 41\) mesh over which the equation was solved as explained in [7] and then, the \(\dot{x}\) values were taken as the noise-free dataset consisting of 51 snapshots. Random Gaussian noise with the standard deviation of 0.25 was added to the noise-free dataset to make the noisy dataset as seen in Figure 1.

The number of DMD modes for the noisy dataset was taken as 6, according to SVHT. Figure 1 shows a few sample snapshots from the reconstructed datasets. The two top rows show the noise-free and noisy datasets, respectively. The reconstructions of the TV and DMD-KF-W are also shown as indicated. Several runs of TV were conducted and the ones resulting in the least RMSE were picked for comparison, but since DMD-KF-W is a parameter-free method, only one run was conducted to generate the results. The computation times were 5-6 seconds for TV and about 0.1-0.2 seconds for DMD-KF-W. For the noisy and the two reconstructed datasets, the overall RMSE values are provided as well. Figure 1 shows the RMSE values calculated for the noisy and the two reconstructed datasets. The solid and dotted lines represent the per-snapshot and overall RMSE values, respectively.

3.2. Test case 2: Wake behind a cylinder at Re=100. As the second test, a dataset representing the 2D velocity field for the wake behind a cylinder at Reynolds number Re = 100 taken from [14] was used. The dataset consists of 151 snapshots with regular time intervals of 0.2 seconds over an \(89 \times 39\) mesh grid. Random
Gaussian noise with the standard deviation of 5% of the maximum velocity magnitude was added to both components. Since the test dataset represents a 2D flow field, the DFW method fails to denoise it properly unless it is turned into a 3D flow field. This was achieved by adding a dummy zero-valued third velocity component \( w \) everywhere. Also, it was assumed the size of the mesh in the \( w \) direction was 39. After the resulting 3D dataset was denoised by DFW method, the denoised dataset was turned back into a 2D dataset by discarding the reconstructed dummy component \( w \) and the extra copies of the components \( u \) and \( v \) along the \( w \) direction.

Figure 2 shows the results of the test for the snapshot number 21. The rank of DMD was taken as 7 as resulted from SVHT. It took about a half a second for DMD-KF-W to denoise the whole dataset in a single run whereas several runs of TV and DFW were performed to find the best for comparison. The runs leading to the best results of TV and DFW took about 33 and 1,800 seconds, respectively. Figure 3 shows the effects of the tuning parameters of TV and DFW on the RMSE values.

**3.3. Test case 3: A synthetic 4D-Flow MRI dataset.** The approach explained in [8] was taken to generate a synthetic 4D-Flow MRI dataset. First, a CFD simulation was conducted to simulate the blood flow through an actual intra-cranial aneurysm geometry where the boundary conditions were taken from a realistic pulsatile flow. The high-resolution time snapshots of the CFD simulation were downsampled into a \( 25 \times 31 \times 86 \) grid to form the noise-free reference dataset. The synthetic dataset was generated by adding noise to the noise-free reference dataset.
Figure 2. The results of second test case (wake behind a cylinder at Re=100). On the left, the reference, noisy, and reconstructed datasets are shown. The images show the snapshot number 21. The bottom row shows the reconstruction errors with the overall RMSE values, where the lower is better. On the right, per-snapshot (solid lines) and overall (dotted) RMSE values of the noisy and the reconstructed datasets are presented.

Figure 3. The effects of the tuning parameters of TV and DFW on the reconstruction error for the second test dataset. For TV (on left), the effect of the weighting parameter of the regularization term is shown where $\lambda = 0.07$ resulted in the best reconstruction. For DFW (on right), the effects of the smallest wavelet level size and the number of cycles are shown. In this case, 5 cycles with the smallest wavelet level size of 10 resulted in the best reconstruction.

in $k$-space [12]. There were a total of 50 snapshots in the dataset. According to the geometric mask, the aneurysm contained 6516 spatial locations which accounts for 9.8% of the total number of points in the grid. The ROI for noise covariance estimation was defined as a set of 50 voxels picked from inside the static tissue mask randomly.
For comparison, several runs of TV and DFW were conducted and then, the ones resulting in the best reconstruction (least RMSE) were picked. Figure 4 shows the effects of the tuning parameters of TV and DFW on the RMSE values. In contrast, the results of DMD-KF-W were acquired by a single run since it is a parameter-free method.

**Figure 4.** The effects of tuning parameters of TV and DFW on the reconstruction error for the synthetic 4D-Flow MRI dataset. For TV (on left), the effect of the weighting parameter of the regularization term is shown where $\lambda = 0.03$ resulted in the best reconstruction. For DFW (on right), the effects of the smallest wavelet level size and the number of cycles are shown. In this case, 4 cycles with the smallest wavelet level size of 12 resulted in the best reconstruction.

The number of dynamic modes was taken as 3 as resulted from SVHT. Figure 5 shows sample slices from the noise-free reference, noisy, and reconstructed datasets. The reconstruction errors are shown as well. The computation times were about 127 seconds for TV, about 19 seconds for DFW, and 3.6 seconds for DMD-KF-W. Figure 5 also shows the plot of per-snapshot RMSE values as the bold lines and the total RMSE values as the dotted lines.

### 3.4. Test case 4: In-vivo 4D-Flow MRI datasets

Two datasets acquired by in-vivo 4D-Flow MRI scans were used in this test case. Unlike the first test case, the noise-free datasets are unknown and so, there is no metric to make comparison based on and evaluate the results. Each dataset consisted of 20 consecutive snapshots. The original in-vivo datasets were cropped to show the aneurysms only. The spatial resolutions were $30 \times 40 \times 103$ and $22 \times 30 \times 48$ after cropping. For either dataset, the geometry of the aneurysm was provided as a binary mask. The geometries of the aneurysms contained 4900 and 1566 spatial locations which respectively accounted for 4.0% and 4.9% of the spatial locations scanned.

The in-vivo datasets are affected by eddy current artifacts which should be removed before denoising. In our method, the eddy current artifact is approximated as a static quadratic polynomial fitted to the noisy data through least squares fitting. The polynomial fitting is performed in areas which are supposed to represent static tissues since the data in those areas consists of noise and eddy current artifact only. For each snapshot, a quadratic polynomial is fitted to the noisy data corresponding to the static tissue mask. Then, the fitted polynomial is taken as an
Figure 5. The results for the synthetic 4D-Flow MRI dataset. The synthetic data resembles a cardiac cycle. (a) The top row shows sample slices where at each point, the norm of the velocity is shown. The bottom row shows the norm of difference between the datasets and the noise-free reference. The bottom row figures are titled with the corresponding total RMSE values. All areas outside the geometry mask are zeroed before RMSE values are calculated. (b) Per-snapshot (solid lines) and total (dotted) RMSE values of the noisy and the reconstructed datasets.

approximation to the eddy current artifact everywhere and is subtracted from the corresponding snapshot to eliminate the eddy current artifact.

The computation times for the first in-vivo dataset were about 100 seconds for TV, about 13 seconds for DFW, and 14 seconds for DMD-KF-W. Similarly, the computation times for the second in-vivo dataset were about 26 seconds for TV, 3-4 seconds for DFW, and 3-4 seconds for DMD-KF-W. According to SVHT, the number of dynamic modes were respectively taken as 8 and 7. The higher the number of modes is, the more time it takes to denoise them. That’s why the computation times increased. Two sample slices of the noisy datasets are shown in Figure 6.

4. Discussion. Four tests were conducted to compare the proposed denoising method versus TV and DFW methods which are well-known state-of-the-art denoising methods. In contrast to TV and DFW, our method considers the dynamics of the underlying dynamic system while denoising. This is done by using the dominant dynamic modes and eigenvalues detected by DMD-KF-W to implement the forward model. Also, our method is parameter-free and runs fast. It didn’t take more than a half a minute for any of the test cases to run our method.

The first test was based on a 2D dataset generated from the unforced Duffing equation. As Figure 1 shows, the per-snapshot RMSE of DMD-KF-W result is initially high but it drops quickly and goes lower than the RMSE of TV as the simulation marches forward in time. It starts to go up again over the last few snapshots. This is because DMD approximates the nonlinear dynamics as a linear
auto-regressive model and so, it fails to estimate the next snapshot properly in the presence of highly nonlinear dynamics. The dotted lines in Figure 1 represent the overall RMSE values. While both methods have resulted in more than 50% reduction in overall RMSE, the overall RMSE of DMD-KF-W has seen more drop than TV’s (0.069 versus 0.111). The last snapshot (#51) in Figure 1 shows the effectiveness of the proposed algorithm in recovering fine features. Even though most fine features seem to be washed out by the noise, the DMD-KF-W method was still able to reconstruct some of them by taking advantage of knowing the dominant dynamic features and their corresponding growth/decay rate and frequency of oscillation.

The second test case represented the 2D flow velocity field of an incompressible fluid. Even though the test data was synthetic and the noise variance was known, the procedure explained in section 2.6.1 was followed to approximate the measurement noise covariance matrix. In contrast to the TV, which can be applied to any dataset, DFW is only applicable to datasets representing a 3D fluid velocity profile. Since the second test dataset was from a 2D problem, we had to convert it to a 3D dataset by adding a dummy third direction and the corresponding zero-valued velocity component in order to be able to use DFW. This resulted in a significantly higher computation time for DFW compared to TV and DMD-KF-W.

The wavelet transform used in Algorithm 2 was the Daubechies 2 (db2) wavelet. There was no solid reason for choosing this wavelet type. To show the effect of changing the type of the wavelet on the final result, various types of Daubechies wavelet were tested and the results were compared. Figure 7 shows the RMSE values obtained from DMD-KF-W by using various Daubechies wavelet types (the type None means no wavelet denoising was performed). As the figure shows, all wavelet types resulted in lower RMSE values than not performing wavelet denoising. But, even without wavelet denoising, DMD-KF-W resulted in lower RMSE values than TV and DFW. For this test case, the Daubechies wavelet type 4 resulted in the least RMSE value but there is no reason to generalize this conclusion.
The third test was based on a synthetic 4D-Flow MRI dataset consisting of 50 snapshots. As seen in Figure 4, for TV and DFW methods there are optimal sets of tuning parameters for which the least RMSE values are obtained whereas there is no systematic way of finding the optimal values of the tuning parameters when the reference dataset is not available. Figure 4 also shows the sensitivity of the TV method with respect to the weighting parameter $\lambda$. As seen, increasing the $\lambda$ value from 0.03 to 0.10, almost doubles the RMSE value. Compared to TV, the DFW method is less sensitive to the two parameters, as the figure shows. The impact of the smallest wavelet level size on the RMSE of DFW seems to be more significant than the number of cycles. For a given wavelet level size, changing the number of cycles slightly impacts the RMSE value, whereas increasing the wavelet level size from 14 to 15 results in almost 70% increase in the RMSE value.

The plot of per-snapshot RMSE values in Figure 5 reveals the effectiveness of DMD-KF-W in denoising the noisy dataset where it has resulted in lower reconstruction error in all snapshots. The total RMSE of DMD-KF-W is less than DFW's and TV's and less than one-third of the noisy dataset's (0.019, 0.037, 0.042, and 0.066, respectively). This test clearly shows the effectiveness of DMD-KF-W in denoising without having to set any parameters.

The last test was based on two actual 4D-Flow MRI in-vivo datasets. Due to the lack of the desired solution, it is not possible to find the RMSE values and make comparison based on them. By observing the reconstruction results, it seems DMD-KF-W has been able to reveal more fine features than TV and DFW. Since DMD-KF-W was shown to perform better than both TV and DFW for the synthetic test cases and also since it is a parameter-free method, we tend to believe it performed better for the last test case as well.

The proposed denoising method involves POD-based reduced-order modeling, forward-backward DMD, Kalman filtering-smoothing, and wavelet denoising. To justify the effectiveness of various components of DMD-KF-W method, its results are compared with the results obtained solely based on the POD vectors (POD), the results obtained by neglecting wavelet denoising (DMD-KF), and the results obtained by replacing forward-backward DMD with exact DMD (Exact DMD). Figure 8 shows the the ratio of the RMSEs of reconstructed datasets to the RMSE of the corresponding noisy input dataset in percent for the three synthetic test cases. The results of TV and DFW methods are also presented for comparison.
As the results show, in all synthetic cases studied, the proposed method has led to the lowest RMSE ratio. This figure shows replacing forward-backward DMD with simple DMD doesn’t have any significant effect on the results. As stated earlier, the eigenvalues computed by the simple DMD method are shifted in presence of noise. So, we tend to use forward-backward DMD instead of simple DMD.

![Figure 8](image_url)  
**Figure 8.** Evaluating the effectiveness of wavelet denoising step of the proposed method in improving the RMSE values for the synthetic test cases. The horizontal axis shows the test case number. The vertical axis shows the ratio of the RMSE of reconstruction to the RMSE of the noisy input dataset in percent. A lower ratio corresponds to better denoising.

In summary, DMD-KF-W outperforms TV and DFW in denoising a dynamic dataset. It runs quickly and is autonomous. Our method relies on fbDMD to approximate the dynamics of the given dataset. The fbDMD algorithm (Algorithm 1) involves finding the square root of a matrix. The square root of a matrix is not unique and so, picking the best square root remains an open question. If the dynamics are highly nonlinear, fbDMD will fail to identify them properly and so, the forward model derived from fbDMD will be inaccurate. This might affect the performance of our method. One possible workaround is to use a variant of DMD which takes into consideration the nonlinear dynamics as well (such as extended DMD or kernel-based DMD). It is also possible to use a more advanced variant of the Kalman filter such as extended Kalman filter or unscented Kalman filter.

5. **Conclusion.** In this paper, a new method for denoising time-resolved datasets was proposed. The proposed method is fast, parameter-free, and not computationally-intensive. It was compared against TV and DFW methods in four test cases. In the first three cases, where the noise-free datasets were available, our method was shown to perform better than TV and DFW methods in terms of the defined error metric.

**REFERENCES**

[1] E. Bostan, S. Lefkimmiatis, O. Vardoulis, N. Stergiopulos and M. Unser, Improved variational denoising of flow fields with application to phase-contrast MRI data, *IEEE Signal Processing Letters*, 22 (2015), 762–766.
[2] 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.

[3] Y. Chen, A. Wiesel and A. O. Hero, Shrinkage Estimation of High Dimensional Covariance Matrices, 2009 IEEE International Conference on Acoustics, Speech and Signal Processing, 2937–2940.

[4] R. R. Coifman and D. L. Donoho, Translation-invariant de-noising, in *Wavelets and Statistics*, Springer, (1995), 125–150.

[5] J. L. Crassidis and J. L. Junkins, *Optimal Estimation of Dynamic Systems*, Chapman and Hall/CRC, 2004.

[6] S. T. M. 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), 1–19.

[7] M. F. Fathi, A. Bakhshinejad, A. Baghaie and R. M. D’Souza, Dynamic denoising and gappy data reconstruction based on dynamic mode decomposition and discrete cosine transform, *Applied Sciences*, 8 (2018), 1515.

[8] M. F. Fathi, A. Bakhshinejad, A. Baghaie, D. Saloner, R. H. Sacho, V. L. Rayz and R. M. D’Souza, Denoising and spatial resolution enhancement of 4D flow MRI using proper orthogonal decomposition and Lasso regularization, *Computerized Medical Imaging and Graphics*, 70 (2018), 165–172.

[9] M. Gavish and D. L. Donoho, The optimal hard threshold for singular values is $4/\sqrt{3}$, *IEEE Trans. Inform. Theory*, 60 (2014), 5040–5053.

[10] A. Gelb, *Applied Optimal Estimation*, MIT press, 1974.

[11] G. V. Iungo, C. Santoni-Ortiz, M. Abkar, F. Porté-Agel, M. A. Rotea and S. Leonardi, Data-driven reduced order model for prediction of wind turbine wakes, *Journal of Physics: Conference Series*, 625.

[12] K. M. Johnson and M. Markl, Improved SNR in phase contrast velocimetry with five-point balanced flow encoding, *Magnetic Resonance in Medicine*, 63 (2010), 349–355.

[13] R. E. Kalman, A new approach to linear filtering and prediction problems, *Trans. ASME Ser. D. J. Basic Engr.*, 82 (1960), 35–45.

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

[15] T. Nonomura, H. Shibata and R. Takaki, Dynamic mode decomposition using a kalman filter for parameter estimation, *AIP Advances*, 8 (2018), 105106.

[16] T. Nonomura, H. Shibata and R. Takaki, Extended-kalman-filter-based dynamic mode decomposition for simultaneous system identification and denoising, *PloS one*, 14, e0209836.

[17] F. Ong, M. Uecker, U. Tariq, A. Hsiao, M. T. Alley, S. S. Vasanawala and M. Lustig, Robust 4D flow denoising using divergence-free wavelet transform, *Magnetic Resonance in Medicine*, 73 (2015), 828–842.

[18] H. E. Rauch, F. Tung and C. T. Striebel, Maximum likelihood estimates of linear dynamic systems, *AIAA J.*, 3 (1965), 1445–1450.

[19] 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.

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

[21] P. J. Schmid and J. Sesterhenn, Dynamic mode decomposition of numerical and experimental data, *Bulletin of the American Physical Society*, 53 (2008), 2008.

[22] 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), 391–421.

[23] E. Wan, Sigma-point filters: An overview with applications to integrated navigation and vision assisted control, in *2006 IEEE Nonlinear Statistical Signal Processing Workshop*, (2006), 201–202.

[24] K. Willcox and J. Peraire, Balanced model reduction via the proper orthogonal decomposition, *AIAA Journal*, 40 (2002), 2323–2330.

[25] M. O. Williams, C. W. Rowley and I. G. Kevrekidis, A kernel-based method for data-driven Koopman spectral analysis, *J. Comput. Dyn.*, 2 (2015), 247–265. arXiv:1411.2260.

[26] M. O. Williams, I. G. Kevrekidis and C. W. Rowley, A data-driven approximation of the Koopman operator: Extending dynamic mode decomposition, *J. Nonlinear Sci.*, 25 (2015), 1307–1346.
[27] M. J. Zimoñ, J. M. Reese and D. R. Emerson, A novel coupling of noise reduction algorithms for particle flow simulations, J. Comput. Phys., 321 (2016), 169–190.

Received for publication September 2019.

E-mail address: fathifi2@uwm.edu
E-mail address: abaghaie@nyit.edu
E-mail address: ali@vasognosis.com
E-mail address: rsacho@mcw.edu
E-mail address: dsouza@uwm.edu