I&EC research
Industrial & Engineering Chemistry Research

Article

Cite This: Ind. Eng. Chem. Res. 2019, 58, 5588−5601

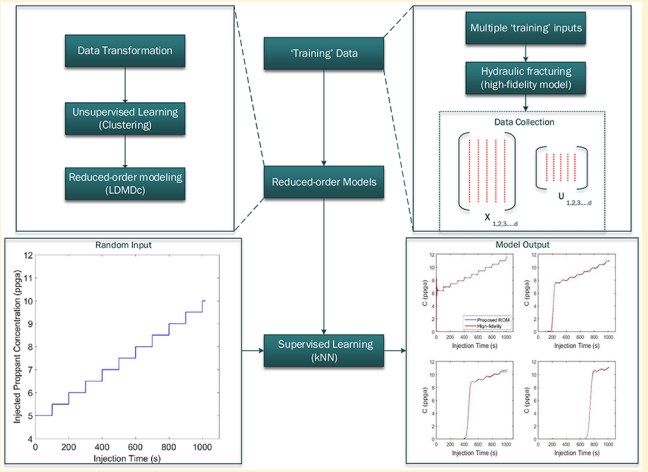
pubs.acs.org/IECR

Enlarging the Domain of Attraction of the Local Dynamic Mode Decomposition with Control Technique: Application to Hydraulic Fracturing

Mohammed Saad Faizan Bangi, Abhinav Narasingam, Prashanth Siddhamshetty, and Joseph Sang-Il Kwon*

Artie McFerrin Department of Chemical Engineering and Texas A&M Energy Institute, Texas A&M University, College Station, Texas 77843, United States

ABSTRACT: The local dynamic mode decomposition with control (LDMDc) technique combines the concept of unsupervised learning and the DMDc technique to extract the relevant local dynamics associated with highly nonlinear processes to build temporally local reduced-order models (ROMs). But the limited domain of attraction (DOA) of LDMDc hinders its widespread use in prediction. To systematically enlarge the DOA of the LDMDc technique, we utilize both the states of the system and the applied inputs from the data generated using multiple "training" inputs. We implement a clustering strategy to divide the data into clusters, use DMDc to build multiple local ROMs, and implement the *k*-nearest neighbors technique to make a selection among the set of ROMs during prediction. The proposed algorithm is applied to hydraulic fracturing to demonstrate the enlarged DOA of the LDMDc technique.



INTRODUCTION

Many chemical processes are usually represented by high-dimensional complex models which accurately describe the dynamics of the system, but their utility in the design of feedback control systems is limited due to the model complexity which puts considerable strain on the computational resources. Nonetheless, the solutions of such large-scale complex systems can be approximately explained by a very specific set of low-dimensional equations. For example, only three ordinary differential equations (ODEs) were required to represent the essential features of a laminar fluid flow passing a 2D cylinder.[1] Many model order reduction techniques are based on this idea and they have been widely used in industrially important engineering problems to deal with high-dimensional models without losing much accuracy. One reduced-order model (ROM) technique is network-based wherein complex chemical systems are divided into a network of small units, the characteristics of each unit can be defined by very few ordinary differential equations, and solving these equations for each unit would give a distribution of properties such as mass, energy, and momentum. More recently, this network-based ROM technique has been applied to gasifier which is represented as a network of ideal reactors consisting of plug flow reactors (PFRs) and continuously stirred tank reactors (CSTRs).[2−4] Two of the commonly used modal decomposition techniques in model order reduction methods are proper orthogonal decomposition (POD) and dynamic mode decomposition (DMD). Both of these techniques extract coherent structures within the system by analyzing sequential data obtained either by simulation of the high-fidelity model of the high-dimensional system or obtained via experimental studies. POD technique extracts structures that capture the most energy[5] and can be used to build a ROM for the system. The POD technique has been applied to build ROMs for various applications.[6−13] But using energy as a criterion for identifying these coherent structures is not always useful as it ignores those structures with zero-energy that are, however, dynamically relevant.[14]

DMD was initially introduced in the fluid community to extract flow structures by observing the high-dimensional data that can accurately represent the dynamics of the flow.[15] In comparison to POD, this method extracts those structures that are dynamically relevant and contribute toward the long-term dynamics of the system[16] rather than selecting those that carry the most energy.

Mathematically, DMD assumes that nonlinear systems with complex models can be represented using a linear form and this may seem inaccurate at first but understanding DMD as a numerical approximation of Koopman spectral analysis has

Received: December 3, 2018
Revised: February 3, 2019
Accepted: March 7, 2019
Published: March 7, 2019

DOI: 10.1021/acs.iecr.8b05995
Ind. Eng. Chem. Res. 2019, 58, 5588−5601

validated this representation.[17−19] DMD has been successfully applied to both numerical[20−24] and experimental[25−31] fluid flow data to represent relevant physical mechanisms in a linear form. Many works have been carried out regarding the numerics of the DMD algorithm which include the development of memory efficient algorithms,[32,33] a method for selection of a sparse basis of DMD modes,[34] and an error analysis of DMD growth rates.[31] Apart from this, theoretical works have been carried out to explore and understand its relationship with other methods such as Fourier analysis,[35] POD,[20] and Koopman spectral analysis.[17−19] Also, different methods have been proposed as variations of DMD such as optimized DMD[35] and optimal mode decomposition.[36,37]

Within this context, dynamic mode decomposition with control (DMDc), a purely data-driven modal decomposition technique, was developed to represent nonlinear systems, especially those whose dynamics are influenced by external inputs, in a discrete state-space form by extracting dynamically relevant spatial structures using both measurements of the system and the external inputs applied on it.[38] DMDc provides an understanding of the input-to-output behavior, which can be utilized to predict and design feedback control systems. However, for a highly nonlinear system, a global linear representation might not be a good approximation considering the fewer degrees of freedom associated with the linear model. Because of this limitation, the global method may fail to capture the effect of the changes in the process parameters such as permeability and Young's modulus of the rock formation on the local dynamics in the case of hydraulic fracturing as these constants are space-dependent. In order to better capture the local dynamics, temporal clustering can be integrated to DMDc to develop local ROMs that better represent the dynamics of the overall system, and this technique was introduced as LDMDc.[39] LDMDc divides the snapshot data into different clusters, obtains a linear operator pair for each cluster, and each pair represents the corresponding local ROM. The local ROM will approximately explain the dynamics of the system under the conditions in which the snapshots belonging to that particular cluster were obtained. It has been shown that LDMDc performs better than global DMDc when a single data set obtained under a particular operating condition is used to build the models.[39] However, the drawback of these models is that their domain of attraction (DOA) is limited by the data used for model training; in other words, these models will perform poorly when used for prediction under other operating conditions.

Our contribution in this work is to enlarge the DOA of LDMDc technique by implementing supervised and unsupervised learning techniques on multiple "training" data sets to build and utilize multiple local ROMs, respectively. These data sets are obtained under different operating conditions by performing simulations of the high-fidelity model. In order to obtain highly accurate local ROMs, we implement a particular clustering strategy instead of the conventional approaches available in the literature. The clustering strategy involves considering each "training" data set individually and clustering it using only the information on the inputs such that the optimal number of clusters are obtained along with the clustered output. The reason we opted for the clustering strategy is that it is easier to implement, and the resulting clustered output satisfies constraints required to build highly accurate ROMs which will be discussed later in this text. Using the clustered output, DMDc-based local ROMs are built and these ROMs will be used for prediction. During prediction, at any instance, the selection of a local ROM is accomplished by utilizing the *k*-nearest neighbors (*k*NN) classification technique. Another novelty is that in our proposed algorithm, we utilize both the states of the system and the external inputs applied on it which is necessary because the dynamics of the system are influenced by both the state and the applied external input. This aspect of our algorithm makes it different from the LDMDc technique proposed by Narasingam and Kwon[39] wherein only the states of the system were utilized to cluster the data. Also, in the LDMDc technique proposed by Narasingam and Kwon[39] selection of ROM was not necessary as only one "training" input was used to build the model. On the contrary, in our proposed algorithm any input profile satisfying an imposed constraint within the enlarged DOA can be utilized for prediction and this necessitates the use of the *k*NN classification technique. Despite the differences, our proposed technique still holds the advantages of the LDMDc technique proposed by Narasingam and Kwon[39] in that it is completely data-driven and captures local dynamics efficiently all while requiring no knowledge in terms of the system model.

The remainder of this text is organized as follows: In Local Dynamic Mode Decomposition with Control we provide the methodology proposed by Narasingam and Kwon[39] to build a LDMDc-based ROM. In Enlarging the DOA of Local DMDc we present our proposed algorithm to expand the DOA of the LDMDc technique. In Application to Hydraulic Fracturing we present the application of our proposed technique on the hydraulic fracturing system which includes a series of numerical simulation results.

LOCAL DYNAMIC MODE DECOMPOSITION WITH CONTROL

Recall, the technique of DMDc represents the underlying dynamics of a nonlinear system in a linear state-space form by utilizing both the measurements of the system and the external input applied on it. Mathematically, this would mean that the snapshots are related to each other by a linear operator pair. But considering that most of the systems are inherently nonlinear, this linear representation will not be accurate. To accurately represent a nonlinear system using DMDc, Narasingam and Kwon[39] proposed a framework to divide the snapshots into clusters wherein in each cluster its underlying local dynamics can be captured and be represented in a linear form by using DMDc on the snapshots within that cluster. To divide the snapshots into clusters, the global optimum search (GOS) algorithm was used, and the set of all local ROMs will together be used to describe the nonlinear system. This technique is discussed in detail below:

Temporal Clustering. The GOS algorithm aims to partition the snapshots into clusters by solving a mixed integer nonlinear programming (MINLP) optimization problem formulated to minimize the distance between the snapshots and the cluster centers.[40] In the case of time-series data, consecutive snapshots will be clustered together to an extent as the optimization problem uses euclidean distance as a metric. Suppose there are n snapshots in the generated data set which are to be partitioned into m clusters. Assuming that these snapshots are sampled at uniform time intervals, they can be represented as $\mathbf{x}_j$ for $j = 1, 2, 3, ..., n$ where n is the total number of snapshots. The MINLP problem is presented below:where i denotes the spatial points such that $i = 1, 2, 3, ..., s$ where s is

DOI: 10.1021/acs.iecr.8b05995
Ind. Eng. Chem. Res. 2019, 58, 5588−5601

$$
\begin{aligned}
&\underset{c_{ki},y_{jk}}{Minimize} && \sum_{i=1}^{s}\sum_{j=1}^{n} x_{ij}^2 - \sum_{i=1}^{s}\sum_{j=1}^{n}\sum_{k=1}^{m}(x_{ij}y_{jk}c_{ki}) \\
&\text{s.t} && c_{ki}\sum_{j=1}^{n} y_{jk} - \sum_{j=1}^{n} x_{ij}y_{jk} = 0, \quad \forall i,k \\
& && \sum_{k=1}^{m} y_{jk} = 1, \quad \forall j = 1,...,n \\
& && 1 \le \sum_{j=1}^{n} y_{jk} \le n-m+1 \\
& && y_{jk} \in \{0,1\}, \quad \forall j,k \\
& && c_{ki}^{L} \le c_{ki} \le c_{ki}^{U}, \quad \forall i,k \\
& && c_{ki}^{L} = min\{x_{ij}\}, \quad \forall i = 1,\ldots,s \\
& && c_{ki}^{U} = max\{x_{ij}\}, \quad \forall i = 1,\ldots,s
\end{aligned} \tag{1}
$$

the dimension of $\mathbf{x}_j$, y_{jk} are binary variables used to indicate whether a snapshot j lies in the kth cluster, c_{ki} are continuous variables which represent the cluster centers, and c_{ki}^{L}, c_{ki}^{U} denote the lower and upper bounds on the cluster center c_{ki}, respectively. The first constraint represents the necessary optimality condition which ensures that the vector distance sum of all the data points within a cluster to the cluster center is zero. The second constraint makes sure that each snapshot can lie in only one cluster. The third constraint makes sure that each cluster will contain at least one and no more than $(n - m + 1)$ snapshots in it. The clusters are obtained by solving the above formulated MINLP optimization problem. The optimal number of clusters is derived from the clustering balance curve[40] where the clustering balance, ε, is defined as

$$
\varepsilon = \frac{1}{2}\left(\sum_{i=1}^{s}\sum_{j=1}^{n}\sum_{k=1}^{m} y_{jk}\,\|x_{ij} - c_{ki}\|_2^2 + \sum_{i=1}^{s}\sum_{k=1}^{m}\|c_{ki} - c_i^{o}\|_2^2\right) \tag{2}
$$

where the first term is the intracluster error sum, and the second term is the intercluster error sum, and $c_i^{o} = \frac{1}{n}\sum_{j=1}^{n} x_{ij}$ is the global cluster center. The above-mentioned m clusters can be obtained by solving the MINLP problem and the resulting cluster configuration can be represented as

$$
C^k = \{\mathbf{x}_j | (\|\mathbf{x}_j - \mathbf{c}_k\|_2^2) \le (\|\mathbf{x}_j - \mathbf{c}_l\|_2^2),\ \forall\, l \ne k\} \tag{3}
$$

where C^k is the kth cluster and $\mathbf{c}_k$ is the center of the kth cluster. The optimal number of clusters is obtained from the clustering balance curve at the minimum value of ε.

Capturing Local Dynamics. DMDc is applied to each cluster using its corresponding snapshots to obtain a linear operator pair $\mathbf{A}$ and $\mathbf{B}$ to build a ROM that will capture the cluster's underlying local dynamics. *Algorithm* 1 describes how to apply DMDc to each cluster and obtain the pairs $(\mathbf{A}_j, \mathbf{B}_j)$ for every cluster in detail. The ROM to describe the dynamics of the system in each cluster can be formulated as follows:

$$
\mathbf{x}_{i+1} = \mathbf{A}_j \times \mathbf{x}_i + \mathbf{B}_j \times \mathbf{u}_i \tag{4}
$$

The above equation represents the system as a discrete-time linear state space model. Therefore, we recognize that DMDc can be used for system identification of a high dimensional, nonlinear system as a linear state-space model by capturing its underlying dynamics using the data.

Algorithm 1 DMDc for each cluster

1: Suppose the j^{th} cluster contains the states $\{\mathbf{x}_1\ \mathbf{x}_2\ \ldots\ \mathbf{x}_n\}_j$ and the corresponding inputs applied on them are $\{\mathbf{u}_1\ \mathbf{u}_2\ \ldots\ \mathbf{u}_n\}_j$. Arrange the data matrix $\{\mathbf{x}_1\ \mathbf{x}_2\ \ldots\ \mathbf{x}_n\}_j$ into matrices $\mathbf{X}$ and $\mathbf{Y}$ such that

$$
\mathbf{X} = \{\mathbf{x}_1\ \mathbf{x}_2\ \ldots\ \mathbf{x}_{n-1}\}_j,\ \mathbf{Y} = \{\mathbf{x}_2\ \mathbf{x}_3\ \ldots\ \mathbf{x}_n\}_j
$$

where $\mathbf{X} \in \mathbb{R}^{s\times(n-1)}$ and $\mathbf{Y} \in \mathbb{R}^{s\times(n-1)}$. The corresponding inputs for the states in $\mathbf{X}$ be $\mathbf{\Gamma}$ such that

$$
\mathbf{\Gamma} = \{\mathbf{u}_1\ \mathbf{u}_2\ \ldots\ \mathbf{u}_{n-1}\}_j
$$

where $\mathbf{\Gamma} \in \mathbb{R}^{l\times(n-1)}$

2: The state space form corresponding to this cluster can be defined as

$$
\mathbf{Y} = \mathbf{A}_j * \mathbf{X} + \mathbf{B}_j * \mathbf{\Gamma}
$$

where $\mathbf{A}_j \in \mathbb{R}^{s\times s}$ and $\mathbf{B}_j \in \mathbb{R}^{s\times l}$

3: The equation can be rewritten in an augmented form as

$$
\mathbf{Y} = [\mathbf{A}_j\ \mathbf{B}_j] * \begin{bmatrix}\mathbf{X}\\ \mathbf{\Gamma}\end{bmatrix} = \mathbf{G} * \mathbf{\Omega}
$$

where $\mathbf{G} \in \mathbb{R}^{s\times(s+l)}$ and $\mathbf{\Omega} \in \mathbb{R}^{(s+l)\times(n-1)}$

4: Compute the Singular Value Decomposition (SVD) of the augmented matrix $\mathbf{\Omega}$ as

$$
\mathbf{\Omega} = \hat{\mathbf{U}}\,\hat{\mathbf{\Sigma}}\,\hat{\mathbf{V}}^*
$$

5: Reduce the order of the augmented system $\mathbf{\Omega}$ from '$s + l$' to 'p' by selecting an appropriate tolerance limit for the singular values of $\mathbf{\Omega}$

6: Compute $\hat{\mathbf{U}}_1^* \in \mathbb{R}^{s\times p}$, and $\hat{\mathbf{U}}_2^* \in \mathbb{R}^{l\times p}$ such that

$$
\hat{\mathbf{U}} = \begin{bmatrix}\hat{\mathbf{U}}_1\\ \hat{\mathbf{U}}_2\end{bmatrix}
$$

7: Compute the SVD of the shifted snapshot sequence $\mathbf{Y}$ as

$$
\mathbf{Y} = \tilde{\mathbf{U}}\,\tilde{\mathbf{\Sigma}}\,\tilde{\mathbf{V}}^*
$$

8: Reduce the order of the subspace of $\mathbf{Y}$ from 's' to 'r' by selecting an appropriate tolerance limit for the singular values of $\mathbf{Y}$. Here, $\tilde{\mathbf{U}} \in \mathbb{R}^{s\times r}$, $\tilde{\mathbf{\Sigma}} \in \mathbb{R}^{r\times r}$, and $\tilde{\mathbf{V}} \in \mathbb{R}^{(n-1)\times r}$

9: Compute the low dimensional representation of the system matrices as

$$
\hat{\mathbf{A}}_j = \tilde{\mathbf{U}}^*\,\mathbf{Y}\,\hat{\mathbf{V}}\,\hat{\mathbf{\Sigma}}^{-1}\,\hat{\mathbf{U}}_1^*\,\tilde{\mathbf{U}} \qquad \hat{\mathbf{B}}_j = \tilde{\mathbf{U}}^*\,\mathbf{Y}\,\hat{\mathbf{V}}\,\hat{\mathbf{\Sigma}}^{-1}\,\hat{\mathbf{U}}_2^*
$$

where $\hat{\mathbf{A}}_j \in \mathbb{R}^{r\times r}$ and $\hat{\mathbf{B}}_j \in \mathbb{R}^{r\times l}$

10: Apply the inverse transformation to project the approximate system matrices, $\hat{\mathbf{A}}_j$ and $\hat{\mathbf{B}}_j$, in the r-dimensional subspace to the full order space

$$
\mathbf{A}_j = \tilde{\mathbf{U}}\,\hat{\mathbf{A}}_j\,\tilde{\mathbf{U}}^* \qquad \mathbf{B}_j = \tilde{\mathbf{U}}\,\hat{\mathbf{B}}_j
$$

where $\mathbf{A}_j \in \mathbb{R}^{s\times s}$ and $\mathbf{B}_j \in \mathbb{R}^{s\times l}$

Remark 1. *If the augmented system matrix is ill-conditioned, it is recommended to calculate its pseudoinverse in order to obtain the linear operators.*

ENLARGING THE DOA OF LOCAL DMDC

As mentioned previously, the DOA of the LDMDc technique proposed by Narasingam and Kwon[39] is narrow with respect to both the input and the state space, meaning that the model built using this technique can only reproduce accurately the "training" data when the "training" input is applied and applying any other input on the model will produce unsatisfactory results. In this work we use a variety of operating conditions to obtain the "training" data which would then be used to enlarge the DOA of LDMDc.

Suppose $\mathbb{X}_h$ is the trajectory followed by the state $x \in \mathbb{R}^s$ when an input $u \in \mathbb{R}^l$ is applied on the high fidelity model, and $\mathbb{X}_r$ is the trajectory followed by the state $x_r \in \mathbb{R}^s$ when the same input $u \in \mathbb{R}^l$ is applied on the reduced-order model obtained from the proposed algorithm. Then

$$
\begin{aligned}
&\mathbb{X}_h := \{x \in \mathbb{R}^s\colon \dot{x}(t) = f(x(t), u(t)) \\
&\quad \text{s.t. } x(0) = 0,\ u \in \mathbb{R}^l,\ \forall\ t \in [0, t_e]\} \\
&\mathbb{X}_r := \{x_r \in \mathbb{R}^s\colon x_r(k+1) = A_k \times x_r(k) + B_k \times u(k)) \\
&\quad \text{s.t. } x_r(0) = 0,\ A_k \in A,\ B_k \in B,\ u \in \mathbb{R}^l,\ \forall\ k \in [0, k_{t_e}]\}
\end{aligned} \tag{5}
$$

where f represents the high-fidelity model as a function of state x and input u, t_e is the end of fracturing time, and A_k and B_k are the linear operator pairs obtained by using the "training" data. The DOA $\mathbb{D}$ is then defined as

DOI: 10.1021/acs.iecr.8b05995
Ind. Eng. Chem. Res. 2019, 58, 5588−5601

$$\mathbb{D} := \{x, x_r \in \mathbb{R}^s : |x - x_r| < \varepsilon\} \quad (6)$$

where ε is the error. In other words, the DOA is the set of all states obtained from the reduced-order models which can satisfactorily describe the system under well-defined conditions.

Data Generation. The "training" data can be obtained by implementing various input profiles on the system either on an experimental basis or by carrying out simulations of the high fidelity model. In this work we choose the latter option for data generation. An important point to remember here is that it is very crucial to identify the DOA for which the model is intended to be built, and to select inputs within this region. To identify this region, it is necessary to understand the application of the ROM. One of the important applications of ROMs is in the design of controllers and one of the most widely used control techniques in these days is an optimization-based control scheme to obtain the optimal control action. Therefore, it would be ideal to select a region which would contain the solution (optimal control action) to the optimization (control) problem. Also, these multiple inputs need to be spread all across this finite region to make sure that the "training" data is "rich". Consequently, the resultant model will be able to accurately predict for any input under a constraint, which will be discussed later, within this finite region. Finally, the number of "training" inputs to be used is up to the discretion of the users. Large amounts of data would definitely improve the accuracy of the model but this would come at the cost of high computational expenses. Having a priori knowledge of the system, and an understanding of the application of the model will help in deciding the number of "training" inputs necessary to build the model. To summarize, the following guidelines should be taken into account when defining the "training" inputs.

1. To identify the region, information from the existing literature/experimental studies must be considered along with other system and practical constraints.
2. The selected "training" inputs should cover the entire identified input region to maximize the predictive capability of the reduced-order model.
3. To reduce the computational expenses, the "training" inputs should be unique but within the defined region.

Once the "training" inputs have been identified, "training" data sets can be generated by performing numerical simulations of the high-fidelity model. Assuming that "d" "training" data sets are generated and each data set contains "n" snapshots, the nomenclature used to represent the state and input matrices are shown in Table 1.

Remark 2. *In the case of experimental data corrupted with noise and unmeasured disturbances, it is advised to denoise the data using filters and then use the proposed algorithm to build the ROMs in order to be used in the design of feedback controllers. When obtaining "training" data in the case of unstable systems, it is recommended that closed-loop identification be utilized in unstable regions.*

Temporal Clustering. Recall that the LDMDc technique divides the generated snapshots of data temporally into clusters. The GOS algorithm was used to partition the snapshots into clusters and it was sufficient to use only the state vectors as only one "training" input profile was used to obtain the local ROMs via Local DMDc.[39] But in this work, since multiple "training" inputs are considered, we propose to use both the state vector and the input by stacking them vertically to form an "augmented" vector, and these "augmented" vectors will now constitute our "training" data sets. Before applying any clustering technique, it is essential to normalize the data as the components of the "augmented" vector operate in two different spaces (i.e., the state space and the input space) and the range of each component varies with the other. To perform normalization, we first concatenate all the "augmented" vectors horizontally to form one "combined" data matrix. To further understand this "combined" data matrix, its rows represent the components of the "augmented" vector and its columns represent the time instances. Each row of the "combined" data matrix must be normalized individually across all the columns of the "combined" data matrix. The nomenclatures used to represent above-mentioned matrices are shown in Table 1. In Algorithm 1, the state vectors and the input vectors have been defined to contain s and l components, respectively. As a result, the "augmented" vector contains $s + l$ components. Therefore, the normalization process is repeated for the entire $s + l$ rows in the "combined" data matrix.

Table 1. Nomenclature

term	mathematical description
states	$\mathbf{X}^d = \{\mathbf{x}_1 \ \mathbf{x}_2 \ \dots \ \mathbf{x}_n\}_d$
inputs	$\mathbf{U}^d = \{\mathbf{u}_1 \ \mathbf{u}_2 \ \dots \ \mathbf{u}_n\}_d$
augmented matrix	$\Omega^d = \begin{bmatrix} \mathbf{X}^d \\ \mathbf{U}^d \end{bmatrix}$
combined matrix	$\Omega = [\Omega^1 \ \Omega^2 \ \dots \ \Omega^d] = \begin{bmatrix} \mathbf{X}^1 & \mathbf{X}^2 & \dots & \mathbf{X}^d \\ \mathbf{U}^1 & \mathbf{U}^2 & \dots & \mathbf{U}^d \end{bmatrix}$
normalized matrix	$\Omega_N = [\Omega_N^1 \ \Omega_N^2 \ \dots \ \Omega_N^d] = \begin{bmatrix} [\mathbf{X}^1 & \mathbf{X}^2 & \dots & \mathbf{X}^d]_N \\ [\mathbf{U}^1 & \mathbf{U}^2 & \dots & \mathbf{U}^d]_N \end{bmatrix}$
weighted matrix	$\Omega_w = [\mathbf{w}_x \ \mathbf{w}_u] * \begin{bmatrix} [\mathbf{X}^1 & \mathbf{X}^2 & \dots & \mathbf{X}^d]_N \\ [\mathbf{U}^1 & \mathbf{U}^2 & \dots & \mathbf{U}^d]_N \end{bmatrix}$

A point to consider here is the dimensions of the state vector and the input vector. It is usually the case where the dimension of the state vector is much larger compared to the dimension of the input vector. Furthermore, considering that clustering algorithms typically use "distance" as a metric based on which snapshots are divided into clusters, it is quite possible that the contribution of the state vector toward this metric might numerically outweigh the contribution of the input vector. To overcome this imbalance, we propose to apply two weights on the "augmented" vector (Table 1), wherein one weight is equally divided among all the components of the "augmented" vector that represent the state vector, and similarly, the other weight is applied on the input component of the "augmented" vector. Numerically, these weights have to be ascertained by trial and error as they depend on the system, and the type of "training" inputs used to generate "training" data to build the model. We then perform principal component analysis (PCA) on the weighted matrix. PCA helps in reducing the number of dimensions that we have to deal with when clustering the data, validating the model, and using the model for prediction. Applying PCA transforms the weighted matrix into its PCA scores (PCSs) which represent the data in the principal component space, and we will only use those scores whose corresponding components can together be used to represent at least 90% of the variance in the data. Now, we have the

DOI: 10.1021/acs.iecr.8b05995
Ind. Eng. Chem. Res. 2019, 58, 5588−5601

necessary transformed data to implement the clustering strategy.

The clustering strategy to be implemented is highly dependent on the "training" data used, and the application in which the resultant model will be used. Nonetheless, it should satisfy the following output criteria: (a) no two data points from two different "training" data sets will lie in the same cluster, (b) no two data points with different inputs will lie in the same cluster. The reasons for the above criteria are that when snapshots belonging to different "training" data sets, or belonging to the same "training" data set but having different inputs are kept in the same cluster, the resultant operator pair (A, B) will capture the local dynamics inaccurately for that cluster and this linear operator pair will give inaccurate results when used for prediction. Conventional clustering techniques can be applied but it would be much easier to satisfy the above-mentioned criteria by implementing a clustering strategy which involves considering each "training" data set individually and clustering based on the inputs such that the optimal number of clusters is obtained along with the clustered output. Once clustering is done, the cluster centers can be calculated in terms of the PCSs by calculating the average of all the PCSs of the data points within each cluster, and these centers will be used in the selection of the local ROM which is explained in the section below.

Remark 3. *The above transformation of the "training" data was performed to aid the algorithm—considering the fact that we need to utilize both the state and the input in implementing a clustering strategy, and in selecting the correct ROM during model validation/model prediction. To build the local ROMs, we will use the "training" data obtained from the simulations of the high fidelity model.*

Local ROM Selection. After building a ROM for each cluster, we use this set of local ROMs for validation and for model prediction. In both the cases we adopt the same approach to select the appropriate local ROM. Recall, at a given time instance, the future state of the system is dependent on both the current state of the system and the input to be applied on it. Hence, we will use both the information in the selection of the appropriate ROM.

At this stage it is important to understand that each local ROM is developed for a cluster of state vectors and their corresponding inputs. At any time instance, given the state and the input, we need to find a snapshot in the "combined" matrix whose state vector and the input closely match with the ones in consideration. Next, we locate the cluster in which this selected snapshot belongs to and use that particular cluster's ROM to predict the future state for an applied input profile. But finding the closest snapshot in the "combined" matrix is a computationally expensive task considering the huge amounts of data in use. Instead it would be much easier to find the nearest cluster center to both the state vector and the input in consideration and use the corresponding local ROM to predict the future state trajectory.

Considering that the state vector and the input operate in two different spaces, it is difficult to make a selection of which ROM to use without an appropriate transformation. To overcome this, we apply the transformation similar to the one used in the clustering step of our algorithm. We first stack them vertically to form the "augmented" vector, normalize each row of the "augmented" vector using the corresponding mean and variance of that row in the "combined" matrix obtained in the clustering step, and apply weights on the state vector and the input in the exact manner as done in the clustering step. Recall, in the clustering step, we perform PCA on the weighted matrix to reduce the number of dimensions we have to deal with in various steps of our algorithm, which includes the selection of the local ROM. Considering that we transformed the "training" data into PCSs of its dominant PCA components in the clustering step, we similarly calculate the PCS of the weighted vector in consideration by using the same dominant PCA components. Now the transformation of the state vector and the input in consideration is complete and the above calculated PCSs will be further used in the ROM selection step.

We use *k*NN technique to select the appropriate local ROM. Recall, *k*NN technique selects "*k*" points from a data set that are closest to the query point with respect to the Euclidean distance metric. In our method the above calculated dominant PCSs of the weighted vector in consideration is the query point. The set of cluster centers given to the *k*NN technique as the input data set will not comprise all the cluster centers. Using all the cluster centers will result in the incorrect selection of the local ROM because it is entirely possible that there exists two "augmented" vectors whose states and inputs are dissimilar respectively but may have approximately the same PCSs. To avoid such scenarios we include the following constraint in our algorithm: at the given time instance, those cluster centers are selected in the subset of cluster centers to the *k*NN technique whose respective clusters contain the snapshot having the same time instance. Implementing the *k*NN technique with the above constraint will help the algorithm in selecting the correct local ROM. Given a $(\mathbf{x}_i, \mathbf{u}_i)$ at the time instance $\mathbf{t}_i$, the right pair of $(\mathbf{A}_i, \mathbf{B}_i)$ can be selected and the next state of the system can be calculated as defined in eq 4.

The proposed methodology is summarized and presented below as *Algorithm* 2.

Remark 4. *In our proposed algorithm, we transform the "training" data set into its PCA scores and reduce the dimensions of the "combined" matrix by selecting only few of the principal components. The resulting dimension depends on the "training" data used, dimensionality of the system, and the weights used. It is not possible to predict this beforehand. The amount of data to be used should be carefully identified considering the above-mentioned parameters in order to avoid any computational issues considering that this proposed method is data-based.*

APPLICATION TO HYDRAULIC FRACTURING

Shale gas is natural gas trapped within rocks of low porosity and low permeability, and hydraulic fracturing is a technique to obtain shale gas by stimulation of such rocks by controlled explosions along the length of the wellbore resulting in the formation of fractures. A clean fluid called pad is then introduced inside the wellbore at high pressures to extend the length of the initial fractures. A fracturing fluid containing water, proppant, and additives is then introduced to further extend the fractures. Once pumping is stopped, the remaining fluid is allowed to leak off into the reservoir resulting in the formation of a medium of proppant in the fractures. The natural stresses in the rocks cause the closure of fractures, thereby, trapping the proppant which would then act as a conductive medium for the extraction of the gas present in the reservoir. Two control objectives usually associated with hydraulic fracturing during proppant injection is to obtain uniform proppant concentration throughout the length of the fracture and to obtain a desired fracture geometry. In this

DOI: 10.1021/acs.iecr.8b05995
Ind. Eng. Chem. Res. 2019, 58, 5588−5601

Algorithm 2 Proposed methodology for enlarging the DOA of LDMDc

1: Say the d^{th} data set contains the state matrix $\mathbf{X}^d = \{\mathbf{x}_1\ \mathbf{x}_2\ \dots\ \mathbf{x}_n\}_d$ and the corresponding inputs applied on them are $\mathbf{U}^d = \{\mathbf{u}_1\ \mathbf{u}_2\ \dots\ \mathbf{u}_n\}_d$, respectively.
2: For every data set, stack the state and input matrices to construct the 'augmented' matrix $\mathbf{\Omega}^d$

$$\mathbf{\Omega}^d = \begin{bmatrix} \mathbf{X}^d \\ \mathbf{U}^d \end{bmatrix}$$

3: Form the 'combined' matrix $\mathbf{\Omega}$ by stacking each 'augmented' matrix horizontally

$$\mathbf{\Omega} = \{\mathbf{\Omega}^1\ \mathbf{\Omega}^2\ \dots\ \mathbf{\Omega}^d\}$$

4: Normalize each row of the matrix $\mathbf{\Omega}$ to form the 'normalized' matrix $\mathbf{\Omega}_N$

$$\mathbf{\Omega}_N = \{\mathbf{\Omega}_N^1\ \mathbf{\Omega}_N^2\ \dots\ \mathbf{\Omega}_N^d\} = \begin{bmatrix} \mathbf{X}_N^1\ \mathbf{X}_N^2\ \dots\ \mathbf{X}_N^d \\ \mathbf{U}_N^1\ \mathbf{U}_N^2\ \dots\ \mathbf{U}_N^d \end{bmatrix}$$

5: Apply weights $\mathbf{w_x}$ and $\mathbf{w_u}$ on the 'normalized' matrix $\mathbf{\Omega}_N$ such that $\mathbf{w_x}$ and $\mathbf{w_u}$ will be equally divided among the components of the state vector and input vector, respectively. The 'weighted' matrix is $\mathbf{\Omega}_w$

$$\mathbf{\Omega}_w = [\mathbf{w}_x\ \mathbf{w}_u] \begin{bmatrix} \mathbf{X}_N^1\ \mathbf{X}_N^2\ \dots\ \mathbf{X}_N^d \\ \mathbf{U}_N^1\ \mathbf{U}_N^2\ \dots\ \mathbf{U}_N^d \end{bmatrix} = \begin{bmatrix} \mathbf{\Omega}_w^1\ \mathbf{\Omega}_w^2\ \dots\ \mathbf{\Omega}_w^d \end{bmatrix}$$

6: Perform PCA on the 'weighted' matrix $\mathbf{\Omega}_w$ and select the first 'p' Principal Components such that the cumulative sum of their Principal Component Variances is at least 90%.
7: Transform the data in the matrix $\mathbf{\Omega}_w$ to their corresponding PCSs. As we are only considering the first 'p' Principal Components, the dimension of the matrix $\mathbf{\Omega}_w$ reduces to 'p'.

$$\mathbf{PCA}(\mathbf{\Omega}_w) = \begin{bmatrix} \mathbf{PCS_1}(\mathbf{\Omega}_w^1)\ \mathbf{PCS_1}(\mathbf{\Omega}_w^2)\ \dots\ \mathbf{PCS_1}(\mathbf{\Omega}_w^d) \\ \mathbf{PCS_2}(\mathbf{\Omega}_w^1)\ \mathbf{PCS_2}(\mathbf{\Omega}_w^2)\ \dots\ \mathbf{PCS_2}(\mathbf{\Omega}_w^d) \\ \vdots\ \ \vdots\ \ \vdots \\ \mathbf{PCS_p}(\mathbf{\Omega}_w^1)\ \mathbf{PCS_p}(\mathbf{\Omega}_w^2)\ \dots\ \mathbf{PCS_p}(\mathbf{\Omega}_w^d) \end{bmatrix}$$

8: Cluster the matrix $\mathbf{PCA}(\mathbf{\Omega}_w)$ using any clustering technique such that the clustering output satisfies the criteria mentioned in the clustering subsection. Obtain the cluster centers in terms of the average PCSs of the snapshots within the respective clusters.
9: For every cluster j, obtain the corresponding linear operator pair $(\mathbf{A}_j, \mathbf{B}_j)$ using the orignial 'training' data of snapshots within that cluster.
10: To select the correct local ROM, implement the kNN technique with k value as 1 as only one local ROM is required to calculate the next state of the system. A subset of cluster centers is selected as explained previously. Also the query point is transformed as described in Steps 2 to 5, its PCSs will be obtained using the same 'p' PCA components obtained in Steps 6 to 7, and then will be used in the kNN technique.

section we applied our proposed methodology to build a LDMDc-based ROM which can be used to predict the proppant concentration at various locations of the fracture at various times of the proppant injection process for a wide range of proppant pumping schedules.

Dynamic Modeling of Hydraulic Fracturing Process. Hydraulic Fracturing can be classified into three subprocesses which are as follows: (1) fracture propagation, (2) proppant transport, and (3) proppant bank formation.

Fracture Propagation. The fracture propagation is assumed to follow the Perkins, Kern, and Nordgren (PKN) model[41,42] which is shown in Figure 1. The other assumptions considered with regard to the fracture propagation are as follows: (1) the fracture length is much greater than its width, and hence, the fluid pressure along the vertical direction remains constant; (2) large stresses in the rock layers above and below the fracture result in the fracture being confined to a single layer; and (3) the rock properties such as Young's modulus and Poisson's ratio remain constant with respect to both time and space and the fracturing fluid is incompressible. Considering the above assumptions, it must be noted that the fracture will take an elliptical shape and its cross-sectional area will be rectangular. Fluid momentum is explained using the lubrication theory which relates the fluid flow-rate in the horizontal direction, q_z, to the sustained pressure gradient, $-\frac{\partial P}{\partial z}\hat{z}$, as follows:

$$q_z = -\frac{\pi H W^3}{64\mu}\frac{\partial P}{\partial z} \tag{7}$$

where P is the net pressure varying with the z coordinate, H is the fracture height, W is the width of the fracture, and μ is the fracturing fluid viscosity. The maximum width of the fracture

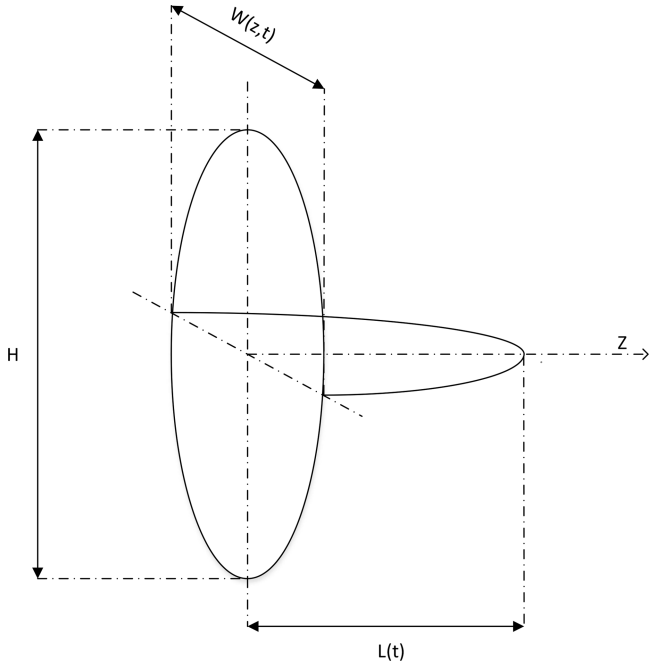


Figure 1. PKN fracture model.

can be related to the net pressure exerted by the fracturing fluids as follows:

$$W = \frac{2PH(1-\nu^2)}{E} \tag{8}$$

where E is the Young's modulus and ν is the Poisson's ratio of the formation. The continuity equation obtained by local mass conservation of an incompressible fluid is given by

$$\frac{\partial A}{\partial t} + \frac{\partial q_z}{\partial z} + HU = 0 \tag{9}$$

where $A = \pi WH/4$ is the cross-sectional area of the fracture, t is the time elapsed since the beginning of the fracturing process, z is the time-dependent spatial coordinate in the horizontal direction, and U is the fluid leak off rate per unit height into the reservoir. The fluid leak off rate is in the orthogonal direction to the fracture plane and is given by[43,44]

$$U = \frac{2C_{\text{leak}}}{\sqrt{t-\tau(z)}} \tag{10}$$

where C_{leak} is the overall leak off coefficient and $\tau(z)$ is the time instance at which the fracturing fluid reached the coordinate z for the first time. Plugging eqs 7−8 into eq 9 results in the following partial differential equation:

$$\frac{\pi H}{4}\frac{\partial W}{\partial t} - \frac{\pi E}{128\mu(1-\nu^2)}\left[3W^2\left(\frac{\partial W}{\partial z}\right)^2 + W^3\frac{\partial^2 W}{\partial z^2}\right] + HU = 0 \tag{11}$$

The two boundary conditions and an initial condition for the process are formulated as follows:[45,46]

$$q_z(0,t) = Q_0 \quad W(L(t),t) = 0 \tag{12}$$

$$W(z,0) = 0 \tag{13}$$

where Q_0 is the fluid injection rate at the wellbore and $L(t)$ is the fracture tip varying with time.

DOI: 10.1021/acs.iecr.8b05995
Ind. Eng. Chem. Res. 2019, 58, 5588−5601

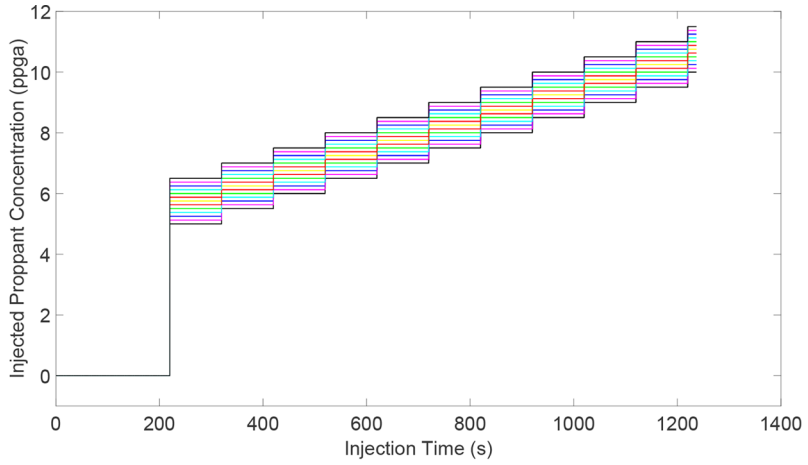


Figure 2. Different "training" input profiles used to generate open-loop simulation data for model training.

Proppant Transport. In this model, the proppant is assumed to travel with the superficial velocity of the fracturing fluid in the horizontal direction, and it is assumed to travel with the settling velocity relative to the fracturing fluid in the vertical direction due to the effect of gravity. The other assumptions adopted are as follows: (1) the proppant particle size is assumed to be large enough to neglect the diffusive flux while only convective flux is taken into consideration; (2) the interactions between the proppant particles are neglected while only drag and gravity effects are considered; and (3) the proppant particles have a uniform size. Based on these assumptions, the advection of proppant in the z direction can be computed as

$$\frac{\partial(WC)}{\partial t} + \frac{\partial}{\partial z}(WCV_p) = 0 \tag{14}$$

$$C(0, t) = C_0(t) \quad \text{and} \quad C(z, 0) = 0 \tag{15}$$

where $C(z, t)$ is the suspended proppant concentration at the coordinate z, and at time t. $C_0(t)$ is the proppant concentration injected at the wellbore. V_p is the net velocity of the proppant particles and is obtained by[47]

$$V_p = V - (1 - C)V_s \tag{16}$$

where V is the superficial fluid velocity in the horizontal direction and V_s is the gravitational settling velocity which can be computed as[48]

$$V_s = \frac{(1 - C)^2(\rho_{sd} - \rho_f)gd^2}{10^{1.82C}18\mu} \tag{17}$$

where ρ_{sd} is the proppant particle density, ρ_f is the pure fluid density, d is the proppant particle diameter, g is the gravitational constant, and μ is the fracture fluid viscosity which can be related to the proppant concentration as follows:[49]

$$\mu(C) = \mu_0\left(1 - \frac{C}{C_{\max}}\right)^{-\alpha} \tag{18}$$

where μ_0 is the pure fluid viscosity, $C_{\max}$ is the maximum theoretical concentration determined by $C_{\max} = (1 - \phi)\rho_{sd}$ where ϕ is the proppant bank porosity, and α is an exponent in the range of 1.2−1.8.

Proppant Bank Formation. The proppant settling results in the formation of a proppant bank and the variation of the bank height, δ, can be explained using the following equations:[45,50]

$$\frac{d(\delta W)}{dt} = \frac{CV_sW}{(1 - \phi)} \tag{19}$$

$$\delta(z, 0) = 0 \tag{20}$$

where eq 20 is the initial condition for eq 19.

Numerical Simulations. In this section, we solve the dynamic model of the hydraulic fracturing process for various input profiles in the selected finite region of the input space. A numerical scheme is adopted considering the highly nonlinear nature of the model and the moving boundary of the system.[51] We used fixed mesh strategy to solve the high-fidelity model, and the numerical scheme used is as follows:[52]

1. At time step t_k, the fracture tip is elongated by Δz to obtain the fracture length $L(t_{k+1})$.
2. The coupled equations of eqs 7−19 are solved together to obtain suspended proppant concentration $C(z, t_{k+1})$, fracture width $W(z, t_{k+1})$, proppant bank height $\delta(z, t_{k+1})$, settling velocity $V_s(z, t_{k+1})$, flow rate $Q(z, t_{k+1})$, and net pressure $P(z, t_{k+1})$ across the fracture using a finite element method.
3. Iteratively solve for $\tau(z_{k+1})$ in eqs 9 and 10 by repeating steps 2 and 3.
4. The time interval Δt_k is obtained. To handle computational efficiency, the numerical integration time step is adopted based on the Courant−Friedrichs−Lewy (CFL) condition which is $\frac{u\Delta t}{\Delta z} \leq 1$, where u is the fracture propagation speed in the width direction.
5. Set $k \leftarrow k + 1$ and go to step 1.

The various parameters used in our process calculations are as follows:[45] $H = 20$ m, $\mu = 0.56$ Pa·s, $E = 5 \times 10^3$ MPa, $\nu = 0.2$, and $C_{\text{leak}} = 6.3 \times 10^{-5}$ m/s$^{1/2}$. A constant flow rate of $Q_0 = 0.03$ m^3/s was utilized throughout the fracturing process. A finite region of the input space was selected in which a total of 13 distinct "training" input profiles were chosen as shown in Figure 2. Note that the proppant injection was started at $t = 220$ s. The reason for this design of the "training" input profiles is to closely imitate the practically viable inlet proppant concentration in the field which is usually an increasing staircase profile. The step increases have been kept constant at

DOI: 10.1021/acs.iecr.8b05995
Ind. Eng. Chem. Res. 2019, 58, 5588−5601

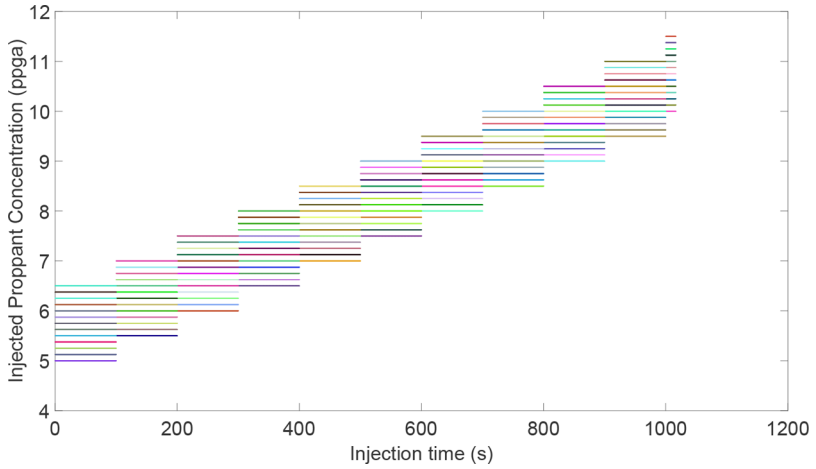


Figure 3. Clustering output representation in the input space.

0.5 ppga in all the cases. Another reason for this kind of pattern is to make sure that we cover the entire finite region so as to obtain rich "training" data sets and the amount of data used in model building would be optimal. Random input profiles can be considered within this region but the number of "training" inputs required to cover the entire region would be larger. Also, to avoid using many "training" inputs and to cover the entire region would require the step increase to be greater than 0.5 ppga, which is usually not practical to be implemented in the field.

Each "training" input profile is implemented on the open-loop system, and the corresponding response of the system is obtained by solving the high fidelity model. The simulations are carried out for t_f = 1236.4 s which resulted in a total of 12365 snapshots in each "training" data set. The high-order discretization scheme resulted in a total of 501 spatial points across the length of the fracture out of which only 101 were selected at equidistant points. The same discretization scheme was applied to the simulations of all the "training" input profiles. The "training" data obtained in these simulations were used in building ROMs through LDMDc.

Building LDMDc-Based ROMs. Our algorithm can be divided into three sections: (1) temporal clustering, (2) building ROMs for each cluster, and (3) model selection for validation/prediction. Note that only the data after the proppant injection began was used in this work.

Temporal Clustering. We first built the "augmented" vectors by stacking the state vector with its corresponding input vertically, formed the "combined" matrix by stacking horizontally all the "augmented" vectors from all the "training" data sets, normalized each row of the "combined" matrix, and applied weights on the resultant "normalized" matrix. In this work we applied equal weights [0.5, 0.5] on the state vector and the input. And, the weight 0.5 on the state vector was equally divided among the components of the state vector whereas the weight on the input was kept the same. The reason we chose this weights is that both the current state of the system and the input applied on it are equally important in propelling the system forward. In other applications it is possible that this may not be the case, and therefore, we suggest that a trial and error scheme needs to be adopted to obtain these weights. We performed PCA on weighted matrix and found that the 1st principal component (PC) was able to capture 99.59% of the total variance. Therefore, it was sufficient for us to just use the 1st PCS of all the data points in the clustering step as well as in the model selection step of the algorithm. The clustering strategy was implemented that satisfies the criteria mentioned previously; that is, no two data points from two different "training" data sets will lie in the same cluster, and no two data points having the same input will lie in the same cluster. We obtained a total of 143 clusters and the output of this clustering strategy in the input space is shown in Figure 3 wherein each color represents a cluster. The cluster centers were computed in terms of the 1st PCS and stored to be used in the local ROM selection step of the algorithm.

Remark 5. *The clustering criteria stated in the proposed algorithm is suited for systems and applications with step input profile. In the cases where nonstep input profiles are used, the clustering criteria need to be modified such that each cluster better captures local dynamics of the system.*

Building ROMs for Each Cluster. We applied the DMDc method to every cluster wherein we set the tolerance limit on the singular values as 1 to determine the corresponding p and r values for the purpose of model order reduction. For each cluster, we obtained a pair of linear operators, $(\mathbf{A}_j, \mathbf{B}_j)$, that captures the underlying local dynamics exhibited by the snapshots of that particular cluster. This pair of linear operators is then used to build the linear state-space model which will be then used for model validation and for prediction in the case of random inputs selected within the finite region.

Model Selection for Validation/Prediction. During model validation or model prediction, at any time step, a local ROM needs to be selected based on the current state of the system and the input to be applied on it to further propagate the system. To achieve this objective, we implemented the kNN technique by setting the k value as 1 as we only need to select one local ROM. To obtain the subset of cluster centers, we implemented the constraint that at any time step only those cluster centers will be used in the selection process whose corresponding clusters contain the snapshot which was obtained at the same time instance during the open-loop data generation process. Also, at every time step, the query points (i.e., the current state and the input) were transformed in the exact manner adopted in the clustering step. Recall, the parameter used in the selection process is the PCSs with respect to the dominant components. In this work, the 1st PCSs of the subset of cluster centers, and of the query point

DOI: 10.1021/acs.iecr.8b05995
Ind. Eng. Chem. Res. 2019, 58, 5588−5601

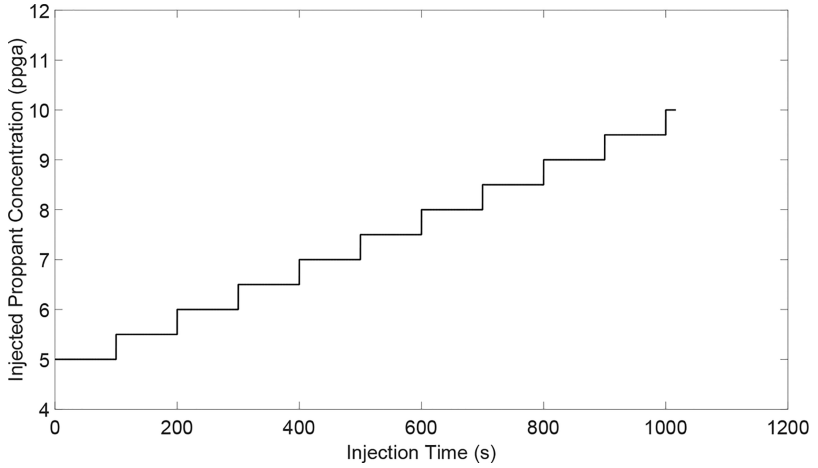
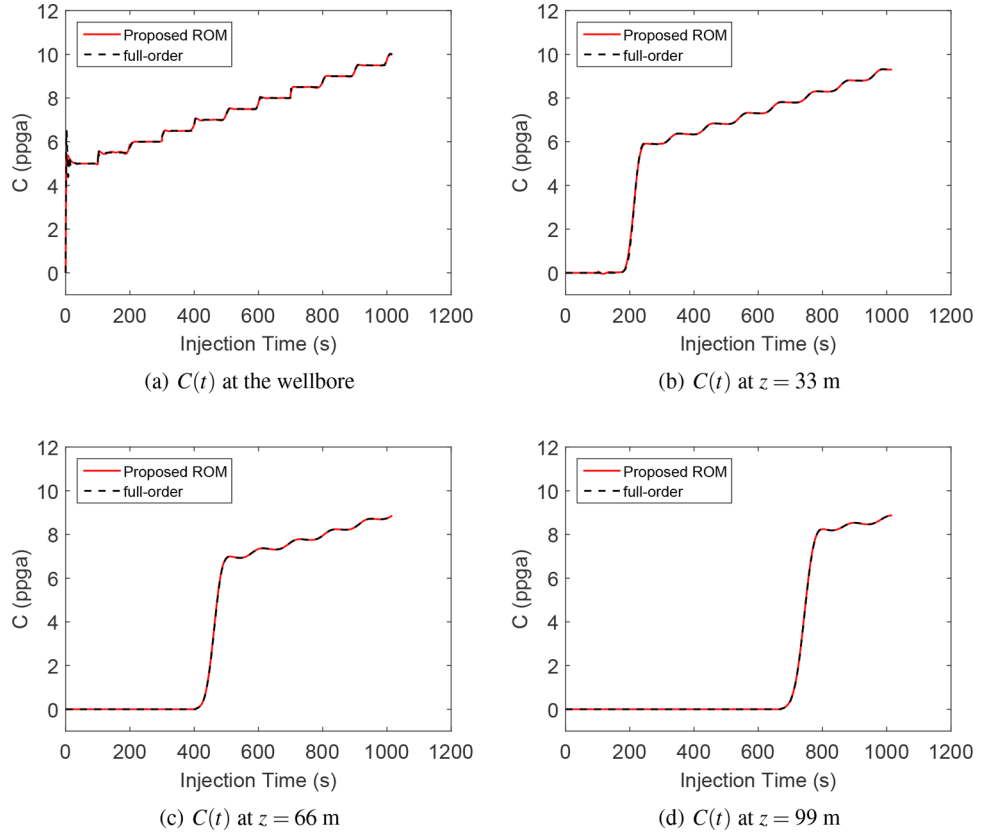


Figure 4. Validation input.



(a) $C(t)$ at the wellbore (b) $C(t)$ at $z = 33$ m

(c) $C(t)$ at $z = 66$ m (d) $C(t)$ at $z = 99$ m

Figure 5. Comparison of the approximate solution computed using LDMDc-based ROMs with the full-order solution.

were used as the 1st PC was able to capture 99.59% of the total variance in the data.

Model Validation. To verify the accuracy of our proposed methodology, we implemented one of the "training" inputs which is shown in Figure 4. We utilized the developed LDMDc-based ROMs to compute the output for the selected "training" input and it is compared against the output of the full-order model. Figure 5 shows the comparison of these two models with respect to the evolution of the proppant concentration at four different locations during the injection process. It can be seen that the output obtained using the proposed algorithm mimics the output from the full-order model. We used a relative error metric, $E(t)$, to quantify the performance of our proposed methodology in comparison to the full-order solution. The relative error is calculated by using the Frobenius norms of the state vectors as follows:

$$E(t) = \frac{\| \mathbf{x}_{\text{full}} - \mathbf{x}_{\text{rom}} \|_{\text{fro}}}{\| \mathbf{x}_{\text{full}} \|_{\text{fro}}} \tag{21}$$

where $\|.\|_{\text{fro}}$ is the Frobenius norm, $\mathbf{x}_{\text{full}}$ is the state vector obtained from the full-order solution, and $\mathbf{x}_{\text{rom}}$ is the state vector obtained from a ROM developed by the proposed methodology. The relative error for the approximate solution obtained from our proposed methodology is presented in Figure 6. From the plot we observe that the proposed methodology is able to provide an accurate approximation

DOI: 10.1021/acs.iecr.8b05995
Ind. Eng. Chem. Res. 2019, 58, 5588−5601

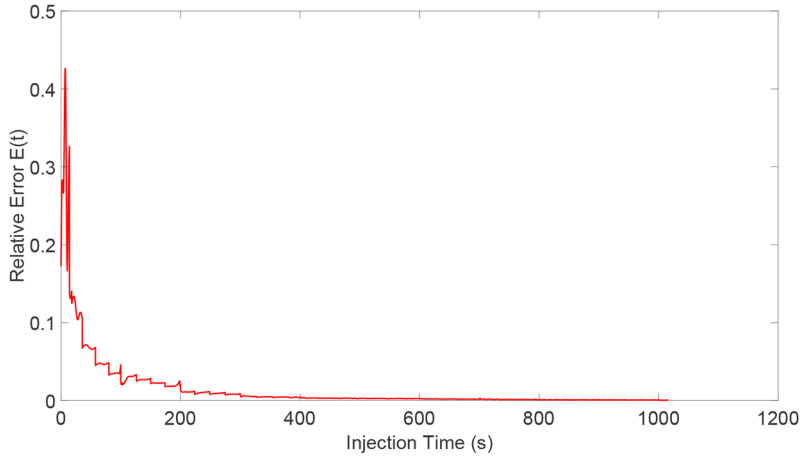


Figure 6. Profile for $E(t)$ with time for solution obtained from our proposed methodology when the validation input is used.

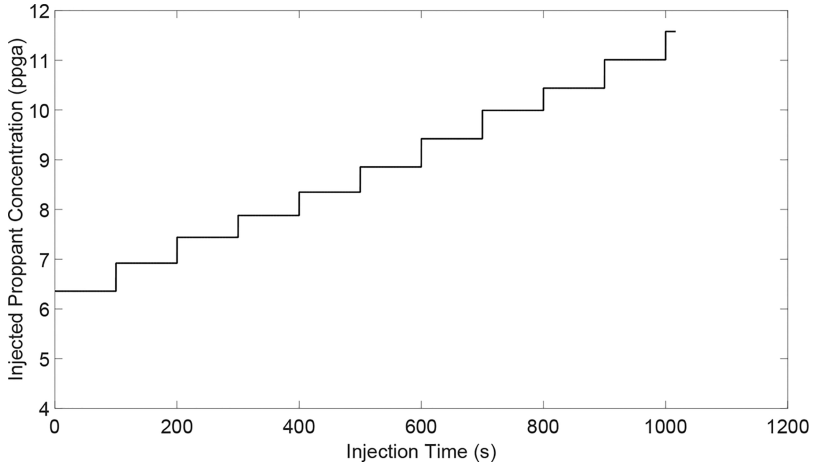


Figure 7. Random input profile used for model prediction.

when compared to the full-order solution. Similarly, our proposed methodology will give accurate solutions when any of the other 12 "training" inputs are used for validation purposes. Thus, these results validate the proposed methodology and warrant its use for the purposes of prediction when random inputs are used within the selected finite input space.

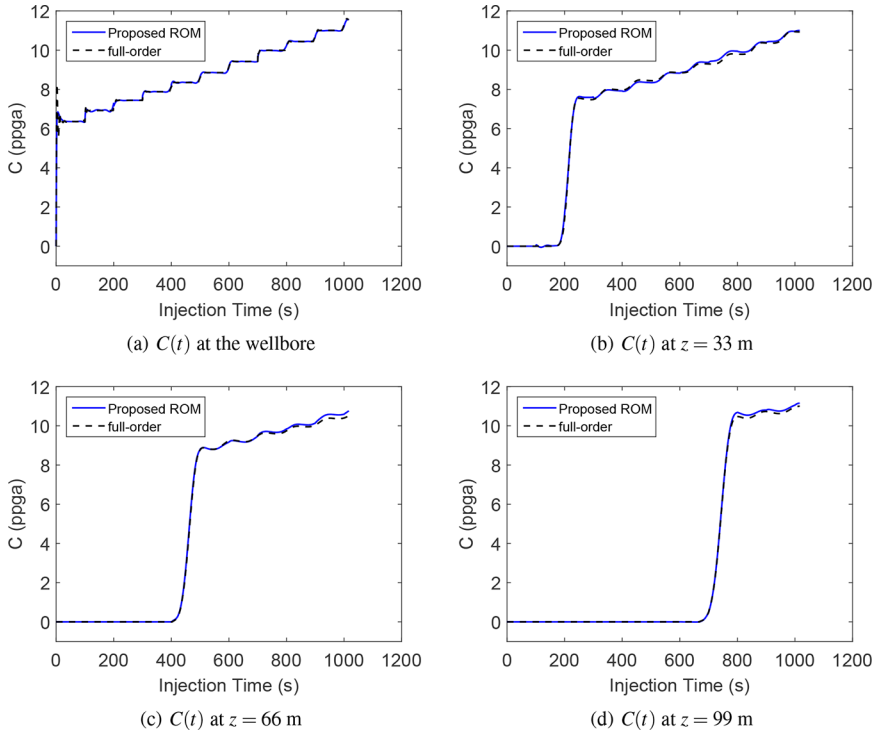
Remark 6. *Considering the distinct "training" input profiles used and the constraint to obtain the subset of cluster centers in the local ROM selection process, it is possible to use just the input information in the proposed methodology to obtain accurate results. This modification can be made by putting a weight of 0 on the states and 1 on the inputs. But for a different set of "random" inputs it is absolutely necessary to use both the information on the states and the inputs as proposed in this work.*

Model Prediction. Equipped with the set of LDMDc-based ROMs, we used our model selection step of the proposed methodology to predict the output when a random input is considered within the selected finite input region.

Random Input. In this case a random input was generated with the constraint that the injected proppant concentration was varied at every 100s as in the "training" inputs. The generated random input is shown in Figure 7. Note that in all the "training" inputs, as described in Figure 3, the step increase of 0.5 ppga in the injected proppant concentration was kept constant throughout the pumping process. But in the case of this model prediction a different constraint was implemented that when generating the random input the step increase in the injected proppant concentration will lie in the range of $[0.425\ 0.575]$. The reason we implemented this constraint is that it makes the generated random input closely mimic the "training" inputs and also allows flexibility to deviate from them to a limited extent. The output predicted by the proposed algorithm is compared with the output from the high-fidelity model at four different locations along the length of the fracture and are presented in Figure 8. From the figure we observe that the proposed methodology is able to accurately predict the concentration at four different locations when compared to the full-order solution. To quantify the accuracy of the prediction, we calculated the relative error as defined in eq 21 and is plotted at various times during the injection process as shown in Figure 9.

Remark 7. *To use our proposed methodology for prediction purposes, the input used should closely mimic the "training" inputs without much deviation for high accuracy. To obtain more flexibility in this regard, we can train our model for a wide variety of inputs. But use of random inputs which have random amount of step increase is not practically feasible in the hydraulic fracturing process. For this reason we considered a uniform step increase in the "training" inputs. For other systems which do not have such practical constraints on the input, random "training" inputs can be used to build the model and consequently similar random inputs*

DOI: 10.1021/acs.iecr.8b05995
Ind. Eng. Chem. Res. 2019, 58, 5588−5601



(a) $C(t)$ at the wellbore (b) $C(t)$ at $z = 33$ m

(c) $C(t)$ at $z = 66$ m (d) $C(t)$ at $z = 99$ m

Figure 8. Comparison of the prediction output computed using LDMDc-based ROMs with the full-order solution.

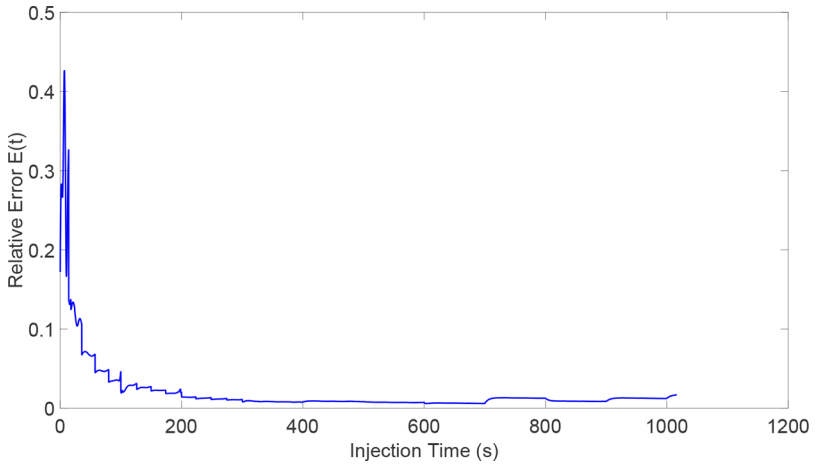


Figure 9. Profile for $E(t)$ with time for the prediction when the random input is used.

can be used for prediction. In such cases a different constraint would be required.

Comparison with LDMDc. In this section, we illustrate the superior performance of our proposed methodology in comparison to the performance of the LDMDc technique proposed by Narasingam and Kwon[39] for a randomly generated, constrained input within the finite region of input space. To do so, we first generated the required data by carrying out open-loop simulations of the high-fidelity model. The "training" input used mimics the ones used in our proposed methodology; in particular, the step increase in the concentration of the injected proppant is kept constant at 0.5 ppga all throughout the injection process as shown in Figure 10. We then used just the information on the states to cluster the data into 11 clusters, where for each cluster we captured the local dynamics using LDMDc-based ROMs. These ROMs are then used to calculate the approximate solution of the full-order model. Now that the LDMDc technique has been used to build the model, we used the prediction input shown in Figure 7 to compare its performance with the performance of our proposed methodology by plotting the relative error profiles for both the techniques as shown in Figure 11. Note that the relative errors for each technique was obtained by comparing their corresponding approximate solutions to the solution of the high-fidelity model. From the plot we observe the limitation of the LDMDc technique, which is its poor performance when a random input is used, which can be overcome by using our proposed methodology.

Remark 8. *The above reasonable performance of the LDMDc technique can be attributed to the fact that the "training" input used and the prediction input are reasonably close. But when an ever wider input domain is considered in our proposed methodology, for any random input in that domain the performance of the LDMDc technique will only get poor.*

DOI: 10.1021/acs.iecr.8b05995
Ind. Eng. Chem. Res. 2019, 58, 5588−5601

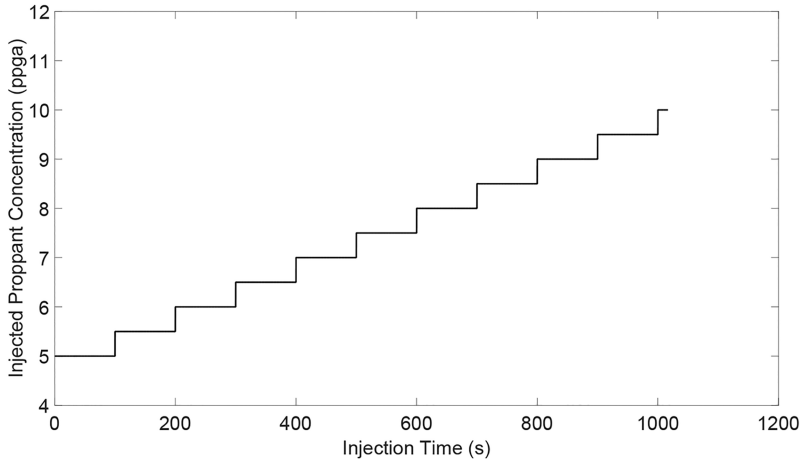


Figure 10. Input used to build a LDMDc-based ROM.

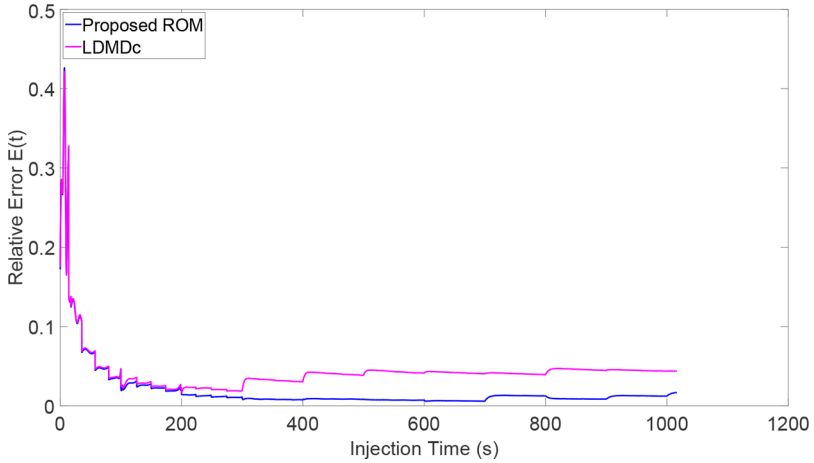


Figure 11. Comparison of the relative error profiles of our proposed methodology and the LDMDc technique.

We further compare the prediction accuracy of our proposed methodology with that of the LDMDc-based model for 100 random input profiles within the selected input region. We calculate the relative root mean squared errors (RMSE), as defined in eq 22, for each technique averaged over the 100 random inputs. The Average RMSE value for both the techniques has been reported in Table 2. The superior performance of our proposed methodology in this comparison warrants its use for prediction purposes and its application in the design of closed-loop controllers for the hydraulic fracturing process.

Table 2. Prediction Comparison—Average RMSE over 100 Random Input Profiles

technique	average RMSE
proposed methodology	1.2586
LDMDc[39]	2.6627

Remark 9. *The proposed methodology can be used to design stable closed-loop controllers for hydraulic fracturing. Recently, lot of efforts have been carried out in this direction which include the use of high-fidelity model to design model predictive control (MPC)-based controller,[53] the use of LDMDc-based ROM in the design of MPC controller,[39] approximate dynamic programming based controller,[54] and the use of POD based ensemble Kalman filter to handle spatial uncertainties during feedback control.*[55]

$$\text{RMSE} = 100 \times \frac{\sqrt{\sum \| \mathbf{x}_{\text{full}} - \mathbf{x}_{\text{pred}} \|_2^2}}{\sqrt{\sum \| \mathbf{x}_{\text{full}} \|_2^2}} \tag{22}$$

CONCLUSION

In this article a novel strategy was proposed utilizing data to enlarge the DOA of the LDMDc method. In this method, "training" inputs within a finite region were used to run numerical simulations of the high-fidelity model to obtain the "training" data. Unlike the conventional LDMDc technique,[39] both the information on the state and the input were utilized in various steps of the algorithm as they both contribute toward the output of the system. The PCA technique was utilized to combine the information on both the variables and transform the data. A clustering strategy was implemented and the cluster centers were defined in terms of the dominant PCSs. Then, DMDc technique was implemented in each cluster and a pair of linear operators (A, B) was obtained to represent the local ROM. Lastly, a model selection strategy was implemented involving the *k*NN technique to select the appropriate pair of (A, B) and predict the system's future behavior. We validated our method using one of the "training" inputs and were able to

DOI: 10.1021/acs.iecr.8b05995
Ind. Eng. Chem. Res. 2019, 58, 5588−5601

predict the output with high accuracy when compared to the output from the high-fidelity model. We proved the superior performance of our proposed methodology to the performance of the LDMDc technique. The advantage of our proposed algorithm is that the DOA of LDMDc can be further increased without compromising on the computational expenses which is usually not expected in a data-based modeling technique. Another advantage is that it enables the use of the ROMs built using LDMDc technique in the design of controllers for a wide variety of operating conditions.

AUTHOR INFORMATION

Corresponding Author

*E-mail: kwonx075@tamu.edu.

ORCID

Joseph Sang-Il Kwon: 0000-0002-7903-5681

Notes

The authors declare no competing financial interest.

ACKNOWLEDGMENTS

The authors gratefully acknowledge financial support from the National Science Foundation (CBET-1804407), the Artie McFerrin Department of Chemical Engineering, and the Texas A&M Energy Institute.

REFERENCES

(1) Noack, B. R.; Afanasiev, K.; Morzynski, M.; Tadmor, G.; Thiele, F. A hierarchy of low- dimensional models for the transient and post-transient cylinder wake. *J. Fluid Mech.* **2003**, *497*, 335−363.

(2) Sahraei, M. H.; Duchesne, M. A.; Hughes, R. W.; Ricardez-Sandoval, L. A. Dynamic reduced order modeling of an entrained-flow slagging gasifier using a new recirculation ratio correlation. *Fuel* **2017**, *196*, 520−531.

(3) Sahraei, M. H.; Duchesne, M. A.; Boisvert, P. G.; Hughes, R. W.; Ricardez-Sandoval, L. A. Reduced-Order Modeling of a Commercial-Scale Gasifier Using a Multielement Injector Feed System. *Ind. Eng. Chem. Res.* **2017**, *56*, 7285−7300.

(4) Hossein Sahraei, M.; Duchesne, M. A.; Yandon, R.; Majeski, A.; Hughes, R. W.; Ricardez- Sandoval, L. A. Reduced order modeling of a short-residence time gasifier. *Fuel* **2015**, *161*, 222−232.

(5) Holmes, P.; Lumley, J. L.; Berkooz, G. *Turbulence, Coherent Structures, Dynamical Systems and Symmetry*; Cambridge University Press: New York, 1996.

(6) Armaou, A.; Christofides, P. D. Finite-dimensional control of nonlinear parabolic PDE systems with time-dependent spatial domains using empirical eigenfunctions. *Int. J. Appl. Math. Comput. Sci.* **2001**, *11*, 287−317.

(7) Berkooz, G.; Holmes, P.; Lumley, J. The proper orthogonal decomposition in the analysis of turbulent flows. *Annu. Rev. Fluid Mech.* **1993**, *25*, 539−575.

(8) Christensen, E.; Brons, M.; Sorensen, J. Evaluation of proper orthogonal decomposition based decomposition techniques applied to parameter-dependent nonturbulent flows. *SIAM J. Sci. Comput.* **1999**, *21*, 1419−1434.

(9) Park, H. M.; Lee, M. W. An efficient method of solving the Navier-Stokes equations for flow control. *Int. J. Numer. Methods Eng.* **1998**, *41*, 1133−1151.

(10) Ravindran, S. Proper orthogonal decomposition in optimal control of fluids. *Int. J. Numer. Methods Fluids* **2000**, *34*, 425−448.

(11) Sidhu, H. S.; Narasingam, A.; Siddhamshetty, P.; Kwon, J. S. Model order reduction of nonlinear parabolic PDE systems with moving boundaries using sparse proper orthogonal decomposition: application to hydraulic fracturing. *Comput. Chem. Eng.* **2018**, *112*, 92−100.

(12) Narasingam, A.; Siddhamshetty, P.; Kwon, J. S. Temporal clustering for order reduction of nonlinear parabolic PDE systems with time-dependent spatial domains: Application to a hydraulic fracturing process. *AIChE J.* **2017**, *63*, 3818−3831.

(13) Pitchaiah, S.; Armaou, A. Output Feedback Control of Distributed Parameter Systems Using Adaptive Proper Orthogonal Decomposition. *Ind. Eng. Chem. Res.* **2010**, *49*, 10496−10509.

(14) Noack, B. R.; Schlegel, M.; Ahlborn, B.; Mutschke, B.; Morzynski, M.; Comte, P.; Tadmor, G. A finite-time thermodynamics formalism for unsteady flows. *J. Non-Equilib. Thermodyn.* **2008**, *33*, 103−148.

(15) Schmid, P. J.; Sesterhenn, J. Dynamic mode decomposition of numerical and experimental data. *Bull. Am. Phys. Soc. 61st APS Meeting, San Antonio, Texas* **2008**, 208.

(16) Ghommem, M.; Calo, V. M.; Efendiev, Y. Mode decomposition methods for flows in high- contrast porous media. A Global approach. *J. Comput. Phys.* **2014**, *257*, 400−413.

(17) Rowley, C. W.; Mezić, I.; Bagheri, S.; Schlatter, P.; Henningson, D. S. Spectral analysis of nonlinear flows. *J. Fluid Mech.* **2009**, *641*, 115−127.

(18) Mezić, I. Analysis of fluid flows via spectral properties of the Koopman operator. *Annu. Rev. Fluid Mech.* **2013**, *45*, 357−378.

(19) Bagheri, S. Koopman-mode decomposition of the cylinder wake. *J. Fluid Mech.* **2013**, *726*, 596−623.

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

(21) Schmid, P. J.; Li, L.; Juniper, M. P.; Pust, O. Applications of the dynamic mode decomposition. *Theor. Comput. Fluid Dyn.* **2011**, *25*, 249−259.

(22) Seena, A.; Sung, H. J. Dynamic mode decomposition of turbulent cavity flows for selfsustained oscillations. *Int. J. Heat Fluid Flow* **2011**, *32*, 1098−1110.

(23) Mizuno, Y.; Duke, D.; Atkinson, C.; Soria, J. Investigation of wall-bounded turbulent flow using dynamic mode decomposition. *J. Phys. Conf. Ser.* **2011**, *318*, 042040.

(24) Muld, T. W.; Efraimsson, G.; Henningson, D. S. Flow structures around high-speed train extracted using proper orthogonal decomposition and dynamic mode decomposition. *Comput. Fluids* **2012**, *57*, 87−97.

(25) Schmid, P. J. Dynamic mode decomposition of experimental data. In *8th International Symposium on Particle Image Velocimetry−PIV09*, 2009.

(26) Schmid, P. J.; Meyer, K. E.; Pust, O. Dynamic mode decomposition and proper orthogonal decomposition of flow in a lid-driven cylindrical cavity. In *8th International Symposium on Particle Image Velocimetry−PIV09*, 2009.

(27) Schmid, P. J. Application of the dynamic mode decomposition to experimental data. *Exp. Fluids* **2011**, *50*, 1123−1130.

(28) Pan, C.; Yu, D.; Wang, J. Dynamical mode decomposition of Gurney flap wake flow. *Theor. Appl. Mech. Lett.* **2011**, *1*, 012002.

(29) Semeraro, O.; Bellani, G.; Lundell, F. Analysis of time-resolved PIV measurements of a confined turbulent jet using POD and Koopman modes. *Exp. Fluids* **2012**, *53*, 1203−1220.

(30) Lusseyran, F.; Gueniat, F.; Basley, J.; Douay, C. L.; Pastur, L. R.; Faure, T. M.; Schmid, P. J. Flow coherent structures and frequency signature: Application of the dynamic modes decomposition to open cavity flow. *J. Phys. Conf. Ser.* **2011**, *318*, 042036.

(31) Duke, D.; Soria, J.; Honnery, D. An error analysis of the dynamic mode decomposition. *Exp. Fluids* **2012**, *52*, 529−542.

(32) Tu, J. H.; Rowley, C. W. An improved algorithm for balanced POD through an analytic treatment of impulse response tails. *J. Comput. Phys.* **2012**, *231*, 5317.

(33) Belson, B. A.; Tu, J. H.; Rowley, C. W. Algorithm 945: modredA parallelized model reduction library. *ACM Trans. Math. Softw* **2014**, *40*, 1.

(34) Jovanović, M. R.; Schmid, P. J.; Nichols, J. W. Sparsity-promoting dynamic mode decomposition. *Phys. Fluids* **2014**, *26*, 024103.

(35) Chen, K. K.; Tu, J. H.; Rowley, C. W. Variants of dynamic mode decomposition: Connections between Koopman and Fourier analyses. *J. Nonlinear Sci.* **2012**, *22*, 887−915.

DOI: 10.1021/acs.iecr.8b05995
Ind. Eng. Chem. Res. 2019, 58, 5588−5601

(36) Goulart, P. J.; Wynn, A.; Pearson, D. S. Optimal mode decomposition for high dimensional systems. In *Proceedings of the 51st IEEE Conference on Decision and Control*, Maui, Hawaii, 2012; pp 4965−4970.
(37) Wynn, A.; Pearson, D. S.; Ganapathisubramani, B.; Goulart, P. J. Optimal mode decomposition for unsteady flows. *J. Fluid Mech.* **2013**, *733*, 473−503.
(38) Proctor, J. L.; Brunton, S. L.; Kutz, J. N. Dynamic mode decomposition with control. *SIAM J. Appl. Dyn. Syst.* **2016**, *15*, 142−161.
(39) Narasingam, A.; Kwon, J. S. Development of local dynamic mode decomposition with control: Application to model predictive control of hydraulic fracturing. *Comput. Chem. Eng.* **2017**, *106*, 501−511.
(40) Tan, M. P.; Broach, J. R.; Floudas, C. A. A novel clustering approach and prediction of optimal number of clusters: global optimum search with enhanced positioning. *J. Glob. Optim.* **2007**, *39*, 323−346.
(41) Perkins, T. K.; Kern, L. R. Widths of hydraulic fractures. *J. Pet. Technol.* **1961**, *13*, 937−949.
(42) Nordgren, R. Propagation of a vertical hydraulic fracture. *SPEJ, Soc. Pet. Eng. J.* **1972**, *12*, 306−314.
(43) Howard, G. C.; Fast, C. R. Optimum fluid characteristics for fracture extension. *Drill. prod. pract.* **1957**, *24*, 261−270.
(44) Economides, M. J.; Nolte, K. G. *Reservoir stimulation*; Wiley: Chichester, 2000.
(45) Gu, Q.; Hoo, K. A. Evaluating the Performance of a Fracturing Treatment Design. *Ind. Eng. Chem. Res.* **2014**, *53*, 10491−10503.
(46) Gu, Q.; Hoo, K. A. Model-based closed-loop control of the hydraulic fracturing process. *Ind. Eng. Chem. Res.* **2015**, *54*, 1585−1594.
(47) Adachi, J.; Siebrits, E.; Peirce, A.; Desroches, J. Computer simulation of hydraulic fractures. *Int. J. Rock Mech. Min. Sci.* **2007**, *44*, 739−757.
(48) Daneshy, A. Numerical solution of sand transport in hydraulic fracturing. *JPT, J. Pet. Technol.* **1978**, *30*, 132−140.
(49) Barree, R.; Conway, M. Experimental and numerical modeling of convective proppant transport. *J. Pet. Technol.* **1995**, *47*, 216−222.
(50) Novotny, E. J. Proppant transport. In *Proceedings of the 52nd SPE annual technical conference and exhibition*, Denver, CO. 1977; SPE 6813.
(51) Yang, S.; Siddhamshetty, P.; Kwon, J. S. Optimal pumping schedule design to achieve a uniform proppant concentration level in hydraulic fracturing. *Comput. Chem. Eng.* **2017**, *101*, 138−147.
(52) Siddhamshetty, P.; Kwon, J. S.; Liu, S.; Valko, P. P. Feedback control of proppant bank heights during hydraulic fracturing for enhanced productivity in shale formations. *AIChE J.* **2018**, *64*, 1638−1650.
(53) Siddhamshetty, P.; Yang, S.; Kwon, J. S.-I. Modeling of hydraulic fracturing and designing of online pumping schedules to achieve uniform proppant concentration in conventional oil reservoirs. *Comput. Chem. Eng.* **2018**, *114*, 306−317. FOCAPO/CPC 2017.
(54) Singh Sidhu, H.; Siddhamshetty, P.; Kwon, J. S. Approximate Dynamic Programming Based Control of Proppant Concentration in Hydraulic Fracturing. *Mathematics* **2018**, *6*, 132.
(55) Narasingam, A.; Siddhamshetty, P.; Kwon, J. S. Handling Spatial Heterogeneity in Reservoir Parameters Using Proper Orthogonal Decomposition Based Ensemble Kalman Filter for Model-Based Feedback Control of Hydraulic Fracturing. *Ind. Eng. Chem. Res.* **2018**, *57*, 3977.

DOI: 10.1021/acs.iecr.8b05995
Ind. Eng. Chem. Res. 2019, 58, 5588−5601