E-ISSN 2586-9027
Homepage : https://tci-thaijo.org/index.php/SciTechAsia Science & Technology Asia
Vol.30 No.2 April - June 2025 Page: [7-20]
Original research article

A Comparative Study of Subspace Measures for Parametric Model Reduction Using K-Medoids Clustering and Neural Networks: A Case Study for Burgers' Equation

Norapon Sukuntee[1], Suvarin Phungngam[2], Saifon Chaturantabut[3,*]

[1]*Faculty of Learning Sciences and Education, Thammasat University, Pathum Thani 12120, Thailand*
[2]*Thammasat Secondary School, Thammasat University, Pathum Thani 12120, Thailand*
[3]*Department of Mathematics and Statistics, Faculty of Science and Technology, Thammasat University, Pathum Thani 12120, Thailand*

Received 6 October 2024; Received in revised form 18 March 2025
Accepted 18 April 2025; Available online 18 June 2025

ABSTRACT

This work investigates the performance of three distance measures including Grassmann, Binet-Cauchy and Chordal distances in the complexity reduction of parameterized nonlinear dynamical systems using K-medoids clustering and neural network. The Grassmann distance provides a geometric perspective by measuring the subspace angles, while the Binet-Cauchy distance utilizes determinants to measure volume distortion between subspaces. The Chordal distance, on the other hand, considers a straightforward metric that uses the Euclidean distance between points on a unit sphere. Each of these distances is employed in the K-medoids clustering process to construct a dictionary of projection basis sets used in reduced-order modeling. The neural network is then trained to automatically select the most suitable basis sets in the dictionary for a given parameter vector. The numerical experiments are demonstrated using the parameterized Burgers' equation, where the governing partial differential equation, initial conditions and boundary conditions are all parameter-dependent. The Grassmann distance is shown to give the most accurate result across different parameter settings when compared with the Binet-Cauchy and Chordal distances.

Keywords: Model order reduction; Proper orthogonal decomposition; Discrete empirical interpolation method; Machine learning; Subspace distances

***Corresponding author:** saifon@mathstat.sci.tu.ac.th **doi:** 10.14456/scitechasia.2025.22

1. Introduction

Reduced-order modeling plays a crucial role in the simulation of complex dynamical systems, offering a computationally efficient alternative to full-order model (FOM) without compromising accuracy. Its importance spans across several fields such as fluid dynamics, control systems and optimization, where high-dimensional simulations can become prohibitively expensive. By reducing the number of degrees of freedom, a reduced-order model (ROM) enables faster computation, providing real-time simulation with reliable accuracy.

Recently, various model order reduction techniques have been proposed. One widely-used method for ROM construction is the POD-Galerkin approach coupled with the discrete empirical interpolation (DEIM), often called POD-DEIM. In [[1]], D. Chen and H. Song proposed the POD-DEIM reduced-order method for stochastic Allen–Cahn equations with multiplicative noise. A. Roy and M. Nabi [2] presented a reduced-order modeling for three dimensional electrothermal microgripper with thermal nonlinearity using POD-DEIM. Also in [3], a POD-DEIM-ROM for compressible gas reservoir flow based on the Peng-Robinson equation of state was introduced. More details on using POD-DEIM in the construction of ROM for various problems can be found in [4–7]. Detailed analyses of the effectiveness of the POD-DEIM approach including error estimation can be found in [8, 9].

Despite the success of the POD-DEIM approach in reduced-order modeling, it still encounters notable challenges when dealing with parameterized systems. In many parameterized problems, a global reduced basis is employed to represent the entire parameter space with the assumption that this basis will sufficiently capture the system's dynamic across all parameter variation. However, it can fail in cases where the system exhibits highly localized behaviors or significant changes in dynamics for different parameter values. To address these limitations, several studies have explored the use of local reduced bases, which are constructed for specific regions of the parameter space. In particular, physics-informed cluster analysis and a priori efficiency criterion for the construction of local reduced-order bases were introduced in [10], where the local bases are generated by the corresponding centroids of the clusters. In [11], J. Cortés et al. presented a local reduced-order method for computing bifurcation diagrams in 2D Rayleigh–Bénard convection problems, where the locality of the method is achieved through K-means clustering as well. In 2024, N. Sukuntee and S. Chaturantabut [12] employed the Grassmann distance to partition the parameter space, allowing for more effective construction of local ROMs by measuring the distance between subspaces. Further details and comprehensive analyses of these approaches can be found in [12, 13].

Previous researches have largely focused on studying the effects of parameters on the system's dynamics without explicitly considering the impact of parameterized initial and boundary conditions. In this article, we extend these approaches by focusing on the construction of local bases depending on different distance measures, including Grassmann, Binet-Cauchy and Chordal distances, specifically applied to parameterized Burgers' equations. We study how these distance measures perform in the context of parameterized partial differential equation (PDE), initial conditions (ICs) and boundary conditions (BCs), comparing their effectiveness in capturing localized solution behaviors and assessing their

impact on the accuracy of the ROMs.

This manuscript is arranged as follows. Section 2 provides a fundamental overview of the methods in constructing ROMs, particularly focusing on proper orthogonal decompostion (POD), Galerkin projection and the discrete empirical interpolation method (DEIM). In Section 3, we introduce our proposed approach for constructing local POD bases using different distance measures across parameter range. Section 4 presents the numerical results of applying our approach to the parameterized Burgers' equation. Finally, we briefly summarize our work in Section 5.

2. Preliminaries

This section provides some background on a projection-based model reduction technique using POD and DEIM.

2.1 Proper orthogonal decomposition

Proper orthogonal decomposition (POD) is a widely-employed technique for constructing a low-dimensional basis that optimally approximates a given dataset. Given a dataset $\{u_1, u_2, \ldots, u_m\} \subset \mathbb{R}^n$ and define $\mathcal{U} = \text{span}\{u_1, u_2, \ldots, u_m\}$, POD identifies an orthonormal basis set $\{v_1, v_2, \ldots, v_l\}$, where $l \ll r = \dim(\mathcal{U})$ such that the subspace spanned by $\{v_1, v_2, \ldots, v_l\}$ approximates the subspace spanned by the original set $\{u_1, u_2, \ldots, u_m\}$, i.e.,

$$\text{span}\{v_1, v_2, \ldots, v_l\} \approx \text{span}\{u_1, u_2, \ldots, u_m\}. \tag{2.1}$$

It is well-known that the aforementioned orthonormal basis can be obtained through singular value decomposition (SVD). In particular, let $U = [u_1, u_2, \ldots, u_m] \in \mathbb{R}^{n\times m}$ be a matrix that represents a given dataset. The SVD of U is given by

$$U = V\Sigma W^{\mathsf{T}}, \tag{2.2}$$

where $V \in \mathbb{R}^{n\times r}$ and $W \in \mathbb{R}^{m\times r}$ are orthogonal matrices and $\Sigma = \text{diag}(\sigma_1, \sigma_2, \ldots, \sigma_r) \in \mathbb{R}^{r\times r}$ is a diagonal matrix containing the singular values in decreasing order. The first l columns of V, denoted by $V_l = [v_1, v_2, \ldots, v_l] \in \mathbb{R}^{n\times l}$ is called the l-dimensional POD basis of U and satisfies Eq. (2.1), making the best approximation in terms of ℓ^2-norm. Equivalently, the l-dimensional POD basis solves the following minimization problem

$$\min_{\{\phi_i\}_{i=1}^{l}} \sum_{j=1}^{m} \|u_j - \sum_{i=1}^{l} (u_j^{\mathsf{T}}\phi_i)\phi_i\|_2^2, \tag{2.3}$$

with constrains $\phi_i^{\mathsf{T}}\phi_j = \delta_{ij}$, where δ_{ij} is the Kronecker delta function for $i, j = 1, \ldots, l$. For a given dimension l, the corresponding minimum error of approximating (2.3) by using the l-dimensional POD basis from the columns of V_l is given by

$$\sum_{j=1}^{m} \|u_j - \sum_{i=1}^{l} (u_j^{\mathsf{T}}v_i)v_i\|_2^2 = \sum_{i=l+1}^{r} \sigma_i^2. \tag{2.4}$$

2.2 Reduced-order modeling via Galerkin projection and DEIM

Consider a system of parameterized ODEs arising from the spatial discretization of nonlinear PDE

$$\frac{d}{dt}u(t;p) = L(p)u(t;p) + f(t, u(t;p)), \tag{2.5}$$

with initial condition $u(0;p) = u_0(p)$, where $L \in \mathbb{R}^{n\times n}$ is the linear part, $f : \mathcal{T} \times \mathbb{R}^n \to \mathbb{R}^n$ is the nonlinear part and $u : \mathcal{T} \to \mathbb{R}^n$ is the time-dependent vector of state variables with parameter $p \in \mathcal{P} \subset \mathbb{R}^q$. Eq. (2.5) is called the full-order model (FOM) of dimension n. To obtain accurate numerical solution for (2.5), the dimension n might have to be large, which often makes simulations slow. To handle this issue, dimensionality reduction techniques can be

employed as discussed next. For simplicity, we will omit the notation of the parameter p for a cleaner and more concise presentation. Suppose that we have a solution matrix U, representing a set of snapshots. From this solution, we can construct a POD basis as described in Subsection 2.1. The POD procedure applied to U yields an l_u-dimensional POD basis, denoted by $\mathrm{V}^u_{l_u} = [\mathrm{v}^u_1, \mathrm{v}^u_2, \ldots, \mathrm{v}^u_{l_u}] \in \mathbb{R}^{n\times l_u}$. This implies that the solution $\mathrm{u}(t)$ can be estimated as

$$\mathrm{u}(t) \approx \mathrm{V}^u_{l_u}\tilde{\mathrm{u}}(t), \tag{2.6}$$

where $\tilde{\mathrm{u}}$ is the reduced-representation of u. By substituting (2.6) into (2.5) and subsequently applying the Galerkin projection, we obtain

$$\frac{d}{dt}\tilde{\mathrm{u}}(t) = \tilde{\mathrm{L}}\tilde{\mathrm{u}}(t) + \overbrace{\underbrace{\mathrm{V}^{u\,\mathsf{T}}_{l_u}}_{l_u\times n}\underbrace{\mathrm{f}(t, \mathrm{V}^u_{l_u}\tilde{\mathrm{u}}(t))}_{n\times 1}}^{\bar{\mathrm{f}}(t,\tilde{\mathrm{u}}(t))}, \tag{2.7}$$

where $\tilde{\mathrm{L}} = \mathrm{V}^{u\,\mathsf{T}}_{l_u}\mathrm{L}\mathrm{V}^u_{l_u} \in \mathbb{R}^{l_u\times l_u}$. The resulting reduced-order model (2.7) is called the POD-Galerkin-ROM or POD-ROM of dimension $l_u \ll n$. However, the computational efficiency of the POD-Galerkin technique is limited to the linear part of the system, as the complexity of the nonlinear term $\bar{\mathrm{f}}(t, \tilde{\mathrm{u}}(t))$ remains largely unreduced depending on the full-order dimension n. To address this issue, the discrete empirical interpolation method (DEIM) is used in conjunction with Galerkin projection to further reduce the computational complexity associated with nonlinear term. Suppose that we also have nonlinear snapshots, denoted by $\mathrm{F} = [\mathrm{f}_1, \mathrm{f}_2, \ldots \mathrm{f}_m] \in \mathbb{R}^{n\times m}$, which are arranged in matrix form. We perform the POD method on F, which can be computed as described in Subsection 2.9, to obtain an l_f-dimensional POD basis of F, denoted by $\mathrm{V}^f_{l_f} = [\mathrm{v}^f_1, \mathrm{v}^f_2, \ldots, \mathrm{v}^f_{l_f}] \in \mathbb{R}^{n\times l_f}$. The DEIM requires the POD basis of nonlinear snapshots as input to construct interpolation matrix. It can be summarized in the following algorithm.

Algorithm 1 DEIM [9]

Input : $\mathrm{V}^f_{l_f} = [\mathrm{v}^f_1, \mathrm{v}^f_2, \ldots, \mathrm{v}^f_{l_f}] \in \mathbb{R}^{n\times l_f}$
procedure
  $[|\rho|, \wp_1] = \max\{|\mathbf{v}^f_1|\}$
  $\hat{\mathbf{V}} = [\mathbf{v}^f_1]$, $\hat{\mathbf{P}} = [\mathbf{e}_{\wp_1}]$, $\hat{\wp} = [\wp_1]$
  for $j = 2 : l_f$ **do**
    $\mathbf{c} = (\hat{\mathbf{P}}^{\mathsf{T}}\hat{\mathbf{V}})^{-1}\hat{\mathbf{P}}^{\mathsf{T}}\mathbf{v}^f_j$
    $\mathbf{r} = \mathbf{v}^f_j - \hat{\mathbf{V}}\mathbf{c}$
    $[|\rho|, \wp_j] = \max\{|\mathbf{r}|\}$
    $\hat{\mathbf{V}} \leftarrow [\hat{\mathbf{V}}\ \mathbf{v}^f_j]$, $\hat{\mathbf{P}} \leftarrow [\hat{\mathbf{P}}\ \mathbf{e}_{\wp_j}]$, $\hat{\wp} \leftarrow \begin{bmatrix}\hat{\wp}\\ \wp_j\end{bmatrix}$
  end for
end procedure
Output : $\mathrm{P} = \hat{\mathrm{P}} = [\mathrm{e}_{\wp_1}, \mathrm{e}_{\wp_2}, \ldots, \mathrm{e}_{\wp_{l_f}}] \in \mathbb{R}^{n\times l_f}$, $\hat{\wp} = [\wp_1, \wp_2, \ldots, \wp_{l_f}]^{\mathsf{T}} \in \mathbb{R}^{l_f}$

By utilizing the interpolation matrix P obtained from Algorithm 1, the nonlinear term can be approximated as follows:

$$\mathrm{f}(t, \mathrm{V}^u_{l_u}\tilde{\mathrm{u}}(t)) \approx \underbrace{\mathrm{V}^f_{l_f}(\mathrm{P}^{\mathsf{T}}\mathrm{V}^f_{l_f})^{-1}\mathrm{P}^{\mathsf{T}}\mathrm{f}(t, \mathrm{V}^u_{l_u}\tilde{\mathrm{u}}(t))}_{\mathrm{f}^{\mathrm{DEIM}}(t)}. \tag{2.8}$$

By substituting (2.8) into (2.7), we obtain the reduced-order model, which is entirely independent of the full computational size shown below.

$$\frac{d}{dt}\tilde{\mathrm{u}}(t) = \tilde{\mathrm{L}}\tilde{\mathrm{u}}(t) + \widehat{F}(t) \tag{2.9}$$

where

$$\begin{aligned}\widehat{F}(t) &= \mathrm{V}^{u\,\mathsf{T}}_{l_u}\mathrm{f}^{\mathrm{DEIM}}(t)\\ &= \underbrace{\mathrm{V}^{u\,\mathsf{T}}_{l_u}\mathrm{V}^f_{l_f}}_{l_u\times l_f}\underbrace{(\mathrm{P}^{\mathsf{T}}\mathrm{V}^f_{l_f})^{-1}}_{l_f\times l_f}\underbrace{\mathrm{P}^{\mathsf{T}}\mathrm{f}(t, \mathrm{V}^u_{l_u}\tilde{\mathrm{u}}(t))}_{l_f\times 1}.\end{aligned}$$

This reduced system (2.9) is called POD-Galerkin-DEIM-ROM or POD-DEIM-ROM.

3. Automated Local Pod Basis Selection from Multi-Distance Clustered Dictionary via K-Medoids and Neural Network

Previously in Subsection 2.2, we have discussed the projection-based technique POD-Galerkin-DEIM-ROM, where the essential ingredients are POD bases. When studying the parameterized system (2.5), the simplest approach is to construct a single global POD basis from snapshots taken across the entire range of parameters, offering a broad approximation of the system's behavior. However, a single global POD basis may not always provide sufficient accuracy for system with complex or highly localized dynamics. In such cases, the approximation can be improved by using local POD bases. Rather than relying on one global basis, the parameter space is divided into clusters based on a suitable distance measure. This allows the construction of POD bases to better suit specific regions of the parameter spaces for capturing localized variations in the system's dynamics more efficiently. To achieve the partitioning of the parameter space, an effective clustering algorithm will be employed and discussed in the following subsection.

3.1 K-medoids clustering algorithm

One of the efficient clustering methods is the K-medoids algorithm, which is particularly beneficial when the data set does not permit straightforward computation of centroids, such as when distances are not derived from standard Euclidean space. Unlike other centroid-based methods, K-medoids algorithm selects actual data points (medoids) as the center of each cluster, making it more robust for data sets where the concept of a centroid may not exist or is difficult to compute. Also, the benefit of using K-medoids is its flexibility in handling arbitrary distance metrics, allowing it to operate effectively even when the distance between data points is defined in complex ways, such as through manifold or subspace distances. The K-medoids algorithm can be summarized in Algorithm 2.

Algorithm 2 K-medoids clustering algorithm

Input : $iter \in \mathbb{N}$, $K < s$, $\{\mathrm{p}_i\}_{i=1}^{s}$
procedure
  Choose randomly K initial medoids, $\hat{\mathrm{p}}_1, \hat{\mathrm{p}}_2, \ldots, \hat{\mathrm{p}}_K$ from $\{\mathrm{p}_i\}_{i=1}^{s}$
  $\ell = \mathrm{zeros}(s, 1)$
  for $q = 1 : iter$ **do**
    for $i = 1 : s$ **do**
      for $k = 1 : K$ **do**
        $d_{ik} = d_*(\mathrm{p}_i, \hat{\mathrm{p}}_k)$
      end for
      $\ell(i) = \underset{k \in \{1,2,\ldots,K\}}{\arg\min}\ d_{ik}$
    end for
    for $k = 1 : K$ **do**
      $\hat{\mathrm{p}}_k \leftarrow \underset{\substack{\mathrm{p}_i \\ \ell(i)=k}}{\arg\min} \sum_{\ell(j)=k} d_*(\mathrm{p}_i, \mathrm{p}_j)$
    end for
  end for
end procedure
Output : $\{\hat{\mathrm{p}}_k\}_{k=1}^{K}$

From Algorithm 2, the inputs are the number of iterations $iter$, the desired number of clusters K, and the data set $\{\mathrm{p}_i\}_{i=1}^{s}$, where s is the total number of samples. It begins by selecting K initial medoids $\{\hat{\mathrm{p}}_k\}_{k=1}^{K}$ from the data set. Each parameter p_i is then assigned to the cluster k corresponding to the closest medoid. Inside each cluster k, the medoid $\hat{\mathrm{p}}_k$ is updated by choosing the parameter belonging to this

cluster, which minimizes the sum of distances to all other points. This procedure is repeated for $q = 1, 2, \ldots, iter$. Once the clustering process is completed, a dictionary of local POD bases can be defined, where each cluster k is represented by the POD basis computed from the snapshots associated with medoid parameter $\hat{\mathrm{p}}_k$.

Note that the distance d_* in Algorithm 2 is used to assign points to cluster and the type of this distance can be chosen to accommodate different types of data of objectives. In this work, d_* is specifically chosen to reflect the differences between the POD bases obtained from solutions corresponding to different parameters. This distance can be based on metrics such as Grassmann distance, Binet-Cauchy distance or Chordal distance, which will be discussed in Subsection 3.2.

3.2 Distances between subspaces

In many applications, the distance between subspaces plays a crucial role in quantifying the similarity or divergence of different vector spaces. Several metrics have been developed to measure these distances, each offering unique insights into subspace alignment and separation. Consider two subspaces $\mathcal{A}$ and $\mathcal{B}$ with dimensions $\dim(\mathcal{A}) = a$ and $\dim(\mathcal{B}) = b$. Let $\mathcal{A}$ and $\mathcal{B}$ be represented by an orthonormal basis $\mathrm{A} \in \mathbb{R}^{n\times a}$ and $\mathrm{B} \in \mathbb{R}^{n\times b}$. This work considers the distances between subspaces based on Grassmann distance, Binet-Cauchy distance and Chordal distance as discussed next.

3.2.1 Grassmann distance

The Grassmann distance measures the separation between subspaces within the Grassmann manifold (Grassmannian), where each point represents a subspace of fixed dimension. This distance captures the angular deviation between subspaces and is commonly used to quantify how much one subspace has rotated relative to another. A general Grassmann distance that can calculate distances between subspaces of different dimensions was defined in [14]. This generalization extends the concept of the Grassmann manifold to the doubly-infinite Grassmannian, denoted by $Gr(\infty, \infty)$, which parametrizes subspaces of varying dimensions without requiring a fixed ambient space. This extension greatly increases the flexibility of Grassmannian geometry in applications where subspaces of different dimensions need to be compared. This Grassmann distance formulation is given by

$$d_{Gr}(\mathcal{A}, \mathcal{B}) := \sqrt{\frac{\pi^2}{4}|a-b| + \sum_{i=1}^{\min(a,b)} \theta_i^2}, \tag{3.1}$$

where $\theta_i = \cos^{-1}(\sigma_i)$ are the principle angles of $\mathrm{A}^\mathsf{T}\mathrm{B}$.

3.2.2 Binet-Cauchy distance

The Binet-Cauchy distance is based on the Binet-Cauchy theorem. It measures the similarity between subspaces by comparing the volumes of the parallelepipeds formed by their basis vectors. The Binet-Cauchy distance is defined as

$$d_{BC}(\mathcal{A}, \mathcal{B}) := \sqrt{1 - \prod_{i=1}^{\min(a,b)} \sigma_i^2}, \tag{3.2}$$

where σ_i are singular values of $\mathrm{A}^\mathsf{T}\mathrm{B}$.

3.2.3 Chordal distance

The Chordal distance measures the Euclidean distance between the subspaces in projective space. It is sensitive to large angular differences, making it suitable for applications where significant deviations between subspaces matter. The Chordal

distance is given by

$$d_{Chor}(\mathcal{A},\mathcal{B}) := \sqrt{\sum_{i=1}^{\min(a,b)} \sin^2(\theta_i)}, \tag{3.3}$$

where $\theta_i = \cos^{-1}(\sigma_i)$ are the principle angles of $A^T B$.

Once the clusters and corresponding POD bases are obtained, the challenge arises when dealing with new unseen parameters. For these parameters, calculating exact distances to determine the appropriate cluster is impractical due to limited information. In particular, the solution of the dynamical system corresponding to the new parameter may not be available. Consequently, we cannot directly assign an unseen parameter to a specific cluster based purely on subspace distances. Instead, a more efficient strategy is required to predict the cluster that should be assigned to a new parameter, based on learning information from known parameters. In the next subsection, we introduce a neural network classifier to automate the cluster assignment process for unseen parameters.

3.3 Predictive cluster classification via neural network

Neural networks, inspired by the structure of the human brain, are powerful tools in machine learning designed to recognize patterns and solve complex systems through a system of interconnected nodes (neurons) between layers. The first layer, also referred to as the input layer, is an array of input parameters and does not incorporate with any activation function. The input signals are then propagated to the subsequent layer (the hidden layer), where they undergo some internal calculations, such as weighted combinations and the application of an activation function, before being transformed into the hidden layer's output. This process can be repeated across multiple hidden layers, allowing the network to progressively transform and refine the input signals before passing them to the final layer as depicted in Figs. 1-2.

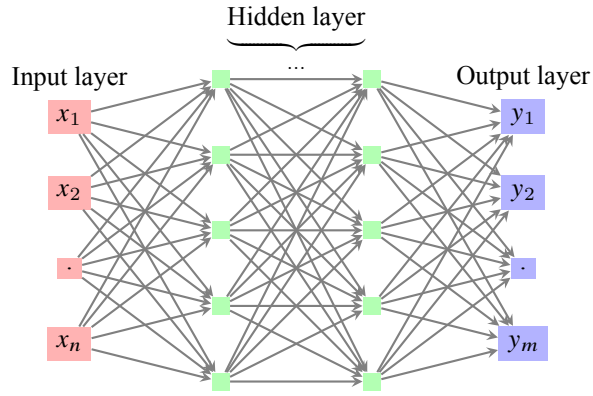


Fig. 1. Example of neural network's structure.

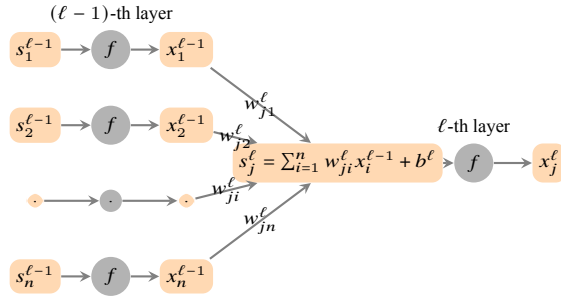


Fig. 2. A perspective view example of transferring the signals from the $(\ell-1)$-th layer to the next ℓ-th layer, where each neuron computes a weighted sum of its inputs (linear combination), adds a bias, and then applies a non-linear activation function f.

Various types of neural networks are commonly used for classification tasks, depending on the nature of the data. For example, feedforward neural networks work well with structured data, while convolutional neural networks (CNNs) are ideal for image classification. Recurrent neural networks (RNNs) and long short-term memory

(LSTM) networks, on the other hand, are particularly suited for handling sequential data such as time series or natural language. The selection of activation functions, like ReLU, sigmoid, or $\tanh$ in the hidden layers is crucial for optimizing learning and performance as well. This work uses optimizable neural network architecture to classify parameters based on distance information discussed in the previous sections. The optimizable feature refers to the automatic tuning of key hyperparameters such as the number of fully connected layers, number of neurons per layer or activation functions. This optimization process ensures that the network achieves optimal performance for the given dataset, which allows the model to capture more intricate patterns in the data. This classification process is done through MATLAB's Classification Learner App. The resulting approach can be summarized as shown in Fig. 3.

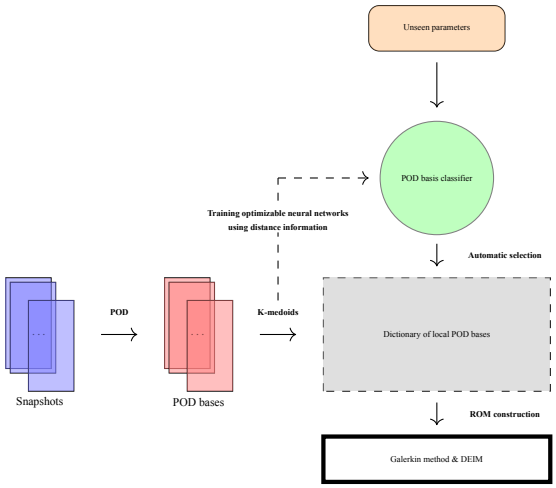


Fig. 3. The workflow of the parametric model reduction procedure proposed in this work.

From Fig. 3, in the offline stage, several steps are carried out to prepare the data and train the model. First, in the POD step, the process begins with computing the POD basis for each snapshot individually. Each snapshot, with dimensions $n \times m$, is processed through SVD, which has a complexity of $O(n \cdot m \cdot \min(n, m))$. Given that there are s snapshots, the total complexity for computing the POD for all snapshots individually is $O(s \cdot n \cdot m \cdot \min(n, m))$. The next step is to calculate the distances between each pair of POD bases. This involves matrix multiplications and the computation of SVD for each pair. The total number of pairwise comparisons is $O(s^2)$, which results in a complexity for this step of $O(s^2 \cdot (n \cdot m)^2)$. Based on the computation of distances, the complexity of this clustering step is $O(K \cdot s^2 \cdot iter)$, where K is the number of clusters, and $iter$ is the number of iterations. Finally, the distance information is used to train a classifier in the offline stage. Since the Classification Learner App automates these steps, it abstracts away the underlying computations. As a result, the computational complexity of this step is difficult to assess precisely. It depends on factors such as the size of the input data (number of samples and features) and the complexity of the neural network architecture (e.g., the number of layers and neurons). While the exact complexity is challenging to determine, the training time will be shown in the numerical experiment to provide insight into the computational cost of training the model with the chosen architecture.

4. Numerical Experiments on the Parameterized Burgers' Equation

The Burgers' equation is a fundamental partial differential equation that combines both nonlinear convection and diffusion effects, often used to simplify models for fluid dynamics, traffic flow and gas dynamics. In its parameterized form, the equation may include parameters that

influence the initial conditions, boundary conditions or the physical properties of the system. In this work, we study the parameterized form of Burgers' equation governed by parameter $(Re, \alpha, \gamma) \in \mathcal{P} = (1, 5) \times (-2, 2) \times (-2, 2)$ given by

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} = \frac{1}{Re}\frac{\partial^2 u}{\partial x^2}, \tag{4.1}$$

$(x, t) \in [0, 2\pi] \times [0, 2]$, where Re is a constant parameter known as Reynolds number satisfies the initial condition

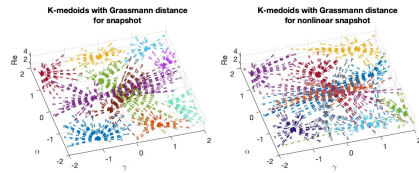
$$u(x, 0) = \alpha \cos(x) + \gamma \sin(x), \tag{4.2}$$

which describes the system's state at time $t = 0$, incorporating both cosine and sine terms parameterized by α and γ and the boundary condition

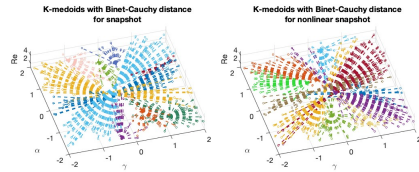
$$u(0, t) = \alpha, \tag{4.3}$$
$$u(2\pi, t) = \alpha, \tag{4.4}$$

indicating that system is subject to constant boundary values of α at both ends. To numerically solve (4.1), we use a simple central finite difference scheme. The spatial and temporal domains are discretized into $n = 100$ and $m = 5000$ grid points, respectively. Furthermore, we generate $s = 500$ samples varying the parameter (Re, α, γ) over a chosen parameter range $\mathcal{P}$.

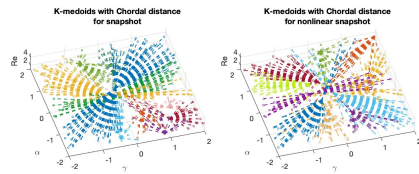
Note that, in Fig. 4, the positions of the cluster labels correspond to the medoids, which are represented by •. Points that belong to the same cluster are color-coded accordingly. For easier visualization, dashed lines are drawn from each data point to its respective medoid, highlighting the connections between the points and their cluster medoids. In Fig. 4, we present the results of the K-medoids with $K = 10$. This is done primarily to make the patterns more visually discernible and easier to interpret. Three different distance measures reveal distinct patterns in the data.



(a) Grassmann distance: Snapshots (left) and nonlinear snapshots (right)



(b) Binet-Cauchy distance: Snapshots (left) and nonlinear snapshots (right)



(c) Chordal distance: Snapshots (left) and nonlinear snapshots (right)

Fig. 4. The results of K-medoids clustering with $K = 10$, using Grassmann (top), Binet-Cauchy (middle), and Chordal (bottom) distances. This comparison highlights how the choice of distance metric affects clustering behavior for each type of data representation.

In particular, the results for Binet-Cauchy and Chordal distances are nearly identical, exhibiting minimal variation in cluster patterns. This observation indicates that these two distance metrics produce the similar clustering outcomes, which might suggest they capture analogous aspects of the data's structure. In contrast, the clustering based on the Grassmann distance demonstrates a more varied distribution of clusters, with several distinct grouping. This difference suggests that the Grassmann distance provides a more nuanced differentiation between clusters compared to those two distances. The Grassmann dis-

tance captures subtler variations in the solution spaces and reveals more complicated patterns that are not apparent with other distance measures. This disparity highlights the sensitivity of the clustering results to the choice of distance metric and underscores the importance of selecting an appropriate distance measure based on the specific characteristics of the problems. As shown in Fig. 5, using a cluster count of $K = 10$ is insufficient for clearly distinguishing local POD bases. Consequently, the resulting ROMs lack stability, as the selected POD bases may not adequately capture the essential characteristics of the parameter space of interest. The small number of clusters also leads to high training accuracy, as the training data remains relatively simple and less complex demonstrated in Table 1. Despite this high accuracy, the number of clusters is still insufficient for effective classification, leading to a failure in properly distinguishing the local POD bases.

Table 1. Comparison of classifier performance based on different distance measures, showing the accuracy of classifiers during training phase for $K = 10$.

Distance type	K	Dictionary type	Training accuracy
Grassmann	10	Galerkin	90.8%
		DEIM	85.4%
Binet-Cauchy	10	Galerkin	87.0%
		DEIM	90.2%
Chordal	10	Galerkin	91.8%
		DEIM	92.4%

Based on more detailed analysis and underlying structure of the data, the actual number of clusters is determined to be $K = 30$, as shown in Fig. 6. It is important to note that while a higher number of clusters can offer a more accurate representation of parameter space, leading to better approximation and potentially enhanced performance in reduced-order modeling. It also comes with increased computational costs.

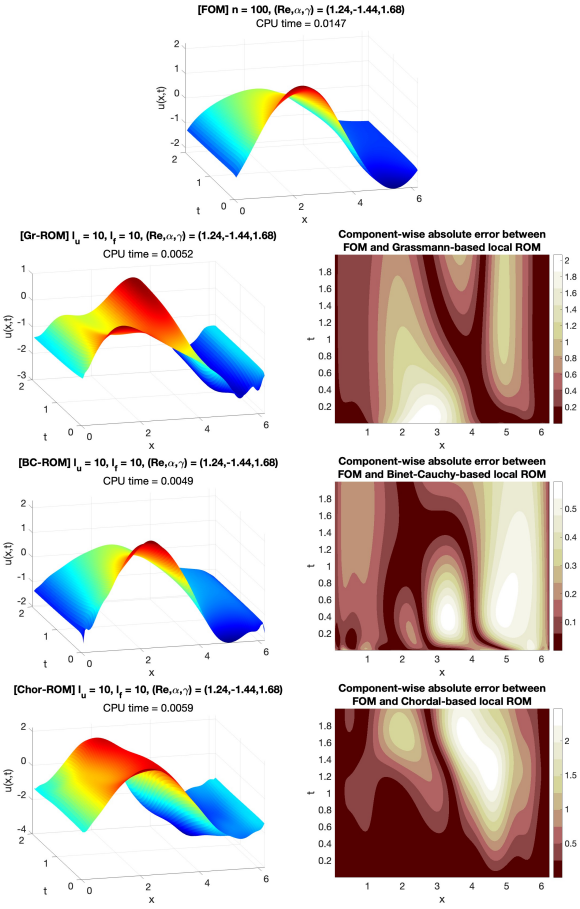
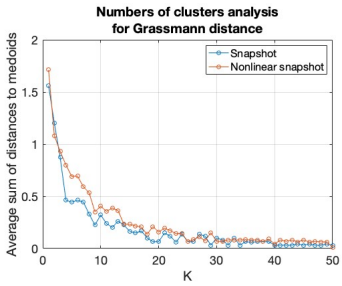


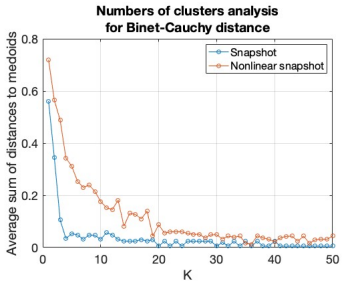
Fig. 5. Using $K = 10$, the ROM solutions corresponding to Grassmann distance (Gr-ROM), Binet-Cauchy distance (BC-ROM) and Chordal distance (Chor-ROM) of size $l_u = l_f = 10$ for parameter $(Re, \alpha, \gamma) = (3, 0, 1)$ are shown and compared to the FOM of size $n = 100$.

In this case, $K = 30$ is used specifically to investigate the effect of different distance measures on the clustering results. Therefore, for each distance metric, we have two local POD basis dictionaries, one for Galerkin projection and one for DEIM. For the classifier task, we employ a 5-fold cross-validation scheme. Optimizable neural networks are trained to serve as POD basis classifiers for both the Galerkin projection (POD basis for snapshots) and DEIM (POD basis for nonlinear snapshots)

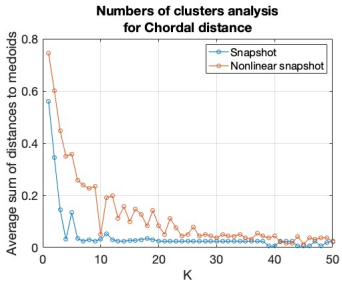
across each distance measure. Note that each classifier takes approximately 200 seconds to complete the training phase. This results in a total of 6 classifiers, with the classification results presented in Table 2.



(a) Grassmann distance: Comparison of snapshots (blue) vs. nonlinear snapshots (orange)



(b) Binet-Cauchy distance: Comparison of snapshots (blue) vs. nonlinear snapshots (orange)



(c) Chordal distance: Comparison of snapshots (blue) vs. nonlinear snapshots (orange)

Fig. 6. Plots of average sum of distances to medoids using different numbers $K = 1, 2, \ldots, 50$ using Grassmann (top), Binet-Cauchy (middle), and Chordal (bottom) distances. For each distance measure, we compare the appropriate K for both snapshots and nonlinear snapshots, highlighting the clustering performance across different values of K.

Table 2. Comparison of classifier performance based on different distance measures, showing the accuracy of classifiers during training phase for $K = 30$.

Distance type	K	Dictionary type	Training accuracy
Grassmann	30	Galerkin	88.8%
		DEIM	83.0%
Binet-Cauchy	30	Galerkin	83.8%
		DEIM	80.0%
Chordal	30	Galerkin	77.4%
		DEIM	78.6%

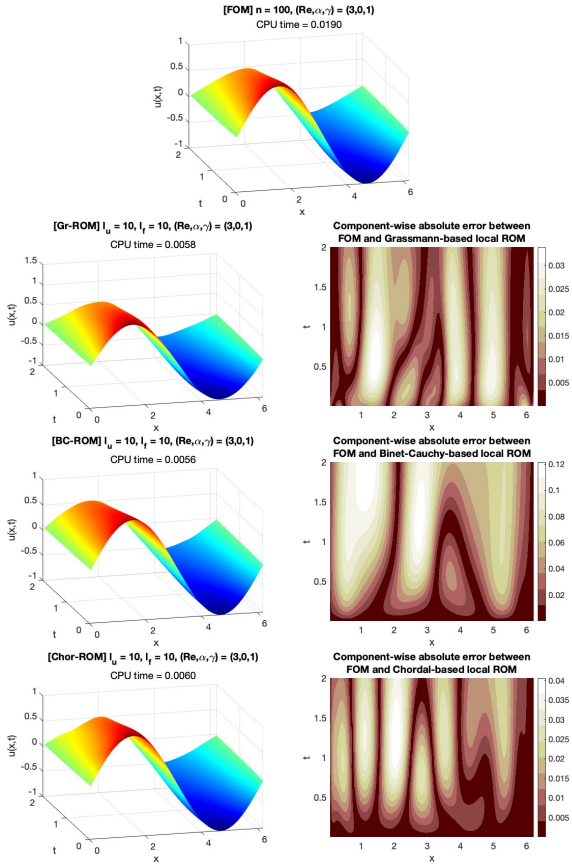


Fig. 7. Using $K = 30$, the ROM solutions corresponding to Grassmann distance (Gr-ROM), Binet-Cauchy distance (BC-ROM) and Chordal distance (Chor-ROM) of size $l_u = l_f = 10$ for parameter $(Re, \alpha, \gamma) = (3, 0, 1)$ are shown and compared to the FOM of size $n = 100$.

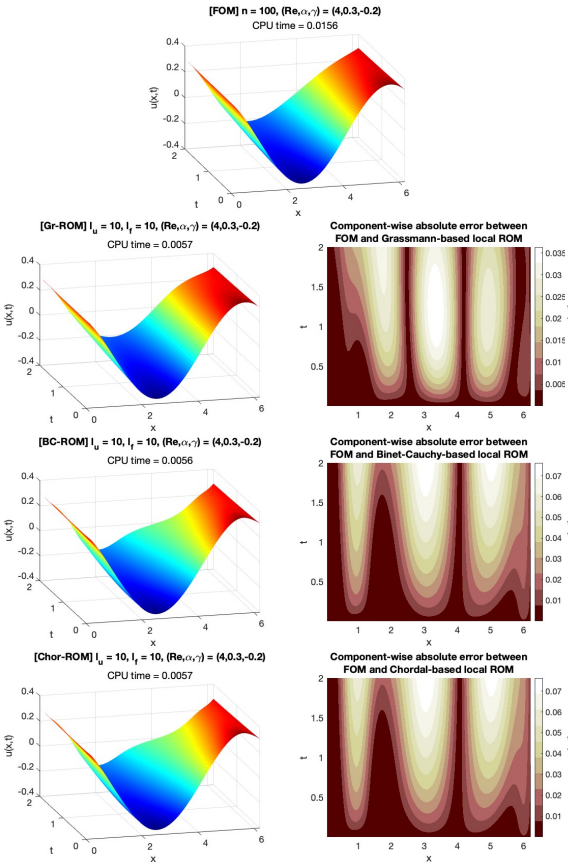


Fig. 8. Using $K = 30$, the ROM solutions corresponding to Grassmann distance (Gr-ROM), Binet-Cauchy distance (BC-ROM) and Chordal distance (Chor-ROM) of size $l_u = l_f = 10$ for parameter $(Re, \alpha, \gamma) = (4, 0.2, -0.2)$ are shown and compared to the FOM of size $n = 100$.

Fig. 9. Using $K = 30$, the ROM solutions corresponding to Grassmann distance (Gr-ROM), Binet-Cauchy distance (BC-ROM) and Chordal distance (Chor-ROM) of size $l_u = l_f = 10$ for parameter $(Re, \alpha, \gamma) = (2.75, -0.4, 0.2)$ are shown and compared to the FOM of size $n = 100$.

Figs. 7-10 clearly demonstrate that the Gr-ROM provides more accurate results compared to the other ROMs. The BC-ROM and Chor-ROM exhibit noticeable instability, which is easily visible in the figures. In contrast, the Gr-ROM maintains stability and closely matches the reference FOM solution, underscoring its superior performance in terms of accuracy and robustness. To demonstrate the efficiency of our proposed method, we define the error between the full-order model (FOM) and the reduced-order model (ROM) approximation as follows:

$$\text{AvgError} = \frac{\sqrt{\sum_{j=1}^{m} \|\mathrm{u}_j - \mathrm{V}^u \tilde{\mathrm{u}}_j\|_2^2}}{nm} \quad (4.5)$$

$$= \frac{\|\mathrm{U}^{\mathrm{FOM}} - \mathrm{U}^{\mathrm{ROM}}\|_F}{nm}. \quad (4.6)$$

To provide more detailed insights, Table 3 presents a comparison between the FOM and various ROMs under different parameter settings.

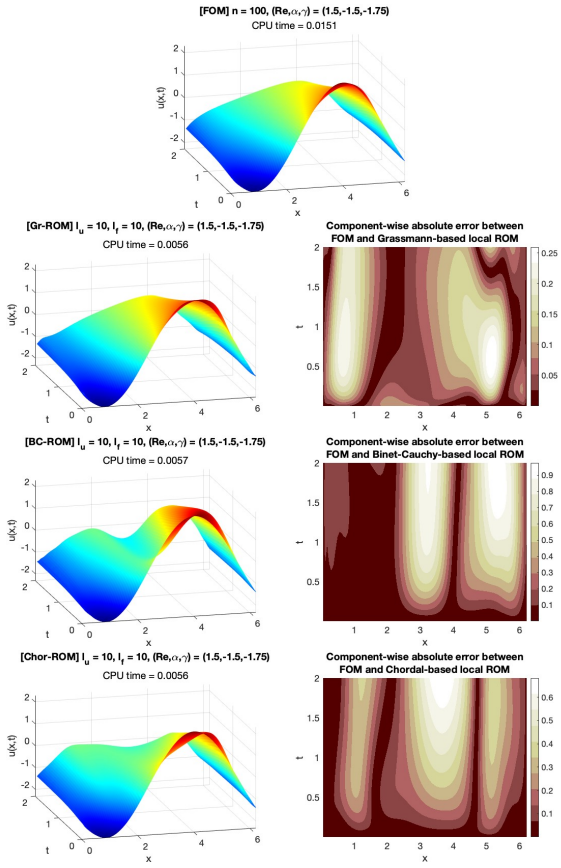


Fig. 10. Using $K = 30$, the ROM solutions corresponding to Grassmann distance (Gr-ROM), Binet-Cauchy distance (BC-ROM) and Chordal distance (Chor-ROM) of size $l_u = l_f = 10$ for parameter $(Re, \alpha, \gamma) = (1.5, -1.5, -1.75)$ are shown and compared to the FOM of size $n = 100$.

Notably, the Gr-ROM consistently outperforms both BC-ROM and the Chor-ROM in terms of accuracy, as reflected in the AvgError across different parameter configurations. Additionally, the CPU time for Gr-ROM, BC-ROM, and Chor-ROM is similar across models since they share the same reduced dimensionality and ROMs require approximately four times less CPU time compared to the FOM.

Table 3. Comparison AvgError as defined in (4.5) and scaled CPU time usage for each parameter (Re, α, γ) corresponding to each distance.

(Re, α, γ)	Approach	Size	AvgError	Scaled CPU time
$(2.4, 1.33, 0.64)$	FOM	100	-	1
	Gr-ROM	10	3.36×10^{-4}	0.2460
	BC-ROM	10	1.90×10^{-3}	0.2503
	Chor-ROM	10	1.10×10^{-3}	0.2648
$(3.33, 1.29, -0.66)$	FOM	100	-	1
	Gr-ROM	10	8.63×10^{-4}	0.2261
	BC-ROM	10	N/A	0.2831
	Chor-ROM	10	2.9×10^{-3}	0.2964
$(4, -1.19, 1.44)$	FOM	100	-	1
	Gr-ROM	10	7.52×10^{-4}	0.2764
	BC-ROM	10	1.30×10^{-3}	0.2845
	Chor-ROM	10	8.34×10^{-4}	0.2882
$(1.76, -1.27, -1.39)$	FOM	100	-	1
	Gr-ROM	10	1.30×10^{-4}	0.2988
	BC-ROM	10	4.52×10^{-4}	0.2832
	Chor-ROM	10	2.84×10^{-4}	0.2838

5. Conclusion

In conclusion, this work investigated the impact of different distance measures in constructing a dictionary of POD bases using the K-medoids clustering algorithm. Specifically, we explored several distance metrics to partition the parameter space for reduced-order modeling of the parameterized Burgers' equation, where the system dynamics, initial conditions, and boundary conditions are all parameter-dependent. The results clearly demonstrate that the Gr-ROM outperforms other ROM approaches, particularly those based on Binet-Cauchy and Chordal distances, providing superior accuracy and stability across varying parameter settings.

Note that, given the nature of the study in this work, some aspects rely on pre-set configurations within the computational framework, making it challenging to pinpoint a unified theoretical contribution. However, the results given in this work would be a crucial starting point for more advanced theoretical and computational investigations.

References

[1] Chen D, Song H. The POD–DEIM reduced-order method for stochastic Allen–Cahn equations with multiplicative noise. Computers & Mathematics with Applications. 2020;80(12):2691-706.

[2] Roy A, Nabi M. Modeling of MEMS Electrothermal Microgripper employing POD-DEIM and POD method. Microelectronics Reliability. 2021;125:114338.

[3] Li J,Fan X,Wang Y,Yu B,Sun S, Sun D. A POD-DEIM reduced model for compressible gas reservoir flow based on the Peng-Robinson equation of state. Journal of Natural Gas Science and Engineering. 2020;79:103367.

[4] Isoz M. POD-DEIM based model order reduction for speed-up of flow parametric studies. Ocean Engineering. 2019;186:106083.

[5] Sipp D, Fosas de Pando M, Schmid PJ. Nonlinear model reduction: A comparison between POD-Galerkin and POD-DEIM methods. Com- puters & Fluids. 2020;208:104628.

[6] Yang H, Veneziani A. Efficient estimation of cardiac conductivities via POD-DEIM model order reduction. Applied Numerical Mathematics. 2017;115:180-99.

[7] Bizon K, Continillo G. Optimal Design of a Non-isothermal Hybrid Catalyst Pellet based on POD-DEIM Reduced-order Methodology. 2020;48:271-6.

[8] García-Archilla B, John V, Novo J. Second order error bounds for POD- ROM methods based on first order divided differences. Applied Mathematics Letters. 2023;146:108836.

[9] Chaturantabut S, Sorensen DC. A State Space Error Estimate for POD-DEIM Nonlinear Model Reduction. SIAM Journal on Numerical Analysis. 2012;50(1):46-63.

[10] Daniel T, Casenave F, Akkari N, Ketata A, Ryckelynck D. Physics-informed cluster analysis and a priori efficiency criterion for the construction of local reduced-order bases. Journal of Com- putational Physics. 2022;458:111120.

[11] Cortés J, Herrero H, Pla F. A local ROM for Rayleigh–Bénard bifurcation problems. Computer Methods in Applied Mechanics and Engineering. 2024;425:116949.

[12] Sukuntee N, Chaturantabut S. Parametric Nonlinear Model Reduction Using K-Means Clustering for Miscible Flow Simulation. Journal of Applied Mathematics. 2020;2020(1):3904606.

[13] Sukuntee N, Chaturantabut S. Dictionary-based Model Order Reduction via POD-DEIM with Support Vector Machine for the Parametrized Burgers' Equation. Thai Journal of Mathematics. 2022 Jan;Special Issue (2022): Annual Meeting in Mathematics 2021:38-52.

[14] Ye K, Lim LH. Schubert Varieties and Distances between Subspaces of Different Dimensions. SIAM Journal on Matrix Analysis and Applications. 2016;37(3):1176-97.