Commutation Error in Reduced Order Modeling

Birgul Koc

Thesis submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Master of Science
in
Mathematics

Traian Iliescu, Chair
Jeffrey T Borggaard
Serkan Gugercin

October 1, 2018
Blacksburg, Virginia

Keywords: Reduced Order Modeling, Data-Driven Modeling, Filtering, Closure Modeling, Commutation Error

Commutation Error in Reduced Order Modeling

Birgul Koc

(ABSTRACT)

We investigate the effect of spatial filtering on the recently proposed data-driven correction reduced order model (DDC-ROM). We compare two filters: the ROM projection, which was originally used to develop the DDC-ROM, and the ROM differential filter, which uses a Helmholtz operator to attenuate the small scales in the input signal. We focus on the following questions: "Do filtering and differentiation with respect to space variable commute, when filtering is applied to the diffusion term?" or in other words "Do we have commutation error (CE) in the diffusion term?" and "If so, is the commutation error data-driven correction ROM (CE-DDC-ROM) more accurate than the original DDC-ROM?" If the CE exists, the DDC-ROM has two different correction terms: one comes from the diffusion term and the other from the nonlinear convection term. We investigate the DDC-ROM and the CE-DDC-ROM equipped with the two ROM spatial filters in the numerical simulation of the Burgers equation with different diffusion coefficients and two different initial conditions (smooth and non-smooth).

Commutation Error in Reduced Order Modeling

Birgul Koc

(THE GENERAL AUDIENCE ABSTRACT)

We propose reduced order models (ROMs) for an efficient and relatively accurate numerical simulation of nonlinear systems. We use the ROM projection and the ROM differential filters to construct a novel data-driven correction ROM (DDC-ROM). We show that the ROM spatial filtering and differentiation do not commute for the diffusion operator. Furthermore, we show that the resulting commutation error has an important effect on the ROM, especially for low viscosity values. As a mathematical model for our numerical study, we use the one-dimensional Burgers equations with smooth and non-smooth initial conditions.

Dedication

To my dear family

Acknowledgments

At some point our lives change suddenly in a good way. In my life, these changes have started with Dr. Songul Kaya Merdan. She encouraged me to start a new life here. I want to thank her. And these good changes continue thanks to many people. This thesis is the result of two years of study. When I look at the difference between my first day and today, I am pleased to notice changes in my knowledge and interest thanks to my advisor, Dr. Traian Iliescu. I am very grateful for his encouragement, endless motivation and support for my research and studies in many ways. He has always told me to move step by step. I hope I will continuously increase my step size. I want thank my academic brothers, Dr. Xuping Xie, Changhong Mou, and Dr. Muhammad Mohebujjaman for their remarkable cooperation. I want to thank Dr. Jeff Borggaard and Dr. Serkan Gugercin for being my committee members and generously offering their time, support, guidance, and goodwill throughout the preparation and review of this document.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Reduced order models (ROMs) [8, 24, 30, 31, 47, 56] are low-dimensional surrogate models that provide an efficient computational alternative to brute force numerical simulations of complex engineering and scientific applications. ROMs are generally constructed in an offline stage from available numerical or experimental data. In an online stage, ROMs are used for longer time intervals and/or parameters that are different from those used in the training stage. ROMs have been used for decades in the efficient numerical simulation of fluid flows that are dominated by relatively few structures [5, 6, 10, 13, 18, 20, 29, 31, 44, 47, 54, 63–65, 73], such as that in Figure 1.1.

Figure 1.1: Vortices around an aiplane's wingtips (https://goo.gl/images/NjTnZg).

However, when the ROM dimension is too low to capture the relevant flow features, ROMs are generally supplemented with a *correction term* [4, 9, 25, 28, 31, 35, 36, 47, 49, 51, 60, 69]. In [71], it was shown that this correction term can be explicitly calculated and modeled with the available data by using a *ROM spatial filter*. ROM spatial filtering has also been used to develop large eddy simulation ROMs, e.g., eddy viscosity ROMs [3, 9, 31, 47, 51, 55, 57, 69] and approximate deconvolution ROMs [72]. To develop all these ROMs, it has been been *assumed* that differentiation and ROM spatial filtering commute:

$$\overline{\frac{\partial u}{\partial x}} = \frac{\partial \overline{u}}{\partial x}, \tag{1.1}$$

where u is a flow variable (such as velocity) and x is a spatial direction. In this thesis, we investigate whether the equality in (1.1) holds, i.e., whether there exists a *commutation error (CE)*. In particular, we investigate whether there is a CE for the Laplacian, which plays a central role in fluid dynamics:

$$\overline{\Delta u} \stackrel{?}{=} \Delta \overline{u}. \tag{1.2}$$

To our knowledge, this represents the *first investigation of the CE in a ROM context*.

When the CE is found to exist in our theoretical and numerical investigation, we check whether the CE has any significant effect on the actual ROM. To this end, we consider the recently proposed *data-driven correction ROM (DDC-ROM)* [71], in which the correction term (which is generally added to improve the ROM's accuracy) is modeled using the available data [16, 42, 43, 50, 53]. To investigate the effect of the CE on the DDC-ROM, we also consider the *commutation error DDC-ROM (CE-DDC-ROM)*, in which available data is used to model not only the correction term, but also the CE. We also consider the *ideal DDC-ROM (IDDC-ROM)*, which is the DDC-ROM supplemented with a fine resolution representation (i.e., without any additional modeling) of the CE. When the CE-DDC-ROM and the IDDC-ROM yield more accurate results than the standard DDC-ROM, we conclude that the CE has a significant effect of the DDC-ROM and, therefore, should be included in the model. As numerical tests, we use the Burgers equation with viscosities $\nu = 10^{-1}$ and $\nu = 10^{-3}$.

Chapter 2

Reduced Order Modeling (ROM)

2.1 Introduction

There are different approaches for constructing ROMs [7, 8, 11, 14, 19, 37–39, 51, 58, 69]. The main idea of all these ROM approaches is to gather significant information from the given system and to convert it to a new reduced system by neglecting the small-scale information. In this thesis, we use the proper orthogonal decomposition (POD) to build the ROM basis. In this chapter, we describe the POD in (Section 2.2) and the Galerkin ROM (G-ROM) in (Section 2.3).

2.2 Proper Orthogonal Decomposition (POD)

Proper orthogonal decomposition (POD) [23, 31, 47, 67] is one of the most popular ROM techniques. POD has been successfully used in various areas, such as signal analysis, pattern recognition, fluid dynamics, control theory, and inverse problems. This technique is a tool to build ROMs by extracting the most dominant spatial structures from direct numerical simulation (DNS). We briefly describe the mathematical formulation of POD [37, 39]. To construct the POD basis, we use snapshots that are obtained by solving the given continuous problem with Finite Element (FE), Finite Volume (FV), Boundary Element Method (BEM), or other numerical methods. Without loss of generality, we use the FE method to find snapshots at different time instances t_j , $j = 1, 2, ..., M$, as follows:

$$u_h(x, t_j) = \sum_{i=1}^{N} \left(u_h^j\right)_i \phi_i^h(x), \tag{2.1}$$

where u_h is FE approximation of u, the solution of the given PDE (e.g., the Navies-Stokes equations or the Burgers equation) and N is the FE space dimension (depending on the

problem, N could be $\mathcal{O}(10^6)$ or even higher) and $\phi_i^h,\ i = 1, ..., N$, are the FE basis functions. We define the snapshot matrix Y such that $Y_{ij} = \left(u_h^j\right)_i\ ,\ i = 1, ..., N,\ \ j = 1, ..., M', (M' \leqslant M)$. If we use all the snapshots to obtain the POD basis, although we get accurate results, the computational cost is often very high. For this reason, we need to truncate the number of snapshots. In this thesis, we use all the snapshots since $M = \mathcal{O}(10^3)$ is not too high. However, if truncation is needed, an important question is which snapshots we need to select and use.

POD seeks a low-dimensional basis that approximates the selected snapshots optimally with respect to a certain norm. In this thesis, we use the $\mathcal{H} = L^2$-norm. To find the ROM basis, i.e., $\{\phi_1, \phi_2, ..., \phi_r\}$, we need to solve the following minimization problem:

$$\min_{(\phi_i,\phi_j)_{\mathcal{H}}=\delta_{ij}} \ \frac{1}{M'} \ \sum_{j=1}^{M'} \left\| u_h(t_j) - \sum_{i=1}^{r} \Big(u_h(t_j), \phi_i(x)\Big)_{\mathcal{H}} \phi_i(x) \right\|_{\mathcal{H}}^2, \tag{2.2}$$

where r represents the number of ROM basis functions (i.e., the ROM dimension). Equation (2.2) shows that there is a relation between ROM (ϕ_i) and FE (ϕ_k^h) basis functions. To find this relation, we first expand the ROM basis functions in terms of the FE basis functions:

$$\phi_i = \sum_{k=1}^{N} C_r(k, i)\phi_k^h. \tag{2.3}$$

Next, we show that (2.2) can be converted to an eigenvalue problem. By solving this eigenvalue problem, we can obtain the POD modes $C_r(:, i)$, where $i = 1, ..., r$. First, we note that

$$\begin{aligned} 0 &\leqslant \left\| u_h(t_j) - \sum_{i=1}^{r} \Big(u_h(t_j), \phi_i\Big)_{\mathcal{H}} \phi_i \right\|_{\mathcal{H}}^2 \\ &= \left(u_h(t_j) - \sum_{i=1}^{r} \Big(u_h(t_j), \phi_i\Big)_{\mathcal{H}} \phi_i \ ,\ u_h(t_j) - \sum_{i=1}^{r} \Big(u_h(t_j), \phi_i\Big)_{\mathcal{H}} \phi_i \right)_{\mathcal{H}}^2 \\ &= \|u_h(t_j)\|_{\mathcal{H}}^2 - 2\sum_{i=1}^{r} \Big(u_h(t_j), \phi_i\Big)_{\mathcal{H}}^2 + \sum_{i=1}^{r}\sum_{k=1}^{r} \Big(u_h(t_j), \phi_i\Big)_{\mathcal{H}}^2 \Big(\phi_i, \phi_i\Big)_{\mathcal{H}} \\ &= \|u_h(t_j)\|_{\mathcal{H}}^2 - \sum_{i=1}^{r} \Big(u_h(t_j), \phi_i\Big)_{\mathcal{H}}^2. \end{aligned} \tag{2.4}$$

By using (2.4), the minimization problem (2.2) can be rewritten as:

$$\min_{(\phi_i,\phi_j)_{\mathcal{H}}=\delta_{ij}} \ \frac{1}{M'} \ \sum_{j=1}^{M'} \left(\|u_h(t_j)\|_{\mathcal{H}}^2 - \sum_{i=1}^{r} \Big(u_h(t_j), \phi_i\Big)_{\mathcal{H}}^2 \right), \tag{2.5}$$

which is equivalent to the following maximization problem:

$$\max_{(\phi_i,\phi_j)_{\mathcal{H}}=\delta_{ij}} \ \frac{1}{M'} \ \sum_{j=1}^{M'} \sum_{i=1}^{r} \Big(u_h(t_j), \phi_i\Big)_{\mathcal{H}}^2. \tag{2.6}$$

To solve the maximization problem (2.6), we use Lagrange multipliers:

$$L\big(\phi_i, \lambda_i\big) = \frac{1}{M'} \sum_{j=1}^{M'} \Big(u_h(t_j), \phi_i\Big)^2_{\mathcal{H}} + \lambda_i\Big(1 - \big(\phi_i, \phi_i\big)_{\mathcal{H}}\Big), \tag{2.7}$$

$$\frac{\partial L(\phi_i, \lambda_i)}{\partial \phi_i} = \frac{2}{M'} \sum_{j=1}^{M'} \Big(u_h(t_j), \phi_i\Big)_{\mathcal{H}} u_h(t_j) - 2\lambda_i \phi_i = 0 \tag{2.8}$$

where λ_i is auxiliary variable. Solving (2.8) for λ_i and using (2.1) and (2.3), the minimization problem (2.2) can be converted to the following eigenvalue problem: $\forall i = 1, ..., N$,

$$\frac{1}{M'} Y Y^T M_h \{C_r\}_i = \lambda_i \{C_r\}_i. \tag{2.9}$$

If the dimension of the FE approximation N is much larger than the number of time instances M', solving eigenvalue problem (2.9) is not efficient. For this reason , by using the method of snapshots, we solve instead the following eigenvalue problem [66]:

$$K v_i = \lambda_i v_i, \tag{2.10}$$

where K is correlation matrix $K = \frac{1}{M'} Y^T M_h Y$. Since matrix K is real and symmetric, the eigenvalues λ_i are real and non-negative: $\lambda_1 \geqslant \lambda_2 \geqslant ... \geqslant \lambda_d > \lambda_{d+1} = ... = \lambda_{M'} = 0$, where d is rank of K. The POD modes $(C_r)_i$ are defined as:

$$C_r(:, i) = \frac{1}{\sqrt{M'}} \frac{1}{\sqrt{\lambda_i}} Y v_i \ , \qquad i = 1, ..., r, \tag{2.11}$$

where $r \leqslant d \leqslant M'$. If we use d as ROM dimension, we get accurate results. For efficiency, however, we use low r values, which yield accurate results and guarantee a low computational cost at the same time.

In Figure 2.1, we illustrate the differences and relationships between FE and POD basis functions. The left graph represents the fifth FE basis function, which satisfies $\phi_i^h(x_j) = \delta_{ij}$. All the FE basis functions are local hat functions with changing positions. The global POD basis function (right graph) is a linear combination of FE basis functions. Indeed, by (2.3), we have $\phi_1 = \sum_{k=1}^{10} C_r(k, 1) \phi_k^h$. Thus, $C_r(k, 1)$ represents the value of the POD basis function ϕ_1 at the FE mesh point $x_k = k\frac{1}{10}$. In (2.12), we list the entries in $(C_r)_1$, which represent the coefficients of FE basis functions used to obtain the first POD basis function, ϕ_1.

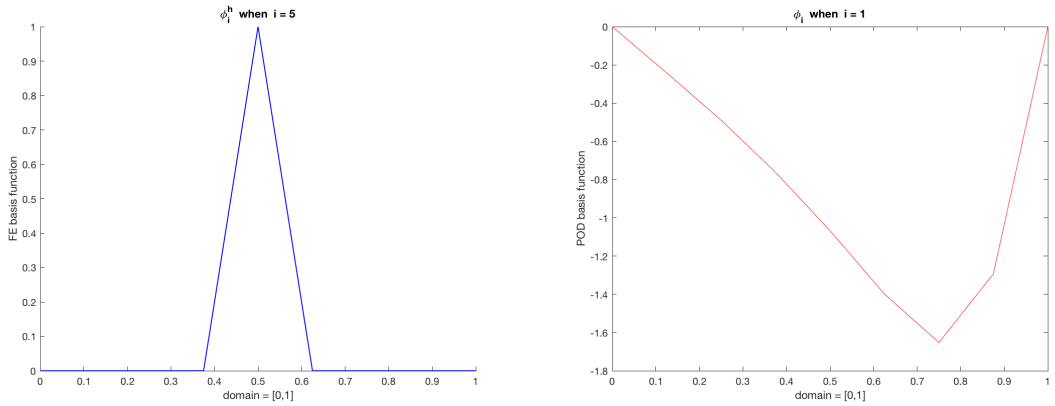

Figure 2.1: The left and right graphs represent FE and POD basis functions, respectively.

$$C_r(:,1) = [0, -2.4175e-01, -4.8960e-01, -7.6206e-01, -1.0672e+00, \\ -1.3988e+00, -1.6519e+00, -1.2938e+00, 0]^T. \tag{2.12}$$

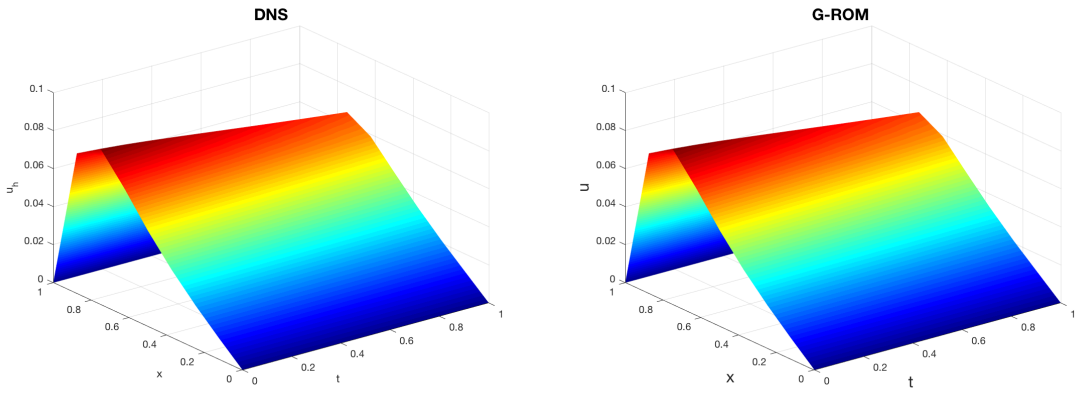

Figure 2.2: The graphs that are labeled as DNS and G-ROM are solutions of Burgers equation by using FE and ROM, respectively. The former uses 10 FE basis functions, whereas the latter uses only 3 ROM basis functions.

2.3 Standard Galerkin ROM (G-ROM)

In this section, we construct the Galerkin ROM (G-ROM), which is one of the most common ROM approaches. To build the G-ROM, we need a mathematical model. In this thesis, we

use the 1D Burgers equation with homogeneous Dirichlet boundary conditions as a mathematical model.

Burgers Equation :

$$\begin{cases} u_t - \nu u_{xx} + u u_x = 0 & in \ \ \Omega \\ u = 0 & on \ \ \partial\Omega \\ u(x,0) = u_0(x) & in \ \ \Omega \ \ , \end{cases} \tag{2.13}$$

where u and ν represent the velocity and diffusion parameter, respectively. Equation (2.13) is continuous and dimensionless. To apply ROMs, the problem should be finite-dimensional. To this end, we first create a POD approximation of the velocity as a linear combination of POD basis functions:

$$u_r(x,t) = \sum_{j=1}^{r} (a_r(t))_j \phi_j(x), \tag{2.14}$$

where $\{(a_r(t))_j\}_{j=1}^{r}$ are time-varying unknowns in the dynamical system for G-ROM. Next, we replace u with u_r in (2.13) and project the equation that is obtained from the previous step onto $X^r \subset X = H_0^1(\Omega)$, which is spanned by the POD basis, i.e., $X^r = span\{\phi_1, \phi_2, ..., \phi_r\}$. This yields the weak form for G-ROM:

$$\Big((u_r)_t, \phi_i\Big)_{\mathcal{H}} + \nu \Big((u_r)_x, (\phi_i)_x\Big)_{\mathcal{H}} + \Big(u_r (u_r)_x, \phi_i\Big)_{\mathcal{H}} = 0 \quad , \quad \forall i = 1, ..., r, \tag{2.15}$$

where we used that $-\nu\Big((u_r)_{xx}, \phi_i\Big)_{\mathcal{H}} = \nu\Big((u_r)_x, (\phi_i)_x\Big)_{\mathcal{H}}$ since the POD basis functions vanish at the boundary. By inserting (2.14) into the weak form (2.15), we get the following nonlinear time-dependent dynamical system:

$$\mathbf{G-ROM}: \quad \boxed{\dot{a_r} = A\, a_r + a_r^T\, B\, a_r} \tag{2.16}$$

and its componentwise formulation: $\forall i = 1, ..., r$,

$$(\dot{a_r})_i = \sum_{m=1}^{r} A_{im} a_m + \sum_{m=1}^{r}\sum_{n=1}^{r} B_{imn} a_n a_m, \tag{2.17}$$

where A_{im} and B_{imn} are given by

$$A_{im} = -\nu \Big((\phi_m)_x, (\phi_i)_x\Big)_{\mathcal{H}} = -\nu \sum_{j_1=1}^{N}\sum_{j_2=1}^{N} C_r(j_1, m) C_r(j_2, i) \Big((\phi_{j_1}^h)_x, (\phi_{j_2}^h)_x\Big)_{\mathcal{H}}, \tag{2.18}$$

$$B_{imn} = -\Big(\phi_m (\phi_n)_x, \phi_i\Big)_{\mathcal{H}} = -\sum_{j_1=1}^{N}\sum_{j_2}^{N}\sum_{j_3=1}^{N} C_r(j_1, m) C_r(j_2, n) C_r(j_3, i) \Big(\phi_{j_1}^h (\phi_{j_2}^h)_x, \phi_{j_3}^h\Big)_{\mathcal{H}}. \tag{2.19}$$

For laminar flows, G-ROM is efficient, since using few POD basis functions is enough to capture the most important information from the dynamics of the given problem. The reason is that the discarded (unresolved) POD modes $\{\phi_i\}_{i=r+1}^{d}$ do not have a significant impact on the resolved modes $\{\phi_i\}_{i=1}^{r}$. However, for convection-dominated flows (where the viscous parameter ν is small), G-ROM could fail because the small structures are important. Thus, for convection-dominated flows we need a correction term that models the interaction between resolved and unresolved modes.

Chapter 3

Spatially Filtered ROMs

3.1 Introduction

As we have explained in the previous chapter, G-ROM can sometimes yield inaccurate results if the system is nonlinear or convection-dominated (i.e., the diffusion parameter is small). In this case, the interaction between resolved and unresolved modes becomes crucial. For this reason, we need to add a correction term [4, 9, 25, 28, 31, 35, 36, 47, 49, 51, 60, 69] to G-ROM (2.16) to capture the missing interaction between resolved and unresolved POD modes. In this section, we explain how ROM spatial filtering can be used to determine the explicit form of this interaction.

3.2 ROM Spatial Filtering

Spatial filtering has been used in ROMs, mainly as a preprocessing tool to eliminate the noise in data. In our approach, we use spatial filtering [1, 2, 17, 21, 26, 48, 51, 68–70] during the construction of the actual ROM, not in the development of the POD basis. The aim of ROM filtering is to get rid of small length scales, since their approximation increases the computational cost. We present two different ROM filtering approaches: the ROM projection and the ROM differential filter. In the next sections, we use the notations u_d, $\overline{u_d}^r$ and $\overline{u_d}^{DF}$, which represent functions in X^d (ROM space of dimension d) and the solutions in X^r (ROM space of dimension r) that are obtained after applying the projection and differential filters to u_d, respectively. The solution $u_d \in X^d$ and $\overline{u_d}^r, \overline{u_d}^{DF} \in X^r$ can be written as linear combinations of ROM basis functions with suitable dimensions in the following way:

$$u_d = \sum_{j=1}^{d} (a_d)_j \phi_j, \tag{3.1}$$

$$\overline{u_d}^r = \sum_{j=1}^{r} \overline{(a_d)}_j^r \phi_j, \tag{3.2}$$

$$\overline{u_d}^{DF} = \sum_{j=1}^{r} \overline{(a_d)}_j^{DF} \phi_j. \tag{3.3}$$

In equations (3.2) and (3.3), when we filter the solution u_d, we need to apply filtering to the time-dependent variable a_d, which is the coefficient of the ROM basis function ϕ_i.

3.2.1 Projection Filter (PF)

For fixed $r \leqslant d$ and a given $u_d \in X^d$, the projection filter [48, 51, 69] seeks $\overline{u_d}^r \in X^r$ such that

$$\left(\overline{u_d}^r, \phi_i\right)_{\mathcal{H}} = \left(u_d, \phi_i\right)_{\mathcal{H}} \quad , \qquad \forall i = 1, ...r. \tag{3.4}$$

In this thesis, we will use the L^2-norm for $\mathcal{H}$; other norms could also be used. By solving (3.4), we get the filtered solution $\overline{u_d}^r$ for the given ROM solution u_d.

3.2.2 Differential Filter (DF)

For any given ROM solution $u_d \in X^d$, by using the ROM differential filter [70], we can find the filtered ROM solution $\overline{u_d}^{DF} \in X^r$, where $r \leq d$ as:

$$\left(\left(\mathbb{I} - \delta^2 \Delta\right)\overline{u_d}^{DF}, \phi_i\right)_{\mathcal{H}} = \left(u_d, \phi_i\right)_{\mathcal{H}} \quad , \qquad \forall i = 1, ..., r, \tag{3.5}$$

where $\mathbb{I}$, δ, Δ, and " $-^{DF}$ " represent the identity matrix, radius of differential filter (DF), Laplace operator, and DF filtering, respectively. It is critical to find the best suitable δ value in the differential filter: If we choose δ in (3.5) too small, the results from (3.4) and (3.5) will be very close. In this thesis, we use $\mathcal{H} = L^2$ norm, but other norms could also be applied.

There are two main approaches [72] to compute $\overline{u_d}^{DF}$. For given $u_d = \sum_{i=1}^{d} (a_d)_i \phi_i \in X^d$, to obtain $\overline{u_d}^{DF}$ with the ROM approach we filter the time-dependent coefficients a_d, and write the filtered solution as $\overline{u_d}^{DF} = \sum_{i=1}^{r} (\overline{a_d}^{DF})_i \phi_i$. In the FE approach, we filter the ROM basis functions ϕ_i and express the filtered solution as $\overline{u_d}^{DF} = \sum_{i=1}^{r} a_r \overline{\phi_i}^{DF}$, where a_r contains first r entries of a_d.

Case 1: ROM Approach

In this approach, we filter only the time-dependent variable a_d when we apply filtering to solution u_d. By inserting (3.1) and (3.3) into (3.5), we find the unknown $\overline{a_d}^{DF}$ from the following equation:

$$(M_r + \delta^2 S_r)\overline{a_d}^{DF} = (M_{r\times d})a_d, \tag{3.6}$$

where $M_r = (\phi_i, \phi_j)_{\mathcal{H}}$, $M_{r\times d} = (\phi_i, \phi_k)_{\mathcal{H}}$ and $S_r = \Big((\phi_i)_x, (\phi_j)_x\Big)_{\mathcal{H}}$ with $i, j = 1, .., r$, $k = 1, .., d$ represent the ROM square and rectangular mass matrices and the square stiffness matrix, respectively. Solving (3.6) at each time step is efficient because the dimension of the linear system is small. Furthermore, we do not need to store $\overline{a_d}^{DF}$ for different time levels.

Case2: FEM Approach

The main idea in this approach is to write ROM solutions in terms of FE basis functions (ϕ_k^h). Moreover, filtering is applied to ROM basis functions $\{\phi_i\}_{i=1}^r$, not to time-dependent coefficients. By using (2.3) and (3.3), we can derive the following equality for filtered solution $\overline{u_d}^{DF}$:

$$\overline{u_d}^{DF} = \sum_{i=1}^{r}(a_r)_i \overline{\sum_{k=1}^{N} C_d(k,i)\phi_k^h}^{DF} = \sum_{i=1}^{r}(a_r)_i \sum_{k=1}^{N} \overline{C_d(k,i)}^{DF}\phi_k^h. \tag{3.7}$$

Then, using (3.7) in (3.5) and (2.3), the following system arises:

$$\overline{C_d}^T(M_h + \delta^2 S_h)C_r a_r = C_r^T M_h C_d a_d = C_r^T M_h C_r a_r, \tag{3.8}$$

where the last equality holds because for $i = 1, ..., r$ and $j = 1, ..., d$, $(\phi_i, \phi_j)_{L^2} = C_r^T M_h C_d = C_r^T M_h C_r$ thanks to the orthogonality of POD basis functions. In (3.8), a_r contains the first r entries of a_d and C_r and C_d represent the first r and d POD modes, respectively. Furthermore, $D_r := \overline{C_d}$ contains the filtered POD modes, which can be found by solving the following equation:

$$C_r^T(M_h + \delta^2 S_h)D_r = C_r^T M_h C_r. \tag{3.9}$$

If a fine mesh is used to obtain the FE solution, the dimension of (3.9) is large. However, the filtered POD modes are computed and stored in the offline stage because there is no term that depends on time.

3.3 Filtered-ROM (F-ROM) Framework and Large Eddy Simulation ROM (LES-ROM)

In this section, our aim is to apply explicit ROM spatial filtering to the given mathematical model (1D Burgers equation). "—" is general notation for both the differential and the projection filter. To construct the F-ROM framework, we follow the standard LES approach [12, 21, 32–34, 40, 59], which contains two main steps:
(i) Use explicit spatial filtering of the given continuous equation. After filtering is applied to the continuous Burgers equation, the weak form of the continuous filtered Burgers equation is:

$$\Big(\overline{u_t},\phi\Big)_{\mathcal{H}} - \nu\Big(\overline{u_{xx}},\phi\Big)_{\mathcal{H}} + \Big((\overline{uu_x},\phi\Big)_{\mathcal{H}} = 0, \tag{3.10}$$

where ϕ represents a test function. The weak form of Burgers equation without filtering is

$$\Big(u_t,\phi\Big)_{\mathcal{H}} - \nu\Big(u_{xx},\phi\Big)_{\mathcal{H}} + \Big(uu_x,\phi\Big)_{\mathcal{H}} = 0. \tag{3.11}$$

When we compare (3.11) with (3.10) in terms of length scales, the latter has smaller length scales than the former. Spatial filtering eliminates small length scales, which require memory and time. Thus, the weak form of the spatially filtered Burgers equation will require fewer POD modes than the continuous case; this is the advantage of filtering in terms of computational cost.
(ii) After filtering has been applied to the continuous Burgers equation, it is still not computationally useful because we need a finite-dimensional system. If we express the continuous solution in terms of the discrete solution by using a ROM approximation, we can apply filtering to this finite discrete solution to obtain a practical ROM.

The ROM approximation for the continuous solution u is

$$\boxed{u \approx u_d = \sum_{i=1}^{d} a_d \phi_i \ ,} \tag{3.12}$$

where u and u_d represent the continuous and discrete solution of the 1D Burgers equation, respectively. The discrete solution u_d, which is a linear combination of d ROM basis functions, is the best available finite-dimensional discrete approximation of the continuous solution u. By using (3.12) into (3.10) and choosing test functions in (3.10) as ROM basis functions $\{\phi_i\}_{i=1}^{r}$, we get the following equation: $\forall i = 1, ..., r$,

$$\Big(\overline{(u_d)_t},\phi_i\Big)_{\mathcal{H}} - \nu\Big(\overline{(u_d)_{xx}},\phi_i\Big)_{\mathcal{H}} + \Big(\overline{u_d(u_d)_x},\phi_i\Big)_{\mathcal{H}} = 0. \tag{3.13}$$

Next, we check whether differentiation with respect to variable x and filtering commute for both the projection and differential filter.

Projection Filter : By using the notation in (3.2) and (3.4), we have the following equations: $\forall i = 1, ..., r$,

$$\begin{aligned}
\Big(\overline{(u_d)_t}^r, \phi_i\Big)_{\mathcal{H}} = \Big((u_d)_t, \phi_i\Big)_{\mathcal{H}} &\implies M_r\ \dot{\overline{a_d}}^r = M_{r\times d}\ \dot{a}_d \\
\Big(\overline{u_d}^r, \phi_i\Big)_{\mathcal{H}} = \Big(u_d, \phi_i\Big)_{\mathcal{H}} &\implies M_r\ \overline{a_d}^r = M_{r\times d}\ a_d \\
\Big((\overline{u_d}^r)_t, \phi_i\Big)_{\mathcal{H}} &= M_r\ \dot{\overline{a_d}}^r = M_{r\times d}\ \dot{a}_d \\
\Big(\overline{(u_d)_t}^r - (\overline{u_d}^r)_t, \phi_i\Big)_{\mathcal{H}} &= M_r\ (\dot{\overline{a_d}}^r - \dot{\overline{a_d}}^r) = 0.
\end{aligned} \tag{3.14}$$

Differential Filter : By using the notation in (3.3) and (3.5),

$$\begin{aligned}
\Big(\big(I - \delta^2\Delta\big)\overline{(u_d)_t}^{DF}, \phi_i\Big)_{\mathcal{H}} = \Big((u_d)_t, \phi_i\Big)_{\mathcal{H}} &\implies (M_r + \delta^2 S_r)\ \dot{\overline{a_d}}^{DF} = M_{r\times d}\ \dot{a}_d \\
\Big(\big(I - \delta^2\Delta\big)\overline{u_d}^{DF}, \phi_i\Big)_{\mathcal{H}} = \Big(u_d, \phi_i\Big)_{\mathcal{H}} &\implies (M_r + \delta^2 S_r)\ \overline{a_d}^{DF} = M_{r\times d}\ a_d \\
\Big((\overline{u_d}^{DF})_t, \phi_i\Big)_{\mathcal{H}} &= M_r\ \dot{\overline{a_d}}^{DF} = M_r(M_r + \delta^2 S_r)^{-1} M_{r\times d}\ \dot{a}_d \\
\Big(\overline{(u_d)_t}^{DF} - (\overline{u_d}^{DF})_t, \phi_i\Big)_{\mathcal{H}} &= M_r\ (\dot{\overline{a_d}}^{DF} - \dot{\overline{a_d}}^{DF}) = 0.
\end{aligned} \tag{3.15}$$

Thanks to (3.14) and (3.15), we conclude that there exists no commutation error for the time derivative for both filters, i.e., $\overline{(u_d)_t} = (\overline{u_d})_t$. For the diffusion term, we investigate the commutation error in chapter 4. To obtain the filtered-ROM (F-ROM), we rewrite (3.13) as follows: $\forall i = 1, ..., r$,

$$\begin{aligned}
\Big((\overline{u_d})_t, \phi_i\Big)_{\mathcal{H}} + \nu\Big((\overline{u_d})_x, (\phi_i)_x\Big)_{\mathcal{H}} + \Big(\overline{u_d}(\overline{u_d})_x, \phi_i\Big)_{\mathcal{H}} - \nu\Big(\overline{(u_d)_{xx}} - (\overline{u_d})_{xx}, \phi_i\Big)_{\mathcal{H}} + \\
\Big(\overline{u_d(u_d)_x} - \overline{u_d}(\overline{u_d})_x, \phi_i\Big)_{\mathcal{H}} = 0.
\end{aligned} \tag{3.16}$$

Since $\overline{u_d} \in X^r$, we can rename it as $u_r := \overline{u_d}$ where $u_r = \sum_{i=1}^{r} a_r \phi_i$. By using this new notation in (3.16), we obtain:

$$\begin{aligned}
\Big((u_r)_t, \phi_i\Big)_{\mathcal{H}} + \nu\Big((u_r)_x, (\phi_i)_x\Big)_{\mathcal{H}} + \Big(u_r(u_r)_x, \phi_i\Big)_{\mathcal{H}} - \nu\Big(\overline{(u_d)_{xx}} - (u_r)_{xx}, \phi_i\Big)_{\mathcal{H}} + \\
\Big(\overline{u_d(u_d)_x} - u_r(u_r)_x, \phi_i\Big)_{\mathcal{H}} = 0,
\end{aligned} \tag{3.17}$$

which yields the

$$\textbf{Filtered} - \textbf{ROM } (\textbf{F} - \textbf{ROM}): \quad \boxed{\dot{a_r} = A\ a_r + a_r^T\ B\ a_r + \mathcal{E} + \tau,} \tag{3.18}$$

where the expressions for the matrix A, tensor B, and correction terms $\mathcal{E}$ and τ are:

$$A_{im} = -\nu\Big((\phi_m)_x, (\phi_i)_x\Big)_{\mathcal{H}} = -\nu \sum_{j_1=1}^{N}\sum_{j_2=1}^{N} C_r(j_1,m)C_r(j_2,i)\Big((\phi^h_{j_1})_x, (\phi^h_{j_2})_x\Big)_{\mathcal{H}}, \tag{3.19}$$

$$B_{imn} = -\Big(\phi_m(\phi_n)_x, \phi_i\Big)_{\mathcal{H}} = -\sum_{j_1=1}^{N}\sum_{j_2=1}^{N}\sum_{j_3=1}^{N} C_r(j_1,m)C_r(j_2,n)C_r(j_3,i)\Big(\phi^h_m(\phi^h_n)_x \ , \ \phi^h_i\Big)_{\mathcal{H}}, \tag{3.20}$$

$$\tau_i = -(\tau_r^{SFS}, \phi_i)_{\mathcal{H}} = -\Big(\overline{u_d(u_d)_x} - u_r(u_r)_x \ , \ \phi_i\Big)_{\mathcal{H}}, \tag{3.21}$$

where ${\tau_r}^{SFS}$ represents the ROM subfilter-stress tensor, and

$$\mathcal{E}_i = \nu\Big(\overline{(u_d)_{xx}} - (u_r)_{xx}, \phi_i\Big)_{\mathcal{H}}, correctionterm \tag{3.22}$$

where $\overline{(u_d)_{xx}} - (\overline{u_d})_{xx} = \overline{(u_d)_{xx}} - (u_r)_{xx}$ will be discussed in chapter 4 in detail. We note that formally, the F-ROM (3.18) can be written as

$$\boxed{\text{F-ROM} \ = \ \text{G-ROM} + \ \tau \ + \ \mathcal{E}} \tag{3.23}$$

3.4 ROM Closure Modeling

The F-ROM (3.18) is an r-dimensional ODE system; solving this system is computationally efficient since r is much lower than N. However, there is one important issue we need to pay attention to: the F-ROM (3.18) is not a closed system of equations, since the ROM stress tensor τ and $\mathcal{E}$ do not only depend on a_r. To close the F-ROM (3.18), we need to solve the ROM closure problem. There are many methods to solve the closure problem, e.g., eddy-viscosity (EV), approximate deconvolution (AD), and data-driven (DD) modeling [15, 22, 27, 32, 40, 41, 51, 52, 61, 62, 68, 69, 71, 72]. In this thesis, we apply the DD method to solve the closure problem. The main idea in the DD method is to find a formula that depends only on a_r for τ or $\tau + \mathcal{E}$:

$$\textbf{Closure Problem 1}: \quad \boxed{\tau \ \approx \ \tau(a_r) \ ,} \tag{3.24}$$

$$\textbf{Closure Problem 2}: \quad \boxed{\tau + \mathcal{E} \ \approx \ (\tau + \mathcal{E})(a_r).} \tag{3.25}$$

We use two different closure problems (3.24) and (3.25) to understand the effects of CE on ROM. We use closure problem 1 to obtain the data-driven correction ROM (DDC-ROM) and ideal data driven correction ROM (IDDC-ROM). By using closure problem 2, we derive the commutation error data driven ROM (CE-DDC-ROM). In chapters 4 and 5, we explain in detail the DDC-ROM, IDDC-ROM, and CE-DDC-ROM.

Chapter 4

Commutation Error (CE)

In this thesis, we address the following two questions: (1) "Do we have Commutation Error (CE) in the diffusion term when we apply the projection or differential filter?" and (2) "If so, does this CE affect ROM significantly or not?"
We have already proved that filtering and differentiation with respect to time commute in (3.14) and (3.15): $\overline{(u_d)_t} = (\overline{u_d})_t$. We have also controlled the nonlinear term with the correction term τ. Thus, we only have to investigate the CE for the diffusion term. To this end, we first define the CE in the diffusion term as [12]

$$\mathcal{E}_\Delta[u_d] := \nu\Big(\overline{(u_d)_{xx}} - (\overline{u_d})_{xx}\Big), \tag{4.1}$$

where ν is a viscous parameter.

4.1 CE with the projection Filter

By using the projection filter, the CE can be written as

$$\mathcal{E}_\Delta[u_d] := \nu\Big(\overline{(u_d)_{xx}}^r - (\overline{u_d}^r)_{xx}\Big), \tag{4.2}$$

where "$\overline{}^r$" represents the projection filter. In this section, we derive the equations for $\overline{(u_d)_{xx}}^r$ and $(\overline{u_d}^r)_{xx}$ by using (3.4) to see whether the CE for the projection filter (4.2) is zero or not.

Proposition 1. *The CE satisfies the following relations:*

$$\mathcal{E}_\Delta[u_d] = \nu\Big(\overline{(u_d)_{xx}}^r - (\overline{u_d}^r)_{xx}\Big) = \nu\Bigg(\sum_{j=1}^{r}\big(c_r - b_r\big)_j\ \phi_j\Bigg) \neq 0, \tag{4.3}$$

where $(\overline{u_d}^r)_{xx} = \sum_{j=1}^{r}(b_r)_j\phi_j$ *and* $\overline{(u_d)_{xx}}^r = \sum_{j=1}^{r}(c_r)_j\phi_j$.

Proof. First, we introduce some notation for Proposition 1: a_r and a_d represent the coefficients of $\overline{u_d}^r$ and u_d, respectively, such that $\overline{u_d}^r = \sum_{j=1}^{r}(a_r)_j\phi_j$ and $u_d = \sum_{j=1}^{d}(a_d)_j\phi_j$. Before starting the derivations, we prove that a_r contains the first r entries of a_d when using the projection filter:

$$\begin{aligned}(\overline{u_d}^r, \phi_i)_{\mathcal{H}} &= (u_d, \phi_i)_{\mathcal{H}} \\ M_r\ a_r &= M_{r\times d}\ a_d.\end{aligned} \tag{4.4}$$

Since the L^2-POD basis is used, the ROM mass matrix is the identity matrix and the second equation in (4.4) holds.

For $\overline{(u_d)_{xx}}^r$: By applying the definition (3.4) for $\overline{(u_d)_{xx}}^r$ and integration by parts, the following equalities for $\overline{(u_d)_{xx}}^r$ are obtained: $\forall i = 1, ..., r$,

$$\left(\overline{(u_d)_{xx}}^r, \phi_i\right)_{\mathcal{H}} = \left((u_d)_{xx}, \phi_i\right)_{\mathcal{H}} = -\left((u_d)_x, (\phi_i)_x\right)_{\mathcal{H}}. \tag{4.5}$$

By expanding $\overline{(u_d)_{xx}}^r$ and u_d in terms of suitable time-dependent coefficients and ROM basis functions, the following relation emerges:

$$M_r\ c_r = -S_{r\times d}\ a_d, \tag{4.6}$$

where M_r and $S_{r\times d}$ represent the square ROM mass matrix and the rectangular ROM stiffness matrix, respectively.

For $(\overline{u_d}^r)_{xx}$: By applying integration by parts to $(\overline{u_d}^r)_{xx}$, the following equality is obtained: $\forall i = 1, ..., r$,

$$\left((\overline{u_d}^r)_{xx}, \phi_i\right)_{\mathcal{H}} = -\left((\overline{u_d}^r)_x, (\phi_i)_x\right)_{\mathcal{H}}. \tag{4.7}$$

By expanding $(\overline{u_d}^r)_{xx}$ and $\overline{u_d}^r$ in terms of suitable time-dependent coefficients and ROM basis functions and using the second equality in (4.4), the following relations emerge:

$$M_r\ b_r = -S_r\ a_r = -S_r M_r^{-1} M_{r\times d}\ a_d. \tag{4.8}$$

Since the POD basis functions are orthonormal, the ROM mass matrix is the identity matrix and b_r is affected only by square stiffness ROM matrix and a_r, but c_r is affected by the rectangular ROM stiffness matrix and a_d. Since the stiffness matrix is not the identity matrix, b_r and c_r are different. Thus, $\mathcal{E}_\Delta[u_d] = \nu\left(\overline{(u_d)_{xx}}^r - (\overline{u_d}^r)_{xx}\right) = \nu\left(\sum_{j=1}^{r}\left(c_r - b_r\right)_j\ \phi_j\right) \neq 0$. On the other hand, if r approaches d, $\mathcal{E}_\Delta[u_d]$ approaches 0. □

4.2 CE with Differential Filter

By using the differential filter, the CE in the diffusion term can be written as:

$$\mathcal{E}_\Delta[u_d] := \nu\Big(\overline{(u_d)_{xx}}^{DF} - (\overline{u_d}^{DF})_{xx}\Big), \tag{4.9}$$

where "$\overline{\quad}^{DF}$" represents the differential filter. We derive equations for $\overline{(u_d)_{xx}}^{DF}$ and $(\overline{u_d}^{DF})_{xx}$ by using definition (3.5), and then we investigate whether there exists CE in the diffusion term when the differential filter is applied.

Proposition 2. *The CE satisfies the following relations:*

$$\mathcal{E}_\Delta[u_d] = \nu\Big(\overline{(u_d)_{xx}}^{DF} - (\overline{u_d}^{DF})_{xx}\Big) = \nu\Bigg(\sum_{j=1}^{r}\big(c_r - b_r\big)_j\ \phi_j\Bigg) \neq 0, \tag{4.10}$$

where $(\overline{u_d}^{DF})_{xx} = \sum_{j=1}^{r}(b_r)_j\phi_j$ *and* $\overline{(u_d)_{xx}}^{DF} = \sum_{j=1}^{r}(c_r)_j\phi_j$.

Proof. First, we introduce some notation for Proposition 2: a_r and a_d represent the coefficients of $\overline{u_d}^{DF}$ and u_d, respectively, such that $\overline{u_d}^{DF} = \sum_{j=1}^{r}(a_r)_j\phi_j$, $u_d = \sum_{j=1}^{d}(a_d)_j\phi_j$. Before starting the derivations, we show the relation between a_d and a_r by using the definition (3.5) as follows: $\forall i = 1, ..., r$,

$$\begin{aligned}\Big((\mathbb{I} - \delta^2\Delta)\overline{u_d}^{DF}, \phi_i\Big)_{\mathcal{H}} &= (u_d, \phi_i)_{\mathcal{H}}\\ (M_r + \delta^2 S_r)\ a_r &= M_{r\times d}\ a_d.\end{aligned} \tag{4.11}$$

Thus, we conclude that a_r does not contain the first r entries of a_d anymore; this is the difference between the projection and the differential filter.

For $\overline{(u_d)_{xx}}^{DF}$: By applying the definition (3.5) and integration by parts: $\forall i = 1, ..., r$,

$$\begin{aligned}\Big((\mathbb{I} - \delta^2\Delta)\overline{(u_d)_{xx}}^{DF}, \phi_i\Big)_{\mathcal{H}} = \Big((u_d)_{xx}, \phi_i\Big)_{\mathcal{H}} &= -\Big((u_d)_x, (\phi_i)_x\Big)_{\mathcal{H}}\\ (M_r + \delta^2 S_r)\ c_r &= -S_{r\times d}\ a_d.\end{aligned} \tag{4.12}$$

We apply integration by parts to $(\overline{u_d}^{DF})_{xx}$ as follows: $\forall i = 1, ..., r$,

$$\begin{aligned}\Big((\overline{u_d}^{DF})_{xx}, \phi_i\Big)_{\mathcal{H}} &= -\Big((\overline{u_d}^{DF})_x, (\phi_i)_x\Big)_{\mathcal{H}}\\ M_r\ b_r &= -S_r\ a_r\\ M_r\ b_r &= -S_r(M_r + \delta^2 S_r)^{-1}M_{r\times d}\ a_d.\end{aligned} \tag{4.13}$$

By combining (4.12) and (4.13), we can express the CE in the diffusion term for the differential filter as $\mathcal{E}_\Delta[u_d] = \nu\left(\overline{(u_d)_{xx}}^{DF} - (\overline{u_d}^{DF})_{xx}\right) = \nu\left(\sum_{j=1}^{r}\left(b_r - c_r\right)_j \phi_j\right) \neq 0$. However, when r approaches d, CE approaches zero. □

4.3 Effect of CE on LES-ROMs

After obtaining the results in sections 4.1 and 4.2, we are ready to answer our first question: "Do we have CE in diffusion term when the projection or differential filter is applied?" Since c_r in (4.6) and b_r in (4.8) are not equal, we conclude that there exists CE in the diffusion term for the projection filter. Likewise, we can conclude the same for the differential filter by looking at c_r in (4.12) and b_r in (4.13). We can easily observe that they are not identically equal unless $r = d$. After ensuring that we have CE, we need to address the second question: "Does the CE in the diffusion term affect ROM significantly or not if we use it as a correction term in ROM?" We investigate the effect of CE on ROM in chapter 6 by comparing the numerical results that are obtained by using CE or not.
As we have seen up to this point, we have exactly two correction terms: τ, which is obtained from the nonlinear convection term after applying filtering, and $\mathcal{E}$, which is obtained by applying filtering to the diffusion term. Since we cannot write these two terms in terms of unknown a_r in G-ROM (2.16), we need to solve closure problems for τ and $\mathcal{E}$. Three questions arise:

- "Do we need to solve closure problem only for τ?"
- "Do we need to solve closure problem only for $\mathcal{E}$?"
- "Do we need to solve closure problem for $\tau + \mathcal{E}$?"

Since our aim is to see how the CE affects ROM, we examine the following ROMs: data-driven correction ROM (DDC-ROM), where closure problem is solved for τ term and CE is not be included; ideal data-driven correction ROM (IDDC-ROM), where closure problem is solved again only for τ and the exact CE is inserted directly in ROM; and the CE data-driven correction ROM (CE-DDC-ROM), where closure problem is solved for both τ and $\mathcal{E}$. We discuss these three approaches in the next sections; their numerical results are presented in chapter 6.

Data Driven Correction ROM (DDC – ROM)

In this approach, we apply data driven modeling only for τ term and we do not include $\mathcal{E}$ into ROM to obtain numerical results without CE correction term. Then, we compare the results

from DDC-ROM with ROMs that includes a CE correction term to investigate whether the CE correction term has a significant impact on ROM. Moreover, we discuss under which circumstances ROM with CE correction term gives more accurate results than ROM model without CE correction term. Since we solve closure problem only for τ, we need to create ansatz only for τ term. The corresponding dynamical system is:

$$\textbf{DDC} - \textbf{ROM}: \quad \boxed{\dot{a}_r = (A + \tilde{A})\ a_r + a_r^T\ (B + \tilde{B})\ a_r,} \tag{4.14}$$

where a combination of linear and quadratic ansatz is applied to obtain a data-driven model for τ term. In other words, we use $\tau \approx \tau(a_r) = \tilde{A}\ a_r + a_r^T\ \tilde{B}\ a_r$ for $\tilde{A}$ and $\tilde{B}$.

Ideal Data Driven ROM (IDDC – ROM)

The difference from the DDC-ROM from [71] is that now we use a CE correction term; However, we do not apply data-driven modeling to this correction term; instead we directly insert the CE correction term into ROM. The corresponding dynamical system is

$$\textbf{IDDC} - \textbf{ROM}: \quad \boxed{\dot{a}_r = (A + \tilde{A})\ a_r + a_r^T\ (B + \tilde{B})\ a_r + \mathcal{E},} \tag{4.15}$$

where $\tilde{A}$ and $\tilde{B}$ are obtained by solving $\tau \approx \tau(a_r) = \tilde{A}\ a_r + a_r^T\ \tilde{B}\ a_r$ for $\tilde{A}$ and $\tilde{B}$ and $\mathcal{E}$ is computed exactly without using data-driven modeling.

CE Data Driven Correction ROM (CE – DDC – ROM)

In this approach, we apply data-driven modeling for both τ and $\mathcal{E}$, [35]. In other words, we solve $(\tau + \mathcal{E}) \approx (\tau + \mathcal{E})(a_r) = \tilde{A}\ a_r + a_r^T\ \tilde{B}\ a_r$ for $\tilde{A}$ and $\tilde{B}$. The corresponding dynamical system is

$$\textbf{CE} - \textbf{DDC} - \textbf{ROM}: \quad \boxed{\dot{a}_r = (A + \tilde{A})\ a_r + a_r^T\ (B + \tilde{B})\ a_r.} \tag{4.16}$$

Chapter 5

Data-Driven Correction ROM (DDC-ROM)

5.1 Introduction

In this section, we construct the data-driven correction ROM (DDC-ROM). Specifically, we use data-driven modeling to solve the ROM closure problem. Since F-ROM (3.23) contains two correction terms τ and $\mathcal{E}$, we can apply data driven modeling to (3.24) or (3.25). To show how the DDC-ROM is constructed, we derive the system only for the τ term. One can easily construct a data driven model for $\mathcal{E}$ based on what we do for the τ term. From now on, we only focus on closure problem (3.24). To make F-ROM (3.23) resemble the G-ROM (2.16) for τ term, we define the following ansatz to solve the closure problem:

$$\tau(a_r) = \tilde{A}\ a_r\ +\ a_r^T\ \tilde{B}\ a_r, \tag{5.1}$$

where the unknowns are $\tilde{A}$ and $\tilde{B}$. Finding $\tilde{A}$ and $\tilde{B}$ solves the ROM closure problem.

5.2 Mathematical Formulation

To find the unknowns $\tilde{A}$ and $\tilde{B}$ in (5.1), we use data-driven modeling. That is, we find $\tilde{A}$ and $\tilde{B}$ that guarantee the highest accuracy between $\tau(a_r)$ in (5.1)and τ in the F-ROM (3.23). Numerically, we minimize the difference between τ and $\tau(a_r)$, which are computed from (3.21) and (5.1), respectively. We compute the values $a_d^{snap}(t_j)$ at different time instances $t_j,\ \forall j = 1, ..., M$, by projecting the FE solutions $u_h(t_j) = \sum_{k=1}^{d} (a_d)_k^{snap}(t_j)\phi_k$ onto the POD basis functions $\phi_i,\ \forall i = 1, ..., d$, and by using the orthogonality of POD basis functions: $\forall i = 1, ..., d$ and $\forall j = 1, ..., M$,

$$
\begin{aligned}
(a_d)_i^{snap}(t_j) &= \Big(u_h(t_j), \phi_i\Big)_{\mathcal{H}} \\
\Big(\sum_{m=1}^{N}(u_h^j)_m\phi_m^h, \phi_i\Big)_{\mathcal{H}} &= \Big(\sum_{m=1}^{N}(u_h^j)_m\phi_m^h, \sum_{n=1}^{N}C_d(n,i)\phi_n^h\Big)_{\mathcal{H}} = C_d^T M_h Y(:,j) \\
\Big(u_h(t_j), \phi_i\Big)_{\mathcal{H}} &= \Big(\sum_{k=1}^{d}(a_d)_k^{snap}(t_j)\phi_k, \phi_i\Big)_{\mathcal{H}} = M_d\ a_d^{snap}(t_j) \\
M_d\ a_d^{snap} &= C_d^T M_h Y.
\end{aligned}
\tag{5.2}
$$

To find a_r^{snap}, we solve (5.2) by using $u_h(t_j) = \sum_{k=1}^{r}(a_r)^{snap}(t_j)\phi_k$, $\forall i = 1, ..., r$, which yields the relation $M_r\ a_r^{snap} = C_r^T M_h Y$. To get $\tilde{A}$ and $\tilde{B}$, we need to solve the following minimization problem:

$$
\min_{\substack{\tilde{A}\in\mathbb{R}^{r\times r}\\ \tilde{B}\in\mathbb{R}^{r\times r\times r}}} \quad \sum_{j=1}^{M}\|\tau^{true}(t_j) - \tau^{ansatz}(t_j)\|_{\mathcal{H}}^2, \tag{5.3}
$$

where $\|\ .\ \|_{\mathcal{H}}$ is the Euclidean norm in $\mathbb{R}^r$. $\tau^{true}(t_j)$ in (5.3) is the exact value of $\tau(t_j)$, which can be computed as follows: $\forall i = 1, ..., r$ and $\forall j = 1, .., M$,

$$
\begin{aligned}
\tau_i^{true}(t_j) &= -\Bigg(\overline{\Big(u_d(t_j)\big(u_d(t_j)\big)_x\Big)} - \Big(u_r(t_j)(u_r(t_j))_x\Big), \phi_i\ \Bigg) \\
&= -\Bigg(\overline{\sum_{k_1=1}^{d}\sum_{k_2=1}^{d}(a_d)_{k_1}^{snap}(t_j)(a_d)_{k_2}^{snap}(t_j)\Big(\phi_{k_1}(\phi_{k_2})_x\Big),\ \phi_i}\Bigg) + \\
&\quad + \Bigg(\sum_{k_3=1}^{r}\sum_{k_4=1}^{r}(a_r)_{k_3}^{snap}(t_j)(a_r)_{k_4}^{snap}(t_j)\Big(\phi_{k_3}(\phi_{k_4})_x\Big),\ \phi_i\Bigg)_{\mathcal{H}},
\end{aligned}
\tag{5.4}
$$

where

$$
u_d^{snap}(t_j) = \sum_{k=1}^{d}(a_d)_k^{snap}(t_j)\phi_k \qquad , \qquad u_r^{snap}(t_j) = \sum_{k=1}^{r}\ (a_r)_k^{snap}(t_j)\phi_k. \tag{5.5}
$$

By plugging a_d^{snap} and a_r^{snap}, which are found by using (5.2), into (5.4) and (5.1), we obtain $\tau^{true}(t_j)$ and derive a formula for $\tau^{ansatz}(t_j)$ as follows:

$$
\tau^{ansatz}(t_j) = \tilde{A}\ (a_r)^{snap}(t_j) + (a_r)^{snap}(t_j)^T\ \tilde{B}\ (a_r)^{snap}(t_j). \tag{5.6}
$$

By inserting $\tau^{ansatz}(t_j)$ into the minimization problem (5.3), we obtain the following minimization problem:

$$
\min_{\substack{\tilde{A}\in\mathbb{R}^{r\times r}\\ \tilde{B}\in\mathbb{R}^{r\times r\times r}}} \sum_{j=1}^{M}\Big\|\tau^{true}(t_j) - \tilde{A}\ a^{snap}(t_j) - a^{snap}(t_j)^T\ \tilde{B}\ a^{snap}(t_j)\Big\|_{\mathcal{H}}^2. \tag{5.7}
$$

The aim is to minimize the Euclidean norm ($\mathcal{H} = L^2$) in (5.7). We can write the minimization problem (5.7) as a least squares problem as follows:

$$\min_{\substack{\tilde{A}\in\mathbb{R}^{r\times r}\\ \tilde{B}\in\mathbb{R}^{r\times r\times r}}} \sum_{j=1}^{M} \left\| \tau^{true}(t_j) - \tilde{A}\, a^{snap}(t_j) - a^{snap}(t_j)^T\, \tilde{B}\, a^{snap}(t_j) \right\|_{\mathcal{H}}^2 = \min_{x\in\mathbb{R}^{(r^3+r^2)}} \|f - Ex\|_{\mathcal{H}}^2, \quad (5.8)$$

where the vector $x \in \mathbb{R}^{(r^2+r^3)\times 1}$ contains all entries of the unknown matrix $\tilde{A}$ and tensor $\tilde{B}$. The vector $f \in \mathbb{R}^{(Mr)\times 1}$ and the matrix $E \in \mathbb{R}^{(Mr)\times(r^2+r^3)}$ are computed by using $\tau^{true}(t_j)$ and $a_r^{snap}(t_j)$, respectively. We use the following notation for the matrix E and vectors f and x: $\forall i = 1, ..., r \quad , \quad j = 1, ..., M,$

$$\begin{aligned} b_j &:= \Big((a_r)_1^{snap}(t_j), (a_r)_2^{snap}(t_j), ..., (a_r)_r^{snap}(t_j) \Big) \in \mathbb{R}^{1\times r}, \\ c_j &:= \Big((a_r)_1^{snap}(t_j)(a_r)_1^{snap}(t_j), a_1^{snap}(t_j)a_2^{snap}(t_j), ..., (a_r)_1^{snap}(t_j)(a_r)_r^{snap}(t_j), \\ &\qquad (a_r)_2^{snap}(t_j)(a_r)_1^{snap}(t_j), ..., (a_r)_r^{snap}(t_j)(a_r)_r^{snap}(t_j) \Big) \in \mathbb{R}^{1\times r^2}, \\ \tau_{ij} &:= \tau_i^{true}(t_j). \end{aligned} \quad (5.9)$$

5.2.1 Case 1:

In the linear least squares problem in (5.8), x represents the unknowns, which are the entries of $\tilde{A}$ and $\tilde{B}$. Since the dimensions of $\tilde{A}$ and $\tilde{B}$ are $r\times r$ and $r\times r\times r$, respectively, the dimension of x is $(r^2+r^3)\times 1$. Furthermore, since the entries of vector f are $\tau_i^{true}(t_j)$ for $i=1,...,r$ and $j=1,...,M$, the dimension of vector f is $Mr\times 1$. Then, to guarantee consistency, the dimension of matrix E should be $Mr\times(r^2+r^3)$. By using the notations in (5.9), the matrix E and vectors f and x are constructed as follows:

$$f = \begin{bmatrix} \tau_{11} \\ \tau_{21} \\ \vdots \\ \tau_{r1} \\ \tau_{12} \\ \tau_{22} \\ \vdots \\ \tau_{r2} \\ \vdots \\ \vdots \\ \tau_{1M} \\ \tau_{2M} \\ \vdots \\ \tau_{rM} \end{bmatrix} \in \mathbb{R}^{Mr\times 1} \quad , \qquad x = \begin{bmatrix} \tilde{A}_{11} \\ \vdots \\ \tilde{A}_{r1} \\ \tilde{A}_{12} \\ \vdots \\ \tilde{A}_{rr} \\ \tilde{B}_{111} \\ \vdots \\ \tilde{B}_{1r1} \\ \vdots \\ \tilde{B}_{rr1} \\ \tilde{B}_{112} \\ \vdots \\ \tilde{B}_{1r2} \\ \vdots \\ \tilde{B}_{rr2} \\ \vdots \\ \tilde{B}_{11r} \\ \vdots \\ \tilde{B}_{1rr} \\ \vdots \\ \tilde{B}_{rrr} \end{bmatrix} \in \mathbb{R}^{(r^2+r^3)\times 1},$$

$$E = \begin{bmatrix}
b_1 & 0 & 0 & ... & 0 & 0 & 0 & c_1 & 0 & 0 & ... & 0 & 0 & 0 \\
0 & b_1 & 0 & ... & 0 & 0 & 0 & 0 & c_1 & 0 & ... & 0 & 0 & 0 \\
0 & 0 & b_1 & ... & 0 & 0 & 0 & 0 & 0 & c_1 & ... & 0 & 0 & 0 \\
: & : & : & ... & : & : & : & : & : & : & ... & : & : & : \\
: & : & : & ... & : & : & : & : & : & : & ... & : & : & : \\
0 & 0 & 0 & ... & 0 & b_1 & 0 & 0 & 0 & 0 & ... & 0 & c_1 & 0 \\
0 & 0 & 0 & ... & 0 & 0 & b_1 & 0 & 0 & 0 & ... & 0 & 0 & c_1 \\
b_2 & 0 & 0 & ... & 0 & 0 & 0 & c_2 & 0 & 0 & ... & 0 & 0 & 0 \\
0 & b_2 & 0 & ... & 0 & 0 & 0 & 0 & c_2 & 0 & ... & 0 & 0 & 0 \\
0 & 0 & b_2 & ... & 0 & 0 & 0 & 0 & 0 & c_2 & ... & 0 & 0 & 0 \\
: & : & : & ... & : & : & : & : & : & : & ... & : & : & : \\
: & : & : & ... & : & : & : & : & : & : & ... & : & : & : \\
0 & 0 & 0 & ... & 0 & b_2 & 0 & 0 & 0 & 0 & ... & 0 & c_2 & 0 \\
0 & 0 & 0 & ... & 0 & 0 & b_2 & 0 & 0 & 0 & ... & 0 & 0 & c_2 \\
: & : & : & ... & : & : & : & : & : & : & ... & : & : & : \\
: & : & : & ... & : & : & : & : & : & : & ... & : & : & : \\
: & : & : & ... & : & : & : & : & : & : & ... & : & : & : \\
: & : & : & ... & : & : & : & : & : & : & ... & : & : & : \\
b_M & 0 & 0 & ... & 0 & 0 & 0 & c_M & 0 & 0 & ... & 0 & 0 & 0 \\
0 & b_M & 0 & ... & 0 & 0 & 0 & 0 & c_M & 0 & ... & 0 & 0 & 0 \\
0 & 0 & b_M & ... & 0 & 0 & 0 & 0 & 0 & c_M & ... & 0 & 0 & 0 \\
: & : & : & ... & : & : & : & : & : & : & ... & : & : & : \\
: & : & : & ... & : & : & : & : & : & : & ... & : & : & : \\
0 & 0 & 0 & ... & 0 & b_M & 0 & 0 & 0 & 0 & ... & 0 & c_M & 0 \\
0 & 0 & 0 & ... & 0 & 0 & b_M & 0 & 0 & 0 & ... & 0 & 0 & c_M
\end{bmatrix} \in \mathbb{R}^{Mr\times(r^2+r^3)}.$$

Solving the linear least squares problem in (5.8) based on vectors x and f, and matrix E is not efficient for large M values even if r is small. In the next section, we show that there exist efficient ways to obtain an equivalent linear least squares problem that is more efficient than (5.8).

5.2.2 Case 2:

We write the linear least squares problem in (5.8) as a a sum of linear least squares problems to guarantee computational efficiency. The updated vectors $\hat{f}$ and $\hat{x}$ and matrix $\hat{E}$ are

permuted forms of vectors f and x and matrix E from the previous section:

$$\hat{f} = \begin{bmatrix} \tau_{11} \\ \tau_{12} \\ \vdots \\ \tau_{1M} \\ \tau_{21} \\ \tau_{22} \\ \vdots \\ \tau_{2M} \\ \vdots \\ \vdots \\ \tau_{r1} \\ \tau_{r2} \\ \vdots \\ \tau_{rM} \end{bmatrix} \in \mathbb{R}^{Mr\times 1} \quad , \quad \hat{x} = \begin{bmatrix} \tilde{A}_{11} \\ \tilde{A}_{12} \\ \vdots \\ \tilde{A}_{1r} \\ \tilde{B}_{111} \\ \tilde{B}_{121} \\ \vdots \\ \tilde{B}_{rr1} \\ \tilde{A}_{21} \\ \tilde{A}_{22} \\ \vdots \\ \tilde{A}_{2r} \\ \tilde{B}_{112} \\ \vdots \\ \tilde{B}_{1r2} \\ \vdots \\ \tilde{B}_{rr2} \\ \vdots \\ \tilde{A}_{r1} \\ \tilde{A}_{r2} \\ \vdots \\ \tilde{A}_{rr} \\ \tilde{B}_{11r} \\ \vdots \\ \tilde{B}_{1rr} \\ \vdots \\ \tilde{B}_{rrr} \end{bmatrix} \in \mathbb{R}^{(r^2+r^3)\times 1},$$

$$\hat{E} = \begin{bmatrix}
b_1 & c_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & ... & 0 & 0 & 0 & 0 \\
b_2 & c_2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & ... & 0 & 0 & 0 & 0 \\
b_3 & c_3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & ... & 0 & 0 & 0 & 0 \\
: & : & : & : & : & : & : & : & : & : & ... & : & : & : & : \\
b_M & c_M & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & ... & 0 & 0 & 0 & 0 \\
0 & 0 & b_1 & c_1 & 0 & 0 & 0 & 0 & 0 & 0 & ... & 0 & 0 & 0 & 0 \\
0 & 0 & b_2 & c_2 & 0 & 0 & 0 & 0 & 0 & 0 & ... & 0 & 0 & 0 & 0 \\
0 & 0 & b_3 & c_3 & 0 & 0 & 0 & 0 & 0 & 0 & ... & 0 & 0 & 0 & 0 \\
: & : & : & : & : & : & : & : & : & : & ... & : & : & : & : \\
0 & 0 & b_M & c_M & 0 & 0 & 0 & 0 & 0 & 0 & ... & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & b_1 & c_1 & 0 & 0 & 0 & 0 & ... & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & b_2 & c_2 & 0 & 0 & 0 & 0 & ... & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & b_3 & c_3 & 0 & 0 & 0 & 0 & ... & 0 & 0 & 0 & 0 \\
: & : & : & : & : & : & : & : & : & : & ... & : & : & : & : \\
0 & 0 & 0 & 0 & b_M & c_M & 0 & 0 & 0 & 0 & ... & 0 & 0 & 0 & 0 \\
: & : & : & : & : & : & : & : & : & : & ... & : & : & : & : \\
: & : & : & : & : & : & : & : & : & : & ... & : & : & : & : \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & ... & b_1 & c_1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & ... & b_2 & c_2 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & ... & b_3 & c_3 & 0 & 0 \\
: & : & : & : & : & : & : & : & : & : & ... & : & : & : & : \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & ... & b_M & c_M & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & ... & 0 & 0 & b_1 & c_1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & ... & 0 & 0 & b_2 & c_2 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & ... & 0 & 0 & b_3 & c_3 \\
: & : & : & : & : & : & : & : & : & : & ... & : & : & : & : \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & ... & 0 & 0 & b_M & c_M
\end{bmatrix} \in \mathbb{R}^{Mr\times(r^2+r^3)}.$$

Up to this point, we have only rearranged the entries of vectors f and x and matrix E to obtain $\hat{f}$, $\hat{x}$, and $\hat{E}$, respectively, while keeping the same dimensions. Although matrix $\hat{E}$ is still high-dimensional, it has a recurrent pattern. This property allows us to convert one linear least problem with huge dimension to a sum of linear least squares problems with smaller dimensions. We label the recurrence pattern in matrix $\hat{E}$ as $\hat{E}_s$. The following explains the relation between $\hat{E}$ and $\hat{E}_s$:

$$\hat{E} = \begin{bmatrix}
\hat{E}_s & \mathcal{O} & \mathcal{O} & ... & \mathcal{O} & \mathcal{O} \\
\mathcal{O} & \hat{E}_s & \mathcal{O} & ... & \mathcal{O} & \mathcal{O} \\
\mathcal{O} & \mathcal{O} & \hat{E}_s & ... & \mathcal{O} & \mathcal{O} \\
: & : & : & ... & : & : \\
: & : & : & ... & : & : \\
\mathcal{O} & \mathcal{O} & \mathcal{O} & ... & \hat{E}_s & \mathcal{O} \\
\mathcal{O} & \mathcal{O} & \mathcal{O} & ... & \mathcal{O} & \hat{E}_s
\end{bmatrix} \in \mathbb{R}^{Mr\times(r^2+r^3)},$$

where the recurrent pattern $\hat{E}_s$ in matrix $\hat{E}$ is:

$$\hat{E}_s := \begin{bmatrix} b_1 & c_1 \\ b_2 & c_2 \\ b_3 & c_3 \\ : & : \\ : & : \\ b_M & c_M \end{bmatrix} \in \mathbb{R}^{M\times(r+r^2)}.$$

We subdivide the vectors $\hat{f}$ and $\hat{x}$ to obtain vectors $\hat{f}_k$ and $\hat{x}_k$. The following vectors f_k and x_k are kth subvectors of $\hat{f}$ and $\hat{x}$, respectively: $\forall k = 1, ..., r$,

$$\hat{f}_k = \begin{bmatrix} \tau_{k1} \\ \tau_{k2} \\ : \\ \tau_{kM} \end{bmatrix} \in \mathbb{R}^{M\times 1} \quad , \quad \hat{x}_k = \begin{bmatrix} \tilde{A}_{k1} \\ \tilde{A}_{k2} \\ : \\ \tilde{A}_{kr} \\ \tilde{B}_{11k} \\ \tilde{B}_{12k} \\ : \\ \tilde{B}_{rrk} \end{bmatrix} \in \mathbb{R}^{(r+r^2)\times 1}.$$

We rewrite the large dimensional linear least squares problem as a sum of smaller dimensional linear least squares problem as follows:

$$\left\| \hat{f} - \hat{E}\hat{x} \right\|_{\mathcal{H}}^2 = \left\| \begin{bmatrix} \hat{f}_1 \\ \hat{f}_2 \\ : \\ \hat{f}_r \end{bmatrix} - \begin{bmatrix} \hat{E}_s & \mathcal{O} & ... & \mathcal{O} \\ \mathcal{O} & \hat{E}_s & ... & \mathcal{O} \\ : & : & ... & : \\ : & : & ... & : \\ \mathcal{O} & \mathcal{O} & & \hat{E}_s \end{bmatrix} \begin{bmatrix} \hat{x}_1 \\ \hat{x}_2 \\ : \\ \hat{x}_r \end{bmatrix} \right\|_{\mathcal{H}}^2 = \sum_{k=1}^{r} \left\| \hat{f}_k - \hat{E}_s\hat{x}_k \right\|_{\mathcal{H}}^2.$$

Thus, the minimization problem in (5.3) has first been converted to a least squares problem and then as an efficient sum of least squares problems as follows:

$$\begin{aligned} \min_{\substack{\tilde{A}\in\mathbb{R}^{r\times r} \\ \tilde{B}\in\mathbb{R}^{r\times r\times r}}} \sum_{j=1}^{M} \left\| \tau^{true}(t_j) - \tilde{A}\ a^{snap}(t_j) - a^{snap}(t_j)^T\ \tilde{B}\ a^{snap}(t_j) \right\|_{\mathcal{H}}^2 & \\ = \min_{x\in\mathbb{R}^{(r^2+r^3)\times 1}} \left\| f - Ex \right\|_{\mathcal{H}}^2 & \\ = \sum_{k=1}^{r} \min_{\hat{x}_k\in\mathbb{R}^{(r+r^2)\times 1}} \left\| \hat{f}_k - \hat{E}_s\hat{x}_k \right\|_{\mathcal{H}}^2. & \end{aligned} \tag{5.10}$$

To find the optimal $\tilde{A}$ and $\tilde{B}$ that satisfy (5.10), we need to solve the least squares problem in the last equality in (5.10) for $\hat{x}_k$, $\forall k = 1, ..., r$, which contains $\tilde{A}$ and $\tilde{B}$. The equalities

in (5.10) have been proven and illustrated numerically in [46]. We emphasize that in (5.10) τ^{true} has been computed by using different ROM spatial filters. In (5.4), we have derived the general equation for τ^{true}. Now, we define τ^{true} specifically for both the differential and the projection filter.
We start with computing $\tau^{true} = \tau$ for the projection filter by using (3.4) as follows: $\forall i = 1, ..., r$,

$$\begin{aligned}
\tau_i = -\left(\tau^{SFS}, \phi_i\right)_{\mathcal{H}} &= -\left(\overline{u_d(u_d)_x}^r - u_r(u_r)_x, \phi_i\right)_{\mathcal{H}} \\
\left(\overline{u_d(u_d)_x}^r, \phi_i\right)_{\mathcal{H}} &= \left(u_d(u_d)_x, \phi_i\right)_{\mathcal{H}} \\
\tau_i = -\Bigg(\sum_{k_1=1}^{d} \sum_{k_2=1}^{d} (a_d)_{k_1} (a_d)_{k_2} &\left(\phi_{k_1}(\phi_{k_2})_x\right), \phi_i \Bigg)_{\mathcal{H}} + \\
\Bigg(\sum_{k_3=1}^{r} \sum_{k_4=1}^{r} (a_r)_{k_3} (a_r)_{k_4} &\left(\phi_{k_3}(\phi_{k_4})_x\right), \phi_i \Bigg)_{\mathcal{H}},
\end{aligned} \tag{5.11}$$

where $a_d = a_d^{snap}$ and $a_r = a_r^{snap}$ are obtained by using (5.2). We continue with the differential filter. We use the self-adjoint property of the differential filter [72]: $\forall i = 1, ..., r$

$$\begin{aligned}
\tau_i = \left(\tau^{SFS}, \phi_i\right)_{\mathcal{H}} &= -\left(\overline{u_d(u_d)_x}^{DF} - u_r(u_r)_x, \phi_i\right)_{\mathcal{H}} \\
\left(\overline{u_d(u_d)_x}^{DF}, \phi_i\right)_{\mathcal{H}} &= \left(u_d(u_d)_x, \overline{\phi_i}^{DF}\right)_{\mathcal{H}} \\
\tau_i = -\Bigg(\sum_{k_1=1}^{d} \sum_{k_2=1}^{d} (a_d)_{k_1} (a_d)_{k_2} &\left(\phi_{k_1}(\phi_{k_2})_x\right), \overline{\phi_i}^{DF} \Bigg)_{\mathcal{H}} + \\
\Bigg(\sum_{k_3=1}^{r} \sum_{k_4=1}^{r} (a_r)_{k_3} (a_r)_{k_4} &\left(\phi_{k_3}(\phi_{k_4})_x\right), \phi_i \Bigg)_{\mathcal{H}},
\end{aligned} \tag{5.12}$$

where we use (5.2) to obtain a_d and a_r. The main difference between the projection and differential filter for the τ terms (5.11) and (5.12) is that the latter uses the filtered POD basis function $\overline{\phi_i}^{DF}$ in the first term of τ instead of the unfiltered POD basis function ϕ_i. By using (2.3) and (3.9), the filtered POD basis function can be derived as $\overline{\phi_i}^{DF} = \sum_{m=1}^{N} D_r(m, i)\phi_m^h$ and by inserting this equality in (5.12), τ is computed for the differential filter. The following algorithm outlines the main steps of the data driven model.

Data Driven Algorithm: As we have explained in sections 5.2.1 and 5.2.2, there are two approaches to solve the least squares problem in (5.8). The main difference between Case 1 and Case 2 is that the latter is more efficient. We construct the data-driven model only for the correction term τ by using Case 2.

1. Consider the following F-ROM:

$$\dot{a}_r = A\ a_r + a_r^T\ B\ a_r + \tau. \tag{5.13}$$

2. Use snapshot data and (5.2) to find $a_d = a_d^{snap}$ and $a_r = a_r^{snap}$, then insert them into (5.11) and (5.12) to compute $\tau = \tau^{true}$ for the projection and differential filters, respectively.

3. Use τ^{true} that is computed in step 2 to create the vector $\hat{f}$, which is defined in Case 2.

4. Construct

$$\tau^{ansatz} = \tilde{A}\ a_r^{snap}\ +\ a_r^{snap}\ \tilde{B}\ a_r^{snap}. \tag{5.14}$$

5. Construct the matrix $\hat{E}$ (by using a_r^{snap}) and the vector $\hat{x}$ (by using the unknowns $\tilde{A}$ and $\tilde{B}$), as defined in Case 2.

6. Use the truncated SVD algorithm to solve the following linear least squares problems:

$$\min_{\hat{x}\in\mathbb{R}^{(r^2+r^3)\times r}} \left\|\hat{f} - \hat{E}\hat{x}\right\|_{\mathcal{H}}^2 \ = \ \sum_{k=1}^{r} \ \min_{\hat{x}_k\in\mathbb{R}^{(r+r^2)\times 1}} \left\|\hat{f}_k - \hat{E}_s\hat{x}_k\right\|_{\mathcal{H}}^2. \tag{5.15}$$

   (a) Compute the SVD form of the matrix $\hat{E}_s$ as follows:

$$\hat{E}_s = U\Sigma V^T. \tag{5.16}$$

   (b) Specify tolerance to determine the smallest singular value that we consider.

   (c) Create matrix $\hat{\Sigma}$ from Σ to collect the singular values that are greater than the tolerance fixed in step (b): $\forall j = 1, ..., i$,

$$\hat{\sigma}_j = \sigma_j, \quad s.t \quad \sigma_{i+1} \leqslant tolerance. \tag{5.17}$$

   Truncate the matrices U and V as follows: $\hat{U} = U(:, 1:i)$ and $\hat{V} = V(:, 1:i)$.

   (d) The solution of the least squares problems (5.15) can be found as follows: $\forall k = 1, ..., r$,

$$\hat{x}_k = \left(\hat{V}\hat{\Sigma}^{-1}\hat{U}^T\right)\hat{f}_k. \tag{5.18}$$

7. Convert the entries of vectors $\hat{x}_k$, (which are defined in Case 2), to obtain $\tilde{A}$ and $\tilde{B}$.

8. Construct DD-ROM as follows:

$$\boxed{\dot{a}_r = \left(A + \tilde{A}\right)\ a_r + a_r^T\ \left(B + \tilde{B}\right)\ a_r.} \tag{5.19}$$

Chapter 6

Numerical Results

We have already concluded that there exists CE for both the differential and the projection filter. To see the CE's effect on ROM, we need to compare different ROM approaches that are computed with or without CE. We need to investigate whether the error between the FE and ROM solutions is getting significantly smaller when the CE is inserted in the system; in that case we can conclude that CE has an important impact on the given problem. We try to find answers to the following questions in our numerical investigation:

1. Under which circumstances the ROM approach with CE gives more accurate results than the ROM approach without CE?
2. Are there any parameters or conditions that significantly affect the terms τ and $\mathcal{E}$?

To obtain numerical results, we need to clearly define the error. In this thesis, our error measures the difference between the ROM solution and the projection of the FE solution and can be expressed as follows:

$$e := \|proj_r(u_h) - u_r\|_{\mathcal{H}}, \tag{6.1}$$

where $proj_r(u_h)$ represents the function that is obtained by projecting the FE solution onto the ROM space (by using (3.4)) and expanding the ROM solution in terms of the FE basis functions). Furthermore, u_h and u_r represent the FE and ROM solutions, respectively. We also consider the following average error:

$$e_{L^2(L^2)} := \sqrt{\frac{1}{M}\sum_{i=1}^{M} \|proj_r(u_h(.,t_i)) - u_r(.,t_i)\|^2_{\mathcal{H}}}. \tag{6.2}$$

In what follows all the errors are computed by using (6.2). As mathematical model, in this section we use the Burgers equation with homogeneous Dirichlet boundary conditions:

$$\begin{cases} u_t - \nu u_{xx} + uu_x = 0\ , & x \in [0,1],\ t \in [0,1], \\ u(0,t) = u(1,t) = 0\ , & t \in [0,1]. \end{cases} \tag{6.3}$$

6.1 Burgers Equation with Smooth IC

In this section, we consider the Burgers equation (6.3) with a smooth initial condition:

$$u_0(x) = \frac{2\nu\beta\pi sin(\pi x)}{\alpha + \beta cos(\pi x)} \ , \quad x \in [0,1], \tag{6.4}$$

with the diffusion parameter $\nu = 10^{-1}$ or $\nu = 10^{-3}$, $\alpha = 5$, and $\beta = 4$. To obtain the snapshots, we use piecewise linear FEs with $h = 1/2048$ and $\Delta t = 10^{-3}$.

For both the projection filter and the differential filter, we investigate whether the CE is large. Furthermore, we test whether adding the CE to the data-driven model (i.e., the IDDC-ROM and CE-DDC-ROM) yields better results than the standard DDC-ROM, which does not model the CE.

6.1.1 Results with the Projection Filter

Tables 6.1 and 6.2 show that the CE obtained by using the the projection filter gradually decreases as r increases.

r	$\|\mathcal{E}_\Delta[u_d]\|_{L^2(L^2)}$
2	1.5350e-01
3	3.2165e-02
4	5.4671e-03
5	8.2634e-04
6	1.1506e-04
7	0

Table 6.1: CE obtained by using the projection filter with different r values when $d = 7$ and $\nu = 10^{-1}$.

r	$\|\mathcal{E}_\Delta[u_d]\|_{L^2(L^2)}$
1	4.3961e-04
2	4.3246e-06
3	0

Table 6.2: CE obtained by using the projection filter with different r values when $d = 3$ and $\nu = 10^{-3}$.

Tables 6.3 and 6.4 show that the CE-DDC-ROM yields much better results than DDC-ROM, especially for low r values. From those tables, we conclude that the CE affects the DDC-ROM significantly: the IDDC-ROM and CE-DDC-ROM that model the CE give much better results than the DDC-ROM that does not model the CE.

r	G-ROM	DDC-ROM	IDDC-ROM	CE-DDC-ROM
2	1.7759e-03	1.3238e-03	1.4648e-05	1.4807e-05
3	1.9301e-04	1.5514e-04	1.7270e-07	1.5514e-07
4	2.0028e-05	7.3988e-06	8.7196e-08	8.8975e-08
5	2.0344e-06	7.2933e-07	9.2271e-08	9.2497e-08
6	2.2325e-07	1.6043e-07	9.3466e-08	9.3467e-08
7	9.5581e-08	9.5581e-08	9.5581e-08	9.5581e-08

Table 6.3: Errors for G-ROM, DDC-ROM, IDDC-ROM, and CE-DDC-ROM with the projection filter for $\nu = 10^{-1}$, $d = 7$, and different r values.

We note that it takes 0.102611 seconds to compute the matrix A and tensor B in the G-ROM when $r = 6$ and the error is 2.2325e-07. The ellapsed time to compute the matrix $\tilde{A}$ and tensor $\tilde{B}$ in the CE-DDC-ROM for $r = 3$ and $d = 7$ is 0.077441, which is about 70% of the CPU time; for these values, the CE-DDC-ROM error is 1.5514e-07, which is only about half of the G-ROM error. Furthermore, the computationally efficiency of the CE-DDC-ROM can be improved by using the approaches proposed in [45, 71].

r	G-ROM	DDC-ROM	IDDC-ROM	CE-DDC-ROM
1	2.2069e-07	1.3548e-07	3.4903e-10	4.0655e-10
2	9.3295e-10	6.6797e-10	5.3548e-13	5.3665e-13
3	3.4616e-12	3.4616e-12	3.4616e-12	3.4616e-12

Table 6.4: Errors for G-ROM, DDC-ROM, IDDC-ROM, and CE-DDC-ROM with the the projection filter for $\nu = 10^{-3}$, $d = 3$, and different r values.

6.1.2 Results with the Differential Filter

Tables 6.5 and 6.6 list the CE obtained by using the differential filter for different r and δ values. The former uses $\nu = 10^{-1}$ and the latter uses $\nu = 10^{-3}$. We observe that the CE from Tables 6.5 and 6.6 decreases as r approaches $d = 7$ and $d = 3$, respectively. For fixed r values in both tables, if δ decreases, then the CE increases slightly.

r	δ	$\|\mathcal{E}_\Delta[u_d]\|_{L^2(L^2)}$	r	δ	$\|\mathcal{E}_\Delta[u_d]\|_{L^2(L^2)}$
2	10^{-1}	9.2269e-02	5	10^{-1}	1.6983e-04
2	10^{-2}	1.5247e-01	5	10^{-2}	7.8755e-04
2	10^{-3}	1.5349e-01	5	10^{-3}	8.2593e-04
2	10^{-5}	1.5350e-01	5	10^{-4}	8.2634e-04
3	10^{-1}	1.2868e-02	6	10^{-1}	1.7999e-05
3	10^{-2}	3.1644e-02	6	10^{-2}	1.0732e-04
3	10^{-3}	3.2160e-02	6	10^{-3}	1.1498e-04
3	10^{-5}	3.2165e-02	6	10^{-4}	1.1506e-04
4	10^{-1}	1.5313e-03	7	10^{-1}	2.9565e-13
4	10^{-2}	5.3046e-03	7	10^{-2}	3.01679e-14
4	10^{-3}	5.4655e-03	7	10^{-3}	2.2503e-15
4	10^{-5}	5.4671e-03	7	10^{-4}	1.9009e-15

Table 6.5: CE obtained by using the differential filter with $\nu = 10^{-1}$, $d = 7$, and different r and δ values.

r	δ	$\|\mathcal{E}_\Delta[u_d]\|_{L^2(L^2)}$
1	10^{-1}	3.6143e-04
1	10^{-2}	4.3866e-04
1	10^{-4}	4.3961e-04
1	10^{-6}	4.3961e-04
2	10^{-1}	2.0926e-06
2	10^{-2}	4.2763e-06
2	10^{-4}	4.3246e-06
2	10^{-6}	4.3246e-06
3	10^{-1}	1.3594e-15
3	10^{-2}	4.8001e-17
3	10^{-4}	3.7982e-17
3	10^{-6}	3.7372e-17

Table 6.6: CE obtained by using the differential filter with $\nu = 10^{-3}$, $d = 3$, and different r and δ values.

Tables 6.7 and 6.8 show that the CE affects the DDC-ROM significantly: the IDDC-ROM and CE-DDC-ROM that model the CE give much better results than the DDC-ROM that does not model the CE. For low r and δ values, these improvements are dramatic.

r	G-ROM	δ	DDC-ROM	δ	IDDC-ROM	δ	CE-DDC-ROM
2	1.7759e-03	10^{-5}	1.3238e-03	10^{-4}	1.4649e-05	10^{-4}	1.4809e-05
3	1.9301e-04	10^{-5}	1.5514e-04	10^{-6}	1.7270e-07	10^{-6}	1.5514e-07
4	2.0028e-05	10^{-4}	1.6878e-05	10^{-6}	9.4597e-08	10^{-6}	9.4866e-08
5	2.0344e-06	10^{-4}	1.7697e-06	10^{-5}	9.2432e-08	10^{-5}	9.2657e-08
6	2.2325e-07	10^{-5}	2.0245e-07	10^{-5}	9.3847e-08	10^{-5}	9.3948e-08
7	9.5581e-08	10^{-6}	9.5582e-08	10^{-8}	9.5581e-08	10^{-5}	9.5581e-08

Table 6.7: Errors for G-ROM, DDC-ROM, IDDC-ROM, and CE-DDC-ROM with the differential filter for $\nu = 10^{-1}$, $d = 7$ and different r and δ values.

r	G-ROM	δ	DDC-ROM	δ	IDDC-ROM	δ	CE-DDC-ROM
1	2.2069e-07	10^{-3}	5.8083e-08	10^{-5}	3.4969e-10	10^{-6}	4.0656e-10
2	9.3295e-10	10^{-8}	6.6797e-10	10^{-8}	5.3548e-13	10^{-8}	5.3661e-13
3	3.4616e-12	10^{-8}	3.4616e-12	10^{-8}	3.4616e-12	10^{-8}	3.4616e-12

Table 6.8: Errors for G-ROM, DDC-ROM, IDDC-ROM, and CE-DDC-ROM with the differential filter for $\nu = 10^{-3}$, $d = 3$ and different r and δ values.

6.2 Burgers Equation with Step Function IC

In this section, we consider the Burgers equation (6.3) with a step-function as an initial condition:

$$u_0(x) = \begin{cases} 1, & x \in (0, 1/2], \\ 0, & x \in (1/2, 1]. \end{cases} \tag{6.5}$$

To obtain the snapshots, we use piecewise linear FEs with $\nu = 10^{-1}$ or $\nu = 10^{-3}$, $h = 1/2048$, and $\Delta t = 10^{-3}$.

For both the projection filter and the differential filter, we investigate whether the CE is large. Furthermore, we test whether adding the CE to the data-driven model (i.e., the IDDC-ROM and CE-DDC-ROM) yields better results than the standard DDC-ROM, which does not model the CE.

6.2.1 Results with the Projection Filter

Tables 6.9 and 6.10 show that the CE obtained by using the projection filter continuously increases as r increases, then gradually decreases when r approaches d.

r	$\|\mathcal{E}_\Delta[u_d]\|_{L^2(L^2)}$
2	4.2870e+00
3	2.7545e+01
5	3.2503e+02
7	3.0400e+02
9	1.1717e+02
11	3.5953e+01
17	5.2702e-01
19	0

Table 6.9: CE obtained by using the projection filter with $\nu = 10^{-1}$, $d = 19$, and different r values.

r	$\|\mathcal{E}_\Delta[u_d]\|_{L^2(L^2)}$
2	3.0343e+00
5	1.0171e+01
9	2.1679e+01
15	4.4718e+01
35	1.9577e+02
50	4.2858e+02
150	1.0887e+02
200	1.3910e+01
257	0

Table 6.10: CE obtained by using the projection filter with $\nu = 10^{-3}$, $d = 257$, and different r values.

Tables 6.11 and 6.12 show that the CE-DDC-ROM yields slightly better results than the DDC-ROM, for low r values. However, there exists improvement, especially for moderate r values.

r	G-ROM	DDC-ROM	IDDC-ROM	CE-DDC-ROM
2	7.2683e-03	6.9310e-03	2.7100e-04	3.7081e-03
3	1.5103e-02	8.1417e-03	5.8945e-05	3.8374e-03
5	4.2247e-03	4.2245e-03	5.9749e-07	3.1367e-04
7	9.5862e-04	9.5851e-04	1.3168e-07	6.4433e-05
9	2.3447e-04	2.3447e-04	1.3876e-07	4.2571e-06
11	5.4444e-05	5.4443e-05	1.3025e-07	1.2845e-07
17	4.9677e-07	4.9963e-07	1.3409e-07	1.1507e-07
19	1.3011e-07	1.3011e-07	1.3011e-07	1.3011e-07

Table 6.11: Errors for G-ROM, DDC-ROM, IDDC-ROM, and CE-DDC-ROM with the projection filter for $\nu = 10^{-1}$, $d = 19$, and different r values.

r	G-ROM	DDC-ROM	IDDC-ROM	CE-DDC-ROM
2	1.0105e-01	1.9634e-03	1.9507e-03	1.9052e-03
5	1.7184e-01	1.4857e-04	2.6786e-05	9.3733e-05
9	1.5024e-01	3.3651e-04	8.3660e-05	3.0930e-05
15	7.9857e-02	5.8386e-04	3.8260e-04	4.5483e-04
35	1.4313e-02	1.2551e-03	4.0831e-04	6.8291e-04
50	5.0154e-03	1.2274e-03	2.1203e-04	2.8249e-04

Table 6.12: Errors for G-ROM, DDC-ROM, IDDC-ROM, and CE-DDC-ROM with the the projection filter for $\nu = 10^{-3}$, $d = 257$, and different r values.

6.2.2 Results with the Differential Filter

Tables 6.13 and 6.14 list the CE obtained by using the differential filter for different r and δ values. The former uses $\nu = 10^{-1}$ and the latter uses $\nu = 10^{-3}$. We observe that the CE from Tables 6.13 and 6.14 decreases as r approaches $d = 19$ and $d = 257$, respectively. For fixed r values in both tables, if δ decreases, the CE increases significantly.

r	δ	$\|\mathcal{E}_\Delta[u_d]\|_{L^2(L^2)}$	r	δ	$\|\mathcal{E}_\Delta[u_d]\|_{L^2(L^2)}$
2	10^{1}	9.1001e-04	9	10^{1}	4.7889e-06
2	10^{-1}	2.8579e+00	9	10^{-1}	4.2561e-02
2	10^{-3}	4.2868e+00	9	10^{-3}	6.2283e+01
2	10^{-5}	4.2870e+00	9	10^{-5}	1.1716e+02
3	10^{1}	7.8066e-04	11	10^{1}	1.1043e-06
3	10^{-1}	5.8377e+00	11	10^{-1}	9.5354e-03
3	10^{-3}	2.7534e+01	11	10^{-3}	1.515e+01
3	10^{-5}	2.7545e+01	11	10^{-5}	3.5948e+01
5	10^{1}	9.2349e-05	17	10^{1}	7.5644e-09
5	10^{-1}	7.8572e-01	17	10^{-1}	7.3077e-05
5	10^{-3}	3.0786e+02	17	10^{-3}	1.3347e-01
5	10^{-5}	3.250e+02	17	10^{-5}	5.2686e-01
7	10^{1}	2.2982e-05	19	10^{1}	1.5531e-12
7	10^{-1}	1.8142e-01	19	10^{-1}	2.2454e-11
7	10^{-3}	2.1421e+02	19	10^{-3}	3.1225e-12
7	10^{-5}	3.0399e+02	19	10^{-5}	1.5531e-12

Table 6.13: CE obtained by using the differential filter with $\nu = 10^{-1}$, $d = 19$, and different r and δ values.

r	δ	$\|\mathcal{E}_\Delta[u_d]\|_{L^2(L^2)}$	r	δ	$\|\mathcal{E}_\Delta[u_d]\|_{L^2(L^2)}$
2	10^{1}	6.9415e-04	35	10^{1}	9.0522e-05
2	10^{-1}	2.0281e+00	35	10^{-1}	6.8350e-01
2	10^{-3}	3.0341e+00	35	10^{-3}	1.8659e+02
2	10^{-5}	3.0343e+00	35	10^{-5}	1.9577e+02
5	10^{1}	3.6547e-04	50	10^{1}	4.3770e-05
5	10^{-1}	1.9895e+00	50	10^{-1}	3.5810e-01
5	10^{-3}	1.0165e+01	50	10^{-3}	3.6679e+02
5	10^{-5}	1.0171e+01	50	10^{-5}	4.2858e+02
9	10^{1}	2.4840e-04	150	10^{1}	1.4596e-07
9	10^{-1}	1.5428e+00	150	10^{-1}	1.4531e-03
9	10^{-3}	2.1634e+01	150	10^{-3}	1.0156e+01
9	10^{-5}	2.1679e+01	150	10^{-5}	1.0875e+02
15	10^{1}	1.8536e-04	257	10^{1}	1.3712e-13
15	10^{-1}	1.2344e+00	257	10^{-1}	1.3258e-10
15	10^{-3}	4.4439e+01	257	10^{-3}	8.3491e-11
15	10^{-5}	4.4718e+01	257	10^{-5}	8.6136e-12

Table 6.14: CE obtained by using the differential filter with $\nu = 10^{-3}$, $d = 257$, and different r and δ values.

Tables 6.15 and 6.16 show that the CE-DDC-ROM yields slightly better results than the DDC-ROM, for low r values. However, there exists improvement, especially for moderate r values.

r	G-ROM	δ	DDC-ROM	δ	IDDC-ROM	δ	CE-DDC-ROM
2	7.2683e-03	10^{-4}	6.9310e-03	10^{-3}	2.6898e-04	10^{-4}	3.7082e-03
3	1.5103e-02	10^{-4}	8.1467e-03	10^{-4}	5.8663e-05	10^{-4}	3.8374e-03
5	4.2247e-03	10^{-4}	4.2247e-03	10^{-7}	5.9749e-07	10^{-4}	3.1347e-04
7	9.5862e-04	10^{-5}	9.5851e-04	10^{-7}	1.3168e-07	10^{-4}	6.3809e-05
9	2.3447e-04	10^{-4}	2.3524e-04	10^{-6}	1.4200e-07	10^{-5}	4.2609e-06
11	5.4444e-05	10^{-6}	5.4443e-05	10^{-7}	1.3758e-07	10^{-6}	1.4834e-07
17	4.9677e-07	10^{-7}	4.9677e-07	10^{-7}	1.3495e-07	10^{-7}	1.1870e-07
19	1.3011e-07	10^{-7}	1.3011e-07	10^{-7}	1.3011e-07	10^{-7}	1.3011e-07

Table 6.15: Errors for G-ROM, DDC-ROM, IDDC-ROM, and CE-DDC-ROM with the differential filter for $\nu = 10^{-1}$, $d = 19$, and different r and δ values.

r	G-ROM	δ	DDC-ROM	δ	IDDC-ROM	δ	CE-DDC-ROM
2	1.0105e-01	10^{-3}	1.9634e-03	10^{-3}	1.9509e-03	10^{-3}	1.9054e-03
5	1.7184e-01	10^{-3}	1.3354e-04	10^{-5}	2.6785e-05	10^{-5}	1.5519e-04
9	1.5024e-01	10^{-3}	3.3477e-04	10^{-5}	1.9275e-04	10^{-5}	1.5581e-04
15	7.9857e-02	10^{-3}	2.9602e-03	10^{-5}	3.8266e-04	10^{-3}	1.5465e-03
35	1.4313e-02	10^{-5}	1.8678e-03	10^{-4}	5.6721e-04	10^{-5}	1.3230e-03
50	5.0154e-03	10^{-5}	1.8283e-03	10^{-5}	2.1537e-04	10^{-5}	3.6398e-04

Table 6.16: Errors for G-ROM, DDC-ROM, IDDC-ROM, and CE-DDC-ROM with the differential filter for $\nu = 10^{-3}$, $d = 257$, and different r and δ values.

Chapter 7

Conclusions and Future Work

In this thesis, we have investigated whether differentiation and ROM spatial filtering commute. Specifically, we studied theoretically and numerically whether the Laplacian commutation error (CE) exists, i.e., whether $\overline{\Delta u} = \Delta \overline{u}$. To this end, we considered two ROM spatial filters: the ROM projection and the ROM differential filter. For both filters, we showed both theoretically and numerically that the Laplacian CE is nonzero. We also investigated whether the CE has any effect on the ROM itself. Specifically, we considered the data-driven correction ROM (DDC-ROM), which was developed by using ROM spatial filtering. We also considered the ideal DDC-ROM (IDDC-ROM), which is the standard DDC-ROM supplemented with a fine resolution representation of the CE. Finally, we investigated a commutation error DDC-ROM (CE-DDC-ROM), which consists of the DDC-ROM and a data-driven model for the CE. We compared all the ROMs in the numerical simulation of the Burgers equation with viscosities $\nu = 10^{-1}$ and $\nu = 10^{-3}$ and with smooth and non-smooth initial conditions. In all the numerical tests, the IDDC-ROM and CE-DDC-ROM performed significantly better than the standard DDC-ROM. Thus, we concluded that the CE has a significant effect on the ROM and should be included in the model. Furthermore, we noticed that as the viscosity decreases, the impact of the CE on the ROM diminishes.

We plan to further investigate the lower bound of the viscosity value for which the CE plays a significant role in the ROM. We also plan to investigate more complex test problems, such as the 3D flow past a circular cylinder at different Reynolds numbers. Finally, we also plan to study the effect of the CE on the approximate deconvolution ROM [72], which is another ROM developed by using ROM spatial filtering.

Bibliography

[1] D. Amsallem and C. Farhat. Interpolation method for adapting reduced-order models and application to aeroelasticity. *AIAA journal*, 46(7):1803–1813, 2008.

[2] D. Amsallem and C. Farhat. Stabilization of projection-based reduced-order models. *Int. J. Num. Meth. Eng.*, 91(4):358–377, 2012.

[3] N. Aubry, P. Holmes, J. L. Lumley, and E. Stone. The dynamics of coherent structures in the wall region of a turbulent boundary layer. *J. Fluid Mech.*, 192:115–173, 1988.

[4] J. Baiges, R. Codina, and S. Idelsohn. Reduced-order subscales for POD models. *Comput. Methods Appl. Mech. Engrg.*, 291:173–196, 2015.

[5] F. Ballarin, E. Faggiano, S. Ippolito, A. Manzoni, A. Quarteroni, G. Rozza, and R. Scrofani. Fast simulations of patient-specific haemodynamics of coronary artery bypass grafts based on a POD–Galerkin method and a vascular shape parametrization. *J. Comput. Phys.*, 315:609–628, 2016.

[6] F. Ballarin, A. Manzoni, A. Quarteroni, and G. Rozza. Supremizer stabilization of POD–Galerkin approximation of parametrized steady incompressible Navier–Stokes equations. *Int. J. Numer. Meth. Engng.*, 102:1136–1161, 2015.

[7] C. A. Beattie, J. Borggaard, S. Gugercin, and T. Iliescu. A domain decomposition approach to pod. In *Decision and Control, 2006 45th IEEE Conference on*, pages 6750–6756. IEEE, 2006.

[8] P. Benner, S. Gugercin, and K. Willcox. A survey of projection-based model reduction methods for parametric dynamical systems. *SIAM Rev.*, 57(4):483–531, 2015.

[9] M. Benosman, J. Borggaard, O. San, and B. Kramer. Learning-based robust stabilization for reduced-order models of 2D and 3D Boussinesq equations. *Appl. Math. Model.*, 49:162–181, 2017.

[10] M. Bergmann, A. Ferrero, A. Iollo, E. Lombardi, A. Scardigli, and H. Telib. A zonal Galerkin-free POD model for incompressible flows. *J. Comput. Phys.*, 352:301–325, 2018.

[11] G. Berkooz, P. Holmes, and J. L. Lumley. The proper orthogonal decomposition in the analysis of turbulent flows. *Annual review of fluid mechanics*, 25(1):539–575, 1993.

[12] L. C. Berselli, T. Iliescu, and W. J. Layton. *Mathematics of Large Eddy Simulation of Turbulent Flows*. Scientific Computation. Springer-Verlag, Berlin, 2006.

[13] D. A. Bistrian and I. M. Navon. An improved algorithm for the shallow water equations model reduction: Dynamic mode decomposition vs POD. *Int. J. Num. Meth. Fluids*, 78(9):552–580, 2015.

[14] J. Borggaard, A. Duggleby, A. Hay, T. Iliescu, and Z. Wang. Reduced-order modeling of turbulent flows. In *Proceedings of MTNS 2008*, 2008.

[15] J. Borggaard and T. Iliescu. Approximate deconvolution boundary conditions for large eddy simulation. *Applied mathematics letters*, 19(8):735–740, 2006.

[16] S. L. Brunton, J. L. Proctor, and J. N. Kutz. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proc. Natl. Acad. Sci.*, 113(15):3932–3937, 2016.

[17] J. R. Bull and A. Jameson. Explicit filtering and exact reconstruction of the sub-filter stresses in large eddy simulation. *Journal of Computational Physics*, 306:117–136, 2016.

[18] K. Carlberg, M. Barone, and H. Antil. Galerkin v. least-squares Petrov–Galerkin projection in nonlinear model reduction. *J. Comput. Phys.*, 330:693–734, 2017.

[19] A. Chatterjee. An introduction to the proper orthogonal decomposition. *Current science*, pages 808–817, 2000.

[20] Y. Choi and K. Carlberg. Space-time least-squares Petrov-Galerkin projection for nonlinear model reduction. *arXiv preprint arXiv:1703.04560*, 2017.

[21] F.K. Chow, R.L. Street, M. Xue, and J.H. Ferziger. Explicit filtering and reconstruction turbulence modeling for large-eddy simulation of neutral boundary layer flow. *Journal of the atmospheric sciences*, 62(7):2058–2077, 2005.

[22] T. Cui, Y. M. Marzouk, and K. E. Willcox. Data-driven model reduction for the bayesian solution of inverse problems. *International Journal for Numerical Methods in Engineering*, 102(5):966–990, 2015.

[23] H. Fareed and J. R. Singler. A note on incremental POD algorithms for continuous time data. *arXiv preprint arXiv:1807.00045*, 2018.

[24] F. Feppon and P. F. J. Lermusiaux. Dynamically orthogonal numerical schemes for efficient stochastic advection and Lagrangian transport. *SIAM Rev.*, 60(3):595–625, 2018.

[25] L. Fick, Y. Maday, A. T. Patera, and T. Taddei. A reduced basis technique for long-time unsteady turbulent flows. *arXiv preprint arXiv:1710.03569*, 2017.

[26] M. Germano. Differential filters of elliptic type. *Phys. Fluids*, 29(6):1757–1758, 1986.

[27] M. Germano, U. Piomelli, P. Moin, and W. H. Cabot. A dynamic subgrid-scale eddy viscosity model. *Physics of Fluids A: Fluid Dynamics*, 3(7):1760–1765, 1991.

[28] A. Gouasmi, E. J. Parish, and K. Duraisamy. A priori estimation of memory effects in reduced-order models of nonlinear systems using the Mori–Zwanzig formalism. *Proc. R. Soc. A*, 473(2205):20170385, 2017.

[29] M. Gunzburger, N. Jiang, and M. Schneier. An ensemble-proper orthogonal decomposition method for the nonstationary Navier-Stokes equations. *SIAM J. Numer. Anal.*, 55(1):286–304, 2017.

[30] J. S. Hesthaven, G. Rozza, and B. Stamm. *Certified Reduced Basis Methods for Parametrized Partial Differential Equations*. Springer, 2015.

[31] P. Holmes, J. L. Lumley, and G. Berkooz. *Turbulence, Coherent Structures, Dynamical Systems and Symmetry*. Cambridge, 1996.

[32] T. Iliescu, V. John, W. J. Layton, G. Matthies, and L. Tobiska. A numerical study of a class of les models. *International Journal of Computational Fluid Dynamics*, 17(1):75–85, 2003.

[33] V. John. *Large eddy simulation of turbulent incompressible flows: analytical and numerical results for a class of LES models*, volume 34. Springer Science & Business Media, 2012.

[34] V. John and W. J. Layton. Analysis of numerical errors in large eddy simulation. *SIAM Journal on Numerical Analysis*, 40(3):995–1020, 2002.

[35] B. Koc, M. Mohebujjaman, C. Mou, and T. Iliescu. Commutation error in reduced order modeling of fluid flows. *arXiv preprint arXiv:1810.00517*, 2018.

[36] D. Kondrashov, M. D. Chekroun, and M. Ghil. Data-driven non-Markovian closure models. *Phys. D*, 297:33–55, 2015.

[37] K. Kunisch and S. Volkwein. Control of the burgers equation by a reduced-order approach using proper orthogonal decomposition. *Journal of optimization theory and applications*, 102(2):345–371, 1999.

[38] K. Kunisch and S. Volkwein. Galerkin proper orthogonal decomposition methods for parabolic problems. *Numerische mathematik*, 90(1):117–148, 2001.

[39] K. Kunisch and S. Volkwein. Galerkin proper orthogonal decomposition methods for a general equation in fluid dynamics. *SIAM Journal on Numerical analysis*, 40(2):492–515, 2002.

[40] W. J. Layton and R. Lewandowski. Analysis of an eddy viscosity model for large eddy simulation of turbulent flows. *Journal of Mathematical Fluid Mechanics*, 4(4):374–399, 2002.

[41] W. J. Layton and L. G. Rebholz. *Approximate Deconvolution Models of Turbulence: Analysis, Phenomenology and Numerical Analysis*, volume 2042. Springer Berlin Heidelberg, 2012.

[42] J. C. Loiseau and S. L. Brunton. Constrained sparse Galerkin regression. *J. Fluid Mech.*, 838:42–67, 2018.

[43] F. Lu, K. K. Lin, and A. J. Chorin. Data-based stochastic model reduction for the Kuramoto–Sivashinsky equation. *Phys. D*, 340:46–57, 2017.

[44] A. Majda and N. Chen. Model error, information barriers, state estimation and prediction in complex multiscale systems. *Entropy*, 20(9):644, 2018.

[45] M. Mohebujjaman, L.G. Rebholz, and T. Iliescu. Physically-constrained data-driven, filtered reduced order modeling of fluid flows. *arXiv preprint arXiv:1806.00350*, 2018.

[46] C. Mou. Cross-validation of data-driven correction reduced order modeling, 2018.

[47] B. R. Noack, M. Morzynski, and G. Tadmor. *Reduced-Order Modelling for Flow Control*, volume 528. Springer Verlag, 2011.

[48] A. A. Oberai and J. Jagalur-Mohan. Approximate optimal projection for reduced-order models. *Int. J. Num. Meth. Engng.*, 105(1):63–80, 2016.

[49] J. Östh, B. R. Noack, S. Krajnović, D. Barros, and J. Borée. On the need for a nonlinear subscale turbulence term in POD models as exemplified for a high-Reynolds-number flow over an Ahmed body. *J. Fluid Mech.*, 747:518–544, 2014.

[50] S. Pan and K. Duraisamy. Data-driven discovery of closure models. *arXiv preprint arXiv:1803.09318*, 2018.

[51] E. J. Parish, C. Wentland, and K. Duraisamy. A residual-based petrov-galerkin reduced-order model with memory effects. *arXiv preprint arXiv:1810.03455*, 2018.

[52] B. Peherstorfer and K. Willcox. Dynamic data-driven reduced-order models. *Comput. Methods Appl. Mech. Engrg.*, 291:21–41, 2015.

[53] B. Peherstorfer and K. Willcox. Data-driven operator inference for nonintrusive projection-based model reduction. *Comput. Methods Appl. Mech. Engrg.*, 306:196–215, 2016.

[54] G. Pitton, A. Quaini, and G. Rozza. Computational reduction strategies for the detection of steady bifurcations in incompressible fluid-dynamics: Applications to Coanda effect in cardiology. *J. Comput. Phys.*, 344:534–557, 2017.

[55] B. Protas, B. R. Noack, and J. Östh. Optimal nonlinear eddy viscosity in Galerkin models of turbulent flows. *J. Fluid Mech.*, 766:337–367, 2015.

[56] A. Quarteroni, A. Manzoni, and F. Negri. *Reduced Basis Methods for Partial Differential Equations: An Introduction*, volume 92. Springer, 2015.

[57] T. C. Rebollo, E. D. Ávila, M. G. Mármol, F. Ballarin, and G. Rozza. On a certified smagorinsky reduced basis turbulence model. *SIAM Journal on Numerical Analysis*, 55(6):3047–3067, 2017.

[58] C. W. Rowley. Model reduction for fluids, using balanced proper orthogonal decomposition. *International Journal of Bifurcation and Chaos*, 15(03):997–1013, 2005.

[59] P. Sagaut. *Large Eddy Simulation for Incompressible Flows*. Scientific Computation. Springer-Verlag, Berlin, third edition, 2006.

[60] O. San and R. Maulik. Neural network closures for nonlinear model order reduction. *Adv. Comput. Math.*, pages 1–34, 2018.

[61] O. San, A. E. Staples, and T. Iliescu. Approximate deconvolution large eddy simulation of a stratified two-layer quasigeostrophic ocean model. *Ocean Modelling*, 63:1–20, 2013. in press.

[62] O. San, A. E. Staples, Z. Wang, and T. Iliescu. Approximate deconvolution large eddy simulation of a barotropic ocean circulation model. *Ocean Modelling*, 40:120–132, 2011.

[63] G. Stabile and G. Rozza. Finite volume POD-Galerkin stabilised reduced order methods for the parametrised incompressible Navier-Stokes equations. *arXiv preprint arXiv:1710.11580*, 2017.

[64] M. Strazzullo, F. Ballarin, R. Mosetti, and G. Rozza. Model reduction for parametrized optimal control problems in environmental marine sciences and engineering. *SIAM J. Sci. Comput.*, 40(4):B1055–B1079, 2018.

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

[66] S. Volkwein. Model reduction using proper orthogonal decomposition. *Lecture Notes, Institute of Mathematics and Scientific Computing, University of Graz. see http://www. uni-graz. at/imawww/volkwein/POD. pdf*, 1025, 2011.

[67] S. Volkwein. Proper orthogonal decomposition: Theory and reduced-order modelling. *Lecture Notes, University of Konstanz*, 4(4), 2013.

[68] Z. Wang, I. Akhtar, J. Borggaard, and T. Iliescu. Two-level discretizations of nonlinear closure models for proper orthogonal decomposition. *J. Comput. Phys.*, 230:126–146, 2011.

[69] Z. Wang, I. Akhtar, J. Borggaard, and T. Iliescu. Proper orthogonal decomposition closure models for turbulent flows: A numerical comparison. *Comput. Meth. Appl. Mech. Eng.*, 237-240:10–26, 2012.

[70] D. Wells, Z. Wang, X. Xie, and T. Iliescu. An evolve-then-filter regularized reduced order model for convection-dominated flows. *Int. J. Num. Meth. Fluids*, 84:598–???615, 2017.

[71] X. Xie, M. Mohebujjaman, L. G. Rebholz, and T. Iliescu. Data-driven filtered reduced order modeling of fluid flows. *SIAM J. Sci. Comput.*, 40(3):B834–B857, 2018.

[72] X. Xie, D. Wells, Z. Wang, and T. Iliescu. Approximate deconvolution reduced order modeling. *Comput. Methods Appl. Mech. Engrg.*, 313:512–534, 2017.

[73] M. Yano. Discontinuous Galerkin reduced basis empirical quadrature procedure for model reduction of parametrized nonlinear conservation laws. 2018.