

# DATA-DRIVEN FILTERED REDUCED ORDER MODELING OF FLUID FLOWS

X. XIE <sup>\*</sup>, M. MOHEBUJJAMAN <sup>†</sup>, L. G. REBHOLZ <sup>‡</sup>, AND T. ILIESCU <sup>§</sup>

**Abstract.** We propose a data-driven filtered reduced order model (DDF-ROM) framework for the numerical simulation of fluid flows. The novel DDF-ROM framework consists of two steps: (i) In the first step, we use explicit ROM spatial filtering of the nonlinear PDE to construct a filtered ROM. This filtered ROM is low-dimensional, but is not closed (because of the nonlinearity in the given PDE). (ii) In the second step, we use data-driven modeling to close the filtered ROM, i.e., to model the interaction between the resolved and unresolved modes. To this end, we use a quadratic ansatz to model this interaction and close the filtered ROM. To find the new coefficients in the closed filtered ROM, we solve an optimization problem that minimizes the difference between the full order model data and our ansatz. We emphasize that the new DDF-ROM is built on general ideas of spatial filtering and optimization and is independent of (restrictive) phenomenological arguments.

We investigate the DDF-ROM in the numerical simulation of a 2D channel flow past a circular cylinder at Reynolds number  $Re = 100$ . The DDF-ROM is significantly more accurate than the standard projection ROM. Furthermore, the computational costs of the DDF-ROM and the standard projection ROM are similar, both costs being orders of magnitude lower than the computational cost of the full order model. We also compare the new DDF-ROM with modern ROM closure models in the numerical simulation of the 1D Burgers equation. The DDF-ROM is more accurate and significantly more efficient than these ROM closure models.

**Key words.** reduced order modeling, data-driven modeling, spatial filter

**AMS subject classifications.** 65M60, 76F65

**1. Introduction.** Reduced order models (ROMs) have been successfully used to reduce the computational cost of scientific and engineering applications that are governed by relatively few recurrent dominant spatial structures [2, 7, 10, 13, 29, 30, 31, 46, 52, 54, 63].

One of the most popular classes of ROMs is the *projection ROMs* (*Proj-ROMs*). For a given general partial differential equation (PDE), the Proj-ROM strategy for approximating the PDE's solution  $\mathbf{u}$ , is straightforward: (i) Choose modes  $\{\varphi_1, \dots, \varphi_d\}$ , which represent the recurrent spatial structures of the given PDE. (ii) Choose the dominant modes  $\{\varphi_1, \dots, \varphi_r\}$ ,  $r \leq d$ , as basis functions for the ROM. (iii) Use a Galerkin truncation  $\mathbf{u}_r = \sum_{j=1}^r a_j \varphi_j$ . (iv) Replace  $\mathbf{u}$  with  $\mathbf{u}_r$  in the given PDE. (v) Use a Galerkin projection of PDE( $\mathbf{u}_r$ ) onto the ROM space  $\mathbf{X}^r := \text{span}\{\varphi_1, \dots, \varphi_r\}$  to obtain a low-dimensional dynamical system, which represents the Proj-ROM. For example, in fluid dynamics, the Proj-ROM often takes the following form:

$$\dot{\mathbf{a}} = A \mathbf{a} + \mathbf{a}^\top B \mathbf{a}, \quad (1.1)$$

where  $\mathbf{a}$  is the vector of unknown ROM coefficients and  $A \in \mathbb{R}^{r \times r}$ ,  $B \in \mathbb{R}^{r \times r \times r}$  are ROM operators. (vi) In an offline stage, compute the ROM operators. (vii) In an online stage, repeatedly use the Proj-ROM (1.1) (for various parameter settings

<sup>\*</sup>Department of Mathematics, Virginia Tech, Blacksburg, VA 24061 Partially supported by NSF DMS1522656, email: xupingxy@vt.edu

<sup>†</sup>Department of Mathematical Sciences, Clemson University, Clemson, SC, 29634; Partially supported by NSF DMS1522191, email: mmohebu@g.clemson.edu

<sup>‡</sup>Department of Mathematical Sciences, Clemson University, Clemson, SC 29634; Partially supported by NSF DMS1522191 and Army Research Office 65294-MA, email: rebholz@clemson.edu

<sup>§</sup>Department of Mathematics, Virginia Tech, Blacksburg, VA 24061 Partially supported by NSF DMS1522656, email: iliescu@vt.edu.

and/or longer time intervals). The Proj-ROM (1.1) is often efficient and relatively accurate [8, 31, 46], but can fail in realistic applications when large numbers of modes are needed to accurately represent the system. To ensure a low computational cost, it is desirable that Proj-ROMs use only a few modes (i.e., low  $r$  values) and discard the remaining modes  $\{\varphi_{r+1}, \dots, \varphi_d\}$ . The resulting Proj-ROM, however, can yield inaccurate results (see, e.g., [1, 5, 9, 11, 12, 18, 27, 49, 53]). The general explanation for these inaccurate results is that the Proj-ROM (1.1) fails to account for the interaction between resolved and unresolved modes [4, 22, 28, 48, 65, 66, 67]. Thus, in practical applications, the following modified Proj-ROM is used:

$$\dot{\mathbf{a}} = A \mathbf{a} + \mathbf{a}^\top B \mathbf{a} + \boldsymbol{\tau}, \quad (1.2)$$

where  $\boldsymbol{\tau}$  models the interaction between resolved modes  $\{\varphi_1, \dots, \varphi_r\}$  and unresolved modes  $\{\varphi_{r+1}, \dots, \varphi_d\}$ . Most often, a dissipation mechanism (e.g., eddy viscosity) is used to model  $\boldsymbol{\tau}$  in the modified Proj-ROM (1.2):

$$\boldsymbol{\tau} \approx EV(\mathbf{a}). \quad (1.3)$$

Another class of ROMs is the *data-driven ROMs (DD-ROMs)*, which are an extremely dynamic research area, and are fundamentally different from the Proj-ROMs presented above. Although both the DD-ROM and the Proj-ROM can be written as in (1.1), the operators  $A$  and  $B$  are constructed using fundamentally different approaches: the Proj-ROMs use the Galerkin projection (as explained above), whereas the DD-ROMs use the available full order model (FOM) or experimental data [16, 36]. Specifically, an optimization problem is solved to find the optimal operators  $A$  and  $B$ , i.e., the operators that ensure that the resulting DD-ROM trajectories are as close as possible (typically in a least-squares sense) to the available data. DD-ROM examples include the dynamic mode decomposition (DMD) [37, 57, 60], Koopman theory [44], the sparse identification of nonlinear dynamics (SINDy) algorithm [14], and the operator inference method [50, 51].

We propose a *hybrid projection/data-driven ROM (Proj-DD-ROM)* [15, 19, 21, 25, 26, 28, 42, 47] in order to combine the best parts of each approach. We use the projection to determine the operators  $A$  and  $B$  in (1.2) and data-driven modeling to determine the unknown  $\boldsymbol{\tau}$  in (1.2), which models the interaction between resolved and unresolved modes. The resulting ROM, which we call the *data-driven filtered ROM (DDF-ROM)*, is schematically illustrated in (1.4), below.

$$\boxed{\text{FOM} \xrightarrow{\text{filtering + projection}} \text{F-ROM} \xrightarrow{\text{data-driven modeling}} \text{DDF-ROM}} \quad (1.4)$$

In the first step of the DDF-ROM (1.4), we use a projection approach to find the operators  $A$  and  $B$  in (1.2) as well as an *explicit formula* for  $\boldsymbol{\tau}$ . To this end, we use a *ROM spatial filter* to filter the FOM, which contains all the information in the underlying system. The resulting *filtered ROM (F-ROM)* approximates only the large spatial structures of the system and, therefore, requires fewer modes than the FOM. The F-ROM takes the following form:

$$\dot{\mathbf{a}} = A \mathbf{a} + \mathbf{a}^\top B \mathbf{a} + \boldsymbol{\tau}(\text{FOM}), \quad (1.5)$$

where  $\boldsymbol{\tau}(\text{FOM})$  denotes the explicit dependence of  $\boldsymbol{\tau}$  on the FOM data.

We emphasize that to make the F-ROM (1.5) usable, one still needs to solve the *ROM closure problem*, i.e., to determine a formula of the form

$$\boldsymbol{\tau}(FOM) \approx \boldsymbol{\tau}(\mathbf{a}). \quad (1.6)$$

To this end, in the second step of the DDF-ROM (1.4), we use *data-driven modeling*. First, we employ a quadratic ansatz to model  $\boldsymbol{\tau}$  in (1.6):

$$\boldsymbol{\tau}(FOM) \approx \tilde{A} \mathbf{a} + \mathbf{a}^\top \tilde{B} \mathbf{a}. \quad (1.7)$$

Then, we find  $\tilde{A}$  and  $\tilde{B}$  in (1.7) by solving a (low-dimensional) optimization problem that minimizes the difference between  $\boldsymbol{\tau}$  calculated with the FOM data, and  $\boldsymbol{\tau}$  calculated with our ansatz:

$$\min_{\tilde{A}, \tilde{B}} \|\boldsymbol{\tau}(FOM) - (\tilde{A} \mathbf{a} + \mathbf{a}^\top \tilde{B} \mathbf{a})\|^2. \quad (1.8)$$

At the end of the two steps of (1.4), we obtain the DDF-ROM:

$$\boxed{\dot{\mathbf{a}} = (A + \tilde{A}) \mathbf{a} + \mathbf{a}^\top (B + \tilde{B}) \mathbf{a}} \quad (1.9)$$

We note that the hybrid Proj-DD-ROM approach used to construct the DDF-ROM (1.9) is different from the DD-ROM approach. Indeed, although we use data-driven modeling to develop the DDF-ROM, we do so to determine *only*  $\tilde{A}$  and  $\tilde{B}$  (and, thus,  $\boldsymbol{\tau}$ ). This is in contrast with standard DD-ROMs, where data-driven modeling is used to build *all* the operators, i.e., not only  $\tilde{A}$  and  $\tilde{B}$ , but also  $A$  and  $B$ .

The DDF-ROM (1.9) is also different from the classic Proj-ROMs, although the latter sometimes employ data-driven modeling. Indeed, Proj-ROMs generally use a dissipation ansatz (e.g., the eddy viscosity ansatz (1.3)) to model  $\boldsymbol{\tau}$  in (1.2). Thus, available data can only be used to determine the tuning parameters of these dissipative mechanisms [11, 53, 65]. In contrast, the DDF-ROM does not make any a priori assumptions regarding  $\boldsymbol{\tau}$  and data is used to determine *all* the components of  $\boldsymbol{\tau}$ . Thus, the DDF-ROM represents a *general* ROM framework that, in principle, can be used for the numerical simulation of any nonlinear PDE. The key tool that allows us to use data-driven modeling to determine all the components of  $\boldsymbol{\tau}$  (as opposed to only the tuning parameters, as in Proj-ROMs) is the ROM spatial filtering, which yields an *explicit formula* for  $\boldsymbol{\tau}$ . Indeed, once we know what exactly we want to model, we can use available data to model it.

Finally, we note that the DDF-ROM framework has some connections to some other popular models, in particular the nonlinear Galerkin [24], large eddy simulation (LES), and variational multiscale [32] methods, since they all use the small-large scale separation. However, DDF-ROM is different from all these methods since it uses a data-driven modeling approach to approximate the interaction with the unresolved modes, whereas the other methods do not (see, however, [38], for a notable exception).

The rest of the paper is organized as follows: In Section 2, we present the standard Proj-ROM. In Section 3, we introduce the filtered ROM. In Section 4, we use data-driven modeling to solve the closure problem in the filtered ROM and to construct the DDF-ROM. In Section 5, we investigate the DDF-ROM in the numerical simulation of a 2D channel flow past a circular cylinder. Finally, in Section 6, we draw conclusions and outline future research directions.

**2. Projection Reduced Order Models (Proj-ROMs).** In this section, we briefly review the proper orthogonal decomposition (Section 2.1) and the standard projection ROM (Section 2.2). Although the new DDF-ROM framework can be applied to many types of nonlinear PDEs, to present the method it is necessary to pick a particular model, and so we select our favorite, the incompressible Navier-Stokes equations (NSE):

$$\frac{\partial \mathbf{u}}{\partial t} - Re^{-1} \Delta \mathbf{u} + \mathbf{u} \cdot \nabla \mathbf{u} + \nabla p = \mathbf{0}, \quad (2.1)$$

$$\nabla \cdot \mathbf{u} = 0, \quad (2.2)$$

where  $\mathbf{u}$  is the velocity,  $p$  the pressure, and  $Re$  the Reynolds number. We use the initial condition  $\mathbf{u}(\mathbf{x}, 0) = \mathbf{u}_0(\mathbf{x})$  and (for simplicity) homogeneous Dirichlet boundary conditions:  $\mathbf{u}(\mathbf{x}, t) = \mathbf{0}$ .

**2.1. Proper Orthogonal Decomposition (POD).** One of the most popular reduced order modeling techniques is the proper orthogonal decomposition (POD) [31, 46, 62]. For the snapshots  $\{\mathbf{u}_h^1, \dots, \mathbf{u}_h^{N_s}\}$ , which are, e.g., finite element (FE) solutions of (2.1)–(2.2) at  $N_s$  different time instances, the POD seeks a low-dimensional basis that approximates the snapshots optimally with respect to a certain norm. In this paper, we choose the commonly used  $L^2$ -norm. The solution of the minimization problem is equivalent to the solution of the eigenvalue problem  $YY^T M_h \boldsymbol{\varphi}_j = \lambda_j \boldsymbol{\varphi}_j$ ,  $j = 1, \dots, N$ , where  $\boldsymbol{\varphi}_j$  and  $\lambda_j$  denote the vector of the FE coefficients of the POD basis functions and the POD eigenvalues, respectively,  $Y$  denotes the snapshot matrix, whose columns correspond to the FE coefficients of the snapshots,  $M_h$  denotes the FE mass matrix, and  $N$  is the dimension of the FE space  $\mathbf{X}^h$ . The eigenvalues are real and non-negative, so they can be ordered as follows:  $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_d \geq \lambda_{d+1} = \dots = \lambda_N = 0$ , where  $d$  is the rank of the snapshot matrix. The POD basis consists of the normalized functions  $\{\boldsymbol{\varphi}_j\}_{j=1}^r$ , which correspond to the first  $r \leq N$  largest eigenvalues. Thus, the POD space is defined as  $\mathbf{X}^r := \text{span}\{\boldsymbol{\varphi}_1, \dots, \boldsymbol{\varphi}_r\}$ .

**2.2. Standard Galerkin ROM (G-ROM).** The POD approximation of the velocity is defined as

$$\mathbf{u}_r(\mathbf{x}, t) \equiv \sum_{j=1}^r a_j(t) \boldsymbol{\varphi}_j(\mathbf{x}), \quad (2.3)$$

where  $\{a_j(t)\}_{j=1}^r$  are the sought time-varying coefficients, which are determined by solving the following system of equations:  $\forall i = 1, \dots, r$ ,

$$\left( \frac{\partial \mathbf{u}_r}{\partial t}, \boldsymbol{\varphi}_i \right) + Re^{-1} (\nabla \mathbf{u}_r, \nabla \boldsymbol{\varphi}_i) + \left( (\mathbf{u}_r \cdot \nabla) \mathbf{u}_r, \boldsymbol{\varphi}_i \right) = 0. \quad (2.4)$$

In (2.4), we assume that the modes  $\{\boldsymbol{\varphi}_1, \dots, \boldsymbol{\varphi}_r\}$  are perpendicular to the discrete pressure space, which is the case if standard, conforming LBB stable elements (such as Taylor-Hood, Scott-Vogelius, the mini-element, etc.) are used for the snapshot creation. Plugging (2.3) into (2.4) yields the *Galerkin ROM (G-ROM)*:

$$\dot{\mathbf{a}} = A \mathbf{a} + \mathbf{a}^\top B \mathbf{a}, \quad (2.5)$$

which can be written componentwise as follows:  $\forall i = 1 \dots r$ ,

$$\dot{a}_i = \sum_{m=1}^r A_{im} a_m(t) + \sum_{m=1}^r \sum_{n=1}^r B_{imn} a_n(t) a_m(t), \quad (2.6)$$

where

$$A_{im} = -Re^{-1} (\nabla \varphi_m, \nabla \varphi_i), \quad B_{imn} = -(\varphi_m \cdot \nabla \varphi_n, \varphi_i). \quad (2.7)$$

**3. Filtered ROMs (F-ROMs).** In this section, we present details regarding the first step of the DDF-ROM framework (1.4). That is, we discuss the creation of the *filtered ROM (F-ROM)*, from which the ROM closure problem will reveal itself. To this end, in Section 3.1 we present the ROM projection, which is the explicit ROM spatial filter that we use to construct the DDF-ROM. In Section 3.2, we develop the F-ROM framework, which has also been used to develop LES-ROMs [67]. Finally, in Section 3.3, we outline the celebrated ROM closure problem, which needs to be solved in the F-ROM. We emphasize that *the ROM closure problem is treated completely differently in DDF-ROM and LES-ROM*: DDF-ROM uses data-driven modeling, while LES-ROMs generally use phenomenological arguments (e.g., energy cascade and eddy viscosity).

**3.1. ROM Spatial Filtering.** Spatial filtering has been used in ROMs, mainly as a preprocessing tool to eliminate the noise in the snapshot data (see, e.g., Section 5 in [3] for a survey of relevant work). However, our approach is fundamentally different: We explicitly use spatial filtering in the construction of the actual ROM, not in the development of the ROM basis. In this paper, we exclusively use the ROM projection [65, 66] as a spatial filter, but we note that we could also use other spatial filters (e.g., the ROM differential filter [66, 67]).

For a fixed  $r \leq d$  and a given  $\mathbf{u} \in \mathbf{X}^h$ , the ROM projection [65, 66] seeks  $\bar{\mathbf{u}}^r \in \mathbf{X}^r$  such that

$$(\bar{\mathbf{u}}^r, \varphi_j) = (\mathbf{u}, \varphi_j) \quad \forall j = 1, \dots, r. \quad (3.1)$$

**3.2. F-ROM Framework.** To outline the F-ROM framework, we use the standard LES approach [39, 56, 58], which consists of the following steps: (i) Use an explicit spatial filter to filter the NSE. (ii) Use the resulting spatially filtered NSE and the ROM approximation to obtain the F-ROM.

Filtering the NSE, assuming that differentiation and filtering commute [58], and projecting the resulting equations onto a space of weakly divergence-free functions  $\phi$ , we obtain the spatially filtered NSE (see equations (35)–(36) in [67]):

$$\left( \frac{\partial \bar{\mathbf{u}}}{\partial t}, \phi \right) + Re^{-1} \left( \nabla \bar{\mathbf{u}}, \nabla \phi \right) + \left( (\bar{\mathbf{u}} \cdot \nabla) \bar{\mathbf{u}}, \phi \right) + \left( \boldsymbol{\tau}^{SFS}, \phi \right) = \mathbf{0}, \quad (3.2)$$

where

$$\boldsymbol{\tau}^{SFS} = \overline{(\mathbf{u} \cdot \nabla) \mathbf{u}} - (\bar{\mathbf{u}} \cdot \nabla) \bar{\mathbf{u}} \quad (3.3)$$

is the *subfilter-scale stress tensor*.

The spatial structures in the spatially filtered NSE (3.2) are larger than the spatial structures in the NSE (2.1). Thus, we expect that for a fixed target numerical accuracy of the ROM, the spatially filtered NSE will require fewer POD modes than the NSE, which is advantageous from a computational point of view.

Of course, to develop a useful ROM from the spatially filtered NSE (3.2), we need to (i) use a ROM approximation for the continuous velocity field  $\bar{\mathbf{u}}$ , and (ii) use a ROM approximation of the spatial filter. We use  $\mathbf{u}_d \sim \mathbf{u}$  in (i) (where  $d$  is the rank of the snapshot matrix) and  $\bar{\mathbf{u}}^r \sim \bar{\mathbf{u}}$  in (ii) (where  $\bar{\mathbf{u}}^r$  is the ROM projection (3.1)).

Using  $\mathbf{u}_d \sim \mathbf{u}$  in (i) means that we employ the best possible approximation of the continuous velocity field  $\mathbf{u}$  in the set of snapshots (i.e., in  $\mathbf{X}^d$ ). Using the ROM projection onto  $\mathbf{X}^r$  as the spatial filter in (ii) means that we are projecting the equations from  $\mathbf{X}^d$  onto  $\mathbf{X}^r$ . With these choices in the spatially filtered NSE (3.2), we obtain:  $\forall i = 1, \dots, r$ ,

$$\left( \frac{\partial \overline{\mathbf{u}_d^r}}{\partial t}, \boldsymbol{\varphi}_i \right) + Re^{-1} \left( \nabla \overline{\mathbf{u}_d^r}, \nabla \boldsymbol{\varphi}_i \right) + \left( (\overline{\mathbf{u}_d^r} \cdot \nabla) \overline{\mathbf{u}_d^r}, \boldsymbol{\varphi}_i \right) + \left( \boldsymbol{\tau}_r^{SFS}, \boldsymbol{\varphi}_i \right) = \mathbf{0} \quad (3.4)$$

where

$$\boldsymbol{\tau}_r^{SFS} = \overline{(\mathbf{u}_d \cdot \nabla) \mathbf{u}_d}^r - (\overline{\mathbf{u}_d^r} \cdot \nabla) \overline{\mathbf{u}_d^r}. \quad (3.5)$$

Since we are using the ROM projection onto  $\mathbf{X}^r$  as the spatial filter, we have

$$\overline{\mathbf{u}_d^r} = \mathbf{u}_r. \quad (3.6)$$

Plugging (3.6) in (3.4)–(3.5), we obtain the following system of equations:

$$\left( \frac{\partial \mathbf{u}_r}{\partial t}, \boldsymbol{\varphi}_i \right) + Re^{-1} \left( \nabla \mathbf{u}_r, \nabla \boldsymbol{\varphi}_i \right) + \left( (\mathbf{u}_r \cdot \nabla) \mathbf{u}_r, \boldsymbol{\varphi}_i \right) + \left( \boldsymbol{\tau}_r^{SFS}, \boldsymbol{\varphi}_i \right) = \mathbf{0}, \quad (3.7)$$

where the ROM stress tensor is

$$\boldsymbol{\tau}_r^{SFS} = \overline{(\mathbf{u}_d \cdot \nabla) \mathbf{u}_d}^r - (\mathbf{u}_r \cdot \nabla) \mathbf{u}_r. \quad (3.8)$$

Now plugging (2.3) into (3.7) yields the *F-ROM*:

$$\dot{\mathbf{a}} = A \mathbf{a} + \mathbf{a}^\top B \mathbf{a} + \boldsymbol{\tau}, \quad (3.9)$$

where  $A$  and  $B$  are given by (2.7) and the components of  $\boldsymbol{\tau}$  are given by

$$\tau_i = \left( \boldsymbol{\tau}_r^{SFS}, \boldsymbol{\varphi}_i \right), \quad i = 1, \dots, r. \quad (3.10)$$

**REMARK 3.1 (F-ROM Consistency).** *We note that the F-ROM is consistent with the NSE. Indeed, the F-ROM (3.9) is just the projection of the full order model (i.e., the best representation of the NSE in the snapshot space,  $\mathbf{X}^d$ ) onto the ROM space,  $\mathbf{X}^r$ . Thus, as  $r \rightarrow d$ , the F-ROM is expected to converge to the best representation of F-ROM in the snapshot space.*

*We emphasize that many other ROMs (e.g., eddy viscosity ROMs [4, 31, 65]) are not consistent with the NSE. Indeed, since the G-ROM is modified empirically, the resulting eddy viscosity ROM no longer corresponds to a Galerkin projection of the NSE [6].*

**3.3. F-ROM Closure Modeling.** The F-ROM (3.9) is an  $r$ -dimensional ODE system for  $\mathbf{u}_r$ . Since  $r \ll N$ , the F-ROM (3.9) is a computationally efficient surrogate model for the FOM (i.e., the FE approximation of the NSE, which is an  $N$ -dimensional ODE system). We emphasize, however, that the F-ROM (3.9) is not a closed system of equations, since the ROM stress tensor  $\boldsymbol{\tau}_r^{SFS}$  (which is used in the definition of  $\boldsymbol{\tau}$ ; see equation (3.10)) depends on  $\mathbf{u}_d$  (see equation (3.8)). Thus, to close the F-ROM (3.9), we need to solve the ROM closure problem [19, 25, 42, 48, 61, 65], i.e., to find a formula  $\boldsymbol{\tau} \approx \boldsymbol{\tau}(\mathbf{a})$ .

Note that by neglecting the last term on the LHS of (3.9), the F-ROM (3.9) is identical to the standard G-ROM (2.5). Thus, formally, one could write the following decomposition:

$$\boxed{\text{F-ROM} = \text{G-ROM} + \boldsymbol{\tau}} \quad (3.11)$$

The decomposition (3.11) is not new. Indeed, for complex flows, the G-ROM is supplemented with extra terms, which generally provide a dissipation mechanism (e.g., eddy viscosity) [4, 65]. However, what is new in the F-ROM is the *explicit formula for  $\boldsymbol{\tau}$*  (see equations (3.8) and (3.10)), *which allows for the first time the use of data-driven modeling of the entire missing ROM information*. We do exactly this in the next section.

**4. Data-Driven Filtered ROM (DDF-ROM).** In this section, we construct the new data-driven filtered ROM. Specifically, we use data-driven modeling [14, 15, 16, 20, 25, 26, 28, 36, 42, 47, 51, 64] to solve the F-ROM closure problem, i.e., to find a formula  $\boldsymbol{\tau} \approx \boldsymbol{\tau}(\mathbf{a})$  in (3.9). To make the F-ROM (3.9) resemble the standard G-ROM (2.5), we make the following ansatz:

$$\boldsymbol{\tau}(\mathbf{a}) = \tilde{A} \mathbf{a} + \mathbf{a}^\top \tilde{B} \mathbf{a}. \quad (4.1)$$

Using ansatz (4.1) in the F-ROM (3.9) yields a closed system of equations.

To find  $\tilde{A}$  and  $\tilde{B}$  in (4.1), we use data-driven modeling. That is, we find  $\tilde{A}$  and  $\tilde{B}$  that ensure the highest accuracy of the vector  $\boldsymbol{\tau}$  in the F-ROM (3.9). To this end, we minimize the  $L^2$ -norm of the difference between  $\boldsymbol{\tau}$  computed with the FOM data and equations (3.8) and (3.10), and  $\boldsymbol{\tau}$  computed with the ansatz (4.1) and the ROM coefficients obtained from the snapshots,  $\mathbf{a}^{snap}$ . The values  $\mathbf{a}^{snap}(t_j)$ , computed at snapshot time instances  $t_j$ ,  $j = 1, \dots, M$ , are obtained by projecting the corresponding snapshots  $\mathbf{u}(t_j) = \sum_{k=1}^d a_k^{snap}(t_j) \boldsymbol{\varphi}_k$  onto the POD basis functions  $\boldsymbol{\varphi}_i$  and using the orthogonality of the POD basis functions:  $\forall i = 1, \dots, d$ ,  $\forall j = 1, \dots, M$ ,

$$a_i^{snap}(t_j) = \left( \mathbf{u}(t_j), \boldsymbol{\varphi}_i \right). \quad (4.2)$$

To find  $\tilde{A}$  and  $\tilde{B}$ , we solve the following optimization problem [47, 51]:

$$\min_{\substack{\tilde{A} \in \mathbb{R}^{r \times r} \\ \tilde{B} \in \mathbb{R}^{r \times r \times r}}} \sum_{j=1}^M \|\boldsymbol{\tau}^{true}(t_j) - \boldsymbol{\tau}^{ansatz}(t_j)\|^2, \quad (4.3)$$

where  $\|\cdot\|$  is the Euclidian norm in  $\mathbb{R}^r$  and  $\boldsymbol{\tau}^{true}(t_j)$  is the *true*  $\boldsymbol{\tau}(t_j)$  computed from the snapshot data:  $\forall i = 1, \dots, r$ ,  $\forall j = 1, \dots, M$ ,

$$\begin{aligned} \tau_i^{true}(t_j) &= \left( \overline{(\mathbf{u}_d^{snap}(t_j) \cdot \nabla) \mathbf{u}_d^{snap}(t_j)}^r - (\mathbf{u}_r^{snap}(t_j) \cdot \nabla) \mathbf{u}_r^{snap}(t_j), \boldsymbol{\varphi}_i \right) \\ &= \left( \overline{\sum_{k_1=1}^d \sum_{k_2=1}^d a_{k_1}^{snap}(t_j) a_{k_2}^{snap}(t_j) (\boldsymbol{\varphi}_{k_1} \cdot \nabla) \boldsymbol{\varphi}_{k_2}}^r \right. \\ &\quad \left. - \sum_{k_3=1}^r \sum_{k_4=1}^r a_{k_3}^{snap}(t_j) a_{k_4}^{snap}(t_j) (\boldsymbol{\varphi}_{k_3} \cdot \nabla) \boldsymbol{\varphi}_{k_4}, \boldsymbol{\varphi}_i \right), \end{aligned} \quad (4.4)$$

where

$$\mathbf{u}_d^{snap}(t_j) = \sum_{k=1}^d a_k^{snap}(t_j) \boldsymbol{\varphi}_k, \quad \mathbf{u}_r^{snap}(t_j) = \sum_{k=1}^r a_k^{snap}(t_j) \boldsymbol{\varphi}_k. \quad (4.5)$$

REMARK 4.1 (Computational Efficiency). *In practical settings, where the rank of the snapshot matrix can be extremely large (e.g.,  $d = \mathcal{O}(1000)$ ), using  $\mathbf{u}_d^{snap} \in \mathbf{X}^d$  in (4.4) would be very costly. One possible solution would be to replace  $\mathbf{u}_d^{snap}$  with, say,  $\mathbf{u}_{2r}^{snap}$  and the ROM projection on  $\mathbf{X}^d$  with the ROM projection on  $\mathbf{X}^{2r}$ . We numerically investigate the accuracy of this approximation in Section 5.*

Finally, we compute  $\boldsymbol{\tau}^{ansatz}(t_j)$  from the ansatz (4.1) and the snapshot data:

$$\boldsymbol{\tau}^{ansatz}(t_j) = \tilde{A} \mathbf{a}^{snap}(t_j) + \mathbf{a}^{snap}(t_j)^\top \tilde{B} \mathbf{a}^{snap}(t_j). \quad (4.6)$$

Plugging (4.2), (4.4), and (4.6) into the minimization problem (4.3), we obtain

$$\min_{\substack{\tilde{A} \in \mathbb{R}^{r \times r} \\ \tilde{B} \in \mathbb{R}^{r \times r \times r}}} \sum_{j=1}^M \left\| \boldsymbol{\tau}^{true}(t_j) - \tilde{A} \mathbf{a}^{snap}(t_j) - \mathbf{a}^{snap}(t_j)^\top \tilde{B} \mathbf{a}^{snap}(t_j) \right\|^2, \quad (4.7)$$

where the vector  $\boldsymbol{\tau}^{true}(t_j) \in \mathbb{R}^{r \times 1}$  is defined in (4.4).

Next, we introduce some notation that allows us to write the optimization problem (4.7) as a least squares problem. To this end, we define the vector  $\mathbf{x} \in \mathbb{R}^{(r^2+r^3) \times 1}$  that contains all the entries of  $\tilde{A}$  and  $\tilde{B}$  (i.e., the unknowns in the optimization problem (4.7)), and the vector  $\mathbf{f} \in \mathbb{R}^{(Mr) \times 1}$  and matrix  $E \in \mathbb{R}^{(Mr) \times (r^2+r^3)}$ , which are computed from  $\mathbf{a}^{snap}(t_j)$  and are chosen to satisfy the following equality [51]:

$$\sum_{j=1}^M \left\| \boldsymbol{\tau}^{true}(t_j) - \tilde{A} \mathbf{a}^{snap}(t_j) - \mathbf{a}^{snap}(t_j)^\top \tilde{B} \mathbf{a}^{snap}(t_j) \right\|^2 = \|\mathbf{f} - E \mathbf{x}\|^2. \quad (4.8)$$

With this notation, the optimization problem (4.7) can be written as a *linear least squares* problem [51]:

$$\min_{\mathbf{x} \in \mathbb{R}^{(r^2+r^3) \times 1}} \|\mathbf{f} - E \mathbf{x}\|^2. \quad (4.9)$$

The optimal  $\tilde{A}^{opt}$  and  $\tilde{B}^{opt}$  (i.e., the entries in  $\mathbf{x}$  that solves the linear least squares problem (4.9)) are used in the F-ROM (3.9), yielding the *data-driven filtered ROM (DDF-ROM)*

$$\dot{\mathbf{a}} = \left( A + \tilde{A} \right) \mathbf{a} + \mathbf{a}^\top \left( B + \tilde{B} \right) \mathbf{a}. \quad (4.10)$$

In [51], the authors note that data-driven least squares problems can be ill-conditioned. To remedy this ill-conditioning, the authors use an empirical remedy: they combine trajectories of different initial conditions. In our numerical experiments in Section 5, the least squares problem (4.9) is also ill-conditioned, just as in [51]. To tackle this challenge, however, we propose an approach that is different from that used in [51]: We use the *truncated singular value decomposition (SVD)* (see Section 3.5 in [23]).



The DDF-ROM with the truncated SVD can be summarized in the following algorithm:

---

**Algorithm 1** DDF-ROM

---

1: Consider the F-ROM (3.9)

$$\dot{\mathbf{a}} = A \mathbf{a} + \mathbf{a}^\top B \mathbf{a} + \boldsymbol{\tau}. \quad (4.11)$$

2: Use snapshot data and (4.4) to compute the true vector  $\boldsymbol{\tau}$  in (4.11),  $\boldsymbol{\tau}^{true}$ :

$$\tau_i^{true}(t_j) = \left( \overline{(\mathbf{u}_d^{snap}(t_j) \cdot \nabla) \mathbf{u}_d^{snap}(t_j)^r} - (\mathbf{u}_r^{snap}(t_j) \cdot \nabla) \mathbf{u}_r^{snap}(t_j), \boldsymbol{\varphi}_i \right). \quad (4.12)$$

3: Use snapshot data and (4.1) to define the ansatz vector  $\boldsymbol{\tau}$  in (4.11),  $\boldsymbol{\tau}^{ansatz}$ :

$$\boldsymbol{\tau}^{ansatz}(t_j) = \tilde{A} \mathbf{a}^{snap}(t_j) + \mathbf{a}^{snap}(t_j)^\top \tilde{B} \mathbf{a}^{snap}(t_j). \quad (4.13)$$

4: Use all the entries of  $\tilde{A}$  and  $\tilde{B}$  in (4.13) to define vector of unknowns,  $\mathbf{x}$ .

5: Use  $\boldsymbol{\tau}^{true}$  in (4.12) and  $\boldsymbol{\tau}^{ansatz}$  in (4.13) to assemble vector  $\mathbf{f}$  and matrix  $E$  that satisfy (4.8).

6: Use the truncated SVD algorithm to solve the linear least squares problem (4.9):

$$\min_{\mathbf{x} \in \mathbb{R}^{(r^2+r^3) \times 1}} \|\mathbf{f} - E \mathbf{x}\|^2. \quad (4.14)$$

(i) Calculate the SVD of  $E$ :

$$E = U \Sigma V^\top. \quad (4.15)$$

(ii) Specify tolerance  $tol$ .

(iii) Construct matrix  $\hat{\Sigma}$  from  $\Sigma$  as follows:  $\hat{\sigma}_i = \sigma_i$  if  $\sigma_i > tol$ . (That is, keep only the entries in  $\Sigma$  that are larger than  $tol$ .)

(iv) Construct  $\hat{E}$ , the truncated SVD of  $E$ :

$$\hat{E} = \hat{U} \hat{\Sigma} \hat{V}^\top, \quad (4.16)$$

where  $\hat{U}$  and  $\hat{V}$  are the entries of  $U$  and  $V$  in (4.15) that correspond to  $\hat{\Sigma}$ .

(v) The solution of the least squares problem (4.14) is

$$\mathbf{x} = \left( \hat{V} \hat{\Sigma}^{-1} \hat{U}^\top \right) \mathbf{f}. \quad (4.17)$$

7: The DDF-ROM has the following form:

$$\dot{\mathbf{a}} = \left( A + \tilde{A} \right) \mathbf{a} + \mathbf{a}^\top \left( B + \tilde{B} \right) \mathbf{a} \quad (4.18)$$

where  $\tilde{A}$  and  $\tilde{B}$  are the appropriate entries of  $\mathbf{x}$  found in (4.17).

---

**5. Numerical Results.** In this section, we investigate the DDF-ROM (4.10) in the numerical simulation of a 2D channel flow past a circular cylinder at a Reynolds number  $Re = 100$ . This is a benchmark problem from [59] that is often used for testing new methods; see, e.g., [45], which is the setting that we adopt here.

In Section 5.1, we describe the mathematical and computational setting of the test problem. In Section 5.2, we outline the snapshot and ROM generation. In Section 5.3, we compare the DDF-ROM with the standard G-ROM, both in terms of numerical accuracy and computational efficiency. In Section 5.4, we perform a sensitivity study with respect to the DDF-ROM parameters. Finally, in Section 5.5, we compare the DDF-ROM with some of the most recent ROMs, both in terms of numerical accuracy and computational efficiency.

**5.1. Test Problem Setup.** The domain is a  $2.2 \times 0.41$  rectangular channel with a radius=0.05 cylinder, centered at  $(0.2, 0.2)$ , see Figure 5.1. No slip boundary conditions are prescribed for the walls and on the cylinder, and the inflow and outflow profiles are given by [33, 55]  $u_1(0, y, t) = u_1(2.2, y, t) = \frac{6}{0.41^2}y(0.41 - y)$ ,  $u_2(0, y, t) = u_2(2.2, y, t) = 0$ . The kinematic viscosity is  $\nu = 10^{-3}$ , there is no forcing, and the flow starts from rest.

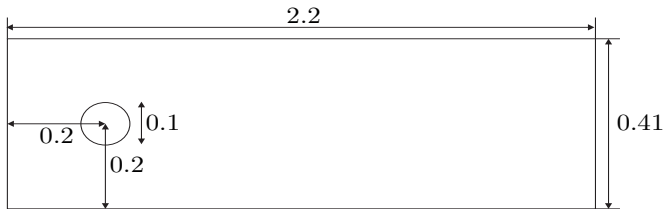


Fig. 5.1: Channel flow around a cylinder domain.

**5.2. Snapshot and ROM Generation.** To compute the snapshots, we use the commonly used linearized BDF2 temporal discretization, together with a FE spatial discretization utilizing the Scott-Vogelius element. On time step 1, we use a backward Euler temporal discretization. All simulations use a time step size of  $\Delta t = 0.002$ , are started from rest, and compute to the end time  $T = 17$ . After an initial spin-up, the flow reaches a periodic-in-time (statistically steady) state by about  $T = 5$  [45]. Snapshots are taken to be the solutions at each time step from  $T = 7$  to  $T = 7.332$ , which corresponds to one period. We compute on 3 different meshes, which provide approximately 103K, 35K, and 23K velocity degrees of freedom. The 103K mesh gives essentially a fully resolved solution, and the lift and drag predictions agree well with results from fine discretizations in [17, 59]:  $c_{d,max} = 3.2261$ ,  $c_{l,max} = 1.0040$ . Results from the 35K meshes are only slightly less accurate, and error is more evident from the 23K simulations.

The ROM modes are created from the snapshots in the usual way. The first mode is chosen to be the snapshot average, which satisfies the boundary conditions. This mode is then subtracted from the snapshots, and finally an eigenvalue problem is solved to find the dominant modes of these adjusted snapshots (see [17] for a more detailed description of the process). The singular values of the snapshot matrix are plotted in Figure 5.2.

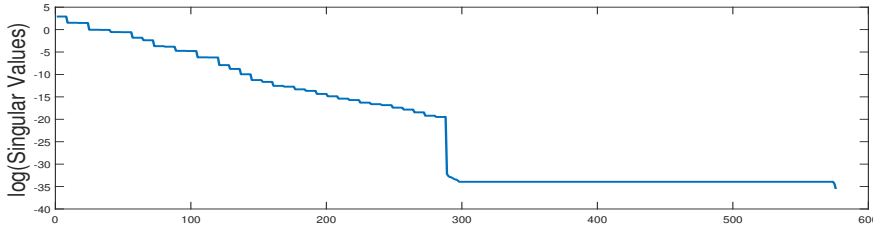


Fig. 5.2: Plots of singular values vs. index, for flow past a cylinder with  $Re = 100$ .

With the dominant modes created, the ROM is constructed as discussed in Section 2 using the BDF2 temporal discretization. In all of our tests, just as in the FE simulations, we take  $\Delta t = 0.002$ ; this choice creates no significant temporal error in any of our simulations (tests were done with varying  $\Delta t$  to verify that 0.002 is sufficiently small). The ROM initial condition at  $T = 6.998$  is the  $L^2$  projection of the FE solution at  $T = 6.998$  into the ROM space. The ROM initial condition at  $T = 7$  is obtained by using the backward Euler method. The ROMs are run from this initial time (now called  $t = 0$ ), and continued to  $t = 10$ . The ROMs are tested using three  $r$  values:  $r = 8, r = 10$ , and  $r = 12$ . Lower  $r$  values yield inaccurate results for all ROMs.

### 5.3. DDF-ROM's Accuracy and Efficiency.

*Accuracy.* In this section, we investigate the accuracy of the DDF-ROM. To this end, we compare the DDF-ROM results with G-ROM results. As benchmark, we use the results obtained on the finest mesh, which has 103K velocity degrees of freedom. To investigate the effect of the particular form of ansatz (4.1) on the numerical results, we consider two DDF-ROM versions: (i) DDF-ROM-quadratic, which is the standard DDF-ROM that considers both  $\tilde{A}$  and  $\tilde{B}$  in ansatz (4.1), and (ii) DDF-ROM-linear, which is the DDF-ROM that considers only  $\tilde{A}$  in ansatz (4.1). Thus, in this section, we investigate three ROMs: DDF-ROM-quadratic, DDF-ROM-linear, and G-ROM. We run all three ROMs with  $r = 8$  ROM modes.

In Figure 5.3, we plot the drag, energy, and lift for all models. For both the DDF-ROM-quadratic and the DDF-ROM-linear, we use  $tol = 10^{-4}$  in the truncated SVD used in Step 6 of Algorithm 1. The main observation is that the DDF-ROM-quadratic performs the best. This is especially true for the drag and energy plots. The G-ROM performs the worst. The DDF-ROM-linear performs better than G-ROM, but worse than the DDF-ROM-quadratic. In Table 5.1, we also list the average errors in the DDF-ROM-quadratic and the G-ROM (comparing to the direct numerical simulation (DNS) solution in the  $L^2$ -norm at each time step, and then taking the average over the time steps). For all  $r$  values, the DDF-ROM-quadratic error is more than 50% lower than the G-ROM error. The plots in Figure 5.3 and the results in Table 5.1 yield the following conclusions: (i) the DDF-ROM-quadratic is significantly more accurate than the G-ROM, and (ii) the DDF-ROM-quadratic is more accurate than the DDF-ROM-linear. Since we have shown that the DDF-ROM-quadratic is significantly more accurate than the DDF-ROM-linear, in what follows we only consider the DDF-ROM-quadratic. Furthermore, since the DDF-ROM-quadratic is the standard DDF-ROM,

we use the latter notation.

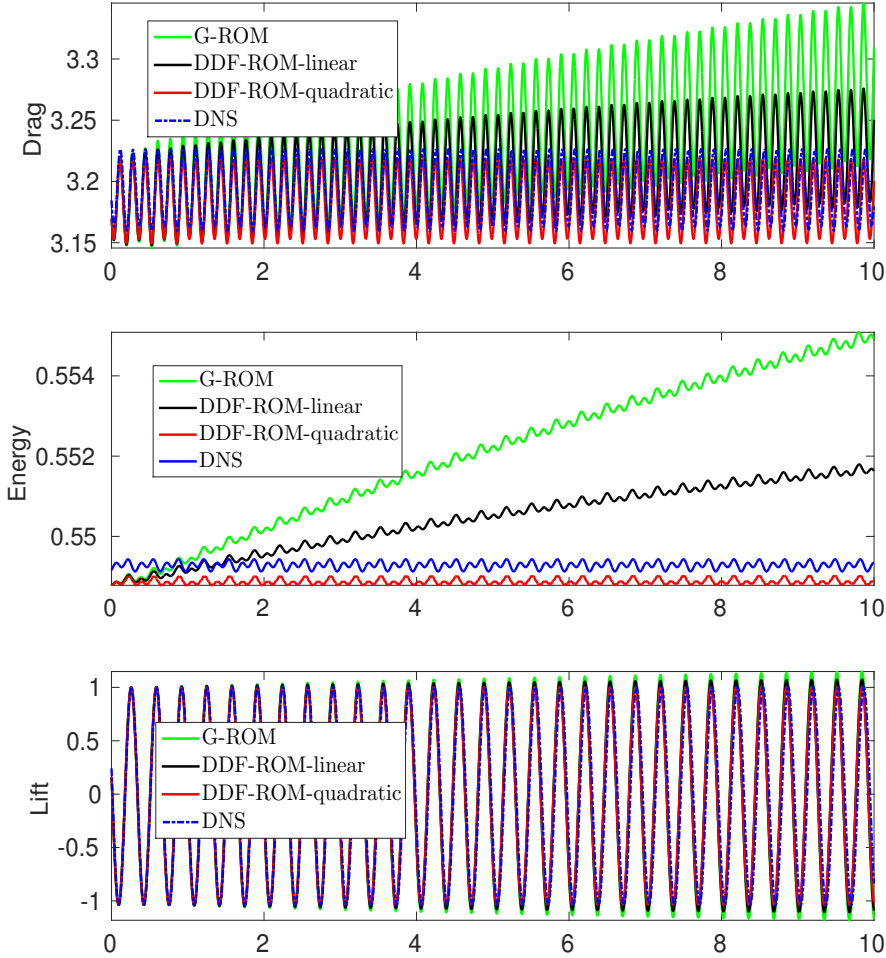


Fig. 5.3: Plots of drag, energy, and lift coefficients vs. time for DDF-ROM-quadratic, DDF-ROM-linear, G-ROM, and DNS for flow past a cylinder with  $\text{Re}=100$ .

*Efficiency.* Although the DDF-ROM is more accurate than the G-ROM, it is less efficient because it must calculate  $\hat{A}$  and  $\hat{B}$ , and so a comparison is in order. We investigate the efficiency of the DDF-ROM and give offline timings for DDF-ROM and G-ROM in Table 5.1. We note that the online timings for DDF-ROM and G-ROM were similar, so we did not include them in Table 5.1. We also note that although the offline timings of both the DDF-ROM and the G-ROM are relatively large, they could be significantly sped up with parallel computations.

In its original form, the DDF-ROM uses (4.4) to compute  $\tau^{\text{true}}(t_j)$ . As mentioned in Remark 4.1, this computation utilizes  $\mathbf{u}_d^{\text{snap}} \in \mathbf{X}^d$ . Since  $d = \mathcal{O}(1000)$  in some practical settings, this could make the DDF-ROM computationally costly. Hence, following Remark 4.1, we replace  $\mathbf{u}_d^{\text{snap}}$  in (4.4) with  $\mathbf{u}_m^{\text{snap}}$ , where  $r \leq m \leq d$ :

$$\mathbf{u}_d^{\text{snap}} \approx \mathbf{u}_m^{\text{snap}} \quad r \leq m \leq d. \quad (5.1)$$

When  $m$  in (5.1) is low (i.e., close to  $r$ ), the DDF-ROM computational cost will be low, but the accuracy will also be low. On the other hand, when  $m$  in (5.1) is large (i.e., close to  $d$ ), the DDF-ROM accuracy will be high, but the computational cost will also be high. Thus, we need to find an  $m$  value in (5.1) that ensures a compromise between accuracy and efficiency in the DDF-ROM. From Table 5.1, we observe that  $m = r + 1$  seems to achieve compromise between accuracy and efficiency in the DDF-ROM.

Method	Proj.	$\ u_{DNS} - u_{ROM}\ $			offline timing (seconds)		
		r=8	r=10	r=12	r=8	r=10	r=12
G-ROM		0.0159	0.0145	0.0039	855.52s	1567.44s	2708.83s
DDF-ROM	$X^r$	0.0159	0.0144	0.0039	1187.21s	1902.04s	3043.01s
DDF-ROM	$X^{r+1}$	0.0099	0.0050	0.0020	1528.12s	2433.48s	3753.97s
DDF-ROM	$X^{2r}$	0.0092	0.0050	0.0019	6373.97s	12162.90s	21028.40s
DDF-ROM	$X^{3r}$	0.0092	0.0050	0.0019	21027.32s	40224.97s	69168.40s

Table 5.1: DDF-ROM and G-ROM errors and offline timings for flow past a cylinder with  $Re=100$ , using varying  $r$  and ROM projection spaces.

**5.4. DDF-ROM’s Parameter Sensitivity.** We also perform a sensitivity study on the DDF-ROM parameters.

In Table 5.2, we list the DDF-ROM’s lift and drag coefficients, and their ranges for different  $r$  values ( $r = 8, r = 10$ , and  $r = 12$ ) and numbers of FE degrees of freedom (23K, 35K, and 103K). We denote  $C_d^{range} = |C_d^{max} - C_d^{min}|$  and  $C_l^{range} = |C_l^{max} - C_l^{min}|$ . The results show that the DDF-ROM has a relatively low sensitivity with respect to  $r$  and the number of FE degrees of freedom.

$r$	FE dof	$C_d^{ave}$	$C_d^{range}$	$C_l^{ave}$	$C_l^{range}$
	103K	3.16	0.07	-0.02	2.04
8	23K	3.16	0.06	-0.02	1.88
8	35K	3.18	0.07	-0.02	2.04
8	103K	3.19	0.07	-0.02	2.05
10	23K	3.16	0.06	-0.02	1.88
10	35K	3.18	0.07	-0.02	2.04
10	103K	3.19	0.07	-0.02	2.04
12	23K	3.16	0.06	-0.02	1.88
12	35K	3.18	0.07	-0.02	2.04
12	103K	3.19	0.07	-0.02	2.04

Table 5.2: DDF-ROM average lift and drag coefficients, and their ranges for flow past a cylinder with  $Re=100$ , using different  $r$  values and numbers of FE degrees of freedom (dof). For comparison purposes, DNS results are listed in second row.

We also perform a DDF-ROM sensitivity study with respect to changes in  $tol$ , which is the tolerance value used in the truncated SVD in Step 6 of Algorithm 1. Values around the value used to generate the plots in Figure 5.3 (i.e.,  $tol = 10^{-4}$ ) yield similar results. However, values that were significantly larger or lower than  $tol = 10^{-4}$  yield inaccurate results. We conclude that the DDF-ROM results are sensitive with respect to  $tol$ .

**5.5. DDF-ROM vs. State-Of-The-Art ROMs.** Above we have shown that the new DDF-ROM is a clear improvement over the standard G-ROM, and a natural question is whether the DDF-ROM is also an improvement over other, more accurate ROMs. To address this question, we consider three recently proposed ROMs for fluid flows: two regularized ROMs (the Leray ROM (L-ROM) and the evolve-then-filter ROM (EF-ROM) [66]) and an LES-ROM (the approximate deconvolution ROM (AD-ROM) [67]).

We compare the new DDF-ROM with the AD-ROM, L-ROM, and EF-ROM. We test all ROMs on the Burgers equation [35] with a steep internal layer (see Figure 5.4) and with a small diffusion coefficient ( $\nu = 10^{-3}$ ). We use the Burgers equation instead of the 2D flow past a circular cylinder that we utilize everywhere else in this section since AD-ROM, L-ROM, and EF-ROM results are available in the literature for the former, but not for the latter.

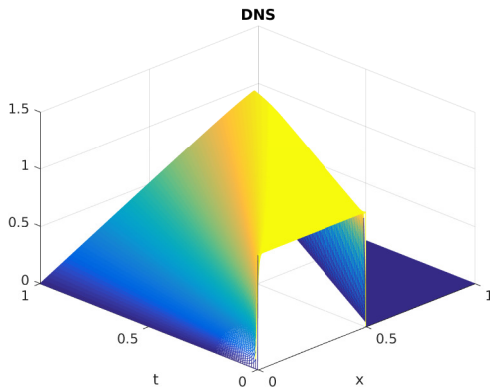


Fig. 5.4: Plot of the solution of the Burgers equation DNS.

We list the errors in Table 5.3. The DDF-ROM errors are slightly lower than all the other ROM errors. We list the CPU times in Table 5.4. The DDF-ROM CPU time is significantly lower than the CPU times of the other ROMs. The results in Table 5.3 and Table 5.4 consistently show that, for this test problem, the DDF-ROM is at least competitive with the other ROMs. (These results are impressive, given that DDF-ROM-linear was used instead of the more accurate DDF-ROM-quadratic, see Section 5.3.)

	L-DF	EF-ROM	AD-ROM	DDF-ROM
$r = 6$	0.1385	0.1005	0.1096	0.0928
$r = 10$	0.1135	0.0699	0.0633	0.0627
$r = 15$	0.1037	0.0549	0.0532	0.0446

Table 5.3: Errors for L-ROM-DF, EF-ROM-DF, AD-ROM, and new DDF-ROM for Burgers equation.

	L-DF	EF-ROM	AD-ROM	DDF-ROM
$r = 6$	4.12	4.25	4.44	2.27
$r = 10$	6.72	6.91	7.26	4.42
$r = 15$	9.97	10.14	10.32	6.67

Table 5.4: CPU times for L-ROM-DF, EF-ROM-DF, AD-ROM, and new DDF-ROM for Burgers equation.

**6. Conclusions and Outlook.** In this paper, we proposed a novel ROM framework for the numerical simulation of fluid flows. This framework was based on explicit ROM spatial filtering and data-driven modeling. The explicit ROM spatial filtering ensured computational efficiency of the filtered ROM, and the data-driven modeling was used to solve the ROM closure problem in the filtered ROM.

We numerically investigated the resulting DDF-ROM in the simulation of a 2D channel flow past a circular cylinder at a Reynolds number  $Re = 100$ . First, we compared the new DDF-ROM with the standard G-ROM. The DDF-ROM was significantly more accurate than the G-ROM. Furthermore, the computational costs of the DDF-ROM and G-ROM were similar, both costs being orders of magnitude lower than the computational cost of the full order model. For the 1D Burgers equation, we also compared the new DDF-ROM with state-of-the-art LES-ROMs. The DDF-ROM was as accurate as state-of-the-art LES-ROMs. However, the DDF-ROM was significantly more efficient than these LES-ROMs.

Although these preliminary results are encouraging, the new DDF-ROM framework’s full potential still needs to be explored. Next, we outline several research directions that could be pursued.

Probably the most important next step in the DDF-ROM development is the data-driven modeling used to solve the ROM closure problem in the filtered ROM. In this paper, we have treated the entries in the subfilter-scale ROM stress tensor  $\tau_r^{SFS}$  as general unknowns. It is well known, however, that in fluid dynamics the subfilter-scale stress tensor satisfies important physical constraints [58]. We plan to replace the unconstrained optimization problem used in the data-driven modeling part of the DDF-ROM with a constrained optimization problem, which includes physical constraints for the subfilter-scale ROM stress tensor, such as energy conservation. Similar approaches have been pursued in [34, 40, 41, 43] in different settings.

Another important research direction is the investigation of the generality of DDF-ROM. Although we constructed and tested the DDF-ROM in a fluid dynamics setting, the DDF-ROM framework can be applied to any type of nonlinear PDE that is amenable to reduced order modeling. Indeed, the only input needed in the DDF-ROM framework is the FOM data. Once those are supplied, the DDF-ROM proceeds in two steps. (i) First, the given nonlinear PDE is spatially filtered. The nonlinearity yields a nonlinear stress tensor (which will generally be different from the stress tensor  $\tau_r^{SFS}$  used in this paper). (ii) In the second step of the DDF-ROM construction, the available FOM data is used to compute an approximation for the true stress tensor in the filtered ROM in (i) and an optimization problem is solved to find the DDF-ROM coefficients. We emphasize again that the entire DDF-ROM procedure does not use any phenomenological arguments that would restrict it to the particular physical system modeled by the given nonlinear PDE. This is in stark contrast with, e.g., ROM closure models of eddy viscosity type [31, 49, 65], which cannot be directly applied

to other classes of PDEs. Since the DDF-ROM is built upon general principles (i.e., filtering and data-driven modeling), we expect it to be successful in the numerical simulation of general mathematical models (e.g., from elasticity or bioengineering).

Finally, the ROM spatial filter used to build the DDF-ROM represents another research direction worthy of investigation. In Section 3, we used the ROM projection (3.1) as a ROM spatial filter to construct the F-ROM (3.9). This choice of ROM spatial filter allowed us to write the F-ROM decomposition in (3.11) and to explain why most ROM closure models amount to adding extra terms to the standard G-ROM. We emphasize, however, that other ROM spatial filters could be used in the new DDF-ROM framework. For example, the ROM differential filter (which was successfully used in developing LES-ROMs [66, 67]) could be used as a ROM spatial filter to construct the F-ROM (3.9).

#### REFERENCES

- [1] D. AMSALLEM AND C. FARHAT, *Stabilization of projection-based reduced-order models*, Int. J. Num. Meth. Eng., 91 (2012), pp. 358–377.
- [2] A. C. ANTOULAS, *Approximation of large-scale dynamical systems*, vol. 6, SIAM, 2005.
- [3] S. ARADAG, S. SIEGEL, J. SEIDEL, K. COHEN, AND T. McLAUGHLIN, *Filtered POD-based low-dimensional modeling of the 3D turbulent flow behind a circular cylinder*, Int. J. Num. Meth. Fluids, 66 (2011), pp. 1–16.
- [4] 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 (1988), pp. 115–173.
- [5] M. J. BALAJEWICZ, E. H. DOWELL, AND B. R. NOACK, *Low-dimensional modelling of high-Reynolds-number shear flows incorporating constraints from the Navier–Stokes equation*, J. Fluid Mech., 729 (2013), pp. 285–308.
- [6] M. J. BALAJEWICZ, I. TEZAUER, AND E. H. DOWELL, *Minimal subspace rotation on the Stiefel manifold for stabilization and enhancement of projection-based reduced order models for the compressible Navier–Stokes equations*, J. Comput. Phys., 321 (2016), pp. 224–241.
- [7] 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 (2016), pp. 609–628.
- [8] 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 (2015), pp. 1136–1161.
- [9] M. F. BARONE, I. KALASHNIKOVA, D. J. SEGALMAN, AND H. K. THORNQUIST, *Stable Galerkin reduced order models for linearized compressible flow*, J. Comput. Phys., 228 (2009), pp. 1932–1946.
- [10] P. BENNER, S. GUGERCIN, AND K. WILLCOX, *A survey of projection-based model reduction methods for parametric dynamical systems*, SIAM Rev., 57 (2015), pp. 483–531.
- [11] 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 (2017), pp. 162–181.
- [12] M. BERGMANN, C. H. BRUNEAU, AND A. IOLLO, *Enablers for robust POD models*, J. Comput. Phys., 228 (2009), pp. 516–538.
- [13] D. A. BISTRAN 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 (2015), pp. 552–580.
- [14] 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 (2016), pp. 3932–3937.
- [15] M. BUFFONI, S. CAMARRI, A. IOLLO, AND M. V. SALVETTI, *Low-dimensional modelling of a confined three-dimensional wake flow*, J. Fluid Mech., 569 (2006), pp. 141–150.
- [16] D. G. CACUCI, I. M. NAVON, AND M. IONESCU-BUJOR, *Computational methods for data evaluation and assimilation*, CRC Press, 2013.
- [17] A. CAIAZZO, T. ILIESCU, V. JOHN, AND S. SCHYSCHLOWA, *A numerical investigation of velocity-pressure reduced order models for incompressible flows*, J. Comput. Phys., 259 (2014),



- pp. 598–616.
- [18] K. CARLBERG, M. BARONE, AND H. ANTIL, *Galerkin v. least-squares Petrov–Galerkin projection in nonlinear model reduction*, J. Comput. Phys., 330 (2017), pp. 693–734.
  - [19] M. D. CHEKROUN, H. LIU, J. C. MCWILLIAMS, AND S. WANG, *Closures for stochastic partial differential equations driven by degenerate noise*, In preparation, (2017).
  - [20] L. CORDIER, B. ABOU EL MAJD, AND J. FAVIER, *Calibration of POD reduced-order models using Tikhonov regularization*, Int. J. Num. Meth. Fluids, 63 (2010), pp. 269–296.
  - [21] M. COUPLET, C. BASDEVANT, AND P. SAGAUT, *Calibrated reduced-order POD–Galerkin system for fluid flow modelling*, J. Comput. Phys., 207 (2005), pp. 192–220.
  - [22] M. COUPLET, P. SAGAUT, AND C. BASDEVANT, *Intermodal energy transfers in a proper orthogonal decomposition–Galerkin representation of a turbulent separated flow*, J. Fluid Mech., 491 (2003), pp. 275–284.
  - [23] J. W. DEMMEL, *Applied numerical linear algebra*, Society for Industrial and Applied Mathematics Philadelphia, 1997.
  - [24] C. FOIAS, O. MANLEY, R. ROSA, AND R. TEMAM, *Navier–Stokes Equations and Turbulence*, Cambridge University Press, 2001.
  - [25] B. GALLETI, A. BOTTARO, C.-H. BRUNEAU, AND A. IOLLO, *Accurate model reduction of transient and forced wakes*, Eur. J. Mech. B-Fluid., 26 (2007), pp. 354–366.
  - [26] B. GALLETI, C. H. BRUNEAU, L. ZANNETTI, AND A. IOLLO, *Low-order modelling of laminar flow regimes past a confined square cylinder*, J. Fluid Mech., 503 (2004), pp. 161–170.
  - [27] S. GIÈRE, T. ILIESCU, V. JOHN, AND D. WELLS, *SUPG reduced order models for convection-dominated convection-diffusion-reaction equations*, Comput. Methods Appl. Mech. Engrg., 289 (2015), pp. 454–474.
  - [28] A. GOUSMI, E. PARISH, AND K. DURAISAMY, *Characterizing memory effects in coarse-grained nonlinear systems using the Mori-Zwanzig formalism*, 2016, <https://arxiv.org/abs/1611.06277>.
  - [29] M. GUNZBURGER, N. JIANG, AND M. SCHNEIER, *An ensemble-proper orthogonal decomposition method for the nonstationary Navier-Stokes equations*, SIAM J. Numer. Anal., (2017). to appear.
  - [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. J. R. HUGHES, G. R. FEIJÓO, L. MAZZEI, AND J.-B. QUINCY, *The variational multiscale method – a paradigm for computational mechanics*, Comput. Methods Appl. Mech. Engrg., 166 (1998), pp. 3–24.
  - [33] V. JOHN, *Reference values for drag and lift of a two dimensional time-dependent flow around a cylinder*, Int. J. Num. Meth. Fluids, 44 (2004), pp. 777–788.
  - [34] D. KONDRASHOV, M. D. CHEKROUN, AND M. GHIL, *Data-driven non-Markovian closure models*, Phys. D, 297 (2015), pp. 33–55.
  - [35] K. KUNISCH AND S. VOLKWEIN, *Galerkin proper orthogonal decomposition methods for parabolic problems*, Numer. Math., 90 (2001), pp. 117–148.
  - [36] J. N. KUTZ, *Data-driven modeling & scientific computation: methods for complex systems & big data*, Oxford University Press, 2013.
  - [37] J. N. KUTZ, S. L. BRUNTON, B. W. BRUNTON, AND J. L. PROCTOR, *Dynamic Mode Decomposition: Data-Driven Modeling of Complex Systems*, SIAM, 2016.
  - [38] J. A. LANGFORD AND R. D. MOSER, *Optimal LES formulations for isotropic turbulence*, J. Fluid Mech., 398 (1999), pp. 321–346.
  - [39] W. J. LAYTON AND L. G. REBHOLZ, *Approximate Deconvolution Models of Turbulence: Analysis, Phenomenology and Numerical Analysis*, vol. 2042, Springer Berlin Heidelberg, 2012.
  - [40] J. LING, R. JONES, AND J. TEMPLETON, *Machine learning strategies for systems with invariance properties*, J. Comput. Phys., 318 (2016), pp. 22–35.
  - [41] J.-C. LOISEAU AND S. L. BRUNTON, *Constrained sparse Galerkin regression*, 2016, <https://arxiv.org/abs/1611.03271>.
  - [42] F. LU, K. K. LIN, AND A. J. CHORIN, *Data-based stochastic model reduction for the Kuramoto–Sivashinsky equation*, Phys. D, 340 (2017), pp. 46–57.
  - [43] A. J. MAJDA AND J. HARLIM, *Physics constrained nonlinear regression models for time series*, Nonlinearity, 26 (2012), p. 201.
  - [44] I. MEZIĆ, *Spectral properties of dynamical systems, model reduction and decompositions*, Nonlinear Dyn., 41 (2005), pp. 309–325.
  - [45] M. MOHEBUJAMAN, L. G. REBHOLZ, X. XIE, AND T. ILIESCU, *Energy balance and mass conservation in reduced order models of fluid flows*, J. Comput. Phys., 346 (2017), pp. 262–277.

- [46] B. R. NOACK, M. MORZYNSKI, AND G. TADMOR, *Reduced-Order Modelling for Flow Control*, vol. 528, Springer Verlag, 2011.
- [47] B. R. NOACK, P. PAPAS, AND P. A. MONKEWITZ, *The need for a pressure-term representation in empirical Galerkin models of incompressible shear flows*, *J. Fluid Mech.*, 523 (2005), pp. 339–365.
- [48] B. R. NOACK, M. SCHLEGEL, B. AHLBORN, G. MUTSCHKE, M. MORZYNSKI, P. COMTE, AND G. TADMOR, *A finite-time thermodynamics of unsteady fluid flows*, *J. Non-Equil. Thermody.*, 33 (2008), pp. 103–148.
- [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 (2014), pp. 518–544.
- [50] B. PEHERSTORFER AND K. WILLCOX, *Dynamic data-driven reduced-order models*, *Comput. Methods Appl. Mech. Engrg.*, 291 (2015), pp. 21–41.
- [51] B. PEHERSTORFER AND K. WILLCOX, *Data-driven operator inference for nonintrusive projection-based model reduction*, *Comput. Methods Appl. Mech. Engrg.*, 306 (2016), pp. 196–215.
- [52] S. PEROTTO, A. REALI, P. RUSCONI, AND A. VENEZIANI, *HIGAMod: A Hierarchical IsoGeometric Approach for MODEL reduction in curved pipes*, *Comput. & Fluids*, 142 (2017), pp. 21–29.
- [53] B. PROTAS, B. R. NOACK, AND J. ÖSTH, *Optimal nonlinear eddy viscosity in Galerkin models of turbulent flows*, *J. Fluid Mech.*, 766 (2015), pp. 337–367.
- [54] A. QUARTERONI, A. MANZONI, AND F. NEGRI, *Reduced Basis Methods for Partial Differential Equations: An Introduction*, vol. 92, Springer, 2015.
- [55] L. REBHOLZ AND M. XIAO, *Improved accuracy in algebraic splitting methods for Navier-Stokes equations*, *SIAM J. Sci. Comput.*, (2017). to appear.
- [56] T. C. REBOLLO AND R. LEWANDOWSKI, *Mathematical and Numerical Foundations of Turbulence Models and Applications*, Springer, 2014.
- [57] C. W. ROWLEY, I. MEZIĆ, S. BAGHERI, P. SCHLATTER, AND D. S. HENNINGSON, *Spectral analysis of nonlinear flows*, *J. Fluid Mech.*, 641 (2009), pp. 115–127.
- [58] P. SAGAUT, *Large Eddy Simulation for Incompressible Flows*, Scientific Computation, Springer-Verlag, Berlin, third ed., 2006.
- [59] M. SCHÄFER AND S. TUREK, *The benchmark problem ‘flow around a cylinder’ flow simulation with high performance computers II*, in E.H. Hirschel (Ed.), *Notes on Numerical Fluid Mechanics*, 52, Braunschweig, Vieweg (1996), pp. 547–566.
- [60] P. J. SCHMID, *Dynamic mode decomposition of numerical and experimental data*, *J. Fluid Mech.*, 656 (2010), pp. 5–28.
- [61] S. SIRISUP AND G. E. KARNIADAKIS, *A spectral viscosity method for correcting the long-term behavior of POD models*, *J. Comput. Phys.*, 194 (2004), pp. 92–116.
- [62] L. SIROVICH, *Turbulence and the dynamics of coherent structures. Parts I–III*, *Quart. Appl. Math.*, 45 (1987), pp. 561–590.
- [63] R. ȘTEFĂNESCU, A. SANDU, AND I. M. NAVON, *POD/DEIM reduced-order strategies for efficient four dimensional variational data assimilation*, *J. Comput. Phys.*, 295 (2015), pp. 569–595.
- [64] J.-X. WANG, J.-L. WU, AND H. XIAO, *Physics-informed machine learning approach for reconstructing Reynolds stress modeling discrepancies based on DNS data*, *Phys. Rev. Fluids*, 2 (2017), p. 034603.
- [65] 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 (2012), pp. 10–26.
- [66] 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 (2017), pp. 598–615.
- [67] X. XIE, D. WELLS, Z. WANG, AND T. ILIESCU, *Approximate deconvolution reduced order modeling*, *Comput. Methods Appl. Mech. Engrg.*, 313 (2017), pp. 512–534.